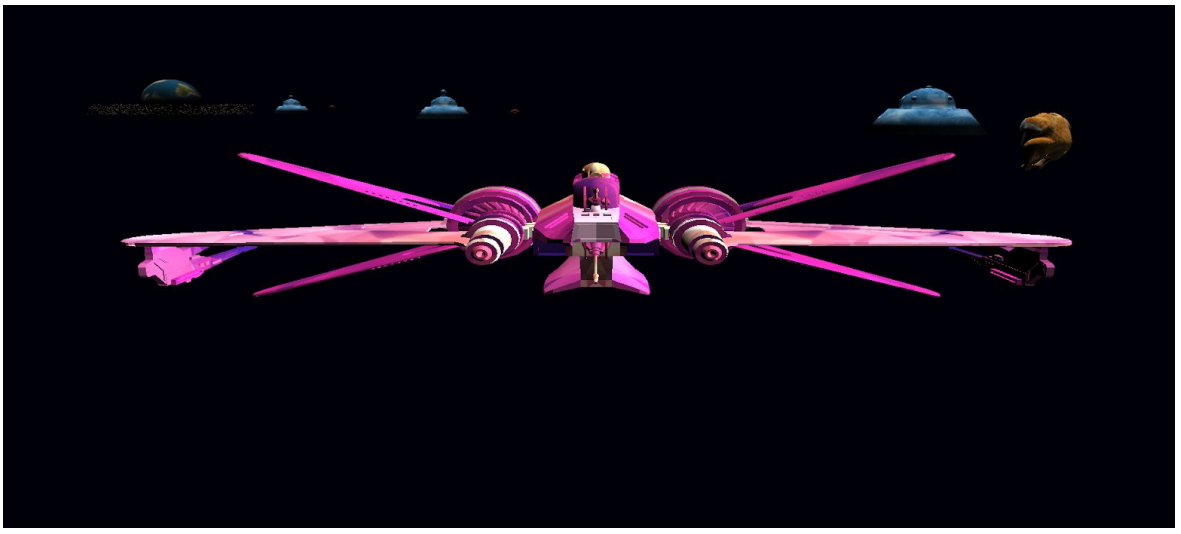
Name: Cheng Ka Pui & Lam Puy Yin
SID: 1155125534 & 1155126240
CSCI 3260 Project Report

