

# The independence of Tarski’s Euclidean axiom

T. J. M. Makarios

February 6, 2026

## Abstract

Tarski’s axioms of plane geometry are formalized and, using the standard real Cartesian model, shown to be consistent. A substantial theory of the projective plane is developed. Building on this theory, the Klein–Beltrami model of the hyperbolic plane is defined and shown to satisfy all of Tarski’s axioms except his Euclidean axiom; thus Tarski’s Euclidean axiom is shown to be independent of his other axioms of plane geometry.

An earlier version of this work was the subject of the author’s MSc thesis [2], which contains natural-language explanations of some of the more interesting proofs.

## Contents

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>Metric and semimetric spaces</b>                               | <b>2</b>  |
| <b>2</b> | <b>Miscellaneous results</b>                                      | <b>4</b>  |
| <b>3</b> | <b>Tarski’s geometry</b>  | <b>11</b> |
| 3.1      | The axioms . . . . .  | 12        |
| 3.2      | Semimetric spaces satisfy the first three axioms . . . . .        | 12        |
| 3.3      | Some consequences of the first three axioms . . . . .             | 13        |
| 3.4      | Some consequences of the first five axioms . . . . .              | 17        |
| 3.5      | Simple theorems about betweenness . . . . .                       | 18        |
| 3.6      | Simple theorems about congruence and betweenness . . . . .        | 20        |
| <b>4</b> | <b>Real Euclidean space and Tarski’s axioms</b>                   | <b>20</b> |
| 4.1      | Real Euclidean space satisfies the first five axioms . . . . .    | 20        |
| 4.2      | Real Euclidean space also satisfies axioms 6, 7, and 11 . . . . . | 25        |
| 4.3      | Real Euclidean space satisfies the Euclidean axiom . . . . .      | 30        |
| 4.4      | The real Euclidean plane . . . . .                                | 31        |
| 4.5      | Special cases of theorems of Tarski’s geometry . . . . .          | 35        |
| <b>5</b> | <b>Linear algebra</b>   | <b>37</b> |
| 5.1      | Matrices . . . . .  | 40        |

|          |   |            |
|----------|---|------------|
| <b>6</b> | <b>Right group actions</b>  | <b>41</b>  |
| <b>7</b> | <b>Projective geometry</b>  | <b>43</b>  |
| 7.1      | Proportionality on non-zero vectors . . . . .                                   | 43         |
| 7.2      | Points of the real projective plane . . . . .                                   | 45         |
| 7.3      | Lines of the real projective plane . . . . .                                    | 49         |
| 7.4      | Collineations of the real projective plane . . . . .                            | 67         |
| 7.4.1    | As a group . . . . .  | 71         |
| 7.4.2    | As a group action . . . . .   | 74         |
| 7.4.3    | Parts of some Statements from [1] . . . . .                                     | 81         |
| 7.5      | Cross ratios . . . . .  | 88         |
| 7.6      | Cartesian subspace of the real projective plane . . . . .                       | 96         |
| <b>8</b> | <b>The hyperbolic plane and Tarski's axioms</b>                                 | <b>105</b> |
| 8.1      | Characterizing a specific conic in the projective plane . . . . .               | 105        |
| 8.2      | Some specific points and lines of the projective plane . . . . .                | 114        |
| 8.3      | Definition of the Klein–Beltrami model of the hyperbolic plane                  | 120        |
| 8.4      | $K$ -isometries map the interior of the conic to itself . . . . .               | 125        |
| 8.5      | The $K$ -isometries form a group action . . . . .                               | 141        |
| 8.6      | The Klein–Beltrami model satisfies Tarski's first three axioms                  | 142        |
| 8.7      | Some lemmas about betweenness . . . . .   | 155        |
| 8.8      | The Klein–Beltrami model satisfies axiom 4 . . . . .                            | 162        |
| 8.9      | More betweenness theorems . . . . .   | 168        |
| 8.10     | Perpendicularity . . . . .  | 177        |
| 8.11     | Functions of distance . . . . .   | 190        |
| 8.11.1   | A formula for a cross ratio involving a perpendicular<br>foot . . . . .         | 207        |
| 8.12     | The Klein–Beltrami model satisfies axiom 5 . . . . .                            | 210        |
| 8.13     | The Klein–Beltrami model satisfies axioms 6, 7, and 11 . . . . .                | 217        |
| 8.14     | The Klein–Beltrami model satisfies the dimension-specific ax-<br>ioms . . . . . | 219        |
| 8.15     | The Klein–Beltrami model violates the Euclidean axiom . . . . .                 | 222        |

## 1 Metric and semimetric spaces

```

theory Metric
imports HOL-Analysis.Multivariate-Analysis
begin

locale semimetric =
  fixes dist :: 'p ⇒ 'p ⇒ real
  assumes nonneg [simp]: dist x y ≥ 0
  and eq-0 [simp]: dist x y = 0 ⟷ x = y
  and symm: dist x y = dist y x
begin

```

```

lemma refl [simp]: dist x x = 0
  by simp
end

locale metric =
  fixes dist :: 'p ⇒ 'p ⇒ real
  assumes [simp]: dist x y = 0 ⟷ x = y
  and triangle: dist x z ≤ dist y x + dist y z

sublocale metric < semimetric
proof
  { fix w
    have dist w w = 0 by simp }
  note [simp] = this
  fix x y
  show 0 ≤ dist x y
  proof -
    from triangle [of y y x] show 0 ≤ dist x y by simp
  qed
  show dist x y = 0 ⟷ x = y by simp
  show dist x y = dist y x
  proof -
    { fix w z
      have dist w z ≤ dist z w
      proof -
        from triangle [of w z z] show dist w z ≤ dist z w by simp
      qed }
    hence dist x y ≤ dist y x and dist y x ≤ dist x y by simp+
    thus dist x y = dist y x by simp
  qed
qed

definition norm-dist :: ('a::real-normed-vector) ⇒ 'a ⇒ real where
[simp]: norm-dist x y ≜ norm (x - y)

interpretation norm-metric: metric norm-dist
proof
  fix x y
  show norm-dist x y = 0 ⟷ x = y by simp
  fix z
  from norm-triangle-ineq [of x - y y - z] have
    norm (x - z) ≤ norm (x - y) + norm (y - z) by simp
  with norm-minus-commute [of x y] show
    norm-dist x z ≤ norm-dist y x + norm-dist y z by simp
qed

end

```

## 2 Miscellaneous results

```
theory Miscellany
imports Metric
begin
```

```
lemma unordered-pair-element-equality:
  assumes {p, q} = {r, s} and p = r
  shows q = s
  using assms by (auto simp: doubleton-eq-iff)
```

```
lemma unordered-pair-equality: {p, q} = {q, p}
  by auto
```

```
lemma cosine-rule:
  fixes a b c :: real ^ ('n::finite)
  shows (norm-dist a c)2 =
    (norm-dist a b)2 + (norm-dist b c)2 + 2 * ((a - b) · (b - c))
proof -
  have (a - b) + (b - c) = a - c by simp
  with dot-norm [of a - b b - c]
  have (a - b) · (b - c) =
    ((norm (a - c))2 - (norm (a - b))2 - (norm (b - c))2) / 2
  by simp
  thus ?thesis by simp
qed
```

```
lemma scalar-equiv: r * s x = r *R x
  by vector
```

```
lemma norm-dist-dot: (norm-dist x y)2 = (x - y) · (x - y)
  by (simp add: power2-norm-eq-inner)
```

```
definition dep2 :: 'a::real-vector ⇒ 'a ⇒ bool where
  dep2 u v ≜ ∃ w r s. u = r *R w ∧ v = s *R w
```

```
lemma real2-eq:
  fixes u v :: real2
  assumes u$1 = v$1 and u$2 = v$2
  shows u = v
  by (simp add: vec-eq-iff [of u v] forall-2 assms)
```

```
definition rotate2 :: real2 ⇒ real2 where
  rotate2 x ≜ vector [-x$2, x$1]
```

```
declare vector-2 [simp]
```

```
lemma rotate2 [simp]:
  (rotate2 x)$1 = -x$2
```

$(\text{rotate2 } x)\$2 = x\$1$   
**by** (*simp add: rotate2-def*)**+**

**lemma rotate2-rotate2** [*simp*]:  $\text{rotate2 } (\text{rotate2 } x) = -x$   
**proof** –  
**have**  $(\text{rotate2 } (\text{rotate2 } x))\$1 = -x\$1$  **and**  $(\text{rotate2 } (\text{rotate2 } x))\$2 = -x\$2$   
**by** *simp+*  
**with** *real2-eq* **show**  $\text{rotate2 } (\text{rotate2 } x) = -x$  **by** *simp*  
**qed**

**lemma rotate2-dot** [*simp*]:  $(\text{rotate2 } u) \cdot (\text{rotate2 } v) = u \cdot v$   
**unfolding** *inner-vec-def*  
**by** (*simp add: sum-2*)

**lemma rotate2-scaleR** [*simp*]:  $\text{rotate2 } (k *_R x) = k *_R (\text{rotate2 } x)$   
**proof** –  
**have**  $(\text{rotate2 } (k *_R x))\$1 = (k *_R (\text{rotate2 } x))\$1$  **and**  
 $(\text{rotate2 } (k *_R x))\$2 = (k *_R (\text{rotate2 } x))\$2$  **by** *simp+*  
**with** *real2-eq* **show** *?thesis* **by** *simp*  
**qed**

**lemma rotate2-uminus** [*simp*]:  $\text{rotate2 } (-x) = -(\text{rotate2 } x)$   
**proof** –  
**from** *scaleR-minus-left* [*of 1*] **have**  
 $-1 *_R x = -x$  **and**  $-1 *_R (\text{rotate2 } x) = -(\text{rotate2 } x)$  **by** *auto*  
**with** *rotate2-scaleR* [*of -1 x*] **show** *?thesis* **by** *simp*  
**qed**

**lemma rotate2-eq** [*iff*]:  $\text{rotate2 } x = \text{rotate2 } y \longleftrightarrow x = y$   
**proof**  
**assume**  $x = y$   
**thus**  $\text{rotate2 } x = \text{rotate2 } y$  **by** *simp*  
**next**  
**assume**  $\text{rotate2 } x = \text{rotate2 } y$   
**hence**  $\text{rotate2 } (\text{rotate2 } x) = \text{rotate2 } (\text{rotate2 } y)$  **by** *simp*  
**hence**  $-(-x) = -(-y)$  **by** *simp*  
**thus**  $x = y$  **by** *simp*  
**qed**

**lemma dot2-rearrange-1**:  
**fixes**  $u x :: \text{real}^2$   
**assumes**  $u \cdot x = 0$  **and**  $x\$1 \neq 0$   
**shows**  $u = (u\$2 / x\$1) *_R (\text{rotate2 } x)$  (**is**  $u = ?u'$ )  
**proof** –  
**from**  $\langle u \cdot x = 0 \rangle$  **have**  $u\$1 * x\$1 = -(u\$2) * (x\$2)$   
**unfolding** *inner-vec-def*  
**by** (*simp add: sum-2*)  
**hence**  $u\$1 * x\$1 / x\$1 = -u\$2 / x\$1 * x\$2$  **by** *simp*  
**with**  $\langle x\$1 \neq 0 \rangle$  **have**  $u\$1 = ?u'\$1$  **by** *simp*

**from**  $\langle x\$1 \neq 0 \rangle$  **have**  $u\$2 = ?u'\$2$  **by** *simp*  
**with**  $\langle u\$1 = ?u'\$1 \rangle$  **and** *real2-eq* **show**  $u = ?u'$  **by** *simp*  
**qed**

**lemma** *dot2-rearrange-2*:  
**fixes**  $u\ x :: \text{real}^2$   
**assumes**  $u \cdot x = 0$  **and**  $x\$2 \neq 0$   
**shows**  $u = -(u\$1 / x\$2) *_R (\text{rotate2 } x)$  (**is**  $u = ?u'$ )  
**proof** –  
**from** *assms* **and** *dot2-rearrange-1* [*of rotate2 u rotate2 x*] **have**  
 $\text{rotate2 } u = \text{rotate2 } ?u'$  **by** *simp*  
**thus**  $u = ?u'$  **by** *blast*  
**qed**

**lemma** *dot2-rearrange*:  
**fixes**  $u\ x :: \text{real}^2$   
**assumes**  $u \cdot x = 0$  **and**  $x \neq 0$   
**shows**  $\exists k. u = k *_R (\text{rotate2 } x)$   
**proof** *cases*  
**assume**  $x\$1 = 0$   
**with** *real2-eq* [*of x 0*] **and**  $\langle x \neq 0 \rangle$  **have**  $x\$2 \neq 0$  **by** *auto*  
**with** *dot2-rearrange-2* **and**  $\langle u \cdot x = 0 \rangle$  **show** *?thesis* **by** *blast*  
**next**  
**assume**  $x\$1 \neq 0$   
**with** *dot2-rearrange-1* **and**  $\langle u \cdot x = 0 \rangle$  **show** *?thesis* **by** *blast*  
**qed**

**lemma** *real2-orthogonal-dep2*:  
**fixes**  $u\ v\ x :: \text{real}^2$   
**assumes**  $x \neq 0$  **and**  $u \cdot x = 0$  **and**  $v \cdot x = 0$   
**shows** *dep2*  $u\ v$   
**proof** –  
**let**  $?w = \text{rotate2 } x$   
**from** *dot2-rearrange* **and** *assms* **have**  
 $\exists r\ s. u = r *_R ?w \wedge v = s *_R ?w$  **by** *simp*  
**with** *dep2-def* **show** *?thesis* **by** *auto*  
**qed**

**lemma** *dot-left-diff-distrib*:  
**fixes**  $u\ v\ x :: \text{real}^n$   
**shows**  $(u - v) \cdot x = (u \cdot x) - (v \cdot x)$   
**proof** –  
**have**  $(u \cdot x) - (v \cdot x) = (\sum_{i \in UNIV}. u\$i * x\$i) - (\sum_{i \in UNIV}. v\$i * x\$i)$   
**unfolding** *inner-vec-def*  
**by** *simp*  
**also from** *sum-subtractf* [*of  $\lambda i. u\$i * x\$i \ \lambda i. v\$i * x\$i$* ] **have**  
 $\dots = (\sum_{i \in UNIV}. u\$i * x\$i - v\$i * x\$i)$  **by** *simp*  
**also from** *left-diff-distrib* [**where**  $'a = \text{real}$ ] **have**  
 $\dots = (\sum_{i \in UNIV}. (u\$i - v\$i) * x\$i)$  **by** *simp*

also have  
 $\dots = (u - v) \cdot x$   
 unfolding *inner-vec-def*  
 by *simp*  
 finally show *?thesis ..*  
 qed

lemma *dot-right-diff-distrib*:  
 fixes  $u v x :: \text{real}^n$   
 shows  $x \cdot (u - v) = (x \cdot u) - (x \cdot v)$   
 proof -  
 from *inner-commute* have  $x \cdot (u - v) = (u - v) \cdot x$  by *auto*  
 also from *dot-left-diff-distrib* [of  $u v x$ ] have  
 $\dots = u \cdot x - v \cdot x$ .  
 also from *inner-commute* [of  $x$ ] have  
 $\dots = x \cdot u - x \cdot v$  by *simp*  
 finally show *?thesis .*  
 qed

lemma *am-gm2*:  
 fixes  $a b :: \text{real}$   
 assumes  $a \geq 0$  and  $b \geq 0$   
 shows  $\text{sqrt } (a * b) \leq (a + b) / 2$   
 and  $\text{sqrt } (a * b) = (a + b) / 2 \iff a = b$   
 proof -  
 have  $0 \leq (a - b) * (a - b)$  and  $0 = (a - b) * (a - b) \iff a = b$  by *simp+*  
 with *right-diff-distrib* [of  $a - b a b$ ] and *left-diff-distrib* [of  $a b$ ] have  
 $0 \leq a * a - 2 * a * b + b * b$   
 and  $0 = a * a - 2 * a * b + b * b \iff a = b$  by *auto*  
 hence  $4 * a * b \leq a * a + 2 * a * b + b * b$   
 and  $4 * a * b = a * a + 2 * a * b + b * b \iff a = b$  by *auto*  
 with *distrib-right* [of  $a + b a b$ ] and *distrib-left* [of  $a b$ ] have  
 $4 * a * b \leq (a + b) * (a + b)$   
 and  $4 * a * b = (a + b) * (a + b) \iff a = b$  by (*simp add: field-simps*)+  
 with *real-sqrt-le-mono* [of  $4 * a * b (a + b) * (a + b)$ ]  
 and *real-sqrt-eq-iff* [of  $4 * a * b (a + b) * (a + b)$ ] have  
 $\text{sqrt } (4 * a * b) \leq \text{sqrt } ((a + b) * (a + b))$   
 and  $\text{sqrt } (4 * a * b) = \text{sqrt } ((a + b) * (a + b)) \iff a = b$  by *simp+*  
 with  $\langle a \geq 0 \rangle$  and  $\langle b \geq 0 \rangle$  have  $\text{sqrt } (4 * a * b) \leq a + b$   
 and  $\text{sqrt } (4 * a * b) = a + b \iff a = b$  by *simp+*  
 with *real-sqrt-abs2* [of 2] and *real-sqrt-mult* [of  $4 a * b$ ] show  
 $\text{sqrt } (a * b) \leq (a + b) / 2$   
 and  $\text{sqrt } (a * b) = (a + b) / 2 \iff a = b$  by (*simp add: ac-simps*)+  
 qed

lemma *refl-on-allrel*: *refl-on*  $A (A \times A)$   
 unfolding *refl-on-def*  
 by *simp*

**lemma** *refl-on-restrict*:  
**assumes** *refl-on A r*  
**shows** *refl-on (A ∩ B) (r ∩ B × B)*  
**proof** –  
**from**  $\langle \text{refl-on } A \ r \rangle$  **and** *refl-on-allrel [of B]* **and** *refl-on-Int*  
**show** *?thesis* **by** *auto*  
**qed**

**lemma** *sym-allrel*: *sym (A × A)*  
**unfolding** *sym-def*  
**by** *simp*

**lemma** *sym-restrict*:  
**assumes** *sym r*  
**shows** *sym (r ∩ A × A)*  
**proof** –  
**from**  $\langle \text{sym } r \rangle$  **and** *sym-allrel* **and** *sym-Int*  
**show** *?thesis* **by** *auto*  
**qed**

**lemma** *trans-allrel*: *trans (A × A)*  
**unfolding** *trans-def*  
**by** *simp*

**lemma** *equiv-Int*:  
**assumes** *equiv A r* **and** *equiv B s*  
**shows** *equiv (A ∩ B) (r ∩ s)*  
**proof** –  
**from** *assms* **and** *refl-on-Int [of A r B s]* **and** *sym-Int* **and** *trans-Int*  
**show** *?thesis*  
**unfolding** *equiv-def*  
**by** *auto*  
**qed**

**lemma** *equiv-allrel*: *equiv A (A × A)*  
**unfolding** *equiv-def*  
**by** (*simp add: refl-on-allrel sym-allrel trans-allrel*)

**lemma** *equiv-restrict*:  
**assumes** *equiv A r*  
**shows** *equiv (A ∩ B) (r ∩ B × B)*  
**proof** –  
**from**  $\langle \text{equiv } A \ r \rangle$  **and** *equiv-allrel [of B]* **and** *equiv-Int*  
**show** *?thesis* **by** *auto*  
**qed**

**lemma** *invertible-times-eq-zero*:  
**fixes**  $x :: \text{real}^n$  **and**  $A :: \text{real}^n \times \text{real}^n$   
**assumes** *invertible A* **and**  $A * v \ x = 0$

**shows**  $x = 0$   
**using** *assms invertible-def matrix-left-invertible-ker* **by** *blast*

**lemma** *times-invertible-eq-zero*:  
**fixes**  $x :: \text{real}^n$  **and**  $A :: \text{real}^{n \times n}$   
**assumes** *invertible A* **and**  $x \cdot A = 0$   
**shows**  $x = 0$   
**using** *transpose-invertible assms invertible-times-eq-zero* **by** *fastforce*

**lemma** *matrix-id-invertible*:  
*invertible (mat 1 :: ('a::semiring-1)^n^n)*  
**by** (*simp add: invertible-def*)

**lemma** *Image-refl-on-nonempty*:  
**assumes** *refl-on A r* **and**  $x \in A$   
**shows**  $x \in r^{\{x\}}$   
**proof**  
**from**  $\langle \text{refl-on } A \ r \rangle$  **and**  $\langle x \in A \rangle$  **show**  $(x, x) \in r$   
**unfolding** *refl-on-def*  
**by** *simp*  
**qed**

**lemma** *quotient-element-nonempty*:  
**assumes** *equiv A r* **and**  $X \in A//r$   
**shows**  $\exists x. x \in X$   
**using** *assms in-quotient-imp-non-empty* **by** *fastforce*

**lemma** *zero-3*:  $(3::3) = 0$   
**by** *simp*

**lemma** *card-suc-ge-insert*:  
**fixes**  $A$  **and**  $x$   
**shows**  $\text{card } A + 1 \geq \text{card } (\text{insert } x \ A)$   
**using** *card-insert-le-m1* **by** *fastforce*

**lemma** *card-le-UNIV*:  
**fixes**  $A :: ('n::\text{finite}) \text{ set}$   
**shows**  $\text{card } A \leq \text{CARD } ('n)$   
**by** (*simp add: card-mono*)

**lemma** *partition-Image-element*:  
**assumes** *equiv A r* **and**  $X \in A//r$  **and**  $x \in X$   
**shows**  $r^{\{x\}} = X$   
**by** (*metis Image-singleton-iff assms equiv-class-eq-iff quotientE*)

**lemma** *card-insert-ge*:  $\text{card } (\text{insert } x \ A) \geq \text{card } A$   
**by** (*metis card.infinite card-insert-le zero-le*)

**lemma** *choose-1*:

**assumes**  $\text{card } S = 1$   
**shows**  $\exists x. S = \{x\}$   
**using**  $\langle \text{card } S = 1 \rangle$  **and**  $\text{card-eq-SucD}$  [of  $S$  0]  
**by** *simp*

**lemma** *choose-2*:

**assumes**  $\text{card } S = 2$   
**shows**  $\exists x y. S = \{x, y\}$   
**proof** –  
**from**  $\langle \text{card } S = 2 \rangle$  **and**  $\text{card-eq-SucD}$  [of  $S$  1]  
**obtain**  $x$  **and**  $T$  **where**  $S = \text{insert } x T$  **and**  $\text{card } T = 1$  **by** *auto*  
**from**  $\langle \text{card } T = 1 \rangle$  **and** *choose-1* **obtain**  $y$  **where**  $T = \{y\}$  **by** *auto*  
**with**  $\langle S = \text{insert } x T \rangle$  **have**  $S = \{x, y\}$  **by** *simp*  
**thus**  $\exists x y. S = \{x, y\}$  **by** *auto*

**qed**

**lemma** *choose-3*:

**assumes**  $\text{card } S = 3$   
**shows**  $\exists x y z. S = \{x, y, z\}$   
**proof** –  
**from**  $\langle \text{card } S = 3 \rangle$  **and**  $\text{card-eq-SucD}$  [of  $S$  2]  
**obtain**  $x$  **and**  $T$  **where**  $S = \text{insert } x T$  **and**  $\text{card } T = 2$  **by** *auto*  
**from**  $\langle \text{card } T = 2 \rangle$  **and** *choose-2* [of  $T$ ] **obtain**  $y$  **and**  $z$  **where**  $T = \{y, z\}$  **by** *auto*  
**with**  $\langle S = \text{insert } x T \rangle$  **have**  $S = \{x, y, z\}$  **by** *simp*  
**thus**  $\exists x y z. S = \{x, y, z\}$  **by** *auto*

**qed**

**lemma** *card-gt-0-diff-singleton*:

**assumes**  $\text{card } S > 0$  **and**  $x \in S$   
**shows**  $\text{card } (S - \{x\}) = \text{card } S - 1$   
**proof** –  
**from**  $\langle \text{card } S > 0 \rangle$  **have** *finite*  $S$  **by** (*rule card-ge-0-finite*)  
**with**  $\langle x \in S \rangle$   
**show**  $\text{card } (S - \{x\}) = \text{card } S - 1$  **by** (*simp add: card-Diff-singleton*)

**qed**

**lemma** *eq-3-or-of-3*:

**fixes**  $j :: 4$   
**shows**  $j = 3 \vee (\exists j'::3. j = \text{of-int } (\text{Rep-bit1 } j'))$   
**proof** (*induct j*)  
**fix**  $j\text{-int} :: \text{int}$   
**assume**  $0 \leq j\text{-int}$   
**assume**  $j\text{-int} < \text{int } \text{CARD}(4)$   
**hence**  $j\text{-int} \leq 3$  **by** *simp*

**show**  $\text{of-int } j\text{-int} = (3::4) \vee (\exists j'::3. \text{of-int } j\text{-int} = \text{of-int } (\text{Rep-bit1 } j'))$

**proof** *cases*

**assume**  $j\text{-int} = 3$

```

thus
  of-int j-int = (3::4) ∨ (∃ j'::3. of-int j-int = of-int (Rep-bit1 j'))
  by simp
next
  assume j-int ≠ 3
  with ⟨j-int ≤ 3⟩ have j-int < 3 by simp
  with ⟨0 ≤ j-int⟩ have j-int ∈ {0..<3} by simp
  hence Rep-bit1 (Abs-bit1 j-int :: 3) = j-int
  by (simp add: bit1.Abs-inverse)
  hence of-int j-int = of-int (Rep-bit1 (Abs-bit1 j-int :: 3)) by simp
thus
  of-int j-int = (3::4) ∨ (∃ j'::3. of-int j-int = of-int (Rep-bit1 j'))
  by auto
qed
qed

```

```

lemma sgn-plus:
  fixes x y :: 'a::linordered-idom
  assumes sgn x = sgn y
  shows sgn (x + y) = sgn x
  by (simp add: assms same-sgn-sgn-add)

```

```

lemma sgn-div:
  fixes x y :: 'a::linordered-field
  assumes y ≠ 0 and sgn x = sgn y
  shows x / y > 0
  using assms sgn-1-pos sgn-eq-0-iff by fastforce

```

```

lemma abs-plus:
  fixes x y :: 'a::linordered-idom
  assumes sgn x = sgn y
  shows |x + y| = |x| + |y|
  by (simp add: assms same-sgn-abs-add)

```

```

lemma sgn-plus-abs:
  fixes x y :: 'a::linordered-idom
  assumes |x| > |y|
  shows sgn (x + y) = sgn x
  by (cases x > 0) (use assms in auto)

```

**end**

### 3 Tarski's geometry

```

theory Tarski
  imports Complex-Main Miscellany Metric
begin

```

### 3.1 The axioms

The axioms, and all theorems beginning with *th* followed by a number, are based on corresponding axioms and theorems in [3].

**locale** *tarski-first3* =

**fixes**  $C :: 'p \Rightarrow 'p \Rightarrow 'p \Rightarrow 'p \Rightarrow \text{bool}$  ( $\langle \cdot - \equiv - \cdot \rangle$  [99,99,99,99] 50)  
**assumes**  $A1: \forall a b. a b \equiv b a$   
**and**  $A2: \forall a b p q r s. a b \equiv p q \wedge a b \equiv r s \longrightarrow p q \equiv r s$   
**and**  $A3: \forall a b c. a b \equiv c c \longrightarrow a = b$

**locale** *tarski-first5* = *tarski-first3* +

**fixes**  $B :: 'p \Rightarrow 'p \Rightarrow 'p \Rightarrow \text{bool}$   
**assumes**  $A4: \forall q a b c. \exists x. B q a x \wedge a x \equiv b c$   
**and**  $A5: \forall a b c d a' b' c' d'. a \neq b \wedge B a b c \wedge B a' b' c'$   
 $\wedge a b \equiv a' b' \wedge b c \equiv b' c' \wedge a d \equiv a' d' \wedge b$   
 $d \equiv b' d'$   
 $\longrightarrow c d \equiv c' d'$

**locale** *tarski-absolute-space* = *tarski-first5* +

**assumes**  $A6: \forall a b. B a b a \longrightarrow a = b$   
**and**  $A7: \forall a b c p q. B a p c \wedge B b q c \longrightarrow (\exists x. B p x b \wedge B q x a)$   
**and**  $A11: \forall X Y. (\exists a. \forall x y. x \in X \wedge y \in Y \longrightarrow B a x y)$   
 $\longrightarrow (\exists b. \forall x y. x \in X \wedge y \in Y \longrightarrow B x b y)$

**locale** *tarski-absolute* = *tarski-absolute-space* +

**assumes**  $A8: \exists a b c. \neg B a b c \wedge \neg B b c a \wedge \neg B c a b$   
**and**  $A9: \forall p q a b c. p \neq q \wedge a p \equiv a q \wedge b p \equiv b q \wedge c p \equiv c q$   
 $\longrightarrow B a b c \vee B b c a \vee B c a b$

**locale** *tarski-space* = *tarski-absolute-space* +

**assumes**  $A10: \forall a b c d t. B a d t \wedge B b d c \wedge a \neq d$   
 $\longrightarrow (\exists x y. B a b x \wedge B a c y \wedge B x t y)$

**locale** *tarski* = *tarski-absolute* + *tarski-space*

### 3.2 Semimetric spaces satisfy the first three axioms

**context** *semimetric*

**begin**

**definition**  $smC :: 'p \Rightarrow 'p \Rightarrow 'p \Rightarrow 'p \Rightarrow \text{bool}$  ( $\langle \cdot - \equiv_{sm} - \cdot \rangle$  [99,99,99,99] 50)  
**where** [*simp*]:  $a b \equiv_{sm} c d \triangleq \text{dist } a b = \text{dist } c d$

**end**

**sublocale** *semimetric* < *tarski-first3 smC*

**proof**

**from** *symm* **show**  $\forall a b. a b \equiv_{sm} b a$  **by** *simp*

**show**  $\forall a b p q r s. a b \equiv_{sm} p q \wedge a b \equiv_{sm} r s \longrightarrow p q \equiv_{sm} r s$  **by** *simp*

**show**  $\forall a b c. a b \equiv_{sm} c c \longrightarrow a = b$  **by** *simp*

**qed**

### 3.3 Some consequences of the first three axioms

context *tarski-first3*

begin

lemma *A1'*:  $a b \equiv b a$   
by (*simp add: A1*)

lemma *A2'*:  $\llbracket a b \equiv p q; a b \equiv r s \rrbracket \Longrightarrow p q \equiv r s$

proof –

assume  $a b \equiv p q$  and  $a b \equiv r s$

with *A2* show ?thesis by blast

qed

lemma *A3'*:  $a b \equiv c c \Longrightarrow a = b$

by (*simp add: A3*)

theorem *th2-1*:  $a b \equiv a b$

proof –

from *A2'* [of  $b a a b a b$ ] and *A1'* [of  $b a$ ] show ?thesis by simp

qed

theorem *th2-2*:  $a b \equiv c d \Longrightarrow c d \equiv a b$

proof –

assume  $a b \equiv c d$

with *A2'* [of  $a b c d a b$ ] and *th2-1* [of  $a b$ ] show ?thesis by simp

qed

theorem *th2-3*:  $\llbracket a b \equiv c d; c d \equiv e f \rrbracket \Longrightarrow a b \equiv e f$

proof –

assume  $a b \equiv c d$

with *th2-2* [of  $a b c d$ ] have  $c d \equiv a b$  by simp

assume  $c d \equiv e f$

with *A2'* [of  $c d a b e f$ ] and  $\langle c d \equiv a b \rangle$  show ?thesis by simp

qed

theorem *th2-4*:  $a b \equiv c d \Longrightarrow b a \equiv c d$

proof –

assume  $a b \equiv c d$

with *th2-3* [of  $b a a b c d$ ] and *A1'* [of  $b a$ ] show ?thesis by simp

qed

theorem *th2-5*:  $a b \equiv c d \Longrightarrow a b \equiv d c$

proof –

assume  $a b \equiv c d$

with *th2-3* [of  $a b c d d c$ ] and *A1'* [of  $c d$ ] show ?thesis by simp

qed

definition *is-segment* :: 'p set  $\Rightarrow$  bool where

*is-segment*  $X \triangleq \exists x y. X = \{x, y\}$

**definition** *segments* :: 'p set set **where**  
*segments* = {X. is-segment X}

**definition** *SC* :: 'p set  $\Rightarrow$  'p set  $\Rightarrow$  bool **where**  
*SC* X Y  $\triangleq$   $\exists w x y z. X = \{w, x\} \wedge Y = \{y, z\} \wedge w x \equiv y z$

**definition** *SC-rel* :: ('p set  $\times$  'p set) set **where**  
*SC-rel* = {(X, Y) | X Y. *SC* X Y}

**lemma** *left-segment-congruence*:

**assumes**  $\{a, b\} = \{p, q\}$  **and**  $p q \equiv c d$   
**shows**  $a b \equiv c d$

**proof** *cases*

**assume**  $a = p$   
**with** *unordered-pair-element-equality* [of a b p q] **and**  $\langle \{a, b\} = \{p, q\} \rangle$   
**have**  $b = q$  **by** *simp*  
**with**  $\langle p q \equiv c d \rangle$  **and**  $\langle a = p \rangle$  **show** *?thesis* **by** *simp*

**next**

**assume**  $a \neq p$   
**with**  $\langle \{a, b\} = \{p, q\} \rangle$  **have**  $a = q$  **by** *auto*  
**with** *unordered-pair-element-equality* [of a b q p] **and**  $\langle \{a, b\} = \{p, q\} \rangle$   
**have**  $b = p$  **by** *auto*  
**with**  $\langle p q \equiv c d \rangle$  **and**  $\langle a = q \rangle$  **have**  $b a \equiv c d$  **by** *simp*  
**with** *th2-4* [of b a c d] **show** *?thesis* **by** *simp*

**qed**

**lemma** *right-segment-congruence*:

**assumes**  $\{c, d\} = \{p, q\}$  **and**  $a b \equiv p q$   
**shows**  $a b \equiv c d$

**proof** –

**from** *th2-2* [of a b p q] **and**  $\langle a b \equiv p q \rangle$  **have**  $p q \equiv a b$  **by** *simp*  
**with** *left-segment-congruence* [of c d p q a b] **and**  $\langle \{c, d\} = \{p, q\} \rangle$   
**have**  $c d \equiv a b$  **by** *simp*  
**with** *th2-2* [of c d a b] **show** *?thesis* **by** *simp*

**qed**

**lemma** *C-SC-equiv*:  $a b \equiv c d = SC \{a, b\} \{c, d\}$

**proof**

**assume**  $a b \equiv c d$   
**with** *SC-def* [of {a, b} {c, d}] **show** *SC* {a, b} {c, d} **by** *auto*

**next**

**assume** *SC* {a, b} {c, d}  
**with** *SC-def* [of {a, b} {c, d}]  
**obtain**  $w x y z$  **where**  $\{a, b\} = \{w, x\}$  **and**  $\{c, d\} = \{y, z\}$  **and**  $w x \equiv y z$   
**by** *blast*  
**from** *left-segment-congruence* [of a b w x y z] **and**  
 $\langle \{a, b\} = \{w, x\} \rangle$  **and**  
 $\langle w x \equiv y z \rangle$   
**have**  $a b \equiv y z$  **by** *simp*

**with** *right-segment-congruence* [of  $c d y z a b$ ] **and**  $\langle \{c, d\} = \{y, z\} \rangle$   
**show**  $a b \equiv c d$  **by** *simp*

**qed**

**lemma** *SC-rel-subset*:  $SC\text{-rel} \subseteq \text{segments} \times \text{segments}$

**proof** –

{ **fix**  $Z$   
**assume**  $Z \in SC\text{-rel}$   
**with** *SC-rel-def* **obtain**  $X Y$  **where**  $Z = (X, Y)$  **and** *SC*  $X Y$  **by** *auto*  
**from**  $\langle SC X Y \rangle$  **and** *SC-def* [of  $X Y$ ]  
**have**  $\exists w x. X = \{w, x\}$  **and**  $\exists y z. Y = \{y, z\}$  **by** *auto*  
**with** *is-segment-def* [of  $X$ ] **and** *is-segment-def* [of  $Y$ ]  
**have** *is-segment*  $X$  **and** *is-segment*  $Y$  **by** *auto*  
**with** *segments-def* **have**  $X \in \text{segments}$  **and**  $Y \in \text{segments}$  **by** *auto*  
**with**  $\langle Z = (X, Y) \rangle$  **have**  $Z \in \text{segments} \times \text{segments}$  **by** *simp* }  
**thus**  $SC\text{-rel} \subseteq \text{segments} \times \text{segments}$  **by** *auto*

**qed**

**lemmas** *SC-refl = th2-1* [simplified]

**lemma** *SC-rel-refl*: *refl-on segments SC-rel*

**proof** –

**note** *refl-on-def* [of *segments SC-rel*]  
**moreover**  
{ **fix**  $X$   
**assume**  $X \in \text{segments}$   
**with** *segments-def* **have** *is-segment*  $X$  **by** *auto*  
**with** *is-segment-def* [of  $X$ ] **obtain**  $x y$  **where**  $X = \{x, y\}$  **by** *auto*  
**with** *SC-def* [of  $X X$ ] **and** *SC-refl* **have**  $SC X X$  **by** (*simp add: C-SC-equiv*)  
**with** *SC-rel-def* **have**  $(X, X) \in SC\text{-rel}$  **by** *simp* }  
**hence**  $\forall X. X \in \text{segments} \longrightarrow (X, X) \in SC\text{-rel}$  **by** *simp*  
**ultimately show** *?thesis* **by** *simp*

**qed**

**lemma** *SC-sym*:

**assumes**  $SC X Y$

**shows**  $SC Y X$

**proof** –

**from** *SC-def* [of  $X Y$ ] **and**  $\langle SC X Y \rangle$   
**obtain**  $w x y z$  **where**  $X = \{w, x\}$  **and**  $Y = \{y, z\}$  **and**  $w x \equiv y z$   
**by** *auto*  
**from** *th2-2* [of  $w x y z$ ] **and**  $\langle w x \equiv y z \rangle$  **have**  $y z \equiv w x$  **by** *simp*  
**with** *SC-def* [of  $Y X$ ] **and**  $\langle X = \{w, x\} \rangle$  **and**  $\langle Y = \{y, z\} \rangle$   
**show**  $SC Y X$  **by** (*simp add: C-SC-equiv*)

**qed**

**lemma** *SC-sym'*:  $SC X Y = SC Y X$

**proof**

**assume**  $SC X Y$

```

  with SC-sym [of X Y] show SC Y X by simp
next
  assume SC Y X
  with SC-sym [of Y X] show SC X Y by simp
qed

```

```

lemma SC-rel-sym: sym SC-rel
proof -
  { fix X Y
    assume  $(X, Y) \in SC\text{-rel}$ 
    with SC-rel-def have SC X Y by simp
    with SC-sym' have SC Y X by simp
    with SC-rel-def have  $(Y, X) \in SC\text{-rel}$  by simp }
  with sym-def [of SC-rel] show ?thesis by blast
qed

```

```

lemma SC-trans:
  assumes SC X Y and SC Y Z
  shows SC X Z
proof -
  from SC-def [of X Y] and  $\langle SC\ X\ Y \rangle$ 
    obtain w x y z where  $X = \{w, x\}$  and  $Y = \{y, z\}$  and  $w\ x \equiv y\ z$ 
    by auto
  from SC-def [of Y Z] and  $\langle SC\ Y\ Z \rangle$ 
    obtain p q r s where  $Y = \{p, q\}$  and  $Z = \{r, s\}$  and  $p\ q \equiv r\ s$  by auto
  from  $\langle Y = \{y, z\} \rangle$  and  $\langle Y = \{p, q\} \rangle$  and  $\langle p\ q \equiv r\ s \rangle$ 
    have  $y\ z \equiv r\ s$  by (simp add: C-SC-equiv)
  with th2-3 [of w x y z r s] and  $\langle w\ x \equiv y\ z \rangle$  have  $w\ x \equiv r\ s$  by simp
  with SC-def [of X Z] and  $\langle X = \{w, x\} \rangle$  and  $\langle Z = \{r, s\} \rangle$ 
    show SC X Z by (simp add: C-SC-equiv)
qed

```

```

lemma SC-rel-trans: trans SC-rel
proof -
  { fix X Y Z
    assume  $(X, Y) \in SC\text{-rel}$  and  $(Y, Z) \in SC\text{-rel}$ 
    with SC-rel-def have SC X Y and SC Y Z by auto
    with SC-trans [of X Y Z] have SC X Z by simp
    with SC-rel-def have  $(X, Z) \in SC\text{-rel}$  by simp }
  with trans-def [of SC-rel] show ?thesis by blast
qed

```

```

lemma A3-reversed:
  assumes  $a\ a \equiv b\ c$ 
  shows  $b = c$ 
proof -
  from  $\langle a\ a \equiv b\ c \rangle$  have  $b\ c \equiv a\ a$  by (rule th2-2)
  thus  $b = c$  by (rule A3')
qed

```

**lemma** *equiv-segments-SC-rel: equiv segments SC-rel*  
**by** (*simp add: SC-rel-subset equiv-def SC-rel-refl SC-rel-sym SC-rel-trans*)

**end**

### 3.4 Some consequences of the first five axioms

**context** *tarski-first5*

**begin**

**lemma**  $A_4'$ :  $\exists x. B\ q\ a\ x \wedge a\ x \equiv b\ c$   
**by** (*simp add: A<sub>4</sub> [simplified]*)

**theorem** *th2-8*:  $a\ a \equiv b\ b$

**proof** –

**from**  $A_4'$  [*of - a b b*] **obtain**  $x$  **where**  $a\ x \equiv b\ b$  **by** *auto*

**with**  $A_3'$  [*of a x b*] **have**  $x = a$  **by** *simp*

**with**  $\langle a\ x \equiv b\ b \rangle$  **show** *?thesis* **by** *simp*

**qed**

**definition** *OFS* ::  $[ 'p, 'p, 'p, 'p, 'p, 'p, 'p ] \Rightarrow bool$  **where**

$OFS\ a\ b\ c\ d\ a'\ b' c' d' \triangleq$

$B\ a\ b\ c \wedge B\ a'\ b' c' \wedge a\ b \equiv a'\ b' \wedge b\ c \equiv b' c' \wedge a\ d \equiv a' d' \wedge b\ d \equiv b' d'$

**lemma**  $A_5'$ :  $\llbracket OFS\ a\ b\ c\ d\ a'\ b' c' d'; a \neq b \rrbracket \Longrightarrow c\ d \equiv c' d'$

**proof** –

**assume**  $OFS\ a\ b\ c\ d\ a'\ b' c' d'$  **and**  $a \neq b$

**with**  $A_5$  **and** *OFS-def* **show** *?thesis* **by** *blast*

**qed**

**theorem** *th2-11*:

**assumes** *hypotheses*:

$B\ a\ b\ c$

$B\ a'\ b' c'$

$a\ b \equiv a'\ b'$

$b\ c \equiv b' c'$

**shows**  $a\ c \equiv a' c'$

**proof** *cases*

**assume**  $a = b$

**with**  $\langle a\ b \equiv a'\ b' \rangle$  **have**  $a' = b'$  **by** (*simp add: A<sub>3</sub>-reversed*)

**with**  $\langle b\ c \equiv b' c' \rangle$  **and**  $\langle a = b \rangle$  **show** *?thesis* **by** *simp*

**next**

**assume**  $a \neq b$

**moreover**

**note**  $A_5'$  [*of a b c a a' b' c' a'*] **and**

*unordered-pair-equality* [*of a c*] **and**

*unordered-pair-equality* [*of a' c'*]

**moreover**

**from** *OFS-def* [*of a b c a a' b' c' a'*] **and**

*hypotheses and*  
*th2-8 [of a a'] and*  
*unordered-pair-equality [of a b] and*  
*unordered-pair-equality [of a' b']*  
**have** *OFS a b c a a' b' c' a'* **by** (*simp add: C-SC-equiv*)  
**ultimately show** *?thesis* **by** (*simp add: C-SC-equiv*)  
**qed**

**lemma** *A4-unique:*

**assumes**  $q \neq a$  **and**  $B q a x$  **and**  $a x \equiv b c$   
**and**  $B q a x'$  **and**  $a x' \equiv b c$   
**shows**  $x = x'$

**proof** –

**from** *SC-sym'* **and** *SC-trans* **and** *C-SC-equiv* **and**  $\langle a x' \equiv b c \rangle$  **and**  $\langle a x \equiv b$   
 $c \rangle$

**have**  $a x \equiv a x'$  **by** *blast*

**with** *th2-11 [of q a x q a x']* **and**  $\langle B q a x \rangle$  **and**  $\langle B q a x' \rangle$  **and** *SC-refl*

**have**  $q x \equiv q x'$  **by** *simp*

**with** *OFS-def [of q a x x q a x']* **and**

$\langle B q a x \rangle$  **and**

*SC-refl* **and**

$\langle a x \equiv a x' \rangle$

**have** *OFS q a x x q a x x'* **by** *simp*

**with** *A5' [of q a x x q a x x']* **and**  $\langle q \neq a \rangle$  **have**  $x x \equiv x x'$  **by** *simp*

**thus**  $x = x'$  **by** (*rule A3-reversed*)

**qed**

**theorem** *th2-12:*

**assumes**  $q \neq a$

**shows**  $\exists!x. B q a x \wedge a x \equiv b c$

**using**  $\langle q \neq a \rangle$  **and** *A4'* **and** *A4-unique*

**by** *blast*

**end**

### 3.5 Simple theorems about betweenness

**theorem** (*in tarski-first5*) *th3-1: B a b b*

**proof** –

**from** *A4 [rule-format, of a b b b]* **obtain**  $x$  **where**  $B a b x$  **and**  $b x \equiv b b$  **by**  
*auto*

**from** *A3 [rule-format, of b x b]* **and**  $\langle b x \equiv b b \rangle$  **have**  $b = x$  **by** *simp*

**with**  $\langle B a b x \rangle$  **show**  $B a b b$  **by** *simp*

**qed**

**context** *tarski-absolute-space*

**begin**

**lemma** *A6':*

**assumes**  $B a b a$

**shows**  $a = b$

**proof** –  
**from**  $A6$  **and**  $\langle B a b a \rangle$  **show**  $a = b$  **by** *simp*  
**qed**

**lemma**  $A7'$ :  
**assumes**  $B a p c$  **and**  $B b q c$   
**shows**  $\exists x. B p x b \wedge B q x a$

**proof** –  
**from**  $A7$  **and**  $\langle B a p c \rangle$  **and**  $\langle B b q c \rangle$  **show** *?thesis* **by** *blast*  
**qed**

**lemma**  $A11'$ :  
**assumes**  $\forall x y. x \in X \wedge y \in Y \longrightarrow B a x y$   
**shows**  $\exists b. \forall x y. x \in X \wedge y \in Y \longrightarrow B x b y$

**proof** –  
**from** *assms* **have**  $\exists a. \forall x y. x \in X \wedge y \in Y \longrightarrow B a x y$  **by** (*rule exI*)  
**thus**  $\exists b. \forall x y. x \in X \wedge y \in Y \longrightarrow B x b y$  **by** (*rule A11 [rule-format]*)  
**qed**

**theorem**  $th3-2$ :  
**assumes**  $B a b c$   
**shows**  $B c b a$

**proof** –  
**from**  $th3-1$  **have**  $B b c c$  **by** *simp*  
**with**  $A7'$  **and**  $\langle B a b c \rangle$  **obtain**  $x$  **where**  $B b x b$  **and**  $B c x a$  **by** *blast*  
**from**  $A6'$  **and**  $\langle B b x b \rangle$  **have**  $x = b$  **by** *auto*  
**with**  $\langle B c x a \rangle$  **show**  $B c b a$  **by** *simp*  
**qed**

**theorem**  $th3-4$ :  
**assumes**  $B a b c$  **and**  $B b a c$   
**shows**  $a = b$

**proof** –  
**from**  $\langle B a b c \rangle$  **and**  $\langle B b a c \rangle$  **and**  $A7'$  [*of a b c b a*]  
**obtain**  $x$  **where**  $B b x b$  **and**  $B a x a$  **by** *auto*  
**hence**  $b = x$  **and**  $a = x$  **by** (*simp-all add: A6'*)  
**thus**  $a = b$  **by** *simp*  
**qed**

**theorem**  $th3-5-1$ :  
**assumes**  $B a b d$  **and**  $B b c d$   
**shows**  $B a b c$

**proof** –  
**from**  $\langle B a b d \rangle$  **and**  $\langle B b c d \rangle$  **and**  $A7'$  [*of a b d b c*]  
**obtain**  $x$  **where**  $B b x b$  **and**  $B c x a$  **by** *auto*  
**from**  $\langle B b x b \rangle$  **have**  $b = x$  **by** (*rule A6'*)  
**with**  $\langle B c x a \rangle$  **have**  $B c b a$  **by** *simp*  
**thus**  $B a b c$  **by** (*rule th3-2*)  
**qed**

**theorem** *th3-6-1*:

assumes  $B a b c$  and  $B a c d$

shows  $B b c d$

**proof** –

from  $\langle B a c d \rangle$  and  $\langle B a b c \rangle$  and *th3-2* have  $B d c a$  and  $B c b a$  by *fast+*

hence  $B d c b$  by (rule *th3-5-1*)

thus  $B b c d$  by (rule *th3-2*)

**qed**

**theorem** *th3-7-1*:

assumes  $b \neq c$  and  $B a b c$  and  $B b c d$

shows  $B a c d$

**proof** –

from  $A_4'$  obtain  $x$  where  $B a c x$  and  $c x \equiv c d$  by *fast*

from  $\langle B a b c \rangle$  and  $\langle B a c x \rangle$  have  $B b c x$  by (rule *th3-6-1*)

have  $c d \equiv c d$  by (rule *th2-1*)

with  $\langle b \neq c \rangle$  and  $\langle B b c x \rangle$  and  $\langle c x \equiv c d \rangle$  and  $\langle B b c d \rangle$

have  $x = d$  by (rule *A4-unique*)

with  $\langle B a c x \rangle$  show  $B a c d$  by *simp*

**qed**

**theorem** *th3-7-2*:

assumes  $b \neq c$  and  $B a b c$  and  $B b c d$

shows  $B a b d$

**proof** –

from  $\langle B b c d \rangle$  and  $\langle B a b c \rangle$  and *th3-2* have  $B d c b$  and  $B c b a$  by *fast+*

with  $\langle b \neq c \rangle$  and *th3-7-1* [*of c b d a*] have  $B d b a$  by *simp*

thus  $B a b d$  by (rule *th3-2*)

**qed**

**end**

### 3.6 Simple theorems about congruence and betweenness

**definition** (in *tarski-first5*)  $Col :: 'p \Rightarrow 'p \Rightarrow 'p \Rightarrow bool$  where

$Col a b c \triangleq B a b c \vee B b c a \vee B c a b$

**end**

## 4 Real Euclidean space and Tarski's axioms

**theory** *Euclid-Tarski*

**imports** *Tarski*

**begin**

### 4.1 Real Euclidean space satisfies the first five axioms

**abbreviation**

$real\text{-euclid-}C :: [real^{(n::finite)}, real^{(n)}, real^{(n)}, real^{(n)}] \Rightarrow bool$

( $\langle - \equiv_{\mathbf{R}} - \rangle$  [99,99,99,99] 50) **where**  
*real-euclid-C*  $\triangleq$  *norm-metric.smC*

**definition** *real-euclid-B* :: [*real*<sup>^</sup>(*n*::*finite*), *real*<sup>^</sup>(*n*), *real*<sup>^</sup>(*n*)]  $\Rightarrow$  *bool*  
( $\langle B_{\mathbf{R}} - - \rangle$  [99,99,99] 50) **where**  
 $B_{\mathbf{R}} a b c \triangleq \exists l. 0 \leq l \wedge l \leq 1 \wedge b - a = l *_{\mathbf{R}} (c - a)$

**interpretation** *real-euclid*: *tarski-first5 real-euclid-C real-euclid-B*  
**proof**

By virtue of being a semimetric space, real Euclidean space is already known to satisfy the first three axioms.

```
{ fix q a b c
  have  $\exists x. B_{\mathbf{R}} q a x \wedge a x \equiv_{\mathbf{R}} b c$ 
  proof cases
    assume  $q = a$ 
    let  $?x = a + c - b$ 
    have  $B_{\mathbf{R}} q a ?x$ 
    proof -
      let  $?l = 0$  :: real
      note real-euclid-B-def [of  $q a ?x$ ]
      moreover
        have  $?l \geq 0$  and  $?l \leq 1$  by auto
      moreover
        from  $\langle q = a \rangle$  have  $a - q = 0$  by simp
        hence  $a - q = ?l *_{\mathbf{R}} (?x - q)$  by simp
        ultimately show ?thesis by auto
    qed
  moreover
    have  $a - ?x = b - c$  by simp
    hence  $a ?x \equiv_{\mathbf{R}} b c$  by (simp add: field-simps)
    ultimately show ?thesis by blast
next
  assume  $q \neq a$ 
  hence norm-dist  $q a > 0$  by simp
  let  $?k = \text{norm-dist } b c / \text{norm-dist } q a$ 
  let  $?x = a + ?k *_{\mathbf{R}} (a - q)$ 
  have  $B_{\mathbf{R}} q a ?x$ 
  proof -
    let  $?l = 1 / (1 + ?k)$ 
    have  $?l > 0$  by (simp add: add-pos-nonneg)
    note real-euclid-B-def [of  $q a ?x$ ]
    moreover
      have  $?l \geq 0$  and  $?l \leq 1$  by (auto simp add: add-pos-nonneg)
    moreover
      from scaleR-left-distrib [of  $1 ?k a - q$ ]
        have  $(1 + ?k) *_{\mathbf{R}} (a - q) = ?x - q$  by simp
      hence  $?l *_{\mathbf{R}} ((1 + ?k) *_{\mathbf{R}} (a - q)) = ?l *_{\mathbf{R}} (?x - q)$  by simp
      with  $\langle ?l > 0 \rangle$  and scaleR-right-diff-distrib [of  $?l ?x q$ ]
        have  $a - q = ?l *_{\mathbf{R}} (?x - q)$  by simp
```

```

ultimately show  $B_{\mathbb{R}} q a \ ?x$  by blast
qed
moreover
  have  $a \ ?x \equiv_{\mathbb{R}} b c$ 
  proof -
    from norm-scaleR [of ?k a - q] have
      norm-dist a ?x = |?k| * norm (a - q) by simp
    also have
      ... = ?k * norm (a - q) by simp
    also from norm-metric.symm [of q a] have
      ... = ?k * norm-dist q a by simp
    finally have
      norm-dist a ?x = norm-dist b c / norm-dist q a * norm-dist q a .
    with <norm-dist q a > 0 show  $a \ ?x \equiv_{\mathbb{R}} b c$  by auto
  qed
ultimately show ?thesis by blast
qed }
thus  $\forall q a b c. \exists x. B_{\mathbb{R}} q a x \wedge a x \equiv_{\mathbb{R}} b c$  by auto
{ fix a b c d a' b' c' d'
  assume  $a \neq b$  and
     $B_{\mathbb{R}} a b c$  and
     $B_{\mathbb{R}} a' b' c'$  and
     $a b \equiv_{\mathbb{R}} a' b'$  and
     $b c \equiv_{\mathbb{R}} b' c'$  and
     $a d \equiv_{\mathbb{R}} a' d'$  and
     $b d \equiv_{\mathbb{R}} b' d'$ 
  have  $c d \equiv_{\mathbb{R}} c' d'$ 
  proof -
    { fix m
      fix  $p q r :: \text{real}^{\wedge}('n::\text{finite})$ 
      assume  $0 \leq m$  and
         $m \leq 1$  and
         $p \neq q$  and
         $q - p = m *_{\mathbb{R}} (r - p)$ 
      from < $p \neq q$ > and < $q - p = m *_{\mathbb{R}} (r - p)$ > have  $m \neq 0$ 
      proof -
        { assume  $m = 0$ 
          with < $q - p = m *_{\mathbb{R}} (r - p)$ > have  $q - p = 0$  by simp
          with < $p \neq q$ > have False by simp }
        thus ?thesis ..
      qed
      with < $m \geq 0$ > have  $m > 0$  by simp
      from < $q - p = m *_{\mathbb{R}} (r - p)$ > and
        scaleR-right-diff-distrib [of m r p]
      have  $q - p = m *_{\mathbb{R}} r - m *_{\mathbb{R}} p$  by simp
      hence  $q - p - q + p - m *_{\mathbb{R}} r =$ 
         $m *_{\mathbb{R}} r - m *_{\mathbb{R}} p - q + p - m *_{\mathbb{R}} r$ 
      by simp
      with scaleR-left-diff-distrib [of 1 m p] and

```

*scaleR-left-diff-distrib* [of 1 m q]  
**have**  $(1 - m) *_{\mathbb{R}} p - (1 - m) *_{\mathbb{R}} q = m *_{\mathbb{R}} q - m *_{\mathbb{R}} r$  **by** *auto*  
**with** *scaleR-right-diff-distrib* [of 1 - m p q] **and**  
*scaleR-right-diff-distrib* [of m q r]  
**have**  $(1 - m) *_{\mathbb{R}} (p - q) = m *_{\mathbb{R}} (q - r)$  **by** *simp*  
**with** *norm-scaleR* [of 1 - m p - q] **and** *norm-scaleR* [of m q - r]  
**have**  $|1 - m| * \text{norm} (p - q) = |m| * \text{norm} (q - r)$  **by** *simp*  
**with**  $\langle m > 0 \rangle$  **and**  $\langle m \leq 1 \rangle$   
**have**  $\text{norm} (q - r) = (1 - m) / m * \text{norm} (p - q)$  **by** *simp*  
**moreover from**  $\langle p \neq q \rangle$  **have**  $\text{norm} (p - q) \neq 0$  **by** *simp*  
**ultimately**  
**have**  $\text{norm} (q - r) / \text{norm} (p - q) = (1 - m) / m$  **by** *simp*  
**with**  $\langle m \neq 0 \rangle$  **have**  
 $\text{norm-dist } q \ r / \text{norm-dist } p \ q = (1 - m) / m$  **and**  $m \neq 0$  **by** *auto* }  
**note** *linelemma* = *this*  
**from** *real-euclid-B-def* [of a b c] **and**  $\langle B_{\mathbb{R}} \ a \ b \ c \rangle$   
**obtain** l **where**  $0 \leq l$  **and**  $l \leq 1$  **and**  $b - a = l *_{\mathbb{R}} (c - a)$  **by** *auto*  
**from** *real-euclid-B-def* [of a' b' c'] **and**  $\langle B_{\mathbb{R}} \ a' \ b' \ c' \rangle$   
**obtain** l' **where**  $0 \leq l'$  **and**  $l' \leq 1$  **and**  $b' - a' = l' *_{\mathbb{R}} (c' - a')$  **by** *auto*  
**from**  $\langle a \neq b \rangle$  **and**  $\langle a \ b \equiv_{\mathbb{R}} \ a' \ b' \rangle$  **have**  $a' \neq b'$  **by** *auto*  
**from** *linelemma* [of l a b c] **and**  
 $\langle l \geq 0 \rangle$  **and**  
 $\langle l \leq 1 \rangle$  **and**  
 $\langle a \neq b \rangle$  **and**  
 $\langle b - a = l *_{\mathbb{R}} (c - a) \rangle$   
**have**  $l \neq 0$  **and**  $(1 - l) / l = \text{norm-dist } b \ c / \text{norm-dist } a \ b$  **by** *auto*  
**from**  $\langle (1 - l) / l = \text{norm-dist } b \ c / \text{norm-dist } a \ b \rangle$  **and**  
 $\langle a \ b \equiv_{\mathbb{R}} \ a' \ b' \rangle$  **and**  
 $\langle b \ c \equiv_{\mathbb{R}} \ b' \ c' \rangle$   
**have**  $(1 - l) / l = \text{norm-dist } b' \ c' / \text{norm-dist } a' \ b'$  **by** *simp*  
**with** *linelemma* [of l' a' b' c'] **and**  
 $\langle l' \geq 0 \rangle$  **and**  
 $\langle l' \leq 1 \rangle$  **and**  
 $\langle a' \neq b' \rangle$  **and**  
 $\langle b' - a' = l' *_{\mathbb{R}} (c' - a') \rangle$   
**have**  $l' \neq 0$  **and**  $(1 - l) / l = (1 - l') / l'$  **by** *auto*  
**from**  $\langle (1 - l) / l = (1 - l') / l' \rangle$   
**have**  $(1 - l) / l * l * l' = (1 - l') / l' * l * l'$  **by** *simp*  
**with**  $\langle l \neq 0 \rangle$  **and**  $\langle l' \neq 0 \rangle$  **have**  $(1 - l) * l' = (1 - l') * l$  **by** *simp*  
**with** *left-diff-distrib* [of 1 l l'] **and** *left-diff-distrib* [of 1 l' l]  
**have**  $l = l'$  **by** *simp*  
**{** **fix** m  
**fix** p q r s :: *real*  $\wedge$  ('n::finite)  
**assume**  $m \neq 0$  **and**  
 $q - p = m *_{\mathbb{R}} (r - p)$   
**with** *scaleR-scaleR* **have**  $r - p = (1/m) *_{\mathbb{R}} (q - p)$  **by** *simp*  
**with** *cosine-rule* [of r s p]  
**have**  $(\text{norm-dist } r \ s)^2 = (\text{norm-dist } r \ p)^2 + (\text{norm-dist } p \ s)^2 +$   
 $2 * (((1/m) *_{\mathbb{R}} (q - p)) \cdot (p - s))$

by *simp*  
 also from *inner-scaleR-left* [of  $1/m \ q - p \ p - s$ ]  
 have ... =  
 $(\text{norm-dist } r \ p)^2 + (\text{norm-dist } p \ s)^2 + 2/m * ((q - p) \cdot (p - s))$   
 by *simp*  
 also from  $\langle m \neq 0 \rangle$  and *cosine-rule* [of  $q \ s \ p$ ]  
 have ... =  $(\text{norm-dist } r \ p)^2 + (\text{norm-dist } p \ s)^2 +$   
 $1/m * ((\text{norm-dist } q \ s)^2 - (\text{norm-dist } q \ p)^2 - (\text{norm-dist } p \ s)^2)$   
 by *simp*  
 finally have  $(\text{norm-dist } r \ s)^2 = (\text{norm-dist } r \ p)^2 + (\text{norm-dist } p \ s)^2 +$   
 $1/m * ((\text{norm-dist } q \ s)^2 - (\text{norm-dist } q \ p)^2 - (\text{norm-dist } p \ s)^2)$  .  
 moreover  
 { from *norm-dist-dot* [of  $r \ p$ ] and  $\langle r - p = (1/m) *_R (q - p) \rangle$   
 have  $(\text{norm-dist } r \ p)^2 = ((1/m) *_R (q - p)) \cdot ((1/m) *_R (q - p))$   
 by *simp*  
 also from *inner-scaleR-left* [of  $1/m \ q - p$ ] and  
*inner-scaleR-right* [of  $- 1/m \ q - p$ ]  
 have ... =  $1/m^2 * ((q - p) \cdot (q - p))$   
 by (*simp add: power2-eq-square*)  
 also from *norm-dist-dot* [of  $q \ p$ ] have ... =  $1/m^2 * (\text{norm-dist } q \ p)^2$   
 by *simp*  
 finally have  $(\text{norm-dist } r \ p)^2 = 1/m^2 * (\text{norm-dist } q \ p)^2$  . }  
 ultimately have  
 $(\text{norm-dist } r \ s)^2 = 1/m^2 * (\text{norm-dist } q \ p)^2 + (\text{norm-dist } p \ s)^2 +$   
 $1/m * ((\text{norm-dist } q \ s)^2 - (\text{norm-dist } q \ p)^2 - (\text{norm-dist } p \ s)^2)$   
 by *simp*  
 with *norm-metric.symm* [of  $q \ p$ ]  
 have  $(\text{norm-dist } r \ s)^2 = 1/m^2 * (\text{norm-dist } p \ q)^2 + (\text{norm-dist } p \ s)^2 +$   
 $1/m * ((\text{norm-dist } q \ s)^2 - (\text{norm-dist } p \ q)^2 - (\text{norm-dist } p \ s)^2)$   
 by *simp* }  
 note *fiveseglemma* = *this*  
 from *fiveseglemma* [of  $l \ b \ a \ c \ d$ ] and  $\langle l \neq 0 \rangle$  and  $\langle b - a = l *_R (c - a) \rangle$   
 have  $(\text{norm-dist } c \ d)^2 = 1/l^2 * (\text{norm-dist } a \ b)^2 + (\text{norm-dist } a \ d)^2 +$   
 $1/l * ((\text{norm-dist } b \ d)^2 - (\text{norm-dist } a \ b)^2 - (\text{norm-dist } a \ d)^2)$   
 by *simp*  
 also from  $\langle l = l' \rangle$  and  
 $\langle a \ b \equiv_{\mathbb{R}} a' \ b' \rangle$  and  
 $\langle a \ d \equiv_{\mathbb{R}} a' \ d' \rangle$  and  
 $\langle b \ d \equiv_{\mathbb{R}} b' \ d' \rangle$   
 have ... =  $1/l'^2 * (\text{norm-dist } a' \ b')^2 + (\text{norm-dist } a' \ d')^2 +$   
 $1/l' * ((\text{norm-dist } b' \ d')^2 - (\text{norm-dist } a' \ b')^2 - (\text{norm-dist } a' \ d')^2)$   
 by *simp*  
 also from *fiveseglemma* [of  $l' \ b' \ a' \ c' \ d'$ ] and  
 $\langle l' \neq 0 \rangle$  and  
 $\langle b' - a' = l' *_R (c' - a') \rangle$   
 have ... =  $(\text{norm-dist } c' \ d')^2$  by *simp*  
 finally have  $(\text{norm-dist } c \ d)^2 = (\text{norm-dist } c' \ d')^2$  .  
 hence *sqrt*  $((\text{norm-dist } c \ d)^2) = \text{sqrt} ((\text{norm-dist } c' \ d')^2)$  by *simp*  
 with *real-sqrt-abs* show  $c \ d \equiv_{\mathbb{R}} c' \ d'$  by *simp*

**qed }**  
**thus**  $\forall a b c d a' b' c' d'.$   
 $a \neq b \wedge B_{\mathbf{R}} a b c \wedge B_{\mathbf{R}} a' b' c' \wedge$   
 $a b \equiv_{\mathbf{R}} a' b' \wedge b c \equiv_{\mathbf{R}} b' c' \wedge a d \equiv_{\mathbf{R}} a' d' \wedge b d \equiv_{\mathbf{R}} b' d' \longrightarrow$   
 $c d \equiv_{\mathbf{R}} c' d'$   
**by blast**  
**qed**

## 4.2 Real Euclidean space also satisfies axioms 6, 7, and 11

**lemma** *rearrange-real-euclid-B:*

**fixes**  $w y z :: \text{real}^{\wedge(n)}$  **and**  $h$

**shows**  $y - w = h *_{\mathbf{R}} (z - w) \longleftrightarrow y = h *_{\mathbf{R}} z + (1 - h) *_{\mathbf{R}} w$

**proof**

**assume**  $y - w = h *_{\mathbf{R}} (z - w)$

**hence**  $y - w + w = h *_{\mathbf{R}} (z - w) + w$  **by simp**

**hence**  $y = h *_{\mathbf{R}} (z - w) + w$  **by simp**

**with** *scaleR-right-diff-distrib* [of  $h z w$ ]

**have**  $y = h *_{\mathbf{R}} z + w - h *_{\mathbf{R}} w$  **by simp**

**with** *scaleR-left-diff-distrib* [of  $1 h w$ ]

**show**  $y = h *_{\mathbf{R}} z + (1 - h) *_{\mathbf{R}} w$  **by simp**

**next**

**assume**  $y = h *_{\mathbf{R}} z + (1 - h) *_{\mathbf{R}} w$

**with** *scaleR-left-diff-distrib* [of  $1 h w$ ]

**have**  $y = h *_{\mathbf{R}} z + w - h *_{\mathbf{R}} w$  **by simp**

**with** *scaleR-right-diff-distrib* [of  $h z w$ ]

**have**  $y = h *_{\mathbf{R}} (z - w) + w$  **by simp**

**hence**  $y - w + w = h *_{\mathbf{R}} (z - w) + w$  **by simp**

**thus**  $y - w = h *_{\mathbf{R}} (z - w)$  **by simp**

**qed**

**interpretation** *real-euclid: tarski-absolute-space real-euclid-C real-euclid-B*

**proof**

**{ fix**  $a b$

**assume**  $B_{\mathbf{R}} a b a$

**with** *real-euclid-B-def* [of  $a b a$ ]

**obtain**  $l$  **where**  $b - a = l *_{\mathbf{R}} (a - a)$  **by auto**

**hence**  $a = b$  **by simp }**

**thus**  $\forall a b. B_{\mathbf{R}} a b a \longrightarrow a = b$  **by auto**

**{ fix**  $a b c p q$

**assume**  $B_{\mathbf{R}} a p c$  **and**  $B_{\mathbf{R}} b q c$

**from** *real-euclid-B-def* [of  $a p c$ ] **and**  $\langle B_{\mathbf{R}} a p c \rangle$

**obtain**  $i$  **where**  $i \geq 0$  **and**  $i \leq 1$  **and**  $p - a = i *_{\mathbf{R}} (c - a)$  **by auto**

**have**  $\exists x. B_{\mathbf{R}} p x b \wedge B_{\mathbf{R}} q x a$

**proof cases**

**assume**  $i = 0$

**with**  $\langle p - a = i *_{\mathbf{R}} (c - a) \rangle$  **have**  $p = a$  **by simp**

**hence**  $p - a = 0 *_{\mathbf{R}} (b - p)$  **by simp**

**moreover have**  $(0::\text{real}) \geq 0$  **and**  $(0::\text{real}) \leq 1$  **by auto**

**moreover note** *real-euclid-B-def* [of  $p$   $a$   $b$ ]  
**ultimately have**  $B_{\mathbb{R}} p a b$  **by** *auto*  
**moreover**  
**{** **have**  $a - q = 1 *_R (a - q)$  **by** *simp*  
**moreover have**  $(1::real) \geq 0$  **and**  $(1::real) \leq 1$  **by** *auto*  
**moreover note** *real-euclid-B-def* [of  $q$   $a$   $a$ ]  
**ultimately have**  $B_{\mathbb{R}} q a a$  **by** *blast* **}**  
**ultimately have**  $B_{\mathbb{R}} p a b \wedge B_{\mathbb{R}} q a a$  **by** *simp*  
**thus**  $\exists x. B_{\mathbb{R}} p x b \wedge B_{\mathbb{R}} q x a$  **by** *auto*  
**next**  
**assume**  $i \neq 0$   
**from** *real-euclid-B-def* [of  $b$   $q$   $c$ ] **and**  $\langle B_{\mathbb{R}} b q c \rangle$   
**obtain**  $j$  **where**  $j \geq 0$  **and**  $j \leq 1$  **and**  $q - b = j *_R (c - b)$  **by** *auto*  
**from**  $\langle i \geq 0 \rangle$  **and**  $\langle i \leq 1 \rangle$   
**have**  $1 - i \geq 0$  **and**  $1 - i \leq 1$  **by** *auto*  
**from**  $\langle j \geq 0 \rangle$  **and**  $\langle 1 - i \geq 0 \rangle$   
**have**  $j * (1 - i) \geq 0$  **by** *auto*  
**with**  $\langle i \geq 0 \rangle$  **and**  $\langle i \neq 0 \rangle$  **have**  $i + j * (1 - i) > 0$  **by** *simp*  
**hence**  $i + j * (1 - i) \neq 0$  **by** *simp*  
**let**  $?l = j * (1 - i) / (i + j * (1 - i))$   
**from** *diff-divide-distrib* [of  $i + j * (1 - i)$   $j * (1 - i)$   $i + j * (1 - i)$ ] **and**  
 $\langle i + j * (1 - i) \neq 0 \rangle$   
**have**  $1 - ?l = i / (i + j * (1 - i))$  **by** *simp*  
**let**  $?k = i * (1 - j) / (j + i * (1 - j))$   
**from** *right-diff-distrib* [of  $i$   $1$   $j$ ] **and**  
*right-diff-distrib* [of  $j$   $1$   $i$ ] **and**  
*mult.commute* [of  $i$   $j$ ] **and**  
*add.commute* [of  $i$   $j$ ]  
**have**  $j + i * (1 - j) = i + j * (1 - i)$  **by** *simp*  
**with**  $\langle i + j * (1 - i) \neq 0 \rangle$  **have**  $j + i * (1 - j) \neq 0$  **by** *simp*  
**with** *diff-divide-distrib* [of  $j + i * (1 - j)$   $i * (1 - j)$   $j + i * (1 - j)$ ]  
**have**  $1 - ?k = j / (j + i * (1 - j))$  **by** *simp*  
**with**  $\langle 1 - ?l = i / (i + j * (1 - i)) \rangle$  **and**  
 $\langle j + i * (1 - j) = i + j * (1 - i) \rangle$  **and**  
*times-divide-eq-left* [of  $-i + j * (1 - i)$ ] **and**  
*mult.commute* [of  $i$   $j$ ]  
**have**  $(1 - ?l) * j = (1 - ?k) * i$  **by** *simp*  
**moreover**  
**{** **from**  $\langle 1 - ?k = j / (j + i * (1 - j)) \rangle$  **and**  
 $\langle j + i * (1 - j) = i + j * (1 - i) \rangle$   
**have**  $?l = (1 - ?k) * (1 - i)$  **by** *simp* **}**  
**moreover**  
**{** **from**  $\langle 1 - ?l = i / (i + j * (1 - i)) \rangle$  **and**  
 $\langle j + i * (1 - j) = i + j * (1 - i) \rangle$   
**have**  $(1 - ?l) * (1 - j) = ?k$  **by** *simp* **}**  
**ultimately**  
**have**  $?l *_R a + ((1 - ?l) * j) *_R c + ((1 - ?l) * (1 - j)) *_R b =$   
 $?k *_R b + ((1 - ?k) * i) *_R c + ((1 - ?k) * (1 - i)) *_R a$   
**by** *simp*

**with** *scaleR-scaleR*  
**have**  $?l *_{\mathbb{R}} a + (1 - ?l) *_{\mathbb{R}} j *_{\mathbb{R}} c + (1 - ?l) *_{\mathbb{R}} (1 - j) *_{\mathbb{R}} b =$   
 $?k *_{\mathbb{R}} b + (1 - ?k) *_{\mathbb{R}} i *_{\mathbb{R}} c + (1 - ?k) *_{\mathbb{R}} (1 - i) *_{\mathbb{R}} a$   
**by** *simp*  
**with** *scaleR-right-distrib* [of  $(1 - ?l) j *_{\mathbb{R}} c (1 - j) *_{\mathbb{R}} b$ ] **and**  
*scaleR-right-distrib* [of  $(1 - ?k) i *_{\mathbb{R}} c (1 - i) *_{\mathbb{R}} a$ ] **and**  
*add.assoc* [of  $?l *_{\mathbb{R}} a (1 - ?l) *_{\mathbb{R}} j *_{\mathbb{R}} c (1 - ?l) *_{\mathbb{R}} (1 - j) *_{\mathbb{R}} b$ ] **and**  
*add.assoc* [of  $?k *_{\mathbb{R}} b (1 - ?k) *_{\mathbb{R}} i *_{\mathbb{R}} c (1 - ?k) *_{\mathbb{R}} (1 - i) *_{\mathbb{R}} a$ ]  
**have**  $?l *_{\mathbb{R}} a + (1 - ?l) *_{\mathbb{R}} (j *_{\mathbb{R}} c + (1 - j) *_{\mathbb{R}} b) =$   
 $?k *_{\mathbb{R}} b + (1 - ?k) *_{\mathbb{R}} (i *_{\mathbb{R}} c + (1 - i) *_{\mathbb{R}} a)$   
**by** *arith*  
**from**  $\langle ?l *_{\mathbb{R}} a + (1 - ?l) *_{\mathbb{R}} (j *_{\mathbb{R}} c + (1 - j) *_{\mathbb{R}} b) =$   
 $?k *_{\mathbb{R}} b + (1 - ?k) *_{\mathbb{R}} (i *_{\mathbb{R}} c + (1 - i) *_{\mathbb{R}} a) \rangle$  **and**  
 $\langle p - a = i *_{\mathbb{R}} (c - a) \rangle$  **and**  
 $\langle q - b = j *_{\mathbb{R}} (c - b) \rangle$  **and**  
*rearrange-real-euclid-B* [of  $p a i c$ ] **and**  
*rearrange-real-euclid-B* [of  $q b j c$ ]  
**have**  $?l *_{\mathbb{R}} a + (1 - ?l) *_{\mathbb{R}} q = ?k *_{\mathbb{R}} b + (1 - ?k) *_{\mathbb{R}} p$  **by** *simp*  
**let**  $?x = ?l *_{\mathbb{R}} a + (1 - ?l) *_{\mathbb{R}} q$   
**from** *rearrange-real-euclid-B* [of  $?x q ?l a$ ]  
**have**  $?x - q = ?l *_{\mathbb{R}} (a - q)$  **by** *simp*  
**from**  $\langle ?x = ?k *_{\mathbb{R}} b + (1 - ?k) *_{\mathbb{R}} p \rangle$  **and**  
*rearrange-real-euclid-B* [of  $?x p ?k b$ ]  
**have**  $?x - p = ?k *_{\mathbb{R}} (b - p)$  **by** *simp*  
**from**  $\langle i + j * (1 - i) > 0 \rangle$  **and**  
 $\langle j * (1 - i) \geq 0 \rangle$  **and**  
*zero-le-divide-iff* [of  $j * (1 - i) i + j * (1 - i)$ ]  
**have**  $?l \geq 0$  **by** *simp*  
**from**  $\langle i + j * (1 - i) > 0 \rangle$  **and**  
 $\langle i \geq 0 \rangle$  **and**  
*zero-le-divide-iff* [of  $i i + j * (1 - i)$ ] **and**  
 $\langle 1 - ?l = i / (i + j * (1 - i)) \rangle$   
**have**  $1 - ?l \geq 0$  **by** *simp*  
**hence**  $?l \leq 1$  **by** *simp*  
**with**  $\langle ?l \geq 0 \rangle$  **and**  
 $\langle ?x - q = ?l *_{\mathbb{R}} (a - q) \rangle$  **and**  
*real-euclid-B-def* [of  $q ?x a$ ]  
**have**  $B_{\mathbb{R}} q ?x a$  **by** *auto*  
**from**  $\langle j \leq 1 \rangle$  **have**  $1 - j \geq 0$  **by** *simp*  
**with**  $\langle 1 - ?l \geq 0 \rangle$  **and**  
 $\langle (1 - ?l) * (1 - j) = ?k \rangle$  **and**  
*zero-le-mult-iff* [of  $1 - ?l 1 - j$ ]  
**have**  $?k \geq 0$  **by** *simp*  
**from**  $\langle j \geq 0 \rangle$  **have**  $1 - j \leq 1$  **by** *simp*  
**from**  $\langle ?l \geq 0 \rangle$  **have**  $1 - ?l \leq 1$  **by** *simp*  
**with**  $\langle 1 - j \leq 1 \rangle$  **and**  
 $\langle 1 - j \geq 0 \rangle$  **and**  
*mult-mono* [of  $1 - ?l 1 1 - j 1$ ] **and**  
 $\langle (1 - ?l) * (1 - j) = ?k \rangle$

**have**  $?k \leq 1$  **by** *simp*  
**with**  $\langle ?k \geq 0 \rangle$  **and**  
 $\langle ?x - p = ?k *_R (b - p) \rangle$  **and**  
*real-euclid-B-def* [of  $p$   $?x$   $b$ ]  
**have**  $B_R p ?x b$  **by** *auto*  
**with**  $\langle B_R q ?x a \rangle$  **show** *?thesis* **by** *auto*  
**qed** }  
**thus**  $\forall a b c p q. B_R a p c \wedge B_R b q c \longrightarrow (\exists x. B_R p x b \wedge B_R q x a)$  **by** *auto*  
**{ fix**  $X Y$   
**assume**  $\exists a. \forall x y. x \in X \wedge y \in Y \longrightarrow B_R a x y$   
**then obtain**  $a$  **where**  $\forall x y. x \in X \wedge y \in Y \longrightarrow B_R a x y$  **by** *auto*  
**have**  $\exists b. \forall x y. x \in X \wedge y \in Y \longrightarrow B_R x b y$   
**proof cases**  
**assume**  $X \subseteq \{a\} \vee Y = \{\}$   
**let**  $?b = a$   
**{ fix**  $x y$   
**assume**  $x \in X$  **and**  $y \in Y$   
**with**  $\langle X \subseteq \{a\} \vee Y = \{\} \rangle$  **have**  $x = a$  **by** *auto*  
**from**  $\langle \forall x y. x \in X \wedge y \in Y \longrightarrow B_R a x y \rangle$  **and**  $\langle x \in X \rangle$  **and**  $\langle y \in Y \rangle$   
**have**  $B_R a x y$  **by** *simp*  
**with**  $\langle x = a \rangle$  **have**  $B_R x ?b y$  **by** *simp* }  
**hence**  $\forall x y. x \in X \wedge y \in Y \longrightarrow B_R x ?b y$  **by** *simp*  
**thus** *?thesis* **by** *auto*  
**next**  
**assume**  $\neg(X \subseteq \{a\} \vee Y = \{\})$   
**hence**  $X - \{a\} \neq \{\}$  **and**  $Y \neq \{\}$  **by** *auto*  
**from**  $\langle X - \{a\} \neq \{\} \rangle$  **obtain**  $c$  **where**  $c \in X$  **and**  $c \neq a$  **by** *auto*  
**from**  $\langle c \neq a \rangle$  **have**  $c - a \neq 0$  **by** *simp*  
**{ fix**  $y$   
**assume**  $y \in Y$   
**with**  $\langle \forall x y. x \in X \wedge y \in Y \longrightarrow B_R a x y \rangle$  **and**  $\langle c \in X \rangle$   
**have**  $B_R a c y$  **by** *simp*  
**with** *real-euclid-B-def* [of  $a$   $c$   $y$ ]  
**obtain**  $l$  **where**  $l \geq 0$  **and**  $l \leq 1$  **and**  $c - a = l *_R (y - a)$  **by** *auto*  
**from**  $\langle c - a = l *_R (y - a) \rangle$  **and**  $\langle c - a \neq 0 \rangle$  **have**  $l \neq 0$  **by** *simp*  
**with**  $\langle l \geq 0 \rangle$  **have**  $l > 0$  **by** *simp*  
**with**  $\langle c - a = l *_R (y - a) \rangle$  **have**  $y - a = (1/l) *_R (c - a)$  **by** *simp*  
**from**  $\langle l > 0 \rangle$  **and**  $\langle l \leq 1 \rangle$  **have**  $1/l \geq 1$  **by** *simp*  
**with**  $\langle y - a = (1/l) *_R (c - a) \rangle$   
**have**  $\exists j \geq 1. y - a = j *_R (c - a)$  **by** *auto* }  
**note** *ylemma = this*  
**from**  $\langle Y \neq \{\} \rangle$  **obtain**  $d$  **where**  $d \in Y$  **by** *auto*  
**with** *ylemma* [of  $d$ ]  
**obtain**  $jd$  **where**  $jd \geq 1$  **and**  $d - a = jd *_R (c - a)$  **by** *auto*  
**{ fix**  $x$   
**assume**  $x \in X$   
**with**  $\langle \forall x y. x \in X \wedge y \in Y \longrightarrow B_R a x y \rangle$  **and**  $\langle d \in Y \rangle$   
**have**  $B_R a x d$  **by** *simp*  
**with** *real-euclid-B-def* [of  $a$   $x$   $d$ ]

**obtain**  $l$  **where**  $l \geq 0$  **and**  $x - a = l *_R (d - a)$  **by** *auto*  
**from**  $\langle x - a = l *_R (d - a) \rangle$  **and**  
 $\langle d - a = jd *_R (c - a) \rangle$  **and**  
*scaleR-scaleR*  
**have**  $x - a = (l * jd) *_R (c - a)$  **by** *simp*  
**hence**  $\exists i. x - a = i *_R (c - a)$  **by** *auto* }  
**note**  $xlemma = this$   
**let**  $?S = \{j. j \geq 1 \wedge (\exists y \in Y. y - a = j *_R (c - a))\}$   
**from**  $\langle d \in Y \rangle$  **and**  $\langle jd \geq 1 \rangle$  **and**  $\langle d - a = jd *_R (c - a) \rangle$   
**have**  $?S \neq \{\}$  **by** *auto*  
**let**  $?k = \text{Inf } ?S$   
**let**  $?b = ?k *_R c + (1 - ?k) *_R a$   
**from** *rearrange-real-euclid-B* [*of ?b a ?k c*]  
**have**  $?b - a = ?k *_R (c - a)$  **by** *simp*  
{ **fix**  $x y$   
**assume**  $x \in X$  **and**  $y \in Y$   
**from**  $xlemma$  [*of x*] **and**  $\langle x \in X \rangle$   
**obtain**  $i$  **where**  $x - a = i *_R (c - a)$  **by** *auto*  
**from**  $ylemma$  [*of y*] **and**  $\langle y \in Y \rangle$   
**obtain**  $j$  **where**  $j \geq 1$  **and**  $y - a = j *_R (c - a)$  **by** *auto*  
**with**  $\langle y \in Y \rangle$  **have**  $j \in ?S$  **by** *auto*  
**then** **have**  $?k \leq j$  **by** (*auto intro: cInf-lower*)  
{ **fix**  $h$   
**assume**  $h \in ?S$   
**hence**  $h \geq 1$  **by** *simp*  
**from**  $\langle h \in ?S \rangle$   
**obtain**  $z$  **where**  $z \in Y$  **and**  $z - a = h *_R (c - a)$  **by** *auto*  
**from**  $\langle \forall x y. x \in X \wedge y \in Y \longrightarrow B_{\mathbb{R}} a x y \rangle$  **and**  $\langle x \in X \rangle$  **and**  $\langle z \in Y \rangle$   
**have**  $B_{\mathbb{R}} a x z$  **by** *simp*  
**with** *real-euclid-B-def* [*of a x z*]  
**obtain**  $l$  **where**  $l \leq 1$  **and**  $x - a = l *_R (z - a)$  **by** *auto*  
**with**  $\langle z - a = h *_R (c - a) \rangle$  **and** *scaleR-scaleR*  
**have**  $x - a = (l * h) *_R (c - a)$  **by** *simp*  
**with**  $\langle x - a = i *_R (c - a) \rangle$   
**have**  $i *_R (c - a) = (l * h) *_R (c - a)$  **by** *auto*  
**with** *scaleR-cancel-right* **and**  $\langle c - a \neq 0 \rangle$  **have**  $i = l * h$  **by** *blast*  
**with**  $\langle l \leq 1 \rangle$  **and**  $\langle h \geq 1 \rangle$  **have**  $i \leq h$  **by** *simp* }  
**with**  $\langle ?S \neq \{\} \rangle$  **and** *cInf-greatest* [*of ?S*] **have**  $i \leq ?k$  **by** *simp*  
**have**  $y - x = (y - a) - (x - a)$  **by** *simp*  
**with**  $\langle y - a = j *_R (c - a) \rangle$  **and**  $\langle x - a = i *_R (c - a) \rangle$   
**have**  $y - x = j *_R (c - a) - i *_R (c - a)$  **by** *simp*  
**with** *scaleR-left-diff-distrib* [*of j i c - a*]  
**have**  $y - x = (j - i) *_R (c - a)$  **by** *simp*  
**have**  $?b - x = (?b - a) - (x - a)$  **by** *simp*  
**with**  $\langle ?b - a = ?k *_R (c - a) \rangle$  **and**  $\langle x - a = i *_R (c - a) \rangle$   
**have**  $?b - x = ?k *_R (c - a) - i *_R (c - a)$  **by** *simp*  
**with** *scaleR-left-diff-distrib* [*of ?k i c - a*]  
**have**  $?b - x = (?k - i) *_R (c - a)$  **by** *simp*  
**have**  $B_{\mathbb{R}} x ?b y$

**proof cases**  
**assume**  $i = j$   
**with**  $\langle i \leq ?k \rangle$  **and**  $\langle ?k \leq j \rangle$  **have**  $?k = i$  **by** *simp*  
**with**  $\langle ?b - x = (?k - i) *_R (c - a) \rangle$  **have**  $?b - x = 0$  **by** *simp*  
**hence**  $?b - x = 0 *_R (y - x)$  **by** *simp*  
**with** *real-euclid-B-def* [of  $x ?b y$ ] **show**  $B_{\mathbf{R}} x ?b y$  **by** *auto*  
**next**  
**assume**  $i \neq j$   
**with**  $\langle i \leq ?k \rangle$  **and**  $\langle ?k \leq j \rangle$  **have**  $j - i > 0$  **by** *simp*  
**with**  $\langle y - x = (j - i) *_R (c - a) \rangle$  **and** *scaleR-scaleR*  
**have**  $c - a = (1 / (j - i)) *_R (y - x)$  **by** *simp*  
**with**  $\langle ?b - x = (?k - i) *_R (c - a) \rangle$  **and** *scaleR-scaleR*  
**have**  $?b - x = ((?k - i) / (j - i)) *_R (y - x)$  **by** *simp*  
**let**  $?l = (?k - i) / (j - i)$   
**from**  $\langle ?k \leq j \rangle$  **have**  $?k - i \leq j - i$  **by** *simp*  
**with**  $\langle j - i > 0 \rangle$  **have**  $?l \leq 1$  **by** *simp*  
**from**  $\langle i \leq ?k \rangle$  **and**  $\langle j - i > 0 \rangle$  **and** *pos-le-divide-eq* [of  $j - i$   $0$   $?k - i$ ]  
**have**  $?l \geq 0$  **by** *simp*  
**with** *real-euclid-B-def* [of  $x ?b y$ ] **and**  
 $\langle ?l \leq 1 \rangle$  **and**  
 $\langle ?b - x = ?l *_R (y - x) \rangle$   
**show**  $B_{\mathbf{R}} x ?b y$  **by** *auto*  
**qed }**  
**thus**  $\exists b. \forall x y. x \in X \wedge y \in Y \longrightarrow B_{\mathbf{R}} x b y$  **by** *auto*  
**qed }**  
**thus**  $\forall X Y. (\exists a. \forall x y. x \in X \wedge y \in Y \longrightarrow B_{\mathbf{R}} a x y) \longrightarrow$   
 $(\exists b. \forall x y. x \in X \wedge y \in Y \longrightarrow B_{\mathbf{R}} x b y)$   
**by** *auto*  
**qed**

### 4.3 Real Euclidean space satisfies the Euclidean axiom

**lemma** *rearrange-real-euclid-B-2*:

**fixes**  $a b c :: \text{real}^{\wedge}('n::\text{finite})$

**assumes**  $l \neq 0$

**shows**  $b - a = l *_R (c - a) \longleftrightarrow c = (1/l) *_R b + (1 - 1/l) *_R a$

**proof**

**from** *scaleR-right-diff-distrib* [of  $1/l$   $b$   $a$ ]

**have**  $(1/l) *_R (b - a) = c - a \longleftrightarrow (1/l) *_R b - (1/l) *_R a + a = c$  **by** *auto*

**also with** *scaleR-left-diff-distrib* [of  $1$   $1/l$   $a$ ]

**have**  $\dots \longleftrightarrow c = (1/l) *_R b + (1 - 1/l) *_R a$  **by** *auto*

**finally have** *eq*:

$(1/l) *_R (b - a) = c - a \longleftrightarrow c = (1/l) *_R b + (1 - 1/l) *_R a .$

**{ assume**  $b - a = l *_R (c - a)$

**with**  $\langle l \neq 0 \rangle$  **have**  $(1/l) *_R (b - a) = c - a$  **by** *simp*

**with** *eq* **show**  $c = (1/l) *_R b + (1 - 1/l) *_R a ..$  **}**

**{ assume**  $c = (1/l) *_R b + (1 - 1/l) *_R a$

**with** *eq* **have**  $(1/l) *_R (b - a) = c - a ..$

**hence**  $l *_R (1/l) *_R (b - a) = l *_R (c - a)$  **by** *simp*

with  $\langle l \neq 0 \rangle$  show  $b - a = l *_R (c - a)$  by simp }  
qed

interpretation *real-euclid: tarski-space real-euclid-C real-euclid-B*  
proof

{ fix  $a b c d t$   
assume  $B_R a d t$  and  $B_R b d c$  and  $a \neq d$   
from *real-euclid-B-def* [of  $a d t$ ] and  $\langle B_R a d t \rangle$   
obtain  $j$  where  $j \geq 0$  and  $j \leq 1$  and  $d - a = j *_R (t - a)$  by *auto*  
from  $\langle d - a = j *_R (t - a) \rangle$  and  $\langle a \neq d \rangle$  have  $j \neq 0$  by *auto*  
with  $\langle d - a = j *_R (t - a) \rangle$  and *rearrange-real-euclid-B-2*  
have  $t = (1/j) *_R d + (1 - 1/j) *_R a$  by *auto*  
let  $?x = (1/j) *_R b + (1 - 1/j) *_R a$   
let  $?y = (1/j) *_R c + (1 - 1/j) *_R a$   
from  $\langle j \neq 0 \rangle$  and *rearrange-real-euclid-B-2* have  
 $b - a = j *_R (?x - a)$  and  $c - a = j *_R (?y - a)$  by *auto*  
with *real-euclid-B-def* and  $\langle j \geq 0 \rangle$  and  $\langle j \leq 1 \rangle$  have  
 $B_R a b ?x$  and  $B_R a c ?y$  by *auto*  
from *real-euclid-B-def* and  $\langle B_R b d c \rangle$  obtain  $k$  where  
 $k \geq 0$  and  $k \leq 1$  and  $d - b = k *_R (c - b)$  by *blast*  
from  $\langle t = (1/j) *_R d + (1 - 1/j) *_R a \rangle$  have  
 $t - ?x = (1/j) *_R d - (1/j) *_R b$  by *simp*  
also from *scaleR-right-diff-distrib* [of  $1/j d b$ ] have  
 $\dots = (1/j) *_R (d - b)$  by *simp*  
also from  $\langle d - b = k *_R (c - b) \rangle$  have  
 $\dots = k *_R (1/j) *_R (c - b)$  by *simp*  
also from *scaleR-right-diff-distrib* [of  $1/j c b$ ] have  
 $\dots = k *_R (?y - ?x)$  by *simp*  
finally have  $t - ?x = k *_R (?y - ?x)$  .  
with *real-euclid-B-def* and  $\langle k \geq 0 \rangle$  and  $\langle k \leq 1 \rangle$  have  $B_R ?x t ?y$  by *blast*  
with  $\langle B_R a b ?x \rangle$  and  $\langle B_R a c ?y \rangle$  have  
 $\exists x y. B_R a b x \wedge B_R a c y \wedge B_R x t y$  by *auto* }  
thus  $\forall a b c d t. B_R a d t \wedge B_R b d c \wedge a \neq d \longrightarrow$   
 $(\exists x y. B_R a b x \wedge B_R a c y \wedge B_R x t y)$   
by *auto*  
qed

#### 4.4 The real Euclidean plane

lemma *Col-dep2*:

*real-euclid.Col a b c*  $\longleftrightarrow$  *dep2 (b - a) (c - a)*

proof -

from *real-euclid.Col-def* have

*real-euclid.Col a b c*  $\longleftrightarrow$   $B_R a b c \vee B_R b c a \vee B_R c a b$  by *auto*

moreover from *dep2-def* have

*dep2 (b - a) (c - a)*  $\longleftrightarrow$   $(\exists w r s. b - a = r *_R w \wedge c - a = s *_R w)$

by *auto*

moreover

{ assume  $B_R a b c \vee B_R b c a \vee B_R c a b$

**moreover**  
 { **assume**  $B_{\mathbb{R}} a b c$   
   **with** *real-euclid-B-def* **obtain**  $l$  **where**  $b - a = l *_{\mathbb{R}} (c - a)$  **by** *blast*  
   **moreover** **have**  $c - a = 1 *_{\mathbb{R}} (c - a)$  **by** *simp*  
   **ultimately** **have**  $\exists w r s. b - a = r *_{\mathbb{R}} w \wedge c - a = s *_{\mathbb{R}} w$  **by** *blast* }  
**moreover**  
 { **assume**  $B_{\mathbb{R}} b c a$   
   **with** *real-euclid-B-def* **obtain**  $l$  **where**  $c - b = l *_{\mathbb{R}} (a - b)$  **by** *blast*  
   **moreover** **have**  $c - a = (c - b) - (a - b)$  **by** *simp*  
   **ultimately** **have**  $c - a = l *_{\mathbb{R}} (a - b) - (a - b)$  **by** *simp*  
   **with** *scaleR-left-diff-distrib* [of  $l$  1  $a - b$ ] **have**  
      $c - a = (l - 1) *_{\mathbb{R}} (a - b)$  **by** *simp*  
   **moreover** **from** *scaleR-minus-left* [of 1  $a - b$ ] **have**  
      $b - a = (-1) *_{\mathbb{R}} (a - b)$  **by** *simp*  
   **ultimately** **have**  $\exists w r s. b - a = r *_{\mathbb{R}} w \wedge c - a = s *_{\mathbb{R}} w$  **by** *blast* }  
**moreover**  
 { **assume**  $B_{\mathbb{R}} c a b$   
   **with** *real-euclid-B-def* **obtain**  $l$  **where**  $a - c = l *_{\mathbb{R}} (b - c)$  **by** *blast*  
   **moreover** **have**  $c - a = -(a - c)$  **by** *simp*  
   **ultimately** **have**  $c - a = -(l *_{\mathbb{R}} (b - c))$  **by** *simp*  
   **with** *scaleR-minus-left* **have**  $c - a = (-l) *_{\mathbb{R}} (b - c)$  **by** *simp*  
   **moreover** **have**  $b - a = (b - c) + (c - a)$  **by** *simp*  
   **ultimately** **have**  $b - a = 1 *_{\mathbb{R}} (b - c) + (-l) *_{\mathbb{R}} (b - c)$  **by** *simp*  
   **with** *scaleR-left-distrib* [of 1  $-l$   $b - c$ ] **have**  
      $b - a = (1 + (-l)) *_{\mathbb{R}} (b - c)$  **by** *simp*  
   **with**  $\langle c - a = (-l) *_{\mathbb{R}} (b - c) \rangle$  **have**  
      $\exists w r s. b - a = r *_{\mathbb{R}} w \wedge c - a = s *_{\mathbb{R}} w$  **by** *blast* }  
**ultimately** **have**  $\exists w r s. b - a = r *_{\mathbb{R}} w \wedge c - a = s *_{\mathbb{R}} w$  **by** *auto* }  
**moreover**  
 { **assume**  $\exists w r s. b - a = r *_{\mathbb{R}} w \wedge c - a = s *_{\mathbb{R}} w$   
   **then** **obtain**  $w r s$  **where**  $b - a = r *_{\mathbb{R}} w$  **and**  $c - a = s *_{\mathbb{R}} w$  **by** *auto*  
   **have**  $B_{\mathbb{R}} a b c \vee B_{\mathbb{R}} b c a \vee B_{\mathbb{R}} c a b$   
   **proof** *cases*  
     **assume**  $s = 0$   
       **with**  $\langle c - a = s *_{\mathbb{R}} w \rangle$  **have**  $a = c$  **by** *simp*  
       **with** *real-euclid.th3-1* **have**  $B_{\mathbb{R}} b c a$  **by** *simp*  
       **thus** *?thesis* **by** *simp*  
   **next**  
     **assume**  $s \neq 0$   
       **with**  $\langle c - a = s *_{\mathbb{R}} w \rangle$  **have**  $w = (1/s) *_{\mathbb{R}} (c - a)$  **by** *simp*  
       **with**  $\langle b - a = r *_{\mathbb{R}} w \rangle$  **have**  $b - a = (r/s) *_{\mathbb{R}} (c - a)$  **by** *simp*  
       **have**  $r/s < 0 \vee (r/s \geq 0 \wedge r/s \leq 1) \vee r/s > 1$  **by** *arith*  
       **moreover**  
         { **assume**  $r/s \geq 0 \wedge r/s \leq 1$   
           **with** *real-euclid-B-def* **and**  $\langle b - a = (r/s) *_{\mathbb{R}} (c - a) \rangle$  **have**  $B_{\mathbb{R}} a b c$   
           **by** *auto*  
           **hence** *?thesis* **by** *simp* }  
       **moreover**  
         { **assume**  $r/s > 1$

**with**  $\langle b - a = (r/s) *_{\mathbb{R}} (c - a) \rangle$  **have**  $c - a = (s/r) *_{\mathbb{R}} (b - a)$  **by** *auto*  
**from**  $\langle r/s > 1 \rangle$  **and** *le-imp-inverse-le* [of 1  $r/s$ ] **have**  
 $s/r \leq 1$  **by** *simp*  
**from**  $\langle r/s > 1 \rangle$  **and** *inverse-positive-iff-positive* [of  $r/s$ ] **have**  
 $s/r \geq 0$  **by** *simp*  
**with** *real-euclid-B-def*  
**and**  $\langle c - a = (s/r) *_{\mathbb{R}} (b - a) \rangle$   
**and**  $\langle s/r \leq 1 \rangle$   
**have**  $B_{\mathbb{R}} a c b$  **by** *auto*  
**with** *real-euclid.th3-2* **have**  $B_{\mathbb{R}} b c a$  **by** *auto*  
**hence** *?thesis* **by** *simp* }  
**moreover**  
{ **assume**  $r/s < 0$   
**have**  $b - c = (b - a) + (a - c)$  **by** *simp*  
**with**  $\langle b - a = (r/s) *_{\mathbb{R}} (c - a) \rangle$  **have**  
 $b - c = (r/s) *_{\mathbb{R}} (c - a) + (a - c)$  **by** *simp*  
**have**  $c - a = -(a - c)$  **by** *simp*  
**with** *scaleR-minus-right* [of  $r/s$   $a - c$ ] **have**  
 $(r/s) *_{\mathbb{R}} (c - a) = -((r/s) *_{\mathbb{R}} (a - c))$  **by** *arith*  
**with**  $\langle b - c = (r/s) *_{\mathbb{R}} (c - a) + (a - c) \rangle$  **have**  
 $b - c = -(r/s) *_{\mathbb{R}} (a - c) + (a - c)$  **by** *simp*  
**with** *scaleR-left-distrib* [of  $-(r/s)$  1  $a - c$ ] **have**  
 $b - c = (-(r/s) + 1) *_{\mathbb{R}} (a - c)$  **by** *simp*  
**moreover from**  $\langle r/s < 0 \rangle$  **have**  $-(r/s) + 1 > 1$  **by** *simp*  
**ultimately have**  $a - c = (1 / (-(r/s) + 1)) *_{\mathbb{R}} (b - c)$  **by** *auto*  
**let**  $?l = 1 / (-(r/s) + 1)$   
**from**  $\langle -(r/s) + 1 > 1 \rangle$  **and** *le-imp-inverse-le* [of 1  $-(r/s) + 1$ ] **have**  
 $?l \leq 1$  **by** *simp*  
**from**  $\langle -(r/s) + 1 > 1 \rangle$   
**and** *inverse-positive-iff-positive* [of  $-(r/s) + 1$ ]  
**have**  
 $?l \geq 0$  **by** *simp*  
**with** *real-euclid-B-def* **and**  $\langle ?l \leq 1 \rangle$  **and**  $\langle a - c = ?l *_{\mathbb{R}} (b - c) \rangle$  **have**  
 $B_{\mathbb{R}} c a b$  **by** *blast*  
**hence** *?thesis* **by** *simp* }  
**ultimately show** *?thesis* **by** *auto*  
**qed** }  
**ultimately show** *?thesis* **by** *blast*  
**qed**

**lemma** *non-Col-example*:

$\neg(\text{real-euclid.Col } 0 \text{ (vector } [1/2, 0] \text{ :: real}^2 \text{) (vector } [0, 1/2]))$   
(is  $\neg(\text{real-euclid.Col } ?a \ ?b \ ?c)$ )

**proof** –

{ **assume**  $\text{dep2 } (?b - ?a) (?c - ?a)$   
**with** *dep2-def* [of  $?b - ?a$   $?c - ?a$ ] **obtain**  $w r s$  **where**  
 $?b - ?a = r *_{\mathbb{R}} w$  **and**  $?c - ?a = s *_{\mathbb{R}} w$  **by** *auto*  
**have**  $?b = 1/2$  **by** *simp*  
**with**  $\langle ?b - ?a = r *_{\mathbb{R}} w \rangle$  **have**  $r * (w = 1/2)$  **by** *simp*

hence  $w\$1 \neq 0$  by *auto*  
 have  $?c\$1 = 0$  by *simp*  
 with  $\langle ?c - ?a = s *_R w \rangle$  have  $s * (w\$1) = 0$  by *simp*  
 with  $\langle w\$1 \neq 0 \rangle$  have  $s = 0$  by *simp*  
 have  $?c\$2 = 1/2$  by *simp*  
 with  $\langle ?c - ?a = s *_R w \rangle$  have  $s * (w\$2) = 1/2$  by *simp*  
 with  $\langle s = 0 \rangle$  have *False* by *simp* }  
 hence  $\neg(\text{dep2 } (?b - ?a) (?c - ?a))$  by *auto*  
 with *Col-dep2* show  $\neg(\text{real-euclid.Col } ?a ?b ?c)$  by *blast*  
 qed

**interpretation** *real-euclid*:

*tarski real-euclid-C::([real^2, real^2, real^2, real^2]  $\Rightarrow$  bool) real-euclid-B*

**proof**

{ let  $?a = 0 :: \text{real}^2$   
 let  $?b = \text{vector } [1/2, 0] :: \text{real}^2$   
 let  $?c = \text{vector } [0, 1/2] :: \text{real}^2$   
 from *non-Col-example* and *real-euclid.Col-def* have  
 $\neg B_{\mathbb{R}} ?a ?b ?c \wedge \neg B_{\mathbb{R}} ?b ?c ?a \wedge \neg B_{\mathbb{R}} ?c ?a ?b$  by *auto* }  
 thus  $\exists a b c :: \text{real}^2. \neg B_{\mathbb{R}} a b c \wedge \neg B_{\mathbb{R}} b c a \wedge \neg B_{\mathbb{R}} c a b$   
 by *auto*  
 { fix  $p q a b c :: \text{real}^2$   
 assume  $p \neq q$  and  $a p \equiv_{\mathbb{R}} a q$  and  $b p \equiv_{\mathbb{R}} b q$  and  $c p \equiv_{\mathbb{R}} c q$   
 let  $?m = (1/2) *_R (p + q)$   
 from *scaleR-right-distrib* [of  $1/2 p q$ ] and  
*scaleR-right-diff-distrib* [of  $1/2 q p$ ] and  
*scaleR-left-diff-distrib* [of  $1/2 1 p$ ]  
 have  $?m - p = (1/2) *_R (q - p)$  by *simp*  
 with  $\langle p \neq q \rangle$  have  $?m - p \neq 0$  by *simp*  
 from *scaleR-right-distrib* [of  $1/2 p q$ ] and  
*scaleR-right-diff-distrib* [of  $1/2 p q$ ] and  
*scaleR-left-diff-distrib* [of  $1/2 1 q$ ]  
 have  $?m - q = (1/2) *_R (p - q)$  by *simp*  
 with  $\langle ?m - p = (1/2) *_R (q - p) \rangle$   
 and *scaleR-minus-right* [of  $1/2 q - p$ ]  
 have  $?m - q = -(?m - p)$  by *simp*  
 with *norm-minus-cancel* [of  $?m - p$ ] have  
 $(\text{norm } (?m - q))^2 = (\text{norm } (?m - p))^2$  by (*simp only: norm-minus-cancel*)  
 { fix  $d$   
 assume  $d p \equiv_{\mathbb{R}} d q$   
 hence  $(\text{norm } (d - p))^2 = (\text{norm } (d - q))^2$  by *simp*  
 have  $(d - ?m) \cdot (?m - p) = 0$   
**proof** -  
 have  $d + (-q) = d - q$  by *simp*  
 have  $d + (-p) = d - p$  by *simp*  
 with *dot-norm* [of  $d - ?m ?m - p$ ] have  
 $(d - ?m) \cdot (?m - p) =$   
 $((\text{norm } (d - p))^2 - (\text{norm } (d - ?m))^2 - (\text{norm } (?m - p))^2) / 2$   
 by *simp*

**also from**  $\langle (\text{norm } (d - p))^2 = (\text{norm } (d - q))^2 \rangle$   
**and**  $\langle (\text{norm } (?m - q))^2 = (\text{norm } (?m - p))^2 \rangle$   
**have**  
 $\dots = ((\text{norm } (d - q))^2 - (\text{norm } (d - ?m))^2 - (\text{norm } (?m - q))^2) / 2$   
**by simp**  
**also from** *dot-norm* [of  $d - ?m$   $?m - q$ ]  
**and**  $\langle d + (-q) = d - q \rangle$   
**have**  
 $\dots = (d - ?m) \cdot (?m - q)$  **by simp**  
**also from** *inner-minus-right* [of  $d - ?m$   $?m - p$ ]  
**and**  $\langle ?m - q = -(?m - p) \rangle$   
**have**  
 $\dots = -((d - ?m) \cdot (?m - p))$  **by (simp only: inner-minus-left)**  
**finally have**  $(d - ?m) \cdot (?m - p) = -((d - ?m) \cdot (?m - p))$  .  
**thus**  $(d - ?m) \cdot (?m - p) = 0$  **by arith**  
**qed }**  
**note** *m-lemma = this*  
**with**  $\langle a \equiv_{\mathbb{R}} a \rangle$  **have**  $(a - ?m) \cdot (?m - p) = 0$  **by simp**  
**{ fix**  $d$   
**assume**  $d \equiv_{\mathbb{R}} d$   
**with** *m-lemma* **have**  $(d - ?m) \cdot (?m - p) = 0$  **by simp**  
**with** *dot-left-diff-distrib* [of  $d - ?m$   $a - ?m$   $?m - p$ ]  
**and**  $\langle a - ?m \rangle \cdot (?m - p) = 0$   
**have**  $(d - a) \cdot (?m - p) = 0$  **by (simp add: inner-diff-left inner-diff-right) }**  
**with**  $\langle b \equiv_{\mathbb{R}} b \rangle$  **and**  $\langle c \equiv_{\mathbb{R}} c \rangle$  **have**  
 $(b - a) \cdot (?m - p) = 0$  **and**  $(c - a) \cdot (?m - p) = 0$  **by simp+**  
**with** *real2-orthogonal-dep2* **and**  $\langle ?m - p \neq 0 \rangle$  **have** *dep2*  $(b - a) (c - a)$   
**by blast**  
**with** *Col-dep2* **have** *real-euclid.Col*  $a \ b \ c$  **by auto**  
**with** *real-euclid.Col-def* **have**  $B_{\mathbb{R}} \ a \ b \ c \vee B_{\mathbb{R}} \ b \ c \ a \vee B_{\mathbb{R}} \ c \ a \ b$  **by auto }**  
**thus**  $\forall p \ q \ a \ b \ c :: \text{real}^2$ .  
 $p \neq q \wedge a \equiv_{\mathbb{R}} a \ q \wedge b \equiv_{\mathbb{R}} b \ q \wedge c \equiv_{\mathbb{R}} c \ q \longrightarrow$   
 $B_{\mathbb{R}} \ a \ b \ c \vee B_{\mathbb{R}} \ b \ c \ a \vee B_{\mathbb{R}} \ c \ a \ b$   
**by blast**  
**qed**

## 4.5 Special cases of theorems of Tarski's geometry

**lemma** *real-euclid-B-disjunction*:

**assumes**  $l \geq 0$  **and**  $b - a = l *_{\mathbb{R}} (c - a)$

**shows**  $B_{\mathbb{R}} \ a \ b \ c \vee B_{\mathbb{R}} \ a \ c \ b$

**proof cases**

**assume**  $l \leq 1$

**with**  $\langle l \geq 0 \rangle$  **and**  $\langle b - a = l *_{\mathbb{R}} (c - a) \rangle$

**have**  $B_{\mathbb{R}} \ a \ b \ c$  **by (unfold real-euclid-B-def) (simp add: exI [of - l])**

**thus**  $B_{\mathbb{R}} \ a \ b \ c \vee B_{\mathbb{R}} \ a \ c \ b$  ..

**next**

**assume**  $\neg (l \leq 1)$

**hence**  $1/l \leq 1$  **by simp**

**from**  $\langle l \geq 0 \rangle$  **have**  $1/l \geq 0$  **by** *simp*

**from**  $\langle b - a = l *_R (c - a) \rangle$

**have**  $(1/l) *_R (b - a) = (1/l) *_R (l *_R (c - a))$  **by** *simp*

**with**  $\langle \neg (l \leq 1) \rangle$  **have**  $c - a = (1/l) *_R (b - a)$  **by** *simp*

**with**  $\langle 1/l \geq 0 \rangle$  **and**  $\langle 1/l \leq 1 \rangle$

**have**  $B_R a c b$  **by** (*unfold real-euclid-B-def*) (*simp add: exI [of - 1/l]*)

**thus**  $B_R a b c \vee B_R a c b$  ..

**qed**

The following are true in Tarski's geometry, but to prove this would require much more development of it, so only the Euclidean case is proven here.

