1 Question 1

The procedure of the rotation is listed as follows:

1. move point P_1 to the origin of the coordinate
2. rotate around the X axis to move the vector U in the XoZ plane
3. rotate around the Y axis to align U with the same direction of Z axis
4. rotate the cube 60 degrees clockwise around the Z axis
5. apply the reverse operations for step 1,2 and 3

Submatrixes of M:

$$M = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 5 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{3}{\sqrt{13}} & \frac{2}{\sqrt{13}} & 0 \\ 0 & -\frac{2}{\sqrt{13}} & \frac{3}{\sqrt{13}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{\sqrt{13}}{\sqrt{14}} & 0 & \frac{1}{\sqrt{14}} & 0 \\ 0 & 1 & 0 & 0 \\ -\frac{1}{\sqrt{14}} & 0 & \frac{\sqrt{13}}{\sqrt{14}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.5 & \frac{\sqrt{3}}{2} & 0 & 0 \\ -\frac{\sqrt{3}}{2} & 0.5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
$$\begin{bmatrix} \frac{\sqrt{13}}{\sqrt{14}} & 0 & -\frac{1}{\sqrt{14}} & 0 \\ 0 & 1 & 0 & 0 \\ \frac{1}{\sqrt{14}} & 0 & \frac{\sqrt{13}}{\sqrt{14}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{3}{\sqrt{13}} & -\frac{2}{\sqrt{13}} & 0 \\ 0 & \frac{2}{\sqrt{13}} & \frac{3}{\sqrt{13}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & -5 \\ 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The value of M is equal to:

$$M = \begin{bmatrix} 0.56 & 0.76 & -0.35 & -2.63 \\ -0.62 & 0.64 & 0.44 & 1.54 \\ -0.34 & 0.58 & 0.71 & -1.98 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

The rotated cube is displayed in Figure 1.

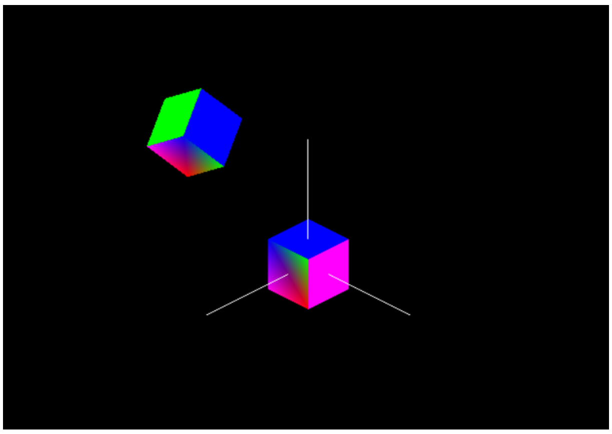

Figure 1: Rotated Cube

2 Question 2

This section will introduce the main steps for question 2, which I will introduce the implementation details in terms of the snowman model and the run-time rotation.

First, the procedure for constructing the snowman is illustrated. A bottom-up methodology is leveraged when displaying the snowman, which constructs the snowman sequentially with the key break-down structures (i.e., the base, the body, the arms and the head). For displaying each body structure, separate functions are called (e.g., *drawBody()* function which draws the body of the snowman). Second, the run-time rotation (both the self-spinning of the snowman and the rotation revolving around a center point in a circle) is achieved with the help of some auxiliary functions such as the *idleFunc()* and the *update()* function. The *idleFunc()* continuously calculates the elapsed time of the program and then trigger *update()* function to update the time-related variables for snowman rotation, such as scaling coefficients, the rotating angles and the coordinates. The snowman will then be re-displayed as the *idleFunc()* calls the built-in *glutPostRedispaly()* after all the variables are updated in the *update()*. Since the *idleFunc()* is called when the program is in idle state, the time-related variables are updated in a fine-granularity, which contributes to the smooth animation of the snowman. More specifically, the scaling coefficient is linear to the elapsed time, which is calculated as follows:

$$scale = \begin{cases} 1 - 1.5t & \text{if } t \leq 0.33 \\ 0.55 - 0.51t & \text{if } 0.33 < t \leq 0.66 \\ 0.33 + 2.01(t - 0.66) & \text{if } 0.66 < t \leq 1 \end{cases} \tag{2}$$

where t represents the ratio of the rotated angle for the current cycle to the full cycle. t is determined by the folloing equation:

$$n = \frac{T}{C} \tag{3}$$

$$t = n - floor(n) \tag{4}$$

where T represents the elapsed time, C represents the rotation time for one cycle, *floor(x)* represents the function that output the greatest integer less than or equal to the input x.

