A HUMAN FOLLOWING ROBOT FOR FALL DETECTION

by

Simeon O. Adebola

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Engineering Technology

Middle Tennessee State University

December 2019

Thesis Committee:

Dr. Lei Miao (Advisor)

Dr. Saleh Sbenaty

Dr. Vahid Khiabani

Dr. Misa Faezipour

This thesis is dedicated to the God and Father of my Lord Jesus Christ for His unfailing mercies that I have received throughout the course of my degree and continue to receive. Secondly, to my parents: Late Rt. Revd Dr. Simeon Oluwemimo Monday and Late Mrs. Abigail Ajibola Adebola whose impact and sacrifices made this possible.

ACKNOWLEDGMENTS

I would like to say a big thank you to my advisor Dr. Lei Miao for his guidance, counsel, patience, and support throughout the course of this thesis. It has been a privilege to learn from him and gain a better grasp of research fundamentals. I will remain eternally grateful.

I am grateful to all members of my committee: Professor Saleh Sbenaty, who is also the Graduate Coordinator for the Engineering Technology Program; Dr. Vahid Khiabani; and Dr. Misa Faezipour. Thank you for your guidance and counsel in the course of my thesis work.

I am grateful also to Dr. Charles Perry and Dr. Mala Beard, whose courses I also took at MTSU and who inspired me in more ways than one.

I am grateful to my parents Late Rt. Revd Dr. Simeon Oluwemimo Monday and Late Mrs. Abigail Ajibola Adebola. Thank you for sowing the seeds of hunger for an education. Thank you for every sacrifice you made on my behalf and for every word of instruction and prayer. I miss you but know you are in a better place.

I am grateful to my siblings: Dr. Dunsin Adebola and Mrs Feyisayo Adebola; Mr Rolu Adebola and Mrs Aramide Adebola; Dr. Rotimi Awoyode and Dr. Mrs Oluwalope Awoyode; Dr Sunmolu Adebola and Mrs Funke Adebola; Mr Dare Oluboba and Mrs Foluso Oluboba; Mr Lanre Adebayo; Ms Bola Akinsete; and Mr Ayo Akinsete and Mrs Caroline

Adelakun for their support in every kind- financial, material and spiritual. Thank you for looking after your aburo!

I am grateful to Wind and Light for every long call, for every counsel, for every prayer, for hours, and days of strategizing and asking questions. Thank you for always being willing to listen and advise and guide and support. I would not be here today without your support, and I am immensely grateful.

Thanks to Late Venerable V.S. Adenugba and Mrs Adenugba for believing in me right from the start and ultimately setting me on this path.

Thanks to Venerable Samson Sodipe, Mr. Bisi Ajayi (Daddy Ibadan), Venerable and Mrs. Ejiwunmi, Mr. and Mrs. Ero-Phillips; Uncle Kola Ejiwunmi, Venerable Adelowo Adesina and Dr. Destiny Ogwah for their financial support, care and prayers for me.

Thanks to Rt Revd Pelu Johnson for his immense financial support as I made my way to the US for my Master's and for his constant care. I remain grateful, Sir.

Thanks to Pastor Olubi Johnson and Pastor Mrs. Sarah Johnson for accepting me as a son and for their support and care for me before and during my thesis period. Thank you to the entire SPCC Ibadan family for prayer and support during this period.

Thanks to Manoj Basnet, Tanzil Chowdhury and Bola Oladele for helping with initially testing FADER.

Thanks to Ben Becker, Bryan Schuder, and all the staff of the MTSU Library Makerspace for their technical support, brainstorming and troubleshooting sessions, and for always being willing to listen.

Thanks to Wenbo Dong for help and guidance with ideating through the changes we made to FADER while working on 2-D navigation.

Thanks to Mary Ellen Sloan for review sessions of my initial thesis drafts.

A big thank you to Dr. Adrian Rosebrock of PyImageSearch for his invaluable tutorials on Computer Vision and Deep Learning. Thank you for making your knowledge available.

Thanks to Mr Toheeb, Ms Ledoux, Tomilayo Bello, Mayowa, Esan and Mr Shamshudeen for volunteering as participants to test FADER.

ABSTRACT

This thesis presents work on a human following robot for detecting falls in the home of the elderly. The goal is to have a robot that can detect a human, follow the human in a cluttered space, and determine when the human falls. A Raspberry Pi based Robot known as Fall Detection Robot (FADER) that had been developed in the Real-time and Embedded Control, Computing, and Communication (REC^3) Lab at Middle Tennessee State University is used, and a number of adjustments are made to its design including adding a Pi Camera and an Arduino microcontroller board. Computer vision deep learning-based object detection is used as the means of detecting the human, and linear regression and threshold-based algorithms are used to estimate the distance to the human, navigate and to determine falls. The advantages of using FADER for fall detection include its being mobile, the user not being required to be involved for the technology to work, and its being non-invasive with respect to the user's body. Furthermore, FADER is low-cost and easily manufacturable. Results show that the modified FADER functions with a high precision of 100% but low sensitivity of 42%.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS/ABBREVIATIONS

ADL: Activities of Daily Living

AHRS: Attitude and Heading Reference System

ANN: Artificial Neural Networks

CDC: Center for Disease Control

CNN: Convolutional Neural Network

GPS: Global Positioning System

I/O: Input/Output

kNN: K-Nearest Neighbor

LCS: Longest Common Subsequence

MMS: Multimedia Messaging Service

NN: Neural Network

PRM: Pattern Recognition Methods

PWM: Pulse Width Modulation

RFID: Radio-frequency Identification

SMS: Short Messaging Service

SVM: Support Vector Machine

SVM: Signal Vector Magnitude

TBA: Threshold Based Algorithm

TBD: Threshold Based Detection

WBASN: Wireless Body Area Sensor Network

YOLO: You Only Look Once

CHAPTER I: INTRODUCTION

Human life expectancy has been on the increase around the world[1], [2]. Japan, for example, in 2013, achieved a life expectancy of "*80.1 years for men and 86.4 years for women*"[3], which was the "*. . .world's longest. . .*"[3]. Furthermore, it has been shown that from 1950 to 2010 the survival rates for people aged from 60 -80 years was accelerating [4]. In the US, it is stated that the number of older adults will grow "*from 35 million in 2000 to an estimated 74 million in 2030*"[5]. Whilst a lot of the aged or aging population live with family members or in retirement/nursing homes, there are some who also live alone. In the US, "*12 million seniors live alone*"[6] Furthermore, falls are known to occur. In fact, it has been shown that "*Yearly incidence of falls among the elderly population aged over 65 years is 30-50% globally*" [7]. In the US, there are so many falls every year that result in a total cost of 34 Billion Dollars[6]. When an elderly person lives with family or in a nursing home, falls can be quickly detected, and they can get help. However, in cases where they live alone, help may not be forthcoming for a long time. Yet, medicine says that the best time for an injured person to get help is within the first hour after the injury occurs: also known as the golden hour. Therefore, there is a need to constantly monitor the state and being of elderly members of the population in a way that also does not contravene their privacy. Our approach is to develop a human-following robot that can track and follow the elderly in their homes, can detect if they fall and can then signal for help.

There are a number of options that have been developed or are being developed to monitor for falls in the home of the elderly. Such options include fall detection using wireless signals in the home, and fall detection using sensors on the body of the elderly person. Sensors on the bodies may be invasive and require the participation of the elderly. On the other hand, using wireless signals in the homes of the elderly have been shown to have low accuracy and are non-social. We have chosen to use a following robot because it is non-invasive with respect to the body of the elderly person. Furthermore, the robot can be non-participatory for detection and tracking. By 'non-participatory,' we mean that it does not require any action on the part of the person in order for it to carry out its monitoring. Moreover, the robot is portable. In this approach, computer vision, specifically object detection is used to detect and then follow the human being. First, work was done on the control and navigation of the current mobile prototype in a one-dimensional(1-D) space using object detection. This was done with deep learning running on the Raspberry Pi and using the Raspberry Pi camera. This allows the robot to detect objects and then move towards or away from them. Furthermore, in moving away from a target(person), the robot might back up close to another object, e.g., wall, thus preventing proper avoidance. The solution to this is to use an ultrasonic Ping)))™ sensor to avoid the robot hitting the object.

The general scope of this thesis is to develop a robot to detect, track and follow a human in a cluttered home environment. Our goals are met when the robot can detect humans, navigate and follow a human, avoid obstacles/objects in the space, detect a human fall, and can communicate when the fall is detected.

The rest of the thesis is divided into four chapters. A review of some previous work in both robotics and fall detection is done in Chapter 2. Chapter 3 covers the methodology. The Rapsberry Pi and Arduino based Fall Detection Robot (FADER) uses computer-vision based deep learning object detection to detect a human in a space and then follow the person. Distance estimation, one-dimensional and two-dimensional navigation and fall detection are all covered in this chapter. Chapter 4 give the results from both our preliminary and extended laboratory tests, evaluates them using performance measures and discusses the implications of our results. Chapter 5 concludes the thesis with future work.

CHAPTER II: LITERATURE REVIEW

This thesis is about a mobile robot. The Merriam-Webster dictionary defines ‘Robot’ as “*a machine that resembles a living creature in being capable of moving independently (as by walking or rolling on wheels) and performing complex actions (such as grasping and moving objects)*”[8]. The word ‘robot’ was first used by the playwright Karl Capek in his play “R.U.R.,” (translated as Rossum’s Universal Robots), which premiered in 1921[9], [10]. Another recent definition that emphasizes embodiment[10] states that a robot “is a physically embodied artificially intelligent agent that can take actions that have effects on the physical world”[11]. While there are so many discussions today around what is and what is not a robot, these definitions are useful for our considerations.

We see this work as occupying the intersection of a number of fields, including social robots, human following robots, and robots in the household. We shall discuss these fields and review some works in them next.

A. Social Robots

The field of social robots deal with robots that are in the household and interact with humans[12]. This field is increasingly under consideration, especially with respect to two aspects of our population: children and the elderly. Social robots are being viewed for opportunities around education and learning for children[13], and also for care, community, and companionship for the elderly[14], [15].

B. Human Following Robots

Human following robots are currently being used in a number of areas in our world today as well as being proposed for use in other sectors too. These sectors include service areas, household, travel, and shopping to name a few. The key issues to consider in a human following robot include detecting the target person, preventing permanent loss of the target person, determining the distance from detected person, and navigating with respect to the target person. Research in human following robots often works on one or more of these issues. We would review some of these research works next.

In 2015, [16] in their work showed that they had developed a human following robot that used a laser range finder to detect the shins of the person to be followed. The laser range finder also allowed the robot to avoid obstacles. The robot consisted of Kobuki Yujin robot, the Hokuyo UST-20LX laser range scanner, a Dell PC and web-camera. Tests on the robot in two environments resulted in an efficiency of 83% and 66% respectively. The second environment was more complex than the first environment.

In 2017, [17] presented work in which they had a robotic cart that could be controlled using a smartphone or could automatically follow a person using ultrasonic sensors. The cart consisted of an Arduino Mega 2560, six HC-SR04 ultrasonic sensors, four wheels, with two of them controlled by 12V DC motors and the two others on springs, a Bluetooth module and a 2kg 12 V, 6.4 Ah battery. The maximum load the cart could carry was 80 kg but at that load it travelled at a very slow speed so it was recommended to only be used with a load of 40 kg to allow a speed of 1m/s. The control application was coded to be run on the smartphone using the Android Studio and used the Bluetooth module to communicate with the microcontroller and to select which mode the robot operates in.

Also in 2017, [18] presented their work on integrating a stereo vision based convolutional neural network (CNN) tracker with a person-following robot. The CNN tracker allows the robot to consistently be able to follow the target person even when the person is occluded or steps out of sight. The CNN was trained using RGB and stereo depth images and the training is done online and in real-time. The robot used was a Pioneer 3AT robot and two stereo cameras were used namely: Point Grey Bumblebee and ZED stereo camera. The robot follows the target using a PID based controller while the CNN trains based on images received. If the person is lost, the robot is then able to reproduce the expected path of the user and follow that to check and retrieve the person.

Additionally in 2017, [19] presented their work on a new algorithm known as Selected Online Ada-Boosting(SOAB) that builds on the Online Ada-Boosting (OAB) algorithm and with images from a stereo camera which allows a mobile robot to be more resilient in person-following. According to the paper, using SOAB, a mobile robot can handle situations including the target picking or wearing a bag, sitting, squatting, illumination, the target facing the side, partial and complete occlusion, the target standing beside someone with the same clothes or having such a person pass in front of them, and the target changing their appearance. The robot used in the paper is Pioneer 3AT robot which was described as "*a four wheeled differential drive robot with an on-board computer*". The paper further stated that the robot is "*configured with a Point Grey Bumblebee Stereo Camera which acts as the only sensor on the robot to sense its environment*".

C. Robots in the Household

Robots in the household carry out a number of roles and would likely take on more diverse roles in the future[12]. For robots who are required to navigate in the household, one of the most important things is a way to plan a path. We would review some recent works in this area next.

In 2017, [20] presented work in which they had implemented an omnidirectional mobile system that could be added to a home robot and which together with gesture recognition brought added function that allowed the robot to follow the user, approach the user and avoid obstacles in its path. A Kinect sensor was used. Notably, the paper stated that "*the omnidirectional wheel used by the power subsystem can help the robot to do translation and spin movement that is difficult for the traditional two-wheeled robot. Among them, the translation movement allow the robot to avoid obstacle in the path faster and more flexible, while the spin movement allow the robot to lock user with large angle rotation when user is turning around in a short time, so that the target can always be located within the sight of Kinect camera*."

In 2018, [21] presented their work developing an improved navigation model for a human-following robot indoors. Even though there was no physical testing or experimentation, the work uses Microsoft Robotics Developer Studio 4 (MRDS) for implementation and tests using Virtual Simulation Environment scenarios. The model uses a "*depth camera, a limited array of proximity sensors and an active IR marker tracking system*". A fusion algorithm is used with these sensors.

Other works related to indoor navigation for robots include [22]–[24].

FALL DETECTION

A fall can be defined as "*an unexpected event in which the participant comes to rest on the ground, floor, or lower level"*[25]. One can classify falls into two groups: fatal and non-fatal. Factors that increase the risk of a fall include age[25]. Thus, the older a person is, the more they are at the risk of a fall[26]. In fact, the "*Yearly incidence of falls among the elderly population aged over 65 years is 30-50% globally"*[7]. On the other hand, human life expectancy is increasing globally. From 1950 to 2010, the survival rates for people aged from 60 -80 years was accelerating and the percentage of people aged 65 and older is expected to reach 21% by 2050 [4], [27]. Therefore, it is expected that an increase in life expectancy would be accompanied by an increased risk of falls.

In the US, statistics echo the global trend. According to the CDC, in 2017, there were 36,338 deaths due to unintentional falls[28]. Of these, 31,190 (approximately 86%) were aged 65 and above[28]. Furthermore, given that the total population was 50,858,679, this was equal to a crude death rate of 61.33 for every 100,000 people[28]. Additionally, there were 8,591,683 injuries due to unintentional/undetermined falls[29]. Of these, 2,970,720 (approximately 35%) were aged 65 and above[29]. Furthermore, given that the total population was 50,858,679, this was equal to a crude fall rate of 5,841.13 for every 100,000 people[29]. Projections are that the number of falls will continue to increase with estimations that by 2030 we would have over 61,000 and maybe as high as 100,000 fatal falls yearly[30]–[32] Costs due to all falls have risen from 34 Billion Dollars in 2013 to over 50 Billion Dollars in 2015 and are projected to rise as high as 67.7 Billion Dollars by 2020 [6], [33]–[35].

Another aspect is the non-fatal consequences of falls. Falls can cause broken bones and head injuries; sometimes, they are indicative of or caused by severe illnesses such as stroke. Sadly, falls often signal the “beginning of the end” of an older person’s life. If the elderly person remains on the ground for an hour or more after falling, the victims may suffer from many medical complications such as dehydration, pressure sores, pneumonia, hypothermia, internal bleeding, and permanent damage to the brain, and half of the people die within 6 months[36]–[38]. A psychological effect also referred to as ‘post-fall syndrome’ with indicators including fear of falling, “*…loss of confidence, loss of muscle and control, problems with balance, and walking disorders …*”; depression, reduced quality of life, and increased medical and familial costs are also possible non-fatal consequences of falls [39]–[41].

When elderly people live in a nursing home or with family, a fall can quickly be detected, and they can get help soon. However, not all elderly people live with other people. In the US at least 12 Million elderly people live alone[6]. Unfortunately, it has also been shown that when such elderly people fall, they may not get help as soon as possible[6]. This is a challenge as “ *An immediate response after a collapse has been shown to be key to soften the most serious consequences of falls”*[35] and *“…there is a close relationship between the delay in assisting to the injury and the mortality rate*”[27]. There has therefore been a lot of research into technology for fall detection in order to ensure that the elderly person gets help as soon as possible after the fall.

FALL DETECTION SYSTEMS

According to [42], “*A typical fall detection system has two major functional components: the detection component and the communication component… the detection component detects falls and the communication component communicates with emergency contact after fall detection*.” [43] showed a general schematic for how fall detection systems work dividing the system into three parts: “*sense*(sensing)*, analysis and communication*”. This schematic is shown in Figure 1.

A primary concern in fall detection systems is usually the way to differentiate falls from other normal occurrences of the human body referred to often as Activities of Daily Living (ADL). In considering fall detection methods/ technologies, we see that virtually all of these technologies use sensors in one form or the other. In addition to this, we observe

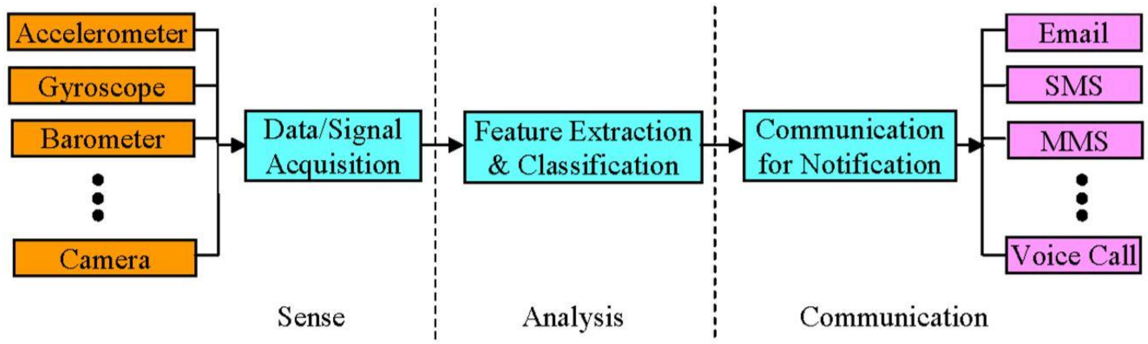

Fig. 1. General schematic of a fall detection system [43].

the following characteristics: participation, invasiveness and mobility. Participation refers to whether the technology requires the elderly person to be involved in the operation of the technology. Examples of participatory technologies are wearable sensors that the elderly person must remember to have on their person every time they are moving. Invasiveness refers to whether the technology infringes on the body or privacy of the elderly person. With respect to privacy, the typical example given for such an invasive technology are cameras. Mobility refers to how easy to transfer the technology to another environment and have it function. We discuss each if these characteristics in the sections below.