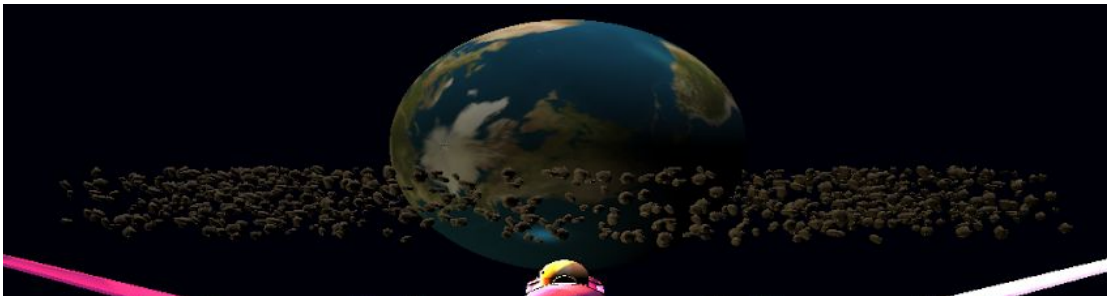
Project Report

1. Overall scene

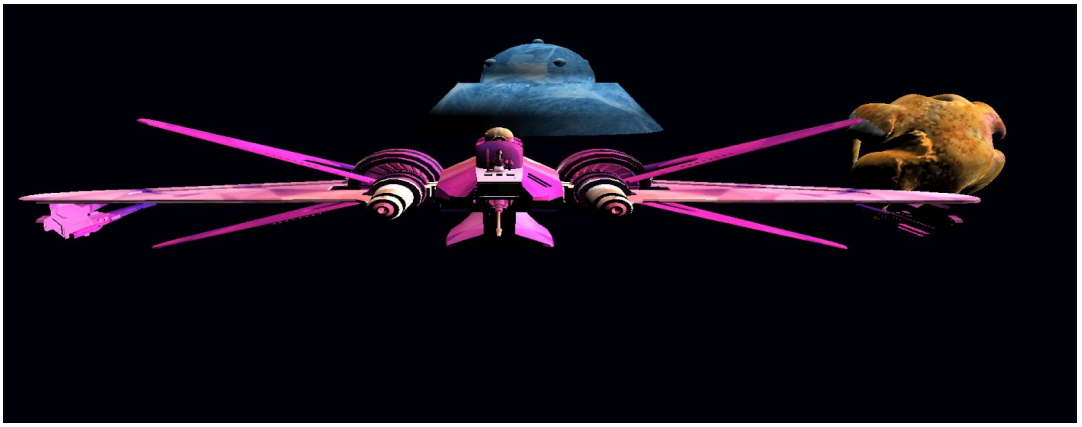


2. Close look at the basic light rendering results on each kind of the objects

Planet and Asteroids

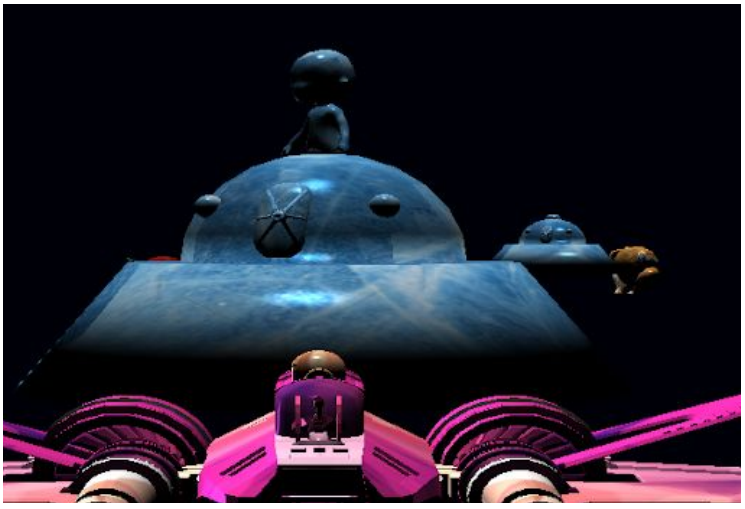


Spacecraft and Chicken

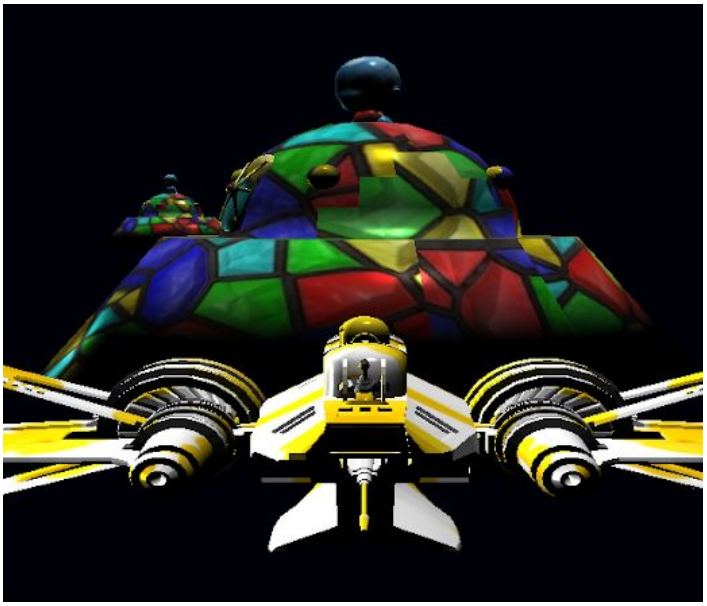


Name: Cheng Ka Pui & Lam Puy Yin
SID: 1155125534 & 1155126240
CSCI 3260 Project Report

