SDP20 – Team 15

Camera Trap 3D

[1]

Anamitra Datta, CSE, CS, Max Haimowitz, CSE,
Xiaoyang Pan, CSE, and Minting Chen, CSE

***Abstract*—The current research practices used by biologists who are studying endangered animals have many limitations. For instance, tranquilization is potentially dangerous to an animal and requires a lot of manpower and training. A better system would allow researchers to collect large amounts of data on animals easily without disturbing them. One system that would accomplish this is a network of cameras that would be designed to take photographs to make 3D models. Ideally, this system would be low-power, easy to deploy and operate in remote areas, and reliable at generating 3D models.**

I. Introduction

[1. THE goal of this project is to create an integrated camera trap system that can be deployed by biologists in the wild to create 3D models of endangered megafauna.

A. Significance

Many people are concerned about the extinction of certain species of animals. This includes governments, scientists, and the public. People do care about the health of the animals, but the way people gather information from the animals can potentially be harmful. In the past, people used to tranquilize endangered animals, set up traps, or utilize other remoting sensing technologies. If scientists and researchers collect information through these methods, the process might be potentially disruptive, harmful, and dangerous for animals and for them. Additionally, certain species of endangered animals, such as the tiger quoll can be elusive [19]. Therefore, a system that gathers meaningful information from animals where minimum interference is required.

B. Context and Existing Products

Currently, there is a wide variety of camera traps which can be bought usually in the range of $40 to $100 [11]. The main market for these traps is hunters and biology researchers who want single still photographs of the animals they are interested in. These traps can also be set to take video.

Last year, Professor Duncan Irschick, who is a biology professor at UMass Amherst and the manager of the 3D Digital Life project, [16] which aims to create high-quality 3D models of animals, had a graduate student (Nino Figliola) from the MIE department investigate whether these traps can be used in a way that would take photos for generating 3D models.

The issue that came up was that if the cameras were all set up to take photos, they would get triggered at different times, and there was really no way around this lack of coordination. A different approach was to set the cameras to take video and then try and stitch all the videos together to find the same frame, and then use that to make the models. The issue with that, however, was that with video the individual frames are of much lower quality (i.e. the fewer number of pixels), and thus not suitable for 3D reconstruction.

It should also be mentioned that there is networking for camera traps, just not the kind we want. Currently there are camera traps that can connect with 4G phones (intended for hunters) that can transmit video over the cell network [1]. The issue is that this cannot be configured with more than 4 cameras at a time, so this would not work for our purposes (since we will need many more cameras), and they are not meant to be used in the same area.

C. Societal Impacts

This project is intended to be used by biologists studying endangered animals. Currently there are about 41,000 endangered species and 16,000 critically endangered species [2]. To give some examples using famous species, there are only about 2,500 giant pandas (Ailuropoda Melanoleuca) left in the wild and only about 4,000 tigers (panthera tigris). The main causes of the deaths of these animals are hunting and the spread of human development into previously wild areas. Also, as bad as it is by itself to see one species go extinct, there are also knock-on effects from extinction since it could destabilize the entire ecosystem.

The reason why this project is important is that it will not only allow researchers to learn about the number of animals in a particular area, but it will help gather data about the health of those animals. In particular, one very important piece of data researchers want to learn is the weight of the animals because that lets the researcher know how much food the animal is getting. By using pre-existing algorithms, a researcher can estimate the weight of an animal from a 3D model, which is what our system helps create. Of course, it is possible to get this data through tranquilization, but this can be difficult and cause political issues. In many of the countries where the research is taking place having foreign researchers go in and interfere with these animals (which can be a source of national pride) may be offensive. So, our design choice was made with the goal in mind of being totally non-invasive to the animal.

D. Requirements Analysis and Specifications

Our system has many specifications and requirements in order to ensure proper synchronization of the images generated by the system, a proper setup of the system so it can

[1] M. Haimowitz (email: mhaimowitz@umass.edu)
A. Datta (email: anamitradatt@umass.edu)
M. Chen (email: mintingchen@umass.edu)
X. Pan (email: xiaoyangpan@umass.edu)

accurately take automatic photos for long periods of time, and proper design of the system so it is scalable, portable, and compact enough to be set up easily and readily deployable in the field.

In order to generate an accurate 3D model, the 2D images must be synchronized within 0.1 seconds of each other, or else the model will be distorted and inaccurate. Our system will be ensured to take photos within 0.1 seconds of each other 95% of the time. This is our most important requirement for our system. Having high-quality images helps to improve 3D models so the system must generate 8K resolution images, which is the highest resolution possible for images. Our system uses a Raspberry Pi Camera V2 module which can generate 8K quality photos, so we meet this specification.

For the design of the system, we need to specify how the cameras will be placed and how long the system will run for. The cameras should be able to be connected wirelessly through Wi-Fi by up to 8 meters of each other. With this networking range, the system should be able to detect the presence of animals using their PIR sensors from 1-5 meters up to 95% of the time. The system should be able to run for at least 72 hours; this requires less than 370mA of power consumption for each device. The system will also have long-distance transmission of information, so a user can get information about the system from a long-range. We will send information about battery life and the number of photos taken within a range of 1-3 km.

It is very important for this system to be easily deployable and operational in the field. We decided that our system should be easy to set up by a non-expert in less than 45 minutes. It should also be portable since we need to set up the system in remote areas without an Internet connection or power outlets. It should also be scalable if someone wants to add more devices into the network without affecting the performance of the system. All requirements and specifications of the system are given in Table 1.

Requirements	**Specifications**
Synchronization	Photographs from the cameras are synchronized within 0.1 seconds, 95% of the time
Photo quality	The best possible resolution, 8K, 3280*2464 resolution
Detection Range	Detects the presence of animals at 1-5 meters, 95% of the time
Networking Range	Cameras need to be connected wirelessly up to 8 meters
Battery Life	Lasts at least 72 hours
Power Consumption	Less than 370 mA
Long distance transmission range	Statistics on battery life and number of photos taken in range of 1-3 kilometers
Easy to deploy and operate	Can be set up in less than 45 minutes by a non-expert
Portable	Needs to be portable to set up in remote environments with no Internet connection or power outlets
Scalable	Can add multiple cameras easily to the network and will not affect performance of the system, up to 25 cameras

Table 1: Requirements and Specifications

II. DESIGN

A. Overview

For our final system, we used three cameras, each attached to its own Raspberry Pi. Our system follows a central/edge paradigm where one of the camera modules is a central listening to the sensor data for when to take a picture and the other modules respond to the main module's message of when to take the picture. These modules are connected through Wi-Fi as part of an ad-hoc network and they communicate messages with each other through a protocol called MQTT (Message Queueing Telemetry Transport). The main module contains a sensor, an RTC (Real-time clock), and a PCB to keep track of how much battery is left in the power brick attached to the module. Our system also has a software UI that can be used to view photos taken by the system from a mobile phone through an app as part of short data transmission.

