Assignment 3: Transformation and Simple Animation

Introduction: In this assignment, you will work on extending **Assignment 3** by applying different types of transformation, i.e., translation, scale, and rotation on the Chinese hand fan you built in Assignment 2. With '**glutTimerFunc**' callback function, you will create a simple animation of simultaneous folding/unfolding of the Chinese hand fan in certain time intervals. For this Assignment, you may find uploaded examples on **glutTimerFunc** helpful. In order to do Assignment 3, you need to have basic part of **Assignment 2** complete.

Specification:

- **Apply scale factor:** **1 point**
  With the press of the keyboard button 's'/'S', the fan scales down to 25% of its original size. Toggling 's'/'S' brings the fan back to its original size and vice versa. In the scaled version, set the point size to 10 pixels.

- **Add multiple fans:** **6 point**
  With both scale ('s'/'S') and multiple ('m'/'M') modes on, 6 fans appear surrounding the center fan. So, you need to add six more fans using different transformation as discussed in this week's lectures. All six fans are placed at an angle of 60 degrees from each other with a radial distance of 0.6 from the center fan as demonstrated in Figure 1b and Figure 1c respectively.

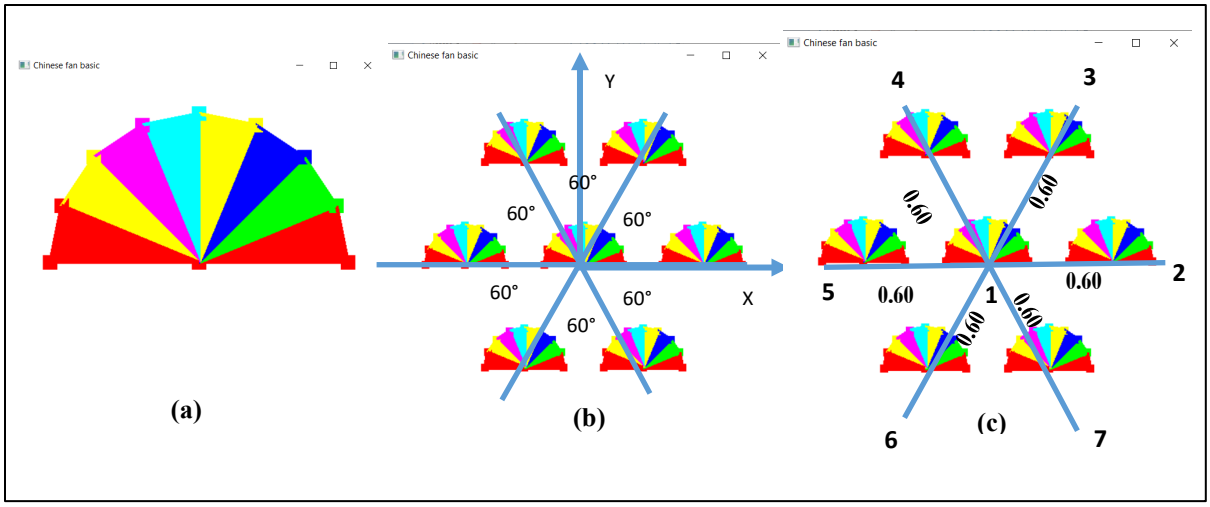

Figure 1: (a) Output from Assignment 2; (b) with 'scale' and 'multiple' modes on, fans are placed at an angle of 60 degrees from each other around the center fan; (c) fans are placed at a radial distance of 0.6 from the center fan.

Fan 2 is placed at a radial distance of **0.6** with an angle of 0 degree with respect to the positive X-axis. Similarly, **Fan 3** is placed at an angle of 60 degrees with respect to the positive X-axis, **Fan 4** is placed at 120 degrees with respect to the positive X-axis and so on. So, the added fans follow the path of a circle with a radius of **0.6**. For **'Fan 2'**, the X and Y translation components are 0.6*cos(angle) and 0.6*sin(angle) respectively where angle is zero degree. Similarly, for '**Fan 3**', the X and Y translation components are 0.6*cos(angle) and 0.6*sin(angle) respectively where angle is 60 degrees. Remember, you need to convert angles from degree to radian while using glm library. Now, for **Fan2**, translation matrix can be defined as follows:

mat4 trans_mat = translate(mat4(1.0f), vec3(0.6*cos(radians(0.0)), 0.6*sin(radians(0.0)), 0.0);

Similarly, for '**fan3**', translation matrix should be defined as follows:

mat4 trans_mat = translate(mat4(1.0f), vec3(0.6*cos(radians(60)), 0.6*sin(radians(60)), 0.0);

and so on….

