Comprehensive Medical Tool Attachment for VR

ECE 445 Design Document

Corey Zeinstra, Vignesh Gopal, Mingrui Zhou
Group 78
TA: Mengze Sha
2/21/19

1 Introduction

1.1 Objective

A laryngoscope is a medical tool used for examining the larynx during a medical procedure called a laryngoscopy. When performing a direct laryngoscopy, a doctor has about a 10 minute window while the patient is under general anesthetic to insert the laryngoscope and cut and remove blockages that they were looking for [4]. This is a complicated procedure that requires a decent amount of practice to ensure high success rates. Consequently, performing this procedure repeatedly on different cadavers is not a viable option for training.

We propose to ameliorate this issue by taking advantage of recent improvements in virtual reality and sensor technology. We propose to create a virtual environment for doctors to train performing this procedure so that they can get the practice they need without burning through resources each trial. Our plan is to develop a laryngoscope with an IMU to track orientation, and a distance sensor to track how far the tube has been inserted in the throat. These sensors would collect information about how the trainee is performing the surgery, and would send this data through a microcontroller to the computer connected to our VR headset. The simulation environment would adapt and change in real time in response to the inputs provided by the trainee. We believe that this would provide medical professionals with adequate training while saving thousands of dollars on cadavers.

1.2 Background

The healthcare industry is one of the largest and fastest growing industries in the world. It is projected to rise from 7.077 trillion dollars to 8.734 trillion dollars by 2020 [1]. Currently, a major issue that medical professionals encounter while training is the lack of resources available to actually simulate complex and delicate procedures. A synthetic human cadaver costs about 40 thousand dollars each, and fewer than 20,000 real cadavers are donated to science each year [2-3]. This pretty much matches the number of medical students in all of the US. With the growing demand for improved health care, doctors must be adequately trained to perform procedures. Clearly, there is a mismatch between the number of doctors and training resources available for training physicians. This leads to a rise in malpractice and botched medical procedures.

The need for improved medical tools and devices is a growing industry. Deloitte projected that the medical technology sector would experience a compound annual growth rate (CAGR) of about 15.9% between 2016 and 2021 [1]. While a majority of this industry focuses on devices for real procedures, because of the rising costs allocated towards training physicians, there is an increasing demand for simulated training for medical students. Recently, virtual reality training has been tested for testing the competency of EMTs. Mcgrath [5] concluded that simulation

softwares are indeed effective for training and testing. However, he noted that the rapidly changing VR industry has made it hard for healthcare professionals to handle the various platforms available. Also, many proposed solutions are relatively high cost, so it is important to ensure that sensors and controllers are used efficiently. We plan to design a cost-efficient solution that would be easy for any general medical professional to use and train on.

1.3 High-Level Requirements List

1) User must be able to calibrate the sensors to meet accuracy requirements as specified in our block requirements
2) Universal Controller should be able to relay information to Unity application with low latency (<100 ms)
3) Power source (batteries) must be able to last the length of at least 2 simulations (~30 minutes total).

