

```

/*
 * canonize_result.c
 *
 * The function
 *
 * Boolean is_canonical_triangulation(Triangulation *manifold);
 *
 * accepts a Triangulation which it assumes to be a subdivision of a canonical
 * cell decomposition (as produced by proto_canonize(), for example), and
 * says whether it is indeed the canonical cell decomposition itself.
 * In other words, is_canonical_triangulation() will return TRUE
 * when the canonical cell decomposition is a triangulation, and FALSE when
 * it is not.
 *
 * is_canonical_triangulation() assumes all Tetrahedra in the Triangulation
 * have valid "tilt" fields, as will be the case following a call to
 * proto_canonize().
 */

#include "kernel.h"
#include "canonize.h"

Boolean is_canonical_triangulation(
    Triangulation *manifold)
{
    Tetrahedron *tet,
                *nbr_tet;
    FaceIndex    f,
                nbr_f;
    double       sum_of_tilts;
    Boolean      result;

    /*
     * We'll assume the canonical cell decomposition is a triangulation
     * unless we discover otherwise.
     */
    result = TRUE;

    for (tet = manifold->tet_list_begin.next;
         tet != &manifold->tet_list_end;
         tet = tet->next)

        for (f = 0; f < 4; f++)
        {
            nbr_tet = tet->neighbor[f];
            nbr_f    = EVALUATE(tet->gluing[f], f);

            sum_of_tilts = tet->tilt[f] + nbr_tet->tilt[nbr_f];

            /*
             * The sum of the tilts should never be positive, because
             * this would mean that we didn't have a subdivision of the
             * canonical cell decomposition after all.
             */
            if (sum_of_tilts > CONCAVITY_EPSILON)
                uFatalError("is_canonical_triangulation", "canonize_result");

            /*
             * If the sum of the tilts is zero, then the canonical cell
             * decomposition contains cell other than tetrahedra.
             */
            if (sum_of_tilts > -CONCAVITY_EPSILON)
                result = FALSE;

            /*
             * Otherwise we're OK.
             */
        }

    return result;
}

```