

# The Banach–Tarski Paradox in Isabelle/HOL

Arthur F. Ramos      David Barros Hulak  
Ruy J. G. B. de Queiroz

## Abstract

We formalise in Isabelle/HOL the classical Banach–Tarski paradox for the closed unit ball in three-dimensional Euclidean space. The development proves the free-group paradox, the free subgroup of  $SO(3)$ , the Hausdorff paradox away from a countable bad set, the absorption arguments for the sphere and the origin, and the final finite-partition theorem for the ball.

The proof follows the standard route: the paradoxical decomposition of the free group on two generators  $F_2$ , the existence of a free subgroup  $F_2 \leq SO(3)$  via the AFP `Free-Groups` entry, the Hausdorff paradox for  $S^2 \setminus D$  where  $D$  is a countable set of fixed points, and absorption arguments extending the partition to  $S^2$ , then radially to  $B^3 \setminus \{0\}$ , and finally to  $B^3$  itself. The argument crucially relies on the axiom of choice.

## Background

The Banach–Tarski paradox was proved by Banach and Tarski [1], building on Hausdorff’s earlier paradoxical decomposition of the sphere [4]. The route formalised here follows the standard free-group presentation from Wagon’s expositions [7, 6] and can be compared with de Rauglaudre’s Coq formalisation [3]. The Isabelle development uses Breitner’s AFP entry on free groups [2].

## Formalisation Profile

The generated Isabelle session is `Banach_Tarski`. It is based on the HOL Analysis library, imports the AFP entry `Free-Groups`, and uses the `Primes` theory for the arithmetic freeness argument. The development introduces no local axiomatization or oracle. Its non-constructive steps use the standard Hilbert-choice operators of HOL, made explicit in the orbit-representative construction and inverse-map uses.

The checked target is Isabelle 2025-2 with AFP 2026-04-09.

## Contents

1	The unit sphere and unit ball in $\mathbb{R}^3$	3
2	Disjoint finite covers	3
3	Transporting finite partitions	5
4	Group actions on a set	8
5	Paradoxical decomposition	9
6	Equidecomposability	10
7	Free actions	11
8	Choice of orbit representatives	11
9	Transport of paradoxical decompositions along free actions	11
10	Two-element generator type	12
11	The free group $F_2$ and its carrier	13
12	The four “starts-with” classes of $F_2$	14
13	The two distinguished generators $a$ and $b$	14
14	The action of $F_2$ on itself by left multiplication	15
15	Behaviour of left multiplication by $a$	15
16	Image computations	16
17	The paradoxical decomposition of $F_2$	16
18	Rotations of three-space	18
19	Two distinguished rotations	18
20	The rotations as bijections of the sphere	21
21	The map sigma: $F_2$ to $\text{SO}(3)$	22
22	Arithmetic freeness invariant	25
23	The bad set of fixed points	35

<b>24 Free action of <math>F_2</math> on <math>S^2 \setminus D</math></b>	<b>37</b>
<b>25 Transporting the free-group paradox to <math>S^2 \setminus D</math></b>	<b>38</b>
<b>26 The Hausdorff paradox</b>	<b>40</b>

```

theory BT-Prelim
  imports
    HOL-Analysis.Analysis
    Free-Groups.FreeGroups
begin

```

## 1 The unit sphere and unit ball in $\mathbb{R}^3$

We work throughout with vectors in  $(real, 3) \text{ vec}$  (the type  $(real, 3) \text{ vec}$  from *HOL-Analysis.Cartesian-Euclidean-Space*). These abbreviations name the principal geometric objects of interest.

**definition** *sphere2* ::  $(real^3)$  set **where**  
*sphere2* =  $\{x. \text{norm } x = 1\}$

**definition** *ball3* ::  $(real^3)$  set **where**  
*ball3* =  $\{x. \text{norm } x \leq 1\}$

**lemma** *sphere2-subset-ball3*:  $\text{sphere2} \subseteq \text{ball3}$   
*<proof>*

**lemma** *zero-in-ball3*:  $0 \in \text{ball3}$   
*<proof>*

**lemma** *zero-notin-sphere2*:  $(0::real^3) \notin \text{sphere2}$   
*<proof>*

## 2 Disjoint finite covers

A *partition* of a set into a finite list of pairwise disjoint pieces is the basic combinatorial object underlying “equidecomposability”.

**definition** *pairwise-disjoint* :: 'a set list  $\Rightarrow$  bool **where**  
*pairwise-disjoint* *Xs*  $\longleftrightarrow$   
 $(\forall i < \text{length } Xs. \forall j < \text{length } Xs. i \neq j \longrightarrow Xs ! i \cap Xs ! j = \{\})$

**lemma** *pairwise-disjoint-Nil* [*simp*]: *pairwise-disjoint* []  
*<proof>*

**lemma** *pairwise-disjoint-singleton* [*simp*]: *pairwise-disjoint* [*X*]  
*<proof>*

**lemma** *pairwise-disjoint-nthD*:  
**assumes** *pairwise-disjoint Xs*  
**and**  $i < \text{length } Xs$  **and**  $j < \text{length } Xs$  **and**  $i \neq j$   
**shows**  $Xs ! i \cap Xs ! j = \{\}$   
 $\langle \text{proof} \rangle$

**lemma** *indexed-union-set*:  
**fixes**  $P :: 'a \text{ set list}$   
**shows**  $(\bigcup_{i < \text{length } P} P ! i) = \bigcup(\text{set } P)$   
 $\langle \text{proof} \rangle$

**lemma** *indexed-union-append*:  
**fixes**  $P Q :: 'a \text{ set list}$   
**shows**  $(\bigcup_{i < \text{length } (P @ Q)} (P @ Q) ! i) =$   
 $(\bigcup_{i < \text{length } P} P ! i) \cup (\bigcup_{i < \text{length } Q} Q ! i)$   
 $\langle \text{proof} \rangle$

**lemma** *indexed-image-union-zip*:  
**fixes**  $P :: 'a \text{ set list}$  **and**  $G :: ('a \Rightarrow 'b) \text{ list}$   
**assumes**  $\text{length } P = \text{length } G$   
**shows**  $(\bigcup_{i < \text{length } P} G ! i \ ' (P ! i)) =$   
 $(\bigcup_{(g, A) \in \text{set } (\text{zip } G P)} g \ ' A)$   
 $\langle \text{proof} \rangle$

**lemma** *indexed-image-union-append*:  
**fixes**  $P Q :: 'a \text{ set list}$  **and**  $G H :: ('a \Rightarrow 'b) \text{ list}$   
**assumes**  $\text{len } P: \text{length } P = \text{length } G$  **and**  $\text{len } Q: \text{length } Q = \text{length } H$   
**shows**  $(\bigcup_{i < \text{length } (P @ Q)} (G @ H) ! i \ ' ((P @ Q) ! i)) =$   
 $(\bigcup_{i < \text{length } P} G ! i \ ' (P ! i)) \cup$   
 $(\bigcup_{i < \text{length } Q} H ! i \ ' (Q ! i))$   
 $\langle \text{proof} \rangle$

**lemma** *indexed-union-map-fst*:  
**fixes**  $xs :: ('a \text{ set} \times 'b) \text{ list}$   
**shows**  $(\bigcup_{i < \text{length } (\text{map } \text{fst } xs)} \text{map } \text{fst } xs ! i) =$   
 $(\bigcup_{x \in \text{set } xs} \text{fst } x)$   
 $\langle \text{proof} \rangle$

**lemma** *indexed-image-union-pairs*:  
**fixes**  $xs :: ('a \text{ set} \times ('a \Rightarrow 'b)) \text{ list}$   
**shows**  $(\bigcup_{i < \text{length } (\text{map } \text{fst } xs)} \text{map } \text{snd } xs ! i \ ' (\text{map } \text{fst } xs ! i)) =$   
 $(\bigcup_{x \in \text{set } xs} \text{snd } x \ ' \text{fst } x)$   
 $\langle \text{proof} \rangle$

**lemma** *pairwise-disjoint-map-fstI*:  
**fixes**  $xs :: ('a \text{ set} \times 'b) \text{ list}$   
**assumes** *distinct: distinct xs*  
**and**  $\text{disj}: \bigwedge x y. \llbracket x \in \text{set } xs; y \in \text{set } xs; x \neq y \rrbracket$

$\implies \text{fst } x \cap \text{fst } y = \{\}$   
**shows** *pairwise-disjoint* (*map fst xs*)  
 <proof>

**lemma** *pairwise-disjoint-appendI*:  
**assumes** *xs: pairwise-disjoint xs*  
**and** *ys: pairwise-disjoint ys*  
**and** *cross:  $\bigwedge x y. \llbracket x \in \text{set } xs; y \in \text{set } ys \rrbracket \implies x \cap y = \{\}$*   
**shows** *pairwise-disjoint* (*xs @ ys*)  
 <proof>

**lemma** *pairwise-disjoint-appendD1*:  
**assumes** *pairwise-disjoint* (*xs @ ys*)  
**shows** *pairwise-disjoint xs*  
 <proof>

**lemma** *pairwise-disjoint-appendD2*:  
**assumes** *pairwise-disjoint* (*xs @ ys*)  
**shows** *pairwise-disjoint ys*  
 <proof>

**lemma** *pairwise-disjoint-append-crossD*:  
**assumes** *pairwise-disjoint* (*xs @ ys*)  
**and** *i < length xs and j < length ys*  
**shows** *xs ! i  $\cap$  ys ! j =  $\{\}$*   
 <proof>

**lemma** *pairwise-disjoint-pair-imagesI*:  
**fixes** *xs :: ('a set  $\times$  ('a  $\Rightarrow$  'b)) list*  
**assumes** *distinct: distinct xs*  
**and** *disj:  $\bigwedge x y. \llbracket x \in \text{set } xs; y \in \text{set } xs; x \neq y \rrbracket$*   
 $\implies \text{snd } x \text{ 'fst } x \cap \text{snd } y \text{ 'fst } y = \{\}$   
**shows** *pairwise-disjoint* (*map2* ( $\lambda g A. g \text{ ' } A$ ) (*map snd xs*) (*map fst xs*))  
 <proof>

### 3 Transporting finite partitions

**definition** *transfer-piece* ::  
*'a set list  $\Rightarrow$  ('a  $\Rightarrow$  'a) list  $\Rightarrow$  'a set list  $\Rightarrow$*   
*('a  $\Rightarrow$  'a) list  $\Rightarrow$  'a set list  $\Rightarrow$  nat  $\Rightarrow$  nat  $\Rightarrow$  nat  $\Rightarrow$  'a set*  
**where**  
*transfer-piece A e P g B k i l =*  
 $\{x \in A ! k. (e ! k) x \in P ! i \wedge (g ! i) ((e ! k) x) \in B ! l\}$

**definition** *transfer-map* ::  
*('a  $\Rightarrow$  'a) list  $\Rightarrow$  ('a  $\Rightarrow$  'a) list  $\Rightarrow$*   
*('a  $\Rightarrow$  'a) list  $\Rightarrow$  nat  $\Rightarrow$  nat  $\Rightarrow$  nat  $\Rightarrow$  ('a  $\Rightarrow$  'a)*  
**where**  
*transfer-map t g e k i l = (t ! l)  $\circ$  (g ! i)  $\circ$  (e ! k)*

**definition** *transfer-pairs* ::

'a set list  $\Rightarrow$  ('a  $\Rightarrow$  'a) list  $\Rightarrow$  'a set list  $\Rightarrow$   
('a  $\Rightarrow$  'a) list  $\Rightarrow$  'a set list  $\Rightarrow$  ('a  $\Rightarrow$  'a) list  $\Rightarrow$   
('a set  $\times$  ('a  $\Rightarrow$  'a)) set

**where**

*transfer-pairs* A e P g B t =  
{(transfer-piece A e P g B k i l, transfer-map t g e k i l) |  
k i l. k < length A  $\wedge$  i < length P  $\wedge$  l < length B}

**lemma** *finite-transfer-pairs [simp]*: finite (transfer-pairs A e P g B t)  
<proof>

**lemma** *transfer-source-cover*:

**assumes** *A-cover*:  $Y = (\bigcup_{k < \text{length } A} A ! k)$   
**and** *B-cover*:  $X = (\bigcup_{k < \text{length } B} B ! k)$   
**and** *e-into-B*:  $\bigwedge k x. \llbracket k < \text{length } A; x \in A ! k \rrbracket \Longrightarrow (e ! k) x \in B ! k$   
**and** *len-AB*:  $\text{length } A = \text{length } B$   
**and** *source-cover*:  $X = (\bigcup_{i < \text{length } P} P ! i) \cup (\bigcup_{j < \text{length } Q} Q ! j)$   
**and** *imageP-cover*:  $X = (\bigcup_{i < \text{length } P} gP ! i \text{ ' } (P ! i))$   
**and** *imageQ-cover*:  $X = (\bigcup_{j < \text{length } Q} gQ ! j \text{ ' } (Q ! j))$

**shows**  $Y =$

$(\bigcup_{p \in \text{transfer-pairs } A e P gP B t. \text{fst } p}) \cup$   
 $(\bigcup_{q \in \text{transfer-pairs } A e Q gQ B t. \text{fst } q})$

<proof>

**lemma** *transfer-image-cover*:

**assumes** *A-cover*:  $Y = (\bigcup_{k < \text{length } A} A ! k)$   
**and** *B-cover*:  $X = (\bigcup_{k < \text{length } B} B ! k)$   
**and** *len-AB*:  $\text{length } A = \text{length } B$   
**and** *e-left*:  $\bigwedge k x. \llbracket k < \text{length } A; x \in A ! k \rrbracket$   
 $\Longrightarrow (e ! k) x \in B ! k \wedge (t ! k) ((e ! k) x) = x$   
**and** *t-left*:  $\bigwedge k y. \llbracket k < \text{length } B; y \in B ! k \rrbracket$   
 $\Longrightarrow (t ! k) y \in A ! k \wedge (e ! k) ((t ! k) y) = y$   
**and** *P-sub*:  $\bigwedge i. i < \text{length } P \Longrightarrow P ! i \subseteq X$   
**and** *image-cover*:  $X = (\bigcup_{i < \text{length } P} g ! i \text{ ' } (P ! i))$

**shows**  $Y = (\bigcup_{p \in \text{transfer-pairs } A e P g B t. \text{snd } p \text{ ' } \text{fst } p})$

<proof>

**lemma** *transfer-source-disjoint-same*:

**assumes** *A-disj*: pairwise-disjoint A  
**and** *P-disj*: pairwise-disjoint P  
**and** *B-disj*: pairwise-disjoint B  
**and** *x-in*:  $x \in \text{transfer-pairs } A e P g B t$   
**and** *y-in*:  $y \in \text{transfer-pairs } A e P g B t$   
**and** *xy*:  $x \neq y$

**shows**  $\text{fst } x \cap \text{fst } y = \{\}$

<proof>

**lemma** *transfer-source-disjoint-cross*:  
**assumes** *A-disj*: pairwise-disjoint *A*  
**and** *PQ-cross*:  $\bigwedge i j. \llbracket i < \text{length } P; j < \text{length } Q \rrbracket \implies P ! i \cap Q ! j = \{\}$   
**and** *B-disj*: pairwise-disjoint *B*  
**and** *x-in*:  $x \in \text{transfer-pairs } A \ e \ P \ gP \ B \ t$   
**and** *y-in*:  $y \in \text{transfer-pairs } A \ e \ Q \ gQ \ B \ t$   
**shows**  $\text{fst } x \cap \text{fst } y = \{\}$   
*<proof>*

**lemma** *transfer-image-disjoint-same*:  
**assumes** *len-AB*:  $\text{length } A = \text{length } B$   
**and** *A-disj*: pairwise-disjoint *A*  
**and** *B-disj*: pairwise-disjoint *B*  
**and** *image-disj*: pairwise-disjoint  $(\text{map2 } (\lambda h \ C. \ h \ ' \ C) \ g \ P)$   
**and** *len*:  $\text{length } P = \text{length } g$   
**and** *g-inj*:  $\bigwedge i. i < \text{length } P \implies \text{inj } (g ! i)$   
**and** *e-into-B*:  $\bigwedge k \ x. \llbracket k < \text{length } A; x \in A ! k \rrbracket \implies (e ! k) \ x \in B ! k$   
**and** *t-left*:  $\bigwedge l \ y. \llbracket l < \text{length } B; y \in B ! l \rrbracket$   
 $\implies (t ! l) \ y \in A ! l \wedge (e ! l) \ ((t ! l) \ y) = y$   
**and** *x-in*:  $x \in \text{transfer-pairs } A \ e \ P \ g \ B \ t$   
**and** *y-in*:  $y \in \text{transfer-pairs } A \ e \ P \ g \ B \ t$   
**and** *xy*:  $x \neq y$   
**shows**  $\text{snd } x \ ' \ \text{fst } x \cap \text{snd } y \ ' \ \text{fst } y = \{\}$   
*<proof>*

