

```

/*
 * complex_length.c
 *
 * This file provides the functions
 *
 *     Complex complex_length_mt(MoebiusTransformation *mt);
 *     Complex complex_length_o3l(O3lMatrix *m);
 *
 * They are identical except that in one case the isometry is specified
 * by a MoebiusTransformation, and in the other by an O3lMatrix.
 *
 * If the isometry is orientation-preserving, then complex_length_*()
 * returns a complex (length + i torsion) with the usual interpretation:
 *
 *     If neither length nor torsion is zero,
 *         the isometry is loxodromic.  It's a screw motion with the given
 *         length and torsion.
 *
 *     If the torsion is zero,
 *         the isometry is hyperbolic.  It's a translation along a
 *         geodesic.  The translation distance is the reported length.
 *
 *     If the length is zero,
 *         the isometry is a rotation about an axis.  The torsion gives
 *         the rotation angle.
 *
 *     If both length and torsion are zero,
 *         the isometry is a parabolic (perhaps the identity).
 *
 * If the isometry is orientation-reversing, then we use the definition
 *
 *     Definition.  An orientation-reversing isometry of  $H^3$  is
 *     elliptic, parabolic or hyperbolic iff it factors as a
 *     reflection in a plane, followed by an orientation-preserving
 *     elliptic, parabolic or hyperbolic isometry, respectively,
 *     which fixes that plane (setwise).
 *
 * The section "Interpreting the trace" (cf. below) explains this
 * definition and proves the necessary supporting lemmas.  For now,
 * note that orientation-reversing loxodromics do not exist.  So a
 * single real parameter suffices to fully describe an orientation-
 * reversing isometry.  complex_length_*() returns a nonzero length or
 * a nonzero torsion (but not both) with the following interpretations:
 *
 *     If the torsion is zero,
 *         the isometry is hyperbolic.  It's a glide reflection along a
 *         geodesic.  The translation distance is the reported length.
 *
 *     If the length is zero,
 *         the isometry is a reflection in a plane followed by a rotation
 *         about an axis orthogonal to that plane.  The torsion gives
 *         the rotation angle.
 *
 *     If both length and torsion are zero,
 *         the isometry is a reflection in a plane, followed by a
 *         parabolic (perhaps the identity) which fixes that plane setwise.
 *
 * I recommend that when the torsion is zero, the user interface not
 * display it to the user.  For example, the presence or absence of a
 * torsion value is a good easy way to see which geodesics are
 * orientation_preserving and which are orientation_reversing.
 *
 * Classification of isometries.
 *
 * We must analyze what kinds of isometries are possible in hyperbolic
 * space.  In the orientable case the answers are part of the standard
 * lore of hyperbolic geometry, but standard textbooks omit the
 * nonorientable case.  So here we'll go ahead and do the orientable and
 * nonorientable cases both.
 *
 * Let  $g$  be an isometry of  $H^3$ .
 *
 * Case 1.   $g$  has a fixed point  $x$  in  $H^3$ .