In considering fall detection systems, two of the most important metrics used are sensitivity (also referred to as recall) and specificity[6], [27]. According to [27], "*Sensitivity represents the capacity to detect a fall and it is calculated as the ratio of true positives and the sum of true positives and false negatives. Specificity identifies the ability to avoid* *detection of a normal event as a fall and it is calculated as the ratio of true negatives and the sum of true negatives and false positives."*

We also see in quite a number of fall detection systems the application of machine learning techniques both traditional and more recently Neural Networks and their many iterations.

[6]classifies fall detection methods into two general groups: wearable and non-wearable technologies. We would look at the various methods under each group and discuss them in the following sections.

A. Wearable Technologies for Fall Detection

These technologies require the user to wear the detection sensor(s) on their body. Examples of these technologies are smartphones, RFID, transportable mobility sensors (e.g. accelerometer, galvanometers and magnetometers).

A1. Sensors for Fall Detection

For years, fall detection was carried out by having the elderly person wear sensor(s) on their body. Such sensors include accelerometers, gyroscopes to mention a few. In a number of cases, different sensor types are combined. For example, accelerometers, gyroscopes and magnetometers are combined in an inertial measurement unit (IMU)[44]. One of the advantages of sensors for wearables is that they are becoming more ubiquitous, more powerful, cheaper while at the same time using less computational power. The main disadvantage of wearable sensors is that they require the participation of users every time i.e. the elderly person need to remember to always have the sensors on their body. However, there is no guarantee that this will be done. Also, recently, [6] had this to say about wearable sensors for fall detection:

"…years of medical research has shown that wearable devices do not work well for the elderly. Seniors are typically encumbered by wearable technologies, and many of them suffer from memory problems and hence may forget to wear or charge their devices. Furthermore, those sensors can be dangerous; recently an elderly woman got strangled with her fall detection pendant…".

We would review some of the work that has been done over the years in wearable sensors for fall detection.

In 2009, [45] presented work that utilized two nodes containing a tri-axial accelerometer and a tri-axial gyroscope each for fall detection. The project sought to detect falls in a more effective way that reduced false positives due to sitting down fast and false negatives from falling on stairs rising from the use of accelerometers alone. The system developed was quite effective dealing with general type of falls but still had some issues "*when people fall against walls ending with a sitting position"* (false negatives) or "*quickly lying down"* such as in bed (false positives). Results showed a sensitivity of 91% and a specificity of 92%.

In 2013, [46] presented work that combined a waist mounted triaxial accelerometer with an Artificial Neural Network(ANN). The system was used to both gather the initial data that was used to train the ANN, and then to carry out the fall detection. During fall detection, data from the accelerometer was "pre-treated" and then fed into the trained ANN to determine whether it was a fall or not. When a fall is detected, a computer receves an alarm. Results presented included a specificity of 98.6% and a sensitivity of 98.4%. As part of testing, a subject had the system for an extended period to ensure that false alarms were not an issue.

In work first presented in 2015 [36], we see the use of a triaxial accelerometer, triaxial gyroscope, and triaxial magnetometer in a combined unit referred to as an "*Attitude and Heading Reference System (AHRS)*". This allowed the location of the device in space, and together with algorithms, including the orientation filter, was used for fall detection. An update was presented in 2016[27], with the addition of a barometer and a complimentary filter to further improve the efficiency by allowing the detection of syncopes and backward falls that end with the person sitting.

In similar work proposed in 2017, [47] also used an AHRS to carry out fall detection. The AHRS was to be mounted on the waist of the user with the aim of detecting all kinds of falls. Other parts of the system included an *"ARM microcontroller, a bypass button, a LCD module, GPS, GSM/GPRS and a battery"*. Both adhoc data fusion and Kalman filter algorithms were to be used to remove noise and for detection. When a fall is detected, a notification containing the location of the person (using GPS) is sent to the notice is sent to a designated guardian's phone as well as to a hospital website.

Work presented in 2017 by [26] shows the gathering of a dataset using Shimmer sensors including triaxial accelerometers, triaxial gyrometer, magnetometers, pressure and temperature sensors placed on the body of subjects to form a wireless body area sensor network (WBASN). The dataset was classified based on age and weight and then three machine learning techniques namely Support Vector Machine (SVM), k -Nearest Neighbor (KNN) and Neural Network (NN) were tested. The results showed KNN to be the most accurate with a sensitivity of 94% and a specificity of 96.23%.

A2. Smartphones for Fall Detection

Smartphones are often considered for use in Fall Detection systems because they contain sensors such as accelerometers and galvanometers already built-in, together with the opportunity to have a software application that can utilize the readings from the sensors. The main advantage of smartphone systems therefore is having both the detection component and the communication component in a single compact unit. As stated by [44], smartphones "*feature embedded motion sensors, increasingly powerful microprocessors, considerable memory capacity, open source operating systems, and telecommunication services, making them ideal candidates for easily programmable and* *customizable fall detection*". We would review some of the recent smartphone-based detection systems.

[42] in 2010 reported the development of a fall detection system that would run on any mobile phone provided the phone has an accelerometer. According to the authors, this was the first proposal and implementation of the mobile phone as a platform for fall detection. The system which was named PerFallD was implemented on an Android G1 phone which contained an accelerometer, *98MB RAM, 70MB of internal storage and a 1150mAh rechargeable lithium ion battery*. PerFallD's algorithm included an alarm that sounds once a fall is detected and the user has a period to turn off the alarm barring which the system reached out to already saved emergency contacts. Results after showed the waist as the optimal position for mounting the phone running PerFallD with false negatives at 2.67% and false positives at 8.7%. Limitations include detection of slow falls.

[48] in their presented research carried out fall detection using a waist mounted smartphone with a built-in accelerometer. The system classified the movement of a person in real-time. The smartphone used ran the Android operating system, had a "*1GHz Snapdragon CPU*" and a "*built-in tri-accelerometer*". Acceleration due to body motion and gravity were used and with their developed algorithms, they were able to classify body motion "*into five different patterns: vertical activity, lying, sitting or static standing, horizontal activity and fall.*" Furthermore, when a fall is detected, an alarm is sent using Multimedia Messaging Service (MMS) which contains the "*time, map of suspected fall location and GPS coordinate*". The main limitation of the system developed was that the

smartphone has to be on waist to work. So, it would not work if it is elsewhere on the body including the person's hand.

[49] work utilize the accelerometer and GPS module of a waist-mounted smartphone to detect falls both indoors and outdoors with a reported accuracy of 94%. They are especially able to pinpoint the direction (left, right, backwards, forwards) of fall using the acceleration changes on the X, Y and Z axes. They were also able to differentiate between falling and jumping, sitting, standing, walking and running.

[43] surveyed existing smartphone-based solutions for fall detection and prevention. The paper discussed the trends in smartphone-based solutions under the three phases: sensing, analysis and communication. The survey showed that the most commonly used sensor in smartphone-based systems was the tri-axial accelerometer. In a group of reviewed solutions, the paper also notes that GPS receiver and the gyroscope were the next most commonly used sensors after the accelerometer. It also showed that most smartphone-based systems used a Threshold-Based Algorithm (TBA) because such an algorithm is *"less complex and hence require the lowest computational power…, which helps to reduce battery power consumption…"*. TBAs could use adaptive or predefined thresholds. The survey also pointed out that with the increasing computational power available in smartphones, it had become possible to run Machine Learning (ML) algorithms on them and gave examples of such projects. Having detected a fall, most systems request confirmation from the person being observed and/or send information to the established external contact. Requesting confirmation helps to prevent acting on

false positives and where possible allows the system to continue to improve itself over time. Methods for informing about falls include audible alarms, Short Message Service (SMS), Multimedia Messaging Service (MMS), automatic voice calls, emails, and Twitter messages. Challenges of smartphone-based solutions highlighted include quality of smartphone sensors, energy consumption and battery life, and smartphone placement and usability issues. The paper also discusses the possibility of having a smartphone operate in a mode where it is not mounted on the user's body yet it can detect things happening in the environment.

[50] carried out a survey of fall detection solutions that specifically used the Android Operating System. Fifty-six works presented between 2009 and 2014 were reviewed. Results showed that the solutions' general architecture could be classified as body-worn (i.e. wearable) or context-aware; the systems consisted of either smartphones alone, smartphone together with external sensors, or specific devices; the android devices used (mainly smartphones) functioned as one or more of the sensor, data analyzer for fall detection, communication gateway or the remote monitoring unit; sensors used included accelerometers (built-in and external), tri-axial accelerometers(built-in and external), magnetic sensors, magnetometers, orientation sensors, gyroscopes, Doppler sensor, sensor tags with inertial units, barometers, temperature and humidity sensors, cameras, microphones, and visual sensors; and the algorithms used could be classified into two groups namely Pattern Recognition Methods(PRM) and Threshold Based Detection (TBD). Pattern Recognition Methods *"…employ data bases, training phases and AI (Artificial*

Intelligence) solutions.". PRMs also include machine learning techniques including neural networks. The paper then proceeded to present a system developed for monitoring elderly people with early dementia. The developed system provided a platform with the option of utilizing any of five algorithms previously developed by other researchers, using preferred user data, and in case of a confirmed fall, notifying approved contacts as well as recording the location of the person.

[51] in work presented in 2015 utilized the acceleration sensor and orientation sensor on a smartphone to detect falls among the elderly. Information gotten from the sensor was converted into character strings and then to check for a fall, a string matching algorithm using the Longest Common Subsequence (LCS) is used to compare a character string with an already known string. The known string referred to as the "*feature string"* is achieved by extracting a common feature from a training set of falls. When a fall is detected, a contact person can be contacted either by SMSs or instant messages. The system developed also gave the option to manually adjust system settings to "*turn on/off the fall alarm, select the alarm music, set the alarm time, choose an emergency contact person, and pre-type the help message*." Results were presented were tested using two metrics: sensitivity and specificity. The paper defines these as follows: *Sensitivity indicates the ability to identify fall events correctly, and specificity indicates the ability to exclude non-fall events correctly.* Sensitivity was 89% and specificity was 98%. Moreover, it was possible to update the feature string as new falls were detected.

Also in 2015, [44] presented ongoing work that sought to use both thresholds and machine learning to carry out fall detection. The smartphone used was an iPhone4 which had built-in tri-axial accelerometer, gyroscope and magnetometer. The paper presented the initial work that had been done using thresholds. Unlike some previous work, the iPhone4 used was put in the left pocket of the user. This was expected to be more comfortable than say the waist. Using the measurements from the smartphone's sensors, five ADLs were detected and differentiated namely: "*…comfortable walking, …stand-to-seated posture, … seated-to-standing posture, …pivoting at the waist to pick up an object, and …stand-to-seated-to-laying transition"*.

In 2017, [52] applied machine learning, specifically a k-Nearest Neighbor(kNN) classifier, to the data gathered using a smartphone's accelerometer. The triaxial data from the accelerometer was first run through a TBA which had thresholds on magnitude and time. After this, the kNN classifier was then used in a bid to improve the accuracy of the algorithm. Moreover, the system developed also had a part that "monitors *the activation level of the sensor and aims keeping battery consumption as low as possible and another that adjusts the model to the user activity patterns."*. The results showed a sensitivity of 97.53% and a specificity of 94.89%. The paper concluded that "*a machine learning classifier definitely improves the detection performance of a threshold based algorithm"*.

More recently, in 2018, [53] presented work using the values gathered by accelerometer, gyroscope, distance, direction and light sensors in a smartphone for action recognition and fall detection. By 'actions', the paper is referring to what in other papers were referred to as ADL. Actions detected included falling, walking, running, moving up and down stairs and sitting. Depending on the source of the data, either the Kalman Filter or the Mean Filter was used. Also, what part of the algorithm was used or run depended on the location of the smartphone. The algorithm developed is reported to have 90% accuracy.

Other smartphone-based solutions not discussed above include [54],. Also [41] presents a smartwatch based solution that makes use of the watch's triaxial accelerometer.

A3. Radio-frequency Identification (RFID) for Fall Detection

Fall detection systems have been proposed that use wearable RFID.

In 2010, [55] presented work in which they inserted RFID modules, consisting of an active tag and a passive reader, into slippers and combined this with several passive tags affixed to the floor at different locations and active readers and a computer. Together this system was referred to as a "*RFID Gait Monitoring System (RGMS)*". The system worked by monitoring the stride length of the user: a stride length less than 25cm was deemed indicative of an abnormal gait and an alarm was sent to caregivers. The system provided both "quantitative and graphical feedback" from both slippers.

In 2013, [56] showed the development of a system that used a wearable RFID tag, consisting of a triaxial accelerometer and a microprocessor, to detect when elderly people exited their beds as a way of preventing falls. RFID Antennae were placed around the room and powered the RFID tag. The algorithm used for determining the activities of the user was based on conditional random fields (CRFs) which are a class of machine learning classifiers. The activities of the user were classified into lying, sitting, out-of-bed. While this was not directly fall detection, the system was used with the aim of allowing caregivers to quickly respond to a situation and prevent a fall.

B. Non-wearable Technologies for Fall Detection

These technologies do not require the user to have anything on their bodies. Instead they are deployed in the environment of the user and carry out their detection in this way. Thus their main advantage is that they do not require the participation of the user [37].

They are referred to by various names including external sensors[57] and passive sensors[6].

B1. Camera/Depth Sensor for Fall Detection

Cameras/Depth Sensors are perhaps the most popular form of non-wearable technologies for fall detection. Their main advantage lies in their being non-participatory. Furthermore, there has been increasing development in the field of computer vision over the recent years. However, the main challenges/disadvantages faced when using cameras/depth sensors for fall detection are those of invasiveness of the privacy kind [58] and Occlusion[57]. We shall review some work in this field next.

In their 2010 work, [59] proposes a realtime video based detection that uses a combination of the subject's posture (skeleton information) and shape to determine if a fall has occurred. A detected fall is confirmed using "*inactivity of a person for a period of time".* Preliminary testing was done using OpenCV and a single camera. Forward, backward and sideways falls were compared with ADLs like walking running and squatting. Results gave a detection rate of 90.9%, a false alarm rate of 6.25% and an execution time of 4.21 seconds.

Moreover, [60] also presented work that used the location of a subject's head and the location of a subject's feet as seen by a single camera to detect falls. The work considered six possible fall scenarios namely: backward fall, forward fall, lateral fall to the right, lateral fall to the left, syncope and neutral. They also compared falls with ADLs specifically walking, bending and seating. Results showed a recognition rate of 96%.

In 2012, [61] presented a posture based fall detection system for use on an elderly system who lives alone. Human features and posture were extracted using background subtraction, ellipse fitting and projection histogram. The extracted postures were then classified using a directed acyclic graph support vector machine. This classification was combined with information about the floor to conclusively determine if a fall had occurred. Tests resulted in a fall detection rate of 97.08% and a false detection rate of 0.8%.

In work presented in [62], [63], researchers developed a fall detection system that used a specially designed stereo vision system, a field programmable gate array(FPGA) and a digital signal processor(DSP) and a wireless communication module. The stereo vision system consists of two detectors and unlike typical cameras, delivers "*asynchronous* *events*". The papers highlight that this is a solution to the privacy concern typically associated with camera fall detection systems. The FPGA is used for stereo matching and the DSP runs a neural network. Tests showed a fall detection rate greater than 96% with false positives less than 5%. The communication module is used to send the information when fall is detected.

B2. Radio Frequency (RF) Technologies for Fall Detection

These are another very common group of non-wearable technologies for fall detection. Unlike cameras, these technologies are said to be less invasive with respect to privacy since the actual image of the subject is not used. We will review some of such works next.

[58] in 2012 presented work that uses a Doppler sensor to detect trip and fall. According to the paper, the Doppler sensor "*emits microwaves, and outputs an electric signal according to the Doppler frequency, which is the difference in frequency between the transmitted wave from the sensor and the reflected wave from a moving object*". A Fast Fourier Transform (FFT) is then used to distinguish signals representing falls from those representing six other ADLs namely: walking, shaking arm and hand, working with hands, stationary posture, standing up, and empty room. The paper defines a trip and fall as "*Falling after walking a straight line*". An embedded computer, the BeagleBoard-xM, running the Android Operating System is used to process the signals. Evaluation was carried out first by testing in a single direction, then in multiple directions (front, left,

right, opposite) and with multiple people. In a single direction, true positives were at 90% while the largest false positive was at 1.9% which occurred when nobody was in the room. In multiple directions, maximum true positive was 96% while the largest false positive was 6.6% which occurred when the subject was walking in the right direction. With multiple participants, maximum true positive was also 96%.

[6] in 2019, presented work that applied a convolutional neural network (CNN) and a state machine to RF signals for fall detection. The project, called Aryokee, was said to present the "*first CNN architecture for RF-based fall detection*", could be applied to new environments and people, and could determine when a subject fell, stood up and how long a fall occurred for and can be used in situations with multiple people. Over 40 ADLs were considered side by side with different fall types including "*falling forward, backward, on position, and sideways*". The fall types were expanded upon based on their causes resulting in eighteen different fall classifications. For testing, the authors used over 140 people in 57 different environments and results showed a recall of 94% and a precision of 92%.

B3. Other Non-wearable Technologies for Fall Detection

[38] in work presented in 2011 utilized several boxes containing accelerometers which were mounted on the floor. The project named 'eHome' utilized Fast Fourier Transform and other algorithms to extract specific details from the vibrations detected by the

network of boxes. This information was then sent to a server which used other algorithms to determine if a fall had occurred based on a determined threshold. The project also provided a way to crosscheck if the subject was alright, after a fall is supposedly discovered, by prompting the user on a touchscreen and also monitoring other sensors in the room for activity before sending notice to points of contact who can take further action. In extensive testing in the laboratory and considering different types of floors, eHome achieved a sensitivity of 87% and a specificity of 97.7% for a specified fall scenario.

C. **Hybrid Technologies** for Fall Detection

While there have been wearable and non-wearable solutions for fall detection, in a number of cases, fall detection systems combine a number of solutions both within and across the groups. This in certain cases allows the leveraging of the advantages that the individual solutions may have, to counterbalance the disadvantages of other solutions.

We review some of such hybrid solutions below.

We have the proposal and development of "wireless sensor networks" defined as a combination of "networks of wearable sensors (i.e. body sensor networks) and networks of ambient sensors´.

[40] in 2014 presented a solution that consisted of tri-axial accelerometer(s), active RFID tags and an RFID reader. The tri-axial accelerometer(s) (the study does not state how many) is integrated into the RFID tag which is then worn on the user. Information received

by the RFID reader from the tag is then analyzed using Machine learning to detect if one of six fall postures, namely *"…frontal fall, back fall, left fall, right fall, kneel fall, and hip fall"*, occurs.