2 Design

2.1 Block Diagram

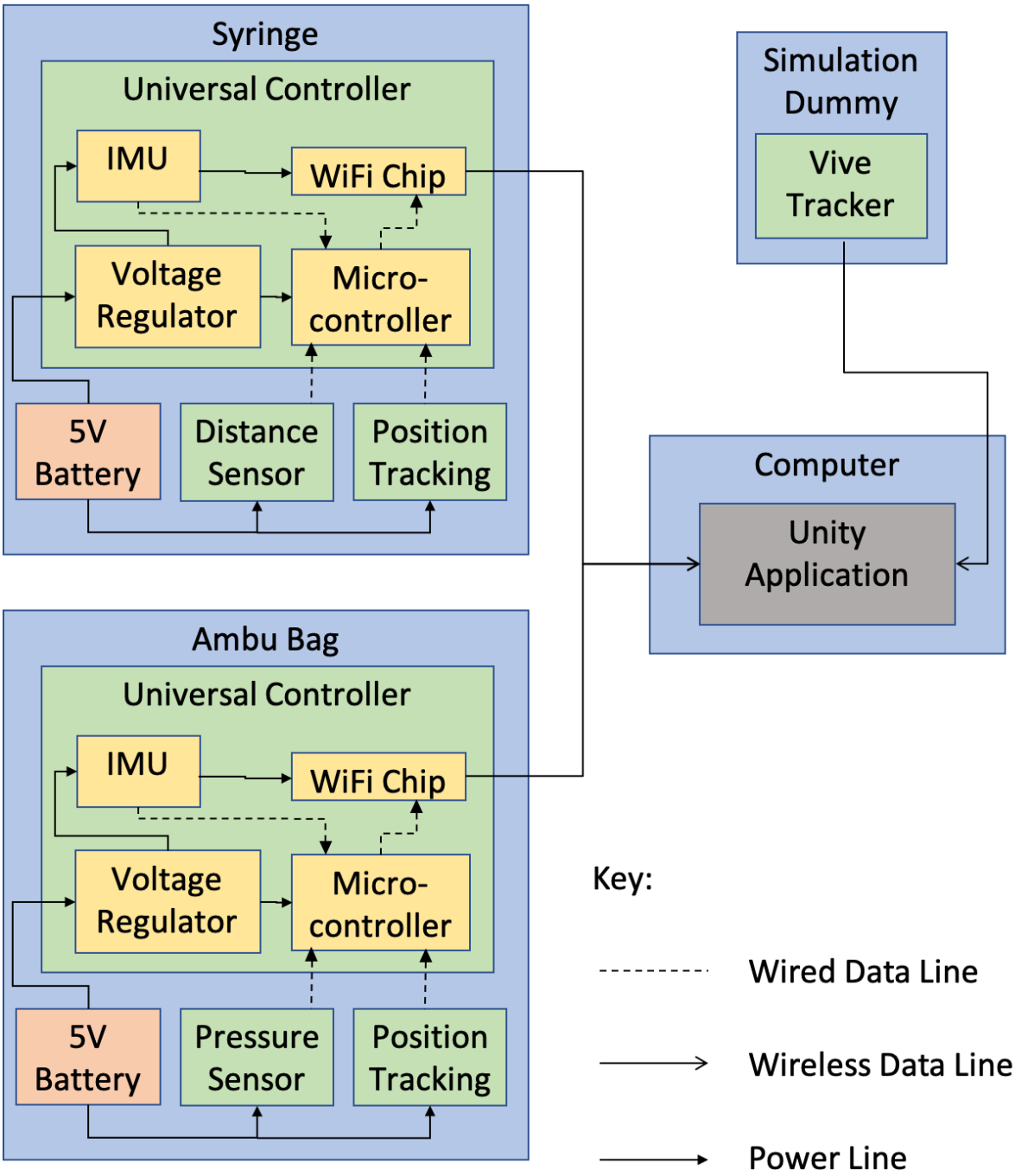

Fig 1. Block Diagram

The 4 main blocks of the diagram are the Syringe, the Ambu Bag, the Computer, and the Simulation Dummy. Most of our physical design will focus around the Syringe and the Ambu Bag. WIth the Dummy and Computer there to show off how the controllers work. The Syringe and Ambu Bag will have their own built in sensors specific to their uses and then attached to each will be a universal controller that will measure more common parameters as well as reading the output from the specific sensor and transfering the data to the computer over wifi. In the diagram you can see the sensors all transferring data to the microcontroller which will then send it to the computer via the WiFi Chip. This fulfills our first requirement of the design. The second requirement is again fulfilled by the microcontroller and WiFi chip but also focuses more on the actual Unity Application that we will build. While the application is part of the block diagram it is really more of a black box of software that will be created with the requirement kept in mind. The third requirement will be met by the fact that our actual design doesn't require that much in the way of hardware. Batteries, Microcontrollers, IMUs, and most other sensors are cheap. The only expensive components will be the LEAP motion for position tracking and the Vive tracker simulation dummy which will be borrowed from HCESC so we won't have to factor in paying for those.

