3D City

Grace Cassidy and Khalil Elbaggari
6.111 Project Proposal
Fall 2015

1. Overview

The project aims to render a 3D city as an immersive map, to serve as entertainment for a user to explore an unknown world. This city would be made of wireframes initially, and later from polygons, and the user would move from a first person perspective. The user's movement would be controlled with the buttons on the labkit. The city would first be implemented with wire frames, and then progress with shaded polygons, and then background sound. This project will involve saving the frames to RAM and accessing the frames with ZBT to write and read quickly. Additionally, the project involves computing the transformations to render the 3D images in 2D. The transformations are computationally intensive and require four multiplies, a divide, and trigonometric calculations for the building angles. The sound will involve storing the .coe files in ROM and later sending the appropriate sounds to the AC97 chip.

There are more features that we could add to the city landscape if we achieve these initial goals. The stretch goals include adding textures to the polygon surfaces, implementing shading of buildings, implementing a z-buffer to determine which surfaces of the building should be displayed, and adding acceleration to implement running or walking. The benefit of these stretch goals is to make the virtual city appear more realistic and improve the user's experience. The running and jumping features would make the game more exciting by providing the user with options on how to explore the city.

2. Design Decisions

The project was designed to have several, increasingly complicated stages of implementation which improve the user's experience by making the virtual city more realistic. Breaking the project into stages will allow us to have a working project for the final project regardless of how complex we are able to make the project, and the high level block diagram is shown in Figure 1. The foundational blocks of the project include reading and writing to RAM using ZBT to display a wireframe city on the VGA display. In this basic implementation, the user will also be able to navigate throughout the city. The movement throughout the city will be controlled by the buttons on the labkit, and later controlled with a joystick if one is available to provide for a more natural user experience.

The next stage of the project involves replacing the wireframes in the imagined city with polygons. This stage will make the city look more realistic, but the polygons will also allow the project to eventually include shading of buildings to give the effect of a sun or other light source.

The next goal of this project is to implement physical laws into the game, such as acceleration of the user and velocity control. This would allow the user to feel like they are actually walking, running, or jumping when navigating the city.

If these steps are accomplished, the final stretch goal is to store various sounds in the ROM. These sounds would be used for background music, collisions with buildings, listening to cars as the user approaches intersections, and different sounds for the user while running or jumping. The background sounds would improve the user experience and make the virtual city appear more realistic.

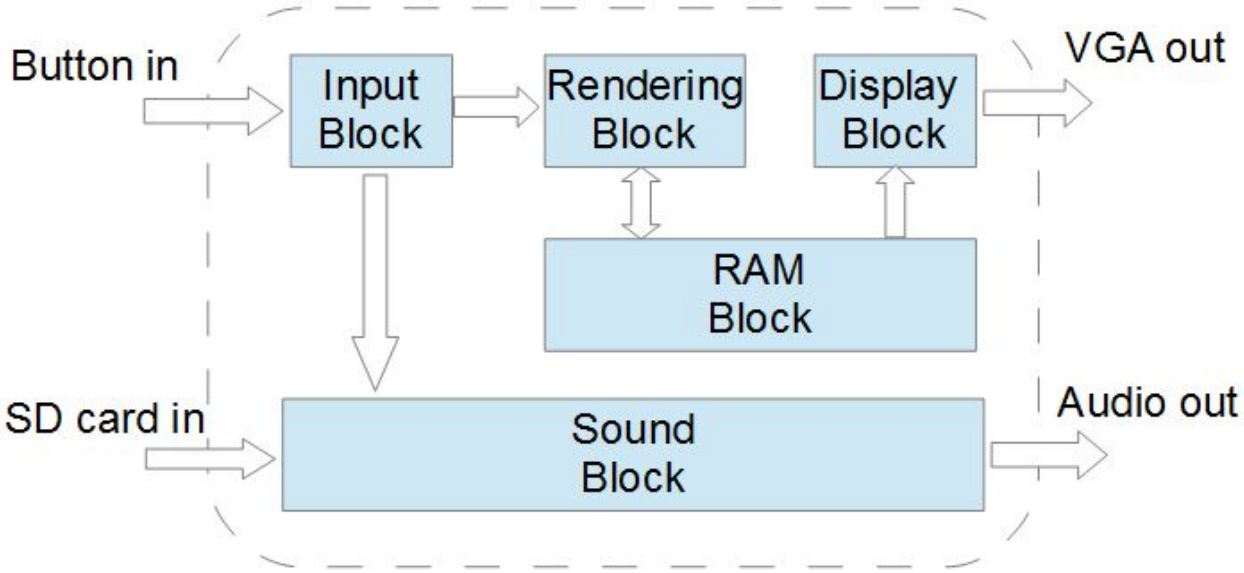

Fig. 1. High-Level Block Diagram

3. Implementation

The project will be implemented first on the labkit and later the Nexys 4 board if all other stretch goals have been accomplished. The labkit has known, working modules for ZBT SRAM unlike the Nexys 4 board. The Nexys has more resources, so it would be possible to render more complex scenes at a reasonable frame rate.

3.1 Input Block

In order to move the user through this virtual city, the buttons on the labkit will modify the user's location in the city. The buttons will move side to side (left and right buttons) or translate (up and down buttons) the user, trigger events in the virtual city, or reset the user's position and locations of all objects. These locations and positions will be passed to the rendering module to be processed for eventual display on the VGA monitor. One of the last stretch goals will be using a GameCube controller to move the user through the virtual city and make movements more natural. This will also allow for more varied actions such as looking to the left while walking forwards. With the buttons, the user has to move in the direction they're facing.