**lemma** *transfer-partitioned-paradox*:  
**assumes** *len-AB*:  $\text{length } A = \text{length } B$   
**and** *A-cover*:  $Y = (\bigcup k < \text{length } A. A ! k)$   
**and** *A-disj*: pairwise-disjoint *A*  
**and** *B-cover*:  $X = (\bigcup k < \text{length } B. B ! k)$   
**and** *B-disj*: pairwise-disjoint *B*  
**and** *e-left*:  $\bigwedge k \ x. \llbracket k < \text{length } A; x \in A ! k \rrbracket$   
 $\implies (e ! k) \ x \in B ! k \wedge (t ! k) \ ((e ! k) \ x) = x$   
**and** *t-left*:  $\bigwedge k \ y. \llbracket k < \text{length } B; y \in B ! k \rrbracket$   
 $\implies (t ! k) \ y \in A ! k \wedge (e ! k) \ ((t ! k) \ y) = y$   
**and** *lenP*:  $\text{length } P = \text{length } gP$   
**and** *lenQ*:  $\text{length } Q = \text{length } gQ$   
**and** *source-disj*: pairwise-disjoint  $(P \ @ \ Q)$   
**and** *source-cover*:  $X = (\bigcup i < \text{length } P. P ! i) \cup (\bigcup j < \text{length } Q. Q ! j)$   
**and** *imageP-disj*: pairwise-disjoint  $(\text{map2 } (\lambda h \ C. \ h \ ' \ C) \ gP \ P)$   
**and** *imageQ-disj*: pairwise-disjoint  $(\text{map2 } (\lambda h \ C. \ h \ ' \ C) \ gQ \ Q)$   
**and** *imageP-cover*:  $X = (\bigcup i < \text{length } P. gP ! i \ ' \ (P ! i))$   
**and** *imageQ-cover*:  $X = (\bigcup j < \text{length } Q. gQ ! j \ ' \ (Q ! j))$   
**and** *gP-inj*:  $\bigwedge i. i < \text{length } P \implies \text{inj } (gP ! i)$   
**and** *gQ-inj*:  $\bigwedge j. j < \text{length } Q \implies \text{inj } (gQ ! j)$   
**and** *mapsP*:  $\bigwedge k \ i \ l. \llbracket k < \text{length } A; i < \text{length } P; l < \text{length } B \rrbracket$   
 $\implies \text{transfer-map } t \ gP \ e \ k \ i \ l \in M$   
**and** *mapsQ*:  $\bigwedge k \ j \ l. \llbracket k < \text{length } A; j < \text{length } Q; l < \text{length } B \rrbracket$   
 $\implies \text{transfer-map } t \ gQ \ e \ k \ j \ l \in M$

```

shows  $\exists P' Q' :: 'a \text{ set list. } \exists gP' gQ' :: ('a \Rightarrow 'a) \text{ list.}$ 
   $\text{length } P' = \text{length } gP' \wedge \text{length } Q' = \text{length } gQ' \wedge$ 
   $\text{set } gP' \subseteq M \wedge \text{set } gQ' \subseteq M \wedge$ 
   $\text{pairwise-disjoint } (P' @ Q') \wedge$ 
   $\text{pairwise-disjoint } (\text{map2 } (\lambda h C. h ' C) gP' P') \wedge$ 
   $\text{pairwise-disjoint } (\text{map2 } (\lambda h C. h ' C) gQ' Q') \wedge$ 
   $Y = (\bigcup_{i < \text{length } P'. P' ! i} \cup (\bigcup_{j < \text{length } Q'. Q' ! j}) \wedge$ 
   $Y = (\bigcup_{i < \text{length } P'. gP' ! i ' (P' ! i)) \wedge$ 
   $Y = (\bigcup_{j < \text{length } Q'. gQ' ! j ' (Q' ! j))$ 
<proof>

```

**end**

```

theory Paradoxical-Decomposition
  imports BT-Prelim
begin

```

## 4 Group actions on a set

An action of a (carrier) group  $G$  on a set  $X$  is a function  $act$  respecting the unit and multiplication on  $X$ . We do not fix the group structure here; rather, we record the algebraic laws that an action must satisfy. Concrete instances will be given in later theories (the action of  $F_2$  on itself by left translation; the action of  $SO(3)$  on  $S^2$ ).

```

locale group-action =
  fixes carrier :: 'g set
    and unit :: 'g
    and mult :: 'g  $\Rightarrow$  'g  $\Rightarrow$  'g (infixl  $\langle \cdot \rangle$  70)
    and act :: 'g  $\Rightarrow$  'a  $\Rightarrow$  'a
    and ground :: 'a set
  assumes unit-carrier [simp]:  $\text{unit} \in \text{carrier}$ 
    and mult-closed [intro]:  $\llbracket g \in \text{carrier}; h \in \text{carrier} \rrbracket \Longrightarrow g \cdot h \in \text{carrier}$ 
    and act-closed [intro]:  $\llbracket g \in \text{carrier}; x \in \text{ground} \rrbracket \Longrightarrow \text{act } g \ x \in \text{ground}$ 
    and act-unit [simp]:  $x \in \text{ground} \Longrightarrow \text{act } \text{unit} \ x = x$ 
    and act-mult:  $\llbracket g \in \text{carrier}; h \in \text{carrier}; x \in \text{ground} \rrbracket$ 
       $\Longrightarrow \text{act } (g \cdot h) \ x = \text{act } g \ (\text{act } h \ x)$ 

```

```

context group-action
begin

```

```

definition orbit :: 'a  $\Rightarrow$  'a set where
  orbit  $x = \{y. \exists g \in \text{carrier. } y = \text{act } g \ x\}$ 

```

```

definition image-set :: 'g  $\Rightarrow$  'a set  $\Rightarrow$  'a set where
  image-set  $g \ A = \text{act } g \ ' A$ 

```

```

lemma image-set-unit:

```

**assumes**  $A \subseteq \text{ground}$   
**shows** *image-set unit*  $A = A$   
 ⟨*proof*⟩

**lemma** *orbit-self*:  
**assumes**  $x \in \text{ground}$   
**shows**  $x \in \text{orbit } x$   
 ⟨*proof*⟩

**end**

## 5 Paradoxical decomposition

A subset  $X$  of the underlying set of a group action is *paradoxical* when it admits two finite collections of pairwise disjoint pieces, all contained in  $X$ , such that each collection can be translated (by group elements) so that the resulting *disjoint* union is the whole of  $X$ .

This is the standard Banach-Tarski definition. We will instantiate it twice: once for  $F_2$  acting on itself, and once for the rotation group acting on  $S^2$  (and ultimately the ball).

**context** *group-action*  
**begin**

**definition** *paradoxical* :: 'a set  $\Rightarrow$  bool **where**  
*paradoxical*  $X \longleftrightarrow$   
 ( $\exists P Q ::$  'a set list.  $\exists gP gQ ::$  'g list.  
 $\text{length } P = \text{length } gP \wedge \text{length } Q = \text{length } gQ \wedge$   
 $\text{set } gP \subseteq \text{carrier} \wedge \text{set } gQ \subseteq \text{carrier} \wedge$   
 $\text{pairwise-disjoint } (P @ Q) \wedge$   
 $\text{pairwise-disjoint } (\text{map2 } \text{image-set } gP P) \wedge$   
 $\text{pairwise-disjoint } (\text{map2 } \text{image-set } gQ Q) \wedge$   
 $(\bigcup_{i < \text{length } P}. P ! i) \cup (\bigcup_{i < \text{length } Q}. Q ! i) \subseteq X \wedge$   
 $X = (\bigcup_{i < \text{length } P}. \text{image-set } (gP ! i) (P ! i)) \wedge$   
 $X = (\bigcup_{i < \text{length } Q}. \text{image-set } (gQ ! i) (Q ! i))$ )

A specialised constructor for the most common case: each of  $P$  and  $Q$  has exactly two pieces. This matches the  $F_2$  decomposition (split into the  $a / a^{-1}$  classes and the  $b / b^{-1}$  classes).

**lemma** *paradoxical-two-two*:  
**assumes**  $p1 \in \text{carrier } p2 \in \text{carrier } q1 \in \text{carrier } q2 \in \text{carrier}$   
**and** *disj-P12*:  $P1 \cap P2 = \{\}$   
**and** *disj-Q12*:  $Q1 \cap Q2 = \{\}$   
**and** *disj-PQ*:  $(P1 \cup P2) \cap (Q1 \cup Q2) = \{\}$   
**and** *sub*:  $P1 \cup P2 \cup Q1 \cup Q2 \subseteq X$   
**and** *disj-imgP*:  $\text{image-set } p1 P1 \cap \text{image-set } p2 P2 = \{\}$   
**and** *disj-imgQ*:  $\text{image-set } q1 Q1 \cap \text{image-set } q2 Q2 = \{\}$   
**and** *cover-P*:  $X = \text{image-set } p1 P1 \cup \text{image-set } p2 P2$

**and** *cover-Q*:  $X = \text{image-set } q1 \ Q1 \cup \text{image-set } q2 \ Q2$   
**shows** *paradoxical X*  
 ⟨*proof*⟩

With the finite-list convention used above, the empty set is paradoxical: take both families of pieces to be empty. The two translated indexed unions are then both the empty union, hence both equal to  $\{\}$ . This degenerate case is harmless in the later non-empty geometric applications, but recording it keeps the abstract definition honest about empty indexed unions.

**lemma** *paradoxical-empty: paradoxical  $\{\}$*   
 ⟨*proof*⟩

**end**

## 6 Equidecomposability

Two subsets  $X$  and  $Y$  are *equidecomposable* under a group action when they admit congruent partitions: the  $i$ -th piece of  $X$  maps onto the  $i$ -th piece of  $Y$  via some group element.

**context** *group-action*  
**begin**

**definition** *equidecomposable* :: '*a set*  $\Rightarrow$  '*a set*  $\Rightarrow$  *bool* **where**  
*equidecomposable X Y*  $\longleftrightarrow$   
 ( $\exists P \ Q ::$  '*a set list*.  $\exists gs ::$  '*g list*.  
 $\text{length } P = \text{length } Q \wedge \text{length } P = \text{length } gs \wedge$   
 $\text{set } gs \subseteq \text{carrier} \wedge$   
 $\text{pairwise-disjoint } P \wedge \text{pairwise-disjoint } Q \wedge$   
 $X = (\bigcup_{i < \text{length } P}. P ! i) \wedge$   
 $Y = (\bigcup_{i < \text{length } Q}. Q ! i) \wedge$   
 $(\forall i < \text{length } P. Q ! i = \text{image-set } (gs ! i) (P ! i))$ )

The main development uses this relation only as background notation: the sphere and ball arguments below work directly with explicit finite lists of pieces and transformations, so the needed partition data is visible at each transfer step.

**end**

**end**

**theory** *Free-Action-Paradox*  
**imports** *Paradoxical-Decomposition*  
**begin**

## 7 Free actions

An action is *free* on a set  $X$  if non-identity group elements have no fixed points in  $X$ :  $g \cdot x = x$  forces  $g = \text{unit}$ .

**context** *group-action*  
**begin**

**definition** *free-on* :: 'a set  $\Rightarrow$  bool **where**  
*free-on*  $X \longleftrightarrow$   
 $(\forall g \in \text{carrier}. \forall x \in X. \text{act } g \ x = x \longrightarrow g = \text{unit})$

**end**

## 8 Choice of orbit representatives

Given a free action of  $G$  on  $X$ , the orbits partition  $X$ , and for any subset of choice representatives  $M \subseteq X$  (one per orbit), every  $x \in X$  can be written uniquely as  $\text{act } g \ m$  for some  $g \in \text{carrier}$  and  $m \in M$ .

This is where the axiom of choice enters: we pick one element from each orbit.

**context** *group-action*  
**begin**

**definition** *orbit-eq* :: 'a  $\Rightarrow$  'a  $\Rightarrow$  bool **where**  
*orbit-eq*  $x \ y \longleftrightarrow (\exists g \in \text{carrier}. y = \text{act } g \ x)$

**lemma** *orbit-eq-refl*:  $x \in \text{ground} \Longrightarrow \text{orbit-eq } x \ x$   
*<proof>*

When the action is free on  $X$ , the orbit relation is symmetric on  $X$ , provided every element has a group inverse in the carrier. We do not require *group*-level inverses here, so we state symmetry conditionally.

**end**

## 9 Transport of paradoxical decompositions along free actions

This is the abstract Banach-Tarski reduction: suppose  $G$  acts freely on  $X$ , and  $G$  itself admits a paradoxical decomposition under the regular action (left multiplication on itself). Choosing one representative per  $G$ -orbit in  $X$  via the axiom of choice transports the paradox: each piece  $P_i \subseteq G$  in the decomposition lifts to the union  $\{\text{act } g \ m \mid g \in P_i, m \in M\}$ , where  $M$  is the chosen set of representatives.

The later Hausdorff theory carries out this lift explicitly for  $S^2 \setminus D$ : it defines orbit representatives, proves the lifted pieces are disjoint, and proves the two translated covers directly. The predicate below only packages the freeness and closure hypotheses that such an argument needs.

**context** *group-action*  
**begin**

A free action is suitable for orbit-by-orbit transport when the set is contained in the ground space, the action is free on it, and the set is closed under the action of every carrier element.

**definition** *transports-paradox* :: 'a set  $\Rightarrow$  bool **where**  
*transports-paradox*  $X \longleftrightarrow$   
 $(X \subseteq \text{ground} \wedge$   
 $\text{free-on } X \wedge$   
 $(\forall x \in X. \forall g \in \text{carrier}. \text{act } g \ x \in X))$

Under the assumptions captured by *transports-paradox*, a paradoxical decomposition of the carrier transports to a paradoxical decomposition of  $X$ . The proof is the standard “orbit-by-orbit” argument and uses the axiom of choice to select one representative per orbit.

**end**

**end**

**theory** *Free-Group-F2*  
**imports**  
*BT-Prelim*  
*Free-Groups.FreeGroups*  
**begin**

## 10 Two-element generator type

We pick a concrete two-element type for the generators of  $F_2$ .

**datatype** *gen2* =  $A \mid B$

**abbreviation** *Gen2* :: *gen2 set* **where**  
 $Gen2 \equiv \{A, B\}$

**lemma** *Gen2-finite* [*simp*]: *finite Gen2*  
 $\langle \text{proof} \rangle$

**lemma** *Gen2-card* [*simp*]:  $\text{card } Gen2 = 2$   
 $\langle \text{proof} \rangle$

**lemma** *Gen2-UNIV*:  $Gen2 = (UNIV :: \text{gen2 set})$   
 $\langle \text{proof} \rangle$

## 11 The free group $F_2$ and its carrier

**abbreviation**  $F2$  :: (*bool* × *gen2*) list monoid **where**

$$F2 \equiv \mathcal{F} Gen2$$

The carrier of  $F_2$  consists of cancelled words over the alphabet  $UNIV \times Gen2 = UNIV \times UNIV$ .

**lemma** *F2-is-group*: group  $F2$

⟨*proof*⟩

**lemma** *F2-carrier-iff*:

$$w \in \text{carrier } F2 \iff w \in \text{lists } (UNIV \times Gen2) \wedge \text{canceled } w$$

⟨*proof*⟩

**lemma** *F2-carrier-alt*:

$$\text{carrier } F2 = \{w. \text{canceled } w\}$$

⟨*proof*⟩

**lemma** *in-F2-canceled* [*dest*]:

**assumes**  $w \in \text{carrier } F2$

**shows** *canceled*  $w$

⟨*proof*⟩

**lemma** *canceled-in-F2* [*intro*]:

**assumes** *canceled*  $w$

**shows**  $w \in \text{carrier } F2$

⟨*proof*⟩

**lemma** *canceled-iff-no-adjacent-canceling*:

$$\text{canceled } w \iff (\forall i. \text{Suc } i < \text{length } w \longrightarrow \neg \text{canceling } (w ! i) (w ! \text{Suc } i))$$

⟨*proof*⟩

**lemma** *canceled-Cons-iff*:

$$\text{canceled } (p \# w) \iff \text{canceled } w \wedge (w = [] \vee \neg \text{canceling } p (\text{hd } w))$$

⟨*proof*⟩

**lemma** *F2-ConsD*:

**assumes**  $p \# w \in \text{carrier } F2$

**shows**  $w \in \text{carrier } F2$  **and**  $w = [] \vee \neg \text{canceling } p (\text{hd } w)$

⟨*proof*⟩

**lemma** *F2-ConsI*:

**assumes**  $w \in \text{carrier } F2$  **and**  $w = [] \vee \neg \text{canceling } p (\text{hd } w)$

**shows**  $p \# w \in \text{carrier } F2$

⟨*proof*⟩

**lemma** *F2-one* [*simp*]:  $\mathbf{1}_{F2} = []$

⟨*proof*⟩

**lemma** *F2-mult*:  $x \otimes_{F_2} y = \text{normalize } (x @ y)$   
 ⟨proof⟩

## 12 The four “starts-with” classes of $F_2$

A non-empty reduced word in  $F_2$  begins with one of four letters:  $a$ ,  $a^{-1}$ ,  $b$ , or  $b^{-1}$ . The corresponding “starts-with” classes partition  $F_2 \setminus \{1\}$ .

**definition** *starts-with* ::  $\text{bool} \Rightarrow \text{gen2} \Rightarrow (\text{bool} \times \text{gen2})$  list set **where**  
*starts-with*  $b$   $x = \{w \in \text{carrier } F_2. w \neq [] \wedge \text{hd } w = (b, x)\}$

**abbreviation** *S-a* ::  $(\text{bool} \times \text{gen2})$  list set **where**  $S-a \equiv \text{starts-with } \text{False } A$

**abbreviation** *S-aI* ::  $(\text{bool} \times \text{gen2})$  list set **where**  $S-aI \equiv \text{starts-with } \text{True } A$

**abbreviation** *S-b* ::  $(\text{bool} \times \text{gen2})$  list set **where**  $S-b \equiv \text{starts-with } \text{False } B$

**abbreviation** *S-bI* ::  $(\text{bool} \times \text{gen2})$  list set **where**  $S-bI \equiv \text{starts-with } \text{True } B$

**lemma** *starts-with-subset*:  $\text{starts-with } b$   $x \subseteq \text{carrier } F_2$   
 ⟨proof⟩

**lemma** *starts-with-disjoint*:  
**assumes**  $(b1, x1) \neq (b2, x2)$   
**shows**  $\text{starts-with } b1$   $x1 \cap \text{starts-with } b2$   $x2 = \{\}$   
 ⟨proof⟩

**lemma** *starts-with-pairwise-disjoint*:  
**shows**  $S-a \cap S-aI = \{\}$   $S-a \cap S-b = \{\}$   $S-a \cap S-bI = \{\}$   
 $S-aI \cap S-b = \{\}$   $S-aI \cap S-bI = \{\}$   $S-b \cap S-bI = \{\}$   
 $[] \notin S-a$   $[] \notin S-aI$   $[] \notin S-b$   $[] \notin S-bI$   
 ⟨proof⟩

The four classes together with the singleton  $\{1\}$  partition the carrier of  $F_2$ . The proof is a case analysis on the head of a reduced word.

