Cat Collar - Petronics

Team Number 42 - Taha Anwar,
ECE 445 Project Proposal - Spring 2019
TA: Dongwei Shi

1. Introduction

1.1 Objective:

Problem:
Cat owners do not always have time to play with their cats. The Mousr is a clever solution that accompanies your cat in your absence. However, Petronics still does not have a way to check if there is a direct correlation between a cat's overall activity and Mousr's activity.

Solution:
The Mousr unit developed by Petronics is already able to make event predictions such as "inactive", "engaged", "needs charging" and so on. The motivation for the cat collar is to use the data collected from it to confirm the event predictions by the Mousr in order to assess the cat's actual engagement with Mousr. This will be instrumental for Petronics, as it will allow them to measure the effectiveness of Mousr in engaging with the cat.

1.2 Background

This project is sponsored by Petronics, a company aiming to build interactive robots for pets to engage with them in the absence of humans. They have already developed a prototype called Mousr, that is able to play with the pet for upto 6 hours. It can also record events such as "stuck in a corner", "needs charging", "not engaged", etc. However, they have not done any substantial work to verify the effectiveness of the Mousr unit. As a result, they don't have the results to show any correlation between the activity of the cat and Mousr in operation. So they have asked tasked us to develop a collar that assesses the basic activities of the cat and reports it wirelessly to a remote desktop or Raspberry Pi for data analytics. This solution will play a crucial role for Petronics in their ability to validate Mousr and market it to the users using the statistics found from our dataset.

1.3 High-level requirements list

1. Must be able to transmit and interpret IMU sensor data to Raspberry Pi wirelessly using the ESP32.
2. The entire PCB circuit should be small enough to be integrated within a traditional collar.
3. The battery lifetime should be at least 1 hour to enable collection of sufficient data for a play session.

2. Design

2.1 Block Diagram

The cat's activities will be measured by the IMU sensor. The time-stamped acceleration data will be sent to the ESP32 by the I2C protocol. The ESP32 will save these data in the SD card using SPI protocol. The IMU data will be sent through WIFI using MQTT protocol to the Raspberry Pi, where it will be formatted into a csv file. The Raspberry Pi will also take in time-stamped mousr activities in a csv file. An algorithm will use the two input files to detect a pattern between the two activities. Another program will analyze the IMU data and generate results. A third program will attempt to detect cat's engagement activity in real time and turn on the camera when engagement is detected. Below in figure 1 we have a high level block diagram explaining the communication between different modules in our design. We also have a physical layout of how everything will look while testing as shown below in figure 2.