[64] in their work combined a Kinect (depth sensor/camera, non-wearable) and a smartphone(wearable). The smartphone's accelerometer is used to detect if a fall has occurred by comparing its tilt angle (TA) and signal vector magnitude(SVM) with determined thresholds: 40 degrees and 2.5g respectively. The Kinect is mounted in a fixed location in the room and utilizes information from the user/subject's skeletal frame. The skeletal information is extracted from both the color and depth information provided by the Kinect. Two key metrics are used to determine a fall: head speed and the head height. As with the smartphone, thresholds are used to determine if the user has fallen. A head speed greater than 0.2m/s and a head height lower than 50 cm are considered to mean the user has fallen. Information from both the smartphone's accelerometer and the Kinect are sent to a web server which then integrates them. The main advantage of this work is that it reduces cases of false positives from each separate detection method by combining them. In a case presented in the paper, the researchers were able to distinguish between a case of sitting down and a fall because of the added layer provided by the Kinect's skeletal information. Where a fall is detected, information including the location of the user is sent to the "family, friends, or help center of the subject immediately"

In 2018, [39] presented a fall detection system that combined a Kinect (depth sensor/camera, non-wearable) and an accelerometer(wearable). The system was event-driven and combined depth maps from the Kinect with readings from the accelerometer. In line with the event-driven approach, readings from the accelerometer that were likely to indicate a fall triggered the depth map from the Kinect to confirm whether a fall has occurred. Furthermore, two positions for the Kinect were considered. In the first case, the Kinect was mounted on a pan-tilt on the ceiling. This allowed the Kinect to be moved thus increasing the total possible viewing area of the depth sensor. In the second case, the Kinect was mounted on the wall and faced the user. Classifiers are used on both the accelerometer data and the depth images to determine falls. Results showed a very high sensitivity and specificity with the wall mounted Kinect facing the User having slightly better results than the ceiling mounted Kinect. However, the best results were gotten when both the wall mounted and ceiling mounted Kinects were used.

D. Similar Work

In 2017, [65] improved on previous work working with a mobile robot(Kobuki), a Kinect sensor, a side sensor(Hokuyo URG - 04LX) for determining distance and a PC. The robot followed the elderly person in their household. In their work, they made use of the skeletal based detection from the Kinect combined with determined thresholds to tell if a fall has occurred. The mobile robot followed the elderly person in the space using the information from the side sensor while the Kinect Sensor focused on the fall detection.

Furthermore, an obstacle avoidance approach was implemented. In testing, the following approach was considered effective while they concluded that their obstacle avoidance approach was not satisfactory. Not much was said about the fall detection except that in a previous work they had found that by moving the sensor instead of having it fixed resulted in an 80% improvement.

CHAPTER III: METHODOLOGY

We start with the problem of following just one person in a mildly occluded space. While not all-embracing, this approach still has real-world relevance. This is because as we stated in previous chapters, a substantial number of seniors do live alone. Such seniors are likely more at risk if they fall than seniors who live in communities, retirement homes, with family or even with other seniors.

Our prototype builds on an existing prototype robot controlled by a Raspberry-Pi and two motor drivers. We are using Computer Vision based Object detection as the detection methodology. In computer vision, there are number of possible options for person detection including Face Detection/Recognition and Object Detection. While Face Recognition provides the opportunity for customization, it is our opinion that based on what is currently available the effort required to retrain the model for every single new person makes it difficult. Object detection on the other hand, currently once trained can detect people as a class and, since we are looking at navigation in a space with only one human, is sufficient for our purpose. We discuss this further in sections below.

A. Existing Robot Prototype

Our existing robot prototype was built at the Real-time and Embedded Control, Computing, and Communication (REC^3) Lab, Middle Tennessee State University (MTSU) by a student, Christopher Secrest, and is named the Fall Detection Robot (FADER). Initially, FADER consisted of a 3D printed frame, four DC motors with encoders, a three cell 4000 mAh 12 V battery, a Raspberry Pi 2 Model B, two L298 motor drivers, an ultrasonic (Ping)))™) sensor and two passive infra-red (PIR) sensors. The four DC motors functioned as the wheels of the robot. Each of the motors is a RF-370CA-15370 metal-brush motor with an operating range of 3V – 12V and a nominal voltage of 12V. The Raspberry Pi served as the brain of the robot and was programmed using the Python programming language. In programming the Raspberry Pi using Python, the following libraries and modules were used: pigpio, RPI.GPIO and time. The pigpio is a library that allows control of the Raspberry Pi's GPIO, PRI.GPIO is a module that is also for control of the Raspberry Pi's GPIO and the time module allows use of the sleep method to handle pauses in the program. Furthermore, the L298 motor drivers allowed the control of the DC motors. The L298, as an IC, consisted of the H-Bridge and allowed it to be able to control the DC motors. Each L298 motor driver could thus control two DC motors at a time. Distance readings from the ultrasonic (Ping)))™) sensor are used to set up a threshold-based algorithm for the robot's navigation. The PIR sensors are used to orient the robot. Signals from both PIR sensors are interpreted by the control program and then instructions are sent to the motors allowing the robot to orient itself. The battery provides power supply for the Raspberry Pi, motors, motor drivers, sensors, and is rechargeable. Figure 2 shows a picture of FADER while Figure 3 shows the circuitry of FADER electronic connections. As discussed in sections below, we eventually made a number of changes to the initial prototype. A table showing the list of parts for FADER is given in the Appendix.

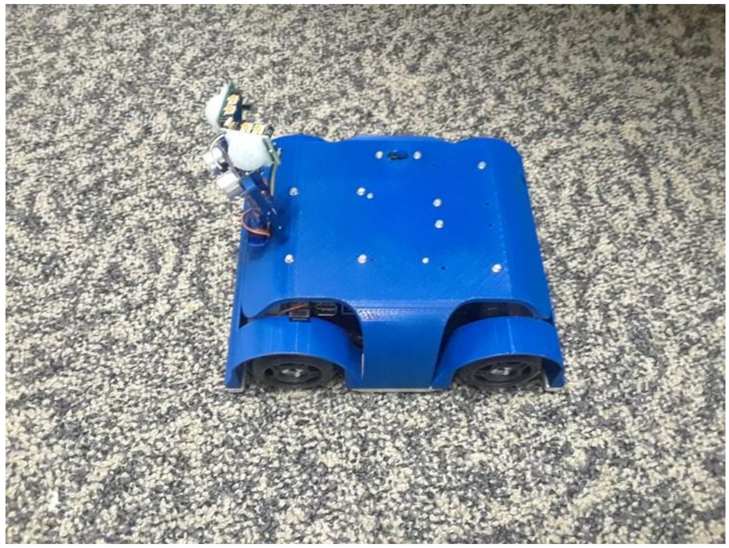

Fig. 2. FADER robot (initial prototype).

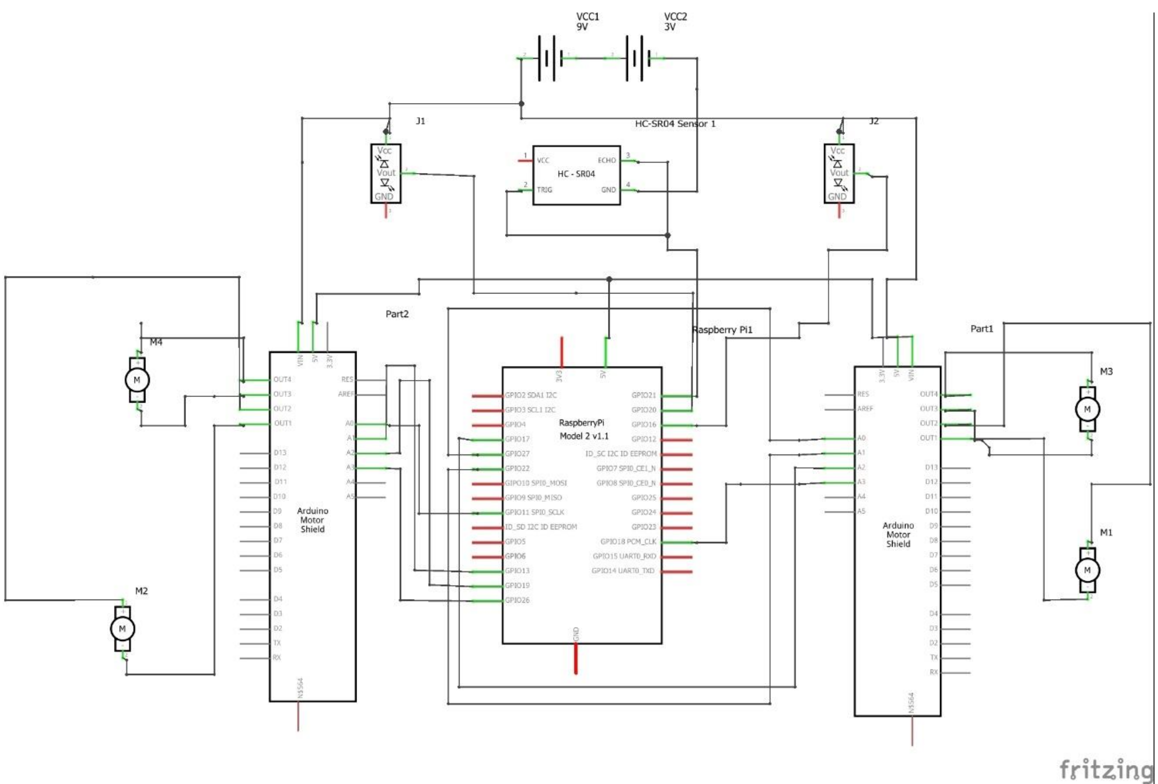

Fig. 3. FADER circuitry.

B. Computer Vision Based Object Detection

We decided to use the Raspberry Pi Camera as the main medium for human detection. This allows us to take advantage of the current advances in computer vision and machine learning.

The Rapsberry Pi Camera comes in two types: the regular which is referred to as the Camera Module and which works with visible light, and the infrared which is referred to as the Pi NoIR and works with infrared light[66]–[68]. The main difference between the two camera types is that the Pi NoIR has no infrared filter thus allowing us take pictures in the dark, when infrared lighting is available[67]. This was a preferred feature for us since such would allow us to carry out detection even in the dark, if necessary. However, we tested both camera module types and compared the images they took. We found that the images were comparable in normal lighting. Figure 4 shows the two camera types and the images we took. Because the images were comparable and because of the advantage of the Pi NoIR camera, we chose to use the Pi NoIR camera. The Pi NoIR camera was mounted on FADER and then connected to the Raspberry Pi using a camera and display extender.

Specifically, we are using deep learning to run object detection on the Pi Camera. A lot of computer vision solutions today use OpenCV which is the de facto open source library for computer visions as well as Neural Networks. Our approach makes use of OpenCV and Neural Networks together with other methods as discussed in this section.

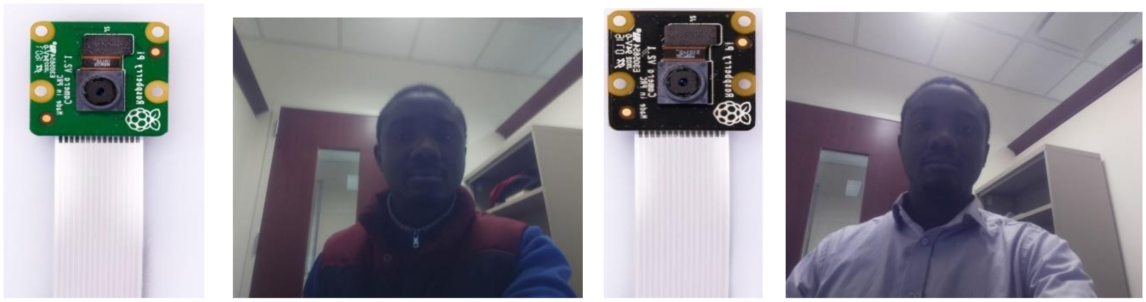

Fig. 4. Regular Camera Module, and Pi NoIR Camera Module and the pictures taken with them respectively.

In discussing computer vision for detection, various uses exist including person tracking, skeletal tracking (also known as pose annotation), Face Detection, Face Recognition and Object Detection. Each of these areas have different techniques that are used to carry them out. Today with machine learning, each of these uses is becoming far more advanced than what was available in past years. Each of these options was considered for our current requirements and we explored specifically Face Detection, Face Recognition and Object Detection. Face Detection occurs when a computer can detect a face in a picture or video. In order to achieve this, the computer is usually “trained” using a positive dataset and a negative dataset. The positive dataset contains many (hundreds-thousands) pictures with faces while the negative dataset contains many (hundreds-thousands) pictures without faces. Once a machine has been trained, it is able to detect a face in a picture or video. One of the disadvantages of this method that we encountered was its ineffectiveness in low lighting. Face Recognition goes one step beyond Face

Detection by allowing the computer to recognize specific individuals. Whereas Face Detection can only detect a face without necessarily identifying who it is, Face Recognition allows the identifying of the face that is detected. In its own case, Face Recognition requires training the machine with the images of the specific person(s) that are to be recognized. As a result, even though specificity improves, Face Recognition has the disadvantage of being more computationally intensive as well as requiring 'retraining' every time a new person has to be recognized. Object Detection allows the detection of objects based on previously trained networks. Objects to be detected are referred to as classes. In Object detection, it is possible to have a class that is for humans. Advantages of this method include our not needing to retrain the machine for every new item/object as already pretrained networks can work as well as the fact that it detects not just humans but as many objects that have been trained as classes, which offers the best of both worlds especially when considering navigation in the home. We chose Object Detection for our project because of this. To carry out our object detection, we settled on a deep learning architecture. Simply put, the difference between traditional machine learning and deep learning is that deep learning uses several layers of machine learning with a previous layer feeding the succeeding layer. Moreover, in deep learning we can have convolutional neural networks (CNN) which have been pre-trained on image datasets function as the backbone architecture or base network to the detection framework[69], [70]. So, though quite a simplification, we can say that deep learning object detection has two parts: the CNN referred to as the base network/backbone architecture and an object detection framework[69], [70]. Figure 5 shows a picture of some one stage deep learning detection frameworks. There are three options that were considered for the object detection framework, the Faster Region Convolutional Neural Networks (Faster R-CNN)[71], Single Shot Detectors(SSD)[72], and You Only Look Once(YOLO)[73]. While Faster R-CNN is a two stage detector, YOLO and SSD are one stage detectors. Faster R-CNN has the advantage of being known as the most accurate of the three methods. However, it is quite slow in running. It is therefore most suitable for tasks where speed is not a premium, but accuracy is priced. YOLO has the fastest speed of the three frameworks but has a lower accuracy. Furthermore, currently implementations of YOLO on a device with limited resources like the Raspberry Pi without GPU support is far from satisfactory with most projects opting for the YOLO tiny variant[74]–[78]. SSD, which was developed by Google, occupies a sort of middle road between both Faster R-CNN and YOLO in terms of accuracy and speed. Also, it has been implemented on the Raspberry Pi. Therefore, even though the speed of YOLO would have been preferred for our application, we are using SSD. The backbone architecture we are using for the SSD framework is MobileNets [79] which was also developed by Google. MobileNets is especially good for mobile or embedded systems which have constrained resources and was shown to have "*significantly reduced computation cost as well as the number of parameters without significant loss in classification accuracy"*[69]. The model we are using after training and finetuning by [80] had 21 total classes it could detect including the background[81]. When running our software, we surround the object detected with a bounding box as well as print a label at the top of the bounding box telling what the object is and showing the percentage of certainty. The percentage of certainty, confidence level, further improves the control that can be exercised during navigation since it is also possible to use this value to handle false positives and false negatives especially in areas where lighting is of concern.

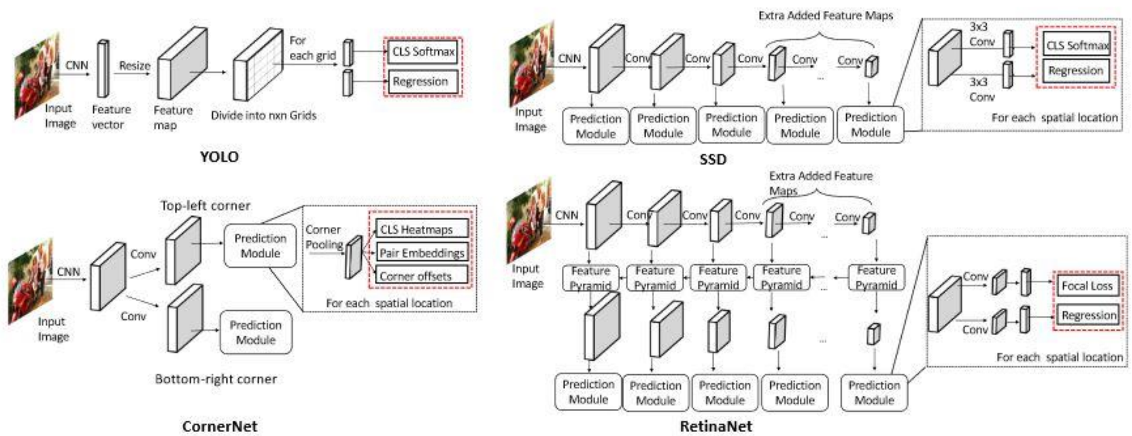

Fig. 5. Some one stage deep learning frameworks [69].

False positives occur when the software detects a person where there is no one and false negatives occur when a person is present, but the software detects something else. Both scenarios have unwanted implications for our research and the general application of human-following robots. This is because if false positives and false negatives are not dealt with, the robot cannot be effective enough. As previously stated, we have noticed that in certain cases, the prevalence of false negatives is high. The percentages/confidence level provide a possible solution to this conundrum. It is therefore possible to finetune the robot's response based on the results from the object detection as it is being run. The robot will simply be instructed to disregard detections that are below a certain percentage. Figure 6 shows an image taken by the Pi Camera with deep learning object detection running during initial tests. The tests were carried out using a Raspberry Pi 3 Model B which eventually replaced the original Raspberry Pi 2 Model B which was in FADER. The code was written in Python 3.0. In addition to pigpio, RPI.GPIO and time, the following libraries and modules were used: imutils[82], math, numpy, argparse, cv2, and serial.

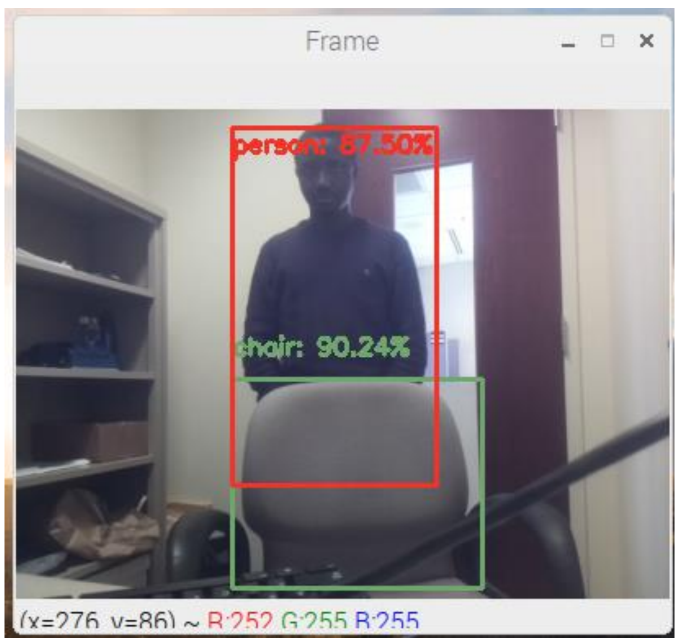

Fig. 6. Object Detection running on the Pi Camera.