**theorem** *real-euclid-th5-1:*

**assumes**  $a \neq b$  **and**  $B_R a b c$  **and**  $B_R a b d$

**shows**  $B_R a c d \vee B_R a d c$

**proof** –

**from**  $\langle B_R a b c \rangle$  **and**  $\langle B_R a b d \rangle$

**obtain**  $l$  **and**  $m$  **where**  $l \geq 0$  **and**  $b - a = l *_R (c - a)$

**and**  $m \geq 0$  **and**  $b - a = m *_R (d - a)$

**by** (*unfold real-euclid-B-def*) *auto*

**from**  $\langle b - a = m *_R (d - a) \rangle$  **and**  $\langle a \neq b \rangle$  **have**  $m \neq 0$  **by** *auto*

**from**  $\langle l \geq 0 \rangle$  **and**  $\langle m \geq 0 \rangle$  **have**  $l/m \geq 0$  **by** (*simp add: zero-le-divide-iff*)

**from**  $\langle b - a = l *_R (c - a) \rangle$  **and**  $\langle b - a = m *_R (d - a) \rangle$

**have**  $m *_R (d - a) = l *_R (c - a)$  **by** *simp*

**hence**  $(1/m) *_R (m *_R (d - a)) = (1/m) *_R (l *_R (c - a))$  **by** *simp*

**with**  $\langle m \neq 0 \rangle$  **have**  $d - a = (l/m) *_R (c - a)$  **by** *simp*

**with**  $\langle l/m \geq 0 \rangle$  **and** *real-euclid-B-disjunction*

**show**  $B_R a c d \vee B_R a d c$  **by** *auto*

**qed**

**theorem** *real-euclid-th5-3:*

**assumes**  $B_R a b d$  **and**  $B_R a c d$

**shows**  $B_R a b c \vee B_R a c b$

**proof** –

**from**  $\langle B_R a b d \rangle$  **and**  $\langle B_R a c d \rangle$

**obtain**  $l$  **and**  $m$  **where**  $l \geq 0$  **and**  $b - a = l *_R (d - a)$

**and**  $m \geq 0$  **and**  $c - a = m *_R (d - a)$

**by** (*unfold real-euclid-B-def*) *auto*

**show**  $B_R a b c \vee B_R a c b$

**proof** *cases*

**assume**  $l = 0$

**with**  $\langle b - a = l *_R (d - a) \rangle$  **have**  $b - a = l *_R (c - a)$  **by** *simp*

**with**  $\langle l = 0 \rangle$

**have**  $B_R a b c$  **by** (*unfold real-euclid-B-def*) (*simp add: exI [of - l]*)

```

    thus  $B_R a b c \vee B_R a c b ..$ 
  next
    assume  $l \neq 0$ 

    from  $\langle l \geq 0 \rangle$  and  $\langle m \geq 0 \rangle$  have  $m/l \geq 0$  by (simp add: zero-le-divide-iff)

    from  $\langle b - a = l *_R (d - a) \rangle$ 
    have  $(1/l) *_R (b - a) = (1/l) *_R (l *_R (d - a))$  by simp
    with  $\langle l \neq 0 \rangle$  have  $d - a = (1/l) *_R (b - a)$  by simp
    with  $\langle c - a = m *_R (d - a) \rangle$  have  $c - a = (m/l) *_R (b - a)$  by simp
    with  $\langle m/l \geq 0 \rangle$  and real-euclid-B-disjunction
    show  $B_R a b c \vee B_R a c b$  by auto
  qed
qed
end

```

## 5 Linear algebra

```

theory Linear-Algebra2
imports Miscellany
begin

```

```

lemma exhaust-4:
  fixes  $x :: 4$ 
  shows  $x = 1 \vee x = 2 \vee x = 3 \vee x = 4$ 
proof (induct x)
  case (of-int z)
  hence  $0 \leq z$  and  $z < 4$  by simp-all
  hence  $z = 0 \vee z = 1 \vee z = 2 \vee z = 3$  by arith
  thus ?case by auto
qed

```

```

lemma forall-4:  $(\forall i::4. P i) \longleftrightarrow P 1 \wedge P 2 \wedge P 3 \wedge P 4$ 
  by (metis exhaust-4)

```

```

lemma UNIV-4:  $(UNIV::(4 set)) = \{1, 2, 3, 4\}$ 
  using exhaust-4
  by auto

```

```

lemma vector-4:
  fixes  $w :: 'a::zero$ 
  shows (vector  $[w, x, y, z] :: 'a^4$ )$1 = w
  and (vector  $[w, x, y, z] :: 'a^4$ )$2 = x
  and (vector  $[w, x, y, z] :: 'a^4$ )$3 = y
  and (vector  $[w, x, y, z] :: 'a^4$ )$4 = z
  unfolding vector-def
  by simp-all

```

**definition**

*is-basis* :: (real<sup>'n</sup>) set ⇒ bool **where**  
*is-basis* S ≜ independent S ∧ span S = UNIV

**lemma** *card-finite*:

**assumes** card S = CARD('n::finite)  
**shows** finite S

**proof** –

**from** ⟨card S = CARD('n)⟩ **have** card S ≠ 0 **by** simp  
**with** card-eq-0-iff [of S] **show** finite S **by** simp

**qed**

**lemma** *independent-is-basis*:

**fixes** B :: (real<sup>'n</sup>) set  
**shows** independent B ∧ card B = CARD('n) ⟷ *is-basis* B

**proof**

**assume** L: independent B ∧ card B = CARD('n)  
**then have** card (Basis::(real<sup>'n</sup>) set) = card B  
**by** simp

**with** L **show** *is-basis* B

**by** (metis (no-types) card-eq-dim dim-UNIV independent-bound *is-basis-def* sub-set-antisym top-greatest)

**next**

**assume** *is-basis* B

**then show** independent B ∧ card B = CARD('n)

**by** (metis DIM-cart DIM-real basis-card-eq-dim dim-UNIV *is-basis-def* mult.right-neutral top.extremum)

**qed**

**lemma** *basis-finite*:

**fixes** B :: (real<sup>'n</sup>) set  
**assumes** *is-basis* B  
**shows** finite B

**proof** –

**from** *independent-is-basis* [of B] **and** ⟨*is-basis* B⟩ **have** card B = CARD('n)  
**by** simp

**with** *card-finite* [of B, where 'n = 'n] **show** finite B **by** simp

**qed**

**lemma** *basis-expand*:

**assumes** *is-basis* B  
**shows** ∃ c. v = (∑ w∈B. (c w) \*<sub>R</sub> w)

**proof** –

**from** ⟨*is-basis* B⟩ **have** v ∈ span B **unfolding** *is-basis-def* **by** simp

**from** *basis-finite* [of B] **and** ⟨*is-basis* B⟩ **have** finite B **by** simp

**with** span-finite [of B] **and** ⟨v ∈ span B⟩

**show** ∃ c. v = (∑ w∈B. (c w) \*<sub>R</sub> w) **by** (simp add: scalar-equiv) auto

**qed**

**lemma** *not-span-independent-insert*:

**fixes**  $v :: ('a::\text{real-vector})^{\wedge}n$   
**assumes** *independent S* **and**  $v \notin \text{span } S$   
**shows** *independent (insert v S)*  
**by** (*simp add: assms independent-insert*)

**lemma** *orthogonal-sum*:

**fixes**  $v :: \text{real}^{\wedge}n$   
**assumes**  $\bigwedge w. w \in S \implies \text{orthogonal } v \ w$   
**shows** *orthogonal v* ( $\sum w \in S. c \ w * s \ w$ )  
**by** (*metis (no-types, lifting) assms orthogonal-clauses(1,2) orthogonal-rvsum scalar-equiv sum.infinite*)

**lemma** *orthogonal-self-eq-0*:

**fixes**  $v :: ('a::\text{real-inner})^{\wedge}n$   
**assumes** *orthogonal v v*  
**shows**  $v = 0$   
**using** *inner-eq-zero-iff [of v]* **and** *assms*  
**unfolding** *orthogonal-def*  
**by** *simp*

**lemma** *orthogonal-in-span-eq-0*:

**fixes**  $v :: \text{real}^{\wedge}n$   
**assumes**  $v \in \text{span } S$  **and**  $\bigwedge w. w \in S \implies \text{orthogonal } v \ w$   
**shows**  $v = 0$   
**using** *assms orthogonal-self orthogonal-to-span* **by** *blast*

**lemma** *orthogonal-independent*:

**fixes**  $v :: \text{real}^{\wedge}n$   
**assumes** *independent S* **and**  $v \neq 0$  **and**  $\bigwedge w. w \in S \implies \text{orthogonal } v \ w$   
**shows** *independent (insert v S)*  
**using** *assms not-span-independent-insert orthogonal-in-span-eq-0* **by** *blast*

**lemma** *dot-scaleR-mult*:

**shows**  $(k *_{\mathbb{R}} a) \cdot b = k * (a \cdot b)$  **and**  $a \cdot (k *_{\mathbb{R}} b) = k * (a \cdot b)$   
**by** *auto*

**lemma** *dependent-explicit-finite*:

**fixes**  $S :: (('a::\{\text{real-vector}, \text{field}\})^{\wedge}n) \ \text{set}$   
**assumes** *finite S*  
**shows** *dependent S*  $\longleftrightarrow (\exists u. (\exists v \in S. u \ v \neq 0) \wedge (\sum v \in S. u \ v *_{\mathbb{R}} v) = 0)$   
**by** (*simp add: assms dependent-finite*)

**lemma** *dependent-explicit-2*:

**fixes**  $v \ w :: ('a::\{\text{field}, \text{real-vector}\})^{\wedge}n$   
**assumes**  $v \neq w$   
**shows** *dependent {v, w}*  $\longleftrightarrow (\exists i \ j. (i \neq 0 \vee j \neq 0) \wedge i *_{\mathbb{R}} v + j *_{\mathbb{R}} w = 0)$

**proof**

**let**  $?S = \{v, w\}$

```

have finite ?S by simp

{ assume dependent ?S
  with dependent-explicit-finite [of ?S] and ⟨finite ?S⟩ and ⟨v ≠ w⟩
  show ∃ i j. (i ≠ 0 ∨ j ≠ 0) ∧ i *R v + j *R w = 0 by auto }

{ assume ∃ i j. (i ≠ 0 ∨ j ≠ 0) ∧ i *R v + j *R w = 0
  then obtain i and j where i ≠ 0 ∨ j ≠ 0 and i *R v + j *R w = 0 by auto
  let ?u = λ x. if x = v then i else j
  from ⟨i ≠ 0 ∨ j ≠ 0⟩ and ⟨v ≠ w⟩ have ∃ x ∈ ?S. ?u x ≠ 0 by simp
  from ⟨i *R v + j *R w = 0⟩ and ⟨v ≠ w⟩
  have (∑ x ∈ ?S. ?u x *R x) = 0 by simp
  with dependent-explicit-finite [of ?S]
  and ⟨finite ?S⟩ and ⟨∃ x ∈ ?S. ?u x ≠ 0⟩
  show dependent ?S by best }
qed

```

## 5.1 Matrices

```

lemma zero-not-invertible:
  ¬ (invertible (0 :: realn))
  using invertible-times-eq-zero matrix-vector-mult-0 by blast

```

Based on matrix-vector-column in HOL/Multivariate\_Analysis/Euclidean\_Space.thy in Isabelle 2009-1:

```

lemma vector-matrix-row:
  fixes x :: ('a::comm-semiring-1)m and A :: ('an)m
  shows x v * A = (∑ i ∈ UNIV. (x $ i) * s (A $ i))
  unfolding vector-matrix-mult-def
  by (simp add: vec-eq-iff mult commute)

```

```

lemma matrix-inv:
  assumes invertible M
  shows matrix-inv M ** M = mat 1
  and M ** matrix-inv M = mat 1
  using ⟨invertible M⟩ and someI-ex [of λ N. M ** N = mat 1 ∧ N ** M = mat
1]
  unfolding invertible-def and matrix-inv-def
  by simp-all

```

```

lemma matrix-inv-invertible:
  assumes invertible M
  shows invertible (matrix-inv M)
  using ⟨invertible M⟩ and matrix-inv
  unfolding invertible-def [of matrix-inv M]
  by auto

```

```

lemma invertible-times-non-zero:
  fixes M :: realn

```

**assumes** *invertible M* **and**  $v \neq 0$   
**shows**  $M * v \neq 0$   
**using**  $\langle \text{invertible } M \rangle$  **and**  $\langle v \neq 0 \rangle$  **and** *invertible-times-eq-zero* [of  $M v$ ]  
**by** *auto*

**lemma** *matrix-right-invertible-ker*:  
**fixes**  $M :: \text{real}^{('m::\text{finite})}{}^n$   
**shows**  $(\exists M'. M ** M' = \text{mat } 1) \longleftrightarrow (\forall x. x v * M = 0 \longrightarrow x = 0)$   
**using** *left-invertible-transpose matrix-left-invertible-ker* **by force**

**lemma** *left-invertible-iff-invertible*:  
**fixes**  $M :: \text{real}^{n}{}^n$   
**shows**  $(\exists N. N ** M = \text{mat } 1) \longleftrightarrow \text{invertible } M$   
**by** (*simp add: invertible-def matrix-left-right-inverse*)

**lemma** *right-invertible-iff-invertible*:  
**fixes**  $M :: \text{real}^{n}{}^n$   
**shows**  $(\exists N. M ** N = \text{mat } 1) \longleftrightarrow \text{invertible } M$   
**by** (*simp add: invertible-def matrix-left-right-inverse*)

**definition** *symmatrix* ::  $'a^{n}{}^n \Rightarrow \text{bool}$  **where**  
*symmatrix*  $M \triangleq \text{transpose } M = M$

**lemma** *symmatrix-preserve*:  
**fixes**  $M N :: ('a::\text{comm-semiring-1})^{n}{}^n$   
**assumes** *symmatrix*  $M$   
**shows** *symmatrix*  $(N ** M ** \text{transpose } N)$   
**proof** –  
**have**  $\text{transpose } (N ** M ** \text{transpose } N) = N ** (M ** \text{transpose } N)$   
**by** (*metis (no-types) transpose-transpose assms matrix-transpose-mul symmatrix-def*)  
**then show** *?thesis*  
**by** (*simp add: matrix-mul-assoc symmatrix-def*)  
**qed**

**lemma** *non-zero-mult-invertible-non-zero*:  
**fixes**  $M :: \text{real}^{n}{}^n$   
**assumes**  $v \neq 0$  **and** *invertible*  $M$   
**shows**  $v v * M \neq 0$   
**using**  $\langle v \neq 0 \rangle$  **and**  $\langle \text{invertible } M \rangle$  **and** *times-invertible-eq-zero*  
**by** *auto*

**end**

## 6 Right group actions

**theory** *Action*  
**imports** *HOL-Algebra.Group*  
**begin**

**locale** *action* = *group* +  
**fixes** *act* :: 'b  $\Rightarrow$  'a  $\Rightarrow$  'b (**infixl**  $\langle < o \rangle$  69)  
**assumes** *id-act* [*simp*]:  $b < o \mathbf{1} = b$   
**and** *act-act'*:  
 $g \in \text{carrier } G \wedge h \in \text{carrier } G \longrightarrow (b < o g) < o h = b < o (g \otimes h)$   
**begin**

**lemma** *act-act*:  
**assumes**  $g \in \text{carrier } G$  **and**  $h \in \text{carrier } G$   
**shows**  $(b < o g) < o h = b < o (g \otimes h)$   
**proof** –  
**from**  $\langle g \in \text{carrier } G \rangle$  **and**  $\langle h \in \text{carrier } G \rangle$  **and** *act-act'*  
**show**  $(b < o g) < o h = b < o (g \otimes h)$  **by** *simp*  
**qed**

**lemma** *act-act-inv* [*simp*]:  
**assumes**  $g \in \text{carrier } G$   
**shows**  $b < o g < o \text{inv } g = b$   
**proof** –  
**from**  $\langle g \in \text{carrier } G \rangle$  **have**  $\text{inv } g \in \text{carrier } G$  **by** (*rule inv-closed*)  
**with**  $\langle g \in \text{carrier } G \rangle$  **have**  $b < o g < o \text{inv } g = b < o g \otimes \text{inv } g$  **by** (*rule act-act*)  
**with**  $\langle g \in \text{carrier } G \rangle$  **show**  $b < o g < o \text{inv } g = b$  **by** *simp*  
**qed**

**lemma** *act-inv-act* [*simp*]:  
**assumes**  $g \in \text{carrier } G$   
**shows**  $b < o \text{inv } g < o g = b$   
**using**  $\langle g \in \text{carrier } G \rangle$  **and** *act-act-inv* [*of inv g*]  
**by** *simp*

**lemma** *act-inv-iff*:  
**assumes**  $g \in \text{carrier } G$   
**shows**  $b < o \text{inv } g = c \longleftrightarrow b = c < o g$   
**proof**  
**assume**  $b < o \text{inv } g = c$   
**hence**  $b < o \text{inv } g < o g = c < o g$  **by** *simp*  
**with**  $\langle g \in \text{carrier } G \rangle$  **show**  $b = c < o g$  **by** *simp*  
**next**  
**assume**  $b = c < o g$   
**hence**  $b < o \text{inv } g = c < o g < o \text{inv } g$  **by** *simp*  
**with**  $\langle g \in \text{carrier } G \rangle$  **show**  $b < o \text{inv } g = c$  **by** *simp*  
**qed**

**end**

**end**

## 7 Projective geometry

```
theory Projective
  imports Linear-Algebra2
         Euclid-Tarski
         Action
begin
```

### 7.1 Proportionality on non-zero vectors

```
context vector-space
begin
```

```
definition proportionality :: ('b × 'b) set where
  proportionality  $\triangleq$   $\{(x, y). x \neq 0 \wedge y \neq 0 \wedge (\exists k. x = \text{scale } k \ y)\}$ 
```

```
definition non-zero-vectors :: 'b set where
  non-zero-vectors  $\triangleq$   $\{x. x \neq 0\}$ 
```

```
lemma proportionality-subset:
  local.proportionality  $\subseteq$  local.non-zero-vectors  $\times$  local.non-zero-vectors
  unfolding proportionality-def non-zero-vectors-def
  by auto
```

```
lemma proportionality-refl-on: refl-on local.non-zero-vectors local.proportionality
proof -
```

```
  have  $\forall x \in \text{local.non-zero-vectors}. (x, x) \in \text{local.proportionality}$ 
```

```
proof
```

```
  fix x
```

```
  assume  $x \in \text{local.non-zero-vectors}$ 
```

```
  hence  $x \neq 0$  unfolding non-zero-vectors-def ..
```

```
  moreover have  $x = \text{scale } 1 \ x$  by simp
```

```
  ultimately show  $(x, x) \in \text{local.proportionality}$ 
```

```
    unfolding proportionality-def
```

```
    by blast
```

```
qed
```

```
thus refl-on local.non-zero-vectors local.proportionality
```

```
  by (simp add: refl-onI)
```

```
qed
```

```
lemma proportionality-sym: sym local.proportionality
```

```
proof -
```

```
{ fix x y
```

```
  assume  $(x, y) \in \text{local.proportionality}$ 
```

```
  hence  $x \neq 0$  and  $y \neq 0$  and  $\exists k. x = \text{scale } k \ y$ 
```

```
    unfolding proportionality-def
```

```
    by simp+
```

```
  from  $\langle \exists k. x = \text{scale } k \ y \rangle$  obtain k where  $x = \text{scale } k \ y$  by auto
```

```
  with  $\langle x \neq 0 \rangle$  have  $k \neq 0$  by simp
```

```
  with  $\langle x = \text{scale } k \ y \rangle$  have  $y = \text{scale } (1/k) \ x$  by simp
```

```

with ⟨x ≠ 0⟩ and ⟨y ≠ 0⟩ have (y, x) ∈ local.proportionality
  unfolding proportionality-def
  by auto
}
thus sym local.proportionality
  unfolding sym-def
  by blast
qed

```

```

lemma proportionality-trans: trans local.proportionality
proof -
  { fix x y z
    assume (x, y) ∈ local.proportionality and (y, z) ∈ local.proportionality
    hence x ≠ 0 and z ≠ 0 and ∃j. x = scale j y and ∃k. y = scale k z
      unfolding proportionality-def
      by simp+
    from ⟨∃j. x = scale j y⟩ and ⟨∃k. y = scale k z⟩
    obtain j and k where x = scale j y and y = scale k z by auto+
    hence x = scale (j * k) z by simp
    with ⟨x ≠ 0⟩ and ⟨z ≠ 0⟩ have (x, z) ∈ local.proportionality
      unfolding proportionality-def
      by auto
  }
thus trans local.proportionality
  unfolding trans-def
  by blast
qed

```

```

theorem proportionality-equiv: equiv local.non-zero-vectors local.proportionality
  unfolding equiv-def
  by (simp add:
    proportionality-subset
    proportionality-refl-on
    proportionality-sym
    proportionality-trans)

```

end

```

definition invertible-proportionality ::
  ((real(n::finite))n × (realn)n) set where
  invertible-proportionality ≜
  real-vector.proportionality ∩ (Collect invertible × Collect invertible)

```

```

lemma invertible-proportionality-equiv:
  equiv (Collect invertible :: (real(n::finite))n set)
  invertible-proportionality
  (is equiv ?invs -)
proof -
  from zero-not-invertible

```

```

have real-vector.non-zero-vectors  $\cap$  ?invs = ?invs
  unfolding real-vector.non-zero-vectors-def
  by auto
from equiv-restrict and real-vector.proportionality-equiv
have equiv (real-vector.non-zero-vectors  $\cap$  ?invs) invertible-proportionality
  unfolding invertible-proportionality-def
  by auto
with  $\langle$ real-vector.non-zero-vectors  $\cap$  ?invs = ?invs $\rangle$ 
show equiv ?invs invertible-proportionality
  by simp
qed

```

## 7.2 Points of the real projective plane

```

typedef proj2 = (real-vector.non-zero-vectors :: (real3) set) // real-vector.proportionality
proof
  have (axis 1 1 :: real3)  $\in$  real-vector.non-zero-vectors
    unfolding real-vector.non-zero-vectors-def
    by (simp add: axis-def vec-eq-iff [where 'a=real])
  thus real-vector.proportionality “ {axis 1 1}  $\in$  (real-vector.non-zero-vectors ::
(real3) set) // real-vector.proportionality
    unfolding quotient-def
    by auto
qed

```

**definition** *proj2-rep* :: *proj2*  $\Rightarrow$  *real*<sup>3</sup> **where**  
*proj2-rep* *x*  $\triangleq$   $\epsilon$  *v*. *v*  $\in$  *Rep-proj2* *x*

**definition** *proj2-abs* :: *real*<sup>3</sup>  $\Rightarrow$  *proj2* **where**  
*proj2-abs* *v*  $\triangleq$  *Abs-proj2* (*real-vector.proportionality* “ {*v*})

**lemma** *proj2-rep-in*: *proj2-rep* *x*  $\in$  *Rep-proj2* *x*

```

proof –
  let ?v = proj2-rep x
  from quotient-element-nonempty and
    real-vector.proportionality-equiv and
    Rep-proj2 [of x]
  have  $\exists$  w. w  $\in$  Rep-proj2 x
    by auto
  with someI-ex [of  $\lambda$  z. z  $\in$  Rep-proj2 x]
  show ?v  $\in$  Rep-proj2 x
    unfolding proj2-rep-def
    by simp
qed

```

**lemma** *proj2-rep-non-zero*: *proj2-rep* *x*  $\neq$  0

```

proof –
  from
    Union-quotient [of real-vector.non-zero-vectors real-vector.proportionality]

```

```

    and real-vector.proportionality-equiv
    and Rep-proj2 [of x] and proj2-rep-in [of x]
  have proj2-rep x ∈ real-vector.non-zero-vectors
    unfolding quotient-def
    by auto
  thus proj2-rep x ≠ 0
    unfolding real-vector.non-zero-vectors-def
    by simp
qed

```

```

lemma proj2-rep-abs:
  fixes v :: real^3
  assumes v ∈ real-vector.non-zero-vectors
  shows (v, proj2-rep (proj2-abs v)) ∈ real-vector.proportionality
proof -
  from ⟨v ∈ real-vector.non-zero-vectors⟩
  have real-vector.proportionality “ {v} ∈ (real-vector.non-zero-vectors :: (real^3)
set)//real-vector.proportionality
    unfolding quotient-def
    by auto
  with Abs-proj2-inverse
  have Rep-proj2 (proj2-abs v) = real-vector.proportionality “ {v}
    unfolding proj2-abs-def
    by simp
  with proj2-rep-in
  have proj2-rep (proj2-abs v) ∈ real-vector.proportionality “ {v} by auto
  thus (v, proj2-rep (proj2-abs v)) ∈ real-vector.proportionality by simp
qed

```

```

lemma proj2-abs-rep: proj2-abs (proj2-rep x) = x
proof -
  from partition-Image-element
  [of real-vector.non-zero-vectors
    real-vector.proportionality
    Rep-proj2 x
    proj2-rep x]
  and real-vector.proportionality-equiv
  and Rep-proj2 [of x] and proj2-rep-in [of x]
  have real-vector.proportionality “ {proj2-rep x} = Rep-proj2 x
    by simp
  with Rep-proj2-inverse show proj2-abs (proj2-rep x) = x
    unfolding proj2-abs-def
    by simp
qed

```

```

lemma proj2-abs-mult:
  assumes c ≠ 0
  shows proj2-abs (c *R v) = proj2-abs v
proof cases

```

```

assume  $v = 0$ 
thus  $\text{proj2-abs } (c *_R v) = \text{proj2-abs } v$  by simp
next
assume  $v \neq 0$ 
with  $\langle c \neq 0 \rangle$ 
have  $(c *_R v, v) \in \text{real-vector.propportionality}$ 
  and  $c *_R v \in \text{real-vector.non-zero-vectors}$ 
  and  $v \in \text{real-vector.non-zero-vectors}$ 
  unfolding real-vector.propportionality-def
  and real-vector.non-zero-vectors-def
  by simp-all
with eq-equiv-class-iff
[of real-vector.non-zero-vectors
 real-vector.propportionality
  $c *_R v$ 
  $v$ ]
and real-vector.propportionality-equiv
have  $\text{real-vector.propportionality} \{c *_R v\} =$ 
   $\text{real-vector.propportionality} \{v\}$ 
  by simp
thus  $\text{proj2-abs } (c *_R v) = \text{proj2-abs } v$ 
  unfolding proj2-abs-def
  by simp
qed

```

```

lemma proj2-abs-mult-rep:
assumes  $c \neq 0$ 
shows  $\text{proj2-abs } (c *_R \text{proj2-rep } x) = x$ 
using proj2-abs-mult and proj2-abs-rep and assms
by simp

```

```

lemma proj2-rep-inj: inj proj2-rep
by (simp add: inj-on-inverseI [of UNIV proj2-abs proj2-rep] proj2-abs-rep)

```

```

lemma proj2-rep-abs2:
assumes  $v \neq 0$ 
shows  $\exists k. k \neq 0 \wedge \text{proj2-rep } (\text{proj2-abs } v) = k *_R v$ 
proof –
from proj2-rep-abs [of v] and  $\langle v \neq 0 \rangle$ 
have  $(v, \text{proj2-rep } (\text{proj2-abs } v)) \in \text{real-vector.propportionality}$ 
  unfolding real-vector.non-zero-vectors-def
  by simp
then obtain  $c$  where  $v = c *_R \text{proj2-rep } (\text{proj2-abs } v)$ 
  unfolding real-vector.propportionality-def
  by auto
with  $\langle v \neq 0 \rangle$  have  $c \neq 0$  by auto
hence  $1/c \neq 0$  by simp

```

```

from  $\langle v = c *_R \text{proj2-rep } (\text{proj2-abs } v) \rangle$ 

```

**have**  $(1/c) *_R v = (1/c) *_R c *_R \text{proj2-rep} (\text{proj2-abs } v)$   
**by** *simp*  
**with**  $\langle c \neq 0 \rangle$  **have**  $\text{proj2-rep} (\text{proj2-abs } v) = (1/c) *_R v$  **by** *simp*  
  
**with**  $\langle 1/c \neq 0 \rangle$  **show**  $\exists k. k \neq 0 \wedge \text{proj2-rep} (\text{proj2-abs } v) = k *_R v$   
**by** *blast*  
**qed**

**lemma** *proj2-abs-abs-mult*:  
**assumes**  $\text{proj2-abs } v = \text{proj2-abs } w$  **and**  $w \neq 0$   
**shows**  $\exists c. v = c *_R w$   
**proof** *cases*  
**assume**  $v = 0$   
**hence**  $v = 0 *_R w$  **by** *simp*  
**thus**  $\exists c. v = c *_R w$  ..

**next**  
**assume**  $v \neq 0$   
**from**  $\langle \text{proj2-abs } v = \text{proj2-abs } w \rangle$   
**have**  $\text{proj2-rep} (\text{proj2-abs } v) = \text{proj2-rep} (\text{proj2-abs } w)$  **by** *simp*  
**with** *proj2-rep-abs2* **and**  $\langle w \neq 0 \rangle$   
**obtain**  $k$  **where**  $\text{proj2-rep} (\text{proj2-abs } v) = k *_R w$  **by** *auto*  
**with** *proj2-rep-abs2* [of  $v$ ] **and**  $\langle v \neq 0 \rangle$   
**obtain**  $j$  **where**  $j \neq 0$  **and**  $j *_R v = k *_R w$  **by** *auto*  
**hence**  $(1/j) *_R j *_R v = (1/j) *_R k *_R w$  **by** *simp*  
**with**  $\langle j \neq 0 \rangle$  **have**  $v = (k/j) *_R w$  **by** *simp*  
**thus**  $\exists c. v = c *_R w$  ..  
**qed**

**lemma** *dependent-proj2-abs*:  
**assumes**  $p \neq 0$  **and**  $q \neq 0$  **and**  $i \neq 0 \vee j \neq 0$  **and**  $i *_R p + j *_R q = 0$   
**shows**  $\text{proj2-abs } p = \text{proj2-abs } q$   
**proof** –  
**have**  $i \neq 0$   
**proof**  
**assume**  $i = 0$   
**with**  $\langle i \neq 0 \vee j \neq 0 \rangle$  **have**  $j \neq 0$  **by** *simp*  
**with**  $\langle i *_R p + j *_R q = 0 \rangle$  **and**  $\langle q \neq 0 \rangle$  **have**  $i *_R p \neq 0$  **by** *auto*  
**with**  $\langle i = 0 \rangle$  **show** *False* **by** *simp*  
**qed**  
**with**  $\langle p \neq 0 \rangle$  **and**  $\langle i *_R p + j *_R q = 0 \rangle$  **have**  $j \neq 0$  **by** *auto*  
  
**from**  $\langle i \neq 0 \rangle$   
**have**  $\text{proj2-abs } p = \text{proj2-abs } (i *_R p)$  **by** (*rule proj2-abs-mult [symmetric]*)  
**also from**  $\langle i *_R p + j *_R q = 0 \rangle$  **and** *proj2-abs-mult* [of  $-1 j *_R q$ ]  
**have**  $\dots = \text{proj2-abs } (j *_R q)$  **by** (*simp add: algebra-simps [symmetric]*)  
**also from**  $\langle j \neq 0 \rangle$  **have**  $\dots = \text{proj2-abs } q$  **by** (*rule proj2-abs-mult*)  
**finally show**  $\text{proj2-abs } p = \text{proj2-abs } q$  .  
**qed**

**lemma** *proj2-rep-dependent*:

**assumes**  $i *_{\mathbb{R}} \text{proj2-rep } v + j *_{\mathbb{R}} \text{proj2-rep } w = 0$

**(is**  $i *_{\mathbb{R}} ?p + j *_{\mathbb{R}} ?q = 0$ )

**and**  $i \neq 0 \vee j \neq 0$

**shows**  $v = w$

**proof** –

**have**  $?p \neq 0$  **and**  $?q \neq 0$  **by** (rule *proj2-rep-non-zero*)+

**with**  $\langle i \neq 0 \vee j \neq 0 \rangle$  **and**  $\langle i *_{\mathbb{R}} ?p + j *_{\mathbb{R}} ?q = 0 \rangle$

**have** *proj2-abs*  $?p = \text{proj2-abs } ?q$  **by** (*simp add: dependent-proj2-abs*)

**thus**  $v = w$  **by** (*simp add: proj2-abs-rep*)

**qed**

**lemma** *proj2-rep-independent*:

**assumes**  $p \neq q$

**shows** *independent*  $\{\text{proj2-rep } p, \text{proj2-rep } q\}$

**proof**

**let**  $?p' = \text{proj2-rep } p$

**let**  $?q' = \text{proj2-rep } q$

**let**  $?S = \{?p', ?q'\}$

**assume** *dependent*  $?S$

**from** *proj2-rep-inj* **and**  $\langle p \neq q \rangle$  **have**  $?p' \neq ?q'$

**unfolding** *inj-on-def*

**by** *auto*

**with** *dependent-explicit-2* [of  $?p' ?q'$ ] **and**  $\langle \text{dependent } ?S \rangle$

**obtain**  $i$  **and**  $j$  **where**  $i *_{\mathbb{R}} ?p' + j *_{\mathbb{R}} ?q' = 0$  **and**  $i \neq 0 \vee j \neq 0$

**by** (*simp add: scalar-equiv*) *auto*

**with** *proj2-rep-dependent* **have**  $p = q$  **by** *simp*

**with**  $\langle p \neq q \rangle$  **show** *False* ..

**qed**

### 7.3 Lines of the real projective plane

**definition** *proj2-Col* ::  $[\text{proj2}, \text{proj2}, \text{proj2}] \Rightarrow \text{bool}$  **where**

*proj2-Col*  $p$   $q$   $r \triangleq$

$(\exists i j k. i *_{\mathbb{R}} \text{proj2-rep } p + j *_{\mathbb{R}} \text{proj2-rep } q + k *_{\mathbb{R}} \text{proj2-rep } r = 0$   
 $\wedge (i \neq 0 \vee j \neq 0 \vee k \neq 0))$

**lemma** *proj2-Col-abs*:

**assumes**  $p \neq 0$  **and**  $q \neq 0$  **and**  $r \neq 0$  **and**  $i \neq 0 \vee j \neq 0 \vee k \neq 0$

**and**  $i *_{\mathbb{R}} p + j *_{\mathbb{R}} q + k *_{\mathbb{R}} r = 0$

**shows** *proj2-Col* (*proj2-abs*  $p$ ) (*proj2-abs*  $q$ ) (*proj2-abs*  $r$ )

(**is** *proj2-Col*  $?pp$   $?pq$   $?pr$ )

**proof** –

**from**  $\langle p \neq 0 \rangle$  **and** *proj2-rep-abs2*

**obtain**  $i'$  **where**  $i' \neq 0$  **and** *proj2-rep*  $?pp = i' *_{\mathbb{R}} p$  (**is**  $?rp = -$ ) **by** *auto*

**from**  $\langle q \neq 0 \rangle$  **and** *proj2-rep-abs2*

**obtain**  $j'$  **where**  $j' \neq 0$  **and** *proj2-rep*  $?pq = j' *_{\mathbb{R}} q$  (**is**  $?rq = -$ ) **by** *auto*

**from**  $\langle r \neq 0 \rangle$  **and** *proj2-rep-abs2*

**obtain**  $k'$  **where**  $k' \neq 0$  **and** *proj2-rep*  $?pr = k' *_{\mathbb{R}} r$  (**is**  $?rr = -$ ) **by** *auto*

**with**  $\langle i *_R p + j *_R q + k *_R r = 0 \rangle$   
**and**  $\langle i' \neq 0 \rangle$  **and**  $\langle \text{proj2-rep } ?pp = i' *_R p \rangle$   
**and**  $\langle j' \neq 0 \rangle$  **and**  $\langle \text{proj2-rep } ?pq = j' *_R q \rangle$   
**have**  $(i/i') *_R ?rp + (j/j') *_R ?rq + (k/k') *_R ?rr = 0$  **by** *simp*  
  
**from**  $\langle i' \neq 0 \rangle$  **and**  $\langle j' \neq 0 \rangle$  **and**  $\langle k' \neq 0 \rangle$  **and**  $\langle i \neq 0 \vee j \neq 0 \vee k \neq 0 \rangle$   
**have**  $i/i' \neq 0 \vee j/j' \neq 0 \vee k/k' \neq 0$  **by** *simp*  
**with**  $\langle (i/i') *_R ?rp + (j/j') *_R ?rq + (k/k') *_R ?rr = 0 \rangle$   
**show** *proj2-Col*  $?pp$   $?pq$   $?pr$  **by** (*unfold proj2-Col-def, best*)  
**qed**

**lemma** *proj2-Col-permute*:

**assumes** *proj2-Col*  $a$   $b$   $c$

**shows** *proj2-Col*  $a$   $c$   $b$

**and** *proj2-Col*  $b$   $a$   $c$

**proof** –

**let**  $?a' = \text{proj2-rep } a$

**let**  $?b' = \text{proj2-rep } b$

**let**  $?c' = \text{proj2-rep } c$

**from**  $\langle \text{proj2-Col } a$   $b$   $c \rangle$

**obtain**  $i$  **and**  $j$  **and**  $k$  **where**

$i *_R ?a' + j *_R ?b' + k *_R ?c' = 0$

**and**  $i \neq 0 \vee j \neq 0 \vee k \neq 0$

**unfolding** *proj2-Col-def*

**by** *auto*

**from**  $\langle i *_R ?a' + j *_R ?b' + k *_R ?c' = 0 \rangle$

**have**  $i *_R ?a' + k *_R ?c' + j *_R ?b' = 0$

**and**  $j *_R ?b' + i *_R ?a' + k *_R ?c' = 0$

**by** (*simp-all add: ac-simps*)

**moreover from**  $\langle i \neq 0 \vee j \neq 0 \vee k \neq 0 \rangle$

**have**  $i \neq 0 \vee k \neq 0 \vee j \neq 0$  **and**  $j \neq 0 \vee i \neq 0 \vee k \neq 0$  **by** *auto*

**ultimately show** *proj2-Col*  $a$   $c$   $b$  **and** *proj2-Col*  $b$   $a$   $c$

**unfolding** *proj2-Col-def*

**by** *auto*

**qed**

**lemma** *proj2-Col-coincide*: *proj2-Col*  $a$   $a$   $c$

**proof** –

**have**  $1 *_R \text{proj2-rep } a + (-1) *_R \text{proj2-rep } a + 0 *_R \text{proj2-rep } c = 0$

**by** *simp*

**moreover have**  $(1::\text{real}) \neq 0$  **by** *simp*

**ultimately show** *proj2-Col*  $a$   $a$   $c$

**unfolding** *proj2-Col-def*

**by** *blast*

**qed**

**lemma** *proj2-Col-iff*:

**assumes**  $a \neq r$

shows  $\text{proj2-Col } a \ r \ t \longleftrightarrow$   
 $t = a \vee (\exists i. t = \text{proj2-abs } (i *_{\mathbb{R}} (\text{proj2-rep } a) + (\text{proj2-rep } r)))$

**proof**

let  $?a' = \text{proj2-rep } a$   
 let  $?r' = \text{proj2-rep } r$   
 let  $?t' = \text{proj2-rep } t$

{ assume  $\text{proj2-Col } a \ r \ t$   
 then obtain  $h$  and  $j$  and  $k$  where  
 $h *_{\mathbb{R}} ?a' + j *_{\mathbb{R}} ?r' + k *_{\mathbb{R}} ?t' = 0$   
 and  $h \neq 0 \vee j \neq 0 \vee k \neq 0$   
 unfolding  $\text{proj2-Col-def}$   
 by *auto*

show  $t = a \vee (\exists i. t = \text{proj2-abs } (i *_{\mathbb{R}} ?a' + ?r'))$

**proof cases**

assume  $j = 0$   
 with  $\langle h \neq 0 \vee j \neq 0 \vee k \neq 0 \rangle$  have  $h \neq 0 \vee k \neq 0$  by *simp*  
 with  $\text{proj2-rep-dependent}$   
 and  $\langle h *_{\mathbb{R}} ?a' + j *_{\mathbb{R}} ?r' + k *_{\mathbb{R}} ?t' = 0 \rangle$   
 and  $\langle j = 0 \rangle$   
 have  $t = a$  by *auto*  
 thus  $t = a \vee (\exists i. t = \text{proj2-abs } (i *_{\mathbb{R}} ?a' + ?r'))$  ..

**next**

assume  $j \neq 0$   
 have  $k \neq 0$   
**proof** (*rule ccontr*)  
 assume  $\neg k \neq 0$   
 with  $\text{proj2-rep-dependent}$   
 and  $\langle h *_{\mathbb{R}} ?a' + j *_{\mathbb{R}} ?r' + k *_{\mathbb{R}} ?t' = 0 \rangle$   
 and  $\langle j \neq 0 \rangle$   
 have  $a = r$  by *simp*  
 with  $\langle a \neq r \rangle$  show *False* ..

**qed**

from  $\langle h *_{\mathbb{R}} ?a' + j *_{\mathbb{R}} ?r' + k *_{\mathbb{R}} ?t' = 0 \rangle$   
 have  $h *_{\mathbb{R}} ?a' + j *_{\mathbb{R}} ?r' + k *_{\mathbb{R}} ?t' - k *_{\mathbb{R}} ?t' = -k *_{\mathbb{R}} ?t'$  by *simp*  
 hence  $h *_{\mathbb{R}} ?a' + j *_{\mathbb{R}} ?r' = -k *_{\mathbb{R}} ?t'$  by *simp*  
 with  $\text{proj2-abs-mult-rep}$  [of  $-k$ ] and  $\langle k \neq 0 \rangle$   
 have  $\text{proj2-abs } (h *_{\mathbb{R}} ?a' + j *_{\mathbb{R}} ?r') = t$  by *simp*  
 with  $\text{proj2-abs-mult}$  [of  $1/j$   $h *_{\mathbb{R}} ?a' + j *_{\mathbb{R}} ?r'$ ] and  $\langle j \neq 0 \rangle$   
 have  $\text{proj2-abs } ((h/j) *_{\mathbb{R}} ?a' + ?r') = t$   
 by (*simp add: scaleR-right-distrib*)  
 hence  $\exists i. t = \text{proj2-abs } (i *_{\mathbb{R}} ?a' + ?r')$  by *auto*  
 thus  $t = a \vee (\exists i. t = \text{proj2-abs } (i *_{\mathbb{R}} ?a' + ?r'))$  ..

**qed**

}

{ assume  $t = a \vee (\exists i. t = \text{proj2-abs } (i *_{\mathbb{R}} ?a' + ?r'))$

```

show proj2-Col a r t
proof cases
  assume t = a
  with proj2-Col-coincide and proj2-Col-permute
  show proj2-Col a r t by blast
next
  assume t ≠ a
  with ⟨t = a ∨ (∃ i. t = proj2-abs (i *R ?a' + ?r'))⟩
  obtain i where t = proj2-abs (i *R ?a' + ?r') by auto
  from proj2-rep-dependent [of i a 1 r] and ⟨a ≠ r⟩
  have i *R ?a' + ?r' ≠ 0 by auto
  with proj2-rep-abs2 and ⟨t = proj2-abs (i *R ?a' + ?r')⟩
  obtain j where ?t' = j *R (i *R ?a' + ?r') by auto
  hence ?t' - ?t' = (j * i) *R ?a' + j *R ?r' + (-1) *R ?t'
    by (simp add: scaleR-right-distrib)
  hence (j * i) *R ?a' + j *R ?r' + (-1) *R ?t' = 0 by simp
  have ∃ h j k. h *R ?a' + j *R ?r' + k *R ?t' = 0
    ∧ (h ≠ 0 ∨ j ≠ 0 ∨ k ≠ 0)
  proof standard+
    from ⟨(j * i) *R ?a' + j *R ?r' + (-1) *R ?t' = 0⟩
    show (j * i) *R ?a' + j *R ?r' + (-1) *R ?t' = 0 .
    show j * i ≠ 0 ∨ j ≠ 0 ∨ (-1::real) ≠ 0 by simp
  qed
  thus proj2-Col a r t
    unfolding proj2-Col-def .
qed
}
qed

definition proj2-Col-coeff :: proj2 ⇒ proj2 ⇒ proj2 ⇒ real where
  proj2-Col-coeff a r t ≜ ε i. t = proj2-abs (i *R proj2-rep a + proj2-rep r)

lemma proj2-Col-coeff:
  assumes proj2-Col a r t and a ≠ r and t ≠ a
  shows t = proj2-abs ((proj2-Col-coeff a r t) *R proj2-rep a + proj2-rep r)
proof -
  from ⟨a ≠ r⟩ and ⟨proj2-Col a r t⟩ and ⟨t ≠ a⟩ and proj2-Col-iff
  have ∃ i. t = proj2-abs (i *R proj2-rep a + proj2-rep r) by simp
  thus t = proj2-abs ((proj2-Col-coeff a r t) *R proj2-rep a + proj2-rep r)
    by (unfold proj2-Col-coeff-def) (rule someI-ex)
qed

lemma proj2-Col-coeff-unique':
  assumes a ≠ 0 and r ≠ 0 and proj2-abs a ≠ proj2-abs r
  and proj2-abs (i *R a + r) = proj2-abs (j *R a + r)
  shows i = j
proof -
  from ⟨a ≠ 0⟩ and ⟨r ≠ 0⟩ and ⟨proj2-abs a ≠ proj2-abs r⟩
  and dependent-proj2-abs [of a r - 1]

```

**have**  $i *_R a + r \neq 0$  **and**  $j *_R a + r \neq 0$  **by** *auto*  
**with** *proj2-rep-abs2* [of  $i *_R a + r$ ]  
**and** *proj2-rep-abs2* [of  $j *_R a + r$ ]  
**obtain**  $k$  **and**  $l$  **where**  $k \neq 0$   
**and** *proj2-rep* (*proj2-abs* ( $i *_R a + r$ )) =  $k *_R (i *_R a + r)$   
**and** *proj2-rep* (*proj2-abs* ( $j *_R a + r$ )) =  $l *_R (j *_R a + r)$   
**by** *auto*  
**with**  $\langle \text{proj2-abs } (i *_R a + r) = \text{proj2-abs } (j *_R a + r) \rangle$   
**have**  $(k * i) *_R a + k *_R r = (l * j) *_R a + l *_R r$   
**by** (*simp add: scaleR-right-distrib*)  
**hence**  $(k * i - l * j) *_R a + (k - l) *_R r = 0$   
**by** (*simp add: algebra-simps vec-eq-iff*)  
**with**  $\langle a \neq 0 \rangle$  **and**  $\langle r \neq 0 \rangle$  **and**  $\langle \text{proj2-abs } a \neq \text{proj2-abs } r \rangle$   
**and** *dependent-proj2-abs* [of  $a$   $r$   $k * i - l * j$   $k - l$ ]  
**have**  $k * i - l * j = 0$  **and**  $k - l = 0$  **by** *auto*  
**from**  $\langle k - l = 0 \rangle$  **have**  $k = l$  **by** *simp*  
**with**  $\langle k * i - l * j = 0 \rangle$  **have**  $k * i = k * j$  **by** *simp*  
**with**  $\langle k \neq 0 \rangle$  **show**  $i = j$  **by** *simp*  
**qed**

**lemma** *proj2-Col-coeff-unique*:

**assumes**  $a \neq r$   
**and** *proj2-abs* ( $i *_R \text{proj2-rep } a + \text{proj2-rep } r$ )  
= *proj2-abs* ( $j *_R \text{proj2-rep } a + \text{proj2-rep } r$ )  
**shows**  $i = j$

**proof** –

**let**  $?a' = \text{proj2-rep } a$   
**let**  $?r' = \text{proj2-rep } r$   
**have**  $?a' \neq 0$  **and**  $?r' \neq 0$  **by** (*rule proj2-rep-non-zero*)+

**from**  $\langle a \neq r \rangle$  **have** *proj2-abs*  $?a' \neq \text{proj2-abs } ?r'$  **by** (*simp add: proj2-abs-rep*)  
**with**  $\langle ?a' \neq 0 \rangle$  **and**  $\langle ?r' \neq 0 \rangle$   
**and**  $\langle \text{proj2-abs } (i *_R ?a' + ?r') = \text{proj2-abs } (j *_R ?a' + ?r') \rangle$   
**and** *proj2-Col-coeff-unique'*  
**show**  $i = j$  **by** *simp*

**qed**

**datatype** *proj2-line* = *P2L* *proj2*

**definition** *L2P* :: *proj2-line*  $\Rightarrow$  *proj2* **where**

*L2P*  $l \triangleq \text{case } l \text{ of } P2L \ p \Rightarrow p$

**lemma** *L2P-P2L* [*simp*]: *L2P* (*P2L*  $p$ ) =  $p$

**unfolding** *L2P-def*

**by** *simp*

**lemma** *P2L-L2P* [*simp*]: *P2L* (*L2P*  $l$ ) =  $l$

**by** (*induct*  $l$ ) *simp*

**lemma** *L2P-inj* [*simp*]:  
**assumes**  $L2P\ l = L2P\ m$   
**shows**  $l = m$   
**using** *P2L-L2P* [*of l*] **and** *assms*  
**by** *simp*

**lemma** *P2L-to-L2P*:  $P2L\ p = l \longleftrightarrow p = L2P\ l$

**proof**  
**assume**  $P2L\ p = l$   
**hence**  $L2P\ (P2L\ p) = L2P\ l$  **by** *simp*  
**thus**  $p = L2P\ l$  **by** *simp*  
**next**  
**assume**  $p = L2P\ l$   
**thus**  $P2L\ p = l$  **by** *simp*  
**qed**

**definition** *proj2-line-abs* ::  $real^3 \Rightarrow proj2-line$  **where**  
*proj2-line-abs*  $v \triangleq P2L\ (proj2-abs\ v)$

**definition** *proj2-line-rep* ::  $proj2-line \Rightarrow real^3$  **where**  
*proj2-line-rep*  $l \triangleq proj2-rep\ (L2P\ l)$

**lemma** *proj2-line-rep-abs*:  
**assumes**  $v \neq 0$   
**shows**  $\exists k. k \neq 0 \wedge proj2-line-rep\ (proj2-line-abs\ v) = k *_{\mathbb{R}} v$   
**unfolding** *proj2-line-rep-def* **and** *proj2-line-abs-def*  
**using** *proj2-rep-abs2* **and**  $\langle v \neq 0 \rangle$   
**by** *simp*

**lemma** *proj2-line-abs-rep* [*simp*]:  $proj2-line-abs\ (proj2-line-rep\ l) = l$   
**unfolding** *proj2-line-abs-def* **and** *proj2-line-rep-def*  
**by** (*simp add: proj2-abs-rep*)

**lemma** *proj2-line-rep-non-zero*:  $proj2-line-rep\ l \neq 0$   
**unfolding** *proj2-line-rep-def*  
**using** *proj2-rep-non-zero*  
**by** *simp*

**lemma** *proj2-line-rep-dependent*:  
**assumes**  $i *_{\mathbb{R}} proj2-line-rep\ l + j *_{\mathbb{R}} proj2-line-rep\ m = 0$   
**and**  $i \neq 0 \vee j \neq 0$   
**shows**  $l = m$   
**using** *proj2-rep-dependent* [*of i L2P l j L2P m*] **and** *assms*  
**unfolding** *proj2-line-rep-def*  
**by** *simp*

**lemma** *proj2-line-abs-mult*:  
**assumes**  $k \neq 0$   
**shows**  $proj2-line-abs\ (k *_{\mathbb{R}} v) = proj2-line-abs\ v$

**unfolding** *proj2-line-abs-def*  
**using**  $\langle k \neq 0 \rangle$   
**by** (*subst proj2-abs-mult*) *simp-all*

**lemma** *proj2-line-abs-abs-mult*:  
**assumes** *proj2-line-abs*  $v = \text{proj2-line-abs } w$  **and**  $w \neq 0$   
**shows**  $\exists k. v = k *_R w$   
**using** *assms*  
**by** (*unfold proj2-line-abs-def*) (*simp add: proj2-abs-abs-mult*)

**definition** *proj2-incident* :: *proj2*  $\Rightarrow$  *proj2-line*  $\Rightarrow$  *bool* **where**  
*proj2-incident*  $p \ l \triangleq (\text{proj2-rep } p) \cdot (\text{proj2-line-rep } l) = 0$

**lemma** *proj2-points-define-line*:  
**shows**  $\exists l. \text{proj2-incident } p \ l \wedge \text{proj2-incident } q \ l$   
**proof** –  
**let**  $?p' = \text{proj2-rep } p$   
**let**  $?q' = \text{proj2-rep } q$   
**let**  $?B = \{?p', ?q'\}$   
**from** *card-suc-ge-insert* [*of*  $?p' \ \{?q'\}$ ] **have**  $\text{card } ?B \leq 2$  **by** *simp*  
**with** *dim-le-card'* [*of*  $?B$ ] **have**  $\text{dim } ?B < 3$  **by** *simp*  
**with** *lowdim-subset-hyperplane* [*of*  $?B$ ]  
**obtain**  $l'$  **where**  $l' \neq 0$  **and**  $\text{span } ?B \subseteq \{x. l' \cdot x = 0\}$  **by** *auto*  
**let**  $?l = \text{proj2-line-abs } l'$   
**let**  $?l'' = \text{proj2-line-rep } ?l$   
**from** *proj2-line-rep-abs* **and**  $\langle l' \neq 0 \rangle$   
**obtain**  $k$  **where**  $?l'' = k *_R l'$  **by** *auto*  
  
**have**  $?p' \in ?B$  **and**  $?q' \in ?B$  **by** *simp-all*  
**with** *span-superset* [*of*  $?B$ ] **and**  $\langle \text{span } ?B \subseteq \{x. l' \cdot x = 0\} \rangle$   
**have**  $l' \cdot ?p' = 0$  **and**  $l' \cdot ?q' = 0$  **by** *auto*  
**hence**  $?p' \cdot l' = 0$  **and**  $?q' \cdot l' = 0$  **by** (*simp-all add: inner-commute*)  
**with** *dot-scaleR-mult(2)* [*of*  $- \ k \ l'$ ] **and**  $\langle ?l'' = k *_R l' \rangle$   
**have** *proj2-incident*  $p \ ?l \wedge \text{proj2-incident } q \ ?l$   
**unfolding** *proj2-incident-def*  
**by** *simp*  
**thus**  $\exists l. \text{proj2-incident } p \ l \wedge \text{proj2-incident } q \ l$  **by** *auto*  
**qed**

**definition** *proj2-line-through* :: *proj2*  $\Rightarrow$  *proj2*  $\Rightarrow$  *proj2-line* **where**  
*proj2-line-through*  $p \ q \triangleq \epsilon \ l. \text{proj2-incident } p \ l \wedge \text{proj2-incident } q \ l$

**lemma** *proj2-line-through-incident*:  
**shows** *proj2-incident*  $p \ (\text{proj2-line-through } p \ q)$   
**and** *proj2-incident*  $q \ (\text{proj2-line-through } p \ q)$   
**unfolding** *proj2-line-through-def*  
**using** *proj2-points-define-line*  
**and** *someI-ex* [*of*  $\lambda \ l. \text{proj2-incident } p \ l \wedge \text{proj2-incident } q \ l$ ]  
**by** *simp-all*

**lemma** *proj2-line-through-unique*:

assumes  $p \neq q$  and *proj2-incident*  $p$   $l$  and *proj2-incident*  $q$   $l$   
shows  $l = \text{proj2-line-through } p$   $q$

**proof** –

let  $?l' = \text{proj2-line-rep } l$

let  $?m = \text{proj2-line-through } p$   $q$

let  $?m' = \text{proj2-line-rep } ?m$

let  $?p' = \text{proj2-rep } p$

let  $?q' = \text{proj2-rep } q$

let  $?A = \{?p', ?q'\}$

let  $?B = \text{insert } ?m' ?A$

from *proj2-line-through-incident*

have *proj2-incident*  $p$   $?m$  and *proj2-incident*  $q$   $?m$  by *simp-all*

with  $\langle \text{proj2-incident } p$   $l \rangle$  and  $\langle \text{proj2-incident } q$   $l \rangle$

have *ortho*:  $\bigwedge w. w \in ?A \implies \text{orthogonal } ?m' w \bigwedge w. w \in ?A \implies \text{orthogonal } ?l' w$

unfolding *proj2-incident-def* and *orthogonal-def*

by (*metis empty-iff inner-commute insert-iff*)+

from *proj2-rep-independent* and  $\langle p \neq q \rangle$  have *independent*  $?A$  by *simp*

from *proj2-line-rep-non-zero* have  $?m' \neq 0$  by *simp*

with *orthogonal-independent*  $\langle \text{independent } ?A \rangle$  *ortho*

have *independent*  $?B$  by *auto*

from *proj2-rep-inj* and  $\langle p \neq q \rangle$  have  $?p' \neq ?q'$

unfolding *inj-on-def*

by *auto*

hence  $\text{card } ?A = 2$  by *simp*

moreover have  $?m' \notin ?A$

using *ortho(1)* *orthogonal-self* *proj2-line-rep-non-zero* by *auto*

ultimately have  $\text{card } ?B = 3$  by *simp*

with *independent-is-basis* [of  $?B$ ] and  $\langle \text{independent } ?B \rangle$

have *is-basis*  $?B$  by *simp*

with *basis-expand* obtain  $c$  where  $?l' = (\sum v \in ?B. c v *_R v)$  by *auto*

let  $?l'' = ?l' - c ?m' *_R ?m'$

from  $\langle ?l' = (\sum v \in ?B. c v *_R v) \rangle$  and  $\langle ?m' \notin ?A \rangle$

have  $?l'' = (\sum v \in ?A. c v *_R v)$  by *simp*

with *orthogonal-sum* [of  $?A$ ] *ortho*

have *orthogonal*  $?l' ?l''$  and *orthogonal*  $?m' ?l''$

by (*simp-all add: scalar-equiv*)

from  $\langle \text{orthogonal } ?m' ?l'' \rangle$

have *orthogonal*  $(c ?m' *_R ?m') ?l''$  by (*simp add: orthogonal-clauses*)

with  $\langle \text{orthogonal } ?l' ?l'' \rangle$

have *orthogonal*  $?l'' ?l''$  by (*simp add: orthogonal-clauses*)

with *orthogonal-self-eq-0* [of  $?l''$ ] have  $?l'' = 0$  by *simp*

with *proj2-line-rep-dependent* [of  $1$   $l - c ?m' ?m$ ] show  $l = ?m$  by *simp*

qed

**lemma** *proj2-incident-unique*:

assumes *proj2-incident*  $p$   $l$

**and** *proj2-incident*  $q\ l$   
**and** *proj2-incident*  $p\ m$   
**and** *proj2-incident*  $q\ m$   
**shows**  $p = q \vee l = m$   
**proof** *cases*  
**assume**  $p = q$   
**thus**  $p = q \vee l = m$  ..  
**next**  
**assume**  $p \neq q$   
**with**  $\langle$ *proj2-incident*  $p\ l$  $\rangle$  **and**  $\langle$ *proj2-incident*  $q\ l$  $\rangle$   
**and** *proj2-line-through-unique*  
**have**  $l =$  *proj2-line-through*  $p\ q$  **by** *simp*  
**moreover from**  $\langle p \neq q \rangle$  **and**  $\langle$ *proj2-incident*  $p\ m$  $\rangle$  **and**  $\langle$ *proj2-incident*  $q\ m$  $\rangle$   
**have**  $m =$  *proj2-line-through*  $p\ q$  **by** (*rule proj2-line-through-unique*)  
**ultimately show**  $p = q \vee l = m$  **by** *simp*  
**qed**

**lemma** *proj2-lines-define-point*:  $\exists p.$  *proj2-incident*  $p\ l \wedge$  *proj2-incident*  $p\ m$   
**proof** –

**let**  $?l' =$  *L2P*  $l$   
**let**  $?m' =$  *L2P*  $m$   
**from** *proj2-points-define-line* [*of*  $?l'$   $?m'$ ]  
**obtain**  $p'$  **where** *proj2-incident*  $?l'\ p' \wedge$  *proj2-incident*  $?m'\ p'$  **by** *auto*  
**hence** *proj2-incident* (*L2P*  $p'$ )  $l \wedge$  *proj2-incident* (*L2P*  $p'$ )  $m$   
**unfolding** *proj2-incident-def* **and** *proj2-line-rep-def*  
**by** (*simp add: inner-commute*)  
**thus**  $\exists p.$  *proj2-incident*  $p\ l \wedge$  *proj2-incident*  $p\ m$  **by** *auto*  
**qed**

**definition** *proj2-intersection* :: *proj2-line*  $\Rightarrow$  *proj2-line*  $\Rightarrow$  *proj2* **where**  
*proj2-intersection*  $l\ m \triangleq$  *L2P* (*proj2-line-through* (*L2P*  $l$ ) (*L2P*  $m$ ))

**lemma** *proj2-incident-switch*:  
**assumes** *proj2-incident*  $p\ l$   
**shows** *proj2-incident* (*L2P*  $l$ ) (*P2L*  $p$ )  
**using** *assms*  
**unfolding** *proj2-incident-def* **and** *proj2-line-rep-def*  
**by** (*simp add: inner-commute*)

**lemma** *proj2-intersection-incident*:  
**shows** *proj2-incident* (*proj2-intersection*  $l\ m$ )  $l$   
**and** *proj2-incident* (*proj2-intersection*  $l\ m$ )  $m$   
**using** *proj2-line-through-incident*(1) [*of* *L2P*  $l$  *L2P*  $m$ ]  
**and** *proj2-line-through-incident*(2) [*of* *L2P*  $m$  *L2P*  $l$ ]  
**and** *proj2-incident-switch* [*of* *L2P*  $l$ ]  
**and** *proj2-incident-switch* [*of* *L2P*  $m$ ]  
**unfolding** *proj2-intersection-def*  
**by** *simp-all*

**lemma** *proj2-intersection-unique*:  
 assumes  $l \neq m$  and *proj2-incident*  $p$   $l$  and *proj2-incident*  $p$   $m$   
 shows  $p = \text{proj2-intersection } l \ m$   
**proof** –  
 from  $\langle l \neq m \rangle$  have  $L2P \ l \neq L2P \ m$  by *auto*  
 from  $\langle \text{proj2-incident } p \ l \rangle$  and  $\langle \text{proj2-incident } p \ m \rangle$   
 and *proj2-incident-switch*  
 have *proj2-incident*  $(L2P \ l)$   $(P2L \ p)$  and *proj2-incident*  $(L2P \ m)$   $(P2L \ p)$   
 by *simp-all*  
 with  $\langle L2P \ l \neq L2P \ m \rangle$  and *proj2-line-through-unique*  
 have  $P2L \ p = \text{proj2-line-through } (L2P \ l) \ (L2P \ m)$  by *simp*  
 thus  $p = \text{proj2-intersection } l \ m$   
 unfolding *proj2-intersection-def*  
 by (*simp add: P2L-to-L2P*)  
**qed**

**lemma** *proj2-not-self-incident*:  
 $\neg (\text{proj2-incident } p \ (P2L \ p))$   
 unfolding *proj2-incident-def* and *proj2-line-rep-def*  
 using *proj2-rep-non-zero* and *inner-eq-zero-iff* [*of proj2-rep p*]  
 by *simp*

**lemma** *proj2-another-point-on-line*:  
 $\exists q. q \neq p \wedge \text{proj2-incident } q \ l$   
**proof** –  
 let  $?m = P2L \ p$   
 let  $?q = \text{proj2-intersection } l \ ?m$   
 from *proj2-intersection-incident*  
 have *proj2-incident*  $?q \ l$  and *proj2-incident*  $?q \ ?m$  by *simp-all*  
 from  $\langle \text{proj2-incident } ?q \ ?m \rangle$  and *proj2-not-self-incident* have  $?q \neq p$  by *auto*  
 with  $\langle \text{proj2-incident } ?q \ l \rangle$  show  $\exists q. q \neq p \wedge \text{proj2-incident } q \ l$  by *auto*  
**qed**

**lemma** *proj2-another-line-through-point*:  
 $\exists m. m \neq l \wedge \text{proj2-incident } p \ m$   
**proof** –  
 from *proj2-another-point-on-line*  
 obtain  $q$  where  $q \neq L2P \ l \wedge \text{proj2-incident } q \ (P2L \ p)$  by *auto*  
 with *proj2-incident-switch* [*of q P2L p*]  
 have  $P2L \ q \neq l \wedge \text{proj2-incident } p \ (P2L \ q)$  by *auto*  
 thus  $\exists m. m \neq l \wedge \text{proj2-incident } p \ m$  ..  
**qed**

**lemma** *proj2-incident-abs*:  
 assumes  $v \neq 0$  and  $w \neq 0$   
 shows *proj2-incident*  $(\text{proj2-abs } v)$   $(\text{proj2-line-abs } w) \longleftrightarrow v \cdot w = 0$   
**proof** –  
 from  $\langle v \neq 0 \rangle$  and *proj2-rep-abs2*  
 obtain  $j$  where  $j \neq 0$  and *proj2-rep*  $(\text{proj2-abs } v) = j \ *_R \ v$  by *auto*

**from**  $\langle w \neq 0 \rangle$  **and** *proj2-line-rep-abs*  
**obtain**  $k$  **where**  $k \neq 0$   
**and** *proj2-line-rep* (*proj2-line-abs*  $w$ ) =  $k *_R w$   
**by** *auto*  
**with**  $\langle j \neq 0 \rangle$  **and**  $\langle \text{proj2-rep } (\text{proj2-abs } v) = j *_R v \rangle$   
**show** *proj2-incident* (*proj2-abs*  $v$ ) (*proj2-line-abs*  $w$ )  $\longleftrightarrow v \cdot w = 0$   
**unfolding** *proj2-incident-def*  
**by** (*simp add: dot-scaleR-mult*)  
**qed**

**lemma** *proj2-incident-left-abs*:  
**assumes**  $v \neq 0$   
**shows** *proj2-incident* (*proj2-abs*  $v$ )  $l \longleftrightarrow v \cdot (\text{proj2-line-rep } l) = 0$   
**proof** –  
**have** *proj2-line-rep*  $l \neq 0$  **by** (*rule proj2-line-rep-non-zero*)  
**with**  $\langle v \neq 0 \rangle$  **and** *proj2-incident-abs* [*of*  $v$  *proj2-line-rep*  $l$ ]  
**show** *proj2-incident* (*proj2-abs*  $v$ )  $l \longleftrightarrow v \cdot (\text{proj2-line-rep } l) = 0$  **by** *simp*  
**qed**

**lemma** *proj2-incident-right-abs*:  
**assumes**  $v \neq 0$   
**shows** *proj2-incident*  $p$  (*proj2-line-abs*  $v$ )  $\longleftrightarrow (\text{proj2-rep } p) \cdot v = 0$   
**proof** –  
**have** *proj2-rep*  $p \neq 0$  **by** (*rule proj2-rep-non-zero*)  
**with**  $\langle v \neq 0 \rangle$  **and** *proj2-incident-abs* [*of* *proj2-rep*  $p$   $v$ ]  
**show** *proj2-incident*  $p$  (*proj2-line-abs*  $v$ )  $\longleftrightarrow (\text{proj2-rep } p) \cdot v = 0$   
**by** (*simp add: proj2-abs-rep*)  
**qed**

**definition** *proj2-set-Col* :: *proj2 set*  $\Rightarrow$  *bool* **where**  
*proj2-set-Col*  $S \triangleq \exists l. \forall p \in S. \text{proj2-incident } p l$

**lemma** *proj2-subset-Col*:  
**assumes**  $T \subseteq S$  **and** *proj2-set-Col*  $S$   
**shows** *proj2-set-Col*  $T$   
**using**  $\langle T \subseteq S \rangle$  **and**  $\langle \text{proj2-set-Col } S \rangle$   
**by** (*unfold proj2-set-Col-def*) *auto*

**definition** *proj2-no-3-Col* :: *proj2 set*  $\Rightarrow$  *bool* **where**  
*proj2-no-3-Col*  $S \triangleq \text{card } S = 4 \wedge (\forall p \in S. \neg \text{proj2-set-Col } (S - \{p\}))$

**lemma** *proj2-Col-iff-not-invertible*:  
*proj2-Col*  $p q r$   
 $\longleftrightarrow \neg \text{invertible } (\text{vector } [\text{proj2-rep } p, \text{proj2-rep } q, \text{proj2-rep } r] :: \text{real}^{\wedge 3})$   
**(is -  $\longleftrightarrow \neg \text{invertible } (\text{vector } [?u, ?v, ?w])$ )**  
**proof** –  
**let**  $?M = \text{vector } [?u, ?v, ?w] :: \text{real}^{\wedge 3}$   
**have** *proj2-Col*  $p q r \longleftrightarrow (\exists x. x \neq 0 \wedge x v * ?M = 0)$

**proof**  
**assume** *proj2-Col p q r*  
**then obtain** *i* **and** *j* **and** *k*  
**where**  $i \neq 0 \vee j \neq 0 \vee k \neq 0$  **and**  $i *_R ?u + j *_R ?v + k *_R ?w = 0$   
**unfolding** *proj2-Col-def*  
**by** *auto*  
**let**  $?x = \text{vector } [i,j,k] :: \text{real}^3$   
**from**  $\langle i \neq 0 \vee j \neq 0 \vee k \neq 0 \rangle$   
**have**  $?x \neq 0$   
**unfolding** *vector-def*  
**by** (*simp add: vec-eq-iff forall-3*)  
**moreover** {  
**from**  $\langle i *_R ?u + j *_R ?v + k *_R ?w = 0 \rangle$   
**have**  $?x v * ?M = 0$   
**unfolding** *vector-def* **and** *vector-matrix-mult-def*  
**by** (*simp add: sum-3 vec-eq-iff algebra-simps*) }  
**ultimately show**  $\exists x. x \neq 0 \wedge x v * ?M = 0$  **by** *auto*  
**next**  
**assume**  $\exists x. x \neq 0 \wedge x v * ?M = 0$   
**then obtain** *x* **where**  $x \neq 0$  **and**  $x v * ?M = 0$  **by** *auto*  
**let**  $?i = x\$1$   
**let**  $?j = x\$2$   
**let**  $?k = x\$3$   
**from**  $\langle x \neq 0 \rangle$  **have**  $?i \neq 0 \vee ?j \neq 0 \vee ?k \neq 0$  **by** (*simp add: vec-eq-iff forall-3*)  
**moreover** {  
**from**  $\langle x v * ?M = 0 \rangle$   
**have**  $?i *_R ?u + ?j *_R ?v + ?k *_R ?w = 0$   
**unfolding** *vector-matrix-mult-def* **and** *sum-3* **and** *vector-def*  
**by** (*simp add: vec-eq-iff algebra-simps*) }  
**ultimately show** *proj2-Col p q r*  
**unfolding** *proj2-Col-def*  
**by** *auto*  
**qed**  
**also from** *matrix-right-invertible-ker* [*of ?M*]  
**have**  $\dots \longleftrightarrow \neg (\exists M'. ?M ** M' = \text{mat } 1)$  **by** *auto*  
**also from** *matrix-left-right-inverse*  
**have**  $\dots \longleftrightarrow \neg \text{invertible } ?M$   
**unfolding** *invertible-def*  
**by** *auto*  
**finally show** *proj2-Col p q r*  $\longleftrightarrow \neg \text{invertible } ?M$  .  
**qed**

**lemma** *not-invertible-iff-proj2-set-Col*:

$\neg \text{invertible } (\text{vector } [\text{proj2-rep } p, \text{proj2-rep } q, \text{proj2-rep } r] :: \text{real}^3^3)$   
 $\longleftrightarrow \text{proj2-set-Col } \{p,q,r\}$   
(is  $\neg \text{invertible } ?M \longleftrightarrow -$ )

**proof** –

**from** *left-invertible-iff-invertible*

**have**  $\neg \text{invertible } ?M \longleftrightarrow \neg (\exists M'. M' ** ?M = \text{mat } 1)$  **by** *auto*

**also from** *matrix-left-invertible-ker* [of ?M]  
**have** ...  $\longleftrightarrow (\exists y. y \neq 0 \wedge ?M *v y = 0)$  **by** *auto*  
**also have** ...  $\longleftrightarrow (\exists l. \forall s \in \{p, q, r\}. \text{proj2-incident } s \ l)$   
**proof**  
    **assume**  $\exists y. y \neq 0 \wedge ?M *v y = 0$   
    **then obtain** *y* **where**  $y \neq 0$  **and**  $?M *v y = 0$  **by** *auto*  
    **let**  $?l = \text{proj2-line-abs } y$   
    **from**  $\langle ?M *v y = 0 \rangle$   
    **have**  $\forall s \in \{p, q, r\}. \text{proj2-rep } s \cdot y = 0$   
    **unfolding** *vector-def*  
    **and** *matrix-vector-mult-def*  
    **and** *inner-vec-def*  
    **and** *sum-3*  
    **by** (*simp add: vec-eq-iff forall-3*)  
    **with**  $\langle y \neq 0 \rangle$  **and** *proj2-incident-right-abs*  
    **have**  $\forall s \in \{p, q, r\}. \text{proj2-incident } s \ ?l$  **by** *simp*  
    **thus**  $\exists l. \forall s \in \{p, q, r\}. \text{proj2-incident } s \ l ..$   
**next**  
    **assume**  $\exists l. \forall s \in \{p, q, r\}. \text{proj2-incident } s \ l$   
    **then obtain** *l* **where**  $\forall s \in \{p, q, r\}. \text{proj2-incident } s \ l ..$   
    **let**  $?y = \text{proj2-line-rep } l$   
    **have**  $?y \neq 0$  **by** (*rule proj2-line-rep-non-zero*)  
    **moreover** {  
        **from**  $\langle \forall s \in \{p, q, r\}. \text{proj2-incident } s \ l \rangle$   
        **have**  $?M *v ?y = 0$   
        **unfolding** *vector-def*  
        **and** *matrix-vector-mult-def*  
        **and** *inner-vec-def*  
        **and** *sum-3*  
        **and** *proj2-incident-def*  
        **by** (*simp add: vec-eq-iff*) }  
    **ultimately show**  $\exists y. y \neq 0 \wedge ?M *v y = 0$  **by** *auto*  
**qed**  
    **finally show**  $\neg \text{invertible } ?M \longleftrightarrow \text{proj2-set-Col } \{p, q, r\}$   
    **unfolding** *proj2-set-Col-def* .  
**qed**

**lemma** *proj2-Col-iff-set-Col*:  
*proj2-Col* *p q r*  $\longleftrightarrow$  *proj2-set-Col*  $\{p, q, r\}$   
**by** (*simp add: proj2-Col-iff-not-invertible*  
*not-invertible-iff-proj2-set-Col*)

**lemma** *proj2-incident-Col*:  
    **assumes** *proj2-incident* *p l* **and** *proj2-incident* *q l* **and** *proj2-incident* *r l*  
    **shows** *proj2-Col* *p q r*  
**proof** –  
    **from**  $\langle \text{proj2-incident } p \ l \rangle$  **and**  $\langle \text{proj2-incident } q \ l \rangle$  **and**  $\langle \text{proj2-incident } r \ l \rangle$   
    **have** *proj2-set-Col*  $\{p, q, r\}$  **by** (*unfold proj2-set-Col-def*) *auto*  
    **thus** *proj2-Col* *p q r* **by** (*subst proj2-Col-iff-set-Col*)

qed

**lemma** *proj2-incident-iff-Col*:

assumes  $p \neq q$  and *proj2-incident*  $p\ l$  and *proj2-incident*  $q\ l$   
shows *proj2-incident*  $r\ l \iff$  *proj2-Col*  $p\ q\ r$

**proof**

assume *proj2-incident*  $r\ l$   
with  $\langle$ *proj2-incident*  $p\ l$  $\rangle$  and  $\langle$ *proj2-incident*  $q\ l$  $\rangle$   
show *proj2-Col*  $p\ q\ r$  by (rule *proj2-incident-Col*)

**next**

assume *proj2-Col*  $p\ q\ r$   
hence *proj2-set-Col*  $\{p,q,r\}$  by (simp add: *proj2-Col-iff-set-Col*)  
then obtain  $m$  where  $\forall s \in \{p,q,r\}. \text{proj2-incident } s\ m$   
unfolding *proj2-set-Col-def* ..  
hence *proj2-incident*  $p\ m$  and *proj2-incident*  $q\ m$  and *proj2-incident*  $r\ m$   
by *simp-all*  
from  $\langle p \neq q \rangle$  and  $\langle$ *proj2-incident*  $p\ l$  $\rangle$  and  $\langle$ *proj2-incident*  $q\ l$  $\rangle$   
and  $\langle$ *proj2-incident*  $p\ m$  $\rangle$  and  $\langle$ *proj2-incident*  $q\ m$  $\rangle$   
and *proj2-incident-unique*  
have  $m = l$  by *auto*  
with  $\langle$ *proj2-incident*  $r\ m$  $\rangle$  show *proj2-incident*  $r\ l$  by *simp*

qed

**lemma** *proj2-incident-iff*:

assumes  $p \neq q$  and *proj2-incident*  $p\ l$  and *proj2-incident*  $q\ l$   
shows *proj2-incident*  $r\ l$   
 $\iff r = p \vee (\exists k. r = \text{proj2-abs } (k *_R \text{proj2-rep } p + \text{proj2-rep } q))$

**proof** –

from  $\langle p \neq q \rangle$  and  $\langle$ *proj2-incident*  $p\ l$  $\rangle$  and  $\langle$ *proj2-incident*  $q\ l$  $\rangle$   
have *proj2-incident*  $r\ l \iff$  *proj2-Col*  $p\ q\ r$  by (rule *proj2-incident-iff-Col*)  
with  $\langle p \neq q \rangle$  and *proj2-Col-iff*  
show *proj2-incident*  $r\ l$   
 $\iff r = p \vee (\exists k. r = \text{proj2-abs } (k *_R \text{proj2-rep } p + \text{proj2-rep } q))$   
by *simp*

qed

**lemma** *not-proj2-set-Col-iff-span*:

assumes  $\text{card } S = 3$   
shows  $\neg \text{proj2-set-Col } S \iff \text{span } (\text{proj2-rep } ` S) = \text{UNIV}$

**proof** –

from  $\langle \text{card } S = 3 \rangle$  and *choose-3* [of  $S$ ]  
obtain  $p$  and  $q$  and  $r$  where  $S = \{p,q,r\}$  by *auto*  
let  $?u = \text{proj2-rep } p$   
let  $?v = \text{proj2-rep } q$   
let  $?w = \text{proj2-rep } r$   
let  $?M = \text{vector } [?u, ?v, ?w] :: \text{real}^{\sim 3}$   
from  $\langle S = \{p,q,r\} \rangle$  and *not-invertible-iff-proj2-set-Col* [of  $p\ q\ r$ ]  
have  $\neg \text{proj2-set-Col } S \iff \text{invertible } ?M$  by *auto*  
also from *left-invertible-iff-invertible*

**have** ...  $\longleftrightarrow (\exists N. N ** ?M = \text{mat } 1) ..$   
**also from** *matrix-left-invertible-span-rows*  
**have** ...  $\longleftrightarrow \text{span } (\text{rows } ?M) = \text{UNIV}$  **by** *auto*  
**finally have**  $\neg \text{proj2-set-Col } S \longleftrightarrow \text{span } (\text{rows } ?M) = \text{UNIV} .$

**have**  $\text{rows } ?M = \{?u, ?v, ?w\}$

**proof**

**{ fix**  $x$   
**assume**  $x \in \text{rows } ?M$   
**then obtain**  $i :: 3$  **where**  $x = ?M \$ i$   
**unfolding** *rows-def* **and** *row-def*  
**by** (*auto simp add: vec-lambda-beta vec-lambda-eta*)  
**with** *exhaust-3* **have**  $x = ?u \vee x = ?v \vee x = ?w$   
**unfolding** *vector-def*  
**by** *auto*  
**hence**  $x \in \{?u, ?v, ?w\}$  **by** *simp* }

**thus**  $\text{rows } ?M \subseteq \{?u, ?v, ?w\} ..$

**{ fix**  $x$   
**assume**  $x \in \{?u, ?v, ?w\}$   
**hence**  $x = ?u \vee x = ?v \vee x = ?w$  **by** *simp*  
**hence**  $x = ?M \$ 1 \vee x = ?M \$ 2 \vee x = ?M \$ 3$   
**unfolding** *vector-def*  
**by** *simp*  
**hence**  $x \in \text{rows } ?M$   
**unfolding** *rows-def row-def vec-lambda-eta*  
**by** *blast* }

**thus**  $\{?u, ?v, ?w\} \subseteq \text{rows } ?M ..$

**qed**

**with**  $\langle S = \{p, q, r\} \rangle$

**have**  $\text{rows } ?M = \text{proj2-rep } ' S$

**unfolding** *image-def*

**by** *auto*

**with**  $\langle \neg \text{proj2-set-Col } S \longleftrightarrow \text{span } (\text{rows } ?M) = \text{UNIV} \rangle$

**show**  $\neg \text{proj2-set-Col } S \longleftrightarrow \text{span } (\text{proj2-rep } ' S) = \text{UNIV}$  **by** *simp*

**qed**

**lemma** *proj2-no-3-Col-span:*

**assumes** *proj2-no-3-Col*  $S$  **and**  $p \in S$

**shows**  $\text{span } (\text{proj2-rep } ' (S - \{p\})) = \text{UNIV}$

**proof** -

**from**  $\langle \text{proj2-no-3-Col } S \rangle$  **have**  $\text{card } S = 4$  **unfolding** *proj2-no-3-Col-def* ..

**with**  $\langle p \in S \rangle$  **and**  $\langle \text{card } S = 4 \rangle$  **and** *card-gt-0-diff-singleton* [*of*  $S$   $p$ ]

**have**  $\text{card } (S - \{p\}) = 3$  **by** *simp*

**from**  $\langle \text{proj2-no-3-Col } S \rangle$  **and**  $\langle p \in S \rangle$

**have**  $\neg \text{proj2-set-Col } (S - \{p\})$

**unfolding** *proj2-no-3-Col-def*

**by** *simp*

**with**  $\langle \text{card } (S - \{p\}) = 3 \rangle$  **and** *not-proj2-set-Col-iff-span*

**show**  $\text{span} (\text{proj2-rep } \langle S - \{p\} \rangle) = \text{UNIV}$  **by** *simp*  
**qed**

**lemma** *fourth-proj2-no-3-Col*:

**assumes**  $\neg \text{proj2-Col } p \ q \ r$

**shows**  $\exists s. \text{proj2-no-3-Col } \{s, r, p, q\}$

**proof** –

**from**  $\langle \neg \text{proj2-Col } p \ q \ r \rangle$  **and** *proj2-Col-coincide* **have**  $p \neq q$  **by** *auto*  
**hence**  $\text{card } \{p, q\} = 2$  **by** *simp*

**from**  $\langle \neg \text{proj2-Col } p \ q \ r \rangle$  **and** *proj2-Col-coincide* **and** *proj2-Col-permute*  
**have**  $r \notin \{p, q\}$  **by** *fast*  
**with**  $\langle \text{card } \{p, q\} = 2 \rangle$  **have**  $\text{card } \{r, p, q\} = 3$  **by** *simp*

**have** *finite*  $\{r, p, q\}$  **by** *simp*

**let**  $?s = \text{proj2-abs } (\sum t \in \{r, p, q\}. \text{proj2-rep } t)$

**have**  $\exists j. (\sum t \in \{r, p, q\}. \text{proj2-rep } t) = j *_{\mathbb{R}} \text{proj2-rep } ?s$

**proof** *cases*

**assume**  $(\sum t \in \{r, p, q\}. \text{proj2-rep } t) = 0$

**hence**  $(\sum t \in \{r, p, q\}. \text{proj2-rep } t) = 0 *_{\mathbb{R}} \text{proj2-rep } ?s$  **by** *simp*

**thus**  $\exists j. (\sum t \in \{r, p, q\}. \text{proj2-rep } t) = j *_{\mathbb{R}} \text{proj2-rep } ?s$  ..

**next**

**assume**  $(\sum t \in \{r, p, q\}. \text{proj2-rep } t) \neq 0$

**with** *proj2-rep-abs2*

**obtain**  $k$  **where**  $k \neq 0$

**and**  $\text{proj2-rep } ?s = k *_{\mathbb{R}} (\sum t \in \{r, p, q\}. \text{proj2-rep } t)$

**by** *auto*

**hence**  $(1/k) *_{\mathbb{R}} \text{proj2-rep } ?s = (\sum t \in \{r, p, q\}. \text{proj2-rep } t)$  **by** *simp*

**from** *this* [*symmetric*]

**show**  $\exists j. (\sum t \in \{r, p, q\}. \text{proj2-rep } t) = j *_{\mathbb{R}} \text{proj2-rep } ?s$  ..

**qed**

**then obtain**  $j$  **where**  $(\sum t \in \{r, p, q\}. \text{proj2-rep } t) = j *_{\mathbb{R}} \text{proj2-rep } ?s$  ..