2.2 Physical Design

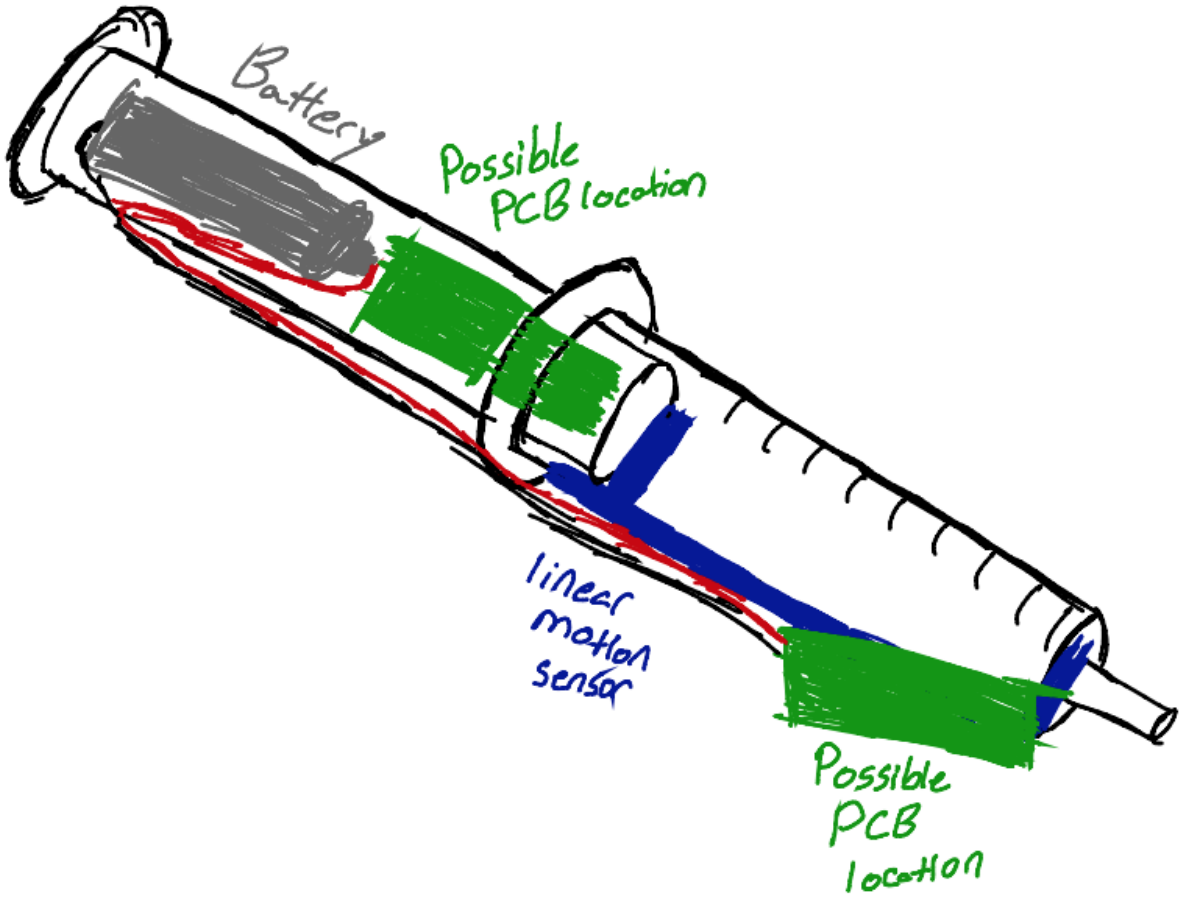

Fig 2. Syringe Physical Design

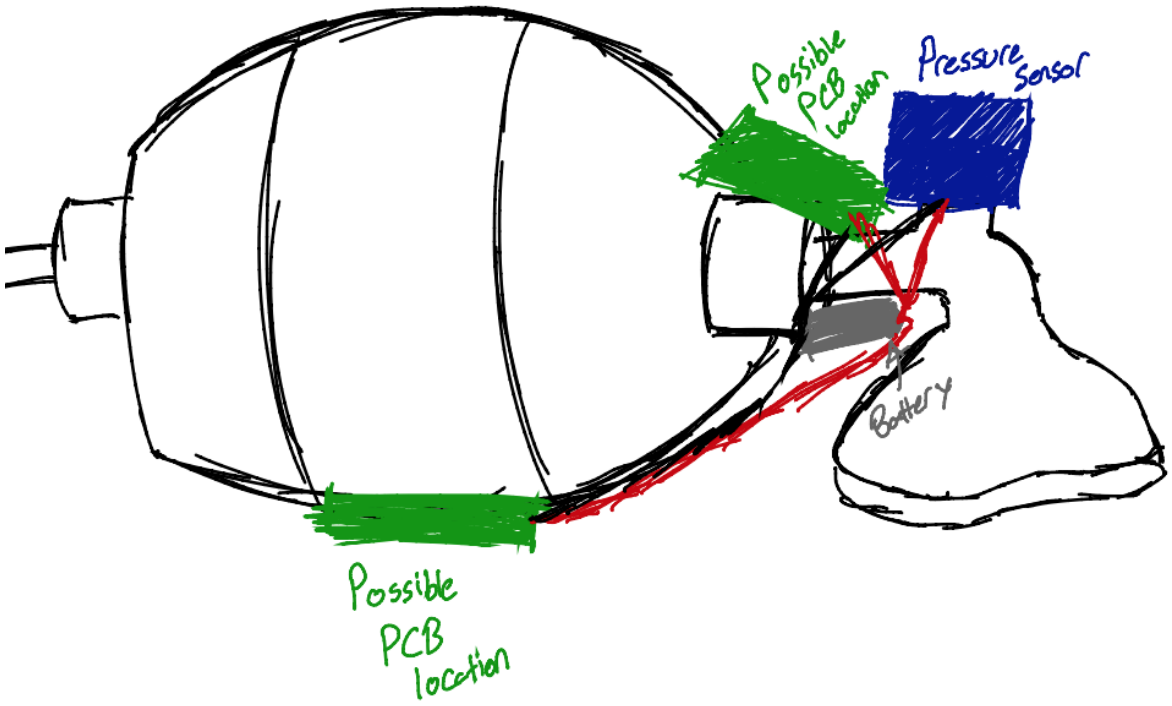

Fig 3. Ambu Bag Physical Design

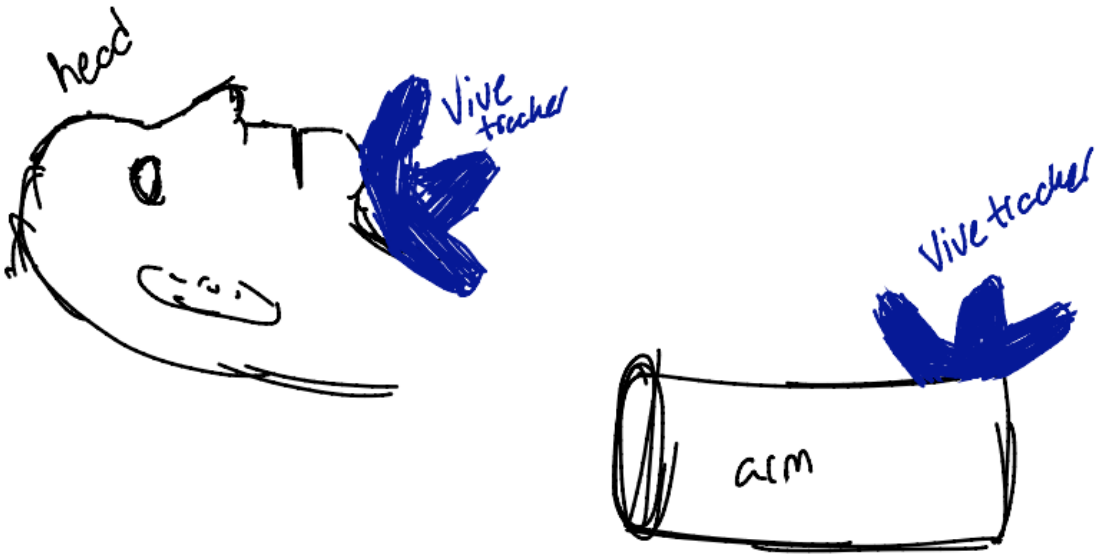

Fig 4. Possible Sim Model Physical Design

2.3 Block Design

2.3.1 Syringe

This is the same as the component shown in figure 2. For simplicity and replicability it will share many components with the Ambu Bag from section 2.3.2. To help keep the document from repeating itself I will only be discussing shared components here in section 2.3.1, so make sure to look at the Block diagram (figure 1) to see any shared components.

2.3.1.1 Distance Sensor

This will be built into the actual syringe itself and will output its reading to the I/O ports on the universal controller which will then go onto the microcontroller and transferred to the unity application. This is required to get a full readout of all the important parameters of the syringe. The distance sensor will be attached to the plunger and it will measure how much the syringe has been pressed. Then in the application we can measure how much fluid would be injected into the patient and at what rate and we will be able to tell if this is the correct amount, as well as giving visual feedback to the user on what exactly they are doing.

In order to actually create this we will have to either find a distance sensor online that fits into our syringe and can run at the voltages we use, or we will have to modify some other device like a small digital caliper. It is likely that we will have to cut open the syringe and have a hole that the distance sensor can stick out of so that it doesn't get in the way of the plunger. We will most likely be running this sensor off of the built in battery hidden in the syringe itself.

Requirement	Verification
Can track simulated volume to within 10% of the actual value (actual syringes can be as high as 5%).	1. Attach distance sensor to I/O ports of PCB and to the power source. 2. Measure the sensor at 5 different positions equally spaced between fully extended and fully closed 3. Check to make sure all of the measurements are within the allowed range.
Doesn't drift outside of the simulated range after multiple uses or prolonged usage.	1. Leave the device running and the program tracking data for 10 minutes and repeat step 2 and 3 from row above.