C. Navigation in One-Dimensional(1-D) Space (the Distance Estimation Problem)

With object detection running on the Pi Camera on FADER, we were able to detect a total of 20 classes in addition to the background[70]. However, for our starting purposes, we are only focusing on the “person” class. So, we ignore all the other classes and only focus on the person class for navigation. This also works because for our initial tasks, we are considering only one person in a room fairly free of obstructions. With the person detection sorted, we set about considering a suitable method for distance estimation. Generally, in distance calculation, it is required to either calibrate the camera or have a reference object whose dimensions are known in the image/video. We could not use the later method since for our purposes, the robot would not always have a reference object in its scope. However, we noticed that the dimensions of the bounding box which is drawn around the detected object generally changed depending on how far or close the object is to the robot. Specifically, the width of the bounding box increased the closer the object was to the robot while it reduced the farther away the object was from the robot. Figure 7 shows the relationship of distance to the width of the bounding box for a person with a given height of 5 feet 10 inches and body width of 20.5 inches. The Pi Camera is slightly inclined on the robot to allow its field of view to include the standing human. As shown in the figure, when the robot is at a distance of six feet or greater the relationship is approximately linear. A similar effect is seen when comparing the distance to the height of the bounding box as shown in Figure 8. However, in our tests we found that the bounding box did not always cover the complete height of the subject even when the closeness of the subject to the camera was not a factor.

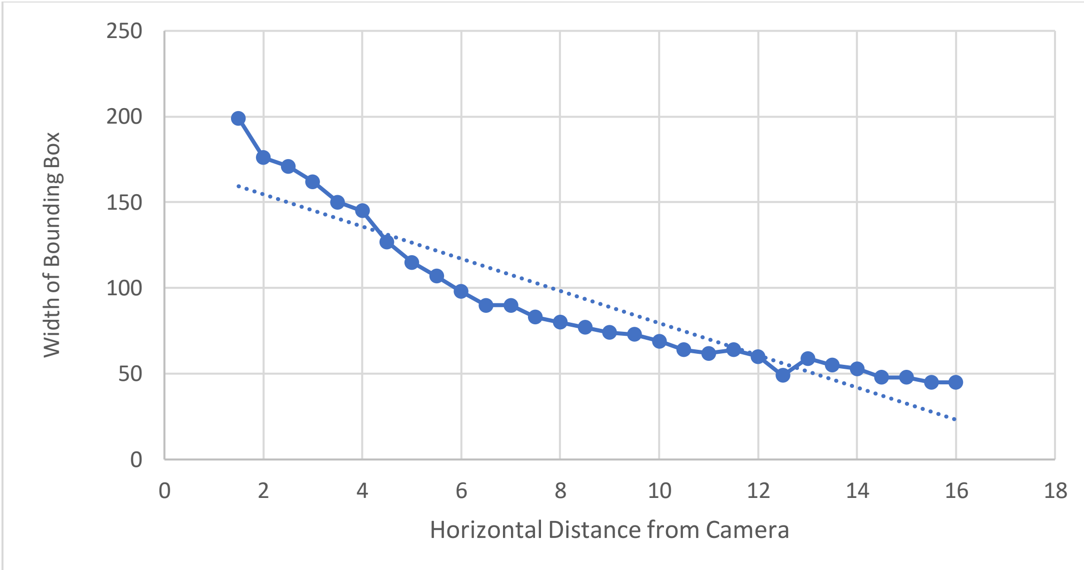

Fig. 7. Width of bounding box versus horizontal distance.

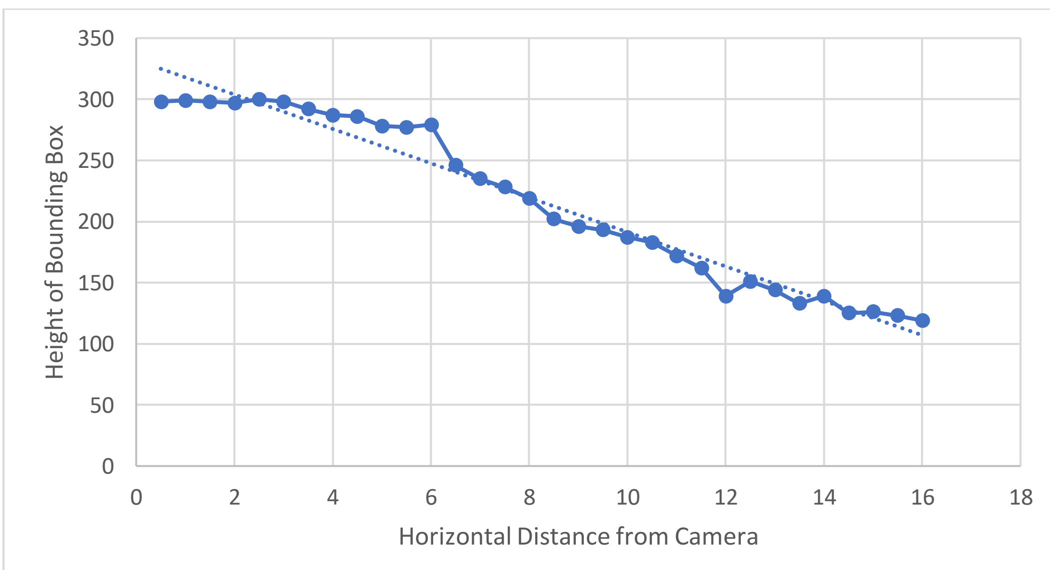

Fig. 8. Height of bounding box versus horizontal distance.

For distance estimation in one-dimensional space, first we used the raw values of the width of the bounding box to set limits. This worked adequately and the robot was able to operate in the three states which we discuss below. Next, we carried out a linear regression analysis using Microsoft Excel to get a formal mathematical relationship between the distance and the width of the bounding box, and between the distance and the height of the bounding box. Tables 1 and 2 show the results of the regression analysis including the coefficients and the ANOVA results. The following equations were thus derived:

$$D = 17.1838 - 0.0924W \quad (3.1)$$

$$D = 23.0591 - 0.0686H \quad (3.2)$$

where D is the horizontal distance to the camera in feet

W is the width of the bounding box in pixels

H is the height of the bounding box in pixels

TABLE 1 REGRESSION ANALYSIS ON WIDTH OF BOUNDING BOX AND HORIZONTAL DISTANCE

SUMMARY OUTPUT								
Regression Statistics								
Multiple R	0.93155509							
R Square	0.86779488							
Adjusted R Square	0.86307327							
Standard Error	1.62878958							
Observations	30							
ANOVA								
	df	*SS*	*MS*	*F*	*Significance F*			
Regression	1	487.5922463	487.5922	183.7920945	7.95374E-14			
Residual	28	74.28275373	2.652955					
Total	29	561.875						
	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	17.1837949	0.689521015	24.92135	1.17898E-20	15.77137517	18.596215	15.7713752	18.5962147
Width of Bounding Box	-0.0924083	0.00681628	-13.557	7.95374E-14	-0.106370789	-0.078446	-0.1063708	-0.07844575

TABLE 2 REGRESSION ANALYSIS ON HEIGHT OF BOUNDING BOX AND HORIZONTAL DISTANCE

SUMMARY OUTPUT									
Regression Statistics									
Multiple R	0.981904003								
R Square	0.964135472								
Adjusted R Square	0.962939988								
Standard Error	0.902950867								
Observations	32								
ANOVA									
	df	*SS*	*MS*	*F*	*Significance F*				
Regression	1	657.5403919	657.5404	806.4811	3.07019E-23				
Residual	30	24.45960806	0.81532						
Total	31	682							
	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*	
Intercept	23.0591015	0.54535546	42.2827	2.67E-28	21.94533706	24.172866	21.9453371	24.1728659	
Height of Bounding Box	-0.068570576	0.002414575	-28.3986	3.07E-23	-0.0735018	-0.063639	-0.0735018	-0.0636394	

Having estimated distance using the two approaches highlighted above, (i.e. raw bounding box dimensions, and regression analysis), FADER operates in one of three states: approaching, waiting and retreating. In 'approaching', the robot is at a distance that is too far away from the person detected and so it moves towards the person until it reaches a predetermined distance at which it stops and just waits (second state). This second state is a distance range within which FADER does not move and keeps observing the detected individual. If the person moves closer causing the distance between them to be less than the preset distance, FADER retreats thus increasing the distance between them until it reaches the preset distance again (third state). The pseudocode for this is shown next

1. Obtain Image from Pi Camera
2. Run object detection on frame

3. Dismiss detections with confidence levels lower than set value
4. Evaluate object detected in frame
5. If object detected is not human

    5a. Discard Detection

    5b. Go back to 4

6. If object detected is human:

    6a. Obtain the dimensions of the bounding box

    6b. Estimate distance using the regression formula

    6c. If distance is less than 6 feet

        i. Retreat

    6d. If distance is greater than 12 feet

        i. Approach

    6e. If distance is between 6 feet and 12 feet

        i. Do nothing

7. Return to Number 1

While the values obtained for distance are not exact, they are mostly useful as an estimate that allow FADER to decide whether to approach, not move or retreat. In most cases, we do not need this preciseness.

D. Navigation in Two-Dimensional (2-D) Space

For 2-D navigation, FADER has to also be able to track the user's movement in the plane perpendicular to the initial plane used during 1-D navigation. Assuming that FADER starts in a 'waiting' state of the 1-D navigation; when the user moves in the perpendicular plane, FADER then turns or pivots to face the user again at which point the navigation becomes a 1-D navigation problem once again. FADER then returns to one of the three states discussed previously: approaching, waiting and retreating.

2-D Navigation = Tracking + Turning/Pivoting + 1- D Navigation

Tracking

To enable FADER to track the user in the perpendicular plane, we used the x-coordinates of centers of the frame and the bounding box drawn around the detected person. The entire displayed frame of the camera is 400 by 300 with the origin being the top leftmost point. The origin has coordinates of 0 by 0. This makes the coordinates of the center of the frame 200 by 150. Furthermore, since the center of the bounding box is a way to approximate the center of the detected person, by comparing the x-coordinates of the center of the bounding box and the center of the frame, we can tell how far away the person is from the center of the frame and in what direction. The aim is to keep the user close to the center or within an acceptable range.

To calculate the center of the bounding box (x_c, y_c) we use the following equations:

$x_c = (x_1 + x_2) / 2$

$y_c = (y_1 + y_2) / 2$

where the coordinates (x_1, y_1) represent the top leftmost point of the bounding box and (x_2, y_2) represent the lowest rightmost point of the bounding box.

Pivoting/Turning

Once we have determined how far from the center the person is, we determine if the robot needs to pivot to keep the user closer to the center of the frame. For FADER, we set a threshold of 100 units to the right and 50 units to the left. If the user is out of this range, then we pivot right or left respectively. The Pseudocode for this is shown below. (X_c, Y_c) represent the coordinates of the center of the frame while (x_c, y_c) represent the center of the bounding box which is approximately the center of the person detected.

1. Calculate $x_c - X_c$
2. If $(x_c - X_c)$ is less than -50:

   2a. Pivot Left

   2b. Go back to 1

3. If $(x_c - X_c)$ is greater than 100:

   3a. Pivot Right

3b. Go back to 1

4. If ($x_c - X_c$) is greater than -50 but less than 100

   4a. Do nothing

5. Continue to 1-D Navigation

Adjustments to FADER Design

While working on two-dimensional navigation, we encountered a number of challenges including non-responsiveness and losing of the human target. To correct for this, we had to make a number of adjustments to the design of FADER. The following changes were made:

1) **Adding an Arduino Uno**: We added an Arduino UNO to handle the motor control. In the initial FADER prototype, the Raspberry Pi handled everything including sending individual navigation instructions to each of the four DC wheels through its GPIO pins. With the addition of deep learning based computer vision to FADER, still having the Raspberry Pi sending individual instructions to the DC wheels seriously slowed down the response time of the robot. Given that the Raspberry Pi 3 is a resource constrained device, running the Object Detection on it was already pushing its limits. We therefore decided to add an Arduino Uno to handle sending individual commands to the DC motors. The Raspberry Pi and Arduino communicated via serial port and functioned in Master-Slave configuration.

According to the Arduino website [83], the Uno is based on the ATmega328P microcontroller and has 14 digital input/output (I/O) pins and 6 analog pins. [83] further stated that six of the 14 digital I/O pins can be used as pulse width modulation (PWM) outputs. Figure 9 shows the specifications of the Arduino board. In controlling the DC motor, we need at least three pins per motor, one of which will be for sending the PWM instructions. The Raspberry Pi continued to be programmed in Python while we coded the Uno on the Arduino IDE which has its own language. According to the Arduino website, the Arduino programming language is "merely *a set of C/C++ functions that can be called from your code*"[84].

2) **Restricting Motor Control**: We also found that sending individual separate instructions to each of the four DC Motors increased the complexity especially when the robot has to pivot to make a turn either left or right. To reduce this complexity, we combined the two wheels on each side of the robot, sending them instructions from the same pins. This way instead of needing at least twelve pins in total, we needed just six. This freed up pins on the Arduino for other possible uses.

3) **Encoder Use**: Each of the DC motors on FADER had a quadrature encoder. Encoders often can be used to tell how far a wheel has moved [85]. We consulted information on similar motors to the ones used in the existing FADER prototype. The quadrature encoders had two outputs and connections for power and ground.

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Fig. 9. Specifications of the Arduino UNO [83].

However, for our purposes, we only used one of the outputs. To use the encoders, we had to manually calibrate motors/encoders. We needed to know how many counts the encoder made per circumference of each wheel. To find this out, we carried out tests as suggested by [85]. First, we calculated the diameter and circumference of each wheel. Next, we wrote a short Arduino program that prints the encoder count to the screen. By moving each wheel one complete cycle, the distance of one circumference, we were able to determine the encoder count per circumference and also then calculate the encoder count per inches. This allowed to be able to estimate how far each wheel robot has gone and stop the robot when needed. Figures 10 and 11 show the Arduino code written for testing and an example of the serial output showing the encoder count; while Tables 3 and 4 show the counts per circumference for each wheel, and the average counts per inches for each wheel.

Based on our calculations, we arrived at an average value of 818 encoder counts per circumference and an average circumference length of 8.38 inches for the robot wheels. We used these values in our motor control code on the Arduino.

4) **Separate Power Supply**: We noticed during testing and using the encoders that different wheels seemed to be running at different speeds. To ensure that the problem was not due to noise, we included a separate power supply for the Raspberry Pi. The Raspberry Pi is powered by a power-bank while the motor wheels continued to be powered by the three cell 4000 mAh 12 V battery. While this reduced the discrepancy in the in the wheel speeds, we eventually had to manually correct for the remaining discrepancy in the Arduino Motor Code.

5) **Seeking Function**: After implementing adjustments 1 – 4, we commenced testing of two-dimensional navigation. While FADER functioned well and was able to

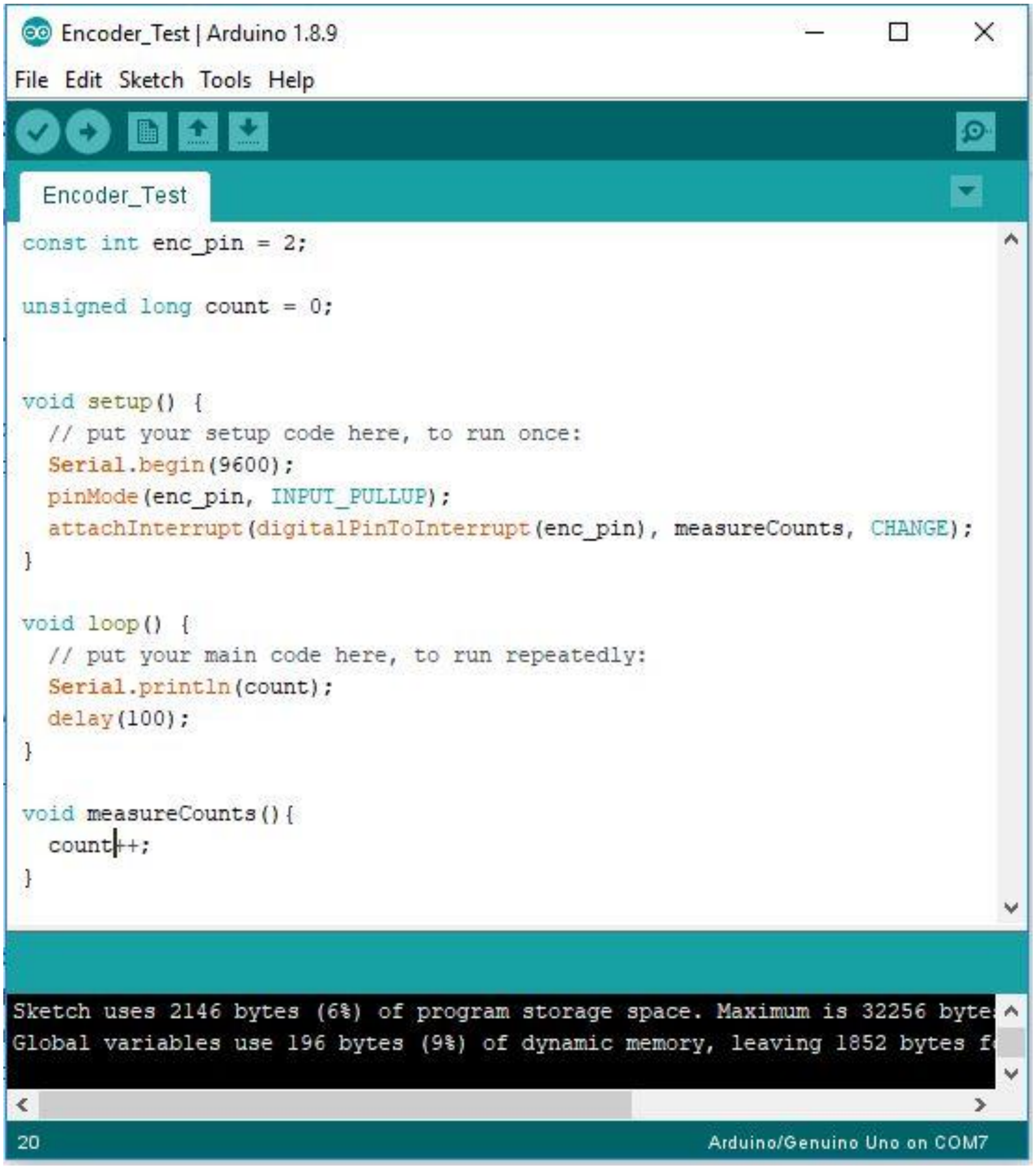

Fig. 10. Arduino Code for Encoder Count.

follow a user in the test space, we noticed that it lost the user once the user stepped out of the field of view of the Pi Camera. To correct for this, we created a separate function for seeking the user If the user is lost. If a user is lost the robot

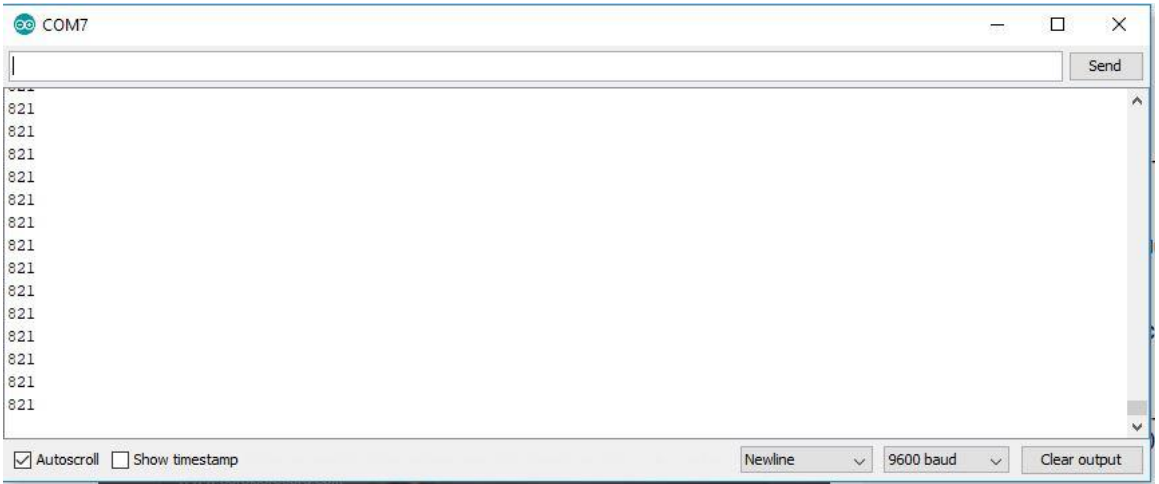

Fig. 11. Serial output showing the encoder count.