**lemma** *F2-decomposition*:  
 $\text{carrier } F_2 = \{\{\}\} \cup S-a \cup S-aI \cup S-b \cup S-bI$   
 ⟨proof⟩

## 13 The two distinguished generators $a$ and $b$

**abbreviation** *a-elt* ::  $(\text{bool} \times \text{gen2})$  list **where**  $a\text{-elt} \equiv [(False, A)]$

**abbreviation** *aI-elt* ::  $(\text{bool} \times \text{gen2})$  list **where**  $aI\text{-elt} \equiv [(True, A)]$

**abbreviation** *b-elt* ::  $(\text{bool} \times \text{gen2})$  list **where**  $b\text{-elt} \equiv [(False, B)]$

**abbreviation** *bI-elt* ::  $(\text{bool} \times \text{gen2})$  list **where**  $bI\text{-elt} \equiv [(True, B)]$

**lemma** *single-letter-canceled*:  
*canceled*  $[(b, x)]$   
 ⟨proof⟩

**lemma** *a-elt-in-F2* [*simp*]:  $a\text{-elt} \in \text{carrier } F_2$

*<proof>*

**lemma** *aI-elt-in-F2* [*simp*]: *aI-elt*  $\in$  *carrier F2*  
*<proof>*

**lemma** *b-elt-in-F2* [*simp*]: *b-elt*  $\in$  *carrier F2*  
*<proof>*

**lemma** *bI-elt-in-F2* [*simp*]: *bI-elt*  $\in$  *carrier F2*  
*<proof>*

**lemma** *inv-a-eq-aI*: *inv<sub>F2</sub> a-elt* = *aI-elt*  
*<proof>*

**lemma** *inv-b-eq-bI*: *inv<sub>F2</sub> b-elt* = *bI-elt*  
*<proof>*

**lemma** *inv-aI-eq-a*: *inv<sub>F2</sub> aI-elt* = *a-elt*  
*<proof>*

**lemma** *inv-bI-eq-b*: *inv<sub>F2</sub> bI-elt* = *b-elt*  
*<proof>*

**end**

**theory** *F2-Paradox*  
**imports**  
    *Free-Group-F2*  
    *Paradoxical-Decomposition*  
**begin**

## 14 The action of $F_2$ on itself by left multiplication

We interpret *group-action* with carrier *carrier F2* and the action being left multiplication.

**lemma** *F2-one-in-carrier* [*simp*]:  $\square \in$  *carrier F2*  
*<proof>*

**interpretation** *F2-act*:  
    *group-action carrier F2*  $\square$  ( $\otimes_{F2}$ ) ( $\otimes_{F2}$ ) *carrier F2*  
*<proof>*

## 15 Behaviour of left multiplication by $a$

When we left-multiply a word that begins with  $a^{-1}$  by  $a$ , the leading pair cancels, and the result is exactly the tail of the original word.

**lemma** *a-mult-starts-with-aI*:

**assumes**  $w \in S\text{-}aI$

**shows**  $a\text{-elt} \otimes_{F_2} w = tl\ w$

*<proof>*

**lemma** *b-mult-starts-with-bI*:

**assumes**  $w \in S\text{-}bI$

**shows**  $b\text{-elt} \otimes_{F_2} w = tl\ w$

*<proof>*

## 16 Image computations

Translating  $S_a^{-1}$  on the left by  $a$  yields exactly the carrier of  $F_2$  minus  $S_a$ .

**lemma** *image-a-S-aI*:

**shows**  $(\otimes_{F_2})\ a\text{-elt} \text{ ' } S\text{-}aI = \text{carrier } F_2 - S\text{-}a$

*<proof>*

**lemma** *image-b-S-bI*:

**shows**  $(\otimes_{F_2})\ b\text{-elt} \text{ ' } S\text{-}bI = \text{carrier } F_2 - S\text{-}b$

*<proof>*

## 17 The paradoxical decomposition of $F_2$

Putting everything together:  $S_a$  together with the  $a$ -translate of  $S_a^{-1}$  covers  $F_2$  (disjointly), and symmetrically for  $b$ . Hence  $F_2$  is paradoxical.

**lemma** *S-a-un-image-a-S-aI*:  $S\text{-}a \cup (\otimes_{F_2})\ a\text{-elt} \text{ ' } S\text{-}aI = \text{carrier } F_2$

*<proof>*

**lemma** *S-b-un-image-b-S-bI*:  $S\text{-}b \cup (\otimes_{F_2})\ b\text{-elt} \text{ ' } S\text{-}bI = \text{carrier } F_2$

*<proof>*

**lemma** *S-a-disj-image-a-S-aI*:  $S\text{-}a \cap (\otimes_{F_2})\ a\text{-elt} \text{ ' } S\text{-}aI = \{\}$

*<proof>*

**lemma** *S-b-disj-image-b-S-bI*:  $S\text{-}b \cap (\otimes_{F_2})\ b\text{-elt} \text{ ' } S\text{-}bI = \{\}$

*<proof>*

**definition** *aI-ray* ::  $(\text{bool} \times \text{gen2})$  list set **where**

$aI\text{-ray} = \{\text{replicate } n\ (\text{True}, A) \mid n. \text{True}\}$

**lemma** *canceled-replicate-aI*:  $\text{canceled} (\text{replicate } n\ (\text{True}, A))$

*<proof>*

**lemma** *aI-ray-subset-carrier*:  $aI\text{-ray} \subseteq \text{carrier } F_2$

*<proof>*

**lemma** *aI-ray-cases*:

**assumes**  $w \in aI\text{-ray}$   
**shows**  $w = [] \vee w \in S\text{-}aI$   
 $\langle \text{proof} \rangle$

**lemma** *a-mult-aI-ray-pos*:  
 $(\otimes_{F2}) \text{ a-elt } ' (aI\text{-ray} - \{[]\}) = aI\text{-ray}$   
 $\langle \text{proof} \rangle$

**lemma** *a-mult-nonray-stays-nonray*:  
**assumes**  $w \in S\text{-}aI$  **and**  $w \notin aI\text{-ray}$   
**shows**  $\text{a-elt } \otimes_{F2} w \notin aI\text{-ray}$   
 $\langle \text{proof} \rangle$

**lemma** *image-a-S-aI-nonray*:  
 $(\otimes_{F2}) \text{ a-elt } ' (S\text{-}aI - aI\text{-ray}) = \text{carrier } F2 - S\text{-}a - aI\text{-ray}$   
 $\langle \text{proof} \rangle$

**theorem** *F2-paradoxical-partition*:  
 $\exists P Q :: ((\text{bool} \times \text{gen2}) \text{ list}) \text{ set list. } \exists gP gQ :: ((\text{bool} \times \text{gen2}) \text{ list}) \text{ list.}$   
 $\text{length } P = \text{length } gP \wedge \text{length } Q = \text{length } gQ \wedge$   
 $\text{set } gP \subseteq \text{carrier } F2 \wedge \text{set } gQ \subseteq \text{carrier } F2 \wedge$   
 $\text{pairwise-disjoint } (P @ Q) \wedge$   
 $\text{pairwise-disjoint } (\text{map2 } F2\text{-act.image-set } gP P) \wedge$   
 $\text{pairwise-disjoint } (\text{map2 } F2\text{-act.image-set } gQ Q) \wedge$   
 $\text{carrier } F2 = (\bigcup i < \text{length } P. P ! i) \cup (\bigcup i < \text{length } Q. Q ! i) \wedge$   
 $\text{carrier } F2 = (\bigcup i < \text{length } P. F2\text{-act.image-set } (gP ! i) (P ! i)) \wedge$   
 $\text{carrier } F2 = (\bigcup i < \text{length } Q. F2\text{-act.image-set } (gQ ! i) (Q ! i))$   
 $\langle \text{proof} \rangle$

We unfold *F2-act.paradoxical* directly to avoid locale-instantiation complications with implicit polymorphic variables.

**theorem** *F2-paradoxical*:  
 $F2\text{-act.paradoxical } (\text{carrier } F2)$   
 $\langle \text{proof} \rangle$

**end**

**theory** *Free-Rotations-SO3*  
**imports**  
*Banach-Tarski.F2-Paradox*  
*Banach-Tarski.Free-Action-Paradox*  
*HOL-Computational-Algebra.Primes*  
*Free-Groups.PingPongLemma*  
**begin**

## 18 Rotations of three-space

A rotation is an orthogonal transformation of determinant 1.

**definition** *rotation* :: ( $\text{real}^3 \Rightarrow \text{real}^3$ )  $\Rightarrow$  *bool* **where**  
*rotation*  $f \iff \text{orthogonal-transformation } f \wedge \det(\text{matrix } f) = 1$

**definition** *SO3* :: ( $\text{real}^3 \Rightarrow \text{real}^3$ ) *set* **where**  
 $\text{SO3} = \{f. \text{rotation } f\}$

SO(3) contains the identity and is closed under composition.

**lemma** *id-in-SO3*:  $\text{id} \in \text{SO3}$   
(*proof*)

**lemma** *SO3-closed-compose*:  
**assumes**  $f \in \text{SO3}$  **and**  $g \in \text{SO3}$   
**shows**  $f \circ g \in \text{SO3}$   
(*proof*)

SO(3) elements preserve the unit sphere.

**lemma** *rotation-preserves-sphere2*:  
**assumes**  $f \in \text{SO3}$  **and**  $x \in \text{sphere2}$   
**shows**  $f x \in \text{sphere2}$   
(*proof*)

**lemma** *SO3-bij*:  
**assumes**  $f \in \text{SO3}$   
**shows** *bij*  $f$   
(*proof*)

## 19 Two distinguished rotations

We use the rotations  $R_x$  (about the x-axis) and  $R_z$  (about the z-axis) by the same angle  $\vartheta$  with  $\cos \vartheta = 1/3$ . The exact-form witnesses are unimportant for the paradoxical decomposition; what matters is the abstract conclusion that these generate a free  $F_2$  inside SO(3).

**definition** *rot-c* :: *real* **where**  
 $\text{rot-c} = 1 / 3$

**definition** *rot-s* :: *real* **where**  
 $\text{rot-s} = \text{sqrt } 8 / 3$

**lemma** *rot-c-sq-plus-rot-s-sq*:  $\text{rot-c}^2 + \text{rot-s}^2 = 1$   
(*proof*)

**lemma** *rot-c-mul-plus-rot-s-mul* [*simp*]:  $\text{rot-c} * \text{rot-c} + \text{rot-s} * \text{rot-s} = 1$   
(*proof*)

**definition**  $Rx\text{-mat} :: \text{real}^3^3$  **where**

$Rx\text{-mat} = \text{vector}$  [  
   $\text{vector}$   $[1, 0, 0]$ ,  
   $\text{vector}$   $[0, \text{rot-c}, -\text{rot-s}]$ ,  
   $\text{vector}$   $[0, \text{rot-s}, \text{rot-c}]$ ]

**definition**  $Rz\text{-mat} :: \text{real}^3^3$  **where**

$Rz\text{-mat} = \text{vector}$  [  
   $\text{vector}$   $[\text{rot-c}, -\text{rot-s}, 0]$ ,  
   $\text{vector}$   $[\text{rot-s}, \text{rot-c}, 0]$ ,  
   $\text{vector}$   $[0, 0, 1]$ ]

**definition**  $Rx\text{-inv-mat} :: \text{real}^3^3$  **where**

$Rx\text{-inv-mat} = \text{vector}$  [  
   $\text{vector}$   $[1, 0, 0]$ ,  
   $\text{vector}$   $[0, \text{rot-c}, \text{rot-s}]$ ,  
   $\text{vector}$   $[0, -\text{rot-s}, \text{rot-c}]$ ]

**definition**  $Rz\text{-inv-mat} :: \text{real}^3^3$  **where**

$Rz\text{-inv-mat} = \text{vector}$  [  
   $\text{vector}$   $[\text{rot-c}, \text{rot-s}, 0]$ ,  
   $\text{vector}$   $[-\text{rot-s}, \text{rot-c}, 0]$ ,  
   $\text{vector}$   $[0, 0, 1]$ ]

**definition**  $Rx :: \text{real}^3 \Rightarrow \text{real}^3$  **where**

$Rx\ v = Rx\text{-mat} * v$

**definition**  $Rz :: \text{real}^3 \Rightarrow \text{real}^3$  **where**

$Rz\ v = Rz\text{-mat} * v$

**definition**  $Rx\text{-inv} :: \text{real}^3 \Rightarrow \text{real}^3$  **where**

$Rx\text{-inv}\ v = Rx\text{-inv-mat} * v$

**definition**  $Rz\text{-inv} :: \text{real}^3 \Rightarrow \text{real}^3$  **where**

$Rz\text{-inv}\ v = Rz\text{-inv-mat} * v$

**lemma**  $Rx\text{-mat-orthogonal}$ : *orthogonal-matrix*  $Rx\text{-mat}$   
(*proof*)

**lemma**  $Rz\text{-mat-orthogonal}$ : *orthogonal-matrix*  $Rz\text{-mat}$   
(*proof*)

**lemma**  $Rx\text{-inv-mat-orthogonal}$ : *orthogonal-matrix*  $Rx\text{-inv-mat}$   
(*proof*)

**lemma**  $Rz\text{-inv-mat-orthogonal}$ : *orthogonal-matrix*  $Rz\text{-inv-mat}$   
(*proof*)

**lemma**  $\text{det-Rx-mat}$  [*simp*]:  $\text{det}\ Rx\text{-mat} = 1$

*<proof>*

**lemma** *det-Rz-mat* [*simp*]: *det Rz-mat = 1*  
*<proof>*

**lemma** *det-Rx-inv-mat* [*simp*]: *det Rx-inv-mat = 1*  
*<proof>*

**lemma** *det-Rz-inv-mat* [*simp*]: *det Rz-inv-mat = 1*  
*<proof>*

**lemma** *Rx-mat-inverse-left*: *Rx-inv-mat \*\* Rx-mat = mat 1*  
*<proof>*

**lemma** *Rx-mat-inverse-right*: *Rx-mat \*\* Rx-inv-mat = mat 1*  
*<proof>*

**lemma** *Rz-mat-inverse-left*: *Rz-inv-mat \*\* Rz-mat = mat 1*  
*<proof>*

**lemma** *Rz-mat-inverse-right*: *Rz-mat \*\* Rz-inv-mat = mat 1*  
*<proof>*

The concrete generators and their inverses lie in  $SO(3)$ .

**lemma** *Rx-in-SO3*: *Rx ∈ SO3*  
*<proof>*

**lemma** *Rz-in-SO3*: *Rz ∈ SO3*  
*<proof>*

**lemma** *Rx-inv-in-SO3*: *Rx-inv ∈ SO3*  
*<proof>*

**lemma** *Rz-inv-in-SO3*: *Rz-inv ∈ SO3*  
*<proof>*

The named inverse rotations are two-sided inverses of the generators.

**lemma** *Rx-inv-left*: *Rx-inv ∘ Rx = id*  
*<proof>*

**lemma** *Rz-inv-left*: *Rz-inv ∘ Rz = id*  
*<proof>*

**lemma** *Rx-inv-right*: *Rx ∘ Rx-inv = id*  
*<proof>*

**lemma** *Rz-inv-right*: *Rz ∘ Rz-inv = id*  
*<proof>*

## 20 The rotations as bijections of the sphere

**definition**  $sphere\text{-}perm :: (real^3 \Rightarrow real^3) \Rightarrow (real^3 \Rightarrow real^3)$  **where**  
 $sphere\text{-}perm\ f = restrict\ f\ sphere2$

**lemma**  $SO3\text{-}bij\text{-}betw\text{-}sphere2$ :  
**assumes**  $f \in SO3$   
**shows**  $bij\text{-}betw\ f\ sphere2\ sphere2$   
 $\langle proof \rangle$

**lemma**  $sphere\text{-}perm\text{-}in\text{-}Bij$ :  
**assumes**  $f \in SO3$   
**shows**  $sphere\text{-}perm\ f \in Bij\ sphere2$   
 $\langle proof \rangle$

**lemma**  $sphere\text{-}perm\text{-}mult$ :  
**assumes**  $sphere\text{-}perm\ f \in Bij\ sphere2\ sphere\text{-}perm\ g \in Bij\ sphere2$   
**shows**  $sphere\text{-}perm\ f \otimes_{BijGroup\ sphere2}\ sphere\text{-}perm\ g = sphere\text{-}perm\ (f \circ g)$   
 $\langle proof \rangle$

**lemma**  $sphere\text{-}perm\text{-}id$ :  $sphere\text{-}perm\ id = \mathbf{1}_{BijGroup\ sphere2}$   
 $\langle proof \rangle$

**lemma**  $sphere\text{-}perm\text{-}Rx\ [simp]$ :  $sphere\text{-}perm\ Rx \in Bij\ sphere2$   
 $\langle proof \rangle$

**lemma**  $sphere\text{-}perm\text{-}Rx\text{-}inv\ [simp]$ :  $sphere\text{-}perm\ Rx\text{-}inv \in Bij\ sphere2$   
 $\langle proof \rangle$

**lemma**  $sphere\text{-}perm\text{-}Rz\ [simp]$ :  $sphere\text{-}perm\ Rz \in Bij\ sphere2$   
 $\langle proof \rangle$

**lemma**  $sphere\text{-}perm\text{-}Rz\text{-}inv\ [simp]$ :  $sphere\text{-}perm\ Rz\text{-}inv \in Bij\ sphere2$   
 $\langle proof \rangle$

**lemma**  $sphere\text{-}perm\text{-}Rx\text{-}inv\text{-}left$ :  
 $sphere\text{-}perm\ Rx\text{-}inv \otimes_{BijGroup\ sphere2}\ sphere\text{-}perm\ Rx = \mathbf{1}_{BijGroup\ sphere2}$   
 $\langle proof \rangle$

**lemma**  $sphere\text{-}perm\text{-}Rx\text{-}inv\text{-}right$ :  
 $sphere\text{-}perm\ Rx \otimes_{BijGroup\ sphere2}\ sphere\text{-}perm\ Rx\text{-}inv = \mathbf{1}_{BijGroup\ sphere2}$   
 $\langle proof \rangle$

**lemma**  $sphere\text{-}perm\text{-}Rz\text{-}inv\text{-}left$ :  
 $sphere\text{-}perm\ Rz\text{-}inv \otimes_{BijGroup\ sphere2}\ sphere\text{-}perm\ Rz = \mathbf{1}_{BijGroup\ sphere2}$   
 $\langle proof \rangle$

**lemma**  $sphere\text{-}perm\text{-}Rz\text{-}inv\text{-}right$ :  
 $sphere\text{-}perm\ Rz \otimes_{BijGroup\ sphere2}\ sphere\text{-}perm\ Rz\text{-}inv = \mathbf{1}_{BijGroup\ sphere2}$

*<proof>*

**lemma** *inv-sphere-perm-Rx:*

$inv_{BijGroup\ sphere2} (sphere-perm\ Rx) = sphere-perm\ Rx-inv$   
*<proof>*

**lemma** *inv-sphere-perm-Rz:*

$inv_{BijGroup\ sphere2} (sphere-perm\ Rz) = sphere-perm\ Rz-inv$   
*<proof>*

**interpretation** *sphere-Bij: group BijGroup sphere2*

*<proof>*

## 21 The map sigma: $F_2$ to $SO(3)$

The map *sigma* sends each reduced word in  $F_2$  to the corresponding composition of  $R_x$ ,  $R_z$ , and their inverses.

