Tutorial: Using a UDF to Control the Dynamic Mesh of a Flexible Oscillating Membrane

Introduction

The purpose of this tutorial is to illustrate how to use a user-defined function (UDF) to control the dynamic mesh of a generic flow device with a rotating blade and a flexible oscillating membrane. The motion of the rotating blade and the oscillating membrane have a large amplitude which requires the use of local remeshing. The rotation of the blade, the oscillation of the membrane, and the sliding of the nodes along the top wall of the housing are defined and controlled by means of a UDF that utilizes the three macros specific to the dynamic mesh model.

This tutorial demonstrates how to do the following:

- Set up the dynamic mesh (DM) model for this problem.
- Use the three DM-specific macros in a UDF to control the dynamic mesh.
- Preview the dynamic mesh before starting the flow computation.

Prerequisites

This tutorial is written with the assumption that you have completed Tutorial 1 from ANSYS FLUENT 12.0 Tutorial Guide, and that you are familiar with the ANSYS FLUENT navigation pane and menu structure. Some steps in the setup and solution procedure will not be shown explicitly.

For more details about UDFs, see ANSYS FLUENT 12.0 UDF Manual.

Problem Description

A generic flow device with a flexible oscillating membrane beneath it is shown in Figure 1. The valve radius is 0.1 m.

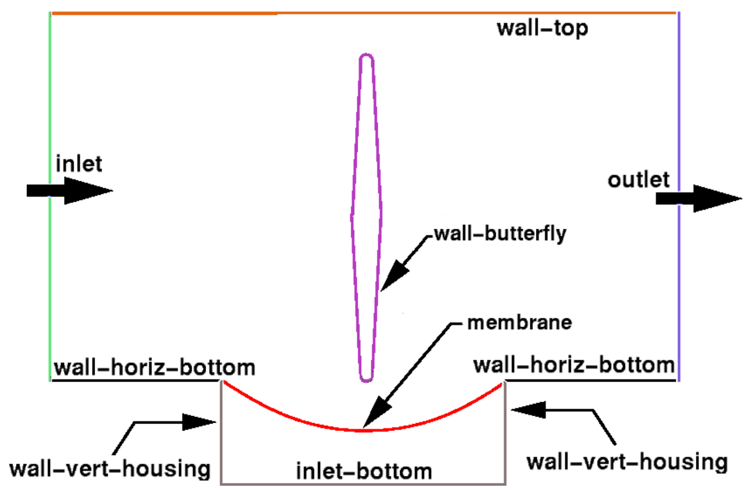


Figure 1: Problem Schematic

Setup and Solution

Preparation

1. Copy the files (`butterfly-flex.msh`, `butterfly-flex.c`) to your working folder.
2. Use FLUENT Launcher to start the 2D version of ANSYS FLUENT.

   For more information about FLUENT Launcher *see Section* 1.1.2, Starting ANSYS FLUENT Using FLUENT Launcher *in* ANSYS FLUENT *12.0 User's Guide.*
3. Enable Double-Precision in the Options list.
4. Click the UDF Compiler tab and ensure that the Setup Compilation Environment for UDF is enabled.

   The path to the .bat file which is required to compile the UDF will be displayed as soon as you enable Setup Compilation Environment for UDF*.*

   If the UDF Compiler *tab does not appear in the* FLUENT Launcher *dialog box by default, click the* Show More >> *button to view the additional settings.*

   The Display Options *are enabled by default. Therefore, after you read in the mesh, it will be displayed in the embedded graphics window.*

Step 1: Mesh

1. Read the mesh file (`butterfly-flex.msh.gz`).

   File ⟶ Read ⟶ Mesh...

   As the mesh file is read, ANSYS FLUENT *will report the progress in the console.*

Step 2: General Settings

1. Define the solver settings.

   General ⟶ Transient

2. Check the mesh.

   General ⟶ Check

   ANSYS FLUENT *will perform various checks on the mesh and will report the progress in the console. Make sure the minimum volume reported is a positive number.*

3. Display the mesh.

   General ⟶ Display...

   (a) Click Colors....

   The Mesh Colors *dialog box opens.*

   i. Select Color by ID from the Options list.

   ii. Click Close.

   (b) Click Display and close Mesh Display dialog box (see Figure 2).

