Matthew Fox, Michael Kelessoglou, Evangelos Taratoris
6.111 Project Proposal
2 November 2013

Virtual Pitch and Catch

Overview

The game of pitch and catch resonates deeply within many cultures as a shared experience. Whether it be strengthening a parent-child relationship, passing time with a friend, or training for the upcoming season, people across the world engage in this game on a regular basis. On another strain, the tech community and the gaming industry as a whole are moving towards virtual reality as the next avenue for the new generation of games. One only has to look as far as the Nintendo Wii, XBox Kinect, Oculus Rift, or Playstation Move to see the trend of virtual reality sweeping the gaming industry.

Our idea for virtual pitch and catch lies at the intersection of these two cultural practices, the analog game of pitch and catch and the digital storm of virtual reality. We hope to bring the interactive game that has been so beloved for over a hundred years to the twenty-first century by creating a virtual interface for two players to throw and catch a digitally produced ball. We will use trackable gloves with multiple types of sensors to track and monitor the user's actions while playing the game, allowing us to show the game's progress on a monitor and simulate the actions of the virtual ball.

Furthermore, the implementation of the pitch and catch game will provide a framework that is highly configurable for interesting applications. Some of these include a virtual basketball game, a three dimensional pitch and catch game, a blue screen addition to the original game, and many more. This flexibility that the implementation allows supports the creation of this project that, in its basis, is already strong.

Design Decisions

In our exploration of this project, we have laid out an overall block diagram detailing the modules of our project and how they interconnect (See Figure 1). The first design decision that we have made is to create a "smart glove" for each player that will inform our system of the hand's acceleration and openness. To detect whether the hand is opened or closed, we have decided to use a pair of flex sensors on each glove with surrounding circuitry to give a clean output to the module. The glove will also have an accelerometer so that we have some reliable data for tracking the glove.

The next major portion of the project is the physics module and rendering. We have made the design decision of making a two dimensional game as to simplify the hand

tracking and the physics involved but mostly to simplify the on-screen rendering. Because of this decision, we will be able to provide a real-time visual approximation of the ball's location to inform the players of the progress of the game.

The last major design decision that we've made prior to starting the project is to track the bright color of each glove through a camera near to the glove. This data combined with the data from the glove's accelerometer should give us a very reliable and robust way of tracking the player's hands throughout the game. In order to allow the cameras to track each player's hand, we've made the decision to fit two cameras to this project, one in front of each player, with the image processing for each done on a separate labkit to avoid running into issues with processing power. Overall, the decisions we've made should allow us to complete a working version of a fun and compelling game.

Implementation

Glove Module – Matthew Fox

The glove and its surrounding circuitry are one of the major modules of this project. We are planning on fitting two gloves that will be tracked by the FPGA camera provided (one for each player). Because of this, we have chosen to buy a pair of bright orange gloves to serve as a bright indicator that the camera can track. These gloves will be fitted with a few sensors, namely two flex sensors and an accelerometer each. The flex sensors (purchased from sparkfun - https://www.sparkfun.com/products/10264) create a variable resistance that increases with bending. We plan to fit these to the outside of the index and ring fingers of the glove so as to discern when the hand closes. In the datasheet of these flex sensors, a suggested circuit is provided for producing an output voltage that will inform us of the level to which the hand is closed. We plan on using an operational amplifier to push this output to the rails so that we can have a binary output from this module of the hand's openness. We have chosen to have two flex sensors per hand to approach assuredness that the hand is intentionally being closed rather than partially flexed. By using a NAND gate as we did in Lab1 (74LS00), we will only accept a hand as closed when both signals are low, showing an intentional closing.

In addition to the bright glove and the flex sensors, we will outfit each glove with an accelerometer to capture the acceleration of the glove leading up to a throwing event. We have ordered two accelerometers and breakout boards (purchased from sparkfun - https://www.sparkfun.com/products/10955). These small boards will be mounted on the back of the hand and will serve to factor into the acceleration given to the ball in the physics module. The accelerometer should give reliable and near noise-free data that can counteract any issues that arise with noise in the hand-tracking module, allowing us to have multiple options should either the accelerometer or tracking fail to produce reliable results.

The glove module will not need inputs from the FPGA as it is simply sending data back to the FPGA from the user's actions. The glove's outputs will be as follows: open, x_Accel, and y_Accel. Because we are currently completing a two dimensional game,

the z acceleration will not matter, and will not appear as an output. These inputs will need to be synchronized to the rest of the FPGA before they may be used. Refer to Figure 2 for the block diagram of this module.

Physics and Display Module – Michael Kelessoglou

The physics module will implement all the logic necessary for the game mechanics, which will attempt to emulate real-world physics. Using input from the visual tracking logic, the flex sensors, and the accelerometer, it will keep track of the position of the ball and the hands and output them to the VGA screen.

The logic will implement a state machine for the ball's position and velocity. It will make it possible to use switches to input how far away the two players are standing, so that the game can be played at various distances. During our first iteration, the hands' positions will be derived solely from the visual tracking logic. If we find that it helps, we may combine the accelerometer data with the visual tracking data to get a more accurate and less glitchy estimate of position. The flex sensor input will determine whether the hand is closed. This signal will most likely need to be synchronized. The ball can be in two states, the "holding" state and the "thrown" state. If the ball is in the "thrown" state and the hand is close enough to the ball when it closes, then we will transition to the "holding" state, in which the ball's position and velocity will match those of the hand. When the hand opens, the ball transitions to the "thrown" state, during which its velocity is affected only by the acceleration of gravity and possibly wind resistance if we decide to implement it. Both state transitions will trigger appropriate sound events.

The display module will display the ball and hands on the screen. The hands will be displayed using two sprites, one for a closed fist and one for an open hand. If we decide to implement more than two states for the hands, though that is not necessary for the game mechanics, then there will be one sprite for each state. The ball will either also be displayed as a sprite or as a monochromatic circle. We may use a separate sprite for a hand holding the ball. The sprites will be stored in the ROM of the FPGA. There will be an object module, similar the the "block module" in Lab 3, that takes in screen coordinates (hcount, vcount, etc.), the the x and y coordinates of the object, and the object's sprite, and outputs the pixel for a particular object (a.k.a. black if the object is not in the current hcount-vcount coordinates or the color of the sprite pixel that should be at the current coordinates). The pixel outputs for each object will be combined to get the output pixel. The method will be that if everything is black except for a single object, then that object's pixel is passed. If more than one object have non-black pixels, then each object will have a priority variable, and the one that will be shown will be decided by which one has the highest priority. The scale of the display will be dependant on the distance between cameras. If this module requires pipelining to meet time constraints, then this will not affect gameplay, as long as we apply the appropriate delays to the screen coordinates (hcount, vcount, etc.). An unpipelined block diagram of the module is shown in Figure 3.

As an additional feature, we may decide to implement a floor or walls. The ball may stick to them until someone picks it up, or it may bounce off them with variable elasticity set by user input. We may also add a game mode in which the second player is replaced by a target, a wall, a user controlled paddle, or an artificial intelligence "goalie."

Hand-Tracking Module - Evangelos Taratoris

The hand-tracking module will implement all the necessary logic that will enable us to take a sensory input from a camera and output the correct position of a virtual hand relative to our absolute defined coordinates. Upon initialization, we will create an absolute coordinate system based on the acceptable visual range of the camera. Then the module will take as input the image from the camera and it will output the location of the hands in the previously defined coordinate system. It will do so using an algorithm that will evaluate the position of the coloured gloves.

The algorithm will operate as follows: We will define an acceptable range of colours in the RGB scale to be perceived as "acceptable" orange. Then, the algorithm will scan the input image and try to find a contiguous region where all the pixels will fall into the acceptable range. We can define a "contiguous" region to be any of the following:

1) A circular disk of diameter r_diam, where r_diam will be a parameter.
2) A horizontal line segment of length l_seg, where l_seg will be a parameter.
3) A square region of diagonal l_diag, where l_diag will be a parameter.

All of the above will be tested during the implementation steps of our module and we will choose one based on how accurately it performs the tracking and how efficiently it does so. The location where the glove is will be a relatively large region containing "orange" pixels. This means that we will have to average over all the acceptable contiguous regions that we have found. We can do so by finding the centroid of all the points identified. This will give a good approximation of the actual location of the center of the hand.

There is an important thing to notice here. If in the background, there are things that fall into the acceptable orange range, the algorithm will produce the wrong result. Therefore it is important to be in a background where there are no orange things (orange fruit, Princeton memorabilia etc)

Timeline

After two weeks we expect all modules to be fully functioning. The Glove should output consistent and valid digital signals from the accelerometer, and flex sensors. The Hand-tracking module should work for static and moving gloves. The physics should be fully implemented, though the scaling and sprite creation will be completed by the end of the third week.

The third week will be focused on debugging and combining the modules. We are giving ourselves a full week to implement the combination of the modules to deal with unexpected consequences of miscommunication or bugs in the code.

The fourth week will be focused on improving the modules, implementing new features or more complex games, and implementing sounds. By this point, the project should be ready for presentation.

Testing

Testing for the glove module will be fairly straightforward. To test the effectiveness of the flex sensors, we will wear the glove and monitor the output with an oscilloscope to ensure that we are getting the proper behavior for an open and closed hand. We will then be able to test the accelerometer by connecting the acceleration outputs to the oscilloscope and moving the hand around.

Testing for the Physics and Display module can be done by giving the ball various initial velocities and observing whether it follows the expected parabolic motion. Catching can be tested by placing one glove and the ball at the same x position, dropping the ball into the glove, and controlling the glove state with a switch/button. After catching, we can test dropping in the same setup, and perhaps throwing if using buttons to control the glove can create a smooth enough motion.

Testing for the hand-tracking module will involve two steps:

1) The first step will be to check whether there is a location returned that corresponds to the location of the actual hand relative to the defined coordinate system. In essence this is a test for the static performance of the module.This can be done statically, i.e. without moving the glove, and by comparing the physical distance of the glove from the "corners" of our real coordinate rectangle to the distance of the (x,y) coordinates from the corners of the virtual coordinate rectangle.

2) The second part will involve testing the dynamic behaviour of the module. This means that we will be now moving our glove and compare whether what is shown on the screen corresponds to reality.

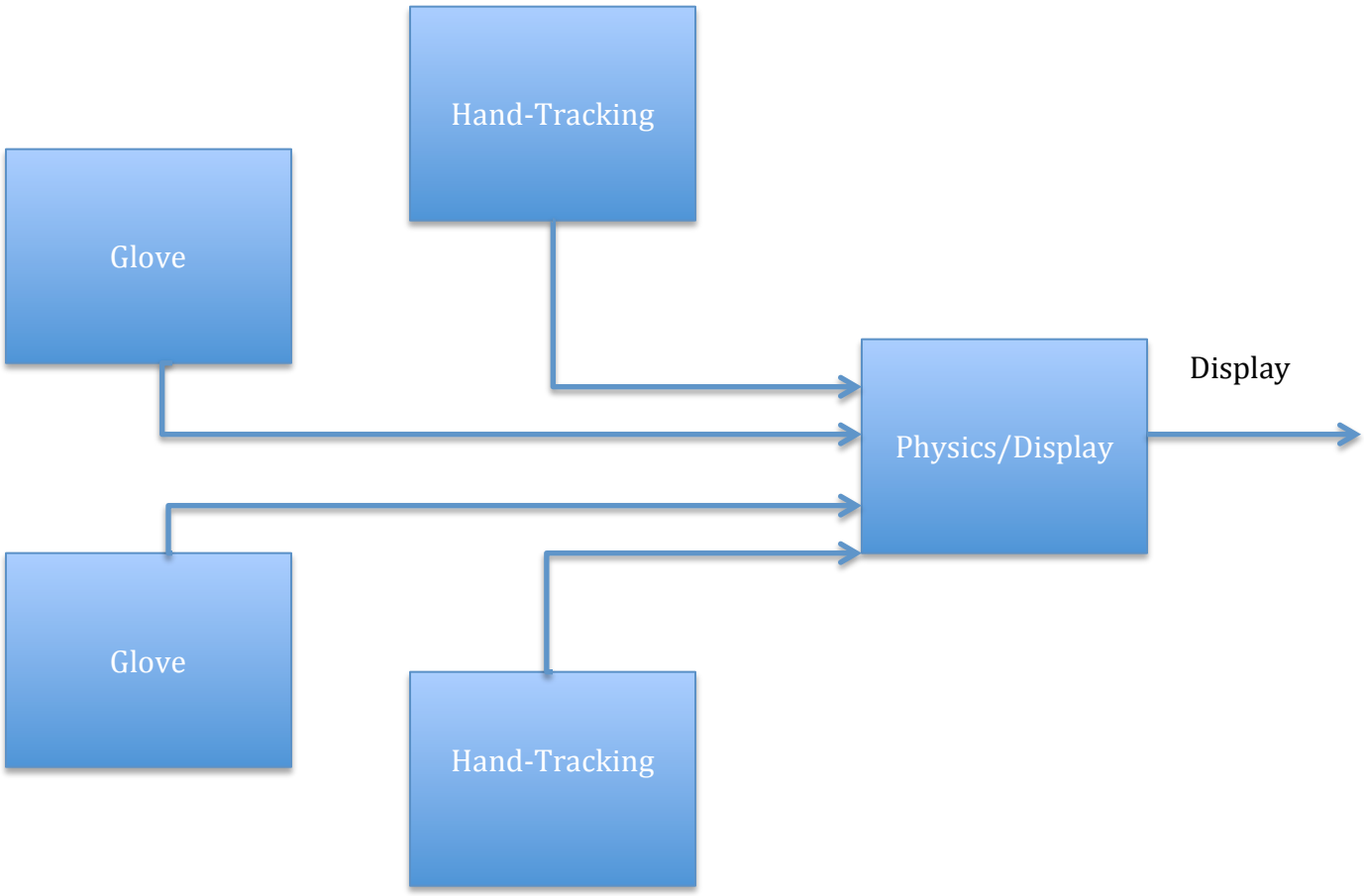

Figure 1: **Pitch and Catch Block Diagram** – This block diagram shows how two instances of the glove module and two instances of the hand-tracking module will interact with the Physics and Display module to simulate the game logic. The Physics/Display module will be able to render the game from the sensor data from the gloves and the tracking data from the cameras to show the progress of the game.

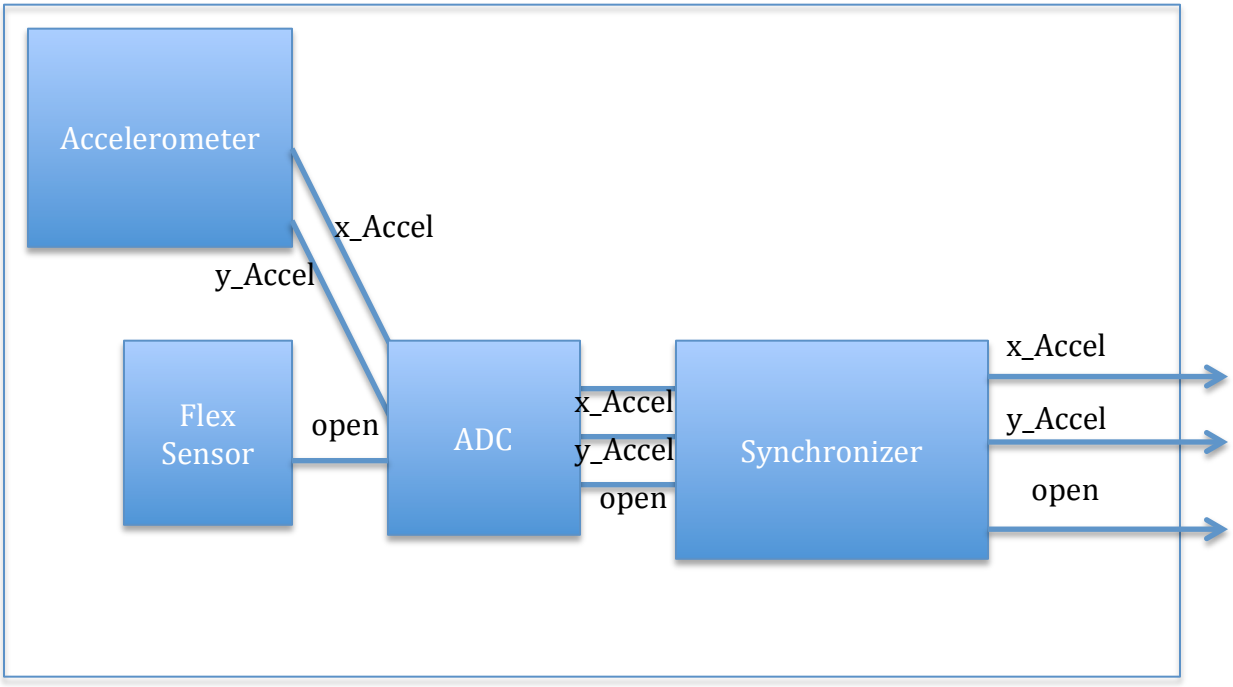

Figure 2: **The "Smart Glove" Module** – the "smart glove" will monitor its two flex sensors as well as its accelerometer. It will then convert these analog signals to digital signals and synchronize them with the system clock. These outputs will then go to the physics and display module.

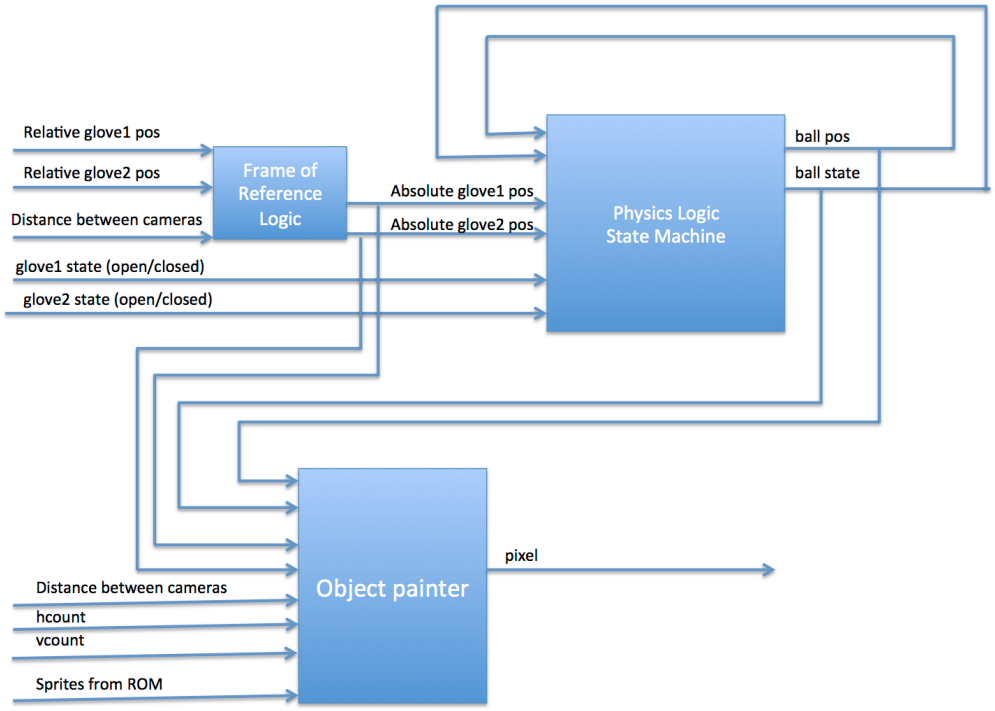

Figure 3: **Physics/Display Module** - The physics module is shown here with its inputs, outputs, and internal signals. This module will keep track of the ball, its motion, and its current state. The frame of reference will be computed in this module given the glove positions and input distance between the cameras. The ball will then be rendered by the object painter given its state from the physics state machine.