Animation:

- **Rotation of center fans and surrounding fans: 7 points**
  After the fans are placed statically around the center fan as demonstrated in Figure 1, you need to add **glutTimerFunc** so that all added fans rotate around the center fan maintaining a constant original angular distance between each other, i.e., 60 degrees. You may add a delay of 100 milliseconds between the frames. The angular increment between the frames can be 5 degrees. Please take a look at the uploaded example "**MultipleHexagonRotated**".

  All fans (including the center fan) will also rotate around their own axes and center. You may add a delay of 500 milliseconds between the frames. Uploaded example on canvas on "**MultipleHexagonRotated**" will be helpful in understanding this part. You may use the same procedure for this assignment. The overall output may be better understood from the attached video. You will be able to start/ stop the rotation at any moment by toggling a keyboard press of **'r'/'R'**. Please note, all rotations are around Z-axis.

- **Simultaneous folding/unfolding of the fan: 1 point**
  The fan should fold/unfold simultaneously within a time interval of 500 milli seconds. Use '**glutTimerFunc**' to accomplish this part of the assignment. Take a look at the uploaded example "**HexagonElementUpdated**" to find how vertex update within a certain time interval has been accomplished by **glutTimerFunc** just toggling '**vertex_update**' variable within a certain time interval.

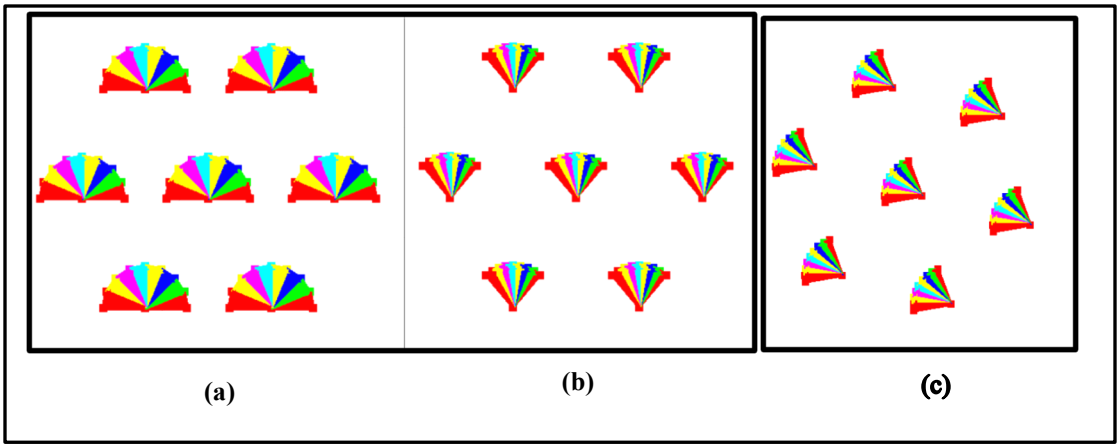

Figure 2: Some screen shots of animated fans.

Bonus part: 10 points

Bonus part for this assignment is a bit challenging in comparison with other assignments. So, I have set aside 10 points. Before you try this, you need to have a solid understanding of the basic part. If you try this part, it will enhance your creativity and imagination at the cost of your hard work and you will benefit in the end.

In bonus part, there will be gradual transition from unfolded to folded fan and vice versa. I would like to give you some tips. Please also watch the accompanied video. So, in the basic part, fans just switch between folded and unfolded options. They just have two states and they switch from one to the other.

In the bonus part, there will be a number of intermediate steps as fans gradually transition from the unfolded to the folded option. A few of these intermediate steps are demonstrated below in Figure 3. In the basic part, you use **glutTimerFunc** to transition between two modes within a certain time interval. In the bonus part, you will do the same for each intermediate step. You may choose to have 3 to 6 intermediate steps. The more there will be intermediate steps, the smoother the transition and the animation will be.