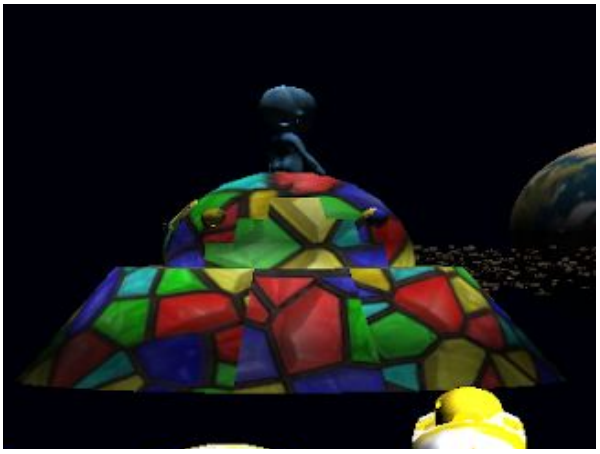
Alien and his alien vehicle (1)



Alien and his alien vehicle (2)

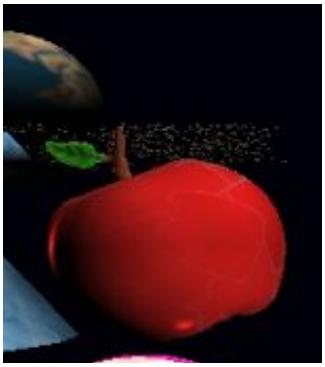


Alien and his alien vehicle (3)



Name: Cheng Ka Pui & Lam Puy Yin
SID: 1155125534 & 1155126240
CSCI 3260 Project Report

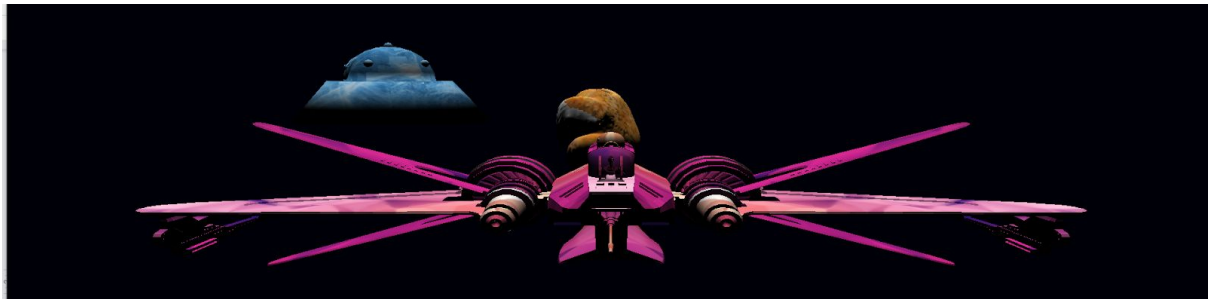
Apple



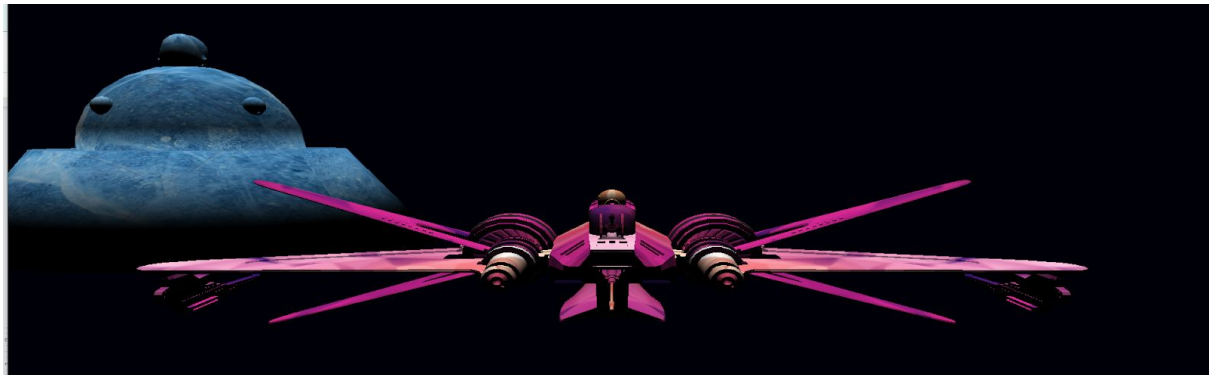
3. Frames that show that the spacecraft is

