CPT205 Computer Graphics (2020-21)

Assessment 3 – Discussion Questions and 3D Modelling Project

Your Name:Yumin Wang

Your ID Number:1825179

Your Programme of Study:Information and computer science

Date: 6th January

PART I. Discussion Questions

Question 1. Briefly describe the graphics pipeline, identify pipeline bottlenecks, and discuss possible ways to tackle the bottlenecks for enhancing the performance from both software (e.g. algorithms) and hardware perspectives.

Q1 Answer:

The graphics pipeline is an important concept in computer graphics[1]. It is actually a bunch of original graph data via a conveying pipeline, through various changes processing finally appears in the screen process.

First of all, the starting point of graphics rendering pipeline is the CPU, and the application stage. The main task of the application stage is to input the geometry that needs to be rendered on the screen into the geometry stage through the rendering pipeline.

Subsequently, the pipeline continues with the geometry. The main responsibility of the geometry stage is operations of each polygon and each vertex.It can be divided into the following functional phases: model and view transformation, vertex shading, projection, clipping and screen mapping. Model transformation types include rotation, translation, scaling, and mirroring operations. View is a transformation in computer graphics, and developers created cameras. Viewpoint and direction transformation are the main contents of viewpoint transformation. In vertex shading, shading is the operation of determining the effect of a light source on a geometry. Lighting and material are both related to shading. Especially, shading can be performed on a per-vertex or per-fragment basis. When it comes to projection, it is divided into orthogonal projection (parallel projection) and perspective projection. The next step is clipping. For partially primitives that are located in the visual vertebral body, they are clipped. And some objects outside the window are clipped. Screen mapping is the final step of the geometry stage. There are three types of screen mappings: perspective division, viewport transformation, and picking.

The final stage is the rasterizer stage. Rasterization, also known as scan conversion. It converts 2d vertices from screen space to pixels on the screen. This stage is divided into 4 several functional stages: triangle setup stage, the stage of triangle traversal, the pixel shading stage, inverse merging phase. It is worth noting that, in some versions, rasterization is implemented before shading.

The graphics rendering pipeline is a single pipeline from the CPU to the GPU, so each stage of the pipeline can be a bottleneck . Identify the bottleneck. For each stage in the pipeline, vary either its workload or its computational ability. If performance varies, you've found a bottleneck[2].

To solve the bottleneck, we should consider both software and hardware. On the one hand, in the software,abundant algorithm resources can optimize the pipeline. For example, an optimization technique suitable for the rasterization stage is to perform appropriate texture compression. On the other hand, in terms of hardware, different graphic pipeline designs should be implemented for different hardware performance levels. For instance, resource locking can be reduced and the number of batches can be minimized. You can try to avoid accessing resources that the GPU is using during rendering to reduce resource locking. Use the Shader branch to increase the size of individual batches to maximize the size of batches[3].

In conclusion, The graphics pipeline is an effective rendering tool. Although bottlenecks can occur at any stage, there are ways to optimize them.

Question 2. Discuss applications of computer graphics incorporating artificial intelligence. This should cover techniques, key issues and possible solutions with directions for future development.

Q2 Answer:

With the development of technology, computer graphics and artificial intelligence technologies are increasingly optimized. However, there are still some problems in computer graphics, and artificial intelligence can provide satisfactory solutions [4]. Using artificial intelligence can improve the modeling and rendering process.

Rendering is one of the most difficult aspects of computer graphics, but with advances in artificial intelligence and neural networks, this problem can be improved. For example, an RNN+Autoencoder neural network can denoise an image sequence rendered with unconverged ray tracing. The purpose of this technique is interactive rendering, first using an optimized path tracker to generate our noise input images,then compute samples, and evaluate eventually. This technology uses artificial intelligence to achieve global illumination in computer graphics.

On the contrary, the technology of computer graphics can also help improve the research level of artificial intelligence. In the field of computer graphics, the details of the semantic annotation of images in AI games is a challenge, but we can hash different rendering resources, such as geometry, texture and shaders. The annotation can through the game to the graphics hardware, and the generated object signature runs through the entire scene and game session[5]. For example, by using the presented approach, people have created pixel-level semantic segmentation ground truth for 25 thousand images extracted from the game Grand Theft Auto V. The labeling process was completed in only 49 hours[5].

In general, there is a lot of space for cooperation between computer graphics and artificial intelligence.

References

[1] "Overview of the Graphics Pipeline,"2017, Fragment.

