INDIVIDUAL LAB REPORT 4
Progress Review
MRSD Project

Name: Keerthana P G
Andrew ID: kgopalak

Team D

10th November 2017

Teamwork

Ritwik: Integrated parts of the neural network and began training
Luka: Made changes to text and speech processing, printed CAD models for hardware assembly
Keerthana: Wrote bidirectional LSTM for text encoding, designed the PCB board and programmed the camera tracker
Luxing: Proposed the initial design of PCB

Individual Responsibility

I was responsible for programming the bidirectional LSTM for text encoding. I also designed the PCB layout from scratch, and implemented face detection and tracking on web camera.

Bidirectional-LSTM for Text Encoding

For text processing, we had decided to use a bidirectional LSTM as detailed in [2]
Given word embedding of T input words $\{w_1, \cdots, w_T\}$, the vectors are fed into the bi-LSTM.

$h(f)_t = \text{LSTM(f)}(x_t, h(f)_{t-1})$
$h(b)_t = \text{LSTM(b)}(x_t, h(b)_{t+1})$,
where $h(f)_t$ and $h(b)_t$ represent the hidden states at time t from the forward and backward LSTMs, respectively.

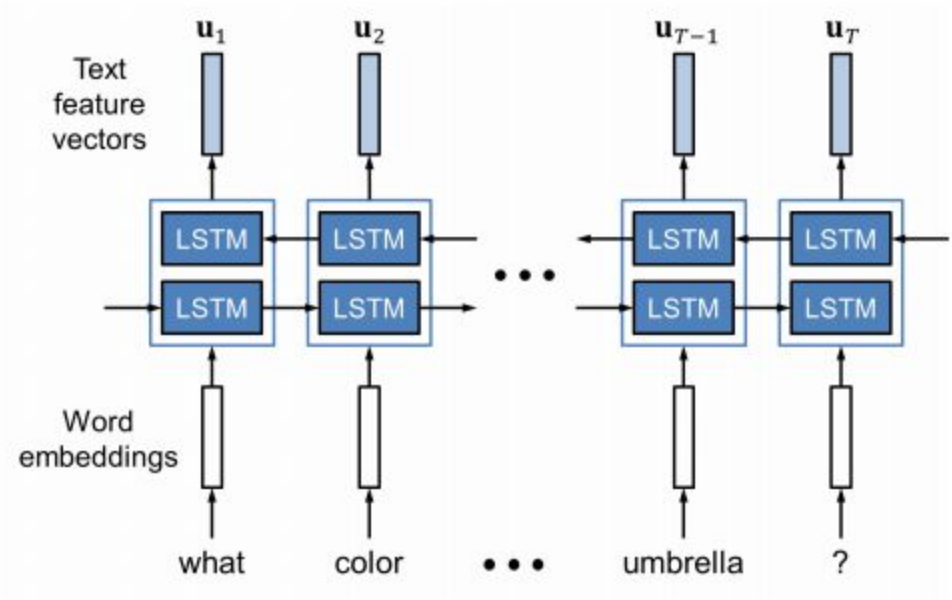

Fig1: Bidirectional LSTM for text encoding. Source:[1]

Then we add the two hidden states at each time step
$u_t = h(f)_t + h(b)_t$

To construct a set of feature vectors $\{u_1, \cdots, u_T\}$ where u_t encodes the semantics of the t-th word in the context of the entire sentence. These context vectors are then passed to the dual attention network.

Stage of Implementation

Currently, we've written the layers and the training function, it runs smoothly on random input. However, we have not integrated this or the attention network in the entire hybrid model as prediction involving text modality is not an FVE requirement.

PCB Design

It was Luxing's responsibility to design the PCB layout for our team, however he was not present that day and the submission was due the next day at 7 am. So, I took the initiative to get it done along with Ritwik.

After looking at the design Luxing submitted for Milestone 2, we figured that he had not considered end requirements and overvoltage/reverse voltage protection. So, we decided to start from scratch. We began by estimating the mass of the camera(upperbound 300g) and the moment of inertia of the aluminium extrusions about the central axis.

Fig 2: Alumium extrusions[2]

Thecross section has dimensions of 25*25 mm, and aluminium has a density of 2700kg/m^3. The moment of inertia about central axis, coming out of cross section was approximated to be 22147.75*10mm^4. After adding the weight of camera and offsets for upper estimate of off-axis moments, we concluded that to move the equipment at an angular velocity of at least 0.1 rad/s, we required a torque of at least 0.1Nm(since torque = moment of inertia about axis * angular velocity). Hence we decided to go ahead with the mercury SM-42BYG011-25 that has a torque rating of 0.23Nm. This motor goes well with the A4988 motor driver chip with a current rating of 1A. These considerations and parameters became the design fundamentals of the PCB. We used a zener diode for overvoltage protection, a schottky diode for reverse voltage protection and a fuse wire of 1.5A for overcurrent protection.

Once the PCB schematic was designed, we generated the board, arranged the labels, calculated an ideal trace width for our current requirements(32mil) and routed the circuit. We paid heed to the feedback we received for Milestone 2 and tried our best to address them.