TABLE 3 COUNTS PER CIRCUMFERENCE FOR EACH WHEEL

Motor	1	2	3	4	5	6	7	8	9	10	Average Counts/Circumference
RightFrontMotor	812	818	816	818	816	820	818	818	822	826	818.4
RightBackMotor	814	818	820	816	819	814	824	818	818	824	818.5
LeftFrontMotor	813	814	822	822	815	824	818	818	822	820	818.8
LeftBackMotor	822	818	816	820	818	814	818	818	818	818	818

TABLE 4 AVERAGE COUNTS PER INCHES, AND MILLIMETERS, FOR EACH WHEEL

Motor	Diameter(In)	Diameter(mm)	Circumference(In)	Circumference(mm)	Average Counts/In	Average Count/mm
RightFrontMotor	2.667	67.73	8.378627607	212.7800704	97.67709443	3.846224876
RightBackMotor	2.653	67.33	8.33464531	211.5234334	98.20453895	3.869547629
LeftFrontMotor	2.669	67.78	8.384910792	212.9371501	97.6516054	3.845266079
LeftBackMotor	2.664	67.67	8.369202829	212.5915749	97.73929688	3.847753612

pivots in the last direction in which the user was heading and continues pivoting in that direction in a 360°. If the distance to the user was estimated to be greater than six feet (6 ft.), then the robot moves forward before starting to pivot in the direction the user was heading in. Once the user is reacquired, the robot estimates the distance and based on the distance either waits, or approaches or retreats or pivots.

Other actions that we carried out included reducing or removing wait periods. Upon testing, FADER functioned satisfactorily and was able to follow a human in an open space (both free of obstacles and with a few obstacles). We tested with the human moving in a clockwise pattern, and in a random fashion as well. One of the main constraints we continue to encounter is the time of detection. Because deep learning is computationally intensive and the Raspberry Pi is resource constrained, detection time is currently 1.5 seconds. As such, if a subject moves too fast the robot would lose them. However, since our target are the elderly, we do not think they would move too fast for the robot to lose them. Figure 12 shows the modified FADER.

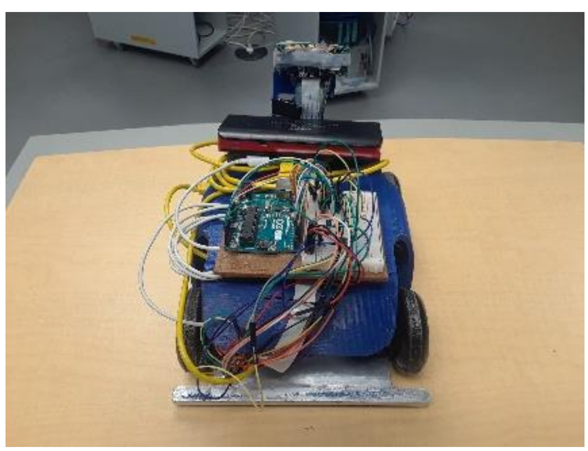

Fig. 12. Modified FADER.

E. Fall Detection

After sorting out 2-D navigation, we proceeded to fall detection. The first thing we did was to run the deep learning object detection on some random videos downloaded from Youtube with people falling to see if the software still recognized people when they fell. We found that the software worked in a number of cases.

After doing this, we recorded videos of people in the lab space moving and simulating falling in the lab space. Falls were simulated in five positions: lying flat on the back, lying flat on the stomach, lying on the right side, lying on the left side and slumping to a seating position with the back rested against the wall. Figures 13 and 14 show images of these videos.

We annotated the videos with our object detection software. The aim was twofold: to see if the software recognized a person in that position, and to see if there were any generalizable characteristics to the detections. Figures 15-20 show that the object detection software recognized people in the various positions tried. The confidence values as percentages are also shown.

From the annotation, we noticed that the ratio of the width to the height of the bounding box changes from typically less than one to close to or greater than one when a person falls. In fact this is one of the metrics used by [39] in their fall detection. The exception was when the person was in the position: slumping to a seating position with the back rested against the wall. Figure 21-26 show the positions with the confidence percentages and the ratio values.

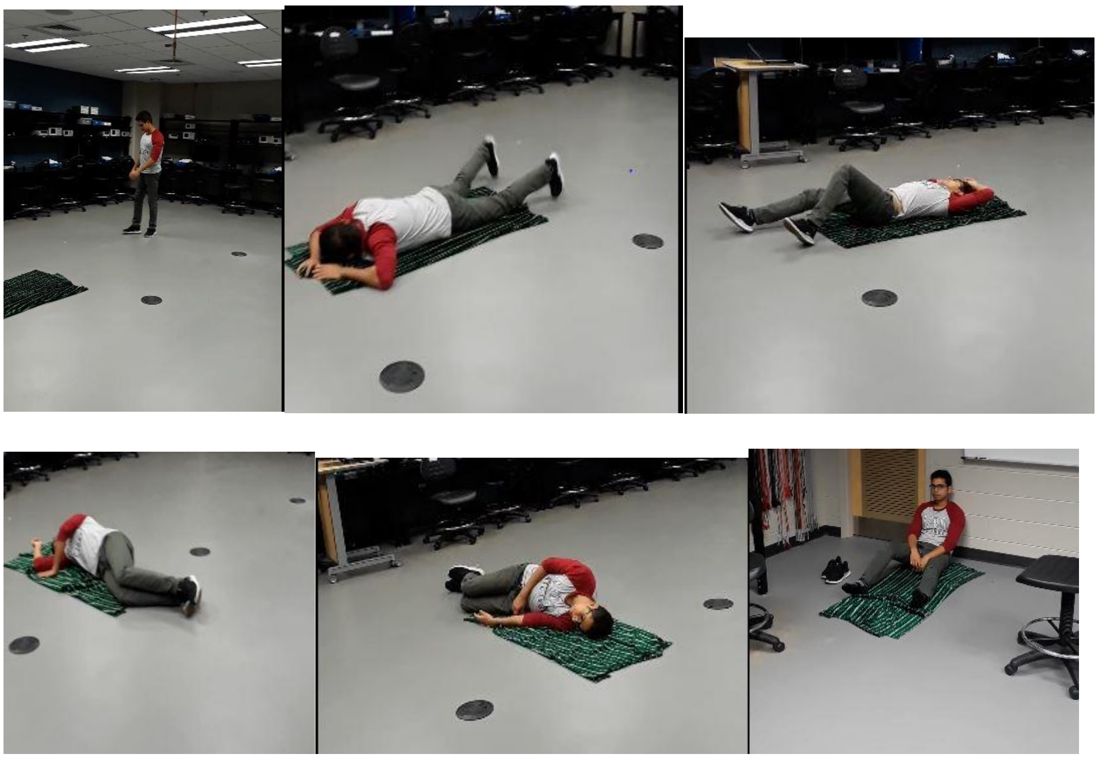

Fig. 13. Moving and Simulating Falls in the Lab (Person 1).

In further annotation, we also calculated the x, y coordinates of the center of the bounding box, which approximates the center of the detected person, to see what happened to these values during a fall. We found that the y-coordinate's value changes and increases. Furthermore, since the person is falling, the y-coordinate value of the center of the bounding box would be greater than the y-coordinate value of the center of the frame.

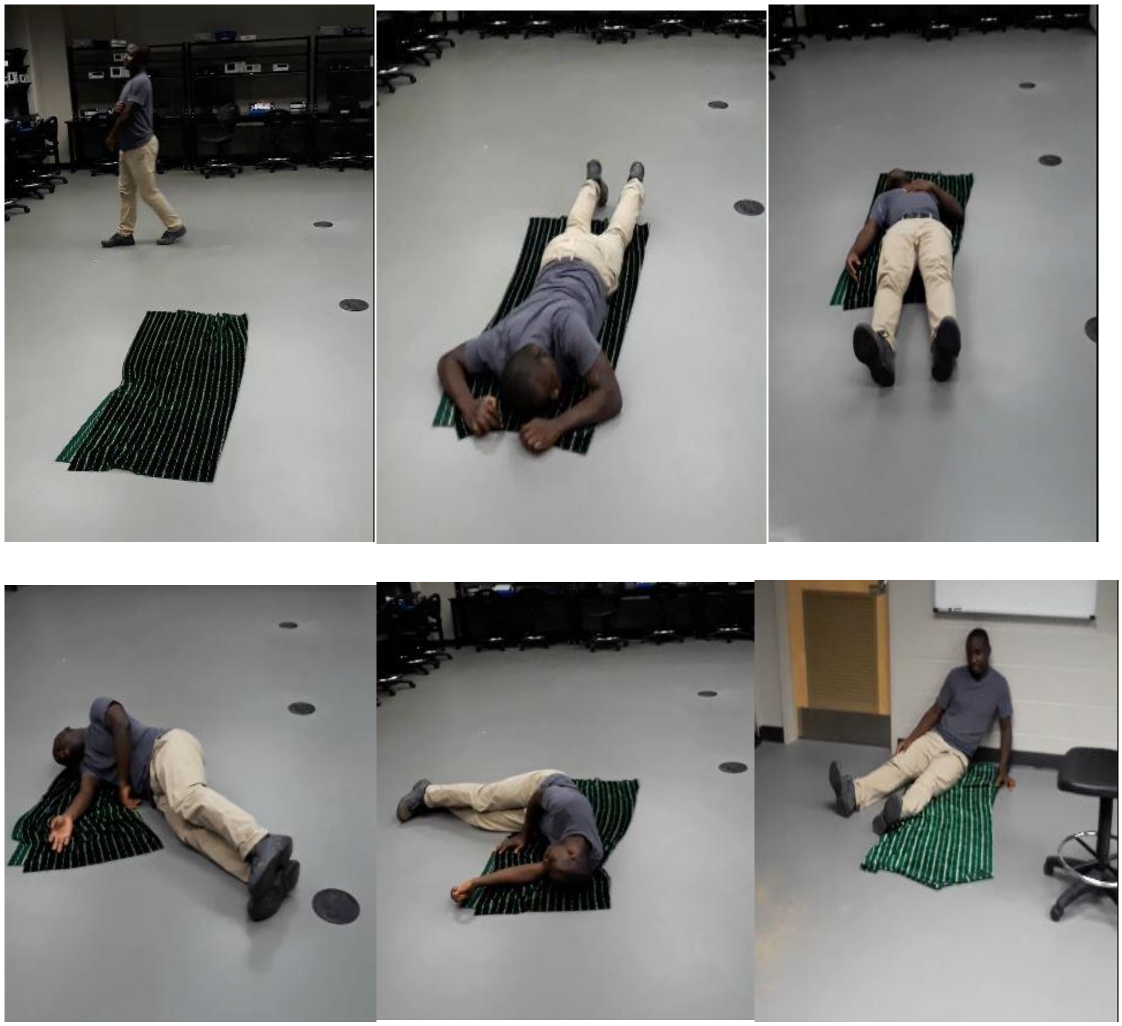

Fig. 14. Moving and Simulating Falls in the Lab (Person 2).

Therefore, our analysis allowed us to use these two metrics for setting a threshold for fall detection:

1. Ratio of width to height of the bounding box greater than 0.8
2. Y-coordinate of the center of the detected person greater than y-coordinate of center of frame.

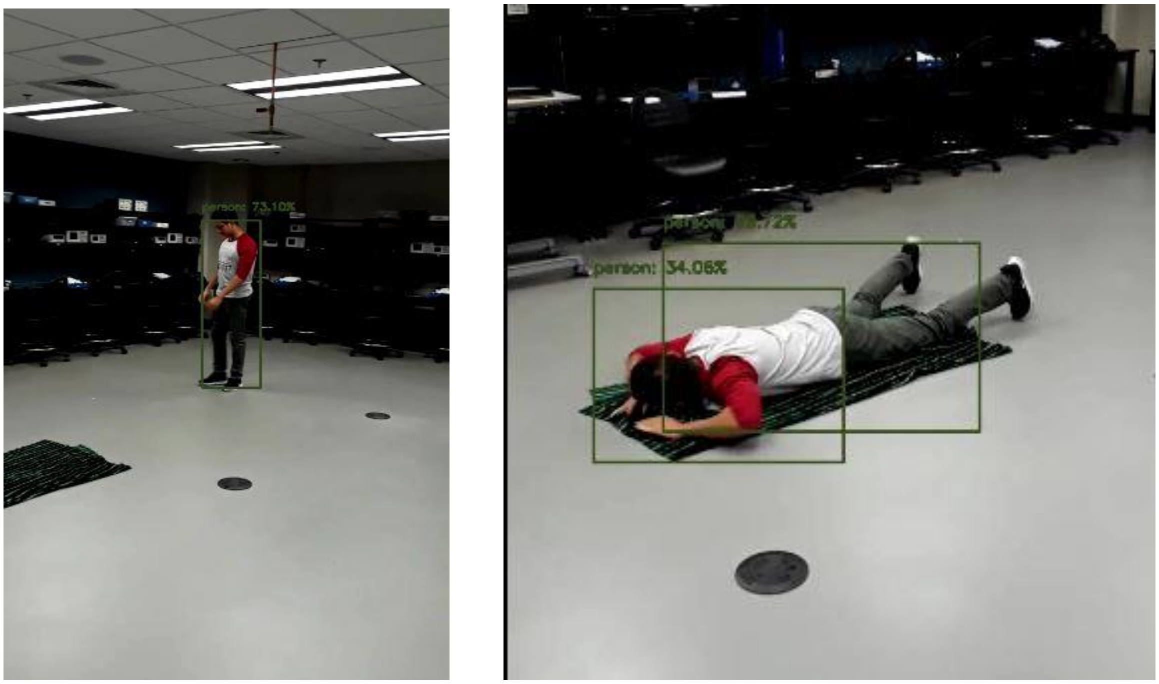

Fig. 15. Person Detections with Confidence (Person 1, Walking & Fall Position 1).

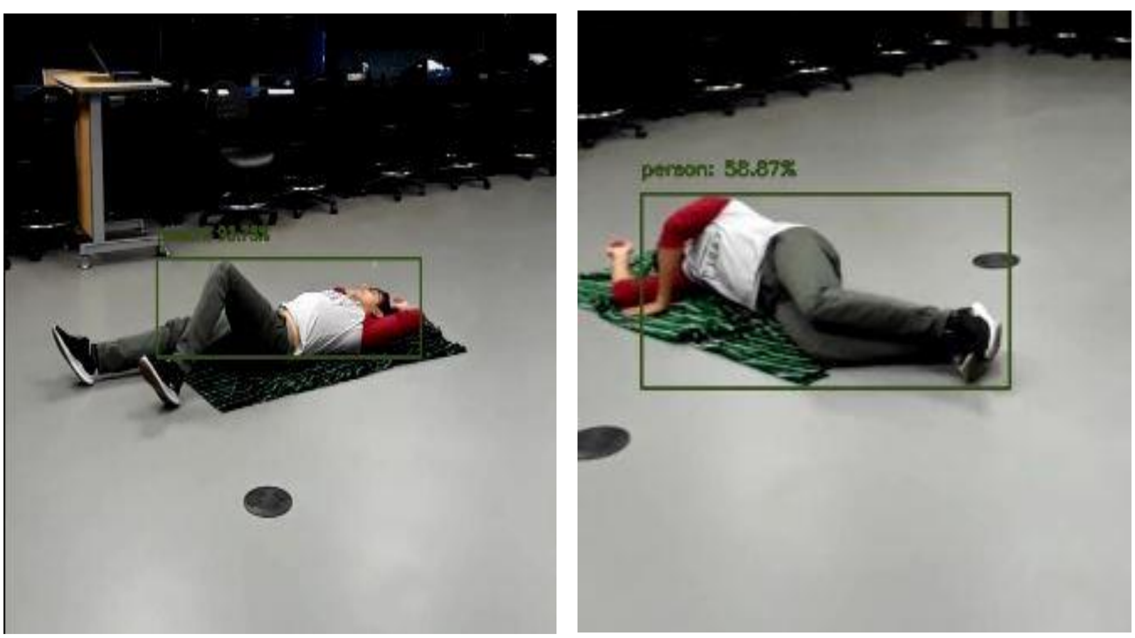

Fig. 16. Person Detections with Confidence (Person 1, Fall Positions 2 & 3).

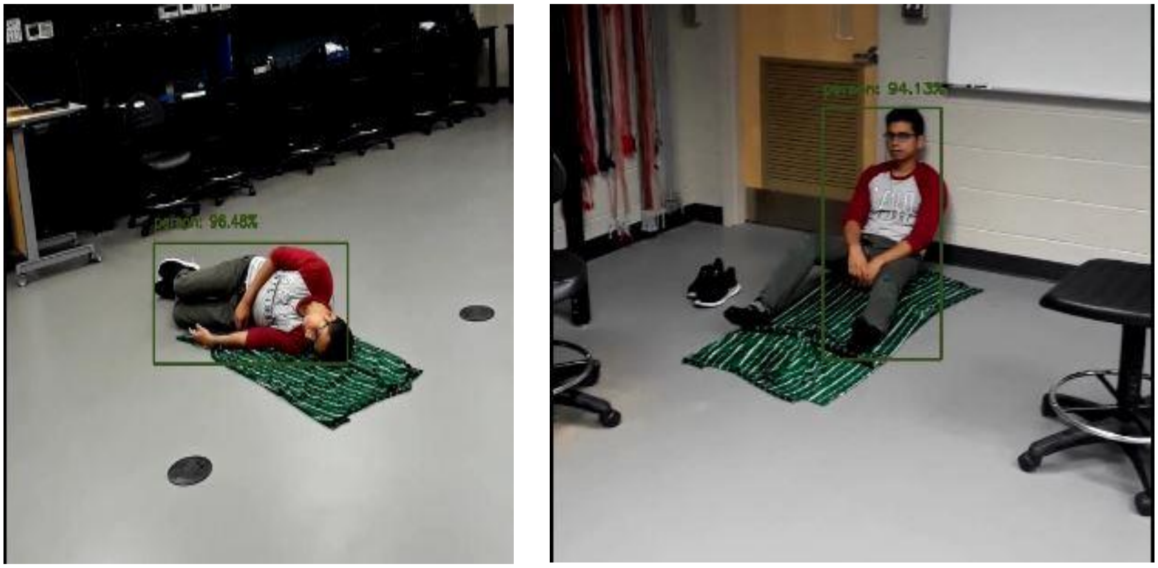

Fig. 17. Person Detections with Confidence (Person 1, Fall Positions 4 & 5).