**definition** *letter-to-rot :: bool × gen2 ⇒ (real<sup>3</sup> ⇒ real<sup>3</sup>) where*

*letter-to-rot p = (case p of*  
    *(False, A) ⇒ Rx*  
    *| (True, A) ⇒ Rx-inv*  
    *| (False, B) ⇒ Rz*  
    *| (True, B) ⇒ Rz-inv)*

**definition** *rho :: gen2 ⇒ (real<sup>3</sup> ⇒ real<sup>3</sup>) where*

*rho x = (case x of A ⇒ sphere-perm Rx | B ⇒ sphere-perm Rz)*

**lemma** *rho-in-BijGroup:*

$rho \in Gen2 \rightarrow carrier (BijGroup\ sphere2)$   
*<proof>*

**lemma** *rho-image-subset-BijGroup:*

$rho \text{ ' } Gen2 \subseteq carrier (BijGroup\ sphere2)$   
*<proof>*

**definition** *sphere-rot-group :: (real<sup>3</sup> ⇒ real<sup>3</sup>) monoid where*

*sphere-rot-group =*  
*(BijGroup sphere2)(carrier := <rho ' Gen2>BijGroup sphere2)*

**lemma** *sphere-rot-subgroup:*

$subgroup (carrier\ sphere-rot-group) (BijGroup\ sphere2)$   
*<proof>*

**lemma** *sphere-rot-group-is-group: group sphere-rot-group*

*<proof>*

**interpretation** *sphere-rot: group sphere-rot-group*

*<proof>*

**lemma** *rho-in-sphere-rot-group*:

$\rho \in \text{Gen2} \rightarrow \text{carrier sphere-rot-group}$   
 $\langle \text{proof} \rangle$

**lemma** *sphere-rot-inclusion-hom*:

$\text{id} \in \text{hom sphere-rot-group (BijGroup sphere2)}$   
 $\langle \text{proof} \rangle$

**lemma** *sphere-rot-generated*:

$\langle \rho \text{ ' Gen2} \rangle \text{sphere-rot-group} = \text{carrier sphere-rot-group}$   
 $\langle \text{proof} \rangle$

**lemma** *sphere-Bij-lift-gi-rho*:

**assumes**  $\text{snd } p \in \text{Gen2}$   
**shows**  $\text{sphere-Bij.lift-gi } \rho \text{ } p = \text{sphere-perm (letter-to-rot } p)$   
 $\langle \text{proof} \rangle$

**lemma** *sphere-rot-inv-rho-eq-sphere-Bij*:

**assumes**  $i \in \text{Gen2}$   
**shows**  $\text{inv}_{\text{sphere-rot-group}} (\rho \text{ } i) = \text{inv}_{\text{BijGroup sphere2}} (\rho \text{ } i)$   
 $\langle \text{proof} \rangle$

**lemma** *sphere-rot-lift-gi-rho-eq-sphere-Bij*:

**assumes**  $\text{snd } p \in \text{Gen2}$   
**shows**  $\text{sphere-rot.lift-gi } \rho \text{ } p = \text{sphere-Bij.lift-gi } \rho \text{ } p$   
 $\langle \text{proof} \rangle$

**lemma** *sphere-rot-lift-rho-eq-sphere-Bij*:

**assumes**  $w \in \text{lists (UNIV } \times \text{ Gen2)}$   
**shows**  $\text{sphere-rot.lift } \rho \text{ } w = \text{sphere-Bij.lift } \rho \text{ } w$   
 $\langle \text{proof} \rangle$

**lemma** *sphere-Bij-lift-rho-injective-if-ping-pong*:

**fixes**  $X_{\text{in}} X_{\text{out}} :: \text{gen2} \Rightarrow (\text{real}^3) \text{ set}$   
**assumes**  $\text{sub-out: } \forall i \in \text{Gen2}. X_{\text{out}} \text{ } i \subseteq \text{sphere2}$   
**and**  $\text{sub-in: } \forall i \in \text{Gen2}. X_{\text{in}} \text{ } i \subseteq \text{sphere2}$   
**and**  $\text{disj-out: } \forall i \in \text{Gen2}. \forall j \in \text{Gen2}. i \neq j \longrightarrow X_{\text{out}} \text{ } i \cap X_{\text{out}} \text{ } j = \{\}$   
**and**  $\text{disj-in: } \forall i \in \text{Gen2}. \forall j \in \text{Gen2}. i \neq j \longrightarrow X_{\text{in}} \text{ } i \cap X_{\text{in}} \text{ } j = \{\}$   
**and**  $\text{disj-cross: } \forall i \in \text{Gen2}. \forall j \in \text{Gen2}. X_{\text{in}} \text{ } i \cap X_{\text{out}} \text{ } j = \{\}$   
**and**  $x\text{-sphere: } x \in \text{sphere2}$   
**and**  $\text{ping: } \forall i \in \text{Gen2}. \rho \text{ } i \text{ ' (sphere2 - } X_{\text{out}} \text{ } i) \subseteq X_{\text{in}} \text{ } i$   
**and**  $\text{pong: } \forall i \in \text{Gen2}. (\text{inv}_{\text{sphere-rot-group}} (\rho \text{ } i)) \text{ ' (sphere2 - } X_{\text{in}} \text{ } i) \subseteq X_{\text{out}} \text{ } i$   
**shows**  $\text{inj-on (sphere-Bij.lift } \rho) (\text{carrier } F2)$   
 $\langle \text{proof} \rangle$

**definition**  $\text{sigma} :: (\text{bool} \times \text{gen2}) \text{ list} \Rightarrow (\text{real}^3 \Rightarrow \text{real}^3) \text{ where}$

$\text{sigma } w = \text{foldr } (\lambda p \text{ } f. \text{letter-to-rot } p \circ f) \text{ } w \text{ } \text{id}$

**lemma** *letter-to-rot-cancel-left* [simp]:

*letter-to-rot* (True, A)  $\circ$  *letter-to-rot* (False, A) = *id*

*letter-to-rot* (False, A)  $\circ$  *letter-to-rot* (True, A) = *id*

*letter-to-rot* (True, B)  $\circ$  *letter-to-rot* (False, B) = *id*

*letter-to-rot* (False, B)  $\circ$  *letter-to-rot* (True, B) = *id*

$\langle$ *proof* $\rangle$

**lemma** *letter-to-rot-canceling*:

**assumes** *canceling* p q

**shows** *letter-to-rot* p  $\circ$  *letter-to-rot* q = *id*

$\langle$ *proof* $\rangle$

**lemma** *sigma-Nil* [simp]: *sigma* [] = *id*

$\langle$ *proof* $\rangle$

**lemma** *sigma-Cons* [simp]: *sigma* (p # w) = *letter-to-rot* p  $\circ$  *sigma* w

$\langle$ *proof* $\rangle$

**lemma** *sigma-append* [simp]: *sigma* (u @ v) = *sigma* u  $\circ$  *sigma* v

$\langle$ *proof* $\rangle$

**lemma** *sigma-singleton* [simp]: *sigma* [p] = *letter-to-rot* p

$\langle$ *proof* $\rangle$

**lemma** *sigma-doubleton-canceling*:

**assumes** *canceling* p q

**shows** *sigma* [p, q] = *id*

$\langle$ *proof* $\rangle$

**lemma** *sigma-cancels-to-1*:

**assumes** *cancels-to-1* x y

**shows** *sigma* x = *sigma* y

$\langle$ *proof* $\rangle$

**lemma** *sigma-cancels-to*:

**assumes** *cancels-to* x y

**shows** *sigma* x = *sigma* y

$\langle$ *proof* $\rangle$

**lemma** *sigma-normalize* [simp]: *sigma* (*normalize* w) = *sigma* w

$\langle$ *proof* $\rangle$

**lemma** *letter-to-rot-in-SO3*: *letter-to-rot* p  $\in$  *SO3*

$\langle$ *proof* $\rangle$

**lemma** *sphere-perm-letter-to-rot-in-Bij* [simp]:

*sphere-perm* (*letter-to-rot* p)  $\in$  *Bij* *sphere2*

$\langle$ *proof* $\rangle$

**lemma** *sigma-in-SO3*:  $\text{sigma } w \in \text{SO}3$   
 ⟨proof⟩

**lemma** *sigma-preserves-sphere2*:  
 assumes  $x \in \text{sphere}2$   
 shows  $\text{sigma } w x \in \text{sphere}2$   
 ⟨proof⟩

**lemma** *sigma-bij*:  $\text{bij } (\text{sigma } w)$   
 ⟨proof⟩

**lemma** *sphere-perm-sigma-in-Bij* [simp]:  
 $\text{sphere-perm } (\text{sigma } w) \in \text{Bij } \text{sphere}2$   
 ⟨proof⟩

**lemma** *sphere-Bij-lift-rho-eq-sigma*:  
 assumes  $w \in \text{lists } (\text{UNIV} \times \text{Gen}2)$   
 shows  $\text{sphere-Bij.lift } \rho w = \text{sphere-perm } (\text{sigma } w)$   
 ⟨proof⟩

**lemma** *sigma-injective-if-sphere-lift-injective*:  
 assumes  $\text{inj-on } (\text{sphere-Bij.lift } \rho) (\text{carrier } F2)$   
 shows  $\text{inj-on } \text{sigma } (\text{carrier } F2)$   
 ⟨proof⟩

**lemma** *sigma-injective-if-ping-pong*:  
 fixes  $Xin Xout :: \text{gen}2 \Rightarrow (\text{real}^3) \text{ set}$   
 assumes  $\text{sub-out}: \forall i \in \text{Gen}2. Xout\ i \subseteq \text{sphere}2$   
 and  $\text{sub-in}: \forall i \in \text{Gen}2. Xin\ i \subseteq \text{sphere}2$   
 and  $\text{disj-out}: \forall i \in \text{Gen}2. \forall j \in \text{Gen}2. i \neq j \longrightarrow Xout\ i \cap Xout\ j = \{\}$   
 and  $\text{disj-in}: \forall i \in \text{Gen}2. \forall j \in \text{Gen}2. i \neq j \longrightarrow Xin\ i \cap Xin\ j = \{\}$   
 and  $\text{disj-cross}: \forall i \in \text{Gen}2. \forall j \in \text{Gen}2. Xin\ i \cap Xout\ j = \{\}$   
 and  $x\text{-sphere}: x \in \text{sphere}2$   
 and  $\text{ping}: \forall i \in \text{Gen}2. \rho\ i \text{ ' } (\text{sphere}2 - Xout\ i) \subseteq Xin\ i$   
 and  $\text{pong}: \forall i \in \text{Gen}2. (\text{inv}_{\text{sphere-rot-group}} (\rho\ i)) \text{ ' } (\text{sphere}2 - Xin\ i) \subseteq Xout\ i$   
 shows  $\text{inj-on } \text{sigma } (\text{carrier } F2)$   
 ⟨proof⟩

The ping-pong criterion gives a reusable injectivity criterion for *sigma*.

## 22 Arithmetic freeness invariant

The classical proof that the above rotations generate a free subgroup is arithmetic rather than geometric: scale each letter by  $\mathfrak{3}$ , so every word has entries in the integer quadratic ring generated by  $\text{sqrt } 2$ . A nonempty reduced word leaves a coefficient nonzero modulo  $\mathfrak{3}$  after applying it to a suitable basis vector, whereas the identity matrix would have all scaled off-axis coefficients divisible by  $\mathfrak{3}$ .

The following definitions make that invariant explicit. They are deliberately separated from real-vector semantics so that the finite modulo-3 induction can be discharged by simplification and case analysis before being connected back to *sigma*.

**type-synonym**  $zsqrt2 = int \times int$

**type-synonym**  $zvec3 = zsqrt2 \times zsqrt2 \times zsqrt2$

**definition**  $zsqrt2-val :: zsqrt2 \Rightarrow real$  **where**

$zsqrt2-val\ q = of-int\ (fst\ q) + of-int\ (snd\ q) * sqrt\ 2$

**definition**  $zsqrt2-add :: zsqrt2 \Rightarrow zsqrt2 \Rightarrow zsqrt2$  **where**

$zsqrt2-add\ p\ q = (fst\ p + fst\ q, snd\ p + snd\ q)$

**definition**  $zsqrt2-neg :: zsqrt2 \Rightarrow zsqrt2$  **where**

$zsqrt2-neg\ p = (-\ fst\ p, -\ snd\ p)$

**definition**  $zsqrt2-sub :: zsqrt2 \Rightarrow zsqrt2 \Rightarrow zsqrt2$  **where**

$zsqrt2-sub\ p\ q = zsqrt2-add\ p\ (zsqrt2-neg\ q)$

**definition**  $zsqrt2-scale :: int \Rightarrow zsqrt2 \Rightarrow zsqrt2$  **where**

$zsqrt2-scale\ n\ p = (n * fst\ p, n * snd\ p)$

**definition**  $zsqrt2-mul-sqrt2 :: zsqrt2 \Rightarrow zsqrt2$  **where**

$zsqrt2-mul-sqrt2\ p = (2 * snd\ p, fst\ p)$

**definition**  $zsqrt2-div3 :: zsqrt2 \Rightarrow bool$  **where**

$zsqrt2-div3\ p \longleftrightarrow 3\ dvd\ fst\ p \wedge 3\ dvd\ snd\ p$

**definition**  $zvec3-div3 :: zvec3 \Rightarrow bool$  **where**

$zvec3-div3\ v \longleftrightarrow$

$(case\ v\ of\ (x, y, z) \Rightarrow zsqrt2-div3\ x \wedge zsqrt2-div3\ y \wedge zsqrt2-div3\ z)$

**definition**  $zvec3-scale :: int \Rightarrow zvec3 \Rightarrow zvec3$  **where**

$zvec3-scale\ n\ v = (case\ v\ of\ (x, y, z) \Rightarrow$   
 $(zsqrt2-scale\ n\ x, zsqrt2-scale\ n\ y, zsqrt2-scale\ n\ z))$

**definition**  $zvec3-val :: zvec3 \Rightarrow real^3$  **where**

$zvec3-val\ v = (case\ v\ of\ (x, y, z) \Rightarrow$   
 $vector\ [zsqrt2-val\ x, zsqrt2-val\ y, zsqrt2-val\ z])$

**definition**  $ze-x :: zvec3$  **where**

$ze-x = ((1, 0), (0, 0), (0, 0))$

**definition**  $ze-y :: zvec3$  **where**

$ze-y = ((0, 0), (1, 0), (0, 0))$

**definition**  $ze-z :: zvec3$  **where**

$ze-z = ((0, 0), (0, 0), (1, 0))$

**definition**  $zRx\text{-step} :: \text{zvec3} \Rightarrow \text{zvec3}$  **where**  
 $zRx\text{-step } v = (\text{case } v \text{ of } (x, y, z) \Rightarrow$   
 $(\text{zsqr2-scale } 3 \ x,$   
 $\text{zsqr2-sub } y \ (\text{zsqr2-scale } 2 \ (\text{zsqr2-mul-sqr2 } z)),$   
 $\text{zsqr2-add } (\text{zsqr2-scale } 2 \ (\text{zsqr2-mul-sqr2 } y)) \ z))$

**definition**  $zRx\text{-inv-step} :: \text{zvec3} \Rightarrow \text{zvec3}$  **where**  
 $zRx\text{-inv-step } v = (\text{case } v \text{ of } (x, y, z) \Rightarrow$   
 $(\text{zsqr2-scale } 3 \ x,$   
 $\text{zsqr2-add } y \ (\text{zsqr2-scale } 2 \ (\text{zsqr2-mul-sqr2 } z)),$   
 $\text{zsqr2-sub } z \ (\text{zsqr2-scale } 2 \ (\text{zsqr2-mul-sqr2 } y))))$

**definition**  $zRz\text{-step} :: \text{zvec3} \Rightarrow \text{zvec3}$  **where**  
 $zRz\text{-step } v = (\text{case } v \text{ of } (x, y, z) \Rightarrow$   
 $(\text{zsqr2-sub } x \ (\text{zsqr2-scale } 2 \ (\text{zsqr2-mul-sqr2 } y)),$   
 $\text{zsqr2-add } (\text{zsqr2-scale } 2 \ (\text{zsqr2-mul-sqr2 } x)) \ y,$   
 $\text{zsqr2-scale } 3 \ z))$

**definition**  $zRz\text{-inv-step} :: \text{zvec3} \Rightarrow \text{zvec3}$  **where**  
 $zRz\text{-inv-step } v = (\text{case } v \text{ of } (x, y, z) \Rightarrow$   
 $(\text{zsqr2-add } x \ (\text{zsqr2-scale } 2 \ (\text{zsqr2-mul-sqr2 } y)),$   
 $\text{zsqr2-sub } y \ (\text{zsqr2-scale } 2 \ (\text{zsqr2-mul-sqr2 } x)),$   
 $\text{zsqr2-scale } 3 \ z))$