[2] C.J.G.G.Cebenoyan , "Graphics Pipeline Performance," ed:Boston, MA: Pearson ,2004, pp. 473-486.

[3] E.Haines , N.Hoffman and T.Moller, Real-Time Rendering ,1997.

[4] D.Plemenos.(2000,January). "Artificial Intelligence techniques for computer graphics." [Online].Available :https://www.researchgate.net/publication/228717511_Artificial_Intelligence_techniques_for_computer_graphics

[5] S.R.Richter, V.Vineet, S . Roth, and V.Koltun, " Playing for Data: Ground Truth from Computer Games," in European conference on computer vision, 2016, pp. 102-118: Springer

PART II. 3D Modelling Project

Written Report

Introduction

The Assessment uses Visual Studio + OpenGL technology to simulate a three-dimensional courtyard and surrounding environment, including objects such as grass, wood floor, floor tiles, rooms, swimming pools, balloons, courtyard walls, path in the yard , bridges to the river, river, shopping mall, trees, sofas, TV sets, beds, tables, clouds and the sun.

In addition, we can use some graphics techniques. For example, create geometry, transform perspectives or viewing , lighting and materials, texture mapping and keyboard interaction.

Design and Features of the Assessment 3

First, set the cube() function and trapezoid() function to assign values to the vertex arrays of cuboid and trapezoid according to the input coordinate parameters, and then create the build () function and build2() function to draw faces according to the vertex coordinates to draw faces, which are spliced into a cuboid and a 3d trapezoid.

Using cube() function and the trapezoid() function to build the courtyard and the surrounding environment.

- glBegin(GL_POLYGON): Drawing surface based on vertex coordinates.
- trape[ ][ ], rect[ ][ ]: Assign values to cuboid and trapezoid vertex arrays.
- glBegin(GL_QUADS): Draw quadrilateral .
- glutSolidSphere: Draw the ball.

  Using the basic figure above, draw what needs to be built.

- Draw grass and floor tiles and river:

Map the texture with glBindTexture(). GlTexCoord2f(); and glVertex3f(); to determine the plane position of the map.

- Draw the balustrade:

  Build a cuboid as a balustrade, using glBindTexture(); to map the balustrade.

- Draw the fence of the yard :

For the courtyard wall, five cuboids were used to draw the main body. The courtyard has a gate, so the lower wall is divided into two parts. Finally, using glBindTexture() to map the wallpaper on the wall. To draw the iron gate, first draw the solid of the door by cuboid() and glRotatef() to change its direction. The door is then textured with an iron door by using glBindTexture().

- Draw the shopping mall :

The shopping mall is a cuboid that uses glBindTexture() to texture the five sides of the mall. Put the door texture on the side of the malls, a fountain texture on the front, tiles texture on the back and a helicopter takeoff texture on the top.

- Draw the lamp on the wall:

To draw the lights on the wall, cuboid(),Build () and for{} loop are firstly used , then the glutSolidSphere() is used to draw the bulbs, finally I using glTranslatef() to combine the bulbs and the poles.

- Draw the sun room:

The sunroom is in the yard. The sunroom consists of a glass roof and two walls that are textured by using the glBindTexture().There are two chairs and a table in the sunroom. The four cuboid legs of the table and the top of a cuboid make up the table. To construct two chairs, I using trapezoid() and biuld2 () functions.

- Draw the roads:

Use the for() loop and cuboid to draw the roads. One road from the house to the door, and one from the land to the river.

- Draw swimming pool:

The swimming pool consists of four walls and water. A balloon built by glutSolidSphere() floats up and down on the water.

- Draw house:

The house has white walls on all sides. The roof is a cuboid with adjustable transparency. The glBindTexture() is used to put a wood floor in the house. Besides, the door and the window of the house are on the front wall of the house. Lastly, there is a bed, a mahjong table, two sofas and a big projector in the room. Cuboid(), built() and glBindTexture() are used. Glcolor4f() can adjust the transparency of the roof.

- Draw sky:

There is a sun and two floating clouds in the sky. The cloud is made up of three spheres. GlTranslatef() is used to change position and make the clouds float.

- Draw tree:

The trunk of the tree is a cuboid. The leaves are made of four spheres.

Texture mapping

Firstly, the texture object is defined and the BMP image required for texture drawing is read in the texture binding function. Then the three steps of starting texture, binding texture and closing texture are carried out.