Further, we generated the manufacturing files and the free-FDM links as prescribed in the submission guidelines.

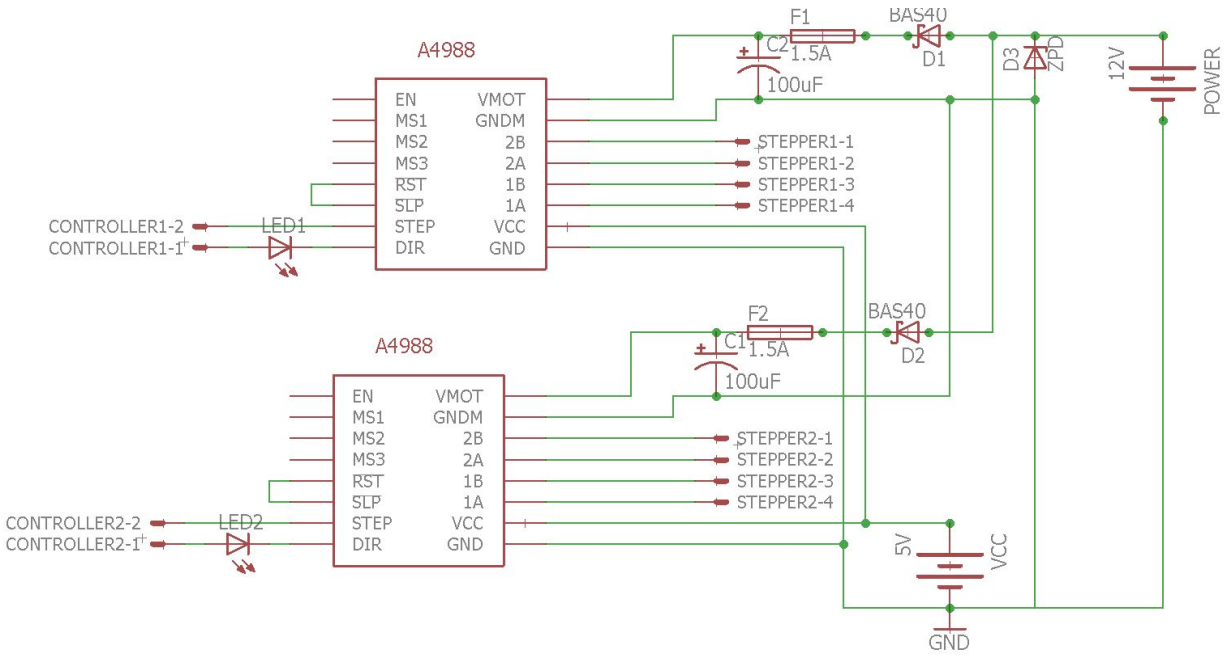

Fig 2: PCB Schematic

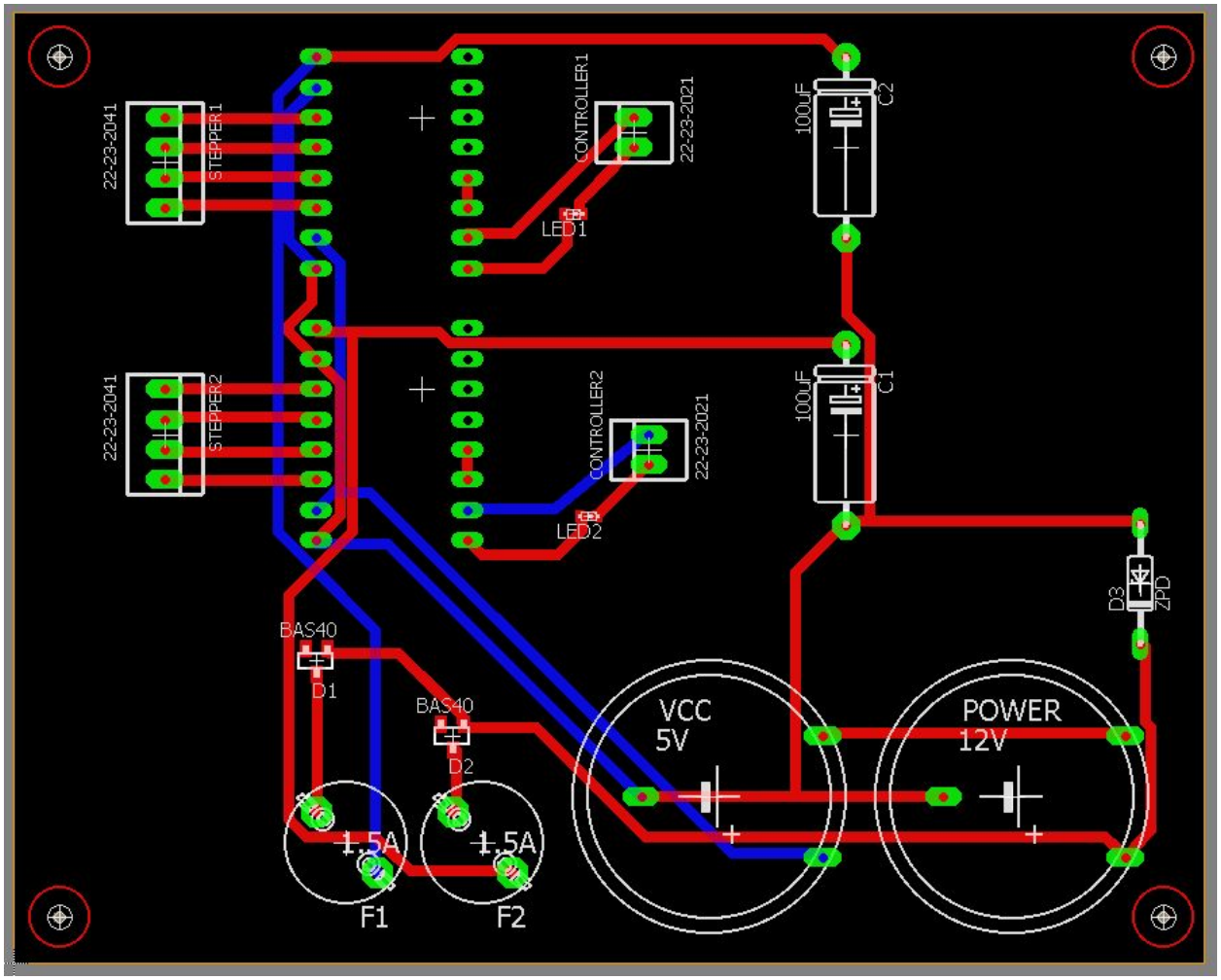

Fig 3: PCB Board Layout

Stage of Implementation
Complete

Face Detection and Tracking

For FVE, one of our requirements is a 1-DOF face tracking mechanism. To get this ready, we wrote a simple face detection function using dlib and openCV, and a newly bought webcam. We used the dlib face detectors(Histogram of Oriented Gradients algorithm[3]) as it was shown to perform better than Haar Cascades of OpenCV(Viola-Jones algorithm)[4], the latter yielding several false positives.

After detecting faces, we calculate the centroid of the face and output this relative to the position of the centroid of the frame. This relative position will be fed as serial input to the Arduino. The objective of the tracking algorithm is to move the camera in a direction along the vector joining frame centroid and face centroid.

Stage of Implementation
This has to be integrated into the hardware after programming the stepper motor and Arduino.

Additional Responsibility

In addition, I inspected the training dataset to view consistency of target ratings and raters across target dimensions and relayed the findings to Luka. We found that a large part of the dataset doesn't have intensity ratings. So we decided to drop that dimension instead of cutting down on training data. Secondly, we found that number of raters varied across the dataset, and due to lack of training data, we decided to include any data sample that has at least one rater for each output dimension.

Challenges

1. The implementation of face detector works well in good lighting, but performance falls when lighting is dimmer. Also, when the subject is farther>2.5m, face is not detected.
2. Team management has hit new lows. One of our team members hasn't showed up for more than a week and half and has been hostile too. We are still figuring out on how to deal with the situation.

Future Plans

1. We hope to have completed the robot from a hardware assembly standpoint.