(1) collecting foods

Before



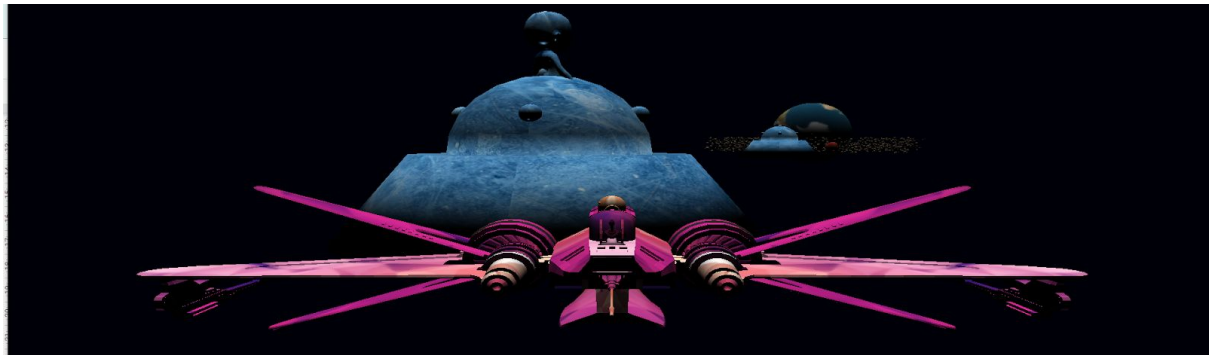
After



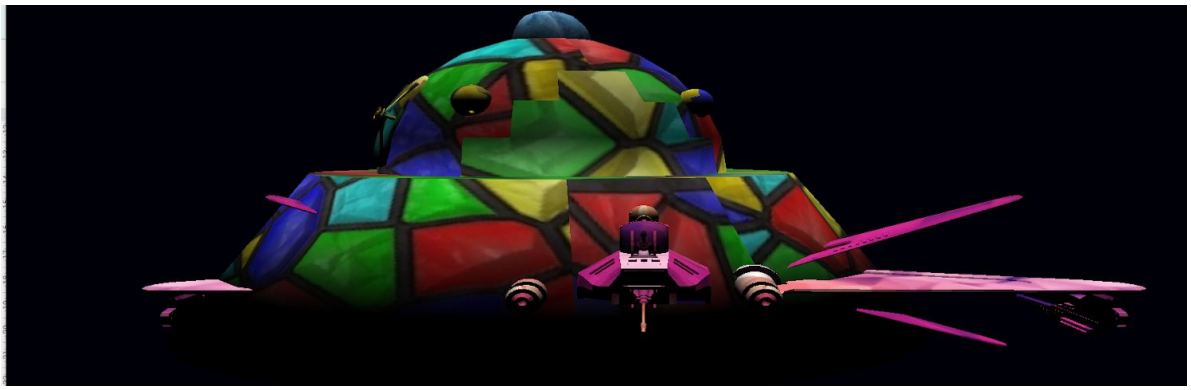
(2) visiting the space vehicle

Name: Cheng Ka Pui & Lam Puy Yin
SID: 1155125534 & 1155126240
CSCI 3260 Project Report

