


1-1-2012

# Sight: a Socially Interactive Gesture-Aware Human-Following Transport System

Andrew D'Souza  
*Ryerson University*

Follow this and additional works at: <http://digitalcommons.ryerson.ca/dissertations>

 Part of the [Computer Engineering Commons](#), and the [Other Mechanical Engineering Commons](#)

---

## Recommended Citation

D'Souza, Andrew, "Sight: a Socially Interactive Gesture-Aware Human-Following Transport System" (2012). *Theses and dissertations*. Paper 721.

This Thesis is brought to you for free and open access by Digital Commons @ Ryerson. It has been accepted for inclusion in Theses and dissertations by an authorized administrator of Digital Commons @ Ryerson. For more information, please contact [bcameron@ryerson.ca](mailto:bcameron@ryerson.ca).

# **SIGHT: A SOCIALLY INTERACTIVE GESTURE-AWARE HUMAN-FOLLOWING TRANSPORT SYSTEM**

By  
Andrew D'Souza  
B.Sc, Ryerson University, June 2009

A thesis  
presented to Ryerson University  
in partial fulfillment of the  
requirements for the degree of  
Master of Science  
in the Program of  
Computer Science

Toronto, Ontario, Canada, 2012  
© Andrew D'Souza 2012

## **Author's Declaration**

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my thesis may be made electronically available to the public.

ANDREW D'SOUZA

## **Abstract**

### **SIGHT: A SOCIALLY INTERACTIVE GESTURE-AWARE HUMAN-FOLLOWING TRANSPORT SYSTEM**

Andrew D'Souza  
MSc, Computer Science, Ryerson University, 2012

Moving a wheelchair is a task which requires both the physical and mental engagement of either an assistant pushing it, or the rider controlling it. Pushing it can be physically demanding and limits the number of wheelchairs that can be moved, to one. Our research introduces a technology that may enable a wheelchair to independently follow the rider or assistant. This will allow the rider to disengage from the task of moving the wheelchair, and allow for more riders to be assisted by an assistant at one time. We have developed a Socially Interactive, Gesture-Aware, Human-Following, Transport system (SIGHT) in order to provide more freedom of action to assistants who are providing the same level of service to riders with less effort. Indeed, SIGHT allows for a human to interact with the wheelchair on a more social level. By using gestures, such as waving, the assistant and wheelchair can communicate.

## Acknowledgements

I would like to thank my parents and sister for all the love and support they have given me not only during the last two years, but for the many years before that while I was an undergraduate. They are the reason why I pursued a graduate degree, and I am truly grateful for their encouragement for me to get it.

To my supervisor, Dr. Alex Ferworn, I owe a great deal of thanks. He accepted me as his student without hesitation or worry, and has been a vital source of encouragement, knowledge, wisdom, and life lessons, that I will carry on for the rest of my life. This thesis would not have been possible without his guidance and help.

To the boys of NCART, especially Jimmy Tran, thank you. Jimmy has been an unfailing foundation of information and help. He has played a major role in helping me complete this thesis, and without his advice, technical knowledge, and friendship, this thesis would not have ended the way it did. Also a big thank you to John Paul Compagnone and Alex Ufkes, who have helped in more ways than one.

Lastly, a big thank you to my girlfriend, Samantha Simons, who has put up with my schooling not only for the past two years, but also while I was an undergraduate. You have been an unwavering source of inspiration and support, for that I am truly grateful.