**definition**  $zletter\text{-step} :: \text{bool} \times \text{gen2} \Rightarrow \text{zvec3} \Rightarrow \text{zvec3}$  **where**  
 $zletter\text{-step } p = (\text{case } p \text{ of}$   
 $(\text{False}, A) \Rightarrow zRx\text{-step}$   
 $| (\text{True}, A) \Rightarrow zRx\text{-inv-step}$   
 $| (\text{False}, B) \Rightarrow zRz\text{-step}$   
 $| (\text{True}, B) \Rightarrow zRz\text{-inv-step})$

**definition**  $zword\text{-step} :: (\text{bool} \times \text{gen2}) \text{ list} \Rightarrow \text{zvec3} \Rightarrow \text{zvec3}$  **where**  
 $zword\text{-step } w \ v = \text{foldr } (\lambda p \ f. \ zletter\text{-step } p \circ f) \ w \ \text{id } v$

**definition**  $zword\text{-witness} :: (\text{bool} \times \text{gen2}) \text{ list} \Rightarrow \text{zvec3}$  **where**  
 $zword\text{-witness } w = (\text{case } \text{snd } (\text{last } w) \text{ of } A \Rightarrow ze\text{-y} \mid B \Rightarrow ze\text{-x})$

**datatype**  $r3 = R0 \mid R1 \mid R2$

**type-synonym**  $rcoef = r3 \times r3$

**type-synonym**  $rvec3 = rcoef \times rcoef \times rcoef$

**fun**  $r3\text{-add} :: r3 \Rightarrow r3 \Rightarrow r3$  **where**

$r3\text{-add } R0 \ x = x$   
 $| r3\text{-add } x \ R0 = x$   
 $| r3\text{-add } R1 \ R1 = R2$   
 $| r3\text{-add } R1 \ R2 = R0$   
 $| r3\text{-add } R2 \ R1 = R0$   
 $| r3\text{-add } R2 \ R2 = R1$

**fun** *r3-neg* :: *r3* ⇒ *r3* **where**

*r3-neg* *R0* = *R0*  
| *r3-neg* *R1* = *R2*  
| *r3-neg* *R2* = *R1*

**definition** *r3-sub* :: *r3* ⇒ *r3* ⇒ *r3* **where**

*r3-sub* *x y* = *r3-add* *x* (*r3-neg* *y*)

**fun** *r3-double* :: *r3* ⇒ *r3* **where**

*r3-double* *R0* = *R0*  
| *r3-double* *R1* = *R2*  
| *r3-double* *R2* = *R1*

**definition** *rcoef-add* :: *rcoef* ⇒ *rcoef* ⇒ *rcoef* **where**

*rcoef-add* *p q* = (*r3-add* (*fst* *p*) (*fst* *q*), *r3-add* (*snd* *p*) (*snd* *q*))

**definition** *rcoef-neg* :: *rcoef* ⇒ *rcoef* **where**

*rcoef-neg* *p* = (*r3-neg* (*fst* *p*), *r3-neg* (*snd* *p*))

**definition** *rcoef-sub* :: *rcoef* ⇒ *rcoef* ⇒ *rcoef* **where**

*rcoef-sub* *p q* = *rcoef-add* *p* (*rcoef-neg* *q*)

**definition** *rcoef-double* :: *rcoef* ⇒ *rcoef* **where**

*rcoef-double* *p* = (*r3-double* (*fst* *p*), *r3-double* (*snd* *p*))

**definition** *rcoef-mul-sqrt2* :: *rcoef* ⇒ *rcoef* **where**

*rcoef-mul-sqrt2* *p* = (*r3-double* (*snd* *p*), *fst* *p*)

**definition** *rRx-step* :: *rvec3* ⇒ *rvec3* **where**

*rRx-step* *v* = (case *v* of (*x*, *y*, *z*) ⇒  
((*R0*, *R0*),  
*rcoef-sub* *y* (*rcoef-double* (*rcoef-mul-sqrt2* *z*)),  
*rcoef-add* (*rcoef-double* (*rcoef-mul-sqrt2* *y*)) *z*))

**definition** *rRx-inv-step* :: *rvec3* ⇒ *rvec3* **where**

*rRx-inv-step* *v* = (case *v* of (*x*, *y*, *z*) ⇒  
((*R0*, *R0*),  
*rcoef-add* *y* (*rcoef-double* (*rcoef-mul-sqrt2* *z*)),  
*rcoef-sub* *z* (*rcoef-double* (*rcoef-mul-sqrt2* *y*))))

**definition** *rRz-step* :: *rvec3* ⇒ *rvec3* **where**

*rRz-step* *v* = (case *v* of (*x*, *y*, *z*) ⇒  
(*rcoef-sub* *x* (*rcoef-double* (*rcoef-mul-sqrt2* *y*)),  
*rcoef-add* (*rcoef-double* (*rcoef-mul-sqrt2* *x*)) *y*,  
(*R0*, *R0*)))

**definition** *rRz-inv-step* :: *rvec3* ⇒ *rvec3* **where**

*rRz-inv-step* *v* = (case *v* of (*x*, *y*, *z*) ⇒

(*rcoef-add* *x* (*rcoef-double* (*rcoef-mul-sqrt2* *y*)),  
*rcoef-sub* *y* (*rcoef-double* (*rcoef-mul-sqrt2* *x*)),  
(*R0*, *R0*))

**definition** *rletter-step* :: *bool* × *gen2* ⇒ *rvec3* ⇒ *rvec3* **where**

*rletter-step* *p* = (case *p* of  
(*False*, *A*) ⇒ *rRx-step*  
| (*True*, *A*) ⇒ *rRx-inv-step*  
| (*False*, *B*) ⇒ *rRz-step*  
| (*True*, *B*) ⇒ *rRz-inv-step*)

**definition** *rword-step* :: (*bool* × *gen2*) *list* ⇒ *rvec3* ⇒ *rvec3* **where**

*rword-step* *w* *v* = *foldr* ( $\lambda p f. rletter-step\ p \circ f$ ) *w* *id* *v*

**definition** *re-x* :: *rvec3* **where**

*re-x* = ((*R1*, *R0*), (*R0*, *R0*), (*R0*, *R0*))

**definition** *re-y* :: *rvec3* **where**

*re-y* = ((*R0*, *R0*), (*R1*, *R0*), (*R0*, *R0*))

**definition** *rword-witness* :: (*bool* × *gen2*) *list* ⇒ *rvec3* **where**

*rword-witness* *w* = (case *snd* (*last* *w*) of *A* ⇒ *re-y* | *B* ⇒ *re-x*)

**definition** *rzero-vec* :: *rvec3* **where**

*rzero-vec* = ((*R0*, *R0*), (*R0*, *R0*), (*R0*, *R0*))

**definition** *rstate* :: *bool* × *gen2* ⇒ *rvec3* ⇒ *bool* **where**

*rstate* *p* *v*  $\longleftrightarrow$   
*v* ∈ (case *p* of  
(*False*, *A*) ⇒ {  
((*R0*, *R0*), (*R0*, *R1*), (*R1*, *R0*)),  
((*R0*, *R0*), (*R0*, *R2*), (*R2*, *R0*)),  
((*R0*, *R0*), (*R1*, *R0*), (*R0*, *R2*)),  
((*R0*, *R0*), (*R2*, *R0*), (*R0*, *R1*))}  
| (*True*, *A*) ⇒ {  
((*R0*, *R0*), (*R0*, *R1*), (*R2*, *R0*)),  
((*R0*, *R0*), (*R0*, *R2*), (*R1*, *R0*)),  
((*R0*, *R0*), (*R1*, *R0*), (*R0*, *R1*)),  
((*R0*, *R0*), (*R2*, *R0*), (*R0*, *R2*))}  
| (*False*, *B*) ⇒ {  
((*R0*, *R1*), (*R1*, *R0*), (*R0*, *R0*)),  
((*R0*, *R2*), (*R2*, *R0*), (*R0*, *R0*)),  
((*R1*, *R0*), (*R0*, *R2*), (*R0*, *R0*)),  
((*R2*, *R0*), (*R0*, *R1*), (*R0*, *R0*))}  
| (*True*, *B*) ⇒ {  
((*R0*, *R1*), (*R2*, *R0*), (*R0*, *R0*)),  
((*R0*, *R2*), (*R1*, *R0*), (*R0*, *R0*)),  
((*R1*, *R0*), (*R0*, *R1*), (*R0*, *R0*)),  
((*R2*, *R0*), (*R0*, *R2*), (*R0*, *R0*))})

**lemma** *rstate-single*:

$rstate\ p\ (rletter\text{-}step\ p\ (case\ snd\ p\ of\ A\ \Rightarrow\ re\text{-}y\ |\ B\ \Rightarrow\ re\text{-}x))$   
 $\langle proof \rangle$

**lemma** *rstate-step-FA-FA*:

$rstate\ (False,\ A)\ v\ \Longrightarrow\ rstate\ (False,\ A)\ (rletter\text{-}step\ (False,\ A)\ v)$   
 $\langle proof \rangle$

**lemma** *rstate-step-FA-FB*:

$rstate\ (False,\ B)\ v\ \Longrightarrow\ rstate\ (False,\ A)\ (rletter\text{-}step\ (False,\ A)\ v)$   
 $\langle proof \rangle$

**lemma** *rstate-step-FA-TB*:

$rstate\ (True,\ B)\ v\ \Longrightarrow\ rstate\ (False,\ A)\ (rletter\text{-}step\ (False,\ A)\ v)$   
 $\langle proof \rangle$

**lemma** *rstate-step-TA-TA*:

$rstate\ (True,\ A)\ v\ \Longrightarrow\ rstate\ (True,\ A)\ (rletter\text{-}step\ (True,\ A)\ v)$   
 $\langle proof \rangle$

**lemma** *rstate-step-TA-FB*:

$rstate\ (False,\ B)\ v\ \Longrightarrow\ rstate\ (True,\ A)\ (rletter\text{-}step\ (True,\ A)\ v)$   
 $\langle proof \rangle$

**lemma** *rstate-step-TA-TB*:

$rstate\ (True,\ B)\ v\ \Longrightarrow\ rstate\ (True,\ A)\ (rletter\text{-}step\ (True,\ A)\ v)$   
 $\langle proof \rangle$

**lemma** *rstate-step-FB-FB*:

$rstate\ (False,\ B)\ v\ \Longrightarrow\ rstate\ (False,\ B)\ (rletter\text{-}step\ (False,\ B)\ v)$   
 $\langle proof \rangle$

**lemma** *rstate-step-FB-FA*:

$rstate\ (False,\ A)\ v\ \Longrightarrow\ rstate\ (False,\ B)\ (rletter\text{-}step\ (False,\ B)\ v)$   
 $\langle proof \rangle$

**lemma** *rstate-step-FB-TA*:

$rstate\ (True,\ A)\ v\ \Longrightarrow\ rstate\ (False,\ B)\ (rletter\text{-}step\ (False,\ B)\ v)$   
 $\langle proof \rangle$

**lemma** *rstate-step-TB-TB*:

$rstate\ (True,\ B)\ v\ \Longrightarrow\ rstate\ (True,\ B)\ (rletter\text{-}step\ (True,\ B)\ v)$   
 $\langle proof \rangle$

**lemma** *rstate-step-TB-FA*:

$rstate\ (False,\ A)\ v\ \Longrightarrow\ rstate\ (True,\ B)\ (rletter\text{-}step\ (True,\ B)\ v)$   
 $\langle proof \rangle$

**lemma** *rstate-step-TB-TA*:  
 $rstate (True, A) v \implies rstate (True, B) (rletter-step (True, B) v)$   
 ⟨proof⟩

**lemmas** *rstate-step-cases* =  
*rstate-step-FA-FA* *rstate-step-FA-FB* *rstate-step-FA-TB*  
*rstate-step-TA-TA* *rstate-step-TA-FB* *rstate-step-TA-TB*  
*rstate-step-FB-FB* *rstate-step-FB-FA* *rstate-step-FB-TA*  
*rstate-step-TB-TB* *rstate-step-TB-FA* *rstate-step-TB-TA*

**lemma** *rstate-step*:  
**assumes**  $rstate\ q\ v$  **and**  $\neg\ canceling\ p\ q$   
**shows**  $rstate\ p\ (rletter-step\ p\ v)$   
 ⟨proof⟩

**lemma** *rstate-nonzero*:  
**assumes**  $rstate\ p\ v$   
**shows**  $v \neq rzero-vec$   
 ⟨proof⟩

**lemma** *rword-step-Cons* [simp]:  
 $rword-step\ (p\ \# w)\ v = rletter-step\ p\ (rword-step\ w\ v)$   
 ⟨proof⟩

**lemma** *zword-step-Cons* [simp]:  
 $zword-step\ (p\ \# w)\ v = zletter-step\ p\ (zword-step\ w\ v)$   
 ⟨proof⟩

**lemma** *rword-witness-Cons-nonempty* [simp]:  
**assumes**  $w \neq []$   
**shows**  $rword-witness\ (p\ \# w) = rword-witness\ w$   
 ⟨proof⟩

**lemma** *rword-step-witness-state*:  
**assumes**  $w \in carrier\ F2$  **and**  $w \neq []$   
**shows**  $rstate\ (hd\ w)\ (rword-step\ w\ (rword-witness\ w))$   
 ⟨proof⟩

**definition** *r3-of-int* ::  $int \Rightarrow r3$  **where**  
 $r3-of-int\ n = (if\ n\ mod\ 3 = 0\ then\ R0\ else\ if\ n\ mod\ 3 = 1\ then\ R1\ else\ R2)$

**definition** *rcoef-of-z* ::  $zsqr2 \Rightarrow rcoef$  **where**  
 $rcoef-of-z\ p = (r3-of-int\ (fst\ p),\ r3-of-int\ (snd\ p))$

**definition** *rvec3-of-z* ::  $zvec3 \Rightarrow rvec3$  **where**  
 $rvec3-of-z\ v = (case\ v\ of\ (x,\ y,\ z) \Rightarrow (rcoef-of-z\ x,\ rcoef-of-z\ y,\ rcoef-of-z\ z))$

**lemma** *r3-of-int-add*:  
 $r3-of-int\ (a + b) = r3-add\ (r3-of-int\ a)\ (r3-of-int\ b)$

*<proof>*

**lemma** *r3-of-int-neg*:

$$r3\text{-of-int } (- a) = r3\text{-neg } (r3\text{-of-int } a)$$

*<proof>*

**lemma** *r3-of-int-sub*:

$$r3\text{-of-int } (a - b) = r3\text{-add } (r3\text{-of-int } a) (r3\text{-neg } (r3\text{-of-int } b))$$

*<proof>*

**lemma** *r3-of-int-double*:

$$r3\text{-of-int } (2 * a) = r3\text{-double } (r3\text{-of-int } a)$$

*<proof>*

**lemma** *r3-of-int-triple*:

$$r3\text{-of-int } (3 * a) = R0$$

*<proof>*

**lemma** *rcoef-of-z-simps [simp]*:

$$rcoef\text{-of-z } (zsqr2\text{-add } p q) = rcoef\text{-add } (rcoef\text{-of-z } p) (rcoef\text{-of-z } q)$$

$$rcoef\text{-of-z } (zsqr2\text{-neg } p) = rcoef\text{-neg } (rcoef\text{-of-z } p)$$

$$rcoef\text{-of-z } (zsqr2\text{-sub } p q) = rcoef\text{-sub } (rcoef\text{-of-z } p) (rcoef\text{-of-z } q)$$

$$rcoef\text{-of-z } (zsqr2\text{-scale } 2 p) = rcoef\text{-double } (rcoef\text{-of-z } p)$$

$$rcoef\text{-of-z } (zsqr2\text{-scale } 3 p) = (R0, R0)$$

$$rcoef\text{-of-z } (zsqr2\text{-mul-sqrt2 } p) = rcoef\text{-mul-sqrt2 } (rcoef\text{-of-z } p)$$

*<proof>*

**lemma** *rvec3-of-z-basis [simp]*:

$$rvec3\text{-of-z } ze\text{-}x = re\text{-}x$$

$$rvec3\text{-of-z } ze\text{-}y = re\text{-}y$$

*<proof>*

**lemma** *rstep-of-z-simps [simp]*:

$$rvec3\text{-of-z } (zRx\text{-step } v) = rRx\text{-step } (rvec3\text{-of-z } v)$$

$$rvec3\text{-of-z } (zRx\text{-inv-step } v) = rRx\text{-inv-step } (rvec3\text{-of-z } v)$$

$$rvec3\text{-of-z } (zRz\text{-step } v) = rRz\text{-step } (rvec3\text{-of-z } v)$$

$$rvec3\text{-of-z } (zRz\text{-inv-step } v) = rRz\text{-inv-step } (rvec3\text{-of-z } v)$$

*<proof>*

**lemma** *rletter-step-of-z [simp]*:

$$rvec3\text{-of-z } (zletter\text{-step } p v) = rletter\text{-step } p (rvec3\text{-of-z } v)$$

*<proof>*

**lemma** *rword-step-of-z [simp]*:

$$rvec3\text{-of-z } (zword\text{-step } w v) = rword\text{-step } w (rvec3\text{-of-z } v)$$

*<proof>*

**lemma** *rword-witness-of-z [simp]*:

$$rvec3\text{-of-z } (zword\text{-witness } w) = rword\text{-witness } w$$

*<proof>*

**lemma** *r3-of-int-eq-R0-iff*:  
 $r3\text{-of-int } n = R0 \iff 3 \text{ dvd } n$   
*<proof>*

**lemma** *rcoef-of-z-zero-iff*:  
 $rcoef\text{-of-z } p = (R0, R0) \iff zsqrt2\text{-div3 } p$   
*<proof>*

**lemma** *rvec3-of-z-zero-iff*:  
 $rvec3\text{-of-z } v = rzero\text{-vec} \iff zvec3\text{-div3 } v$   
*<proof>*

**lemma** *zsqr2-val-simps [simp]*:  
 $zsqr2\text{-val } (zsqr2\text{-add } p \ q) = zsqr2\text{-val } p + zsqr2\text{-val } q$   
 $zsqr2\text{-val } (zsqr2\text{-neg } p) = - zsqr2\text{-val } p$   
 $zsqr2\text{-val } (zsqr2\text{-sub } p \ q) = zsqr2\text{-val } p - zsqr2\text{-val } q$   
 $zsqr2\text{-val } (zsqr2\text{-scale } n \ p) = of\text{-int } n * zsqr2\text{-val } p$   
*<proof>*