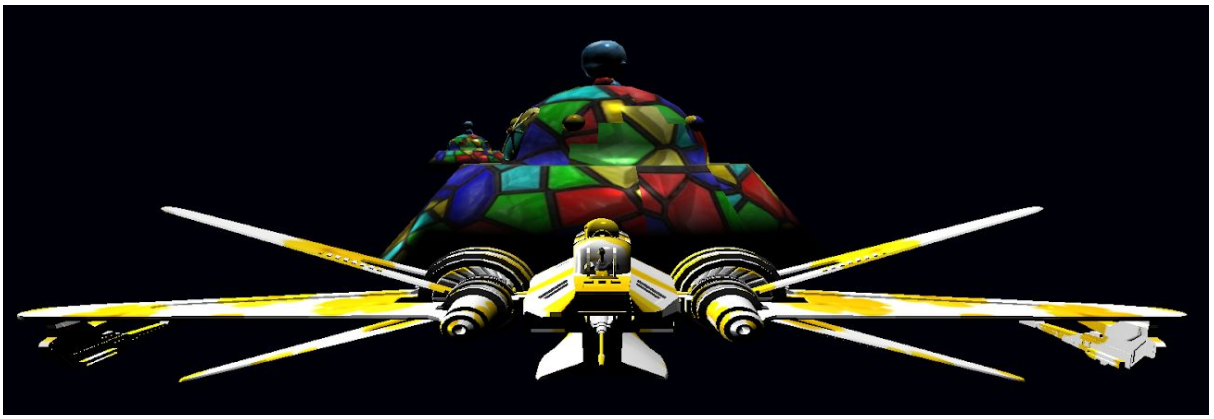
Before



After



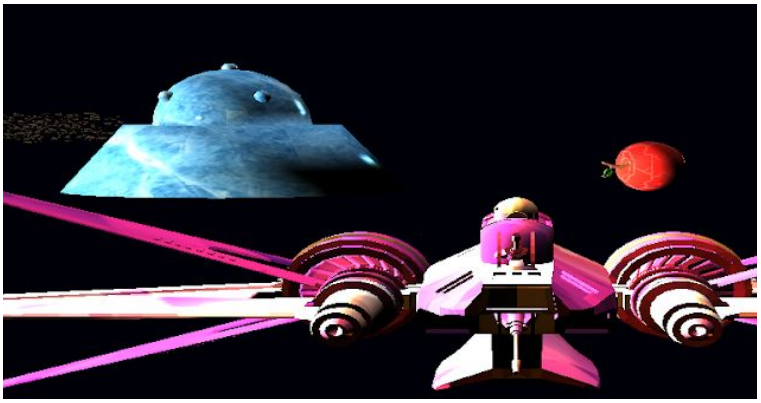
(3) changing the texture of spacecraft after visiting finished



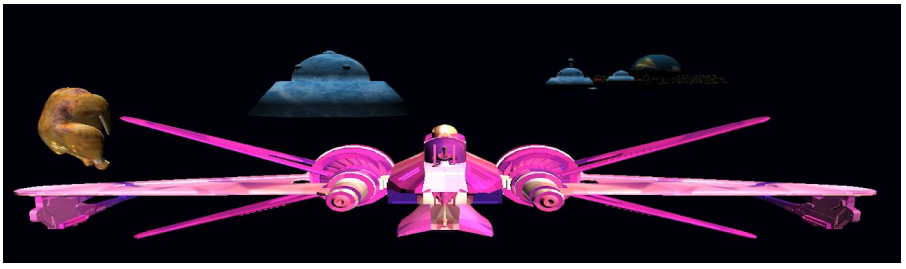
Name: Cheng Ka Pui & Lam Puy Yin
SID: 1155125534 & 1155126240
CSCI 3260 Project Report

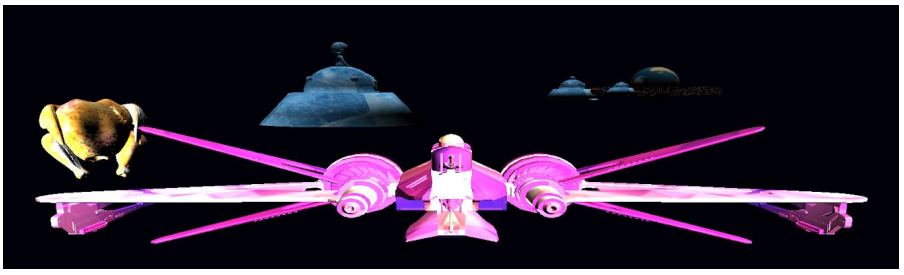
4. The frames that can represent any bonus features that you have implemented

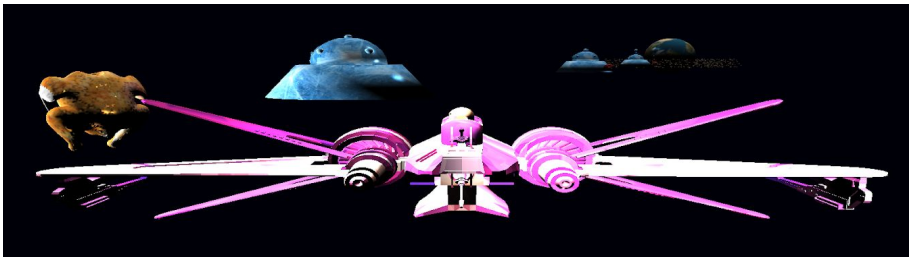
Extra food(apple)



Extra light source
Spotlight (press S and D to move the light position)