# Table of Contents

Author's Declaration.....	ii
Abstract .....	iii
Acknowledgements .....	iv
Table of Contents .....	v
List of Tables .....	vii
List of Figures .....	viii
List of Equations .....	x
List of Appendices .....	xii
Chapter 1. Introduction.....	1
1.1 Thesis Statement .....	1
1.2 Motivation .....	1
1.3 Objectives .....	5
1.4 Contributions.....	6
1.5 Thesis Outline .....	7
Chapter 2. Background and Related Work.....	8
2.1 Tracking basics.....	8
2.2 Representing an Object .....	10
2.3 Detecting an Object.....	11
2.4 Tracking an Object .....	12
2.4.1 Point Tracking.....	13
2.4.2 Kernel Tracking .....	15
2.4.3 Silhouette Tracking.....	16
2.5 Social Interaction.....	17
2.6 Applications of Social Robotics .....	19
2.6.1 Elderly.....	20
2.6.2 Disabled and Rehabilitation.....	25
2.6.3 Human Friendships .....	26
2.7 Designing Assistive Robots .....	27
2.8 Assistive Robotic Wheelchairs.....	31
2.9 Summary .....	34
Chapter 3. Methodology .....	35
3.1 Architecture of SIGHT.....	35
3.2 Object Layer .....	35
3.3 Imaging Layer .....	38
3.4 System Layer.....	40
3.4.1 Vision Tracking Algorithm.....	41
3.4.2 PID Motor Control.....	53
3.4.3 Gesturing Engine .....	59
3.5 Physical and Output Layer .....	60
Chapter 4. Experiments and Results.....	61
4.1 ROI Selection .....	61
4.2 Data Logging.....	67
4.3 Static Tracking .....	68
4.4 Dynamic Tracking and Following.....	77

Chapter 5. Conclusion and Future Work .....	86
5.1 Limitations .....	87
5.2 Advantages .....	88
5.3 Future Research.....	89
5.4 Concluding Remarks .....	90
Appendices.....	91
Bibliography .....	95
Nomenclature .....	105

## List of Tables

Table 1 - Types of ROI Selections.....	64
Table 2 - OSC messages used in SIGHT .....	68
Table 3 - Track 1 Testing.....	79
Table 4 - Complete Track testing.....	81



## List of Figures

Figure 1.1 – Overview .....	1
Figure 1.2 - Population of people in wheelchairs over time .....	5
Figure 2.1 - Taxonomy of tracking methods [25] .....	10
Figure 2.2 - Object representations [25] .....	11
Figure 2.3 - Pearl the nursebot [101] .....	20
Figure 2.4 - Ratio of people 65 years old and over to the total .....	21
Figure 2.5 - Population Pyramid for the United States in 1950 [106] .....	22
Figure 2.6 - Population Pyramid for the United States in 1990 [106] .....	22
Figure 2.7 - Population Pyramid for the United States in 2010 [106] .....	22
Figure 2.8 - Estimated Population Pyramid for the United States in 2050 [106] .....	23
Figure 2.9 - Paro, the seal robot [112] .....	24
Figure 2.10 - AIBO, the robotic dog [116] .....	25
Figure 2.11 - Robovie the communication robot [130] .....	27
Figure 2.12 - Kismet, a robot that mimics and responds to human emotions [140] .....	30
Figure 2.13 - A semi-autonomous robotic wheelchair [146] .....	33
Figure 2.14 - MAid, the robotic wheelchair [149] .....	33
Figure 2.15 - SENA, a robotic wheelchair [150] .....	34
Figure 3.1 - Architecture of SIGHT .....	36
Figure 3.2 - An ROI selected on a POI .....	37
Figure 3.3 - The object layer subroutine .....	38
Figure 3.4 - Sony PlayStation 3 Eye [154] .....	39
Figure 3.5 - Microsoft Xbox Kinect [155] .....	39
Figure 3.6 - Two-Tiered High Level Architecture of SIGHT .....	41
Figure 3.7 - An RGB image of Lena .....	44
Figure 3.8 - A Hue-Saturation Histogram of Lena .....	44
Figure 3.9 - Lucas-Kanade Optical Flow Assumptions [87] .....	48
Figure 3.10 - FAST and BRIEF matching a POI .....	53
Figure 3.11 - A graphical representation of a PID controller [171] .....	54
Figure 4.1 – ROI selections: (a) choosing just the head, (b) choosing the top frame, (c) ideal ROI .....	63
Figure 4.2 - A representation of the CPU load percentage vs. time for various numbers of particles .....	66
Figure 4.3 - NCART Lab .....	69
Figure 4.4 - Corners, Depth, and Histogram vs. Time in a stationary position .....	71
Figure 4.5 - View used for re-finding blocked objects .....	73
Figure 4.6 – Re-finding an object: corners, depth, and histograms, vs. time .....	75
Figure 4.7 - Re-finding an object after occlusion of the object occurs .....	76
Figure 4.8 - Track 1 .....	78
Figure 4.9 - Track 2 .....	78
Figure 4.10 - Complete track .....	78
Figure 4.11 - Start to 17m, fast speed .....	83
Figure 4.12 - 17m to start, fast speed .....	83
Figure 4.13 - Start to 17m, medium speed .....	83
Figure 4.14 - 17m to start, medium speed .....	84