```

```

*
*   g maps a unit sphere centered at x to itself.
*
*   Case 1.1.  g is orientation-preserving.
*
*       It's a standard theorem of geometry that an orientation-
*       preserving isometry from a 2-sphere to itself is a rotation
*       about some axis.
*
*       1.1.1.  In the upper half space model we may choose coordinates
*       so that g has the matrix
*
*           exp(i*theta/2)      0
*           0      exp(-i*theta/2)
*
*       The trace is  $\exp(i*theta/2) + \exp(-i*theta/2) = 2 \cos(theta/2)$ .
*
*       1.1.2.  In the Minkowski space model we may choose coordinates
*       so that g has the matrix
*
*           1      0      0      0
*           0      1      0      0
*           0      0      cos(theta)  -sin(theta)
*           0      0      sin(theta)  cos(theta)
*
*       The trace is  $2 + 2 \cos(theta)$ .
*
*   Case 1.2.  g is orientation-reversing.
*
*       Let h be reflection through x.  Then  $g = g(hh) = (gh)h$ .
*       The map gh is orientation preserving, so must have a matrix
*       as in 1.1.2.  The coordinate system of 1.1.2 puts the fixed
*       point x at the origin (1, 0, 0, 0), so h has matrix
*
*           1      0      0      0
*           0      -1     0      0
*           0      0      -1     0
*           0      0      0      -1
*
*       and therefore  $g = (gh)h$  has matrix
*
*           1      0      0      0
*           0      -1     0      0
*           0      0      -cos(phi)  sin(phi)
*           0      0      -sin(phi)  -cos(phi)
*
*       The trace is  $-2 \cos(phi)$ .
*
*       For most purposes it's more convenient to factor g as  $(gh')h'$ ,
*       where h' is a reflection in the plane through x orthogonal
*       to the rotation axis.  The rotation gh' has the same axis as gh,
*       but its rotation angle theta is phi + pi.  Thus gh' has matrix
*
*           1      0      0      0
*           0      -1     0      0
*           0      0      cos(theta)  -sin(theta)
*           0      0      sin(theta)  cos(theta)
*
*       The trace is  $2 \cos(theta)$ .
*
*   Case 2.  g has no fixed point in  $H^3$ .
*
*       By the Brouwer fixed point theorem, g must have at least one
*       fixed point on the sphere at infinity.
*
*   Case 2.1.  g has at least three fixed points on the sphere at infinity.
*
*       g fixes the ideal triangle spanned by the three points.  Ideal
*       triangles are rigid, so it follows that g fixes the entire plane
*       spanned by the three points.  Either g is the identity with
*       matrix
*
*           1  0  0  0
*           0  1  0  0
*           0  0  1  0

```

```

*           0  0  0  1
*
* and trace 4, or g is a reflection in the fixed plane, with
* matrix conjugate to
*
*           1  0  0  0
*           0  1  0  0
*           0  0  1  0
*           0  0  0 -1
*
* and trace 2.
*
* Case 2.2. g has exactly two fixed points on the sphere at infinity.
*
* Let L be the line spanned by the two fixed points. g fixes
* L as a set, but not pointwise (because throughout case 2 we
* are assuming g has no fixed point in  $H^3$ ). Similarly, g must
* preserve the direction of L, since otherwise there'd be a
* fixed point somewhere on L.
*
* Case 2.2.1. g is orientation-preserving.
*
* g is a translation along L combined with a rotation about L.
*
* 2.2.1.1. If we choose coordinates in the upper half space
* model so that one fixed point is at 0 and the other at
* infinity, then  $g(z) = kz$ , the matrix of g is
*
*           sqrt(k)    0
*           0    1/sqrt(k)
*
* and the trace is  $\sqrt{k} + 1/\sqrt{k}$ . Here it's more
* convenient to work with the square of the trace, which
* is  $k + 2 + 1/k$ .
*
* 2.2.1.2. In the Minkowski space model we may choose
* coordinates so that g has matrix
*
*           cosh s  sinh s    0    0
*           sinh s  cosh s    0    0
*           0       0       cos t  -sin t
*           0       0       sin t   cos t
*
* and trace  $2 \cosh s + 2 \cos t$ . Note that it's not so easy
* to analyze the trace in this case. If we computed the
* characteristic polynomial we could factor it to obtain
*  $|s|$  and  $|t|$ , but this won't be necessary because in the
* orientation-preserving case we'll stick to  $PSL(2, \mathbb{C})$  where
* we can also obtain the sign (handedness) of the twist.
*
* Case 2.2.2. g is orientation-reversing.
*
* g is a translation along L combined with a reflection
* in a plane through L.
*
* In the Minkowski space model we may choose coordinates
* so that g has matrix
*
*           cosh s  sinh s    0    0
*           sinh s  cosh s    0    0
*           0       0       -1    0
*           0       0        0    1
*
* with trace  $2 \cosh s$ .
*
* Case 2.3. g has exactly one fixed point on the sphere at infinity.
*
* Let x be the fixed point on the sphere at infinity, and let
* H be any horosphere centered at x.
*
* Lemma. g maps H to itself.
*
* Proof. Since x is a fixed point, we know g must send H to
* some horosphere H' centered at x. Let p be the map which