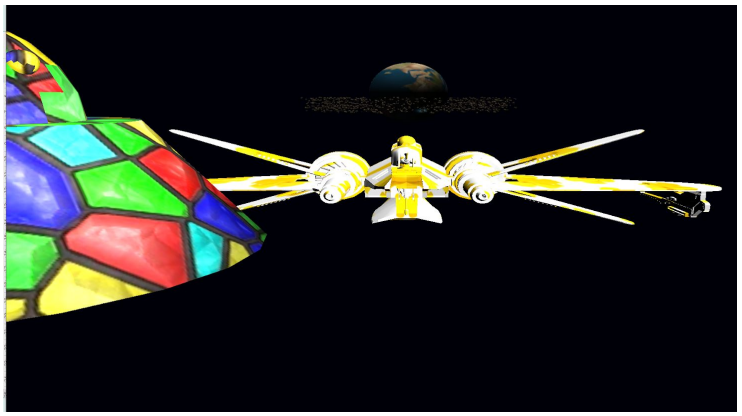




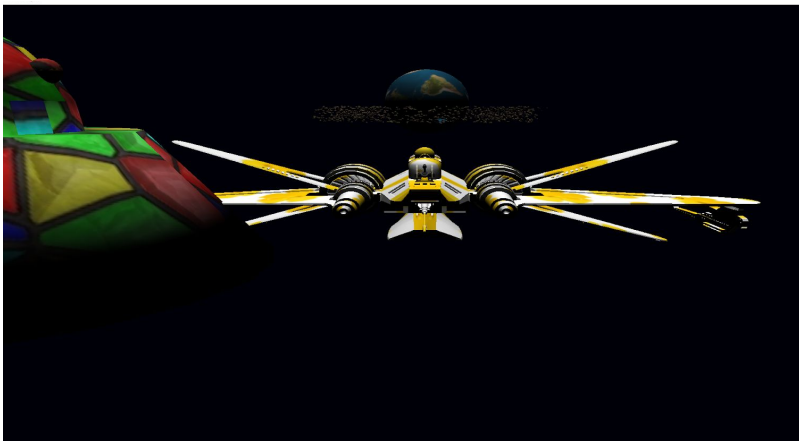
Name: Cheng Ka Pui & Lam Puy Yin
SID: 1155125534 & 1155126240
CSCI 3260 Project Report

Light control by pressing Z

Light On



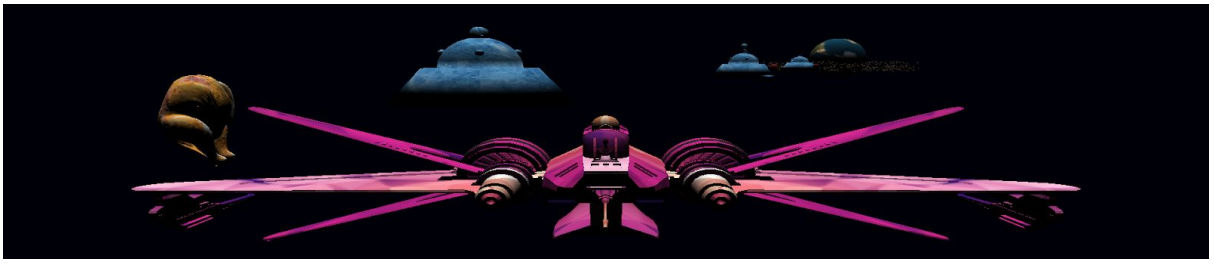
Light Off



Point Light (automatic light color change)

Name: Cheng Ka Pui & Lam Puy Yin
SID: 1155125534 & 1155126240
CSCI 3260 Project Report

Light intensity control (press W and S to control the brightness)



5. Some brief and necessary descriptions of your implementation details

Collision Handling (Distancing)

When the spacecraft is near a specific object (eg. chicken),the scale of the object will set to 0(e.g. chicken_trigger). The food will be “collected” and disappear. The leisure_trigger will then add up by 1, the corresponding trigger_detection boolean variable will also be activated to avoid the same collision again.

```
if ((SCInitialPos[0] + SCTranslation[0])< (chicken_location.x *0.01f + collision_range)
    && (SCInitialPos[0] + SCTranslation[0]) > (chicken_location.x * 0.01f - collision_range)
    && (SCInitialPos[2] + SCTranslation[2]) < (chicken_location.z *0.01f + collision_range)
    && (SCInitialPos[0] + SCTranslation[2]) > (chicken_location.z * 0.01f - collision_range)
    && trigger_detection[0] == false) {
    chicken_trigger = 0.0;
    leisure_trigger++;
    trigger_detection[0] = true;
    winning_detection();
}
```

After finishing all the visiting tasks (foods and vehicles), the texture of the spacecraft will change to a leisure style(winning_dectection).