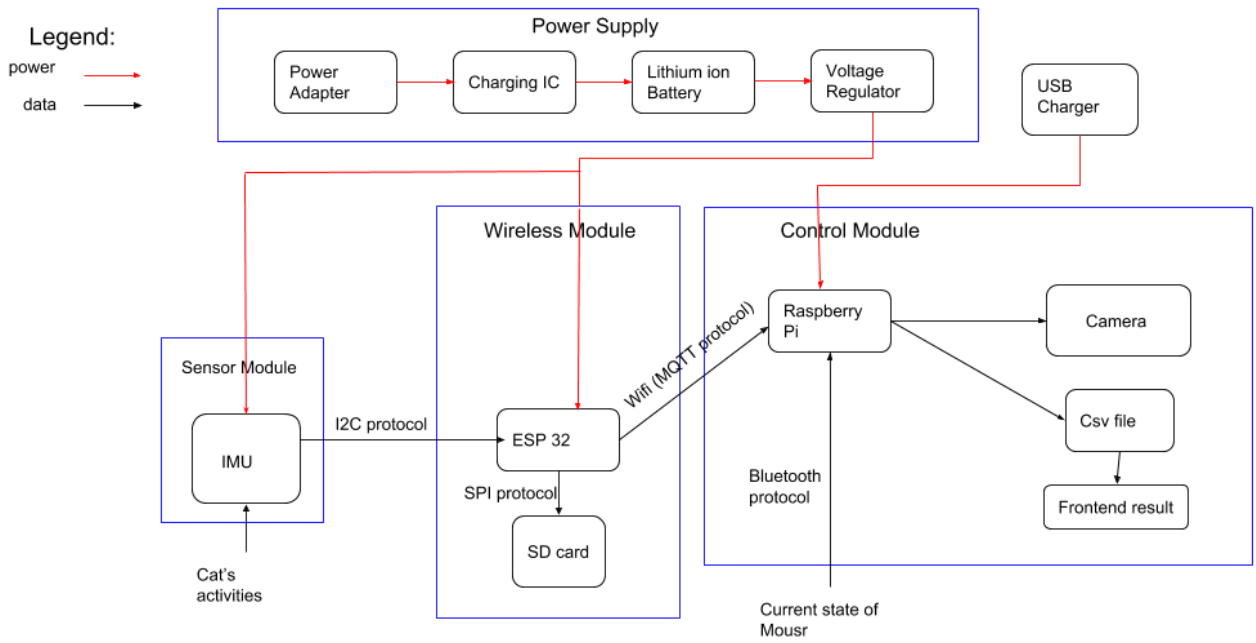

Fig.1. High level block diagram

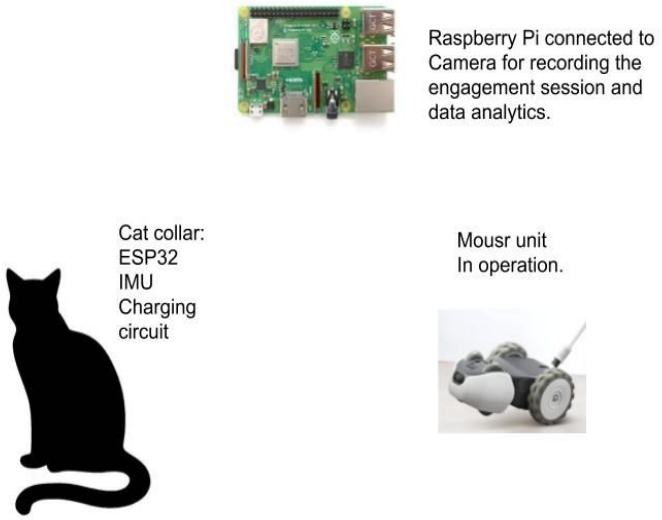

Fig.2. Physical Layout

2.2 Block Design

2.2.1/ 2.2.2/2.2.3 Functional Overview & Requirements and Verifications and Supporting Material

Power Unit:

1) Lithium Ion Battery:
   We will be using a the Sparkfun 3.7 V, 400mAh Lithium Ion Battery in order to power our entire collar circuit [3] . The main components it will need to power are the ESP32 and the IMU. We are estimating ~180 mA of current draw from the ESP and ~6mA from the IMU. Rounding that to 200mA, we see that this will ideally put as at 2 hours of constant usage.

Requirements:	*Verification:*
1) Battery must supply at least 3.3V for a minimum of 1 hour *2) Battery must supply at least 180 mA to the ESP32 and 6.1mA to the IMU for a minimum of 1 hour*	*1)* *a) Connect fully charged lithium Ion battery as the battery shown in figure 3 circuit below* *b) Discharge battery at 400mA for 1 hour (Use R_1 =*

	1.25/(400mA) = 3.125 Ω) *c) Use multimeter to verify that the battery voltage remains above 3.3 V.* *2)* *a) Connect the lithium ion battery as the "battery" in the constant current test circuit (fig. 3) and draw 200 mA* *b) Measure the output current using a multimeter and ensure that there is sufficient current being supplied for 1 hour.*

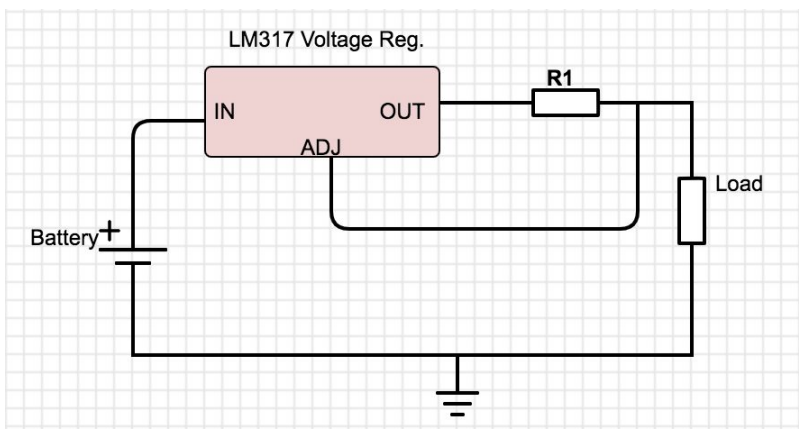

Fig. 3. Constant Current Test Circuit, $I_L = 1.25/R_1$

2) Battery Charging IC:

We will be using the Sparkfun PRT-10217 Lipo charger. The charger charges 3.7V LiPo cells at a maximum rate of 500mA and includes a micro-USB connector, a charging IC, status LEDs, and an appropriate port to connect to the lithium ion battery. We will initially be using this board but we can implement the board on our PCB afterwards. The schematic for its implementation is shown in figure 5 below.