In our final product we had to find balance and make compromises in order to meet all of our requirements. For instance, in our power specification we could have gotten much more battery life if we hadn't gone with a Raspberry Pi, but we needed to do that in order to get a camera that had good resolution. Similarly, making our system easy to use meant that we probably had to sacrifice a decent amount of performance. During our initial prototyping phase in the fall we controlled the system through SSH (command line), which made improving our code very fast. But, for using our final product we developed a mobile app, which can interact with the software on the Raspberry Pi which is a lot more complicated to implement and gives the user less control. A further limitation we had to consider was how big a battery we should get. Of course a bigger battery gives our system a longer life, but this comes at the expense of money and also makes the system larger and more difficult to move around.

To further elaborate on our system, we are required to take multiple 2D images of an animal that is within the range of our system. We need to take multiple photos of the same animal from different angles to construct the 3D image for the purpose of analysis by the researchers. For taking multiple photos over a long time, we need multiple cameras that consume very little

power. These cameras must also be connected wirelessly for proper photo synchronization. We need synchronization to a high degree of accuracy because the animals are always moving, and we can only construct a high-quality 3D model if the photos are taken at the same instant. For wireless connection within our system, we considered using either Wi-Fi or Bluetooth. Wi-Fi has a bigger range (up to 300 feet) than Bluetooth and it can connect many devices together through a DHCP server, whereas Bluetooth can only connect 7 or 8 devices with the present technology, which does not meet our scalable requirement. However, Wi-Fi does use more power than Bluetooth, but not by much, and we can lower power consumption within our system through using a less intensive Operating System, disabling Input/output ports, etc. which is discussed in other sections.

To connect the cameras wirelessly through Wi-Fi, we attach them to a separate module or computer. We decided to attach the camera module to a Raspberry Pi. The other alternative we considered is to network the cameras themselves (that is trying to build around some pre-existing camera trap). Adding networking to a camera trap is much more difficult and complex. It also does not serve a purpose in the long term. Firstly, every camera has different hardware and software, so to network them, we must commit to a technology that can ultimately defeat the purpose of having a scalable, open, and configurable system. Not to mention, adding all the features to the cameras themselves makes the system as complex as a computer. Presumably, after adding all the features, it will consume as much power as a normal Operating System. Also, if, at a later time, you want to make the cameras more intelligent based on synchronization, it will be limited by the capabilities of the cameras which are not within our control. Also, adding a Pi to the camera module gives us the power of making a more robust, configurable, scalable, and open system. Every Pi comes with an operating system that runs a TCP/IP stack. So, we can implement any kind of networking protocol or software package/product on top of this if our requirements change. The module we chose to use is the Raspberry Pi Zero W. This Raspberry Pi model uses very little power and can last a long time. The Pi also has a built-in Wi-Fi chip that can connect to Wi-Fi. The DHCP server for Wi-Fi will be set up on the central Raspberry Pi as part of an ad-hoc network.

These photographs are synchronized to within 0.1 seconds, with an accuracy of 95%. These photos will be used to generate an accurate 3D reconstruction of the animal captured using a 3D reconstruction software such as Meshroom or Blender. The photos must be in good quality to generate an accurate 3D model. We have used a Raspberry Pi Camera Module V2, which can take photos with 8K resolution, 3280 * 2464 resolution. The photo session (when multiple cameras capture an image of an animal at the same time) will be triggered by a central camera module(s) using a PIR (Passive Infrared) sensor (which detects motion by an animal within the camera's field of view). The central camera broadcasts a message to all other cameras through a protocol called MQTT (Message Queueing Telemetry Transport). We can ensure that timing is correct among all cameras if we add an RTC (real-time clock) into the system. The Pi does not have a real-time clock. It gets from the Internet from an NTP (Network Time Protocol) server. Since our system is designed to work without the Internet (is not needed), we will add an RTC on the main Raspberry Pi, which will serve as the central clock for our captive camera network.

Another part of the project is telemetry. For this project, we plan to send updates about the status of the system within a couple of kilometers (1 – 3 km). This is useful because the researcher may not want to approach the area where the cameras are set up since that could leave a scent trail that could scare away animals. To accomplish this, we will use XBee Radio Frequency modules to send information about the network such as battery life and statistics on photos taken Our block diagram of the system is shown in Figure I.