**let**  $?c = \lambda t. \text{if } t = ?s \text{ then } 1 - j \text{ else } 1$

**from**  $\langle p \neq q \rangle$  **have**  $?c \ p \neq 0 \vee ?c \ q \neq 0$  **by** *simp*

**let**  $?d = \lambda t. \text{if } t = ?s \text{ then } j \text{ else } -1$

**let**  $?S = \{?s, r, p, q\}$

**have**  $?s \notin \{r, p, q\}$

**proof**

**assume**  $?s \in \{r, p, q\}$

**from**  $\langle r \notin \{p, q\} \rangle$  **and**  $\langle p \neq q \rangle$

**have**  $?c \ r *_{\mathbb{R}} \text{proj2-rep } r + ?c \ p *_{\mathbb{R}} \text{proj2-rep } p + ?c \ q *_{\mathbb{R}} \text{proj2-rep } q$

$= (\sum t \in \{r, p, q\}. ?c \ t *_{\mathbb{R}} \text{proj2-rep } t)$

**by** (*simp add: sum.insert [of - -  $\lambda t. ?c \ t *_{\mathbb{R}} \text{proj2-rep } t]$* )

**also from**  $\langle \text{finite } \{r, p, q\} \rangle$  **and**  $\langle ?s \in \{r, p, q\} \rangle$

**have**  $\dots = ?c \ ?s *R \text{proj2-rep } ?s + (\sum t \in \{r,p,q\} - \{?s\}. ?c \ t *R \text{proj2-rep } t)$   
**by** (*simp only*):  
 $\text{sum.remove [of } \{r,p,q\} \ ?s \ \lambda \ t. \ ?c \ t *R \text{proj2-rep } t]$   
**also have**  $\dots$   
 $= -j *R \text{proj2-rep } ?s + (\text{proj2-rep } ?s + (\sum t \in \{r,p,q\} - \{?s\}. \text{proj2-rep } t))$   
**by** (*simp add: algebra-simps*)  
**also from**  $\langle \text{finite } \{r,p,q\} \rangle$  **and**  $\langle ?s \in \{r,p,q\} \rangle$   
**have**  $\dots = -j *R \text{proj2-rep } ?s + (\sum t \in \{r,p,q\}. \text{proj2-rep } t)$   
**by** (*simp only*):  
 $\text{sum.remove [of } \{r,p,q\} \ ?s \ \lambda \ t. \ \text{proj2-rep } t, \text{symmetric}]$   
**also from**  $\langle (\sum t \in \{r,p,q\}. \text{proj2-rep } t) = j *R \text{proj2-rep } ?s \rangle$   
**have**  $\dots = 0$  **by** *simp*  
**finally**  
**have**  $?c \ r *R \text{proj2-rep } r + ?c \ p *R \text{proj2-rep } p + ?c \ q *R \text{proj2-rep } q = 0$   
 $\cdot$   
**with**  $\langle ?c \ p \neq 0 \vee ?c \ q \neq 0 \rangle$   
**have** *proj2-Col p q r*  
**by** (*unfold proj2-Col-def*) (*auto simp add: algebra-simps*)  
**with**  $\langle \neg \text{proj2-Col p q r} \rangle$  **show** *False ..*  
**qed**  
**with**  $\langle \text{card } \{r,p,q\} = 3 \rangle$  **have**  $\text{card } ?S = 4$  **by** *simp*  
  
**from**  $\langle \neg \text{proj2-Col p q r} \rangle$  **and** *proj2-Col-permute*  
**have**  $\neg \text{proj2-Col r p q}$  **by** *fast*  
**hence**  $\neg \text{proj2-set-Col } \{r,p,q\}$  **by** (*subst proj2-Col-iff-set-Col [symmetric]*)  
  
**have**  $\forall u \in ?S. \neg \text{proj2-set-Col } (?S - \{u\})$   
**proof**  
**fix**  $u$   
**assume**  $u \in ?S$   
**with**  $\langle \text{card } ?S = 4 \rangle$  **have**  $\text{card } (?S - \{u\}) = 3$  **by** *simp*  
**show**  $\neg \text{proj2-set-Col } (?S - \{u\})$   
**proof cases**  
**assume**  $u = ?s$   
**with**  $\langle ?s \notin \{r,p,q\} \rangle$  **have**  $?S - \{u\} = \{r,p,q\}$  **by** *simp*  
**with**  $\langle \neg \text{proj2-set-Col } \{r,p,q\} \rangle$  **show**  $\neg \text{proj2-set-Col } (?S - \{u\})$  **by** *simp*  
**next**  
**assume**  $u \neq ?s$   
**hence**  $\text{insert } ?s (\{r,p,q\} - \{u\}) = ?S - \{u\}$  **by** *auto*  
  
**from**  $\langle \text{finite } \{r,p,q\} \rangle$  **have**  $\text{finite } (\{r,p,q\} - \{u\})$  **by** *simp*  
  
**from**  $\langle ?s \notin \{r,p,q\} \rangle$  **have**  $?s \notin \{r,p,q\} - \{u\}$  **by** *simp*  
**hence**  $\forall t \in \{r,p,q\} - \{u\}. ?d \ t = -1$  **by** *auto*  
  
**from**  $\langle u \neq ?s \rangle$  **and**  $\langle u \in ?S \rangle$  **have**  $u \in \{r,p,q\}$  **by** *simp*  
**hence**  $(\sum t \in \{r,p,q\}. \text{proj2-rep } t)$   
 $= \text{proj2-rep } u + (\sum t \in \{r,p,q\} - \{u\}. \text{proj2-rep } t)$   
**by** (*simp add: sum.remove*)

**with**  $\langle \sum_{t \in \{r,p,q\}} \text{proj2-rep } t \rangle = j *_R \text{proj2-rep } ?s$   
**have**  $\text{proj2-rep } u$   
 $= j *_R \text{proj2-rep } ?s - (\sum_{t \in \{r,p,q\} - \{u\}} \text{proj2-rep } t)$   
**by** *simp*  
**also from**  $\langle \forall t \in \{r,p,q\} - \{u\}. ?d t = -1 \rangle$   
**have**  $\dots = j *_R \text{proj2-rep } ?s + (\sum_{t \in \{r,p,q\} - \{u\}} ?d t *_R \text{proj2-rep } t)$   
**by** (*simp add: sum-negf*)  
**also from**  $\langle \text{finite } (\{r,p,q\} - \{u\}) \rangle$  **and**  $\langle ?s \notin \{r,p,q\} - \{u\} \rangle$   
**have**  $\dots = (\sum_{t \in \text{insert } ?s (\{r,p,q\} - \{u\})} ?d t *_R \text{proj2-rep } t)$   
**by** (*simp add: sum.insert*)  
**also from**  $\langle \text{insert } ?s (\{r,p,q\} - \{u\}) = ?S - \{u\} \rangle$   
**have**  $\dots = (\sum_{t \in ?S - \{u\}} ?d t *_R \text{proj2-rep } t)$  **by** *simp*  
**finally have**  $\text{proj2-rep } u = (\sum_{t \in ?S - \{u\}} ?d t *_R \text{proj2-rep } t)$  .  
**moreover**  
**have**  $\forall t \in ?S - \{u\}. ?d t *_R \text{proj2-rep } t \in \text{span } (\text{proj2-rep } ' (?S - \{u\}))$   
**by** (*simp add: span-clauses*)  
**ultimately have**  $\text{proj2-rep } u \in \text{span } (\text{proj2-rep } ' (?S - \{u\}))$   
**by** (*metis (no-types, lifting) span-sum*)

**have**  $\forall t \in \{r,p,q\}. \text{proj2-rep } t \in \text{span } (\text{proj2-rep } ' (?S - \{u\}))$   
**proof**  
**fix**  $t$   
**assume**  $t \in \{r,p,q\}$   
**show**  $\text{proj2-rep } t \in \text{span } (\text{proj2-rep } ' (?S - \{u\}))$   
**proof cases**  
**assume**  $t = u$   
**from**  $\langle \text{proj2-rep } u \in \text{span } (\text{image } \text{proj2-rep } (?S - \{u\})) \rangle$   
**show**  $\text{proj2-rep } t \in \text{span } (\text{proj2-rep } ' (?S - \{u\}))$   
**by** (*subst <t = u>*)  
**next**  
**assume**  $t \neq u$   
**with**  $\langle t \in \{r,p,q\} \rangle$   
**have**  $\text{proj2-rep } t \in \text{proj2-rep } ' (?S - \{u\})$  **by** *simp*  
**with** *span-superset [of proj2-rep ' (?S - {u})]*  
**show**  $\text{proj2-rep } t \in \text{span } (\text{proj2-rep } ' (?S - \{u\}))$  **by** *fast*  
**qed**  
**qed**  
**hence**  $\text{proj2-rep } ' \{r,p,q\} \subseteq \text{span } (\text{proj2-rep } ' (?S - \{u\}))$   
**by** (*simp only: image-subset-iff*)  
**hence**  
 $\text{span } (\text{proj2-rep } ' \{r,p,q\}) \subseteq \text{span } (\text{span } (\text{proj2-rep } ' (?S - \{u\})))$   
**by** (*simp only: span-mono*)  
**hence**  $\text{span } (\text{proj2-rep } ' \{r,p,q\}) \subseteq \text{span } (\text{proj2-rep } ' (?S - \{u\}))$   
**by** (*simp only: span-span*)  
**moreover**  
**from**  $\langle \neg \text{proj2-set-Col } \{r,p,q\} \rangle$   
**and**  $\langle \text{card } \{r,p,q\} = 3 \rangle$   
**and** *not-proj2-set-Col-iff-span*  
**have**  $\text{span } (\text{proj2-rep } ' \{r,p,q\}) = \text{UNIV}$  **by** *simp*

**ultimately have**  $\text{span } (\text{proj2-rep } (\{S - \{u\}\})) = \text{UNIV}$  **by** *auto*  
**with**  $\langle \text{card } (\{S - \{u\}\}) = 3 \rangle$  **and** *not-proj2-set-Col-iff-span*  
**show**  $\neg \text{proj2-set-Col } (\{S - \{u\}\})$  **by** *simp*  
**qed**  
**qed**  
**with**  $\langle \text{card } ?S = 4 \rangle$   
**have** *proj2-no-3-Col ?S* **by** (*unfold proj2-no-3-Col-def*) *fast*  
**thus**  $\exists s. \text{proj2-no-3-Col } \{s, r, p, q\}$  **..**  
**qed**

**lemma** *proj2-set-Col-expand*:

**assumes** *proj2-set-Col S* **and**  $\{p, q, r\} \subseteq S$  **and**  $p \neq q$  **and**  $r \neq p$   
**shows**  $\exists k. r = \text{proj2-abs } (k *_R \text{proj2-rep } p + \text{proj2-rep } q)$   
**proof** –  
**from**  $\langle \text{proj2-set-Col } S \rangle$   
**obtain**  $l$  **where**  $\forall t \in S. \text{proj2-incident } t \ l$  **unfolding** *proj2-set-Col-def* **..**  
**with**  $\langle \{p, q, r\} \subseteq S \rangle$  **and**  $\langle p \neq q \rangle$  **and**  $\langle r \neq p \rangle$  **and** *proj2-incident-iff* [*of p q l r*]  
**show**  $\exists k. r = \text{proj2-abs } (k *_R \text{proj2-rep } p + \text{proj2-rep } q)$  **by** *simp*  
**qed**

## 7.4 Collineations of the real projective plane

**typedef** *cltn2* =

(*Collect invertible* ::  $(\text{real}^{\mathbb{3}\mathbb{3}})$  *set*)//*invertible-proportionality*

**proof**

**from** *matrix-id-invertible* **have**  $(\text{mat } 1 :: \text{real}^{\mathbb{3}\mathbb{3}}) \in \text{Collect } \text{invertible}$   
**by** *simp*

**thus** *invertible-proportionality* “  $\{\text{mat } 1\} \in$   
(*Collect invertible* ::  $(\text{real}^{\mathbb{3}\mathbb{3}})$  *set*)//*invertible-proportionality*  
unfolding *quotient-def*

**by** *auto*

**qed**

**definition** *cltn2-rep* :: *cltn2*  $\Rightarrow$   $\text{real}^{\mathbb{3}\mathbb{3}}$  **where**

*cltn2-rep*  $A \triangleq \epsilon \ B. B \in \text{Rep-cltn2 } A$

**definition** *cltn2-abs* ::  $\text{real}^{\mathbb{3}\mathbb{3}}$   $\Rightarrow$  *cltn2* **where**

*cltn2-abs*  $B \triangleq \text{Abs-cltn2 } (\text{invertible-proportionality } \{B\})$

**definition** *cltn2-independent* :: *cltn2 set*  $\Rightarrow$  *bool* **where**

*cltn2-independent*  $X \triangleq \text{independent } \{\text{cltn2-rep } A \mid A. A \in X\}$

**definition** *apply-cltn2* :: *proj2*  $\Rightarrow$  *cltn2*  $\Rightarrow$  *proj2* **where**

*apply-cltn2*  $x \ A \triangleq \text{proj2-abs } (\text{proj2-rep } x \ v * \text{cltn2-rep } A)$

**lemma** *cltn2-rep-in*: *cltn2-rep*  $B \in \text{Rep-cltn2 } B$

**proof** –

**let**  $?A = \text{cltn2-rep } B$

**from** *quotient-element-nonempty* **and**

*invertible-proportionality-equiv* **and**  
*Rep-cltn2* [of *B*]  
**have**  $\exists C. C \in \text{Rep-cltn2 } B$   
**by** *auto*  
**with** *someI-ex* [of  $\lambda C. C \in \text{Rep-cltn2 } B$ ]  
**show**  $?A \in \text{Rep-cltn2 } B$   
**unfolding** *cltn2-rep-def*  
**by** *simp*  
**qed**

**lemma** *cltn2-rep-invertible*: *invertible* (*cltn2-rep* *A*)  
**proof** –  
**from**  
*Union-quotient* [of *Collect invertible invertible-proportionality*]  
**and** *invertible-proportionality-equiv*  
**and** *Rep-cltn2* [of *A*] **and** *cltn2-rep-in* [of *A*]  
**have** *cltn2-rep* *A*  $\in$  *Collect invertible*  
**unfolding** *quotient-def*  
**by** *auto*  
**thus** *invertible* (*cltn2-rep* *A*)  
**unfolding** *invertible-proportionality-def*  
**by** *simp*  
**qed**

**lemma** *cltn2-rep-abs*:  
**fixes** *A* ::  $\text{real}^{\mathbb{3}}^{\mathbb{3}}$   
**assumes** *invertible* *A*  
**shows**  $(A, \text{cltn2-rep } (\text{cltn2-abs } A)) \in \text{invertible-proportionality}$   
**proof** –  
**from**  $\langle \text{invertible } A \rangle$   
**have** *invertible-proportionality* “ $\{A\} \in (\text{Collect invertible} :: (\text{real}^{\mathbb{3}}^{\mathbb{3}}) \text{ set}) // \text{invertible-proportionality}$ ”  
**unfolding** *quotient-def*  
**by** *auto*  
**with** *Abs-cltn2-inverse*  
**have** *Rep-cltn2* (*cltn2-abs* *A*) = *invertible-proportionality* “ $\{A\}$ ”  
**unfolding** *cltn2-abs-def*  
**by** *simp*  
**with** *cltn2-rep-in*  
**have** *cltn2-rep* (*cltn2-abs* *A*)  $\in$  *invertible-proportionality* “ $\{A\}$ ” **by** *auto*  
**thus**  $(A, \text{cltn2-rep } (\text{cltn2-abs } A)) \in \text{invertible-proportionality}$  **by** *simp*  
**qed**

**lemma** *cltn2-rep-abs2*:  
**assumes** *invertible* *A*  
**shows**  $\exists k. k \neq 0 \wedge \text{cltn2-rep } (\text{cltn2-abs } A) = k *_R A$   
**proof** –  
**from**  $\langle \text{invertible } A \rangle$  **and** *cltn2-rep-abs*  
**have**  $(A, \text{cltn2-rep } (\text{cltn2-abs } A)) \in \text{invertible-proportionality}$  **by** *simp*  
**then obtain** *c* **where**  $A = c *_R \text{cltn2-rep } (\text{cltn2-abs } A)$

**unfolding** *invertible-proportionality-def* **and** *real-vector.proportionality-def*  
**by** *auto*  
**with**  $\langle \text{invertible } A \rangle$  **and** *zero-not-invertible* **have**  $c \neq 0$  **by** *auto*  
**hence**  $1/c \neq 0$  **by** *simp*  
  
**let**  $?k = 1/c$   
**from**  $\langle A = c *_{\mathbb{R}} \text{cltn2-rep } (\text{cltn2-abs } A) \rangle$   
**have**  $?k *_{\mathbb{R}} A = ?k *_{\mathbb{R}} c *_{\mathbb{R}} \text{cltn2-rep } (\text{cltn2-abs } A)$  **by** *simp*  
**with**  $\langle c \neq 0 \rangle$  **have**  $\text{cltn2-rep } (\text{cltn2-abs } A) = ?k *_{\mathbb{R}} A$  **by** *simp*  
**with**  $\langle ?k \neq 0 \rangle$   
**show**  $\exists k. k \neq 0 \wedge \text{cltn2-rep } (\text{cltn2-abs } A) = k *_{\mathbb{R}} A$  **by** *blast*  
**qed**

**lemma** *cltn2-abs-rep*:  $\text{cltn2-abs } (\text{cltn2-rep } A) = A$

**proof** –

**from** *partition-Image-element*  
[*of Collect invertible*  
*invertible-proportionality*  
*Rep-cltn2 A*  
*cltn2-rep A*]  
**and** *invertible-proportionality-equiv*  
**and** *Rep-cltn2 [of A]* **and** *cltn2-rep-in [of A]*  
**have** *invertible-proportionality* “  $\{\text{cltn2-rep } A\} = \text{Rep-cltn2 } A$   
**by** *simp*  
**with** *Rep-cltn2-inverse*  
**show**  $\text{cltn2-abs } (\text{cltn2-rep } A) = A$   
**unfolding** *cltn2-abs-def*  
**by** *simp*  
**qed**

**lemma** *cltn2-abs-mult*:

**assumes**  $k \neq 0$  **and** *invertible A*  
**shows**  $\text{cltn2-abs } (k *_{\mathbb{R}} A) = \text{cltn2-abs } A$   
**proof** –  
**from**  $\langle k \neq 0 \rangle$  **and**  $\langle \text{invertible } A \rangle$  **and** *scalar-invertible*  
**have** *invertible*  $(k *_{\mathbb{R}} A)$  **by** *auto*  
**with**  $\langle \text{invertible } A \rangle$   
**have**  $(k *_{\mathbb{R}} A, A) \in \text{invertible-proportionality}$   
**unfolding** *invertible-proportionality-def*  
**and** *real-vector.proportionality-def*  
**by** *(auto simp add: zero-not-invertible)*  
**with** *eq-equiv-class-iff*  
[*of Collect invertible invertible-proportionality k \*\_{\mathbb{R}} A A*  
**and** *invertible-proportionality-equiv*  
**and**  $\langle \text{invertible } A \rangle$  **and**  $\langle \text{invertible } (k *_{\mathbb{R}} A) \rangle$   
**have** *invertible-proportionality* “  $\{k *_{\mathbb{R}} A\}$   
= *invertible-proportionality* “  $\{A\}$   
**by** *simp*  
**thus**  $\text{cltn2-abs } (k *_{\mathbb{R}} A) = \text{cltn2-abs } A$

**unfolding** *cltn2-abs-def*  
**by** *simp*  
**qed**

**lemma** *cltn2-abs-mult-rep*:  
**assumes**  $k \neq 0$   
**shows**  $\text{cltn2-abs } (k *_R \text{cltn2-rep } A) = A$   
**using** *cltn2-rep-invertible* **and** *cltn2-abs-mult* **and** *cltn2-abs-rep* **and** *assms*  
**by** *simp*

**lemma** *apply-cltn2-abs*:  
**assumes**  $x \neq 0$  **and** *invertible A*  
**shows**  $\text{apply-cltn2 } (\text{proj2-abs } x) (\text{cltn2-abs } A) = \text{proj2-abs } (x v * A)$   
**proof** –  
**from** *proj2-rep-abs2* **and**  $\langle x \neq 0 \rangle$   
**obtain**  $k$  **where**  $k \neq 0$  **and**  $\text{proj2-rep } (\text{proj2-abs } x) = k *_R x$  **by** *auto*  
  
**from** *cltn2-rep-abs2* **and**  $\langle \text{invertible } A \rangle$   
**obtain**  $c$  **where**  $c \neq 0$  **and**  $\text{cltn2-rep } (\text{cltn2-abs } A) = c *_R A$  **by** *auto*  
  
**from**  $\langle k \neq 0 \rangle$  **and**  $\langle c \neq 0 \rangle$  **have**  $k * c \neq 0$  **by** *simp*

**from**  $\langle \text{proj2-rep } (\text{proj2-abs } x) = k *_R x \rangle$  **and**  $\langle \text{cltn2-rep } (\text{cltn2-abs } A) = c *_R A \rangle$   
**have**  $\text{proj2-rep } (\text{proj2-abs } x) v * \text{cltn2-rep } (\text{cltn2-abs } A) = (k * c) *_R (x v * A)$   
**by** (*simp add: scaleR-vector-matrix-assoc vector-scaleR-matrix-ac*)  
**with**  $\langle k * c \neq 0 \rangle$   
**show**  $\text{apply-cltn2 } (\text{proj2-abs } x) (\text{cltn2-abs } A) = \text{proj2-abs } (x v * A)$   
**unfolding** *apply-cltn2-def*  
**by** (*simp add: proj2-abs-mult*)  
**qed**

**lemma** *apply-cltn2-left-abs*:  
**assumes**  $v \neq 0$   
**shows**  $\text{apply-cltn2 } (\text{proj2-abs } v) C = \text{proj2-abs } (v v * \text{cltn2-rep } C)$   
**proof** –  
**have**  $\text{cltn2-abs } (\text{cltn2-rep } C) = C$  **by** (*rule cltn2-abs-rep*)  
**with**  $\langle v \neq 0 \rangle$  **and** *cltn2-rep-invertible* **and** *apply-cltn2-abs* [*of v cltn2-rep C*]  
**show**  $\text{apply-cltn2 } (\text{proj2-abs } v) C = \text{proj2-abs } (v v * \text{cltn2-rep } C)$   
**by** *simp*  
**qed**

**lemma** *apply-cltn2-right-abs*:  
**assumes** *invertible M*  
**shows**  $\text{apply-cltn2 } p (\text{cltn2-abs } M) = \text{proj2-abs } (\text{proj2-rep } p v * M)$   
**proof** –  
**from** *proj2-rep-non-zero* **and**  $\langle \text{invertible } M \rangle$  **and** *apply-cltn2-abs*  
**have**  $\text{apply-cltn2 } (\text{proj2-abs } (\text{proj2-rep } p)) (\text{cltn2-abs } M)$   
 $= \text{proj2-abs } (\text{proj2-rep } p v * M)$   
**by** *simp*

**thus**  $\text{apply-cltn2 } p \text{ (cltn2-abs } M) = \text{proj2-abs (proj2-rep } p \text{ } v * M)$   
**by** (*simp add: proj2-abs-rep*)  
**qed**

**lemma** *non-zero-mult-rep-non-zero*:  
**assumes**  $v \neq 0$   
**shows**  $v * \text{cltn2-rep } C \neq 0$   
**using**  $\langle v \neq 0 \rangle$  **and** *cltn2-rep-invertible* **and** *times-invertible-eq-zero*  
**by** *auto*

**lemma** *rep-mult-rep-non-zero*:  $\text{proj2-rep } p \text{ } v * \text{cltn2-rep } A \neq 0$   
**using** *proj2-rep-non-zero*  
**by** (*rule non-zero-mult-rep-non-zero*)

**definition** *cltn2-image* ::  $\text{proj2 set} \Rightarrow \text{cltn2} \Rightarrow \text{proj2 set}$  **where**  
 $\text{cltn2-image } P \ A \triangleq \{\text{apply-cltn2 } p \ A \mid p. p \in P\}$

#### 7.4.1 As a group

**definition** *cltn2-id* ::  $\text{cltn2}$  **where**  
 $\text{cltn2-id} \triangleq \text{cltn2-abs (mat 1)}$

**definition** *cltn2-compose* ::  $\text{cltn2} \Rightarrow \text{cltn2} \Rightarrow \text{cltn2}$  **where**  
 $\text{cltn2-compose } A \ B \triangleq \text{cltn2-abs (cltn2-rep } A \ ** \ \text{cltn2-rep } B)$

**definition** *cltn2-inverse* ::  $\text{cltn2} \Rightarrow \text{cltn2}$  **where**  
 $\text{cltn2-inverse } A \triangleq \text{cltn2-abs (matrix-inv (cltn2-rep } A))$

**lemma** *cltn2-compose-abs*:  
**assumes** *invertible*  $M$  **and** *invertible*  $N$   
**shows**  $\text{cltn2-compose (cltn2-abs } M) \text{ (cltn2-abs } N) = \text{cltn2-abs (} M \ ** \ N)$

**proof** –

**from**  $\langle \text{invertible } M \rangle$  **and**  $\langle \text{invertible } N \rangle$  **and** *invertible-mult*  
**have** *invertible*  $(M \ ** \ N)$  **by** *auto*

**from**  $\langle \text{invertible } M \rangle$  **and**  $\langle \text{invertible } N \rangle$  **and** *cltn2-rep-abs2*  
**obtain**  $j$  **and**  $k$  **where**  $j \neq 0$  **and**  $k \neq 0$   
**and**  $\text{cltn2-rep (cltn2-abs } M) = j *_{\mathbb{R}} M$   
**and**  $\text{cltn2-rep (cltn2-abs } N) = k *_{\mathbb{R}} N$   
**by** *blast*

**from**  $\langle j \neq 0 \rangle$  **and**  $\langle k \neq 0 \rangle$  **have**  $j * k \neq 0$  **by** *simp*

**from**  $\langle \text{cltn2-rep (cltn2-abs } M) = j *_{\mathbb{R}} M \rangle$  **and**  $\langle \text{cltn2-rep (cltn2-abs } N) = k *_{\mathbb{R}} N \rangle$

**have**  $\text{cltn2-rep (cltn2-abs } M) \ ** \ \text{cltn2-rep (cltn2-abs } N)$   
 $= (j * k) *_{\mathbb{R}} (M \ ** \ N)$   
**by** (*simp add: matrix-scalar-ac scalar-matrix-assoc [symmetric]*)  
**with**  $\langle j * k \neq 0 \rangle$  **and**  $\langle \text{invertible (} M \ ** \ N) \rangle$

**show**  $\text{cltn2-compose} (\text{cltn2-abs } M) (\text{cltn2-abs } N) = \text{cltn2-abs} (M ** N)$   
**unfolding**  $\text{cltn2-compose-def}$   
**by** (*simp add: cltn2-abs-mult*)  
**qed**

**lemma**  $\text{cltn2-compose-left-abs}$ :  
**assumes**  $\text{invertible } M$   
**shows**  $\text{cltn2-compose} (\text{cltn2-abs } M) A = \text{cltn2-abs} (M ** \text{cltn2-rep } A)$   
**proof** –  
**from**  $\langle \text{invertible } M \rangle$  **and**  $\text{cltn2-rep-invertible}$  **and**  $\text{cltn2-compose-abs}$   
**have**  $\text{cltn2-compose} (\text{cltn2-abs } M) (\text{cltn2-abs} (\text{cltn2-rep } A))$   
 $= \text{cltn2-abs} (M ** \text{cltn2-rep } A)$   
**by** *simp*  
**thus**  $\text{cltn2-compose} (\text{cltn2-abs } M) A = \text{cltn2-abs} (M ** \text{cltn2-rep } A)$   
**by** (*simp add: cltn2-abs-rep*)  
**qed**

**lemma**  $\text{cltn2-compose-right-abs}$ :  
**assumes**  $\text{invertible } M$   
**shows**  $\text{cltn2-compose } A (\text{cltn2-abs } M) = \text{cltn2-abs} (\text{cltn2-rep } A ** M)$   
**proof** –  
**from**  $\langle \text{invertible } M \rangle$  **and**  $\text{cltn2-rep-invertible}$  **and**  $\text{cltn2-compose-abs}$   
**have**  $\text{cltn2-compose} (\text{cltn2-abs} (\text{cltn2-rep } A)) (\text{cltn2-abs } M)$   
 $= \text{cltn2-abs} (\text{cltn2-rep } A ** M)$   
**by** *simp*  
**thus**  $\text{cltn2-compose } A (\text{cltn2-abs } M) = \text{cltn2-abs} (\text{cltn2-rep } A ** M)$   
**by** (*simp add: cltn2-abs-rep*)  
**qed**

**lemma**  $\text{cltn2-abs-rep-abs-mult}$ :  
**assumes**  $\text{invertible } M$  **and**  $\text{invertible } N$   
**shows**  $\text{cltn2-abs} (\text{cltn2-rep} (\text{cltn2-abs } M) ** N) = \text{cltn2-abs} (M ** N)$   
**proof** –  
**from**  $\langle \text{invertible } M \rangle$  **and**  $\langle \text{invertible } N \rangle$   
**have**  $\text{invertible} (M ** N)$  **by** (*simp add: invertible-mult*)  
  
**from**  $\langle \text{invertible } M \rangle$  **and**  $\text{cltn2-rep-abs2}$   
**obtain**  $k$  **where**  $k \neq 0$  **and**  $\text{cltn2-rep} (\text{cltn2-abs } M) = k *_R M$  **by** *auto*  
**from**  $\langle \text{cltn2-rep} (\text{cltn2-abs } M) = k *_R M \rangle$   
**have**  $\text{cltn2-rep} (\text{cltn2-abs } M) ** N = k *_R M ** N$  **by** *simp*  
**with**  $\langle k \neq 0 \rangle$  **and**  $\langle \text{invertible} (M ** N) \rangle$  **and**  $\text{cltn2-abs-mult}$   
**show**  $\text{cltn2-abs} (\text{cltn2-rep} (\text{cltn2-abs } M) ** N) = \text{cltn2-abs} (M ** N)$   
**by** (*simp add: scalar-matrix-assoc [symmetric]*)  
**qed**

**lemma**  $\text{cltn2-assoc}$ :  
 $\text{cltn2-compose} (\text{cltn2-compose } A B) C = \text{cltn2-compose } A (\text{cltn2-compose } B C)$   
**proof** –  
**let**  $?A' = \text{cltn2-rep } A$

```

let ?B' = cltn2-rep B
let ?C' = cltn2-rep C
from cltn2-rep-invertible
have invertible ?A' and invertible ?B' and invertible ?C' by simp-all
with invertible-mult
have invertible (?A' ** ?B') and invertible (?B' ** ?C')
  and invertible (?A' ** ?B' ** ?C')
  by auto
from ⟨invertible (?A' ** ?B')⟩ and ⟨invertible ?C'⟩ and cltn2-abs-rep-abs-mult
have cltn2-abs (cltn2-rep (cltn2-abs (?A' ** ?B')) ** ?C')
  = cltn2-abs (?A' ** ?B' ** ?C')
  by simp

from ⟨invertible (?B' ** ?C')⟩ and cltn2-rep-abs2 [of ?B' ** ?C']
obtain k where k ≠ 0
  and cltn2-rep (cltn2-abs (?B' ** ?C')) = k *R (?B' ** ?C')
  by auto
from ⟨cltn2-rep (cltn2-abs (?B' ** ?C')) = k *R (?B' ** ?C')⟩
have ?A' ** cltn2-rep (cltn2-abs (?B' ** ?C')) = k *R (?A' ** ?B' ** ?C')
  by (simp add: matrix-scalar-ac matrix-mul-assoc scalar-matrix-assoc)
with ⟨k ≠ 0⟩ and ⟨invertible (?A' ** ?B' ** ?C')⟩
  and cltn2-abs-mult [of k ?A' ** ?B' ** ?C']
have cltn2-abs (?A' ** cltn2-rep (cltn2-abs (?B' ** ?C')))
  = cltn2-abs (?A' ** ?B' ** ?C')
  by simp
with ⟨cltn2-abs (cltn2-rep (cltn2-abs (?A' ** ?B')) ** ?C')
  = cltn2-abs (?A' ** ?B' ** ?C')⟩
show
  cltn2-compose (cltn2-compose A B) C = cltn2-compose A (cltn2-compose B C)
  unfolding cltn2-compose-def
  by simp
qed

```

```

lemma cltn2-left-id: cltn2-compose cltn2-id A = A
proof –
  let ?A' = cltn2-rep A
  from cltn2-rep-invertible have invertible ?A' by simp
  with matrix-id-invertible and cltn2-abs-rep-abs-mult [of mat 1 ?A']
  have cltn2-compose cltn2-id A = cltn2-abs (cltn2-rep A)
  unfolding cltn2-compose-def and cltn2-id-def
  by (auto simp add: matrix-mul-lid)
  with cltn2-abs-rep show cltn2-compose cltn2-id A = A by simp
qed

```

```

lemma cltn2-left-inverse: cltn2-compose (cltn2-inverse A) A = cltn2-id
proof –
  let ?M = cltn2-rep A
  let ?M' = matrix-inv ?M
  from cltn2-rep-invertible have invertible ?M by simp

```

**with** *matrix-inv-invertible* **have** *invertible ?M'* **by** *auto*  
**with**  $\langle$ *invertible ?M* $\rangle$  **and** *cltn2-abs-rep-abs-mult*  
**have** *cltn2-compose (cltn2-inverse A) A = cltn2-abs (?M' \*\* ?M)*  
**unfolding** *cltn2-compose-def* **and** *cltn2-inverse-def*  
**by** *simp*  
**with**  $\langle$ *invertible ?M* $\rangle$   
**show** *cltn2-compose (cltn2-inverse A) A = cltn2-id*  
**unfolding** *cltn2-id-def*  
**by** (*simp add: matrix-inv*)  
**qed**

**lemma** *cltn2-left-inverse-ex*:  
 $\exists B. \text{cltn2-compose } B \ A = \text{cltn2-id}$   
**using** *cltn2-left-inverse ..*

**interpretation** *cltn2*:  
*group (|carrier = UNIV, mult = cltn2-compose, one = cltn2-id|)*  
**using** *cltn2-assoc* **and** *cltn2-left-id* **and** *cltn2-left-inverse-ex*  
**and** *groupI [of (|carrier = UNIV, mult = cltn2-compose, one = cltn2-id|)]*  
**by** *simp-all*

**lemma** *cltn2-inverse-inv [simp]*:  
 $\text{inv}(|\text{carrier} = \text{UNIV}, \text{mult} = \text{cltn2-compose}, \text{one} = \text{cltn2-id}|) \ A$   
 $= \text{cltn2-inverse } A$   
**using** *cltn2-left-inverse [of A]* **and** *cltn2.inv-equality*  
**by** *simp*

**lemmas** *cltn2-inverse-id [simp] = cltn2.inv-one [simplified]*  
**and** *cltn2-inverse-compose = cltn2.inv-mult-group [simplified]*

## 7.4.2 As a group action

**lemma** *apply-cltn2-id [simp]*:  $\text{apply-cltn2 } p \ \text{cltn2-id} = p$   
**proof** –  
**from** *matrix-id-invertible* **and** *apply-cltn2-right-abs*  
**have**  $\text{apply-cltn2 } p \ \text{cltn2-id} = \text{proj2-abs } (\text{proj2-rep } p \ v * \text{mat } 1)$   
**unfolding** *cltn2-id-def* **by** *blast*  
**thus**  $\text{apply-cltn2 } p \ \text{cltn2-id} = p$   
**by** (*simp add: proj2-abs-rep*)  
**qed**

**lemma** *apply-cltn2-compose*:  
 $\text{apply-cltn2 } (\text{apply-cltn2 } p \ A) \ B = \text{apply-cltn2 } p \ (\text{cltn2-compose } A \ B)$   
**proof** –  
**from** *rep-mult-rep-non-zero* **and** *cltn2-rep-invertible* **and** *apply-cltn2-abs*  
**have**  $\text{apply-cltn2 } (\text{apply-cltn2 } p \ A) \ (\text{cltn2-abs } (\text{cltn2-rep } B))$   
 $= \text{proj2-abs } ((\text{proj2-rep } p \ v * \text{cltn2-rep } A) \ v * \text{cltn2-rep } B)$   
**unfolding** *apply-cltn2-def [of p A]*  
**by** *simp*

**hence**  $\text{apply-cltn2} (\text{apply-cltn2 } p \ A) \ B$   
 $= \text{proj2-abs} (\text{proj2-rep } p \ v * (\text{cltn2-rep } A \ ** \ \text{cltn2-rep } B))$   
**by** (*simp add: cltn2-abs-rep vector-matrix-mul-assoc*)

**from** *cltn2-rep-invertible* **and** *invertible-mult*  
**have** *invertible (cltn2-rep A \*\* cltn2-rep B)* **by** *auto*  
**with** *apply-cltn2-right-abs*  
**have**  $\text{apply-cltn2 } p \ (\text{cltn2-compose } A \ B)$   
 $= \text{proj2-abs} (\text{proj2-rep } p \ v * (\text{cltn2-rep } A \ ** \ \text{cltn2-rep } B))$   
**unfolding** *cltn2-compose-def*  
**by** *simp*  
**with**  $\langle \text{apply-cltn2} (\text{apply-cltn2 } p \ A) \ B$   
 $= \text{proj2-abs} (\text{proj2-rep } p \ v * (\text{cltn2-rep } A \ ** \ \text{cltn2-rep } B)) \rangle$   
**show**  $\text{apply-cltn2} (\text{apply-cltn2 } p \ A) \ B = \text{apply-cltn2 } p \ (\text{cltn2-compose } A \ B)$   
**by** *simp*  
**qed**

**interpretation** *cltn2*:

*action* ( $| \text{carrier} = \text{UNIV}, \text{mult} = \text{cltn2-compose}, \text{one} = \text{cltn2-id} |$ ) *apply-cltn2*

**proof**

**let**  $?G = (| \text{carrier} = \text{UNIV}, \text{mult} = \text{cltn2-compose}, \text{one} = \text{cltn2-id} |)$

**fix**  $p$

**show**  $\text{apply-cltn2 } p \ \mathbf{1}_{?G} = p$  **by** *simp*

**fix**  $A \ B$

**have**  $\text{apply-cltn2} (\text{apply-cltn2 } p \ A) \ B = \text{apply-cltn2 } p \ (A \otimes_{?G} B)$

**by** *simp (rule apply-cltn2-compose)*

**thus**  $A \in \text{carrier } ?G \wedge B \in \text{carrier } ?G$

$\longrightarrow \text{apply-cltn2} (\text{apply-cltn2 } p \ A) \ B = \text{apply-cltn2 } p \ (A \otimes_{?G} B)$

..

**qed**

**definition** *cltn2-transpose* :: *cltn2*  $\Rightarrow$  *cltn2* **where**

$\text{cltn2-transpose } A \triangleq \text{cltn2-abs} (\text{transpose} (\text{cltn2-rep } A))$

**definition** *apply-cltn2-line* :: *proj2-line*  $\Rightarrow$  *cltn2*  $\Rightarrow$  *proj2-line* **where**

*apply-cltn2-line*  $l \ A$

$\triangleq P2L (\text{apply-cltn2} (L2P \ l) (\text{cltn2-transpose} (\text{cltn2-inverse } A)))$

**lemma** *cltn2-transpose-abs*:

**assumes** *invertible*  $M$

**shows**  $\text{cltn2-transpose} (\text{cltn2-abs } M) = \text{cltn2-abs} (\text{transpose } M)$

**proof** –

**from**  $\langle \text{invertible } M \rangle$  **and** *transpose-invertible* **have** *invertible (transpose M)* **by** *auto*

**from**  $\langle \text{invertible } M \rangle$  **and** *cltn2-rep-abs2*

**obtain**  $k$  **where**  $k \neq 0$  **and**  $\text{cltn2-rep} (\text{cltn2-abs } M) = k *_{\mathbb{R}} M$  **by** *auto*

**from**  $\langle \text{cltn2-rep} (\text{cltn2-abs } M) = k *_{\mathbb{R}} M \rangle$

**have**  $\text{transpose} (\text{cltn2-rep} (\text{cltn2-abs } M)) = k *_R \text{transpose } M$   
**by** (*simp add: transpose-scalar*)  
**with**  $\langle k \neq 0 \rangle$  **and**  $\langle \text{invertible} (\text{transpose } M) \rangle$   
**show**  $\text{cltn2-transpose} (\text{cltn2-abs } M) = \text{cltn2-abs} (\text{transpose } M)$   
**unfolding** *cltn2-transpose-def*  
**by** (*simp add: cltn2-abs-mult*)  
**qed**

**lemma** *cltn2-transpose-compose:*

$\text{cltn2-transpose} (\text{cltn2-compose } A B)$   
 $= \text{cltn2-compose} (\text{cltn2-transpose } B) (\text{cltn2-transpose } A)$

**proof** –

**from** *cltn2-rep-invertible*  
**have**  $\text{invertible} (\text{cltn2-rep } A)$  **and**  $\text{invertible} (\text{cltn2-rep } B)$   
**by** *simp-all*  
**with** *transpose-invertible*  
**have**  $\text{invertible} (\text{transpose} (\text{cltn2-rep } A))$   
**and**  $\text{invertible} (\text{transpose} (\text{cltn2-rep } B))$   
**by** *auto*

**from**  $\langle \text{invertible} (\text{cltn2-rep } A) \rangle$  **and**  $\langle \text{invertible} (\text{cltn2-rep } B) \rangle$   
**and** *invertible-mult*

**have**  $\text{invertible} (\text{cltn2-rep } A ** \text{cltn2-rep } B)$  **by** *auto*  
**with**  $\langle \text{invertible} (\text{cltn2-rep } A ** \text{cltn2-rep } B) \rangle$  **and** *cltn2-transpose-abs*  
**have**  $\text{cltn2-transpose} (\text{cltn2-compose } A B)$   
 $= \text{cltn2-abs} (\text{transpose} (\text{cltn2-rep } A ** \text{cltn2-rep } B))$   
**unfolding** *cltn2-compose-def*  
**by** *simp*

**also have**  $\dots = \text{cltn2-abs} (\text{transpose} (\text{cltn2-rep } B) ** \text{transpose} (\text{cltn2-rep } A))$   
**by** (*simp add: matrix-transpose-mul*)

**also from**  $\langle \text{invertible} (\text{transpose} (\text{cltn2-rep } B)) \rangle$   
**and**  $\langle \text{invertible} (\text{transpose} (\text{cltn2-rep } A)) \rangle$   
**and** *cltn2-compose-abs*

**have**  $\dots = \text{cltn2-compose} (\text{cltn2-transpose } B) (\text{cltn2-transpose } A)$   
**unfolding** *cltn2-transpose-def*  
**by** *simp*

**finally show**  $\text{cltn2-transpose} (\text{cltn2-compose } A B)$   
 $= \text{cltn2-compose} (\text{cltn2-transpose } B) (\text{cltn2-transpose } A)$  .

**qed**

**lemma** *cltn2-transpose-transpose:*  $\text{cltn2-transpose} (\text{cltn2-transpose } A) = A$

**proof** –

**from** *cltn2-rep-invertible* **have**  $\text{invertible} (\text{cltn2-rep } A)$  **by** *simp*  
**with** *transpose-invertible* **have**  $\text{invertible} (\text{transpose} (\text{cltn2-rep } A))$  **by** *auto*  
**with** *cltn2-transpose-abs* [*of transpose (cltn2-rep A)*]

**have**

$\text{cltn2-transpose} (\text{cltn2-transpose } A) = \text{cltn2-abs} (\text{transpose} (\text{transpose} (\text{cltn2-rep } A)))$

**unfolding** *cltn2-transpose-def* [*of A*]

**by** *simp*  
**with** *cltn2-abs-rep* **and** *transpose-transpose* [*of cltn2-rep A*]  
**show** *cltn2-transpose* (*cltn2-transpose A*) = *A* **by** *simp*  
**qed**

**lemma** *cltn2-transpose-id* [*simp*]: *cltn2-transpose cltn2-id* = *cltn2-id*  
**using** *cltn2-transpose-abs*  
**unfolding** *cltn2-id-def*  
**by** (*simp add: transpose-mat matrix-id-invertible*)

**lemma** *apply-cltn2-line-id* [*simp*]: *apply-cltn2-line l cltn2-id* = *l*  
**unfolding** *apply-cltn2-line-def*  
**by** *simp*

**lemma** *apply-cltn2-line-compose*:  
*apply-cltn2-line* (*apply-cltn2-line l A*) *B*  
= *apply-cltn2-line l* (*cltn2-compose A B*)

**proof** –  
**have** *cltn2-compose*  
(*cltn2-transpose* (*cltn2-inverse A*)) (*cltn2-transpose* (*cltn2-inverse B*))  
= *cltn2-transpose* (*cltn2-inverse* (*cltn2-compose A B*))  
**by** (*simp add: cltn2-transpose-compose cltn2-inverse-compose*)  
**thus** *apply-cltn2-line* (*apply-cltn2-line l A*) *B*  
= *apply-cltn2-line l* (*cltn2-compose A B*)  
**unfolding** *apply-cltn2-line-def*  
**by** (*simp add: apply-cltn2-compose*)  
**qed**

**interpretation** *cltn2-line*:

*action*  
(|*carrier* = *UNIV*, *mult* = *cltn2-compose*, *one* = *cltn2-id*|)  
*apply-cltn2-line*

**proof**  
**let** *?G* = (|*carrier* = *UNIV*, *mult* = *cltn2-compose*, *one* = *cltn2-id*|)  
**fix** *l*  
**show** *apply-cltn2-line l 1 ?G* = *l* **by** *simp*  
**fix** *A B*  
**have** *apply-cltn2-line* (*apply-cltn2-line l A*) *B*  
= *apply-cltn2-line l* (*A*  $\otimes_{?G}$  *B*)  
**by** *simp* (*rule apply-cltn2-line-compose*)  
**thus** *A*  $\in$  *carrier ?G*  $\wedge$  *B*  $\in$  *carrier ?G*  
 $\longrightarrow$  *apply-cltn2-line* (*apply-cltn2-line l A*) *B*  
= *apply-cltn2-line l* (*A*  $\otimes_{?G}$  *B*)

**qed**

**lemmas** *apply-cltn2-inv* [*simp*] = *cltn2.act-act-inv* [*simplified*]

**lemmas** *apply-cltn2-line-inv* [*simp*] = *cltn2-line.act-act-inv* [*simplified*]

**lemma** *apply-cltn2-line-alt-def*:  
*apply-cltn2-line l A*  
 $= \text{proj2-line-abs } (\text{cltn2-rep } (\text{cltn2-inverse } A) *v \text{proj2-line-rep } l)$   
**proof** –  
**have** *invertible (cltn2-rep (cltn2-inverse A))* **by** (*rule cltn2-rep-invertible*)  
**hence** *invertible (transpose (cltn2-rep (cltn2-inverse A)))*  
**by** (*rule transpose-invertible*)  
**hence**  
*apply-cltn2 (L2P l) (cltn2-transpose (cltn2-inverse A))*  
 $= \text{proj2-abs } (\text{proj2-rep } (L2P l) v * \text{transpose } (\text{cltn2-rep } (\text{cltn2-inverse } A)))$   
**unfolding** *cltn2-transpose-def*  
**by** (*rule apply-cltn2-right-abs*)  
**hence** *apply-cltn2 (L2P l) (cltn2-transpose (cltn2-inverse A))*  
 $= \text{proj2-abs } (\text{cltn2-rep } (\text{cltn2-inverse } A) *v \text{proj2-line-rep } l)$   
**unfolding** *proj2-line-rep-def*  
**by** *simp*  
**thus** *apply-cltn2-line l A*  
 $= \text{proj2-line-abs } (\text{cltn2-rep } (\text{cltn2-inverse } A) *v \text{proj2-line-rep } l)$   
**unfolding** *apply-cltn2-line-def* **and** *proj2-line-abs-def* ..  
**qed**

**lemma** *rep-mult-line-rep-non-zero*: *cltn2-rep A \*v proj2-line-rep l  $\neq 0$*   
**using** *proj2-line-rep-non-zero* **and** *cltn2-rep-invertible*  
**and** *invertible-times-eq-zero*  
**by** *auto*

**lemma** *apply-cltn2-incident*:  
*proj2-incident p (apply-cltn2-line l A)*  
 $\longleftrightarrow \text{proj2-incident } (\text{apply-cltn2 } p (\text{cltn2-inverse } A)) l$   
**proof** –  
**have** *proj2-rep p v \* cltn2-rep (cltn2-inverse A)  $\neq 0$*   
**by** (*rule rep-mult-line-rep-non-zero*)  
**with** *proj2-rep-abs2*  
**obtain** *j where j  $\neq 0$*   
**and** *proj2-rep (proj2-abs (proj2-rep p v \* cltn2-rep (cltn2-inverse A)))*  
 $= j *_R (\text{proj2-rep } p v * \text{cltn2-rep } (\text{cltn2-inverse } A))$   
**by** *auto*  
  
**let** *?v = cltn2-rep (cltn2-inverse A) \*v proj2-line-rep l*  
**have** *?v  $\neq 0$*  **by** (*rule rep-mult-line-rep-non-zero*)  
**with** *proj2-line-rep-abs [of ?v]*  
**obtain** *k where k  $\neq 0$*   
**and** *proj2-line-rep (proj2-line-abs ?v) = k \*\_R ?v*  
**by** *auto*  
**hence** *proj2-incident p (apply-cltn2-line l A)*  
 $\longleftrightarrow \text{proj2-rep } p \cdot (\text{cltn2-rep } (\text{cltn2-inverse } A) *v \text{proj2-line-rep } l) = 0$   
**unfolding** *proj2-incident-def* **and** *apply-cltn2-line-alt-def*  
**by** (*simp add: dot-scaleR-mult*)  
**also from** *dot-lmul-matrix [of proj2-rep p cltn2-rep (cltn2-inverse A)]*

**have**  
 $\dots \longleftrightarrow (\text{proj2-rep } p \ v* \ \text{cltn2-rep } (\text{cltn2-inverse } A)) \cdot \text{proj2-line-rep } l = 0$   
**by** *simp*  
**also from**  $\langle j \neq 0 \rangle$   
**and**  $\langle \text{proj2-rep } (\text{proj2-abs } (\text{proj2-rep } p \ v* \ \text{cltn2-rep } (\text{cltn2-inverse } A)))$   
 $= j *_{\mathbb{R}} (\text{proj2-rep } p \ v* \ \text{cltn2-rep } (\text{cltn2-inverse } A)) \rangle$   
**have**  $\dots \longleftrightarrow \text{proj2-incident } (\text{apply-cltn2 } p \ (\text{cltn2-inverse } A)) \ l$   
**unfolding** *proj2-incident-def* **and** *apply-cltn2-def*  
**by** (*simp add: dot-scaleR-mult*)  
**finally show** *?thesis* .  
**qed**

**lemma** *apply-cltn2-preserve-incident [iff]*:  
 $\text{proj2-incident } (\text{apply-cltn2 } p \ A) \ (\text{apply-cltn2-line } l \ A)$   
 $\longleftrightarrow \text{proj2-incident } p \ l$   
**by** (*simp add: apply-cltn2-incident*)

**lemma** *apply-cltn2-preserve-set-Col*:  
**assumes** *proj2-set-Col S*  
**shows** *proj2-set-Col*  $\{\text{apply-cltn2 } p \ C \mid p. p \in S\}$   
**proof** –  
**from**  $\langle \text{proj2-set-Col } S \rangle$   
**obtain** *l* **where**  $\forall p \in S. \text{proj2-incident } p \ l$  **unfolding** *proj2-set-Col-def* ..  
**hence**  $\forall q \in \{\text{apply-cltn2 } p \ C \mid p. p \in S\}.$   
 $\text{proj2-incident } q \ (\text{apply-cltn2-line } l \ C)$   
**by** *auto*  
**thus** *proj2-set-Col*  $\{\text{apply-cltn2 } p \ C \mid p. p \in S\}$   
**unfolding** *proj2-set-Col-def* ..  
**qed**

**lemma** *apply-cltn2-injective*:  
**assumes**  $\text{apply-cltn2 } p \ C = \text{apply-cltn2 } q \ C$   
**shows**  $p = q$   
**proof** –  
**from**  $\langle \text{apply-cltn2 } p \ C = \text{apply-cltn2 } q \ C \rangle$   
**have**  $\text{apply-cltn2 } (\text{apply-cltn2 } p \ C) \ (\text{cltn2-inverse } C)$   
 $= \text{apply-cltn2 } (\text{apply-cltn2 } q \ C) \ (\text{cltn2-inverse } C)$   
**by** *simp*  
**thus**  $p = q$  **by** *simp*  
**qed**

**lemma** *apply-cltn2-line-injective*:  
**assumes**  $\text{apply-cltn2-line } l \ C = \text{apply-cltn2-line } m \ C$   
**shows**  $l = m$   
**proof** –  
**from**  $\langle \text{apply-cltn2-line } l \ C = \text{apply-cltn2-line } m \ C \rangle$   
**have**  $\text{apply-cltn2-line } (\text{apply-cltn2-line } l \ C) \ (\text{cltn2-inverse } C)$   
 $= \text{apply-cltn2-line } (\text{apply-cltn2-line } m \ C) \ (\text{cltn2-inverse } C)$   
**by** *simp*

thus  $l = m$  by *simp*  
qed

**lemma** *apply-cltn2-line-unique*:

assumes  $p \neq q$  and *proj2-incident*  $p\ l$  and *proj2-incident*  $q\ l$   
and *proj2-incident* (*apply-cltn2*  $p\ C$ )  $m$   
and *proj2-incident* (*apply-cltn2*  $q\ C$ )  $m$   
shows *apply-cltn2-line*  $l\ C = m$

**proof** –

**from**  $\langle$ *proj2-incident*  $p\ l$  $\rangle$   
**have** *proj2-incident* (*apply-cltn2*  $p\ C$ ) (*apply-cltn2-line*  $l\ C$ ) by *simp*

**from**  $\langle$ *proj2-incident*  $q\ l$  $\rangle$   
**have** *proj2-incident* (*apply-cltn2*  $q\ C$ ) (*apply-cltn2-line*  $l\ C$ ) by *simp*

**from**  $\langle$  $p \neq q$  $\rangle$  and *apply-cltn2-injective* [*of*  $p\ C\ q$ ]  
**have** *apply-cltn2*  $p\ C \neq$  *apply-cltn2*  $q\ C$  by *auto*  
**with**  $\langle$ *proj2-incident* (*apply-cltn2*  $p\ C$ ) (*apply-cltn2-line*  $l\ C$ ) $\rangle$   
and  $\langle$ *proj2-incident* (*apply-cltn2*  $q\ C$ ) (*apply-cltn2-line*  $l\ C$ ) $\rangle$   
and  $\langle$ *proj2-incident* (*apply-cltn2*  $p\ C$ )  $m$  $\rangle$   
and  $\langle$ *proj2-incident* (*apply-cltn2*  $q\ C$ )  $m$  $\rangle$   
and *proj2-incident-unique*

**show** *apply-cltn2-line*  $l\ C = m$  by *fast*

qed

**lemma** *apply-cltn2-unique*:

assumes  $l \neq m$  and *proj2-incident*  $p\ l$  and *proj2-incident*  $p\ m$   
and *proj2-incident*  $q$  (*apply-cltn2-line*  $l\ C$ )  
and *proj2-incident*  $q$  (*apply-cltn2-line*  $m\ C$ )  
shows *apply-cltn2*  $p\ C = q$

**proof** –

**from**  $\langle$ *proj2-incident*  $p\ l$  $\rangle$   
**have** *proj2-incident* (*apply-cltn2*  $p\ C$ ) (*apply-cltn2-line*  $l\ C$ ) by *simp*

**from**  $\langle$ *proj2-incident*  $p\ m$  $\rangle$   
**have** *proj2-incident* (*apply-cltn2*  $p\ C$ ) (*apply-cltn2-line*  $m\ C$ ) by *simp*

**from**  $\langle$  $l \neq m$  $\rangle$  and *apply-cltn2-line-injective* [*of*  $l\ C\ m$ ]  
**have** *apply-cltn2-line*  $l\ C \neq$  *apply-cltn2-line*  $m\ C$  by *auto*  
**with**  $\langle$ *proj2-incident* (*apply-cltn2*  $p\ C$ ) (*apply-cltn2-line*  $l\ C$ ) $\rangle$   
and  $\langle$ *proj2-incident* (*apply-cltn2*  $p\ C$ ) (*apply-cltn2-line*  $m\ C$ ) $\rangle$   
and  $\langle$ *proj2-incident*  $q$  (*apply-cltn2-line*  $l\ C$ ) $\rangle$   
and  $\langle$ *proj2-incident*  $q$  (*apply-cltn2-line*  $m\ C$ ) $\rangle$   
and *proj2-incident-unique*

**show** *apply-cltn2*  $p\ C = q$  by *fast*

qed

### 7.4.3 Parts of some Statements from [1]

All theorems with names beginning with *statement* are based on corresponding theorems in [1].

**lemma** *statement52-existence*:

**fixes**  $a :: \text{proj2}^{\wedge}3$  **and**  $a3 :: \text{proj2}$   
**assumes**  $\text{proj2-no-3-Col } (\text{insert } a3 \ (\text{range } ((\$) \ a)))$   
**shows**  $\exists A. \text{apply-cltn2 } (\text{proj2-abs } (\text{vector } [1,1,1])) \ A = a3 \wedge$   
 $(\forall j. \text{apply-cltn2 } (\text{proj2-abs } (\text{axis } j \ 1))) \ A = a\$j$

**proof** –

**let**  $?v = \text{proj2-rep } a3$   
**let**  $?B = \text{proj2-rep } \langle \text{range } ((\$) \ a) \rangle$

**from**  $\langle \text{proj2-no-3-Col } (\text{insert } a3 \ (\text{range } ((\$) \ a))) \rangle$   
**have**  $\text{card } (\text{insert } a3 \ (\text{range } ((\$) \ a))) = 4$  **unfolding** *proj2-no-3-Col-def* ..

**from** *card-image-le* [of *UNIV*  $(\$) \ a$ ]  
**have**  $\text{card } (\text{range } ((\$) \ a)) \leq 3$  **by** *simp*  
**with** *card-insert-if* [of  $\text{range } ((\$) \ a) \ a3$ ]  
**and**  $\langle \text{card } (\text{insert } a3 \ (\text{range } ((\$) \ a))) = 4 \rangle$   
**have**  $a3 \notin \text{range } ((\$) \ a)$  **by** *auto*  
**hence**  $(\text{insert } a3 \ (\text{range } ((\$) \ a))) - \{a3\} = \text{range } ((\$) \ a)$  **by** *simp*  
**with**  $\langle \text{proj2-no-3-Col } (\text{insert } a3 \ (\text{range } ((\$) \ a))) \rangle$   
**and** *proj2-no-3-Col-span* [of  $\text{insert } a3 \ (\text{range } ((\$) \ a)) \ a3$ ]  
**have**  $\text{span } ?B = \text{UNIV}$  **by** *simp*

**from** *card-suc-ge-insert* [of  $a3 \ \text{range } ((\$) \ a)$ ]  
**and**  $\langle \text{card } (\text{insert } a3 \ (\text{range } ((\$) \ a))) = 4 \rangle$   
**and**  $\langle \text{card } (\text{range } ((\$) \ a)) \leq 3 \rangle$   
**have**  $\text{card } (\text{range } ((\$) \ a)) = 3$  **by** *simp*  
**with** *card-image* [of  $\text{proj2-rep } \text{range } ((\$) \ a)$ ]  
**and** *proj2-rep-inj*  
**and** *inj-on-subset*  
**have**  $\text{card } ?B = 3$  **by** *auto*  
**hence** *finite*  $?B$  **by** *simp*  
**with**  $\langle \text{span } ?B = \text{UNIV} \rangle$  **and** *span-finite* [of  $?B$ ]  
**obtain**  $c$  **where**  $(\sum w \in ?B. (c \ w) \ *_R \ w) = ?v$   
**by** (*auto simp add: scalar-equiv*) (*metis (no-types, lifting) UNIV-I rangeE*)  
**let**  $?C = \chi \ i. c \ (\text{proj2-rep } (a\$i)) \ *_R \ (\text{proj2-rep } (a\$i))$   
**let**  $?A = \text{cltn2-abs } ?C$

**from** *proj2-rep-inj* **and**  $\langle a3 \notin \text{range } ((\$) \ a) \rangle$  **have**  $?v \notin ?B$   
**unfolding** *inj-on-def*  
**by** *auto*

**have**  $\forall i. c \ (\text{proj2-rep } (a\$i)) \neq 0$

**proof**

**fix**  $i$   
**let**  $?Bi = \text{proj2-rep } \langle \text{range } ((\$) \ a) - \{a\$i\} \rangle$

**have**  $a\$i \in \text{insert } a3 \text{ (range ((\$) a))}$  **by simp**

**have**  $\text{proj2-rep } (a\$i) \in ?B$  **by auto**

**from** *image-set-diff* [of *proj2-rep*] **and** *proj2-rep-inj*

**have**  $?Bi = ?B - \{\text{proj2-rep } (a\$i)\}$  **by simp**

**with** *sum-diff1* [of  $?B \lambda w. (c w) *_R w$ ]

**and**  $\langle \text{finite } ?B \rangle$

**and**  $\langle \text{proj2-rep } (a\$i) \in ?B \rangle$

**have**  $(\sum w \in ?Bi. (c w) *_R w) =$   
 $(\sum w \in ?B. (c w) *_R w) - c (\text{proj2-rep } (a\$i)) *_R \text{proj2-rep } (a\$i)$

**by simp**

**from**  $\langle a3 \notin \text{range ((\$) a)} \rangle$  **have**  $a3 \neq a\$i$  **by auto**

**hence**  $\text{insert } a3 \text{ (range ((\$) a))} - \{a\$i\} =$   
 $\text{insert } a3 \text{ (range ((\$) a)} - \{a\$i\}$  **by auto**

**hence**  $\text{proj2-rep } '(\text{insert } a3 \text{ (range ((\$) a))} - \{a\$i\}) = \text{insert } ?v ?Bi$

**by simp**

**moreover from**  $\langle \text{proj2-no-3-Col } (\text{insert } a3 \text{ (range ((\$) a))) \rangle$

**and**  $\langle a\$i \in \text{insert } a3 \text{ (range ((\$) a))} \rangle$

**have**  $\text{span } (\text{proj2-rep } '(\text{insert } a3 \text{ (range ((\$) a))} - \{a\$i\})) = \text{UNIV}$

**by** (*rule proj2-no-3-Col-span*)

**ultimately have**  $\text{span } (\text{insert } ?v ?Bi) = \text{UNIV}$  **by simp**

**from**  $\langle ?Bi = ?B - \{\text{proj2-rep } (a\$i)\} \rangle$

**and**  $\langle \text{proj2-rep } (a\$i) \in ?B \rangle$

**and**  $\langle \text{card } ?B = 3 \rangle$

**have**  $\text{card } ?Bi = 2$  **by** (*simp add: card-gt-0-diff-singleton*)

**hence** *finite*  $?Bi$  **by simp**

**with**  $\langle \text{card } ?Bi = 2 \rangle$  **and** *dim-le-card'* [of  $?Bi$ ] **have**  $\text{dim } ?Bi \leq 2$  **by simp**

**hence**  $\text{dim } (\text{span } ?Bi) \leq 2$  **by** (*subst dim-span*)

**then have**  $\text{span } ?Bi \neq \text{UNIV}$

**by** *clarify* (*auto simp: dim-UNIV*)

**with**  $\langle \text{span } (\text{insert } ?v ?Bi) = \text{UNIV} \rangle$  **and** *span-redundant*

**have**  $?v \notin \text{span } ?Bi$  **by auto**

**{ assume**  $c (\text{proj2-rep } (a\$i)) = 0$

**with**  $\langle (\sum w \in ?Bi. (c w) *_R w) =$   
 $(\sum w \in ?B. (c w) *_R w) - c (\text{proj2-rep } (a\$i)) *_R \text{proj2-rep } (a\$i) \rangle$

**and**  $\langle (\sum w \in ?B. (c w) *_R w) = ?v \rangle$

**have**  $?v = (\sum w \in ?Bi. (c w) *_R w)$

**by simp**

**with** *span-finite* [of  $?Bi$ ] **and**  $\langle \text{finite } ?Bi \rangle$

**have**  $?v \in \text{span } ?Bi$  **by** (*simp add: scalar-equiv*)

**with**  $\langle ?v \notin \text{span } ?Bi \rangle$  **have** *False ..* **}**

**thus**  $c (\text{proj2-rep } (a\$i)) \neq 0$  **..**

**qed**

**hence**  $\forall w \in ?B. c w \neq 0$

```

unfolding image-def
by auto

have  $\text{rows } ?C = (\lambda w. (c\ w) *_{R} w) \text{ ' } ?B$ 
unfolding rows-def
and row-def
and image-def
by (auto simp: vec-lambda-eta)

have  $\forall x. x \in \text{span } (\text{rows } ?C)$ 
proof
  fix  $x :: \text{real}^3$ 
  from  $\langle \text{finite } ?B \rangle$  and span-finite [of ?B] and  $\langle \text{span } ?B = \text{UNIV} \rangle$ 
  obtain  $ub$  where  $(\sum w \in ?B. (ub\ w) *_{R} w) = x$ 
  by (auto simp add: scalar-equiv) (metis (no-types, lifting) UNIV-I rangeE)
  have  $\forall w \in ?B. (ub\ w) *_{R} w \in \text{span } (\text{rows } ?C)$ 
  proof
    fix  $w$ 
    assume  $w \in ?B$ 
    with span-superset [of rows ?C] and  $\langle \text{rows } ?C = \text{image } (\lambda w. (c\ w) *_{R} w) \text{ ' } ?B \rangle$ 
    have  $(c\ w) *_{R} w \in \text{span } (\text{rows } ?C)$  by auto
    with span-mul [of (c w) *R w rows ?C (ub w)/(c w)]
    have  $((ub\ w)/(c\ w)) *_{R} ((c\ w) *_{R} w) \in \text{span } (\text{rows } ?C)$ 
    by (simp add: scalar-equiv)
    with  $\langle \forall w \in ?B. c\ w \neq 0 \rangle$  and  $\langle w \in ?B \rangle$ 
    show  $(ub\ w) *_{R} w \in \text{span } (\text{rows } ?C)$  by auto
  qed
  with span-sum [of ?B  $\lambda w. (ub\ w) *_{R} w$ ] and  $\langle \text{finite } ?B \rangle$ 
  have  $(\sum w \in ?B. (ub\ w) *_{R} w) \in \text{span } (\text{rows } ?C)$  by blast
  with  $\langle (\sum w \in ?B. (ub\ w) *_{R} w) = x \rangle$  show  $x \in \text{span } (\text{rows } ?C)$  by simp
qed
hence  $\text{span } (\text{rows } ?C) = \text{UNIV}$  by auto
with matrix-left-invertible-span-rows [of ?C]
have  $\exists C'. C' ** ?C = \text{mat } 1 \dots$ 
with left-invertible-iff-invertible
have invertible ?C ..

have  $(\text{vector } [1,1,1] :: \text{real}^3) \neq 0$ 
unfolding vector-def
by (simp add: vec-eq-iff forall-3)
with apply-cltn2-abs and  $\langle \text{invertible } ?C \rangle$ 
have apply-cltn2 (proj2-abs (vector [1,1,1])) ?A =
  proj2-abs (vector [1,1,1] v* ?C)
by simp
from inj-on-iff-eq-card [of UNIV ( $\$$ ) a] and  $\langle \text{card } (\text{range } ((\$)\ a)) = 3 \rangle$ 
have inj ((\$) a) by simp
from exhaust-3 have  $\forall i :: 3. (\text{vector } [1 :: \text{real}, 1, 1]) \$ i = 1$ 
unfolding vector-def

```

**by auto**  
**with** *vector-matrix-row* [of vector [1,1,1] ?C]  
**have** (vector [1,1,1] v\* ?C =  
 $(\sum_{i \in UNIV}. (c (proj2\text{-rep} (a\$i))) *_R (proj2\text{-rep} (a\$i)))$ )  
**by simp**  
**also from** *sum.reindex*  
[of (\$) a UNIV  $\lambda x. (c (proj2\text{-rep} x)) *_R (proj2\text{-rep} x)$ ]  
**and**  $\langle inj ((\$ a) \rangle$   
**have** ... =  $(\sum_{x \in (range ((\$ a)).} (c (proj2\text{-rep} x)) *_R (proj2\text{-rep} x))$   
**by simp**  
**also from** *sum.reindex*  
[of *proj2-rep* range ((\\$ a)  $\lambda w. (c w) *_R w$ ]  
**and** *proj2-rep-inj* **and** *inj-on-subset* [of *proj2-rep* UNIV range ((\\$ a) ]  
**have** ... =  $(\sum_{w \in ?B. (c w) *_R w)$  **by simp**  
**also from**  $\langle (\sum w \in ?B. (c w) *_R w) = ?v \rangle$  **have** ... = ?v **by simp**  
**finally have** (vector [1,1,1] v\* ?C = ?v .  
**with**  $\langle apply\text{-cltn2} (proj2\text{-abs} (vector [1,1,1])) ?A =$   
 $proj2\text{-abs} (vector [1,1,1] v* ?C) \rangle$   
**have** *apply-cltn2* (proj2-abs (vector [1,1,1])) ?A = *proj2-abs* ?v **by simp**  
**with** *proj2-abs-rep* **have** *apply-cltn2* (proj2-abs (vector [1,1,1])) ?A = a3  
**by simp**  
**have**  $\forall j. apply\text{-cltn2} (proj2\text{-abs} (axis j 1)) ?A = a\$j$   
**proof**  
**fix** j :: 3  
**have** ((axis j 1)::real^3)  $\neq 0$  **by** (*simp add: vec-eq-iff axis-def*)  
**with** *apply-cltn2-abs* **and**  $\langle invertible ?C \rangle$   
**have** *apply-cltn2* (proj2-abs (axis j 1)) ?A = *proj2-abs* (axis j 1 v\* ?C)  
**by simp**  
  
**have**  $\forall i \in (UNIV - \{j\}).$   
 $((axis j 1)\$i * c (proj2\text{-rep} (a\$i))) *_R (proj2\text{-rep} (a\$i)) = 0$   
**by** (*simp add: axis-def*)  
**with** *sum.mono-neutral-left* [of UNIV {j}]  
 $\lambda i. ((axis j 1)\$i * c (proj2\text{-rep} (a\$i))) *_R (proj2\text{-rep} (a\$i))$   
**and** *vector-matrix-row* [of axis j 1 ?C]  
**have** (axis j 1) v\* ?C = ?C\\$j **by** (*simp add: scalar-equiv*)  
**hence** (axis j 1) v\* ?C = c (proj2-rep (a\\$j)) \*\_R (proj2-rep (a\\$j)) **by simp**  
**with** *proj2-abs-mult-rep* **and**  $\langle \forall i. c (proj2\text{-rep} (a\$i)) \neq 0 \rangle$   
**and**  $\langle apply\text{-cltn2} (proj2\text{-abs} (axis j 1)) ?A = proj2\text{-abs} (axis j 1 v* ?C) \rangle$   
**show** *apply-cltn2* (proj2-abs (axis j 1)) ?A = a\\$j  
**by simp**  
**qed**  
**with**  $\langle apply\text{-cltn2} (proj2\text{-abs} (vector [1,1,1])) ?A = a3 \rangle$   
**show**  $\exists A. apply\text{-cltn2} (proj2\text{-abs} (vector [1,1,1])) A = a3 \wedge$   
 $(\forall j. apply\text{-cltn2} (proj2\text{-abs} (axis j 1)) A = a\$j)$   
**by auto**  
**qed**

**lemma** *statement53-existence*:

```

fixes  $p :: \text{proj2}^{\wedge}4^{\wedge}2$ 
assumes  $\forall i. \text{proj2-no-3-Col } (\text{range } ((\$) (p\$i)))$ 
shows  $\exists C. \forall j. \text{apply-cltn2 } (p\$0\$j) C = p\$1\$j$ 
proof -
let  $?q = \chi i. \chi j::3. p\$i \$ (\text{of-int } (\text{Rep-bit1 } j))$ 
let  $?D = \chi i. \epsilon D. \text{apply-cltn2 } (\text{proj2-abs } (\text{vector } [1,1,1])) D = p\$i\$3$ 
 $\wedge (\forall j'. \text{apply-cltn2 } (\text{proj2-abs } (\text{axis } j' 1)) D = ?q\$i\$j')$ 
have  $\forall i. \text{apply-cltn2 } (\text{proj2-abs } (\text{vector } [1,1,1])) (?D\$i) = p\$i\$3$ 
 $\wedge (\forall j'. \text{apply-cltn2 } (\text{proj2-abs } (\text{axis } j' 1)) (?D\$i) = ?q\$i\$j')$ 
proof
fix  $i$ 
have  $\text{range } ((\$) (p\$i)) = \text{insert } (p\$i\$3) (\text{range } ((\$) (?q\$i)))$ 
proof
show  $\text{range } ((\$) (p\$i)) \supseteq \text{insert } (p\$i\$3) (\text{range } ((\$) (?q\$i)))$  by auto
show  $\text{range } ((\$) (p\$i)) \subseteq \text{insert } (p\$i\$3) (\text{range } ((\$) (?q\$i)))$ 
proof
fix  $r$ 
assume  $r \in \text{range } ((\$) (p\$i))$ 
then obtain  $j$  where  $r = p\$i\$j$  by auto
with  $\text{eq-3-or-of-3 } [\text{of } j]$ 
show  $r \in \text{insert } (p\$i\$3) (\text{range } ((\$) (?q\$i)))$  by auto
qed
qed
moreover from  $\langle \forall i. \text{proj2-no-3-Col } (\text{range } ((\$) (p\$i))) \rangle$ 
have  $\text{proj2-no-3-Col } (\text{range } ((\$) (p\$i))) ..$ 
ultimately have  $\text{proj2-no-3-Col } (\text{insert } (p\$i\$3) (\text{range } ((\$) (?q\$i))))$ 
by simp
hence  $\exists D. \text{apply-cltn2 } (\text{proj2-abs } (\text{vector } [1,1,1])) D = p\$i\$3$ 
 $\wedge (\forall j'. \text{apply-cltn2 } (\text{proj2-abs } (\text{axis } j' 1)) D = ?q\$i\$j')$ 
by  $(\text{rule statement52-existence})$ 
with  $\text{someI-ex } [\text{of } \lambda D. \text{apply-cltn2 } (\text{proj2-abs } (\text{vector } [1,1,1])) D = p\$i\$3$ 
 $\wedge (\forall j'. \text{apply-cltn2 } (\text{proj2-abs } (\text{axis } j' 1)) D = ?q\$i\$j')]$ 
show  $\text{apply-cltn2 } (\text{proj2-abs } (\text{vector } [1,1,1])) (?D\$i) = p\$i\$3$ 
 $\wedge (\forall j'. \text{apply-cltn2 } (\text{proj2-abs } (\text{axis } j' 1)) (?D\$i) = ?q\$i\$j')$ 
by simp
qed
hence  $\text{apply-cltn2 } (\text{proj2-abs } (\text{vector } [1,1,1])) (?D\$0) = p\$0\$3$ 
and  $\text{apply-cltn2 } (\text{proj2-abs } (\text{vector } [1,1,1])) (?D\$1) = p\$1\$3$ 
and  $\forall j'. \text{apply-cltn2 } (\text{proj2-abs } (\text{axis } j' 1)) (?D\$0) = ?q\$0\$j'$ 
and  $\forall j'. \text{apply-cltn2 } (\text{proj2-abs } (\text{axis } j' 1)) (?D\$1) = ?q\$1\$j'$ 
by simp-all

let  $?C = \text{cltn2-compose } (\text{cltn2-inverse } (?D\$0)) (?D\$1)$ 
have  $\forall j. \text{apply-cltn2 } (p\$0\$j) ?C = p\$1\$j$ 
proof
fix  $j$ 
show  $\text{apply-cltn2 } (p\$0\$j) ?C = p\$1\$j$ 
proof cases
assume  $j = 3$ 

```

**with**  $\langle \text{apply-cltn2} (\text{proj2-abs} (\text{vector} [1,1,1])) (?D\$0) = p\$0\$3 \rangle$   
**and**  $\text{cltn2.act-inv-iff}$   
**have**  
 $\text{apply-cltn2} (p\$0\$j) (\text{cltn2-inverse} (?D\$0)) = \text{proj2-abs} (\text{vector} [1,1,1])$   
**by** *simp*  
**with**  $\langle \text{apply-cltn2} (\text{proj2-abs} (\text{vector} [1,1,1])) (?D\$1) = p\$1\$3 \rangle$   
**and**  $\langle j = 3 \rangle$   
**and**  $\text{cltn2.act-act} [\text{of cltn2-inverse} (?D\$0) ?D\$1 p\$0\$j]$   
**show**  $\text{apply-cltn2} (p\$0\$j) ?C = p\$1\$j$  **by** *simp*  
**next**  
**assume**  $j \neq 3$   
**with** *eq-3-or-of-3* **obtain**  $j' :: 3$  **where**  $j = \text{of-int} (\text{Rep-bit1 } j')$   
**by** *metis*  
**with**  $\langle \forall j'. \text{apply-cltn2} (\text{proj2-abs} (\text{axis } j' 1)) (?D\$0) = ?q\$0\$j' \rangle$   
**and**  $\langle \forall j'. \text{apply-cltn2} (\text{proj2-abs} (\text{axis } j' 1)) (?D\$1) = ?q\$1\$j' \rangle$   
**have**  $p\$0\$j = \text{apply-cltn2} (\text{proj2-abs} (\text{axis } j' 1)) (?D\$0)$   
**and**  $p\$1\$j = \text{apply-cltn2} (\text{proj2-abs} (\text{axis } j' 1)) (?D\$1)$   
**by** *simp-all*  
**from**  $\langle p\$0\$j = \text{apply-cltn2} (\text{proj2-abs} (\text{axis } j' 1)) (?D\$0) \rangle$   
**and**  $\text{cltn2.act-inv-iff}$   
**have**  $\text{apply-cltn2} (p\$0\$j) (\text{cltn2-inverse} (?D\$0)) = \text{proj2-abs} (\text{axis } j' 1)$   
**by** *simp*  
**with**  $\langle p\$1\$j = \text{apply-cltn2} (\text{proj2-abs} (\text{axis } j' 1)) (?D\$1) \rangle$   
**and**  $\text{cltn2.act-act} [\text{of cltn2-inverse} (?D\$0) ?D\$1 p\$0\$j]$   
**show**  $\text{apply-cltn2} (p\$0\$j) ?C = p\$1\$j$  **by** *simp*  
**qed**  
**qed**  
**thus**  $\exists C. \forall j. \text{apply-cltn2} (p\$0\$j) C = p\$1\$j$  **by** (*rule exI [of - ?C]*)  
**qed**

**lemma** *apply-cltn2-linear*:  
**assumes**  $j *_R v + k *_R w \neq 0$   
**shows**  $j *_R (v *_R \text{cltn2-rep } C) + k *_R (w *_R \text{cltn2-rep } C) \neq 0$   
**(is**  $?u \neq 0$ **)**  
**and**  $\text{apply-cltn2} (\text{proj2-abs} (j *_R v + k *_R w)) C$   
 $= \text{proj2-abs} (j *_R (v *_R \text{cltn2-rep } C) + k *_R (w *_R \text{cltn2-rep } C))$   
**proof** –  
**have**  $?u = (j *_R v + k *_R w) *_R \text{cltn2-rep } C$   
**by** (*simp only: vector-matrix-left-distrib scaleR-vector-matrix-assoc*)  
**with**  $\langle j *_R v + k *_R w \neq 0 \rangle$  **and** *non-zero-mult-rep-non-zero*  
**show**  $?u \neq 0$  **by** *simp*  
  
**from**  $\langle ?u = (j *_R v + k *_R w) *_R \text{cltn2-rep } C \rangle$   
**and**  $\langle j *_R v + k *_R w \neq 0 \rangle$   
**and** *apply-cltn2-left-abs*  
**show**  $\text{apply-cltn2} (\text{proj2-abs} (j *_R v + k *_R w)) C = \text{proj2-abs } ?u$   
**by** *simp*  
**qed**

**lemma** *apply-cltn2-imp-mult*:  
**assumes** *apply-cltn2*  $p\ C = q$   
**shows**  $\exists k. k \neq 0 \wedge \text{proj2-rep } p\ v^*\ \text{cltn2-rep } C = k *_R \text{proj2-rep } q$   
**proof** –  
**have** *proj2-rep*  $p\ v^*\ \text{cltn2-rep } C \neq 0$  **by** (*rule rep-mult-rep-non-zero*)  
  
**from**  $\langle \text{apply-cltn2 } p\ C = q \rangle$   
**have** *proj2-abs* (*proj2-rep*  $p\ v^*\ \text{cltn2-rep } C$ ) =  $q$  **by** (*unfold apply-cltn2-def*)  
**hence** *proj2-rep* (*proj2-abs* (*proj2-rep*  $p\ v^*\ \text{cltn2-rep } C$ )) = *proj2-rep*  $q$   
**by** *simp*  
**with**  $\langle \text{proj2-rep } p\ v^*\ \text{cltn2-rep } C \neq 0 \rangle$  **and** *proj2-rep-abs2* [*of proj2-rep*  $p\ v^*\ \text{cltn2-rep } C$ ]  
**have**  $\exists j. j \neq 0 \wedge \text{proj2-rep } q = j *_R (\text{proj2-rep } p\ v^*\ \text{cltn2-rep } C)$  **by** *simp*  
**then obtain**  $j$  **where**  $j \neq 0$   
**and** *proj2-rep*  $q = j *_R (\text{proj2-rep } p\ v^*\ \text{cltn2-rep } C)$  **by** *auto*  
**hence** *proj2-rep*  $p\ v^*\ \text{cltn2-rep } C = (1/j) *_R \text{proj2-rep } q$   
**by** (*simp add: field-simps*)  
**with**  $\langle j \neq 0 \rangle$   
**show**  $\exists k. k \neq 0 \wedge \text{proj2-rep } p\ v^*\ \text{cltn2-rep } C = k *_R \text{proj2-rep } q$   
**by** (*simp add: exI [of - 1/j]*)  
**qed**

**lemma** *statement55*:  
**assumes**  $p \neq q$   
**and** *apply-cltn2*  $p\ C = q$   
**and** *apply-cltn2*  $q\ C = p$   
**and** *proj2-incident*  $p\ l$   
**and** *proj2-incident*  $q\ l$   
**and** *proj2-incident*  $r\ l$   
**shows** *apply-cltn2* (*apply-cltn2*  $r\ C$ )  $C = r$

**proof** *cases*  
**assume**  $r = p$   
**with**  $\langle \text{apply-cltn2 } p\ C = q \rangle$  **and**  $\langle \text{apply-cltn2 } q\ C = p \rangle$   
**show** *apply-cltn2* (*apply-cltn2*  $r\ C$ )  $C = r$  **by** *simp*

**next**  
**assume**  $r \neq p$

**from**  $\langle \text{apply-cltn2 } p\ C = q \rangle$  **and** *apply-cltn2-imp-mult* [*of*  $p\ C\ q$ ]  
**obtain**  $i$  **where**  $i \neq 0$  **and** *proj2-rep*  $p\ v^*\ \text{cltn2-rep } C = i *_R \text{proj2-rep } q$   
**by** *auto*

**from**  $\langle \text{apply-cltn2 } q\ C = p \rangle$  **and** *apply-cltn2-imp-mult* [*of*  $q\ C\ p$ ]  
**obtain**  $j$  **where**  $j \neq 0$  **and** *proj2-rep*  $q\ v^*\ \text{cltn2-rep } C = j *_R \text{proj2-rep } p$   
**by** *auto*

**from**  $\langle p \neq q \rangle$   
**and**  $\langle \text{proj2-incident } p\ l \rangle$   
**and**  $\langle \text{proj2-incident } q\ l \rangle$   
**and**  $\langle \text{proj2-incident } r\ l \rangle$

**and** *proj2-incident-iff*  
**have**  $r = p \vee (\exists k. r = \text{proj2-abs } (k *_R \text{proj2-rep } p + \text{proj2-rep } q))$   
**by** *fast*  
**with**  $\langle r \neq p \rangle$   
**obtain**  $k$  **where**  $r = \text{proj2-abs } (k *_R \text{proj2-rep } p + \text{proj2-rep } q)$  **by** *auto*

**from**  $\langle p \neq q \rangle$  **and** *proj2-rep-dependent* [of  $k$   $p$   $1$   $q$ ]  
**have**  $k *_R \text{proj2-rep } p + \text{proj2-rep } q \neq 0$  **by** *auto*  
**with**  $\langle r = \text{proj2-abs } (k *_R \text{proj2-rep } p + \text{proj2-rep } q) \rangle$   
**and** *apply-cltn2-linear* [of  $k$  *proj2-rep*  $p$   $1$  *proj2-rep*  $q$ ]  
**have**  $k *_R (\text{proj2-rep } p \text{ v* } \text{cltn2-rep } C) + \text{proj2-rep } q \text{ v* } \text{cltn2-rep } C \neq 0$   
**and** *apply-cltn2*  $r$   $C$   
 $= \text{proj2-abs}$   
 $(k *_R (\text{proj2-rep } p \text{ v* } \text{cltn2-rep } C) + \text{proj2-rep } q \text{ v* } \text{cltn2-rep } C)$   
**by** *simp-all*

**with**  $\langle \text{proj2-rep } p \text{ v* } \text{cltn2-rep } C = i *_R \text{proj2-rep } q \rangle$   
**and**  $\langle \text{proj2-rep } q \text{ v* } \text{cltn2-rep } C = j *_R \text{proj2-rep } p \rangle$   
**have**  $(k * i) *_R \text{proj2-rep } q + j *_R \text{proj2-rep } p \neq 0$   
**and** *apply-cltn2*  $r$   $C$   
 $= \text{proj2-abs } ((k * i) *_R \text{proj2-rep } q + j *_R \text{proj2-rep } p)$   
**by** *simp-all*

**with** *apply-cltn2-linear*  
**have** *apply-cltn2* (*apply-cltn2*  $r$   $C$ )  $C$   
 $= \text{proj2-abs}$   
 $((k * i) *_R (\text{proj2-rep } q \text{ v* } \text{cltn2-rep } C)$   
 $+ j *_R (\text{proj2-rep } p \text{ v* } \text{cltn2-rep } C))$   
**by** *simp*

**with**  $\langle \text{proj2-rep } p \text{ v* } \text{cltn2-rep } C = i *_R \text{proj2-rep } q \rangle$   
**and**  $\langle \text{proj2-rep } q \text{ v* } \text{cltn2-rep } C = j *_R \text{proj2-rep } p \rangle$   
**have** *apply-cltn2* (*apply-cltn2*  $r$   $C$ )  $C$   
 $= \text{proj2-abs } ((k * i * j) *_R \text{proj2-rep } p + (j * i) *_R \text{proj2-rep } q)$   
**by** *simp*

**also have**  $\dots = \text{proj2-abs } ((i * j) *_R (k *_R \text{proj2-rep } p + \text{proj2-rep } q))$   
**by** (*simp add: algebra-simps*)

**also from**  $\langle i \neq 0 \rangle$  **and**  $\langle j \neq 0 \rangle$  **and** *proj2-abs-mult*  
**have**  $\dots = \text{proj2-abs } (k *_R \text{proj2-rep } p + \text{proj2-rep } q)$  **by** *simp*

**also from**  $\langle r = \text{proj2-abs } (k *_R \text{proj2-rep } p + \text{proj2-rep } q) \rangle$   
**have**  $\dots = r$  **by** *simp*

**finally show** *apply-cltn2* (*apply-cltn2*  $r$   $C$ )  $C = r$  .

qed

## 7.5 Cross ratios

**definition** *cross-ratio* :: *proj2*  $\Rightarrow$  *proj2*  $\Rightarrow$  *proj2*  $\Rightarrow$  *proj2*  $\Rightarrow$  *real* **where**  
*cross-ratio*  $p$   $q$   $r$   $s \triangleq \text{proj2-Col-coeff } p$   $q$   $s / \text{proj2-Col-coeff } p$   $q$   $r$

**definition** *cross-ratio-correct* :: *proj2*  $\Rightarrow$  *proj2*  $\Rightarrow$  *proj2*  $\Rightarrow$  *proj2*  $\Rightarrow$  *bool* **where**  
*cross-ratio-correct*  $p$   $q$   $r$   $s \triangleq$   
*proj2-set-Col*  $\{p, q, r, s\} \wedge p \neq q \wedge r \neq p \wedge s \neq p \wedge r \neq q$

**lemma** *proj2-Col-coeff-abs*:  
**assumes**  $p \neq q$  **and**  $j \neq 0$   
**shows**  $\text{proj2-Col-coeff } p \ q \ (\text{proj2-abs } (i \ *_R \text{proj2-rep } p + j \ *_R \text{proj2-rep } q))$   
 $= i/j$   
**(is**  $\text{proj2-Col-coeff } p \ q \ ?r = i/j$ **)**  
**proof** –  
**from**  $\langle j \neq 0 \rangle$   
**and**  $\text{proj2-abs-mult}$  [*of*  $1/j \ i \ *_R \text{proj2-rep } p + j \ *_R \text{proj2-rep } q$ ]  
**have**  $?r = \text{proj2-abs } ((i/j) \ *_R \text{proj2-rep } p + \text{proj2-rep } q)$   
**by** (*simp add: scaleR-right-distrib*)  
  
**from**  $\langle p \neq q \rangle$  **and**  $\text{proj2-rep-dependent}$  [*of*  $- \ p \ 1 \ q$ ]  
**have**  $(i/j) \ *_R \text{proj2-rep } p + \text{proj2-rep } q \neq 0$  **by** *auto*  
**with**  $\langle ?r = \text{proj2-abs } ((i/j) \ *_R \text{proj2-rep } p + \text{proj2-rep } q) \rangle$   
**and**  $\text{proj2-rep-abs2}$   
**obtain**  $k$  **where**  $k \neq 0$   
**and**  $\text{proj2-rep } ?r = k \ *_R ((i/j) \ *_R \text{proj2-rep } p + \text{proj2-rep } q)$   
**by** *auto*  
**hence**  $(k*i/j) \ *_R \text{proj2-rep } p + k \ *_R \text{proj2-rep } q - \text{proj2-rep } ?r = 0$   
**by** (*simp add: scaleR-right-distrib*)  
**hence**  $\exists l. (k*i/j) \ *_R \text{proj2-rep } p + k \ *_R \text{proj2-rep } q + l \ *_R \text{proj2-rep } ?r = 0$   
 $\wedge (k*i/j \neq 0 \vee k \neq 0 \vee l \neq 0)$   
**by** (*simp add: exI [of - -1]*)  
**hence**  $\text{proj2-Col } p \ q \ ?r$  **by** (*unfold proj2-Col-def*) *auto*  
  
**have**  $?r \neq p$   
**proof**  
**assume**  $?r = p$   
**with**  $\langle (k*i/j) \ *_R \text{proj2-rep } p + k \ *_R \text{proj2-rep } q - \text{proj2-rep } ?r = 0 \rangle$   
**have**  $(k*i/j - 1) \ *_R \text{proj2-rep } p + k \ *_R \text{proj2-rep } q = 0$   
**by** (*simp add: algebra-simps*)  
**with**  $\langle k \neq 0 \rangle$  **and**  $\text{proj2-rep-dependent}$  **have**  $p = q$  **by** *simp*  
**with**  $\langle p \neq q \rangle$  **show** *False ..*  
**qed**  
**with**  $\langle \text{proj2-Col } p \ q \ ?r \rangle$  **and**  $\langle p \neq q \rangle$   
**have**  $?r = \text{proj2-abs } (\text{proj2-Col-coeff } p \ q \ ?r \ *_R \text{proj2-rep } p + \text{proj2-rep } q)$   
**by** (*rule proj2-Col-coeff*)  
**with**  $\langle p \neq q \rangle$  **and**  $\langle ?r = \text{proj2-abs } ((i/j) \ *_R \text{proj2-rep } p + \text{proj2-rep } q) \rangle$   
**and**  $\text{proj2-Col-coeff-unique}$   
**show**  $\text{proj2-Col-coeff } p \ q \ ?r = i/j$  **by** *simp*  
**qed**

**lemma** *proj2-set-Col-coeff*:  
**assumes**  $\text{proj2-set-Col } S$  **and**  $\{p, q, r\} \subseteq S$  **and**  $p \neq q$  **and**  $r \neq p$   
**shows**  $r = \text{proj2-abs } (\text{proj2-Col-coeff } p \ q \ r \ *_R \text{proj2-rep } p + \text{proj2-rep } q)$   
**(is**  $r = \text{proj2-abs } (?i \ *_R ?u + ?v)$ **)**  
**proof** –  
**from**  $\langle \{p, q, r\} \subseteq S \rangle$  **and**  $\langle \text{proj2-set-Col } S \rangle$

**have** *proj2-set-Col* {*p,q,r*} **by** (*rule proj2-subset-Col*)  
**hence** *proj2-Col* *p q r* **by** (*subst proj2-Col-iff-set-Col*)  
**with**  $\langle p \neq q \rangle$  **and**  $\langle r \neq p \rangle$  **and** *proj2-Col-coeff*  
**show**  $r = \text{proj2-abs } (?i *_{\mathbb{R}} ?u + ?v)$  **by** *simp*  
**qed**

**lemma** *cross-ratio-abs*:

**fixes**  $u v :: \text{real}^3$  **and**  $i j k l :: \text{real}$   
**assumes**  $u \neq 0$  **and**  $v \neq 0$  **and**  $\text{proj2-abs } u \neq \text{proj2-abs } v$   
**and**  $j \neq 0$  **and**  $l \neq 0$   
**shows** *cross-ratio* (*proj2-abs*  $u$ ) (*proj2-abs*  $v$ )  
(*proj2-abs* ( $i *_{\mathbb{R}} u + j *_{\mathbb{R}} v$ ))  
(*proj2-abs* ( $k *_{\mathbb{R}} u + l *_{\mathbb{R}} v$ ))  
 $= j * k / (i * l)$   
**(is** *cross-ratio*  $?p ?q ?r ?s = -$ )

**proof** –

**from**  $\langle u \neq 0 \rangle$  **and** *proj2-rep-abs2*  
**obtain**  $g$  **where**  $g \neq 0$  **and**  $\text{proj2-rep } ?p = g *_{\mathbb{R}} u$  **by** *auto*

**from**  $\langle v \neq 0 \rangle$  **and** *proj2-rep-abs2*

**obtain**  $h$  **where**  $h \neq 0$  **and**  $\text{proj2-rep } ?q = h *_{\mathbb{R}} v$  **by** *auto*

**with**  $\langle g \neq 0 \rangle$  **and**  $\langle \text{proj2-rep } ?p = g *_{\mathbb{R}} u \rangle$

**have**  $?r = \text{proj2-abs } ((i/g) *_{\mathbb{R}} \text{proj2-rep } ?p + (j/h) *_{\mathbb{R}} \text{proj2-rep } ?q)$

**and**  $?s = \text{proj2-abs } ((k/g) *_{\mathbb{R}} \text{proj2-rep } ?p + (l/h) *_{\mathbb{R}} \text{proj2-rep } ?q)$

**by** (*simp-all add: field-simps*)

**with**  $\langle ?p \neq ?q \rangle$  **and**  $\langle h \neq 0 \rangle$  **and**  $\langle j \neq 0 \rangle$  **and**  $\langle l \neq 0 \rangle$  **and** *proj2-Col-coeff-abs*

**have** *proj2-Col-coeff*  $?p ?q ?r = h*i/(g*j)$

**and** *proj2-Col-coeff*  $?p ?q ?s = h*k/(g*l)$

**by** *simp-all*

**with**  $\langle g \neq 0 \rangle$  **and**  $\langle h \neq 0 \rangle$

**show** *cross-ratio*  $?p ?q ?r ?s = j*k/(i*l)$

**by** (*unfold cross-ratio-def*) (*simp add: field-simps*)

**qed**

**lemma** *cross-ratio-abs2*:

**assumes**  $p \neq q$

**shows** *cross-ratio*  $p q$

(*proj2-abs* ( $i *_{\mathbb{R}} \text{proj2-rep } p + \text{proj2-rep } q$ ))

(*proj2-abs* ( $j *_{\mathbb{R}} \text{proj2-rep } p + \text{proj2-rep } q$ ))

$= j/i$

**(is** *cross-ratio*  $p q ?r ?s = -$ )

**proof** –

**let**  $?u = \text{proj2-rep } p$

**let**  $?v = \text{proj2-rep } q$

**have**  $?u \neq 0$  **and**  $?v \neq 0$  **by** (*rule proj2-rep-non-zero*)+

**have** *proj2-abs*  $?u = p$  **and** *proj2-abs*  $?v = q$  **by** (*rule proj2-abs-rep*)+

**with**  $\langle ?u \neq 0 \rangle$  **and**  $\langle ?v \neq 0 \rangle$  **and**  $\langle p \neq q \rangle$  **and** *cross-ratio-abs* [*of*  $?u ?v$  1 1  $i j$ ]

**show** *cross-ratio*  $p q ?r ?s = j/i$  **by** *simp*

qed

**lemma** *cross-ratio-correct-cltn2*:

**assumes** *cross-ratio-correct*  $p\ q\ r\ s$

**shows** *cross-ratio-correct* (*apply-cltn2*  $p\ C$ ) (*apply-cltn2*  $q\ C$ )

(*apply-cltn2*  $r\ C$ ) (*apply-cltn2*  $s\ C$ )

(**is** *cross-ratio-correct*  $?pC\ ?qC\ ?rC\ ?sC$ )

**proof** –

**from**  $\langle \textit{cross-ratio-correct}\ p\ q\ r\ s \rangle$

**have** *proj2-set-Col*  $\{p,q,r,s\}$

**and**  $p \neq q$  **and**  $r \neq p$  **and**  $s \neq p$  **and**  $r \neq q$

**by** (*unfold cross-ratio-correct-def*) *simp-all*

**have**  $\{\textit{apply-cltn2}\ t\ C \mid t \in \{p,q,r,s\}\} = \{?pC, ?qC, ?rC, ?sC\}$  **by** *auto*

**with**  $\langle \textit{proj2-set-Col}\ \{p,q,r,s\} \rangle$

**and** *apply-cltn2-preserve-set-Col* [*of*  $\{p,q,r,s\}\ C$ ]

**have** *proj2-set-Col*  $\{?pC, ?qC, ?rC, ?sC\}$  **by** *simp*

**from**  $\langle p \neq q \rangle$  **and**  $\langle r \neq p \rangle$  **and**  $\langle s \neq p \rangle$  **and**  $\langle r \neq q \rangle$  **and** *apply-cltn2-injective*

**have**  $?pC \neq ?qC$  **and**  $?rC \neq ?pC$  **and**  $?sC \neq ?pC$  **and**  $?rC \neq ?qC$  **by** *fast+*

**with**  $\langle \textit{proj2-set-Col}\ \{?pC, ?qC, ?rC, ?sC\} \rangle$

**show** *cross-ratio-correct*  $?pC\ ?qC\ ?rC\ ?sC$

**by** (*unfold cross-ratio-correct-def*) *simp*

qed

**lemma** *cross-ratio-cltn2*:

**assumes** *proj2-set-Col*  $\{p,q,r,s\}$  **and**  $p \neq q$  **and**  $r \neq p$  **and**  $s \neq p$

**shows** *cross-ratio* (*apply-cltn2*  $p\ C$ ) (*apply-cltn2*  $q\ C$ )

(*apply-cltn2*  $r\ C$ ) (*apply-cltn2*  $s\ C$ )

$=$  *cross-ratio*  $p\ q\ r\ s$

(**is** *cross-ratio*  $?pC\ ?qC\ ?rC\ ?sC = -$ )

**proof** –

**let**  $?u = \textit{proj2-rep}\ p$

**let**  $?v = \textit{proj2-rep}\ q$

**let**  $?i = \textit{proj2-Col-coeff}\ p\ q\ r$

**let**  $?j = \textit{proj2-Col-coeff}\ p\ q\ s$

**from**  $\langle \textit{proj2-set-Col}\ \{p,q,r,s\} \rangle$  **and**  $\langle p \neq q \rangle$  **and**  $\langle r \neq p \rangle$  **and**  $\langle s \neq p \rangle$

**and** *proj2-set-Col-coeff*

**have**  $r = \textit{proj2-abs}\ (?i *_{\mathbb{R}} ?u + ?v)$  **and**  $s = \textit{proj2-abs}\ (?j *_{\mathbb{R}} ?u + ?v)$

**by** *simp-all*

**let**  $?uC = ?u\ v * \textit{cltn2-rep}\ C$

**let**  $?vC = ?v\ v * \textit{cltn2-rep}\ C$

**have**  $?uC \neq 0$  **and**  $?vC \neq 0$  **by** (*rule rep-mult-rep-non-zero*) $+$

**have**  $\textit{proj2-abs}\ ?uC = ?pC$  **and**  $\textit{proj2-abs}\ ?vC = ?qC$

**by** (*unfold apply-cltn2-def*) *simp-all*

**from**  $\langle p \neq q \rangle$  **and** *apply-cltn2-injective* **have**  $?pC \neq ?qC$  **by** *fast*

**from**  $\langle p \neq q \rangle$  **and** *proj2-rep-dependent* [of - p 1 q]  
**have**  $?i *_{\mathbb{R}} ?u + ?v \neq 0$  **and**  $?j *_{\mathbb{R}} ?u + ?v \neq 0$  **by** *auto*  
**with**  $\langle r = \text{proj2-abs } (?i *_{\mathbb{R}} ?u + ?v) \rangle$  **and**  $\langle s = \text{proj2-abs } (?j *_{\mathbb{R}} ?u + ?v) \rangle$   
**and** *apply-cltn2-linear* [of ?i ?u 1 ?v]  
**and** *apply-cltn2-linear* [of ?j ?u 1 ?v]  
**have**  $?rC = \text{proj2-abs } (?i *_{\mathbb{R}} ?uC + ?vC)$   
**and**  $?sC = \text{proj2-abs } (?j *_{\mathbb{R}} ?uC + ?vC)$   
**by** *simp-all*  
**with**  $\langle ?uC \neq 0 \rangle$  **and**  $\langle ?vC \neq 0 \rangle$  **and**  $\langle \text{proj2-abs } ?uC = ?pC \rangle$   
**and**  $\langle \text{proj2-abs } ?vC = ?qC \rangle$  **and**  $\langle ?pC \neq ?qC \rangle$   
**and** *cross-ratio-abs* [of ?uC ?vC 1 1 ?i ?j]  
**have** *cross-ratio* ?pC ?qC ?rC ?sC = ?j/?i **by** *simp*  
**thus** *cross-ratio* ?pC ?qC ?rC ?sC = *cross-ratio* p q r s  
**unfolding** *cross-ratio-def* [of p q r s] .  
**qed**

**lemma** *cross-ratio-unique*:

**assumes** *cross-ratio-correct* p q r s **and** *cross-ratio-correct* p q r t  
**and** *cross-ratio* p q r s = *cross-ratio* p q r t  
**shows** s = t

**proof** –

**from**  $\langle \text{cross-ratio-correct } p \ q \ r \ s \rangle$  **and**  $\langle \text{cross-ratio-correct } p \ q \ r \ t \rangle$   
**have** *proj2-set-Col* {p,q,r,s} **and** *proj2-set-Col* {p,q,r,t}  
**and**  $p \neq q$  **and**  $r \neq p$  **and**  $r \neq q$  **and**  $s \neq p$  **and**  $t \neq p$   
**by** (*unfold cross-ratio-correct-def*) *simp-all*

**let** ?u = *proj2-rep* p  
**let** ?v = *proj2-rep* q  
**let** ?i = *proj2-Col-coeff* p q r  
**let** ?j = *proj2-Col-coeff* p q s  
**let** ?k = *proj2-Col-coeff* p q t  
**from**  $\langle \text{proj2-set-Col } \{p,q,r,s\} \rangle$  **and**  $\langle \text{proj2-set-Col } \{p,q,r,t\} \rangle$   
**and**  $\langle p \neq q \rangle$  **and**  $\langle r \neq p \rangle$  **and**  $\langle s \neq p \rangle$  **and**  $\langle t \neq p \rangle$  **and** *proj2-set-Col-coeff*  
**have**  $r = \text{proj2-abs } (?i *_{\mathbb{R}} ?u + ?v)$   
**and**  $s = \text{proj2-abs } (?j *_{\mathbb{R}} ?u + ?v)$   
**and**  $t = \text{proj2-abs } (?k *_{\mathbb{R}} ?u + ?v)$   
**by** *simp-all*

**from**  $\langle r \neq q \rangle$  **and**  $\langle r = \text{proj2-abs } (?i *_{\mathbb{R}} ?u + ?v) \rangle$   
**have** ?i  $\neq 0$  **by** (*auto simp add: proj2-abs-rep*)  
**with**  $\langle \text{cross-ratio } p \ q \ r \ s = \text{cross-ratio } p \ q \ r \ t \rangle$   
**have** ?j = ?k **by** (*unfold cross-ratio-def*) *simp*  
**with**  $\langle s = \text{proj2-abs } (?j *_{\mathbb{R}} ?u + ?v) \rangle$  **and**  $\langle t = \text{proj2-abs } (?k *_{\mathbb{R}} ?u + ?v) \rangle$   
**show** s = t **by** *simp*

**qed**

**lemma** *cltn2-three-point-line*:

**assumes**  $p \neq q$  **and**  $r \neq p$  **and**  $r \neq q$

**and** *proj2-incident*  $p\ l$  **and** *proj2-incident*  $q\ l$  **and** *proj2-incident*  $r\ l$   
**and** *apply-cltn2*  $p\ C = p$  **and** *apply-cltn2*  $q\ C = q$  **and** *apply-cltn2*  $r\ C = r$   
**and** *proj2-incident*  $s\ l$   
**shows** *apply-cltn2*  $s\ C = s$  (**is**  $?sC = s$ )

**proof** *cases*

**assume**  $s = p$   
**with**  $\langle \text{apply-cltn2 } p\ C = p \rangle$  **show**  $?sC = s$  **by** *simp*  
**next**  
**assume**  $s \neq p$

**let**  $?pC = \text{apply-cltn2 } p\ C$   
**let**  $?qC = \text{apply-cltn2 } q\ C$   
**let**  $?rC = \text{apply-cltn2 } r\ C$

**from**  $\langle \text{proj2-incident } p\ l \rangle$  **and**  $\langle \text{proj2-incident } q\ l \rangle$  **and**  $\langle \text{proj2-incident } r\ l \rangle$   
**and**  $\langle \text{proj2-incident } s\ l \rangle$   
**have** *proj2-set-Col*  $\{p, q, r, s\}$  **by** (*unfold proj2-set-Col-def*) *auto*  
**with**  $\langle p \neq q \rangle$  **and**  $\langle r \neq p \rangle$  **and**  $\langle s \neq p \rangle$  **and**  $\langle r \neq q \rangle$   
**have** *cross-ratio-correct*  $p\ q\ r\ s$  **by** (*unfold cross-ratio-correct-def*) *simp*  
**hence** *cross-ratio-correct*  $?pC\ ?qC\ ?rC\ ?sC$   
**by** (*rule cross-ratio-correct-cltn2*)  
**with**  $\langle ?pC = p \rangle$  **and**  $\langle ?qC = q \rangle$  **and**  $\langle ?rC = r \rangle$   
**have** *cross-ratio-correct*  $p\ q\ r\ ?sC$  **by** *simp*

**from**  $\langle \text{proj2-set-Col } \{p, q, r, s\} \rangle$  **and**  $\langle p \neq q \rangle$  **and**  $\langle r \neq p \rangle$  **and**  $\langle s \neq p \rangle$   
**have** *cross-ratio*  $?pC\ ?qC\ ?rC\ ?sC = \text{cross-ratio } p\ q\ r\ s$   
**by** (*rule cross-ratio-cltn2*)  
**with**  $\langle ?pC = p \rangle$  **and**  $\langle ?qC = q \rangle$  **and**  $\langle ?rC = r \rangle$   
**have** *cross-ratio*  $p\ q\ r\ ?sC = \text{cross-ratio } p\ q\ r\ s$  **by** *simp*  
**with**  $\langle \text{cross-ratio-correct } p\ q\ r\ ?sC \rangle$  **and**  $\langle \text{cross-ratio-correct } p\ q\ r\ s \rangle$   
**show**  $?sC = s$  **by** (*rule cross-ratio-unique*)