```

projects H' onto H radially from x . The restriction of g to H is an isometry from H to H' , but if $H' \neq H$ then the projection p from H' back to H is a similarity which expands distances by some factor $r \neq 1$. Let f be p composed with the restriction of g to H , and let y be an arbitrary point on H . I claim f has a fixed point. If $r < 1$, consider the sequence of points $\{y, f(y), f(f(y)), \dots\}$. The distances between consecutive points form a convergent geometric series, so the points themselves converge to a fixed point z . If $r > 1$, the same argument applies using f^{-1} instead of f . The (ideal) point x and the (finite) point z determine a geodesic which is fixed (setwise, not pointwise) by g . The geodesic's other endpoint is therefore a second fixed point on the sphere at infinity, contradicting the Case 2.3 assumption that g has a unique fixed point on the sphere at infinity. Therefore our assumption that $H' \neq H$ must have been wrong. Q.E.D.

In view of the preceding Lemma, it suffices to analyze the action of g on a horosphere H centered at x .

Case 2.3.1. g is orientation-preserving.

Lemma. $g|_H$ is a pure translation.

Proof. If $g|_H$ had a rotational component, then for any point y on H , the points $\{y, g(y), g(g(y))\}$ would not be colinear (g couldn't map the vector from y to $g(y)$ to a parallel vector). The perpendicular bisectors of the segments from y to $g(y)$ and from $g(y)$ to $g(g(y))$ would intersect at a fixed point, which is impossible because a fixed point on H would imply a second fixed point on the sphere at infinity, as in the Lemma above. Q.E.D.

2.3.1.1. In $PSL(2, \mathbb{C})$ any translation is conjugate to

$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

which has trace 2.

2.3.1.2. In $O(3, 1)$ any translation is conjugate to

$$\begin{pmatrix} 3/2 & -1/2 & 1 & 0 \\ 1/2 & 1/2 & 1 & 0 \\ 1 & -1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

which has trace 4.

Case 2.3.2. g is orientation-reversing.

Lemma. g preserves some direction on H .

Proof. Draw a clock face on H (the old-fashioned analog kind -- no digital clocks please) and look at its image under g . As the second hand sweeps clockwise on the original clock, its image under g sweeps counterclockwise. Within 30 seconds the second hand and its image will point in the same direction. Q.E.D. (It's a good thing we used a second hand instead of an hour hand, or this might have been a much longer proof.)

Because g is orientation-reversing, it flips lines perpendicular to the fixed direction. It follows that g is a glide reflection. That is, it's a parabolic as in Case 2.3.1 above, composed with a reflection in the perpendicular direction. In $O(3, 1)$ its matrix is