Requirements:	*Verification:*

1) *Must be able to fully charge the lithium battery to 3.7 within 1-2 hours*	1) *Once the battery has depleted, begin charging and monitor how long it takes for the battery to become fully charged*

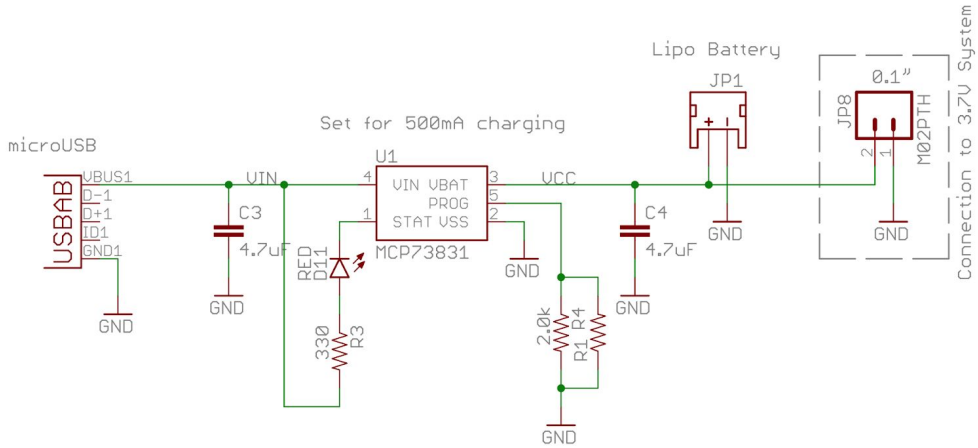

Fig. 4. Battery Charging Circuit Schematic [3]

3) USB charger/Adapter:

The USB charger will just be a wall adapter with a USB cable that has a regular usb port on one end and a micro-USB connector on the other end.

4) Voltage Regulator:

Our voltage regulator should take our Lithium Batteries as an input and output around 3.3 V to the ESP32 and IMU. We plan on using the TLV755P Voltage regulator IC Chip in order to do this. The chip can take input voltage of 1.45 to 5.5 V and provides can provide an output voltage of 0.6 to 5 V with a low 238 mV dropout voltage [6]. The regulator can also source up to 500 mA of current which should be sufficient for our purposes.

Requirements:	*Verification:*
1) *Output a voltage of 3.3V (+/- 5%) to the ESP32 and IMU* 2) *Must be able to output the necessary current to the ESP and IMU (~200mA)*	*1&2)* a) *Connect the output of the voltage regulator as the "battery" in the constant current test circuit (fig. 3) and draw 200 mA* b) *Measure the output using an oscilloscope to ensure that the output voltage is +/- 5% of 3.3 V*

Wireless Module:

A) ESP 32 Microcontroller:

The ESP32 microcontroller will be programmed with the raspberry pi to take the IMU sensor signals and transmit the data wireless to the raspberry pi. This will be implemented using MQTT protocol or data stream with microPython.

SD Card:
To save data on the microSD card with the ESP32, we use the following microSD card module that communicates with the ESP32 using SPI communication protocol. This unit is an extra security in case wireless data is transmitted erroneously. The two modes are SD and SPI.

With 32GB Sandisk Extreme UHS-II card, 6.46 MB/s transfer rate means that the storage can last up to 32GB/6.46MB/s = 4953.56 seconds = 82.56 minutes. Our target battery life is 1hr and 6.46MB/s is the optimal transfer rate, so this is a safe boundary [7].

ESP32 pin	**SD card pin**	**SPI pin**
GPIO14 (MTMS)	CLK	SCK
GPIO15 (MTDO)	CMD	MOSI
GPIO2	D0	MISO
GPIO4	D1	N/C
GPIO12 (MTDI)	D2	N/C
GPIO13 (MTCK)	D3	CS

ESP 32 Development Board:
Initially, we will be using a ESP32 development board for testing purposes which has a built-in voltage regulator and supports USB-based programming. However, we plan on actually designing our custom development board where we will have a voltage regulator IC and a mounted USB connector and ESP32 chip on the PCB. The ESP32 transmits through wifi 802.11g OFDM at 54 mbps at 190mA power consumption [8].