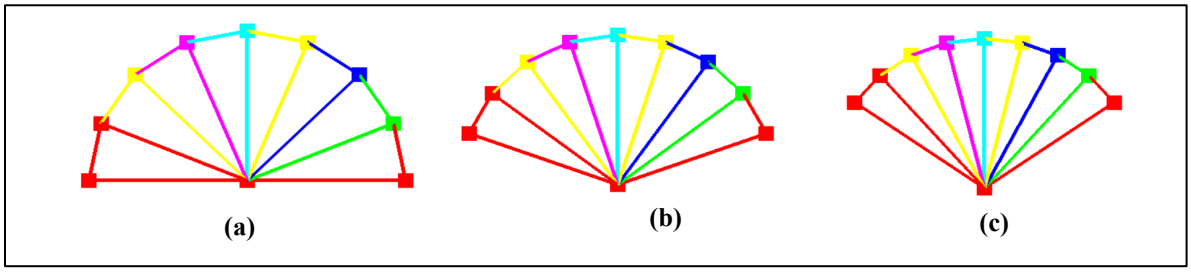

Figure 3: Some screen shots of animated fans in wireframe as the fan transitions from unfolded to folded mode via a number of intermediate steps.

Below is what you need to do. Find out the angular difference between corresponding points i.e., A and A', B and B', and more in unfolded and folded versions respectively. Next, divide the differences in a fixed intervals i.e., 3 to 6. Update the vertices accordingly for the intermediate

steps and use **glutTimeFunc** to switch between intermediate steps. Once the fan switches from unfolded to folded mode, again you will need to gradually switch from the folded to unfolded mode in a reverse way.

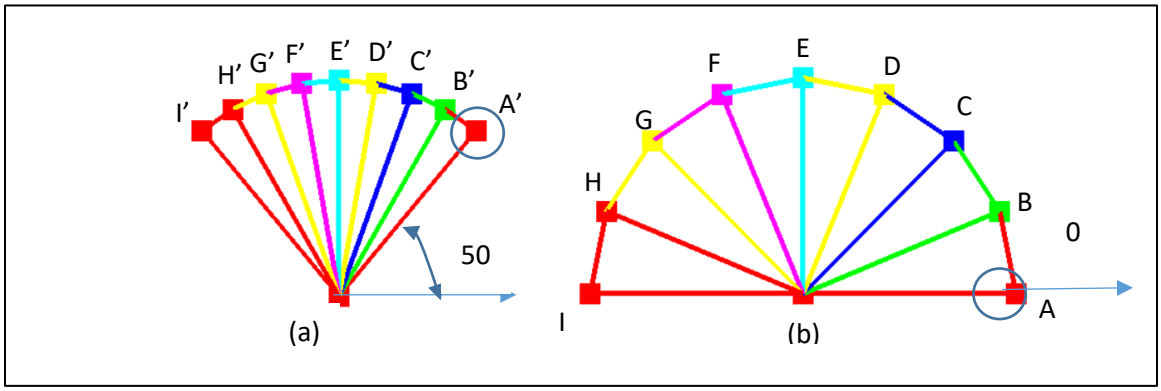

Figure 4: Determining angular distance between the corresponding points, i.e., A and A' between the unfolded and folded modes.

GLM Library:

- Use **glm** library for applying different transformation, i.e., translation, rotation, and scale. All transformation for a particular fan should be reduced to a single transformation by concatenation of transformation and be declared as a uniform 4X4 matrix variable named "model_matrix" or "model_transformation".

Submission:

Basic Part:

Place your solution in a zipped file named with your last name followed by the first initial of your first name followed by 3 (ex: **CSCD377YasminS3Basic.zip**) and submit the solution via canvas. Thus, your zip should contain the following:

- A file named **ChineseFanBasic.h**, **ChineseFanBasicAnimated.cpp**, and shader files **ChineseFan.vs** and **ChineseFan.fs**.

Bonus Part:

Place your solution in a zipped file named with your last name followed by the first initial of your first name followed by 3 (ex: **CSCD377YasminS3Bonus.zip**) and submit the solution via canvas. Thus, your zip should contain the following:

- A file named **ChineseFanBonus.h, ChineseFanBonusAnimated.cpp**, and shader files **ChineseFan.vs** and **ChineseFan.fs**.

Submission deadline is Friday, May 1, 11:59 pm. This assignment carries a weightage of 15% of this course.