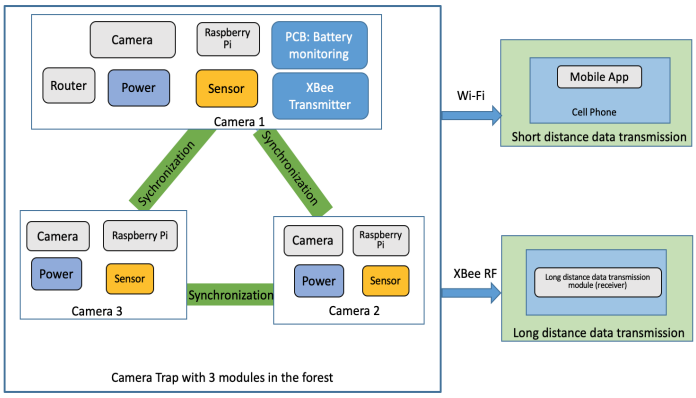


Fig 1. Block Diagram of the Integrated Camera Trap System

B. Technical Block – Networking and Synchronization

For our system, we are using Wi-Fi to connect each of the different modules so they can send messages between each other. The network design is based on a central and edge paradigm, in which one module listens to commands and sends the commands to the other edge nodes within the network. For this reason, we have chosen a star topology for our network. The central module is the main node and the edge cameras are the peripheral nodes. The central module runs a DHCP server for the entire network of camera modules. How the cameras will send information between each other is with Message Queueing Telemetry Transport (MQTT) Protocol. MQTT is a lightweight messaging protocol in which devices send messages to each other with a publish and subscribe model. One module (the central module) will act as the publisher, who will broadcast a message to the other edge modules, who will serve as subscribers. The subscribers connect to a topic or channel where they will receive information. The distribution of messages through these different channels is maintained by a MQTT Broker. For the MQTT Broker, we are using a free, open-source software called Mosquitto which provides the service for our network. Figure 2 shows the model of the MQTT Protocol The broker is also going to run on the central module. A passive infrared (PIR) sensor is attached to the central module. [17] When the PIR sensor detects an animal within the camera's field of view, the central module will send a message using MQTT to the other camera modules connected to the MQTT Broker and the Wi-Fi network. The central module will start the process of taking a photo. When these edge

modules receive the message, they will start the process of taking a photo. The process is asynchronous, the central module does not wait for the edges to respond. For that reason, the whole system is responsive to new events, like another animal triggering the cameras. An asynchronous process does not mean that the photos themselves will not be synchronized. This whole process takes less than 0.1 seconds without the overhead of being monitored by the central module. This allows taking synchronized photos without interruption within 0.1 seconds latency. In the photo session, all cameras have a photo of the animal at that moment. The idea is to get all the images of the edge cameras onto the central module. How we achieve this is through FTP (File transfer protocol). Each camera module will log into the main camera module and place its photo in a separate directory for the photo set. The photos have a camera number, set number, and time by which they can be tracked, identified, and processed.

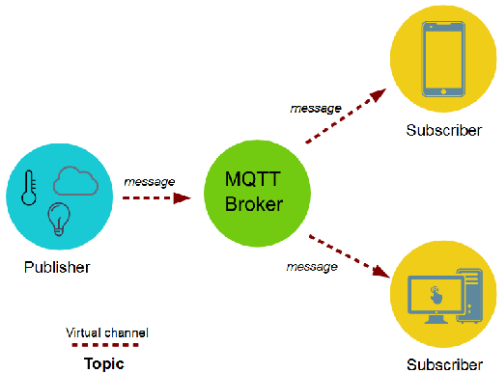


Fig. 2. Structure of a Publish-Subscribe model using MQTT

The central module will have a static IP address and a hostname which the edge modules can easily connect to. We will set up a local name server for naming the central module, the MQTT Broker. This will ensure that even if the IP address of the broker and the central module changes, it will not affect the operation of the system. The system will be flexible to host the different servers on different machines without changing the programs. However, the edge modules can run on dynamic IP addresses managed by the DHCP server. If, however, the client feels the necessity to have a set naming convention for the edges, it can be done with minimum configuration changes.

To ensure synchronization, all modules need to have the same set time. However, the Raspberry Pi does not have an in-built clock [12]. The clock is set using the Internet. The clock is synced from an NTP (Network Time Protocol) Server. The problem with this is that we cannot ensure Internet access in remote environments where our devices will be placed. As a result, every device in our network will have an arbitrary software clock time once they are rebooted. In addition, since the modules themselves are not synced together, and software clocks are unreliable and prone to delay, each module will have different times, which is a huge deterrent to measure our goal. Also, it will be impossible to find out the time and synchronization of the photos, which will be used to make the 3D model. For this reason, we are adding a real-time clock (PCF 8523) [18] to the central module, which will keep track of the time. The edge modules can receive the real-time from the central module through SSH (Secure Shell). The edge modules will routinely query the central module for the real-time with a cron job (automated running script) and will update their software clocks to the main module's time.

C. Technical Block - Power Consumption and Monitoring

An important specification we need to meet is having good power consumption because that enables us to either get longer total battery life or buy cheaper (low capacity) batteries. The specification was 72 hours. We are using 26800mAh batteries, which give us enough power for the system to meet this requirement. [13]. For MDR, we agreed on meeting a 24-hour battery life specification with our evaluators and we exceeded this significantly.

To give a rough estimate in order to meet our 72-hour spec we will need to consume roughly 370mA on average. This is somewhat less than what a full Raspberry Pi (3B+) does while running code, so we had to figure out ways of reducing current draw. The primary thing that can be done is manually disabling certain buses on the Raspberry Pi chip and this reduced power considerably. Turning off the bus that goes to the USB ports can kill about 130mA of power, likewise, killing HDMI reduces by about 20mA. All these methods of reducing power should not affect performance for our system.

So, that takes us to about maybe 220mA, and if we want to get more we have a couple more easy things we can do, but there are also certain power-saving techniques we could do on top of that to get really-low power, but some of those would come with downsides. For the tricky things, we could do things like underclock the CPU and try and change the way our code is structured. Also, we could do things like try to kill daemons running in the background, turn off LEDs or change to a more lightweight operating system. These latter things do not make too big of a dent, but they can add up, so that overall we might be able to get something approximating 150mA if things go well. This would mean a 172-hour battery life, which would be 148% more than what we originally had as our specifications.

One other feature of the Raspberry Pi that should be mentioned is the lack of any way to measure current from the Pi itself (it can display voltages on certain buses but there is no way of displaying current). In order to get these measurements, we used a USB ammeter [14]. Part of our hardware component is an analog ammeter that will feed back into the central Raspberry Pi via the GPIO pins. This will allow the raspberry pi to know how much power has been consumed and can send this information to the software user interface and through long-distance transmission.

D. Technical Block – 3D Reconstruction

Part of the software running on the Raspberry Pi will have to calibrate settings for the photography so that the photos that get taken of the animal are suitable for 3D reconstruction. There is a 'sport' mode which is better for photographing moving objects than the default settings. But even in that case the shutter

speed can be too slow because the algorithm is trying to still get a good amount of lighting. This will make the 3D reconstruction difficult because the photographs need to be nearly simultaneous for the reconstruction to work since the software uses common reference points. So, if the object is in different places in different photos, it will confuse the software.