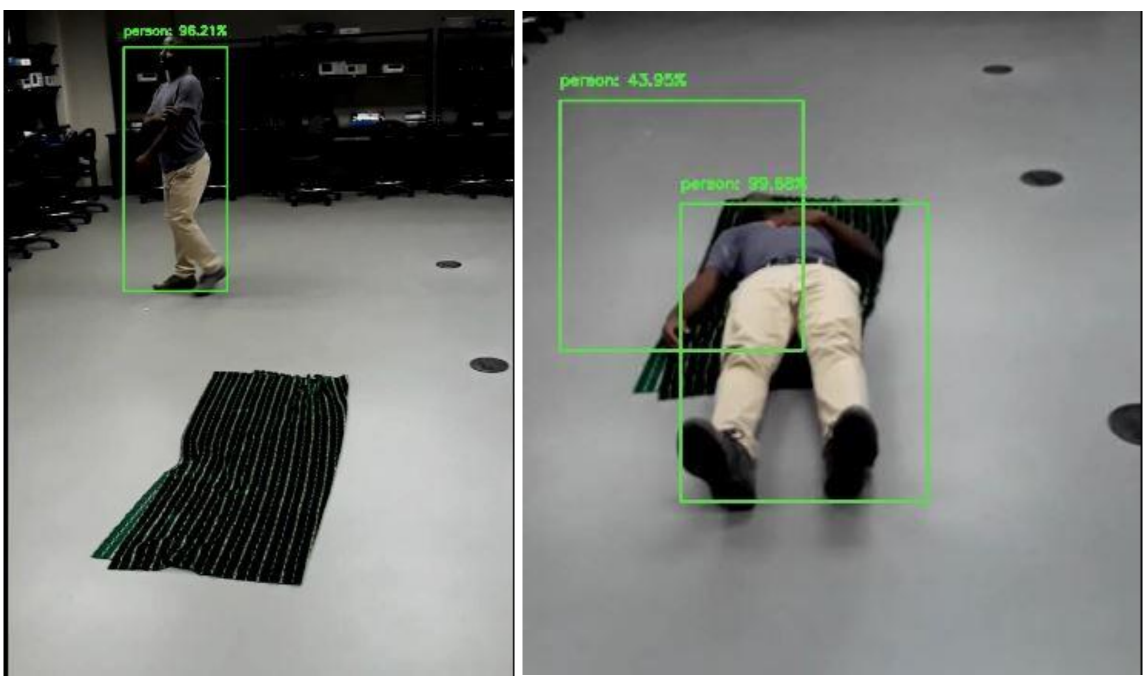

Fig. 18. Person Detections with Confidence (Person 2, Walking & Fall Position 1).

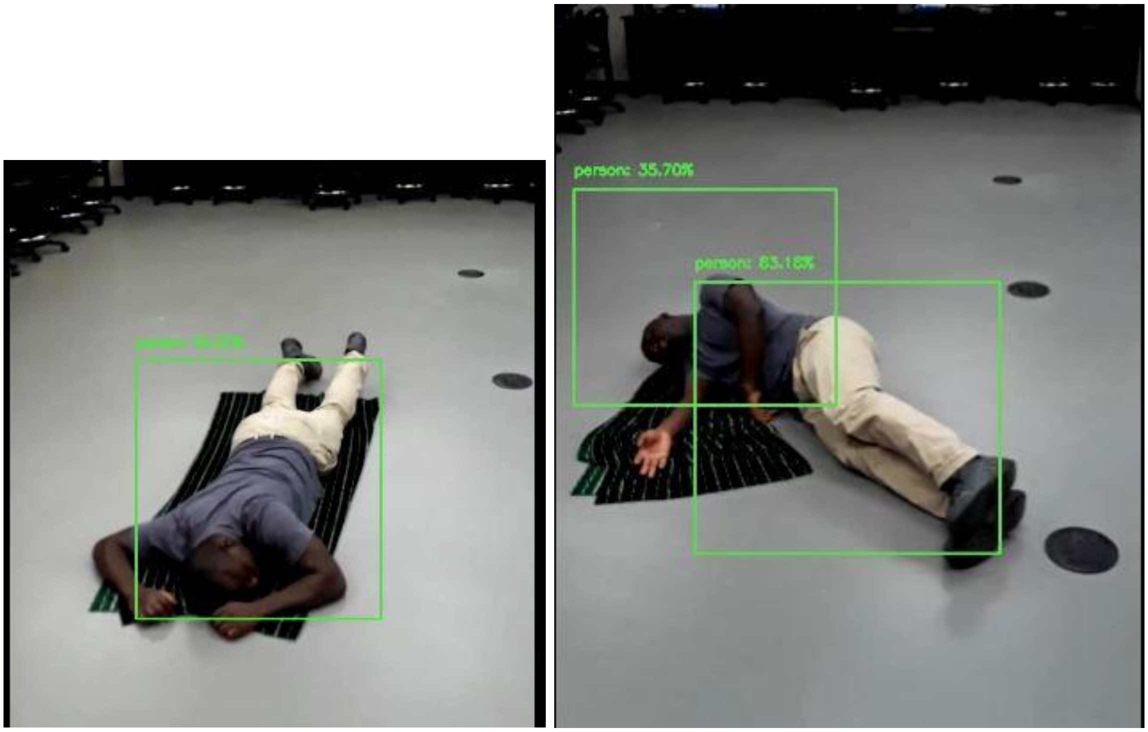

Fig. 19. Person Detections with Confidence (Person 2, Fall Positions 2 & 3).

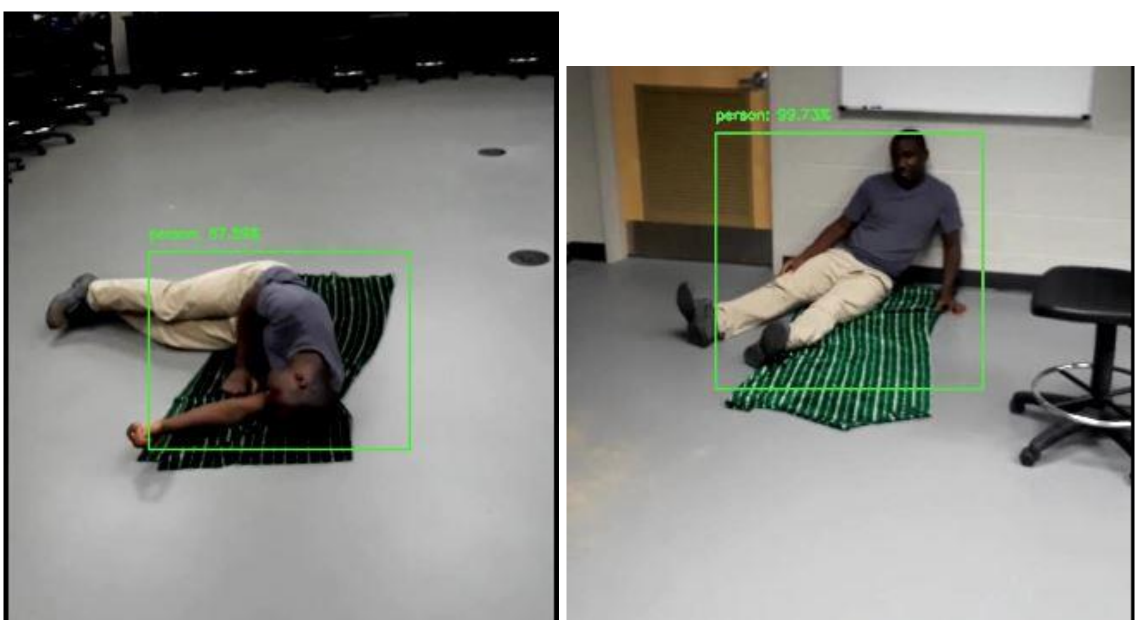

Fig. 20. Person Detections with Confidence (Person 2, Fall Positions 4 & 5).

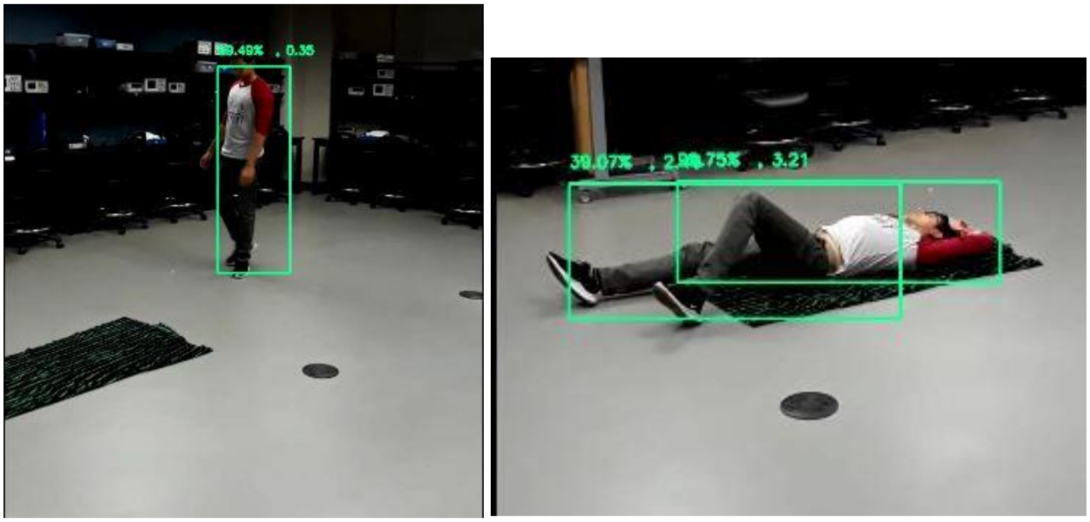

Fig. 21. Moving Ratio = 0.35, Fall Ratios = 2.46, 3.21 (Person 1).

With these results, we added a new function in our Python code that tests for a fall. The fall testing is done in two stages as shown in the pseudocode below. Stage 1 is a preliminary determination and then Stage 2 serves to conclusively prove that a fall has occurred. Y_c represents the y-coordinate of the center of the camera frame and y_c represents y-coordinate of the center of the bounding box which is approximately the center of the person detected.

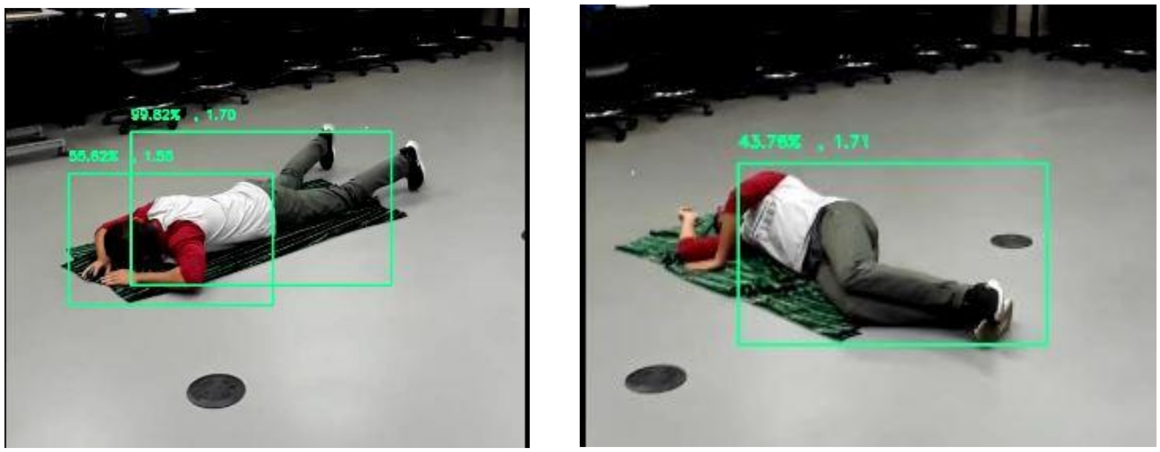

Fig. 22. Fall Ratios = 1.55, 1.70; Fall Ratio = 1.71 (Person 1).

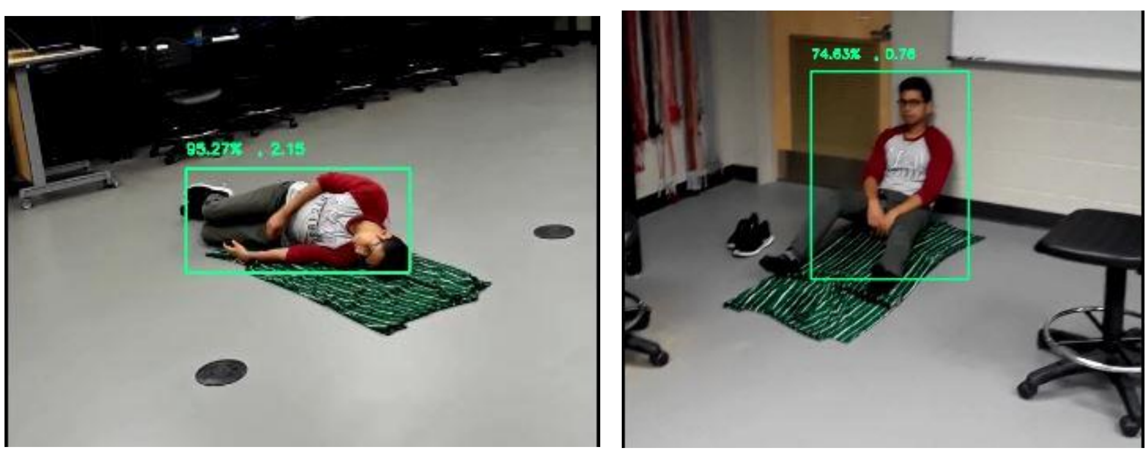

Fig. 23. Fall Ratio = 2.15; Fall Ratio = 0.78 (Person 1).

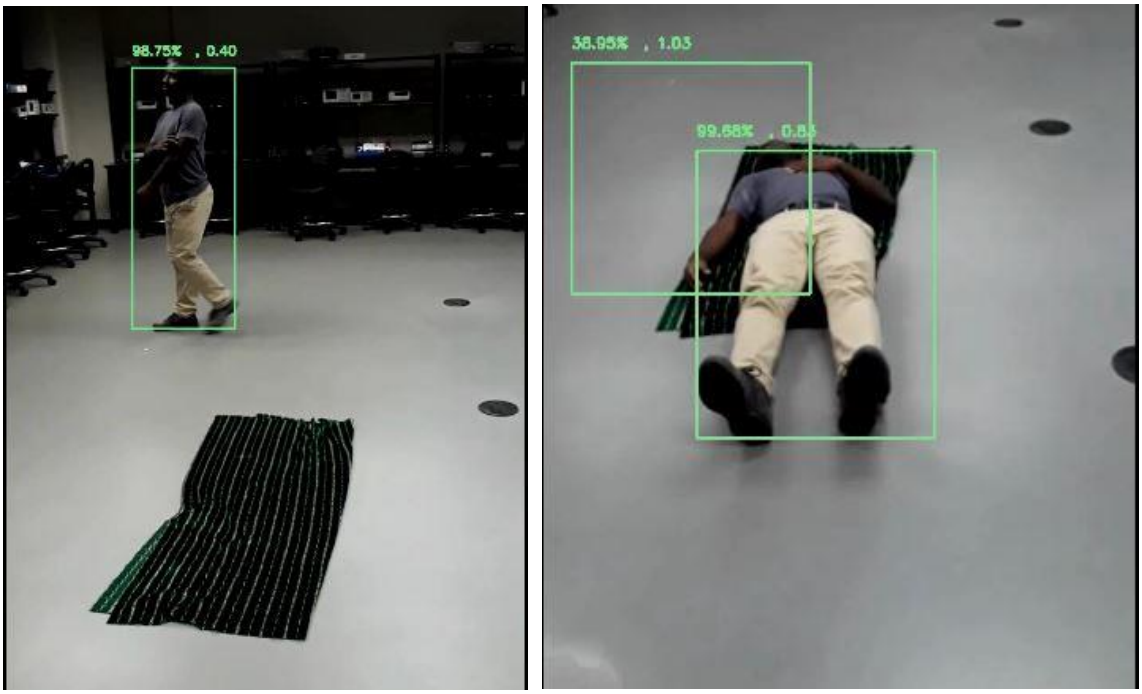

Fig. 24. Moving Ratio = 0.40; Fall Ratio = 0.83, 1.03 (Person 2).

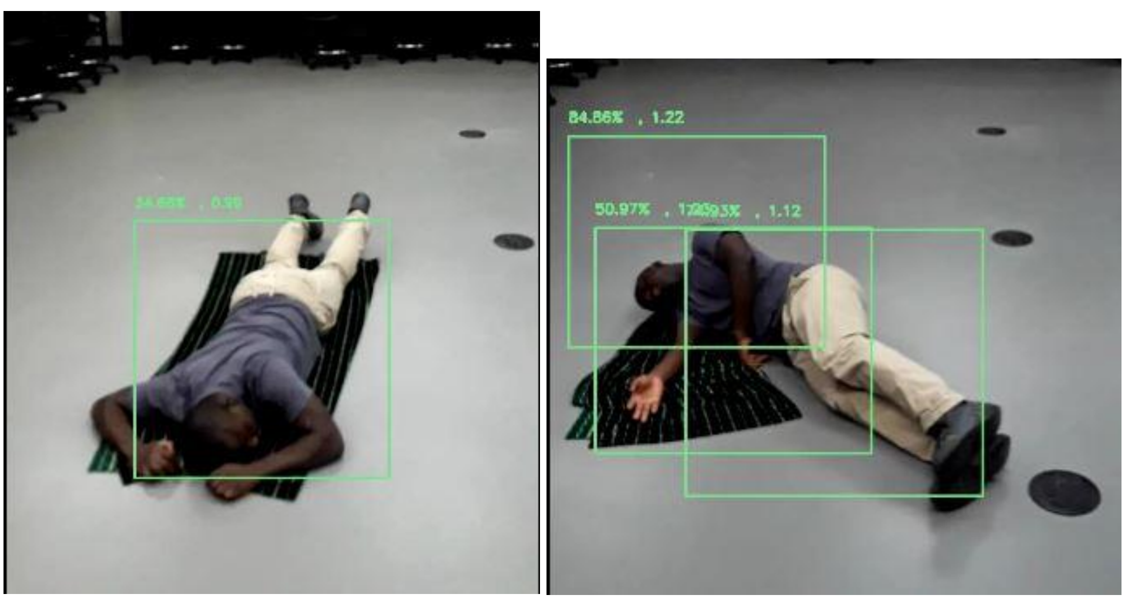

Fig. 25. Fall Ratio = 0.99; Fall Ratios = 1.22, 1.12 (Person 2).