Figure 4.15 - Start to 17m, slow speed .....	84
Figure 4.16 - 17m to start, slow speed .....	84
Figure 4.17 - Start to finish, medium speed.....	85
Figure 4.18 - Start to finish - lost at the 32 m mark: depth values.....	85
Figure 4.19 - Start to finish – lost at the 32 m mark: histogram values.....	85
Figure 5.1 - SIGHT, an autonomous people-following gesture aware robotic wheelchair .....	88

## List of Equations

(2.1).....	14
(2.2).....	14
(2.3).....	14
(2.4).....	15
(3.1).....	42
(3.2).....	42
(3.3).....	42
(3.4).....	42
(3.5).....	42
(3.6).....	45
(3.7).....	45
(3.8).....	45
(3.9).....	45
(3.10).....	45
(3.11).....	46
(3.12).....	46
(3.13).....	46
(3.14).....	46
(3.15).....	46
(3.16).....	47
(3.17).....	47
(3.18).....	47
(3.19).....	47
(3.20).....	47
(3.21).....	48
(3.22).....	48
(3.23).....	49
(3.24).....	49
(3.25).....	50
(3.26).....	50
(3.27).....	50
(3.28).....	54
(3.29).....	55
(3.30).....	55
(3.31).....	55
(3.32).....	55
(3.33).....	55
(3.34).....	55
(3.35).....	57
(3.36).....	57
(3.37).....	57
(3.38).....	57
(3.39).....	57
(3.40).....	58

(3.41) .....	58
(3.42) .....	58
(3.43) .....	58
(3.44) .....	58
(3.45) .....	58
(3.46) .....	58
(3.47) .....	58
(4.1) .....	65

## List of Appendices

Appendix A – Particle Filter Flowchart .....	91
Appendix B – Calculate New Position Flowchart .....	92
Appendix C – Lost Object Flowchart .....	93
Appendix D – Complete SIGHT Flowchart .....	94

# Chapter 1. Introduction

## 1.1 Thesis Statement

The goal of this thesis is to demonstrate that it is possible to form an artificial cooperative relationship between a sufficiently equipped “intelligent” wheelchair, and an external human “assistant” attempting to move the chair out. The relationship is based on the chair’s ability to use sensed data and develop appropriate algorithmic responses implementing the desires of the human assistant to have the wheelchair follow at a distance that allows for interaction with a human “rider” being transported in the chair.

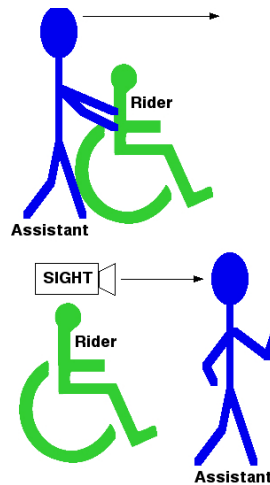


Figure 1.1 – Overview

## 1.2 Motivation

In today’s world, robots can be found across a wide swath of domains. They can be autonomous or remotely controlled. They exist in factories and manufacturing plants, in people’s homes, schools, workplaces, and even in space. Robots can look after jobs that may be too dangerous (or even too laborious) for humans to do. Common examples are the use of robots for big jobs such as bomb disposal and weapon delivery [1], heavy transportation [2], and mass production

[3], to smaller jobs such as vacuuming [4], floor washing [5], gutter cleaning [6], or simply to entertain [7].

These various types of robots can fall into one of three main categories: industrial robotics, professional service robotics, and personal service robotics [8] [9]. Personal service robots are those that perform some task for people directly, within social contexts, through physical means, with the main goal of rendering assistance. These task-based robots, in close proximity to humans, are sometimes called *socially assistive robots (SAR)*. Common examples of SAR are robotic arm manipulators, walker-aides, and robotic shopping carts. Closely related to SAR, are *socially interactive robots (SIR)*. These are “human-friendly” robots that are able to coexist with humans and support them effectively through direct interaction with them [10]. Examples of SIR are rehabilitation (physical and/or mental) robots, friendship/companion robots, and butler (serving) robots.