Good shutter speed for our purposes is approximately 8ms, but this can cause modest dimming/darkening. Since Raspberry Pis are not able to open the aperture, we would have to compensate for this by increasing ISO/Brightness which reduces how nice the photos will look but won't necessarily make them harder to use for 3D model generation. It may also help to artificially boost contrast since that would make common reference points/pixels more pronounced. This is because the way the Meshroom (3D Reconstruction Software Works) is that the algorithms try and look for common reference points and shapes in all of the input photos and then use that to estimate the placement of the cameras[**Citation 15].** Also, since each camera comes with a different focal length and lens width, the program can estimate the volumes of different objects that are in the scene.

Currently we can configure our cameras so that they can communicate, and all choose the same photography setting, and that worked in our MDR for getting good photos, but we may want to improve on this.

To demonstrate that our photos were of the appropriate quality we made a 3D model of a stuffed moose we had by taking photos of it using our raspberry pi camera. We did two versions of this. First, we took 26 photos using 'auto' mode, then we took 32 photos using our custom anti-blur settings (8ms shutter speed, 60/100 brightness, 60/100 contrast) and that also was turned into a 3D model. Before, we started Irschick said that we would probably need about 25 – 30 cameras to get a good model of an animal since there are a lot of different angles and crevices to cover.

In both cases, the model was successfully generated, and we got good detail on all parts of the animal (legs, underbelly, back, sides). We have attached our 3D reconstructions which are in the standard .obj format to our website.

E. Technical Block – Power Monitoring System

The power monitoring system is composed of a current sensor, model ACS-723 [3], an analog to digital converter, model MCP-3008 [20], two USB connectors, some resistors, and some capacitors. It is attached to the central Raspberry Pi. The main goal is to collect the power information so that the users can receive this information through long-distance transmission. Then, they would know when the rechargeable battery should be replaced—the connections are shown in Figure 3. Two USB connectors connect to the power output of the power brick and the power supply of Raspberry Pi Zero. The current is measured by the current sensor ACS 723; the output, analog voltage, of the current sensor, is converted into digital signals by the analog-to-digital converter. The ADC connects to the Raspberry Pi through Serial Peripheral Interface (SPI), so digital information can be received by the Raspberry Pi. The voltage is obtained using the equation V = x*3.3/1023, where x is the output of the ADC. The instantaneous current measurement is obtained with the formula I = (V - 2.5)/0.4, where V is the output of the current sensor. By summing the instantaneous current, the current measurement over a specified period can be computed, and multiplied by 5, which is the constant output voltage of the GPIO pin of Raspberry Pi, to obtain the power information. Power usage is calculated from dividing the power information by the total amount of power of a power brick. The users will know when they should replace the power brick after they obtain power usage.

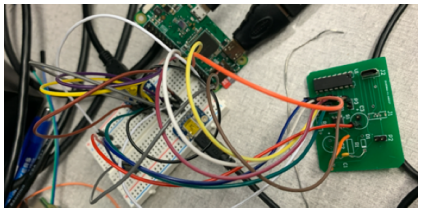


Fig. 3. Power Monitoring System.

F. Technical Block – Implementing System to Detect Presence of Animals Using Multiple Sensors

Our goal is to trigger the system to take pictures when there is an animal in the middle of the field of view of all cameras. Ideally, the modules of the camera trap will be arranged in a circle as shown in Figure 1 (consider a small system with four modules for now.) The field of view of the camera on each module is 62.2 degrees, [4] and the maximum distance between cameras is 8 meters apart based on our project specification. Therefore, the common field of view of all cameras will be a circle-like area in the center of the camera trap with a maximum approximate diameter of 4.83 meters., represented as the blue area in Figure 1. To avoid false-positive triggering of using only one sensor, we decided to use two sensors for the triggering system. For example, shown in Figure 2, assuming there is only one sensor, the appearance of animals outside of the common field of view (blue circle) could trigger the system too. However, the use of two sensors solves the problem with the fields of view of two sensors overlapping inside of the blue circle, as shown in Figure 2, with two sensors installed in two separate camera modules at a 90-degree angle ideally. One sensor is on the central module and the other is on a secondary module. The system could only be triggered when both sensors sense something. The sensors we use are Passive Infrared (PIR) sensors, which measure infrared light radiating from an object in the field of view. [5] By assuming the size of animals is small or equal to 1.5 meters, we could calculate that the sensing area has a diameter of 1.83 meters and the field of view of the sensor is 25.8 degrees in this case. Therefore, with the system set up as shown in Figure 2, the system could successfully trigger the modules to take a picture only if there is an animal in the common field of view of all cameras.

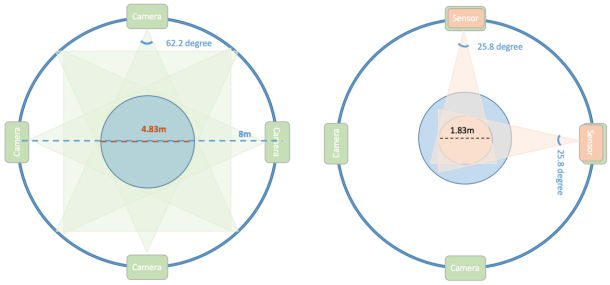


Fig 4. (left) Common view of all cameras
(right) Arrangement of two sensors

G. *Technical Block – Long Distance Transmission*

Long-range transmission is an add-on feature of our system. We aim for transmitting a short message about the remaining battery life and the number of pictures has been taken to the user every day in a range of two miles so the user could know when the battery should be replaced. For next semester, we will use the XBee modules, which transmit data via radio frequency [6], to accomplish this goal. These XBee modules have been FCC certified. The transmitter of the XBee module will be attached to the central module since we will have a battery monitoring system on the mater module too. The receiver of the XBee module could be connected to devices like a laptop or a mobile phone to receive the message from the transmitter.

H. *Technical Block - Mobile App*

The mobile app was created so that a non-expert could operate the system, and easily view photos. The app was designed for Android phones, and designed using android studio. The reason android was chosen was that it allows the app to be easily shared with new phones, in contrast to the iphone system which requires the app to be distributed through the app store, which is an involved process.

The basic functions of the app are debugging, turning on the system, and viewing the photos and statistics. The debugging part is important because it is possible that one of the sensors or cameras could not be working, or the network could be not working too. If the researcher set up the system in a location and thought he was properly turning on the system without debugging then he could come back a day later, and the pictures wouldn't be there. So allowing for debugging makes sure the researcher has a basic awareness of what's going on.