3.2 Rendering Block

The rendering block contains the rendering pipeline: edge module and matrix transformation. The edge module contains a counter to look up the IDs of the edges and their associated 8 bit vertices in 3D space. The number of bits, corresponding to spatial resolution, can be increased after testing. Since many of the operations in the matrix transformation module are computationally expensive, thus slow, the module will be pipelined. The first part of the transform pipeline is computing cosines and sines from a fourth order, possibly fifth order, polynomial that approximate a quarter period of a cosine (fourth order) or sine (fifth order). The fourth order polynomial is easier computationally, but the fifth order has lower overall error. These calculations can be done in parallel and used in the transformation matrix. The transformation matrix requires three multiplications, and to generate the 2D projection on the VGA display, another multiply and divide are necessary. To keep the ID of the edge and transformed vertices associated, the ID needs to be passed down the rendering pipeline.

Once the rendering block is tested, constructing the buildings from polygons will make the virtual city appear more realistic. More computations are needed to implement the polygon faces, which may make the frame rendering slower. To implement the polygons, another point needs to be transformed from 3D to 2D, but the biggest computation will be determining the angle the surface forms with the light source.

3.3 RAM Block

To store the vertices of the buildings to be displayed on the VGA display, the RAM block will allow the system to read and write into the memory on the labkit. The RAM block includes three main modules. The first is the read module, the second is the write module, and the third is the scheduler to control the timing for these modules. These are separate modules to allow both partners to work on separate modules simultaneously. ZBT will be implemented within these blocks, and the sample ZBT code from the 6.111 website for the labkit will be used in the initial implementation.

3.4 Display Block

To display the virtual city on a VGA display as the user navigates throughout the city, a display block will be incorporated into the project. The display block will take the data read from the ZBT SRAM and will display the image on the monitor using 24-bit XVGA. Since accessing RAM takes longer than displaying the image on the display, there will be two frame buffers. One frame buffer will be used to display the current frame, and the other frame buffer will be used to read data from the RAM for the next frame to be displayed. We will try to implement the display block with a 65 MHz clock, but a slower clock, around 15-30 MHz, might be necessary to account for time to access the RAM.

3.5 Sound Block

In order to play sound while navigating through the virtual city, a sound block will be implemented in the project. The sound block takes the sound ID and play sound signal from the input block to select when to play a certain stored sound. Sample MATLAB code from the 6.111 website to convert .wav sound files to .coe files will be used to store the sounds in BRAM. When the play sound signal is high, the sound module will access BRAM to send the correct sound information to the AC97 chip. The AC97 chip samples at 48 kHz, so depending on the stored sound's frequency, the sound may need to be upsampled or downsampled. The AC97 chip will send the output audio to a headset or speakers.

4. Timeline

Figure 2 is a chart displaying the proposed schedule for the project with elaboration on the steps below. In the chart, Khalil will complete modules labeled "a," and Grace will complete modules labeled "b." During the first three weeks of the project, we will be able to work individually on the modules. The final weeks include testing and integration of the modules, requiring a collaborative effort.

	Week of 11/2	Week of 11/9	Week of 11/16	Week of 11/23	Week of 11/30	Week of 12/7
1) Finalize project ideas						
2a) Rendering module, input module 2b) Display module						
3a) Writing to RAM module 3b) Reading from RAM module						
4) Testing						
5a) Sound module 5b) Replace wireframes with polygons						
6a) Integration of modules and testing 6b) Stretch goals						
7) Demo and final checkoff						

Figure 2: The proposed schedule.

Description of work during each week:

1. Finalize project ideas and discuss project details with Prof. Hom.

2. Make the rendering and display modules. These are the most complex modules and we expect these modules to take the most time to design and build. They will be completed early to ensure the project will be completed in time.
3. Make the modules to write to RAM and read from RAM. The scheduler for the write and read modules will also need to be completed during this time by the person who finishes their other module first. After the RAM block is completed, the modules will be able to be integrated to test the system.
4. Test the existing modules individually.
5. Add sound effects to the city with sounds stored in the labkit's ROM, replace the wireframes with polygons, and start working on any stretch goals.
6. Integrate all modules, test, and continue working on the stretch goals.
7. Demonstrate the final project to staff and complete the checkoff.

5. Testing

After completion of each module, the module will be tested with Verilog test benches to help reduce errors and aid in debugging. Test benches will also be created to test groups of modules to further reduce debugging time. Testing each module individually will make it easier to find errors than waiting to debug until all modules have been completed and integrated.

The VGA display will also be used to help debug the display modules such as the reading from RAM, construction of the wireframes, and construction of the polygons with shading. If there is time, the testbenches and the display will be used to debug the optional physics modules including jumping, walking, and running modes.

6. Resources

In addition to the 6.111 labkit, this project will need a GameCube controller and a GameCube socket. The controller and socket would be used to implement one of the stretch goals of switching the navigation control from labkit buttons to a more natural joystick control. Both of these are available on ebay, but since the GameCube controller will not be modified, the controller could also be borrowed from someone. A pack of four GameCube sockets is available on Ebay for $6.77.

7. Conclusion

Overall, this project presents many challenges. Both the memory control and 3D to 2D transformations will pose the greatest implementation obstacles. In addition to incorporating features learned in the previous 6.111 labs, the project also requires use of ZBT SRAM and more intensive calculations such as multiplies and divides. Implementation of these features will further our knowledge of digital systems skills. The basic implementation involves a virtual,

wireframe city, but after success of this initial goal, there are many options to add more features and improvements to the project. These additional features include velocity control, sounds, and shading of buildings.