**lemma** *zsqr2-mul-sqr2-val [simp]*:  
 $zsqr2\text{-val } (zsqr2\text{-mul-sqr2 } p) = \text{sqrt } 2 * zsqr2\text{-val } p$   
*<proof>*

**lemma** *sqr2-prime-irrational-dev*:  
**fixes**  $p :: \text{nat}$   
**assumes**  $\text{prime } p$   
**shows**  $\text{sqrt } p \notin \mathbb{Q}$   
*<proof>*

**lemma** *sqr2-not-rat-dev*:  $\text{sqrt } 2 \notin \mathbb{Q}$   
*<proof>*

**lemma** *zsqr2-val-eq-iff*:  
 $zsqr2\text{-val } p = zsqr2\text{-val } q \iff p = q$   
*<proof>*

**lemma** *zvec3-val-eq-iff*:  
 $zvec3\text{-val } u = zvec3\text{-val } v \iff u = v$   
*<proof>*

**lemma** *zvec3-scale-val [simp]*:  
 $zvec3\text{-val } (zvec3\text{-scale } n \ v) = \text{scaleR } (of\text{-int } n) (zvec3\text{-val } v)$   
*<proof>*

**lemma** *sqr8-eq-2-sqr2*:  $\text{sqrt } 8 = 2 * \text{sqrt } 2$   
*<proof>*

**lemma** *zletter-step-single-mod3*:

$\neg \text{zvec3-div3 } (\text{zletter-step } (\text{False}, A) \text{ ze-y})$   
 $\neg \text{zvec3-div3 } (\text{zletter-step } (\text{True}, A) \text{ ze-y})$   
 $\neg \text{zvec3-div3 } (\text{zletter-step } (\text{False}, B) \text{ ze-x})$   
 $\neg \text{zvec3-div3 } (\text{zletter-step } (\text{True}, B) \text{ ze-x})$   
(proof)

**lemma** *zRx-step-val*:

$\text{zvec3-val } (\text{zRx-step } v) = \text{scaleR } 3 (\text{Rx } (\text{zvec3-val } v))$   
(proof)

**lemma** *zRx-inv-step-val*:

$\text{zvec3-val } (\text{zRx-inv-step } v) = \text{scaleR } 3 (\text{Rx-inv } (\text{zvec3-val } v))$   
(proof)

**lemma** *zRz-step-val*:

$\text{zvec3-val } (\text{zRz-step } v) = \text{scaleR } 3 (\text{Rz } (\text{zvec3-val } v))$   
(proof)

**lemma** *zRz-inv-step-val*:

$\text{zvec3-val } (\text{zRz-inv-step } v) = \text{scaleR } 3 (\text{Rz-inv } (\text{zvec3-val } v))$   
(proof)

**lemma** *zletter-step-val*:

$\text{zvec3-val } (\text{zletter-step } p \ v) = \text{scaleR } 3 (\text{letter-to-rot } p (\text{zvec3-val } v))$   
(proof)

**lemma** *letter-to-rot-linear*: *linear* (*letter-to-rot* *p*)

(proof)

**lemma** *zword-step-val*:

$\text{zvec3-val } (\text{zword-step } w \ v) =$   
 $\text{scaleR } ((3::\text{real}) \wedge \text{length } w) (\text{sigma } w (\text{zvec3-val } v))$   
(proof)

**lemma** *zvec3-scale-power3-div3*:

**assumes**  $n > 0$   
**shows**  $\text{zvec3-div3 } (\text{zvec3-scale } ((3::\text{int}) \wedge n) \ v)$   
(proof)

**lemma** *zword-step-identity-div3*:

**assumes**  $w \neq []$  **and**  $\text{sigma } w = \text{id}$   
**shows**  $\text{zvec3-div3 } (\text{zword-step } w (\text{zword-witness } w))$   
(proof)

**lemma** *zword-step-witness-not-div3-if-nonempty-F2*:

**assumes**  $w \in \text{carrier } F2$  **and**  $w \neq []$   
**shows**  $\neg \text{zvec3-div3 } (\text{zword-step } w (\text{zword-witness } w))$   
(proof)

The word interpretation is an injective homomorphism into  $SO(3)$ .

**theorem** *sigma-homomorphism:*

**assumes**  $w1 \in \text{carrier } F2$   $w2 \in \text{carrier } F2$

**shows**  $\text{sigma } (w1 \otimes_{F2} w2) = \text{sigma } w1 \circ \text{sigma } w2$

*<proof>*

**theorem** *sigma-nontrivial-word-not-id:*

**assumes**  $w \in \text{carrier } F2$  **and**  $w \neq []$

**shows**  $\text{sigma } w \neq \text{id}$

*<proof>*

**lemma** *sigma-injective-if-trivial-kernel:*

**assumes** *kernel:*  $\bigwedge w. w \in \text{carrier } F2 \implies w \neq [] \implies \text{sigma } w \neq \text{id}$

**shows** *inj-on sigma* ( $\text{carrier } F2$ )

*<proof>*

**theorem** *sigma-injective-on-F2:*

*inj-on sigma* ( $\text{carrier } F2$ )

*<proof>*

**theorem** *sigma-image-in-SO3:*

**assumes**  $w \in \text{carrier } F2$

**shows**  $\text{sigma } w \in SO3$

*<proof>*

These three theorems together establish that *sigma* is a group monomorphism from  $F_2$  to  $SO(3)$  – i.e., the image is a copy of  $F_2$  inside  $SO(3)$ . This supplies the free subgroup used in the geometric part of the Banach-Tarski theorem.

**end**

**theory** *Hausdorff-Paradox*

**imports** *Free-Rotations-SO3*

**begin**

## 23 The bad set of fixed points

**definition** *fixed-point-set* ::  $(\text{real}^3 \Rightarrow \text{real}^3) \Rightarrow (\text{real}^3)$  **set** **where**

*fixed-point-set*  $f = \{x \in \text{sphere2}. f x = x\}$

**definition** *bad-set-D* ::  $(\text{real}^3)$  **set** **where**

*bad-set-D* =  $(\bigcup w \in \text{carrier } F2 - \{[]\}. \text{fixed-point-set } (\text{sigma } w))$

**lemma** *carrier-F2-countable:* *countable* ( $\text{carrier } F2$ )

*<proof>*

**lemma** *bad-set-D-index-countable:* *countable* ( $\text{carrier } F2 - \{[]\}$ )

*<proof>*

**lemma** *fixed-point-set-subset-sphere2*: *fixed-point-set*  $f \subseteq$  *sphere2*  
*<proof>*

**lemma** *SO3-fixed-cross*:  
assumes  $f \in SO3$  and  $f x = x$  and  $f y = y$   
shows  $f$  (*cross3*  $x y$ ) = *cross3*  $x y$   
*<proof>*

**lemma** *SO3-fixed-cross-nonzero-imp-id*:  
assumes  $f \in SO3$  and  $f x = x$  and  $f y = y$  and *cross3*  $x y \neq 0$   
shows  $f = id$   
*<proof>*

**lemma** *nonidentity-SO3-fixed-points-cross-zero*:  
assumes  $f \in SO3$  and  $f \neq id$  and  $x \in$  *fixed-point-set*  $f$  and  $y \in$  *fixed-point-set*  $f$   
shows *cross3*  $x y = 0$   
*<proof>*

**lemma** *collinear-sphere2-two-points*:  
assumes  $a \in$  *sphere2* and  $y \in$  *sphere2* and *cross3*  $a y = 0$   
shows  $y = a \vee y = - a$   
*<proof>*

**lemma** *bad-set-D-countable-if-fixed-point-sets-countable*:  
assumes  $\bigwedge w. w \in$  *carrier*  $F2 - \{\emptyset\} \implies$  *countable* (*fixed-point-set* (*sigma*  $w$ ))  
shows *countable* *bad-set-D*  
*<proof>*

**lemma** *nonidentity-SO3-fixed-point-set-countable*:  
assumes  $f \in SO3$  and  $f \neq id$   
shows *countable* (*fixed-point-set*  $f$ )  
*<proof>*

**lemma** *sigma-fixed-point-set-countable*:  
assumes  $w \in$  *carrier*  $F2 - \{\emptyset\}$   
shows *countable* (*fixed-point-set* (*sigma*  $w$ ))  
*<proof>*

**lemma** *bad-set-D-countable*: *countable* *bad-set-D*  
*<proof>*

**lemma** *bad-set-D-subset*: *bad-set-D*  $\subseteq$  *sphere2*  
*<proof>*

**lemma** *sphere2-minus-bad-subset*: *sphere2*  $-$  *bad-set-D*  $\subseteq$  *sphere2*  
*<proof>*

**interpretation** *sigma-sphere-action:*

*group-action carrier F2 [] (λw1 w2. w1 ⊗<sub>F2</sub> w2) λw x. sigma w x sphere2*  
⟨proof⟩

**lemma** *F2-conjugate-nontrivial:*

**assumes**  $w \in \text{carrier } F2$   $v \in \text{carrier } F2$   $v \neq []$   
**shows**  $(\text{inv}_{F2} w \otimes_{F2} v) \otimes_{F2} w \neq []$   
⟨proof⟩

**lemma** *sigma-inverse-left:*

**assumes**  $w \in \text{carrier } F2$   
**shows**  $\text{sigma} (\text{inv}_{F2} w) (\text{sigma } w x) = x$   
⟨proof⟩

**lemma** *sigma-conjugate-fixed-pullback:*

**assumes**  $w \in \text{carrier } F2$   
**and**  $v \in \text{carrier } F2$   
**and**  $\text{sigma } v (\text{sigma } w x) = \text{sigma } w x$   
**shows**  $\text{sigma} ((\text{inv}_{F2} w \otimes_{F2} v) \otimes_{F2} w) x = x$   
⟨proof⟩

**lemma** *sigma-preimage-bad-set:*

**assumes**  $w \in \text{carrier } F2$   
**and**  $x \in \text{sphere2}$   
**and**  $\text{sigma } w x \in \text{bad-set-}D$   
**shows**  $x \in \text{bad-set-}D$   
⟨proof⟩

**lemma** *sigma-preserves-sphere2-minus-bad-set:*

**assumes**  $w \in \text{carrier } F2$   
**and**  $x \in \text{sphere2} - \text{bad-set-}D$   
**shows**  $\text{sigma } w x \in \text{sphere2} - \text{bad-set-}D$   
⟨proof⟩

**interpretation** *sigma-sphere-minus-bad-action:*

*group-action carrier F2 [] (λw1 w2. w1 ⊗<sub>F2</sub> w2) λw x. sigma w x*  
*sphere2 - bad-set-D*  
⟨proof⟩

## 24 Free action of $F_2$ on $S^2 \setminus D$

By construction, every point of  $S^2 \setminus D$  avoids the fixed-point set of every non-trivial group element, hence is moved by every non-trivial  $\text{sigma } w$ .

The action is free away from the fixed-point bad set.

**lemma** *sigma-action-free-off-D:*

**assumes**  $x \in \text{sphere2} - \text{bad-set-}D$

**and**  $w \in \text{carrier } F2$   
**and**  $\text{sigma } w \ x = x$   
**shows**  $w = []$   
 ⟨proof⟩

**lemma** *sigma-sphere-minus-bad-action-free*:  
 $\text{sigma-sphere-minus-bad-action.free-on } (\text{sphere2} - \text{bad-set-}D)$   
 ⟨proof⟩

**lemma** *sigma-orbit-eqI*:  
**assumes**  $x: x \in \text{sphere2} - \text{bad-set-}D$   
**and**  $y: y \in \text{sphere2} - \text{bad-set-}D$   
**and**  $g: g \in \text{carrier } F2$   
**and**  $y\text{-eq}: y = \text{sigma } g \ x$   
**shows**  $\text{sigma-sphere-minus-bad-action.orbit } y =$   
 $\text{sigma-sphere-minus-bad-action.orbit } x$   
 ⟨proof⟩

**lemma** *sigma-orbit-eq-if-common-point*:  
**assumes**  $x: x \in \text{sphere2} - \text{bad-set-}D$   
**and**  $y: y \in \text{sphere2} - \text{bad-set-}D$   
**and**  $z: z \in \text{sigma-sphere-minus-bad-action.orbit } x$   
**and**  $z': z \in \text{sigma-sphere-minus-bad-action.orbit } y$   
**shows**  $\text{sigma-sphere-minus-bad-action.orbit } x =$   
 $\text{sigma-sphere-minus-bad-action.orbit } y$   
 ⟨proof⟩

## 25 Transporting the free-group paradox to $S^2 \setminus D$

**definition** *hausdorff-rep* ::  $\text{real}^3 \Rightarrow \text{real}^3$  **where**  
 $\text{hausdorff-rep } x = (\text{SOME } y. y \in \text{sigma-sphere-minus-bad-action.orbit } x)$

**definition** *hausdorff-lift* ::  $((\text{bool} \times \text{gen2}) \text{ list}) \text{ set} \Rightarrow (\text{real}^3) \text{ set}$  **where**  
 $\text{hausdorff-lift } u =$   
 $\{\text{sigma } g (\text{hausdorff-rep } x) \mid g \ x. g \in u \wedge x \in \text{sphere2} - \text{bad-set-}D\}$

**definition** *hausdorff-sigma-image-set* ::  
 $(\text{bool} \times \text{gen2}) \text{ list} \Rightarrow (\text{real}^3) \text{ set} \Rightarrow (\text{real}^3) \text{ set}$  **where**  
 $\text{hausdorff-sigma-image-set } g \ u = \text{sigma } g \ ` u$

**lemma** *map2-hausdorff-sigma-image-set-eq*:  
 $\text{map2 } \text{sigma-sphere-minus-bad-action.image-set } g s \ A s =$   
 $\text{map2 } \text{hausdorff-sigma-image-set } g s \ A s$   
 ⟨proof⟩

**lemma** *hausdorff-rep-in-orbit*:  
**assumes**  $x \in \text{sphere2} - \text{bad-set-}D$   
**shows**  $\text{hausdorff-rep } x \in \text{sigma-sphere-minus-bad-action.orbit } x$   
 ⟨proof⟩

**lemma** *hausdorff-rep-in-X*:  
**assumes**  $x \in \text{sphere2} - \text{bad-set-D}$   
**shows**  $\text{hausdorff-rep } x \in \text{sphere2} - \text{bad-set-D}$   
 $\langle \text{proof} \rangle$

**lemma** *hausdorff-rep-eq-if-orbit-eq*:  
**assumes**  $\text{sigma-sphere-minus-bad-action.orbit } x =$   
 $\text{sigma-sphere-minus-bad-action.orbit } y$   
**shows**  $\text{hausdorff-rep } x = \text{hausdorff-rep } y$   
 $\langle \text{proof} \rangle$

**lemma** *hausdorff-rep-idem*:  
**assumes**  $x: x \in \text{sphere2} - \text{bad-set-D}$   
**shows**  $\text{hausdorff-rep } (\text{hausdorff-rep } x) = \text{hausdorff-rep } x$   
 $\langle \text{proof} \rangle$

**lemma** *hausdorff-lift-subset-X*:  
**assumes**  $u \subseteq \text{carrier } F2$   
**shows**  $\text{hausdorff-lift } u \subseteq \text{sphere2} - \text{bad-set-D}$   
 $\langle \text{proof} \rangle$

**lemma** *F2-act-image-set-subset-carrier*:  
**assumes**  $g \in \text{carrier } F2$  **and**  $u \subseteq \text{carrier } F2$   
**shows**  $F2\text{-act.image-set } g \ u \subseteq \text{carrier } F2$   
 $\langle \text{proof} \rangle$

**lemma** *sigma-free-same-point-eq*:  
**assumes**  $x: x \in \text{sphere2} - \text{bad-set-D}$   
**and**  $p: p \in \text{carrier } F2$   
**and**  $q: q \in \text{carrier } F2$   
**and**  $\text{eq}: \text{sigma } p \ x = \text{sigma } q \ x$   
**shows**  $p = q$   
 $\langle \text{proof} \rangle$

**lemma** *hausdorff-lift-disjoint*:  
**assumes**  $\text{disj}: u \cap v = \{\}$   
**and**  $u: u \subseteq \text{carrier } F2$   
**and**  $v: v \subseteq \text{carrier } F2$   
**shows**  $\text{hausdorff-lift } u \cap \text{hausdorff-lift } v = \{\}$   
 $\langle \text{proof} \rangle$

**lemma** *hausdorff-lift-pairwise-disjoint*:  
**assumes**  $\text{disj}: \text{pairwise-disjoint } As$   
**and**  $\text{sub}: \forall u \in \text{set } As. u \subseteq \text{carrier } F2$   
**shows**  $\text{pairwise-disjoint } (\text{map } \text{hausdorff-lift } As)$   
 $\langle \text{proof} \rangle$

**lemma** *hausdorff-sigma-image-lift*:

**assumes**  $g: g \in \text{carrier } F2$   
**and**  $u: u \subseteq \text{carrier } F2$   
**shows**  $\text{hausdorff-sigma-image-set } g (\text{hausdorff-lift } u) =$   
 $\text{hausdorff-lift } (F2\text{-act.image-set } g u)$   
 $\langle \text{proof} \rangle$

**lemma** *hausdorff-translated-lift-pairwise-disjoint:*

**assumes**  $\text{len}: \text{length } As = \text{length } gs$   
**and**  $gs: \text{set } gs \subseteq \text{carrier } F2$   
**and**  $\text{sub}: \forall u \in \text{set } As. u \subseteq \text{carrier } F2$   
**and**  $\text{disj}: \text{pairwise-disjoint } (\text{map2 } F2\text{-act.image-set } gs As)$   
**shows**  $\text{pairwise-disjoint } (\text{map2 } \text{hausdorff-sigma-image-set } gs (\text{map } \text{hausdorff-lift } As))$   
 $\langle \text{proof} \rangle$

**lemma** *hausdorff-lift-cover:*

**assumes**  $\text{len}: \text{length } As = \text{length } gs$   
**and**  $gs: \text{set } gs \subseteq \text{carrier } F2$   
**and**  $\text{sub}: \forall u \in \text{set } As. u \subseteq \text{carrier } F2$   
**and**  $\text{cover}: \text{carrier } F2 =$   
 $(\bigcup_{i < \text{length } As} F2\text{-act.image-set } (gs ! i) (As ! i))$   
**shows**  $\text{sphere2} - \text{bad-set-}D =$   
 $(\bigcup_{i < \text{length } As} \text{hausdorff-sigma-image-set } (gs ! i) (\text{hausdorff-lift } (As ! i)))$   
 $\langle \text{proof} \rangle$

## 26 The Hausdorff paradox

Define the rotation-action of  $F_2$  on the sphere by evaluation:  $\text{sigma } w \cdot x = \text{sigma } w x$ . Off the bad set, this is a free action, and the paradoxical decomposition of  $F_2$  transports along orbits.