Once the researcher is sure that the system is set up properly he can enable the system to take photos, and can leave the system to work on its own. Another part of the system is monitoring and viewing the photos while the system is running. The app allows for moving the photos from the master raspberry pi to the phone, and also see how many photos total have been taken. There is also a way to view statistics on how much power is being consumed, and this information comes from the PCB. All of this information allows the user to have confidence that the system is working correctly in real time.

In order for the app to function there needs to be code working on the phone (client side) and on the raspberry pi (server side) that sets up a socket and transfers information back and forth. On the client side the coding is in java and the UI is in the standard Android studio format, and what happens when a button is clicked is that a text string is sent over the connection to the Raspberry Pi telling it what to do. On the Raspberry Pi side the code that receives the messages from the phone interacts with the code that controls whether to take a photo or not, and also can transmit data back to the phone. For instance, if the Raspberry Pi gets a request to send over the photos it will do that over the same socket connection, and deposit the photos in the SD card through FTP.

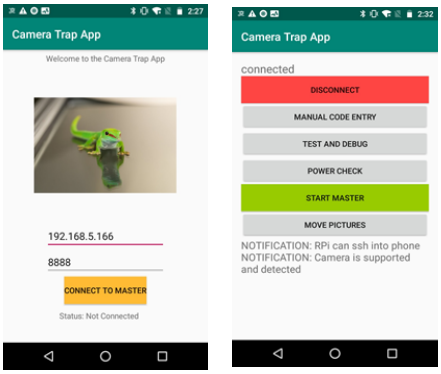


Fig. 4. Pages of Mobile App

III. THE PRODUCT

A. *Product Overview*

For our final product we built on what we did during our first semester, and primarily added on features that made the system easier to use and more reliable.

For our hardware we created an enclosure and also created a PCB to measure power consumption. The purpose of the enclosure is to make it so that the person using our system (a biologist) can deploy our system remotely without being affected by the weather. The enclosure has been designed so that it can fit the battery, and the camera module, and so things are organized for the researcher and there aren't a tangle of wires.

The purpose of the PCB which we added this semester is to measure power in a way that can be sent to the user. One way to measure power is to use a USB ammeter, and these are fairly cheap. But the data produced by the ammeter is only shown on an LCD display, and can't be recorded by the user. Therefore what our PCB does is it reads the information on the current then feeds that information back into the pi through the GPIO pins, and then that information can be stored and/or sent to the user to analyze battery power consumption.

For software this semester, our team worked on creating an app and integrating that with our existing software in order to communicate and control what the Master raspberry pi was doing. The app is designed to be easy to use and allows the user to set up the system so that he or she can check and make sure that the motion sensors are working, and the cameras are working. Also, when the researcher enables the system the researcher can get notifications, and also view the pictures on the phone. This makes it much easier to use for the average person who may not know about how to use a command line tool.

Lastly, one more subsystem we added this semester in order to make our system into a better product was adding a long distance transmission system that can send information to the researcher up to about 2km. This means the researcher can get information on battery life and also see how many photos were taken over the previous day. This is important to the researcher because humans can leave scent trails that animals may pick up, so they may be wary of going near an area that the researcher had previously visited.

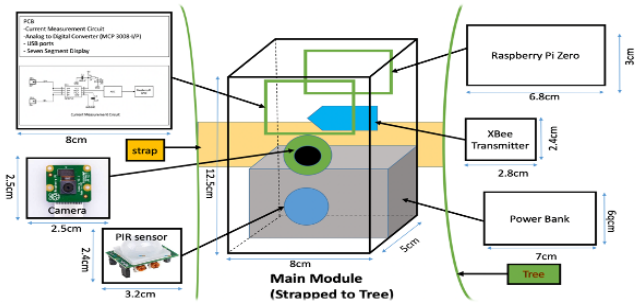


Fig. 5. Product Sketch

B. *Electronic Hardware Component*

A schematics of the power monitoring system is demonstrated in Figure 5. The design is done using Altium and fabricated. A circuit is built in a breadboard according to the schematics, using a current sensor ACS 723, an analog-to-digital converter MCP 3008, a Raspberry Pi Zero, and several capacitors and resistors. A voltage source and a resistor are utilized to test the efficiency of the system, and the values computed by the Raspberry Pi is similar to the theoretical results, by dividing the voltage by the resistor values. Then, the printed circuit board and a hand-soldered protoboard are implemented to measure the current delivered from the power brick to the Raspberry Pi Zero. The physical implementation is shown in Figure 3. From the power monitoring system, the Raspberry Pi Zero can receive continuous reading about power consumption.

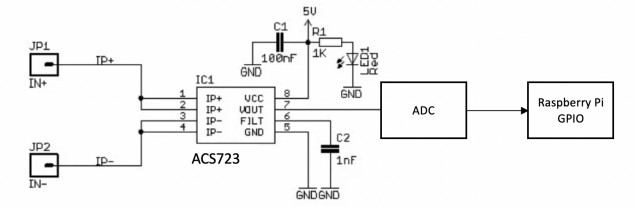


Fig 6. A sample schematics of the powering monitoring system

C. *Product Functionality*

For our CDR we were able to implement everything shown in our block diagram in Fig. 1. For CDR our demo consisted of 3 modules, a phone, a router, a PCB, and a long distance transmission system. During our demo we showed that the system was synchronized, as was the case during MDR, by holding up phones with clocks on them to demonstrate that our system was synchronized to within 100ms. We also had a demo for our PCB which involved showing that the power readouts could be seen on the phone. For our router we had it set up so that anyone could connect, and we showed this using our phones.

One critical part of our demo was being able to run the system via the mobile app which we did. Before our demo we were able to take screenshot videos of what it was like to use the mobile app. These videos show that the system can be enabled by a user, and also when the user requests it the photos can be moved over to the phone and viewed. We included these videos as links during our CDR presentation.

One part of our demo that only partially worked however was the router system we had set up. Originally we had wanted to create a router that ran off one of our Raspberry Pis, but since this was unreliable we went back to using a store-bought router for our demo since it was more reliable.