The social relationships that humans have with SAR and SIR are deliberately intended to be similar to the social relationships humans have with each other. It is social in the sense that there exists natural languages, gestures, mimicry, situation understanding, physical interactions, and coordination between the robot and it’s (human) user [11]. Social interaction, in a general sense, is the way people communicate with each other within the context of a social setting – including overt and implied rules for mutual engagement. It can be defined as actions taken by an individual that relate to the behavior of others, and therefore dictate the individual’s further reactions and directions [12].

Social interaction can be decomposed into three main categories – social presence, social awareness, and connectedness. Social presence itself can be defined by the relation of immediacy [13] and intimacy [14], where immediacy is a measure of psychological distance, e.g., smiling, nodding, and intimacy is the interpretation of interpersonal interactions. Immediacy behaviors are used to create and maintain intimacy, while enhancing social presence [15]. Social awareness is complimentary to social presence, and can be defined as an “understanding of the activities of others, which in turn provides a context for your own activity” [16]. A third important aspect of social interaction is connectedness. This can be defined as the lingering after-effects of social presence; specifically as it is an “emotional experience, evoked by, but independent of, the other’s presence” [15].

It is the combination of all three of these interaction pieces that define companionship: immediacy and intimacy, awareness, and connectedness. We believe these types of social interactions can be found between a robot and a human. For robots to be socially interactive, Breazeal [17] explains that their “behavior and actions must fully meet a person’s social model for it, whilst remaining active in the entirety of the humans environment”. She expands on this by giving four distinct modes of interactions, between robots and humans, which define the social intelligence of a robot: These interactions closely follow the criteria of social interactions discussed above. They require that robots be: socially evocative, socially communicative, socially responsive, and sociable. Socially evocative is a class that is effectively meant for the human side of the relationship – it encourages humans to anthropomorphize (personify) the robot. This means that this mode of social interaction inspires humans to believe that a robot can possess human-like qualities, i.e. react to emotions. Social communication is an interaction type



that says that robots should make contact and communication with a human more natural and familiar. Interaction isn't necessarily a one-way street from human to robot – the socially receptive class says that robots should also benefit from interactions with people, i.e. learning to physically move itself based on movements of a human. The last class, the sociable class, encourages robots to engage and interact with humans in a social manner to not only benefit the person, but also to benefit itself, e.g. through learning.

A robot that is able to follow a person is an important requirement for a socially interactive robotic companion [18]. Not only would such a robot be a physical presence to a person, but it would also act as an emotional presence as well.

There are many possible applications where a robot being able to follow a human would be advantageous. One such application might involve robots that carry heavy equipment for the people they are following, like soldiers, construction workers, and luggage handlers. As there is a specific way that humans interact with each other, robots would necessarily have to fit into these social contexts in an appropriate way. Another major application for people following robots occurs in medical and health care environments. Robotic health monitors, or even beds, might be able to follow a healthcare practitioner from one location to the other, saving valuable time in finding the people to move the equipment around manually. This also saves health care costs as a surprising number of health care workers become injured when trying to move patients and equipment around [19]. Arguably, one of the most critical applications for people following robots may involve wheelchairs. In a 1994/95 U.S. census, 1.8 million people were found to use wheelchairs [20]. In 2002, this number grew to 2.7 million [21]. A year later, in 2003, the

number increased yet again to 3.3 million [22]. Figure 1.2 shows a graphical representation of this. Clearly the number of people dependent on wheelchairs today has grown substantially. In fact, the percentage of senior citizens in the population has increased dramatically because the first segment of the “baby boomer” generation are reaching the age of 65 and are becoming dependent on assistive technology such as wheelchairs [23]. The U.S. Administration of Aging predicts by the year 2050, 25% of the U.S. population will be 65 years or older [24]. With the advance of age comes the necessity for assistive tools; the dependency on wheelchairs will be higher than ever. Adding a people following feature, (attached with a sense of social awareness), to these wheelchairs may greatly and positively impact the lives that depend on the wheelchairs.