Fig. 26. Fall Ratios = 2.21, 1.51; Fall Ratio = 1.03 (Person 2).

1. Calculate ratio = width of bounding box divided by height of bounding box
2. Calculate y_c
3. If ratio is greater than 0.8, and y_c is greater than ($Y_c + 50$)

   3a. Call Fall Function

   3b. In fall function, repeat detection

   3c. If human is detected, calculate ratio and y_c

   3d. If ratio is greater than 0.8

      3di. If y_c is greater than ($Y_c + 60$)

         Notify that ‘Fall is detected’

      3dii. If y_c is less than or equal to ($Y_c + 60$)

Return to Main Function.

3e. If ratio is less than or equal to 0.8

3ei. Return to Main Function

4. If ratio is less than or equal to 0.8, or y_c is less than or equal to (Y_c + 50)

4a. Do nothing

5. Continue to 1-D Navigation.

F. FADER and the Ping)))™ Sensor

One of the main challenges with using computer vision on FADER is that when moving away from an approaching human, the robot needs a way to be able to detect if it is backing into another object or a wall. This is the use to which we have repurposed the ultrasonic Ping)))™ sensor on the FADER. The PING)))™ sensor is now at the rear of FADER while the Pi Camera is placed in front of it. When FADER is in the retreating state and is backing up, we use the ultrasonic Ping)))™ sensor to ensure that it does not hit the wall for example. If while backing up from a person, FADER gets closer than three feet to a wall or another object, the ultrasonic Ping)))™ sensor detects the wall or object and the FADER is able to turn to the side to avoid hitting the object or wall. Thus, the ultrasonic Ping)))™ sensor prevents the robot from hitting an object or wall while the Pi Camera allows it to maintain its focus on the person detected.

We discuss our initial test results and ongoing results in the next chapter.

CHAPTER IV: RESULTS AND DISCUSSION

A. Initial Evaluation Tests

Having set our initial thresholds to be used for comparing ratio, and the y-coordinates of the center of the detected person, we set about testing in the lab. For our initial tests, we used Person 1 from our earlier annotation sessions. Person 1 took a two minute 50 second walk around the lab and for the last twenty seconds simulated a fall. FADER was able to follow Person 1 around the entire space as well as detect conclusively, by passing through the two stages we set, that a fall had occurred. The notifications “Testing Fall Now” and “Fall Detected” were both printed to the screen of the Raspberry Pi computer. “Testing Fall Now” is printed when the robot passes through the preliminary stage while “Fall Detected” is printed when the Robot conclusively determines that a fall has occurred. Figures 29 and 30 below show the view from both the Rapsberry Pi screen and from a recording in the space. It should be noted that we are only showing the very last frame FADER processes before the fall is detected. While FADER continues to process subsequent frames as seen by the multiple “Fall Detected” notifications.

B. Test with Other Participants

In total we have had six participants come to the lab space and carry out different Activities of Daily Living and simulate falls in different environments and positions. Two of the tests took place a lab space with more obstacles while the rest took place in a lab space with chairs at the side and a few obstacles, mainly at the center of the room.

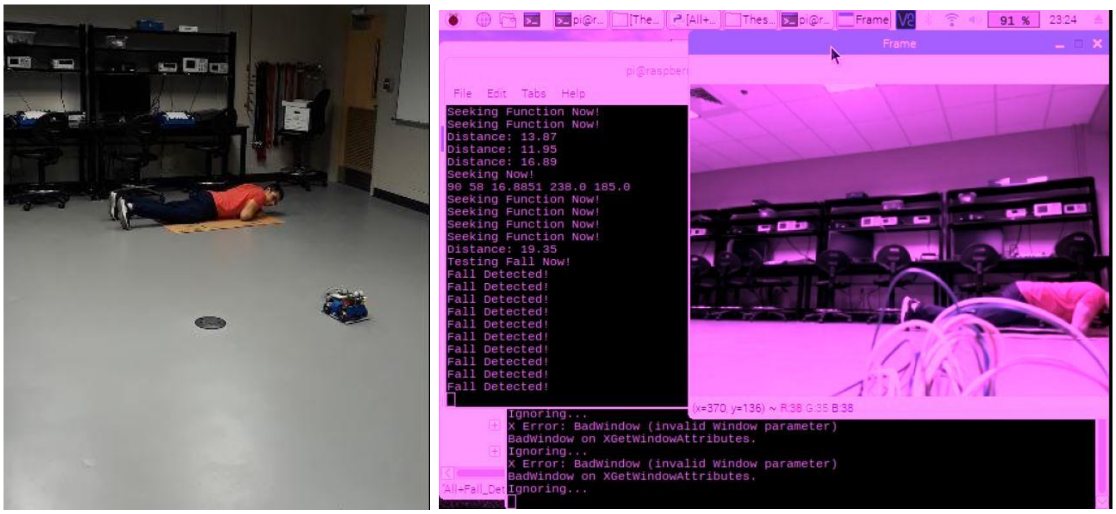

Fig. 27. FADER Detecting a Fall and Computer Screen showing FADER detects the fall activity.

In some cases, FADER conclusively detected a fall passing through the two stages; in other cases, FADER only detected the preliminary fall stage; and in other cases, FADER did not detect the fall. We present the results next.

Participant 1 is a black male with height 5 feet 8.5 Inches. We carried out two tests in a different Lab space which had more obstacles. For test 1, participant walked around, stood and then simulated two falls, lying on his back in both cases. While FADER responded appropriately to the walking and standing, in the case of the two falls, FADER only passed through one preliminary stage of fall detection. There were no conclusive detections. Instead, FADER executed the Seeking Function in both cases. For test 2, the participant walked, stood and then simulated two falls, lying on his side in both cases (he faced FADER in one case). FADER responded appropriately to the walking and standing but there were no detections in either fall cases.

Participant 2 is a white female of height 5 feet 7 inches. In the test session, the participant stood, walked around, approached FADER, then simulated a fall lying on her back. FADER responded appropriately in all the ADL cases but did not detect a fall at all executing the Seeking Function instead. Next the participant crawled around, then stood upside down. In both ADL cases, FADER responded appropriately either approaching or retreating depending on the distance. Next the participant simulated falling in two positions. In the first position, she lay on her back and then in position two she lay on her side facing FADER. While FADER did not detect the participant in the first position, executing the Seeking Function instead, it conclusively detected the fall in the second position passing through both the preliminary and final fall detection stages. Next, Participant 2 stood up again at which point FADER detected that she had stood up and began to follow her as required. She simulated another fall this time lying on her stomach facing FADER. In this case, FADER did not detect her at all and executed the Seeking Function. Participant 2 crawled again with FADER responding appropriately and then simulated another fall this time lying on her side with her back to FADER. Initially FADER did not respond but after she moved forward while maintaining the same position, FADER conclusively detected the fall passing through both stages.

Participant 3 is a black male of height 5 feet 8 inches. In the test session the participant walked and then simulated a fall lying flat on his back. While walking, FADER responded appropriately approaching and seeking as needed. In the fall position, FADER repeatedly passed through the preliminary stage of fall detection without conclusively determining that a fall had occurred. However just as participant 3 was standing up, FADER conclusively determined it was a fall. Next, participant walked and then simulated another fall lying on his side facing FADER. This time FADER responded correctly, and also conclusively detected the fall. Next, Participant 3 walked and simulated a fall lying on his side with his back to FADER. This time, FADER only reached the preliminary stage of Fall Detection. Furthermore, it had drawn too close to the participant.

Participant 4 is a Middle Eastern male of height 5 feet 8 inches. In the test session Participant 4 walked back and forth, moved towards FADER, and stood. In all ADL cases, FADER responded appropriately. Next, the participant simulated a fall by lying on his side with his back to FADER. FADER conclusively detected a fall passing through both stages of detection. When Participant 4 stood up and walked, FADER also detected this and followed as required. Next, Participant 4 simulated a fall lying on his side facing FADER. In this case, FADER conclusively detected a fall passing through both stages of detection. Lastly, Participant 4 Simulated a fall lying on his stomach. In this position, FADER did not detect a fall. Instead it executed the Seeking Function.

Participant 5 is a black female of height 5 feet 3 inches. In the test session, the participant walked, then half-bent and then stood. FADER responding appropriately to the walking but lost the participant when she half bent leading it to execute the Seeking Function. It then reacquired her when she stood up. Next, she fully bent, and FADER recognized her this time and appropriately responded. Participant 5 then went on to execute the following fall simulations: lying down on her side, lying down on her stomach, lying down on her side diagonally, lying down on her side facing FADER. In all of these positions, FADER did not detect the fall, executing the Seeking Function instead, and reacquiring her when she stood up. Next Participant 5 lay down on her side with her back to FADER and then also turned to face it. FADER responding by first executing the Seeking Function moving towards her then passing the preliminary stage of Fall Detection. It remained in this stage even when she turned to face it.

Participant 6 is a black female of height 5 feet 2.8 inches. In this test session, FADER followed participant 6 while she was moving. It briefly lost her when it misidentified a sweater on a chair as a person while executing the Seeking Function. However, it discarded that detection within seven seconds and continued till it reacquired her. Participant 6 first simulated a fall lying on her stomach. FADER executed the Seeking Function and then detected her when she raised her head while still in the fall position. It entered the preliminary stage of fall detection but exited it and then approached her, coming too close and then reentering the preliminary stage of fall detection. In the next instance, while executing the Seeking Function to track Participant 6, FADER misidentified the same sweater as before as a human but once again corrected itself. We also had to abort the next instance where participant 6 lay down on her side facing FADER because

FADER got caught in a chair. We restarted the same fall simulation and this time FADER completely detected a fall. Lastly, Participant 6 simulated a fall lying on her side with her back to FADER. In this case, FADER originally entered the preliminary stage of fall detection before exiting and moving towards the participant.

Tables 5-7 summarize the results of the seven tests sessions we had with the participants.

TABLE 5 PARTICIPANTS AND TEST SESSION BREAKDOWN

	Height	Gender	Skin Color	Location of Session	Time of Total Session
Person 1	5 ft 7 in	Male	Black	DSB 150	6 min 55 sec
Person 1	5 ft 7 in	Male	Black	DSB 150	3 min 32 sec
Person 2	5 ft 7 in	Female	White	DSB 122	7 min 54 sec
Person 3	5 ft 8 in	Male	Black	DSB 122	5 min 57 sec
Person 4	5 ft 8 in	Male	Middle Eastern	DSB 122	4 min 56 sec
Person 5	5 ft 3 in	Female	Black	DSB 122	12 min 24 sec
Person 6	5 ft 2.8 in	Female	Black	DSB 122	16 min 14 sec

These results as summarized in Tables 5 -7 show that FADER responds to 22 out of 23 ADL cases correctly and does not misinterpret any of them as the user falling. The case of the half-bend is the exception to this and even in that case FADER merely does not detect the target as human and so concludes that the human has stepped out of view and executes the Seeking Function.

TABLE 6 DIFFERENT ADLS CARRIED OUT BY PARTICIPANTS AND FADER'S RESPONSE

ADL	Total Number	Responded Appropriately	Did Not Respond Appropriately	Misclassified as a Fall	Comment
Walking (Approaching)	3	3	0	0	Participants often walked or stood in between fall simulations, but we are counting all instances as one.
Walking (Retreating)	7	7	0	0	
Standing	7	7	0	0	
Half-Bend	1	0	1	0	Executed Seeking Function.
Full Bend	2	2	0	0	
Crawling	2	2	0	0	
Standing Upside Down	1	1	0	0	

Out of 24 simulated falls, FADER correctly deduced the first stage of a fall 13 times but then only conclusively determined a fall 7 times. Twelve out of the remaining seventeen times, FADER executed the Seeking Function continuously which means it did not detect a human in the frame before it. The issue therefore is not that FADER did not recognize a fallen human in those cases but that it did not recognize a human at all. Of the remaining five cases, FADER remained in the preliminary detection stage four times; and in only one case does FADER move towards the person instead of conclusively detecting a fall (this case is marked in the “Did Something Else” column in Table 7.

TABLE 7 DIFFERENT FALL TYPES SIMULATED AND FADER’S RESPONSE

		FADER RESPONSE			
FALL TYPE	**TOTAL**	Executed Initial Stage	Completely Detected Fall	Executed Seeking Function	Did Something Else
Lying on stomach	4	1	-	3	-
Lying on back	5	2	1	3	-
Lying on side facing FADER	7	5	4	2	-
Lying on side with back to FADER	5	5	2	-	1
Lying on side	2	-	-	2	-
Lying on side diagonally	1	-	-	1	-

In evaluating real-world fall detection, [37] discusses a number of performance measures based on the possible outcome of true positives, true negatives, false positives and false negatives. The measures included Sensitivity, Specificity, False Positive Rate Over Time, Precision, Negative Predictive Value, Accuracy, F-Measure, Informedness, Markedness, Matthews Correlation Coefficient.

We will be evaluating FADER based on Sensitivity, Precision and the F-Measure together with the definitions and equations presented by [37]. This is because [37] concluded that these three measures are the measures to be focused on when considering real-world Fall Detection. Here is a quote from the paper*:*

> *Sensitivity and precision together quantify the ability to detect falls and avoid false alarms, therefore providing a complete portrayal of performance. In addition to sensitivity and precision it is important to have a single measure which can quantify the trade-off between them. PR AUC is one possible option; however it considers the performance of multiple sub-optimum versions of the system as the system's parameters are adjusted. Since only the optimised system can be deployed, it is the optimised version which should be the focal point of the evaluation. F-measure, the harmonic mean of sensitivity and precision, appears to be the most suitable single measure for objective comparison. This trio of measures has two major advantages in robustness: (1) it does not rely on non-falls and (2) it is resistant to issues surrounding wear time and time in the capture area.*

First, we consider the outcomes as defined by [37]:

True Positive (TP)—Correctly detected fall

True Negative (TN)—Non-fall movement not detected as a fall

False Positive (FP)—Classified as a fall when none occurred

False Negative (FN)—A fall which was not detected

We however note that while [37] considers fall detection as a binary classification, a positive case or a negative case; in the case of our experiments, we give a score of 0.5 to the cases where FADER reached the initial stage of fall detection but did not completely detect the fall.

TP = 7 + (0.5*6) = 10

TN = 23

FP = 0

FN = 11 + (0.5*6) = 14

B1. Sensitivity (also known as recall and true positive rate)**:**

This was defined the proportion of falls which are correctly detected.

Sensitivity = TP /(TP+FN)

= 10/ (10+ 14)

= 0.417

B2. Precision (also known as positive predictive value)

This was defined as the proportion of alarms which are true falls

Precision = TP /(TP+FP)

= 10/ (10+ 0)

= 1

B3. F-Measure (also known as F-Score)

This was defined as the harmonic mean of sensitivity and precision.

[37] further states that the F-measure "*considers all outcomes except true negatives (non-falls). In fall detection, the priorities are detected falls (TP), missed falls (FN) and false alarms (FP). F-measure considers all of these outcomes and therefore provides a good overview of performance*".

F-Measure = 2 * [(Precision * sensitivity)/ (Precision + Sensitivity)]

= 2*[(0.417*1)/ (0.417+1)

= 2 *(0.417/1.417)

= 0.589

C. Improvements of Results

While it has a high precision of 100%, FADER currently has a low sensitivity of 41.7%. This is due to its not detecting humans in some fallen positions. It is however consistently spot on when it conclusively detects a fall.

We propose two possible solutions to improve FADER's sensitivity:

1. Carry out transfer learning to adapt our current deep learning model to detect people in more fall positions as people.

2. Currently the Pi camera is in a fixed position on FADER. Implementing a servo that allows the camera to turn and also move up and down might allow the human to be

detected. As previously stated, [65] found that replacing a fixed sensor with an active one resulted in an 80% improvement. This could substantially reduce the number of false negatives that FADER has.

CHAPTER V: CONCLUSION AND FUTURE WORK

Robots today have the potential to become more ubiquitous as social robots that live in our households and provide care, support and companionship.

We have presented in this thesis preliminary work on the development of a computer vision based human following robot for fall detection in the home of the elderly.

Our Robot, FADER, is Raspberry-Pi and Arduino Based, mobile, can sufficiently follow a person in a mildly occluded space without permanent loss; and can detect falls. While we initially considered other non-vision-based options due to potential concerns about privacy; we eventually decided to have the robot operate as an isolated system, i.e., without connection to the internet. Thus, we are able to address the privacy concern and yet take advantage of the advances in the field of computer vision. We considered two Pi Camera options and after testing chose the Pi NoIR because of its promise for functioning in the dark whilst remaining robust in normal conditions. We implemented deep-learning based object detection on the Raspberry Pi and the Pi Camera. We achieved Distance Estimation, based on Linear Regression using Microsoft Excel. We implemented a threshold-based algorithm based on estimated distance for one-dimensional navigation. We made hardware adjustments to the existing robot prototype. We achieved two-dimensional navigation as a linear combination of tracking, pivoting and one-dimensional navigation. We tested and realized two representative metrics for Fall Detection namely: the ratio of width to the height the bounding box, and the Y-Coordinate of Center of

Person. We implemented Threshold-Based Algorithms using the above metrics to test Fall Detection. We carried out tests with different people and different environments to test our system and have proven that it is possible to use a mobile robot to follow a human in an occluded space and to detect falls.

Our current results show Fall Detection Robot (FADER) has a perfect precision of 100% but a low sensitivity of 42%. Future work is being done to improve this.

Compared to other works, FADER has the following advantages:

- It is mobile/portable
- It is non-participatory
- It is non-invasive with respect to the body
- It is designed as an isolated system. This takes care of non-invasiveness with respect to privacy
- It is low-cost and easily assembled

It is not difficult to envisage a future where FADER is deployed in the home of seniors across the world and functions as a platform for many more functions including offering quick response for when the seniors fall thus extending the length and quality of their lives.

As proposed in the previous chapter, future work would focus first on improving the sensitivity of FADER by either training a new detection model with a person class that detects more people in fallen positions, or by introducing a servo motor that allows the

Pi Camera to be move both vertically and horizontally, thus adjusting the angles at which a tracked person is being viewed. We would also in future work add infrared lighting to FADER and test it a dark environment thus fully maximizing the potential of the Pi NoIR camera.

Moreover, additional work would focus on considering when multiple people and many more household objects are in the space and how that could affect FADER's detection[57]. Possible approaches for detection in such a space include reinforcement learning algorithms.

Also, while the notification of fall detection currently happens on the screen, work would focus on the notification of the external responder either by SMS or by the use of Near Field Communication. These two approaches are preferred because they allow us to preserve the 'no connection to the internet' rule that protects the privacy of FADER's potential users.

REFERENCES

[1] “Leading economics, finance commentator outlines major global trends,” *UWIRE Text*, p. 1, 07-Apr-2016.

[2] C. J. L. Murray *et al.*, “Global, regional, and national disability-adjusted life years (DALYs) for 306 diseases and injuries and healthy life expectancy (HALE) for 188 countries, 1990–2013: quantifying the epidemiological transition,” *The Lancet*, vol. 386, no. 10009, pp. 2145–2191, Nov. 2015.