**qed**

**lemma** *cross-ratio-equal-cltn2*:

**assumes** *cross-ratio-correct*  $p\ q\ r\ s$   
**and** *cross-ratio-correct*  $(\text{apply-cltn2 } p\ C)\ (\text{apply-cltn2 } q\ C)$   
 $(\text{apply-cltn2 } r\ C)\ t$   
**(is** *cross-ratio-correct*  $?pC\ ?qC\ ?rC\ t$ )  
**and** *cross-ratio*  $(\text{apply-cltn2 } p\ C)\ (\text{apply-cltn2 } q\ C)\ (\text{apply-cltn2 } r\ C)\ t$   
 $= \text{cross-ratio } p\ q\ r\ s$   
**shows**  $t = \text{apply-cltn2 } s\ C$  (**is**  $t = ?sC$ )

**proof** –

**from**  $\langle \text{cross-ratio-correct } p\ q\ r\ s \rangle$   
**have** *cross-ratio-correct*  $?pC\ ?qC\ ?rC\ ?sC$  **by** (*rule cross-ratio-correct-cltn2*)

**from**  $\langle \text{cross-ratio-correct } p\ q\ r\ s \rangle$   
**have** *proj2-set-Col*  $\{p, q, r, s\}$  **and**  $p \neq q$  **and**  $r \neq p$  **and**  $s \neq p$   
**by** (*unfold cross-ratio-correct-def*) *simp-all*  
**hence** *cross-ratio*  $?pC\ ?qC\ ?rC\ ?sC = \text{cross-ratio } p\ q\ r\ s$

by (rule cross-ratio-cltn2)  
 with  $\langle \text{cross-ratio } ?pC \ ?qC \ ?rC \ t = \text{cross-ratio } p \ q \ r \ s \rangle$   
 have  $\text{cross-ratio } ?pC \ ?qC \ ?rC \ t = \text{cross-ratio } ?pC \ ?qC \ ?rC \ ?sC$  by simp  
 with  $\langle \text{cross-ratio-correct } ?pC \ ?qC \ ?rC \ t \rangle$   
 and  $\langle \text{cross-ratio-correct } ?pC \ ?qC \ ?rC \ ?sC \rangle$   
 show  $t = ?sC$  by (rule cross-ratio-unique)  
 qed

**lemma** *proj2-Col-distinct-coeff-non-zero*:  
 assumes *proj2-Col*  $p \ q \ r$  and  $p \neq q$  and  $r \neq p$  and  $r \neq q$   
 shows *proj2-Col-coeff*  $p \ q \ r \neq 0$   
**proof**  
 assume *proj2-Col-coeff*  $p \ q \ r = 0$

from  $\langle \text{proj2-Col } p \ q \ r \rangle$  and  $\langle p \neq q \rangle$  and  $\langle r \neq p \rangle$   
 have  $r = \text{proj2-abs } ((\text{proj2-Col-coeff } p \ q \ r) *_{\mathbb{R}} \text{proj2-rep } p + \text{proj2-rep } q)$   
 by (rule *proj2-Col-coeff*)  
 with  $\langle \text{proj2-Col-coeff } p \ q \ r = 0 \rangle$  have  $r = q$  by (simp add: *proj2-abs-rep*)  
 with  $\langle r \neq q \rangle$  show *False* ..  
 qed

**lemma** *cross-ratio-product*:  
 assumes *proj2-Col*  $p \ q \ s$  and  $p \neq q$  and  $s \neq p$  and  $s \neq q$   
 shows  $\text{cross-ratio } p \ q \ r \ s * \text{cross-ratio } p \ q \ s \ t = \text{cross-ratio } p \ q \ r \ t$   
**proof** –  
 from  $\langle \text{proj2-Col } p \ q \ s \rangle$  and  $\langle p \neq q \rangle$  and  $\langle s \neq p \rangle$  and  $\langle s \neq q \rangle$   
 have *proj2-Col-coeff*  $p \ q \ s \neq 0$  by (rule *proj2-Col-distinct-coeff-non-zero*)  
 thus  $\text{cross-ratio } p \ q \ r \ s * \text{cross-ratio } p \ q \ s \ t = \text{cross-ratio } p \ q \ r \ t$   
 by (*unfold cross-ratio-def*) simp  
 qed

**lemma** *cross-ratio-equal-1*:  
 assumes *proj2-Col*  $p \ q \ r$  and  $p \neq q$  and  $r \neq p$  and  $r \neq q$   
 shows  $\text{cross-ratio } p \ q \ r \ r = 1$   
**proof** –  
 from  $\langle \text{proj2-Col } p \ q \ r \rangle$  and  $\langle p \neq q \rangle$  and  $\langle r \neq p \rangle$  and  $\langle r \neq q \rangle$   
 have *proj2-Col-coeff*  $p \ q \ r \neq 0$  by (rule *proj2-Col-distinct-coeff-non-zero*)  
 thus  $\text{cross-ratio } p \ q \ r \ r = 1$  by (*unfold cross-ratio-def*) simp  
 qed

**lemma** *cross-ratio-1-equal*:  
 assumes *cross-ratio-correct*  $p \ q \ r \ s$  and  $\text{cross-ratio } p \ q \ r \ s = 1$   
 shows  $r = s$   
**proof** –  
 from  $\langle \text{cross-ratio-correct } p \ q \ r \ s \rangle$   
 have *proj2-set-Col*  $\{p, q, r, s\}$  and  $p \neq q$  and  $r \neq p$  and  $r \neq q$   
 by (*unfold cross-ratio-correct-def*) simp-all

from  $\langle \text{proj2-set-Col } \{p, q, r, s\} \rangle$

**have** *proj2-set-Col* {*p,q,r*}  
**by** (*simp add: proj2-subset-Col [of {p,q,r} {p,q,r,s}]*)  
**with**  $\langle p \neq q \rangle$  **and**  $\langle r \neq p \rangle$  **and**  $\langle r \neq q \rangle$   
**have** *cross-ratio-correct* *p q r r* **by** (*unfold cross-ratio-correct-def*) *simp*  
  
**from**  $\langle \text{proj2-set-Col } \{p,q,r\} \rangle$   
**have** *proj2-Col* *p q r* **by** (*subst proj2-Col-iff-set-Col*)  
**with**  $\langle p \neq q \rangle$  **and**  $\langle r \neq p \rangle$  **and**  $\langle r \neq q \rangle$   
**have** *cross-ratio* *p q r r = 1* **by** (*simp add: cross-ratio-equal-1*)  
**with**  $\langle \text{cross-ratio } p q r s = 1 \rangle$   
**have** *cross-ratio* *p q r r = cross-ratio p q r s* **by** *simp*  
**with**  $\langle \text{cross-ratio-correct } p q r r \rangle$  **and**  $\langle \text{cross-ratio-correct } p q r s \rangle$   
**show**  $r = s$  **by** (*rule cross-ratio-unique*)  
**qed**

**lemma** *cross-ratio-swap-34*:  
**shows** *cross-ratio* *p q s r = 1 / (cross-ratio p q r s)*  
**by** (*unfold cross-ratio-def*) *simp*

**lemma** *cross-ratio-swap-13-24*:  
**assumes** *cross-ratio-correct* *p q r s* **and**  $r \neq s$   
**shows** *cross-ratio* *r s p q = cross-ratio p q r s*  
**proof** –  
**from**  $\langle \text{cross-ratio-correct } p q r s \rangle$   
**have** *proj2-set-Col* {*p,q,r,s*} **and**  $p \neq q$  **and**  $r \neq p$  **and**  $s \neq p$  **and**  $r \neq q$   
**by** (*unfold cross-ratio-correct-def, simp-all*)  
  
**have** *proj2-rep* *p*  $\neq 0$  (**is**  $?u \neq 0$ ) **and** *proj2-rep* *q*  $\neq 0$  (**is**  $?v \neq 0$ )  
**by** (*rule proj2-rep-non-zero*)**+**

**have**  $p = \text{proj2-abs } ?u$  **and**  $q = \text{proj2-abs } ?v$   
**by** (*simp-all add: proj2-abs-rep*)  
**with**  $\langle p \neq q \rangle$  **have**  $\text{proj2-abs } ?u \neq \text{proj2-abs } ?v$  **by** *simp*

**let**  $?i = \text{proj2-Col-coeff } p q r$   
**let**  $?j = \text{proj2-Col-coeff } p q s$   
**from**  $\langle \text{proj2-set-Col } \{p,q,r,s\} \rangle$  **and**  $\langle p \neq q \rangle$  **and**  $\langle r \neq p \rangle$  **and**  $\langle s \neq p \rangle$   
**have**  $r = \text{proj2-abs } (?i *_{\mathbb{R}} ?u + ?v)$  (**is**  $r = \text{proj2-abs } ?w$ )  
**and**  $s = \text{proj2-abs } (?j *_{\mathbb{R}} ?u + ?v)$  (**is**  $s = \text{proj2-abs } ?x$ )  
**by** (*simp-all add: proj2-set-Col-coeff*)  
**with**  $\langle r \neq s \rangle$  **have**  $?i \neq ?j$  **by** *auto*

**from**  $\langle ?u \neq 0 \rangle$  **and**  $\langle ?v \neq 0 \rangle$  **and**  $\langle \text{proj2-abs } ?u \neq \text{proj2-abs } ?v \rangle$   
**and** *dependent-proj2-abs [of ?u ?v - 1]*  
**have**  $?w \neq 0$  **and**  $?x \neq 0$  **by** *auto*

**from**  $\langle r = \text{proj2-abs } (?i *_{\mathbb{R}} ?u + ?v) \rangle$  **and**  $\langle r \neq q \rangle$   
**have**  $?i \neq 0$  **by** (*auto simp add: proj2-abs-rep*)

**have**  $?w - ?x = (?i - ?j) *_{\mathbb{R}} ?u$  **by** (*simp add: algebra-simps*)  
**with**  $\langle ?i \neq ?j \rangle$   
**have**  $p = \text{proj2-abs } (?w - ?x)$  **by** (*simp add: proj2-abs-mult-rep*)  
  
**have**  $?j *_{\mathbb{R}} ?w - ?i *_{\mathbb{R}} ?x = (?j - ?i) *_{\mathbb{R}} ?v$  **by** (*simp add: algebra-simps*)  
**with**  $\langle ?i \neq ?j \rangle$   
**have**  $q = \text{proj2-abs } (?j *_{\mathbb{R}} ?w - ?i *_{\mathbb{R}} ?x)$  **by** (*simp add: proj2-abs-mult-rep*)  
**with**  $\langle ?w \neq 0 \rangle$  **and**  $\langle ?x \neq 0 \rangle$  **and**  $\langle r \neq s \rangle$  **and**  $\langle ?i \neq 0 \rangle$  **and**  $\langle r = \text{proj2-abs } ?w \rangle$   
**and**  $\langle s = \text{proj2-abs } ?x \rangle$  **and**  $\langle p = \text{proj2-abs } (?w - ?x) \rangle$   
**and** *cross-ratio-abs [of ?w ?x -1 -?i 1 ?j]*  
**have** *cross-ratio r s p q = ?j / ?i* **by** (*simp add: algebra-simps*)  
**thus** *cross-ratio r s p q = cross-ratio p q r s*  
**by** (*unfold cross-ratio-def [of p q r s], simp*)  
**qed**

**lemma** *cross-ratio-swap-12:*

**assumes** *cross-ratio-correct p q r s* **and** *cross-ratio-correct q p r s*  
**shows** *cross-ratio q p r s = 1 / (cross-ratio p q r s)*

**proof** *cases*

**assume**  $r = s$

**from**  $\langle \text{cross-ratio-correct } p \ q \ r \ s \rangle$

**have** *proj2-set-Col {p,q,r,s}* **and**  $p \neq q$  **and**  $r \neq p$  **and**  $r \neq q$   
**by** (*unfold cross-ratio-correct-def*) *simp-all*

**from**  $\langle \text{proj2-set-Col } \{p,q,r,s\} \rangle$  **and**  $\langle r = s \rangle$

**have** *proj2-Col p q r* **by** (*simp-all add: proj2-Col-iff-set-Col*)

**hence** *proj2-Col q p r* **by** (*rule proj2-Col-permute*)

**with**  $\langle \text{proj2-Col } p \ q \ r \rangle$  **and**  $\langle p \neq q \rangle$  **and**  $\langle r \neq p \rangle$  **and**  $\langle r \neq q \rangle$  **and**  $\langle r = s \rangle$

**have** *cross-ratio p q r s = 1* **and** *cross-ratio q p r s = 1*

**by** (*simp-all add: cross-ratio-equal-1*)

**thus** *cross-ratio q p r s = 1 / (cross-ratio p q r s)* **by** *simp*

**next**

**assume**  $r \neq s$

**with**  $\langle \text{cross-ratio-correct } q \ p \ r \ s \rangle$

**have** *cross-ratio q p r s = cross-ratio r s q p*

**by** (*simp add: cross-ratio-swap-13-24*)

**also have**  $\dots = 1 / (\text{cross-ratio } r \ s \ p \ q)$  **by** (*rule cross-ratio-swap-34*)

**also from**  $\langle \text{cross-ratio-correct } p \ q \ r \ s \rangle$  **and**  $\langle r \neq s \rangle$

**have**  $\dots = 1 / (\text{cross-ratio } p \ q \ r \ s)$  **by** (*simp add: cross-ratio-swap-13-24*)

**finally show** *cross-ratio q p r s = 1 / (cross-ratio p q r s)* .

**qed**

## 7.6 Cartesian subspace of the real projective plane

**definition** *vector2-append1* ::  $\text{real}^2 \Rightarrow \text{real}^3$  **where**

*vector2-append1*  $v = \text{vector } [v\$1, v\$2, 1]$

**lemma** *vector2-append1-non-zero*: *vector2-append1 v*  $\neq 0$

**proof** –

**have** (*vector2-append1 v*)  $\neq 0$

**unfolding** *vector2-append1-def* **and** *vector-def*

**by** *simp*

**thus** *vector2-append1 v*  $\neq 0$  **by** *auto*

**qed**

**definition** *proj2-pt* ::  $\text{real}^2 \Rightarrow \text{proj2}$  **where**

*proj2-pt v*  $\triangleq \text{proj2-abs } (\text{vector2-append1 } v)$

**lemma** *proj2-pt-scalar*:

$\exists c. c \neq 0 \wedge \text{proj2-rep } (\text{proj2-pt } v) = c *_R \text{vector2-append1 } v$

**unfolding** *proj2-pt-def*

**by** (*simp add: proj2-rep-abs2 vector2-append1-non-zero*)

**abbreviation** *z-non-zero* ::  $\text{proj2} \Rightarrow \text{bool}$  **where**

*z-non-zero p*  $\triangleq (\text{proj2-rep } p) \neq 0$

**definition** *cart2-pt* ::  $\text{proj2} \Rightarrow \text{real}^2$  **where**

*cart2-pt p*  $\triangleq$

$\text{vector } [(\text{proj2-rep } p)_1 / (\text{proj2-rep } p)_3, (\text{proj2-rep } p)_2 / (\text{proj2-rep } p)_3]$

**definition** *cart2-append1* ::  $\text{proj2} \Rightarrow \text{real}^3$  **where**

*cart2-append1 p*  $\triangleq (1 / ((\text{proj2-rep } p)_3)) *_R \text{proj2-rep } p$

**lemma** *cart2-append1-z*:

**assumes** *z-non-zero p*

**shows** (*cart2-append1 p*)  $= 1$

**using**  $\langle z\text{-non-zero } p \rangle$

**by** (*unfold cart2-append1-def simp*)

**lemma** *cart2-append1-non-zero*:

**assumes** *z-non-zero p*

**shows** *cart2-append1 p*  $\neq 0$

**proof** –

**from**  $\langle z\text{-non-zero } p \rangle$  **have** (*cart2-append1 p*)  $= 1$  **by** (*rule cart2-append1-z*)

**thus** *cart2-append1 p*  $\neq 0$  **by** (*simp add: vec-eq-iff exI [of - 3]*)

**qed**

**lemma** *proj2-rep-cart2-append1*:

**assumes** *z-non-zero p*

**shows**  $\text{proj2-rep } p = ((\text{proj2-rep } p)_3) *_R \text{cart2-append1 } p$

**using**  $\langle z\text{-non-zero } p \rangle$

**by** (*unfold cart2-append1-def simp*)

**lemma** *proj2-abs-cart2-append1*:

**assumes** *z-non-zero p*

**shows**  $\text{proj2-abs } (\text{cart2-append1 } p) = p$

**proof** –  
**from**  $\langle z\text{-non-zero } p \rangle$   
**have**  $\text{proj2-abs } (\text{cart2-append1 } p) = \text{proj2-abs } (\text{proj2-rep } p)$   
**by**  $(\text{unfold cart2-append1-def}) (\text{simp add: proj2-abs-mult})$   
**thus**  $\text{proj2-abs } (\text{cart2-append1 } p) = p$  **by**  $(\text{simp add: proj2-abs-rep})$   
**qed**

**lemma** *cart2-append1-inj*:  
**assumes**  $z\text{-non-zero } p$  **and**  $\text{cart2-append1 } p = \text{cart2-append1 } q$   
**shows**  $p = q$

**proof** –  
**from**  $\langle z\text{-non-zero } p \rangle$  **have**  $(\text{cart2-append1 } p)\$3 = 1$  **by**  $(\text{rule cart2-append1-z})$   
**with**  $\langle \text{cart2-append1 } p = \text{cart2-append1 } q \rangle$   
**have**  $(\text{cart2-append1 } q)\$3 = 1$  **by** *simp*  
**hence**  $z\text{-non-zero } q$  **by**  $(\text{unfold cart2-append1-def})$  *auto*  
  
**from**  $\langle \text{cart2-append1 } p = \text{cart2-append1 } q \rangle$   
**have**  $\text{proj2-abs } (\text{cart2-append1 } p) = \text{proj2-abs } (\text{cart2-append1 } q)$  **by** *simp*  
**with**  $\langle z\text{-non-zero } p \rangle$  **and**  $\langle z\text{-non-zero } q \rangle$   
**show**  $p = q$  **by**  $(\text{simp add: proj2-abs-cart2-append1})$   
**qed**

**lemma** *cart2-append1*:  
**assumes**  $z\text{-non-zero } p$   
**shows**  $\text{vector2-append1 } (\text{cart2-pt } p) = \text{cart2-append1 } p$   
**using**  $\langle z\text{-non-zero } p \rangle$   
**unfolding** *vector2-append1-def*  
**and** *cart2-append1-def*  
**and** *cart2-pt-def*  
**and** *vector-def*  
**by**  $(\text{simp add: vec-eq-iff forall-3})$

**lemma** *cart2-proj2*:  $\text{cart2-pt } (\text{proj2-pt } v) = v$

**proof** –  
**let**  $?v' = \text{vector2-append1 } v$   
**let**  $?p = \text{proj2-pt } v$   
**from** *proj2-pt-scalar*  
**obtain**  $c$  **where**  $c \neq 0$  **and**  $\text{proj2-rep } ?p = c *_{\mathbb{R}} ?v'$  **by** *auto*  
**hence**  $(\text{cart2-pt } ?p)\$1 = v\$1$  **and**  $(\text{cart2-pt } ?p)\$2 = v\$2$   
**unfolding** *cart2-pt-def* **and** *vector2-append1-def* **and** *vector-def*  
**by** *simp+*  
**thus**  $\text{cart2-pt } ?p = v$  **by**  $(\text{simp add: vec-eq-iff forall-2})$   
**qed**

**lemma** *z-non-zero-proj2-pt*:  $z\text{-non-zero } (\text{proj2-pt } v)$

**proof** –  
**from** *proj2-pt-scalar*  
**obtain**  $c$  **where**  $c \neq 0$  **and**  $\text{proj2-rep } (\text{proj2-pt } v) = c *_{\mathbb{R}} (\text{vector2-append1 } v)$   
**by** *auto*

**from**  $\langle \text{proj2-rep } (\text{proj2-pt } v) = c *_{\mathbb{R}} (\text{vector2-append1 } v) \rangle$   
**have**  $(\text{proj2-rep } (\text{proj2-pt } v))\$3 = c$   
**unfolding** *vector2-append1-def* **and** *vector-def*  
**by** *simp*  
**with**  $\langle c \neq 0 \rangle$  **show** *z-non-zero*  $(\text{proj2-pt } v)$  **by** *simp*  
**qed**

**lemma** *cart2-append1-proj2*:  $\text{cart2-append1 } (\text{proj2-pt } v) = \text{vector2-append1 } v$   
**proof** –  
**from** *z-non-zero-proj2-pt*  
**have**  $\text{cart2-append1 } (\text{proj2-pt } v) = \text{vector2-append1 } (\text{cart2-pt } (\text{proj2-pt } v))$   
**by** *(simp add: cart2-append1)*  
**thus**  $\text{cart2-append1 } (\text{proj2-pt } v) = \text{vector2-append1 } v$   
**by** *(simp add: cart2-proj2)*  
**qed**

**lemma** *proj2-pt-inj*: *inj* *proj2-pt*  
**by** *(simp add: inj-on-inverseI [of UNIV cart2-pt proj2-pt] cart2-proj2)*

**lemma** *proj2-cart2*:  
**assumes** *z-non-zero*  $p$   
**shows**  $\text{proj2-pt } (\text{cart2-pt } p) = p$   
**proof** –  
**from**  $\langle \text{z-non-zero } p \rangle$   
**have**  $(\text{proj2-rep } p)\$3 *_{\mathbb{R}} \text{vector2-append1 } (\text{cart2-pt } p) = \text{proj2-rep } p$   
**unfolding** *vector2-append1-def* **and** *cart2-pt-def* **and** *vector-def*  
**by** *(simp add: vec-eq-iff forall-3)*  
**with**  $\langle \text{z-non-zero } p \rangle$   
**and** *proj2-abs-mult [of (proj2-rep p)\\$3 vector2-append1 (cart2-pt p)]*  
**have**  $\text{proj2-abs } (\text{vector2-append1 } (\text{cart2-pt } p)) = \text{proj2-abs } (\text{proj2-rep } p)$   
**by** *simp*  
**thus**  $\text{proj2-pt } (\text{cart2-pt } p) = p$   
**by** *(unfold proj2-pt-def) (simp add: proj2-abs-rep)*  
**qed**

**lemma** *cart2-injective*:  
**assumes** *z-non-zero*  $p$  **and** *z-non-zero*  $q$  **and**  $\text{cart2-pt } p = \text{cart2-pt } q$   
**shows**  $p = q$   
**proof** –  
**from**  $\langle \text{z-non-zero } p \rangle$  **and**  $\langle \text{z-non-zero } q \rangle$   
**have**  $\text{proj2-pt } (\text{cart2-pt } p) = p$  **and**  $\text{proj2-pt } (\text{cart2-pt } q) = q$   
**by** *(simp-all add: proj2-cart2)*  
  
**from**  $\langle \text{proj2-pt } (\text{cart2-pt } p) = p \rangle$  **and**  $\langle \text{cart2-pt } p = \text{cart2-pt } q \rangle$   
**have**  $\text{proj2-pt } (\text{cart2-pt } q) = p$  **by** *simp*  
**with**  $\langle \text{proj2-pt } (\text{cart2-pt } q) = q \rangle$  **show**  $p = q$  **by** *simp*  
**qed**

**lemma** *proj2-Col-iff-euclid*:

$proj2\text{-}Col (proj2\text{-}pt\ a) (proj2\text{-}pt\ b) (proj2\text{-}pt\ c) \longleftrightarrow real\text{-}euclid.Col\ a\ b\ c$   
(is  $proj2\text{-}Col\ ?p\ ?q\ ?r \longleftrightarrow -$ )

**proof**

let  $?a' = vector2\text{-}append1\ a$   
let  $?b' = vector2\text{-}append1\ b$   
let  $?c' = vector2\text{-}append1\ c$   
let  $?a'' = proj2\text{-}rep\ ?p$   
let  $?b'' = proj2\text{-}rep\ ?q$   
let  $?c'' = proj2\text{-}rep\ ?r$

from  $proj2\text{-}pt\text{-}scalar$  obtain  $i$  and  $j$  and  $k$  where

$i \neq 0$  and  $?a'' = i *_{\mathbb{R}} ?a'$   
and  $j \neq 0$  and  $?b'' = j *_{\mathbb{R}} ?b'$   
and  $k \neq 0$  and  $?c'' = k *_{\mathbb{R}} ?c'$   
by *metis*

hence  $?a' = (1/i) *_{\mathbb{R}} ?a''$   
and  $?b' = (1/j) *_{\mathbb{R}} ?b''$   
and  $?c' = (1/k) *_{\mathbb{R}} ?c''$   
by *simp-all*

{ assume  $proj2\text{-}Col\ ?p\ ?q\ ?r$

then obtain  $i'$  and  $j'$  and  $k'$  where

$i' *_{\mathbb{R}} ?a'' + j' *_{\mathbb{R}} ?b'' + k' *_{\mathbb{R}} ?c'' = 0$  and  $i' \neq 0 \vee j' \neq 0 \vee k' \neq 0$   
unfolding  $proj2\text{-}Col\text{-}def$   
by *auto*

let  $?i'' = i * i'$   
let  $?j'' = j * j'$   
let  $?k'' = k * k'$   
from  $\langle i \neq 0 \rangle$  and  $\langle j \neq 0 \rangle$  and  $\langle k \neq 0 \rangle$  and  $\langle i' \neq 0 \vee j' \neq 0 \vee k' \neq 0 \rangle$   
have  $?i'' \neq 0 \vee ?j'' \neq 0 \vee ?k'' \neq 0$  by *simp*

from  $\langle i' *_{\mathbb{R}} ?a'' + j' *_{\mathbb{R}} ?b'' + k' *_{\mathbb{R}} ?c'' = 0 \rangle$   
and  $\langle ?a'' = i *_{\mathbb{R}} ?a' \rangle$   
and  $\langle ?b'' = j *_{\mathbb{R}} ?b' \rangle$   
and  $\langle ?c'' = k *_{\mathbb{R}} ?c' \rangle$

have  $?i'' *_{\mathbb{R}} ?a' + ?j'' *_{\mathbb{R}} ?b' + ?k'' *_{\mathbb{R}} ?c' = 0$   
by (*simp add: ac-simps*)

hence  $(?i'' *_{\mathbb{R}} ?a' + ?j'' *_{\mathbb{R}} ?b' + ?k'' *_{\mathbb{R}} ?c')\$3 = 0$   
by *simp*

hence  $?i'' + ?j'' + ?k'' = 0$   
unfolding  $vector2\text{-}append1\text{-}def$  and  $vector\text{-}def$   
by *simp*

have  $(?i'' *_{\mathbb{R}} ?a' + ?j'' *_{\mathbb{R}} ?b' + ?k'' *_{\mathbb{R}} ?c')\$1 =$   
 $(?i'' *_{\mathbb{R}} a + ?j'' *_{\mathbb{R}} b + ?k'' *_{\mathbb{R}} c)\$1$   
and  $(?i'' *_{\mathbb{R}} ?a' + ?j'' *_{\mathbb{R}} ?b' + ?k'' *_{\mathbb{R}} ?c')\$2 =$   
 $(?i'' *_{\mathbb{R}} a + ?j'' *_{\mathbb{R}} b + ?k'' *_{\mathbb{R}} c)\$2$   
unfolding  $vector2\text{-}append1\text{-}def$  and  $vector\text{-}def$   
by *simp+*

**with**  $\langle ?i'' *_{\mathbb{R}} ?a' + ?j'' *_{\mathbb{R}} ?b' + ?k'' *_{\mathbb{R}} ?c' = 0 \rangle$   
**have**  $?i'' *_{\mathbb{R}} a + ?j'' *_{\mathbb{R}} b + ?k'' *_{\mathbb{R}} c = 0$   
**by** (*simp add: vec-eq-iff forall-2*)

**have**  $dep2 (b - a) (c - a)$

**proof cases**

**assume**  $?k'' = 0$   
**with**  $\langle ?i'' + ?j'' + ?k'' = 0 \rangle$  **have**  $?j'' = -?i''$  **by** *simp*  
**with**  $\langle ?i'' \neq 0 \vee ?j'' \neq 0 \vee ?k'' \neq 0 \rangle$  **and**  $\langle ?k'' = 0 \rangle$  **have**  $?i'' \neq 0$  **by** *simp*

**from**  $\langle ?i'' *_{\mathbb{R}} a + ?j'' *_{\mathbb{R}} b + ?k'' *_{\mathbb{R}} c = 0 \rangle$   
**and**  $\langle ?k'' = 0 \rangle$  **and**  $\langle ?j'' = -?i'' \rangle$   
**have**  $?i'' *_{\mathbb{R}} a + (-?i'' *_{\mathbb{R}} b) = 0$  **by** *simp*  
**with**  $\langle ?i'' \neq 0 \rangle$  **have**  $a = b$  **by** (*simp add: algebra-simps*)  
**hence**  $b - a = 0 *_{\mathbb{R}} (c - a)$  **by** *simp*  
**moreover** **have**  $c - a = 1 *_{\mathbb{R}} (c - a)$  **by** *simp*  
**ultimately** **have**  $\exists x t s. b - a = t *_{\mathbb{R}} x \wedge c - a = s *_{\mathbb{R}} x$   
**by** *blast*

**thus**  $dep2 (b - a) (c - a)$  **unfolding** *dep2-def* .

**next**

**assume**  $?k'' \neq 0$   
**from**  $\langle ?i'' + ?j'' + ?k'' = 0 \rangle$  **have**  $?i'' = -(?j'' + ?k'')$  **by** *simp*  
**with**  $\langle ?i'' *_{\mathbb{R}} a + ?j'' *_{\mathbb{R}} b + ?k'' *_{\mathbb{R}} c = 0 \rangle$   
**have**  $-(?j'' + ?k'') *_{\mathbb{R}} a + ?j'' *_{\mathbb{R}} b + ?k'' *_{\mathbb{R}} c = 0$  **by** *simp*  
**hence**  $?k'' *_{\mathbb{R}} (c - a) = -?j'' *_{\mathbb{R}} (b - a)$   
**by** (*simp add: scaleR-left-distrib*  
*scaleR-right-diff-distrib*  
*scaleR-left-diff-distrib*  
*algebra-simps*)  
**hence**  $(1 / ?k'') *_{\mathbb{R}} ?k'' *_{\mathbb{R}} (c - a) = (-?j'' / ?k'') *_{\mathbb{R}} (b - a)$   
**by** *simp*  
**with**  $\langle ?k'' \neq 0 \rangle$  **have**  $c - a = (-?j'' / ?k'') *_{\mathbb{R}} (b - a)$  **by** *simp*  
**moreover** **have**  $b - a = 1 *_{\mathbb{R}} (b - a)$  **by** *simp*  
**ultimately** **have**  $\exists x t s. b - a = t *_{\mathbb{R}} x \wedge c - a = s *_{\mathbb{R}} x$  **by** *blast*  
**thus**  $dep2 (b - a) (c - a)$  **unfolding** *dep2-def* .

**qed**

**with** *Col-dep2* **show** *real-euclid.Col a b c* **by** *auto*

**}**

**{** *assume real-euclid.Col a b c*

**with** *Col-dep2* **have**  $dep2 (b - a) (c - a)$  **by** *auto*

**then** **obtain**  $x$  **and**  $t$  **and**  $s$  **where**  $b - a = t *_{\mathbb{R}} x$  **and**  $c - a = s *_{\mathbb{R}} x$

**unfolding** *dep2-def*

**by** *auto*

**show** *proj2-Col ?p ?q ?r*

**proof cases**

**assume**  $t = 0$

**with**  $\langle b - a = t *_{\mathbb{R}} x \rangle$  **have**  $a = b$  **by** *simp*

with *proj2-Col-coincide* show *proj2-Col* ?p ?q ?r by *simp*  
 next  
 assume  $t \neq 0$   
  
 from  $\langle b - a = t *_R x \rangle$  and  $\langle c - a = s *_R x \rangle$   
 have  $s *_R (b - a) = t *_R (c - a)$  by *simp*  
 hence  $(s - t) *_R a + (-s) *_R b + t *_R c = 0$   
   by (*simp* add: *scaleR-right-diff-distrib*  
       *scaleR-left-diff-distrib*  
       *algebra-simps*)  
 hence  $((s - t) *_R ?a' + (-s) *_R ?b' + t *_R ?c')\$1 = 0$   
   and  $((s - t) *_R ?a' + (-s) *_R ?b' + t *_R ?c')\$2 = 0$   
   unfolding *vector2-append1-def* and *vector-def*  
   by (*simp*-all add: *vec-eq-iff*)  
 moreover have  $((s - t) *_R ?a' + (-s) *_R ?b' + t *_R ?c')\$3 = 0$   
   unfolding *vector2-append1-def* and *vector-def*  
   by *simp*  
 ultimately have  $(s - t) *_R ?a' + (-s) *_R ?b' + t *_R ?c' = 0$   
   by (*simp* add: *vec-eq-iff forall-3*)  
 with  $\langle ?a' = (1/i) *_R ?a'' \rangle$   
   and  $\langle ?b' = (1/j) *_R ?b'' \rangle$   
   and  $\langle ?c' = (1/k) *_R ?c'' \rangle$   
 have  $((s - t)/i) *_R ?a'' + (-s/j) *_R ?b'' + (t/k) *_R ?c'' = 0$   
   by *simp*  
 moreover from  $\langle t \neq 0 \rangle$  and  $\langle k \neq 0 \rangle$  have  $t/k \neq 0$  by *simp*  
 ultimately show *proj2-Col* ?p ?q ?r  
   unfolding *proj2-Col-def*  
   by *blast*  
 qed  
 }  
 qed

**lemma** *proj2-Col-iff-euclid-cart2*:  
 assumes *z-non-zero* p and *z-non-zero* q and *z-non-zero* r  
 shows  
*proj2-Col* p q r  $\longleftrightarrow$  *real-euclid.Col* (*cart2-pt* p) (*cart2-pt* q) (*cart2-pt* r)  
 (is -  $\longleftrightarrow$  *real-euclid.Col* ?a ?b ?c)  
**proof** –  
 from  $\langle \text{z-non-zero } p \rangle$  and  $\langle \text{z-non-zero } q \rangle$  and  $\langle \text{z-non-zero } r \rangle$   
 have *proj2-pt* ?a = p and *proj2-pt* ?b = q and *proj2-pt* ?c = r  
   by (*simp*-all add: *proj2-cart2*)  
 with *proj2-Col-iff-euclid* [of ?a ?b ?c]  
 show *proj2-Col* p q r  $\longleftrightarrow$  *real-euclid.Col* ?a ?b ?c by *simp*  
 qed

**lemma** *euclid-Col-cart2-incident*:  
 assumes *z-non-zero* p and *z-non-zero* q and *z-non-zero* r and  $p \neq q$   
 and *proj2-incident* p l and *proj2-incident* q l  
 and *real-euclid.Col* (*cart2-pt* p) (*cart2-pt* q) (*cart2-pt* r)

(is real-euclid.Col ?cp ?cq ?cr)  
 shows proj2-incident r l  
**proof** –  
 from ⟨z-non-zero p⟩ and ⟨z-non-zero q⟩ and ⟨z-non-zero r⟩  
 and ⟨real-euclid.Col ?cp ?cq ?cr⟩  
 have proj2-Col p q r by (subst proj2-Col-iff-euclid-cart2, simp-all)  
 hence proj2-set-Col {p,q,r} by (simp add: proj2-Col-iff-set-Col)  
 then obtain m where  
 proj2-incident p m and proj2-incident q m and proj2-incident r m  
 by (unfold proj2-set-Col-def, auto)  
  
 from ⟨p ≠ q⟩ and ⟨proj2-incident p l⟩ and ⟨proj2-incident q l⟩  
 and ⟨proj2-incident p m⟩ and ⟨proj2-incident q m⟩ and proj2-incident-unique  
 have l = m by auto  
 with ⟨proj2-incident r m⟩ show proj2-incident r l by simp  
**qed**

**lemma** euclid-B-cart2-common-line:  
 assumes z-non-zero p and z-non-zero q and z-non-zero r  
 and  $B_{\mathbb{R}}$  (cart2-pt p) (cart2-pt q) (cart2-pt r)  
 (is  $B_{\mathbb{R}}$  ?cp ?cq ?cr)  
 shows  $\exists l.$  proj2-incident p l  $\wedge$  proj2-incident q l  $\wedge$  proj2-incident r l  
**proof** –  
 from ⟨z-non-zero p⟩ and ⟨z-non-zero q⟩ and ⟨z-non-zero r⟩  
 and ⟨ $B_{\mathbb{R}}$  ?cp ?cq ?cr⟩ and proj2-Col-iff-euclid-cart2  
 have proj2-Col p q r by (unfold real-euclid.Col-def) simp  
 hence proj2-set-Col {p,q,r} by (simp add: proj2-Col-iff-set-Col)  
 thus  $\exists l.$  proj2-incident p l  $\wedge$  proj2-incident q l  $\wedge$  proj2-incident r l  
 by (unfold proj2-set-Col-def) simp  
**qed**

**lemma** cart2-append1-between:  
 assumes z-non-zero p and z-non-zero q and z-non-zero r  
 shows  $B_{\mathbb{R}}$  (cart2-pt p) (cart2-pt q) (cart2-pt r)  
 $\longleftrightarrow (\exists k \geq 0. k \leq 1$   
 $\wedge \text{cart2-append1 } q = k *_{\mathbb{R}} \text{cart2-append1 } r + (1 - k) *_{\mathbb{R}} \text{cart2-append1 } p)$   
**proof** –  
 let ?cp = cart2-pt p  
 let ?cq = cart2-pt q  
 let ?cr = cart2-pt r  
 let ?cp1 = vector2-append1 ?cp  
 let ?cq1 = vector2-append1 ?cq  
 let ?cr1 = vector2-append1 ?cr  
 from ⟨z-non-zero p⟩ and ⟨z-non-zero q⟩ and ⟨z-non-zero r⟩  
 have ?cp1 = cart2-append1 p  
 and ?cq1 = cart2-append1 q  
 and ?cr1 = cart2-append1 r  
 by (simp-all add: cart2-append1)

**have**  $\forall k. ?cq - ?cp = k *_R (?cr - ?cp) \longleftrightarrow ?cq = k *_R ?cr + (1 - k) *_R ?cp$   
**by** (*simp add: algebra-simps*)  
**hence**  $\forall k. ?cq - ?cp = k *_R (?cr - ?cp)$   
 $\longleftrightarrow ?cq1 = k *_R ?cr1 + (1 - k) *_R ?cp1$   
**unfolding** *vector2-append1-def* **and** *vector-def*  
**by** (*simp add: vec-eq-iff forall-2 forall-3*)  
**with**  $\langle ?cp1 = \text{cart2-append1 } p \rangle$   
**and**  $\langle ?cq1 = \text{cart2-append1 } q \rangle$   
**and**  $\langle ?cr1 = \text{cart2-append1 } r \rangle$   
**have**  $\forall k. ?cq - ?cp = k *_R (?cr - ?cp)$   
 $\longleftrightarrow \text{cart2-append1 } q = k *_R \text{cart2-append1 } r + (1 - k) *_R \text{cart2-append1 } p$   
**by** *simp*  
**thus**  $B_{\mathbb{R}} (\text{cart2-pt } p) (\text{cart2-pt } q) (\text{cart2-pt } r)$   
 $\longleftrightarrow (\exists k \geq 0. k \leq 1$   
 $\wedge \text{cart2-append1 } q = k *_R \text{cart2-append1 } r + (1 - k) *_R \text{cart2-append1 } p)$   
**by** (*unfold real-euclid-B-def*) *simp*  
**qed**

**lemma** *cart2-append1-between-right-strict*:

**assumes** *z-non-zero*  $p$  **and** *z-non-zero*  $q$  **and** *z-non-zero*  $r$   
**and**  $B_{\mathbb{R}} (\text{cart2-pt } p) (\text{cart2-pt } q) (\text{cart2-pt } r)$  **and**  $q \neq r$   
**shows**  $\exists k \geq 0. k < 1$   
 $\wedge \text{cart2-append1 } q = k *_R \text{cart2-append1 } r + (1 - k) *_R \text{cart2-append1 } p$

**proof** –

**from**  $\langle \text{z-non-zero } p \rangle$  **and**  $\langle \text{z-non-zero } q \rangle$  **and**  $\langle \text{z-non-zero } r \rangle$   
**and**  $\langle B_{\mathbb{R}} (\text{cart2-pt } p) (\text{cart2-pt } q) (\text{cart2-pt } r) \rangle$  **and** *cart2-append1-between*  
**obtain**  $k$  **where**  $k \geq 0$  **and**  $k \leq 1$   
**and**  $\text{cart2-append1 } q = k *_R \text{cart2-append1 } r + (1 - k) *_R \text{cart2-append1 } p$   
**by** *auto*

**have**  $k \neq 1$

**proof**

**assume**  $k = 1$   
**with**  $\langle \text{cart2-append1 } q = k *_R \text{cart2-append1 } r + (1 - k) *_R \text{cart2-append1 } p \rangle$   
**have**  $\text{cart2-append1 } q = \text{cart2-append1 } r$  **by** *simp*  
**with**  $\langle \text{z-non-zero } q \rangle$  **have**  $q = r$  **by** (*rule cart2-append1-inj*)  
**with**  $\langle q \neq r \rangle$  **show** *False ..*

**qed**

**with**  $\langle k \leq 1 \rangle$  **have**  $k < 1$  **by** *simp*

**with**  $\langle k \geq 0 \rangle$

**and**  $\langle \text{cart2-append1 } q = k *_R \text{cart2-append1 } r + (1 - k) *_R \text{cart2-append1 } p \rangle$

**show**  $\exists k \geq 0. k < 1$

$\wedge \text{cart2-append1 } q = k *_R \text{cart2-append1 } r + (1 - k) *_R \text{cart2-append1 } p$

**by** (*simp add: exI [of - k]*)

**qed**

**lemma** *cart2-append1-between-strict*:

**assumes** *z-non-zero*  $p$  **and** *z-non-zero*  $q$  **and** *z-non-zero*  $r$   
**and**  $B_{\mathbb{R}} (\text{cart2-pt } p) (\text{cart2-pt } q) (\text{cart2-pt } r)$  **and**  $q \neq p$  **and**  $q \neq r$

```

shows  $\exists k > 0. k < 1$ 
 $\wedge \text{cart2-append1 } q = k *_R \text{cart2-append1 } r + (1 - k) *_R \text{cart2-append1 } p$ 
proof -
from  $\langle z\text{-non-zero } p \rangle$  and  $\langle z\text{-non-zero } q \rangle$  and  $\langle z\text{-non-zero } r \rangle$ 
and  $\langle B_R (\text{cart2-pt } p) (\text{cart2-pt } q) (\text{cart2-pt } r) \rangle$  and  $\langle q \neq r \rangle$ 
and  $\text{cart2-append1-between-right-strict } [\text{of } p \ q \ r]$ 
obtain  $k$  where  $k \geq 0$  and  $k < 1$ 
and  $\text{cart2-append1 } q = k *_R \text{cart2-append1 } r + (1 - k) *_R \text{cart2-append1 } p$ 
by auto

have  $k \neq 0$ 
proof
assume  $k = 0$ 
with  $\langle \text{cart2-append1 } q = k *_R \text{cart2-append1 } r + (1 - k) *_R \text{cart2-append1 } p \rangle$ 
have  $\text{cart2-append1 } q = \text{cart2-append1 } p$  by simp
with  $\langle z\text{-non-zero } q \rangle$  have  $q = p$  by (rule cart2-append1-inj)
with  $\langle q \neq p \rangle$  show False ..
qed
with  $\langle k \geq 0 \rangle$  have  $k > 0$  by simp
with  $\langle k < 1 \rangle$ 
and  $\langle \text{cart2-append1 } q = k *_R \text{cart2-append1 } r + (1 - k) *_R \text{cart2-append1 } p \rangle$ 
show  $\exists k > 0. k < 1$ 
 $\wedge \text{cart2-append1 } q = k *_R \text{cart2-append1 } r + (1 - k) *_R \text{cart2-append1 } p$ 
by (simp add: exI [of - k])
qed

end

```

## 8 The hyperbolic plane and Tarski's axioms

```

theory Hyperbolic-Tarski
imports Euclid-Tarski
Projective
HOL-Library.Quadratic-Discriminant
begin

```

### 8.1 Characterizing a specific conic in the projective plane

**definition**  $M :: \text{real}^3 \times \text{real}^3$  **where**

```

 $M \triangleq \text{vector } [$ 
 $\text{vector } [1, 0, 0],$ 
 $\text{vector } [0, 1, 0],$ 
 $\text{vector } [0, 0, -1]]$ 

```

**lemma** *M-symmatrix: symmatrix M*

```

unfolding symmatrix-def and transpose-def and M-def
by (simp add: vec-eq-iff forall-3 vector-3)

```

**lemma** *M-self-inverse: M \*\* M = mat 1*

**unfolding** *M-def* **and** *matrix-matrix-mult-def* **and** *mat-def* **and** *vector-def*  
**by** (*simp add: sum-3 vec-eq-iff forall-3*)

**lemma** *M-invertible: invertible M*  
**unfolding** *invertible-def*  
**using** *M-self-inverse*  
**by** *auto*

**definition** *polar* :: *proj2*  $\Rightarrow$  *proj2-line* **where**  
*polar* *p*  $\triangleq$  *proj2-line-abs* (*M* \**v* *proj2-rep* *p*)

**definition** *pole* :: *proj2-line*  $\Rightarrow$  *proj2* **where**  
*pole* *l*  $\triangleq$  *proj2-abs* (*M* \**v* *proj2-line-rep* *l*)

**lemma** *polar-abs*:  
**assumes** *v*  $\neq$  0  
**shows** *polar* (*proj2-abs* *v*) = *proj2-line-abs* (*M* \**v* *v*)  
**proof** –  
**from**  $\langle v \neq 0 \rangle$  **and** *proj2-rep-abs2*  
**obtain** *k* **where** *k*  $\neq$  0 **and** *proj2-rep* (*proj2-abs* *v*) = *k* \*<sub>R</sub> *v* **by** *auto*  
**from**  $\langle \text{proj2-rep } (\text{proj2-abs } v) = k *_R v \rangle$   
**have** *polar* (*proj2-abs* *v*) = *proj2-line-abs* (*k* \*<sub>R</sub> (*M* \**v* *v*))  
**unfolding** *polar-def*  
**by** (*simp add: matrix-scaleR-vector-ac scaleR-matrix-vector-assoc*)  
**with**  $\langle k \neq 0 \rangle$  **and** *proj2-line-abs-mult*  
**show** *polar* (*proj2-abs* *v*) = *proj2-line-abs* (*M* \**v* *v*) **by** *simp*  
**qed**

**lemma** *pole-abs*:  
**assumes** *v*  $\neq$  0  
**shows** *pole* (*proj2-line-abs* *v*) = *proj2-abs* (*M* \**v* *v*)  
**proof** –  
**from**  $\langle v \neq 0 \rangle$  **and** *proj2-line-rep-abs*  
**obtain** *k* **where** *k*  $\neq$  0 **and** *proj2-line-rep* (*proj2-line-abs* *v*) = *k* \*<sub>R</sub> *v*  
**by** *auto*  
**from**  $\langle \text{proj2-line-rep } (\text{proj2-line-abs } v) = k *_R v \rangle$   
**have** *pole* (*proj2-line-abs* *v*) = *proj2-abs* (*k* \*<sub>R</sub> (*M* \**v* *v*))  
**unfolding** *pole-def*  
**by** (*simp add: matrix-scaleR-vector-ac scaleR-matrix-vector-assoc*)  
**with**  $\langle k \neq 0 \rangle$  **and** *proj2-abs-mult*  
**show** *pole* (*proj2-line-abs* *v*) = *proj2-abs* (*M* \**v* *v*) **by** *simp*  
**qed**

**lemma** *polar-rep-non-zero*: *M* \**v* *proj2-rep* *p*  $\neq$  0  
**proof** –  
**have** *proj2-rep* *p*  $\neq$  0 **by** (*rule* *proj2-rep-non-zero*)  
**with** *M-invertible*  
**show** *M* \**v* *proj2-rep* *p*  $\neq$  0 **by** (*rule* *invertible-times-non-zero*)  
**qed**

**lemma** *pole-polar*:  $\text{pole} (\text{polar } p) = p$   
**proof** –  
    **from** *polar-rep-non-zero*  
    **have**  $\text{pole} (\text{polar } p) = \text{proj2-abs } (M *v (M *v \text{proj2-rep } p))$   
    **unfolding** *polar-def*  
    **by** (*rule pole-abs*)  
    **with** *M-self-inverse*  
    **show**  $\text{pole} (\text{polar } p) = p$   
    **by** (*simp add: matrix-vector-mul-assoc proj2-abs-rep*)  
**qed**

**lemma** *pole-rep-non-zero*:  $M *v \text{proj2-line-rep } l \neq 0$   
**proof** –  
    **have**  $\text{proj2-line-rep } l \neq 0$  **by** (*rule proj2-line-rep-non-zero*)  
    **with** *M-invertible*  
    **show**  $M *v \text{proj2-line-rep } l \neq 0$  **by** (*rule invertible-times-non-zero*)  
**qed**

**lemma** *polar-pole*:  $\text{polar} (\text{pole } l) = l$   
**proof** –  
    **from** *pole-rep-non-zero*  
    **have**  $\text{polar} (\text{pole } l) = \text{proj2-line-abs } (M *v (M *v \text{proj2-line-rep } l))$   
    **unfolding** *pole-def*  
    **by** (*rule polar-abs*)  
    **with** *M-self-inverse*  
    **show**  $\text{polar} (\text{pole } l) = l$   
    **by** (*simp add: matrix-vector-mul-assoc proj2-line-abs-rep matrix-vector-mul-lid*)  
**qed**

**lemma** *polar-inj*:  
    **assumes**  $\text{polar } p = \text{polar } q$   
    **shows**  $p = q$   
**proof** –  
    **from**  $\langle \text{polar } p = \text{polar } q \rangle$  **have**  $\text{pole} (\text{polar } p) = \text{pole} (\text{polar } q)$  **by** *simp*  
    **thus**  $p = q$  **by** (*simp add: pole-polar*)  
**qed**

**definition** *conic-sgn* ::  $\text{proj2} \Rightarrow \text{real}$  **where**  
     $\text{conic-sgn } p \triangleq \text{sgn } (\text{proj2-rep } p \cdot (M *v \text{proj2-rep } p))$

**lemma** *conic-sgn-abs*:  
    **assumes**  $v \neq 0$   
    **shows**  $\text{conic-sgn } (\text{proj2-abs } v) = \text{sgn } (v \cdot (M *v v))$   
**proof** –  
    **from**  $\langle v \neq 0 \rangle$  **and** *proj2-rep-abs2*  
    **obtain**  $j$  **where**  $j \neq 0$  **and**  $\text{proj2-rep } (\text{proj2-abs } v) = j *_R v$  **by** *auto*

**from**  $\langle \text{proj2-rep } (\text{proj2-abs } v) = j *_{\mathbb{R}} v \rangle$   
**have**  $\text{conic-sgn } (\text{proj2-abs } v) = \text{sgn } (j^2 * (v \cdot (M * v v)))$   
**unfolding**  $\text{conic-sgn-def}$   
**by** (*simp add:*  
*matrix-scaleR-vector-ac*  
*scaleR-matrix-vector-assoc [symmetric]*  
*dot-scaleR-mult*  
*power2-eq-square*  
*algebra-simps*)  
**also have**  $\dots = \text{sgn } (j^2) * \text{sgn } (v \cdot (M * v v))$  **by** (*rule sgn-mult*)  
**also from**  $\langle j \neq 0 \rangle$  **have**  $\dots = \text{sgn } (v \cdot (M * v v))$   
**by** (*simp add: power2-eq-square sgn-mult*)  
**finally show**  $\text{conic-sgn } (\text{proj2-abs } v) = \text{sgn } (v \cdot (M * v v))$  .  
**qed**

**lemma**  $\text{sgn-conic-sgn: } \text{sgn } (\text{conic-sgn } p) = \text{conic-sgn } p$   
**by** (*unfold conic-sgn-def*) *simp*

**definition**  $S :: \text{proj2 set where}$   
 $S \triangleq \{p. \text{conic-sgn } p = 0\}$

**definition**  $K2 :: \text{proj2 set where}$   
 $K2 \triangleq \{p. \text{conic-sgn } p < 0\}$

**lemma**  $S\text{-}K2\text{-empty: } S \cap K2 = \{\}$   
**unfolding**  $S\text{-def}$  **and**  $K2\text{-def}$   
**by** *auto*

**lemma**  $K2\text{-abs:}$   
**assumes**  $v \neq 0$   
**shows**  $\text{proj2-abs } v \in K2 \longleftrightarrow v \cdot (M * v v) < 0$   
**proof** –  
**have**  $\text{proj2-abs } v \in K2 \longleftrightarrow \text{conic-sgn } (\text{proj2-abs } v) < 0$   
**by** (*simp add: K2-def*)  
**with**  $\langle v \neq 0 \rangle$  **and**  $\text{conic-sgn-abs}$   
**show**  $\text{proj2-abs } v \in K2 \longleftrightarrow v \cdot (M * v v) < 0$  **by** *simp*  
**qed**

**definition**  $K2\text{-centre} = \text{proj2-abs } (\text{vector } [0,0,1])$

**lemma**  $K2\text{-centre-non-zero: } \text{vector } [0,0,1] \neq (0 :: \text{real}^3)$   
**by** (*unfold vector-def*) (*simp add: vec-eq-iff forall-3*)

**lemma**  $K2\text{-centre-in-}K2: K2\text{-centre} \in K2$   
**proof** –  
**from**  $K2\text{-centre-non-zero}$  **and**  $\text{proj2-rep-abs2}$   
**obtain**  $k$  **where**  $k \neq 0$  **and**  $\text{proj2-rep } K2\text{-centre} = k *_{\mathbb{R}} \text{vector } [0,0,1]$   
**by** (*unfold K2-centre-def*) *auto*  
**from**  $\langle k \neq 0 \rangle$  **have**  $0 < k^2$  **by** *simp*

**with**  $\langle \text{proj2-rep } K2\text{-centre} = k *_R \text{vector } [0,0,1] \rangle$   
**show**  $K2\text{-centre} \in K2$   
**unfolding**  $K2\text{-def}$   
**and**  $\text{conic-sgn-def}$   
**and**  $M\text{-def}$   
**and**  $\text{matrix-vector-mult-def}$   
**and**  $\text{inner-vec-def}$   
**and**  $\text{vector-def}$   
**by**  $(\text{simp add: vec-eq-iff sum-3 power2-eq-square})$   
**qed**

**lemma**  $K2\text{-imp-}M\text{-neg}$ :  
**assumes**  $v \neq 0$  **and**  $\text{proj2-abs } v \in K2$   
**shows**  $v \cdot (M *v v) < 0$   
**using**  $\text{assms}$   
**by**  $(\text{simp add: } K2\text{-abs})$

**lemma**  $M\text{-neg-imp-}z\text{-squared-big}$ :  
**assumes**  $v \cdot (M *v v) < 0$   
**shows**  $(v\$3)^2 > (v\$1)^2 + (v\$2)^2$   
**using**  $\langle v \cdot (M *v v) < 0 \rangle$   
**unfolding**  $\text{matrix-vector-mult-def}$  **and**  $M\text{-def}$  **and**  $\text{vector-def}$   
**by**  $(\text{simp add: inner-vec-def sum-3 power2-eq-square})$

**lemma**  $M\text{-neg-imp-}z\text{-non-zero}$ :  
**assumes**  $v \cdot (M *v v) < 0$   
**shows**  $v\$3 \neq 0$   
**proof** –  
**have**  $(v\$1)^2 + (v\$2)^2 \geq 0$  **by**  $\text{simp}$   
**with**  $M\text{-neg-imp-}z\text{-squared-big}$   $[\text{of } v]$  **and**  $\langle v \cdot (M *v v) < 0 \rangle$   
**have**  $(v\$3)^2 > 0$  **by**  $\text{arith}$   
**thus**  $v\$3 \neq 0$  **by**  $\text{simp}$   
**qed**

**lemma**  $M\text{-neg-imp-}K2$ :  
**assumes**  $v \cdot (M *v v) < 0$   
**shows**  $\text{proj2-abs } v \in K2$   
**proof** –  
**from**  $\langle v \cdot (M *v v) < 0 \rangle$  **have**  $v\$3 \neq 0$  **by**  $(\text{rule } M\text{-neg-imp-}z\text{-non-zero})$   
**hence**  $v \neq 0$  **by**  $\text{auto}$   
**with**  $\langle v \cdot (M *v v) < 0 \rangle$  **and**  $K2\text{-abs}$  **show**  $\text{proj2-abs } v \in K2$  **by**  $\text{simp}$   
**qed**

**lemma**  $M\text{-reverse}$ :  $a \cdot (M *v b) = b \cdot (M *v a)$   
**unfolding**  $\text{matrix-vector-mult-def}$  **and**  $M\text{-def}$  **and**  $\text{vector-def}$   
**by**  $(\text{simp add: inner-vec-def sum-3})$

**lemma**  $S\text{-abs}$ :  
**assumes**  $v \neq 0$

**shows**  $\text{proj2-abs } v \in S \longleftrightarrow v \cdot (M *v v) = 0$   
**proof** –  
**have**  $\text{proj2-abs } v \in S \longleftrightarrow \text{conic-sgn } (\text{proj2-abs } v) = 0$   
**unfolding**  $S\text{-def}$   
**by**  $\text{simp}$   
**also from**  $\langle v \neq 0 \rangle$  **and**  $\text{conic-sgn-abs}$   
**have**  $\dots \longleftrightarrow \text{sgn } (v \cdot (M *v v)) = 0$  **by**  $\text{simp}$   
**finally show**  $\text{proj2-abs } v \in S \longleftrightarrow v \cdot (M *v v) = 0$  **by**  $(\text{simp add: sgn-0-0})$   
**qed**

**lemma**  $S\text{-alt-def}$ :  $p \in S \longleftrightarrow \text{proj2-rep } p \cdot (M *v \text{proj2-rep } p) = 0$   
**proof** –  
**have**  $\text{proj2-rep } p \neq 0$  **by**  $(\text{rule proj2-rep-non-zero})$   
**hence**  $\text{proj2-abs } (\text{proj2-rep } p) \in S \longleftrightarrow \text{proj2-rep } p \cdot (M *v \text{proj2-rep } p) = 0$   
**by**  $(\text{rule } S\text{-abs})$   
**thus**  $p \in S \longleftrightarrow \text{proj2-rep } p \cdot (M *v \text{proj2-rep } p) = 0$   
**by**  $(\text{simp add: proj2-abs-rep})$   
**qed**

**lemma**  $\text{incident-polar}$ :  
 $\text{proj2-incident } p (\text{polar } q) \longleftrightarrow \text{proj2-rep } p \cdot (M *v \text{proj2-rep } q) = 0$   
**using**  $\text{polar-rep-non-zero}$   
**unfolding**  $\text{polar-def}$   
**by**  $(\text{rule proj2-incident-right-abs})$

**lemma**  $\text{incident-own-polar-in-S}$ :  $\text{proj2-incident } p (\text{polar } p) \longleftrightarrow p \in S$   
**using**  $\text{incident-polar}$  **and**  $S\text{-alt-def}$   
**by**  $\text{simp}$

**lemma**  $\text{incident-polar-swap}$ :  
**assumes**  $\text{proj2-incident } p (\text{polar } q)$   
**shows**  $\text{proj2-incident } q (\text{polar } p)$   
**proof** –  
**from**  $\langle \text{proj2-incident } p (\text{polar } q) \rangle$   
**have**  $\text{proj2-rep } p \cdot (M *v \text{proj2-rep } q) = 0$  **by**  $(\text{unfold incident-polar})$   
**hence**  $\text{proj2-rep } q \cdot (M *v \text{proj2-rep } p) = 0$  **by**  $(\text{simp add: } M\text{-reverse})$   
**thus**  $\text{proj2-incident } q (\text{polar } p)$  **by**  $(\text{unfold incident-polar})$   
**qed**

**lemma**  $\text{incident-pole-polar}$ :  
**assumes**  $\text{proj2-incident } p l$   
**shows**  $\text{proj2-incident } (\text{pole } l) (\text{polar } p)$   
**proof** –  
**from**  $\langle \text{proj2-incident } p l \rangle$   
**have**  $\text{proj2-incident } p (\text{polar } (\text{pole } l))$  **by**  $(\text{subst polar-pole})$   
**thus**  $\text{proj2-incident } (\text{pole } l) (\text{polar } p)$  **by**  $(\text{rule incident-polar-swap})$   
**qed**

**definition**  $z\text{-zero}$  ::  $\text{proj2-line}$  **where**

$z\text{-zero} \triangleq \text{proj2-line-abs } (\text{vector } [0,0,1])$

**lemma**  $z\text{-zero}$ :

**assumes**  $(\text{proj2-rep } p)\$3 = 0$

**shows**  $\text{proj2-incident } p \ z\text{-zero}$

**proof** –

**from**  $K2\text{-centre-non-zero}$  **and**  $\text{proj2-line-rep-abs}$

**obtain**  $k$  **where**  $\text{proj2-line-rep } z\text{-zero} = k *_R \text{vector } [0,0,1]$

**by**  $(\text{unfold } z\text{-zero-def}) \text{ auto}$

**with**  $\langle (\text{proj2-rep } p)\$3 = 0 \rangle$

**show**  $\text{proj2-incident } p \ z\text{-zero}$

**unfolding**  $\text{proj2-incident-def}$  **and**  $\text{inner-vec-def}$  **and**  $\text{vector-def}$

**by**  $(\text{simp add: sum-3})$

**qed**

**lemma**  $z\text{-zero-conic-sgn-1}$ :

**assumes**  $\text{proj2-incident } p \ z\text{-zero}$

**shows**  $\text{conic-sgn } p = 1$

**proof** –

**let**  $?v = \text{proj2-rep } p$

**have**  $(\text{vector } [0,0,1] :: \text{real}^3) \neq 0$

**unfolding**  $\text{vector-def}$

**by**  $(\text{simp add: vec-eq-iff})$

**with**  $\langle \text{proj2-incident } p \ z\text{-zero} \rangle$

**have**  $?v \cdot \text{vector } [0,0,1] = 0$

**unfolding**  $z\text{-zero-def}$

**by**  $(\text{simp add: proj2-incident-right-abs})$

**hence**  $?v\$3 = 0$

**unfolding**  $\text{inner-vec-def}$  **and**  $\text{vector-def}$

**by**  $(\text{simp add: sum-3})$

**hence**  $?v \cdot (M *_v ?v) = (?v\$1)^2 + (?v\$2)^2$

**unfolding**  $\text{inner-vec-def}$

**and**  $\text{power2-eq-square}$

**and**  $\text{matrix-vector-mult-def}$

**and**  $M\text{-def}$

**and**  $\text{vector-def}$

**and**  $\text{sum-3}$

**by**  $\text{simp}$

**have**  $?v \neq 0$  **by**  $(\text{rule } \text{proj2-rep-non-zero})$

**with**  $\langle ?v\$3 = 0 \rangle$  **have**  $?v\$1 \neq 0 \vee ?v\$2 \neq 0$  **by**  $(\text{simp add: vec-eq-iff forall-3})$

**hence**  $(?v\$1)^2 > 0 \vee (?v\$2)^2 > 0$  **by**  $\text{simp}$

**with**  $\text{add-sign-intros [of } (?v\$1)^2 \ (?v\$2)^2]$

**have**  $(?v\$1)^2 + (?v\$2)^2 > 0$  **by**  $\text{auto}$

**with**  $\langle ?v \cdot (M *_v ?v) = (?v\$1)^2 + (?v\$2)^2 \rangle$

**have**  $?v \cdot (M *_v ?v) > 0$  **by**  $\text{simp}$

**thus**  $\text{conic-sgn } p = 1$

**unfolding**  $\text{conic-sgn-def}$

**by**  $\text{simp}$

qed

lemma *conic-sgn-not-1-z-non-zero*:

assumes *conic-sgn*  $p \neq 1$

shows *z-non-zero*  $p$

proof –

from  $\langle \text{conic-sgn } p \neq 1 \rangle$

have  $\neg \text{proj2-incident } p \text{ z-zero}$  by (auto simp add: *z-zero-conic-sgn-1*)

thus *z-non-zero*  $p$  by (auto simp add: *z-zero*)

qed

lemma *z-zero-not-in-S*:

assumes *proj2-incident*  $p \text{ z-zero}$

shows  $p \notin S$

proof –

from  $\langle \text{proj2-incident } p \text{ z-zero} \rangle$  have *conic-sgn*  $p = 1$

by (rule *z-zero-conic-sgn-1*)

thus  $p \notin S$

unfolding *S-def*

by *simp*

qed

lemma *line-incident-point-not-in-S*:  $\exists p. p \notin S \wedge \text{proj2-incident } p \ l$

proof –

let  $?p = \text{proj2-intersection } l \text{ z-zero}$

have *proj2-incident*  $?p \ l$  and *proj2-incident*  $?p \ \text{z-zero}$

by (rule *proj2-intersection-incident*)+

from  $\langle \text{proj2-incident } ?p \ \text{z-zero} \rangle$  have  $?p \notin S$  by (rule *z-zero-not-in-S*)

with  $\langle \text{proj2-incident } ?p \ l \rangle$

show  $\exists p. p \notin S \wedge \text{proj2-incident } p \ l$  by auto

qed

lemma *apply-cltn2-abs-abs-in-S*:

assumes  $v \neq 0$  and *invertible*  $J$

shows *apply-cltn2* (*proj2-abs*  $v$ ) (*cltn2-abs*  $J$ )  $\in S$

$\longleftrightarrow v \cdot (J ** M ** \text{transpose } J * v \ v) = 0$

proof –

from  $\langle v \neq 0 \rangle$  and  $\langle \text{invertible } J \rangle$

have  $v \ v * J \neq 0$  by (rule *non-zero-mult-invertible-non-zero*)

from  $\langle v \neq 0 \rangle$  and  $\langle \text{invertible } J \rangle$

have *apply-cltn2* (*proj2-abs*  $v$ ) (*cltn2-abs*  $J$ ) = *proj2-abs* ( $v \ v * J$ )

by (rule *apply-cltn2-abs*)

also from  $\langle v \ v * J \neq 0 \rangle$

have  $\dots \in S \longleftrightarrow (v \ v * J) \cdot (M * v \ (v \ v * J)) = 0$  by (rule *S-abs*)

finally show *apply-cltn2* (*proj2-abs*  $v$ ) (*cltn2-abs*  $J$ )  $\in S$

$\longleftrightarrow v \cdot (J ** M ** \text{transpose } J * v \ v) = 0$

by (*simp* add: *dot-lmul-matrix matrix-vector-mul-assoc* [*symmetric*])

qed

**lemma** *apply-cltn2-right-abs-in-S*:

**assumes** *invertible J*

**shows** *apply-cltn2 p (cltn2-abs J) ∈ S*

$\longleftrightarrow$  *(proj2-rep p) · (J \*\* M \*\* transpose J \*v (proj2-rep p)) = 0*

**proof** –

**have** *proj2-rep p ≠ 0* **by** (*rule proj2-rep-non-zero*)

**with**  $\langle$ *invertible J* $\rangle$

**have** *apply-cltn2 (proj2-abs (proj2-rep p)) (cltn2-abs J) ∈ S*

$\longleftrightarrow$  *proj2-rep p · (J \*\* M \*\* transpose J \*v proj2-rep p) = 0*

**by** (*simp add: apply-cltn2-abs-abs-in-S*)

**thus** *apply-cltn2 p (cltn2-abs J) ∈ S*

$\longleftrightarrow$  *proj2-rep p · (J \*\* M \*\* transpose J \*v proj2-rep p) = 0*

**by** (*simp add: proj2-abs-rep*)

**qed**

**lemma** *apply-cltn2-abs-in-S*:

**assumes** *v ≠ 0*

**shows** *apply-cltn2 (proj2-abs v) C ∈ S*

$\longleftrightarrow$  *v · (cltn2-rep C \*\* M \*\* transpose (cltn2-rep C) \*v v) = 0*

**proof** –

**have** *invertible (cltn2-rep C)* **by** (*rule cltn2-rep-invertible*)

**with**  $\langle$ *v ≠ 0* $\rangle$

**have** *apply-cltn2 (proj2-abs v) (cltn2-abs (cltn2-rep C)) ∈ S*

$\longleftrightarrow$  *v · (cltn2-rep C \*\* M \*\* transpose (cltn2-rep C) \*v v) = 0*

**by** (*rule apply-cltn2-abs-abs-in-S*)

**thus** *apply-cltn2 (proj2-abs v) C ∈ S*

$\longleftrightarrow$  *v · (cltn2-rep C \*\* M \*\* transpose (cltn2-rep C) \*v v) = 0*

**by** (*simp add: cltn2-abs-rep*)

**qed**

**lemma** *apply-cltn2-in-S*:

*apply-cltn2 p C ∈ S*

$\longleftrightarrow$  *proj2-rep p · (cltn2-rep C \*\* M \*\* transpose (cltn2-rep C) \*v proj2-rep p)*  
*= 0*

**proof** –

**have** *proj2-rep p ≠ 0* **by** (*rule proj2-rep-non-zero*)

**hence** *apply-cltn2 (proj2-abs (proj2-rep p)) C ∈ S*

$\longleftrightarrow$  *proj2-rep p · (cltn2-rep C \*\* M \*\* transpose (cltn2-rep C) \*v proj2-rep p)*  
*= 0*

**by** (*rule apply-cltn2-abs-in-S*)

**thus** *apply-cltn2 p C ∈ S*

$\longleftrightarrow$  *proj2-rep p · (cltn2-rep C \*\* M \*\* transpose (cltn2-rep C) \*v proj2-rep p)*  
*= 0*

**by** (*simp add: proj2-abs-rep*)

**qed**

**lemma** *norm-M*:  $(\text{vector2-append1 } v) \cdot (M *v \text{vector2-append1 } v) = (\text{norm } v)^2 - 1$

**proof** –  
**have**  $(\text{norm } v)^2 = (v\$1)^2 + (v\$2)^2$   
**unfolding** *norm-vec-def*  
**and** *L2-set-def*  
**by** (*simp add: sum-2*)  
**thus**  $(\text{vector2-append1 } v) \cdot (M *v \text{vector2-append1 } v) = (\text{norm } v)^2 - 1$   
**unfolding** *vector2-append1-def*  
**and** *inner-vec-def*  
**and** *matrix-vector-mult-def*  
**and** *vector-def*  
**and** *M-def*  
**and** *power2-norm-eq-inner*  
**by** (*simp add: sum-3 power2-eq-square*)  
**qed**

## 8.2 Some specific points and lines of the projective plane

**definition** *east* = *proj2-abs* (*vector* [1,0,1])  
**definition** *west* = *proj2-abs* (*vector* [-1,0,1])  
**definition** *north* = *proj2-abs* (*vector* [0,1,1])  
**definition** *south* = *proj2-abs* (*vector* [0,-1,1])  
**definition** *far-north* = *proj2-abs* (*vector* [0,1,0])

**lemmas** *compass-defs* = *east-def west-def north-def south-def*

**lemma** *compass-non-zero*:  
**shows** *vector* [1,0,1]  $\neq$  (0 :: *real*<sup>3</sup>)  
**and** *vector* [-1,0,1]  $\neq$  (0 :: *real*<sup>3</sup>)  
**and** *vector* [0,1,1]  $\neq$  (0 :: *real*<sup>3</sup>)  
**and** *vector* [0,-1,1]  $\neq$  (0 :: *real*<sup>3</sup>)  
**and** *vector* [0,1,0]  $\neq$  (0 :: *real*<sup>3</sup>)  
**and** *vector* [1,0,0]  $\neq$  (0 :: *real*<sup>3</sup>)  
**unfolding** *vector-def*  
**by** (*simp-all add: vec-eq-iff forall-3*)

**lemma** *east-west-distinct*: *east*  $\neq$  *west*

**proof**  
**assume** *east* = *west*  
**with** *compass-non-zero*  
**and** *proj2-abs-abs-mult* [*of vector* [1,0,1] *vector* [-1,0,1]]  
**obtain** *k* **where** (*vector* [1,0,1] :: *real*<sup>3</sup>) = *k* \*<sub>R</sub> *vector* [-1,0,1]  
**unfolding** *compass-defs*  
**by** *auto*  
**thus** *False*  
**unfolding** *vector-def*  
**by** (*auto simp add: vec-eq-iff forall-3*)  
**qed**

**lemma** *north-south-distinct*: *north*  $\neq$  *south*

**proof**  
**assume**  $north = south$   
**with** *compass-non-zero*  
**and** *proj2-abs-abs-mult* [of vector  $[0,1,1]$  vector  $[0,-1,1]$ ]  
**obtain**  $k$  **where**  $(vector\ [0,1,1] :: real^3) = k *_R vector\ [0,-1,1]$   
**unfolding** *compass-defs*  
**by** *auto*  
**thus** *False*  
**unfolding** *vector-def*  
**by**  $(auto\ simp\ add:\ vec\ eq\ iff\ forall\ 3)$   
**qed**

**lemma** *north-not-east-or-west*:  $north \notin \{east, west\}$   
**proof**

**assume**  $north \in \{east, west\}$   
**hence**  $east = north \vee west = north$  **by** *auto*  
**with** *compass-non-zero*  
**and** *proj2-abs-abs-mult* [of - vector  $[0,1,1]$ ]  
**obtain**  $k$  **where**  $(vector\ [1,0,1] :: real^3) = k *_R vector\ [0,1,1]$   
 $\vee (vector\ [-1,0,1] :: real^3) = k *_R vector\ [0,1,1]$   
**unfolding** *compass-defs*  
**by** *auto*  
**thus** *False*  
**unfolding** *vector-def*  
**by**  $(simp\ add:\ vec\ eq\ iff\ forall\ 3)$   
**qed**

**lemma** *compass-in-S*:

**shows**  $east \in S$  **and**  $west \in S$  **and**  $north \in S$  **and**  $south \in S$   
**using** *compass-non-zero* **and** *S-abs*  
**unfolding** *compass-defs*  
**and** *M-def*  
**and** *inner-vec-def*  
**and** *matrix-vector-mult-def*  
**and** *vector-def*  
**by**  $(simp\ all\ add:\ sum\ 3)$

**lemma** *east-west-tangents*:

**shows**  $polar\ east = proj2\ line\ abs\ (vector\ [-1,0,1])$   
**and**  $polar\ west = proj2\ line\ abs\ (vector\ [1,0,1])$   
**proof** –  
**have**  $M *_v vector\ [1,0,1] = (-1) *_R vector\ [-1,0,1]$   
**and**  $M *_v vector\ [-1,0,1] = (-1) *_R vector\ [1,0,1]$   
**unfolding** *M-def* **and** *matrix-vector-mult-def* **and** *vector-def*  
**by**  $(simp\ all\ add:\ vec\ eq\ iff\ sum\ 3)$   
**with** *compass-non-zero* **and** *polar-abs*  
**have**  $polar\ east = proj2\ line\ abs\ ((-1) *_R vector\ [-1,0,1])$   
**and**  $polar\ west = proj2\ line\ abs\ ((-1) *_R vector\ [1,0,1])$   
**unfolding** *compass-defs*

by *simp-all*  
 with *proj2-line-abs-mult* [of  $-1$ ]  
 show *polar east* = *proj2-line-abs* (*vector*  $[-1,0,1]$ )  
 and *polar west* = *proj2-line-abs* (*vector*  $[1,0,1]$ )  
 by *simp-all*  
 qed

**lemma** *east-west-tangents-distinct*: *polar east*  $\neq$  *polar west*  
**proof**

assume *polar east* = *polar west*  
 hence *east* = *west* by (*rule polar-inj*)  
 with *east-west-distinct* show *False* ..  
 qed

**lemma** *east-west-tangents-incident-far-north*:  
 shows *proj2-incident far-north* (*polar east*)  
 and *proj2-incident far-north* (*polar west*)  
 using *compass-non-zero* and *proj2-incident-abs*  
 unfolding *far-north-def* and *east-west-tangents* and *inner-vec-def*  
 by (*simp-all add: sum-3 vector-3*)

**lemma** *east-west-tangents-far-north*:  
*proj2-intersection* (*polar east*) (*polar west*) = *far-north*  
 using *east-west-tangents-distinct* and *east-west-tangents-incident-far-north*  
 by (*rule proj2-intersection-unique [symmetric]*)

**instantiation** *proj2* :: *zero*  
**begin**  
**definition** *proj2-zero-def*:  $0 = \text{proj2-pt } 0$   
**instance** ..  
**end**

**definition** *equator*  $\triangleq$  *proj2-line-abs* (*vector*  $[0,1,0]$ )  
**definition** *meridian*  $\triangleq$  *proj2-line-abs* (*vector*  $[1,0,0]$ )

**lemma** *equator-meridian-distinct*: *equator*  $\neq$  *meridian*  
**proof**  
 assume *equator* = *meridian*  
 with *compass-non-zero*  
 and *proj2-line-abs-abs-mult* [of *vector*  $[0,1,0]$  *vector*  $[1,0,0]$ ]  
 obtain *k* where (*vector*  $[0,1,0]$  ::  $\text{real}^3$ ) = *k* \*<sub>R</sub> *vector*  $[1,0,0]$   
 by (*unfold equator-def meridian-def*) *auto*  
 thus *False* by (*unfold vector-def*) (*auto simp add: vec-eq-iff forall-3*)  
 qed

**lemma** *east-west-on-equator*:  
 shows *proj2-incident east equator* and *proj2-incident west equator*  
 unfolding *east-def* and *west-def* and *equator-def*  
 using *compass-non-zero*

by (simp-all add: proj2-incident-abs inner-vec-def vector-def sum-3)

**lemma** north-far-north-distinct: north  $\neq$  far-north

**proof**

assume north = far-north

with compass-non-zero

and proj2-abs-abs-mult [of vector [0,1,1] vector [0,1,0]]

obtain k where (vector [0,1,1] :: real<sup>3</sup>) = k \*<sub>R</sub> vector [0,1,0]

by (unfold north-def far-north-def) auto

thus False

unfolding vector-def

by (auto simp add: vec-eq-iff forall-3)

qed

**lemma** north-south-far-north-on-meridian:

shows proj2-incident north meridian and proj2-incident south meridian

and proj2-incident far-north meridian

unfolding compass-defs and far-north-def and meridian-def

using compass-non-zero

by (simp-all add: proj2-incident-abs inner-vec-def vector-def sum-3)

**lemma** K2-centre-on-equator-meridian:

shows proj2-incident K2-centre equator

and proj2-incident K2-centre meridian

unfolding K2-centre-def and equator-def and meridian-def

using K2-centre-non-zero and compass-non-zero

by (simp-all add: proj2-incident-abs inner-vec-def vector-def sum-3)

**lemma** on-equator-meridian-is-K2-centre:

assumes proj2-incident a equator and proj2-incident a meridian

shows a = K2-centre

using assms and K2-centre-on-equator-meridian and equator-meridian-distinct

and proj2-incident-unique

by auto

**definition** rep-equator-reflect  $\triangleq$  vector [

vector [1, 0,0],

vector [0,-1,0],

vector [0, 0,1]] :: real<sup>3</sup><sup>3</sup>

**definition** rep-meridian-reflect  $\triangleq$  vector [

vector [-1,0,0],

vector [ 0,1,0],

vector [ 0,0,1]] :: real<sup>3</sup><sup>3</sup>

**definition** equator-reflect  $\triangleq$  cltn2-abs rep-equator-reflect

**definition** meridian-reflect  $\triangleq$  cltn2-abs rep-meridian-reflect

**lemmas** compass-reflect-defs = equator-reflect-def meridian-reflect-def

rep-equator-reflect-def rep-meridian-reflect-def

**lemma** *compass-reflect-self-inverse*:  
**shows** *rep-equator-reflect* \*\* *rep-equator-reflect* = *mat 1*  
**and** *rep-meridian-reflect* \*\* *rep-meridian-reflect* = *mat 1*  
**unfolding** *compass-reflect-defs matrix-matrix-mult-def mat-def*  
**by** (*simp-all add: vec-eq-iff forall-3 sum-3 vector-3*)

**lemma** *compass-reflect-invertible*:  
**shows** *invertible rep-equator-reflect* **and** *invertible rep-meridian-reflect*  
**unfolding** *invertible-def*  
**using** *compass-reflect-self-inverse*  
**by** *auto*

**lemma** *compass-reflect-compass*:  
**shows** *apply-cltn2 east meridian-reflect* = *west*  
**and** *apply-cltn2 west meridian-reflect* = *east*  
**and** *apply-cltn2 north meridian-reflect* = *north*  
**and** *apply-cltn2 south meridian-reflect* = *south*  
**and** *apply-cltn2 K2-centre meridian-reflect* = *K2-centre*  
**and** *apply-cltn2 east equator-reflect* = *east*  
**and** *apply-cltn2 west equator-reflect* = *west*  
**and** *apply-cltn2 north equator-reflect* = *south*  
**and** *apply-cltn2 south equator-reflect* = *north*  
**and** *apply-cltn2 K2-centre equator-reflect* = *K2-centre*

**proof** –

**have** (*vector [1,0,1] :: real^3*) *v*\* *rep-meridian-reflect* = *vector [-1,0,1]*  
**and** (*vector [-1,0,1] :: real^3*) *v*\* *rep-meridian-reflect* = *vector [1,0,1]*  
**and** (*vector [0,1,1] :: real^3*) *v*\* *rep-meridian-reflect* = *vector [0,1,1]*  
**and** (*vector [0,-1,1] :: real^3*) *v*\* *rep-meridian-reflect* = *vector [0,-1,1]*  
**and** (*vector [0,0,1] :: real^3*) *v*\* *rep-meridian-reflect* = *vector [0,0,1]*  
**and** (*vector [1,0,1] :: real^3*) *v*\* *rep-equator-reflect* = *vector [1,0,1]*  
**and** (*vector [-1,0,1] :: real^3*) *v*\* *rep-equator-reflect* = *vector [-1,0,1]*  
**and** (*vector [0,1,1] :: real^3*) *v*\* *rep-equator-reflect* = *vector [0,-1,1]*  
**and** (*vector [0,-1,1] :: real^3*) *v*\* *rep-equator-reflect* = *vector [0,1,1]*  
**and** (*vector [0,0,1] :: real^3*) *v*\* *rep-equator-reflect* = *vector [0,0,1]*  
**unfolding** *rep-meridian-reflect-def* **and** *rep-equator-reflect-def*  
**and** *vector-matrix-mult-def*  
**by** (*simp-all add: vec-eq-iff forall-3 vector-3 sum-3*)

**with** *compass-reflect-invertible* **and** *compass-non-zero* **and** *K2-centre-non-zero*  
**show** *apply-cltn2 east meridian-reflect* = *west*  
**and** *apply-cltn2 west meridian-reflect* = *east*  
**and** *apply-cltn2 north meridian-reflect* = *north*  
**and** *apply-cltn2 south meridian-reflect* = *south*  
**and** *apply-cltn2 K2-centre meridian-reflect* = *K2-centre*  
**and** *apply-cltn2 east equator-reflect* = *east*  
**and** *apply-cltn2 west equator-reflect* = *west*  
**and** *apply-cltn2 north equator-reflect* = *south*  
**and** *apply-cltn2 south equator-reflect* = *north*  
**and** *apply-cltn2 K2-centre equator-reflect* = *K2-centre*  
**unfolding** *compass-defs* **and** *K2-centre-def*

and *meridian-reflect-def* and *equator-reflect-def*  
 by (*simp-all add: apply-cltn2-abs*)

qed

**lemma** *on-equator-rep*:

assumes *z-non-zero a* and *proj2-incident a equator*

shows  $\exists x. a = \text{proj2-abs } (\text{vector } [x,0,1])$

**proof** –

let  $?ra = \text{proj2-rep } a$

let  $?ca1 = \text{cart2-append1 } a$

let  $?x = ?ca1\$1$

from *compass-non-zero* and  $\langle \text{proj2-incident } a \text{ equator} \rangle$

have  $?ra \cdot \text{vector } [0,1,0] = 0$

by (*unfold equator-def*) (*simp add: proj2-incident-right-abs*)

hence  $?ra\$2 = 0$  by (*unfold inner-vec-def vector-def*) (*simp add: sum-3*)

hence  $?ca1\$2 = 0$  by (*unfold cart2-append1-def*) *simp*

moreover

from  $\langle z\text{-non-zero } a \rangle$  have  $?ca1\$3 = 1$  by (*rule cart2-append1-z*)

ultimately

have  $?ca1 = \text{vector } [?x,0,1]$

by (*unfold vector-def*) (*simp add: vec-eq-iff forall-3*)

with  $\langle z\text{-non-zero } a \rangle$

have  $\text{proj2-abs } (\text{vector } [?x,0,1]) = a$  by (*simp add: proj2-abs-cart2-append1*)

thus  $\exists x. a = \text{proj2-abs } (\text{vector } [x,0,1])$  by (*simp add: exI [of - ?x]*)

qed

**lemma** *on-meridian-rep*:

assumes *z-non-zero a* and *proj2-incident a meridian*

shows  $\exists y. a = \text{proj2-abs } (\text{vector } [0,y,1])$

**proof** –

let  $?ra = \text{proj2-rep } a$

let  $?ca1 = \text{cart2-append1 } a$

let  $?y = ?ca1\$2$

from *compass-non-zero* and  $\langle \text{proj2-incident } a \text{ meridian} \rangle$

have  $?ra \cdot \text{vector } [1,0,0] = 0$

by (*unfold meridian-def*) (*simp add: proj2-incident-right-abs*)

hence  $?ra\$1 = 0$  by (*unfold inner-vec-def vector-def*) (*simp add: sum-3*)

hence  $?ca1\$1 = 0$  by (*unfold cart2-append1-def*) *simp*

moreover

from  $\langle z\text{-non-zero } a \rangle$  have  $?ca1\$3 = 1$  by (*rule cart2-append1-z*)

ultimately

have  $?ca1 = \text{vector } [0,?y,1]$

by (*unfold vector-def*) (*simp add: vec-eq-iff forall-3*)

with  $\langle z\text{-non-zero } a \rangle$

have  $\text{proj2-abs } (\text{vector } [0,?y,1]) = a$  by (*simp add: proj2-abs-cart2-append1*)

thus  $\exists y. a = \text{proj2-abs } (\text{vector } [0,y,1])$  by (*simp add: exI [of - ?y]*)

qed

### 8.3 Definition of the Klein–Beltrami model of the hyperbolic plane

abbreviation  $hyp2 == K2$

typedef  $hyp2 = K2$   
 using  $K2\text{-centre-in-}K2$   
 by  $auto$

definition  $hyp2\text{-rep} :: hyp2 \Rightarrow real^2$  where  
 $hyp2\text{-rep } p \triangleq cart2\text{-pt } (Rep\text{-hyp2 } p)$

definition  $hyp2\text{-abs} :: real^2 \Rightarrow hyp2$  where  
 $hyp2\text{-abs } v = Abs\text{-hyp2 } (proj2\text{-pt } v)$

lemma  $norm\text{-lt-1-iff-in-hyp2}$ :

shows  $norm\ v < 1 \longleftrightarrow proj2\text{-pt } v \in hyp2$

proof –

let  $?v' = vector2\text{-append1 } v$

have  $?v' \neq 0$  by (rule  $vector2\text{-append1-non-zero}$ )

from  $real\text{-less-rsqrt}$  [of  $norm\ v\ 1$ ]

and  $abs\text{-square-less-1}$  [of  $norm\ v$ ]

have  $norm\ v < 1 \longleftrightarrow (norm\ v)^2 < 1$  by  $auto$

hence  $norm\ v < 1 \longleftrightarrow ?v' \cdot (M *v ?v') < 0$  by ( $simp\ add: norm\text{-}M$ )

with  $\langle ?v' \neq 0 \rangle$  have  $norm\ v < 1 \longleftrightarrow proj2\text{-abs } ?v' \in K2$  by ( $subst\ K2\text{-abs}$ )

thus  $norm\ v < 1 \longleftrightarrow proj2\text{-pt } v \in hyp2$  by ( $unfold\ proj2\text{-pt-def}$ )

qed

lemma  $norm\text{-eq-1-iff-in-S}$ :

shows  $norm\ v = 1 \longleftrightarrow proj2\text{-pt } v \in S$

proof –

let  $?v' = vector2\text{-append1 } v$

have  $?v' \neq 0$  by (rule  $vector2\text{-append1-non-zero}$ )

from  $real\text{-sqrt-unique}$  [of  $norm\ v\ 1$ ]

have  $norm\ v = 1 \longleftrightarrow (norm\ v)^2 = 1$  by  $auto$

hence  $norm\ v = 1 \longleftrightarrow ?v' \cdot (M *v ?v') = 0$  by ( $simp\ add: norm\text{-}M$ )

with  $\langle ?v' \neq 0 \rangle$  have  $norm\ v = 1 \longleftrightarrow proj2\text{-abs } ?v' \in S$  by ( $subst\ S\text{-abs}$ )

thus  $norm\ v = 1 \longleftrightarrow proj2\text{-pt } v \in S$  by ( $unfold\ proj2\text{-pt-def}$ )

qed

lemma  $norm\text{-le-1-iff-in-hyp2-S}$ :

$norm\ v \leq 1 \longleftrightarrow proj2\text{-pt } v \in hyp2 \cup S$

using  $norm\text{-lt-1-iff-in-hyp2}$  [of  $v$ ] and  $norm\text{-eq-1-iff-in-S}$  [of  $v$ ]

by  $auto$

lemma  $proj2\text{-pt-hyp2-rep}$ :  $proj2\text{-pt } (hyp2\text{-rep } p) = Rep\text{-hyp2 } p$

proof –

let  $?p' = Rep\text{-hyp2 } p$

**let**  $?v = \text{proj2-rep } ?p'$   
**have**  $?v \neq 0$  **by** (rule *proj2-rep-non-zero*)  
  
**have**  $\text{proj2-abs } ?v = ?p'$  **by** (rule *proj2-abs-rep*)  
  
**have**  $?p' \in \text{hyp2}$  **by** (rule *Rep-hyp2*)  
**with**  $\langle ?v \neq 0 \rangle$  **and**  $\langle \text{proj2-abs } ?v = ?p' \rangle$   
**have**  $?v \cdot (M *v ?v) < 0$  **by** (simp add: *K2-imp-M-neg*)  
**hence**  $?v \neq 0$  **by** (rule *M-neg-imp-z-non-zero*)  
**hence**  $\text{proj2-pt } (\text{cart2-pt } ?p') = ?p'$  **by** (rule *proj2-cart2*)  
**thus**  $\text{proj2-pt } (\text{hyp2-rep } p) = ?p'$  **by** (unfold *hyp2-rep-def*)  
**qed**

**lemma** *hyp2-rep-abs*:  
**assumes**  $\text{norm } v < 1$   
**shows**  $\text{hyp2-rep } (\text{hyp2-abs } v) = v$   
**proof** –  
**from**  $\langle \text{norm } v < 1 \rangle$   
**have**  $\text{proj2-pt } v \in \text{hyp2}$  **by** (simp add: *norm-lt-1-iff-in-hyp2*)  
**hence**  $\text{Rep-hyp2 } (\text{Abs-hyp2 } (\text{proj2-pt } v)) = \text{proj2-pt } v$   
**by** (simp add: *Abs-hyp2-inverse*)  
**hence**  $\text{hyp2-rep } (\text{hyp2-abs } v) = \text{cart2-pt } (\text{proj2-pt } v)$   
**by** (unfold *hyp2-rep-def hyp2-abs-def*) simp  
**thus**  $\text{hyp2-rep } (\text{hyp2-abs } v) = v$  **by** (simp add: *cart2-proj2*)  
**qed**

**lemma** *hyp2-abs-rep*:  $\text{hyp2-abs } (\text{hyp2-rep } p) = p$   
**by** (unfold *hyp2-abs-def*) (simp add: *proj2-pt-hyp2-rep Rep-hyp2-inverse*)

**lemma** *norm-hyp2-rep-lt-1*:  $\text{norm } (\text{hyp2-rep } p) < 1$   
**proof** –  
**have**  $\text{proj2-pt } (\text{hyp2-rep } p) = \text{Rep-hyp2 } p$  **by** (rule *proj2-pt-hyp2-rep*)  
**hence**  $\text{proj2-pt } (\text{hyp2-rep } p) \in \text{hyp2}$  **by** (simp add: *Rep-hyp2*)  
**thus**  $\text{norm } (\text{hyp2-rep } p) < 1$  **by** (simp add: *norm-lt-1-iff-in-hyp2*)  
**qed**

**lemma** *hyp2-S-z-non-zero*:  
**assumes**  $p \in \text{hyp2} \cup S$   
**shows**  $z\text{-non-zero } p$   
**proof** –  
**from**  $\langle p \in \text{hyp2} \cup S \rangle$   
**have**  $\text{conic-sgn } p \leq 0$  **by** (unfold *K2-def S-def*) auto  
**hence**  $\text{conic-sgn } p \neq 1$  **by** simp  
**thus**  $z\text{-non-zero } p$  **by** (rule *conic-sgn-not-1-z-non-zero*)  
**qed**

**lemma** *hyp2-S-not-equal*:  
**assumes**  $a \in \text{hyp2}$  **and**  $p \in S$   
**shows**  $a \neq p$

using *assms* and *S-K2-empty*  
by *auto*

**lemma** *hyp2-S-cart2-inj*:

assumes  $p \in \text{hyp2} \cup S$  and  $q \in \text{hyp2} \cup S$  and  $\text{cart2-pt } p = \text{cart2-pt } q$   
shows  $p = q$

**proof** –

from  $\langle p \in \text{hyp2} \cup S \rangle$  and  $\langle q \in \text{hyp2} \cup S \rangle$   
have *z-non-zero*  $p$  and *z-non-zero*  $q$  by (*simp-all add: hyp2-S-z-non-zero*)  
hence  $\text{proj2-pt } (\text{cart2-pt } p) = p$  and  $\text{proj2-pt } (\text{cart2-pt } q) = q$   
by (*simp-all add: proj2-cart2*)

from  $\langle \text{cart2-pt } p = \text{cart2-pt } q \rangle$   
have  $\text{proj2-pt } (\text{cart2-pt } p) = \text{proj2-pt } (\text{cart2-pt } q)$  by *simp*  
with  $\langle \text{proj2-pt } (\text{cart2-pt } p) = p \rangle$  [*symmetric*] and  $\langle \text{proj2-pt } (\text{cart2-pt } q) = q \rangle$   
show  $p = q$  by *simp*

qed

**lemma** *on-equator-in-hyp2-rep*:

assumes  $a \in \text{hyp2}$  and *proj2-incident*  $a$  *equator*  
shows  $\exists x. |x| < 1 \wedge a = \text{proj2-abs } (\text{vector } [x,0,1])$

**proof** –

from  $\langle a \in \text{hyp2} \rangle$  have *z-non-zero*  $a$  by (*simp add: hyp2-S-z-non-zero*)  
with  $\langle \text{proj2-incident } a \text{ equator} \rangle$  and *on-equator-rep*  
obtain  $x$  where  $a = \text{proj2-abs } (\text{vector } [x,0,1])$  (**is**  $a = \text{proj2-abs } ?v$ )  
by *auto*

have  $?v \neq 0$  by (*simp add: vec-eq-iff forall-3 vector-3*)  
with  $\langle a \in \text{hyp2} \rangle$  and  $\langle a = \text{proj2-abs } ?v \rangle$   
have  $?v \cdot (M *v ?v) < 0$  by (*simp add: K2-abs*)  
hence  $x^2 < 1$

unfolding *M-def matrix-vector-mult-def inner-vec-def*  
by (*simp add: sum-3 vector-3 power2-eq-square*)  
with *real-sqrt-abs* [*of x*] and *real-sqrt-less-iff* [*of x<sup>2</sup> 1*]  
have  $|x| < 1$  by *simp*  
with  $\langle a = \text{proj2-abs } ?v \rangle$   
show  $\exists x. |x| < 1 \wedge a = \text{proj2-abs } (\text{vector } [x,0,1])$   
by (*simp add: exI* [*of - x*])

qed

**lemma** *on-meridian-in-hyp2-rep*:

assumes  $a \in \text{hyp2}$  and *proj2-incident*  $a$  *meridian*  
shows  $\exists y. |y| < 1 \wedge a = \text{proj2-abs } (\text{vector } [0,y,1])$

**proof** –

from  $\langle a \in \text{hyp2} \rangle$  have *z-non-zero*  $a$  by (*simp add: hyp2-S-z-non-zero*)  
with  $\langle \text{proj2-incident } a \text{ meridian} \rangle$  and *on-meridian-rep*  
obtain  $y$  where  $a = \text{proj2-abs } (\text{vector } [0,y,1])$  (**is**  $a = \text{proj2-abs } ?v$ )  
by *auto*

**have**  $?v \neq 0$  **by** (*simp add: vec-eq-iff forall-3 vector-3*)  
**with**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle a = \text{proj2-abs } ?v \rangle$   
**have**  $?v \cdot (M *v ?v) < 0$  **by** (*simp add: K2-abs*)  
**hence**  $y^2 < 1$   
**unfolding** *M-def matrix-vector-mult-def inner-vec-def*  
**by** (*simp add: sum-3 vector-3 power2-eq-square*)  
**with** *real-sqrt-abs [of y]* **and** *real-sqrt-less-iff [of y^2 1]*  
**have**  $|y| < 1$  **by** *simp*  
**with**  $\langle a = \text{proj2-abs } ?v \rangle$   
**show**  $\exists y. |y| < 1 \wedge a = \text{proj2-abs } (\text{vector } [0, y, 1])$   
**by** (*simp add: exI [of - y]*)  
**qed**

**definition** *hyp2-cltn2* :: *hyp2*  $\Rightarrow$  *cltn2*  $\Rightarrow$  *hyp2* **where**  
*hyp2-cltn2*  $p$   $A \triangleq \text{Abs-hyp2 } (\text{apply-cltn2 } (\text{Rep-hyp2 } p) A)$

**definition** *is-K2-isometry* :: *cltn2*  $\Rightarrow$  *bool* **where**  
*is-K2-isometry*  $J \triangleq (\forall p. \text{apply-cltn2 } p J \in S \longleftrightarrow p \in S)$

**lemma** *cltn2-id-is-K2-isometry*: *is-K2-isometry* *cltn2-id*  
**unfolding** *is-K2-isometry-def*  
**by** *simp*

**lemma** *J-M-J-transpose-K2-isometry*:  
**assumes**  $k \neq 0$   
**and**  $\text{repJ} ** M ** \text{transpose repJ} = k *_R M$  (**is**  $?N = -$ )  
**shows** *is-K2-isometry* (*cltn2-abs*  $\text{repJ}$ ) (**is** *is-K2-isometry*  $?J$ )

**proof** –  
**from**  $\langle ?N = k *_R M \rangle$   
**have**  $?N ** ((1/k) *_R M) = \text{mat } 1$   
**by** (*simp add: matrix-scalar-ac*  $\langle k \neq 0 \rangle$  *M-self-inverse*)  
**with** *right-invertible-iff-invertible [of repJ]*  
**have** *invertible*  $\text{repJ}$   
**by** (*simp add: matrix-mul-assoc*  
*exI [of - M \*\* transpose repJ \*\* ((1/k) \*\_R M)]*)

**have**  $\forall t. \text{apply-cltn2 } t ?J \in S \longleftrightarrow t \in S$

**proof**  
**fix**  $t :: \text{proj2}$   
**have**  $\text{proj2-rep } t \cdot ((k *_R M) *v \text{proj2-rep } t)$   
 $= k * (\text{proj2-rep } t \cdot (M *v \text{proj2-rep } t))$   
**by** (*simp add: scaleR-matrix-vector-assoc [symmetric] dot-scaleR-mult*)  
**with**  $\langle ?N = k *_R M \rangle$   
**have**  $\text{proj2-rep } t \cdot (?N *v \text{proj2-rep } t)$   
 $= k * (\text{proj2-rep } t \cdot (M *v \text{proj2-rep } t))$   
**by** *simp*  
**hence**  $\text{proj2-rep } t \cdot (?N *v \text{proj2-rep } t) = 0$   
 $\longleftrightarrow k * (\text{proj2-rep } t \cdot (M *v \text{proj2-rep } t)) = 0$   
**by** *simp*

**with**  $\langle k \neq 0 \rangle$   
**have**  $\text{proj2-rep } t \cdot (?N *v \text{proj2-rep } t) = 0$   
 $\longleftrightarrow \text{proj2-rep } t \cdot (M *v \text{proj2-rep } t) = 0$   
**by** *simp*  
**with**  $\langle \text{invertible rep } J \rangle$   
**have**  $\text{apply-cltn2 } t ?J \in S \longleftrightarrow \text{proj2-rep } t \cdot (M *v \text{proj2-rep } t) = 0$   
**by** (*simp add: apply-cltn2-right-abs-in-S*)  
**thus**  $\text{apply-cltn2 } t ?J \in S \longleftrightarrow t \in S$  **by** (*unfold S-alt-def*)  
**qed**  
**thus** *is-K2-isometry*  $?J$  **by** (*unfold is-K2-isometry-def*)  
**qed**

**lemma** *equator-reflect-K2-isometry*:  
**shows** *is-K2-isometry equator-reflect*  
**unfolding** *compass-reflect-defs*  
**by** (*rule J-M-J-transpose-K2-isometry [of 1]*)  
(*simp-all add: M-def matrix-matrix-mult-def transpose-def*  
*vec-eq-iff forall-3 sum-3 vector-3*)

**lemma** *meridian-reflect-K2-isometry*:  
**shows** *is-K2-isometry meridian-reflect*  
**unfolding** *compass-reflect-defs*  
**by** (*rule J-M-J-transpose-K2-isometry [of 1]*)  
(*simp-all add: M-def matrix-matrix-mult-def transpose-def*  
*vec-eq-iff forall-3 sum-3 vector-3*)

**lemma** *cltn2-compose-is-K2-isometry*:  
**assumes** *is-K2-isometry H* **and** *is-K2-isometry J*  
**shows** *is-K2-isometry (cltn2-compose H J)*  
**using**  $\langle \text{is-K2-isometry } H \rangle$  **and**  $\langle \text{is-K2-isometry } J \rangle$   
**unfolding** *is-K2-isometry-def*  
**by** (*simp add: cltn2.act-act [simplified, symmetric]*)