Requirements:	*Verification:*
1) *Must be able to send time-stamped ESP32 data to Raspberry Pi wirelessly using MQTT protocol* 2) *Must be able to store the sensor data to SD card*	1) *We will simply start by printing an acknowledgement message from the Raspberry Pi, once it has received the data packet via MQTT protocol. Then move on to sending the time-stamped sensor data.* 2) *Output the csv file's acceleration data into a graph using excel and compare that with our actual IMU motion: stationary is 0 m/s^2 and free fall is 9.8 m/s^2*

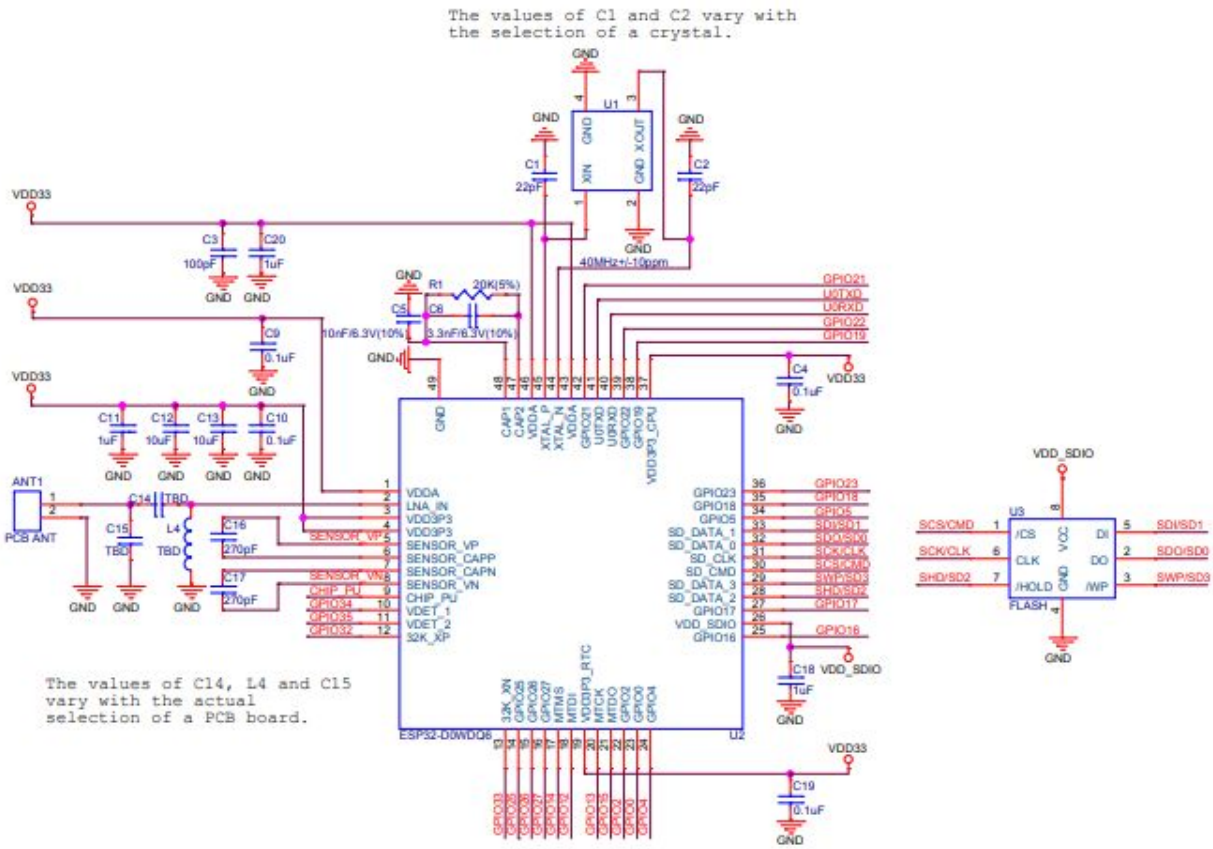

Fig. 5. ESP32 Wroom Schematic(Wireless Module)

Sensor Unit:

1) IMU Sensor:

The cat's activities will be measured by the IMU sensor. The time-stamped acceleration data will be sent to the ESP32 by the IC2 protocol. The IMU has 4 ranges of acceleration to select from: *±2/±4/±8/±16 g linear acceleration [4],* where each has a different sensitivity, with a larger scale having greater sensitivity. Since a cat is not expect to have a linear acceleration exceeding +-2g, even in jumping from free fall, we will choose +-2g as our scale.

Requirements:	*Verification:*
1) Must be able to send information to ESP32.	*1) We will keep the circuit stationary on a flat surface, so that it reports only an acceleration of g m/s^2 in the z-axis and confirm that is reflected in the output values of the ESP32, which is stored in a CSV file in the SD card.*

Control Module:

This part of the system contains the Raspberry pi and a Picamera. The raspberry pi receives the IMU data from the esp32 Wifi signals. We will be using the Picamera to record the entire play session to serve as ground truth for our algorithm. The raspberry pi simultaneously receives bluetooth signals, the Mousr's activity, from the Mousr.

Mousr activities and cat's sensor activities are inputs to our analysis program. This software stores the timestamps of these two activities every 5ms in a csv file. It also runs an algorithm at the end of each session to determine when the cat is in engagement. At the end of each session, lasting approximately two hours depending on the battery life, an analysis report will be generated. The report consists of various information like, how much time the cat spends on sleeping, walking or engagement with Mousr, and what are Mousr's activities during these scenarios. This report will be generated in a csv as well as a frontend web page hosted locally.