D. *Product Performance*

All of our specifications were met during CDR and we demonstrated this by repeating our demos from MDR and also adding new demos to show that the app worked, the enclosure worked, the long distance transmission worked, and the PCB worked. We recorded videos of all of our demos and included these as links in our CDR powerpoint.

The first and most basic demo we had was proving timing synchronization and we did this by showing a clock taken from all three cameras, and this was included in our CDR powerpoint as a set of three photos. For power system monitoring we far exceeded our requirement since we switched to Raspberry Pi zeros which mean that we were using about 150mA of power which was far under our requirement.

The other requirements we met this semester were related to the hardware we incorporated this semester. For our power monitoring system we had a PCB and this gave accurate readouts to the phone that matched the current reading the USB ammeter was giving us, so we have high confidence in its accuracy. For the long distance transmission system we demonstrated the range by sending messages from a known location at umass (near physical sciences) to the art center and calculating the distances, and we included this video in our CDR presentation.

The last requirement we had was to demonstrate the functionality of the mobile app, we showed this during our in-person demos, but also took recordings of the app being used and included this as links in our presentation.

Finally, we had to demonstrate that our project was portable, compact, and could be used in remote locations. This was accomplished by using 3D printed enclosures that were waterproof and could store the cameras and the batteries. This means that the camera modules would be easy to handle and move around.

IV. CONCLUSION

We built a system of Raspberry Pis where each Raspberry Pi takes photos within 0.2 seconds. The MQTT is the network protocol used to exchange messages between a central Raspberry Pi and edge Raspberry Pis through IP. Then, we test the power consumption of one Raspberry Pi. A USB amps and power meter measures the power consumption for 15 hours

without consistently taking the images, and the average is 127mAh. Based on the information, each camera will be able to run for at least 24 hours with a portable battery of 26800 mAh. There is a hardware user interface. It keeps track of how much power is left in the battery of a camera module. We also test infrared triggering systems. We first initialize only one infrared sensor, attached to the central Raspberry Pi. Once the sensor detects any motion, all three cameras will take images. Then, we add one sensor to one edge Raspberry Pi. Only when two infrared sensors both detect motion, then all the cameras take photos together. Adding one sensor to the system ensures animals are in the common views of all cameras when the Raspberry Pis take a photo. Then, we transform 2D images of a toy moose into a 3D model. The 3D model presents meaningful information on the body of the moose and demonstrates that the image quality is sufficient to create a great 3D model.

In the following semester, we will both improve on what we have done and add new features to our system. For improving the system, we will test the power consumption of cameras continuously taking photos and transmitting data. We want to know what the battery life in the worst case will be. At the same time, we must control the number of triggers when animals stay in the view of the cameras to avoid unnecessary memory usage and battery usage. The shutter speed of cameras is also a concern because blurring might occur in the image depending on the speed of the animal. Moreover, the new features will include a mobile user interface, a battery monitoring system, a long-distance transmission system, and outdoor implementation. Every new feature gets us closer to the goal of a robust and simple system for the users, but it also led the main problem, which is the integration of individual components. More tests will be done next semester to check if the system altogether.

ACKNOWLEDGMENT

We would like to thank Professor Duarte for being our advisor for this project. He has given a lot of insight and guidance in this project. We would like to thank our evaluators Professors Irwin and Vouvakis for their useful questions and feedback. We would like to thank Professor Hollot, Baird Soules, Chuck Malloch, and Francis Caron for their advice and setting up and maintaining the SDP program. Last and most importantly, we would like to thank Professor Irschick and Professor Schmidt for the idea of the project and their continuous support, and the help with their invaluable knowledge in the subject domain of camera traps.

REFERENCES

[2. "Overcoming the Challenges of Studying Endangered Animals," The Scientist Magazine®. [Online]. Available: https://www.the-scientist.com/careers/overcoming-the-challenges-of-studying-endangered-animals-64368. [Accessed: 19-Dec-2019].

[3. S. Bs, "4x4 Matrix Membrane Keypad (#27899)," p. 5.

[4. "ACS723LLCTR-05AB-T Allegro MicroSystems | Sensors, Transducers | DigiKey" [Online]. Available: https://www.digikey.com/product-detail/en/allegro-microsystems/ACS723LLCTR-05AB-T/620-1641-1-ND/4948877. [Accessed: 19-Feb-2020].

[5. "Field of view of V2 camera - Raspberry Pi Forums." [Online]. Available: https://www.raspberrypi.org/forums/viewtopic.php?t=154155. [Accessed: 19-Dec-2019].

[6. "Passive infrared sensor," Wikipedia. 10-Dec-2019.

[7. "XBee 3 Pro Module - RP-SMA Antenna - WRL-15131 - SparkFun Electronics." [Online]. Available: https://www.sparkfun.com/products/15131. [Accessed: 19-Dec-2019].

[8. IUCN 2019. The IUCN Red List of Threatened Species. Version 2019-3. http://www.iucnredlist.org. Downloaded on 10 December 2019.

[9. "2019 Camera Traps for Researchers," Trail Cam Pro. [Online]. Available: https://www.trailcampro.com/pages/camera-traps-for-researchers.

[10. D. Soni and A. Makwana, "A Survey On MQTT: A Protocol of Internet of Things," International Conference On Telecommunication, Power Analysis And Computing Techniques, Apr. 2017.

[11. H. Mohammadmoradi, O. Gnawali, and A. Szalay, "Accurately initializing real time clocks to provide synchronized time in sensor networks," 2017 International Conference on Computing, Networking and Communications (ICNC), 2017.

[12. "TRAIL CAMERA SELECTION GUIDE," *Trail Cam Pro*. [Online]. Available: https://www.trailcampro.com/pages/trail-camera-selection-guide.

[13. "Adding a real time clock," *Adafruit*. [Online]. Available: https://learn.adafruit.com/adding-a-real-time-clock-to-raspberry-pi.

[14. "Prime 26800mAh 5.5A 3-Port Power Bank," *Ravpower*. [Online]. Available: https://www.ravpower.com/products/rp-pb41-26800mah-power-bank.

[15. "YOTINO USB Volatage/Amps Power Meter Computer Cables Tester Multimeter Test Speed of Charger Power Bank," *NorthStar Data*. [Online]. Available: http://northstar-data.com/Multimeter-Test-Speed-of-Charger-Power-Bank-YOTINO-USB-476256/.