**lemma** *cltn2-inverse-is-K2-isometry*:  
**assumes** *is-K2-isometry J*  
**shows** *is-K2-isometry (cltn2-inverse J)*  
**proof** –  
{ **fix**  $p$   
**from**  $\langle \text{is-K2-isometry } J \rangle$   
**have**  $\text{apply-cltn2 } p (\text{cltn2-inverse } J) \in S$   
 $\longleftrightarrow \text{apply-cltn2 } (\text{apply-cltn2 } p (\text{cltn2-inverse } J)) J \in S$   
**unfolding** *is-K2-isometry-def*  
**by** *simp*  
**hence**  $\text{apply-cltn2 } p (\text{cltn2-inverse } J) \in S \longleftrightarrow p \in S$   
**by** (*simp add: cltn2.act-inv-act [simplified]*) }  
**thus** *is-K2-isometry (cltn2-inverse J)*  
**unfolding** *is-K2-isometry-def ..*  
**qed**



*matrix-scaleR-vector-ac*  
*inner-scaleR-right*  
*scaleR-matrix-vector-assoc* [*symmetric*]  
*M-reverse*  
*power2-eq-square*  
*algebra-simps*)

qed

**lemma** *S-quadratic'*:

**assumes**  $p \neq 0$  **and**  $q \neq 0$  **and**  $\text{proj2-abs } p \neq \text{proj2-abs } q$   
**shows**  $\text{proj2-abs } (k *_R p + q) \in S$   
 $\longleftrightarrow p \cdot (M *v p) * k^2 + p \cdot (M *v q) * 2 * k + q \cdot (M *v q) = 0$

**proof** –

**let**  $?r = k *_R p + q$   
**from**  $\langle p \neq 0 \rangle$  **and**  $\langle q \neq 0 \rangle$  **and**  $\langle \text{proj2-abs } p \neq \text{proj2-abs } q \rangle$   
**and** *dependent-proj2-abs* [*of p q k 1*]  
**have**  $?r \neq 0$  **by** *auto*  
**hence**  $\text{proj2-abs } ?r \in S \longleftrightarrow ?r \cdot (M *v ?r) = 0$  **by** (*rule S-abs*)  
**with** *collinear-quadratic* [*of ?r k p q*]  
**show**  $\text{proj2-abs } ?r \in S$   
 $\longleftrightarrow p \cdot (M *v p) * k^2 + p \cdot (M *v q) * 2 * k + q \cdot (M *v q) = 0$   
**by** (*simp add: dot-lmul-matrix* [*symmetric*] *algebra-simps*)

qed

**lemma** *S-quadratic*:

**assumes**  $p \neq q$  **and**  $r = \text{proj2-abs } (k *_R \text{proj2-rep } p + \text{proj2-rep } q)$   
**shows**  $r \in S$   
 $\longleftrightarrow \text{proj2-rep } p \cdot (M *v \text{proj2-rep } p) * k^2$   
 $\quad + \text{proj2-rep } p \cdot (M *v \text{proj2-rep } q) * 2 * k$   
 $\quad + \text{proj2-rep } q \cdot (M *v \text{proj2-rep } q)$   
 $= 0$

**proof** –

**let**  $?u = \text{proj2-rep } p$   
**let**  $?v = \text{proj2-rep } q$   
**let**  $?w = k *_R ?u + ?v$   
**have**  $?u \neq 0$  **and**  $?v \neq 0$  **by** (*rule proj2-rep-non-zero*)  
  
**from**  $\langle p \neq q \rangle$  **have**  $\text{proj2-abs } ?u \neq \text{proj2-abs } ?v$  **by** (*simp add: proj2-abs-rep*)  
**with**  $\langle ?u \neq 0 \rangle$  **and**  $\langle ?v \neq 0 \rangle$  **and**  $\langle r = \text{proj2-abs } ?w \rangle$   
**show**  $r \in S$   
 $\longleftrightarrow ?u \cdot (M *v ?u) * k^2 + ?u \cdot (M *v ?v) * 2 * k + ?v \cdot (M *v ?v) = 0$   
**by** (*simp add: S-quadratic'*)

qed

**definition** *quarter-discrim* ::  $\text{real}^3 \Rightarrow \text{real}^3 \Rightarrow \text{real}$  **where**

*quarter-discrim*  $p \ q \triangleq (p \cdot (M *v q))^2 - p \cdot (M *v p) * (q \cdot (M *v q))$

**lemma** *quarter-discrim-invariant*:

**assumes**  $t = i *_R a + r$

shows *quarter-discrim a t = quarter-discrim a r*  
**proof** –  
**from**  $\langle t = i *_{R} a + r \rangle$   
**have**  $a \cdot (M *v t) = i * (a \cdot (M *v a)) + a \cdot (M *v r)$   
**by** (*simp add:*  
*matrix-vector-right-distrib*  
*inner-add-right*  
*matrix-scaleR-vector-ac*  
*scaleR-matrix-vector-assoc [symmetric]*)  
**hence**  $(a \cdot (M *v t))^2 =$   
 $(a \cdot (M *v a))^2 * i^2 +$   
 $2 * (a \cdot (M *v a)) * (a \cdot (M *v r)) * i +$   
 $(a \cdot (M *v r))^2$   
**by** (*simp add: power2-eq-square algebra-simps*)  
**moreover from** *collinear-quadratic* **and**  $\langle t = i *_{R} a + r \rangle$   
**have**  $a \cdot (M *v a) * (t \cdot (M *v t)) =$   
 $(a \cdot (M *v a))^2 * i^2 +$   
 $2 * (a \cdot (M *v a)) * (a \cdot (M *v r)) * i +$   
 $a \cdot (M *v a) * (r \cdot (M *v r))$   
**by** (*simp add: power2-eq-square algebra-simps*)  
**ultimately show** *quarter-discrim a t = quarter-discrim a r*  
**by** (*unfold quarter-discrim-def, simp*)  
**qed**

**lemma** *quarter-discrim-positive:*

**assumes**  $p \neq 0$  **and**  $q \neq 0$  **and** *proj2-abs p ≠ proj2-abs q (is ?pp ≠ ?pq)*  
**and** *proj2-abs p ∈ K2*  
**shows** *quarter-discrim p q > 0*

**proof** –

**let**  $?i = -q^3/p^3$   
**let**  $?t = ?i *_{R} p + q$

**from**  $\langle p \neq 0 \rangle$  **and**  $\langle ?pp \in K2 \rangle$   
**have**  $p \cdot (M *v p) < 0$  **by** (*subst K2-abs [symmetric]*)  
**hence**  $p^3 \neq 0$  **by** (*rule M-neg-imp-z-non-zero*)  
**hence**  $?t^3 = 0$  **by** *simp*  
**hence**  $?t \cdot (M *v ?t) = (?t^1)^2 + (?t^2)^2$   
**unfolding** *matrix-vector-mult-def* **and** *M-def* **and** *vector-def*  
**by** (*simp add: inner-vec-def sum-3 power2-eq-square*)

**from**  $\langle p^3 \neq 0 \rangle$  **have**  $p \neq 0$  **by** *auto*  
**with**  $\langle q \neq 0 \rangle$  **and**  $\langle ?pp \neq ?pq \rangle$  **and** *dependent-proj2-abs [of p q ?i 1]*  
**have**  $?t \neq 0$  **by** *auto*  
**with**  $\langle ?t^3 = 0 \rangle$  **have**  $?t^1 \neq 0 \vee ?t^2 \neq 0$  **by** (*simp add: vec-eq-iff forall-3*)  
**hence**  $(?t^1)^2 > 0 \vee (?t^2)^2 > 0$  **by** *simp*  
**moreover have**  $(?t^2)^2 \geq 0$  **and**  $(?t^1)^2 \geq 0$  **by** *simp-all*  
**ultimately have**  $(?t^1)^2 + (?t^2)^2 > 0$  **by** *arith*  
**with**  $\langle ?t \cdot (M *v ?t) = (?t^1)^2 + (?t^2)^2 \rangle$  **have**  $?t \cdot (M *v ?t) > 0$  **by** *simp*  
**with** *mult-neg-pos [of p · (M \*v p)]* **and**  $\langle p \cdot (M *v p) < 0 \rangle$

**have**  $p \cdot (M *v p) * (?t \cdot (M *v ?t)) < 0$  **by** *simp*  
**moreover have**  $(p \cdot (M *v ?t))^2 \geq 0$  **by** *simp*  
**ultimately**  
**have**  $(p \cdot (M *v ?t))^2 - p \cdot (M *v p) * (?t \cdot (M *v ?t)) > 0$  **by** *arith*  
**with** *quarter-discrim-invariant* [of ?t ?i p q]  
**show** *quarter-discrim* p q > 0 **by** (*unfold quarter-discrim-def*, *simp*)  
**qed**

**lemma** *quarter-discrim-self-zero*:  
**assumes** *proj2-abs* a = *proj2-abs* b  
**shows** *quarter-discrim* a b = 0  
**proof** *cases*  
**assume** b = 0  
**thus** *quarter-discrim* a b = 0 **by** (*unfold quarter-discrim-def*, *simp*)  
**next**  
**assume** b  $\neq$  0  
**with**  $\langle$ *proj2-abs* a = *proj2-abs* b  $\rangle$  **and** *proj2-abs-abs-mult*  
**obtain** k **where** a = k \*<sub>R</sub> b **by** *auto*  
**thus** *quarter-discrim* a b = 0  
**unfolding** *quarter-discrim-def*  
**by** (*simp add: power2-eq-square*  
*matrix-scaleR-vector-ac*  
*scaleR-matrix-vector-assoc* [*symmetric*])  
**qed**

**definition** *S-intersection-coeff1* ::  $\text{real}^3 \Rightarrow \text{real}^3 \Rightarrow \text{real}$  **where**  
*S-intersection-coeff1* p q  
 $\triangleq (-p \cdot (M *v q) + \text{sqrt} (\text{quarter-discrim } p \ q)) / (p \cdot (M *v p))$

**definition** *S-intersection-coeff2* ::  $\text{real}^3 \Rightarrow \text{real}^3 \Rightarrow \text{real}$  **where**  
*S-intersection-coeff2* p q  
 $\triangleq (-p \cdot (M *v q) - \text{sqrt} (\text{quarter-discrim } p \ q)) / (p \cdot (M *v p))$

**definition** *S-intersection1-rep* ::  $\text{real}^3 \Rightarrow \text{real}^3 \Rightarrow \text{real}^3$  **where**  
*S-intersection1-rep* p q  $\triangleq (S\text{-intersection-coeff1 } p \ q) *_{\mathbb{R}} p + q$

**definition** *S-intersection2-rep* ::  $\text{real}^3 \Rightarrow \text{real}^3 \Rightarrow \text{real}^3$  **where**  
*S-intersection2-rep* p q  $\triangleq (S\text{-intersection-coeff2 } p \ q) *_{\mathbb{R}} p + q$

**definition** *S-intersection1* ::  $\text{real}^3 \Rightarrow \text{real}^3 \Rightarrow \text{proj2}$  **where**  
*S-intersection1* p q  $\triangleq \text{proj2-abs } (S\text{-intersection1-rep } p \ q)$

**definition** *S-intersection2* ::  $\text{real}^3 \Rightarrow \text{real}^3 \Rightarrow \text{proj2}$  **where**  
*S-intersection2* p q  $\triangleq \text{proj2-abs } (S\text{-intersection2-rep } p \ q)$

**lemmas** *S-intersection-coeffs-defs* =  
*S-intersection-coeff1-def* *S-intersection-coeff2-def*

**lemmas** *S-intersections-defs* =

*S-intersection1-def S-intersection2-def*  
*S-intersection1-rep-def S-intersection2-rep-def*

**lemma** *S-intersection-coeffs-distinct*:

**assumes**  $p \neq 0$  **and**  $q \neq 0$  **and**  $\text{proj2-abs } p \neq \text{proj2-abs } q$  (**is**  $?pp \neq ?pq$ )  
**and**  $\text{proj2-abs } p \in K2$   
**shows**  $S\text{-intersection-coeff1 } p \ q \neq S\text{-intersection-coeff2 } p \ q$

**proof** –

**from**  $\langle p \neq 0 \rangle$  **and**  $\langle ?pp \in K2 \rangle$   
**have**  $p \cdot (M *v p) < 0$  **by** (*subst K2-abs [symmetric]*)

**from** *assms* **have**  $\text{quarter-discrim } p \ q > 0$  **by** (*rule quarter-discrim-positive*)  
**with**  $\langle p \cdot (M *v p) < 0 \rangle$   
**show**  $S\text{-intersection-coeff1 } p \ q \neq S\text{-intersection-coeff2 } p \ q$   
**by** (*unfold S-intersection-coeffs-defs, simp*)

**qed**

**lemma** *S-intersections-distinct*:

**assumes**  $p \neq 0$  **and**  $q \neq 0$  **and**  $\text{proj2-abs } p \neq \text{proj2-abs } q$  (**is**  $?pp \neq ?pq$ )  
**and**  $\text{proj2-abs } p \in K2$   
**shows**  $S\text{-intersection1 } p \ q \neq S\text{-intersection2 } p \ q$

**proof** –

**from**  $\langle p \neq 0 \rangle$  **and**  $\langle q \neq 0 \rangle$  **and**  $\langle ?pp \neq ?pq \rangle$  **and**  $\langle ?pp \in K2 \rangle$   
**have**  $S\text{-intersection-coeff1 } p \ q \neq S\text{-intersection-coeff2 } p \ q$   
**by** (*rule S-intersection-coeffs-distinct*)  
**with**  $\langle p \neq 0 \rangle$  **and**  $\langle q \neq 0 \rangle$  **and**  $\langle ?pp \neq ?pq \rangle$  **and** *proj2-Col-coeff-unique'*  
**show**  $S\text{-intersection1 } p \ q \neq S\text{-intersection2 } p \ q$   
**by** (*unfold S-intersections-defs, auto*)

**qed**

**lemma** *S-intersections-in-S*:

**assumes**  $p \neq 0$  **and**  $q \neq 0$  **and**  $\text{proj2-abs } p \neq \text{proj2-abs } q$  (**is**  $?pp \neq ?pq$ )  
**and**  $\text{proj2-abs } p \in K2$   
**shows**  $S\text{-intersection1 } p \ q \in S$  **and**  $S\text{-intersection2 } p \ q \in S$

**proof** –

**let**  $?j = S\text{-intersection-coeff1 } p \ q$   
**let**  $?k = S\text{-intersection-coeff2 } p \ q$   
**let**  $?a = p \cdot (M *v p)$   
**let**  $?b = 2 * (p \cdot (M *v q))$   
**let**  $?c = q \cdot (M *v q)$

**from**  $\langle p \neq 0 \rangle$  **and**  $\langle ?pp \in K2 \rangle$  **have**  $?a < 0$  **by** (*subst K2-abs [symmetric]*)

**have**  $qd: \text{discrim } ?a \ ?b \ ?c = 4 * \text{quarter-discrim } p \ q$   
**unfolding** *discrim-def quarter-discrim-def*  
**by** (*simp add: power2-eq-square*)  
**with** *times-divide-times-eq* [of  
 $2 \ 2 \ \text{sqrt } (\text{quarter-discrim } p \ q) - p \cdot (M *v q) \ ?a$ ]  
**and** *times-divide-times-eq* [of

$2 \cdot 2 - p \cdot (M * v \ q) - \text{sqrt}(\text{quarter-discrim } p \ q) \ ?a]$   
**and** *real-sqrt-mult* **and** *real-sqrt-abs* [of 2]  
**have**  $?j = (-?b + \text{sqrt}(\text{discrim } ?a \ ?b \ ?c)) / (2 * ?a)$   
**and**  $?k = (-?b - \text{sqrt}(\text{discrim } ?a \ ?b \ ?c)) / (2 * ?a)$   
**by** (*unfold S-intersection-coeffs-defs, simp-all add: algebra-simps*)

**from** *assms* **have** *quarter-discrim*  $p \ q > 0$  **by** (*rule quarter-discrim-positive*)  
**with** *qd*  
**have** *discrim*  $(p \cdot (M * v \ p)) (2 * (p \cdot (M * v \ q))) (q \cdot (M * v \ q)) > 0$   
**by** *simp*  
**with**  $\langle ?j = (-?b + \text{sqrt}(\text{discrim } ?a \ ?b \ ?c)) / (2 * ?a) \rangle$   
**and**  $\langle ?k = (-?b - \text{sqrt}(\text{discrim } ?a \ ?b \ ?c)) / (2 * ?a) \rangle$   
**and**  $\langle ?a < 0 \rangle$  **and** *discriminant-nonneg* [of  $?a \ ?b \ ?c \ ?j$ ]  
**and** *discriminant-nonneg* [of  $?a \ ?b \ ?c \ ?k$ ]  
**have**  $p \cdot (M * v \ p) * ?j^2 + 2 * (p \cdot (M * v \ q)) * ?j + q \cdot (M * v \ q) = 0$   
**and**  $p \cdot (M * v \ p) * ?k^2 + 2 * (p \cdot (M * v \ q)) * ?k + q \cdot (M * v \ q) = 0$   
**by** (*unfold S-intersection-coeffs-defs, auto*)  
**with**  $\langle p \neq 0 \rangle$  **and**  $\langle q \neq 0 \rangle$  **and**  $\langle ?pp \neq ?pq \rangle$  **and** *S-quadratic'*  
**show** *S-intersection1*  $p \ q \in S$  **and** *S-intersection2*  $p \ q \in S$   
**by** (*unfold S-intersections-defs, simp-all*)  
**qed**

**lemma** *S-intersections-Col*:  
**assumes**  $p \neq 0$  **and**  $q \neq 0$   
**shows** *proj2-Col* (*proj2-abs*  $p$ ) (*proj2-abs*  $q$ ) (*S-intersection1*  $p \ q$ )  
(is *proj2-Col*  $?pp \ ?pq \ ?pr$ )  
**and** *proj2-Col* (*proj2-abs*  $p$ ) (*proj2-abs*  $q$ ) (*S-intersection2*  $p \ q$ )  
(is *proj2-Col*  $?pp \ ?pq \ ?ps$ )  
**proof** –  
{ **assume**  $?pp = ?pq$   
**hence** *proj2-Col*  $?pp \ ?pq \ ?pr$  **and** *proj2-Col*  $?pp \ ?pq \ ?ps$   
**by** (*simp-all add: proj2-Col-coincide*) }  
**moreover**  
{ **assume**  $?pp \neq ?pq$   
**with**  $\langle p \neq 0 \rangle$  **and**  $\langle q \neq 0 \rangle$  **and** *dependent-proj2-abs* [of  $p \ q \ 1$ ]  
**have** *S-intersection1-rep*  $p \ q \neq 0$  (is  $?r \neq 0$ )  
**and** *S-intersection2-rep*  $p \ q \neq 0$  (is  $?s \neq 0$ )  
**by** (*unfold S-intersection1-rep-def S-intersection2-rep-def, auto*)  
**with**  $\langle p \neq 0 \rangle$  **and**  $\langle q \neq 0 \rangle$   
**and** *proj2-Col-abs* [of  $p \ q \ ?r \ S\text{-intersection-coeff1 } p \ q \ 1 \ -1$ ]  
**and** *proj2-Col-abs* [of  $p \ q \ ?s \ S\text{-intersection-coeff2 } p \ q \ 1 \ -1$ ]  
**have** *proj2-Col*  $?pp \ ?pq \ ?pr$  **and** *proj2-Col*  $?pp \ ?pq \ ?ps$   
**by** (*unfold S-intersections-defs, simp-all*) }  
**ultimately show** *proj2-Col*  $?pp \ ?pq \ ?pr$  **and** *proj2-Col*  $?pp \ ?pq \ ?ps$  **by** *fast+*  
**qed**

**lemma** *S-intersections-incident*:  
**assumes**  $p \neq 0$  **and**  $q \neq 0$  **and** *proj2-abs*  $p \neq \text{proj2-abs } q$  (is  $?pp \neq ?pq$ )  
**and** *proj2-incident* (*proj2-abs*  $p$ )  $l$  **and** *proj2-incident* (*proj2-abs*  $q$ )  $l$

**shows** *proj2-incident* (*S-intersection1* p q) l (**is** *proj2-incident* ?pr l)  
**and** *proj2-incident* (*S-intersection2* p q) l (**is** *proj2-incident* ?ps l)  
**proof** –  
**from**  $\langle p \neq 0 \rangle$  **and**  $\langle q \neq 0 \rangle$   
**have** *proj2-Col* ?pp ?pq ?pr **and** *proj2-Col* ?pp ?pq ?ps  
**by** (*rule S-intersections-Col*)+  
**with**  $\langle ?pp \neq ?pq \rangle$  **and**  $\langle \text{proj2-incident } ?pp \text{ l} \rangle$  **and**  $\langle \text{proj2-incident } ?pq \text{ l} \rangle$   
**and** *proj2-incident-iff-Col*  
**show** *proj2-incident* ?pr l **and** *proj2-incident* ?ps l **by** *fast+*  
**qed**

**lemma** *K2-line-intersect-twice*:

**assumes**  $a \in K2$  **and**  $a \neq r$

**shows**  $\exists s u. s \neq u \wedge s \in S \wedge u \in S \wedge \text{proj2-Col } a \ r \ s \wedge \text{proj2-Col } a \ r \ u$

**proof** –

**let** ?a' = *proj2-rep* a

**let** ?r' = *proj2-rep* r

**from** *proj2-rep-non-zero* **have** ?a'  $\neq 0$  **and** ?r'  $\neq 0$  **by** *simp-all*

**from**  $\langle ?a' \neq 0 \rangle$  **and** *K2-imp-M-neg* **and** *proj2-abs-rep* **and**  $\langle a \in K2 \rangle$

**have** ?a'  $\cdot (M * v \ ?a') < 0$  **by** *simp*

**from**  $\langle a \neq r \rangle$  **have** *proj2-abs* ?a'  $\neq$  *proj2-abs* ?r' **by** (*simp add: proj2-abs-rep*)

**from**  $\langle a \in K2 \rangle$  **have** *proj2-abs* ?a'  $\in K2$  **by** (*simp add: proj2-abs-rep*)

**with**  $\langle ?a' \neq 0 \rangle$  **and**  $\langle ?r' \neq 0 \rangle$  **and**  $\langle \text{proj2-abs } ?a' \neq \text{proj2-abs } ?r' \rangle$

**have** *S-intersection1* ?a' ?r'  $\neq$  *S-intersection2* ?a' ?r' (**is** ?s  $\neq$  ?u)

**by** (*rule S-intersections-distinct*)

**from**  $\langle ?a' \neq 0 \rangle$  **and**  $\langle ?r' \neq 0 \rangle$  **and**  $\langle \text{proj2-abs } ?a' \neq \text{proj2-abs } ?r' \rangle$

**and**  $\langle \text{proj2-abs } ?a' \in K2 \rangle$

**have** ?s  $\in S$  **and** ?u  $\in S$  **by** (*rule S-intersections-in-S*)+

**from**  $\langle ?a' \neq 0 \rangle$  **and**  $\langle ?r' \neq 0 \rangle$

**have** *proj2-Col* (*proj2-abs* ?a') (*proj2-abs* ?r') ?s

**and** *proj2-Col* (*proj2-abs* ?a') (*proj2-abs* ?r') ?u

**by** (*rule S-intersections-Col*)+

**hence** *proj2-Col* a r ?s **and** *proj2-Col* a r ?u

**by** (*simp-all add: proj2-abs-rep*)

**with**  $\langle ?s \neq ?u \rangle$  **and**  $\langle ?s \in S \rangle$  **and**  $\langle ?u \in S \rangle$

**show**  $\exists s u. s \neq u \wedge s \in S \wedge u \in S \wedge \text{proj2-Col } a \ r \ s \wedge \text{proj2-Col } a \ r \ u$

**by** *auto*

**qed**

**lemma** *point-in-S-polar-is-tangent*:

**assumes**  $p \in S$  **and**  $q \in S$  **and** *proj2-incident* q (*polar* p)

**shows** q = p

**proof** –

**from**  $\langle p \in S \rangle$  **have** *proj2-incident* p (*polar* p)

by (*subst incident-own-polar-in-S*)  
 from *line-incident-point-not-in-S*  
 obtain  $r$  where  $r \notin S$  and *proj2-incident*  $r$  (*polar*  $p$ ) by *auto*  
 let  $?u = \text{proj2-rep } r$   
 let  $?v = \text{proj2-rep } p$   
 from  $\langle r \notin S \rangle$  and  $\langle p \in S \rangle$  and  $\langle q \in S \rangle$  have  $r \neq p$  and  $q \neq r$  by *auto*  
 with  $\langle \text{proj2-incident } p \text{ (polar } p) \rangle$   
   and  $\langle \text{proj2-incident } q \text{ (polar } p) \rangle$   
   and  $\langle \text{proj2-incident } r \text{ (polar } p) \rangle$   
   and *proj2-incident-iff* [*of*  $r$   $p$  *polar*  $p$   $q$ ]  
 obtain  $k$  where  $q = \text{proj2-abs } (k *_{\mathbb{R}} ?u + ?v)$  by *auto*  
 with  $\langle r \neq p \rangle$  and  $\langle q \in S \rangle$  and *S-quadratic*  
 have  $?u \cdot (M *v ?u) * k^2 + ?u \cdot (M *v ?v) * 2 * k + ?v \cdot (M *v ?v) = 0$   
   by *simp*  
 moreover from  $\langle p \in S \rangle$  have  $?v \cdot (M *v ?v) = 0$  by (*unfold S-alt-def*)  
 moreover from  $\langle \text{proj2-incident } r \text{ (polar } p) \rangle$   
 have  $?u \cdot (M *v ?v) = 0$  by (*unfold incident-polar*)  
 moreover from  $\langle r \notin S \rangle$  have  $?u \cdot (M *v ?u) \neq 0$  by (*unfold S-alt-def*)  
 ultimately have  $k = 0$  by *simp*  
 with  $\langle q = \text{proj2-abs } (k *_{\mathbb{R}} ?u + ?v) \rangle$   
 show  $q = p$  by (*simp add: proj2-abs-rep*)  
 qed

**lemma** *line-through-K2-intersect-S-twice*:  
 assumes  $p \in K2$  and *proj2-incident*  $p$   $l$   
 shows  $\exists q r. q \neq r \wedge q \in S \wedge r \in S \wedge \text{proj2-incident } q$   $l \wedge \text{proj2-incident } r$   $l$   
**proof** –  
 from *proj2-another-point-on-line*  
 obtain  $s$  where  $s \neq p$  and *proj2-incident*  $s$   $l$  by *auto*  
 from  $\langle p \in K2 \rangle$  and  $\langle s \neq p \rangle$  and *K2-line-intersect-twice* [*of*  $p$   $s$ ]  
 obtain  $q$  and  $r$  where  $q \neq r$  and  $q \in S$  and  $r \in S$   
   and *proj2-Col*  $p$   $s$   $q$  and *proj2-Col*  $p$   $s$   $r$   
   by *auto*  
 with  $\langle s \neq p \rangle$  and  $\langle \text{proj2-incident } p$   $l \rangle$  and  $\langle \text{proj2-incident } s$   $l \rangle$   
   and *proj2-incident-iff-Col* [*of*  $p$   $s$ ]  
 have *proj2-incident*  $q$   $l$  and *proj2-incident*  $r$   $l$  by *fast+*  
 with  $\langle q \neq r \rangle$  and  $\langle q \in S \rangle$  and  $\langle r \in S \rangle$   
 show  $\exists q r. q \neq r \wedge q \in S \wedge r \in S \wedge \text{proj2-incident } q$   $l \wedge \text{proj2-incident } r$   $l$   
   by *auto*  
 qed

**lemma** *line-through-K2-intersect-S-again*:  
 assumes  $p \in K2$  and *proj2-incident*  $p$   $l$   
 shows  $\exists r. r \neq q \wedge r \in S \wedge \text{proj2-incident } r$   $l$   
**proof** –  
 from  $\langle p \in K2 \rangle$  and  $\langle \text{proj2-incident } p$   $l \rangle$   
   and *line-through-K2-intersect-S-twice* [*of*  $p$   $l$ ]  
 obtain  $s$  and  $t$  where  $s \neq t$  and  $s \in S$  and  $t \in S$

**and** *proj2-incident s l* **and** *proj2-incident t l*  
**by** *auto*  
**show**  $\exists r. r \neq q \wedge r \in S \wedge \text{proj2-incident } r \ l$   
**proof** *cases*  
**assume**  $t = q$   
**with**  $\langle s \neq t \rangle$  **and**  $\langle s \in S \rangle$  **and**  $\langle \text{proj2-incident } s \ l \rangle$   
**have**  $s \neq q \wedge s \in S \wedge \text{proj2-incident } s \ l$  **by** *simp*  
**thus**  $\exists r. r \neq q \wedge r \in S \wedge \text{proj2-incident } r \ l$  ..  
**next**  
**assume**  $t \neq q$   
**with**  $\langle t \in S \rangle$  **and**  $\langle \text{proj2-incident } t \ l \rangle$   
**have**  $t \neq q \wedge t \in S \wedge \text{proj2-incident } t \ l$  **by** *simp*  
**thus**  $\exists r. r \neq q \wedge r \in S \wedge \text{proj2-incident } r \ l$  ..  
**qed**  
**qed**

**lemma** *line-through-K2-intersect-S*:  
**assumes**  $p \in K2$  **and** *proj2-incident p l*  
**shows**  $\exists r. r \in S \wedge \text{proj2-incident } r \ l$   
**proof** –  
**from** *assms*  
**have**  $\exists r. r \neq p \wedge r \in S \wedge \text{proj2-incident } r \ l$   
**by** (*rule line-through-K2-intersect-S-again*)  
**thus**  $\exists r. r \in S \wedge \text{proj2-incident } r \ l$  **by** *auto*  
**qed**

**lemma** *line-intersect-S-at-most-twice*:  
 $\exists p \ q. \forall r \in S. \text{proj2-incident } r \ l \longrightarrow r = p \vee r = q$   
**proof** –  
**from** *line-incident-point-not-in-S*  
**obtain**  $s$  **where**  $s \notin S$  **and** *proj2-incident s l* **by** *auto*  
**let**  $?v = \text{proj2-rep } s$   
**from** *proj2-another-point-on-line*  
**obtain**  $t$  **where**  $t \neq s$  **and** *proj2-incident t l* **by** *auto*  
**let**  $?w = \text{proj2-rep } t$   
**have**  $?v \neq 0$  **and**  $?w \neq 0$  **by** (*rule proj2-rep-non-zero*)  
  
**let**  $?a = ?v \cdot (M *v ?v)$   
**let**  $?b = 2 * (?v \cdot (M *v ?w))$   
**let**  $?c = ?w \cdot (M *v ?w)$   
**from**  $\langle s \notin S \rangle$  **have**  $?a \neq 0$   
**unfolding** *S-def* **and** *conic-sgn-def*  
**by** *auto*  
**let**  $?j = (-?b + \text{sqrt } (\text{discrim } ?a \ ?b \ ?c)) / (2 * ?a)$   
**let**  $?k = (-?b - \text{sqrt } (\text{discrim } ?a \ ?b \ ?c)) / (2 * ?a)$   
**let**  $?p = \text{proj2-abs } (?j *_R ?v + ?w)$   
**let**  $?q = \text{proj2-abs } (?k *_R ?v + ?w)$   
**have**  $\forall r \in S. \text{proj2-incident } r \ l \longrightarrow r = ?p \vee r = ?q$   
**proof**

**fix**  $r$   
**assume**  $r \in S$   
**with**  $\langle s \notin S \rangle$  **have**  $r \neq s$  **by** *auto*  
**{** **assume** *proj2-incident*  $r\ l$   
**with**  $\langle t \neq s \rangle$  **and**  $\langle r \neq s \rangle$  **and**  $\langle \text{proj2-incident } s\ l \rangle$  **and**  $\langle \text{proj2-incident } t\ l \rangle$   
**and** *proj2-incident-iff* [of  $s\ t\ l\ r$ ]  
**obtain**  $i$  **where**  $r = \text{proj2-abs } (i *_R ?v + ?w)$  **by** *auto*  
**with**  $\langle r \in S \rangle$  **and**  $\langle t \neq s \rangle$  **and** *S-quadratic*  
**have**  $?a * i^2 + ?b * i + ?c = 0$  **by** *simp*  
**with**  $\langle ?a \neq 0 \rangle$  **and** *discriminant-iff* **have**  $i = ?j \vee i = ?k$  **by** *simp*  
**with**  $\langle r = \text{proj2-abs } (i *_R ?v + ?w) \rangle$  **have**  $r = ?p \vee r = ?q$  **by** *auto* }  
**thus** *proj2-incident*  $r\ l \longrightarrow r = ?p \vee r = ?q$  ..  
**qed**  
**thus**  $\exists p\ q. \forall r \in S. \text{proj2-incident } r\ l \longrightarrow r = p \vee r = q$  **by** *auto*  
**qed**

**lemma** *card-line-intersect-S*:

**assumes**  $T \subseteq S$  **and** *proj2-set-Col*  $T$   
**shows**  $\text{card } T \leq 2$

**proof** –

**from**  $\langle \text{proj2-set-Col } T \rangle$   
**obtain**  $l$  **where**  $\forall p \in T. \text{proj2-incident } p\ l$  **unfolding** *proj2-set-Col-def* ..  
**from** *line-intersect-S-at-most-twice* [of  $l$ ]  
**obtain**  $b$  **and**  $c$  **where**  $\forall a \in S. \text{proj2-incident } a\ l \longrightarrow a = b \vee a = c$  **by** *auto*  
**with**  $\langle \forall p \in T. \text{proj2-incident } p\ l \rangle$  **and**  $\langle T \subseteq S \rangle$   
**have**  $T \subseteq \{b, c\}$  **by** *auto*  
**hence**  $\text{card } T \leq \text{card } \{b, c\}$  **by** (*simp add: card-mono*)  
**also from** *card-suc-ge-insert* [of  $b\ \{c\}$ ] **have**  $\dots \leq 2$  **by** *simp*  
**finally show**  $\text{card } T \leq 2$  .

**qed**

**lemma** *line-S-two-intersections-only*:

**assumes**  $p \neq q$  **and**  $p \in S$  **and**  $q \in S$  **and**  $r \in S$   
**and** *proj2-incident*  $p\ l$  **and** *proj2-incident*  $q\ l$  **and** *proj2-incident*  $r\ l$   
**shows**  $r = p \vee r = q$

**proof** –

**from**  $\langle p \neq q \rangle$  **have**  $\text{card } \{p, q\} = 2$  **by** *simp*

**from**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$  **and**  $\langle r \in S \rangle$  **have**  $\{r, p, q\} \subseteq S$  **by** *simp-all*

**from**  $\langle \text{proj2-incident } p\ l \rangle$  **and**  $\langle \text{proj2-incident } q\ l \rangle$  **and**  $\langle \text{proj2-incident } r\ l \rangle$   
**have** *proj2-set-Col*  $\{r, p, q\}$

**by** (*unfold proj2-set-Col-def*) (*simp add: exI* [of  $l$ ])

**with**  $\langle \{r, p, q\} \subseteq S \rangle$  **have**  $\text{card } \{r, p, q\} \leq 2$  **by** (*rule card-line-intersect-S*)

**show**  $r = p \vee r = q$

**proof** (*rule ccontr*)

**assume**  $\neg (r = p \vee r = q)$

**hence**  $r \notin \{p, q\}$  **by** *simp*

**with**  $\langle \text{card } \{p, q\} = 2 \rangle$  **and** *card-insert-disjoint* [of  $\{p, q\}$   $r$ ]  
**have**  $\text{card } \{r, p, q\} = 3$  **by** *simp*  
**with**  $\langle \text{card } \{r, p, q\} \leq 2 \rangle$  **show** *False* **by** *simp*  
**qed**  
**qed**

**lemma** *line-through-K2-intersect-S-exactly-twice*:

**assumes**  $p \in K2$  **and** *proj2-incident*  $p$   $l$   
**shows**  $\exists q r. q \neq r \wedge q \in S \wedge r \in S \wedge \text{proj2-incident } q$   $l \wedge \text{proj2-incident } r$   $l$   
 $\wedge (\forall s \in S. \text{proj2-incident } s$   $l \longrightarrow s = q \vee s = r)$   
**proof** –  
**from**  $\langle p \in K2 \rangle$  **and**  $\langle \text{proj2-incident } p$   $l \rangle$   
**and** *line-through-K2-intersect-S-twice* [of  $p$   $l$ ]  
**obtain**  $q$  **and**  $r$  **where**  $q \neq r$  **and**  $q \in S$  **and**  $r \in S$   
**and** *proj2-incident*  $q$   $l$  **and** *proj2-incident*  $r$   $l$   
**by** *auto*  
**with** *line-S-two-intersections-only*  
**show**  $\exists q r. q \neq r \wedge q \in S \wedge r \in S \wedge \text{proj2-incident } q$   $l \wedge \text{proj2-incident } r$   $l$   
 $\wedge (\forall s \in S. \text{proj2-incident } s$   $l \longrightarrow s = q \vee s = r)$   
**by** *blast*  
**qed**

**lemma** *tangent-not-through-K2*:

**assumes**  $p \in S$  **and**  $q \in K2$   
**shows**  $\neg \text{proj2-incident } q$  (*polar*  $p$ )  
**proof**  
**assume** *proj2-incident*  $q$  (*polar*  $p$ )  
**with**  $\langle q \in K2 \rangle$  **and** *line-through-K2-intersect-S-again* [of  $q$  *polar*  $p$   $p$ ]  
**obtain**  $r$  **where**  $r \neq p$  **and**  $r \in S$  **and** *proj2-incident*  $r$  (*polar*  $p$ ) **by** *auto*  
**from**  $\langle p \in S \rangle$  **and**  $\langle r \in S \rangle$  **and**  $\langle \text{proj2-incident } r$  (*polar*  $p$ )  $\rangle$   
**have**  $r = p$  **by** (*rule point-in-S-polar-is-tangent*)  
**with**  $\langle r \neq p \rangle$  **show** *False* ..  
**qed**

**lemma** *outside-exists-line-not-intersect-S*:

**assumes** *conic-sgn*  $p = 1$   
**shows**  $\exists l. \text{proj2-incident } p$   $l \wedge (\forall q. \text{proj2-incident } q$   $l \longrightarrow q \notin S)$   
**proof** –  
**let**  $?r = \text{proj2-intersection}$  (*polar*  $p$ )  $z\text{-zero}$   
**have** *proj2-incident*  $?r$  (*polar*  $p$ ) **and** *proj2-incident*  $?r$   $z\text{-zero}$   
**by** (*rule proj2-intersection-incident*)**+**  
**from**  $\langle \text{proj2-incident } ?r$   $z\text{-zero} \rangle$   
**have** *conic-sgn*  $?r = 1$  **by** (*rule z-zero-conic-sgn-1*)  
**with**  $\langle \text{conic-sgn } p = 1 \rangle$   
**have** *proj2-rep*  $p \cdot (M *v \text{proj2-rep } p) > 0$   
**and** *proj2-rep*  $?r \cdot (M *v \text{proj2-rep } ?r) > 0$   
**by** (*unfold conic-sgn-def*) (*simp-all add: sgn-1-pos*)  
  
**from**  $\langle \text{proj2-incident } ?r$  (*polar*  $p$ )  $\rangle$

**have**  $\text{proj2-incident } p \text{ (polar } ?r)$  **by**  $(\text{rule incident-polar-swap})$   
**hence**  $\text{proj2-rep } p \cdot (M *v \text{proj2-rep } ?r) = 0$  **by**  $(\text{simp add: incident-polar})$

**have**  $p \neq ?r$

**proof**

**assume**  $p = ?r$

**with**  $\langle \text{proj2-incident } ?r \text{ (polar } p) \rangle$  **have**  $\text{proj2-incident } p \text{ (polar } p)$  **by**  $\text{simp}$

**hence**  $\text{proj2-rep } p \cdot (M *v \text{proj2-rep } p) = 0$  **by**  $(\text{simp add: incident-polar})$

**with**  $\langle \text{proj2-rep } p \cdot (M *v \text{proj2-rep } p) > 0 \rangle$  **show**  $\text{False}$  **by**  $\text{simp}$

**qed**

**let**  $?l = \text{proj2-line-through } p \ ?r$

**have**  $\text{proj2-incident } p \ ?l$  **and**  $\text{proj2-incident } ?r \ ?l$

**by**  $(\text{rule proj2-line-through-incident})+$

**have**  $\forall q. \text{proj2-incident } q \ ?l \longrightarrow q \notin S$

**proof**

**fix**  $q$

**show**  $\text{proj2-incident } q \ ?l \longrightarrow q \notin S$

**proof**

**assume**  $\text{proj2-incident } q \ ?l$

**with**  $\langle p \neq ?r \rangle$  **and**  $\langle \text{proj2-incident } p \ ?l \rangle$  **and**  $\langle \text{proj2-incident } ?r \ ?l \rangle$

**have**  $q = p \vee (\exists k. q = \text{proj2-abs } (k *_R \text{proj2-rep } p + \text{proj2-rep } ?r))$

**by**  $(\text{simp add: proj2-incident-iff [of } p \ ?r \ ?l \ q])$

**show**  $q \notin S$

**proof** *cases*

**assume**  $q = p$

**with**  $\langle \text{conic-sgn } p = 1 \rangle$  **show**  $q \notin S$  **by**  $(\text{unfold } S\text{-def}) \text{ simp}$

**next**

**assume**  $q \neq p$

**with**  $\langle q = p \vee (\exists k. q = \text{proj2-abs } (k *_R \text{proj2-rep } p + \text{proj2-rep } ?r)) \rangle$

**obtain**  $k$  **where**  $q = \text{proj2-abs } (k *_R \text{proj2-rep } p + \text{proj2-rep } ?r)$

**by**  $\text{auto}$

**from**  $\langle \text{proj2-rep } p \cdot (M *v \text{proj2-rep } p) > 0 \rangle$

**have**  $\text{proj2-rep } p \cdot (M *v \text{proj2-rep } p) * k^2 \geq 0$

**by**  $\text{simp}$

**with**  $\langle \text{proj2-rep } p \cdot (M *v \text{proj2-rep } ?r) = 0 \rangle$

**and**  $\langle \text{proj2-rep } ?r \cdot (M *v \text{proj2-rep } ?r) > 0 \rangle$

**have**  $\text{proj2-rep } p \cdot (M *v \text{proj2-rep } p) * k^2$

$+ \text{proj2-rep } p \cdot (M *v \text{proj2-rep } ?r) * 2 * k$

$+ \text{proj2-rep } ?r \cdot (M *v \text{proj2-rep } ?r)$

$> 0$

**by**  $\text{simp}$

**with**  $\langle p \neq ?r \rangle$  **and**  $\langle q = \text{proj2-abs } (k *_R \text{proj2-rep } p + \text{proj2-rep } ?r) \rangle$

**show**  $q \notin S$  **by**  $(\text{simp add: } S\text{-quadratic})$

**qed**

**qed**

**qed**

**with**  $\langle \text{proj2-incident } p \ ?l \rangle$   
**show**  $\exists l. \text{proj2-incident } p \ l \wedge (\forall q. \text{proj2-incident } q \ l \longrightarrow q \notin S)$   
**by** (*simp add: exI [of - ?l]*)  
**qed**

**lemma** *lines-through-intersect-S-twice-in-K2:*

**assumes**  $\forall l. \text{proj2-incident } p \ l$   
 $\longrightarrow (\exists q \ r. q \neq r \wedge q \in S \wedge r \in S \wedge \text{proj2-incident } q \ l \wedge \text{proj2-incident } r \ l)$   
**shows**  $p \in K2$   
**proof** (*rule ccontr*)  
**assume**  $p \notin K2$   
**hence** *conic-sgn*  $p \geq 0$  **by** (*unfold K2-def*) *simp*

**have**  $\neg (\forall l. \text{proj2-incident } p \ l \longrightarrow (\exists q \ r. q \neq r \wedge q \in S \wedge r \in S \wedge \text{proj2-incident } q \ l \wedge \text{proj2-incident } r \ l))$

**proof** *cases*

**assume** *conic-sgn*  $p = 0$   
**hence**  $p \in S$  **unfolding** *S-def* ..  
**hence** *proj2-incident*  $p$  (*polar*  $p$ ) **by** (*simp add: incident-own-polar-in-S*)  
**let**  $?l = \text{polar } p$   
**have**  $\neg (\exists q \ r. q \neq r \wedge q \in S \wedge r \in S \wedge \text{proj2-incident } q \ ?l \wedge \text{proj2-incident } r \ ?l)$

**proof**

**assume**  $\exists q \ r. q \neq r \wedge q \in S \wedge r \in S \wedge \text{proj2-incident } q \ ?l \wedge \text{proj2-incident } r \ ?l$   
**then obtain**  $q$  **and**  $r$  **where**  $q \neq r$  **and**  $q \in S$  **and**  $r \in S$   
**and** *proj2-incident*  $q \ ?l$  **and** *proj2-incident*  $r \ ?l$   
**by** *auto*  
**from**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$  **and**  $\langle \text{proj2-incident } q \ ?l \rangle$   
**and**  $\langle r \in S \rangle$  **and**  $\langle \text{proj2-incident } r \ ?l \rangle$   
**have**  $q = p$  **and**  $r = p$  **by** (*simp add: point-in-S-polar-is-tangent*) +  
**with**  $\langle q \neq r \rangle$  **show** *False* **by** *simp*

**qed**

**with**  $\langle \text{proj2-incident } p \ ?l \rangle$   
**show**  $\neg (\forall l. \text{proj2-incident } p \ l \longrightarrow (\exists q \ r. q \neq r \wedge q \in S \wedge r \in S \wedge \text{proj2-incident } q \ l \wedge \text{proj2-incident } r \ l))$   
**by** *auto*

**next**

**assume** *conic-sgn*  $p \neq 0$   
**with**  $\langle \text{conic-sgn } p \geq 0 \rangle$  **have** *conic-sgn*  $p > 0$  **by** *simp*  
**hence** *sgn* (*conic-sgn*  $p$ ) = 1 **by** *simp*  
**hence** *conic-sgn*  $p = 1$  **by** (*simp add: sgn-conic-sgn*)  
**with** *outside-exists-line-not-intersect-S*  
**obtain**  $l$  **where** *proj2-incident*  $p \ l$  **and**  $\forall q. \text{proj2-incident } q \ l \longrightarrow q \notin S$   
**by** *auto*  
**have**  $\neg (\exists q \ r. q \neq r \wedge q \in S \wedge r \in S \wedge \text{proj2-incident } q \ l \wedge \text{proj2-incident } r \ l)$   
**proof**  
**assume**  $\exists q \ r.$

$q \neq r \wedge q \in S \wedge r \in S \wedge \text{proj2-incident } q \ l \wedge \text{proj2-incident } r \ l$   
**then obtain  $q$  where  $q \in S$  and  $\text{proj2-incident } q \ l$  by auto**  
**from  $\langle \text{proj2-incident } q \ l \rangle$  and  $\langle \forall q. \text{proj2-incident } q \ l \longrightarrow q \notin S \rangle$**   
**have  $q \notin S$  by simp**  
**with  $\langle q \in S \rangle$  show *False* by simp**  
**qed**  
**with  $\langle \text{proj2-incident } p \ l \rangle$**   
**show  $\neg (\forall l. \text{proj2-incident } p \ l \longrightarrow (\exists q r. q \neq r \wedge q \in S \wedge r \in S \wedge \text{proj2-incident } q \ l \wedge \text{proj2-incident } r \ l))$**   
**by auto**  
**qed**  
**with  $\langle \forall l. \text{proj2-incident } p \ l \longrightarrow (\exists q r. q \neq r \wedge q \in S \wedge r \in S \wedge \text{proj2-incident } q \ l \wedge \text{proj2-incident } r \ l) \rangle$**   
**show *False* by simp**  
**qed**

**lemma *line-through-hyp2-pole-not-in-hyp2*:**

**assumes  $a \in \text{hyp2}$  and  $\text{proj2-incident } a \ l$**   
**shows  $\text{pole } l \notin \text{hyp2}$**

**proof –**

**from *assms* and *line-through-K2-intersect-S***  
**obtain  $p$  where  $p \in S$  and  $\text{proj2-incident } p \ l$  by auto**

**from  $\langle \text{proj2-incident } p \ l \rangle$**   
**have  $\text{proj2-incident } (\text{pole } l) \ (\text{polar } p)$  by (*rule incident-pole-polar*)**  
**with  $\langle p \in S \rangle$**   
**show  $\text{pole } l \notin \text{hyp2}$**   
**by (*auto simp add: tangent-not-through-K2*)**

**qed**

**lemma *statement60-one-way*:**

**assumes *is-K2-isometry*  $J$  and  $p \in K2$**   
**shows  $\text{apply-cltn2 } p \ J \in K2$  (is  $?p' \in K2$ )**

**proof –**

**let  $?J' = \text{cltn2-inverse } J$**

**have  $\forall l'. \text{proj2-incident } ?p' \ l' \longrightarrow (\exists q' r'. q' \neq r' \wedge q' \in S \wedge r' \in S \wedge \text{proj2-incident } q' \ l' \wedge \text{proj2-incident } r' \ l')$**

**proof**

**fix  $l'$**

**let  $?l = \text{apply-cltn2-line } l' \ ?J'$**

**show  $\text{proj2-incident } ?p' \ l' \longrightarrow (\exists q' r'. q' \neq r' \wedge q' \in S \wedge r' \in S \wedge \text{proj2-incident } q' \ l' \wedge \text{proj2-incident } r' \ l')$**

**proof**

**assume  $\text{proj2-incident } ?p' \ l'$**

**hence  $\text{proj2-incident } p \ ?l$**

**by (*simp add: apply-cltn2-incident* [of  $p \ l' \ ?J'$ ]  
 $\text{cltn2.inv-inv}$  [simplified])**

**with  $\langle p \in K2 \rangle$  and *line-through-K2-intersect-S-twice* [of  $p \ ?l$ ]**

**obtain**  $q$  **and**  $r$  **where**  $q \neq r$  **and**  $q \in S$  **and**  $r \in S$   
**and** *proj2-incident*  $q$   $?l$  **and** *proj2-incident*  $r$   $?l$   
**by** *auto*  
**let**  $?q' = \text{apply-cltn2 } q \ J$   
**let**  $?r' = \text{apply-cltn2 } r \ J$   
**from**  $\langle q \neq r \rangle$  **and** *apply-cltn2-injective* [*of*  $q \ J \ r$ ] **have**  $?q' \neq ?r'$  **by** *auto*

**from**  $\langle q \in S \rangle$  **and**  $\langle r \in S \rangle$  **and**  $\langle \text{is-K2-isometry } J \rangle$   
**have**  $?q' \in S$  **and**  $?r' \in S$  **by** (*unfold is-K2-isometry-def*) *simp-all*

**from**  $\langle \text{proj2-incident } q \ ?l \rangle$  **and**  $\langle \text{proj2-incident } r \ ?l \rangle$   
**have** *proj2-incident*  $?q' \ l'$  **and** *proj2-incident*  $?r' \ l'$   
**by** (*simp-all add: apply-cltn2-incident [of - l' ?J]*)  
*cltn2.inv-inv [simplified]*)  
**with**  $\langle ?q' \neq ?r' \rangle$  **and**  $\langle ?q' \in S \rangle$  **and**  $\langle ?r' \in S \rangle$   
**show**  $\exists q' r'$ .  
 $q' \neq r' \wedge q' \in S \wedge r' \in S \wedge \text{proj2-incident } q' \ l' \wedge \text{proj2-incident } r' \ l'$   
**by** *auto*

**qed**  
**qed**  
**thus**  $?p' \in K2$  **by** (*rule lines-through-intersect-S-twice-in-K2*)  
**qed**

**lemma** *is-K2-isometry-hyp2-S*:  
**assumes**  $p \in \text{hyp2} \cup S$  **and** *is-K2-isometry*  $J$   
**shows** *apply-cltn2*  $p \ J \in \text{hyp2} \cup S$   
**proof** *cases*  
**assume**  $p \in \text{hyp2}$   
**with**  $\langle \text{is-K2-isometry } J \rangle$   
**have** *apply-cltn2*  $p \ J \in \text{hyp2}$  **by** (*rule statement60-one-way*)  
**thus** *apply-cltn2*  $p \ J \in \text{hyp2} \cup S$  ..  
**next**  
**assume**  $p \notin \text{hyp2}$   
**with**  $\langle p \in \text{hyp2} \cup S \rangle$  **have**  $p \in S$  **by** *simp*  
**with**  $\langle \text{is-K2-isometry } J \rangle$   
**have** *apply-cltn2*  $p \ J \in S$  **by** (*unfold is-K2-isometry-def*) *simp*  
**thus** *apply-cltn2*  $p \ J \in \text{hyp2} \cup S$  ..  
**qed**

**lemma** *is-K2-isometry-z-non-zero*:  
**assumes**  $p \in \text{hyp2} \cup S$  **and** *is-K2-isometry*  $J$   
**shows** *z-non-zero* (*apply-cltn2*  $p \ J$ )  
**proof** –  
**from**  $\langle p \in \text{hyp2} \cup S \rangle$  **and**  $\langle \text{is-K2-isometry } J \rangle$   
**have** *apply-cltn2*  $p \ J \in \text{hyp2} \cup S$  **by** (*rule is-K2-isometry-hyp2-S*)  
**thus** *z-non-zero* (*apply-cltn2*  $p \ J$ ) **by** (*rule hyp2-S-z-non-zero*)  
**qed**

**lemma** *cart2-append1-apply-cltn2*:

**assumes**  $p \in \text{hyp2} \cup S$  **and** *is-K2-isometry J*  
**shows**  $\exists k. k \neq 0$   
 $\wedge \text{cart2-append1 } p \ v^* \ \text{cltn2-rep } J = k *_{\mathbb{R}} \text{cart2-append1 } (\text{apply-cltn2 } p \ J)$   
**proof** –  
**have**  $\text{cart2-append1 } p \ v^* \ \text{cltn2-rep } J$   
 $= (1 / (\text{proj2-rep } p)\$3) *_{\mathbb{R}} (\text{proj2-rep } p \ v^* \ \text{cltn2-rep } J)$   
**by** (*unfold cart2-append1-def*) (*simp add: scaleR-vector-matrix-assoc*)  
  
**from**  $\langle p \in \text{hyp2} \cup S \rangle$  **have**  $(\text{proj2-rep } p)\$3 \neq 0$  **by** (*rule hyp2-S-z-non-zero*)  
  
**from** *apply-cltn2-imp-mult* [*of p J*]  
**obtain**  $j$  **where**  $j \neq 0$   
**and**  $\text{proj2-rep } p \ v^* \ \text{cltn2-rep } J = j *_{\mathbb{R}} \text{proj2-rep } (\text{apply-cltn2 } p \ J)$   
**by** *auto*  
  
**from**  $\langle p \in \text{hyp2} \cup S \rangle$  **and**  $\langle \text{is-K2-isometry } J \rangle$   
**have** *z-non-zero* (*apply-cltn2 p J*) **by** (*rule is-K2-isometry-z-non-zero*)  
**hence**  $\text{proj2-rep } (\text{apply-cltn2 } p \ J)$   
 $= (\text{proj2-rep } (\text{apply-cltn2 } p \ J))\$3 *_{\mathbb{R}} \text{cart2-append1 } (\text{apply-cltn2 } p \ J)$   
**by** (*rule proj2-rep-cart2-append1*)  
  
**let**  $?k = 1 / (\text{proj2-rep } p)\$3 * j * (\text{proj2-rep } (\text{apply-cltn2 } p \ J))\$3$   
**from**  $\langle (\text{proj2-rep } p)\$3 \neq 0 \rangle$  **and**  $\langle j \neq 0 \rangle$   
**and**  $\langle (\text{proj2-rep } (\text{apply-cltn2 } p \ J))\$3 \neq 0 \rangle$   
**have**  $?k \neq 0$  **by** *simp*  
  
**from**  $\langle \text{cart2-append1 } p \ v^* \ \text{cltn2-rep } J$   
 $= (1 / (\text{proj2-rep } p)\$3) *_{\mathbb{R}} (\text{proj2-rep } p \ v^* \ \text{cltn2-rep } J) \rangle$   
**and**  $\langle \text{proj2-rep } p \ v^* \ \text{cltn2-rep } J = j *_{\mathbb{R}} \text{proj2-rep } (\text{apply-cltn2 } p \ J) \rangle$   
**have**  $\text{cart2-append1 } p \ v^* \ \text{cltn2-rep } J$   
 $= (1 / (\text{proj2-rep } p)\$3 * j) *_{\mathbb{R}} \text{proj2-rep } (\text{apply-cltn2 } p \ J)$   
**by** *simp*  
  
**from**  $\langle \text{proj2-rep } (\text{apply-cltn2 } p \ J)$   
 $= (\text{proj2-rep } (\text{apply-cltn2 } p \ J))\$3 *_{\mathbb{R}} \text{cart2-append1 } (\text{apply-cltn2 } p \ J) \rangle$   
**have**  $(1 / (\text{proj2-rep } p)\$3 * j) *_{\mathbb{R}} \text{proj2-rep } (\text{apply-cltn2 } p \ J)$   
 $= (1 / (\text{proj2-rep } p)\$3 * j) *_{\mathbb{R}} ((\text{proj2-rep } (\text{apply-cltn2 } p \ J))\$3$   
 $*_{\mathbb{R}} \text{cart2-append1 } (\text{apply-cltn2 } p \ J))$   
**by** *simp*  
**with**  $\langle \text{cart2-append1 } p \ v^* \ \text{cltn2-rep } J$   
 $= (1 / (\text{proj2-rep } p)\$3 * j) *_{\mathbb{R}} \text{proj2-rep } (\text{apply-cltn2 } p \ J) \rangle$   
**have**  $\text{cart2-append1 } p \ v^* \ \text{cltn2-rep } J = ?k *_{\mathbb{R}} \text{cart2-append1 } (\text{apply-cltn2 } p \ J)$   
**by** *simp*  
**with**  $\langle ?k \neq 0 \rangle$   
**show**  $\exists k. k \neq 0$   
 $\wedge \text{cart2-append1 } p \ v^* \ \text{cltn2-rep } J = k *_{\mathbb{R}} \text{cart2-append1 } (\text{apply-cltn2 } p \ J)$   
**by** (*simp add: exI* [*of - ?k*])  
**qed**

## 8.5 The $K$ -isometries form a group action

**lemma** *hyp2-cltn2-id* [simp]:  $\text{hyp2-cltn2 } p \text{ cltn2-id} = p$   
 by (unfold *hyp2-cltn2-def*) (simp add: *Rep-hyp2-inverse*)

**lemma** *apply-cltn2-Rep-hyp2*:  
 assumes *is-K2-isometry J*  
 shows *apply-cltn2 (Rep-hyp2 p) J*  $\in$  *hyp2*

**proof** –  
 from  $\langle \text{is-K2-isometry } J \rangle$  and *Rep-hyp2* [of *p*]  
 show *apply-cltn2 (Rep-hyp2 p) J*  $\in$  *K2* by (rule *statement60-one-way*)  
 qed

**lemma** *Rep-hyp2-cltn2*:  
 assumes *is-K2-isometry J*  
 shows *Rep-hyp2 (hyp2-cltn2 p J) = apply-cltn2 (Rep-hyp2 p) J*

**proof** –  
 from  $\langle \text{is-K2-isometry } J \rangle$   
 have *apply-cltn2 (Rep-hyp2 p) J*  $\in$  *hyp2* by (rule *apply-cltn2-Rep-hyp2*)  
 thus *Rep-hyp2 (hyp2-cltn2 p J) = apply-cltn2 (Rep-hyp2 p) J*  
 by (unfold *hyp2-cltn2-def*) (rule *Abs-hyp2-inverse*)  
 qed

**lemma** *hyp2-cltn2-compose*:  
 assumes *is-K2-isometry H*  
 shows *hyp2-cltn2 (hyp2-cltn2 p H) J = hyp2-cltn2 p (cltn2-compose H J)*

**proof** –  
 from  $\langle \text{is-K2-isometry } H \rangle$   
 have *apply-cltn2 (Rep-hyp2 p) H*  $\in$  *hyp2* by (rule *apply-cltn2-Rep-hyp2*)  
 thus *hyp2-cltn2 (hyp2-cltn2 p H) J = hyp2-cltn2 p (cltn2-compose H J)*  
 by (unfold *hyp2-cltn2-def*) (simp add: *Abs-hyp2-inverse apply-cltn2-compose*)  
 qed

**interpretation** *K2-isometry: action*

(|*carrier = Collect is-K2-isometry, mult = cltn2-compose, one = cltn2-id*|)  
*hyp2-cltn2*

**proof**  
 let  $?G =$   
 (|*carrier = Collect is-K2-isometry, mult = cltn2-compose, one = cltn2-id*|)  
 fix *p*  
 show *hyp2-cltn2 p 1 ?G = p*  
 by (unfold *hyp2-cltn2-def*) (simp add: *Rep-hyp2-inverse*)  
 fix *H J*  
 show *H*  $\in$  *carrier ?G*  $\wedge$  *J*  $\in$  *carrier ?G*  
 $\rightarrow$  *hyp2-cltn2 (hyp2-cltn2 p H) J = hyp2-cltn2 p (H  $\otimes_{?G}$  J)*  
 by (simp add: *hyp2-cltn2-compose*)  
 qed

## 8.6 The Klein–Beltrami model satisfies Tarski’s first three axioms

**lemma** *three-in-S-tangent-intersection-no-3-Col:*

**assumes**  $p \in S$  **and**  $q \in S$  **and**  $r \in S$

**and**  $p \neq q$  **and**  $r \notin \{p, q\}$

**shows** *proj2-no-3-Col*  $\{proj2\text{-intersection } (polar\ p) (polar\ q), r, p, q\}$

(**is** *proj2-no-3-Col*  $\{?s, r, p, q\}$ )

**proof** –

**let**  $?T = \{?s, r, p, q\}$

**from**  $\langle p \neq q \rangle$  **have**  $card\ \{p, q\} = 2$  **by** *simp*

**with**  $\langle r \notin \{p, q\} \rangle$  **have**  $card\ \{r, p, q\} = 3$  **by** *simp*

**from**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$  **and**  $\langle r \in S \rangle$  **have**  $\{r, p, q\} \subseteq S$  **by** *simp*

**have** *proj2-incident*  $?s$  (polar  $p$ ) **and** *proj2-incident*  $?s$  (polar  $q$ )

**by** (*rule proj2-intersection-incident*) $+$

**have**  $?s \notin S$

**proof**

**assume**  $?s \in S$

**with**  $\langle p \in S \rangle$  **and**  $\langle proj2\text{-incident } ?s (polar\ p) \rangle$

**and**  $\langle q \in S \rangle$  **and**  $\langle proj2\text{-incident } ?s (polar\ q) \rangle$

**have**  $?s = p$  **and**  $?s = q$  **by** (*simp-all add: point-in-S-polar-is-tangent*)

**hence**  $p = q$  **by** *simp*

**with**  $\langle p \neq q \rangle$  **show** *False* ..

**qed**

**with**  $\langle \{r, p, q\} \subseteq S \rangle$  **have**  $?s \notin \{r, p, q\}$  **by** *auto*

**with**  $\langle card\ \{r, p, q\} = 3 \rangle$  **have**  $card\ \{?s, r, p, q\} = 4$  **by** *simp*

**have**  $\forall t \in ?T. \neg proj2\text{-set-Col } (?T - \{t\})$

**proof** *standard* $+$

**fix**  $t$

**assume**  $t \in ?T$

**assume** *proj2-set-Col*  $(?T - \{t\})$

**then obtain**  $l$  **where**  $\forall a \in (?T - \{t\}). proj2\text{-incident } a\ l$

**unfolding** *proj2-set-Col-def* ..

**from**  $\langle proj2\text{-set-Col } (?T - \{t\}) \rangle$

**have** *proj2-set-Col*  $(S \cap (?T - \{t\}))$

**by** (*simp add: proj2-subset-Col [of  $(S \cap (?T - \{t\}))$   $?T - \{t\}$ ]*)

**hence**  $card\ (S \cap (?T - \{t\})) \leq 2$  **by** (*simp add: card-line-intersect-S*)

**show** *False*

**proof** *cases*

**assume**  $t = ?s$

**with**  $\langle ?s \notin \{r, p, q\} \rangle$  **have**  $?T - \{t\} = \{r, p, q\}$  **by** *simp*

**with**  $\langle \{r, p, q\} \subseteq S \rangle$  **have**  $S \cap (?T - \{t\}) = \{r, p, q\}$  **by** *simp*

**with**  $\langle card\ \{r, p, q\} = 3 \rangle$  **and**  $\langle card\ (S \cap (?T - \{t\})) \leq 2 \rangle$  **show** *False* **by**

*simp*  
**next**  
**assume**  $t \neq ?s$   
**hence**  $?s \in ?T - \{t\}$  **by** *simp*  
**with**  $\langle \forall a \in (?T - \{t\}). \text{proj2-incident } a \ l \rangle$  **have** *proj2-incident ?s l ..*  
  
**from**  $\langle p \neq q \rangle$  **have**  $\{p, q\} \cap ?T - \{t\} \neq \{\}$  **by** *auto*  
**then obtain**  $d$  **where**  $d \in \{p, q\}$  **and**  $d \in ?T - \{t\}$  **by** *auto*  
**from**  $\langle d \in ?T - \{t\} \rangle$  **and**  $\langle \forall a \in (?T - \{t\}). \text{proj2-incident } a \ l \rangle$   
**have** *proj2-incident d l by simp*  
  
**from**  $\langle d \in \{p, q\} \rangle$   
**and**  $\langle \text{proj2-incident } ?s \ (\text{polar } p) \rangle$   
**and**  $\langle \text{proj2-incident } ?s \ (\text{polar } q) \rangle$   
**have** *proj2-incident ?s (polar d) by auto*  
  
**from**  $\langle d \in \{p, q\} \rangle$  **and**  $\langle \{r, p, q\} \subseteq S \rangle$  **have**  $d \in S$  **by** *auto*  
**hence** *proj2-incident d (polar d) by (unfold incident-own-polar-in-S)*  
  
**from**  $\langle d \in S \rangle$  **and**  $\langle ?s \notin S \rangle$  **have**  $d \neq ?s$  **by** *auto*  
**with**  $\langle \text{proj2-incident } ?s \ l \rangle$   
**and**  $\langle \text{proj2-incident } d \ l \rangle$   
**and**  $\langle \text{proj2-incident } ?s \ (\text{polar } d) \rangle$   
**and**  $\langle \text{proj2-incident } d \ (\text{polar } d) \rangle$   
**and** *proj2-incident-unique*  
**have**  $l = \text{polar } d$  **by** *auto*  
**with**  $\langle d \in S \rangle$  **and** *point-in-S-polar-is-tangent*  
**have**  $\forall a \in S. \text{proj2-incident } a \ l \longrightarrow a = d$  **by** *simp*  
**with**  $\langle \forall a \in (?T - \{t\}). \text{proj2-incident } a \ l \rangle$   
**have**  $S \cap (?T - \{t\}) \subseteq \{d\}$  **by** *auto*  
**with** *card-mono [of {d}]* **have**  $\text{card } (S \cap (?T - \{t\})) \leq 1$  **by** *simp*  
**hence**  $\text{card } ((S \cap ?T) - \{t\}) \leq 1$  **by** (*simp add: Int-Diff*)  
  
**have**  $S \cap ?T \subseteq \text{insert } t \ ((S \cap ?T) - \{t\})$  **by** *auto*  
**with** *card-suc-ge-insert [of t (S ∩ ?T) - {t}]*  
**and** *card-mono [of insert t ((S ∩ ?T) - {t}) S ∩ ?T]*  
**have**  $\text{card } (S \cap ?T) \leq \text{card } ((S \cap ?T) - \{t\}) + 1$  **by** *simp*  
**with**  $\langle \text{card } ((S \cap ?T) - \{t\}) \leq 1 \rangle$  **have**  $\text{card } (S \cap ?T) \leq 2$  **by** *simp*  
  
**from**  $\langle \{r, p, q\} \subseteq S \rangle$  **have**  $\{r, p, q\} \subseteq S \cap ?T$  **by** *simp*  
**with**  $\langle \text{card } \{r, p, q\} = 3 \rangle$  **and** *card-mono [of S ∩ ?T {r, p, q}]*  
**have**  $\text{card } (S \cap ?T) \geq 3$  **by** *simp*  
**with**  $\langle \text{card } (S \cap ?T) \leq 2 \rangle$  **show** *False by simp*  
**qed**  
**qed**  
**with**  $\langle \text{card } ?T = 4 \rangle$  **show** *proj2-no-3-Col ?T unfolding proj2-no-3-Col-def ..*  
**qed**

**lemma** *statement65-special-case:*

**assumes**  $p \in S$  **and**  $q \in S$  **and**  $r \in S$  **and**  $p \neq q$  **and**  $r \notin \{p, q\}$   
**shows**  $\exists J. \text{is-K2-isometry } J$   
 $\wedge \text{apply-cltn2 east } J = p$   
 $\wedge \text{apply-cltn2 west } J = q$   
 $\wedge \text{apply-cltn2 north } J = r$   
 $\wedge \text{apply-cltn2 far-north } J = \text{proj2-intersection (polar } p) (\text{polar } q)$   
**proof** –  
**let**  $?s = \text{proj2-intersection (polar } p) (\text{polar } q)$   
**let**  $?t = \text{vector [vector [?s,r,p,q], vector [far-north, north, east, west]]}$   
 $:: \text{proj2}^{\wedge 4} \text{ }^{\wedge 2}$   
**have**  $\text{range } ((\$) (?t\$1)) = \{?s, r, p, q\}$   
**unfolding** *image-def*  
**by** (*auto simp add: UNIV-4 vector-4*)  
**with**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$  **and**  $\langle r \in S \rangle$  **and**  $\langle p \neq q \rangle$  **and**  $\langle r \notin \{p, q\} \rangle$   
**have**  $\text{proj2-no-3-Col } (\text{range } ((\$) (?t\$1)))$   
**by** (*simp add: three-in-S-tangent-intersection-no-3-Col*)  
**moreover** **have**  $\text{range } ((\$) (?t\$2)) = \{\text{far-north, north, east, west}\}$   
**unfolding** *image-def*  
**by** (*auto simp add: UNIV-4 vector-4*)  
**with** *compass-in-S and east-west-distinct and north-not-east-or-west*  
**and** *east-west-tangents-far-north*  
**and** *three-in-S-tangent-intersection-no-3-Col [of east west north]*  
**have**  $\text{proj2-no-3-Col } (\text{range } ((\$) (?t\$2)))$  **by** *simp*  
**ultimately** **have**  $\forall i. \text{proj2-no-3-Col } (\text{range } ((\$) (?t\$i)))$   
**by** (*simp add: forall-2*)  
**hence**  $\exists J. \forall j. \text{apply-cltn2 } (?t\$0\$j) J = ?t\$1\$j$   
**by** (*rule statement53-existence*)  
**moreover** **have**  $0 = (2::2)$  **by** *simp*  
**ultimately** **obtain**  $J$  **where**  $\forall j. \text{apply-cltn2 } (?t\$2\$j) J = ?t\$1\$j$  **by** *auto*  
**hence**  $\text{apply-cltn2 } (?t\$2\$1) J = ?t\$1\$1$   
**and**  $\text{apply-cltn2 } (?t\$2\$2) J = ?t\$1\$2$   
**and**  $\text{apply-cltn2 } (?t\$2\$3) J = ?t\$1\$3$   
**and**  $\text{apply-cltn2 } (?t\$2\$4) J = ?t\$1\$4$   
**by** *simp-all*  
**hence**  $\text{apply-cltn2 east } J = p$   
**and**  $\text{apply-cltn2 west } J = q$   
**and**  $\text{apply-cltn2 north } J = r$   
**and**  $\text{apply-cltn2 far-north } J = ?s$   
**by** (*simp-all add: vector-2 vector-4*)  
**with** *compass-non-zero*  
**have**  $p = \text{proj2-abs (vector [1,0,1] v* cltn2-rep J)}$   
**and**  $q = \text{proj2-abs (vector [-1,0,1] v* cltn2-rep J)}$   
**and**  $r = \text{proj2-abs (vector [0,1,1] v* cltn2-rep J)}$   
**and**  $?s = \text{proj2-abs (vector [0,1,0] v* cltn2-rep J)}$   
**unfolding** *compass-defs and far-north-def*  
**by** (*simp-all add: apply-cltn2-left-abs*)  
  
**let**  $?N = \text{cltn2-rep } J ** M ** \text{transpose (cltn2-rep } J)$   
**from** *M-symmatrix* **have** *symmatrix ?N* **by** (*rule symmatrix-preserve*)

hence  $?N\$2\$1 = ?N\$1\$2$  and  $?N\$3\$1 = ?N\$1\$3$  and  $?N\$3\$2 = ?N\$2\$3$   
 unfolding *symmatrix-def* and *transpose-def*  
 by (*simp-all add: vec-eq-iff*)

from *compass-non-zero* and  $\langle \text{apply-cltn2 east } J = p \rangle$  and  $\langle p \in S \rangle$   
 and *apply-cltn2-abs-in-S* [of vector  $[1,0,1]$   $J$ ]  
 have  $(\text{vector } [1,0,1] :: \text{real}^3) \cdot (?N *v \text{vector } [1,0,1]) = 0$   
 unfolding *east-def*  
 by *simp*  
 hence  $?N\$1\$1 + ?N\$1\$3 + ?N\$3\$1 + ?N\$3\$3 = 0$   
 unfolding *inner-vec-def* and *matrix-vector-mult-def*  
 by (*simp add: sum-3 vector-3*)  
 with  $\langle ?N\$3\$1 = ?N\$1\$3 \rangle$  have  $?N\$1\$1 + 2 * (?N\$1\$3) + ?N\$3\$3 = 0$  by  
*simp*

from *compass-non-zero* and  $\langle \text{apply-cltn2 west } J = q \rangle$  and  $\langle q \in S \rangle$   
 and *apply-cltn2-abs-in-S* [of vector  $[-1,0,1]$   $J$ ]  
 have  $(\text{vector } [-1,0,1] :: \text{real}^3) \cdot (?N *v \text{vector } [-1,0,1]) = 0$   
 unfolding *west-def*  
 by *simp*  
 hence  $?N\$1\$1 - ?N\$1\$3 - ?N\$3\$1 + ?N\$3\$3 = 0$   
 unfolding *inner-vec-def* and *matrix-vector-mult-def*  
 by (*simp add: sum-3 vector-3*)  
 with  $\langle ?N\$3\$1 = ?N\$1\$3 \rangle$  have  $?N\$1\$1 - 2 * (?N\$1\$3) + ?N\$3\$3 = 0$  by  
*simp*

with  $\langle ?N\$1\$1 + 2 * (?N\$1\$3) + ?N\$3\$3 = 0 \rangle$   
 have  $?N\$1\$1 + 2 * (?N\$1\$3) + ?N\$3\$3 = ?N\$1\$1 - 2 * (?N\$1\$3) +$   
 $?N\$3\$3$   
 by *simp*  
 hence  $?N\$1\$3 = 0$  by *simp*  
 with  $\langle ?N\$1\$1 + 2 * (?N\$1\$3) + ?N\$3\$3 = 0 \rangle$  have  $?N\$3\$3 = - (?N\$1\$1)$   
 by *simp*

from *compass-non-zero* and  $\langle \text{apply-cltn2 north } J = r \rangle$  and  $\langle r \in S \rangle$   
 and *apply-cltn2-abs-in-S* [of vector  $[0,1,1]$   $J$ ]  
 have  $(\text{vector } [0,1,1] :: \text{real}^3) \cdot (?N *v \text{vector } [0,1,1]) = 0$   
 unfolding *north-def*  
 by *simp*  
 hence  $?N\$2\$2 + ?N\$2\$3 + ?N\$3\$2 + ?N\$3\$3 = 0$   
 unfolding *inner-vec-def* and *matrix-vector-mult-def*  
 by (*simp add: sum-3 vector-3*)  
 with  $\langle ?N\$3\$2 = ?N\$2\$3 \rangle$  have  $?N\$2\$2 + 2 * (?N\$2\$3) + ?N\$3\$3 = 0$  by  
*simp*

have *proj2-incident ?s (polar p)* and *proj2-incident ?s (polar q)*  
 by (*rule proj2-intersection-incident*)+

from *compass-non-zero*  
 have *vector*  $[1,0,1]$  *v\* cltn2-rep*  $J \neq 0$

**and**  $\langle \text{vector } [-1,0,1] \ v * \text{cltn2-rep } J \neq 0 \rangle$   
**and**  $\langle \text{vector } [0,1,0] \ v * \text{cltn2-rep } J \neq 0 \rangle$   
**by** (*simp-all add: non-zero-mult-rep-non-zero*)  
**from**  $\langle \text{vector } [1,0,1] \ v * \text{cltn2-rep } J \neq 0 \rangle$   
**and**  $\langle \text{vector } [-1,0,1] \ v * \text{cltn2-rep } J \neq 0 \rangle$   
**and**  $\langle p = \text{proj2-abs } (\text{vector } [1,0,1] \ v * \text{cltn2-rep } J) \rangle$   
**and**  $\langle q = \text{proj2-abs } (\text{vector } [-1,0,1] \ v * \text{cltn2-rep } J) \rangle$   
**have**  $\text{polar } p = \text{proj2-line-abs } (M * v (\text{vector } [1,0,1] \ v * \text{cltn2-rep } J))$   
**and**  $\text{polar } q = \text{proj2-line-abs } (M * v (\text{vector } [-1,0,1] \ v * \text{cltn2-rep } J))$   
**by** (*simp-all add: polar-abs*)

**from**  $\langle \text{vector } [1,0,1] \ v * \text{cltn2-rep } J \neq 0 \rangle$   
**and**  $\langle \text{vector } [-1,0,1] \ v * \text{cltn2-rep } J \neq 0 \rangle$   
**and**  $M$ -invertible  
**have**  $M * v (\text{vector } [1,0,1] \ v * \text{cltn2-rep } J) \neq 0$   
**and**  $M * v (\text{vector } [-1,0,1] \ v * \text{cltn2-rep } J) \neq 0$   
**by** (*simp-all add: invertible-times-non-zero*)  
**with**  $\langle \text{vector } [0,1,0] \ v * \text{cltn2-rep } J \neq 0 \rangle$   
**and**  $\langle \text{polar } p = \text{proj2-line-abs } (M * v (\text{vector } [1,0,1] \ v * \text{cltn2-rep } J)) \rangle$   
**and**  $\langle \text{polar } q = \text{proj2-line-abs } (M * v (\text{vector } [-1,0,1] \ v * \text{cltn2-rep } J)) \rangle$   
**and**  $\langle ?s = \text{proj2-abs } (\text{vector } [0,1,0] \ v * \text{cltn2-rep } J) \rangle$   
**have**  $\text{proj2-incident } ?s (\text{polar } p)$   
 $\longleftrightarrow (\text{vector } [0,1,0] \ v * \text{cltn2-rep } J)$   
 $\cdot (M * v (\text{vector } [1,0,1] \ v * \text{cltn2-rep } J)) = 0$   
**and**  $\text{proj2-incident } ?s (\text{polar } q)$   
 $\longleftrightarrow (\text{vector } [0,1,0] \ v * \text{cltn2-rep } J)$   
 $\cdot (M * v (\text{vector } [-1,0,1] \ v * \text{cltn2-rep } J)) = 0$   
**by** (*simp-all add: proj2-incident-abs*)  
**with**  $\langle \text{proj2-incident } ?s (\text{polar } p) \rangle$  **and**  $\langle \text{proj2-incident } ?s (\text{polar } q) \rangle$   
**have**  $(\text{vector } [0,1,0] \ v * \text{cltn2-rep } J)$   
 $\cdot (M * v (\text{vector } [1,0,1] \ v * \text{cltn2-rep } J)) = 0$   
**and**  $(\text{vector } [0,1,0] \ v * \text{cltn2-rep } J)$   
 $\cdot (M * v (\text{vector } [-1,0,1] \ v * \text{cltn2-rep } J)) = 0$   
**by** *simp-all*  
**hence**  $\text{vector } [0,1,0] \cdot (?N * v \text{vector } [1,0,1]) = 0$   
**and**  $\text{vector } [0,1,0] \cdot (?N * v \text{vector } [-1,0,1]) = 0$   
**by** (*simp-all add: dot-lmul-matrix matrix-vector-mul-assoc [symmetric]*)  
**hence**  $?N\$2\$1 + ?N\$2\$3 = 0$  **and**  $-(?N\$2\$1) + ?N\$2\$3 = 0$   
**unfolding** *inner-vec-def* **and** *matrix-vector-mult-def*  
**by** (*simp-all add: sum-3 vector-3*)  
**hence**  $?N\$2\$1 + ?N\$2\$3 = -(?N\$2\$1) + ?N\$2\$3$  **by** *simp*  
**hence**  $?N\$2\$1 = 0$  **by** *simp*  
**with**  $\langle ?N\$2\$1 + ?N\$2\$3 = 0 \rangle$  **have**  $?N\$2\$3 = 0$  **by** *simp*  
**with**  $\langle ?N\$2\$2 + 2 * (?N\$2\$3) + ?N\$3\$3 = 0 \rangle$  **and**  $\langle ?N\$3\$3 = -(?N\$1\$1) \rangle$   
**have**  $?N\$2\$2 = ?N\$1\$1$  **by** *simp*  
**with**  $\langle ?N\$1\$3 = 0 \rangle$  **and**  $\langle ?N\$2\$1 = ?N\$1\$2 \rangle$  **and**  $\langle ?N\$1\$3 = 0 \rangle$   
**and**  $\langle ?N\$2\$1 = 0 \rangle$  **and**  $\langle ?N\$2\$2 = ?N\$1\$1 \rangle$  **and**  $\langle ?N\$2\$3 = 0 \rangle$   
**and**  $\langle ?N\$3\$1 = ?N\$1\$3 \rangle$  **and**  $\langle ?N\$3\$2 = ?N\$2\$3 \rangle$  **and**  $\langle ?N\$3\$3 =$   
 $-(?N\$1\$1) \rangle$

**have**  $?N = (?N\$1\$1) *_R M$   
**unfolding**  $M\text{-def}$   
**by** (*simp add: vec-eq-iff vector-3 forall-3*)

**have** *invertible* ( $\text{cltn2-rep } J$ ) **by** (*rule cltn2-rep-invertible*)  
**with**  $M\text{-invertible}$   
**have** *invertible*  $?N$  **by** (*simp add: invertible-mult transpose-invertible*)  
**hence**  $?N \neq 0$  **by** (*auto simp add: zero-not-invertible*)  
**with**  $\langle ?N = (?N\$1\$1) *_R M \rangle$  **have**  $?N\$1\$1 \neq 0$  **by** *auto*  
**with**  $\langle ?N = (?N\$1\$1) *_R M \rangle$   
**have** *is-K2-isometry* ( $\text{cltn2-abs } (\text{cltn2-rep } J)$ )  
**by** (*simp add: J-M-J-transpose-K2-isometry*)  
**hence** *is-K2-isometry*  $J$  **by** (*simp add: cltn2-abs-rep*)  
**with**  $\langle \text{apply-cltn2 east } J = p \rangle$   
**and**  $\langle \text{apply-cltn2 west } J = q \rangle$   
**and**  $\langle \text{apply-cltn2 north } J = r \rangle$   
**and**  $\langle \text{apply-cltn2 far-north } J = ?s \rangle$   
**show**  $\exists J. \text{is-K2-isometry } J$   
 $\wedge \text{apply-cltn2 east } J = p$   
 $\wedge \text{apply-cltn2 west } J = q$   
 $\wedge \text{apply-cltn2 north } J = r$   
 $\wedge \text{apply-cltn2 far-north } J = ?s$   
**by** *auto*

qed

**lemma** *statement66-existence:*

**assumes**  $a1 \in K2$  **and**  $a2 \in K2$  **and**  $p1 \in S$  **and**  $p2 \in S$

**shows**  $\exists J. \text{is-K2-isometry } J \wedge \text{apply-cltn2 } a1 J = a2 \wedge \text{apply-cltn2 } p1 J = p2$

**proof** –

**let**  $?a = \text{vector } [a1, a2] :: \text{proj2}^{\wedge}2$

**from**  $\langle a1 \in K2 \rangle$  **and**  $\langle a2 \in K2 \rangle$  **have**  $\forall i. ?a\$i \in K2$  **by** (*simp add: forall-2*)

**let**  $?p = \text{vector } [p1, p2] :: \text{proj2}^{\wedge}2$

**from**  $\langle p1 \in S \rangle$  **and**  $\langle p2 \in S \rangle$  **have**  $\forall i. ?p\$i \in S$  **by** (*simp add: forall-2*)

**let**  $?l = \chi i. \text{proj2-line-through } (?a\$i) (?p\$i)$

**have**  $\forall i. \text{proj2-incident } (?a\$i) (?l\$i)$

**by** (*simp add: proj2-line-through-incident*)

**hence**  $\text{proj2-incident } (?a\$1) (?l\$1)$  **and**  $\text{proj2-incident } (?a\$2) (?l\$2)$

**by** *fast+*

**have**  $\forall i. \text{proj2-incident } (?p\$i) (?l\$i)$

**by** (*simp add: proj2-line-through-incident*)

**hence**  $\text{proj2-incident } (?p\$1) (?l\$1)$  **and**  $\text{proj2-incident } (?p\$2) (?l\$2)$

**by** *fast+*

**let**  $?q = \chi i. \in qi. qi \neq ?p\$i \wedge qi \in S \wedge \text{proj2-incident } qi (?l\$i)$

**have**  $\forall i. ?q\$i \neq ?p\$i \wedge ?q\$i \in S \wedge \text{proj2-incident } (?q\$i) (?l\$i)$

**proof**

```

fix i
from ⟨∀ i. ?a$i ∈ K2⟩ have ?a$i ∈ K2 ..

from ⟨∀ i. proj2-incident (?a$i) (?l$i)⟩
have proj2-incident (?a$i) (?l$i) ..
with ⟨?a$i ∈ K2⟩
have ∃ qi. qi ≠ ?p$i ∧ qi ∈ S ∧ proj2-incident qi (?l$i)
  by (rule line-through-K2-intersect-S-again)
with someI-ex [of λ qi. qi ≠ ?p$i ∧ qi ∈ S ∧ proj2-incident qi (?l$i)]
show ?q$i ≠ ?p$i ∧ ?q$i ∈ S ∧ proj2-incident (?q$i) (?l$i) by simp
qed
hence ?q$1 ≠ ?p$1 and proj2-incident (?q$1) (?l$1)
and proj2-incident (?q$2) (?l$2)
by fast+

let ?r = χ i. proj2-intersection (polar (?q$i)) (polar (?p$i))
let ?m = χ i. proj2-line-through (?a$i) (?r$i)
have ∀ i. proj2-incident (?a$i) (?m$i)
  by (simp add: proj2-line-through-incident)
hence proj2-incident (?a$1) (?m$1) and proj2-incident (?a$2) (?m$2)
  by fast+

have ∀ i. proj2-incident (?r$i) (?m$i)
  by (simp add: proj2-line-through-incident)
hence proj2-incident (?r$1) (?m$1) and proj2-incident (?r$2) (?m$2)
  by fast+

let ?s = χ i. ε si. si ≠ ?r$i ∧ si ∈ S ∧ proj2-incident si (?m$i)
have ∀ i. ?s$i ≠ ?r$i ∧ ?s$i ∈ S ∧ proj2-incident (?s$i) (?m$i)
proof
fix i
from ⟨∀ i. ?a$i ∈ K2⟩ have ?a$i ∈ K2 ..

from ⟨∀ i. proj2-incident (?a$i) (?m$i)⟩
have proj2-incident (?a$i) (?m$i) ..
with ⟨?a$i ∈ K2⟩
have ∃ si. si ≠ ?r$i ∧ si ∈ S ∧ proj2-incident si (?m$i)
  by (rule line-through-K2-intersect-S-again)
with someI-ex [of λ si. si ≠ ?r$i ∧ si ∈ S ∧ proj2-incident si (?m$i)]
show ?s$i ≠ ?r$i ∧ ?s$i ∈ S ∧ proj2-incident (?s$i) (?m$i) by simp
qed
hence ?s$1 ≠ ?r$1 and proj2-incident (?s$1) (?m$1)
and proj2-incident (?s$2) (?m$2)
by fast+

have ∀ i . ∀ u. proj2-incident u (?m$i) ⟶ ¬ (u = ?p$i ∨ u = ?q$i)
proof standard+
fix i :: 2
fix u :: proj2

```

**assume**  $\text{proj2-incident } u \text{ } (?m\$i)$   
**assume**  $u = ?p\$i \vee u = ?q\$i$

**from**  $\langle \forall i. ?p\$i \in S \rangle$  **have**  $?p\$i \in S \dots$

**from**  $\langle \forall i. ?q\$i \neq ?p\$i \wedge ?q\$i \in S \wedge \text{proj2-incident } (?q\$i) (?l\$i) \rangle$   
**have**  $?q\$i \neq ?p\$i$  **and**  $?q\$i \in S$   
**by** *simp-all*

**from**  $\langle ?p\$i \in S \rangle$  **and**  $\langle ?q\$i \in S \rangle$  **and**  $\langle u = ?p\$i \vee u = ?q\$i \rangle$   
**have**  $u \in S$  **by** *auto*  
**hence**  $\text{proj2-incident } u \text{ } (\text{polar } u)$   
**by** (*simp add: incident-own-polar-in-S*)

**have**  $\text{proj2-incident } (?r\$i) \text{ } (\text{polar } (?p\$i))$   
**and**  $\text{proj2-incident } (?r\$i) \text{ } (\text{polar } (?q\$i))$   
**by** (*simp-all add: proj2-intersection-incident*)  
**with**  $\langle u = ?p\$i \vee u = ?q\$i \rangle$   
**have**  $\text{proj2-incident } (?r\$i) \text{ } (\text{polar } u)$  **by** *auto*

**from**  $\langle \forall i. \text{proj2-incident } (?r\$i) (?m\$i) \rangle$   
**have**  $\text{proj2-incident } (?r\$i) (?m\$i) \dots$

**from**  $\langle \forall i. \text{proj2-incident } (?a\$i) (?m\$i) \rangle$   
**have**  $\text{proj2-incident } (?a\$i) (?m\$i) \dots$

**from**  $\langle \forall i. ?a\$i \in K2 \rangle$  **have**  $?a\$i \in K2 \dots$

**have**  $u \neq ?r\$i$   
**proof**  
**assume**  $u = ?r\$i$   
**with**  $\langle \text{proj2-incident } (?r\$i) \text{ } (\text{polar } (?p\$i)) \rangle$   
**and**  $\langle \text{proj2-incident } (?r\$i) \text{ } (\text{polar } (?q\$i)) \rangle$   
**have**  $\text{proj2-incident } u \text{ } (\text{polar } (?p\$i))$   
**and**  $\text{proj2-incident } u \text{ } (\text{polar } (?q\$i))$   
**by** *simp-all*  
**with**  $\langle u \in S \rangle$  **and**  $\langle ?p\$i \in S \rangle$  **and**  $\langle ?q\$i \in S \rangle$   
**have**  $u = ?p\$i$  **and**  $u = ?q\$i$   
**by** (*simp-all add: point-in-S-polar-is-tangent*)  
**with**  $\langle ?q\$i \neq ?p\$i \rangle$  **show** *False* **by** *simp*

**qed**  
**with**  $\langle \text{proj2-incident } (u) \text{ } (\text{polar } u) \rangle$   
**and**  $\langle \text{proj2-incident } (?r\$i) \text{ } (\text{polar } u) \rangle$   
**and**  $\langle \text{proj2-incident } u \text{ } (?m\$i) \rangle$   
**and**  $\langle \text{proj2-incident } (?r\$i) (?m\$i) \rangle$   
**and** *proj2-incident-unique*  
**have**  $?m\$i = \text{polar } u$  **by** *auto*  
**with**  $\langle \text{proj2-incident } (?a\$i) (?m\$i) \rangle$   
**have**  $\text{proj2-incident } (?a\$i) \text{ } (\text{polar } u)$  **by** *simp*

**with**  $\langle u \in S \rangle$  **and**  $\langle ?a\$i \in K2 \rangle$  **and** *tangent-not-through-K2*  
**show** *False* **by** *simp*  
**qed**

**let**  $?H = \chi \ i. \ \epsilon \ Hi. \ is\text{-}K2\text{-isometry} \ Hi$   
 $\wedge \ apply\text{-}cltn2 \ east \ Hi = ?q\$i$   
 $\wedge \ apply\text{-}cltn2 \ west \ Hi = ?p\$i$   
 $\wedge \ apply\text{-}cltn2 \ north \ Hi = ?s\$i$   
 $\wedge \ apply\text{-}cltn2 \ far\text{-}north \ Hi = ?r\$i$   
**have**  $\forall \ i. \ is\text{-}K2\text{-isometry} \ (?H\$i)$   
 $\wedge \ apply\text{-}cltn2 \ east \ (?H\$i) = ?q\$i$   
 $\wedge \ apply\text{-}cltn2 \ west \ (?H\$i) = ?p\$i$   
 $\wedge \ apply\text{-}cltn2 \ north \ (?H\$i) = ?s\$i$   
 $\wedge \ apply\text{-}cltn2 \ far\text{-}north \ (?H\$i) = ?r\$i$   
**proof**  
**fix**  $i :: 2$   
**from**  $\langle \forall \ i. \ ?p\$i \in S \rangle$  **have**  $?p\$i \in S \ ..$

**from**  $\langle \forall \ i. \ ?q\$i \neq ?p\$i \wedge ?q\$i \in S \wedge \text{proj2-incident} \ (?q\$i) \ (?!\$i) \rangle$   
**have**  $?q\$i \neq ?p\$i$  **and**  $?q\$i \in S$   
**by** *simp-all*

**from**  $\langle \forall \ i. \ ?s\$i \neq ?r\$i \wedge ?s\$i \in S \wedge \text{proj2-incident} \ (?s\$i) \ (?m\$i) \rangle$   
**have**  $?s\$i \in S$  **and**  $\text{proj2-incident} \ (?s\$i) \ (?m\$i)$  **by** *simp-all*  
**from**  $\langle \text{proj2-incident} \ (?s\$i) \ (?m\$i) \rangle$   
**and**  $\langle \forall \ i. \ \forall \ u. \ \text{proj2-incident} \ u \ (?m\$i) \longrightarrow \neg (u = ?p\$i \vee u = ?q\$i) \rangle$   
**have**  $?s\$i \notin \{?q\$i, ?p\$i\}$  **by** *fast*  
**with**  $\langle ?q\$i \in S \rangle$  **and**  $\langle ?p\$i \in S \rangle$  **and**  $\langle ?s\$i \in S \rangle$  **and**  $\langle ?q\$i \neq ?p\$i \rangle$   
**have**  $\exists \ Hi. \ is\text{-}K2\text{-isometry} \ Hi$   
 $\wedge \ apply\text{-}cltn2 \ east \ Hi = ?q\$i$   
 $\wedge \ apply\text{-}cltn2 \ west \ Hi = ?p\$i$   
 $\wedge \ apply\text{-}cltn2 \ north \ Hi = ?s\$i$   
 $\wedge \ apply\text{-}cltn2 \ far\text{-}north \ Hi = ?r\$i$   
**by** (*simp add: statement65-special-case*)  
**with** *someI-ex* [of  $\lambda \ Hi. \ is\text{-}K2\text{-isometry} \ Hi$   
 $\wedge \ apply\text{-}cltn2 \ east \ Hi = ?q\$i$   
 $\wedge \ apply\text{-}cltn2 \ west \ Hi = ?p\$i$   
 $\wedge \ apply\text{-}cltn2 \ north \ Hi = ?s\$i$   
 $\wedge \ apply\text{-}cltn2 \ far\text{-}north \ Hi = ?r\$i$ ]  
**show**  $is\text{-}K2\text{-isometry} \ (?H\$i)$   
 $\wedge \ apply\text{-}cltn2 \ east \ (?H\$i) = ?q\$i$   
 $\wedge \ apply\text{-}cltn2 \ west \ (?H\$i) = ?p\$i$   
 $\wedge \ apply\text{-}cltn2 \ north \ (?H\$i) = ?s\$i$   
 $\wedge \ apply\text{-}cltn2 \ far\text{-}north \ (?H\$i) = ?r\$i$   
**by** *simp*

**qed**  
**hence**  $is\text{-}K2\text{-isometry} \ (?H\$1)$   
**and**  $apply\text{-}cltn2 \ east \ (?H\$1) = ?q\$1$   
**and**  $apply\text{-}cltn2 \ west \ (?H\$1) = ?p\$1$

**and** *apply-cltn2 north* (?H\$1) = ?s\$1  
**and** *apply-cltn2 far-north* (?H\$1) = ?r\$1  
**and** *is-K2-isometry* (?H\$2)  
**and** *apply-cltn2 east* (?H\$2) = ?q\$2  
**and** *apply-cltn2 west* (?H\$2) = ?p\$2  
**and** *apply-cltn2 north* (?H\$2) = ?s\$2  
**and** *apply-cltn2 far-north* (?H\$2) = ?r\$2  
**by** *fast+*

**let** ?J = *cltn2-compose* (*cltn2-inverse* (?H\$1)) (?H\$2)  
**from**  $\langle$ *is-K2-isometry* (?H\$1) $\rangle$  **and**  $\langle$ *is-K2-isometry* (?H\$2) $\rangle$   
**have** *is-K2-isometry* ?J  
**by** (*simp only: cltn2-inverse-is-K2-isometry cltn2-compose-is-K2-isometry*)

**from**  $\langle$ *apply-cltn2 west* (?H\$1) = ?p\$1 $\rangle$   
**have** *apply-cltn2 p1* (*cltn2-inverse* (?H\$1)) = *west*  
**by** (*simp add: cltn2.act-inv-iff [simplified]*)  
**with**  $\langle$ *apply-cltn2 west* (?H\$2) = ?p\$2 $\rangle$   
**have** *apply-cltn2 p1* ?J = *p2*  
**by** (*simp add: cltn2.act-act [simplified, symmetric]*)

**from**  $\langle$ *apply-cltn2 east* (?H\$1) = ?q\$1 $\rangle$   
**have** *apply-cltn2* (?q\$1) (*cltn2-inverse* (?H\$1)) = *east*  
**by** (*simp add: cltn2.act-inv-iff [simplified]*)  
**with**  $\langle$ *apply-cltn2 east* (?H\$2) = ?q\$2 $\rangle$   
**have** *apply-cltn2* (?q\$1) ?J = ?q\$2  
**by** (*simp add: cltn2.act-act [simplified, symmetric]*)  
**with**  $\langle$ ?q\$1  $\neq$  ?p\$1 $\rangle$  **and**  $\langle$ *apply-cltn2 p1* ?J = *p2* $\rangle$   
**and**  $\langle$ *proj2-incident* (?p\$1) (?l\$1) $\rangle$   
**and**  $\langle$ *proj2-incident* (?q\$1) (?l\$1) $\rangle$   
**and**  $\langle$ *proj2-incident* (?p\$2) (?l\$2) $\rangle$   
**and**  $\langle$ *proj2-incident* (?q\$2) (?l\$2) $\rangle$   
**have** *apply-cltn2-line* (?l\$1) ?J = (?l\$2)  
**by** (*simp add: apply-cltn2-line-unique*)  
**moreover from**  $\langle$ *proj2-incident* (?a\$1) (?l\$1) $\rangle$   
**have** *proj2-incident* (*apply-cltn2* (?a\$1) ?J) (*apply-cltn2-line* (?l\$1) ?J)  
**by** *simp*  
**ultimately have** *proj2-incident* (*apply-cltn2* (?a\$1) ?J) (?l\$2) **by** *simp*

**from**  $\langle$ *apply-cltn2 north* (?H\$1) = ?s\$1 $\rangle$   
**have** *apply-cltn2* (?s\$1) (*cltn2-inverse* (?H\$1)) = *north*  
**by** (*simp add: cltn2.act-inv-iff [simplified]*)  
**with**  $\langle$ *apply-cltn2 north* (?H\$2) = ?s\$2 $\rangle$   
**have** *apply-cltn2* (?s\$1) ?J = ?s\$2  
**by** (*simp add: cltn2.act-act [simplified, symmetric]*)

**from**  $\langle$ *apply-cltn2 far-north* (?H\$1) = ?r\$1 $\rangle$   
**have** *apply-cltn2* (?r\$1) (*cltn2-inverse* (?H\$1)) = *far-north*  
**by** (*simp add: cltn2.act-inv-iff [simplified]*)

**with**  $\langle \text{apply-cltn2 far-north } (?H\$2) = ?r\$2 \rangle$   
**have**  $\text{apply-cltn2 } (?r\$1) ?J = ?r\$2$   
**by** (*simp add: cltn2.act-act [simplified, symmetric]*)  
**with**  $\langle ?s\$1 \neq ?r\$1 \rangle$  **and**  $\langle \text{apply-cltn2 } (?s\$1) ?J = (?s\$2) \rangle$   
**and**  $\langle \text{proj2-incident } (?r\$1) (?m\$1) \rangle$   
**and**  $\langle \text{proj2-incident } (?s\$1) (?m\$1) \rangle$   
**and**  $\langle \text{proj2-incident } (?r\$2) (?m\$2) \rangle$   
**and**  $\langle \text{proj2-incident } (?s\$2) (?m\$2) \rangle$   
**have**  $\text{apply-cltn2-line } (?m\$1) ?J = (?m\$2)$   
**by** (*simp add: apply-cltn2-line-unique*)  
**moreover from**  $\langle \text{proj2-incident } (?a\$1) (?m\$1) \rangle$   
**have**  $\text{proj2-incident } (\text{apply-cltn2 } (?a\$1) ?J) (\text{apply-cltn2-line } (?m\$1) ?J)$   
**by** *simp*  
**ultimately have**  $\text{proj2-incident } (\text{apply-cltn2 } (?a\$1) ?J) (?m\$2)$  **by** *simp*