1) Raspberry Pi:

The Raspberry Pi will act as the processing unit to interpret the data received from the IMU as mentioned above.

Requirements:	*Verification:*
1) *Must be able to communicate with ESP32 through bluetooth or wifi*	1) *The ESP32 transmits through wifi 802.11g OFDM. Start with sending a simple string message from ESP to Pi. Then timestamped data at at 54 mbps. [8]*

2) Software Module:

Requirements:	*Verification:*
1) *Must generate timestamps of IMU sensor data and mousr activity in a csv file* 2) *Must detect cat's engagement with mousr with 75% accuracy* 3) *Must turn on accessory camera when cat's engagement is detected*	1) *Check the csv file to verify that the output contains rows of data, where the columns represent time, acceleration data from IMU and mousr activity.* 2) *We will check our final prediction result against ground truth from our camera. Perform the experiment 20* times to obtain a percentage accuracy 3) We will check whether the camera turns on when the cat is engaging. This will be performed as many times as possible.

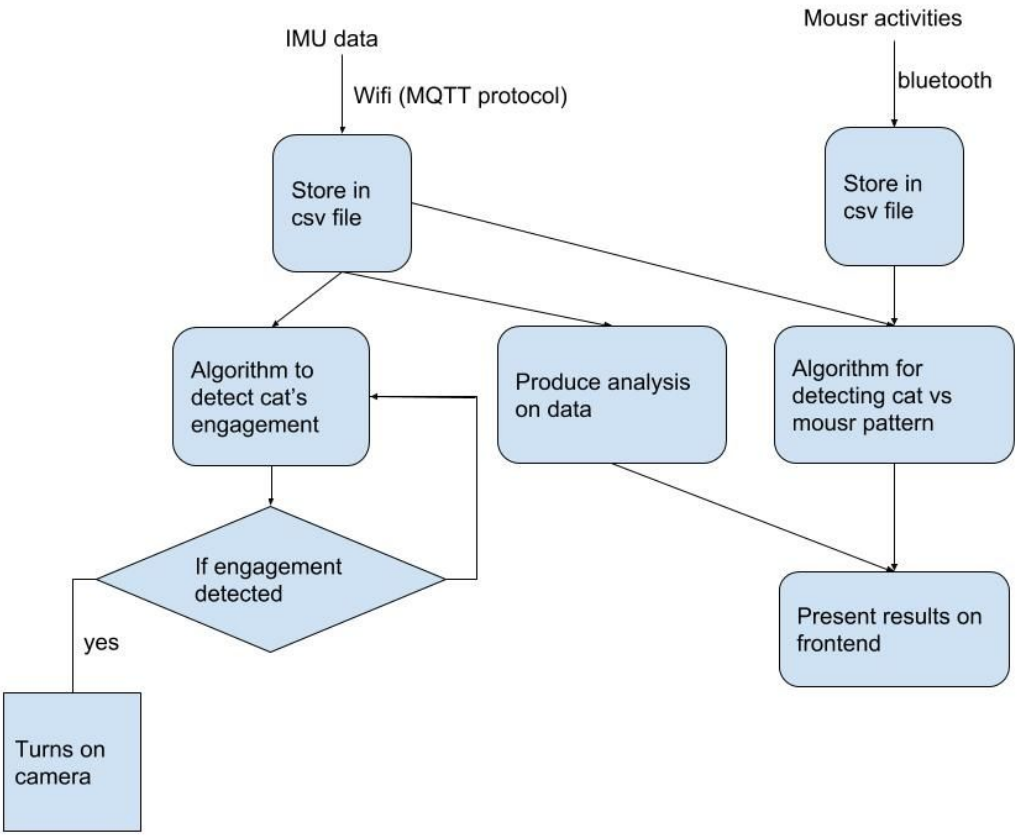

Fig. 6. Overall Software Flowchart

An algorithm will take in the IMU data csv file and mousr activities csv file and detect a pattern between the two activities. Another program will analyze the IMU data and generate analysis on the cat's activities, such as average velocity and total playtime, etc. The results from these two programs will be presented on a frontend platform. A third program will analyze the IMU data csv file and try to detect when the cat is engaging with the mousr and turn on the camera.