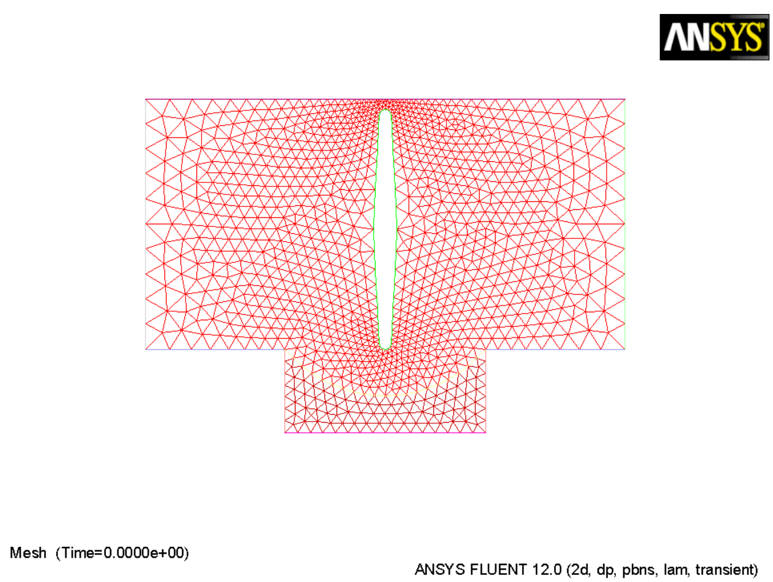
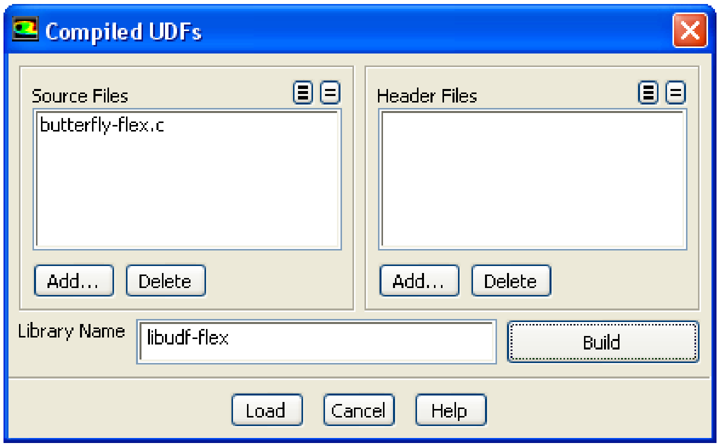


Figure 2: Mesh Display—Colored by ID

Step 3: User-Defined Functions

Compile the UDF.

Define ⟶ User-Defined ⟶ Functions ⟶ Compiled...



1. Click Add... and select the source file, butterfly-flex.c.
2. Enter `libudf-flex` for the Library Name.
3. Click Build to build the library.

   A Warning *dialog box opens, asking you to ensure that the UDF source files are in the same folder that contains the case and data files. Click* OK*.*
4. Click Load to load the newly created UDF library.

Step 4: Dynamic Mesh Setup

The mesh is controlled using a combination of local remeshing with sizing functions and spring smoothing.

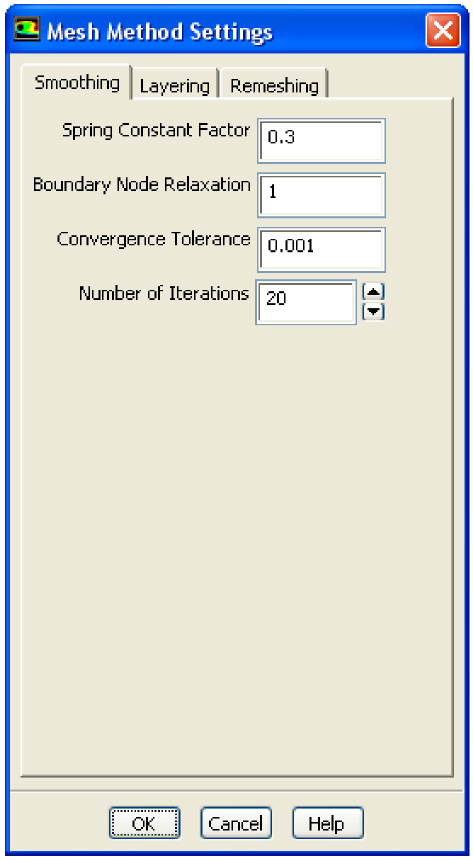
1. Enable dynamic mesh motion and specify the associated parameters.

   Dynamic Mesh

   (a) Enable Dynamic Mesh and Remeshing.

   The dialog box expands to show relevant inputs.

(b) Click Settings... to open Mesh Method Settings dialog box.



i. Click Smoothing tab and enter `0.3` for the Spring Constant Factor.