**from**  $\langle \forall i. \forall u. \text{proj2-incident } u (?m\$i) \longrightarrow \neg (u = ?p\$i \vee u = ?q\$i) \rangle$   
**have**  $\neg \text{proj2-incident } (?p\$2) (?m\$2)$  **by** *fast*  
**with**  $\langle \text{proj2-incident } (?p\$2) (?l\$2) \rangle$  **have**  $?m\$2 \neq ?l\$2$  **by** *auto*  
**with**  $\langle \text{proj2-incident } (?a\$2) (?l\$2) \rangle$   
**and**  $\langle \text{proj2-incident } (?a\$2) (?m\$2) \rangle$   
**and**  $\langle \text{proj2-incident } (\text{apply-cltn2 } (?a\$1) ?J) (?l\$2) \rangle$   
**and**  $\langle \text{proj2-incident } (\text{apply-cltn2 } (?a\$1) ?J) (?m\$2) \rangle$   
**and** *proj2-incident-unique*  
**have**  $\text{apply-cltn2 } a1 ?J = a2$  **by** *auto*  
**with**  $\langle \text{is-K2-isometry } ?J \rangle$  **and**  $\langle \text{apply-cltn2 } p1 ?J = p2 \rangle$   
**show**  $\exists J. \text{is-K2-isometry } J \wedge \text{apply-cltn2 } a1 J = a2 \wedge \text{apply-cltn2 } p1 J = p2$   
**by** *auto*

qed

**lemma** *K2-isometry-swap:*

**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$

**shows**  $\exists J. \text{is-K2-isometry } J \wedge \text{apply-cltn2 } a J = b \wedge \text{apply-cltn2 } b J = a$

**proof** –

**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$

**have**  $a \in K2$  **and**  $b \in K2$  **by** *simp-all*

**let**  $?l = \text{proj2-line-through } a b$

**have**  $\text{proj2-incident } a ?l$  **and**  $\text{proj2-incident } b ?l$

**by** (*rule proj2-line-through-incident*)**+**

**from**  $\langle a \in K2 \rangle$  **and**  $\langle \text{proj2-incident } a ?l \rangle$

**and** *line-through-K2-intersect-S-exactly-twice [of a ?l]*

**obtain**  $p$  **and**  $q$  **where**  $p \neq q$

**and**  $p \in S$  **and**  $q \in S$

**and**  $\text{proj2-incident } p ?l$  **and**  $\text{proj2-incident } q ?l$

**and**  $\forall r \in S. \text{proj2-incident } r ?l \longrightarrow r = p \vee r = q$

**by** *auto*

**from**  $\langle a \in K2 \rangle$  **and**  $\langle b \in K2 \rangle$  **and**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$

**and** *statement66-existence [of a b p q]*

**obtain**  $J$  **where**  $\text{is-K2-isometry } J$  **and**  $\text{apply-cltn2 } a J = b$

**and**  $\text{apply-cltn2 } p \ J = q$   
**by** *auto*  
**from**  $\langle \text{apply-cltn2 } a \ J = b \rangle$  **and**  $\langle \text{apply-cltn2 } p \ J = q \rangle$   
**and**  $\langle \text{proj2-incident } b \ ?l \rangle$  **and**  $\langle \text{proj2-incident } q \ ?l \rangle$   
**have**  $\text{proj2-incident } (\text{apply-cltn2 } a \ J) \ ?l$   
**and**  $\text{proj2-incident } (\text{apply-cltn2 } p \ J) \ ?l$   
**by** *simp-all*

**from**  $\langle a \in K2 \rangle$  **and**  $\langle p \in S \rangle$  **have**  $a \neq p$   
**unfolding** *S-def* **and** *K2-def*  
**by** *auto*  
**with**  $\langle \text{proj2-incident } a \ ?l \rangle$   
**and**  $\langle \text{proj2-incident } p \ ?l \rangle$   
**and**  $\langle \text{proj2-incident } (\text{apply-cltn2 } a \ J) \ ?l \rangle$   
**and**  $\langle \text{proj2-incident } (\text{apply-cltn2 } p \ J) \ ?l \rangle$   
**have**  $\text{apply-cltn2-line } ?l \ J = ?l$  **by** (*simp add: apply-cltn2-line-unique*)  
**with**  $\langle \text{proj2-incident } q \ ?l \rangle$  **and**  $\text{apply-cltn2-preserve-incident } [\text{of } q \ J \ ?l]$   
**have**  $\text{proj2-incident } (\text{apply-cltn2 } q \ J) \ ?l$  **by** *simp*

**from**  $\langle q \in S \rangle$  **and**  $\langle \text{is-K2-isometry } J \rangle$   
**have**  $\text{apply-cltn2 } q \ J \in S$  **by** (*unfold is-K2-isometry-def*) *simp*  
**with**  $\langle \text{proj2-incident } (\text{apply-cltn2 } q \ J) \ ?l \rangle$   
**and**  $\langle \forall r \in S. \text{proj2-incident } r \ ?l \longrightarrow r = p \vee r = q \rangle$   
**have**  $\text{apply-cltn2 } q \ J = p \vee \text{apply-cltn2 } q \ J = q$  **by** *simp*

**have**  $\text{apply-cltn2 } q \ J \neq q$   
**proof**  
**assume**  $\text{apply-cltn2 } q \ J = q$   
**with**  $\langle \text{apply-cltn2 } p \ J = q \rangle$   
**have**  $\text{apply-cltn2 } p \ J = \text{apply-cltn2 } q \ J$  **by** *simp*  
**hence**  $p = q$  **by** (*rule apply-cltn2-injective [of p J q]*)  
**with**  $\langle p \neq q \rangle$  **show** *False ..*

**qed**  
**with**  $\langle \text{apply-cltn2 } q \ J = p \vee \text{apply-cltn2 } q \ J = q \rangle$   
**have**  $\text{apply-cltn2 } q \ J = p$  **by** *simp*  
**with**  $\langle p \neq q \rangle$   
**and**  $\langle \text{apply-cltn2 } p \ J = q \rangle$   
**and**  $\langle \text{proj2-incident } p \ ?l \rangle$   
**and**  $\langle \text{proj2-incident } q \ ?l \rangle$   
**and**  $\langle \text{proj2-incident } a \ ?l \rangle$   
**and** *statement55*  
**have**  $\text{apply-cltn2 } (\text{apply-cltn2 } a \ J) \ J = a$  **by** *simp*  
**with**  $\langle \text{apply-cltn2 } a \ J = b \rangle$  **have**  $\text{apply-cltn2 } b \ J = a$  **by** *simp*  
**with**  $\langle \text{is-K2-isometry } J \rangle$  **and**  $\langle \text{apply-cltn2 } a \ J = b \rangle$   
**show**  $\exists J. \text{is-K2-isometry } J \wedge \text{apply-cltn2 } a \ J = b \wedge \text{apply-cltn2 } b \ J = a$   
**by** (*simp add: exI [of - J]*)

**qed**

**theorem** *hyp2-axiom1*:  $\forall a \ b. a \ b \equiv_K b \ a$

**proof standard+**  
**fix**  $a\ b$   
**let**  $?a' = \text{Rep-hyp2 } a$   
**let**  $?b' = \text{Rep-hyp2 } b$   
**from**  $\text{Rep-hyp2}$  **and**  $K2\text{-isometry-swap}$  [of  $?a' ?b'$ ]  
**obtain**  $J$  **where**  $\text{is-K2-isometry } J$  **and**  $\text{apply-cltn2 } ?a' J = ?b'$   
**and**  $\text{apply-cltn2 } ?b' J = ?a'$   
**by** *auto*

**from**  $\langle \text{apply-cltn2 } ?a' J = ?b' \rangle$  **and**  $\langle \text{apply-cltn2 } ?b' J = ?a' \rangle$   
**have**  $\text{hyp2-cltn2 } a J = b$  **and**  $\text{hyp2-cltn2 } b J = a$   
**unfolding**  $\text{hyp2-cltn2-def}$  **by** (*simp-all add: Rep-hyp2-inverse*)  
**with**  $\langle \text{is-K2-isometry } J \rangle$   
**show**  $a b \equiv_K b a$   
**by** (*unfold real-hyp2-C-def*) (*simp add: exI [of - J]*)

**qed**

**theorem hyp2-axiom2:**  $\forall a\ b\ p\ q\ r\ s. a\ b \equiv_K p\ q \wedge a\ b \equiv_K r\ s \longrightarrow p\ q \equiv_K r\ s$

**proof standard+**  
**fix**  $a\ b\ p\ q\ r\ s$   
**assume**  $a\ b \equiv_K p\ q \wedge a\ b \equiv_K r\ s$   
**then obtain**  $G$  **and**  $H$  **where**  $\text{is-K2-isometry } G$  **and**  $\text{is-K2-isometry } H$   
**and**  $\text{hyp2-cltn2 } a G = p$  **and**  $\text{hyp2-cltn2 } b G = q$   
**and**  $\text{hyp2-cltn2 } a H = r$  **and**  $\text{hyp2-cltn2 } b H = s$   
**by** (*unfold real-hyp2-C-def*) *auto*  
**let**  $?J = \text{cltn2-compose } (\text{cltn2-inverse } G) H$   
**from**  $\langle \text{is-K2-isometry } G \rangle$  **have**  $\text{is-K2-isometry } (\text{cltn2-inverse } G)$   
**by** (*rule cltn2-inverse-is-K2-isometry*)  
**with**  $\langle \text{is-K2-isometry } H \rangle$   
**have**  $\text{is-K2-isometry } ?J$  **by** (*simp only: cltn2-compose-is-K2-isometry*)

**from**  $\langle \text{is-K2-isometry } G \rangle$  **and**  $\langle \text{hyp2-cltn2 } a G = p \rangle$  **and**  $\langle \text{hyp2-cltn2 } b G = q \rangle$   
**and**  $K2\text{-isometry.act-inv-iff}$   
**have**  $\text{hyp2-cltn2 } p (\text{cltn2-inverse } G) = a$   
**and**  $\text{hyp2-cltn2 } q (\text{cltn2-inverse } G) = b$   
**by** *simp-all*  
**with**  $\langle \text{hyp2-cltn2 } a H = r \rangle$  **and**  $\langle \text{hyp2-cltn2 } b H = s \rangle$   
**and**  $\langle \text{is-K2-isometry } (\text{cltn2-inverse } G) \rangle$  **and**  $\langle \text{is-K2-isometry } H \rangle$   
**and**  $K2\text{-isometry.act-act}$  [*symmetric*]  
**have**  $\text{hyp2-cltn2 } p ?J = r$  **and**  $\text{hyp2-cltn2 } q ?J = s$  **by** *simp-all*  
**with**  $\langle \text{is-K2-isometry } ?J \rangle$   
**show**  $p\ q \equiv_K r\ s$   
**by** (*unfold real-hyp2-C-def*) (*simp add: exI [of - ?J]*)

**qed**

**theorem hyp2-axiom3:**  $\forall a\ b\ c. a\ b \equiv_K c\ c \longrightarrow a = b$

**proof standard+**  
**fix**  $a\ b\ c$   
**assume**  $a\ b \equiv_K c\ c$

**then obtain  $J$  where  $is\text{-}K2\text{-isometry } J$**   
**and  $hyp2\text{-cltn2 } a J = c$  and  $hyp2\text{-cltn2 } b J = c$**   
**by  $(unfold\ real\text{-}hyp2\text{-}C\text{-}def)$   $auto$**   
**from  $\langle hyp2\text{-cltn2 } a J = c \rangle$  and  $\langle hyp2\text{-cltn2 } b J = c \rangle$**   
**have  $hyp2\text{-cltn2 } a J = hyp2\text{-cltn2 } b J$  by  $simp$**   
  
**from  $\langle is\text{-}K2\text{-isometry } J \rangle$**   
**have  $apply\text{-cltn2 } (Rep\text{-}hyp2\ a) J \in hyp2$**   
**and  $apply\text{-cltn2 } (Rep\text{-}hyp2\ b) J \in hyp2$**   
**by  $(rule\ apply\text{-cltn2}\text{-}Rep\text{-}hyp2)$ +**  
**with  $\langle hyp2\text{-cltn2 } a J = hyp2\text{-cltn2 } b J \rangle$**   
**have  $apply\text{-cltn2 } (Rep\text{-}hyp2\ a) J = apply\text{-cltn2 } (Rep\text{-}hyp2\ b) J$**   
**by  $(unfold\ hyp2\text{-cltn2}\text{-}def)$   $(simp\ add:\ Abs\text{-}hyp2\text{-}inject)$**   
**hence  $Rep\text{-}hyp2\ a = Rep\text{-}hyp2\ b$  by  $(rule\ apply\text{-cltn2}\text{-}injective)$**   
**thus  $a = b$  by  $(simp\ add:\ Rep\text{-}hyp2\text{-}inject)$**   
**qed**

**interpretation  $hyp2:\ tarski\text{-}first3\ real\text{-}hyp2\text{-}C$**   
**using  $hyp2\text{-axiom1}$  and  $hyp2\text{-axiom2}$  and  $hyp2\text{-axiom3}$**   
**by  $unfold\text{-}locales$**

## 8.7 Some lemmas about betweenness

**lemma  $S\text{-at}\text{-}edge$ :**

**assumes  $p \in S$  and  $q \in hyp2 \cup S$  and  $r \in hyp2 \cup S$  and  $proj2\text{-}Col\ p\ q\ r$**   
**shows  $B_{\mathbf{R}} (cart2\text{-}pt\ p) (cart2\text{-}pt\ q) (cart2\text{-}pt\ r)$**   
 $\vee B_{\mathbf{R}} (cart2\text{-}pt\ p) (cart2\text{-}pt\ r) (cart2\text{-}pt\ q)$   
**(is  $B_{\mathbf{R}}\ ?cp\ ?cq\ ?cr \vee -$ )**

**proof** –

**from  $\langle p \in S \rangle$  and  $\langle q \in hyp2 \cup S \rangle$  and  $\langle r \in hyp2 \cup S \rangle$**   
**have  $z\text{-non}\text{-}zero\ p$  and  $z\text{-non}\text{-}zero\ q$  and  $z\text{-non}\text{-}zero\ r$**   
**by  $(simp\text{-}all\ add:\ hyp2\text{-}S\text{-}z\text{-non}\text{-}zero)$**   
**with  $\langle proj2\text{-}Col\ p\ q\ r \rangle$**   
**have  $real\text{-}euclid.\text{Col}\ ?cp\ ?cq\ ?cr$  by  $(simp\ add:\ proj2\text{-}Col\text{-}iff\text{-}euclid\text{-}cart2)$**

**with  $\langle z\text{-non}\text{-}zero\ p \rangle$  and  $\langle z\text{-non}\text{-}zero\ q \rangle$  and  $\langle z\text{-non}\text{-}zero\ r \rangle$**   
**have  $proj2\text{-}pt\ ?cp = p$  and  $proj2\text{-}pt\ ?cq = q$  and  $proj2\text{-}pt\ ?cr = r$**   
**by  $(simp\text{-}all\ add:\ proj2\text{-}cart2)$**   
**from  $\langle proj2\text{-}pt\ ?cp = p \rangle$  and  $\langle p \in S \rangle$**   
**have  $norm\ ?cp = 1$  by  $(simp\ add:\ norm\text{-}eq\text{-}1\text{-}iff\text{-}in\text{-}S)$**

**from  $\langle proj2\text{-}pt\ ?cq = q \rangle$  and  $\langle proj2\text{-}pt\ ?cr = r \rangle$**   
**and  $\langle q \in hyp2 \cup S \rangle$  and  $\langle r \in hyp2 \cup S \rangle$**   
**have  $norm\ ?cq \leq 1$  and  $norm\ ?cr \leq 1$**   
**by  $(simp\text{-}all\ add:\ norm\text{-}le\text{-}1\text{-}iff\text{-}in\text{-}hyp2\text{-}S)$**

**show  $B_{\mathbf{R}}\ ?cp\ ?cq\ ?cr \vee B_{\mathbf{R}}\ ?cp\ ?cr\ ?cq$**   
**proof cases**  
**assume  $B_{\mathbf{R}}\ ?cr\ ?cp\ ?cq$**

**then obtain  $k$  where  $k \geq 0$  and  $k \leq 1$**   
**and  $?cp - ?cr = k *_R (?cq - ?cr)$**   
**by (unfold real-euclid-B-def) auto**  
**from  $\langle ?cp - ?cr = k *_R (?cq - ?cr) \rangle$**   
**have  $?cp = k *_R ?cq + (1 - k) *_R ?cr$  by (simp add: algebra-simps)**  
**with  $\langle \text{norm } ?cp = 1 \rangle$  have  $\text{norm } (k *_R ?cq + (1 - k) *_R ?cr) = 1$  by simp**  
**with norm-triangle-ineq [of  $k *_R ?cq$   $(1 - k) *_R ?cr$ ]**  
**have  $\text{norm } (k *_R ?cq) + \text{norm } ((1 - k) *_R ?cr) \geq 1$  by simp**

**from  $\langle k \geq 0 \rangle$  and  $\langle k \leq 1 \rangle$**   
**have  $\text{norm } (k *_R ?cq) + \text{norm } ((1 - k) *_R ?cr)$**   
 $= k * \text{norm } ?cq + (1 - k) * \text{norm } ?cr$   
**by simp**  
**with  $\langle \text{norm } (k *_R ?cq) + \text{norm } ((1 - k) *_R ?cr) \geq 1 \rangle$**   
**have  $k * \text{norm } ?cq + (1 - k) * \text{norm } ?cr \geq 1$  by simp**

**from  $\langle \text{norm } ?cq \leq 1 \rangle$  and  $\langle k \geq 0 \rangle$  and mult-mono [of  $k$   $k$   $\text{norm } ?cq$  1]**  
**have  $k * \text{norm } ?cq \leq k$  by simp**

**from  $\langle \text{norm } ?cr \leq 1 \rangle$  and  $\langle k \leq 1 \rangle$**   
**and mult-mono [of  $1 - k$   $1 - k$   $\text{norm } ?cr$  1]**  
**have  $(1 - k) * \text{norm } ?cr \leq 1 - k$  by simp**  
**with  $\langle k * \text{norm } ?cq \leq k \rangle$**   
**have  $k * \text{norm } ?cq + (1 - k) * \text{norm } ?cr \leq 1$  by simp**  
**with  $\langle k * \text{norm } ?cq + (1 - k) * \text{norm } ?cr \geq 1 \rangle$**   
**have  $k * \text{norm } ?cq + (1 - k) * \text{norm } ?cr = 1$  by simp**  
**with  $\langle k * \text{norm } ?cq \leq k \rangle$  have  $(1 - k) * \text{norm } ?cr \geq 1 - k$  by simp**  
**with  $\langle (1 - k) * \text{norm } ?cr \leq 1 - k \rangle$  have  $(1 - k) * \text{norm } ?cr = 1 - k$  by**  
*simp*

**with  $\langle k * \text{norm } ?cq + (1 - k) * \text{norm } ?cr = 1 \rangle$  have  $k * \text{norm } ?cq = k$  by**  
*simp*

**have  $?cp = ?cq \vee ?cq = ?cr \vee ?cr = ?cp$**   
**proof cases**  
**assume  $k = 0 \vee k = 1$**   
**with  $\langle ?cp = k *_R ?cq + (1 - k) *_R ?cr \rangle$**   
**show  $?cp = ?cq \vee ?cq = ?cr \vee ?cr = ?cp$  by auto**  
**next**  
**assume  $\neg (k = 0 \vee k = 1)$**   
**hence  $k \neq 0$  and  $k \neq 1$  by simp-all**  
**with  $\langle k * \text{norm } ?cq = k \rangle$  and  $\langle (1 - k) * \text{norm } ?cr = 1 - k \rangle$**   
**have  $\text{norm } ?cq = 1$  and  $\text{norm } ?cr = 1$  by simp-all**  
**with  $\langle \text{proj2-pt } ?cq = q \rangle$  and  $\langle \text{proj2-pt } ?cr = r \rangle$**   
**have  $q \in S$  and  $r \in S$  by (simp-all add: norm-eq-1-iff-in-S)**  
**with  $\langle p \in S \rangle$  have  $\{p, q, r\} \subseteq S$  by simp**

**from  $\langle \text{proj2-Col } p \ q \ r \rangle$**   
**have  $\text{proj2-set-Col } \{p, q, r\}$  by (simp add: proj2-Col-iff-set-Col)**  
**with  $\langle \{p, q, r\} \subseteq S \rangle$  have  $\text{card } \{p, q, r\} \leq 2$  by (rule card-line-intersect-S)**

```

have  $p = q \vee q = r \vee r = p$ 
proof (rule ccontr)
  assume  $\neg (p = q \vee q = r \vee r = p)$ 
  hence  $p \neq q$  and  $q \neq r$  and  $r \neq p$  by simp-all
  from  $\langle q \neq r \rangle$  have  $\text{card } \{q, r\} = 2$  by simp
  with  $\langle p \neq q \rangle$  and  $\langle r \neq p \rangle$  have  $\text{card } \{p, q, r\} = 3$  by simp
  with  $\langle \text{card } \{p, q, r\} \leq 2 \rangle$  show False by simp
qed
thus  $?cp = ?cq \vee ?cq = ?cr \vee ?cr = ?cp$  by auto
qed
thus  $B_{\mathbf{R}} ?cp ?cq ?cr \vee B_{\mathbf{R}} ?cp ?cr ?cq$ 
  by (auto simp add: real-euclid.th3-1 real-euclid.th3-2)
next
  assume  $\neg B_{\mathbf{R}} ?cr ?cp ?cq$ 
  with  $\langle \text{real-euclid.Col } ?cp ?cq ?cr \rangle$ 
  show  $B_{\mathbf{R}} ?cp ?cq ?cr \vee B_{\mathbf{R}} ?cp ?cr ?cq$ 
    unfolding real-euclid.Col-def
    by (auto simp add: real-euclid.th3-1 real-euclid.th3-2)
qed
qed

lemma hyp2-in-middle:
  assumes  $p \in S$  and  $q \in S$  and  $r \in \text{hyp2} \cup S$  and  $\text{proj2-Col } p \ q \ r$ 
  and  $p \neq q$ 
  shows  $B_{\mathbf{R}} (\text{cart2-pt } p) (\text{cart2-pt } r) (\text{cart2-pt } q)$  (is  $B_{\mathbf{R}} ?cp ?cr ?cq$ )
proof (rule ccontr)
  assume  $\neg B_{\mathbf{R}} ?cp ?cr ?cq$ 
  hence  $\neg B_{\mathbf{R}} ?cq ?cr ?cp$ 
    by (auto simp add: real-euclid.th3-2 [of ?cq ?cr ?cp])

  from  $\langle p \in S \rangle$  and  $\langle q \in S \rangle$  and  $\langle r \in \text{hyp2} \cup S \rangle$  and  $\langle \text{proj2-Col } p \ q \ r \rangle$ 
  have  $B_{\mathbf{R}} ?cp ?cq ?cr \vee B_{\mathbf{R}} ?cp ?cr ?cq$  by (simp add: S-at-edge)
  with  $\langle \neg B_{\mathbf{R}} ?cp ?cr ?cq \rangle$  have  $B_{\mathbf{R}} ?cp ?cq ?cr$  by simp

  from  $\langle \text{proj2-Col } p \ q \ r \rangle$  and  $\text{proj2-Col-permute}$  have  $\text{proj2-Col } q \ p \ r$  by fast
  with  $\langle q \in S \rangle$  and  $\langle p \in S \rangle$  and  $\langle r \in \text{hyp2} \cup S \rangle$ 
  have  $B_{\mathbf{R}} ?cq ?cp ?cr \vee B_{\mathbf{R}} ?cq ?cr ?cp$  by (simp add: S-at-edge)
  with  $\langle \neg B_{\mathbf{R}} ?cq ?cr ?cp \rangle$  have  $B_{\mathbf{R}} ?cq ?cp ?cr$  by simp
  with  $\langle B_{\mathbf{R}} ?cp ?cq ?cr \rangle$  have  $?cp = ?cq$  by (rule real-euclid.th3-4)
  hence  $\text{proj2-pt } ?cp = \text{proj2-pt } ?cq$  by simp

  from  $\langle p \in S \rangle$  and  $\langle q \in S \rangle$ 
  have  $z\text{-non-zero } p$  and  $z\text{-non-zero } q$  by (simp-all add: hyp2-S-z-non-zero)
  hence  $\text{proj2-pt } ?cp = p$  and  $\text{proj2-pt } ?cq = q$  by (simp-all add: proj2-cart2)
  with  $\langle \text{proj2-pt } ?cp = \text{proj2-pt } ?cq \rangle$  have  $p = q$  by simp
  with  $\langle p \neq q \rangle$  show False ..
qed

```

**lemma** *hyp2-incident-in-middle*:  
 assumes  $p \neq q$  and  $p \in S$  and  $q \in S$  and  $a \in \text{hyp2} \cup S$   
 and *proj2-incident*  $p$   $l$  and *proj2-incident*  $q$   $l$  and *proj2-incident*  $a$   $l$   
 shows  $B_{\mathbb{R}}$  (*cart2-pt*  $p$ ) (*cart2-pt*  $a$ ) (*cart2-pt*  $q$ )  
**proof** –  
 from  $\langle \text{proj2-incident } p \ l \rangle$  and  $\langle \text{proj2-incident } q \ l \rangle$  and  $\langle \text{proj2-incident } a \ l \rangle$   
 have *proj2-Col*  $p$   $q$   $a$  by (*rule proj2-incident-Col*)  
 from  $\langle p \in S \rangle$  and  $\langle q \in S \rangle$  and  $\langle a \in \text{hyp2} \cup S \rangle$  and *this* and  $\langle p \neq q \rangle$   
 show  $B_{\mathbb{R}}$  (*cart2-pt*  $p$ ) (*cart2-pt*  $a$ ) (*cart2-pt*  $q$ )  
 by (*rule hyp2-in-middle*)  
**qed**

**lemma** *extend-to-S*:  
 assumes  $p \in \text{hyp2} \cup S$  and  $q \in \text{hyp2} \cup S$   
 shows  $\exists r \in S. B_{\mathbb{R}}$  (*cart2-pt*  $p$ ) (*cart2-pt*  $q$ ) (*cart2-pt*  $r$ )  
 (is  $\exists r \in S. B_{\mathbb{R}}$  *?cp* *?cq* (*cart2-pt*  $r$ ))  
**proof** *cases*  
 assume  $q \in S$   
  
 have  $B_{\mathbb{R}}$  *?cp* *?cq* *?cq* by (*rule real-euclid.th3-1*)  
 with  $\langle q \in S \rangle$  show  $\exists r \in S. B_{\mathbb{R}}$  *?cp* *?cq* (*cart2-pt*  $r$ ) by *auto*  
**next**  
 assume  $q \notin S$   
 with  $\langle q \in \text{hyp2} \cup S \rangle$  have  $q \in K2$  by *simp*  
  
 let  $?l = \text{proj2-line-through } p \ q$   
 have *proj2-incident*  $p$   $?l$  and *proj2-incident*  $q$   $?l$   
 by (*rule proj2-line-through-incident*)  
 from  $\langle q \in K2 \rangle$  and  $\langle \text{proj2-incident } q \ ?l \rangle$   
 and *line-through-K2-intersect-S-twice* [*of*  $q$   $?l$ ]  
 obtain  $s$  and  $t$  where  $s \neq t$  and  $s \in S$  and  $t \in S$   
 and *proj2-incident*  $s$   $?l$  and *proj2-incident*  $t$   $?l$   
 by *auto*  
 let  $?cs = \text{cart2-pt } s$   
 let  $?ct = \text{cart2-pt } t$   
  
 from  $\langle \text{proj2-incident } s \ ?l \rangle$   
 and  $\langle \text{proj2-incident } t \ ?l \rangle$   
 and  $\langle \text{proj2-incident } p \ ?l \rangle$   
 and  $\langle \text{proj2-incident } q \ ?l \rangle$   
 have *proj2-Col*  $s$   $p$   $q$  and *proj2-Col*  $t$   $p$   $q$  and *proj2-Col*  $s$   $t$   $q$   
 by (*simp-all add: proj2-incident-Col*)  
 from  $\langle \text{proj2-Col } s \ p \ q \rangle$  and  $\langle \text{proj2-Col } t \ p \ q \rangle$   
 and  $\langle s \in S \rangle$  and  $\langle t \in S \rangle$  and  $\langle p \in \text{hyp2} \cup S \rangle$  and  $\langle q \in \text{hyp2} \cup S \rangle$   
 have  $B_{\mathbb{R}}$  *?cs* *?cp* *?cq*  $\vee$   $B_{\mathbb{R}}$  *?cs* *?cq* *?cp* and  $B_{\mathbb{R}}$  *?ct* *?cp* *?cq*  $\vee$   $B_{\mathbb{R}}$  *?ct* *?cq* *?cp*  
 by (*simp-all add: S-at-edge*)  
 with *real-euclid.th3-2*  
 have  $B_{\mathbb{R}}$  *?cq* *?cp* *?cs*  $\vee$   $B_{\mathbb{R}}$  *?cp* *?cq* *?cs* and  $B_{\mathbb{R}}$  *?cq* *?cp* *?ct*  $\vee$   $B_{\mathbb{R}}$  *?cp* *?cq* *?ct*  
 by *fast+*

from  $\langle s \in S \rangle$  and  $\langle t \in S \rangle$  and  $\langle q \in \text{hyp2} \cup S \rangle$  and  $\langle \text{proj2-Col } s \ t \ q \rangle$  and  $\langle s \neq t \rangle$

have  $B_{\mathbf{R}} \ ?cs \ ?cq \ ?ct$  by (rule *hyp2-in-middle*)  
hence  $B_{\mathbf{R}} \ ?ct \ ?cq \ ?cs$  by (rule *real-euclid.th3-2*)

have  $B_{\mathbf{R}} \ ?cp \ ?cq \ ?cs \ \vee \ B_{\mathbf{R}} \ ?cp \ ?cq \ ?ct$

proof (rule *ccontr*)

assume  $\neg (B_{\mathbf{R}} \ ?cp \ ?cq \ ?cs \ \vee \ B_{\mathbf{R}} \ ?cp \ ?cq \ ?ct)$

hence  $\neg B_{\mathbf{R}} \ ?cp \ ?cq \ ?cs$  and  $\neg B_{\mathbf{R}} \ ?cp \ ?cq \ ?ct$  by *simp-all*

with  $\langle B_{\mathbf{R}} \ ?cq \ ?cp \ ?cs \ \vee \ B_{\mathbf{R}} \ ?cp \ ?cq \ ?cs \rangle$

and  $\langle B_{\mathbf{R}} \ ?cq \ ?cp \ ?ct \ \vee \ B_{\mathbf{R}} \ ?cp \ ?cq \ ?ct \rangle$

have  $B_{\mathbf{R}} \ ?cq \ ?cp \ ?cs$  and  $B_{\mathbf{R}} \ ?cq \ ?cp \ ?ct$  by *simp-all*

from  $\langle \neg B_{\mathbf{R}} \ ?cp \ ?cq \ ?cs \rangle$  and  $\langle B_{\mathbf{R}} \ ?cq \ ?cp \ ?cs \rangle$  have  $?cp \neq ?cq$  by *auto*

with  $\langle B_{\mathbf{R}} \ ?cq \ ?cp \ ?cs \rangle$  and  $\langle B_{\mathbf{R}} \ ?cq \ ?cp \ ?ct \rangle$

have  $B_{\mathbf{R}} \ ?cq \ ?cs \ ?ct \ \vee \ B_{\mathbf{R}} \ ?cq \ ?ct \ ?cs$

by (*simp add: real-euclid-th5-1 [of ?cq ?cp ?cs ?ct]*)

with  $\langle B_{\mathbf{R}} \ ?cs \ ?cq \ ?ct \rangle$  and  $\langle B_{\mathbf{R}} \ ?ct \ ?cq \ ?cs \rangle$

have  $?cq = ?cs \ \vee \ ?cq = ?ct$  by (*auto simp add: real-euclid.th3-4*)

with  $\langle q \in \text{hyp2} \cup S \rangle$  and  $\langle s \in S \rangle$  and  $\langle t \in S \rangle$

have  $q = s \ \vee \ q = t$  by (*auto simp add: hyp2-S-cart2-inj*)

with  $\langle s \in S \rangle$  and  $\langle t \in S \rangle$  have  $q \in S$  by *auto*

with  $\langle q \notin S \rangle$  show *False ..*

qed

with  $\langle s \in S \rangle$  and  $\langle t \in S \rangle$  show  $\exists r \in S. B_{\mathbf{R}} \ ?cp \ ?cq \ (\text{cart2-pt } r)$  by *auto*

qed

**definition** *endpoint-in-S* :: *proj2*  $\Rightarrow$  *proj2*  $\Rightarrow$  *proj2* where

*endpoint-in-S*  $a \ b$

$\triangleq \epsilon \ p. p \in S \wedge B_{\mathbf{R}} \ (\text{cart2-pt } a) \ (\text{cart2-pt } b) \ (\text{cart2-pt } p)$

**lemma** *endpoint-in-S*:

assumes  $a \in \text{hyp2} \cup S$  and  $b \in \text{hyp2} \cup S$

shows *endpoint-in-S*  $a \ b \in S$  (is  $?p \in S$ )

and  $B_{\mathbf{R}} \ (\text{cart2-pt } a) \ (\text{cart2-pt } b) \ (\text{cart2-pt } (\text{endpoint-in-S } a \ b))$

(is  $B_{\mathbf{R}} \ ?ca \ ?cb \ ?cp$ )

proof –

from  $\langle a \in \text{hyp2} \cup S \rangle$  and  $\langle b \in \text{hyp2} \cup S \rangle$  and *extend-to-S*

have  $\exists p. p \in S \wedge B_{\mathbf{R}} \ ?ca \ ?cb \ (\text{cart2-pt } p)$  by *auto*

hence  $?p \in S \wedge B_{\mathbf{R}} \ ?ca \ ?cb \ ?cp$

by (*unfold endpoint-in-S-def*) (rule *someI-ex*)

thus  $?p \in S$  and  $B_{\mathbf{R}} \ ?ca \ ?cb \ ?cp$  by *simp-all*

qed

**lemma** *endpoint-in-S-swap*:

assumes  $a \neq b$  and  $a \in \text{hyp2} \cup S$  and  $b \in \text{hyp2} \cup S$

shows *endpoint-in-S*  $a \ b \neq \text{endpoint-in-S } b \ a$  (is  $?p \neq ?q$ )

proof

let  $?ca = \text{cart2-pt } a$

**let**  $?cb = \text{cart2-pt } b$   
**let**  $?cp = \text{cart2-pt } ?p$   
**let**  $?cq = \text{cart2-pt } ?q$   
**from**  $\langle a \neq b \rangle$  **and**  $\langle a \in \text{hyp2} \cup S \rangle$  **and**  $\langle b \in \text{hyp2} \cup S \rangle$   
**have**  $B_{\mathbb{R}} ?ca ?cb ?cp$  **and**  $B_{\mathbb{R}} ?cb ?ca ?cq$   
**by** (*simp-all add: endpoint-in-S*)

**assume**  $?p = ?q$   
**with**  $\langle B_{\mathbb{R}} ?cb ?ca ?cq \rangle$  **have**  $B_{\mathbb{R}} ?cb ?ca ?cp$  **by** *simp*  
**with**  $\langle B_{\mathbb{R}} ?ca ?cb ?cp \rangle$  **have**  $?ca = ?cb$  **by** (*rule real-euclid.th3-4*)  
**with**  $\langle a \in \text{hyp2} \cup S \rangle$  **and**  $\langle b \in \text{hyp2} \cup S \rangle$  **have**  $a = b$  **by** (*rule hyp2-S-cart2-inj*)  
**with**  $\langle a \neq b \rangle$  **show** *False ..*

**qed**

**lemma** *endpoint-in-S-incident:*

**assumes**  $a \neq b$  **and**  $a \in \text{hyp2} \cup S$  **and**  $b \in \text{hyp2} \cup S$   
**and** *proj2-incident*  $a l$  **and** *proj2-incident*  $b l$   
**shows** *proj2-incident* (*endpoint-in-S*  $a b$ )  $l$  (**is** *proj2-incident*  $?p l$ )

**proof** –

**from**  $\langle a \in \text{hyp2} \cup S \rangle$  **and**  $\langle b \in \text{hyp2} \cup S \rangle$   
**have**  $?p \in S$  **and**  $B_{\mathbb{R}} (\text{cart2-pt } a) (\text{cart2-pt } b) (\text{cart2-pt } ?p)$   
**(is**  $B_{\mathbb{R}} ?ca ?cb ?cp$ )  
**by** (*rule endpoint-in-S*)**+**

**from**  $\langle a \in \text{hyp2} \cup S \rangle$  **and**  $\langle b \in \text{hyp2} \cup S \rangle$  **and**  $\langle ?p \in S \rangle$   
**have** *z-non-zero*  $a$  **and** *z-non-zero*  $b$  **and** *z-non-zero*  $?p$   
**by** (*simp-all add: hyp2-S-z-non-zero*)

**from**  $\langle B_{\mathbb{R}} ?ca ?cb ?cp \rangle$   
**have** *real-euclid.Col*  $?ca ?cb ?cp$  **unfolding** *real-euclid.Col-def ..*  
**with**  $\langle \text{z-non-zero } a \rangle$  **and**  $\langle \text{z-non-zero } b \rangle$  **and**  $\langle \text{z-non-zero } ?p \rangle$  **and**  $\langle a \neq b \rangle$   
**and**  $\langle \text{proj2-incident } a l \rangle$  **and**  $\langle \text{proj2-incident } b l \rangle$   
**show** *proj2-incident*  $?p l$  **by** (*rule euclid-Col-cart2-incident*)

**qed**

**lemma** *endpoints-in-S-incident-unique:*

**assumes**  $a \neq b$  **and**  $a \in \text{hyp2} \cup S$  **and**  $b \in \text{hyp2} \cup S$  **and**  $p \in S$   
**and** *proj2-incident*  $a l$  **and** *proj2-incident*  $b l$  **and** *proj2-incident*  $p l$   
**shows**  $p = \text{endpoint-in-S } a b \vee p = \text{endpoint-in-S } b a$   
**(is**  $p = ?q \vee p = ?r$ )

**proof** –

**from**  $\langle a \neq b \rangle$  **and**  $\langle a \in \text{hyp2} \cup S \rangle$  **and**  $\langle b \in \text{hyp2} \cup S \rangle$   
**have**  $?q \neq ?r$  **by** (*rule endpoint-in-S-swap*)

**from**  $\langle a \in \text{hyp2} \cup S \rangle$  **and**  $\langle b \in \text{hyp2} \cup S \rangle$   
**have**  $?q \in S$  **and**  $?r \in S$  **by** (*simp-all add: endpoint-in-S*)

**from**  $\langle a \neq b \rangle$  **and**  $\langle a \in \text{hyp2} \cup S \rangle$  **and**  $\langle b \in \text{hyp2} \cup S \rangle$   
**and**  $\langle \text{proj2-incident } a l \rangle$  **and**  $\langle \text{proj2-incident } b l \rangle$

**have** *proj2-incident* ?q l **and** *proj2-incident* ?r l  
**by** (*simp-all add: endpoint-in-S-incident*)  
**with**  $\langle ?q \neq ?r \rangle$  **and**  $\langle ?q \in S \rangle$  **and**  $\langle ?r \in S \rangle$  **and**  $\langle p \in S \rangle$  **and**  $\langle \text{proj2-incident } p \text{ l} \rangle$   
**show**  $p = ?q \vee p = ?r$  **by** (*simp add: line-S-two-intersections-only*)  
**qed**

**lemma** *endpoint-in-S-unique*:

**assumes**  $a \neq b$  **and**  $a \in \text{hyp2} \cup S$  **and**  $b \in \text{hyp2} \cup S$  **and**  $p \in S$   
**and**  $B_{\mathbb{R}}$  (*cart2-pt* a) (*cart2-pt* b) (*cart2-pt* p) (**is**  $B_{\mathbb{R}}$  ?ca ?cb ?cp)  
**shows**  $p = \text{endpoint-in-S } a \ b$  (**is**  $p = ?q$ )

**proof** (*rule ccontr*)

**from**  $\langle a \in \text{hyp2} \cup S \rangle$  **and**  $\langle b \in \text{hyp2} \cup S \rangle$  **and**  $\langle p \in S \rangle$

**have** *z-non-zero* a **and** *z-non-zero* b **and** *z-non-zero* p

**by** (*simp-all add: hyp2-S-z-non-zero*)

**with**  $\langle B_{\mathbb{R}} \ ?ca \ ?cb \ ?cp \rangle$  **and** *euclid-B-cart2-common-line* [*of* a b p]

**obtain** l **where**

*proj2-incident* a l **and** *proj2-incident* b l **and** *proj2-incident* p l

**by** *auto*

**with**  $\langle a \neq b \rangle$  **and**  $\langle a \in \text{hyp2} \cup S \rangle$  **and**  $\langle b \in \text{hyp2} \cup S \rangle$  **and**  $\langle p \in S \rangle$

**have**  $p = ?q \vee p = \text{endpoint-in-S } b \ a$  (**is**  $p = ?q \vee p = ?r$ )

**by** (*rule endpoints-in-S-incident-unique*)

**assume**  $p \neq ?q$

**with**  $\langle p = ?q \vee p = ?r \rangle$  **have**  $p = ?r$  **by** *simp*

**with**  $\langle b \in \text{hyp2} \cup S \rangle$  **and**  $\langle a \in \text{hyp2} \cup S \rangle$

**have**  $B_{\mathbb{R}}$  ?cb ?ca ?cp **by** (*simp add: endpoint-in-S*)

**with**  $\langle B_{\mathbb{R}} \ ?ca \ ?cb \ ?cp \rangle$  **have** ?ca = ?cb **by** (*rule real-euclid.th3-4*)

**with**  $\langle a \in \text{hyp2} \cup S \rangle$  **and**  $\langle b \in \text{hyp2} \cup S \rangle$  **have**  $a = b$  **by** (*rule hyp2-S-cart2-inj*)

**with**  $\langle a \neq b \rangle$  **show** *False* ..

**qed**

**lemma** *between-hyp2-S*:

**assumes**  $p \in \text{hyp2} \cup S$  **and**  $r \in \text{hyp2} \cup S$  **and**  $k \geq 0$  **and**  $k \leq 1$

**shows** *proj2-pt* ( $k *_{\mathbb{R}}$  (*cart2-pt* r) +  $(1 - k) *_{\mathbb{R}}$  (*cart2-pt* p))  $\in \text{hyp2} \cup S$   
**(is** *proj2-pt* ?cq  $\in$  -)

**proof** -

**let** ?cp = *cart2-pt* p

**let** ?cr = *cart2-pt* r

**let** ?q = *proj2-pt* ?cq

**from**  $\langle p \in \text{hyp2} \cup S \rangle$  **and**  $\langle r \in \text{hyp2} \cup S \rangle$

**have** *z-non-zero* p **and** *z-non-zero* r **by** (*simp-all add: hyp2-S-z-non-zero*)

**hence** *proj2-pt* ?cp = p **and** *proj2-pt* ?cr = r **by** (*simp-all add: proj2-cart2*)

**with**  $\langle p \in \text{hyp2} \cup S \rangle$  **and**  $\langle r \in \text{hyp2} \cup S \rangle$

**have** *norm* ?cp  $\leq 1$  **and** *norm* ?cr  $\leq 1$

**by** (*simp-all add: norm-le-1-iff-in-hyp2-S*)

**from**  $\langle k \geq 0 \rangle$  **and**  $\langle k \leq 1 \rangle$

**and** *norm-triangle-ineq* [*of*  $k *_{\mathbb{R}}$  ?cr  $(1 - k) *_{\mathbb{R}}$  ?cp]

**have**  $\text{norm } ?cq \leq k * \text{norm } ?cr + (1 - k) * \text{norm } ?cp$  **by** *simp*  
**from**  $\langle k \geq 0 \rangle$  **and**  $\langle \text{norm } ?cr \leq 1 \rangle$  **and** *mult-mono* [*of k k norm ?cr 1*]  
**have**  $k * \text{norm } ?cr \leq k$  **by** *simp*  
  
**from**  $\langle k \leq 1 \rangle$  **and**  $\langle \text{norm } ?cp \leq 1 \rangle$   
**and** *mult-mono* [*of 1 - k 1 - k norm ?cp 1*]  
**have**  $(1 - k) * \text{norm } ?cp \leq 1 - k$  **by** *simp*  
**with**  $\langle \text{norm } ?cq \leq k * \text{norm } ?cr + (1 - k) * \text{norm } ?cp \rangle$  **and**  $\langle k * \text{norm } ?cr \leq k \rangle$   
**have**  $\text{norm } ?cq \leq 1$  **by** *simp*  
**thus**  $?q \in \text{hyp2} \cup S$  **by** (*simp add: norm-le-1-iff-in-hyp2-S*)  
**qed**

## 8.8 The Klein–Beltrami model satisfies axiom 4

**definition** *expansion-factor* :: *proj2*  $\Rightarrow$  *cltn2*  $\Rightarrow$  *real* **where**  
*expansion-factor*  $p J \triangleq (\text{cart2-append1 } p \ v * \text{cltn2-rep } J) \$ 3$

**lemma** *expansion-factor*:

**assumes**  $p \in \text{hyp2} \cup S$  **and** *is-K2-isometry*  $J$   
**shows** *expansion-factor*  $p J \neq 0$   
**and** *cart2-append1*  $p \ v * \text{cltn2-rep } J$   
 $= \text{expansion-factor } p J *_{\mathbb{R}} \text{cart2-append1 } (\text{apply-cltn2 } p J)$

**proof** –

**from**  $\langle p \in \text{hyp2} \cup S \rangle$  **and**  $\langle \text{is-K2-isometry } J \rangle$   
**have** *z-non-zero* (*apply-cltn2*  $p J$ ) **by** (*rule is-K2-isometry-z-non-zero*)

**from**  $\langle p \in \text{hyp2} \cup S \rangle$  **and**  $\langle \text{is-K2-isometry } J \rangle$

**and** *cart2-append1-apply-cltn2*

**obtain**  $k$  **where**  $k \neq 0$

**and** *cart2-append1*  $p \ v * \text{cltn2-rep } J = k *_{\mathbb{R}} \text{cart2-append1 } (\text{apply-cltn2 } p J)$   
**by** *auto*

**from**  $\langle \text{cart2-append1 } p \ v * \text{cltn2-rep } J = k *_{\mathbb{R}} \text{cart2-append1 } (\text{apply-cltn2 } p J) \rangle$

**and**  $\langle \text{z-non-zero } (\text{apply-cltn2 } p J) \rangle$

**have** *expansion-factor*  $p J = k$

**by** (*unfold expansion-factor-def*) (*simp add: cart2-append1-z*)

**with**  $\langle k \neq 0 \rangle$

**and** *cart2-append1*  $p \ v * \text{cltn2-rep } J = k *_{\mathbb{R}} \text{cart2-append1 } (\text{apply-cltn2 } p J)$

**show** *expansion-factor*  $p J \neq 0$

**and** *cart2-append1*  $p \ v * \text{cltn2-rep } J$

$= \text{expansion-factor } p J *_{\mathbb{R}} \text{cart2-append1 } (\text{apply-cltn2 } p J)$

**by** *simp-all*

**qed**

**lemma** *expansion-factor-linear-apply-cltn2*:

**assumes**  $p \in \text{hyp2} \cup S$  **and**  $q \in \text{hyp2} \cup S$  **and**  $r \in \text{hyp2} \cup S$

**and** *is-K2-isometry*  $J$

**and** *cart2-pt*  $r = k *_{\mathbb{R}} \text{cart2-pt } p + (1 - k) *_{\mathbb{R}} \text{cart2-pt } q$

**shows**  $\text{expansion-factor } r \ J \ *_{\mathbb{R}} \ \text{cart2-append1} \ (\text{apply-cltn2 } r \ J)$   
 $= (k \ * \ \text{expansion-factor } p \ J) \ *_{\mathbb{R}} \ \text{cart2-append1} \ (\text{apply-cltn2 } p \ J)$   
 $+ ((1 - k) \ * \ \text{expansion-factor } q \ J) \ *_{\mathbb{R}} \ \text{cart2-append1} \ (\text{apply-cltn2 } q \ J)$   
**(is**  $?er \ *_{\mathbb{R}} \ - = (k \ * \ ?ep) \ *_{\mathbb{R}} \ - + ((1 - k) \ * \ ?eq) \ *_{\mathbb{R}} \ -$

**proof** –

**let**  $?cp = \text{cart2-pt } p$   
**let**  $?cq = \text{cart2-pt } q$   
**let**  $?cr = \text{cart2-pt } r$   
**let**  $?cp1 = \text{cart2-append1 } p$   
**let**  $?cq1 = \text{cart2-append1 } q$   
**let**  $?cr1 = \text{cart2-append1 } r$   
**let**  $?repJ = \text{cltn2-rep } J$   
**from**  $\langle p \in \text{hyp2} \cup S \rangle$  **and**  $\langle q \in \text{hyp2} \cup S \rangle$  **and**  $\langle r \in \text{hyp2} \cup S \rangle$   
**have**  $z\text{-non-zero } p$  **and**  $z\text{-non-zero } q$  **and**  $z\text{-non-zero } r$   
**by** (*simp-all add: hyp2-S-z-non-zero*)

**from**  $\langle ?cr = k \ *_{\mathbb{R}} \ ?cp + (1 - k) \ *_{\mathbb{R}} \ ?cq \rangle$   
**have**  $\text{vector2-append1 } ?cr$   
 $= k \ *_{\mathbb{R}} \ \text{vector2-append1 } ?cp + (1 - k) \ *_{\mathbb{R}} \ \text{vector2-append1 } ?cq$   
**by** (*unfold vector2-append1-def vector-def*) (*simp add: vec-eq-iff*)  
**with**  $\langle z\text{-non-zero } p \rangle$  **and**  $\langle z\text{-non-zero } q \rangle$  **and**  $\langle z\text{-non-zero } r \rangle$   
**have**  $?cr1 = k \ *_{\mathbb{R}} \ ?cp1 + (1 - k) \ *_{\mathbb{R}} \ ?cq1$  **by** (*simp add: cart2-append1*)  
**hence**  $?cr1 \ v* \ ?repJ = k \ *_{\mathbb{R}} \ (?cp1 \ v* \ ?repJ) + (1 - k) \ *_{\mathbb{R}} \ (?cq1 \ v* \ ?repJ)$   
**by** (*simp add: vector-matrix-left-distrib scaleR-vector-matrix-assoc [symmetric]*)  
**with**  $\langle p \in \text{hyp2} \cup S \rangle$  **and**  $\langle q \in \text{hyp2} \cup S \rangle$  **and**  $\langle r \in \text{hyp2} \cup S \rangle$   
**and**  $\langle \text{is-K2-isometry } J \rangle$   
**show**  $?er \ *_{\mathbb{R}} \ \text{cart2-append1} \ (\text{apply-cltn2 } r \ J)$   
 $= (k \ * \ ?ep) \ *_{\mathbb{R}} \ \text{cart2-append1} \ (\text{apply-cltn2 } p \ J)$   
 $+ ((1 - k) \ * \ ?eq) \ *_{\mathbb{R}} \ \text{cart2-append1} \ (\text{apply-cltn2 } q \ J)$   
**by** (*simp add: expansion-factor*)

**qed**

**lemma** *expansion-factor-linear*:

**assumes**  $p \in \text{hyp2} \cup S$  **and**  $q \in \text{hyp2} \cup S$  **and**  $r \in \text{hyp2} \cup S$   
**and**  $\text{is-K2-isometry } J$   
**and**  $\text{cart2-pt } r = k \ *_{\mathbb{R}} \ \text{cart2-pt } p + (1 - k) \ *_{\mathbb{R}} \ \text{cart2-pt } q$   
**shows**  $\text{expansion-factor } r \ J$   
 $= k \ * \ \text{expansion-factor } p \ J + (1 - k) \ * \ \text{expansion-factor } q \ J$   
**(is**  $?er = k \ * \ ?ep + (1 - k) \ * \ ?eq$

**proof** –

**from**  $\langle p \in \text{hyp2} \cup S \rangle$  **and**  $\langle q \in \text{hyp2} \cup S \rangle$  **and**  $\langle r \in \text{hyp2} \cup S \rangle$   
**and**  $\langle \text{is-K2-isometry } J \rangle$   
**have**  $z\text{-non-zero} \ (\text{apply-cltn2 } p \ J)$   
**and**  $z\text{-non-zero} \ (\text{apply-cltn2 } q \ J)$   
**and**  $z\text{-non-zero} \ (\text{apply-cltn2 } r \ J)$   
**by** (*simp-all add: is-K2-isometry-z-non-zero*)

**from**  $\langle p \in \text{hyp2} \cup S \rangle$  **and**  $\langle q \in \text{hyp2} \cup S \rangle$  **and**  $\langle r \in \text{hyp2} \cup S \rangle$   
**and**  $\langle \text{is-K2-isometry } J \rangle$

**and**  $\langle \text{cart2-pt } r = k *_R \text{ cart2-pt } p + (1 - k) *_R \text{ cart2-pt } q \rangle$   
**have**  $\langle ?er *_R \text{ cart2-append1 } (\text{apply-cltn2 } r J) \rangle$   
 $= (k *_R ?ep) *_R \text{ cart2-append1 } (\text{apply-cltn2 } p J)$   
 $+ ((1 - k) *_R ?eq) *_R \text{ cart2-append1 } (\text{apply-cltn2 } q J)$   
**by**  $(\text{rule expansion-factor-linear-apply-cltn2})$   
**hence**  $(?er *_R \text{ cart2-append1 } (\text{apply-cltn2 } r J))\$3$   
 $= ((k *_R ?ep) *_R \text{ cart2-append1 } (\text{apply-cltn2 } p J)$   
 $+ ((1 - k) *_R ?eq) *_R \text{ cart2-append1 } (\text{apply-cltn2 } q J))\$3$   
**by**  $\text{simp}$   
**with**  $\langle z\text{-non-zero } (\text{apply-cltn2 } p J) \rangle$   
**and**  $\langle z\text{-non-zero } (\text{apply-cltn2 } q J) \rangle$   
**and**  $\langle z\text{-non-zero } (\text{apply-cltn2 } r J) \rangle$   
**show**  $?er = k *_R ?ep + (1 - k) *_R ?eq$  **by**  $(\text{simp add: cart2-append1-z})$   
**qed**

**lemma** *expansion-factor-sgn-invariant*:

**assumes**  $p \in \text{hyp2} \cup S$  **and**  $q \in \text{hyp2} \cup S$  **and** *is-K2-isometry*  $J$   
**shows**  $\text{sgn } (\text{expansion-factor } p J) = \text{sgn } (\text{expansion-factor } q J)$   
**(is**  $\text{sgn } ?ep = \text{sgn } ?eq$ **)**

**proof**  $(\text{rule ccontr})$

**assume**  $\text{sgn } ?ep \neq \text{sgn } ?eq$

**from**  $\langle p \in \text{hyp2} \cup S \rangle$  **and**  $\langle q \in \text{hyp2} \cup S \rangle$  **and**  $\langle \text{is-K2-isometry } J \rangle$

**have**  $?ep \neq 0$  **and**  $?eq \neq 0$  **by**  $(\text{simp-all add: expansion-factor})$

**hence**  $\text{sgn } ?ep \in \{-1, 1\}$  **and**  $\text{sgn } ?eq \in \{-1, 1\}$

**by**  $(\text{simp-all add: sgn-real-def})$

**with**  $\langle \text{sgn } ?ep \neq \text{sgn } ?eq \rangle$  **have**  $\text{sgn } ?ep = -\text{sgn } ?eq$  **by** *auto*

**hence**  $\text{sgn } ?ep = \text{sgn } (-?eq)$  **by**  $(\text{subst sgn-minus})$

**with** *sgn-plus*  $[\text{of } ?ep - ?eq]$

**have**  $\text{sgn } (?ep - ?eq) = \text{sgn } ?ep$  **by**  $(\text{simp add: algebra-simps})$

**with**  $\langle \text{sgn } ?ep \in \{-1, 1\} \rangle$  **have**  $?ep - ?eq \neq 0$  **by**  $(\text{auto simp add: sgn-real-def})$

**let**  $?k = -?eq / (?ep - ?eq)$

**from**  $\langle \text{sgn } (?ep - ?eq) = \text{sgn } ?ep \rangle$  **and**  $\langle \text{sgn } ?ep = \text{sgn } (-?eq) \rangle$

**have**  $\text{sgn } (?ep - ?eq) = \text{sgn } (-?eq)$  **by** *simp*

**with**  $\langle ?ep - ?eq \neq 0 \rangle$  **and** *sgn-div*  $[\text{of } ?ep - ?eq - ?eq]$

**have**  $?k > 0$  **by** *simp*

**from**  $\langle ?ep - ?eq \neq 0 \rangle$

**have**  $1 - ?k = ?ep / (?ep - ?eq)$  **by**  $(\text{simp add: field-simps})$

**with**  $\langle \text{sgn } (?ep - ?eq) = \text{sgn } ?ep \rangle$  **and**  $\langle ?ep - ?eq \neq 0 \rangle$

**have**  $1 - ?k > 0$  **by**  $(\text{simp add: sgn-div})$

**hence**  $?k < 1$  **by** *simp*

**let**  $?cp = \text{cart2-pt } p$

**let**  $?cq = \text{cart2-pt } q$

**let**  $?cr = ?k *_R ?cp + (1 - ?k) *_R ?cq$

**let**  $?r = \text{proj2-pt } ?cr$

**let**  $?er = \text{expansion-factor } ?r J$

**have**  $\text{cart2-pt } ?r = ?cr$  **by** (rule  $\text{cart2-proj2}$ )  
**from**  $\langle p \in \text{hyp2} \cup S \rangle$  **and**  $\langle q \in \text{hyp2} \cup S \rangle$  **and**  $\langle ?k > 0 \rangle$  **and**  $\langle ?k < 1 \rangle$   
**and**  $\text{between-hyp2-S [of } q \text{ } p \text{ } ?k]$   
**have**  $?r \in \text{hyp2} \cup S$  **by**  $\text{simp}$   
**with**  $\langle p \in \text{hyp2} \cup S \rangle$  **and**  $\langle q \in \text{hyp2} \cup S \rangle$  **and**  $\langle \text{is-K2-isometry } J \rangle$   
**and**  $\langle \text{cart2-pt } ?r = ?cr \rangle$   
**and**  $\text{expansion-factor-linear [of } p \text{ } q \text{ } ?r \text{ } J \text{ } ?k]$   
**have**  $?er = ?k * ?ep + (1 - ?k) * ?eq$  **by**  $\text{simp}$   
**with**  $\langle ?ep - ?eq \neq 0 \rangle$  **have**  $?er = 0$  **by** (simp add:  $\text{field-simps}$ )  
**with**  $\langle ?r \in \text{hyp2} \cup S \rangle$  **and**  $\langle \text{is-K2-isometry } J \rangle$   
**show**  $\text{False}$  **by** (simp add:  $\text{expansion-factor}$ )  
**qed**

**lemma**  $\text{statement-63}$ :

**assumes**  $p \in \text{hyp2} \cup S$  **and**  $q \in \text{hyp2} \cup S$  **and**  $r \in \text{hyp2} \cup S$   
**and**  $\text{is-K2-isometry } J$  **and**  $B_{\mathbb{R}}$  ( $\text{cart2-pt } p$ ) ( $\text{cart2-pt } q$ ) ( $\text{cart2-pt } r$ )  
**shows**  $B_{\mathbb{R}}$   
 $(\text{cart2-pt } (\text{apply-cltn2 } p \text{ } J))$   
 $(\text{cart2-pt } (\text{apply-cltn2 } q \text{ } J))$   
 $(\text{cart2-pt } (\text{apply-cltn2 } r \text{ } J))$

**proof** –

**let**  $?cp = \text{cart2-pt } p$   
**let**  $?cq = \text{cart2-pt } q$   
**let**  $?cr = \text{cart2-pt } r$   
**let**  $?ep = \text{expansion-factor } p \text{ } J$   
**let**  $?eq = \text{expansion-factor } q \text{ } J$   
**let**  $?er = \text{expansion-factor } r \text{ } J$   
**from**  $\langle q \in \text{hyp2} \cup S \rangle$  **and**  $\langle \text{is-K2-isometry } J \rangle$   
**have**  $?eq \neq 0$  **by** (rule  $\text{expansion-factor}$ )

**from**  $\langle p \in \text{hyp2} \cup S \rangle$  **and**  $\langle q \in \text{hyp2} \cup S \rangle$  **and**  $\langle r \in \text{hyp2} \cup S \rangle$   
**and**  $\langle \text{is-K2-isometry } J \rangle$  **and**  $\text{expansion-factor-sgn-invariant}$   
**have**  $\text{sgn } ?ep = \text{sgn } ?eq$  **and**  $\text{sgn } ?er = \text{sgn } ?eq$  **by**  $\text{fast+}$   
**with**  $\langle ?eq \neq 0 \rangle$   
**have**  $?ep / ?eq > 0$  **and**  $?er / ?eq > 0$  **by** (simp-all add:  $\text{sgn-div}$ )

**from**  $\langle B_{\mathbb{R}} \text{ } ?cp \text{ } ?cq \text{ } ?cr \rangle$   
**obtain**  $k$  **where**  $k \geq 0$  **and**  $k \leq 1$  **and**  $?cq = k *_{\mathbb{R}} ?cr + (1 - k) *_{\mathbb{R}} ?cp$   
**by** (unfold  $\text{real-euclid-B-def}$ ) (auto simp add:  $\text{algebra-simps}$ )

**let**  $?c = k * ?er / ?eq$   
**from**  $\langle k \geq 0 \rangle$  **and**  $\langle ?er / ?eq > 0 \rangle$  **and**  $\text{mult-nonneg-nonneg [of } k \text{ } ?er / ?eq]$   
**have**  $?c \geq 0$  **by**  $\text{simp}$

**from**  $\langle r \in \text{hyp2} \cup S \rangle$  **and**  $\langle p \in \text{hyp2} \cup S \rangle$  **and**  $\langle q \in \text{hyp2} \cup S \rangle$   
**and**  $\langle \text{is-K2-isometry } J \rangle$  **and**  $\langle ?cq = k *_{\mathbb{R}} ?cr + (1 - k) *_{\mathbb{R}} ?cp \rangle$   
**have**  $?eq = k * ?er + (1 - k) * ?ep$  **by** (rule  $\text{expansion-factor-linear}$ )  
**with**  $\langle ?eq \neq 0 \rangle$  **have**  $1 - ?c = (1 - k) * ?ep / ?eq$  **by** (simp add:  $\text{field-simps}$ )

**with**  $\langle k \leq 1 \rangle$  **and**  $\langle ?ep / ?eq > 0 \rangle$   
**and** *mult-nonneg-nonneg* [of  $1 - k$   $?ep / ?eq$ ]  
**have**  $?c \leq 1$  **by** *simp*

**let**  $?pJ = \text{apply-cltn2 } p \ J$   
**let**  $?qJ = \text{apply-cltn2 } q \ J$   
**let**  $?rJ = \text{apply-cltn2 } r \ J$   
**let**  $?cpJ = \text{cart2-pt } ?pJ$   
**let**  $?cqJ = \text{cart2-pt } ?qJ$   
**let**  $?crJ = \text{cart2-pt } ?rJ$   
**let**  $?cpJ1 = \text{cart2-append1 } ?pJ$   
**let**  $?cqJ1 = \text{cart2-append1 } ?qJ$   
**let**  $?crJ1 = \text{cart2-append1 } ?rJ$   
**from**  $\langle p \in \text{hyp2} \cup S \rangle$  **and**  $\langle q \in \text{hyp2} \cup S \rangle$  **and**  $\langle r \in \text{hyp2} \cup S \rangle$   
**and**  $\langle \text{is-K2-isometry } J \rangle$   
**have** *z-non-zero*  $?pJ$  **and** *z-non-zero*  $?qJ$  **and** *z-non-zero*  $?rJ$   
**by** (*simp-all add: is-K2-isometry-z-non-zero*)

**from**  $\langle r \in \text{hyp2} \cup S \rangle$  **and**  $\langle p \in \text{hyp2} \cup S \rangle$  **and**  $\langle q \in \text{hyp2} \cup S \rangle$   
**and**  $\langle \text{is-K2-isometry } J \rangle$  **and**  $\langle ?cq = k *_{\mathbb{R}} ?cr + (1 - k) *_{\mathbb{R}} ?cp \rangle$   
**have**  $?eq *_{\mathbb{R}} ?cqJ1 = (k * ?er) *_{\mathbb{R}} ?crJ1 + ((1 - k) * ?ep) *_{\mathbb{R}} ?cpJ1$   
**by** (*rule expansion-factor-linear-apply-cltn2*)  
**hence**  $(1 / ?eq) *_{\mathbb{R}} (?eq *_{\mathbb{R}} ?cqJ1)$   
 $= (1 / ?eq) *_{\mathbb{R}} ((k * ?er) *_{\mathbb{R}} ?crJ1 + ((1 - k) * ?ep) *_{\mathbb{R}} ?cpJ1)$  **by** *simp*  
**with**  $\langle 1 - ?c = (1 - k) * ?ep / ?eq \rangle$  **and**  $\langle ?eq \neq 0 \rangle$   
**have**  $?cqJ1 = ?c *_{\mathbb{R}} ?crJ1 + (1 - ?c) *_{\mathbb{R}} ?cpJ1$   
**by** (*simp add: scaleR-right-distrib*)  
**with**  $\langle \text{z-non-zero } ?pJ \rangle$  **and**  $\langle \text{z-non-zero } ?qJ \rangle$  **and**  $\langle \text{z-non-zero } ?rJ \rangle$   
**have** *vector2-append1*  $?cqJ$   
 $= ?c *_{\mathbb{R}} \text{vector2-append1 } ?crJ + (1 - ?c) *_{\mathbb{R}} \text{vector2-append1 } ?cpJ$   
**by** (*simp add: cart2-append1*)  
**hence**  $?cqJ = ?c *_{\mathbb{R}} ?crJ + (1 - ?c) *_{\mathbb{R}} ?cpJ$   
**unfolding** *vector2-append1-def* **and** *vector-def*  
**by** (*simp add: vec-eq-iff forall-2 forall-3*)  
**with**  $\langle ?c \geq 0 \rangle$  **and**  $\langle ?c \leq 1 \rangle$   
**show**  $B_{\mathbb{R}} ?cpJ ?cqJ ?crJ$   
**by** (*unfold real-euclid-B-def*) (*simp add: algebra-simps exI [of - ?c]*)

qed

**theorem** *hyp2-axiom4*:  $\forall q a b c. \exists x. B_K q a x \wedge a x \equiv_K b c$

**proof** (*rule allI*)+

**fix**  $q a b c :: \text{hyp2}$

**let**  $?pq = \text{Rep-hyp2 } q$

**let**  $?pa = \text{Rep-hyp2 } a$

**let**  $?pb = \text{Rep-hyp2 } b$

**let**  $?pc = \text{Rep-hyp2 } c$

**have**  $?pq \in \text{hyp2}$  **and**  $?pa \in \text{hyp2}$  **and**  $?pb \in \text{hyp2}$  **and**  $?pc \in \text{hyp2}$

**by** (*rule Rep-hyp2*)+

**let**  $?cq = \text{cart2-pt } ?pq$

**let**  $?ca = \text{cart2-pt } ?pa$   
**let**  $?cb = \text{cart2-pt } ?pb$   
**let**  $?cc = \text{cart2-pt } ?pc$   
**let**  $?pp = \epsilon p. p \in S \wedge B_{\mathbf{R}} ?cb ?cc (\text{cart2-pt } p)$   
**let**  $?cp = \text{cart2-pt } ?pp$   
**from**  $\langle ?pb \in \text{hyp2} \rangle$  **and**  $\langle ?pc \in \text{hyp2} \rangle$  **and**  $\text{extend-to-}S$  [of  $?pb ?pc$ ]  
**and**  $\text{someI-ex}$  [of  $\lambda p. p \in S \wedge B_{\mathbf{R}} ?cb ?cc (\text{cart2-pt } p)$ ]  
**have**  $?pp \in S$  **and**  $B_{\mathbf{R}} ?cb ?cc ?cp$  **by** *auto*

**let**  $?pr = \epsilon r. r \in S \wedge B_{\mathbf{R}} ?cq ?ca (\text{cart2-pt } r)$   
**let**  $?cr = \text{cart2-pt } ?pr$   
**from**  $\langle ?pq \in \text{hyp2} \rangle$  **and**  $\langle ?pa \in \text{hyp2} \rangle$  **and**  $\text{extend-to-}S$  [of  $?pq ?pa$ ]  
**and**  $\text{someI-ex}$  [of  $\lambda r. r \in S \wedge B_{\mathbf{R}} ?cq ?ca (\text{cart2-pt } r)$ ]  
**have**  $?pr \in S$  **and**  $B_{\mathbf{R}} ?cq ?ca ?cr$  **by** *auto*

**from**  $\langle ?pb \in \text{hyp2} \rangle$  **and**  $\langle ?pa \in \text{hyp2} \rangle$  **and**  $\langle ?pp \in S \rangle$  **and**  $\langle ?pr \in S \rangle$   
**and**  $\text{statement66-existence}$  [of  $?pb ?pa ?pp ?pr$ ]  
**obtain**  $J$  **where**  $\text{is-}K2\text{-isometry } J$   
**and**  $\text{apply-cltn2 } ?pb J = ?pa$  **and**  $\text{apply-cltn2 } ?pp J = ?pr$   
**by** *auto*

**let**  $?px = \text{apply-cltn2 } ?pc J$   
**let**  $?cx = \text{cart2-pt } ?px$   
**let**  $?x = \text{Abs-hyp2 } ?px$   
**from**  $\langle \text{is-}K2\text{-isometry } J \rangle$  **and**  $\langle ?pc \in \text{hyp2} \rangle$   
**have**  $?px \in \text{hyp2}$  **by** (*rule statement60-one-way*)  
**hence**  $\text{Rep-hyp2 } ?x = ?px$  **by** (*rule Abs-hyp2-inverse*)

**from**  $\langle ?pb \in \text{hyp2} \rangle$  **and**  $\langle ?pc \in \text{hyp2} \rangle$  **and**  $\langle ?pp \in S \rangle$  **and**  $\langle \text{is-}K2\text{-isometry } J \rangle$   
**and**  $\langle B_{\mathbf{R}} ?cb ?cc ?cp \rangle$  **and**  $\text{statement-63}$   
**have**  $B_{\mathbf{R}} (\text{cart2-pt } (\text{apply-cltn2 } ?pb J)) ?cx (\text{cart2-pt } (\text{apply-cltn2 } ?pp J))$   
**by** *simp*

**with**  $\langle \text{apply-cltn2 } ?pb J = ?pa \rangle$  **and**  $\langle \text{apply-cltn2 } ?pp J = ?pr \rangle$   
**have**  $B_{\mathbf{R}} ?ca ?cx ?cr$  **by** *simp*

**with**  $\langle B_{\mathbf{R}} ?cq ?ca ?cr \rangle$  **have**  $B_{\mathbf{R}} ?cq ?ca ?cx$  **by** (*rule real-euclid.th3-5-1*)  
**with**  $\langle \text{Rep-hyp2 } ?x = ?px \rangle$   
**have**  $B_K q a ?x$   
**unfolding**  $\text{real-hyp2-B-def}$  **and**  $\text{hyp2-rep-def}$   
**by** *simp*

**have**  $\text{Abs-hyp2 } ?pa = a$  **by** (*rule Rep-hyp2-inverse*)  
**with**  $\langle \text{apply-cltn2 } ?pb J = ?pa \rangle$   
**have**  $\text{hyp2-cltn2 } b J = a$  **by** (*unfold hyp2-cltn2-def*) *simp*

**have**  $\text{hyp2-cltn2 } c J = ?x$  **unfolding**  $\text{hyp2-cltn2-def}$  **..**  
**with**  $\langle \text{is-}K2\text{-isometry } J \rangle$  **and**  $\langle \text{hyp2-cltn2 } b J = a \rangle$   
**have**  $b c \equiv_K a ?x$   
**by** (*unfold real-hyp2-C-def*) (*simp add: exI [of - J]*)  
**hence**  $a ?x \equiv_K b c$  **by** (*rule hyp2.th2-2*)  
**with**  $\langle B_K q a ?x \rangle$

**show**  $\exists x. B_K q a x \wedge a x \equiv_K b c$  **by** (*simp add: exI [of - ?x]*)  
**qed**

## 8.9 More betweenness theorems

**lemma** *hyp2-S-points-fix-line*:

**assumes**  $a \in \text{hyp2}$  **and**  $p \in S$  **and** *is-K2-isometry*  $J$   
**and** *apply-cltn2*  $a J = a$  (**is**  $?aJ = a$ )  
**and** *apply-cltn2*  $p J = p$  (**is**  $?pJ = p$ )  
**and** *proj2-incident*  $a l$  **and** *proj2-incident*  $p l$  **and** *proj2-incident*  $b l$   
**shows** *apply-cltn2*  $b J = b$  (**is**  $?bJ = b$ )

**proof** –

**let**  $?lJ = \text{apply-cltn2-line } l J$   
**from**  $\langle \text{proj2-incident } a l \rangle$  **and**  $\langle \text{proj2-incident } p l \rangle$   
**have** *proj2-incident*  $?aJ ?lJ$  **and** *proj2-incident*  $?pJ ?lJ$  **by** *simp-all*  
**with**  $\langle ?aJ = a \rangle$  **and**  $\langle ?pJ = p \rangle$   
**have** *proj2-incident*  $a ?lJ$  **and** *proj2-incident*  $p ?lJ$  **by** *simp-all*

**from**  $\langle a \in \text{hyp2} \rangle$   $\langle \text{proj2-incident } a l \rangle$  **and** *line-through-K2-intersect-S-again* [*of*  
 $a l$ ]

**obtain**  $q$  **where**  $q \neq p$  **and**  $q \in S$  **and** *proj2-incident*  $q l$  **by** *auto*  
**let**  $?qJ = \text{apply-cltn2 } q J$

**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$   
**have**  $a \neq p$  **and**  $a \neq q$  **by** (*simp-all add: hyp2-S-not-equal*)

**from**  $\langle a \neq p \rangle$  **and**  $\langle \text{proj2-incident } a l \rangle$  **and**  $\langle \text{proj2-incident } p l \rangle$   
**and**  $\langle \text{proj2-incident } a ?lJ \rangle$  **and**  $\langle \text{proj2-incident } p ?lJ \rangle$   
**and** *proj2-incident-unique*  
**have**  $?lJ = l$  **by** *auto*

**from**  $\langle \text{proj2-incident } q l \rangle$  **have** *proj2-incident*  $?qJ ?lJ$  **by** *simp*  
**with**  $\langle ?lJ = l \rangle$  **have** *proj2-incident*  $?qJ l$  **by** *simp*

**from**  $\langle q \in S \rangle$  **and**  $\langle \text{is-K2-isometry } J \rangle$   
**have**  $?qJ \in S$  **by** (*unfold is-K2-isometry-def*) *simp*  
**with**  $\langle q \neq p \rangle$  **and**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$  **and**  $\langle \text{proj2-incident } p l \rangle$   
**and**  $\langle \text{proj2-incident } q l \rangle$  **and**  $\langle \text{proj2-incident } ?qJ l \rangle$   
**and** *line-S-two-intersections-only*  
**have**  $?qJ = p \vee ?qJ = q$  **by** *simp*

**have**  $?qJ = q$

**proof** (*rule ccontr*)

**assume**  $?qJ \neq q$

**with**  $\langle ?qJ = p \vee ?qJ = q \rangle$  **have**  $?qJ = p$  **by** *simp*

**with**  $\langle ?pJ = p \rangle$  **have**  $?qJ = ?pJ$  **by** *simp*

**with** *apply-cltn2-injective* **have**  $q = p$  **by** *fast*

**with**  $\langle q \neq p \rangle$  **show** *False ..*

**qed**

**with**  $\langle q \neq p \rangle$  **and**  $\langle a \neq p \rangle$  **and**  $\langle a \neq q \rangle$  **and**  $\langle \text{proj2-incident } p \ l \rangle$   
**and**  $\langle \text{proj2-incident } q \ l \rangle$  **and**  $\langle \text{proj2-incident } a \ l \rangle$   
**and**  $\langle ?pJ = p \rangle$  **and**  $\langle ?aJ = a \rangle$  **and**  $\langle \text{proj2-incident } b \ l \rangle$   
**and**  $\text{cltn2-three-point-line [of } p \ q \ a \ l \ J \ b \ ]$   
**show**  $?bJ = b$  **by** *simp*  
**qed**

**lemma** *K2-isometry-endpoint-in-S:*

**assumes**  $a \neq b$  **and**  $a \in \text{hyp2} \cup S$  **and**  $b \in \text{hyp2} \cup S$  **and** *is-K2-isometry*  $J$   
**shows**  $\text{apply-cltn2 (endpoint-in-S } a \ b) \ J$   
 $= \text{endpoint-in-S (apply-cltn2 } a \ J) (apply-cltn2 \ b \ J)$   
**(is**  $?pJ = \text{endpoint-in-S } ?aJ \ ?bJ)$

**proof** –

**let**  $?p = \text{endpoint-in-S } a \ b$

**from**  $\langle a \neq b \rangle$  **and** *apply-cltn2-injective* **have**  $?aJ \neq ?bJ$  **by** *fast*

**from**  $\langle a \in \text{hyp2} \cup S \rangle$  **and**  $\langle b \in \text{hyp2} \cup S \rangle$  **and**  $\langle \text{is-K2-isometry } J \rangle$   
**and** *is-K2-isometry-hyp2-S*  
**have**  $?aJ \in \text{hyp2} \cup S$  **and**  $?bJ \in \text{hyp2} \cup S$  **by** *simp-all*

**let**  $?ca = \text{cart2-pt } a$

**let**  $?cb = \text{cart2-pt } b$

**let**  $?cp = \text{cart2-pt } ?p$

**from**  $\langle a \in \text{hyp2} \cup S \rangle$  **and**  $\langle b \in \text{hyp2} \cup S \rangle$

**have**  $?p \in S$  **and**  $B_{\mathbb{R}} \ ?ca \ ?cb \ ?cp$  **by**  $(\text{rule endpoint-in-S})+$

**from**  $\langle ?p \in S \rangle$  **and**  $\langle \text{is-K2-isometry } J \rangle$

**have**  $?pJ \in S$  **by**  $(\text{unfold is-K2-isometry-def}) \text{ simp}$

**let**  $?caJ = \text{cart2-pt } ?aJ$

**let**  $?cbJ = \text{cart2-pt } ?bJ$

**let**  $?cpJ = \text{cart2-pt } ?pJ$

**from**  $\langle a \in \text{hyp2} \cup S \rangle$  **and**  $\langle b \in \text{hyp2} \cup S \rangle$  **and**  $\langle ?p \in S \rangle$  **and**  $\langle \text{is-K2-isometry } J \rangle$

**and**  $\langle B_{\mathbb{R}} \ ?ca \ ?cb \ ?cp \rangle$  **and** *statement-63*

**have**  $B_{\mathbb{R}} \ ?caJ \ ?cbJ \ ?cpJ$  **by** *simp*

**with**  $\langle ?aJ \neq ?bJ \rangle$  **and**  $\langle ?aJ \in \text{hyp2} \cup S \rangle$  **and**  $\langle ?bJ \in \text{hyp2} \cup S \rangle$  **and**  $\langle ?pJ \in S \rangle$

**show**  $?pJ = \text{endpoint-in-S } ?aJ \ ?bJ$  **by**  $(\text{rule endpoint-in-S-unique})$

**qed**

**lemma** *between-endpoint-in-S:*

**assumes**  $a \neq b$  **and**  $b \neq c$

**and**  $a \in \text{hyp2} \cup S$  **and**  $b \in \text{hyp2} \cup S$  **and**  $c \in \text{hyp2} \cup S$

**and**  $B_{\mathbb{R}} \ (\text{cart2-pt } a) \ (\text{cart2-pt } b) \ (\text{cart2-pt } c)$  **(is**  $B_{\mathbb{R}} \ ?ca \ ?cb \ ?cc)$

**shows**  $\text{endpoint-in-S } a \ b = \text{endpoint-in-S } b \ c$  **(is**  $?p = ?q)$

**proof** –

**from**  $\langle b \neq c \rangle$  **and**  $\langle b \in \text{hyp2} \cup S \rangle$  **and**  $\langle c \in \text{hyp2} \cup S \rangle$  **and** *hyp2-S-cart2-inj*

**have**  $?cb \neq ?cc$  **by** *auto*

**let**  $?cq = \text{cart2-pt } ?q$   
**from**  $\langle b \in \text{hyp2} \cup S \rangle$  **and**  $\langle c \in \text{hyp2} \cup S \rangle$   
**have**  $?q \in S$  **and**  $B_{\mathbf{R}} ?cb ?cc ?cq$  **by** (*rule endpoint-in-S*)+

**from**  $\langle ?cb \neq ?cc \rangle$  **and**  $\langle B_{\mathbf{R}} ?ca ?cb ?cc \rangle$  **and**  $\langle B_{\mathbf{R}} ?cb ?cc ?cq \rangle$   
**have**  $B_{\mathbf{R}} ?ca ?cb ?cq$  **by** (*rule real-euclid.th3-7-2*)  
**with**  $\langle a \neq b \rangle$  **and**  $\langle a \in \text{hyp2} \cup S \rangle$  **and**  $\langle b \in \text{hyp2} \cup S \rangle$  **and**  $\langle ?q \in S \rangle$   
**have**  $?q = ?p$  **by** (*rule endpoint-in-S-unique*)  
**thus**  $?p = ?q$  ..

**qed**

**lemma** *hyp2-extend-segment-unique*:  
**assumes**  $a \neq b$  **and**  $B_K a b c$  **and**  $B_K a b d$  **and**  $b c \equiv_K b d$   
**shows**  $c = d$

**proof** *cases*  
**assume**  $b = c$   
**with**  $\langle b c \equiv_K b d \rangle$  **show**  $c = d$  **by** (*simp add: hyp2.A3-reversed*)

**next**  
**assume**  $b \neq c$

**have**  $b \neq d$   
**proof** (*rule ccontr*)  
**assume**  $\neg b \neq d$   
**hence**  $b = d$  **by** *simp*  
**with**  $\langle b c \equiv_K b d \rangle$  **have**  $b c \equiv_K b b$  **by** *simp*  
**hence**  $b = c$  **by** (*rule hyp2.A3'*)  
**with**  $\langle b \neq c \rangle$  **show** *False* ..

**qed**

**with**  $\langle a \neq b \rangle$  **and**  $\langle b \neq c \rangle$   
**have**  $\text{Rep-hyp2 } a \neq \text{Rep-hyp2 } b$  (**is**  $?pa \neq ?pb$ )  
**and**  $\text{Rep-hyp2 } b \neq \text{Rep-hyp2 } c$  (**is**  $?pb \neq ?pc$ )  
**and**  $\text{Rep-hyp2 } b \neq \text{Rep-hyp2 } d$  (**is**  $?pb \neq ?pd$ )  
**by** (*simp-all add: Rep-hyp2-inject*)

**have**  $?pa \in \text{hyp2}$  **and**  $?pb \in \text{hyp2}$  **and**  $?pc \in \text{hyp2}$  **and**  $?pd \in \text{hyp2}$   
**by** (*rule Rep-hyp2*)+

**let**  $?pp = \text{endpoint-in-S } ?pb ?pc$   
**let**  $?ca = \text{cart2-pt } ?pa$   
**let**  $?cb = \text{cart2-pt } ?pb$   
**let**  $?cc = \text{cart2-pt } ?pc$   
**let**  $?cd = \text{cart2-pt } ?pd$   
**let**  $?cp = \text{cart2-pt } ?pp$   
**from**  $\langle ?pb \in \text{hyp2} \rangle$  **and**  $\langle ?pc \in \text{hyp2} \rangle$   
**have**  $?pp \in S$  **and**  $B_{\mathbf{R}} ?cb ?cc ?cp$  **by** (*simp-all add: endpoint-in-S*)

**from**  $\langle b c \equiv_K b d \rangle$

**obtain**  $J$  **where**  $is\text{-}K2\text{-isometry } J$   
**and**  $hyp2\text{-cltn2 } b J = b$  **and**  $hyp2\text{-cltn2 } c J = d$   
**by** ( $unfold\ real\text{-}hyp2\text{-}C\text{-}def$ )  $auto$

**from**  $\langle hyp2\text{-cltn2 } b J = b \rangle$  **and**  $\langle hyp2\text{-cltn2 } c J = d \rangle$   
**have**  $Rep\text{-}hyp2 (hyp2\text{-cltn2 } b J) = ?pb$   
**and**  $Rep\text{-}hyp2 (hyp2\text{-cltn2 } c J) = ?pd$   
**by**  $simp\text{-}all$   
**with**  $\langle is\text{-}K2\text{-isometry } J \rangle$   
**have**  $apply\text{-}cltn2 ?pb J = ?pb$  **and**  $apply\text{-}cltn2 ?pc J = ?pd$   
**by** ( $simp\text{-}all\ add: Rep\text{-}hyp2\text{-}cltn2$ )

**from**  $\langle B_K a b c \rangle$  **and**  $\langle B_K a b d \rangle$   
**have**  $B_R ?ca ?cb ?cc$  **and**  $B_R ?ca ?cb ?cd$   
**unfolding**  $real\text{-}hyp2\text{-}B\text{-}def$  **and**  $hyp2\text{-}rep\text{-}def$  .

**from**  $\langle ?pb \neq ?pc \rangle$  **and**  $\langle ?pb \in hyp2 \rangle$  **and**  $\langle ?pc \in hyp2 \rangle$  **and**  $\langle is\text{-}K2\text{-isometry } J \rangle$   
**have**  $apply\text{-}cltn2 ?pp J$   
 $= endpoint\text{-}in\text{-}S (apply\text{-}cltn2 ?pb J) (apply\text{-}cltn2 ?pc J)$   
**by** ( $simp\ add: K2\text{-isometry}\text{-}endpoint\text{-}in\text{-}S$ )  
**also from**  $\langle apply\text{-}cltn2 ?pb J = ?pb \rangle$  **and**  $\langle apply\text{-}cltn2 ?pc J = ?pd \rangle$   
**have**  $\dots = endpoint\text{-}in\text{-}S ?pb ?pd$  **by**  $simp$   
**also from**  $\langle ?pa \neq ?pb \rangle$  **and**  $\langle ?pb \neq ?pd \rangle$   
**and**  $\langle ?pa \in hyp2 \rangle$  **and**  $\langle ?pb \in hyp2 \rangle$  **and**  $\langle ?pd \in hyp2 \rangle$  **and**  $\langle B_R ?ca ?cb ?cd \rangle$   
**have**  $\dots = endpoint\text{-}in\text{-}S ?pa ?pb$  **by** ( $simp\ add: between\text{-}endpoint\text{-}in\text{-}S$ )  
**also from**  $\langle ?pa \neq ?pb \rangle$  **and**  $\langle ?pb \neq ?pc \rangle$   
**and**  $\langle ?pa \in hyp2 \rangle$  **and**  $\langle ?pb \in hyp2 \rangle$  **and**  $\langle ?pc \in hyp2 \rangle$  **and**  $\langle B_R ?ca ?cb ?cc \rangle$   
**have**  $\dots = endpoint\text{-}in\text{-}S ?pb ?pc$  **by** ( $simp\ add: between\text{-}endpoint\text{-}in\text{-}S$ )  
**finally have**  $apply\text{-}cltn2 ?pp J = ?pp$  .

**from**  $\langle ?pb \in hyp2 \rangle$  **and**  $\langle ?pc \in hyp2 \rangle$  **and**  $\langle ?pp \in S \rangle$   
**have**  $z\text{-}non\text{-}zero ?pb$  **and**  $z\text{-}non\text{-}zero ?pc$  **and**  $z\text{-}non\text{-}zero ?pp$   
**by** ( $simp\text{-}all\ add: hyp2\text{-}S\text{-}z\text{-}non\text{-}zero$ )  
**with**  $\langle B_R ?cb ?cc ?cp \rangle$  **and**  $euclid\text{-}B\text{-}cart2\text{-}common\text{-}line [of ?pb ?pc ?pp]$   
**obtain**  $l$  **where**  $proj2\text{-}incident ?pb l$  **and**  $proj2\text{-}incident ?pp l$   
**and**  $proj2\text{-}incident ?pc l$   
**by**  $auto$   
**with**  $\langle ?pb \in hyp2 \rangle$  **and**  $\langle ?pp \in S \rangle$  **and**  $\langle is\text{-}K2\text{-isometry } J \rangle$   
**and**  $\langle apply\text{-}cltn2 ?pb J = ?pb \rangle$  **and**  $\langle apply\text{-}cltn2 ?pp J = ?pp \rangle$   
**have**  $apply\text{-}cltn2 ?pc J = ?pc$  **by** ( $rule\ hyp2\text{-}S\text{-}points\text{-}fix\text{-}line$ )  
**with**  $\langle apply\text{-}cltn2 ?pc J = ?pd \rangle$  **have**  $?pc = ?pd$  **by**  $simp$   
**thus**  $c = d$  **by** ( $subst\ Rep\text{-}hyp2\text{-}inject [symmetric]$ )

qed

**lemma**  $line\text{-}S\text{-}match\text{-}intersections$ :  
**assumes**  $p \neq q$  **and**  $r \neq s$  **and**  $p \in S$  **and**  $q \in S$  **and**  $r \in S$  **and**  $s \in S$   
**and**  $proj2\text{-}set\text{-}Col \{p, q, r, s\}$   
**shows**  $(p = r \wedge q = s) \vee (q = r \wedge p = s)$   
**proof** –

```

from ⟨proj2-set-Col {p,q,r,s}⟩
obtain l where proj2-incident p l and proj2-incident q l
  and proj2-incident r l and proj2-incident s l
  by (unfold proj2-set-Col-def) auto
with ⟨r ≠ s⟩ and ⟨p ∈ S⟩ and ⟨q ∈ S⟩ and ⟨r ∈ S⟩ and ⟨s ∈ S⟩
have p = r ∨ p = s and q = r ∨ q = s
  by (simp-all add: line-S-two-intersections-only)

show (p = r ∧ q = s) ∨ (q = r ∧ p = s)
proof cases
  assume p = r
  with ⟨p ≠ q⟩ and ⟨q = r ∨ q = s⟩
  show (p = r ∧ q = s) ∨ (q = r ∧ p = s) by simp
next
  assume p ≠ r
  with ⟨p = r ∨ p = s⟩ have p = s by simp
  with ⟨p ≠ q⟩ and ⟨q = r ∨ q = s⟩
  show (p = r ∧ q = s) ∨ (q = r ∧ p = s) by simp
qed
qed

definition are-endpoints-in-S :: [proj2, proj2, proj2, proj2] ⇒ bool where
  are-endpoints-in-S p q a b
  ≙ p ≠ q ∧ p ∈ S ∧ q ∈ S ∧ a ∈ hyp2 ∧ b ∈ hyp2 ∧ proj2-set-Col {p,q,a,b}

lemma are-endpoints-in-S':
  assumes p ≠ q and a ≠ b and p ∈ S and q ∈ S and a ∈ hyp2 ∪ S
  and b ∈ hyp2 ∪ S and proj2-set-Col {p,q,a,b}
  shows (p = endpoint-in-S a b ∧ q = endpoint-in-S b a)
  ∨ (q = endpoint-in-S a b ∧ p = endpoint-in-S b a)
  (is (p = ?r ∧ q = ?s) ∨ (q = ?r ∧ p = ?s))
proof –
  from ⟨a ≠ b⟩ and ⟨a ∈ hyp2 ∪ S⟩ and ⟨b ∈ hyp2 ∪ S⟩
  have ?r ≠ ?s by (simp add: endpoint-in-S-swap)

  from ⟨a ∈ hyp2 ∪ S⟩ and ⟨b ∈ hyp2 ∪ S⟩
  have ?r ∈ S and ?s ∈ S by (simp-all add: endpoint-in-S)

  from ⟨proj2-set-Col {p,q,a,b}⟩
  obtain l where proj2-incident p l and proj2-incident q l
    and proj2-incident a l and proj2-incident b l
    by (unfold proj2-set-Col-def) auto

  from ⟨a ≠ b⟩ and ⟨a ∈ hyp2 ∪ S⟩ and ⟨b ∈ hyp2 ∪ S⟩ and ⟨proj2-incident a l⟩
    and ⟨proj2-incident b l⟩
  have proj2-incident ?r l and proj2-incident ?s l
    by (simp-all add: endpoint-in-S-incident)
  with ⟨proj2-incident p l⟩ and ⟨proj2-incident q l⟩
  have proj2-set-Col {p,q,?r,?s}

```

**by** (*unfold proj2-set-Col-def*) (*simp add: exI [of - l]*)  
**with**  $\langle p \neq q \rangle$  **and**  $\langle ?r \neq ?s \rangle$  **and**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$  **and**  $\langle ?r \in S \rangle$  **and**  $\langle ?s \in S \rangle$   
**show**  $(p = ?r \wedge q = ?s) \vee (q = ?r \wedge p = ?s)$   
**by** (*rule line-S-match-intersections*)  
**qed**

**lemma** *are-endpoints-in-S*:

**assumes**  $a \neq b$  **and** *are-endpoints-in-S*  $p$   $q$   $a$   $b$   
**shows**  $(p = \text{endpoint-in-S } a \ b \wedge q = \text{endpoint-in-S } b \ a)$   
 $\vee (q = \text{endpoint-in-S } a \ b \wedge p = \text{endpoint-in-S } b \ a)$   
**using** *assms*  
**by** (*unfold are-endpoints-in-S-def*) (*simp add: are-endpoints-in-S'*)

**lemma** *S-intersections-endpoints-in-S*:

**assumes**  $a \neq 0$  **and**  $b \neq 0$  **and** *proj2-abs*  $a \neq \text{proj2-abs } b$  (**is**  $?pa \neq ?pb$ )  
**and** *proj2-abs*  $a \in \text{hyp2}$  **and** *proj2-abs*  $b \in \text{hyp2} \cup S$   
**shows**  $(S\text{-intersection1 } a \ b = \text{endpoint-in-S } ?pa \ ?pb$   
 $\wedge S\text{-intersection2 } a \ b = \text{endpoint-in-S } ?pb \ ?pa)$   
 $\vee (S\text{-intersection2 } a \ b = \text{endpoint-in-S } ?pa \ ?pb$   
 $\wedge S\text{-intersection1 } a \ b = \text{endpoint-in-S } ?pb \ ?pa)$   
**(is**  $(?pp = ?pr \wedge ?pq = ?ps) \vee (?pq = ?pr \wedge ?pp = ?ps)$ )

**proof** –

**from**  $\langle a \neq 0 \rangle$  **and**  $\langle b \neq 0 \rangle$  **and**  $\langle ?pa \neq ?pb \rangle$  **and**  $\langle ?pa \in \text{hyp2} \rangle$   
**have**  $?pp \neq ?pq$  **by** (*simp add: S-intersections-distinct*)

**from**  $\langle a \neq 0 \rangle$  **and**  $\langle b \neq 0 \rangle$  **and**  $\langle ?pa \neq ?pb \rangle$  **and**  $\langle \text{proj2-abs } a \in \text{hyp2} \rangle$   
**have**  $?pp \in S$  **and**  $?pq \in S$   
**by** (*simp-all add: S-intersections-in-S*)

**let**  $?l = \text{proj2-line-through } ?pa \ ?pb$

**have** *proj2-incident*  $?pa$   $?l$  **and** *proj2-incident*  $?pb$   $?l$

**by** (*rule proj2-line-through-incident*)**+**

**with**  $\langle a \neq 0 \rangle$  **and**  $\langle b \neq 0 \rangle$  **and**  $\langle ?pa \neq ?pb \rangle$

**have** *proj2-incident*  $?pp$   $?l$  **and** *proj2-incident*  $?pq$   $?l$

**by** (*rule S-intersections-incident*)**+**

**with**  $\langle \text{proj2-incident } ?pa \ ?l \rangle$  **and**  $\langle \text{proj2-incident } ?pb \ ?l \rangle$

**have** *proj2-set-Col*  $\{?pp, ?pq, ?pa, ?pb\}$

**by** (*unfold proj2-set-Col-def*) (*simp add: exI [of - ?l]*)

**with**  $\langle ?pp \neq ?pq \rangle$  **and**  $\langle ?pa \neq ?pb \rangle$  **and**  $\langle ?pp \in S \rangle$  **and**  $\langle ?pq \in S \rangle$  **and**  $\langle ?pa \in \text{hyp2} \rangle$

**and**  $\langle ?pb \in \text{hyp2} \cup S \rangle$

**show**  $(?pp = ?pr \wedge ?pq = ?ps) \vee (?pq = ?pr \wedge ?pp = ?ps)$

**by** (*simp add: are-endpoints-in-S'*)

**qed**

**lemma** *between-endpoints-in-S*:

**assumes**  $a \neq b$  **and**  $a \in \text{hyp2} \cup S$  **and**  $b \in \text{hyp2} \cup S$   
**shows**  $B_{\mathbb{R}}$

$(\text{cart2-pt } (\text{endpoint-in-}S \ a \ b)) \ (\text{cart2-pt } \ a) \ (\text{cart2-pt } (\text{endpoint-in-}S \ b \ a))$   
 $(\text{is } B_{\mathbb{R}} \ ?cp \ ?ca \ ?cq)$   
**proof** –  
**let**  $?cb = \text{cart2-pt } b$   
**from**  $\langle b \in \text{hyp2} \cup S \rangle$  **and**  $\langle a \in \text{hyp2} \cup S \rangle$  **and**  $\langle a \neq b \rangle$   
**have**  $?cb \neq ?ca$  **by**  $(\text{auto simp add: hyp2-S-cart2-inj})$   
  
**from**  $\langle a \in \text{hyp2} \cup S \rangle$  **and**  $\langle b \in \text{hyp2} \cup S \rangle$   
**have**  $B_{\mathbb{R}} \ ?ca \ ?cb \ ?cp$  **and**  $B_{\mathbb{R}} \ ?cb \ ?ca \ ?cq$  **by**  $(\text{simp-all add: endpoint-in-}S)$   
  
**from**  $\langle B_{\mathbb{R}} \ ?ca \ ?cb \ ?cp \rangle$  **have**  $B_{\mathbb{R}} \ ?cp \ ?cb \ ?ca$  **by**  $(\text{rule real-euclid.th3-2})$   
**with**  $\langle ?cb \neq ?ca \rangle$  **and**  $\langle B_{\mathbb{R}} \ ?cb \ ?ca \ ?cq \rangle$   
**show**  $B_{\mathbb{R}} \ ?cp \ ?ca \ ?cq$  **by**  $(\text{simp add: real-euclid.th3-7-1})$   
**qed**

**lemma** *S-hyp2-S-cart2-append1*:  
**assumes**  $p \neq q$  **and**  $p \in S$  **and**  $q \in S$  **and**  $a \in \text{hyp2}$   
**and** *proj2-incident*  $p \ l$  **and** *proj2-incident*  $q \ l$  **and** *proj2-incident*  $a \ l$   
**shows**  $\exists k. k > 0 \wedge k < 1$   
 $\wedge \text{cart2-append1 } a = k *_{\mathbb{R}} \text{cart2-append1 } q + (1 - k) *_{\mathbb{R}} \text{cart2-append1 } p$   
**proof** –  
**from**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$   
**have** *z-non-zero*  $p$  **and** *z-non-zero*  $q$  **and** *z-non-zero*  $a$   
**by**  $(\text{simp-all add: hyp2-S-z-non-zero})$   
  
**from** *assms*  
**have**  $B_{\mathbb{R}} \ (\text{cart2-pt } p) \ (\text{cart2-pt } a) \ (\text{cart2-pt } q)$  **(is**  $B_{\mathbb{R}} \ ?cp \ ?ca \ ?cq)$   
**by**  $(\text{simp add: hyp2-incident-in-middle})$   
  
**from**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$   
**have**  $a \neq p$  **and**  $a \neq q$  **by**  $(\text{simp-all add: hyp2-S-not-equal})$   
  
**with**  $\langle \text{z-non-zero } p \rangle$  **and**  $\langle \text{z-non-zero } a \rangle$  **and**  $\langle \text{z-non-zero } q \rangle$   
**and**  $\langle B_{\mathbb{R}} \ ?cp \ ?ca \ ?cq \rangle$   
**show**  $\exists k. k > 0 \wedge k < 1$   
 $\wedge \text{cart2-append1 } a = k *_{\mathbb{R}} \text{cart2-append1 } q + (1 - k) *_{\mathbb{R}} \text{cart2-append1 } p$   
**by**  $(\text{rule cart2-append1-between-strict})$   
**qed**

**lemma** *are-endpoints-in-S-swap-34*:  
**assumes** *are-endpoints-in-S*  $p \ q \ a \ b$   
**shows** *are-endpoints-in-S*  $p \ q \ b \ a$   
**proof** –  
**have**  $\{p, q, b, a\} = \{p, q, a, b\}$  **by** *auto*  
**with**  $\langle \text{are-endpoints-in-S } p \ q \ a \ b \rangle$   
**show** *are-endpoints-in-S*  $p \ q \ b \ a$  **by**  $(\text{unfold are-endpoints-in-S-def})$  *simp*  
**qed**

**lemma** *proj2-set-Col-endpoints-in-S*:

**assumes**  $a \neq b$  **and**  $a \in \text{hyp2} \cup S$  **and**  $b \in \text{hyp2} \cup S$   
**shows**  $\text{proj2-set-Col } \{\text{endpoint-in-}S \ a \ b, \text{endpoint-in-}S \ b \ a, \ a, \ b\}$   
(is  $\text{proj2-set-Col } \{?p, ?q, a, b\}$ )  
**proof** –  
**let**  $?l = \text{proj2-line-through } a \ b$   
**have**  $\text{proj2-incident } a \ ?l$  **and**  $\text{proj2-incident } b \ ?l$   
**by** (rule  $\text{proj2-line-through-incident}$ )  
**with**  $\langle a \neq b \rangle$  **and**  $\langle a \in \text{hyp2} \cup S \rangle$  **and**  $\langle b \in \text{hyp2} \cup S \rangle$   
**have**  $\text{proj2-incident } ?p \ ?l$  **and**  $\text{proj2-incident } ?q \ ?l$   
**by** (simp-all add:  $\text{endpoint-in-}S\text{-incident}$ )  
**with**  $\langle \text{proj2-incident } a \ ?l \rangle$  **and**  $\langle \text{proj2-incident } b \ ?l \rangle$   
**show**  $\text{proj2-set-Col } \{?p, ?q, a, b\}$   
**by** (unfold  $\text{proj2-set-Col-def}$ ) (simp add:  $\text{exI [of - ?l]}$ )  
**qed**

**lemma**  $\text{endpoints-in-}S\text{-are-endpoints-in-}S$ :  
**assumes**  $a \neq b$  **and**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$   
**shows**  $\text{are-endpoints-in-}S \ (\text{endpoint-in-}S \ a \ b) \ (\text{endpoint-in-}S \ b \ a) \ a \ b$   
(is  $\text{are-endpoints-in-}S \ ?p \ ?q \ a \ b$ )