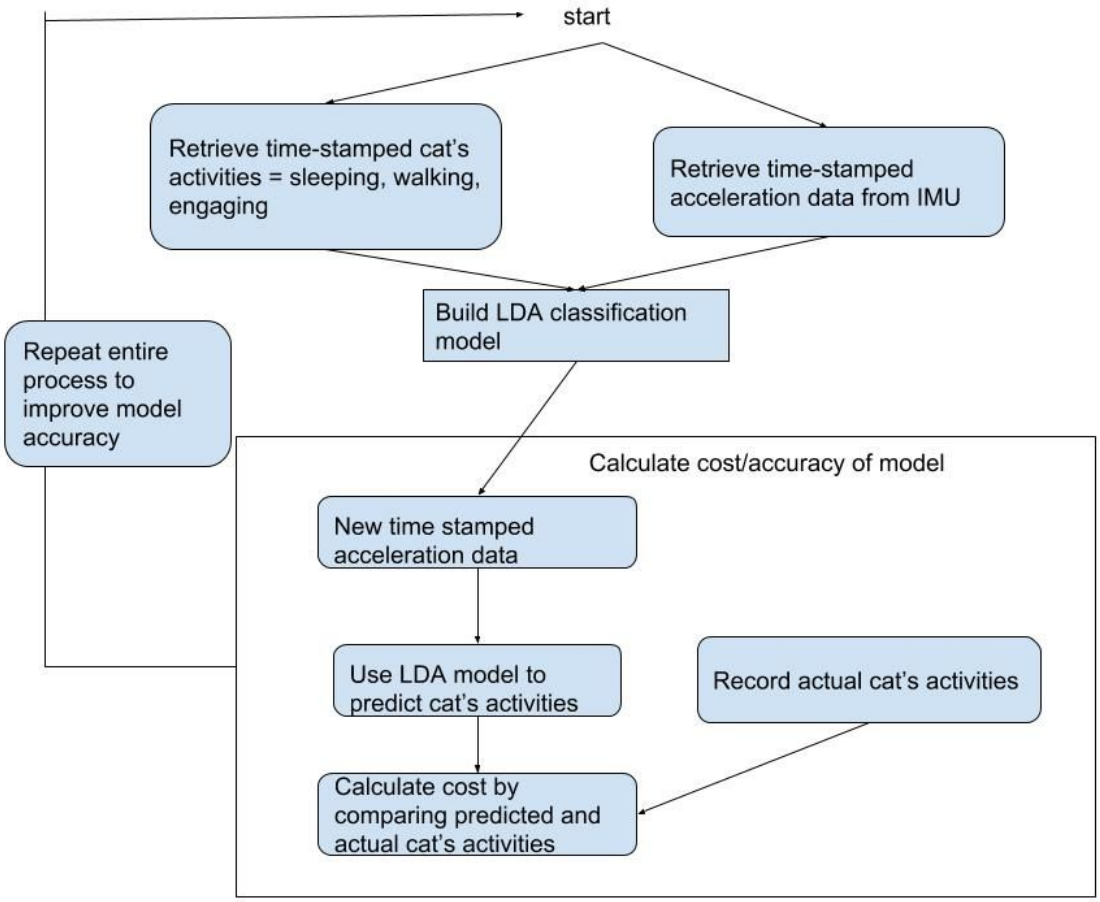

Fig. 7. Building LDA classification model

Algorithm For Detecting Cat's Engagement:
Linear Discriminant Analysis will be used to build a model for classification of cat's activities: sleeping, walking, engaging versus cat's acceleration values. Scikit, a python library will be used for the algorithm model. This machine learning model will be trained with our actual cat experiments, where accuracy will increase with more and more training.

2.3 Tolerance Analysis

The most critical requirement in our project is the ability to characterize the motion of the pet using the accelerometer. Therefore, the tolerance of the accelerometer will govern the final accuracy of our results. The accelerometer has a linear acceleration sensitivity, which depends on the scale of operation. Another sensitivity that is relevant to the accelerometer is the zero-g level offset(TyOff), which describes the deviation of an actual output signal from the ideal output signal if no acceleration is present.

The accelerometer in a steady state on a horizontal surface will measure 0 g on both the X-axis and Y-axis, whereas the Z-axis will measure 1 g. Offset can slightly change after mounting the sensor onto a printed circuit board or exposing it to extensive mechanical stress. Therefore, we plan to recalibrate for the sensor bias after mounting on the PCB as well.

Sensitivity using ±2g scale of operation. *In the table below, every time the raw values increments by one, the final calculated value (which is MilliG) increments by 0.061*

Binary	*LSB for +- 2G*	*Calculated miG (G=9.8m/s^2)*
10000	*0.061*	*0.976*
10001	*0.061*	*1.037*
10010	*0.061*	*1.098*

The smallest interval increment is 0.061 miG for an IMU at +-2g sensitivity, as shown in the above table. For last significant bit (LSB) in the binary, the measured miG increases by 0.061 at the minimum.

