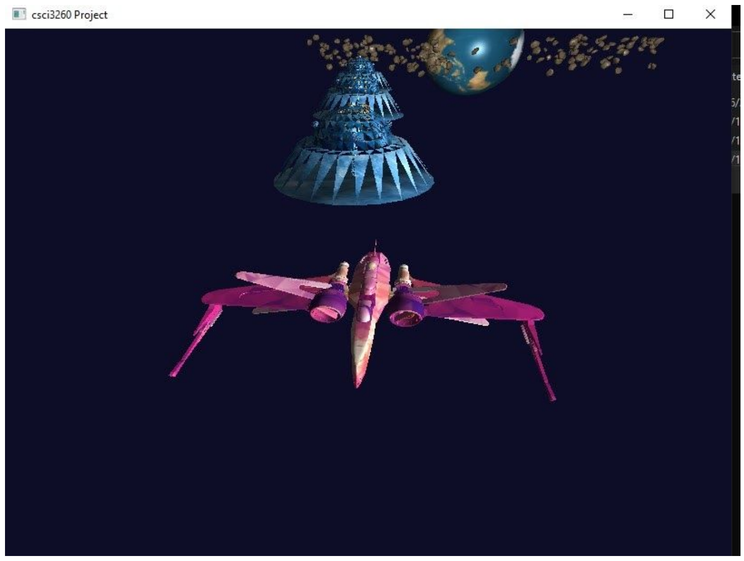
CSCI 3260 Group Project Report

Name :	Ng Chi Hon	Yau Chun Hong
SID :	1155116317	1155083264

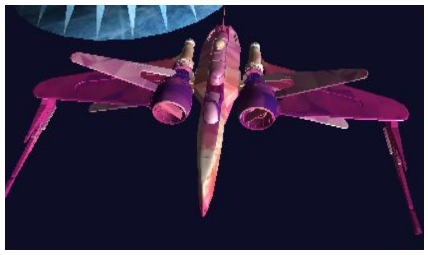
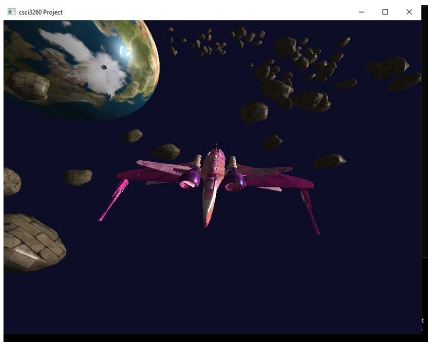
Section 1

The Basic Scene of the project

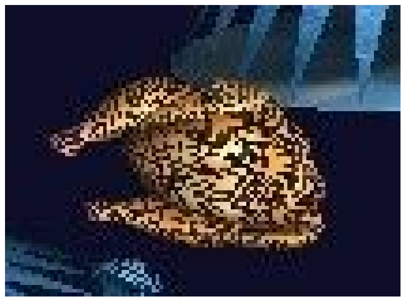
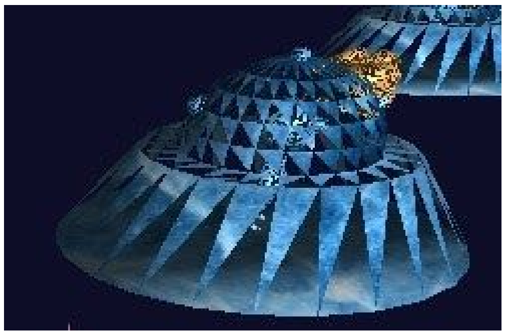


Section 2

The basic light rendering results on each kind of the objects.