$$\begin{pmatrix} 3/2 & -1/2 & 1 & 0 \\ 1/2 & 1/2 & 1 & 0 \\ 1 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

```

*           and the trace is 2.
*
*
* Interpreting the trace.
*
* PSL(2,C)
*
* Orientation-preserving isometries.
*
* Here it's best to look at the square of the trace rather than
* the trace itself, because the trace itself is well-defined
* only up to sign. The preceding classification of isometries
* show that
*
* g is elliptic   => trace^2(g) = 4 (cos(theta/2))^2
*                  = 4 (cosh(i theta/2))^2
* g is parabolic => trace^2(g) = 4
* g is hyperbolic => trace^2(g) = 4 (cosh(length/2))^2
* g is loxodromic => trace^2(g) = 4 (cosh((length + i theta)/2))^2
*
* This chart is good news. By computing the trace^2, we can
* deduce the type of isometry, and excluding the parabolic case
* we can even deduce the length and torsion!
*
* Note that elliptic and hyperbolic isometries are special cases
* of loxodromic one.
*
* By recalling the origin of the trace^2 as  $k + 2 + 1/k$  in the
* loxodromic case, we see that g is strictly loxodromic iff
* the trace^squared does not fall on the nonnegative real axis.
*
* Orientation-reversing isometries.
*
* PSL(2,C) is not well suited to orientation-reversing
* isometries (you need to work with  $\bar{z}$  instead of  $z$ )
* so we won't bother with this case.
*
* O(3,1)
*
* Orientation-preserving isometries.
*
* As remarked in 2.2.1.2 above, the trace alone does not provide
* enough information in O(3,1). Indeed, we can hardly expect
* a single real number to provide both length and torsion
* information. The characteristic polynomial would give us the
* magnitudes of both the length and the torsion, but we still
* wouldn't know the handedness of the twist (because it's not a
* conjugacy invariant in a group like O(3,1) which contains
* orientation-reversing elements). So it's best to use PSL(2,C)
* in the orientation-preserving case.
*
* Orientation-reversing isometries.
*
* Here the terminology is not well established, so let me first
* say what I mean by elliptic, parabolic and hyperbolic isometries.
*
* Lemma. Each orientation-reversing isometry of  $H^3$  is a
* reflection in a plane, followed by an orientation-preserving
* isometry which fixes that plane (setwise, not pointwise!).
*
* Proof. Follows from the classification of isometries
* given above. Q.E.D.
*
* Definition. An orientation-reversing isometry of  $H^3$  is
* elliptic, parabolic or hyperbolic iff it factors as a
* reflection in a plane, followed by an orientation-preserving
* elliptic, parabolic or hyperbolic isometry, respectively,
* which fixes that plane (setwise).
*
* Comment. A simple reflection in a plane corresponds to the
* identity in the orientation-preserving case. So it may be
* considered a degenerate elliptic, parabolic or hyperbolic
* isometry.

```

```

*      Lemma.  Orientation-reversing isometries have traces as follows:
*
*      type of isometry      trace
*      -----
*      elliptic              2 cosh(i theta) = 2 cos(theta)
*      parabolic             2
*      hyperbolic            2 cosh(length)
*
*      Proof.  Follows from the classification of isometries
*      given above.  Q.E.D.
*
*      Corollary.  We're in luck again.  The type of an orientation-
*      reversing isometry may be recognized from the trace of its
*      matrix in O(3,1).
*
*      Comment.  Orientation-reversing loxodromics do not exist.
*      This is a good thing.  An orientation-reversing isometry is
*      fully specified up to conjugacy by a single real parameter,
*      and that single real parameter is easily computed from the
*      (real valued!) trace of the matrix in O(3,1).
*/

#include "kernel.h"

#define TRACE_ERROR_EPSILON 1e-3
#define TORSION_EPSILON     1e-5

static Complex  orientation_preserving_complex_length(MoebiusTransformation *mt);
static Complex  orientation_reversing_complex_length(O3lMatrix m);
static Complex  signed_rotation_angle(MoebiusTransformation *mt);

Complex complex_length_mt(
    MoebiusTransformation  *mt)
{
    O3lMatrix  m;
    Complex    length;

    /*
     * Unfortunately we have to split into two cases, depending
     * on the parity of the MoebiusTransformation.  In the
     * orientation_preserving case we work with the SL2CMatrix
     * directly.  In the orientation_reversing case we convert
     * to an O3lMatrix.
     *
     * SL2CMatrices can't be used in the orientation_reversing
     * case because they are incapable of representing
     * orientation_reversing isometries.  O3lMatrices are inadequate
     * for the orientation_preserving case because their
     * characteristic polynomials carry the length and the magnitude
     * of the torsion, but not the sign of the torsion (indeed, it
     * couldn't possibly carry the sign, because the sign of the
     * torsion is not a conjugacy invariant relative to a group such
     * as O(3,1) which contains orientation_reversing elements).
     */

    if (mt->parity == orientation_preserving)
    {
        length = orientation_preserving_complex_length(mt);
    }
    else
    {
        Moebius_to_O3l(mt, m);
        length = orientation_reversing_complex_length(m);
    }

    return length;
}

Complex complex_length_o3l(
    O3lMatrix  m)
{
    MoebiusTransformation  mt;