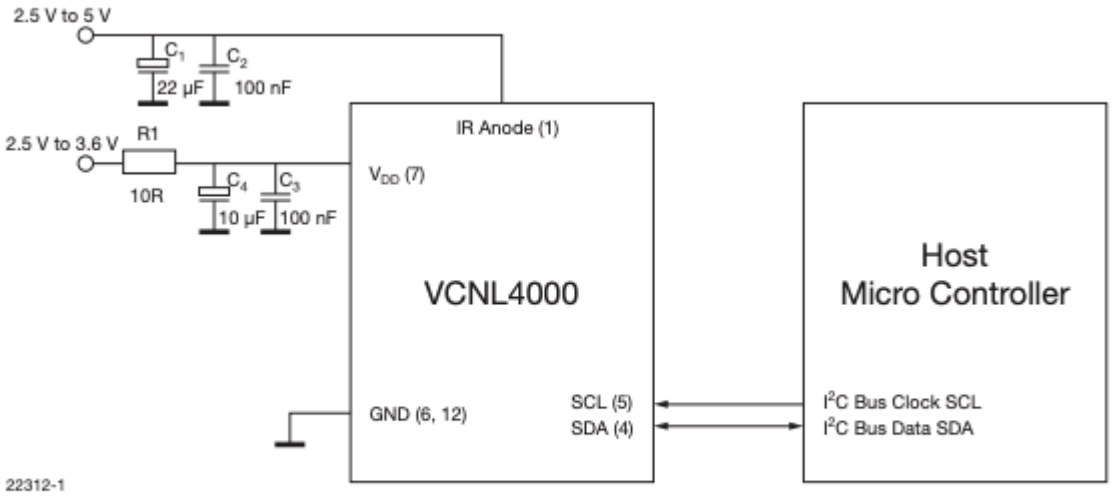

Fig. 12 - Application Circuit
(x) = Pin Number

Fig 5. Proximity Sensor Diagram [9]

Figure 5 shows a diagram and link to the distance sensor we are currently intending on using. Although the diagram doesn't show it correctly, the device runs off 5V so there will be no need to step down the voltage from the battery. The current plan is to use an off the shelf sensor, so the testing and verification for it will be light, however it both directly and indirectly connected to several other components that we are in charge of making so these R&V tests will still be run to make sure everything along the line is working properly.

2.3.1.2 Universal Controller

This will either be one large chip or several smaller chips and will contain all the components necessary to make a medical tool into a VR controller. It will attach to the controllers battery and will track all of the default parameters that every VR controller needs and will process and send that information to the application. It will also have several I/O ports that can be attached to external sensors if needed. The idea is that we will be able to design one chip that will cover most of the functionality necessary for creating a VR application, then we will be able to easily reproduce several copies of that chip and reuse them for several devices. The universal controller currently refers to the IMU, I/O ports, Microcontroller, and WiFi chip, and may refer to the position sensor depending on which method we end up using.

Requirement	Verification
Must be able to read data from at least 2 external sensors.	1. Plug 2 sensors into the I/O ports of the board and have the board send their data to the computer, check on the computer if both devices data are

	being tracked.
Must have a low consistent latency. An end to end latency of <50ms is ideal <100ms is acceptable	1. Move the IMU quickly and measure how long it takes for the computer to get the signal. 2. Repeat this 10 times to make sure that the time is consistently below the required time
Must be able to send data with good reliability and accuracy, 90% of data within 1 degree of the actual value	1. Place the IMU at an arbitrary angle and measure it, see what the raw output data is and make sure it corresponds to the correct value. 2. Repeat 10 times at 10 different angles to make sure it works consistently.

2.3.1.3 IMU

This will be built into the universal controller itself, it will be powered by the battery attached to the medical device and will send its data to the microcontroller. It will be comprised of a built in accelerometer, gyroscope, and magnetometer in order to consistently track the orientation of the chip and its attached device. IMUs are a crucial part of tracking the device and is used in pretty much all VR controllers so they will be used in all of our controllers as well.

Requirement	Verification
Should be able to work within the bounds of that the Voltage regulator is constrained to 3.6V to 3.0V	1. Connect the IMU to the power supply and slowly move it from 3.0V to 3.6V make sure that the IMU is reading correct values the entire time.
Should be able to correctly measure angle of device with respect to all 3 axes and send to the microcontroller correctly.	1. Hook the IMU up to the Microcontroller. 2. Have the Microcontroller output the raw data from the IMU and hold the IMU at 10 different positions. 3. Make sure this data aligns with actual values to within 1 degree.