[16. "Natural Feature Extraction," *Alice Vision*. [Online]. Available: https://alicevision.org/#photogrammetry.

[17. *Digital Life*. [Online]. Available: http://digitallife3d.org/.

[18. "PIR Motion Sensor," *Ada Fruit*. [Online]. Available: https://cdn-learn.adafruit.com/downloads/pdf/pir-passive-infrared-proximity-motion-sensor.pdf.

[19. "Real-Time Clock (RTC) and calendar," *NXP Semiconductors*. [Online]. Available: https://www.nxp.com/docs/en/data-sheet/PCF8523.pdf.

[20. J. Crothers, "Victoria's threatened tiger quolls prove camera-shy in conservation bid," *ABC News*. [Online]. Available: https://www.abc.net.au/news/2017-02-11/conservationists-hunt-to-catch-the-elusive-tiger-quoll-on-camera/8258990

[21. "MCP3008-I/P Microchip Technology | Integrated Circuits (ICs) | DigiKey" [Online]. Available:

https://www.digikey.com/product-detail/en/microchip-technology/MCP3008-I-P/MCP3008-I-P-ND/319422. [Accessed: 19-Feb-2020].

APPENDIX

A. *Design Alternatives*

Large parts of this project were determined already before we started working because Prof. Irschick had a general sense of what he wanted and what the general design of our system should look like. However, our team had independence within this overall framework to make certain design choices and make judgements on what the costs vs. benefits of certain decisions were.

One of the biggest design choices we had to make from the start was deciding whether to base our camera system around using store-bought camera traps or building our own system from scratch. The first approach was suggested by Shira Epstein, and Prof. Duarte brought this up as a possibility initially. There are a couple of issues with this approach. Firstly, having store-bought cameras be networked together would mean we would have to crack open these cameras and start soldering things to them which would allow them to be network. The issue with this is that although we would be able to do this for our SDP project, the goal is for this to be deployed and used by biologists who would not have a lot of technical training. So, it made much more sense to go with Raspberry Pis which anyone can assemble, and where a lot of the software is easy to install.

Also, another issue with the store-bought camera traps is that they use proprietary software and so interfacing with them would have been a challenge. Furthermore, if we want our project to be open source, if we were using these proprietary cameras, that could mean we could be at the whims of these camera-trap companies. If, for instance, the company we were buying from discontinued their camera in a year or two that could mean our project would be in a bad situation because the biology researchers could not get the hardware platform anymore. For all these reasons, it made the most sense to go with the Raspberry Pis as the main platform for our system and not store-bought camera traps.

B. *Technical Standards*

One of the main standards we wanted to follow for our project was having it be open source so that it could be easily replicated by researchers elsewhere. In order to do this we published all of our code publicly on Github, and so anyone can download that.

For our hardware we made sure to use publicly available electronic devices like the Raspberry Pi that are widely used and can be bought easily on Amazon or other online retailers. This means that if future engineers want to improve on our system they can since it is a widely used system, and the parts are easily accessible and cheap to buy.

For software, the code was in Python on the Raspberry Pi side which is standard, and for the Android app we used Android studio code which is also standard and widely used.

For our long distance transmission system we used XBEE which is a signal centered around 2.4GHz which is available for amateur systems like our own.

C. *Testing Methods*

We have done two kinds of tests on our system: unit testing and integration testing. In unit testing, we have tested the central module and edge modules separately for proper functionality such as capturing photographs, response to the sensors, connecting to the MQTT Broker, etc. In integration testing, we have combined the central and edge modules to test the functionality of our whole system to take photographs when an object is sensed in the cameras' field of view by exchanging MQTT signals between the edge and central modules and sending these photos to the central over Wi-Fi through FTP and creating a photo set which can be used for the 3D reconstruction software. This integration testing also included tests of many different parts of our system such as the hardware UI, photo settings testing, multiple sensors testing, proper synchronization, power consumption, and 3D reconstruction.

The first and basic test we did was to test the synchronization of the photo session event to ensure the timestamp between all photos captured in a single session was not beyond 0.1 seconds. To test this, we had all cameras take a synchronous photo of a clock with the time up to millisecond measurement. To test the latency, we read the milliseconds of the clock on each photo of each camera and made sure no two cameras differed greater than 0.1 seconds in time.

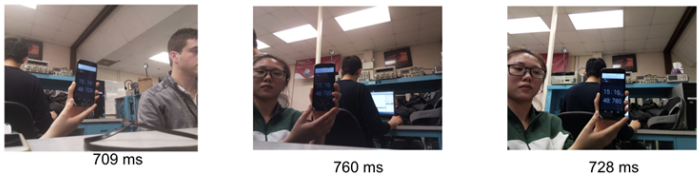
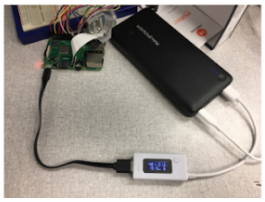


Fig 7. Synchronized images of a mile-second clock

Our system needs to run for at least 72 hours without a power outlet source. The battery we are using for each Raspberry Pi is 26800 mAh, so anything running 1100mA will last at least 24 hours. To achieve our goal of 72 hours, we need each Pi to run under 370mA. Our Pi running the code in low power mode achieved a power of 230 mA, and a 15 hour test of our idle system resulted in 1915 mA (127 mA).

Code running in Low-power Mode: 230mA



15 hour test on idle: 1915mAh (127mA)

Fig 8. Power measurement

Our system can be configured to use more than one sensor for infrared triggering. We tested our system to see if it can use two sensors to trigger the photo session. If the majority of the sensors in the network sense an object in the vicinity of the camera's field of view, they will send MQTT signals among each other and trigger the other cameras to take a photo at the same instant. The figure below shows the model of our system needed for proper photos of animals to be taken,

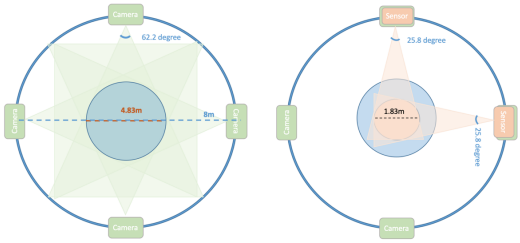


Fig 9. (left) Common view of all cameras
(right) Arrangement of two sensors

The photo below shows the results of multiple cameras taking synced photos at the same instant by motion sensor triggering

Cam 1 Cam 2 Cam 3

Fig 10. Synchronized images of a walking person

One other kind of test we performed was on the mobile application. This was done by running the app many times to check that all the functionality was working properly, and that the debugging, enable function, and the power check were all working and photos were being sent from the main module as shown in Figure 11.