```

```

Complex          length;

/*
 * This is the same as complex_length_mt() above,
 * only the input matrix is given in O(3,1).
 */

if (gl4R_determinant(m) > 0.0)
{
    O31_to_Moebius(m, &mt);
    length = orientation_preserving_complex_length(&mt);
}
else
{
    length = orientation_reversing_complex_length(m);
}

return length;
}

static Complex orientation_preserving_complex_length(
    MoebiusTransformation *mt)
{
    Complex trace,
           trace_squared,
           k,
           length;

    /*
     * The complex length depends on the trace, as explained below.
     */
    trace = complex_plus(mt->matrix[0][0], mt->matrix[1][1]);
    trace_squared = complex_mult(trace, trace);

    /*
     * 96/1/12 Craig has requested that for flat solutions SnapPea
     * provide consistent signs for rotation angles of elliptic
     * isometries of  $H^2$ . To avoid confusing flat solutions with nearly
     * flat solutions, do_Deahn_filling() in hyperbolic_structure.c now
     * sets the imaginary parts to zero when a solution is provably flat.
     */
    if (sl2c_matrix_is_real(mt->matrix) == TRUE
        && trace_squared.real < 4.0)
        return signed_rotation_angle(mt);

    /*
     * If the isometry represented by the MoebiusTransformation
     * is hyperbolic (i.e. a translation along a geodesic, with
     * a possible rotation), then it's conjugate to the isometry
     *  $f(z) = kz$ , for some complex number  $k$ . The complex log
     * of  $k$  gives the complex length of the geodesic.
     *
     * As a matrix,  $f()$  is written as
     *
     * 
$$\begin{pmatrix} k & 0 \\ 0 & 1 \end{pmatrix}$$

     *
     * When the determinant is normalized to One,
     * this becomes
     *
     * 
$$\begin{pmatrix} \sqrt{k} & 0 \\ 0 & 1/\sqrt{k} \end{pmatrix}$$

     *
     * This matrix is well-defined up to sign; therefore
     * the square of its trace is completely well-defined.
     * Because the trace is a conjugacy invariant, the
     * trace squared of this matrix equals that of the matrix  $m$ .
     */

    /*
     * We can now use the relationship
     *
     * 
$$\text{trace} = \sqrt{k} + 1/\sqrt{k}$$


```

```

*
* to solve for k in terms of the trace_squared.
* We follow our noses:
*
*      trace^2 = (sqrt(k) + 1/sqrt(k))^2
*
*      trace^2 = k + 1/k + 2
*
*      k^2 + (2 - trace^2)k + 1 = 0
*
*      (trace^2 - 2) +- sqrt(tr^2(tr^2 - 4))
*      k = -----
*              2
*
* It doesn't matter which of the two possible values
* of k we choose, since they are inverses of one another,
* and therefore their complex logs are negatives of
* one another.
*/

k = complex_real_mult(
    0.5,
    complex_plus (
        complex_minus(trace_squared, Two),
        complex_sqrt(
            complex_mult(
                trace_squared,
                complex_minus(trace_squared, Four)
            )
        )
    )
);

/*
* Now compute the complex length as log(k);
*/

length = complex_log(k, 0.0);

/*
* Make sure the length is positive.
*/

if (length.real < 0.0)
    length = complex_negate(length);

/*
* We want torsions of +-pi to be reported consistently
* as +pi, rather than letting roundoff error choose
* between +pi and -pi.
*/

if (length.imag < - PI + TORSION_EPSILON)
    length.imag += TWO_PI;

/*
* If the isometry represented by mt is parabolic,
* then it is conjugate to
*
*      1   1
*      0   1
*
* The above computation gives
*
*      trace_squared   = 4
*      k               = 1
*      length          = 0.0 + 0.0 i
*
* which is a sensible answer.
*/

return length;
}