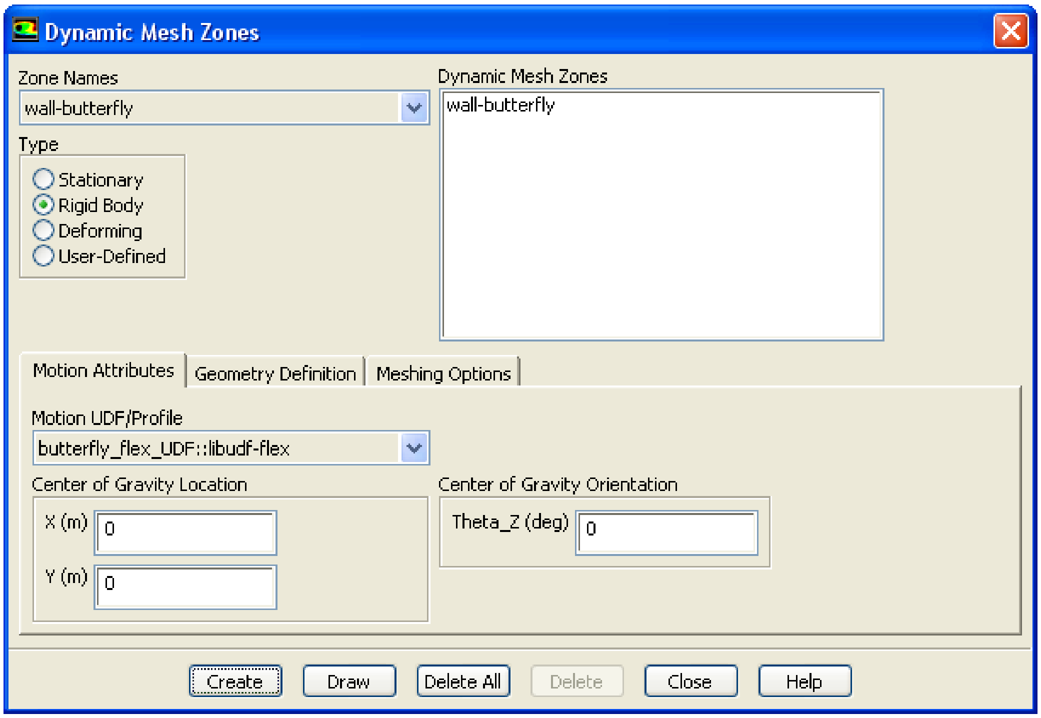
ii. Retain the default settings for other parameters.

iii. Click Remeshing tab and specify the parameters as given in the following table:

Parameter	**Value**
Minimum Length Scale (m)	0.00236
Maximum Length Scale (m)	0.02
Maximum Cell Skewness	0.65
Size Remesh Interval	1

2. Specify the wall-butterfly zone as a rigid body.

   Dynamic Mesh (Dynamic Mesh Zones)⟶ Create/Edit...



   (a) Select wall-butterfly from Zone Names drop-down list

   (b) Retain the default selection of Rigid Body in the Type group box.

   (c) Select butterfly_flex_UDF::libudf-flex from the Motion UDF/Profile drop-down list.

   (d) Retain the Center of Gravity Location as (0,0).

   The valve motion is defined about its center of gravity (CG) located at (0,0). This is important in the case of rotation.

   (e) In the Meshing Options tab, specify a value of `0.002` m for Cell Height.

   (f) Click Create.

3. Specify the wall-top zone as a deforming zone.

   The objective is to allow all nodes located on the top wall to slide along that wall as the tip of the valve moves past it. Thus define a horizontal line and set wall-top *zone as a deforming zone.*

   (a) Select wall-top from the Zone Names drop-down list.

   (b) Select Deforming in the Type list.

   (c) Click the Geometry Definition tab.

      i. Select user-defined from the Definition drop-down list.

      ii. Select plane::libudf-flex from the Geometry UDF drop-down list.

(d) Click the Meshing Options tab.

i. Disable Remeshing in the Methods list.

ii. Retain the default values for other parameters.

iii. Click Create.

4. Specify the membrane zone as a dynamic zone.

(a) Select membrane from the Zone Names drop-down list.

(b) Select User-Defined from the Type list.

(c) Select moving_arc::libudf-flex from the Mesh Motion UDF drop-down list.

The motion of membrane *and* membrane-shadow *wall is governed by the function* moving_arc *from the UDF.*

(d) Click Create.

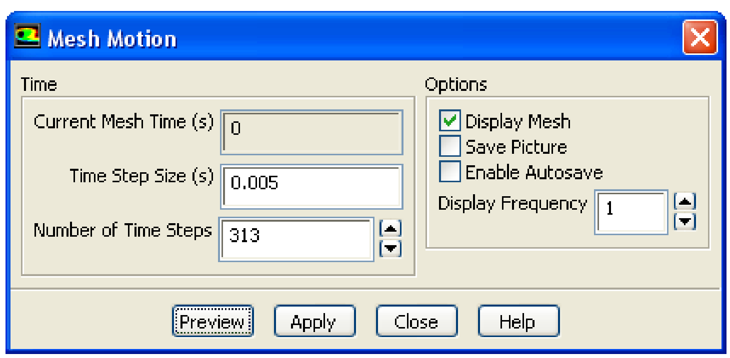
5. Similarly, define the membrane-shadow zone as a dynamic zone.