Figure 2 shows the screenshots of my snowman.

3 Question 3

Similar to the main steps for Question 2, the main steps of the answer to Question 3 also consists of two parts, the model design and the implementation of the rotation. First, the turntable is also built in a bottom-up order, which is in the order of the base, the sliced and colorful patches on the base and the center the pillar at the center , and the bars/pipes on the turntable. The model is first displayed along the negative direction of the z zxis and the rotated around the x axis to stand vertically. Second, the implementation of the rotation is similar to the answer to Question 2, which is achieved by the *idleFunc()* function. The *idleFunc()* continuously calculate the rotated angle compared to the initial state and call *glutPostRedisplay()* function to update the display of the turntable.

Figure 3 shows the screenshots of the turntable.

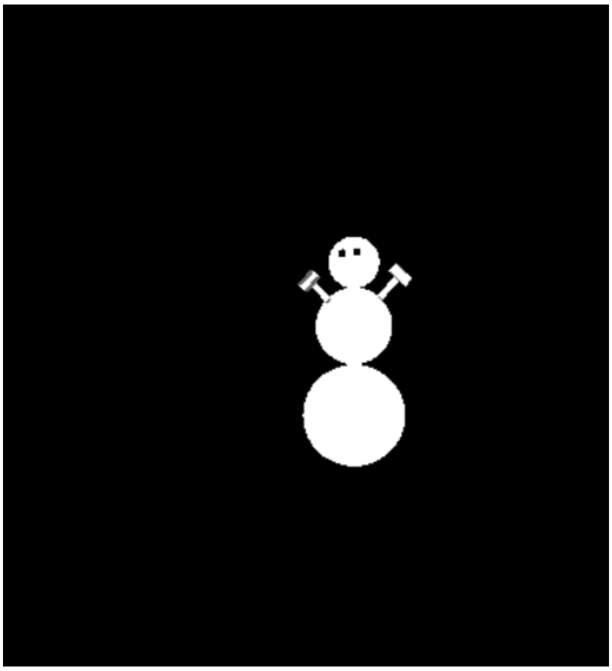

Figure 2: Screenshots of the snoman

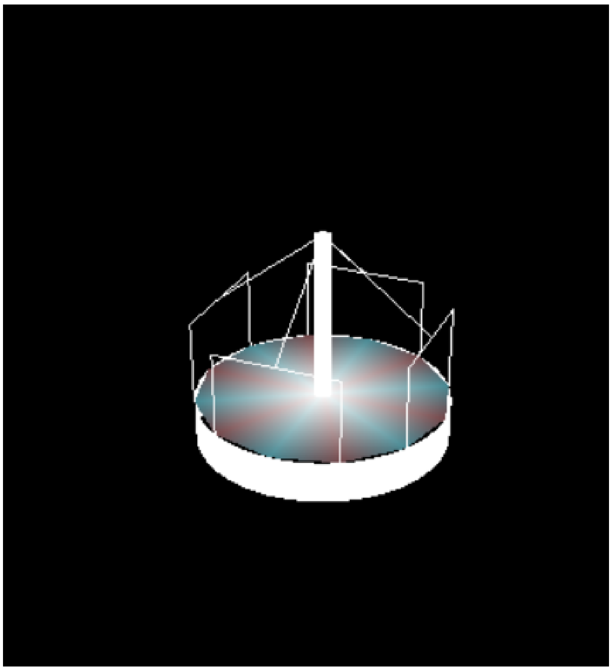

Figure 3: The screenshot of the turntable