Name: Cheng Ka Pui & Lam Puy Yin
SID: 1155125534 & 1155126240
CSCI 3260 Project Report

Asteroid Ring Rendering (Intancing)

As we are going to rendern the rock.obj repeatedly for at least 200 times, we implement an intancing function called “Asteroid”. Also, we use random function (rand) to generate objects in random locations.

```
void Asteroids(int amount) {
    glm::mat4* modelMatrices;
    modelMatrices = new glm::mat4[amount];
    srand(glfwGetTime() * 0.000001); // initialize random seed
    float radius = 16.5;
    float offset = 4.5f;

    for (GLuint i = 0; i < amount; i++)
    {
        glm::mat4 model = glm::mat4(1.0f);

        // rotate at earth centre
        model = glm::translate(model, glm::vec3(earth_location.x * 3, earth_location.y, earth_location.z * 3));
        model = glm::rotate(model, glm::radians((float)glfwGetTime() * 10), glm::vec3(0.0f, 1.0f, 0.0f));

        // 1. translation: displace along circle with 'radius' in range [-offset, offset]
        float angle = (float)i / (float)amount * 360.0f;
        float displacement = (rand() % (int)(2 * offset * 100)) / 100.0f - offset;
        float x = sin(angle) * radius + displacement;
        displacement = (rand() % (int)(2 * offset * 100)) / 100.0f - offset;
        float y = displacement * 0.4f; // keep height of field smaller compared to width of x and z
        displacement = (rand() % (int)(2 * offset * 100)) / 100.0f - offset;
        float z = cos(angle) * radius + displacement;
        model = glm::translate(model, glm::vec3(x, y, z));

        // 2. scale: scale between 0.01 and 0.04f
        float scale = (rand() % 100) / 1000.0f + 0.1;
        model = glm::scale(model, glm::vec3(scale));

        // 3. rotation: add random rotation around a (semi)randomly picked rotation axis vector
        float rotAngle = (rand() % 360);
        model = glm::rotate(model, rotAngle, glm::vec3(0.4f, 0.6f, 0.8f));
        // 4. now add to list of matrices
        modelMatrices[i] = model;
    }
```

Update Camera and SpaceCraft Status

We implement a “UpdateStatus” function so that when the SpaceCraft moves, the camera will also move along. Using the model matrix (spaceship local), the camera’s position and target location will then be standardised. As well as the front direction and right direction are uniformed by the spaceship model matrix too.

```
//camera and spaceship system update
void UpdateStatus() {
    float scale = 0.005;

    glm::mat4 SC_scale_M = glm::scale(glm::mat4(1.0f), glm::vec3(scale));
    glm::mat4 SC_trans_M = glm::translate(glm::mat4(1.0f),
        glm::vec3(SCInitialPos[0] + SCTranslation[0], SCInitialPos[1] + SCTranslation[1],
            SCInitialPos[2] + SCTranslation[2]));
    SC_Rot_M = glm::rotate(glm::mat4(1.0f), glm::radians(theta), glm::vec3(0.0f, 1.0f, 0.0f));

    spaceshipLocal = SC_trans_M * SC_Rot_M * SC_scale_M;
    SC_world_pos = spaceshipLocal * glm::vec4(SC_local_front, 1.0f);
    SC_world_Front_Direction = spaceshipLocal * glm::vec4(SC_local_front, 1.0f);
    SC_world_Right_Direction = spaceshipLocal * glm::vec4(SC_local_right, 1.0f);
    SC_world_Front_Direction = normalize(SC_world_Front_Direction);
    SC_world_Right_Direction = normalize(SC_world_Right_Direction);
    cameraLocation = spaceshipLocal * glm::vec4(cameraFront, 1.0f);
}
```