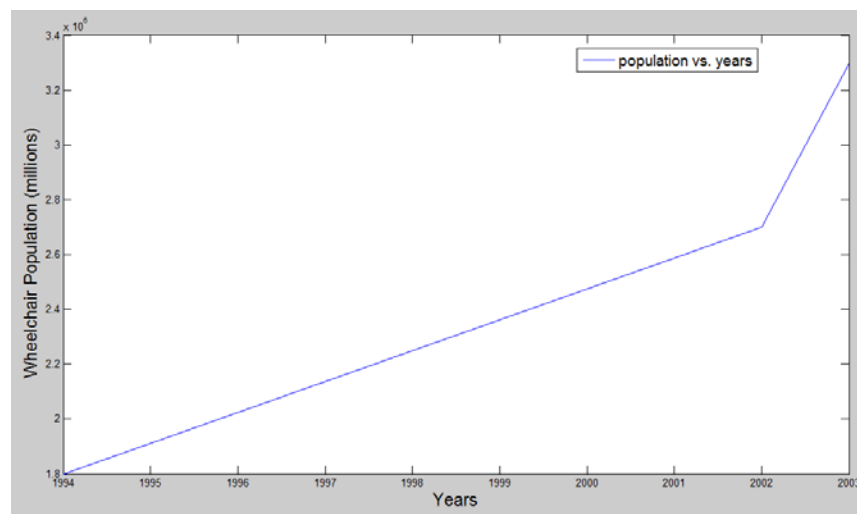


Figure 1.2 - Population of people in wheelchairs over time

### 1.3 Objectives

The purposes of this thesis are twofold. The first objective is intended to demonstrate feasibility. We intend to demonstrate that a typical electric wheelchair can be modified to follow a *person of interest (POI)* through a human-occupied environment. An important aspect of this thesis is to show that using *common off the shelf (COTS)* parts, a cheap vision tracking system could be implemented on any electric wheelchair. Our second objective demonstrates social practicality for such a system. It will be demonstrated that the wheelchair could have a rudimentary form of

social awareness – the chair will be able to interact with the assistant via gestures. The wheelchair (from this point on, *wheelchair*, *robot*, and *robotic wheelchair* are used interchangeably) will employ a vision system in order to detect where the POI is, relative to itself, and move correspondingly in reaction to the POI’s movements.

A method for visually tracking the POI will be examined and explained. An algorithm will then be presented and tested to demonstrate that the robot is socially aware. The various modes and terms of social interaction will be shown and applied to the system. In particular, a gesture engine will be used to interact with the wheelchair.

## **1.4 Contributions**

Several works have been documented in the field of robotics, social awareness, and people following (more detail to follow in chapter 2). The research in this thesis aims to combine various tracking (vision) methods with social rules. Most of the previous works done with people following robots satisfy only the simple ‘following’ goal – to follow an *object of interest (OOI)*, or POI. This thesis contributes by associating social aspects to the ‘following’ functionality. This thesis also targets a more unique way of visually tracking the POI than most of the previous works do. Our vision system can be run in real time, which is a necessity for use in the real world.

Through social integration, our research may help provide users a form of companionship, connectedness, maneuverability, and freedom that they may not have seen in a long time. This work, although targeted towards wheelchairs, can also be extended to other types of

transportation (like the ones already mentioned) without many modifications. Our system is a Social Interactive Gesture-Aware Human-Following Transport (SIGHT).

## **1.5 Thesis Outline**

This chapter serves as an introduction for this thesis. It presents to the reader the motivations, objectives, and contributions of our work. The following chapter will give an overview of existing practices, techniques, solutions, and results that address the objectives of this thesis. In particular, chapter 2 will give the reader the background information necessary to understand our work. It will discuss common vision techniques in detail so the reader can grasp later ideas. It will also show the reader examples of robots who currently exhibit some form of social awareness.

Chapter 3 describes in detail the methodologies used in creating a wheelchair to make it follow a POI and become socially aware via gestures. Algorithms that visually track the POI will be explained. The method of applying a social sense to the robot will also be discussed.