Rendering includes the implementation of reflection and refraction effects, the addition of highlights, translucency and so on. Through the timer to achieve the automatic sink or fall of the ball and clouds.

Interactive instructions.

The interactive control of the keyboard is realized by the inputKey() function, whose parameter is the key pressed by the keyboard.

1/2/3/4 control the movement of the cloud2 and ball in the pool is controlled by changing the offset bx and bz of the cloud 2and ball coordinates.

W/S/A/D controls the observation perspective of the 3D scene by changing the rotation variables rotate_x and rotate_y of the scene axis.

Q/E controls the zooming in and out of 3d scenes, which is achieved by modifying the zooming array sca [ ].5/6 changes the transparency of the roof, which is achieved by changing ln. 7/8 changed the perspective position by changing tra[ ].

Keyboard Interaction:

Key p: Exit and close the window.

Key q: Zoom in

Key e: Zoom out

Key w: Rotate in the positive direction about the X-axis

Key s: Rotate in the negative direction about the X-axis

Key a: Rotate in the positive direction about the Y-axis

Key d: Rotate in the negative direction about the Y-axis

Key 1: The balloon is floating forward along the first quadrant bisector of the X-Z plane, and cloud 2 is floating backwards along the X-axis.

Key 2: The balloon is floating backward along the first quadrant bisector of the X-Z plane, and cloud 2 is floating forwards along the X-axis.

Key 3:The balloon is floating forward along the second quadrant bisector of the X-Z plane.

Key 4: The balloon is floating backward along the second quadrant bisector of the X-Z plane.

Key 5: Increase the opacity of the roof

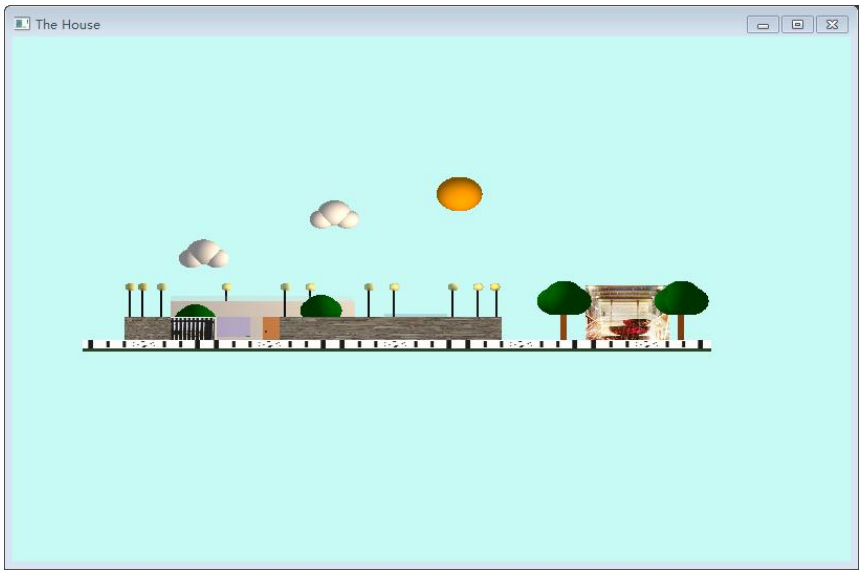
Key 6: Decrease the opacity of the roof

Key 7: Move the whole project forward along XYZ coordinate system

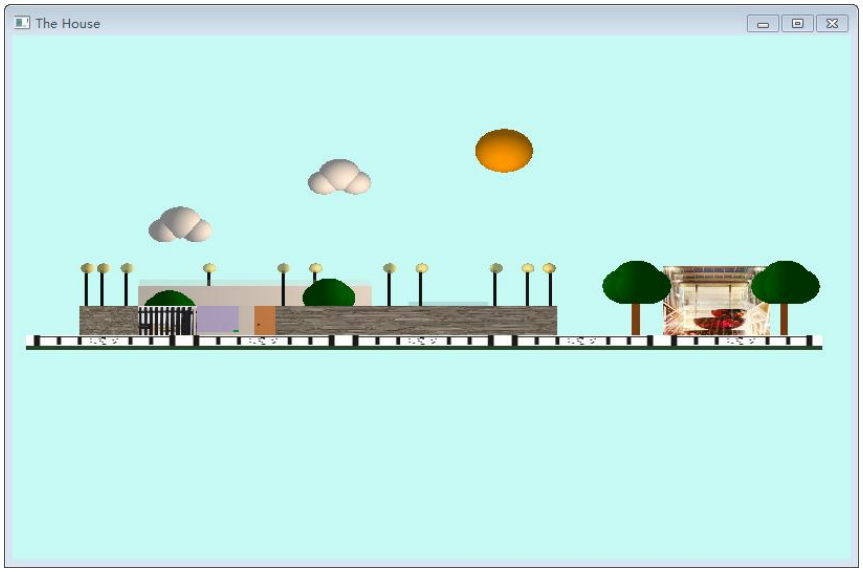
Key 8: Move the whole project backward along XYZ coordinate system