[3] S. Tokudome, S. Hashimoto, and A. Igata, “Life expectancy and healthy life expectancy of Japan: the fastest graying society in the world,” *BMC Res. Notes*, vol. 9, no. 1, Dec. 2016.

[4] R. J. Hum, S. Verguet, Y.-L. Cheng, A. M. McGahan, and P. Jha, “Are Global and Regional Improvements in Life Expectancy and in Child, Adult and Senior Survival Slowing?,” *PLOS ONE*, vol. 10, no. 5, May 2015.

[5] J. Fan, L. Beuscher, P. Newhouse, L. C. Mion, and N. Sarkar, “A Collaborative Virtual Game to Support Activity and Social Engagement for Older Adults,” in *Universal Access in Human-Computer Interaction. Methods, Technologies, and Users*, 2018, pp. 192–204.

[6] Y. Tian, G.-H. Lee, H. He, C.-Y. Hsu, and D. Katabi, “RF-Based Fall Monitoring Using Convolutional Neural Networks,” *Proc ACM Interact Mob Wearable Ubiquitous Technol*, vol. 2, no. 3, pp. 137:1–137:24, Sep. 2018.

[7] “JMU-Web-Based Telepresence Exercise Program for Community-Dwelling Elderly Women With a High Risk of Falling: Randomized Controlled Trial | Hong | JMIR mHealth and uHealth.” [Online]. Available: https://mhealth.jmir.org/2018/5/e132/. [Accessed: 14-Nov-2018].

[8] “Definition of ROBOT.” [Online]. Available: https://www.merriam-webster.com/dictionary/robot. [Accessed: 27-Oct-2019].

[9] “R.U.R. | play by Čapek,” *Encyclopedia Britannica*. [Online]. Available: https://www.britannica.com/topic/RUR. [Accessed: 27-Oct-2019].

[10] A. LaFrance, “What Is a Robot?,” *The Atlantic*, 22-Mar-2016.

[11] “What Is a Robot?,” *Wired*.

[12] C. Breazeal, “Social robots will become family members in the homes of the future,” *Quartz*. [Online]. Available: https://qz.com/1383665/social-robots-will-become-family-members-in-the-homes-of-the-future/. [Accessed: 27-Oct-2019].

[13] J. M. Kory-Westlund and C. Breazeal, “Exploring the Effects of a Social Robot’s Speech Entrainment and Backstory on Young Children’s Emotion, Rapport, Relationship, and Learning,” *Front. Robot. AI*, vol. 6, 2019.

[14] J. Abdi, A. Al-Hindawi, T. Ng, and M. P. Vizcaychipi, “Scoping review on the use of socially assistive robot technology in elderly care,” *BMJ Open*, vol. 8, no. 2, p. e018815, Feb. 2018.

[15] “Project Overview ‹ Designing social robots for older adults,” *MIT Media Lab*. [Online]. Available: https://www.media.mit.edu/projects/designing-social-robots-for-older-adults/overview/. [Accessed: 27-Oct-2019].

[16] N. Kawarazaki, L. T. Kuwae, and T. Yoshidome, “Development of Human Following Mobile Robot System Using Laser Range Scanner,” *Procedia Comput. Sci.*, vol. 76, pp. 455–460, 2015.

[17] N. Rawashdeh, R. M. Haddad, O. A. Jadallah, and A. E. To’ma, “A person-following robotic cart controlled via a smartphone application: design and evaluation,” 2017, pp. 1–5.

[18] B. X. Chen, R. Sahdev, and J. K. Tsotsos, “Integrating Stereo Vision with a CNN Tracker for a Person-Following Robot,” in *Computer Vision Systems*, vol. 10528, M. Liu, H. Chen, and M. Vincze, Eds. Cham: Springer International Publishing, 2017, pp. 300–313.

[19] B. X. Chen, R. Sahdev, and J. K. Tsotsos, “Person Following Robot Using Selected Online Ada-Boosting with Stereo Camera,” in *2017 14th Conference on Computer and Robot Vision (CRV)*, Edmonton, AB, 2017, pp. 48–55.

[20] C.-W. Chen, S.-P. Tseng, Y.-T. Hsu, and J.-F. Wang, “Design and implementation of human following for separable omnidirectional mobile system of smart home robot,” in *2017 International Conference on Orange Technologies (ICOT)*, Singapore, Singapore, 2017, pp. 210–213.

[21] Mark Tee Kit Tsun, Bee Lau, and Hudyjaya Siswoyo Jo, “An Improved Indoor Robot Human-Following Navigation Model Using Depth Camera, Active IR Marker and Proximity Sensors Fusion,” *Robotics*, vol. 7, no. 1, p. 4, Jan. 2018.

[22] N. O. Eraghi, F. Lopez-Colino, A. de Castro, and J. Garrido, “NafisNav: An indoor navigation algorithm for embedded systems and based on grid maps,” in *2015 IEEE International Conference on Industrial Technology (ICIT)*, Seville, 2015, pp. 345–350.

[23] S. A. Fadzli, S. I. Abdulkadir, M. Makhtar, and A. A. Jamal, “Robotic Indoor Path Planning Using Dijkstra’s Algorithm with Multi-Layer Dictionaries,” in *2015 2nd International Conference on Information Science and Security (ICISS)*, Seoul, Korea (South), 2015, pp. 1–4.

[24] C. Gomez, A. C. Hernandez, J. Crespo, and R. Barber, “A topological navigation system for indoor environments based on perception events,” *Int. J. Adv. Robot. Syst.*, vol. 14, no. 1, p. 172988141667813, Jan. 2017.

[25] A. F. Ambrose, G. Paul, and J. M. Hausdorff, “Risk factors for falls among older adults: A review of the literature,” *Maturitas*, vol. 75, no. 1, pp. 51–61, May 2013.

[26] M. Ahmed, N. Mehmood, A. Nadeem, A. Mehmood, and K. Rizwan, “Fall Detection System for the Elderly Based on the Classification of Shimmer Sensor Prototype Data,” *Healthc. Inform. Res.*, vol. 23, no. 3, pp. 147–158, Jul. 2017.

[27] P. Pierleoni *et al.*, “A Wearable Fall Detector for Elderly People Based on AHRS and Barometric Sensor,” *IEEE Sens. J.*, vol. 16, no. 17, pp. 6733–6744, Sep. 2016.

[28] “Fatal Injury Data | WISQARS | Injury Center | CDC,” 18-Jan-2019. [Online]. Available: https://www.cdc.gov/injury/wisqars/fatal.html. [Accessed: 10-Feb-2019].

[29] “NonFatal Data | WISQARS | Injury Center | CDC,” 18-Jan-2019. [Online]. Available: https://www.cdc.gov/injury/wisqars/nonfatal.html. [Accessed: 10-Feb-2019].

[30] “Costs of Falls Among Older Adults | Home and Recreational Safety | CDC Injury Center,” 21-Sep-2018. [Online]. Available:

https://www.cdc.gov/homeandrecreationalsafety/falls/fallcost.html. [Accessed: 12-Feb-2019].

[31] "Important Facts about Falls | Home and Recreational Safety | CDC Injury Center," 24-Aug-2018. [Online]. Available: https://www.cdc.gov/homeandrecreationalsafety/falls/adultfalls.html. [Accessed: 12-Feb-2019].

[32] D. Houry, C. Florence, G. Baldwin, J. Stevens, and R. McClure, "The CDC Injury Center's Response to the Growing Public Health Problem of Falls Among Older Adults," *Am. J. Lifestyle Med.*, vol. 10, no. 1, pp. 74–77, Jan. 2016.

[33] "ARTICLE: Falls Cost U.S. Hospitals $34 billion in Direct Medical Costs," *Johns Hopkins HealthCare Solutions*. [Online]. Available: https://www.johnshopkinssolutions.com/newsletters/falls-cost-u-s-hospitals-30-billion-in-direct-medical-costs/. [Accessed: 12-Feb-2019].

[34] C. S. Florence, G. Bergen, A. Atherly, E. Burns, J. Stevens, and C. Drake, "Medical Costs of Fatal and Nonfatal Falls in Older Adults," *J. Am. Geriatr. Soc.*, vol. 66, no. 4, pp. 693–698, 2018.

[35] E. Casilari, J. A. Santoyo-Ramón, and J. M. Cano-García, "Analysis of a Smartphone-Based Architecture with Multiple Mobility Sensors for Fall Detection," *PLOS ONE*, vol. 11, no. 12, p. e0168069, Dec. 2016.

[36] P. Pierleoni, A. Belli, L. Palma, M. Pellegrini, L. Pernini, and S. Valenti, "A High Reliability Wearable Device for Elderly Fall Detection," *IEEE Sens. J.*, vol. 15, no. 8, pp. 4544–4553, Aug. 2015.

[37] R. W. Broadley, J. Klenk, S. B. Thies, L. P. J. Kenney, and M. H. Granat, "Methods for the Real-World Evaluation of Fall Detection Technology: A Scoping Review," *Sensors*, vol. 18, no. 7, p. 2060, Jul. 2018.

[38] F. Werner, J. Diermaier, S. Schmid, and P. Panek, "Fall detection with distributed floor-mounted accelerometers: An overview of the development and evaluation of a fall detection system within the project eHome," in *2011 5th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth) and Workshops*, 2011, pp. 354–361.

[39] M. Kepski and B. Kwolek, "Event-driven system for fall detection using body-worn accelerometer and depth sensor," *IET Comput. Vis.*, vol. 12, no. 1, pp. 48–58, 2018.

[40] S. Cheng, "An intelligent fall detection system using triaxial accelerometer integrated by active RFID," in *2014 International Conference on Machine Learning and Cybernetics*, 2014, vol. 2, pp. 517–522.

[41] P. Kostopoulos, T. Nunes, K. Salvi, M. Deriaz, and J. Torrent, "F2D: A fall detection system tested with real data from daily life of elderly people," in *2015 17th International Conference on E-health Networking, Application Services (HealthCom)*, 2015, pp. 397–403.

[42] J. Dai, X. Bai, Z. Yang, Z. Shen, and D. Xuan, "PerFallD: A pervasive fall detection system using mobile phones," in *2010 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, 2010, pp. 292–297.

[43] M. A. Habib, M. S. Mohktar, S. B. Kamaruzzaman, K. S. Lim, T. M. Pin, and F. Ibrahim, “Smartphone-Based Solutions for Fall Detection and Prevention: Challenges and Open Issues,” *Sensors*, vol. 14, no. 4, pp. 7181–7208, Apr. 2014.

[44] S. Madansingh, T. A. Thrasher, C. S. Layne, and B. Lee, “Smartphone based fall detection system,” in *2015 15th International Conference on Control, Automation and Systems (ICCAS)*, 2015, pp. 370–374.

[45] Q. Li, J. A. Stankovic, M. A. Hanson, A. T. Barth, J. Lach, and G. Zhou, “Accurate, Fast Fall Detection Using Gyroscopes and Accelerometer-Derived Posture Information,” in *2009 Sixth International Workshop on Wearable and Implantable Body Sensor Networks*, 2009, pp. 138–143.

[46] M. Vallejo, C. V. Isaza, and J. D. López, “Artificial Neural Networks as an alternative to traditional fall detection methods,” in *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2013, pp. 1648–1651.

[47] L. H. Nair and Ragimol, “AHRS based body orientation estimation for real time fall detection,” in *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*, 2017, pp. 1–4.

[48] Y. He, Y. Li, and C. Yin, “Falling-Incident Detection and Alarm by Smartphone with Multimedia Messaging Service (MMS),” *E-Health Telecommun. Syst. Netw.*, vol. 01, p. 1, Mar. 2012.

[49] Y. Bai, S. Wu, and C. H. Yu, “Recognition of direction of fall by smartphone,” in *2013 26th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, 2013, pp. 1–6.

[50] R. Luque, E. Casilari, M.-J. Morón, and G. Redondo, “Comparison and Characterization of Android-Based Fall Detection Systems,” *Sensors*, vol. 14, no. 10, pp. 18543–18574, Oct. 2014.

[51] M. Sie and S. Lo, “The design of a smartphone-based fall detection system,” in *2015 IEEE 12th International Conference on Networking, Sensing and Control*, 2015, pp. 456–461.

[52] P. Tsinganos and A. Skodras, “A smartphone-based fall detection system for the elderly,” in *Proceedings of the 10th International Symposium on Image and Signal Processing and Analysis*, 2017, pp. 53–58.

[53] Y. Ning, S. Hu, X. Nie, S. Liang, H. Li, and G. Zhao, “Real-time Action Recognition and Fall Detection Based on Smartphone,” in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2018, pp. 4418–4422.

[54] B. Aguiar, T. Rocha, J. Silva, and I. Sousa, “Accelerometer-based fall detection for smartphones,” in *2014 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*, 2014, pp. 1–6.

[55] Y. Chen and Y. Lin, “Indoor RFID gait monitoring system for fall detection,” in *2010 2nd International Symposium on Aware Computing*, 2010, pp. 207–212.

[56] R. L. S. Torres, D. C. Ranasinghe, and and A. P. Sample, “Sensor enabled wearable RFID technology for mitigating the risk of falls near beds,” in *2013 IEEE International Conference on RFID (RFID)*, 2013, pp. 191–198.

[57] Y. S. Delahoz and M. A. Labrador, “Survey on Fall Detection and Fall Prevention Using Wearable and External Sensors,” *Sensors*, vol. 14, no. 10, pp. 19806–19842, Oct. 2014.

[58] M. Uegami, T. Iwamoto, and M. Matsumoto, “A study of detection of trip and fall using Doppler sensor on embedded computer,” in *2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2012, pp. 3263–3268.

[59] Y.-T. Chen, Y.-C. Lin, and W.-H. Fang, “A hybrid human fall detection scheme,” in *2010 IEEE International Conference on Image Processing*, 2010, pp. 3485–3488.

[60] M. Shoaib, R. Dragon, and J. Ostermann, “View-invariant Fall Detection for Elderly in Real Home Environment,” in *2010 Fourth Pacific-Rim Symposium on Image and Video Technology*, 2010, pp. 52–57.

[61] M. Yu, A. Rhuma, S. M. Naqvi, L. Wang, and J. Chambers, “A Posture Recognition-Based Fall Detection System for Monitoring an Elderly Person in a Smart Home Environment,” *IEEE Trans. Inf. Technol. Biomed.*, vol. 16, no. 6, pp. 1274–1286, Nov. 2012.

[62] A. N. Belbachir *et al.*, “CARE: A dynamic stereo vision sensor system for fall detection,” in *2012 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2012, pp. 731–734.

[63] M. Humenberger, S. Schraml, C. Sulzbachner, A. N. Belbachir, Á. Srp, and F. Vajda, “Embedded fall detection with a neural network and bio-inspired stereo vision,” in *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2012, pp. 60–67.

[64] H.-J. Chu and G.-J. Tsai, “INTEGRATING SMARTPHONE AND KINECT FOR FALL DETECTION,” p. 15, 2017.

[65] A. Tomoya, S. Nakayama, A. Hoshina, and M. Sugaya, “A Mobile Robot for Following, Watching and Detecting Falls for Elderly Care,” *Procedia Comput. Sci.*, vol. 112, pp. 1994–2003, Jan. 2017.

[66] “Buy a Camera Module V2 – Raspberry Pi.” [Online]. Available: https://www.raspberrypi.org. [Accessed: 28-Oct-2019].

[67] “Buy a Pi NoIR Camera V2 – Raspberry Pi.” [Online]. Available: https://www.raspberrypi.org. [Accessed: 28-Oct-2019].

[68] “Camera Module - Raspberry Pi Documentation.” [Online]. Available: https://www.raspberrypi.org/documentation/hardware/camera/README.md. [Accessed: 28-Oct-2019].

[69] X. Wu, D. Sahoo, and S. C. H. Hoi, “Recent Advances in Deep Learning for Object Detection,” *ArXiv190803673 Cs*, Aug. 2019.

[70] A. Rosebrock, “A gentle guide to deep learning object detection,” *PyImageSearch*, 14-May-2018. .

[71] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” *ArXiv150601497 Cs*, Jun. 2015.

[72] W. Liu *et al.*, "SSD: Single Shot MultiBox Detector," *ArXiv151202325 Cs*, vol. 9905, pp. 21–37, 2016.

[73] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *ArXiv150602640 Cs*, Jun. 2015.

[74] C. P. Brokate, "Real time motion detection in Raspberry Pi," *Cristian Perez Brokate*, 09-Aug-2019. [Online]. Available: https://cristianpb.github.io/blog/motion-monitor-opencv. [Accessed: 23-Oct-2019].

[75] C. P. Brokate, "Object detection using a Raspberry Pi with Yolo and SSD Mobilenet," *Cristian Perez Brokate*, 06-Mar-2019. [Online]. Available: https://cristianpb.github.io/blog/ssd-yolo. [Accessed: 23-Oct-2019].

[76] B. Blagojevic, "How to (actually) easily detect objects with deep learning," *Medium*, 05-Apr-2018. [Online]. Available: https://medium.com/ml-everything/how-to-actually-easily-detect-objects-with-deep-learning-on-raspberry-pi-4fd40af84fee. [Accessed: 23-Oct-2019].

[77] K. Kaiser, "Building a Poor Man's Deep Learning Camera in Python," *Make Art with Python*, 10-Dec-2017. [Online]. Available: https://www.makeartwithpython.com/blog/poor-mans-deep-learning-camera/. [Accessed: 23-Oct-2019].

[78] "Troubles with YOLO on Raspberry Pi 3 B+ - Raspberry Pi Forums." [Online]. Available: https://www.raspberrypi.org/forums/viewtopic.php?t=219601. [Accessed: 23-Oct-2019].

[79] A. G. Howard *et al.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," *ArXiv170404861 Cs*, Apr. 2017.

[80] chuanqi305, "chuanqi305/MobileNet-SSD," 27-Oct-2019. [Online]. Available: https://github.com/chuanqi305/MobileNet-SSD. [Accessed: 27-Oct-2019].

[81] A. Rosebrock, "Object detection with deep learning and OpenCV," *PyImageSearch*, 11-Sep-2017. .

[82] A. Rosebrock, "jrosebr1/imutils," 25-Oct-2019. [Online]. Available: https://github.com/jrosebr1/imutils. [Accessed: 26-Oct-2019].

[83] "Arduino Uno Rev3." [Online]. Available: https://store.arduino.cc/usa/arduino-uno-rev3. [Accessed: 25-Oct-2019].

[84] "Arduino - FAQ." [Online]. Available: https://www.arduino.cc/en/main/FAQ. [Accessed: 25-Oct-2019].

[85] SparkFun Electronics, *Adventures in Science: How to Use Rotary Encoders*. .

APPENDIX

APPENDIX A: LIST OF PARTS FOR FADER

TABLE 8 LIST OF PARTS FOR FADER

S/N	PARTS	Quantity
1	Rapsberry Pi 3 Model B	1
2	Pi Camera v2	1
3	DC Motors	4
4	LiPo Battery (3 cell 4000 mAh 12 V)	1
5	L298 motor drivers	2
6	Arduino UNO	1
7	Ultrasonic Ping Sensor	1
8	Power Bank (10000mAh)	1
9	Plastic wheels	4
10	Raspberry Camera & Display Extender	1
11	LiPo Battery Voltage Tester	1
12	15 W DC/DC Converter	1
13	3D Printed Frame	-