Chapter 4 will outline the experiments that were conducted in order to prove the effectiveness of our work. Any criteria or prerequisites used will be disclosed here. Results that were obtained through the experiments will be presented. Any strengths or weakness' found will be described in this chapter.

Finally chapter 5 will present to the reader a summary of the work done, the results achieved, and the research that is left for the future.

## **Chapter 2. Background and Related Work**

In this chapter we present the relevant background research related to vision, vision tracking, and social awareness. This existing body of knowledge provides the foundation for our work.

### **2.1 Tracking basics**

Tracking is defined [25] by the ability to estimate the trajectory of an object in an image scene as it moves around an environment. In essence, it is the process of updating an objects position, motion, shape, and appearance over a period of time through space. There are many artificial mechanisms that can be used to track objects, including sonar [26], radar [27], and infrared [28]. However, the case for employing some form of machine vision to track remains compelling. Most other techniques provide only limited information about the target – usually limited to positional information. Visual tracking holds the promise that it may be possible to detect variations in the target which can be used to enhance the ability to track and provide additional functionality within the tracking system. There are many reasons why objects should be tracked using computer vision. In the realm of situated and mobile robotics interacting with humans, machine vision may be the only practical means of detecting subtle signals given by humans during their interaction with other humans and the rest of the world. For example, it is unlikely that a human head nodding in agreement will ever be detected by anything other than through visual observation. Tracking itself is necessary in the human world because humans move. Keeping track of this movement is a necessary prerequisite for almost all other interactions – if you do not know where a human is, you cannot interact with them.

Sighted humans are natural visual trackers. For example take a baby, who only at a few months old, is able to gaze into a human's face, and follow that human's face as he or she moves around. Reproducing and describing a natural process is often a matter of "downhill design and uphill analysis" [29], where we can observe a phenomenon like tracking easily but not be able to explain what underlying mechanisms allow it to happen. Inevitably, artificial visual tracking will remain an analog of the natural process but we can describe the problems in terms of what we can observe. Some of the factors that can arise are related to noise in the image, fast object motion, smooth contours, occlusions, dynamic lighting, and general clutter [25] [30]. A primary step in tracking an object is to determine what kind of object it is. The object can be physically represented by points, primitive geometric shapes, silhouettes and contours, articulated shape models, and skeletal models [25]. After determining an object type, it is important to determine the kinds of features this object has. Typical features found in most kind of objects are color, edges, optical flow, and texture. Methods for tracking an object can change if the object is in a static environment, (which means that only the object is moving, while the environment stays the same), versus if the object is in a dynamic environment, (which means that both the object and the environment can change). Figure 2.1 shows the swath of physical and feature representations of the object mentioned above.

Point tracking [31] - [32], kernel tracking [33] - [34], silhouette tracking using contours [35] - [36], and silhouette tracking using matching shapes [37] - [38], are the three major tracking methods used in computer vision today. This thesis uses a combination of all three types (point, kernel, and silhouette) to form a more robust (less failure-rates) tracking method.

