MSIM 441/541 & ECE 406/506

Computer Graphics & Visualization

Programming Assignment Three

Texture Mapping and Imaging Pipeline

Jeffrey McAteer

Modeling & Simulation

jmcat001@odu.edu

540-842-6017

Virtual

2022-12-12

**Introduction**

Dear Mr. Thompson, as you know I have recently been demoted from Chief Information Officer after our latest Project Lead, Mr. Jones, completely missed a number of important customer demands. During his leave I have implemented these requests, and if you were familiar with the project before there are several new things that should be pointed out about the current system. I’d also like to thank you for stepping up to fill the role of Project Lead for our company’s safety simulation program, Traffic Simulation.

Customers had a need to place advertising within the simulation, such as Broderick Dining Commons. We have implemented a sign they requested and added animated lights around it. Several other signs have been added, and the requested snapshot utility is complete. Use the ‘s’ key to save a snapshot to the “snapshots” folder in your working directory. A safety measure has also been added, if the car is facing NS and is within the stoplight box when the NS light is red a snapshot will be recorded. The same thing is implemented for the EW light, but only one snapshot is taken per traffic infraction. I know there are some local cops who may want that modified to take a video, so please keep an eye on those requirements.

Good luck with what’s left of the team, and welcome to the simulation division!

**Program Design**

See the doxygen/ folder in JeffreyMcAteerProject03.zip.

**Results**

**3.1**

The same ‘\r’-ignoring bug was patched in ObjModel.cpp so .obj files could be read in, and ImageMagic is dynamically invoked to convert .jpg texture files to a binary .ppm file. The binary format (P6) is used because it is smaller than the ASCII (P3) format.

The license plate file was modified in GIMP and the VA ODU plate template was obtained from [https://www.dmv.virginia.gov/vehicles/#splates/info.asp?idnm=ODM2](https://www.dmv.virginia.gov/vehicles/" \l "splates/info.asp?idnm=ODM2)

**3.2**

2 static billboards were added, and 2 animated billboards are created from a directory of frame files. The additional methods Billboard::IsAnimated, Billboard::ActiveTextureNumber, and Billboard::SetDelayMs were added to support a dynamic texture number during rendering and a frame delay duration.

The y-axis value of Billboard::location is used as the elevation of the billboard itself, all pillars begin at y=0 and extend until Billboard::location.y, which is where the billboard rectangle begins.

The behavior of Billboard::SetSize was updated to save width and height, and auto-calculate one dimension if it is -1.

glBindTexture is used to map a chunk of pixel memory to a number which the GL runtime may use later for rendering without having to re-read the pixel data over again. This is especially important because the pixel data may no longer be in main memory – they likely reside within the GPU hardware.

GlTexParameteri sets texture parameters; our program sets the GL\_TEXTURE\_WRAP\_\* parameters to repeat images at their edges. GL\_REPEAT causes the texture to be duplicated until the edge of the face is rendered. GL\_CLAMP was deprecated from OpenGL version 3 in 2008, and fully removed in 2009 for OpenGL 3.2. Instead of GL\_CLAMP, modern options are

* GL\_CLAMP\_TO\_EDGE paints the edge pixels from the end of texel space over the remaining area, looking as if the edge was simply stretched across the remaining portion of the rectangle.
* GL\_CLAMP\_TO\_BORDER does not paint pixels outside the texel area, instead using the given color as a solid background.

glEnable(GL\_TEXTURE\_2D) is required because OpenGL can paint textures from many types of bitmaps, including 3d voxel bitmaps. By telling the runtime the geometry we want to use it can setup render pipelines for planar images. Turning them off is important so the runtime can differentiate one set of draw commands from another, otherwise everything would share the same transformation matrix.

****

****

****

**3.3**

WASD controls were removed from this version of the project to support the ‘s’ key for snapshots. The c++17 chrono library is used to get the system time and format it into “%d-%m-%Y\_%H-%M-%S” for use in the snapshot filename.

PPMImage::AllocateMemory must be called before manipulating PPMImage::image to dynamically allocate pixel memory and save the image width+height in the class.

The PPMImage destructor frees PPMImage::image if it is not null. C++ uses a global implicit allocator which, among other things, remembers array dimensions so users do not have to remember how many items are in a dynamic array. Because allocating memory for a single object and looking up an address → length are two different operations, c++ has both a delete keyword and a delete[] keyword which perform the two behaviors. delete[] is an instruction to ask the allocator for the length of the array when it was created, then call the type destructor for every element at each index of the array. *delete[]* should line up with calls to *new[]*, and *delete* should line up with calls to *new*.

PPM images have a binary and ASCII variant. The binary variant packs pixel data as RGB unsigned chars while the ASCII variant writes pixel data as RGB space-delimited text. PPMIMage::WriteFile() writes ASCII if given “P3” as a type designator by opening the ofstream file in text mode and using stream operators to represent the unsigned characters as integers. When given “P6” the header is written and then pixel bytes are directly written to the file in binary mode by casting the unsigned char\* pointer to char\* (signed char pointer).

Traffic violations are detected in the main update() routine. The car direction is determined by boxing the carDirection into NS and EW 90-degree areas, and the position of the car is used to determine if it is within the stoplight box. If the light associated with the carDirection is red, a snapshot is taken and the words “Illegal car position!” will be added to the HUD. This clears when the car is no longer violating the light, and only one snapshot per violation is saved.

**Conclusion**

Adding signs and animation to Traffic Simulator was a great way to experience video game design and opened the path to realistic simulations using textures for detail. One difficulty I had was managing pixel data in-memory; If you do not explicitly tell OpenGL how the memory is laid out using glPixelStorei(GL\_UNPACK\_ALIGNMENT, 1); then only some images will render correctly; others may have an alpha value or GL may assume only 2 bytes per pixel, which skew some of the images (see below for a broken render, the speed limit and dining signs are broken). I’d love to add a physical contact detection to add barriers to the world, but that will have to wait until another semester.