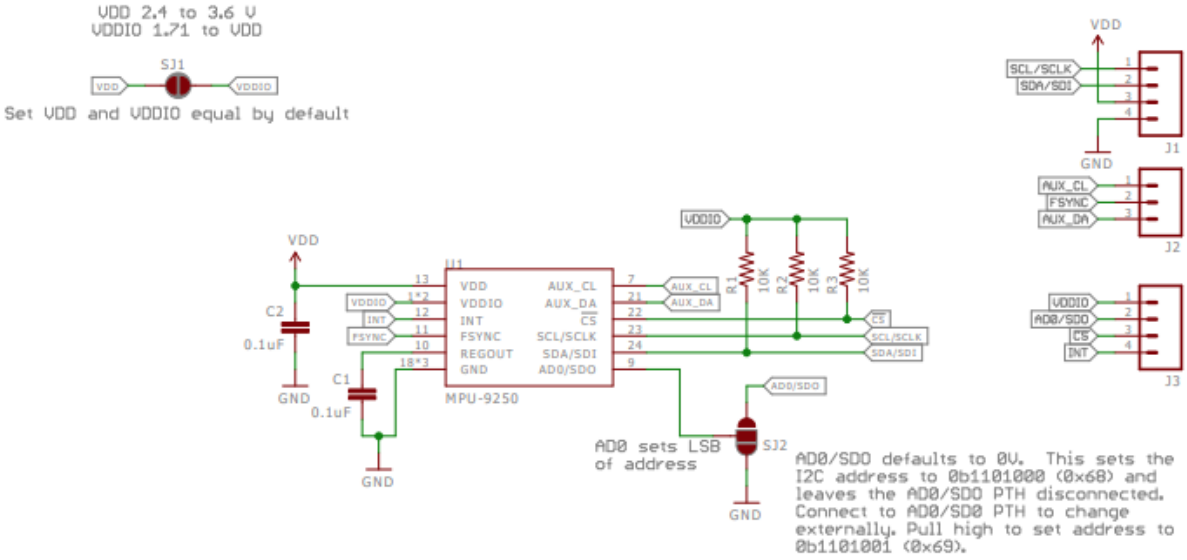

Fig 6. IMU Diagram [10]

Figure 6 shows one possible IMU chip that we will be using and how to correctly wire it into our universal controller.

2.3.1.4 Position Tracking

We aren't 100% sure what we are going to use for position tracking, there are a couple options on the table and we need to do some hands on tests on a couple before we come to our final decision. The current plan is to use LEAP motion to track the controllers. This is nice because it removes the need for extra sensors on the controller itself and instead puts them on the VR headset. Otherwise we will have to find a way to track the controllers using the built in IR camera system built into the VIVE. Then we would have to create a chip and some IR LEDS and power them with the inbuilt battery and then send the data to do the microcontroller. Position Tracking is also an important requirement as it allows us to know exactly where our controller is in 3D space and will be needed to accurately map our controllers into our application.

The current plan for position tracking is to use an off the shelf unit (LEAP motion), therefore we haven't included any R&V, however if our position tracking changes we will accommodate with an actual R&V section.

2.3.1.5 Microcontroller

This will be powered by the built in battery and attached to the universal controller. It will have several inputs from the I/O ports, the position tracking, and the IMU and will output data to the WiFi chip. It will read all of these signals and do any necessary computations on them and then tell the WiFi chip to send them to the computer where they will be used in the Unity application.

The microcontroller will be in charge of controlling the chip itself along with all its peripherals and signals going to and from the device.

Requirement	Verification
Must be able to operate within the bounds of 3.0V to 3.6V.	1. Connect the microcontroller to a power supply and vary the voltage between 3.0V and 3.6V. 2. Have the microcontroller constantly running some function to make sure it keeps working consistently.
Must be able to read data from all of the sensors and output it correctly with low latency (<50 ms) and good consistency (90% accuracy).	1. Connect 2 external sensors and all the internal sensors to the microcontroller and have it perform its operations and send it onto the WiFi chip. (Part of the universal controller verification)

2.3.1.6 Wifi Chip

This will be attached to the microcontroller and will be powered by the built in battery. Its job will be to take whatever the microcontroller is outputting and send it to the computer so that it can be used in the unity application. This will be the main form of communication between our controllers and the unity application.

Requirement	Verification
Must be able to send data to receiver correctly and consistently.	1. Feed some arbitrary data into the Wifi chip and have it send it to another computer. Have the computer read the output and verify it is correct.
Must be able to operate within the bounds of 3.0V to 3.6V.	1. Connect the microcontroller to a power supply and vary the voltage between 3.0V and 3.6V. 2. Have the microcontroller constantly running some function to make sure it keeps working consistently.

2.3.1.7 Voltage Regulator

This will be integrated into the universal controller. All of the chips on the universal controller should be designed to run off of 3.3V but the battery and some of the external sensors are built around a 5V system. We will need to step down the voltage from the battery so all the components on the universal controller can run off of it.

Requirement	Verification
Must be able to maintain a voltage of between 3.0V and 3.6V throughout the entire lifecycle of the battery.	1. Connect the battery to the Voltage Regulator and add in a circuit that will drain the battery. 2. Constantly measure the voltage coming out of the regulator to make sure it is constantly within the required bounds. 3. Recharge the battery and repeat step 2 twice more to ensure it works consistently.

