# Report for PA3

20170690 최종윤

## 1. Variables

*transMode* variable can be changed by key press, 'm' and 'v'. And *moveOrRotate* boolean variable state whether translation mode or rotation mode in viewing space.

```
char transMode = 'm';          // m : modeling space, v : viewing space
bool moveOrRotate = true;
```

## 2. Mode Change

In *onKeyPress()*, I put some code for changing the *transMode* variable by key pressing 'm' or 'v'. And the modeling translation parts that I implemented in PA2 is inside of the *if (transMode == 'm')* statement. Also the viewing space translation parts that I implemented in PA3 is inside of the *if (transMode == 'v')* statement.

In *onMouseDrag(),* I branch two condition with modeling space and viewing space, same with *onKeyPress()* function.

```
if (key == 'm') {
    printf("modeling space\n");
    transMode = 'm';
}
if (key == 'v') {
    printf("viewing space\n");
    transMode = 'v';
}

if (transMode == 'm') { … }
else if (transMode == 'v') { … }
```

## 3. Translation

There are two translation situations, x-y plane and z axis. So I set some *amount* value to make it move in x-y plane or z axis. And these translation set *moveOrRotate* value as 'false', so that it can execute the translation codes in *onMouseDrag()*.

The cow should move in x-y plane or z axis in 'viewing space'. So I transform the cow matrix from world to camera space, by multiplying the *wld2cam.matrix(), cam2wld.matrix()* in sequence. And I put the translation with the amount of mouse drag between two matrix multiplications, so that this translation can be performed in viewing space.

```
if (key == 'x' || key == 'y') {
    moveOrRotate = false;
    amount[0] = 0.05;   amount[1] = 0.05;   amount[2] = 0.0;
}
if (key == 'z') {
    moveOrRotate = false;
    amount[0] = 0.0;    amount[1] = 0.0;    amount[2] = 0.05;
}

else {
    glMultMatrixd(cam2wld[cameraIndex].matrix());
    glTranslated(amount[0] * (x - trans_oldX), amount[1] * (y - trans_oldY), amount[2]
        * (x - trans_oldX));
    glMultMatrixd(wld2cam[cameraIndex].matrix());
    glMultMatrixd(cow2wld.matrix());
}
```

## 4. Rotation

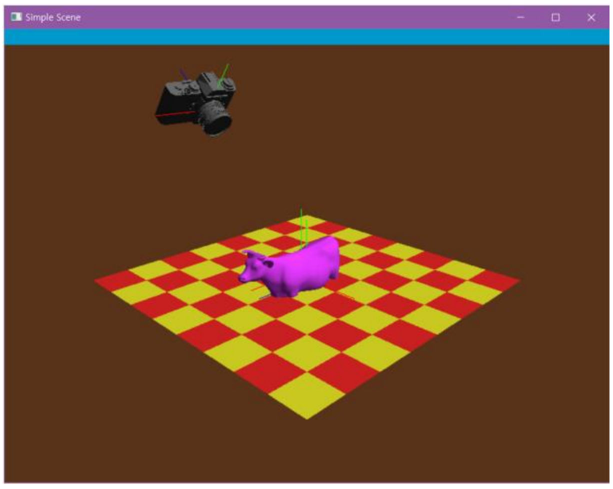
If I press the key 'r', the moveOrRotate variable is set to 'true' value, and this can execute the rotation codes in *onMouseDrag().*

The cow should rotate along the x-axis of viewing space. So I move the cow to the camera position in world space. The arguments of *glTranslated()* are the amount of the movement from current cow position to camera position. And change the frame to viewing space, and rotate the cow along the x-axis of viewing space with the amount of mouse drag. And then come back to the world space, and move the cow to the original position.

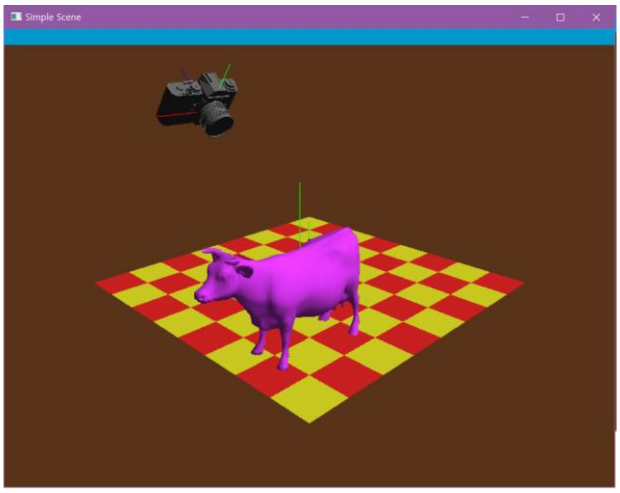
```
if ((key == 'r')) {
    moveOrRotate = true;
}

if (moveOrRotate) {
    glTranslated(cow2wld.matrix()[12] - cameras[cameraIndex][0], cow2wld.matrix()[13]
        - cameras[cameraIndex][1], cow2wld.matrix()[14] - cameras[cameraIndex][2]);
    glMultMatrixd(cam2wld[cameraIndex].matrix());
    glRotated((x - trans_oldX), 1, 0, 0);
    glMultMatrixd(wld2cam[cameraIndex].matrix());
    glTranslated(-cow2wld.matrix()[12] + cameras[cameraIndex][0], -cow2wld.matrix()[13]
        + cameras[cameraIndex][1], -cow2wld.matrix()[14] + cameras[cameraIndex][2]);
    glMultMatrixd(cow2wld.matrix());
}
```

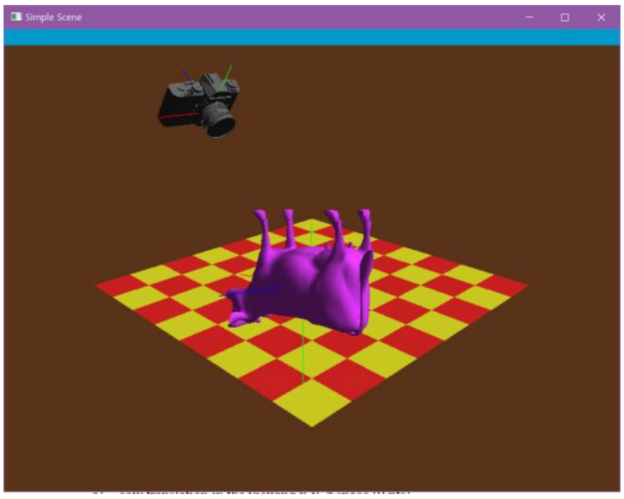
- **The images of executing**



(Cow translation in viewing space, x-y plane)



(Cow translation in viewing space, z-axis)



(Cow rotation in viewing space along x-axis)