Hausdorff's theorem:  $S^2 \setminus D$  is paradoxical under the action of  $\text{SO}(3)$ .

**theorem** *hausdorff-paradox-strong:*

$\exists P Q :: (\text{real}^3) \text{ set list. } \exists gP gQ :: ((\text{bool} \times \text{gen2}) \text{ list}) \text{ list.}$   
 $\text{length } P = \text{length } gP \wedge \text{length } Q = \text{length } gQ \wedge$   
 $\text{set } gP \subseteq \text{carrier } F2 \wedge \text{set } gQ \subseteq \text{carrier } F2 \wedge$   
 $\text{set } (\text{map } \text{sigma } gP) \subseteq \text{SO3} \wedge \text{set } (\text{map } \text{sigma } gQ) \subseteq \text{SO3} \wedge$   
 $\text{pairwise-disjoint } (P @ Q) \wedge$   
 $\text{pairwise-disjoint } (\text{map2 } \text{hausdorff-sigma-image-set } gP P) \wedge$   
 $\text{pairwise-disjoint } (\text{map2 } \text{hausdorff-sigma-image-set } gQ Q) \wedge$   
 $(\forall i < \text{length } P. P ! i \subseteq \text{sphere2} - \text{bad-set-}D) \wedge$   
 $(\forall i < \text{length } Q. Q ! i \subseteq \text{sphere2} - \text{bad-set-}D) \wedge$   
 $(\text{sphere2} - \text{bad-set-}D) =$   
 $(\bigcup_{i < \text{length } P} P ! i) \cup (\bigcup_{i < \text{length } Q} Q ! i) \wedge$   
 $(\text{sphere2} - \text{bad-set-}D) =$   
 $(\bigcup_{i < \text{length } P} \text{hausdorff-sigma-image-set } (gP ! i) (P ! i)) \wedge$   
 $(\text{sphere2} - \text{bad-set-}D) =$   
 $(\bigcup_{i < \text{length } Q} \text{hausdorff-sigma-image-set } (gQ ! i) (Q ! i))$

*<proof>*

**theorem** *hausdorff-paradoxical*:

*sigma-sphere-minus-bad-action.paradoxical (sphere2 - bad-set-D)*

*<proof>*

**theorem** *hausdorff-paradox-rot-strong*:

$\exists P Q :: (\text{real}^3) \text{ set list. } \exists gP gQ :: ((\text{real}^3) \Rightarrow (\text{real}^3)) \text{ list.}$

$\text{length } P = \text{length } gP \wedge \text{length } Q = \text{length } gQ \wedge$   
 $\text{set } gP \subseteq SO3 \wedge \text{set } gQ \subseteq SO3 \wedge$   
 $\text{pairwise-disjoint } (P @ Q) \wedge$   
 $\text{pairwise-disjoint } (\text{map2 } (\lambda g A. g ' A) gP P) \wedge$   
 $\text{pairwise-disjoint } (\text{map2 } (\lambda g A. g ' A) gQ Q) \wedge$   
 $(\forall i < \text{length } P. P ! i \subseteq \text{sphere2} - \text{bad-set-D}) \wedge$   
 $(\forall i < \text{length } Q. Q ! i \subseteq \text{sphere2} - \text{bad-set-D}) \wedge$   
 $(\text{sphere2} - \text{bad-set-D}) =$   
 $(\bigcup_{i < \text{length } P} P ! i) \cup (\bigcup_{i < \text{length } Q} Q ! i) \wedge$   
 $(\text{sphere2} - \text{bad-set-D}) =$   
 $(\bigcup_{i < \text{length } P} (gP ! i) ' (P ! i)) \wedge$   
 $(\text{sphere2} - \text{bad-set-D}) =$   
 $(\bigcup_{i < \text{length } Q} (gQ ! i) ' (Q ! i))$

*<proof>*

**theorem** *hausdorff-paradox*:

$\exists P Q :: (\text{real}^3) \text{ set list. } \exists gP gQ :: ((\text{real}^3) \Rightarrow (\text{real}^3)) \text{ list.}$

$\text{length } P = \text{length } gP \wedge \text{length } Q = \text{length } gQ \wedge$   
 $\text{set } gP \subseteq SO3 \wedge \text{set } gQ \subseteq SO3 \wedge$   
 $(\forall i < \text{length } P. P ! i \subseteq \text{sphere2} - \text{bad-set-D}) \wedge$   
 $(\forall i < \text{length } Q. Q ! i \subseteq \text{sphere2} - \text{bad-set-D}) \wedge$   
 $(\text{sphere2} - \text{bad-set-D}) =$   
 $(\bigcup_{i < \text{length } P} (gP ! i) ' (P ! i)) \wedge$   
 $(\text{sphere2} - \text{bad-set-D}) =$   
 $(\bigcup_{i < \text{length } Q} (gQ ! i) ' (Q ! i))$

*<proof>*

**end**

**theory** *Sphere-Decomposition*

**imports** *Hausdorff-Paradox*

**begin**

Equidecomposability of  $S^2 \setminus D$  and  $S^2$ .

**definition** *Rz-angle-mat* ::  $\text{real} \Rightarrow \text{real}^3^3$  **where**

*Rz-angle-mat*  $t = \text{vector } [$   
 $\text{vector } [\cos t, -\sin t, 0],$   
 $\text{vector } [\sin t, \cos t, 0],$   
 $\text{vector } [0, 0, 1]]$

**definition** *Rz-angle* ::  $\text{real} \Rightarrow \text{real}^3 \Rightarrow \text{real}^3$  **where**  
*Rz-angle*  $t\ v = \text{Rz-angle-mat } t * v$

**definition** *e1-3* ::  $\text{real}^3$  **where**  
*e1-3* = *vector* [1, 0, 0]

**definition** *e3-3* ::  $\text{real}^3$  **where**  
*e3-3* = *vector* [0, 0, 1]

**lemma** *Rz-angle-mat-orthogonal*: *orthogonal-matrix* (*Rz-angle-mat*  $t$ )  
 ⟨*proof*⟩

**lemma** *det-Rz-angle-mat* [*simp*]:  $\text{det } (\text{Rz-angle-mat } t) = 1$   
 ⟨*proof*⟩

**lemma** *Rz-angle-in-SO3*:  $\text{Rz-angle } t \in \text{SO3}$   
 ⟨*proof*⟩

**lemma** *Rz-angle-mat-add*:  
*Rz-angle-mat*  $a ** \text{Rz-angle-mat } b = \text{Rz-angle-mat } (a + b)$   
 ⟨*proof*⟩

**lemma** *Rz-angle-compose*:  
*Rz-angle*  $a \circ \text{Rz-angle } b = \text{Rz-angle } (a + b)$   
 ⟨*proof*⟩

**lemma** *Rz-angle-funpow*:  
*Rz-angle*  $t \hat{\sim} n = \text{Rz-angle } (\text{real } n * t)$   
 ⟨*proof*⟩

**lemma** *Rz-angle-components* [*simp*]:  
*Rz-angle*  $t\ x\ \$\ 1 = \cos t * x\ \$\ 1 - \sin t * x\ \$\ 2$   
*Rz-angle*  $t\ x\ \$\ 2 = \sin t * x\ \$\ 1 + \cos t * x\ \$\ 2$   
*Rz-angle*  $t\ x\ \$\ 3 = x\ \$\ 3$   
 ⟨*proof*⟩

**lemma** *Rz-angle-eq-non-axis-sin-cos*:  
**assumes** *Rz-angle*  $a\ x = \text{Rz-angle } b\ x$  **and**  $x\ \$\ 1 \neq 0 \vee x\ \$\ 2 \neq 0$   
**shows**  $\sin a = \sin b \wedge \cos a = \cos b$   
 ⟨*proof*⟩

**lemma** *Rz-angle-collision-angles-countable*:  
**assumes** *nonaxis*:  $x\ \$\ 1 \neq 0 \vee x\ \$\ 2 \neq 0$   
**shows** *countable*  $\{t. \text{Rz-angle } t\ x = y\}$   
 ⟨*proof*⟩

**lemma** *Rz-angle-scaled-collision-angles-countable*:  
**assumes**  $k > 0$  **and** *nonaxis*:  $x\ \$\ 1 \neq 0 \vee x\ \$\ 2 \neq 0$   
**shows** *countable*  $\{t. \text{Rz-angle } (\text{real } k * t)\ x = y\}$

*<proof>*

**lemma** *e1-3-in-sphere2*:  $e1-3 \in \text{sphere2}$

*<proof>*

**lemma** *e1-3-nonaxis*:  $e1-3 \ \$ \ 1 \neq 0 \vee e1-3 \ \$ \ 2 \neq 0$

*<proof>*

**lemma** *e3-3-in-sphere2*:  $e3-3 \in \text{sphere2}$

*<proof>*

**lemma** *z-axis-sphere2-cases*:

**assumes**  $x \in \text{sphere2}$  **and**  $x \ \$ \ 1 = 0$  **and**  $x \ \$ \ 2 = 0$

**shows**  $x = e3-3 \vee x = - e3-3$

*<proof>*

**lemma** *SO3-funpow*:

**assumes**  $R \in SO3$

**shows**  $(R \ \hat{\sim} \ n) \in SO3$

*<proof>*

**lemma** *SO3-inj*:

**assumes**  $f \in SO3$

**shows** *inj*  $f$

*<proof>*

**lemma** *SO3-surj*:

**assumes**  $f \in SO3$

**shows** *surj*  $f$

*<proof>*

**lemma** *SO3-linear*:

**assumes**  $f \in SO3$

**shows** *linear*  $f$

*<proof>*

**lemma** *SO3-inverse*:

**assumes**  $f \in SO3$

**shows** *Hilbert-Choice.inv*  $f \in SO3$

*<proof>*

**lemma** *conjugate-funpow*:

**assumes** *bijS*:  $\text{bij } S$

**shows**  $((S \circ R \circ \text{Hilbert-Choice.inv } S) \ \hat{\sim} \ n) =$   
 $S \circ (R \ \hat{\sim} \ n) \circ \text{Hilbert-Choice.inv } S$

*<proof>*

**lemma** *conjugate-funpow-image*:

**assumes** *bijS*:  $\text{bij } S$

**shows**  $((S \circ R \circ \text{Hilbert-Choice.inv } S) \overset{\sim}{\sim} n) \text{ ' } D =$   
 $S \text{ ' } ((R \overset{\sim}{\sim} n) \text{ ' } (\text{Hilbert-Choice.inv } S \text{ ' } D))$   
 $\langle \text{proof} \rangle$

**lemma** *exists-antipodal-axis-avoiding-countable:*

**assumes** *count-D: countable D*

**shows**  $\exists a \in \text{sphere2}. a \notin D \wedge -a \notin D$

$\langle \text{proof} \rangle$

**lemma** *exists-Rz-angle-absorbing:*

**assumes** *count-D: countable D*

**and** *nonaxis:  $\bigwedge x. x \in D \implies x \$ 1 \neq 0 \vee x \$ 2 \neq 0$*

**shows**  $\exists t. \forall n \ m. n \neq m \longrightarrow ((Rz\text{-angle } t \overset{\sim}{\sim} n) \text{ ' } D) \cap ((Rz\text{-angle } t \overset{\sim}{\sim} m) \text{ ' } D)$   
 $= \{\}$

$\langle \text{proof} \rangle$

**lemma** *exists-absorbing-rotation:*

$\exists R \in SO3. \forall n \ m. n \neq m \longrightarrow ((R \overset{\sim}{\sim} n) \text{ ' } \text{bad-set-D}) \cap ((R \overset{\sim}{\sim} m) \text{ ' } \text{bad-set-D}) =$   
 $\{\}$

$\langle \text{proof} \rangle$

**lemma** *SO3-funpow-preserves-sphere2:*

**assumes**  $R \in SO3$  **and**  $x \in \text{sphere2}$

**shows**  $(R \overset{\sim}{\sim} n) x \in \text{sphere2}$

$\langle \text{proof} \rangle$

**lemma** *funpow-Suc-image:*

$((R \overset{\sim}{\sim} \text{Suc } n) \text{ ' } S) = R \text{ ' } ((R \overset{\sim}{\sim} n) \text{ ' } S)$

$\langle \text{proof} \rangle$

**lemma** *absorbing-rotation-shift:*

**assumes** *disj:  $\forall n \ m. n \neq m \longrightarrow ((R \overset{\sim}{\sim} n) \text{ ' } D) \cap ((R \overset{\sim}{\sim} m) \text{ ' } D) = \{\}$*

**defines**  $E \equiv (\bigcup n. (R \overset{\sim}{\sim} n) \text{ ' } D)$

**shows**  $R \text{ ' } E = E - D$

$\langle \text{proof} \rangle$

**lemma** *sphere2-absorb-bad-set:*

$\exists P \ Q :: (\text{real}^3) \text{ set list}. \exists gs :: ((\text{real}^3) \Rightarrow (\text{real}^3)) \text{ list}.$

$\text{length } P = \text{length } Q \wedge \text{length } P = \text{length } gs \wedge$

$\text{set } gs \subseteq SO3 \wedge$

$\text{sphere2} = (\bigcup i < \text{length } P. P ! i) \wedge$

$(\text{sphere2} - \text{bad-set-D}) = (\bigcup i < \text{length } Q. Q ! i) \wedge$

$(\forall i < \text{length } P. Q ! i = (gs ! i) \text{ ' } (P ! i))$

$\langle \text{proof} \rangle$

Combining the Hausdorff paradox with the absorption argument yields the paradoxical decomposition of the full sphere.

**lemma** *sphere-indexed-union-append-singleton:*

**fixes**  $P :: 'a \text{ set list}$  **and**  $S :: 'a \text{ set}$

shows  $(\bigcup_{i < \text{length } P} (P @ [S]). (P @ [S]) ! i) =$   
 $(\bigcup_{i < \text{length } P} P ! i) \cup S$   
 ⟨proof⟩

**lemma** *sphere-indexed-image-union-append-singleton*:  
 fixes  $P :: 'a \text{ set list}$  and  $G :: ('a \Rightarrow 'b) \text{ list}$  and  $S :: 'a \text{ set}$   
 and  $g :: 'a \Rightarrow 'b$   
 assumes  $\text{length } P = \text{length } G$   
 shows  $(\bigcup_{i < \text{length } P} (P @ [S]). (G @ [g]) ! i \text{ ' } ((P @ [S]) ! i)) =$   
 $(\bigcup_{i < \text{length } P} G ! i \text{ ' } (P ! i)) \cup g \text{ ' } S$   
 ⟨proof⟩

**lemma** *sphere2-absorb-cover*:  
 assumes  $R\text{-}SO3: R \in SO3$   
 and  $\text{disj}: \forall n m. n \neq m \longrightarrow ((R \hat{\sim} n) \text{ ' } \text{bad-set-}D) \cap ((R \hat{\sim} m) \text{ ' } \text{bad-set-}D) =$   
 {}  
 and  $\text{len}: \text{length } P = \text{length } g$   
 and  $g\text{-}SO3: \text{set } g \subseteq SO3$   
 and  $P\text{-sub}: \forall i < \text{length } P. P ! i \subseteq \text{sphere2} - \text{bad-set-}D$   
 and  $X\text{-cover}: \text{sphere2} - \text{bad-set-}D = (\bigcup_{i < \text{length } P} (g ! i) \text{ ' } (P ! i))$   
 shows  $\exists P' :: (\text{real}^3) \text{ set list}. \exists g' :: ((\text{real}^3) \Rightarrow (\text{real}^3)) \text{ list}.$   
 $\text{length } P' = \text{length } g' \wedge$   
 $\text{set } g' \subseteq SO3 \wedge$   
 $(\forall i < \text{length } P'. P' ! i \subseteq \text{sphere2}) \wedge$   
 $\text{sphere2} = (\bigcup_{i < \text{length } P'. P' ! i) \wedge$   
 $\text{sphere2} = (\bigcup_{i < \text{length } P'. (g' ! i) \text{ ' } (P' ! i))$   
 ⟨proof⟩

**theorem** *sphere2-paradoxical-strong*:  
 $\exists P Q :: (\text{real}^3) \text{ set list}. \exists gP gQ :: ((\text{real}^3) \Rightarrow (\text{real}^3)) \text{ list}.$   
 $\text{length } P = \text{length } gP \wedge \text{length } Q = \text{length } gQ \wedge$   
 $\text{set } gP \subseteq SO3 \wedge \text{set } gQ \subseteq SO3 \wedge$   
 $\text{pairwise-disjoint } (P @ Q) \wedge$   
 $\text{pairwise-disjoint } (\text{map2 } (\lambda g A. g \text{ ' } A) gP P) \wedge$   
 $\text{pairwise-disjoint } (\text{map2 } (\lambda g A. g \text{ ' } A) gQ Q) \wedge$   
 $(\forall i < \text{length } P. P ! i \subseteq \text{sphere2}) \wedge$   
 $(\forall i < \text{length } Q. Q ! i \subseteq \text{sphere2}) \wedge$   
 $\text{sphere2} = (\bigcup_{i < \text{length } P} P ! i) \cup (\bigcup_{i < \text{length } Q} Q ! i) \wedge$   
 $\text{sphere2} = (\bigcup_{i < \text{length } P} (gP ! i) \text{ ' } (P ! i)) \wedge$   
 $\text{sphere2} = (\bigcup_{i < \text{length } Q} (gQ ! i) \text{ ' } (Q ! i))$   
 ⟨proof⟩

**theorem** *sphere2-paradoxical*:  
 $\exists P Q :: (\text{real}^3) \text{ set list}. \exists gP gQ :: ((\text{real}^3) \Rightarrow (\text{real}^3)) \text{ list}.$   
 $\text{length } P = \text{length } gP \wedge \text{length } Q = \text{length } gQ \wedge$   
 $\text{set } gP \subseteq SO3 \wedge \text{set } gQ \subseteq SO3 \wedge$   
 $(\forall i < \text{length } P. P ! i \subseteq \text{sphere2}) \wedge$   
 $(\forall i < \text{length } Q. Q ! i \subseteq \text{sphere2}) \wedge$   
 $\text{sphere2} = (\bigcup_{i < \text{length } P} P ! i) \wedge$

$$\begin{aligned} \text{sphere2} &= (\bigcup_{i < \text{length } Q}. Q ! i) \wedge \\ \text{sphere2} &= (\bigcup_{i < \text{length } P}. (gP ! i) \text{ ' } (P ! i)) \wedge \\ \text{sphere2} &= (\bigcup_{i < \text{length } Q}. (gQ ! i) \text{ ' } (Q ! i)) \end{aligned}$$

*<proof>*

**end**

**theory** *Ball-Decomposition*  
**imports** *Sphere-Decomposition*  
**begin**

Radial cone construction: take a sphere subset and form the open radial cone in the ball.