2.3.2 Ambu Bag

This is the component shown in figure 3. It by design shares a lot of components with the syringe. Any overlapping components will be found in section 2.3.1 instead of this section.

2.3.2.1 Pressure Sensor

This will be attached to the Ambu Bag itself and will output its readings to the I/O ports on the universal controller much the same way as the distance sensor does in the syringe. This will then read how much air pressure is being exerted into the patient's lungs, some actual Ambu Bags actually have pressure monitors attached to the bag which can show the doctor if they are sending the correct amount of pressure, but these won't work in VR so a digital solution must be created. The actual integration of the pressure sensor should be easy enough as long as we are able to source an Ambu Bag with a built in location for the standard pressure monitor to be attached. This sensor will allow us to track both how much air the user is putting into the patient's lungs as well at what intervals they are squeezing the bag.

Requirement	Verification
Must measure a limit of at least 0-60 cm H_2O (range of an actual bag) and an accuracy of at least 2 cm H_20 (based off actual gauge).	1. Attach an actual pressure sensor to the Ambu bag as a control. 2. Wire up our pressure sensor. 3. Use the Ambu bag and see if their outputs are within range.
Must be able to accurately signal when the user squeezes the bag so that interval can be tracked.	1. Squeeze the Ambu bag at a regular 5 second interval for 2 minutes and record the results. 2. Make sure these results correspond to the actions you performed.

2.3.3 Simulation Model

2.3.3.1 Vive Tracker

This is an off the shelf stand alone unit that can track position and rotation and send its data to the unity application. It is rather large and expensive so we will most likely not be using it on our controllers themselves but instead borrowing a few examples to attach to the simulation dummy in order to give the user some haptic feedback while our controllers. They will be attached to a specific point on the dummy and will be mapped to a model in VR the model will then follow the dummy so that your visual and touch senses will line up properly.

This will be an off the shelf component so no R&V is required.

2.3.4 Computer

The computer will be attached to the HTC Vive headset and will be running the unity application, the only component of it that we will be working on ourselves is the unity application.

2.3.4.1 Unity Application

This is a piece of software instead of a physical device like the rest of the blocks. Within it we will create a small virtual simulation tutorial that will walk the user through a simple medical procedure that uses our controllers and dummy. It will be required to accurately display the locations of the devices as their data is sent to them via the microcontrollers and wifi chips. This will be a large part of our demo and will be used to make sure that we actually meet our requirements. People will be able to use the application to test out if the controllers work properly and track accurately in VR.

Requirement	Verification
Must correctly react to the data coming from the controllers and external sensors. All 3D models must move accordingly.	1. Connect the controllers to the unity worldspace and measure how they map in the virtual world to how they are set up in the real world.
Must perform operations quickly to keep the end to end latency low.	1. Move the controllers in the real world and measure how long it takes for it to react in the virtual world. 2. Repeat step 2 10 times and the leave the controller stationary for 2 minutes. 3. Repeat step 2 another 10 times.

2.4 Tolerance Analysis

The most important tolerance we are going to have to deal with is making the VR world line up as closely as possible with the real world. This will be a pretty wide and not very strict tolerance but it will very important in order to make the VR application work properly. We were able to find a document called The Accuracy and Precision of Position and Orientation Tracking in the HTC Vive Virtual Reality System for Scientific Research[7] that goes over in depth the tolerances you can expect to see from commercial grade VR tracking systems. In the end they conclude that the rotation of a Vive usually stayed within about 5 degrees of the correct value. We decided that because this value is based on the IMU we should be able to reach similar values with our device. The paper goes on to talk about the latency of the Vive tracker and said that it was around 22ms. Our controller will probably use a different form of position tracking so we will not expect to get that low of a latency. Our goal is about 50ms because we found a source[8] that said that was a bottom end latency for most input devices. Finally the tolerance for position tracking will be tricky and will require more hands on research into what is acceptable. We are confident that because we are using a store bought system to help track our position and then supplementing its information with data from our own system we will be able to get to commercial levels of accuracy in position tracking.

3 Cost and Schedule

3.1 Cost Analysis

Based on the information we collected, our development costs are about $45/ hour, and about 10-15 hours/week for three people. And we consider to about 10 weeks this semester.
So our labor cost is about:

Max: 3 * 45 *15*10 = 20250
Min : 3 * 45 *10*10 = 13500

Part	Cost(prototype)	Cost(bulk)
Syringe	Average $15.00	$2.50
Distance Sensor	Average $7.50	$2.00
Universal Controller (PCB)	Average $3.10	$0.10
IMU	Average $13.00	$2.10
Microcontroller	Average $7.00	$0.60
Ambu Bag	Average $20.00	$4.00