The following two figures show the different response of the accelerometer data in the vertical(z-axis) for walking and running respectively. The average difference between the amplitudes of the peaks between walking and running is approximately 0.71g. Therefore, a linear sensitivity of 1.037 milli-g is still smaller than the threshold for differentiating the two activities.[9]

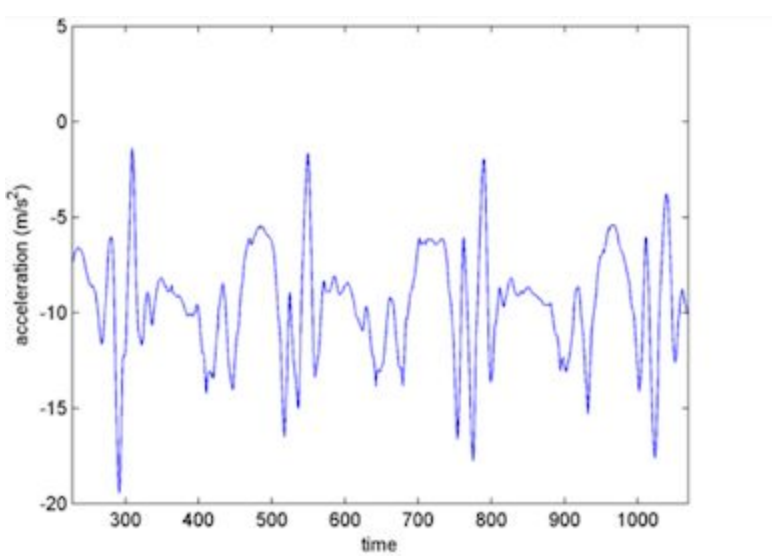

Figure: Accelerometer(z-axis) data from walking motion

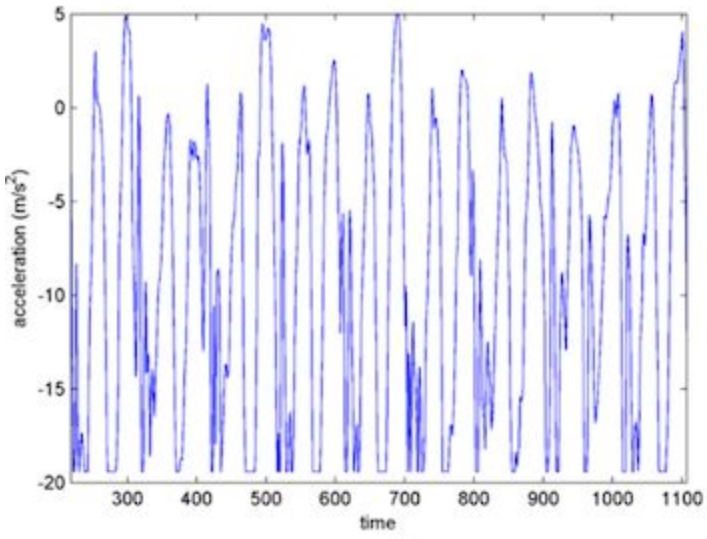

Figure: Accelerometer Data for Running motion

3. Cost and Schedule

3.1 Cost Analysis

3.1.1 Labor

According to Engineering Report 2016-2017, reported by the Engineering Career Services(ECS), the average salary for an Engineering graduate was $75,450. So, on a regular work schedule of 40 hours per week and 52 weeks a year, it sums to a total of 2080 hours. The hourly labor rate is then approximately $36 for each of the three members of our Senior Design team. We are expecting to work around 10 hours per week for 10 weeks, so the total cost is as follows:

$36 x 10 hrs/week x 10 weeks x 3 members = $10800

3.1.2 Parts

Description	Manufacturer	Part #	Quantity	Cost
ESP32 Dev board	Expressif	DevKitC V4	1	$10.64

IMU	Sparkfun	LSM9DS1	1	$15.95
Lipo Charger	Sparkfun	PRT-10217	1	$8.95
Lithium Ion Battery	Sparkfun	PRT-1385	1	$4.95
USB Charger	Dericam	B07D3QSXQJ	1	$7.99
Lipo Charging IC	Mouser Electronics	MCP73831	1	$0.57
Voltage Regulator IC	Texas Instruments	TLV755P	1	$0.62
ESP32 Module Chip	Expressif	Wroom 32	1	$3.80
Total				$53.47

3.1.3 Grand Total

$10,800 + $53.47 = $ 10,853.47.