**proof** –  
**from**  $\langle a \neq b \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$   
**have**  $?p \neq ?q$  **by** (simp add:  $\text{endpoint-in-}S\text{-swap}$ )  
  
**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$   
**have**  $?p \in S$  **and**  $?q \in S$  **by** (simp-all add:  $\text{endpoint-in-}S$ )  
  
**from**  $\text{assms}$   
**have**  $\text{proj2-set-Col } \{?p, ?q, a, b\}$  **by** (simp add:  $\text{proj2-set-Col-endpoints-in-}S$ )  
**with**  $\langle ?p \neq ?q \rangle$  **and**  $\langle ?p \in S \rangle$  **and**  $\langle ?q \in S \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$   
**show**  $\text{are-endpoints-in-}S \ ?p \ ?q \ a \ b$  **by** (unfold  $\text{are-endpoints-in-}S\text{-def}$ ) simp  
**qed**

**lemma**  $\text{endpoint-in-}S\text{-}S\text{-hyp2-distinct}$ :  
**assumes**  $p \in S$  **and**  $a \in \text{hyp2} \cup S$  **and**  $p \neq a$   
**shows**  $\text{endpoint-in-}S \ p \ a \neq p$

**proof**  
**from**  $\langle p \neq a \rangle$  **and**  $\langle p \in S \rangle$  **and**  $\langle a \in \text{hyp2} \cup S \rangle$   
**have**  $B_{\mathbb{R}} \ (\text{cart2-pt } p) \ (\text{cart2-pt } a) \ (\text{cart2-pt } (\text{endpoint-in-}S \ p \ a))$   
**by** (simp add:  $\text{endpoint-in-}S$ )  
  
**assume**  $\text{endpoint-in-}S \ p \ a = p$   
**with**  $\langle B_{\mathbb{R}} \ (\text{cart2-pt } p) \ (\text{cart2-pt } a) \ (\text{cart2-pt } (\text{endpoint-in-}S \ p \ a)) \rangle$   
**have**  $\text{cart2-pt } p = \text{cart2-pt } a$  **by** (simp add:  $\text{real-euclid.A6}$ )  
**with**  $\langle p \in S \rangle$  **and**  $\langle a \in \text{hyp2} \cup S \rangle$  **have**  $p = a$  **by** (simp add:  $\text{hyp2-}S\text{-cart2-inj}$ )  
**with**  $\langle p \neq a \rangle$  **show**  $\text{False} \ ..$   
**qed**

**lemma**  $\text{endpoint-in-}S\text{-}S\text{-strict-hyp2-distinct}$ :  
**assumes**  $p \in S$  **and**  $a \in \text{hyp2}$

**shows** *endpoint-in-S*  $p \ a \neq p$   
**proof** –  
**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle p \in S \rangle$   
**have**  $p \neq a$  **by** (rule *hyp2-S-not-equal* [*symmetric*])  
**with** *assms*  
**show** *endpoint-in-S*  $p \ a \neq p$  **by** (*simp add: endpoint-in-S-S-hyp2-distinct*)  
**qed**

**lemma** *end-and-opposite-are-endpoints-in-S*:  
**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and**  $p \in S$   
**and** *proj2-incident*  $a \ l$  **and** *proj2-incident*  $b \ l$  **and** *proj2-incident*  $p \ l$   
**shows** *are-endpoints-in-S*  $p \ (\text{endpoint-in-S } p \ b) \ a \ b$   
**(is** *are-endpoints-in-S*  $p \ ?q \ a \ b$ )

**proof** –  
**from**  $\langle p \in S \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$   
**have**  $p \neq ?q$  **by** (rule *endpoint-in-S-S-strict-hyp2-distinct* [*symmetric*])  
  
**from**  $\langle p \in S \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **have**  $?q \in S$  **by** (*simp add: endpoint-in-S*)

**from**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle p \in S \rangle$   
**have**  $p \neq b$  **by** (rule *hyp2-S-not-equal* [*symmetric*])  
**with**  $\langle p \in S \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } p \ l \rangle$  **and**  $\langle \text{proj2-incident } b \ l \rangle$   
**have** *proj2-incident*  $?q \ l$  **by** (*simp add: endpoint-in-S-incident*)  
**with**  $\langle \text{proj2-incident } p \ l \rangle$  **and**  $\langle \text{proj2-incident } a \ l \rangle$  **and**  $\langle \text{proj2-incident } b \ l \rangle$   
**have** *proj2-set-Col*  $\{p, ?q, a, b\}$   
**by** (*unfold proj2-set-Col-def*) (*simp add: exI [of - l]*)  
**with**  $\langle p \neq ?q \rangle$  **and**  $\langle p \in S \rangle$  **and**  $\langle ?q \in S \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$   
**show** *are-endpoints-in-S*  $p \ ?q \ a \ b$  **by** (*unfold are-endpoints-in-S-def*) *simp*  
**qed**

**lemma** *real-hyp2-B-hyp2-cltn2*:  
**assumes** *is-K2-isometry*  $J$  **and**  $B_K \ a \ b \ c$   
**shows**  $B_K \ (\text{hyp2-cltn2 } a \ J) \ (\text{hyp2-cltn2 } b \ J) \ (\text{hyp2-cltn2 } c \ J)$   
**(is**  $B_K \ ?aJ \ ?bJ \ ?cJ$ )

**proof** –  
**from**  $\langle B_K \ a \ b \ c \rangle$   
**have**  $B_{\mathbb{R}} \ (\text{hyp2-rep } a) \ (\text{hyp2-rep } b) \ (\text{hyp2-rep } c)$  **by** (*unfold real-hyp2-B-def*)  
**with**  $\langle \text{is-K2-isometry } J \rangle$   
**have**  $B_{\mathbb{R}} \ (\text{cart2-pt } (\text{apply-cltn2 } (\text{Rep-hyp2 } a) \ J))$   
 $\quad (\text{cart2-pt } (\text{apply-cltn2 } (\text{Rep-hyp2 } b) \ J))$   
 $\quad (\text{cart2-pt } (\text{apply-cltn2 } (\text{Rep-hyp2 } c) \ J))$   
**by** (*unfold hyp2-rep-def*) (*simp add: Rep-hyp2 statement-63*)  
**moreover from**  $\langle \text{is-K2-isometry } J \rangle$   
**have** *apply-cltn2*  $(\text{Rep-hyp2 } a) \ J \in \text{hyp2}$   
**and** *apply-cltn2*  $(\text{Rep-hyp2 } b) \ J \in \text{hyp2}$   
**and** *apply-cltn2*  $(\text{Rep-hyp2 } c) \ J \in \text{hyp2}$   
**by** (rule *apply-cltn2-Rep-hyp2*)  
**ultimately show**  $B_K \ (\text{hyp2-cltn2 } a \ J) \ (\text{hyp2-cltn2 } b \ J) \ (\text{hyp2-cltn2 } c \ J)$   
**unfolding** *hyp2-cltn2-def* **and** *real-hyp2-B-def* **and** *hyp2-rep-def*

by (*simp add: Abs-hyp2-inverse*)  
**qed**

**lemma** *real-hyp2-C-hyp2-cltn2*:  
**assumes** *is-K2-isometry J*  
**shows**  $a \equiv_K (hyp2-cltn2\ a\ J)$  ( $hyp2-cltn2\ b\ J$ ) (**is**  $a \equiv_K\ ?aJ\ ?bJ$ )  
**using** *assms* **by** (*unfold real-hyp2-C-def*) (*simp add: exI [of - J]*)

## 8.10 Perpendicularity

**definition** *M-perp* :: *proj2-line*  $\Rightarrow$  *proj2-line*  $\Rightarrow$  *bool* **where**  
*M-perp l m*  $\triangleq$  *proj2-incident (pole l) m*

**lemma** *M-perp-sym*:  
**assumes** *M-perp l m*  
**shows** *M-perp m l*  
**proof** –  
**from**  $\langle M-perp\ l\ m \rangle$  **have** *proj2-incident (pole l) m* **by** (*unfold M-perp-def*)  
**hence** *proj2-incident (pole m) (polar (pole l))* **by** (*rule incident-pole-polar*)  
**hence** *proj2-incident (pole m) l* **by** (*simp add: polar-pole*)  
**thus** *M-perp m l* **by** (*unfold M-perp-def*)  
**qed**

**lemma** *M-perp-to-compass*:  
**assumes** *M-perp l m* **and**  $a \in hyp2$  **and** *proj2-incident a l*  
**and**  $b \in hyp2$  **and** *proj2-incident b m*  
**shows**  $\exists J. is-K2-isometry\ J$   
 $\wedge$  *apply-cltn2-line equator J = l*  $\wedge$  *apply-cltn2-line meridian J = m*  
**proof** –  
**from**  $\langle a \in K2 \rangle$  **and**  $\langle proj2-incident\ a\ l \rangle$   
**and** *line-through-K2-intersect-S-twice [of a l]*  
**obtain**  $p$  **and**  $q$  **where**  $p \neq q$  **and**  $p \in S$  **and**  $q \in S$   
**and** *proj2-incident p l* **and** *proj2-incident q l*  
**by** *auto*

**have**  $\exists r. r \in S \wedge r \notin \{p, q\} \wedge proj2-incident\ r\ m$

**proof** *cases*  
**assume** *proj2-incident p m*

**from**  $\langle b \in K2 \rangle$  **and**  $\langle proj2-incident\ b\ m \rangle$   
**and** *line-through-K2-intersect-S-again [of b m]*  
**obtain**  $r$  **where**  $r \in S$  **and**  $r \neq p$  **and** *proj2-incident r m* **by** *auto*

**have**  $r \notin \{p, q\}$

**proof**  
**assume**  $r \in \{p, q\}$   
**with**  $\langle r \neq p \rangle$  **have**  $r = q$  **by** *simp*  
**with**  $\langle proj2-incident\ r\ m \rangle$  **have** *proj2-incident q m* **by** *simp*  
**with**  $\langle proj2-incident\ p\ l \rangle$  **and**  $\langle proj2-incident\ q\ l \rangle$

**and**  $\langle \text{proj2-incident } p \ m \rangle$  **and**  $\langle \text{proj2-incident } q \ m \rangle$  **and**  $\langle p \neq q \rangle$   
**and**  $\text{proj2-incident-unique } [of \ p \ l \ q \ m]$   
**have**  $l = m$  **by** *simp*  
**with**  $\langle M\text{-perp } l \ m \rangle$  **have**  $M\text{-perp } l \ l$  **by** *simp*  
**hence**  $\text{proj2-incident } (\text{pole } l) \ l$  **(is**  $\text{proj2-incident } ?s \ l)$   
**by** *(unfold M-perp-def)*  
**hence**  $\text{proj2-incident } ?s$   $(\text{polar } ?s)$  **by** *(subst polar-pole)*  
**hence**  $?s \in S$  **by** *(simp add: incident-own-polar-in-S)*  
**with**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$  **and**  $\langle \text{proj2-incident } p \ l \rangle$  **and**  $\langle \text{proj2-incident } q \ l \rangle$   
**and**  $\text{point-in-S-polar-is-tangent } [of \ ?s]$   
**have**  $p = ?s$  **and**  $q = ?s$  **by** *(auto simp add: polar-pole)*  
**with**  $\langle p \neq q \rangle$  **show** *False* **by** *simp*  
**qed**  
**with**  $\langle r \in S \rangle$  **and**  $\langle \text{proj2-incident } r \ m \rangle$   
**show**  $\exists r. r \in S \wedge r \notin \{p, q\} \wedge \text{proj2-incident } r \ m$   
**by** *(simp add: exI [of - r])*  
**next**  
**assume**  $\neg \text{proj2-incident } p \ m$   
  
**from**  $\langle b \in K2 \rangle$  **and**  $\langle \text{proj2-incident } b \ m \rangle$   
**and**  $\text{line-through-K2-intersect-S-again } [of \ b \ m]$   
**obtain**  $r$  **where**  $r \in S$  **and**  $r \neq q$  **and**  $\text{proj2-incident } r \ m$  **by** *auto*  
  
**from**  $\langle \neg \text{proj2-incident } p \ m \rangle$  **and**  $\langle \text{proj2-incident } r \ m \rangle$  **have**  $r \neq p$  **by** *auto*  
**with**  $\langle r \in S \rangle$  **and**  $\langle r \neq q \rangle$  **and**  $\langle \text{proj2-incident } r \ m \rangle$   
**show**  $\exists r. r \in S \wedge r \notin \{p, q\} \wedge \text{proj2-incident } r \ m$   
**by** *(simp add: exI [of - r])*  
**qed**  
**then obtain**  $r$  **where**  $r \in S$  **and**  $r \notin \{p, q\}$  **and**  $\text{proj2-incident } r \ m$  **by** *auto*  
  
**from**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$  **and**  $\langle r \in S \rangle$  **and**  $\langle p \neq q \rangle$  **and**  $\langle r \notin \{p, q\} \rangle$   
**and**  $\text{statement65-special-case } [of \ p \ q \ r]$   
**obtain**  $J$  **where**  $\text{is-K2-isometry } J$  **and**  $\text{apply-cltn2 east } J = p$   
**and**  $\text{apply-cltn2 west } J = q$  **and**  $\text{apply-cltn2 north } J = r$   
**and**  $\text{apply-cltn2 far-north } J = \text{proj2-intersection } (\text{polar } p) (\text{polar } q)$   
**by** *auto*  
  
**from**  $\langle \text{apply-cltn2 east } J = p \rangle$  **and**  $\langle \text{apply-cltn2 west } J = q \rangle$   
**and**  $\langle \text{proj2-incident } p \ l \rangle$  **and**  $\langle \text{proj2-incident } q \ l \rangle$   
**have**  $\text{proj2-incident } (\text{apply-cltn2 east } J) \ l$   
**and**  $\text{proj2-incident } (\text{apply-cltn2 west } J) \ l$   
**by** *simp-all*  
**with**  $\text{east-west-distinct}$  **and**  $\text{east-west-on-equator}$   
**have**  $\text{apply-cltn2-line equator } J = l$  **by** *(rule apply-cltn2-line-unique)*  
  
**from**  $\langle \text{apply-cltn2 north } J = r \rangle$  **and**  $\langle \text{proj2-incident } r \ m \rangle$   
**have**  $\text{proj2-incident } (\text{apply-cltn2 north } J) \ m$  **by** *simp*  
  
**from**  $\langle p \neq q \rangle$  **and**  $\text{polar-inj}$  **have**  $\text{polar } p \neq \text{polar } q$  **by** *fast*

**from**  $\langle \text{proj2-incident } p \ l \rangle$  **and**  $\langle \text{proj2-incident } q \ l \rangle$   
**have**  $\text{proj2-incident } (\text{pole } l) (\text{polar } p)$   
**and**  $\text{proj2-incident } (\text{pole } l) (\text{polar } q)$   
**by** (*simp-all add: incident-pole-polar*)  
**with**  $\langle \text{polar } p \neq \text{polar } q \rangle$   
**have**  $\text{pole } l = \text{proj2-intersection } (\text{polar } p) (\text{polar } q)$   
**by** (*rule proj2-intersection-unique*)  
**with**  $\langle \text{apply-cltn2 far-north } J = \text{proj2-intersection } (\text{polar } p) (\text{polar } q) \rangle$   
**have**  $\text{apply-cltn2 far-north } J = \text{pole } l$  **by** *simp*  
**with**  $\langle M\text{-perp } l \ m \rangle$   
**have**  $\text{proj2-incident } (\text{apply-cltn2 far-north } J) \ m$  **by** (*unfold M-perp-def*) *simp*  
**with** *north-far-north-distinct* **and** *north-south-far-north-on-meridian*  
**and**  $\langle \text{proj2-incident } (\text{apply-cltn2 north } J) \ m \rangle$   
**have**  $\text{apply-cltn2-line meridian } J = m$  **by** (*simp add: apply-cltn2-line-unique*)  
**with**  $\langle \text{is-K2-isometry } J \rangle$  **and**  $\langle \text{apply-cltn2-line equator } J = l \rangle$   
**show**  $\exists J. \text{is-K2-isometry } J$   
 $\wedge \text{apply-cltn2-line equator } J = l \wedge \text{apply-cltn2-line meridian } J = m$   
**by** (*simp add: exI [of - J]*)  
**qed**

**definition** *drop-perp* :: *proj2*  $\Rightarrow$  *proj2-line*  $\Rightarrow$  *proj2-line* **where**  
*drop-perp*  $p \ l \triangleq \text{proj2-line-through } p (\text{pole } l)$

**lemma** *drop-perp-incident*:  $\text{proj2-incident } p (\text{drop-perp } p \ l)$   
**by** (*unfold drop-perp-def*) (*rule proj2-line-through-incident*)

**lemma** *drop-perp-perp*:  $M\text{-perp } l (\text{drop-perp } p \ l)$   
**by** (*unfold drop-perp-def M-perp-def*) (*rule proj2-line-through-incident*)

**definition** *perp-foot* :: *proj2*  $\Rightarrow$  *proj2-line*  $\Rightarrow$  *proj2* **where**  
*perp-foot*  $p \ l \triangleq \text{proj2-intersection } l (\text{drop-perp } p \ l)$

**lemma** *perp-foot-incident*:  
**shows**  $\text{proj2-incident } (\text{perp-foot } p \ l) \ l$   
**and**  $\text{proj2-incident } (\text{perp-foot } p \ l) (\text{drop-perp } p \ l)$   
**by** (*unfold perp-foot-def*) (*rule proj2-intersection-incident*)+

**lemma** *M-perp-hyp2*:  
**assumes**  $M\text{-perp } l \ m$  **and**  $a \in \text{hyp2}$  **and**  $\text{proj2-incident } a \ l$  **and**  $b \in \text{hyp2}$   
**and**  $\text{proj2-incident } b \ m$  **and**  $\text{proj2-incident } c \ l$  **and**  $\text{proj2-incident } c \ m$   
**shows**  $c \in \text{hyp2}$

**proof** –  
**from**  $\langle M\text{-perp } l \ m \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } a \ l \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$   
**and**  $\langle \text{proj2-incident } b \ m \rangle$  **and** *M-perp-to-compass* [*of l m a b*]  
**obtain**  $J$  **where** *is-K2-isometry*  $J$  **and**  $\text{apply-cltn2-line equator } J = l$   
**and**  $\text{apply-cltn2-line meridian } J = m$   
**by** *auto*

**from**  $\langle \text{is-}K2\text{-isometry } J \rangle$  **and**  $K2\text{-centre-in-}K2$   
**have**  $\text{apply-cltn2 } K2\text{-centre } J \in \text{hyp2}$   
**by** (*rule statement60-one-way*)

**from**  $\langle \text{proj2-incident } c \ l \rangle$  **and**  $\langle \text{apply-cltn2-line equator } J = l \rangle$   
**and**  $\langle \text{proj2-incident } c \ m \rangle$  **and**  $\langle \text{apply-cltn2-line meridian } J = m \rangle$   
**have**  $\text{proj2-incident } c \ (\text{apply-cltn2-line equator } J)$   
**and**  $\text{proj2-incident } c \ (\text{apply-cltn2-line meridian } J)$   
**by** *simp-all*  
**with** *equator-meridian-distinct* **and**  $K2\text{-centre-on-equator-meridian}$   
**have**  $\text{apply-cltn2 } K2\text{-centre } J = c$  **by** (*rule apply-cltn2-unique*)  
**with**  $\langle \text{apply-cltn2 } K2\text{-centre } J \in \text{hyp2} \rangle$  **show**  $c \in \text{hyp2}$  **by** *simp*  
**qed**

**lemma** *perp-foot-hyp2*:  
**assumes**  $a \in \text{hyp2}$  **and**  $\text{proj2-incident } a \ l$  **and**  $b \in \text{hyp2}$   
**shows**  $\text{perp-foot } b \ l \in \text{hyp2}$   
**using**  $\text{drop-perp-perp } [of \ l \ b]$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } a \ l \rangle$   
**and**  $\langle b \in \text{hyp2} \rangle$  **and**  $\text{drop-perp-incident } [of \ b \ l]$   
**and**  $\text{perp-foot-incident } [of \ b \ l]$   
**by** (*rule M-perp-hyp2*)

**definition** *perp-up* ::  $\text{proj2} \Rightarrow \text{proj2-line} \Rightarrow \text{proj2}$  **where**  
 $\text{perp-up } a \ l$   
 $\triangleq$  *if*  $\text{proj2-incident } a \ l$  *then*  $\epsilon \ p. \ p \in S \wedge \text{proj2-incident } p \ (\text{drop-perp } a \ l)$   
*else*  $\text{endpoint-in-}S \ (\text{perp-foot } a \ l) \ a$

**lemma** *perp-up-degenerate-in-S-incident*:  
**assumes**  $a \in \text{hyp2}$  **and**  $\text{proj2-incident } a \ l$   
**shows**  $\text{perp-up } a \ l \in S$  (**is**  $?p \in S$ )  
**and**  $\text{proj2-incident } (\text{perp-up } a \ l) \ (\text{drop-perp } a \ l)$

**proof** –  
**from**  $\langle \text{proj2-incident } a \ l \rangle$   
**have**  $?p = (\epsilon \ p. \ p \in S \wedge \text{proj2-incident } p \ (\text{drop-perp } a \ l))$   
**by** (*unfold perp-up-def*) *simp*

**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\text{drop-perp-incident } [of \ a \ l]$   
**have**  $\exists \ p. \ p \in S \wedge \text{proj2-incident } p \ (\text{drop-perp } a \ l)$   
**by** (*rule line-through-K2-intersect-S*)  
**hence**  $?p \in S \wedge \text{proj2-incident } ?p \ (\text{drop-perp } a \ l)$   
**unfolding**  $\langle ?p = (\epsilon \ p. \ p \in S \wedge \text{proj2-incident } p \ (\text{drop-perp } a \ l)) \rangle$   
**by** (*rule someI-ex*)  
**thus**  $?p \in S$  **and**  $\text{proj2-incident } ?p \ (\text{drop-perp } a \ l)$  **by** *simp-all*  
**qed**

**lemma** *perp-up-non-degenerate-in-S-at-end*:  
**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and**  $\text{proj2-incident } b \ l$   
**and**  $\neg \text{proj2-incident } a \ l$   
**shows**  $\text{perp-up } a \ l \in S$

**and**  $B_R$  ( $\text{cart2-pt}$  ( $\text{perp-foot}$   $a$   $l$ )) ( $\text{cart2-pt}$   $a$ ) ( $\text{cart2-pt}$  ( $\text{perp-up}$   $a$   $l$ ))  
**proof** –  
**from**  $\langle \neg \text{proj2-incident } a \ l \rangle$   
**have**  $\text{perp-up } a \ l = \text{endpoint-in-}S$  ( $\text{perp-foot } a \ l$ )  $a$   
**by** ( $\text{unfold perp-up-def}$ )  $\text{simp}$   
  
**from**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } b \ l \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$   
**have**  $\text{perp-foot } a \ l \in \text{hyp2}$  **by** ( $\text{rule perp-foot-hyp2}$ )  
**with**  $\langle a \in \text{hyp2} \rangle$   
**show**  $\text{perp-up } a \ l \in S$   
**and**  $B_R$  ( $\text{cart2-pt}$  ( $\text{perp-foot } a \ l$ )) ( $\text{cart2-pt } a$ ) ( $\text{cart2-pt}$  ( $\text{perp-up } a \ l$ ))  
**unfolding**  $\langle \text{perp-up } a \ l = \text{endpoint-in-}S$  ( $\text{perp-foot } a \ l$ )  $a \rangle$   
**by** ( $\text{simp-all add: endpoint-in-}S$ )  
**qed**

**lemma**  $\text{perp-up-in-}S$ :  
**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and**  $\text{proj2-incident } b \ l$   
**shows**  $\text{perp-up } a \ l \in S$   
**proof**  $\text{cases}$   
**assume**  $\text{proj2-incident } a \ l$   
**with**  $\langle a \in \text{hyp2} \rangle$   
**show**  $\text{perp-up } a \ l \in S$  **by** ( $\text{rule perp-up-degenerate-in-}S\text{-incident}$ )  
**next**  
**assume**  $\neg \text{proj2-incident } a \ l$   
**with**  $\text{assms}$   
**show**  $\text{perp-up } a \ l \in S$  **by** ( $\text{rule perp-up-non-degenerate-in-}S\text{-at-end}$ )  
**qed**

**lemma**  $\text{perp-up-incident}$ :  
**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and**  $\text{proj2-incident } b \ l$   
**shows**  $\text{proj2-incident}$  ( $\text{perp-up } a \ l$ ) ( $\text{drop-perp } a \ l$ )  
**(is**  $\text{proj2-incident } ?p \ ?m$ **)**  
**proof**  $\text{cases}$   
**assume**  $\text{proj2-incident } a \ l$   
**with**  $\langle a \in \text{hyp2} \rangle$   
**show**  $\text{proj2-incident } ?p \ ?m$  **by** ( $\text{rule perp-up-degenerate-in-}S\text{-incident}$ )  
**next**  
**assume**  $\neg \text{proj2-incident } a \ l$   
**hence**  $?p = \text{endpoint-in-}S$  ( $\text{perp-foot } a \ l$ )  $a$  **(is**  $?p = \text{endpoint-in-}S$   $?c \ a$ **)**  
**by** ( $\text{unfold perp-up-def}$ )  $\text{simp}$

**from**  $\text{perp-foot-incident}$  [ $\text{of } a \ l$ ] **and**  $\langle \neg \text{proj2-incident } a \ l \rangle$   
**have**  $?c \neq a$  **by**  $\text{auto}$

**from**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } b \ l \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$   
**have**  $?c \in \text{hyp2}$  **by** ( $\text{rule perp-foot-hyp2}$ )  
**with**  $\langle ?c \neq a \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\text{drop-perp-incident}$  [ $\text{of } a \ l$ ]  
**and**  $\text{perp-foot-incident}$  [ $\text{of } a \ l$ ]  
**show**  $\text{proj2-incident } ?p \ ?m$

by (unfold ⟨?p = endpoint-in-S ?c a⟩) (simp add: endpoint-in-S-incident)  
qed

**lemma** *drop-perp-same-line-pole-in-S*:

assumes *drop-perp*  $p \ l = l$

shows *pole*  $l \in S$

**proof** –

from ⟨*drop-perp*  $p \ l = l$ ⟩

have  $l = \text{proj2-line-through } p \ (\text{pole } l)$  by (unfold *drop-perp-def*) simp

with *proj2-line-through-incident* [of *pole*  $l \ p$ ]

have *proj2-incident* (*pole*  $l$ )  $l$  by simp

hence *proj2-incident* (*pole*  $l$ ) (*polar* (*pole*  $l$ )) by (subst *polar-pole*)

thus *pole*  $l \in S$  by (unfold *incident-own-polar-in-S*)

qed

**lemma** *hyp2-drop-perp-not-same-line*:

assumes  $a \in \text{hyp2}$

shows *drop-perp*  $a \ l \neq l$

**proof**

assume *drop-perp*  $a \ l = l$

hence *pole*  $l \in S$  by (rule *drop-perp-same-line-pole-in-S*)

with ⟨ $a \in \text{hyp2}$ ⟩

have  $\neg \text{proj2-incident } a \ (\text{polar } (\text{pole } l))$

by (simp add: *tangent-not-through-K2*)

with ⟨*drop-perp*  $a \ l = l$ ⟩

have  $\neg \text{proj2-incident } a \ (\text{drop-perp } a \ l)$  by (simp add: *polar-pole*)

with *drop-perp-incident* [of  $a \ l$ ] show *False* by simp

qed

**lemma** *hyp2-incident-perp-foot-same-point*:

assumes  $a \in \text{hyp2}$  and *proj2-incident*  $a \ l$

shows *perp-foot*  $a \ l = a$

**proof** –

from ⟨ $a \in \text{hyp2}$ ⟩

have *drop-perp*  $a \ l \neq l$  by (rule *hyp2-drop-perp-not-same-line*)

with *perp-foot-incident* [of  $a \ l$ ] and ⟨*proj2-incident*  $a \ l$ ⟩

and *drop-perp-incident* [of  $a \ l$ ] and *proj2-incident-unique*

show *perp-foot*  $a \ l = a$  by fast

qed

**lemma** *perp-up-at-end*:

assumes  $a \in \text{hyp2}$  and  $b \in \text{hyp2}$  and *proj2-incident*  $b \ l$

shows  $B_{\mathbb{R}} \ (\text{cart2-pt } (\text{perp-foot } a \ l)) \ (\text{cart2-pt } a) \ (\text{cart2-pt } (\text{perp-up } a \ l))$

**proof** *cases*

assume *proj2-incident*  $a \ l$

with ⟨ $a \in \text{hyp2}$ ⟩

have *perp-foot*  $a \ l = a$  by (rule *hyp2-incident-perp-foot-same-point*)

thus  $B_{\mathbb{R}} \ (\text{cart2-pt } (\text{perp-foot } a \ l)) \ (\text{cart2-pt } a) \ (\text{cart2-pt } (\text{perp-up } a \ l))$

by (simp add: *real-euclid.th3-1* *real-euclid.th3-2*)

**next**  
**assume**  $\neg \text{proj2-incident } a \ l$   
**with** *assms*  
**show**  $B_{\mathbb{R}} (\text{cart2-pt } (\text{perp-foot } a \ l)) (\text{cart2-pt } a) (\text{cart2-pt } (\text{perp-up } a \ l))$   
**by** (*rule perp-up-non-degenerate-in-S-at-end*)  
**qed**

**definition** *perp-down* :: *proj2*  $\Rightarrow$  *proj2-line*  $\Rightarrow$  *proj2* **where**  
*perp-down*  $a \ l \triangleq \text{endpoint-in-S } (\text{perp-up } a \ l) \ a$

**lemma** *perp-down-in-S*:  
**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and** *proj2-incident*  $b \ l$   
**shows** *perp-down*  $a \ l \in S$   
**proof** –  
**from** *assms* **have** *perp-up*  $a \ l \in S$  **by** (*rule perp-up-in-S*)  
**with**  $\langle a \in \text{hyp2} \rangle$   
**show** *perp-down*  $a \ l \in S$  **by** (*unfold perp-down-def*) (*simp add: endpoint-in-S*)  
**qed**

**lemma** *perp-down-incident*:  
**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and** *proj2-incident*  $b \ l$   
**shows** *proj2-incident* (*perp-down*  $a \ l$ ) (*drop-perp*  $a \ l$ )  
**proof** –  
**from** *assms* **have** *perp-up*  $a \ l \in S$  **by** (*rule perp-up-in-S*)  
**with**  $\langle a \in \text{hyp2} \rangle$  **have** *perp-up*  $a \ l \neq a$  **by** (*rule hyp2-S-not-equal [symmetric]*)  
  
**from** *assms*  
**have** *proj2-incident* (*perp-up*  $a \ l$ ) (*drop-perp*  $a \ l$ ) **by** (*rule perp-up-incident*)  
**with**  $\langle \text{perp-up } a \ l \neq a \rangle$  **and**  $\langle \text{perp-up } a \ l \in S \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$   
**and** *drop-perp-incident [of a l]*  
**show** *proj2-incident* (*perp-down*  $a \ l$ ) (*drop-perp*  $a \ l$ )  
**by** (*unfold perp-down-def*) (*simp add: endpoint-in-S-incident*)  
**qed**

**lemma** *perp-up-down-distinct*:  
**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and** *proj2-incident*  $b \ l$   
**shows** *perp-up*  $a \ l \neq \text{perp-down } a \ l$   
**proof** –  
**from** *assms* **have** *perp-up*  $a \ l \in S$  **by** (*rule perp-up-in-S*)  
**with**  $\langle a \in \text{hyp2} \rangle$   
**show** *perp-up*  $a \ l \neq \text{perp-down } a \ l$   
**unfolding** *perp-down-def*  
**by** (*simp add: endpoint-in-S-S-strict-hyp2-distinct [symmetric]*)  
**qed**

**lemma** *perp-up-down-foot-are-endpoints-in-S*:  
**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and** *proj2-incident*  $b \ l$   
**shows** *are-endpoints-in-S* (*perp-up*  $a \ l$ ) (*perp-down*  $a \ l$ ) (*perp-foot*  $a \ l$ )  $a$   
**proof** –

**from**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } b \ l \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$   
**have**  $\text{perp-foot } a \ l \in \text{hyp2}$  **by** (rule  $\text{perp-foot-hyp2}$ )

**from** *assms* **have**  $\text{perp-up } a \ l \in S$  **by** (rule  $\text{perp-up-in-S}$ )

**from** *assms*  
**have**  $\text{proj2-incident } (\text{perp-up } a \ l) (\text{drop-perp } a \ l)$  **by** (rule  $\text{perp-up-incident}$ )  
**with**  $\langle \text{perp-foot } a \ l \in \text{hyp2} \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle \text{perp-up } a \ l \in S \rangle$   
**and**  $\text{perp-foot-incident}(2)$  [of  $a \ l$ ] **and**  $\text{drop-perp-incident}$  [of  $a \ l$ ]  
**show**  $\text{are-endpoints-in-S } (\text{perp-up } a \ l) (\text{perp-down } a \ l) (\text{perp-foot } a \ l) a$   
**by** ( $\text{unfold } \text{perp-down-def}$ ) (rule  $\text{end-and-opposite-are-endpoints-in-S}$ )

qed

**lemma**  $\text{perp-foot-opposite-endpoint-in-S}$ :  
**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and**  $c \in \text{hyp2}$  **and**  $a \neq b$   
**shows**  
 $\text{endpoint-in-S } (\text{endpoint-in-S } a \ b) (\text{perp-foot } c (\text{proj2-line-through } a \ b))$   
 $= \text{endpoint-in-S } b \ a$   
(is  $\text{endpoint-in-S } ?p \ ?d = \text{endpoint-in-S } b \ a$ )

**proof** –  
**let**  $?q = \text{endpoint-in-S } ?p \ ?d$

**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **have**  $?p \in S$  **by** ( $\text{simp add: endpoint-in-S}$ )

**let**  $?l = \text{proj2-line-through } a \ b$   
**have**  $\text{proj2-incident } a \ ?l$  **and**  $\text{proj2-incident } b \ ?l$   
**by** (rule  $\text{proj2-line-through-incident}$ )  
**with**  $\langle a \neq b \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$   
**have**  $\text{proj2-incident } ?p \ ?l$   
**by** ( $\text{simp-all add: endpoint-in-S-incident}$ )

**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } a \ ?l \rangle$  **and**  $\langle c \in \text{hyp2} \rangle$   
**have**  $?d \in \text{hyp2}$  **by** (rule  $\text{perp-foot-hyp2}$ )  
**with**  $\langle ?p \in S \rangle$  **have**  $?q \neq ?p$  **by** (rule  $\text{endpoint-in-S-S-strict-hyp2-distinct}$ )

**from**  $\langle ?p \in S \rangle$  **and**  $\langle ?d \in \text{hyp2} \rangle$  **have**  $?q \in S$  **by** ( $\text{simp add: endpoint-in-S}$ )

**from**  $\langle ?d \in \text{hyp2} \rangle$  **and**  $\langle ?p \in S \rangle$   
**have**  $?p \neq ?d$  **by** (rule  $\text{hyp2-S-not-equal}$  [symmetric])  
**with**  $\langle ?p \in S \rangle$  **and**  $\langle ?d \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } ?p \ ?l \rangle$   
**and**  $\text{perp-foot-incident}(1)$  [of  $c \ ?l$ ]  
**have**  $\text{proj2-incident } ?q \ ?l$  **by** ( $\text{simp add: endpoint-in-S-incident}$ )  
**with**  $\langle a \neq b \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle ?q \in S \rangle$   
**and**  $\langle \text{proj2-incident } a \ ?l \rangle$  **and**  $\langle \text{proj2-incident } b \ ?l \rangle$   
**have**  $?q = ?p \vee ?q = \text{endpoint-in-S } b \ a$   
**by** ( $\text{simp add: endpoints-in-S-incident-unique}$ )  
**with**  $\langle ?q \neq ?p \rangle$  **show**  $?q = \text{endpoint-in-S } b \ a$  **by** *simp*

qed

**lemma** *endpoints-in-S-perp-foot-are-endpoints-in-S*:  
**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and**  $c \in \text{hyp2}$  **and**  $a \neq b$   
**and** *proj2-incident*  $a$   $l$  **and** *proj2-incident*  $b$   $l$   
**shows** *are-endpoints-in-S*  
(*endpoint-in-S*  $a$   $b$ ) (*endpoint-in-S*  $b$   $a$ )  $a$  (*perp-foot*  $c$   $l$ )

**proof** –  
**define**  $p$   $q$   $d$   
**where**  $p = \text{endpoint-in-S } a$   $b$   
**and**  $q = \text{endpoint-in-S } b$   $a$   
**and**  $d = \text{perp-foot } c$   $l$

**from**  $\langle a \neq b \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$   
**have**  $p \neq q$  **by** (*unfold p-def q-def*) (*simp add: endpoint-in-S-swap*)

**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$   
**have**  $p \in S$  **and**  $q \in S$  **by** (*unfold p-def q-def*) (*simp-all add: endpoint-in-S*)

**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } a$   $l \rangle$  **and**  $\langle c \in \text{hyp2} \rangle$   
**have**  $d \in \text{hyp2}$  **by** (*unfold d-def*) (*rule perp-foot-hyp2*)

**from**  $\langle a \neq b \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } a$   $l \rangle$   
**and**  $\langle \text{proj2-incident } b$   $l \rangle$   
**have** *proj2-incident*  $p$   $l$  **and** *proj2-incident*  $q$   $l$   
**by** (*unfold p-def q-def*) (*simp-all add: endpoint-in-S-incident*)  
**with**  $\langle \text{proj2-incident } a$   $l \rangle$  **and** *perp-foot-incident*(1) [*of*  $c$   $l$ ]  
**have** *proj2-set-Col*  $\{p, q, a, d\}$   
**by** (*unfold d-def proj2-set-Col-def*) (*simp add: exI [of - l]*)  
**with**  $\langle p \neq q \rangle$  **and**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle d \in \text{hyp2} \rangle$   
**show** *are-endpoints-in-S*  $p$   $q$   $a$   $d$  **by** (*unfold are-endpoints-in-S-def*) *simp*  
**qed**

**definition** *right-angle* :: *proj2*  $\Rightarrow$  *proj2*  $\Rightarrow$  *proj2*  $\Rightarrow$  *bool* **where**  
*right-angle*  $p$   $a$   $q$   
 $\triangleq p \in S \wedge q \in S \wedge a \in \text{hyp2}$   
 $\wedge M\text{-perp } (\text{proj2-line-through } p$   $a)$  (*proj2-line-through*  $a$   $q$ )

**lemma** *perp-foot-up-right-angle*:  
**assumes**  $p \in S$  **and**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and** *proj2-incident*  $p$   $l$   
**and** *proj2-incident*  $b$   $l$   
**shows** *right-angle*  $p$  (*perp-foot*  $a$   $l$ ) (*perp-up*  $a$   $l$ )

**proof** –  
**define**  $c$  **where**  $c = \text{perp-foot } a$   $l$   
**define**  $q$  **where**  $q = \text{perp-up } a$   $l$   
**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } b$   $l \rangle$   
**have**  $q \in S$  **by** (*unfold q-def*) (*rule perp-up-in-S*)

**from**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } b$   $l \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$   
**have**  $c \in \text{hyp2}$  **by** (*unfold c-def*) (*rule perp-foot-hyp2*)  
**with**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$  **have**  $c \neq p$  **and**  $c \neq q$

**by** (*simp-all add: hyp2-S-not-equal*)

**from**  $\langle c \neq p \rangle$  [*symmetric*] **and**  $\langle \text{proj2-incident } p \ l \rangle$   
**and**  $\text{perp-foot-incident}(1)$  [*of a l*]  
**have**  $l = \text{proj2-line-through } p \ c$   
**by** (*unfold c-def*) (*rule proj2-line-through-unique*)

**define**  $m$  **where**  $m = \text{drop-perp } a \ l$   
**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } b \ l \rangle$   
**have**  $\text{proj2-incident } q \ m$  **by** (*unfold q-def m-def*) (*rule perp-up-incident*)  
**with**  $\langle c \neq q \rangle$  **and**  $\text{perp-foot-incident}(2)$  [*of a l*]  
**have**  $m = \text{proj2-line-through } c \ q$   
**by** (*unfold c-def m-def*) (*rule proj2-line-through-unique*)  
**with**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$  **and**  $\langle c \in \text{hyp2} \rangle$  **and**  $\text{drop-perp-perp}$  [*of l a*]  
**and**  $\langle l = \text{proj2-line-through } p \ c \rangle$   
**show**  $\text{right-angle } p$  ( $\text{perp-foot } a \ l$ ) ( $\text{perp-up } a \ l$ )  
**by** (*unfold right-angle-def q-def c-def m-def*) *simp*

**qed**

**lemma** *M-perp-unique*:  
**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and**  $\text{proj2-incident } a \ l$   
**and**  $\text{proj2-incident } b \ m$  **and**  $\text{proj2-incident } b \ n$  **and**  $M\text{-perp } l \ m$   
**and**  $M\text{-perp } l \ n$   
**shows**  $m = n$

**proof** –  
**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } a \ l \rangle$   
**have**  $\text{pole } l \notin \text{hyp2}$  **by** (*rule line-through-hyp2-pole-not-in-hyp2*)  
**with**  $\langle b \in \text{hyp2} \rangle$  **have**  $b \neq \text{pole } l$  **by** *auto*  
**with**  $\langle \text{proj2-incident } b \ m \rangle$  **and**  $\langle M\text{-perp } l \ m \rangle$  **and**  $\langle \text{proj2-incident } b \ n \rangle$   
**and**  $\langle M\text{-perp } l \ n \rangle$  **and**  $\text{proj2-incident-unique}$   
**show**  $m = n$  **by** (*unfold M-perp-def*) *auto*

**qed**

**lemma** *perp-foot-eq-implies-drop-perp-eq*:  
**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and**  $\text{proj2-incident } a \ l$   
**and**  $\text{perp-foot } b \ l = \text{perp-foot } c \ l$   
**shows**  $\text{drop-perp } b \ l = \text{drop-perp } c \ l$

**proof** –  
**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } a \ l \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$   
**have**  $\text{perp-foot } b \ l \in \text{hyp2}$  **by** (*rule perp-foot-hyp2*)

**from**  $\langle \text{perp-foot } b \ l = \text{perp-foot } c \ l \rangle$   
**have**  $\text{proj2-incident } (\text{perp-foot } b \ l)$  ( $\text{drop-perp } c \ l$ )  
**by** (*simp add: perp-foot-incident*)  
**with**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle \text{perp-foot } b \ l \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } a \ l \rangle$   
**and**  $\text{perp-foot-incident}(2)$  [*of b l*] **and**  $\text{drop-perp-perp}$  [*of l*]  
**show**  $\text{drop-perp } b \ l = \text{drop-perp } c \ l$  **by** (*simp add: M-perp-unique*)

**qed**

**lemma** *right-angle-to-compass*:  
**assumes** *right-angle p a q*  
**shows**  $\exists J. \text{is-K2-isometry } J \wedge \text{apply-cltn2 } p \ J = \text{east}$   
 $\wedge \text{apply-cltn2 } a \ J = \text{K2-centre} \wedge \text{apply-cltn2 } q \ J = \text{north}$   
**proof** –  
**from**  $\langle \text{right-angle } p \ a \ q \rangle$   
**have**  $p \in S$  **and**  $q \in S$  **and**  $a \in \text{hyp2}$   
**and**  $M\text{-perp } (\text{proj2-line-through } p \ a) \ (\text{proj2-line-through } a \ q)$   
**(is**  $M\text{-perp } ?l \ ?m)$   
**by**  $(\text{unfold right-angle-def}) \ \text{simp-all}$   
  
**have**  $\text{proj2-incident } p \ ?l$  **and**  $\text{proj2-incident } a \ ?l$   
**and**  $\text{proj2-incident } q \ ?m$  **and**  $\text{proj2-incident } a \ ?m$   
**by**  $(\text{rule proj2-line-through-incident})+$   
  
**from**  $\langle M\text{-perp } ?l \ ?m \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } a \ ?l \rangle$   
**and**  $\langle \text{proj2-incident } a \ ?m \rangle$  **and**  $M\text{-perp-to-compass } [\text{of } ?l \ ?m \ a \ a]$   
**obtain**  $J''i$  **where**  $\text{is-K2-isometry } J''i$   
**and**  $\text{apply-cltn2-line equator } J''i = ?l$   
**and**  $\text{apply-cltn2-line meridian } J''i = ?m$   
**by** *auto*  
**let**  $?J'' = \text{cltn2-inverse } J''i$   
  
**from**  $\langle \text{apply-cltn2-line equator } J''i = ?l \rangle$   
**and**  $\langle \text{apply-cltn2-line meridian } J''i = ?m \rangle$   
**and**  $\langle \text{proj2-incident } p \ ?l \rangle$  **and**  $\langle \text{proj2-incident } a \ ?l \rangle$   
**and**  $\langle \text{proj2-incident } q \ ?m \rangle$  **and**  $\langle \text{proj2-incident } a \ ?m \rangle$   
**have**  $\text{proj2-incident } (\text{apply-cltn2 } p \ ?J'') \ \text{equator}$   
**and**  $\text{proj2-incident } (\text{apply-cltn2 } a \ ?J'') \ \text{equator}$   
**and**  $\text{proj2-incident } (\text{apply-cltn2 } q \ ?J'') \ \text{meridian}$   
**and**  $\text{proj2-incident } (\text{apply-cltn2 } a \ ?J'') \ \text{meridian}$   
**by**  $(\text{simp-all add: apply-cltn2-incident } [\text{symmetric}])$   
  
**from**  $\langle \text{proj2-incident } (\text{apply-cltn2 } a \ ?J'') \ \text{equator} \rangle$   
**and**  $\langle \text{proj2-incident } (\text{apply-cltn2 } a \ ?J'') \ \text{meridian} \rangle$   
**have**  $\text{apply-cltn2 } a \ ?J'' = \text{K2-centre}$   
**by**  $(\text{rule on-equator-meridian-is-K2-centre})$   
  
**from**  $\langle \text{is-K2-isometry } J''i \rangle$   
**have**  $\text{is-K2-isometry } ?J''$  **by**  $(\text{rule cltn2-inverse-is-K2-isometry})$   
**with**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$   
**have**  $\text{apply-cltn2 } p \ ?J'' \in S$  **and**  $\text{apply-cltn2 } q \ ?J'' \in S$   
**by**  $(\text{unfold is-K2-isometry-def}) \ \text{simp-all}$   
**with**  $\text{east-west-distinct}$  **and**  $\text{north-south-distinct}$  **and**  $\text{compass-in-S}$   
**and**  $\text{east-west-on-equator}$  **and**  $\text{north-south-far-north-on-meridian}$   
**and**  $\langle \text{proj2-incident } (\text{apply-cltn2 } p \ ?J'') \ \text{equator} \rangle$   
**and**  $\langle \text{proj2-incident } (\text{apply-cltn2 } q \ ?J'') \ \text{meridian} \rangle$   
**have**  $\text{apply-cltn2 } p \ ?J'' = \text{east} \vee \text{apply-cltn2 } p \ ?J'' = \text{west}$   
**and**  $\text{apply-cltn2 } q \ ?J'' = \text{north} \vee \text{apply-cltn2 } q \ ?J'' = \text{south}$

by (*simp-all add: line-S-two-intersections-only*)

**have**  $\exists J'. \text{is-K2-isometry } J' \wedge \text{apply-cltn2 } p \ J' = \text{east}$   
 $\wedge \text{apply-cltn2 } a \ J' = \text{K2-centre}$   
 $\wedge (\text{apply-cltn2 } q \ J' = \text{north} \vee \text{apply-cltn2 } q \ J' = \text{south})$

**proof cases**

**assume**  $\text{apply-cltn2 } p \ ?J'' = \text{east}$   
**with**  $\langle \text{is-K2-isometry } ?J'' \rangle$  **and**  $\langle \text{apply-cltn2 } a \ ?J'' = \text{K2-centre} \rangle$   
**and**  $\langle \text{apply-cltn2 } q \ ?J'' = \text{north} \vee \text{apply-cltn2 } q \ ?J'' = \text{south} \rangle$   
**show**  $\exists J'. \text{is-K2-isometry } J' \wedge \text{apply-cltn2 } p \ J' = \text{east}$   
 $\wedge \text{apply-cltn2 } a \ J' = \text{K2-centre}$   
 $\wedge (\text{apply-cltn2 } q \ J' = \text{north} \vee \text{apply-cltn2 } q \ J' = \text{south})$   
**by** (*simp add: exI [of - ?J'']*)

**next**

**assume**  $\text{apply-cltn2 } p \ ?J'' \neq \text{east}$   
**with**  $\langle \text{apply-cltn2 } p \ ?J'' = \text{east} \vee \text{apply-cltn2 } p \ ?J'' = \text{west} \rangle$   
**have**  $\text{apply-cltn2 } p \ ?J'' = \text{west}$  **by** *simp*

**let**  $?J' = \text{cltn2-compose } ?J'' \ \text{meridian-reflect}$   
**from**  $\langle \text{is-K2-isometry } ?J'' \rangle$  **and**  $\langle \text{meridian-reflect-K2-isometry} \rangle$   
**have**  $\text{is-K2-isometry } ?J'$  **by** (*rule cltn2-compose-is-K2-isometry*)  
**moreover**  
**from**  $\langle \text{apply-cltn2 } p \ ?J'' = \text{west} \rangle$  **and**  $\langle \text{apply-cltn2 } a \ ?J'' = \text{K2-centre} \rangle$   
**and**  $\langle \text{apply-cltn2 } q \ ?J'' = \text{north} \vee \text{apply-cltn2 } q \ ?J'' = \text{south} \rangle$   
**and** *compass-reflect-compass*  
**have**  $\text{apply-cltn2 } p \ ?J' = \text{east}$  **and**  $\text{apply-cltn2 } a \ ?J' = \text{K2-centre}$   
**and**  $\text{apply-cltn2 } q \ ?J' = \text{north} \vee \text{apply-cltn2 } q \ ?J' = \text{south}$   
**by** (*auto simp add: cltn2.act-act [simplified, symmetric]*)  
**ultimately**  
**show**  $\exists J'. \text{is-K2-isometry } J' \wedge \text{apply-cltn2 } p \ J' = \text{east}$   
 $\wedge \text{apply-cltn2 } a \ J' = \text{K2-centre}$   
 $\wedge (\text{apply-cltn2 } q \ J' = \text{north} \vee \text{apply-cltn2 } q \ J' = \text{south})$   
**by** (*simp add: exI [of - ?J']*)

**qed**

**then obtain**  $J'$  **where**  $\text{is-K2-isometry } J'$  **and**  $\text{apply-cltn2 } p \ J' = \text{east}$   
**and**  $\text{apply-cltn2 } a \ J' = \text{K2-centre}$   
**and**  $\text{apply-cltn2 } q \ J' = \text{north} \vee \text{apply-cltn2 } q \ J' = \text{south}$   
**by** *auto*

**show**  $\exists J. \text{is-K2-isometry } J \wedge \text{apply-cltn2 } p \ J = \text{east}$   
 $\wedge \text{apply-cltn2 } a \ J = \text{K2-centre} \wedge \text{apply-cltn2 } q \ J = \text{north}$

**proof cases**

**assume**  $\text{apply-cltn2 } q \ J' = \text{north}$   
**with**  $\langle \text{is-K2-isometry } J' \rangle$  **and**  $\langle \text{apply-cltn2 } p \ J' = \text{east} \rangle$   
**and**  $\langle \text{apply-cltn2 } a \ J' = \text{K2-centre} \rangle$   
**show**  $\exists J. \text{is-K2-isometry } J \wedge \text{apply-cltn2 } p \ J = \text{east}$   
 $\wedge \text{apply-cltn2 } a \ J = \text{K2-centre} \wedge \text{apply-cltn2 } q \ J = \text{north}$   
**by** (*simp add: exI [of - J']*)

**next**

**assume**  $\text{apply-cltn2 } q \ J' \neq \text{north}$   
**with**  $\langle \text{apply-cltn2 } q \ J' = \text{north} \vee \text{apply-cltn2 } q \ J' = \text{south} \rangle$   
**have**  $\text{apply-cltn2 } q \ J' = \text{south}$  **by** *simp*

**let**  $?J = \text{cltn2-compose } J' \ \text{equator-reflect}$   
**from**  $\langle \text{is-K2-isometry } J' \rangle$  **and**  $\text{equator-reflect-K2-isometry}$   
**have**  $\text{is-K2-isometry } ?J$  **by** (*rule cltn2-compose-is-K2-isometry*)  
**moreover**  
**from**  $\langle \text{apply-cltn2 } p \ J' = \text{east} \rangle$  **and**  $\langle \text{apply-cltn2 } a \ J' = \text{K2-centre} \rangle$   
**and**  $\langle \text{apply-cltn2 } q \ J' = \text{south} \rangle$  **and**  $\text{compass-reflect-compass}$   
**have**  $\text{apply-cltn2 } p \ ?J = \text{east}$  **and**  $\text{apply-cltn2 } a \ ?J = \text{K2-centre}$   
**and**  $\text{apply-cltn2 } q \ ?J = \text{north}$   
**by** (*auto simp add: cltn2.act-act [simplified, symmetric]*)  
**ultimately**  
**show**  $\exists J. \text{is-K2-isometry } J \wedge \text{apply-cltn2 } p \ J = \text{east}$   
 $\wedge \text{apply-cltn2 } a \ J = \text{K2-centre} \wedge \text{apply-cltn2 } q \ J = \text{north}$   
**by** (*simp add: exI [of - ?J]*)

**qed**  
**qed**

**lemma** *right-angle-to-right-angle*:

**assumes**  $\text{right-angle } p \ a \ q$  **and**  $\text{right-angle } r \ b \ s$   
**shows**  $\exists J. \text{is-K2-isometry } J$

$\wedge \text{apply-cltn2 } p \ J = r \wedge \text{apply-cltn2 } a \ J = b \wedge \text{apply-cltn2 } q \ J = s$

**proof** –

**from**  $\langle \text{right-angle } p \ a \ q \rangle$  **and**  $\text{right-angle-to-compass [of } p \ a \ q]$   
**obtain**  $H$  **where**  $\text{is-K2-isometry } H$  **and**  $\text{apply-cltn2 } p \ H = \text{east}$   
**and**  $\text{apply-cltn2 } a \ H = \text{K2-centre}$  **and**  $\text{apply-cltn2 } q \ H = \text{north}$   
**by** *auto*

**from**  $\langle \text{right-angle } r \ b \ s \rangle$  **and**  $\text{right-angle-to-compass [of } r \ b \ s]$   
**obtain**  $K$  **where**  $\text{is-K2-isometry } K$  **and**  $\text{apply-cltn2 } r \ K = \text{east}$   
**and**  $\text{apply-cltn2 } b \ K = \text{K2-centre}$  **and**  $\text{apply-cltn2 } s \ K = \text{north}$   
**by** *auto*

**let**  $?Ki = \text{cltn2-inverse } K$   
**let**  $?J = \text{cltn2-compose } H \ ?Ki$   
**from**  $\langle \text{is-K2-isometry } H \rangle$  **and**  $\langle \text{is-K2-isometry } K \rangle$   
**have**  $\text{is-K2-isometry } ?J$   
**by** (*simp add: cltn2-inverse-is-K2-isometry cltn2-compose-is-K2-isometry*)

**from**  $\langle \text{apply-cltn2 } r \ K = \text{east} \rangle$  **and**  $\langle \text{apply-cltn2 } b \ K = \text{K2-centre} \rangle$   
**and**  $\langle \text{apply-cltn2 } s \ K = \text{north} \rangle$   
**have**  $\text{apply-cltn2 } \text{east } ?Ki = r$  **and**  $\text{apply-cltn2 } \text{K2-centre } ?Ki = b$   
**and**  $\text{apply-cltn2 } \text{north } ?Ki = s$   
**by** (*simp-all add: cltn2.act-inv-iff [simplified]*)  
**with**  $\langle \text{apply-cltn2 } p \ H = \text{east} \rangle$  **and**  $\langle \text{apply-cltn2 } a \ H = \text{K2-centre} \rangle$   
**and**  $\langle \text{apply-cltn2 } q \ H = \text{north} \rangle$   
**have**  $\text{apply-cltn2 } p \ ?J = r$  **and**  $\text{apply-cltn2 } a \ ?J = b$

**and**  $\text{apply-cltn2 } q \ ?J = s$   
**by** ( $\text{simp-all add: cltn2.act-act [simplified,symmetric]}$ )  
**with**  $\langle \text{is-K2-isometry } ?J \rangle$   
**show**  $\exists J. \text{is-K2-isometry } J$   
 $\wedge \text{apply-cltn2 } p \ J = r \wedge \text{apply-cltn2 } a \ J = b \wedge \text{apply-cltn2 } q \ J = s$   
**by** ( $\text{simp add: exI [of - ?J]}$ )  
**qed**

## 8.11 Functions of distance

**definition**  $\text{exp-2dist} :: \text{proj2} \Rightarrow \text{proj2} \Rightarrow \text{real}$  **where**  
 $\text{exp-2dist } a \ b$   
 $\triangleq \text{if } a = b$   
 $\text{then } 1$   
 $\text{else cross-ratio (endpoint-in-S } a \ b) \ (\text{endpoint-in-S } b \ a) \ a \ b$

**definition**  $\text{cosh-dist} :: \text{proj2} \Rightarrow \text{proj2} \Rightarrow \text{real}$  **where**  
 $\text{cosh-dist } a \ b \triangleq (\text{sqrt } (\text{exp-2dist } a \ b) + \text{sqrt } (1 / (\text{exp-2dist } a \ b))) / 2$

**lemma**  $\text{exp-2dist-formula}$ :

**assumes**  $a \neq 0$  **and**  $b \neq 0$  **and**  $\text{proj2-abs } a \in \text{hyp2}$  (**is**  $?pa \in \text{hyp2}$ )  
**and**  $\text{proj2-abs } b \in \text{hyp2}$  (**is**  $?pb \in \text{hyp2}$ )  
**shows**  $\text{exp-2dist } (\text{proj2-abs } a) \ (\text{proj2-abs } b)$   
 $= (a \cdot (M *v b) + \text{sqrt } (\text{quarter-discrim } a \ b))$   
 $/ (a \cdot (M *v b) - \text{sqrt } (\text{quarter-discrim } a \ b))$   
 $\vee \text{exp-2dist } (\text{proj2-abs } a) \ (\text{proj2-abs } b)$   
 $= (a \cdot (M *v b) - \text{sqrt } (\text{quarter-discrim } a \ b))$   
 $/ (a \cdot (M *v b) + \text{sqrt } (\text{quarter-discrim } a \ b))$   
**(is**  $?e2d = (?aMb + ?sqd) / (?aMb - ?sqd)$   
 $\vee ?e2d = (?aMb - ?sqd) / (?aMb + ?sqd)$ )

**proof cases**

**assume**  $?pa = ?pb$   
**hence**  $?e2d = 1$  **by** ( $\text{unfold exp-2dist-def, simp}$ )

**from**  $\langle ?pa = ?pb \rangle$   
**have**  $\text{quarter-discrim } a \ b = 0$  **by** ( $\text{rule quarter-discrim-self-zero}$ )  
**hence**  $?sqd = 0$  **by**  $\text{simp}$

**from**  $\langle \text{proj2-abs } a = \text{proj2-abs } b \rangle$  **and**  $\langle b \neq 0 \rangle$  **and**  $\text{proj2-abs-abs-mult}$   
**obtain**  $k$  **where**  $a = k *_R b$  **by**  $\text{auto}$

**from**  $\langle b \neq 0 \rangle$  **and**  $\langle \text{proj2-abs } b \in \text{hyp2} \rangle$   
**have**  $b \cdot (M *v b) < 0$  **by** ( $\text{subst K2-abs [symmetric]}$ )  
**with**  $\langle a \neq 0 \rangle$  **and**  $\langle a = k *_R b \rangle$  **have**  $?aMb \neq 0$  **by**  $\text{simp}$   
**with**  $\langle ?e2d = 1 \rangle$  **and**  $\langle ?sqd = 0 \rangle$   
**show**  $?e2d = (?aMb + ?sqd) / (?aMb - ?sqd)$   
 $\vee ?e2d = (?aMb - ?sqd) / (?aMb + ?sqd)$   
**by**  $\text{simp}$

**next**

**assume**  $?pa \neq ?pb$   
**let**  $?l = \text{proj2-line-through } ?pa \ ?pb$   
**have**  $\text{proj2-incident } ?pa \ ?l$  **and**  $\text{proj2-incident } ?pb \ ?l$   
**by** (rule  $\text{proj2-line-through-incident}$ )  
**with**  $\langle a \neq 0 \rangle$  **and**  $\langle b \neq 0 \rangle$  **and**  $\langle ?pa \neq ?pb \rangle$   
**have**  $\text{proj2-incident } (S\text{-intersection1 } a \ b) \ ?l$  (**is**  $\text{proj2-incident } ?Si1 \ ?l$ )  
**and**  $\text{proj2-incident } (S\text{-intersection2 } a \ b) \ ?l$  (**is**  $\text{proj2-incident } ?Si2 \ ?l$ )  
**by** (rule  $S\text{-intersections-incident}$ )  
**with**  $\langle \text{proj2-incident } ?pa \ ?l \rangle$  **and**  $\langle \text{proj2-incident } ?pb \ ?l \rangle$   
**have**  $\text{proj2-set-Col } \{?pa, ?pb, ?Si1, ?Si2\}$  **by** (unfold  $\text{proj2-set-Col-def}$ , auto)

**have**  $\{?pa, ?pb, ?Si2, ?Si1\} = \{?pa, ?pb, ?Si1, ?Si2\}$  **by** auto

**from**  $\langle a \neq 0 \rangle$  **and**  $\langle b \neq 0 \rangle$  **and**  $\langle ?pa \neq ?pb \rangle$  **and**  $\langle ?pa \in \text{hyp2} \rangle$   
**have**  $?Si1 \in S$  **and**  $?Si2 \in S$   
**by** (simp-all add:  $S\text{-intersections-in-S}$ )  
**with**  $\langle ?pa \in \text{hyp2} \rangle$  **and**  $\langle ?pb \in \text{hyp2} \rangle$   
**have**  $?Si1 \neq ?pa$  **and**  $?Si2 \neq ?pa$  **and**  $?Si1 \neq ?pb$  **and**  $?Si2 \neq ?pb$   
**by** (simp-all add:  $\text{hyp2-S-not-equal}$  [symmetric])  
**with**  $\langle \text{proj2-set-Col } \{?pa, ?pb, ?Si1, ?Si2\} \rangle$  **and**  $\langle ?pa \neq ?pb \rangle$   
**have**  $\text{cross-ratio-correct } ?pa \ ?pb \ ?Si1 \ ?Si2$   
**and**  $\text{cross-ratio-correct } ?pa \ ?pb \ ?Si2 \ ?Si1$   
**unfolding**  $\text{cross-ratio-correct-def}$   
**by** (simp-all add:  $\langle \{?pa, ?pb, ?Si2, ?Si1\} = \{?pa, ?pb, ?Si1, ?Si2\} \rangle$ )

**from**  $\langle a \neq 0 \rangle$  **and**  $\langle b \neq 0 \rangle$  **and**  $\langle ?pa \neq ?pb \rangle$  **and**  $\langle ?pa \in \text{hyp2} \rangle$   
**have**  $?Si1 \neq ?Si2$  **by** (simp add:  $S\text{-intersections-distinct}$ )  
**with**  $\langle \text{cross-ratio-correct } ?pa \ ?pb \ ?Si1 \ ?Si2 \rangle$   
**and**  $\langle \text{cross-ratio-correct } ?pa \ ?pb \ ?Si2 \ ?Si1 \rangle$   
**have**  $\text{cross-ratio } ?Si1 \ ?Si2 \ ?pa \ ?pb = \text{cross-ratio } ?pa \ ?pb \ ?Si1 \ ?Si2$   
**and**  $\text{cross-ratio } ?Si2 \ ?Si1 \ ?pa \ ?pb = \text{cross-ratio } ?pa \ ?pb \ ?Si2 \ ?Si1$   
**by** (simp-all add:  $\text{cross-ratio-swap-13-24}$ )

**from**  $\langle a \neq 0 \rangle$  **and**  $\langle \text{proj2-abs } a \in \text{hyp2} \rangle$   
**have**  $a \cdot (M *v a) < 0$  **by** (subst  $K2\text{-abs}$  [symmetric])  
**with**  $\langle a \neq 0 \rangle$  **and**  $\langle b \neq 0 \rangle$  **and**  $\langle ?pa \neq ?pb \rangle$  **and**  $\text{cross-ratio-abs}$  [of  $a \ b \ 1 \ 1$ ]  
**have**  $\text{cross-ratio } ?pa \ ?pb \ ?Si1 \ ?Si2 = (-?aMb - ?sqd) / (-?aMb + ?sqd)$   
**by** (unfold  $S\text{-intersections-defs } S\text{-intersection-coeffs-defs}$ , simp)  
**with**  $\text{times-divide-times-eq}$  [of  $-1 \ -1 \ -?aMb - ?sqd \ -?aMb + ?sqd$ ]  
**have**  $\text{cross-ratio } ?pa \ ?pb \ ?Si1 \ ?Si2 = (?aMb + ?sqd) / (?aMb - ?sqd)$  **by** (simp  
add:  $\text{ac-simps}$ )  
**with**  $\langle \text{cross-ratio } ?Si1 \ ?Si2 \ ?pa \ ?pb = \text{cross-ratio } ?pa \ ?pb \ ?Si1 \ ?Si2 \rangle$   
**have**  $\text{cross-ratio } ?Si1 \ ?Si2 \ ?pa \ ?pb = (?aMb + ?sqd) / (?aMb - ?sqd)$  **by** simp

**from**  $\langle \text{cross-ratio } ?pa \ ?pb \ ?Si1 \ ?Si2 = (?aMb + ?sqd) / (?aMb - ?sqd) \rangle$   
**and**  $\text{cross-ratio-swap-34}$  [of  $?pa \ ?pb \ ?Si2 \ ?Si1$ ]  
**have**  $\text{cross-ratio } ?pa \ ?pb \ ?Si2 \ ?Si1 = (?aMb - ?sqd) / (?aMb + ?sqd)$  **by** simp  
**with**  $\langle \text{cross-ratio } ?Si2 \ ?Si1 \ ?pa \ ?pb = \text{cross-ratio } ?pa \ ?pb \ ?Si2 \ ?Si1 \rangle$   
**have**  $\text{cross-ratio } ?Si2 \ ?Si1 \ ?pa \ ?pb = (?aMb - ?sqd) / (?aMb + ?sqd)$  **by** simp

**from**  $\langle a \neq 0 \rangle$  **and**  $\langle b \neq 0 \rangle$  **and**  $\langle ?pa \neq ?pb \rangle$  **and**  $\langle ?pa \in \text{hyp2} \rangle$  **and**  $\langle ?pb \in \text{hyp2} \rangle$   
**have**  $(?Si1 = \text{endpoint-in-S } ?pa ?pb \wedge ?Si2 = \text{endpoint-in-S } ?pb ?pa)$   
 $\vee (?Si2 = \text{endpoint-in-S } ?pa ?pb \wedge ?Si1 = \text{endpoint-in-S } ?pb ?pa)$   
**by** (*simp add: S-intersections-endpoints-in-S*)  
**with**  $\langle \text{cross-ratio } ?Si1 ?Si2 ?pa ?pb = (?aMb + ?sqd) / (?aMb - ?sqd) \rangle$   
**and**  $\langle \text{cross-ratio } ?Si2 ?Si1 ?pa ?pb = (?aMb - ?sqd) / (?aMb + ?sqd) \rangle$   
**and**  $\langle ?pa \neq ?pb \rangle$   
**show**  $?e2d = (?aMb + ?sqd) / (?aMb - ?sqd)$   
 $\vee ?e2d = (?aMb - ?sqd) / (?aMb + ?sqd)$   
**by** (*unfold exp-2dist-def, auto*)  
**qed**

**lemma** *cosh-dist-formula*:

**assumes**  $a \neq 0$  **and**  $b \neq 0$  **and** *proj2-abs*  $a \in \text{hyp2}$  (**is**  $?pa \in \text{hyp2}$ )  
**and** *proj2-abs*  $b \in \text{hyp2}$  (**is**  $?pb \in \text{hyp2}$ )  
**shows** *cosh-dist* (*proj2-abs*  $a$ ) (*proj2-abs*  $b$ )  
 $= |a \cdot (M *v b)| / \text{sqrt } (a \cdot (M *v a) * (b \cdot (M *v b)))$   
(**is** *cosh-dist*  $?pa ?pb = |?aMb| / \text{sqrt } (?aMa * ?bMb)$ )

**proof** –

**let**  $?qd = \text{quarter-discrim } a b$   
**let**  $?sqd = \text{sqrt } ?qd$   
**let**  $?e2d = \text{exp-2dist } ?pa ?pb$   
**from** *assms*  
**have**  $?e2d = (?aMb + ?sqd) / (?aMb - ?sqd)$   
 $\vee ?e2d = (?aMb - ?sqd) / (?aMb + ?sqd)$   
**by** (*rule exp-2dist-formula*)  
**hence** *cosh-dist*  $?pa ?pb$   
 $= (\text{sqrt } ((?aMb + ?sqd) / (?aMb - ?sqd))$   
 $+ \text{sqrt } ((?aMb - ?sqd) / (?aMb + ?sqd)))$   
 $/ 2$   
**by** (*unfold cosh-dist-def, auto*)

**have**  $?qd \geq 0$

**proof** *cases*

**assume**  $?pa = ?pb$   
**thus**  $?qd \geq 0$  **by** (*simp add: quarter-discrim-self-zero*)  
**next**  
**assume**  $?pa \neq ?pb$   
**with**  $\langle a \neq 0 \rangle$  **and**  $\langle b \neq 0 \rangle$  **and**  $\langle ?pa \in \text{hyp2} \rangle$   
**have**  $?qd > 0$  **by** (*simp add: quarter-discrim-positive*)  
**thus**  $?qd \geq 0$  **by** *simp*

**qed**

**with** *real-sqrt-pow2* [*of*  $?qd$ ] **have**  $?sqd^2 = ?qd$  **by** *simp*

**hence**  $(?aMb + ?sqd) * (?aMb - ?sqd) = ?aMa * ?bMb$

**by** (*unfold quarter-discrim-def, simp add: algebra-simps power2-eq-square*)

**from** *times-divide-times-eq* [*of*

$?aMb + ?sqd$   $?aMb + ?sqd$   $?aMb + ?sqd$   $?aMb - ?sqd$   
**have**  $(?aMb + ?sqd) / (?aMb - ?sqd)$   
 $= (?aMb + ?sqd)^2 / ((?aMb + ?sqd) * (?aMb - ?sqd))$   
**by** (*simp add: power2-eq-square*)  
**with**  $\langle (?aMb + ?sqd) * (?aMb - ?sqd) = ?aMa * ?bMb \rangle$   
**have**  $(?aMb + ?sqd) / (?aMb - ?sqd) = (?aMb + ?sqd)^2 / (?aMa * ?bMb)$  **by**  
*simp*  
**hence**  $\sqrt{(?aMb + ?sqd) / (?aMb - ?sqd)}$   
 $= |?aMb + ?sqd| / \sqrt{?aMa * ?bMb}$   
**by** (*simp add: real-sqrt-divide*)

**from** *times-divide-times-eq* [of  
 $?aMb + ?sqd$   $?aMb - ?sqd$   $?aMb - ?sqd$   $?aMb - ?sqd$ ]  
**have**  $(?aMb - ?sqd) / (?aMb + ?sqd)$   
 $= (?aMb - ?sqd)^2 / ((?aMb + ?sqd) * (?aMb - ?sqd))$   
**by** (*simp add: power2-eq-square*)  
**with**  $\langle (?aMb + ?sqd) * (?aMb - ?sqd) = ?aMa * ?bMb \rangle$   
**have**  $(?aMb - ?sqd) / (?aMb + ?sqd) = (?aMb - ?sqd)^2 / (?aMa * ?bMb)$  **by**  
*simp*  
**hence**  $\sqrt{(?aMb - ?sqd) / (?aMb + ?sqd)}$   
 $= |?aMb - ?sqd| / \sqrt{?aMa * ?bMb}$   
**by** (*simp add: real-sqrt-divide*)

**from**  $\langle a \neq 0 \rangle$  **and**  $\langle b \neq 0 \rangle$  **and**  $\langle ?pa \in hyp2 \rangle$  **and**  $\langle ?pb \in hyp2 \rangle$   
**have**  $?aMa < 0$  **and**  $?bMb < 0$   
**by** (*simp-all add: K2-imp-M-neg*)  
**with**  $\langle (?aMb + ?sqd) * (?aMb - ?sqd) = ?aMa * ?bMb \rangle$   
**have**  $(?aMb + ?sqd) * (?aMb - ?sqd) > 0$  **by** (*simp add: mult-neg-neg*)  
**hence**  $?aMb + ?sqd \neq 0$  **and**  $?aMb - ?sqd \neq 0$  **by** *auto*  
**hence**  $\text{sgn} (?aMb + ?sqd) \in \{-1, 1\}$  **and**  $\text{sgn} (?aMb - ?sqd) \in \{-1, 1\}$   
**by** (*simp-all add: sgn-real-def*)

**from**  $\langle (?aMb + ?sqd) * (?aMb - ?sqd) > 0 \rangle$   
**have**  $\text{sgn} ((?aMb + ?sqd) * (?aMb - ?sqd)) = 1$  **by** *simp*  
**hence**  $\text{sgn} (?aMb + ?sqd) * \text{sgn} (?aMb - ?sqd) = 1$  **by** (*simp add: sgn-mult*)  
**with**  $\langle \text{sgn} (?aMb + ?sqd) \in \{-1, 1\} \rangle$  **and**  $\langle \text{sgn} (?aMb - ?sqd) \in \{-1, 1\} \rangle$   
**have**  $\text{sgn} (?aMb + ?sqd) = \text{sgn} (?aMb - ?sqd)$  **by** *auto*  
**with** *abs-plus* [of  $?aMb + ?sqd$   $?aMb - ?sqd$ ]  
**have**  $|?aMb + ?sqd| + |?aMb - ?sqd| = 2 * |?aMb|$  **by** *simp*  
**with**  $\langle \sqrt{(?aMb + ?sqd) / (?aMb - ?sqd)}$   
 $= |?aMb + ?sqd| / \sqrt{?aMa * ?bMb} \rangle$   
**and**  $\langle \sqrt{(?aMb - ?sqd) / (?aMb + ?sqd)}$   
 $= |?aMb - ?sqd| / \sqrt{?aMa * ?bMb} \rangle$   
**and** *add-divide-distrib* [of  
 $|?aMb + ?sqd|$   $|?aMb - ?sqd|$   $\sqrt{?aMa * ?bMb}$ ]  
**have**  $\sqrt{(?aMb + ?sqd) / (?aMb - ?sqd)}$   
 $+ \sqrt{(?aMb - ?sqd) / (?aMb + ?sqd)}$   
 $= 2 * |?aMb| / \sqrt{?aMa * ?bMb}$   
**by** *simp*

**with**  $\langle \text{cosh-dist } ?pa \ ?pb$   
 $= (\text{sqrt } ((?aMb + ?sqd) / (?aMb - ?sqd))$   
 $+ \text{sqrt } ((?aMb - ?sqd) / (?aMb + ?sqd)))$   
 $/ 2 \rangle$   
**show**  $\text{cosh-dist } ?pa \ ?pb = |?aMb| / \text{sqrt } (?aMa * ?bMb)$  **by** *simp*  
**qed**

**lemma** *cosh-dist-perp-special-case:*

**assumes**  $|x| < 1$  **and**  $|y| < 1$   
**shows**  $\text{cosh-dist } (\text{proj2-abs } (\text{vector } [x,0,1])) (\text{proj2-abs } (\text{vector } [0,y,1]))$   
 $= (\text{cosh-dist } K2\text{-centre } (\text{proj2-abs } (\text{vector } [x,0,1])))$   
 $* (\text{cosh-dist } K2\text{-centre } (\text{proj2-abs } (\text{vector } [0,y,1])))$   
**(is**  $\text{cosh-dist } ?pa \ ?pb = (\text{cosh-dist } ?po \ ?pa) * (\text{cosh-dist } ?po \ ?pb)$

**proof** –

**have**  $\text{vector } [x,0,1] \neq (0::\text{real}^3)$  **(is**  $?a \neq 0$ )  
**and**  $\text{vector } [0,y,1] \neq (0::\text{real}^3)$  **(is**  $?b \neq 0$ )  
**by** (*unfold vector-def, simp-all add: vec-eq-iff forall-3*)

**have**  $?a \cdot (M *v ?a) = x^2 - 1$  **(is**  $?aMa = x^2 - 1$ )  
**and**  $?b \cdot (M *v ?b) = y^2 - 1$  **(is**  $?bMb = y^2 - 1$ )  
**unfolding** *vector-def and M-def and inner-vec-def*  
**and** *matrix-vector-mult-def*

**by** (*simp-all add: sum-3 power2-eq-square*)

**with**  $\langle |x| < 1 \rangle$  **and**  $\langle |y| < 1 \rangle$

**have**  $?aMa < 0$  **and**  $?bMb < 0$  **by** (*simp-all add: abs-square-less-1*)

**hence**  $?pa \in \text{hyp2}$  **and**  $?pb \in \text{hyp2}$

**by** (*simp-all add: M-neg-imp-K2*)

**with**  $\langle ?a \neq 0 \rangle$  **and**  $\langle ?b \neq 0 \rangle$

**have**  $\text{cosh-dist } ?pa \ ?pb = |?a \cdot (M *v ?b)| / \text{sqrt } (?aMa * ?bMb)$   
**(is**  $\text{cosh-dist } ?pa \ ?pb = |?aMb| / \text{sqrt } (?aMa * ?bMb)$

**by** (*rule cosh-dist-formula*)

**also from**  $\langle ?aMa = x^2 - 1 \rangle$  **and**  $\langle ?bMb = y^2 - 1 \rangle$

**have**  $\dots = |?aMb| / \text{sqrt } ((x^2 - 1) * (y^2 - 1))$  **by** *simp*

**finally have**  $\text{cosh-dist } ?pa \ ?pb = 1 / \text{sqrt } ((1 - x^2) * (1 - y^2))$

**unfolding** *vector-def and M-def and inner-vec-def*  
**and** *matrix-vector-mult-def*

**by** (*simp add: sum-3 algebra-simps*)

**let**  $?o = \text{vector } [0,0,1]$

**let**  $?oMa = ?o \cdot (M *v ?a)$

**let**  $?oMb = ?o \cdot (M *v ?b)$

**let**  $?oMo = ?o \cdot (M *v ?o)$

**from** *K2-centre-non-zero* **and**  $\langle ?a \neq 0 \rangle$  **and**  $\langle ?b \neq 0 \rangle$

**and** *K2-centre-in-K2* **and**  $\langle ?pa \in \text{hyp2} \rangle$  **and**  $\langle ?pb \in \text{hyp2} \rangle$

**and** *cosh-dist-formula [of ?o]*

**have**  $\text{cosh-dist } ?po \ ?pa = |?oMa| / \text{sqrt } (?oMo * ?aMa)$

**and**  $\text{cosh-dist } ?po \ ?pb = |?oMb| / \text{sqrt } (?oMo * ?bMb)$

**by** (*unfold K2-centre-def, simp-all*)

**hence**  $\text{cosh-dist } ?po \ ?pa = 1 / \text{sqrt } (1 - x^2)$

**and**  $\text{cosh-dist } ?po \ ?pb = 1 / \text{sqrt } (1 - y^2)$   
**unfolding**  $\text{vector-def}$  **and**  $\text{M-def}$  **and**  $\text{inner-vec-def}$   
**and**  $\text{matrix-vector-mult-def}$   
**by** ( $\text{simp-all add: sum-3 power2-eq-square}$ )  
**with**  $\langle \text{cosh-dist } ?pa \ ?pb = 1 / \text{sqrt } ((1 - x^2) * (1 - y^2)) \rangle$   
**show**  $\text{cosh-dist } ?pa \ ?pb = \text{cosh-dist } ?po \ ?pa * \text{cosh-dist } ?po \ ?pb$   
**by** ( $\text{simp add: real-sqrt-mult}$ )  
**qed**

**lemma**  $K2\text{-isometry-cross-ratio-endpoints-in-S}$ :  
**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and**  $\text{is-K2-isometry } J$  **and**  $a \neq b$   
**shows**  $\text{cross-ratio } (\text{apply-cltn2 } (\text{endpoint-in-S } a \ b) \ J)$   
 $(\text{apply-cltn2 } (\text{endpoint-in-S } b \ a) \ J) (\text{apply-cltn2 } a \ J) (\text{apply-cltn2 } b \ J)$   
 $= \text{cross-ratio } (\text{endpoint-in-S } a \ b) (\text{endpoint-in-S } b \ a) \ a \ b$   
**(is**  $\text{cross-ratio } ?pJ \ ?qJ \ ?aJ \ ?bJ = \text{cross-ratio } ?p \ ?q \ a \ b)$

**proof** –  
**let**  $?l = \text{proj2-line-through } a \ b$   
**have**  $\text{proj2-incident } a \ ?l$  **and**  $\text{proj2-incident } b \ ?l$   
**by** ( $\text{rule proj2-line-through-incident}$ )  
**with**  $\langle a \neq b \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$   
**have**  $\text{proj2-incident } ?p \ ?l$  **and**  $\text{proj2-incident } ?q \ ?l$   
**by** ( $\text{simp-all add: endpoint-in-S-incident}$ )  
**with**  $\langle \text{proj2-incident } a \ ?l \rangle$  **and**  $\langle \text{proj2-incident } b \ ?l \rangle$   
**have**  $\text{proj2-set-Col } \{?p, ?q, a, b\}$   
**by** ( $\text{unfold proj2-set-Col-def}$ ) ( $\text{simp add: exI [of - ?l]}$ )

**from**  $\langle a \neq b \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$   
**have**  $?p \neq ?q$  **by** ( $\text{simp add: endpoint-in-S-swap}$ )

**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **have**  $?p \in S$  **by** ( $\text{simp add: endpoint-in-S}$ )  
**with**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$   
**have**  $a \neq ?p$  **and**  $b \neq ?p$  **by** ( $\text{simp-all add: hyp2-S-not-equal}$ )  
**with**  $\langle \text{proj2-set-Col } \{?p, ?q, a, b\} \rangle$  **and**  $\langle ?p \neq ?q \rangle$   
**show**  $\text{cross-ratio } ?pJ \ ?qJ \ ?aJ \ ?bJ = \text{cross-ratio } ?p \ ?q \ a \ b$   
**by** ( $\text{rule cross-ratio-cltn2}$ )

**qed**

**lemma**  $K2\text{-isometry-exp-2dist}$ :  
**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and**  $\text{is-K2-isometry } J$   
**shows**  $\text{exp-2dist } (\text{apply-cltn2 } a \ J) (\text{apply-cltn2 } b \ J) = \text{exp-2dist } a \ b$   
**(is**  $\text{exp-2dist } ?aJ \ ?bJ = -)$

**proof** *cases*  
**assume**  $a = b$   
**thus**  $\text{exp-2dist } ?aJ \ ?bJ = \text{exp-2dist } a \ b$  **by** ( $\text{unfold exp-2dist-def}$ ) *simp*  
**next**  
**assume**  $a \neq b$   
**with**  $\text{apply-cltn2-injective}$  **have**  $?aJ \neq ?bJ$  **by** *fast*

**let**  $?p = \text{endpoint-in-S } a \ b$

**let**  $?q = \text{endpoint-in-}S\ b\ a$   
**let**  $?aJ = \text{apply-cltn2}\ a\ J$   
**and**  $?bJ = \text{apply-cltn2}\ b\ J$   
**and**  $?pJ = \text{apply-cltn2}\ ?p\ J$   
**and**  $?qJ = \text{apply-cltn2}\ ?q\ J$   
**from**  $\langle a \neq b \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle \text{is-}K2\text{-isometry}\ J \rangle$   
**have**  $\text{endpoint-in-}S\ ?aJ\ ?bJ = ?pJ$  **and**  $\text{endpoint-in-}S\ ?bJ\ ?aJ = ?qJ$   
**by** (*simp-all add: K2-isometry-endpoint-in-S*)

**from** *assms* **and**  $\langle a \neq b \rangle$   
**have**  $\text{cross-ratio}\ ?pJ\ ?qJ\ ?aJ\ ?bJ = \text{cross-ratio}\ ?p\ ?q\ a\ b$   
**by** (*rule K2-isometry-cross-ratio-endpoints-in-S*)  
**with**  $\langle \text{endpoint-in-}S\ ?aJ\ ?bJ = ?pJ \rangle$  **and**  $\langle \text{endpoint-in-}S\ ?bJ\ ?aJ = ?qJ \rangle$   
**and**  $\langle a \neq b \rangle$  **and**  $\langle ?aJ \neq ?bJ \rangle$   
**show**  $\text{exp-2dist}\ ?aJ\ ?bJ = \text{exp-2dist}\ a\ b$  **by** (*unfold exp-2dist-def*) *simp*  
**qed**

**lemma** *K2-isometry-cosh-dist*:  
**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and**  $\text{is-}K2\text{-isometry}\ J$   
**shows**  $\text{cosh-dist}\ (\text{apply-cltn2}\ a\ J)\ (\text{apply-cltn2}\ b\ J) = \text{cosh-dist}\ a\ b$   
**using** *assms*  
**by** (*unfold cosh-dist-def*) (*simp add: K2-isometry-exp-2dist*)

**lemma** *cosh-dist-perp*:  
**assumes**  $M\text{-perp}\ l\ m$  **and**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and**  $c \in \text{hyp2}$   
**and**  $\text{proj2-incident}\ a\ l$  **and**  $\text{proj2-incident}\ b\ l$   
**and**  $\text{proj2-incident}\ b\ m$  **and**  $\text{proj2-incident}\ c\ m$   
**shows**  $\text{cosh-dist}\ a\ c = \text{cosh-dist}\ b\ a * \text{cosh-dist}\ b\ c$   
**proof** –  
**from**  $\langle M\text{-perp}\ l\ m \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident}\ b\ l \rangle$   
**and**  $\langle \text{proj2-incident}\ b\ m \rangle$  **and**  $M\text{-perp-to-compass}\ [of\ l\ m\ b\ b]$   
**obtain**  $J$  **where**  $\text{is-}K2\text{-isometry}\ J$  **and**  $\text{apply-cltn2-line}\ \text{equator}\ J = l$   
**and**  $\text{apply-cltn2-line}\ \text{meridian}\ J = m$   
**by** *auto*

**let**  $?Ji = \text{cltn2-inverse}\ J$   
**let**  $?aJi = \text{apply-cltn2}\ a\ ?Ji$   
**let**  $?bJi = \text{apply-cltn2}\ b\ ?Ji$   
**let**  $?cJi = \text{apply-cltn2}\ c\ ?Ji$   
**from**  $\langle \text{apply-cltn2-line}\ \text{equator}\ J = l \rangle$  **and**  $\langle \text{apply-cltn2-line}\ \text{meridian}\ J = m \rangle$   
**and**  $\langle \text{proj2-incident}\ a\ l \rangle$  **and**  $\langle \text{proj2-incident}\ b\ l \rangle$   
**and**  $\langle \text{proj2-incident}\ b\ m \rangle$  **and**  $\langle \text{proj2-incident}\ c\ m \rangle$   
**have**  $\text{proj2-incident}\ ?aJi\ \text{equator}$  **and**  $\text{proj2-incident}\ ?bJi\ \text{equator}$   
**and**  $\text{proj2-incident}\ ?bJi\ \text{meridian}$  **and**  $\text{proj2-incident}\ ?cJi\ \text{meridian}$   
**by** (*auto simp add: apply-cltn2-incident*)

**from**  $\langle \text{is-}K2\text{-isometry}\ J \rangle$   
**have**  $\text{is-}K2\text{-isometry}\ ?Ji$  **by** (*rule cltn2-inverse-is-K2-isometry*)  
**with**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle c \in \text{hyp2} \rangle$

**have**  $?aJi \in \text{hyp2}$  **and**  $?cJi \in \text{hyp2}$   
**by** (*simp-all add: statement60-one-way*)

**from**  $\langle ?aJi \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } ?aJi \text{ equator} \rangle$   
**and** *on-equator-in-hyp2-rep*  
**obtain**  $x$  **where**  $|x| < 1$  **and**  $?aJi = \text{proj2-abs (vector [x,0,1])}$  **by** *auto*  
**moreover**  
**from**  $\langle ?cJi \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } ?cJi \text{ meridian} \rangle$   
**and** *on-meridian-in-hyp2-rep*  
**obtain**  $y$  **where**  $|y| < 1$  **and**  $?cJi = \text{proj2-abs (vector [0,y,1])}$  **by** *auto*  
**moreover**  
**from**  $\langle \text{proj2-incident } ?bJi \text{ equator} \rangle$  **and**  $\langle \text{proj2-incident } ?bJi \text{ meridian} \rangle$   
**have**  $?bJi = K2\text{-centre}$  **by** (*rule on-equator-meridian-is-K2-centre*)  
**ultimately**  
**have**  $\text{cosh-dist } ?aJi ?cJi = \text{cosh-dist } ?bJi ?aJi * \text{cosh-dist } ?bJi ?cJi$   
**by** (*simp add: cosh-dist-perp-special-case*)  
**with**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle c \in \text{hyp2} \rangle$  **and**  $\langle \text{is-K2-isometry } ?Ji \rangle$   
**show**  $\text{cosh-dist } a c = \text{cosh-dist } b a * \text{cosh-dist } b c$   
**by** (*simp add: K2-isometry-cosh-dist*)  
**qed**

**lemma** *are-endpoints-in-S-ordered-cross-ratio:*

**assumes** *are-endpoints-in-S*  $p q a b$   
**and**  $B_{\mathbb{R}} (\text{cart2-pt } a) (\text{cart2-pt } b) (\text{cart2-pt } p)$  (**is**  $B_{\mathbb{R}} ?ca ?cb ?cp$ )  
**shows** *cross-ratio*  $p q a b \geq 1$

**proof** –

**from**  $\langle \text{are-endpoints-in-S } p q a b \rangle$   
**have**  $p \neq q$  **and**  $p \in S$  **and**  $q \in S$  **and**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$   
**and**  $\text{proj2-set-Col } \{p,q,a,b\}$   
**by** (*unfold are-endpoints-in-S-def*) *simp-all*

**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$   
**have** *z-non-zero*  $a$  **and** *z-non-zero*  $b$  **and** *z-non-zero*  $p$  **and** *z-non-zero*  $q$   
**by** (*simp-all add: hyp2-S-z-non-zero*)  
**hence**  $\text{proj2-abs (cart2-append1 } p) = p$  (**is**  $\text{proj2-abs } ?cp1 = p$ )  
**and**  $\text{proj2-abs (cart2-append1 } q) = q$  (**is**  $\text{proj2-abs } ?cq1 = q$ )  
**and**  $\text{proj2-abs (cart2-append1 } a) = a$  (**is**  $\text{proj2-abs } ?ca1 = a$ )  
**and**  $\text{proj2-abs (cart2-append1 } b) = b$  (**is**  $\text{proj2-abs } ?cb1 = b$ )  
**by** (*simp-all add: proj2-abs-cart2-append1*)

**from**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle p \in S \rangle$  **have**  $b \neq p$  **by** (*rule hyp2-S-not-equal*)  
**with**  $\langle \text{z-non-zero } a \rangle$  **and**  $\langle \text{z-non-zero } b \rangle$  **and**  $\langle \text{z-non-zero } p \rangle$   
**and**  $\langle B_{\mathbb{R}} ?ca ?cb ?cp \rangle$  **and** *cart2-append1-between-right-strict* [*of*  $a b p$ ]  
**obtain**  $j$  **where**  $j \geq 0$  **and**  $j < 1$  **and**  $?cb1 = j *_{\mathbb{R}} ?cp1 + (1-j) *_{\mathbb{R}} ?ca1$   
**by** *auto*

**from**  $\langle \text{proj2-set-Col } \{p,q,a,b\} \rangle$   
**obtain**  $l$  **where**  $\text{proj2-incident } q l$  **and**  $\text{proj2-incident } p l$   
**and**  $\text{proj2-incident } a l$

by (unfold proj2-set-Col-def) auto  
 with  $\langle p \neq q \rangle$  and  $\langle q \in S \rangle$  and  $\langle p \in S \rangle$  and  $\langle a \in \text{hyp2} \rangle$   
 and  $S\text{-hyp2-}S\text{-cart2-append1}$  [of  $q$   $p$   $a$   $l$ ]  
 obtain  $k$  where  $k > 0$  and  $k < 1$  and  $?ca1 = k *_R ?cp1 + (1-k) *_R ?cq1$   
 by auto

from  $\langle z\text{-non-zero } p \rangle$  and  $\langle z\text{-non-zero } q \rangle$   
 have  $?cp1 \neq 0$  and  $?cq1 \neq 0$  by (simp-all add: cart2-append1-non-zero)

from  $\langle p \neq q \rangle$  and  $\langle \text{proj2-abs } ?cp1 = p \rangle$  and  $\langle \text{proj2-abs } ?cq1 = q \rangle$   
 have  $\text{proj2-abs } ?cp1 \neq \text{proj2-abs } ?cq1$  by simp

from  $\langle k < 1 \rangle$  have  $1-k \neq 0$  by simp  
 with  $\langle j < 1 \rangle$  have  $(1-j)*(1-k) \neq 0$  by simp

from  $\langle j < 1 \rangle$  and  $\langle k > 0 \rangle$  have  $(1-j)*k > 0$  by simp

from  $\langle ?cb1 = j *_R ?cp1 + (1-j) *_R ?ca1 \rangle$   
 have  $?cb1 = (j+(1-j)*k) *_R ?cp1 + ((1-j)*(1-k)) *_R ?cq1$   
 by (unfold  $\langle ?ca1 = k *_R ?cp1 + (1-k) *_R ?cq1 \rangle$ ) (simp add: algebra-simps)  
 with  $\langle ?ca1 = k *_R ?cp1 + (1-k) *_R ?cq1 \rangle$   
 have  $\text{proj2-abs } ?ca1 = \text{proj2-abs } (k *_R ?cp1 + (1-k) *_R ?cq1)$   
 and  $\text{proj2-abs } ?cb1$   
 $= \text{proj2-abs } ((j+(1-j)*k) *_R ?cp1 + ((1-j)*(1-k)) *_R ?cq1)$   
 by simp-all  
 with  $\langle \text{proj2-abs } ?ca1 = a \rangle$  and  $\langle \text{proj2-abs } ?cb1 = b \rangle$   
 have  $a = \text{proj2-abs } (k *_R ?cp1 + (1-k) *_R ?cq1)$   
 and  $b = \text{proj2-abs } ((j+(1-j)*k) *_R ?cp1 + ((1-j)*(1-k)) *_R ?cq1)$   
 by simp-all  
 with  $\langle \text{proj2-abs } ?cp1 = p \rangle$  and  $\langle \text{proj2-abs } ?cq1 = q \rangle$   
 have  $\text{cross-ratio } p$   $q$   $a$   $b$   
 $= \text{cross-ratio } (\text{proj2-abs } ?cp1)$   $(\text{proj2-abs } ?cq1)$   
 $(\text{proj2-abs } (k *_R ?cp1 + (1-k) *_R ?cq1))$   
 $(\text{proj2-abs } ((j+(1-j)*k) *_R ?cp1 + ((1-j)*(1-k)) *_R ?cq1))$   
 by simp  
 also from  $\langle ?cp1 \neq 0 \rangle$  and  $\langle ?cq1 \neq 0 \rangle$  and  $\langle \text{proj2-abs } ?cp1 \neq \text{proj2-abs } ?cq1 \rangle$   
 and  $\langle 1-k \neq 0 \rangle$  and  $\langle (1-j)*(1-k) \neq 0 \rangle$   
 have  $\dots = (1-k)*(j+(1-j)*k) / (k*((1-j)*(1-k)))$  by (rule cross-ratio-abs)  
 also from  $\langle 1-k \neq 0 \rangle$  have  $\dots = (j+(1-j)*k) / ((1-j)*k)$  by simp  
 also from  $\langle j \geq 0 \rangle$  and  $\langle (1-j)*k > 0 \rangle$  have  $\dots \geq 1$  by simp  
 finally show  $\text{cross-ratio } p$   $q$   $a$   $b \geq 1$  .

qed

lemma  $\text{cross-ratio-}S\text{-}S\text{-hyp2-hyp2-positive}$ :

assumes  $\text{are-endpoints-in-}S$   $p$   $q$   $a$   $b$

shows  $\text{cross-ratio } p$   $q$   $a$   $b > 0$

proof cases

assume  $B_{\mathbb{R}}$  (cart2-pt  $p$ ) (cart2-pt  $b$ ) (cart2-pt  $a$ )

hence  $B_{\mathbb{R}}$  (cart2-pt  $a$ ) (cart2-pt  $b$ ) (cart2-pt  $p$ )

by (rule real-euclid.th3-2)  
 with *assms* have *cross-ratio*  $p\ q\ a\ b \geq 1$   
 by (rule are-endpoints-in-S-ordered-cross-ratio)  
 thus *cross-ratio*  $p\ q\ a\ b > 0$  by *simp*  
**next**  
 assume  $\neg B_{\mathbb{R}}(\text{cart2-pt } p)(\text{cart2-pt } b)(\text{cart2-pt } a)$  (is  $\neg B_{\mathbb{R}}\ ?cp\ ?cb\ ?ca$ )  
  
 from  $\langle \text{are-endpoints-in-S } p\ q\ a\ b \rangle$   
 have *are-endpoints-in-S*  $p\ q\ b\ a$  by (rule are-endpoints-in-S-swap-34)  
  
 from  $\langle \text{are-endpoints-in-S } p\ q\ a\ b \rangle$   
 have  $p \in S$  and  $a \in \text{hyp2}$  and  $b \in \text{hyp2}$  and *proj2-set-Col*  $\{p, q, a, b\}$   
 by (unfold are-endpoints-in-S-def) *simp-all*  
  
 from  $\langle \text{proj2-set-Col } \{p, q, a, b\} \rangle$   
 have *proj2-set-Col*  $\{p, a, b\}$   
 by (*simp add: proj2-subset-Col [of {p, a, b} {p, q, a, b}]*)  
 hence *proj2-Col*  $p\ a\ b$  by (*subst proj2-Col-iff-set-Col*)  
 with  $\langle p \in S \rangle$  and  $\langle a \in \text{hyp2} \rangle$  and  $\langle b \in \text{hyp2} \rangle$   
 have  $B_{\mathbb{R}}\ ?cp\ ?ca\ ?cb \vee B_{\mathbb{R}}\ ?cp\ ?cb\ ?ca$  by (*simp add: S-at-edge*)  
 with  $\langle \neg B_{\mathbb{R}}\ ?cp\ ?cb\ ?ca \rangle$  have  $B_{\mathbb{R}}\ ?cp\ ?ca\ ?cb$  by *simp*  
 hence  $B_{\mathbb{R}}\ ?cb\ ?ca\ ?cp$  by (rule real-euclid.th3-2)  
 with  $\langle \text{are-endpoints-in-S } p\ q\ b\ a \rangle$   
 have *cross-ratio*  $p\ q\ b\ a \geq 1$   
 by (rule are-endpoints-in-S-ordered-cross-ratio)  
 thus *cross-ratio*  $p\ q\ a\ b > 0$  by (*subst cross-ratio-swap-34*) *simp*  
**qed**

**lemma** *cosh-dist-general*:

assumes *are-endpoints-in-S*  $p\ q\ a\ b$   
 shows *cosh-dist*  $a\ b$   
 = (*sqrt* (*cross-ratio*  $p\ q\ a\ b$ ) + 1 / *sqrt* (*cross-ratio*  $p\ q\ a\ b$ )) / 2  
**proof** –

from  $\langle \text{are-endpoints-in-S } p\ q\ a\ b \rangle$   
 have  $p \neq q$  and  $p \in S$  and  $q \in S$  and  $a \in \text{hyp2}$  and  $b \in \text{hyp2}$   
 and *proj2-set-Col*  $\{p, q, a, b\}$   
 by (unfold are-endpoints-in-S-def) *simp-all*

from  $\langle a \in \text{hyp2} \rangle$  and  $\langle b \in \text{hyp2} \rangle$  and  $\langle p \in S \rangle$  and  $\langle q \in S \rangle$   
 have  $a \neq p$  and  $a \neq q$  and  $b \neq p$  and  $b \neq q$   
 by (*simp-all add: hyp2-S-not-equal*)

**show** *cosh-dist*  $a\ b$

= (*sqrt* (*cross-ratio*  $p\ q\ a\ b$ ) + 1 / *sqrt* (*cross-ratio*  $p\ q\ a\ b$ )) / 2

**proof** *cases*

assume  $a = b$

hence *cosh-dist*  $a\ b = 1$  by (*unfold cosh-dist-def exp-2dist-def*) *simp*

from  $\langle \text{proj2-set-Col } \{p, q, a, b\} \rangle$

**have** *proj2-Col*  $p$   $q$   $a$  **by** (*unfold*  $\langle a = b \rangle$ ) (*simp add: proj2-Col-iff-set-Col*)  
**with**  $\langle p \neq q \rangle$  **and**  $\langle a \neq p \rangle$  **and**  $\langle a \neq q \rangle$   
**have** *cross-ratio*  $p$   $q$   $a$   $b = 1$  **by** (*simp add:  $\langle a = b \rangle$  cross-ratio-equal-1*)  
**hence** (*sqrt* (*cross-ratio*  $p$   $q$   $a$   $b$ ) + 1 / *sqrt* (*cross-ratio*  $p$   $q$   $a$   $b$ )) / 2  
= 1  
**by** *simp*  
**with**  $\langle \text{cosh-dist } a \ b = 1 \rangle$   
**show** *cosh-dist*  $a$   $b$   
= (*sqrt* (*cross-ratio*  $p$   $q$   $a$   $b$ ) + 1 / *sqrt* (*cross-ratio*  $p$   $q$   $a$   $b$ )) / 2  
**by** *simp*  
**next**  
**assume**  $a \neq b$   
  