Wifi Chip	Average $7	$1.2

3. 2 Schedule

Week	Duties	Corey	David	Vignesh
26-Feb	Buy parts and controllers	x	x	x
4-Mar	Configure Sensors			x
11-Mar	Assemble PCB	x		x
25-Mar	Program Wi-Fi controller with sensors and VR headset		x	
1-Apr	Build Unity Application	x	x	
8-Apr	Begin testing device	x	x	x
April 15 to end	Refine prototype	x	x	x

4 Ethics and Safety

We must ensure that the real environment that the trainee is in completely safe as the user would not be able to see what is actually around them. For this, we think that the best course of action is to not use blunted tips on the laryngoscope device (this is fine because the actual device does not need to cut into anything). Also, we plan on having the actual devices tethered to the work station that we develop. This way, the user would not be able to move far away from the dummy while they are essentially blinded to their actual environment. Overall, the design does not have too much room for safety hazards, but we find it important to address the ones that do exist.

Ethically, VR in medicine has had a history of contentious points. Historically, many medical professionals have raised concerns regarding the accuracy and verisimilitude of simulations such as ours [6]. Also, many professionals have raised concerns regarding biases in VR

research applications stating that researchers would be motivated to make advancements in their simulations at the expense of patients that would be later impacted by this technology. We believe that our project will not violate any of these ethical concerns as our device will be used as ancillary training tool for students who will have to perform the procedure on cadavers regardless. Essentially, this tool would not be a replacement for practicing on an actual body, but would rather be a training tool used before actually carrying out the procedure. Also, we plan on using well-accepted 3D models of a human larynx/throat in our simulation to ensure adequate verisimilitude.

The IEEE ethics code rule number 6 states that we must be careful when discussing the limitations of our work as one could be encouraged to give more credit to their project than actually deserved [6]. I think that it is important for us to make sure that our device is clearly only intended for supplemental training purposes. Ethically, our product must explicitly state the differences between the simulation and how the procedure would be in real life.

References

[1] S. Burrill, “2019 US and Global Health Care Industry Outlook,” *Deloitte United States*, 25-Jan-2019. [Online]. Available: https://www2.deloitte.com/us/en/pages/life-sciences-and-health-care/articles/us-and-global-health-care-industry-trends-outlook.html. [Accessed: 07-Feb-2019].

[2] S. Bachai, “Synthetic Human Cadaver To Be Used By Medical Students Instead Of Actual Humans,” 13-Nov-2013. [Online]. Available: https://www.medicaldaily.com/synthetic-human-cadaver-be-used-medical-students-university-arizona-instead-actual-humans-262768. [Accessed: 07-Feb-2019].

[3] M. Maccall, “The Secret Lives of Cadavers,” *National Geographic*, 29-Jul-2016. [Online]. Available: https://news.nationalgeographic.com/2016/07/body-donation-cadavers-anatomy-medical-education/. [Accessed: 07-Feb-2019].

[4] “Laryngoscopy: Purpose, Procedure, Types, And Complications.” [Online]. Available: https://www.webmd.com/oral-health/what-is-laryngoscopy. [Accessed: 07-Feb-2019].

[5] J. L. Mcgrath, J. M. Taekman, P. Dev, D. R. Danforth, D. Mohan, N. Kman, A. Crichlow, and W. F. Bond, “Using Virtual Reality Simulation Environments to Assess Competence for Emergency Medicine Learners,” *Academic Emergency Medicine*, vol. 25, no. 2, pp. 186–195, 2017.

[6] Ieee.org. (2019). *IEEE Code of Ethics*. [online] Available at: https://www.ieee.org/about/corporate/governance/p7-8.html [Accessed 22 Feb. 2019].

[7] Niehorster, D., Li, L. and Lappe, M. (2017). The Accuracy and Precision of Position and Orientation Tracking in the HTC Vive Virtual Reality System for Scientific Research. *i-Perception*, 8(3), p.204166951770820.

[8] Orland, K. (2019). How fast does “virtual reality” have to be to look like “actual reality”?. [online] Ars Technica. Available at: https://arstechnica.com/gaming/2013/01/how-fast-does-virtual-reality-have-to-be-to-look-like-actual-reality/ [Accessed 22 Feb. 2019].

[9] Adafruit.com. (2019). *Fully Integrated Proximity and Ambient Light Sensor with Infrared Emitter and I2C Interface*. [online] Available at: https://cdn-shop.adafruit.com/datasheets/vcnl4000.pdf [Accessed 26 Feb. 2019].

[10] Sparkfun.com. (2019). *MPU-9250 Product Specification Revision 1.0*. [online] Available at: https://cdn.sparkfun.com/assets/learn_tutorials/5/5/0/MPU9250REV1.0.pdf [Accessed 26 Feb. 2019].