Screenshots :

- original state:

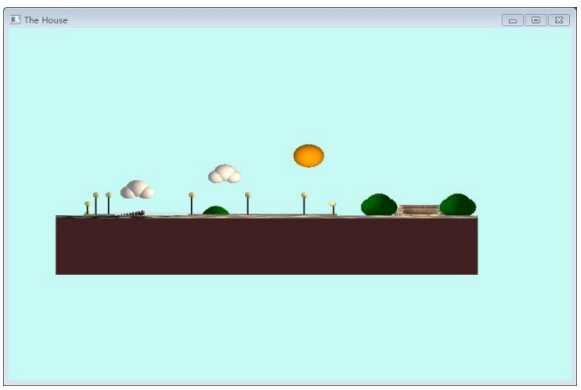


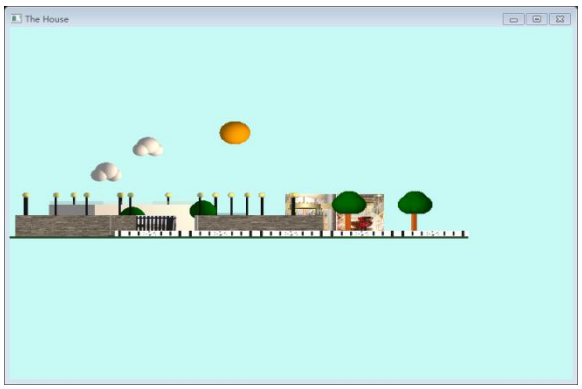
Press q :

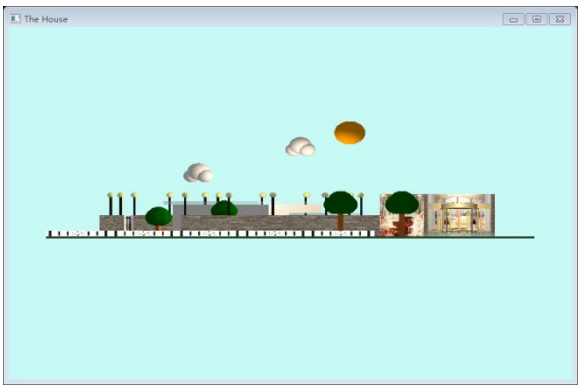


Press e :

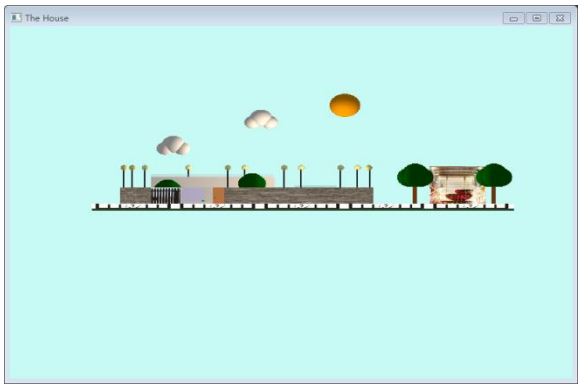
Press w/s/a/d:

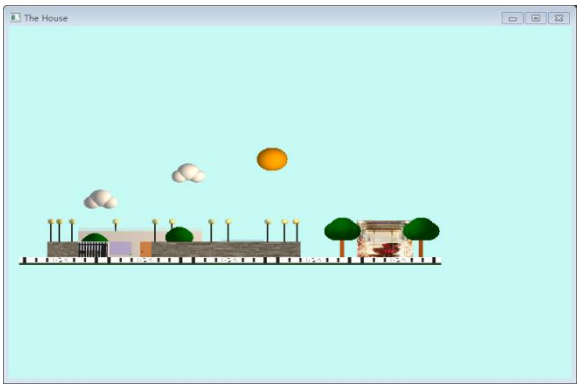






Press 7/8:





Press 5/6: