Lightsaber Training

Landon Carter (lcarter), Rachel Yang (rsyang), Linda Zhang (lolzhang)
6.111 Final Project Proposal | Fall 2016

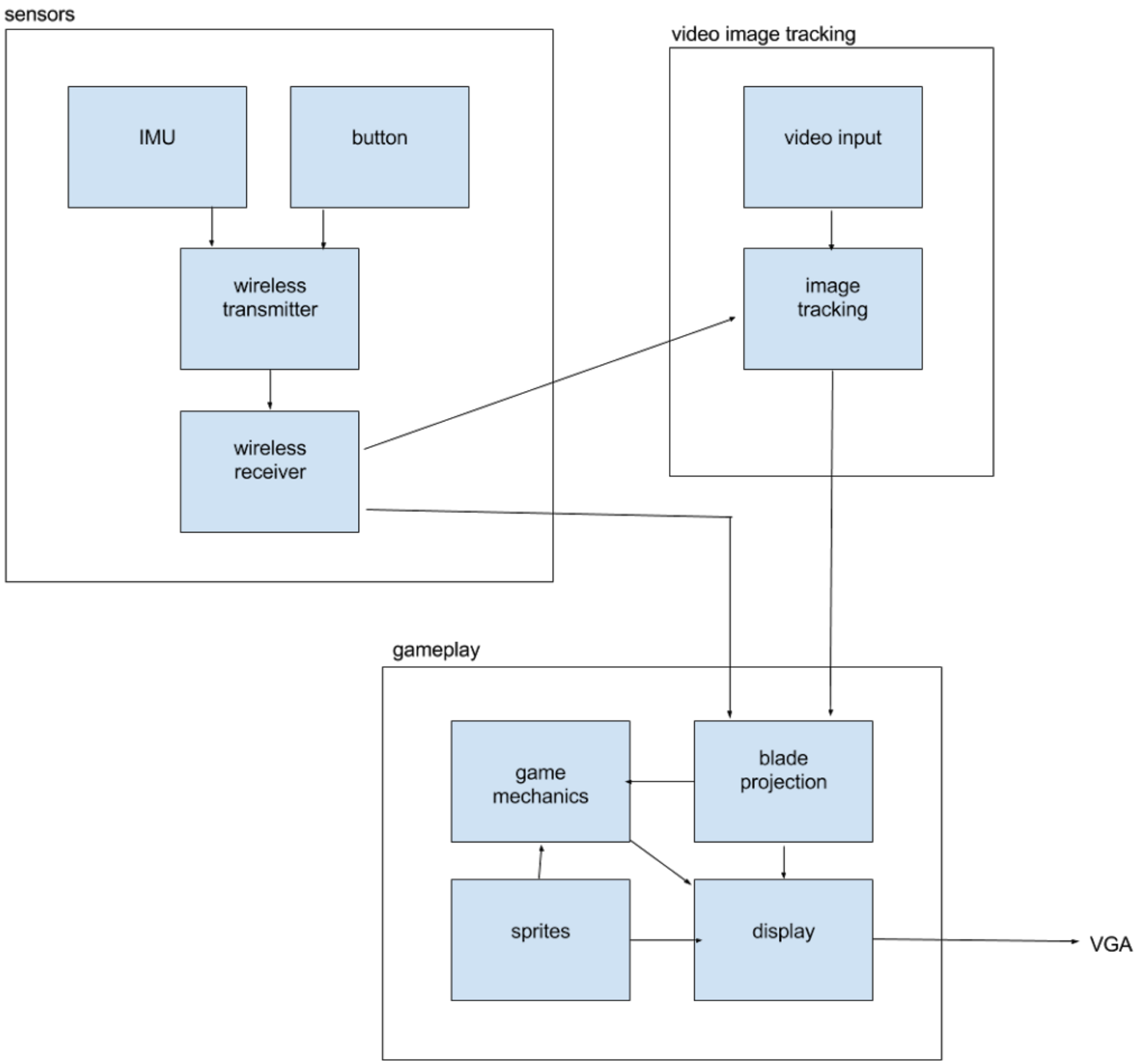
1 Overview

On October 18, 2016, Professor Gim Hom proclaimed his desire for a final project to design a Star Wars virtual lightsaber. Inspired by his enthusiasm, as well as by the weekly Thursday night lightsaber duels that occur in the lobby of building 34, we plan to implement an augmented reality game in which a user, wielding a lightsaber handle, attempts to hit objects displayed on-screen.

Our project will use image tracking from a camera connected to the labkit and data from a wireless bluetooth gyroscope module embedded within the handle to project the lightsaber blade on-screen, overlaid against the real-time video stream from the camera. A wireless bluetooth button also on the handle will let the user extend and retract the blade. The user's movements will be continually tracked so that the projected lightsaber blade moves with the handle. Gameplay will include sprites that appear on-screen and a scoring system that tracks when the user hits a sprite with the lightsaber blade. The game will be projected onto a nearby wall for maximum user engagement.

As a stretch goal, we hope to add a human-controlled quadcopter to the game. The position of the quadcopter will be tracked by the camera, and the user can score points by hitting the quadcopter with the virtual blade. Additionally, sound effects and other special effects can be added to make gameplay more immersive.

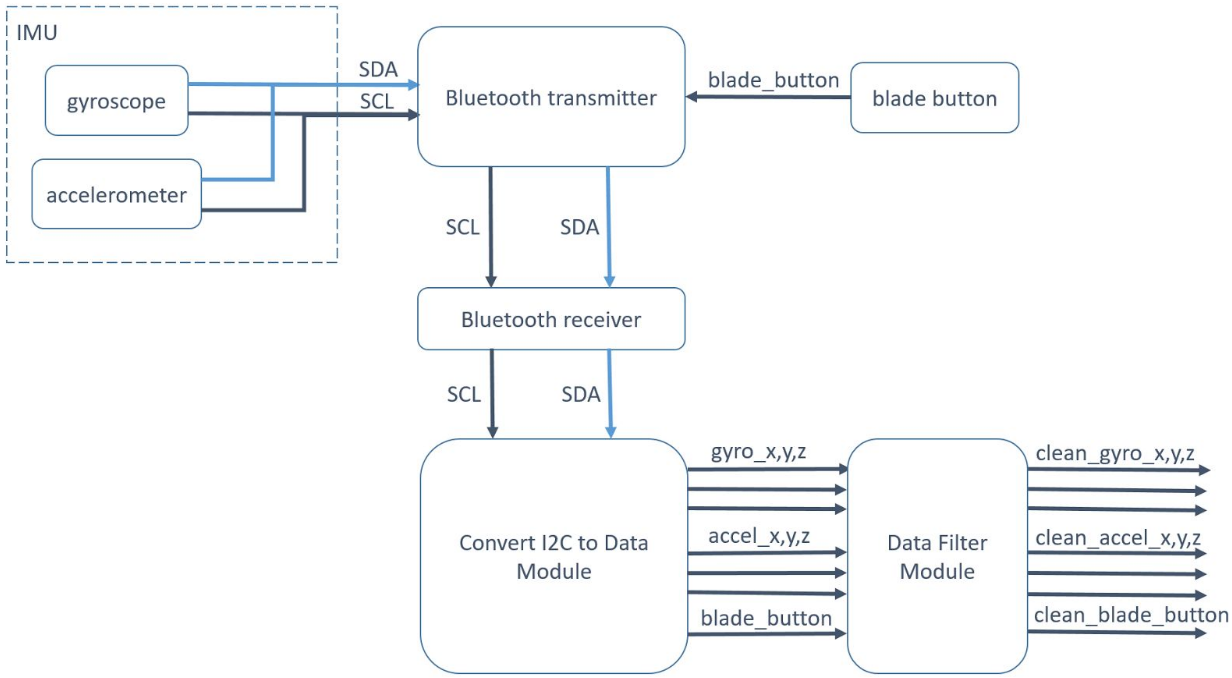
2 Design



Our project has three main blocks: sensor data (rsyang), video image tracking (lcarter), and gameplay (lolzhang). Implementation and testing of each block should be relatively independent of the others, allowing us to work in parallel until we are ready for integration. Accomplishing the minimum working project requires each major block to pass its own minimum threshold.

3 Implementation

3.1 Sensor Data Block Diagram (rsyang)

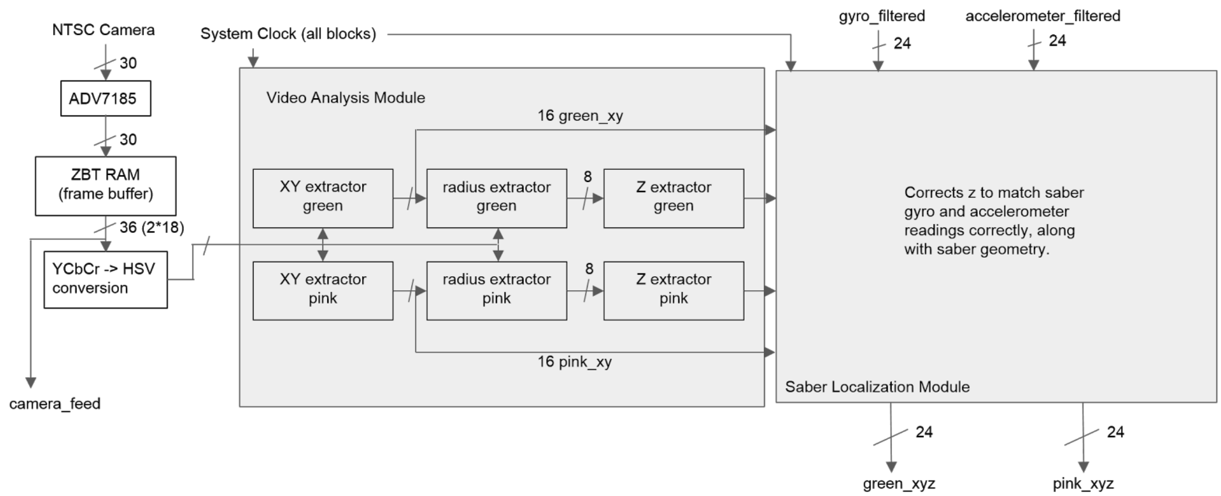


To help determine the location of the lightsaber handle in physical space, we will be putting an IMU on the handle to gather gyroscope and accelerometer data. The handle will also have a button that serves as an input into the game as to when the blade should extend or retract. To allow ease of use and not have wires dangling from the handle, the IMU and blade button data will be transmitted via Bluetooth. The Bluetooth transmitter will receive the IMU data via I2C to transmit.

The Bluetooth receiver will then extract the I2C data from the wireless signals. This I2C data is then passed onto a module to convert the I2C into usable gyroscope, accelerometer, and blade button data. The gyroscope and accelerometer data will both be 3-axis, and each axis will be represented with 8 bits.

The outputs of the convert I2C to data module will then be passed into the data filter module, where the IMU data will be averaged to filter out noise and the blade button data will be debounced.

3.2 Video Block Diagram (lcarter)

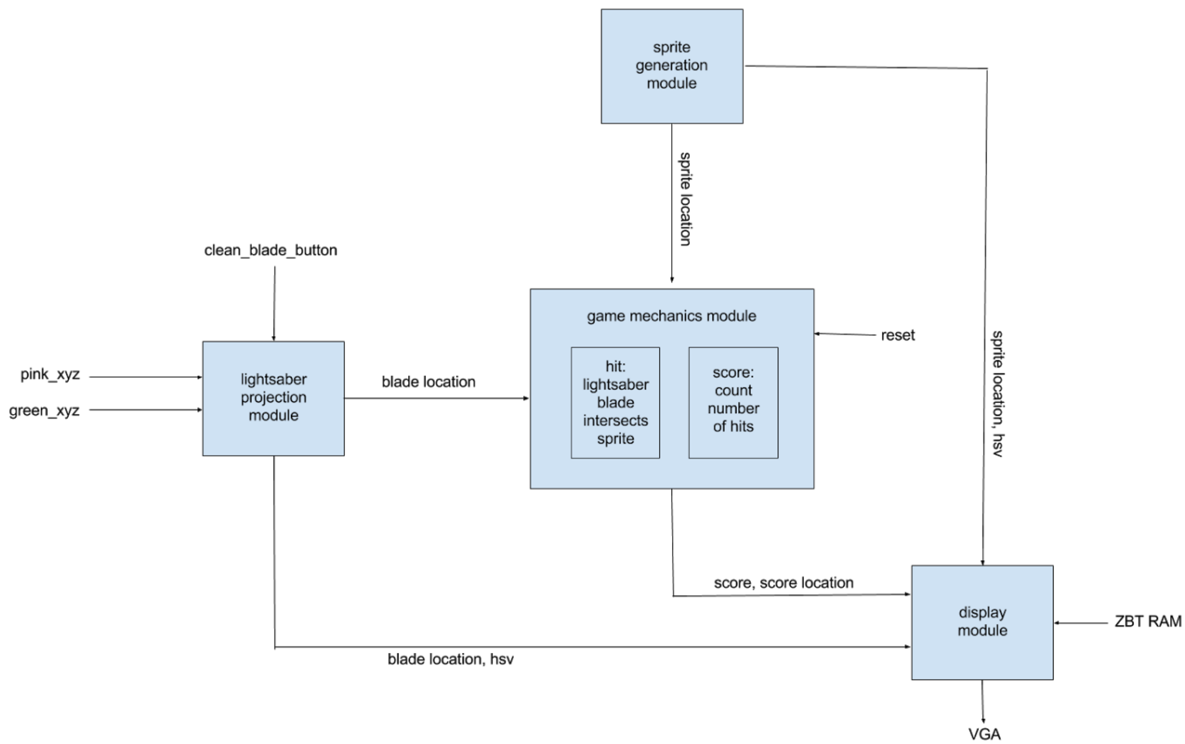


The second component in determining the position of the lightsaber handle in physical space is the video/image tracking module. We will be using a standard NTSC camera with the labkit to detect the lightsaber handle, which will have a green ball and pink ball on opposite sides, along the axis of the blade. This will allow for easy detection once converted to the HSV color space, since these hues/saturations are very unique and separated from standard clothing and lab environment, allowing for easy thresholding.

From the video feed, x and y values will be extracted for the pink and green balls, as well as the approximate radius. From the radius, we will attempt to estimate z-distance. This approximation will be refined by incorporating the gyro and accelerometer data, plus physical requirements on the handle dimensions to better approximate blade angle, refining the x, y, and z positions calculated in the video analysis model.

Finally, the xyz coordinates of the green and pink balls will be output to the gameplay module, along with passing through the camera feed. Overall, the latency of this module must be less than a single frame - ideally, all calculations should be able to be done during the blanking period of the eventual VGA output to minimize perceived lag. The YCbCr -> HSV consists of a simple linear transformation to RGB followed by another relatively simple conversion to HSV. This should have a base latency of a few clock cycles, still much less than the blanking period. The ZBT RAM will need to hold a single frame if all processing can be done during the blanking period, which will be stored as paired 18-bit YCrCb pixels, each packed into a single half of a memory location.

3.3 Gameplay Block Diagram (lolzhang)



The gameplay component of the project will take the outputs of the sensor and video/image tracking modules to create a projection of the lightsaber blade. Additionally, the sprite generator will create objects on screen that can be hit by the lightsaber, increasing the player's score according to the game logic. The lightsaber blade, sprites, and score will be overlaid onto the video image held in the ZBT RAM.

The lightsaber projection module will take the clean button data from the bluetooth receiver to either extend or retract the blade. The x, y, and z positions of the pink and green balls on the handle will be used to compute the blade's angle and start point. Assuming that the blade will have a fixed length, the end point of the blade can then be calculated, and a vector can be drawn between the points to create the projection. The xyz coordinates will then be converted to hcount and vcount coordinates to be displayed on screen and for use in the game logic. This is done by assuming the camera captures an 8ft x 8ft x 8ft space, with xyz point (0, 0, 0) at the bottom left corner of the image.

Objects for the user to hit will be created by the sprite generator module. The most basic implementation will create one motionless, uniformly colored, square sprite somewhere on screen. However, we hope to create more engaging sprites that may have different shapes and colors and can move or disappear. The sprite generator will output hcount and vcount coordinates to the display and game logic modules.

The game logic module will take the output locations of the blade project module and sprite generation module to determine whether the user has hit an object with the lightsaber. This should be a simple check of whether or not the coordinates of the blade and sprite are the same. If there is a hit, then the user's score should increase, and the sprite should disappear from the screen. The game logic module will output the score to the display module.

Finally, the display module will combine the video image saved in the ZBT RAM with the blade, sprite, and score overlays to create the gameplay screen, which will be projected onto a nearby wall. The overlay will use alpha blending to create a more realistic blade projection. The display module will generate all signals needed for a VGA output on screen.

3.4 Extra Components Needed

- IMU (ordering cost: $5-8)
- Bluetooth transmitter and receiver (ordering cost: $10-15)
- Level-shifter for Bluetooth to FPGA interface (ordering cost: ~$3)
- NTSC camera
- Projector
- Quadcopter (stretch goal)

4 Timeline

Note: All deliverables are to be completed by the end of their declared week

Week of 11/07

- Order needed parts
- Background research and refine block diagrams

Week of 11/14

- Have minimum working project achieved
  - Sensor data: have this working without Bluetooth, i.e. hardwiring sensors
  - Video interface: be able to detect one ball and output to gameplay
  - Gameplay: have vertical line to indicate where one end of lightsaber is

Week of 11/21

- Have individual parts working on their own

Week of 11/28

- Test all modules together, i.e. interface all three parts of the project and debug

Week of 12/05

- Extra week for buffer / testing
- Work on stretch goal if time

5 Testing

5.1 Sensor Data Testing

The IMU, Bluetooth transmitter, and Bluetooth receiver will be tested to ensure that these pre-made modules are indeed working as expected and not broken.

The I2C to Data conversion module and the data filter module will be written in Verilog and each tested separately via a test fixture. Once this is done, the IMU and blade button will be hard wired to the Labkit FPGA and tested with the conversion and filter modules. After this step is verified, Bluetooth will be incorporated as the medium between the IMU and blade button data to the Labkit FPGA.

5.2 Video Module Testing

The camera will be tested to ensure the somewhat pre-made NTSC video capture/feed modules are indeed working as expected (in particular, once color is added to the provided code).

The YCbCr -> HSV module will be tested for correctness via a test fixture. The initial xyz extraction will be completed independently of the sensor module, to ensure thresholding conversions are set up correctly before interfacing to the sensor data. Finally, the final lightsaber hilt detection including sensor data corrections will be tested by attempting to track an actual lightsaber hilt before interfacing with the gameplay and projection modules.

5.3 Gameplay Testing

The lightsaber blade projection module can be tested using a test fixture and then a display without video input. The test fixture would determine the correctness of the endpoint calculations given two handle points. We can then pick two points on screen and displaying the blade without any video input to determine the correctness of the vector calculations and hcount vcount outputs. Sprite generation testing will be fairly simple, as we can easily view the sprite on screen.

We can use a test fixture to test the hit and scoring portions of the game logic module as well. This can also be done without video or sensor input, given that the blade and sprite modules are functioning. Further testing can be done visually with the screen display.

The final display will test the integration of all project modules.

6 Conclusion

We hope to learn a lot about different systems on the labkit, along with many different types of FPGA-related things such as sensor/bluetooth interfacing, video analysis, and game/FSM mechanics, collision detection, and video overlay. Also, swatting virtual fruit with a virtual lightsaber will be really cool and hopefully very visually impactful.