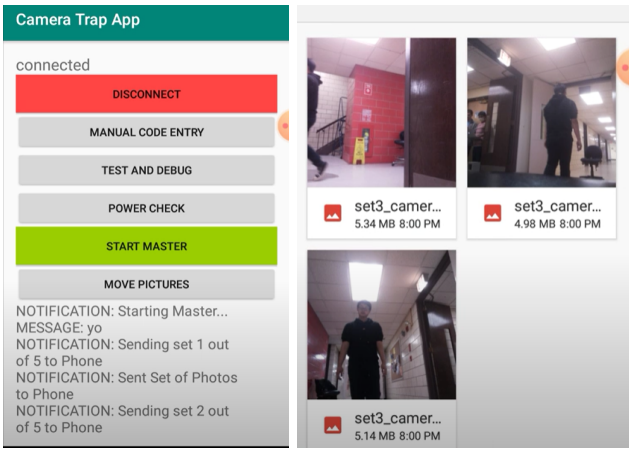


Fig. 11. Photos being received by phone through mobile app

We tested power consumption by sending the power reading from the Pi to the mobile app. Figure 12 shows the output on the mobile app.

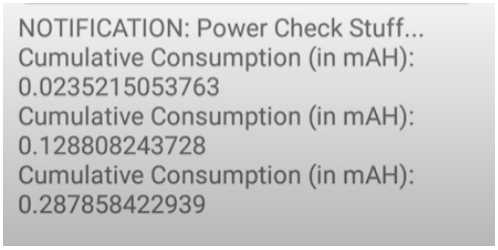


Fig. 12. Output from power monitoring system from mobile app

We also tested long distance transmission by sending a message from long distances (500-800 meters) from a Raspberry Pi through XBee to an Arduino Xbee Receiver. The test output is shown in figure 13 below.

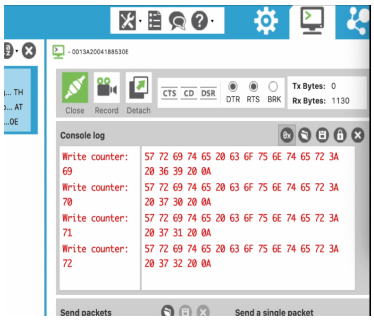
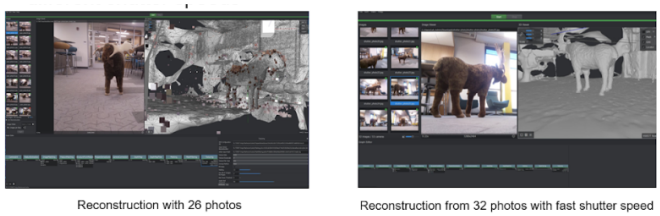


Fig. 13. Test output of Xbee Receiver

We also tested the 3D reconstruction using the synced photos to see whether we can generate a proper 3D model by testing different camera settings. One test consisted of the cameras having default settings with max resolution. The other test consisted of photos captured by a camera with a higher shutter

speed, which produced more accurate results because of less motion blurring. We also considered how many photos are needed in the photo set to generate an accurate 3D model. The photo below shows testing 3D reconstruction on a toy moose.



Reconstruction with 26 photos

Reconstruction from 32 photos with fast shutter speed

Fig 14. 3D reconstruction result of the synchronized images taken by the system using Meshroom

D. Team Organization

We have organized our team into different sections, each plays an important part within our project. Anamitra is in charge of networking and synchronization. Max is in charge of photo settings, 3D reconstruction, power distribution, and software UI. Xiaoyang is in charge of the hardware UI and Minting is in charge of multiple sensors connectivity and long-range transmission. Max is the team manager and Xiaoyang is our PCB lead. We have solid teamwork and organizational skills. We meet every week to discuss how far we have progressed in the project and what we need to accomplish.

E. Beyond the Classroom

Anamitra: I have learned how a microcontroller can be used in a real-life situation. I gained expertise in the area of IoT networks which can communicate with each other to make a decision. I have also learned the engineering design aspect of an autonomous system which can run by itself with very little maintenance. I learned the different tools for manipulating and monitoring IoT networks such as MQTT. I also learned system administration of a distributed system.

Max: During SDP I have gotten experience writing software in Python and using Linux, which is useful to me since I will be working in software after graduation. There are a lot of good resources online (e.g. Message forums, Raspberry Pi Documentation) for learning about the software we are using. One thing I have also had to learn along the way with this project is about photography (Aperture, ISO, Shutter Speed, etc.) and how to get photos that will work well for our 3D reconstruction. Lastly, as team manager I have gained good teamwork and project management experience.

Minting: For this project, I have learned programming languages like Python, MQTT, socket programming for coding the raspberry pi an HTML for creating our website. I also have a chance to work with the hardware like Raspberry Pis and PIR sensors. I also learned to think like an engineer, considering multiple methods and possible outcomes. All these skills will be very helpful in my career because they are very basic and useful.

Xiaoyang: As the hardware lead for this project, I learned about how to use Altium to design circuits. I worked with various hardware components and looked through the data sheets to find out how they should be implemented. I also have more experience in Python. I understand how to write codes according to the physical implementation of the system. The connection between hardware and software is important to produce a desirable outcome.

F. Budget

Part	Total Development Cost (Quantity)	Cost for Each Additional Module
Raspberry Pi Zero W	$120 (6)	20
RasPi 8 MP Camera	$66 (6)	$11
PCB	$40 (1)	$0
XBEE Module	$40 (1)	$0
Portable Power Bank	$0 (Donated)	$45
Router	$70 (2)	$0
SD Card	$42	$6
PIR Sensors	$10 (10)	$1
3D Printed Enclosure	$15 (3)	$5
Total	$403	$88

Table 5: Project budget