

There are five tasks each carrying the same weight.

1. Fully Controllable Camera (1.exe)
2. Sphere to/from Cube (1.exe)
3. Torus (2.exe)
4. Hand Simulation (3.exe)
5. Wheel Simulation (4.exe)

1. Fully Controllable Camera (1.exe)

up arrow - move forward

down arrow - move backward

right arrow - move right

left arrow - move left

PgUp - move up

PgDn - move down

1 - rotate/look right

2 - rotate/look left

3 - look up

4 - look down

5 - tilt clockwise

6 - tilt counterclockwise

Hint (for Section A students):

Maintain 4 global variables: 1 3d point `pos` to indicate the position of the camera and 3 3d **unit** vectors `u`, `r`, and `l` to indicate the up, right, and look directions respectively.

The first 6 operations listed above are move operations, where the position of the camera changes but the up, right, and look directions do not. The last 6 operations are rotate operations, where the camera position does not change, but the direction vectors do.

In case of a move operation, move `pos` a certain amount along the appropriate direction, but leave the direction vectors unchanged. For example, in the move right operation, move `pos` along `r` by 2 (or by any amount you find appropriate) units.

In case of a rotate operation, rotate two appropriate direction vectors a certain amount around the other direction vector, but leave the position of the camera unchanged. For example, in the look up operation, rotate `l` and `u` counterclockwise with respect to `r` by 3 (or by any amount you find appropriate) degrees.

If you maintain `pos`, `u`, `r`, and `l` in this way, your `gluLookAt` statement will look as follows:

```
gluLookAt(pos.x, pos.y, pos.z,  
          pos.x + l.x, pos.y + l.y, pos.z + l.z,  
          u.x, u.y, u.z);
```

2. Sphere to/from Cube (1.exe)

Home - cube to sphere

End - sphere to cube

Draw one eighth of a sphere, one fourth of a cylinder and a square once.

Use transformations (translation, rotation etc.) to put them in the right places.

It is not required to imitate the color patterns, but the shape of the object must be discernible.

3. Torus (2.exe)

1 - increase inner radius

2 - decrease inner radius

3 - increase outer radius

4 - decrease outer radius

Use arrow keys to move the camera.

You are not allowed to use any library function provided by OpenGL to draw the torus.

Generate the points on the torus and draw quads using them.

It is **required** to imitate the color patterns.

4. Hand Simulation (3.exe)

Press the keys 1, 2, 3, ..., 9, 0, q, and w to find out how they work.

Also observe that after a certain amount, each joint ceases to rotate.

Use arrow keys to move the camera.

You can use the OpenGL library function `glutWireSphere` and scale it to draw the parts of the arm.

5. Wheel Simulation (4.exe)

w - move forward

s - move backward

a - rotate left

d - rotate right

Use arrow keys to move the camera.

Keep in mind that a full (360 degree) rotation of the wheel moves the wheel linearly by a length equal to the perimeter of the wheel.

It is not required to imitate the color patterns, but the shape of the wheel must be discernible.