**definition** *cone-of* ::  $(\text{real}^3)$  set  $\Rightarrow$   $(\text{real}^3)$  set **where**  
*cone-of*  $S = \{t *_R x \mid t x. 0 < t \wedge t \leq 1 \wedge x \in S\}$

**lemma** *cone-of-in-ball3*:  
**assumes**  $S \subseteq \text{sphere2}$   
**shows**  $\text{cone-of } S \subseteq \text{ball3}$   
*<proof>*

**lemma** *cone-of-in-ball3-minus-origin*:  
**assumes**  $S \subseteq \text{sphere2}$   
**shows**  $\text{cone-of } S \subseteq \text{ball3} - \{0\}$   
*<proof>*

**lemma** *ball3-minus-origin-eq-cone-sphere2*:  
 $\text{ball3} - \{0::\text{real}^3\} = \text{cone-of } \text{sphere2}$   
*<proof>*

**lemma** *cone-of-UN*:  
 $\text{cone-of } (\bigcup_{i \in I}. S i) = (\bigcup_{i \in I}. \text{cone-of } (S i))$   
*<proof>*

**lemma** *SO3-image-cone-of*:  
**assumes**  $g \in \text{SO3}$   
**shows**  $g \text{ ' } \text{cone-of } S = \text{cone-of } (g \text{ ' } S)$   
*<proof>*

**lemma** *cone-of-disjoint*:  
**assumes** *S-sub*:  $S \subseteq \text{sphere2}$   
**and** *T-sub*:  $T \subseteq \text{sphere2}$   
**and** *disj*:  $S \cap T = \{\}$   
**shows**  $\text{cone-of } S \cap \text{cone-of } T = \{\}$   
*<proof>*

**lemma** *cone-of-pairwise-disjoint*:  
**assumes** *disj*: *pairwise-disjoint*  $P$

**and** *sub*:  $\forall i < \text{length } P. P ! i \subseteq \text{sphere2}$   
**shows** *pairwise-disjoint* (*map cone-of* *P*)  
*<proof>*

**lemma** *SO3-cone-images-pairwise-disjoint*:  
**assumes** *len*:  $\text{length } P = \text{length } g$   
**and** *g-SO3*:  $\text{set } g \subseteq \text{SO3}$   
**and** *P-sub*:  $\forall i < \text{length } P. P ! i \subseteq \text{sphere2}$   
**and** *disj*: *pairwise-disjoint* (*map2* ( $\lambda h A. h \text{ ' } A$ ) *g* *P*)  
**shows** *pairwise-disjoint* (*map2* ( $\lambda h A. h \text{ ' } A$ ) *g* (*map cone-of* *P*))  
*<proof>*

**lemma** *zero-in-ball3 [simp]*:  $(0::\text{real}^3) \in \text{ball3}$   
*<proof>*

**lemma** *indexed-image-union-append-singleton*:  
**fixes** *P* :: 'a set list **and** *G* :: ('a  $\Rightarrow$  'b) list **and** *S* :: 'a set  
**and** *g* :: 'a  $\Rightarrow$  'b  
**assumes**  $\text{length } P = \text{length } G$   
**shows**  $(\bigcup i < \text{length } (P @ [S]). (G @ [g]) ! i \text{ ' } ((P @ [S]) ! i)) =$   
 $(\bigcup i < \text{length } P. G ! i \text{ ' } (P ! i)) \cup g \text{ ' } S$   
*<proof>*

**lemma** *indexed-union-append-singleton*:  
**fixes** *P* :: 'a set list **and** *S* :: 'a set  
**shows**  $(\bigcup i < \text{length } (P @ [S]). (P @ [S]) ! i) =$   
 $(\bigcup i < \text{length } P. P ! i) \cup S$   
*<proof>*

The punctured ball is paradoxical via radial cones over the sphere pieces.

**theorem** *ball3-minus-origin-paradoxical*:  
 $\exists P Q :: (\text{real}^3) \text{ set list. } \exists gP gQ :: ((\text{real}^3) \Rightarrow (\text{real}^3)) \text{ list.}$   
 $\text{length } P = \text{length } gP \wedge \text{length } Q = \text{length } gQ \wedge$   
 $\text{set } gP \subseteq \text{SO3} \wedge \text{set } gQ \subseteq \text{SO3} \wedge$   
 $(\forall i < \text{length } P. P ! i \subseteq \text{ball3} - \{0\}) \wedge$   
 $(\forall i < \text{length } Q. Q ! i \subseteq \text{ball3} - \{0\}) \wedge$   
 $\text{ball3} - \{0\} = (\bigcup i < \text{length } P. P ! i) \wedge$   
 $\text{ball3} - \{0\} = (\bigcup i < \text{length } Q. Q ! i) \wedge$   
 $\text{ball3} - \{0\} = (\bigcup i < \text{length } P. (gP ! i) \text{ ' } (P ! i)) \wedge$   
 $\text{ball3} - \{0\} = (\bigcup i < \text{length } Q. (gQ ! i) \text{ ' } (Q ! i))$   
*<proof>*

**theorem** *ball3-minus-origin-paradoxical-strong*:  
 $\exists P Q :: (\text{real}^3) \text{ set list. } \exists gP gQ :: ((\text{real}^3) \Rightarrow (\text{real}^3)) \text{ list.}$   
 $\text{length } P = \text{length } gP \wedge \text{length } Q = \text{length } gQ \wedge$   
 $\text{set } gP \subseteq \text{SO3} \wedge \text{set } gQ \subseteq \text{SO3} \wedge$   
 $\text{pairwise-disjoint } (P @ Q) \wedge$   
 $\text{pairwise-disjoint } (\text{map2 } (\lambda g A. g \text{ ' } A) gP P) \wedge$   
 $\text{pairwise-disjoint } (\text{map2 } (\lambda g A. g \text{ ' } A) gQ Q) \wedge$

$$\begin{aligned}
& (\forall i < \text{length } P. P ! i \subseteq \text{ball3} - \{0\}) \wedge \\
& (\forall i < \text{length } Q. Q ! i \subseteq \text{ball3} - \{0\}) \wedge \\
& \text{ball3} - \{0\} = (\bigcup_{i < \text{length } P} P ! i) \cup (\bigcup_{i < \text{length } Q} Q ! i) \wedge \\
& \text{ball3} - \{0\} = (\bigcup_{i < \text{length } P} (gP ! i) ' (P ! i)) \wedge \\
& \text{ball3} - \{0\} = (\bigcup_{i < \text{length } Q} (gQ ! i) ' (Q ! i))
\end{aligned}$$

*<proof>*

Adjoining the origin gives a useful  $SO(3)$ -only finite-cover decomposition of the full ball. The partition-level absorption is completed in the final theory, where off-centre isometries are available.

**theorem** *ball3-paradoxical:*

$$\begin{aligned}
& \exists P Q :: (\text{real}^3) \text{ set list. } \exists gP gQ :: ((\text{real}^3) \Rightarrow (\text{real}^3)) \text{ list.} \\
& \text{length } P = \text{length } gP \wedge \text{length } Q = \text{length } gQ \wedge \\
& \text{set } gP \subseteq SO3 \wedge \text{set } gQ \subseteq SO3 \wedge \\
& (\forall i < \text{length } P. P ! i \subseteq \text{ball3}) \wedge \\
& (\forall i < \text{length } Q. Q ! i \subseteq \text{ball3}) \wedge \\
& \text{ball3} = (\bigcup_{i < \text{length } P} P ! i) \wedge \\
& \text{ball3} = (\bigcup_{i < \text{length } Q} Q ! i) \wedge \\
& \text{ball3} = (\bigcup_{i < \text{length } P} (gP ! i) ' (P ! i)) \wedge \\
& \text{ball3} = (\bigcup_{i < \text{length } Q} (gQ ! i) ' (Q ! i))
\end{aligned}$$

*<proof>*

**end**

**theory** *Banach-Tarski-Theorem*

**imports** *Ball-Decomposition*

**begin**

Two copies of the ball, embedded in disjoint regions of  $\mathbb{R}^3$ .

**definition** *shift3* ::  $\text{real}^3$  **where**

$$\text{shift3} = \text{vector } [3, 0, 0]$$

**definition** *ball3-copy-left* ::  $(\text{real}^3)$  **set where**

$$\text{ball3-copy-left} = \text{ball3}$$

**definition** *ball3-copy-right* ::  $(\text{real}^3)$  **set where**

$$\text{ball3-copy-right} = (+) \text{shift3} ' \text{ball3}$$

**lemma** *norm-shift3* [*simp*]:  $\text{norm shift3} = 3$

*<proof>*

**lemma** *ball3-copies-disjoint:*

$$\text{ball3-copy-left} \cap \text{ball3-copy-right} = \{\}$$

*<proof>*

Isometries of  $\mathbb{R}^3$  are the motions used in the final assembly. Rotations from  $SO(3)$ , translations, and their compositions are isometries.

**definition** *isometry* ::  $(\text{real}^3 \Rightarrow \text{real}^3) \Rightarrow \text{bool}$  **where**

$isometry\ f \longleftrightarrow (\forall x\ y. dist\ (f\ x)\ (f\ y) = dist\ x\ y)$

**lemma** *id-isometry* [*simp*]: *isometry id*  
*<proof>*

**lemma** *translation-isometry* [*simp*]: *isometry ((+) a)*  
*<proof>*

**lemma** *SO3-isometry*:  
**assumes**  $g \in SO3$   
**shows** *isometry g*  
*<proof>*

**lemma** *isometry-compose*:  
**assumes** *isometry f isometry g*  
**shows** *isometry (f o g)*  
*<proof>*

**lemma** *translation-compose-SO3-isometry* [*simp*]:  
**assumes**  $g \in SO3$   
**shows** *isometry ((+) a o g)*  
*<proof>*

**lemma** *offcenter-rotation-isometry*:  
**assumes**  $R \in SO3$   
**shows** *isometry ( $\lambda x. c + R (x - c)$ )*  
*<proof>*

**lemma** *offcenter-rotation-inverse-left*:  
**fixes**  $c :: real^3$   
**assumes** *inv-left: Rinv o R = id*  
**defines**  $T \equiv \lambda x. c + R (x - c)$   
**and**  $U \equiv \lambda x. c + Rinv (x - c)$   
**shows**  $U (T\ x) = x$   
*<proof>*

**lemma** *offcenter-rotation-inverse-right*:  
**fixes**  $c :: real^3$   
**assumes** *inv-right: R o Rinv = id*  
**defines**  $T \equiv \lambda x. c + R (x - c)$   
**and**  $U \equiv \lambda x. c + Rinv (x - c)$   
**shows**  $T (U\ x) = x$   
*<proof>*

**lemma** *offcenter-rotation-funpow-zero*:  
**fixes**  $c :: real^3$  **and**  $R :: real^3 \Rightarrow real^3$   
**defines**  $T \equiv \lambda x. c + R (x - c)$   
**shows**  $(T \hat{\ } n)\ 0 = c + (R \hat{\ } n)\ (-c)$   
*<proof>*

**lemma** *sigma-replicate-b*:

$$\text{sigma } (\text{replicate } n \text{ (False, B)}) = (\text{Rz } \sim n)$$

*<proof>*

**lemma** *canceled-replicate-b*: *canceled* (*replicate* *n* (*False*, *B*))

*<proof>*

**lemma** *replicate-b-in-F2*: *replicate* *n* (*False*, *B*)  $\in$  *carrier F2*

*<proof>*

**theorem** *ball3-paradoxical-strong*:

$$\begin{aligned} \exists P Q :: (\text{real}^3) \text{ set list. } \exists gP gQ :: ((\text{real}^3) \Rightarrow (\text{real}^3)) \text{ list.} \\ \text{length } P = \text{length } gP \wedge \text{length } Q = \text{length } gQ \wedge \\ (\forall g \in \text{set } gP. \text{isometry } g) \wedge (\forall g \in \text{set } gQ. \text{isometry } g) \wedge \\ \text{pairwise-disjoint } (P @ Q) \wedge \\ \text{pairwise-disjoint } (\text{map2 } (\lambda g A. g \text{ ` } A) gP P) \wedge \\ \text{pairwise-disjoint } (\text{map2 } (\lambda g A. g \text{ ` } A) gQ Q) \wedge \\ (\forall i < \text{length } P. P ! i \subseteq \text{ball3}) \wedge \\ (\forall i < \text{length } Q. Q ! i \subseteq \text{ball3}) \wedge \\ \text{ball3} = (\bigcup i < \text{length } P. P ! i) \cup (\bigcup i < \text{length } Q. Q ! i) \wedge \\ \text{ball3} = (\bigcup i < \text{length } P. (gP ! i) \text{ ` } (P ! i)) \wedge \\ \text{ball3} = (\bigcup i < \text{length } Q. (gQ ! i) \text{ ` } (Q ! i)) \end{aligned}$$

*<proof>*

An auxiliary finite-cover theorem for the closed unit ball.

**theorem** *banach-tarski-finite-cover*:

$$\begin{aligned} \exists P :: (\text{real}^3) \text{ set list. } \exists gs :: ((\text{real}^3) \Rightarrow (\text{real}^3)) \text{ list.} \\ \text{length } P = \text{length } gs \wedge \\ (\forall g \in \text{set } gs. \text{isometry } g) \wedge \\ (\forall i < \text{length } P. P ! i \subseteq \text{ball3}) \wedge \\ \text{ball3} = (\bigcup i < \text{length } P. P ! i) \wedge \\ (\text{ball3-copy-left} \cup \text{ball3-copy-right}) = \\ (\bigcup i < \text{length } P. (gs ! i) \text{ ` } (P ! i)) \wedge \\ \text{ball3-copy-left} \cap \text{ball3-copy-right} = \{\} \end{aligned}$$

*<proof>*

The Banach–Tarski theorem for the closed unit ball, as a partition theorem.

**theorem** *banach-tarski*:

$$\begin{aligned} \exists P :: (\text{real}^3) \text{ set list. } \exists gs :: ((\text{real}^3) \Rightarrow (\text{real}^3)) \text{ list.} \\ \text{length } P = \text{length } gs \wedge \\ (\forall g \in \text{set } gs. \text{isometry } g) \wedge \\ \text{pairwise-disjoint } P \wedge \\ \text{pairwise-disjoint } (\text{map2 } (\lambda g A. g \text{ ` } A) gs P) \wedge \\ (\forall i < \text{length } P. P ! i \subseteq \text{ball3}) \wedge \\ \text{ball3} = (\bigcup i < \text{length } P. P ! i) \wedge \\ (\text{ball3-copy-left} \cup \text{ball3-copy-right}) = \\ (\bigcup i < \text{length } P. (gs ! i) \text{ ` } (P ! i)) \wedge \\ \text{ball3-copy-left} \cap \text{ball3-copy-right} = \{\} \end{aligned}$$

⟨proof⟩

end

## References

- [1] S. Banach and A. Tarski. Sur la décomposition des ensembles de points en parties respectivement congruentes. *Fundamenta Mathematicae*, 6(1):244–277, 1924. DOI: <https://doi.org/10.4064/fm-6-1-244-277>. Also available at <https://eudml.org/doc/214280>.
- [2] J. Breitner. Free groups. *Archive of Formal Proofs*, June 2010. <https://isa-afp.org/entries/Free-Groups.html>, Formal proof development.
- [3] D. de Rauglaudre. Formal proof of Banach–Tarski paradox. *Journal of Formalized Reasoning*, 10(1):37–49, 2017. DOI: <https://doi.org/10.6092/issn.1972-5787/6927>.
- [4] F. Hausdorff. Bemerkung über den Inhalt von Punktmengen. *Mathematische Annalen*, 75:428–433, 1914. DOI: <https://doi.org/10.1007/BF01563735>. Also available at <https://eudml.org/doc/158671>.
- [5] T. Tao. *An Introduction to Measure Theory*, volume 126 of *Graduate Studies in Mathematics*. American Mathematical Society, 2011. DOI: <https://doi.org/10.1090/gsm/126>.
- [6] G. Tomkowicz and S. Wagon. *The Banach–Tarski Paradox*, volume 163 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 2nd edition, 2016. DOI: <https://doi.org/10.1017/CBO9781107337145>.
- [7] S. Wagon. *The Banach–Tarski Paradox*, volume 24 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 1993. First paperback edition. DOI: <https://doi.org/10.1017/CBO9780511609596>.