3.2 Schedule

Week	Task	Person
02/25/19	1) Experiment with IMU data, measure sensor bias to ensure IMU is working correctly 2) Set up raspberry pi	1) Junnun, Jeff 2) Taha
03/04/19	1) Order battery circuit, voltage regulator, and pi camera 2) Go through the tutorial for setting up ESP32 and interfacing with it.	1) Taha 2) Jeff, Junnun
03/11/19	1) Store IMU data into the SD card of ESP32 2) Implement communication (I2C protocol) between IMU and ESP32 dev board 3) Test/Verify battery circuit, voltage regulator and charging circuit 4) Setup pi camera with raspberry pi 5) Create Eagle schematic for power circuit	1) Jeff 2) Junnun 3) Taha, Junnun 4) Taha 5) Taha

03/18/19	1) Interpret and Process IMU data from ESP32 on the Raspberry Pi 2) Prototype final circuit on breadboard 3) Create Eagle schematic for rest of the circuit and Order PCB	1) Jeff, Junnun 2) All 3) Taha
03/25/19	1) Establish communication between Mousr and raspberry pi	1) Junnun, Jeff
04/01/19	1) Develop algorithms for physical activity detection from accelerometer data. 2) Write code for data analytics of cat's engagement by comparing with data from Mousr unit.	1) Junnun 2) Jeff
04/08/19	1) Mount ESP32 and solder PCB, perform unit testing and final debugging if applicable.	1) All
04/15/19	1) Perform mock demo 2) Work on final report	1) All 2) All
04/22/19	1) Final demonstration 2) Work on final report	1) All 2) All

4. Ethics and Safety

One of the safety concerns in our project is battery's temperature. Misuse of the batteries can cause to overheating and burning the circuit. For most cells, charging significantly above 100% state of charge (SOC) can lead to rapid, exothermic degradation of the electrodes. Charging above the manufacturer's high voltage specification is referred to as overcharge [2]. A few indications of these hazards can be noticed.If a typical fully charged (or overcharged) Li-ion cell undergoes a thermal runaway reaction, a number of things occur, including [2]:

- Cell internal temperature increases;
- Cell internal pressure increases;
- Cell undergoes venting;
- Cell vent gases may ignite;
- Cell contents may be ejected; and
- Cell thermal runaway may propagate to adjacent cells.

In order to prevent this, we plan to use a thermistor to avoid overheating and make sure to use full batteries during demos. It is also to know the root causes of hazards and prevent them. Generally, the root causes of energetic cell and battery failures can be classified as [2]:

- Thermal abuse (e.g., external heating);
- Mechanical abuse (e.g., denting, dropping);
- Electrical abuse (e.g., overcharge, external short circuit, over discharge);
- Poor cell electrochemical design (e.g., imbalance between positive and negative electrodes); and
- Internal cell faults associated with cell manufacturing defects (e.g., foreign metallic particles, poor electrode alignment).

Another issue is using cats for experiment. Our project itself does not pose great danger on the cat, since it is only a cat collar. However, we will take extra measures to prevent battery related hazards as mentioned above or any physical discomfort for the cat when wearing the collar.

5. Citations

References:

[1] Ieee.org, "IEEE Code of Ethics", 2016 [Online]. Available: www.ieee.org/about/corporate/governance/p7-8.html

[2] SFPE, “Lithium-Ion Battery Hazards”, 2012. [Online]. Available: www.sfpe.org/page/2012_Q4_2/LithiumIon-Battery-Hazards.htm

[3] SparkFun Electronics, “SparkFun LiPo Charger Basic - Micro-USB.” 2015. [Online]. Available: www.sparkfun.com/products/10217?_ga=2.229262751.2001694764.1550419447-1820395587.1543635422.

[4] Sparkfun Electronics, “LSM9DS1 Datasheet”, 2015. [Online]. Available: https://cdn.sparkfun.com/assets/learn_tutorials/3/7/3/LSM9DS1_Datasheet.pdf

[5] UIUC College of Engineering, “Engineering Report 2016-17”, 2017. [Online]. Available: https://ecs.engineering.illinois.edu/files/2018/03/Engineering_Report_2016-2017_FINAL.pdf

[6] Texas Instruments, “TLV755P Datasheet”, 2018 [Online]. Available: http://www.ti.com/lit/ds/symlink/tlv755p.pdf?HQS=TI-null-null-mousermode-df-pf-null-wwe&DCM=yes&ref_url=https%3A%2F%2Fwww.mouser.com%2F
card

[7] GitHub. (2019). *espressif/esp-idf*, 2019 [Online] Available at: https://github.com/espressif/esp-idf/tree/master/examples/storage/sd_card

https://randomnerdtutorials.com/esp32-data-logging-temperature-to-microsd-card/

[8] Espressif, “ESP32 WROOM Datasheet”, 2018 [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf

[9] Resna.org, “Simple Activity Recognition”, 2015 [Online]. Available: https://www.resna.org/sites/default/files/conference/2015/other/gani.html