2. We also hope to have gotten the stepper to rotate the camera based on where the human is in the scene.
3. We hope to have done a large-scale training operation and began fine-tuning the hyper-parameters in order to improve our training accuracy.

References

1. Nam H, Ha JW, Kim j, "Dual Attention Networks for Multimodal Reasoning and Matching", arXiv:1611.00471, https://arxiv.org/abs/1611.00471
2. https://www.ebay.com/itm/80-20-Inc-10-Series-1-x-1-Aluminum-Extrusion-Part-1010-x-60-Long-N-/220776275460
3. https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78
4. https://news.ycombinator.com/item?id=14435569

Codes

facedet.py

```
 #our camera currently is 1280*720 p

import cv2
import dlib
import serial
detector = dlib.get_frontal_face_detector()
def show_webcam(mirror=False):
    cam = cv2.VideoCapture(1)
    while True:
        ret_val, img = cam.read()
        if mirror:
            img = cv2.flip(img, 1)
            try:
                dets = detector(img)
                #print("Number of faces detected: {}".format(len(dets)))

                for i, d in enumerate(dets):
                    #print("Detection {}: Left: {} Top: {} Right: {} Bottom: {}".format(
                      # i, d.left(), d.top(), d.right(), d.bottom()))
                    cv2.rectangle(img, (d.left(), d.top()), ( d.right(), d.bottom()), (255,0,0), 2)
                    centroid=((d.left()+d.right())/2, (d.top()+ d.bottom())/2)
                    val=centroid[0]-320
                    print((centroid[0]-320))
                    #the frame seems to have 620 horizontal pixels
                    #the val tells you how many pixels from the centre he centroid of face is ocated.
                    #-ve means image centroid is at the right f face centroid
                    #+ve means viceversa
                    #magnitude of val indicates how shifted it is

                cv2.imshow('my webcam', img)
                if cv2.waitKey(1) == 27:
                    break  # esc to quit
            except:
                continue
    cv2.destroyAllWindows()

def main():
    show_webcam(mirror=True)

if __name__ == '__main__':
```

main()