The spaceship The Earth

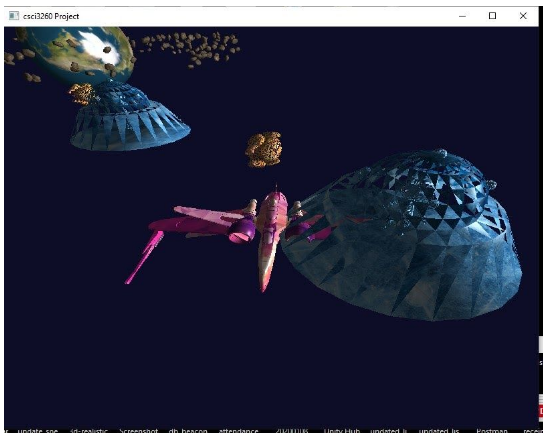



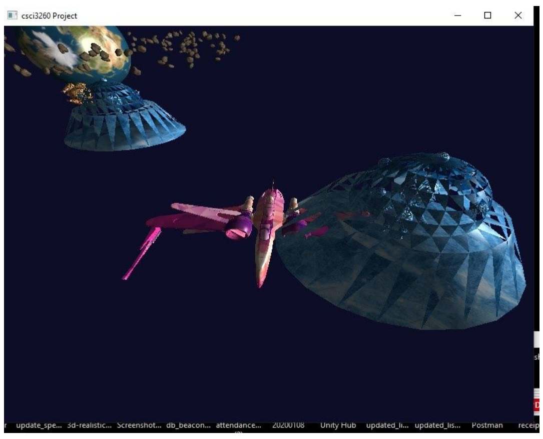
Chicken Space Vehicle with aliens

Section 3

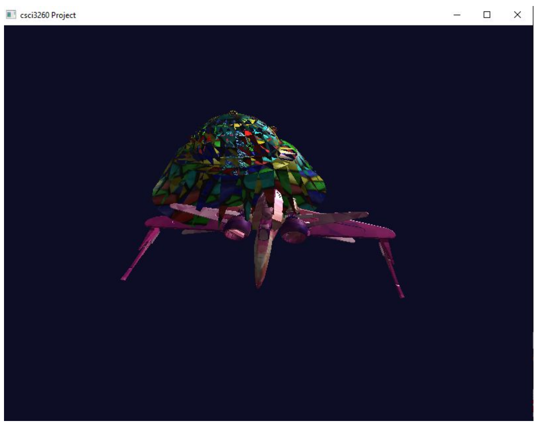
Frames that show that the spacecraft is

(1) collecting foods,

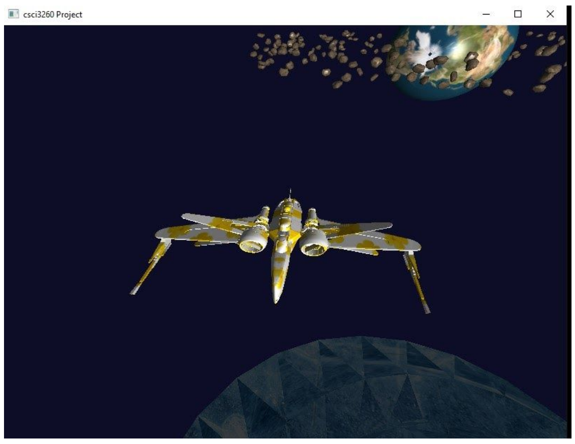



csci3260 Project
update_spe... 3d-realistic... Screenshot... db_beacon... attendance... 20200108 Unity Hub updated_li... updated_lis... Postman receip

(2) visiting the space vehicle,



(3) changing the texture of spacecraft after visiting finished.



Section 5.

Some brief and necessary descriptions of your implementation details.

The detail of an object will save as below,

```
typedef struct entity {
        int objID;
        int texture;
        float collisionRadius;
        int collisionHandler;
        vec3 location;
        glm::mat4 transform; //for aditional scaling or rotational visual transform
        int status;
        float orbitRadius;
        float radian;
        float orbitVOffset;
        vec3 orbitCentre;
        glm::mat4 scale;
}entity;
```

int initEntity(int objID, int texture, int x, int y, int z, float radius, int collisionHandler);
int initEntity(int objID, int texture, int x, int y, int z, glm::mat4 transform, float radius, int collisionHandler);

Both of the functions are used to initial the object position, size, texture of each object and save it in the EntytyList[]. If they do not have transform value, it will pass back the previous value of the object.

void initialiseEntities ();
After loading all the objects, the details of each object are mapped in EntytyList[], such as rock, sun, etc. It includes the size, color, position of each object.

int initRock(float radiusMin, float radiusMax, float vOffset, vec3 centre)
Rock is randomly create around the plant and the radius and centre is set in here.

void setupLight();
setup the basic light spot to each object.

void drawEntity(entity* e, glm::mat4 viewMatrix, glm::mat4 projectionMatrix);
Basically draw an object with the position and extract the entity object to call the following function.

void drawTextureObject(int index, int Texture, glm::mat4 transformMatrix, glm::mat4 viewMatrix, glm::mat4 projectionMatrix);
It would draw the texture of the specific object by object id and position.

int handleCollision(entity * primary, entity * secondary);
int handleExit(entity * primary, entity * secondary);

Both are to handle collision situations. when there are collisions, the function will check their collision status. Some disappear when there is a collision (like chicken rock and plants.) Some change texture, like aliens and alien vehicles.
when there are collisions and the collision object is status that change texture, it will check exit when they are not in collision, change back to original texture.
For collision of chicken, there is variable check to keep checking the collision time of chicken. When the variable is 3 times, the texture of the spacecraft will change to texture2.

int checkCollision(entity * e1, entity * e2);
This function check the spacecraft and all the other objects collide or not. This checking runs each time to make sure the collisions will be marked on each move. If the case reaches, it will run the above function by case. In the entity, there are location matrices for comparing.

void keyboard(unsigned char key, int x, int y)
Key D move to right
Key A move to the left
Key W move forward
Key S move backward
Key F move the light position to the left
Key H move the light position to the right
Key G move the light position upward
Key Tmove the light position downward

void PassiveMouse(int x, int y)
Move the angle of the view by mouse. It will calculate the of mouse movement distance and adjust the angle of view