```



```

static Complex signed_rotation_angle(
    MoebiusTransformation *mt)
{
    /*
     * Here we handle the special case of an orientation preserving
     * isometry whose matrix is all real and whose trace is in the
     * range (-2, 2). (This is an elliptic isometry of H^2.)
     * We want to compute the correct sign for the rotation, for use
     * in studying Seifert fibered spaces' base orbifolds.
     *
     * Thanks to Craig Hodgson for suggesting this improvement,
     * and providing the means to implement it.
     */

    /*
     * Let the isometry be
     *
     *          az + b
     * f(z) = -----
     *          cz + d
     *
     * We'll think of this both as an isometry of the upper half space
     * model of H^3, and also (when z is real) as an isometry of the
     * copy of H^2 which lies above the real axis. The fact that
     * the trace is in the range (-2, 2) implies this is an elliptic
     * isometry. Its rotation axis must be perpendicular to the
     * aforementioned copy of H^2. The rotation axis terminates in
     * a pair of fixed points on the boundary of upper half space.
     * The rotation angle (in H^2) is  $\arg(f'(w)) = -\arg(f'(\bar{w}))$ ,
     * where w (resp.  $\bar{w}$ ) is the fixed point with positive (resp.
     * negative) imaginary part. Let's compute  $\arg(f'(w))$ .
     *
     * First solve for w.
     *
     *          aw + b
     * w = -----
     *          cw + d
     *
     * =>
     *          c w^2 + (d - a)w - b = 0
     *
     * =>
     *          (a - d) +- sqrt((a - d)^2 + 4bc)
     * w = -----
     *                  2c
     *
     * =>
     *          (using ad - bc = 1)
     *
     *          (a - d) +- i sqrt(4 - (a + d)^2)
     * w = -----
     *                  2c
     *
     * Important note: to insure that w has positive imaginary part,
     * we must choose the + sign (in the "+-") when c > 0, and
     * the - sign when c < 0. (The case c = 0 can't occur when det = 1
     * and trace^2 < 4.)
     *
     * Now compute
     *
     *          1
     * f'(z) = -----
     *          (cz + d)^2
     *
     * Substitute in the above value for w to get
     *
     *          1
     * f'(w) = -----
     *          (cw + d)^2
     *
     *
     *          4
     *          -----
     *          ((a + d) +- i sqrt(4 - (a + d)^2))^2
     *
     *
     *          4
     *          -----
     *          (tr +- i sqrt(4 - tr^2))^2
     */
}

```

```

*
*      (tr -+ i sqrt(4 - tr^2))^2
*      = -----
*              4
*
*      tr^2      tr
*      = ---- - 1  -+ i -- sqrt(4 - tr^2)
*        2          2
*
* It follows that the magnitude of the angle is
*
*      tr^2
*      acos( ---- - 1 )
*        2
*
* If c > 0 (resp. c < 0) then the angle and the trace will have
* opposite (resp. the same) signs.
*/

double  tr,
        c;
Complex length;

tr = mt->matrix[0][0].real + mt->matrix[1][1].real;
c  = mt->matrix[1][0].real;

length.real = 0.0;
length.imag = safe_acos(0.5*tr*tr - 1.0); /* in the range (0, +pi] */
if ((c > 0.0) == (tr > 0.0))
    length.imag = - length.imag;

return length;
}

static Complex orientation_reversing_complex_length(
O3lMatrix  m)
{
double  trace;
Complex length;
int     i;

/*
* The section "Interpreting the trace" in the top-of-file
* documentation gives the trace of an orientation-reversing
* isometry as
*
*      type of isometry      trace
*      -----
*      elliptic              2 cosh(i theta) = 2 cos(theta)
*      parabolic              2
*      hyperbolic             2 cosh(length)
*/

trace = 0.0;
for (i = 0; i < 4; i++)
    trace += m[i][i];

if (trace < 2.0 - TRACE_ERROR_EPSILON)
{
    /*
    * elliptic
    */
    length.real = 0.0;
    length.imag = safe_acos(trace/2.0);
}
else if (trace > 2.0 + TRACE_ERROR_EPSILON)
{
    /*
    * hyperbolic
    *
    * The standard value of acosh() is nonnegative.
    */
    length.real = arccosh(trace/2.0);
}
}

```

```
        length.imag = 0.0;
    }
    else
    {
        /*
         *   parabolic
         */
        length.real = 0.0;
        length.imag = 0.0;
    }

    return length;
}
```