6. Save the case file (`membrane-moving-init.cas.gz`).

Step 5: Mesh Motion Preview

1. Specify the parameters for the mesh motion.

Dynamic Mesh ⟶ Preview Mesh Motion...



(a) Enter `0.005` for Time Step Size.

(b) Enter 313 for the Number of Time Steps.

(c) Click Preview.

The valve will move through 90 degrees and assume the position shown in Figure 3.

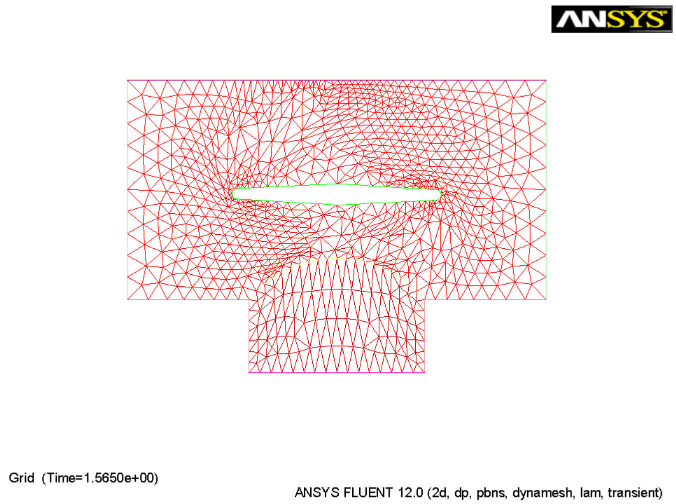


Figure 3: Mesh Display After Mesh Motion

Appendix

The contents of the UDF are as follows:

```
#include "udf.h"

#define  omega      1.0         /* rotational speed, rad/sec        */
#define  R          0.109       /* radius of the arc, meters        */

DEFINE_CG_MOTION(butterfly_flex_UDF, dt, cg_vel, cg_omega, time, dtime)
{
  cg_vel[0] = 0.0;
  cg_vel[1] = 0.0;
  cg_vel[2] = 0.0;

  cg_omega[0] = 0.0;
  cg_omega[1] = 0.0;
  cg_omega[2] = omega;
}

DEFINE_GEOM(plane, domain, dt, position)
{
position[1] = R;
}

DEFINE_GRID_MOTION(moving_arc, domain, dt, time, dtime)
{
  Thread *tf = DT_THREAD (dt);
  face_t f;
  Node *node_p;
  real alpha, theta, x, ymag, yfull, y;
  int n;

/* Set/activate the deforming flag on adjacent cell zone, which      */
/* means that the cells adjacent to the deforming wall will also be  */
/* deformed, in order to avoid skewness.                             */
  SET_DEFORMING_THREAD_FLAG (THREAD_T0 (tf));

  alpha = omega * CURRENT_TIME;
  theta = 2.0 * alpha + 3.0 * M_PI / 2.0;

  begin_f_loop (f, tf)
   {
     f_node_loop (f, tf, n)
      {
        node_p = F_NODE (f, tf, n);
```

```
   /* Update the current node only if it has not been         */
          /* previously visited:                                     */
          if (NODE_POS_NEED_UPDATE (node_p))




   {
              /* Set flag to indicate that the current node's            */
              /* position has been updated, so that it will not be       */
              /* updated during a future pass through the loop:          */


NODE_POS_UPDATED (node_p);

  x     = NODE_X (node_p);
  ymag  = sqrt (R*R - x*x) + 0.03;
  yfull = ymag - 0.1;
  y     = - 0.1 + yfull * sin(theta);
  NODE_Y (node_p) = y;
            }
        }
    }
  end_f_loop (f, tf);
}
```

The first section (under `DEFINE_CG_MOTION`*) prescribes a constant angular rotation about the z-axis, and is used to spin the valve. The second section (under* `DEFINE_GEOM`*) defines a line at $y = R = 0.109$ m; this line is coincident with the top wall of the housing and guides the nodes as they slide along the top wall. The third section (under* `DEFINE_GRID_MOTION`*) controls the motion of the nodes belonging to the membrane. The initial shape of the membrane is that of a circular arc, and then it moves up and down harmonically.*

Summary

In this tutorial you modeled an oscillating membrane in a generic flow device with a rotating blade using the DM model in ANSYS FLUENT. The local remeshing scheme was used to define the mesh motion. DM specific macros of a UDF were used to govern the rotational motion of the blade, deformation of the membrane, and the sliding of the nodes on the top wall of the housing.