**let**  $?r = \text{endpoint-in-}S \ a \ b$   
**let**  $?s = \text{endpoint-in-}S \ b \ a$   
**from**  $\langle a \neq b \rangle$   
**have** *exp-2dist*  $a$   $b = \text{cross-ratio } ?r \ ?s \ a \ b$  **by** (*unfold exp-2dist-def*) *simp*  
  
**from**  $\langle a \neq b \rangle$  **and**  $\langle \text{are-endpoints-in-}S \ p \ q \ a \ b \rangle$   
**have** ( $p = ?r \wedge q = ?s$ )  $\vee$  ( $q = ?r \wedge p = ?s$ ) **by** (*rule are-endpoints-in-S*)  
  
**show** *cosh-dist*  $a$   $b$   
= (*sqrt* (*cross-ratio*  $p$   $q$   $a$   $b$ ) + 1 / *sqrt* (*cross-ratio*  $p$   $q$   $a$   $b$ )) / 2  
**proof cases**  
**assume**  $p = ?r \wedge q = ?s$   
**with**  $\langle \text{exp-2dist } a \ b = \text{cross-ratio } ?r \ ?s \ a \ b \rangle$   
**have** *exp-2dist*  $a$   $b = \text{cross-ratio } p \ q \ a \ b$  **by** *simp*  
**thus** *cosh-dist*  $a$   $b$   
= (*sqrt* (*cross-ratio*  $p$   $q$   $a$   $b$ ) + 1 / *sqrt* (*cross-ratio*  $p$   $q$   $a$   $b$ )) / 2  
**by** (*unfold cosh-dist-def*) (*simp add: real-sqrt-divide*)  
**next**  
**assume**  $\neg (p = ?r \wedge q = ?s)$   
**with**  $\langle (p = ?r \wedge q = ?s) \vee (q = ?r \wedge p = ?s) \rangle$   
**have**  $q = ?r$  **and**  $p = ?s$  **by** *simp-all*  
**with**  $\langle \text{exp-2dist } a \ b = \text{cross-ratio } ?r \ ?s \ a \ b \rangle$   
**have** *exp-2dist*  $a$   $b = \text{cross-ratio } q \ p \ a \ b$  **by** *simp*  
  
**have**  $\{q, p, a, b\} = \{p, q, a, b\}$  **by** *auto*  
**with**  $\langle \text{proj2-set-Col } \{p, q, a, b\} \rangle$  **and**  $\langle p \neq q \rangle$  **and**  $\langle a \neq p \rangle$  **and**  $\langle b \neq p \rangle$   
**and**  $\langle a \neq q \rangle$  **and**  $\langle b \neq q \rangle$   
**have** *cross-ratio-correct*  $p$   $q$   $a$   $b$  **and** *cross-ratio-correct*  $q$   $p$   $a$   $b$   
**by** (*unfold cross-ratio-correct-def*) *simp-all*  
**hence** *cross-ratio*  $q$   $p$   $a$   $b = 1 / (\text{cross-ratio } p \ q \ a \ b)$   
**by** (*rule cross-ratio-swap-12*)  
**with**  $\langle \text{exp-2dist } a \ b = \text{cross-ratio } q \ p \ a \ b \rangle$   
**have** *exp-2dist*  $a$   $b = 1 / (\text{cross-ratio } p \ q \ a \ b)$  **by** *simp*  
**thus** *cosh-dist*  $a$   $b$   
= (*sqrt* (*cross-ratio*  $p$   $q$   $a$   $b$ ) + 1 / *sqrt* (*cross-ratio*  $p$   $q$   $a$   $b$ )) / 2  
**by** (*unfold cosh-dist-def*) (*simp add: real-sqrt-divide*)

qed  
 qed  
 qed

**lemma** *exp-2dist-positive*:

**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$   
**shows**  $\text{exp-2dist } a \ b > 0$

**proof** *cases*

**assume**  $a = b$

**thus**  $\text{exp-2dist } a \ b > 0$  **by** (*unfold exp-2dist-def*) *simp*

**next**

**assume**  $a \neq b$

**let**  $?p = \text{endpoint-in-}S \ a \ b$

**let**  $?q = \text{endpoint-in-}S \ b \ a$

**from**  $\langle a \neq b \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$

**have**  $\text{are-endpoints-in-}S \ ?p \ ?q \ a \ b$

**by** (*rule endpoints-in-S-are-endpoints-in-S*)

**hence**  $\text{cross-ratio } ?p \ ?q \ a \ b > 0$  **by** (*rule cross-ratio-S-S-hyp2-hyp2-positive*)

**with**  $\langle a \neq b \rangle$  **show**  $\text{exp-2dist } a \ b > 0$  **by** (*unfold exp-2dist-def*) *simp*

qed

**lemma** *cosh-dist-at-least-1*:

**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$

**shows**  $\text{cosh-dist } a \ b \geq 1$

**proof** –

**from** *assms* **have**  $\text{exp-2dist } a \ b > 0$  **by** (*rule exp-2dist-positive*)

**with**  $\text{am-gm2}(1)$  [*of sqrt (exp-2dist a b) sqrt (1 / exp-2dist a b)*]

**show**  $\text{cosh-dist } a \ b \geq 1$

**by** (*unfold cosh-dist-def*) (*simp add: real-sqrt-mult [symmetric]*)

qed

**lemma** *cosh-dist-positive*:

**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$

**shows**  $\text{cosh-dist } a \ b > 0$

**proof** –

**from** *assms* **have**  $\text{cosh-dist } a \ b \geq 1$  **by** (*rule cosh-dist-at-least-1*)

**thus**  $\text{cosh-dist } a \ b > 0$  **by** *simp*

qed

**lemma** *cosh-dist-perp-divide*:

**assumes**  $M\text{-perp } l \ m$  **and**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and**  $c \in \text{hyp2}$

**and**  $\text{proj2-incident } a \ l$  **and**  $\text{proj2-incident } b \ l$  **and**  $\text{proj2-incident } b \ m$

**and**  $\text{proj2-incident } c \ m$

**shows**  $\text{cosh-dist } b \ c = \text{cosh-dist } a \ c / \text{cosh-dist } b \ a$

**proof** –

**from**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$

**have**  $\text{cosh-dist } b \ a > 0$  **by** (*rule cosh-dist-positive*)

**from** *assms*  
**have**  $\text{cosh-dist } a \ c = \text{cosh-dist } b \ a * \text{cosh-dist } b \ c$  **by** (*rule cosh-dist-perp*)  
**with**  $\langle \text{cosh-dist } b \ a > 0 \rangle$   
**show**  $\text{cosh-dist } b \ c = \text{cosh-dist } a \ c / \text{cosh-dist } b \ a$  **by** *simp*  
**qed**

**lemma** *real-hyp2-C-cross-ratio-endpoints-in-S*:

**assumes**  $a \neq b$  **and**  $a \ b \equiv_K \ c \ d$   
**shows**  $\text{cross-ratio } (\text{endpoint-in-S } (\text{Rep-hyp2 } a) (\text{Rep-hyp2 } b))$   
 $(\text{endpoint-in-S } (\text{Rep-hyp2 } b) (\text{Rep-hyp2 } a)) (\text{Rep-hyp2 } a) (\text{Rep-hyp2 } b)$   
 $= \text{cross-ratio } (\text{endpoint-in-S } (\text{Rep-hyp2 } c) (\text{Rep-hyp2 } d))$   
 $(\text{endpoint-in-S } (\text{Rep-hyp2 } d) (\text{Rep-hyp2 } c)) (\text{Rep-hyp2 } c) (\text{Rep-hyp2 } d)$   
**(is**  $\text{cross-ratio } ?p \ ?q \ ?a' \ ?b' = \text{cross-ratio } ?r \ ?s \ ?c' \ ?d'$ **)**

**proof** –

**from**  $\langle a \neq b \rangle$  **and**  $\langle a \ b \equiv_K \ c \ d \rangle$  **have**  $c \neq d$  **by** (*auto simp add: hyp2.A3'*)  
**with**  $\langle a \neq b \rangle$  **have**  $?a' \neq ?b'$  **and**  $?c' \neq ?d'$  **by** (*unfold Rep-hyp2-inject*)

**from**  $\langle a \ b \equiv_K \ c \ d \rangle$

**obtain**  $J$  **where** *is-K2-isometry*  $J$  **and** *hyp2-cltn2*  $a \ J = c$

**and** *hyp2-cltn2*  $b \ J = d$

**by** (*unfold real-hyp2-C-def*) *auto*

**hence** *apply-cltn2*  $?a' \ J = ?c'$  **and** *apply-cltn2*  $?b' \ J = ?d'$

**by** (*simp-all add: Rep-hyp2-cltn2 [symmetric]*)

**with**  $\langle ?a' \neq ?b' \rangle$  **and**  $\langle \text{is-K2-isometry } J \rangle$

**have** *apply-cltn2*  $?p \ J = ?r$  **and** *apply-cltn2*  $?q \ J = ?s$

**by** (*simp-all add: Rep-hyp2 K2-isometry-endpoint-in-S*)

**from**  $\langle ?a' \neq ?b' \rangle$

**have** *proj2-set-Col*  $\{?p, ?q, ?a', ?b'\}$

**by** (*simp add: Rep-hyp2 proj2-set-Col-endpoints-in-S*)

**from**  $\langle ?a' \neq ?b' \rangle$  **have**  $?p \neq ?q$  **by** (*simp add: Rep-hyp2 endpoint-in-S-swap*)

**have**  $?p \in S$  **by** (*simp add: Rep-hyp2 endpoint-in-S*)

**hence**  $?a' \neq ?p$  **and**  $?b' \neq ?p$  **by** (*simp-all add: Rep-hyp2 hyp2-S-not-equal*)

**with**  $\langle \text{proj2-set-Col } \{?p, ?q, ?a', ?b'\} \rangle$  **and**  $\langle ?p \neq ?q \rangle$

**have**  $\text{cross-ratio } ?p \ ?q \ ?a' \ ?b'$

$= \text{cross-ratio } (\text{apply-cltn2 } ?p \ J) (\text{apply-cltn2 } ?q \ J)$

$(\text{apply-cltn2 } ?a' \ J) (\text{apply-cltn2 } ?b' \ J)$

**by** (*rule cross-ratio-cltn2 [symmetric]*)

**with**  $\langle \text{apply-cltn2 } ?p \ J = ?r \rangle$  **and**  $\langle \text{apply-cltn2 } ?q \ J = ?s \rangle$

**and**  $\langle \text{apply-cltn2 } ?a' \ J = ?c' \rangle$  **and**  $\langle \text{apply-cltn2 } ?b' \ J = ?d' \rangle$

**show**  $\text{cross-ratio } ?p \ ?q \ ?a' \ ?b' = \text{cross-ratio } ?r \ ?s \ ?c' \ ?d'$  **by** *simp*

**qed**

**lemma** *real-hyp2-C-exp-2dist*:

**assumes**  $a \ b \equiv_K \ c \ d$

**shows**  $\text{exp-2dist } (\text{Rep-hyp2 } a) (\text{Rep-hyp2 } b)$

$= \text{exp-2dist } (\text{Rep-hyp2 } c) (\text{Rep-hyp2 } d)$

(is exp-2dist ?a' ?b' = exp-2dist ?c' ?d')

**proof** –

**from**  $\langle a \ b \equiv_K \ c \ d \rangle$

**obtain**  $J$  **where** *is-K2-isometry*  $J$  **and** *hyp2-cltn2*  $a \ J = c$

**and** *hyp2-cltn2*  $b \ J = d$

**by** (*unfold real-hyp2-C-def*) *auto*

**hence** *apply-cltn2* ?a'  $J = ?c'$  **and** *apply-cltn2* ?b'  $J = ?d'$

**by** (*simp-all add: Rep-hyp2-cltn2 [symmetric]*)

**from** *Rep-hyp2* [of  $a$ ] **and** *Rep-hyp2* [of  $b$ ] **and**  $\langle$ *is-K2-isometry*  $J \rangle$

**have** *exp-2dist* (*apply-cltn2* ?a'  $J$ ) (*apply-cltn2* ?b'  $J$ ) = *exp-2dist* ?a' ?b'

**by** (*rule K2-isometry-exp-2dist*)

**with**  $\langle$ *apply-cltn2* ?a'  $J = ?c' \rangle$  **and**  $\langle$ *apply-cltn2* ?b'  $J = ?d' \rangle$

**show** *exp-2dist* ?a' ?b' = *exp-2dist* ?c' ?d' **by** *simp*

**qed**

**lemma** *real-hyp2-C-cosh-dist*:

**assumes**  $a \ b \equiv_K \ c \ d$

**shows** *cosh-dist* (*Rep-hyp2*  $a$ ) (*Rep-hyp2*  $b$ )

= *cosh-dist* (*Rep-hyp2*  $c$ ) (*Rep-hyp2*  $d$ )

**using** *assms*

**by** (*unfold cosh-dist-def*) (*simp add: real-hyp2-C-exp-2dist*)

**lemma** *cross-ratio-in-terms-of-cosh-dist*:

**assumes** *are-endpoints-in-S*  $p \ q \ a \ b$

**and**  $B_{\mathbb{R}}$  (*cart2-pt*  $a$ ) (*cart2-pt*  $b$ ) (*cart2-pt*  $p$ )

**shows** *cross-ratio*  $p \ q \ a \ b$

=  $2 * (\cosh\text{-dist } a \ b)^2 + 2 * \cosh\text{-dist } a \ b * \sqrt{(\cosh\text{-dist } a \ b)^2 - 1} - 1$

(is ?pqab =  $2 * ?ab^2 + 2 * ?ab * \sqrt{?ab^2 - 1} - 1$ )

**proof** –

**from**  $\langle$ *are-endpoints-in-S*  $p \ q \ a \ b \rangle$

**have** ?ab = ( $\sqrt{?pqab} + 1 / \sqrt{?pqab}$ ) / 2 **by** (*rule cosh-dist-general*)

**hence**  $\sqrt{?pqab} - 2 * ?ab + 1 / \sqrt{?pqab} = 0$  **by** *simp*

**hence**  $\sqrt{?pqab} * (\sqrt{?pqab} - 2 * ?ab + 1 / \sqrt{?pqab}) = 0$  **by** *simp*

**moreover from** *assms*

**have** ?pqab  $\geq 1$  **by** (*rule are-endpoints-in-S-ordered-cross-ratio*)

**ultimately have** ?pqab -  $2 * ?ab * (\sqrt{?pqab}) + 1 = 0$

**by** (*simp add: algebra-simps real-sqrt-mult [symmetric]*)

**with**  $\langle$ ?pqab  $\geq 1 \rangle$  **and** *discriminant-iff* [of 1  $\sqrt{?pqab} - 2 * ?ab$  1]

**have**  $\sqrt{?pqab} = (2 * ?ab + \sqrt{4 * ?ab^2 - 4}) / 2$

$\vee \sqrt{?pqab} = (2 * ?ab - \sqrt{4 * ?ab^2 - 4}) / 2$

**unfolding** *discrim-def*

**by** (*simp add: real-sqrt-mult [symmetric] power2-eq-square*)

**moreover have**  $\sqrt{4 * ?ab^2 - 4} = \sqrt{4 * (?ab^2 - 1)}$  **by** *simp*

**hence**  $\sqrt{4 * ?ab^2 - 4} = 2 * \sqrt{?ab^2 - 1}$

**by** (*unfold real-sqrt-mult*) *simp*

**ultimately have**  $\sqrt{?pqab} = 2 * (?ab + \sqrt{?ab^2 - 1}) / 2$

$\vee \sqrt{?pqab} = 2 * (?ab - \sqrt{?ab^2 - 1}) / 2$

**by** *simp*

hence  $\text{sqrt } ?pqab = ?ab + \text{sqrt } (?ab^2 - 1)$   
 $\vee \text{sqrt } ?pqab = ?ab - \text{sqrt } (?ab^2 - 1)$   
 by (simp only: nonzero-mult-div-cancel-left [of 2])

from  $\langle \text{are-endpoints-in-}S \ p \ q \ a \ b \rangle$   
 have  $a \in \text{hyp2}$  and  $b \in \text{hyp2}$  by (unfold are-endpoints-in- $S$ -def) simp-all  
 hence  $?ab \geq 1$  by (rule cosh-dist-at-least-1)  
 hence  $?ab^2 \geq 1$  by simp  
 hence  $\text{sqrt } (?ab^2 - 1) \geq 0$  by simp  
 hence  $\text{sqrt } (?ab^2 - 1) * \text{sqrt } (?ab^2 - 1) = ?ab^2 - 1$   
 by (simp add: real-sqrt-mult [symmetric])  
 hence  $(?ab + \text{sqrt } (?ab^2 - 1)) * (?ab - \text{sqrt } (?ab^2 - 1)) = 1$   
 by (simp add: algebra-simps power2-eq-square)

have  $?ab - \text{sqrt } (?ab^2 - 1) \leq 1$   
 proof (rule ccontr)  
 assume  $\neg (?ab - \text{sqrt } (?ab^2 - 1) \leq 1)$   
 hence  $1 < ?ab - \text{sqrt } (?ab^2 - 1)$  by simp  
 also from  $\langle \text{sqrt } (?ab^2 - 1) \geq 0 \rangle$   
 have  $\dots \leq ?ab + \text{sqrt } (?ab^2 - 1)$  by simp  
 finally have  $1 < ?ab + \text{sqrt } (?ab^2 - 1)$  by simp  
 with  $\langle 1 < ?ab - \text{sqrt } (?ab^2 - 1) \rangle$   
 and mult-strict-mono' [of  
 $1 \ ?ab + \text{sqrt } (?ab^2 - 1) \ 1 \ ?ab - \text{sqrt } (?ab^2 - 1)$ ]  
 have  $1 < (?ab + \text{sqrt } (?ab^2 - 1)) * (?ab - \text{sqrt } (?ab^2 - 1))$  by simp  
 with  $\langle (?ab + \text{sqrt } (?ab^2 - 1)) * (?ab - \text{sqrt } (?ab^2 - 1)) = 1 \rangle$   
 show False by simp  
 qed

have  $\text{sqrt } ?pqab = ?ab + \text{sqrt } (?ab^2 - 1)$   
 proof (rule ccontr)  
 assume  $\text{sqrt } ?pqab \neq ?ab + \text{sqrt } (?ab^2 - 1)$   
 with  $\langle \text{sqrt } ?pqab = ?ab + \text{sqrt } (?ab^2 - 1) \vee \text{sqrt } ?pqab = ?ab - \text{sqrt } (?ab^2 - 1) \rangle$   
 have  $\text{sqrt } ?pqab = ?ab - \text{sqrt } (?ab^2 - 1)$  by simp  
 with  $\langle ?ab - \text{sqrt } (?ab^2 - 1) \leq 1 \rangle$  have  $\text{sqrt } ?pqab \leq 1$  by simp  
 with  $\langle ?pqab \geq 1 \rangle$  have  $\text{sqrt } ?pqab = 1$  by simp  
 with  $\langle \text{sqrt } ?pqab = ?ab - \text{sqrt } (?ab^2 - 1) \rangle$   
 and  $\langle (?ab + \text{sqrt } (?ab^2 - 1)) * (?ab - \text{sqrt } (?ab^2 - 1)) = 1 \rangle$   
 have  $?ab + \text{sqrt } (?ab^2 - 1) = 1$  by simp  
 with  $\langle \text{sqrt } ?pqab = 1 \rangle$  have  $\text{sqrt } ?pqab = ?ab + \text{sqrt } (?ab^2 - 1)$  by simp  
 with  $\langle \text{sqrt } ?pqab \neq ?ab + \text{sqrt } (?ab^2 - 1) \rangle$  show False ..  
 qed

moreover from  $\langle ?pqab \geq 1 \rangle$  have  $?pqab = (\text{sqrt } ?pqab)^2$  by simp  
 ultimately have  $?pqab = (?ab + \text{sqrt } (?ab^2 - 1))^2$  by simp  
 with  $\langle \text{sqrt } (?ab^2 - 1) * \text{sqrt } (?ab^2 - 1) = ?ab^2 - 1 \rangle$   
 show  $?pqab = 2 * ?ab^2 + 2 * ?ab * \text{sqrt } (?ab^2 - 1) - 1$   
 by (simp add: power2-eq-square algebra-simps)

qed

**lemma** *are-endpoints-in-S-cross-ratio-correct*:  
**assumes** *are-endpoints-in-S*  $p$   $q$   $a$   $b$   
**shows** *cross-ratio-correct*  $p$   $q$   $a$   $b$   
**proof** –  
**from**  $\langle \text{are-endpoints-in-S } p \ q \ a \ b \rangle$   
**have**  $p \neq q$  **and**  $p \in S$  **and**  $q \in S$  **and**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$   
**and** *proj2-set-Col*  $\{p, q, a, b\}$   
**by** (*unfold are-endpoints-in-S-def*) *simp-all*

**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$   
**have**  $a \neq p$  **and**  $b \neq p$  **and**  $a \neq q$  **by** (*simp-all add: hyp2-S-not-equal*)  
**with**  $\langle \text{proj2-set-Col } \{p, q, a, b\} \rangle$  **and**  $\langle p \neq q \rangle$   
**show** *cross-ratio-correct*  $p$   $q$   $a$   $b$  **by** (*unfold cross-ratio-correct-def*) *simp*  
**qed**

**lemma** *endpoints-in-S-cross-ratio-correct*:  
**assumes**  $a \neq b$  **and**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$   
**shows** *cross-ratio-correct* (*endpoint-in-S*  $a$   $b$ ) (*endpoint-in-S*  $b$   $a$ )  $a$   $b$   
**proof** –  
**from** *assms*  
**have** *are-endpoints-in-S* (*endpoint-in-S*  $a$   $b$ ) (*endpoint-in-S*  $b$   $a$ )  $a$   $b$   
**by** (*rule endpoints-in-S-are-endpoints-in-S*)  
**thus** *cross-ratio-correct* (*endpoint-in-S*  $a$   $b$ ) (*endpoint-in-S*  $b$   $a$ )  $a$   $b$   
**by** (*rule are-endpoints-in-S-cross-ratio-correct*)  
**qed**

**lemma** *endpoints-in-S-perp-foot-cross-ratio-correct*:  
**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and**  $c \in \text{hyp2}$  **and**  $a \neq b$   
**and** *proj2-incident*  $a$   $l$  **and** *proj2-incident*  $b$   $l$   
**shows** *cross-ratio-correct*  
(*endpoint-in-S*  $a$   $b$ ) (*endpoint-in-S*  $b$   $a$ )  $a$  (*perp-foot*  $c$   $l$ )  
(*is cross-ratio-correct*  $?p$   $?q$   $a$   $?d$ )  
**proof** –  
**from** *assms*  
**have** *are-endpoints-in-S*  $?p$   $?q$   $a$   $?d$   
**by** (*rule endpoints-in-S-perp-foot-are-endpoints-in-S*)  
**thus** *cross-ratio-correct*  $?p$   $?q$   $a$   $?d$   
**by** (*rule are-endpoints-in-S-cross-ratio-correct*)  
**qed**

**lemma** *cosh-dist-unique*:  
**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$  **and**  $c \in \text{hyp2}$  **and**  $p \in S$   
**and**  $B_{\mathbb{R}}$  (*cart2-pt*  $a$ ) (*cart2-pt*  $b$ ) (*cart2-pt*  $p$ ) (*is*  $B_{\mathbb{R}}$   $?ca$   $?cb$   $?cp$ )  
**and**  $B_{\mathbb{R}}$  (*cart2-pt*  $a$ ) (*cart2-pt*  $c$ ) (*cart2-pt*  $p$ ) (*is*  $B_{\mathbb{R}}$   $?ca$   $?cc$   $?cp$ )  
**and** *cosh-dist*  $a$   $b$  = *cosh-dist*  $a$   $c$  (*is*  $?ab$  =  $?ac$ )  
**shows**  $b = c$   
**proof** –  
**let**  $?q = \text{endpoint-in-S } p \ a$

**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle c \in \text{hyp2} \rangle$  **and**  $\langle p \in S \rangle$   
**have** *z-non-zero*  $a$  **and** *z-non-zero*  $b$  **and** *z-non-zero*  $c$  **and** *z-non-zero*  $p$   
**by** (*simp-all add: hyp2-S-z-non-zero*)  
**with**  $\langle B_{\mathbb{R}} \text{ ?ca ?cb ?cp} \rangle$  **and**  $\langle B_{\mathbb{R}} \text{ ?ca ?cc ?cp} \rangle$   
**have**  $\exists l. \text{proj2-incident } a \ l \wedge \text{proj2-incident } b \ l \wedge \text{proj2-incident } p \ l$   
**and**  $\exists m. \text{proj2-incident } a \ m \wedge \text{proj2-incident } c \ m \wedge \text{proj2-incident } p \ m$   
**by** (*simp-all add: euclid-B-cart2-common-line*)  
**then obtain**  $l$  **and**  $m$  **where**  
*proj2-incident*  $a \ l$  **and** *proj2-incident*  $b \ l$  **and** *proj2-incident*  $p \ l$   
**and** *proj2-incident*  $a \ m$  **and** *proj2-incident*  $c \ m$  **and** *proj2-incident*  $p \ m$   
**by** *auto*

**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle p \in S \rangle$  **have**  $a \neq p$  **by** (*rule hyp2-S-not-equal*)  
**with**  $\langle \text{proj2-incident } a \ l \rangle$  **and**  $\langle \text{proj2-incident } p \ l \rangle$   
**and**  $\langle \text{proj2-incident } a \ m \rangle$  **and**  $\langle \text{proj2-incident } p \ m \rangle$  **and** *proj2-incident-unique*  
**have**  $l = m$  **by** *fast*  
**with**  $\langle \text{proj2-incident } c \ m \rangle$  **have** *proj2-incident*  $c \ l$  **by** *simp*  
**with**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle c \in \text{hyp2} \rangle$  **and**  $\langle p \in S \rangle$   
**and**  $\langle \text{proj2-incident } a \ l \rangle$  **and**  $\langle \text{proj2-incident } b \ l \rangle$  **and**  $\langle \text{proj2-incident } p \ l \rangle$   
**have** *are-endpoints-in-S*  $p \ ?q \ b \ a$  **and** *are-endpoints-in-S*  $p \ ?q \ c \ a$   
**by** (*simp-all add: end-and-opposite-are-endpoints-in-S*)  
**with** *are-endpoints-in-S-swap-34*  
**have** *are-endpoints-in-S*  $p \ ?q \ a \ b$  **and** *are-endpoints-in-S*  $p \ ?q \ a \ c$  **by** *fast+*  
**hence** *cross-ratio-correct*  $p \ ?q \ a \ b$  **and** *cross-ratio-correct*  $p \ ?q \ a \ c$   
**by** (*simp-all add: are-endpoints-in-S-cross-ratio-correct*)  
**moreover**  
**from**  $\langle \text{are-endpoints-in-S } p \ ?q \ a \ b \rangle$  **and**  $\langle \text{are-endpoints-in-S } p \ ?q \ a \ c \rangle$   
**and**  $\langle B_{\mathbb{R}} \text{ ?ca ?cb ?cp} \rangle$  **and**  $\langle B_{\mathbb{R}} \text{ ?ca ?cc ?cp} \rangle$   
**have** *cross-ratio*  $p \ ?q \ a \ b = 2 * ?ab^2 + 2 * ?ab * \text{sqrt} (?ab^2 - 1) - 1$   
**and** *cross-ratio*  $p \ ?q \ a \ c = 2 * ?ac^2 + 2 * ?ac * \text{sqrt} (?ac^2 - 1) - 1$   
**by** (*simp-all add: cross-ratio-in-terms-of-cosh-dist*)  
**with**  $\langle ?ab = ?ac \rangle$  **have** *cross-ratio*  $p \ ?q \ a \ b = \text{cross-ratio } p \ ?q \ a \ c$  **by** *simp*  
**ultimately show**  $b = c$  **by** (*rule cross-ratio-unique*)

qed

**lemma** *cosh-dist-swap*:

**assumes**  $a \in \text{hyp2}$  **and**  $b \in \text{hyp2}$   
**shows** *cosh-dist*  $a \ b = \text{cosh-dist } b \ a$

**proof** –

**from** *assms* **and** *K2-isometry-swap*  
**obtain**  $J$  **where** *is-K2-isometry*  $J$  **and** *apply-cltn2*  $a \ J = b$   
**and** *apply-cltn2*  $b \ J = a$   
**by** *auto*

**from**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle \text{is-K2-isometry } J \rangle$   
**have** *cosh-dist* (*apply-cltn2*  $b \ J$ ) (*apply-cltn2*  $a \ J$ ) = *cosh-dist*  $b \ a$   
**by** (*rule K2-isometry-cosh-dist*)  
**with**  $\langle \text{apply-cltn2 } a \ J = b \rangle$  **and**  $\langle \text{apply-cltn2 } b \ J = a \rangle$

show  $\text{cosh-dist } a \ b = \text{cosh-dist } b \ a$  by *simp*  
qed

**lemma** *exp-2dist-1-equal*:

assumes  $a \in \text{hyp2}$  and  $b \in \text{hyp2}$  and  $\text{exp-2dist } a \ b = 1$   
shows  $a = b$

**proof** (rule *ccontr*)

assume  $a \neq b$

with  $\langle a \in \text{hyp2} \rangle$  and  $\langle b \in \text{hyp2} \rangle$

have *cross-ratio-correct* (*endpoint-in-S*  $a \ b$ ) (*endpoint-in-S*  $b \ a$ )  $a \ b$   
(is *cross-ratio-correct*  $?p \ ?q \ a \ b$ )

by (*simp add: endpoints-in-S-cross-ratio-correct*)

moreover

from  $\langle a \neq b \rangle$

have  $\text{exp-2dist } a \ b = \text{cross-ratio } ?p \ ?q \ a \ b$  by (*unfold exp-2dist-def*) *simp*

with  $\langle \text{exp-2dist } a \ b = 1 \rangle$  have *cross-ratio*  $?p \ ?q \ a \ b = 1$  by *simp*

ultimately have  $a = b$  by (*rule cross-ratio-1-equal*)

with  $\langle a \neq b \rangle$  show *False* ..

qed

### 8.11.1 A formula for a cross ratio involving a perpendicular foot

**lemma** *described-perp-foot-cross-ratio-formula*:

assumes  $a \neq b$  and  $c \in \text{hyp2}$  and *are-endpoints-in-S*  $p \ q \ a \ b$

and *proj2-incident*  $p \ l$  and *proj2-incident*  $q \ l$  and *M-perp*  $l \ m$

and *proj2-incident*  $d \ l$  and *proj2-incident*  $d \ m$  and *proj2-incident*  $c \ m$

shows *cross-ratio*  $p \ q \ d \ a$

$= (\text{cosh-dist } b \ c * \text{sqrt} (\text{cross-ratio } p \ q \ a \ b) - \text{cosh-dist } a \ c)$

$/ (\text{cosh-dist } a \ c * \text{cross-ratio } p \ q \ a \ b$

$- \text{cosh-dist } b \ c * \text{sqrt} (\text{cross-ratio } p \ q \ a \ b))$

(is  $?pqda = (?bc * \text{sqrt } ?pqab - ?ac) / (?ac * ?pqab - ?bc * \text{sqrt } ?pqab)$ )

**proof** –

let  $?da = \text{cosh-dist } d \ a$

let  $?db = \text{cosh-dist } d \ b$

let  $?dc = \text{cosh-dist } d \ c$

let  $?pqdb = \text{cross-ratio } p \ q \ d \ b$

from  $\langle \text{are-endpoints-in-S } p \ q \ a \ b \rangle$

have  $p \neq q$  and  $p \in S$  and  $q \in S$  and  $a \in \text{hyp2}$  and  $b \in \text{hyp2}$

and *proj2-set-Col*  $\{p, q, a, b\}$

by (*unfold are-endpoints-in-S-def*) *simp-all*

from  $\langle \text{proj2-set-Col } \{p, q, a, b\} \rangle$

obtain  $l'$  where *proj2-incident*  $p \ l'$  and *proj2-incident*  $q \ l'$

and *proj2-incident*  $a \ l'$  and *proj2-incident*  $b \ l'$

by (*unfold proj2-set-Col-def*) *auto*

from  $\langle p \neq q \rangle$  and  $\langle \text{proj2-incident } p \ l' \rangle$  and  $\langle \text{proj2-incident } q \ l' \rangle$

and  $\langle \text{proj2-incident } p \ l \rangle$  and  $\langle \text{proj2-incident } q \ l \rangle$  and *proj2-incident-unique*

**have**  $l' = l$  **by** *fast*  
**with**  $\langle \text{proj2-incident } a \ l' \rangle$  **and**  $\langle \text{proj2-incident } b \ l' \rangle$   
**have**  $\text{proj2-incident } a \ l$  **and**  $\text{proj2-incident } b \ l$  **by** *simp-all*

**from**  $\langle M\text{-perp } l \ m \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle \text{proj2-incident } a \ l \rangle$  **and**  $\langle c \in \text{hyp2} \rangle$   
**and**  $\langle \text{proj2-incident } c \ m \rangle$  **and**  $\langle \text{proj2-incident } d \ l \rangle$  **and**  $\langle \text{proj2-incident } d \ m \rangle$   
**have**  $d \in \text{hyp2}$  **by** (*rule M-perp-hyp2*)  
**with**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle c \in \text{hyp2} \rangle$   
**have**  $?bc > 0$  **and**  $?da > 0$  **and**  $?ac > 0$   
**by** (*simp-all add: cosh-dist-positive*)

**from**  $\langle \text{proj2-incident } p \ l \rangle$  **and**  $\langle \text{proj2-incident } q \ l \rangle$  **and**  $\langle \text{proj2-incident } d \ l \rangle$   
**and**  $\langle \text{proj2-incident } a \ l \rangle$  **and**  $\langle \text{proj2-incident } b \ l \rangle$   
**have**  $\text{proj2-set-Col } \{p, q, d, a\}$  **and**  $\text{proj2-set-Col } \{p, q, d, b\}$   
**and**  $\text{proj2-set-Col } \{p, q, a, b\}$   
**by** (*unfold proj2-set-Col-def*) (*simp-all add: exI [of - l]*)  
**with**  $\langle p \neq q \rangle$  **and**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$  **and**  $\langle d \in \text{hyp2} \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$   
**and**  $\langle b \in \text{hyp2} \rangle$   
**have**  $\text{are-endpoints-in-S } p \ q \ d \ a$  **and**  $\text{are-endpoints-in-S } p \ q \ d \ b$   
**and**  $\text{are-endpoints-in-S } p \ q \ a \ b$   
**by** (*unfold are-endpoints-in-S-def*) *simp-all*  
**hence**  $?pqda > 0$  **and**  $?pqdb > 0$  **and**  $?pqab > 0$   
**by** (*simp-all add: cross-ratio-S-S-hyp2-hyp2-positive*)

**from**  $\langle \text{proj2-incident } p \ l \rangle$  **and**  $\langle \text{proj2-incident } q \ l \rangle$  **and**  $\langle \text{proj2-incident } a \ l \rangle$   
**have**  $\text{proj2-Col } p \ q \ a$  **by** (*rule proj2-incident-Col*)

**from**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle p \in S \rangle$  **and**  $\langle q \in S \rangle$   
**have**  $a \neq p$  **and**  $a \neq q$  **and**  $b \neq p$  **by** (*simp-all add: hyp2-S-not-equal*)

**from**  $\langle \text{proj2-Col } p \ q \ a \rangle$  **and**  $\langle p \neq q \rangle$  **and**  $\langle a \neq p \rangle$  **and**  $\langle a \neq q \rangle$   
**have**  $?pqdb = ?pqda * ?pqab$  **by** (*rule cross-ratio-product [symmetric]*)

**from**  $\langle M\text{-perp } l \ m \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$  **and**  $\langle c \in \text{hyp2} \rangle$  **and**  $\langle d \in \text{hyp2} \rangle$   
**and**  $\langle \text{proj2-incident } a \ l \rangle$  **and**  $\langle \text{proj2-incident } b \ l \rangle$  **and**  $\langle \text{proj2-incident } d \ l \rangle$   
**and**  $\langle \text{proj2-incident } d \ m \rangle$  **and**  $\langle \text{proj2-incident } c \ m \rangle$   
**and**  $\text{cosh-dist-perp-divide } [of \ l \ m \ - \ d \ c]$   
**have**  $?dc = ?ac / ?da$  **and**  $?dc = ?bc / ?db$  **by** *fast+*  
**hence**  $?ac / ?da = ?bc / ?db$  **by** *simp*  
**with**  $\langle ?bc > 0 \rangle$  **and**  $\langle ?da > 0 \rangle$   
**have**  $?ac / ?bc = ?da / ?db$  **by** (*simp add: field-simps*)  
**also from**  $\langle \text{are-endpoints-in-S } p \ q \ d \ a \rangle$  **and**  $\langle \text{are-endpoints-in-S } p \ q \ d \ b \rangle$   
**have** ...  
 $= 2 * (\text{sqrt } ?pqda + 1 / (\text{sqrt } ?pqda))$   
 $/ (2 * (\text{sqrt } ?pqdb + 1 / (\text{sqrt } ?pqdb)))$   
**by** (*simp add: cosh-dist-general*)  
**also**  
**have** ...  $= (\text{sqrt } ?pqda + 1 / (\text{sqrt } ?pqda)) / (\text{sqrt } ?pqdb + 1 / (\text{sqrt } ?pqdb))$

by (*simp only: mult-divide-mult-cancel-left-if*) *simp*  
 also have ...  

$$= \text{sqrt } ?pqdb * (\text{sqrt } ?pqda + 1 / (\text{sqrt } ?pqda))$$

$$/ (\text{sqrt } ?pqdb * (\text{sqrt } ?pqdb + 1 / (\text{sqrt } ?pqdb)))$$
 by *simp*  
 also from  $\langle ?pqdb > 0 \rangle$   
 have ... =  $(\text{sqrt } (?pqdb * ?pqda) + \text{sqrt } (?pqdb / ?pqda)) / (?pqdb + 1)$   
 by (*simp add: real-sqrt-mult [symmetric] real-sqrt-divide algebra-simps*)  
 also from  $\langle ?pqdb = ?pqda * ?pqab \rangle$  and  $\langle ?pqda > 0 \rangle$  and *real-sqrt-pow2*  
 have ... =  $(?pqda * \text{sqrt } ?pqab + \text{sqrt } ?pqab) / (?pqda * ?pqab + 1)$   
 by (*simp add: real-sqrt-mult power2-eq-square*)  
 finally  
 have  $?ac / ?bc = (?pqda * \text{sqrt } ?pqab + \text{sqrt } ?pqab) / (?pqda * ?pqab + 1)$  .  
  
 from  $\langle ?pqda > 0 \rangle$  and  $\langle ?pqab > 0 \rangle$   
 have  $?pqda * ?pqab + 1 > 0$  by (*simp add: add-pos-pos*)  
 with  $\langle ?bc > 0 \rangle$   
 and  $\langle ?ac / ?bc = (?pqda * \text{sqrt } ?pqab + \text{sqrt } ?pqab) / (?pqda * ?pqab + 1) \rangle$   
 have  $?ac * (?pqda * ?pqab + 1) = ?bc * (?pqda * \text{sqrt } ?pqab + \text{sqrt } ?pqab)$   
 by (*simp add: field-simps*)  
 hence  $?pqda * (?ac * ?pqab - ?bc * \text{sqrt } ?pqab) = ?bc * \text{sqrt } ?pqab - ?ac$   
 by (*simp add: algebra-simps*)  
  
 from  $\langle \text{proj2-set-Col } \{p, q, a, b\} \rangle$  and  $\langle p \neq q \rangle$  and  $\langle a \neq p \rangle$  and  $\langle a \neq q \rangle$   
 and  $\langle b \neq p \rangle$   
 have *cross-ratio-correct p q a b* by (*unfold cross-ratio-correct-def*) *simp*  
  
 have  $?ac * ?pqab - ?bc * \text{sqrt } ?pqab \neq 0$   
 proof  
 assume  $?ac * ?pqab - ?bc * \text{sqrt } ?pqab = 0$   
 with  $\langle ?pqda * (?ac * ?pqab - ?bc * \text{sqrt } ?pqab) = ?bc * \text{sqrt } ?pqab - ?ac \rangle$   
 have  $?bc * \text{sqrt } ?pqab - ?ac = 0$  by *simp*  
 with  $\langle ?ac * ?pqab - ?bc * \text{sqrt } ?pqab = 0 \rangle$  and  $\langle ?ac > 0 \rangle$   
 have  $?pqab = 1$  by *simp*  
 with  $\langle \text{cross-ratio-correct p q a b} \rangle$   
 have  $a = b$  by (*rule cross-ratio-1-equal*)  
 with  $\langle a \neq b \rangle$  show *False ..*  
 qed  
 with  $\langle ?pqda * (?ac * ?pqab - ?bc * \text{sqrt } ?pqab) = ?bc * \text{sqrt } ?pqab - ?ac \rangle$   
 show  $?pqda = (?bc * \text{sqrt } ?pqab - ?ac) / (?ac * ?pqab - ?bc * \text{sqrt } ?pqab)$   
 by (*simp add: field-simps*)  
 qed  
  
 lemma *perp-foot-cross-ratio-formula*:  
 assumes  $a \in \text{hyp2}$  and  $b \in \text{hyp2}$  and  $c \in \text{hyp2}$  and  $a \neq b$   
 shows *cross-ratio (endpoint-in-S a b) (endpoint-in-S b a)*  
 $(\text{perp-foot } c (\text{proj2-line-through } a b)) a$   
 $= (\text{cosh-dist } b c * \text{sqrt } (\text{exp-2dist } a b) - \text{cosh-dist } a c)$   
 $/ (\text{cosh-dist } a c * \text{exp-2dist } a b - \text{cosh-dist } b c * \text{sqrt } (\text{exp-2dist } a b))$

(is cross-ratio ?p ?q ?d a  
 = (?bc \* sqrt ?pqab - ?ac) / (?ac \* ?pqab - ?bc \* sqrt ?pqab))

**proof** –

**from**  $\langle a \neq b \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$   
**have** are-endpoints-in-S ?p ?q a b  
**by** (rule endpoints-in-S-are-endpoints-in-S)

**let** ?l = proj2-line-through a b  
**have** proj2-incident a ?l **and** proj2-incident b ?l  
**by** (rule proj2-line-through-incident)+  
**with**  $\langle a \neq b \rangle$  **and**  $\langle a \in \text{hyp2} \rangle$  **and**  $\langle b \in \text{hyp2} \rangle$   
**have** proj2-incident ?p ?l **and** proj2-incident ?q ?l  
**by** (simp-all add: endpoint-in-S-incident)

**let** ?m = drop-perp c ?l  
**have** M-perp ?l ?m **by** (rule drop-perp-perp)

**have** proj2-incident ?d ?l **and** proj2-incident ?d ?m  
**by** (rule perp-foot-incident)+

**have** proj2-incident c ?m **by** (rule drop-perp-incident)  
**with**  $\langle a \neq b \rangle$  **and**  $\langle c \in \text{hyp2} \rangle$  **and**  $\langle \text{are-endpoints-in-S } ?p ?q a b \rangle$   
**and**  $\langle \text{proj2-incident } ?p ?l \rangle$  **and**  $\langle \text{proj2-incident } ?q ?l \rangle$  **and**  $\langle M\text{-perp } ?l ?m \rangle$   
**and**  $\langle \text{proj2-incident } ?d ?l \rangle$  **and**  $\langle \text{proj2-incident } ?d ?m \rangle$   
**have** cross-ratio ?p ?q ?d a  
 = (?bc \* sqrt (cross-ratio ?p ?q a b) - ?ac)  
 / (?ac \* (cross-ratio ?p ?q a b) - ?bc \* sqrt (cross-ratio ?p ?q a b))  
**by** (rule described-perp-foot-cross-ratio-formula)  
**with**  $\langle a \neq b \rangle$   
**show** cross-ratio ?p ?q ?d a  
 = (?bc \* sqrt ?pqab - ?ac) / (?ac \* ?pqab - ?bc \* sqrt ?pqab)  
**by** (unfold exp-2dist-def) simp

**qed**

## 8.12 The Klein–Beltrami model satisfies axiom 5

**lemma** *statement69*:

**assumes**  $a b \equiv_K a' b'$  **and**  $b c \equiv_K b' c'$  **and**  $a c \equiv_K a' c'$

**shows**  $\exists J. \text{is-K2-isometry } J$

$\wedge \text{hyp2-cltn2 } a J = a' \wedge \text{hyp2-cltn2 } b J = b' \wedge \text{hyp2-cltn2 } c J = c'$

**proof** *cases*

**assume**  $a = b$

**with**  $\langle a b \equiv_K a' b' \rangle$  **have**  $a' = b'$  **by** (simp add: hyp2.A3-reversed)

**with**  $\langle a = b \rangle$  **and**  $\langle b c \equiv_K b' c' \rangle$

**show**  $\exists J. \text{is-K2-isometry } J$

$\wedge \text{hyp2-cltn2 } a J = a' \wedge \text{hyp2-cltn2 } b J = b' \wedge \text{hyp2-cltn2 } c J = c'$

**by** (unfold real-hyp2-C-def) simp

**next**

**assume**  $a \neq b$

**with**  $\langle a \equiv_K b \rangle$   
**have**  $a' \neq b'$  **by** (*auto simp add: hyp2.A3*)

**let**  $?pa = \text{Rep-hyp2 } a$   
**and**  $?pb = \text{Rep-hyp2 } b$   
**and**  $?pc = \text{Rep-hyp2 } c$   
**and**  $?pa' = \text{Rep-hyp2 } a'$   
**and**  $?pb' = \text{Rep-hyp2 } b'$   
**and**  $?pc' = \text{Rep-hyp2 } c'$   
**define**  $pp \ pq \ l \ pp' \ pq' \ l'$   
**where**  $pp = \text{endpoint-in-S } ?pa \ ?pb$   
**and**  $pq = \text{endpoint-in-S } ?pb \ ?pa$   
**and**  $l = \text{proj2-line-through } ?pa \ ?pb$   
**and**  $pp' = \text{endpoint-in-S } ?pa' \ ?pb'$   
**and**  $pq' = \text{endpoint-in-S } ?pb' \ ?pa'$   
**and**  $l' = \text{proj2-line-through } ?pa' \ ?pb'$   
**define**  $pd \ ps \ m \ pd' \ ps' \ m'$   
**where**  $pd = \text{perp-foot } ?pc \ l$   
**and**  $ps = \text{perp-up } ?pc \ l$   
**and**  $m = \text{drop-perp } ?pc \ l$   
**and**  $pd' = \text{perp-foot } ?pc' \ l'$   
**and**  $ps' = \text{perp-up } ?pc' \ l'$   
**and**  $m' = \text{drop-perp } ?pc' \ l'$

**have**  $pp \in S$  **and**  $pp' \in S$  **and**  $pq \in S$  **and**  $pq' \in S$   
**unfolding**  $pp\text{-def}$  **and**  $pp'\text{-def}$  **and**  $pq\text{-def}$  **and**  $pq'\text{-def}$   
**by** (*simp-all add: Rep-hyp2 endpoint-in-S*)

**from**  $\langle a \neq b \rangle$  **and**  $\langle a' \neq b' \rangle$   
**have**  $?pa \neq ?pb$  **and**  $?pa' \neq ?pb'$  **by** (*unfold Rep-hyp2-inject*)  
**moreover**  
**have**  $\text{proj2-incident } ?pa \ l$  **and**  $\text{proj2-incident } ?pb \ l$   
**and**  $\text{proj2-incident } ?pa' \ l'$  **and**  $\text{proj2-incident } ?pb' \ l'$   
**by** (*unfold l-def l'-def*) (*rule proj2-line-through-incident*)  
**ultimately have**  $\text{proj2-incident } pp \ l$  **and**  $\text{proj2-incident } pp' \ l'$   
**and**  $\text{proj2-incident } pq \ l$  **and**  $\text{proj2-incident } pq' \ l'$   
**unfolding**  $pp\text{-def}$  **and**  $pp'\text{-def}$  **and**  $pq\text{-def}$  **and**  $pq'\text{-def}$   
**by** (*simp-all add: Rep-hyp2 endpoint-in-S-incident*)

**from**  $\langle pp \in S \rangle$  **and**  $\langle pp' \in S \rangle$  **and**  $\langle \text{proj2-incident } pp \ l \rangle$   
**and**  $\langle \text{proj2-incident } pp' \ l' \rangle$  **and**  $\langle \text{proj2-incident } ?pa \ l \rangle$   
**and**  $\langle \text{proj2-incident } ?pa' \ l' \rangle$   
**have**  $\text{right-angle } pp \ pd \ ps$  **and**  $\text{right-angle } pp' \ pd' \ ps'$   
**unfolding**  $pd\text{-def}$  **and**  $ps\text{-def}$  **and**  $pd'\text{-def}$  **and**  $ps'\text{-def}$   
**by** (*simp-all add: Rep-hyp2*  
 $\text{perp-foot-up-right-angle [of } pp \ ?pc \ ?pa \ l]$   
 $\text{perp-foot-up-right-angle [of } pp' \ ?pc' \ ?pa' \ l']$ )  
**with**  $\text{right-angle-to-right-angle [of } pp \ pd \ ps \ pp' \ pd' \ ps']$   
**obtain**  $J$  **where**  $\text{is-K2-isometry } J$  **and**  $\text{apply-cltn2 } pp \ J = pp'$

**and** *apply-cltn2* *pd J = pd'* **and** *apply-cltn2* *ps J = ps'*  
**by** *auto*

**let** *?paJ = apply-cltn2 ?pa J*  
**and** *?pbJ = apply-cltn2 ?pb J*  
**and** *?pcJ = apply-cltn2 ?pc J*  
**and** *?pdJ = apply-cltn2 pd J*  
**and** *?ppJ = apply-cltn2 pp J*  
**and** *?pqJ = apply-cltn2 pq J*  
**and** *?psJ = apply-cltn2 ps J*  
**and** *?lJ = apply-cltn2-line l J*  
**and** *?mJ = apply-cltn2-line m J*

**have** *proj2-incident pd l* **and** *proj2-incident pd' l'*  
**and** *proj2-incident pd m* **and** *proj2-incident pd' m'*  
**by** (*unfold pd-def pd'-def m-def m'-def*) (*rule perp-foot-incident*)+

**from**  $\langle$ *proj2-incident pp l* $\rangle$  **and**  $\langle$ *proj2-incident pq l* $\rangle$   
**and**  $\langle$ *proj2-incident pd l* $\rangle$  **and**  $\langle$ *proj2-incident ?pa l* $\rangle$   
**and**  $\langle$ *proj2-incident ?pb l* $\rangle$   
**have** *proj2-set-Col {pp,pq,pd,?pa}* **and** *proj2-set-Col {pp,pq,?pa,?pb}*  
**by** (*unfold pd-def proj2-set-Col-def*) (*simp-all add: exI [of - l]*)

**from**  $\langle$ *?pa  $\neq$  ?pb* $\rangle$  **and**  $\langle$ *?pa'  $\neq$  ?pb'* $\rangle$   
**have** *pp  $\neq$  pq* **and** *pp'  $\neq$  pq'*  
**unfolding** *pp-def* **and** *pq-def* **and** *pp'-def* **and** *pq'-def*  
**by** (*simp-all add: Rep-hyp2 endpoint-in-S-swap*)

**from**  $\langle$ *proj2-incident ?pa l* $\rangle$  **and**  $\langle$ *proj2-incident ?pa' l'* $\rangle$   
**have** *pd  $\in$  hyp2* **and** *pd'  $\in$  hyp2*  
**unfolding** *pd-def* **and** *pd'-def*  
**by** (*simp-all add: Rep-hyp2 perp-foot-hyp2 [of ?pa l ?pc]*  
*perp-foot-hyp2 [of ?pa' l' ?pc']*)

**from**  $\langle$ *proj2-incident ?pa l* $\rangle$  **and**  $\langle$ *proj2-incident ?pa' l'* $\rangle$   
**have** *ps  $\in$  S* **and** *ps'  $\in$  S*  
**unfolding** *ps-def* **and** *ps'-def*  
**by** (*simp-all add: Rep-hyp2 perp-up-in-S [of ?pc ?pa l]*  
*perp-up-in-S [of ?pc' ?pa' l']*)

**from**  $\langle$ *pd  $\in$  hyp2* $\rangle$  **and**  $\langle$ *pp  $\in$  S* $\rangle$  **and**  $\langle$ *ps  $\in$  S* $\rangle$   
**have** *pd  $\neq$  pp* **and** *?pa  $\neq$  pp* **and** *?pb  $\neq$  pp* **and** *pd  $\neq$  ps*  
**by** (*simp-all add: Rep-hyp2 hyp2-S-not-equal*)

**from**  $\langle$ *is-K2-isometry J* $\rangle$  **and**  $\langle$ *pq  $\in$  S* $\rangle$   
**have** *?pqJ  $\in$  S* **by** (*unfold is-K2-isometry-def*) *simp*

**from**  $\langle$ *pd  $\neq$  pp* $\rangle$  **and**  $\langle$ *proj2-incident pd l* $\rangle$  **and**  $\langle$ *proj2-incident pp l* $\rangle$   
**and**  $\langle$ *proj2-incident pd' l'* $\rangle$  **and**  $\langle$ *proj2-incident pp' l'* $\rangle$

**have**  $?lJ = l'$   
**unfolding**  $\langle ?pdJ = pd' \rangle$  [*symmetric*] **and**  $\langle ?ppJ = pp' \rangle$  [*symmetric*]  
**by** (*rule apply-cltn2-line-unique*)  
**from**  $\langle \text{proj2-incident } pq \ l \rangle$  **and**  $\langle \text{proj2-incident } ?pa \ l \rangle$   
**and**  $\langle \text{proj2-incident } ?pb \ l \rangle$   
**have**  $\text{proj2-incident } ?pqJ \ l'$  **and**  $\text{proj2-incident } ?paJ \ l'$   
**and**  $\text{proj2-incident } ?pbJ \ l'$   
**by** (*unfold*  $\langle ?lJ = l' \rangle$  [*symmetric*]) *simp-all*

**from**  $\langle ?pa' \neq ?pb' \rangle$  **and**  $\langle ?pqJ \in S \rangle$  **and**  $\langle \text{proj2-incident } ?pa' \ l' \rangle$   
**and**  $\langle \text{proj2-incident } ?pb' \ l' \rangle$  **and**  $\langle \text{proj2-incident } ?pqJ \ l' \rangle$   
**have**  $?pqJ = pp' \vee ?pqJ = pq'$   
**unfolding** *pp'-def* **and** *pq'-def*  
**by** (*simp add: Rep-hyp2 endpoints-in-S-incident-unique*)  
**moreover**  
**from**  $\langle pp \neq pq \rangle$  **and** *apply-cltn2-injective*  
**have**  $pp' \neq ?pqJ$  **by** (*unfold*  $\langle ?ppJ = pp' \rangle$  [*symmetric*]) *fast*  
**ultimately have**  $?pqJ = pq'$  **by** *simp*

**from**  $\langle ?pa' \neq ?pb' \rangle$   
**have** *cross-ratio*  $pp' \ pq' \ pd' \ ?pa'$   
 $= (\text{cosh-dist } ?pb' \ ?pc' * \text{sqrt} (\text{exp-2dist } ?pa' \ ?pb') - \text{cosh-dist } ?pa' \ ?pc')$   
 $/ (\text{cosh-dist } ?pa' \ ?pc' * \text{exp-2dist } ?pa' \ ?pb'$   
 $- \text{cosh-dist } ?pb' \ ?pc' * \text{sqrt} (\text{exp-2dist } ?pa' \ ?pb'))$   
**unfolding** *pp'-def* **and** *pq'-def* **and** *pd'-def* **and** *l'-def*  
**by** (*simp add: Rep-hyp2 perp-foot-cross-ratio-formula*)  
**also from** *assms*  
**have**  $\dots = (\text{cosh-dist } ?pb \ ?pc * \text{sqrt} (\text{exp-2dist } ?pa \ ?pb) - \text{cosh-dist } ?pa \ ?pc)$   
 $/ (\text{cosh-dist } ?pa \ ?pc * \text{exp-2dist } ?pa \ ?pb$   
 $- \text{cosh-dist } ?pb \ ?pc * \text{sqrt} (\text{exp-2dist } ?pa \ ?pb))$   
**by** (*simp add: real-hyp2-C-exp-2dist real-hyp2-C-cosh-dist*)  
**also from**  $\langle ?pa \neq ?pb \rangle$   
**have**  $\dots = \text{cross-ratio } pp \ pq \ pd \ ?pa$   
**unfolding** *pp-def* **and** *pq-def* **and** *pd-def* **and** *l-def*  
**by** (*simp add: Rep-hyp2 perp-foot-cross-ratio-formula*)  
**also from**  $\langle \text{proj2-set-Col } \{pp, pq, pd, ?pa\} \rangle$  **and**  $\langle pp \neq pq \rangle$  **and**  $\langle pd \neq pp \rangle$   
**and**  $\langle ?pa \neq pp \rangle$   
**have**  $\dots = \text{cross-ratio } ?ppJ \ ?pqJ \ ?pdJ \ ?paJ$  **by** (*simp add: cross-ratio-cltn2*)  
**also from**  $\langle ?ppJ = pp' \rangle$  **and**  $\langle ?pqJ = pq' \rangle$  **and**  $\langle ?pdJ = pd' \rangle$   
**have**  $\dots = \text{cross-ratio } pp' \ pq' \ pd' \ ?paJ$  **by** *simp*  
**finally**  
**have** *cross-ratio*  $pp' \ pq' \ pd' \ ?paJ = \text{cross-ratio } pp' \ pq' \ pd' \ ?pa'$  **by** *simp*

**from**  $\langle \text{is-K2-isometry } J \rangle$   
**have**  $?paJ \in \text{hyp2}$  **and**  $?pbJ \in \text{hyp2}$  **and**  $?pcJ \in \text{hyp2}$   
**by** (*rule apply-cltn2-Rep-hyp2*) $+$

**from**  $\langle \text{proj2-incident } pp' \ l' \rangle$  **and**  $\langle \text{proj2-incident } pq' \ l' \rangle$   
**and**  $\langle \text{proj2-incident } pd' \ l' \rangle$  **and**  $\langle \text{proj2-incident } ?paJ \ l' \rangle$

**and**  $\langle \text{proj2-incident } ?pa' l' \rangle$  **and**  $\langle \text{proj2-incident } ?pbJ l' \rangle$   
**and**  $\langle \text{proj2-incident } ?pb' l' \rangle$   
**have**  $\text{proj2-set-Col } \{pp', pq', pd', ?paJ\}$  **and**  $\text{proj2-set-Col } \{pp', pq', pd', ?pa'\}$   
**and**  $\text{proj2-set-Col } \{pp', pq', ?pa', ?pbJ\}$   
**and**  $\text{proj2-set-Col } \{pp', pq', ?pa', ?pb'\}$   
**by**  $(\text{unfold proj2-set-Col-def}) (\text{simp-all add: exI [of - l']})$   
**with**  $\langle pp' \neq pq' \rangle$  **and**  $\langle pp' \in S \rangle$  **and**  $\langle pq' \in S \rangle$  **and**  $\langle pd' \in \text{hyp2} \rangle$   
**and**  $\langle ?paJ \in \text{hyp2} \rangle$  **and**  $\langle ?pbJ \in \text{hyp2} \rangle$   
**have**  $\text{are-endpoints-in-S } pp' pq' pd' ?paJ$   
**and**  $\text{are-endpoints-in-S } pp' pq' pd' ?pa'$   
**and**  $\text{are-endpoints-in-S } pp' pq' ?pa' ?pbJ$   
**and**  $\text{are-endpoints-in-S } pp' pq' ?pa' ?pb'$   
**by**  $(\text{unfold are-endpoints-in-S-def}) (\text{simp-all add: Rep-hyp2})$   
**hence**  $\text{cross-ratio-correct } pp' pq' pd' ?paJ$   
**and**  $\text{cross-ratio-correct } pp' pq' pd' ?pa'$   
**and**  $\text{cross-ratio-correct } pp' pq' ?pa' ?pbJ$   
**and**  $\text{cross-ratio-correct } pp' pq' ?pa' ?pb'$   
**by**  $(\text{simp-all add: are-endpoints-in-S-cross-ratio-correct})$

**from**  $\langle \text{cross-ratio-correct } pp' pq' pd' ?paJ \rangle$   
**and**  $\langle \text{cross-ratio-correct } pp' pq' pd' ?pa' \rangle$   
**and**  $\langle \text{cross-ratio } pp' pq' pd' ?paJ = \text{cross-ratio } pp' pq' pd' ?pa' \rangle$   
**have**  $?paJ = ?pa'$  **by**  $(\text{simp add: cross-ratio-unique})$   
**with**  $\langle ?ppJ = pp' \rangle$  **and**  $\langle ?pqJ = pq' \rangle$   
**have**  $\text{cross-ratio } pp' pq' ?pa' ?pbJ = \text{cross-ratio } ?ppJ ?pqJ ?paJ ?pbJ$  **by**  $\text{simp}$   
**also from**  $\langle \text{proj2-set-Col } \{pp, pq, ?pa, ?pb\} \rangle$  **and**  $\langle pp \neq pq \rangle$  **and**  $\langle ?pa \neq pp \rangle$   
**and**  $\langle ?pb \neq pp \rangle$   
**have**  $\dots = \text{cross-ratio } pp pq ?pa ?pb$  **by**  $(\text{rule cross-ratio-cltn2})$   
**also from**  $\langle a \neq b \rangle$  **and**  $\langle a b \equiv_K a' b' \rangle$   
**have**  $\dots = \text{cross-ratio } pp' pq' ?pa' ?pb'$   
**unfolding**  $pp\text{-def } pq\text{-def } pp'\text{-def } pq'\text{-def}$   
**by**  $(\text{rule real-hyp2-C-cross-ratio-endpoints-in-S})$   
**finally have**  $\text{cross-ratio } pp' pq' ?pa' ?pbJ = \text{cross-ratio } pp' pq' ?pa' ?pb' .$   
**with**  $\langle \text{cross-ratio-correct } pp' pq' ?pa' ?pbJ \rangle$   
**and**  $\langle \text{cross-ratio-correct } pp' pq' ?pa' ?pb' \rangle$   
**have**  $?pbJ = ?pb'$  **by**  $(\text{rule cross-ratio-unique})$

**let**  $?cc = \text{cart2-pt } ?pc$   
**and**  $?cd = \text{cart2-pt } pd$   
**and**  $?cs = \text{cart2-pt } ps$   
**and**  $?cc' = \text{cart2-pt } ?pc'$   
**and**  $?cd' = \text{cart2-pt } pd'$   
**and**  $?cs' = \text{cart2-pt } ps'$   
**and**  $?ccJ = \text{cart2-pt } ?pcJ$   
**and**  $?cdJ = \text{cart2-pt } ?pdJ$   
**and**  $?csJ = \text{cart2-pt } ?psJ$

**from**  $\langle \text{proj2-incident } ?pa l \rangle$  **and**  $\langle \text{proj2-incident } ?pa' l' \rangle$   
**have**  $B_{\mathbb{R}} ?cd ?cc ?cs$  **and**  $B_{\mathbb{R}} ?cd' ?cc' ?cs'$

**unfolding**  $pd\text{-def}$  **and**  $ps\text{-def}$  **and**  $pd'\text{-def}$  **and**  $ps'\text{-def}$   
**by** (*simp-all add: Rep-hyp2 perp-up-at-end* [of  $?pc$   $?pa$   $l$ ]  
*perp-up-at-end* [of  $?pc'$   $?pa'$   $l'$ ])

**from**  $\langle pd \in hyp2 \rangle$  **and**  $\langle ps \in S \rangle$  **and**  $\langle is\text{-}K2\text{-isometry } J \rangle$   
**and**  $\langle B_{\mathbb{R}} ?cd ?cc ?cs \rangle$   
**have**  $B_{\mathbb{R}} ?cdJ ?ccJ ?csJ$  **by** (*simp add: Rep-hyp2 statement-63*)  
**hence**  $B_{\mathbb{R}} ?cd' ?ccJ ?cs'$  **by** (*unfold*  $\langle ?pdJ = pd' \rangle$   $\langle ?psJ = ps' \rangle$ )

**from**  $\langle ?paJ = ?pa' \rangle$  **have**  $cosh\text{-}dist ?pa' ?pcJ = cosh\text{-}dist ?paJ ?pcJ$  **by** *simp*  
**also from**  $\langle is\text{-}K2\text{-isometry } J \rangle$   
**have**  $\dots = cosh\text{-}dist ?pa ?pc$  **by** (*simp add: Rep-hyp2 K2-isometry-cosh-dist*)  
**also from**  $\langle a c \equiv_K a' c' \rangle$   
**have**  $\dots = cosh\text{-}dist ?pa' ?pc'$  **by** (*rule real-hyp2-C-cosh-dist*)  
**finally have**  $cosh\text{-}dist ?pa' ?pcJ = cosh\text{-}dist ?pa' ?pc'$  .

**have**  $M\text{-perp } l' m'$  **by** (*unfold*  $m'\text{-def}$ ) (*rule drop-perp-perp*)

**have**  $proj2\text{-incident } ?pc m$  **and**  $proj2\text{-incident } ?pc' m'$   
**by** (*unfold*  $m\text{-def}$   $m'\text{-def}$ ) (*rule drop-perp-incident*)+

**from**  $\langle proj2\text{-incident } ?pa l \rangle$  **and**  $\langle proj2\text{-incident } ?pa' l' \rangle$   
**have**  $proj2\text{-incident } ps m$  **and**  $proj2\text{-incident } ps' m'$   
**unfolding**  $ps\text{-def}$  **and**  $m\text{-def}$  **and**  $ps'\text{-def}$  **and**  $m'\text{-def}$   
**by** (*simp-all add: Rep-hyp2 perp-up-incident* [of  $?pc$   $?pa$   $l$ ]  
*perp-up-incident* [of  $?pc'$   $?pa'$   $l'$ ])

**with**  $\langle pd \neq ps \rangle$  **and**  $\langle proj2\text{-incident } pd m \rangle$  **and**  $\langle proj2\text{-incident } pd' m' \rangle$   
**have**  $?mJ = m'$   
**unfolding**  $\langle ?pdJ = pd' \rangle$  [*symmetric*] **and**  $\langle ?psJ = ps' \rangle$  [*symmetric*]  
**by** (*simp add: apply-cltn2-line-unique*)

**from**  $\langle proj2\text{-incident } ?pc m \rangle$   
**have**  $proj2\text{-incident } ?pcJ m'$  **by** (*unfold*  $\langle ?mJ = m' \rangle$  [*symmetric*]) *simp*  
**with**  $\langle M\text{-perp } l' m' \rangle$  **and**  $Rep\text{-hyp2}$  [of  $a'$ ] **and**  $\langle pd' \in hyp2 \rangle$  **and**  $\langle ?pcJ \in hyp2 \rangle$   
**and**  $Rep\text{-hyp2}$  [of  $c'$ ] **and**  $\langle proj2\text{-incident } ?pa' l' \rangle$   
**and**  $\langle proj2\text{-incident } pd' l' \rangle$  **and**  $\langle proj2\text{-incident } pd' m' \rangle$   
**and**  $\langle proj2\text{-incident } ?pc' m' \rangle$

**have**  $cosh\text{-}dist pd' ?pcJ = cosh\text{-}dist ?pa' ?pcJ / cosh\text{-}dist pd' ?pa'$   
**and**  $cosh\text{-}dist pd' ?pc' = cosh\text{-}dist ?pa' ?pc' / cosh\text{-}dist pd' ?pa'$   
**by** (*simp-all add: cosh-dist-perp-divide*)

**with**  $\langle cosh\text{-}dist ?pa' ?pcJ = cosh\text{-}dist ?pa' ?pc' \rangle$   
**have**  $cosh\text{-}dist pd' ?pcJ = cosh\text{-}dist pd' ?pc'$  **by** *simp*

**with**  $\langle pd' \in hyp2 \rangle$  **and**  $\langle ?pcJ \in hyp2 \rangle$  **and**  $\langle ?pc' \in hyp2 \rangle$  **and**  $\langle ps' \in S \rangle$   
**and**  $\langle B_{\mathbb{R}} ?cd' ?ccJ ?cs' \rangle$  **and**  $\langle B_{\mathbb{R}} ?cd' ?cc' ?cs' \rangle$

**have**  $?pcJ = ?pc'$  **by** (*rule cosh-dist-unique*)

**with**  $\langle ?paJ = ?pa' \rangle$  **and**  $\langle ?pbJ = ?pb' \rangle$

**have**  $hyp2\text{-cltn2 } a J = a'$  **and**  $hyp2\text{-cltn2 } b J = b'$  **and**  $hyp2\text{-cltn2 } c J = c'$   
**by** (*unfold*  $hyp2\text{-cltn2-def}$ ) (*simp-all add: Rep-hyp2-inverse*)

**with**  $\langle is\text{-}K2\text{-isometry } J \rangle$   
**show**  $\exists J. is\text{-}K2\text{-isometry } J$

$\wedge \text{hyp2-cltn2 } a J = a' \wedge \text{hyp2-cltn2 } b J = b' \wedge \text{hyp2-cltn2 } c J = c'$   
 by (*simp add: exI [of - J]*)

qed

**theorem** *hyp2-axiom5*:

$\forall a b c d a' b' c' d'.$   
 $a \neq b \wedge B_K a b c \wedge B_K a' b' c' \wedge a b \equiv_K a' b' \wedge b c \equiv_K b' c'$   
 $\wedge a d \equiv_K a' d' \wedge b d \equiv_K b' d'$   
 $\longrightarrow c d \equiv_K c' d'$

**proof** *standard+*

**fix**  $a b c d a' b' c' d'$   
**assume**  $a \neq b \wedge B_K a b c \wedge B_K a' b' c' \wedge a b \equiv_K a' b' \wedge b c \equiv_K b' c'$   
 $\wedge a d \equiv_K a' d' \wedge b d \equiv_K b' d'$   
**hence**  $a \neq b$  **and**  $B_K a b c$  **and**  $B_K a' b' c'$  **and**  $a b \equiv_K a' b'$   
**and**  $b c \equiv_K b' c'$  **and**  $a d \equiv_K a' d'$  **and**  $b d \equiv_K b' d'$   
 by *simp-all*

**from**  $\langle a b \equiv_K a' b' \rangle$  **and**  $\langle b d \equiv_K b' d' \rangle$  **and**  $\langle a d \equiv_K a' d' \rangle$  **and** *statement69*  
 [*of a b a' b' d d'*]

**obtain**  $J$  **where** *is-K2-isometry*  $J$  **and**  $\text{hyp2-cltn2 } a J = a'$   
**and**  $\text{hyp2-cltn2 } b J = b'$  **and**  $\text{hyp2-cltn2 } d J = d'$   
 by *auto*

**let**  $?aJ = \text{hyp2-cltn2 } a J$   
**and**  $?bJ = \text{hyp2-cltn2 } b J$   
**and**  $?cJ = \text{hyp2-cltn2 } c J$   
**and**  $?dJ = \text{hyp2-cltn2 } d J$

**from**  $\langle a \neq b \rangle$  **and**  $\langle a b \equiv_K a' b' \rangle$   
**have**  $a' \neq b'$  **by** (*auto simp add: hyp2.A3'*)

**from**  $\langle \text{is-K2-isometry } J \rangle$  **and**  $\langle B_K a b c \rangle$   
**have**  $B_K ?aJ ?bJ ?cJ$  **by** (*rule real-hyp2-B-hyp2-cltn2*)  
**hence**  $B_K a' b' ?cJ$  **by** (*unfold*  $\langle ?aJ = a' \rangle$   $\langle ?bJ = b' \rangle$ )

**from**  $\langle \text{is-K2-isometry } J \rangle$   
**have**  $b c \equiv_K ?bJ ?cJ$  **by** (*rule real-hyp2-C-hyp2-cltn2*)  
**hence**  $b c \equiv_K b' ?cJ$  **by** (*unfold*  $\langle ?bJ = b' \rangle$ )  
**from** *this* **and**  $\langle b c \equiv_K b' c' \rangle$  **have**  $b' ?cJ \equiv_K b' c'$  **by** (*rule hyp2.A2'*)  
**with**  $\langle a' \neq b' \rangle$  **and**  $\langle B_K a' b' ?cJ \rangle$  **and**  $\langle B_K a' b' c' \rangle$   
**have**  $?cJ = c'$  **by** (*rule hyp2-extend-segment-unique*)  
**from**  $\langle \text{is-K2-isometry } J \rangle$   
**show**  $c d \equiv_K c' d'$   
**unfolding**  $\langle ?cJ = c' \rangle$  [*symmetric*] **and**  $\langle ?dJ = d' \rangle$  [*symmetric*]  
**by** (*rule real-hyp2-C-hyp2-cltn2*)

qed

**interpretation** *hyp2: tarski-first5 real-hyp2-C real-hyp2-B*  
 using *hyp2-axiom4* **and** *hyp2-axiom5*

by *unfold-locales*

### 8.13 The Klein–Beltrami model satisfies axioms 6, 7, and 11

**theorem** *hyp2-axiom6*:  $\forall a b. B_K a b a \longrightarrow a = b$

**proof** *standard+*

**fix**  $a b$

**let**  $?ca = \text{cart2-pt } (\text{Rep-hyp2 } a)$

**and**  $?cb = \text{cart2-pt } (\text{Rep-hyp2 } b)$

**assume**  $B_K a b a$

**hence**  $B_{\mathbb{R}} ?ca ?cb ?ca$  **by** (*unfold real-hyp2-B-def hyp2-rep-def*)

**hence**  $?ca = ?cb$  **by** (*rule real-euclid.A6'*)

**hence**  $\text{Rep-hyp2 } a = \text{Rep-hyp2 } b$  **by** (*simp add: Rep-hyp2 hyp2-S-cart2-inj*)

**thus**  $a = b$  **by** (*unfold Rep-hyp2-inject*)

**qed**

**lemma** *between-inverse*:

**assumes**  $B_{\mathbb{R}} (\text{hyp2-rep } p) v (\text{hyp2-rep } q)$

**shows**  $\text{hyp2-rep } (\text{hyp2-abs } v) = v$

**proof**  $-$

**let**  $?u = \text{hyp2-rep } p$

**let**  $?w = \text{hyp2-rep } q$

**have**  $\text{norm } ?u < 1$  **and**  $\text{norm } ?w < 1$  **by** (*rule norm-hyp2-rep-lt-1*) $+$

**from**  $\langle B_{\mathbb{R}} ?u v ?w \rangle$

**obtain**  $l$  **where**  $l \geq 0$  **and**  $l \leq 1$  **and**  $v - ?u = l *_R (?w - ?u)$

**by** (*unfold real-euclid-B-def*) *auto*

**from**  $\langle v - ?u = l *_R (?w - ?u) \rangle$

**have**  $v = l *_R ?w + (1 - l) *_R ?u$  **by** (*simp add: algebra-simps*)

**hence**  $\text{norm } v \leq \text{norm } (l *_R ?w) + \text{norm } ((1 - l) *_R ?u)$

**by** (*simp only: norm-triangle-ineq [of  $l *_R ?w (1 - l) *_R ?u$ ]*)

**with**  $\langle l \geq 0 \rangle$  **and**  $\langle l \leq 1 \rangle$

**have**  $\text{norm } v \leq l *_R \text{norm } ?w + (1 - l) *_R \text{norm } ?u$  **by** *simp*

**have**  $\text{norm } v < 1$

**proof** *cases*

**assume**  $l = 0$

**with**  $\langle v = l *_R ?w + (1 - l) *_R ?u \rangle$

**have**  $v = ?u$  **by** *simp*

**with**  $\langle \text{norm } ?u < 1 \rangle$  **show**  $\text{norm } v < 1$  **by** *simp*

**next**

**assume**  $l \neq 0$

**with**  $\langle \text{norm } ?w < 1 \rangle$  **and**  $\langle l \geq 0 \rangle$  **have**  $l *_R \text{norm } ?w < l$  **by** *simp*

**with**  $\langle \text{norm } ?u < 1 \rangle$  **and**  $\langle l \leq 1 \rangle$

**and** *mult-mono* [*of  $1 - l$   $1 - l$   $\text{norm } ?u$   $1$ ]*

**have**  $(1 - l) *_R \text{norm } ?u \leq 1 - l$  **by** *simp*

**with**  $\langle l *_R \text{norm } ?w < l \rangle$

**have**  $l *_R \text{norm } ?w + (1 - l) *_R \text{norm } ?u < 1$  **by** *simp*

**with**  $\langle \text{norm } v \leq l *_R \text{norm } ?w + (1 - l) *_R \text{norm } ?u \rangle$   
**show**  $\text{norm } v < 1$  **by** *simp*  
**qed**  
**thus**  $\text{hyp2-rep } (\text{hyp2-abs } v) = v$  **by** (*rule hyp2-rep-abs*)  
**qed**

**lemma** *between-switch*:

**assumes**  $B_R (\text{hyp2-rep } p) v (\text{hyp2-rep } q)$   
**shows**  $B_K p (\text{hyp2-abs } v) q$   
**proof** –  
**from** *assms* **have**  $\text{hyp2-rep } (\text{hyp2-abs } v) = v$  **by** (*rule between-inverse*)  
**with** *assms* **show**  $B_K p (\text{hyp2-abs } v) q$  **by** (*unfold real-hyp2-B-def*) *simp*  
**qed**

**theorem** *hyp2-axiom7*:

$\forall a b c p q. B_K a p c \wedge B_K b q c \longrightarrow (\exists x. B_K p x b \wedge B_K q x a)$   
**proof** *auto*  
**fix**  $a b c p q$   
**let**  $?ca = \text{hyp2-rep } a$   
**and**  $?cb = \text{hyp2-rep } b$   
**and**  $?cc = \text{hyp2-rep } c$   
**and**  $?cp = \text{hyp2-rep } p$   
**and**  $?cq = \text{hyp2-rep } q$   
**assume**  $B_K a p c$  **and**  $B_K b q c$   
**hence**  $B_R ?ca ?cp ?cc$  **and**  $B_R ?cb ?cq ?cc$  **by** (*unfold real-hyp2-B-def*)  
**with** *real-euclid.A7'* [*of ?ca ?cp ?cc ?cb ?cq*]  
**obtain**  $cx$  **where**  $B_R ?cp cx ?cb$  **and**  $B_R ?cq cx ?ca$  **by** *auto*  
**hence**  $B_K p (\text{hyp2-abs } cx) b$  **and**  $B_K q (\text{hyp2-abs } cx) a$   
**by** (*simp-all add: between-switch*)  
**thus**  $\exists x. B_K p x b \wedge B_K q x a$  **by** (*simp add: exI [of - hyp2-abs cx]*)  
**qed**

**theorem** *hyp2-axiom11*:

$\forall X Y. (\exists a. \forall x y. x \in X \wedge y \in Y \longrightarrow B_K a x y)$   
 $\longrightarrow (\exists b. \forall x y. x \in X \wedge y \in Y \longrightarrow B_K x b y)$   
**proof** (*rule allI*)  
**fix**  $X Y :: \text{hyp2 set}$   
**show**  $(\exists a. \forall x y. x \in X \wedge y \in Y \longrightarrow B_K a x y)$   
 $\longrightarrow (\exists b. \forall x y. x \in X \wedge y \in Y \longrightarrow B_K x b y)$   
**proof** *cases*  
**assume**  $X = \{\} \vee Y = \{\}$   
**thus**  $(\exists a. \forall x y. x \in X \wedge y \in Y \longrightarrow B_K a x y)$   
 $\longrightarrow (\exists b. \forall x y. x \in X \wedge y \in Y \longrightarrow B_K x b y)$  **by** *auto*  
**next**  
**assume**  $\neg (X = \{\} \vee Y = \{\})$   
**hence**  $X \neq \{\}$  **and**  $Y \neq \{\}$  **by** *simp-all*  
**then obtain**  $w$  **and**  $z$  **where**  $w \in X$  **and**  $z \in Y$  **by** *auto*  
  
**show**  $(\exists a. \forall x y. x \in X \wedge y \in Y \longrightarrow B_K a x y)$

$\longrightarrow (\exists b. \forall x y. x \in X \wedge y \in Y \longrightarrow B_K x b y)$

**proof**

**assume**  $\exists a. \forall x y. x \in X \wedge y \in Y \longrightarrow B_K a x y$

**then obtain a where**  $\forall x y. x \in X \wedge y \in Y \longrightarrow B_K a x y ..$

**let**  $?cX = \text{hyp2-rep } \langle X$   
**and**  $?cY = \text{hyp2-rep } \langle Y$   
**and**  $?ca = \text{hyp2-rep } a$   
**and**  $?cw = \text{hyp2-rep } w$   
**and**  $?cz = \text{hyp2-rep } z$

**from**  $\langle \forall x y. x \in X \wedge y \in Y \longrightarrow B_K a x y \rangle$

**have**  $\forall cx cy. cx \in ?cX \wedge cy \in ?cY \longrightarrow B_{\mathbb{R}} ?ca cx cy$

**by**  $(\text{unfold real-hyp2-B-def}) \text{ auto}$

**with**  $\text{real-euclid.A11' [of ?cX ?cY ?ca]}$

**obtain cb where**  $\forall cx cy. cx \in ?cX \wedge cy \in ?cY \longrightarrow B_{\mathbb{R}} cx cb cy$  **by**  $\text{auto}$

**with**  $\langle w \in X \rangle$  **and**  $\langle z \in Y \rangle$  **have**  $B_{\mathbb{R}} ?cw cb ?cz$  **by**  $\text{simp}$

**hence**  $\text{hyp2-rep } (\text{hyp2-abs } cb) = cb$  **(is**  $\text{hyp2-rep } ?b = cb)$

**by**  $(\text{rule between-inverse})$

**with**  $\langle \forall cx cy. cx \in ?cX \wedge cy \in ?cY \longrightarrow B_{\mathbb{R}} cx cb cy \rangle$

**have**  $\forall x y. x \in X \wedge y \in Y \longrightarrow B_K x ?b y$

**by**  $(\text{unfold real-hyp2-B-def}) \text{ simp}$

**thus**  $\exists b. \forall x y. x \in X \wedge y \in Y \longrightarrow B_K x b y$  **by**  $(\text{rule exI})$

**qed**

**qed**

**qed**

**interpretation**  $\text{tarski-absolute-space real-hyp2-C real-hyp2-B}$

**using**  $\text{hyp2-axiom6}$  **and**  $\text{hyp2-axiom7}$  **and**  $\text{hyp2-axiom11}$

**by**  $\text{unfold-locales}$

## 8.14 The Klein–Beltrami model satisfies the dimension-specific axioms

**lemma**  $\text{hyp2-rep-abs-examples}$ :

**shows**  $\text{hyp2-rep } (\text{hyp2-abs } 0) = 0$  **(is**  $\text{hyp2-rep } ?a = ?ca)$

**and**  $\text{hyp2-rep } (\text{hyp2-abs } (\text{vector } [1/2, 0])) = \text{vector } [1/2, 0]$

**(is**  $\text{hyp2-rep } ?b = ?cb)$

**and**  $\text{hyp2-rep } (\text{hyp2-abs } (\text{vector } [0, 1/2])) = \text{vector } [0, 1/2]$

**(is**  $\text{hyp2-rep } ?c = ?cc)$

**and**  $\text{hyp2-rep } (\text{hyp2-abs } (\text{vector } [1/4, 1/4])) = \text{vector } [1/4, 1/4]$

**(is**  $\text{hyp2-rep } ?d = ?cd)$

**and**  $\text{hyp2-rep } (\text{hyp2-abs } (\text{vector } [1/2, 1/2])) = \text{vector } [1/2, 1/2]$

**(is**  $\text{hyp2-rep } ?t = ?ct)$

**proof** –

**have**  $\text{norm } ?ca < 1$  **and**  $\text{norm } ?cb < 1$  **and**  $\text{norm } ?cc < 1$  **and**  $\text{norm } ?cd < 1$

**and**  $\text{norm } ?ct < 1$

**by**  $(\text{unfold norm-vec-def L2-set-def}) (\text{simp-all add: sum-2 power2-eq-square})$

**thus**  $\text{hyp2-rep } ?a = ?ca$  **and**  $\text{hyp2-rep } ?b = ?cb$  **and**  $\text{hyp2-rep } ?c = ?cc$

and *hyp2-rep* ?d = ?cd and *hyp2-rep* ?t = ?ct  
 by (*simp-all add: hyp2-rep-abs*)

qed

**theorem** *hyp2-axiom8*:  $\exists a b c. \neg B_K a b c \wedge \neg B_K b c a \wedge \neg B_K c a b$

**proof** –

let ?ca = 0 :: real<sup>2</sup>

and ?cb = vector [1/2,0] :: real<sup>2</sup>

and ?cc = vector [0,1/2] :: real<sup>2</sup>

let ?a = *hyp2-abs* ?ca

and ?b = *hyp2-abs* ?cb

and ?c = *hyp2-abs* ?cc

from *hyp2-rep-abs-examples* and *non-Col-example*

have  $\neg$  (*hyp2.Col* ?a ?b ?c)

by (*unfold hyp2.Col-def real-euclid.Col-def real-hyp2-B-def*) *simp*

thus  $\exists a b c. \neg B_K a b c \wedge \neg B_K b c a \wedge \neg B_K c a b$

unfolding *hyp2.Col-def*

by *simp (rule exI)*+

qed

**theorem** *hyp2-axiom9*:

$\forall p q a b c. p \neq q \wedge a p \equiv_K a q \wedge b p \equiv_K b q \wedge c p \equiv_K c q$

$\longrightarrow B_K a b c \vee B_K b c a \vee B_K c a b$

**proof** (*rule allI*)+

fix p q a b c

show  $p \neq q \wedge a p \equiv_K a q \wedge b p \equiv_K b q \wedge c p \equiv_K c q$

$\longrightarrow B_K a b c \vee B_K b c a \vee B_K c a b$

**proof**

assume  $p \neq q \wedge a p \equiv_K a q \wedge b p \equiv_K b q \wedge c p \equiv_K c q$

hence  $p \neq q$  and  $a p \equiv_K a q$  and  $b p \equiv_K b q$  and  $c p \equiv_K c q$  by *simp-all*

let ?pp = *Rep-hyp2* p

and ?pq = *Rep-hyp2* q

and ?pa = *Rep-hyp2* a

and ?pb = *Rep-hyp2* b

and ?pc = *Rep-hyp2* c

define l where l = *proj2-line-through* ?pp ?pq

define m ps pt stpq

where m = *drop-perp* ?pa l

and ps = *endpoint-in-S* ?pp ?pq

and pt = *endpoint-in-S* ?pq ?pp

and stpq = *exp-2dist* ?pp ?pq

from  $\langle p \neq q \rangle$  have ?pp  $\neq$  ?pq by (*simp add: Rep-hyp2-inject*)

from *Rep-hyp2*

have stpq > 0 by (*unfold stpq-def*) (*simp add: exp-2dist-positive*)

hence sqrt stpq \* sqrt stpq = stpq

by (*simp add: real-sqrt-mult [symmetric]*)

**from** *Rep-hyp2* **and**  $\langle ?pp \neq ?pq \rangle$   
**have**  $stpq \neq 1$  **by** (*unfold stpq-def*) (*auto simp add: exp-2dist-1-equal*)

**have** *z-non-zero*  $?pa$  **and** *z-non-zero*  $?pb$  **and** *z-non-zero*  $?pc$   
**by** (*simp-all add: Rep-hyp2 hyp2-S-z-non-zero*)

**have**  $\forall pd \in \{?pa, ?pb, ?pc\}$ .  
*cross-ratio ps pt (perp-foot pd l) ?pp = 1 / (sqrt stpq)*

**proof**  
**fix**  $pd$   
**assume**  $pd \in \{?pa, ?pb, ?pc\}$   
**with** *Rep-hyp2* **have**  $pd \in hyp2$  **by** *auto*

**define**  $pe\ x$   
**where**  $pe = perp-foot\ pd\ l$   
**and**  $x = cosh-dist\ ?pp\ pd$

**from**  $\langle pd \in \{?pa, ?pb, ?pc\} \rangle$  **and**  $\langle a\ p \equiv_K\ a\ q \rangle$  **and**  $\langle b\ p \equiv_K\ b\ q \rangle$   
**and**  $\langle c\ p \equiv_K\ c\ q \rangle$   
**have**  $cosh-dist\ pd\ ?pp = cosh-dist\ pd\ ?pq$   
**by** (*auto simp add: real-hyp2-C-cosh-dist*)  
**with**  $\langle pd \in hyp2 \rangle$  **and** *Rep-hyp2*  
**have**  $x = cosh-dist\ ?pq\ pd$  **by** (*unfold x-def*) (*simp add: cosh-dist-swap*)

**from** *Rep-hyp2* [*of p*] **and**  $\langle pd \in hyp2 \rangle$  **and** *cosh-dist-positive* [*of ?pp pd*]  
**have**  $x \neq 0$  **by** (*unfold x-def*) *simp*

**from** *Rep-hyp2* **and**  $\langle pd \in hyp2 \rangle$  **and**  $\langle ?pp \neq ?pq \rangle$   
**have** *cross-ratio ps pt pe ?pp*  
 $= (cosh-dist\ ?pq\ pd * sqrt\ stpq - cosh-dist\ ?pp\ pd)$   
 $/ (cosh-dist\ ?pp\ pd * sqrt\ stpq - cosh-dist\ ?pq\ pd * sqrt\ stpq)$   
**unfolding** *ps-def* **and** *pt-def* **and** *pe-def* **and** *l-def* **and** *stpq-def*  
**by** (*simp add: perp-foot-cross-ratio-formula*)  
**also from** *x-def* **and**  $\langle x = cosh-dist\ ?pq\ pd \rangle$   
**have**  $\dots = (x * sqrt\ stpq - x) / (x * sqrt\ stpq - x * sqrt\ stpq)$  **by** *simp*  
**also from**  $\langle sqrt\ stpq * sqrt\ stpq = stpq \rangle$   
**have**  $\dots = (x * sqrt\ stpq - x) / ((x * sqrt\ stpq - x) * sqrt\ stpq)$   
**by** (*simp add: algebra-simps*)  
**also from**  $\langle x \neq 0 \rangle$  **and**  $\langle stpq \neq 1 \rangle$  **have**  $\dots = 1 / sqrt\ stpq$  **by** *simp*  
**finally show** *cross-ratio ps pt pe ?pp = 1 / sqrt stpq* .

**qed**  
**hence** *cross-ratio ps pt (perp-foot ?pa l) ?pp = 1 / sqrt stpq* **by** *simp*

**have**  $\forall pd \in \{?pa, ?pb, ?pc\}$ . *proj2-incident pd m*

**proof**  
**fix**  $pd$   
**assume**  $pd \in \{?pa, ?pb, ?pc\}$   
**with** *Rep-hyp2* **have**  $pd \in hyp2$  **by** *auto*

**with** *Rep-hyp2* **and**  $\langle ?pp \neq ?pq \rangle$  **and** *proj2-line-through-incident*  
**have** *cross-ratio-correct* *ps pt ?pp* (*perp-foot* *pd l*)  
**and** *cross-ratio-correct* *ps pt ?pp* (*perp-foot* *?pa l*)  
**unfolding** *ps-def* **and** *pt-def* **and** *l-def*  
**by** (*simp-all add: endpoints-in-S-perp-foot-cross-ratio-correct*)

**from**  $\langle pd \in \{?pa, ?pb, ?pc\} \rangle$   
**and**  $\langle \forall pd \in \{?pa, ?pb, ?pc\}. \text{cross-ratio } ps \ pt \ (perp-foot \ pd \ l) \ ?pp = 1 / (sqrt \ stpq) \rangle$   
**have** *cross-ratio* *ps pt* (*perp-foot* *pd l*)  $?pp = 1 / sqrt \ stpq$  **by** *auto*  
**with**  $\langle \text{cross-ratio } ps \ pt \ (perp-foot \ ?pa \ l) \ ?pp = 1 / sqrt \ stpq \rangle$   
**have** *cross-ratio* *ps pt* (*perp-foot* *pd l*)  $?pp$   
 $= \text{cross-ratio } ps \ pt \ (perp-foot \ ?pa \ l) \ ?pp$   
**by** *simp*  
**hence** *cross-ratio* *ps pt*  $?pp$  (*perp-foot* *pd l*)  
 $= \text{cross-ratio } ps \ pt \ ?pp$  (*perp-foot* *?pa l*)  
**by** (*simp add: cross-ratio-swap-34* [*of ps pt - ?pp*])  
**with**  $\langle \text{cross-ratio-correct } ps \ pt \ ?pp \ (perp-foot \ pd \ l) \rangle$   
**and**  $\langle \text{cross-ratio-correct } ps \ pt \ ?pp \ (perp-foot \ ?pa \ l) \rangle$   
**have** *perp-foot* *pd l*  $= \text{perp-foot } ?pa \ l$  **by** (*rule cross-ratio-unique*)  
**with** *Rep-hyp2* [*of p*] **and**  $\langle pd \in hyp2 \rangle$   
**and** *proj2-line-through-incident* [*of ?pp ?pq*]  
**and** *perp-foot-eq-implies-drop-perp-eq* [*of ?pp pd l ?pa*]  
**have** *drop-perp* *pd l*  $= m$  **by** (*unfold m-def l-def*) *simp*  
**with** *drop-perp-incident* [*of pd l*] **show** *proj2-incident* *pd m* **by** *simp*  
**qed**

**hence** *proj2-set-Col*  $\{?pa, ?pb, ?pc\}$   
**by** (*unfold proj2-set-Col-def*) (*simp add: exI* [*of - m*])  
**hence** *proj2-Col* *?pa ?pb ?pc* **by** (*simp add: proj2-Col-iff-set-Col*)  
**with**  $\langle z\text{-non-zero } ?pa \rangle$  **and**  $\langle z\text{-non-zero } ?pb \rangle$  **and**  $\langle z\text{-non-zero } ?pc \rangle$   
**have** *real-euclid.Col* (*hyp2-rep a*) (*hyp2-rep b*) (*hyp2-rep c*)  
**by** (*unfold hyp2-rep-def*) (*simp add: proj2-Col-iff-euclid-cart2*)  
**thus**  $B_K \ a \ b \ c \vee B_K \ b \ c \ a \vee B_K \ c \ a \ b$   
**by** (*unfold real-hyp2-B-def real-euclid.Col-def*)  
**qed**

**qed**

**interpretation** *hyp2: tarski-absolute real-hyp2-C real-hyp2-B*  
**using** *hyp2-axiom8* **and** *hyp2-axiom9*  
**by** *unfold-locales*

## 8.15 The Klein–Beltrami model violates the Euclidean axiom

**theorem** *hyp2-axiom10-false*:

**shows**  $\neg (\forall a \ b \ c \ d \ t. B_K \ a \ d \ t \wedge B_K \ b \ d \ c \wedge a \neq d$   
 $\longrightarrow (\exists x \ y. B_K \ a \ b \ x \wedge B_K \ a \ c \ y \wedge B_K \ x \ t \ y))$

**proof**

**assume**  $\forall a \ b \ c \ d \ t. B_K \ a \ d \ t \wedge B_K \ b \ d \ c \wedge a \neq d$

→ (∃ x y. B<sub>K</sub> a b x ∧ B<sub>K</sub> a c y ∧ B<sub>K</sub> x t y)

**let** ?ca = 0 :: real<sup>2</sup>  
**and** ?cb = vector [1/2,0] :: real<sup>2</sup>  
**and** ?cc = vector [0,1/2] :: real<sup>2</sup>  
**and** ?cd = vector [1/4,1/4] :: real<sup>2</sup>  
**and** ?ct = vector [1/2,1/2] :: real<sup>2</sup>  
**let** ?a = hyp2-abs ?ca  
**and** ?b = hyp2-abs ?cb  
**and** ?c = hyp2-abs ?cc  
**and** ?d = hyp2-abs ?cd  
**and** ?t = hyp2-abs ?ct  
  
**have** ?cd = (1/2) \*<sub>R</sub> ?ct **and** ?cd - ?cb = (1/2) \*<sub>R</sub> (?cc - ?cb)  
**by** (unfold vector-def) (simp-all add: vec-eq-iff)  
**hence** B<sub>R</sub> ?ca ?cd ?ct **and** B<sub>R</sub> ?cb ?cd ?cc  
**by** (unfold real-euclid-B-def) (simp-all add: exI [of - 1/2])  
**hence** B<sub>K</sub> ?a ?d ?t **and** B<sub>K</sub> ?b ?d ?c  
**by** (unfold real-hyp2-B-def) (simp-all add: hyp2-rep-abs-examples)  
  
**have** ?a ≠ ?d  
**proof**  
**assume** ?a = ?d  
**hence** hyp2-rep ?a = hyp2-rep ?d **by** simp  
**hence** ?ca = ?cd **by** (simp add: hyp2-rep-abs-examples)  
**thus** False **by** (simp add: vec-eq-iff forall-2)  
**qed**  
**with** ⟨B<sub>K</sub> ?a ?d ?t⟩ **and** ⟨B<sub>K</sub> ?b ?d ?c⟩  
**and** ⟨∀ a b c d t. B<sub>K</sub> a d t ∧ B<sub>K</sub> b d c ∧ a ≠ d  
→ (∃ x y. B<sub>K</sub> a b x ∧ B<sub>K</sub> a c y ∧ B<sub>K</sub> x t y)⟩  
**obtain** x **and** y **where** B<sub>K</sub> ?a ?b x **and** B<sub>K</sub> ?a ?c y **and** B<sub>K</sub> x ?t y  
**by** blast  
  
**let** ?cx = hyp2-rep x  
**and** ?cy = hyp2-rep y  
**from** ⟨B<sub>K</sub> ?a ?b x⟩ **and** ⟨B<sub>K</sub> ?a ?c y⟩ **and** ⟨B<sub>K</sub> x ?t y⟩  
**have** B<sub>R</sub> ?ca ?cb ?cx **and** B<sub>R</sub> ?ca ?cc ?cy **and** B<sub>R</sub> ?cx ?ct ?cy  
**by** (unfold real-hyp2-B-def) (simp-all add: hyp2-rep-abs-examples)  
  
**from** ⟨B<sub>R</sub> ?ca ?cb ?cx⟩ **and** ⟨B<sub>R</sub> ?ca ?cc ?cy⟩ **and** ⟨B<sub>R</sub> ?cx ?ct ?cy⟩  
**obtain** j **and** k **and** l **where** ?cb - ?ca = j \*<sub>R</sub> (?cx - ?ca)  
**and** ?cc - ?ca = k \*<sub>R</sub> (?cy - ?ca)  
**and** l ≥ 0 **and** l ≤ 1 **and** ?ct - ?cx = l \*<sub>R</sub> (?cy - ?cx)  
**by** (unfold real-euclid-B-def) fast  
  
**from** ⟨?cb - ?ca = j \*<sub>R</sub> (?cx - ?ca)⟩ **and** ⟨?cc - ?ca = k \*<sub>R</sub> (?cy - ?ca)⟩  
**have** j ≠ 0 **and** k ≠ 0 **by** (auto simp add: vec-eq-iff forall-2)  
**with** ⟨?cb - ?ca = j \*<sub>R</sub> (?cx - ?ca)⟩ **and** ⟨?cc - ?ca = k \*<sub>R</sub> (?cy - ?ca)⟩  
**have** ?cx = (1/j) \*<sub>R</sub> ?cb **and** ?cy = (1/k) \*<sub>R</sub> ?cc **by** simp-all

hence  $?cx\$2 = 0$  and  $?cy\$1 = 0$  by *simp-all*

from  $\langle ?ct - ?cx = l *_R (?cy - ?cx) \rangle$   
have  $?ct = (1 - l) *_R ?cx + l *_R ?cy$  by (*simp add: algebra-simps*)  
with  $\langle ?cx\$2 = 0 \rangle$  and  $\langle ?cy\$1 = 0 \rangle$   
have  $?ct\$1 = (1 - l) * (?cx\$1)$  and  $?ct\$2 = l * (?cy\$2)$  by *simp-all*  
hence  $l * (?cy\$2) = 1/2$  and  $(1 - l) * (?cx\$1) = 1/2$  by *simp-all*

have  $?cx\$1 \leq |?cx\$1|$  by *simp*  
also have  $\dots \leq \text{norm } ?cx$  by (*rule component-le-norm-cart*)  
also have  $\dots < 1$  by (*rule norm-hyp2-rep-lt-1*)  
finally have  $?cx\$1 < 1$  .  
with  $\langle l \leq 1 \rangle$  and *mult-less-cancel-left* [of  $1 - l ?cx\$1$  1]  
have  $(1 - l) * ?cx\$1 \leq 1 - l$  by *auto*  
with  $\langle (1 - l) * (?cx\$1) = 1/2 \rangle$  have  $l \leq 1/2$  by *simp*

have  $?cy\$2 \leq |?cy\$2|$  by *simp*  
also have  $\dots \leq \text{norm } ?cy$  by (*rule component-le-norm-cart*)  
also have  $\dots < 1$  by (*rule norm-hyp2-rep-lt-1*)  
finally have  $?cy\$2 < 1$  .  
with  $\langle l \geq 0 \rangle$  and *mult-less-cancel-left* [of  $l ?cy\$2$  1]  
have  $l * ?cy\$2 \leq l$  by *auto*  
with  $\langle l * (?cy\$2) = 1/2 \rangle$  have  $l \geq 1/2$  by *simp*  
with  $\langle l \leq 1/2 \rangle$  have  $l = 1/2$  by *simp*  
with  $\langle l * (?cy\$2) = 1/2 \rangle$  have  $?cy\$2 = 1$  by *simp*  
with  $\langle ?cy\$2 < 1 \rangle$  show *False* by *simp*

qed

**theorem** *hyp2-not-tarski*:  $\neg (\text{tarski real-hyp2-C real-hyp2-B})$   
using *hyp2-axiom10-false*  
by (*unfold tarski-def tarski-space-def tarski-space-axioms-def*) *simp*

Therefore axiom 10 is independent.

end

## References

- [1] K. Borsuk and W. Szmielew. *Foundations of Geometry: Euclidean and Bolyai-Lobachevskian Geometry; Projective Geometry*. North-Holland Publishing Company, 1960. Translated from Polish by Erwin Marquit.
- [2] T. J. M. Makarios. A mechanical verification of the independence of Tarski's Euclidean axiom. Master's thesis, Victoria University of Wellington, New Zealand, 2012. <http://researcharchive.vuw.ac.nz/handle/10063/2315>.
- [3] W. Schwabhäuser, W. Szmielew, and A. Tarski. *Metamathematische Methoden in der Geometrie*. Springer-Verlag, 1983.