# PA10: Pinball

By Team Tessellation
Natalie Arnold
Terra Williams

November 16, 2018

# Table of Contents

Sections

# Overview

# Dependencies, Building, and Running

## Dependency Instructions

For both of the operating systems to run this project installation of these three programs are required GLEW, GLM, and SDL2. (http://glew.sourceforge.net/) (http://glm.g-truc.net/0.9.7/index.html) (http://glm.g-truc.net/0.9.7/index.html)

This project uses OpenGL 3.3. Some computers, such as virtual machines in the ECC, cannot run this version. In in order to run OpenGL 2.7 follow the instructions at Using OpenGL 2.7 (https://github.com/HPC-Vis/computer-graphics/wiki/Using-OpenGL-2.7)

This project uses the Assimp library. To download and install Assimp. (http://www.assimp.org/)

This project uses the ImageMagick 7.0.8. To download, install, and build for linux, follow the instructions at ImageMagick. (https://linuxconfig.org/how-to-install-imagemagick-7-on-ubuntu-18-04-linux)

This project uses the Bullet Physics Library, installation instructions can be found here for Bullet. (https://github.com/bulletphysics/bullet3)

## Ubuntu/Linux

```
sudo apt-get install libglew-dev libglm-dev libsdl2-dev
```

## Mac OSX

Installation of brew is suggested to easily install the libs. Ensure that the latest version of the Developer Tools is installed.

```
brew install glew glm sdl2
```

# Extra Credit

1. Spotlight following the ball
2. Adjusting ambient color
3. Bumpers are bouncy and change color when hit
4. Plunger intensity changes via keyboard
5. Game Replay

# User Manual

## Building and Running

To build this project there are two options. One is to use CMake which makes including new libraries easier, and handles new files added automatically to the src and include directory. CMake is a small new learning curve but makes things easier in the future. The second option is to use the provided Makefile which is used as usual.

Running the make in a separate directory will allow easy cleanup of the build data, and an easy way to prevent unnecessary data to be added to the git repository.

### CMake Instructions

The building of the project is done using CMake, installation with apt-get or brew may be necessary. Later use with CMake and Shader files will be require the copy of a directory where those files are stored (ex. shaders). To do this in the add_custom_target function place
COMMAND ${CMAKE_COMMAND} -E copy_directory ${PROJECT_SOURCE_DIR}/shaders/ ${CMAKE_CURRENT_BINARY_DIR}/shaders

```
mkdir build
cd build
cmake ..
make
./PA_Pinball
```

### Makefile Instructions

The makefile works as expected and must be updated with new files added in.

```
mkdir build
cd build
cp ../makefile .
make
./PA_Pinball
```

## Ubuntu.cse.unr.edu

OpenGL 3.3 will run on the ubuntu.cse.unr.edu website. To do so follow the build instructions, but when running the Tutorial executable use this line to execute.

```
/usr/NX/scripts/vgl/vglrun ./PA_Pinball
```

# Keyboard Inputs

## Board Controls

| Action | Control |
|---|---|
| Left Flipper | Z |
| Right Flipper | M |
| Launch Ball | ↓ |
| Reset Ball | Menu |

## Camera Controls

| Action | Control |
|---|---|
| Enable/Disable FPS Camera Mode | E |
| Move Forward | W |
| Move Left | A |
| Move Right | D |
| Move Backward | S |
| Move Up | SPACE |

# Lighting Controls

Our lighting controls include:

- Switching between Phong and Gourand shading
- Adjusting ambient strength
- Adjusting ambient color (RGB)
- Turning a spotlight on and off
- Switching between a spotlight with a sharp edge and a spotlight with a soft edge
- Adjusting spotlight size
- Adjusting spotlight brightness
- Adjusting specular brightness on the bumpers, ball, flippers, and board

All of our lighting controls can be found in the menus labeled appropriately.

# Figures


**Figure 1. Pinball board at start state.**


**Figure 2. Pinball board lights out**

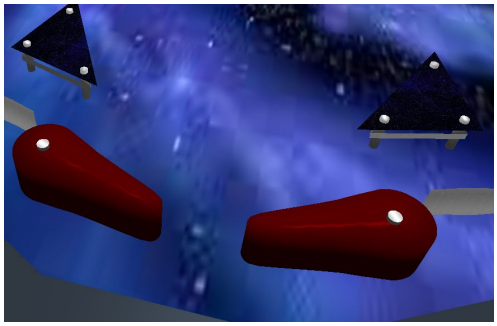

**Figure 3. Flippers rendered with Phong Shader**

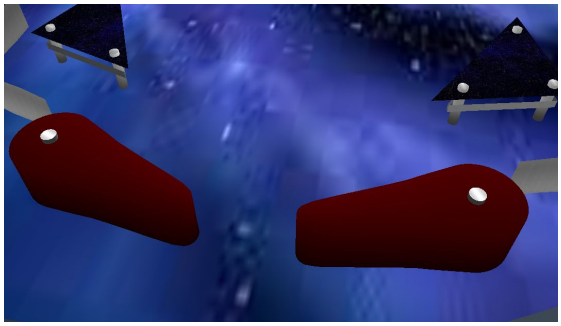

**Figure 4. Flippers rendered with Gourand Shader**


**Figure 5. Bumpers (specular highlights and colored lights)**


**Figure 6. Bumpers (lights out and colored lights)**


**Figure 7. Spotlight following the ball.**

# Technical Manual

## Issues

Bullet was, of course, an absolute delight to work with. It took quite a while to get triangle meshes working and even then we had to slow everything down in order to ensure bullet could detect collisions quickly enough so as to keep our ball from getting stuck in various objects. It was difficult to understand when the gourand shader was and was not working, as specular highlighting was practically nonexistent on most of our objects. We struggled to implement proper game logic, mostly due to complications using bullet. Figuring out a way to animate the flippers as triangle meshes was very difficult and the ball still sometimes moves too quickly for Bullet to detect the collision in time.

## Things We Could Have Done Differently

We would have liked to create a configuration file and generally cleaned up the code since it is currently quite a mess. We also wanted to animate the plunger but were not able to. The code in general could be much neater, there were random objects and flags everywhere.

## Changes

Since Wednesday, we animated the flippers, assembled the whole board with backsplash, bumpers, and plunger. The point light sources on the bumpers were added. The ability to put the ball into play and game logic were incorporated as well as the varying force applied by the plunger.