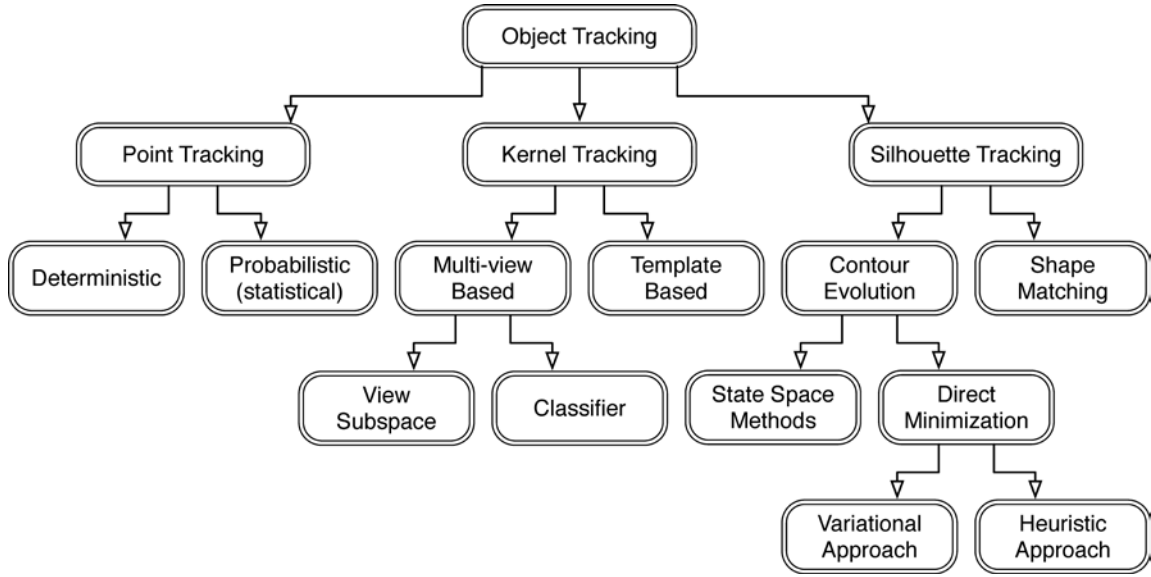
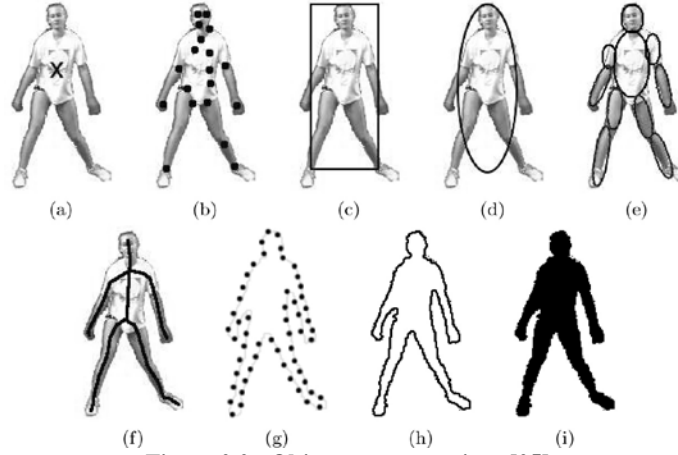


Figure 2.1 - Taxonomy of tracking methods [25]

## 2.2 Representing an Object

One of the easiest ways to describe an object would be to describe its physical characteristics. For example, one can easily identify a human's hand by noticing its shapes, curves, and contours. The authors from [25] describes representation of an object in more detail. They break representation into nine categories, as shown in Figure 2.2. Figure 2.2(a) *Points*: shows the object represented by a centroid point [39], or Figure 2.2(b) a set of points [40]. This type of representation is useful for tracking small regions in an image. Figure 2.2(c) and Figure 2.2(d) *Primitive Geometric Shapes*: shows object shapes which are represented by a rectangle or ellipse [33]. In general, this type is good for tracking simple rigid objects; however it can still be used for tracking non rigid objects. Figure 2.2(e) *Articulated shape models*: follows the primitive geometric shapes in the fact that many shapes are connected together by joints. Figure 2.2(f) *Skeletal models*: this model describes the skeleton of an object by “applying medial axis transform to the object silhouette” [41], and is commonly used to track connected and rigid

objects [42]. Figure 2.2(g) and Figure 2.2(h) *Object silhouette and contour*: shows the contour or the boundary of an object. The region inside the contour is called the silhouette of the object Figure 2.2(i). Silhouette and contour object representation are good for tracking complex non rigid shapes [28] [43].



**Figure 2.2 - Object representations [25]**

Besides an object's physical characteristics, there are several other ways of representing the appearance features of objects [25]. Two in particular are *Probability densities*, and *Templates*. Probability densities can be Gaussian [44], or a mixture of Gaussian [45], or even histograms [33]. Templates [46] use geometric shapes or silhouettes in a single view – they are good for tracking objects with static poses. Selecting unique features of the object itself is a critical aspect of tracking an object. There are four main visual features that are commonly used for tracking [25]: *color*, *edges*, *optical flow*, and *texture* [33] [47] [48].

### 2.3 Detecting an Object

Being able to detect the object regardless of whether it is the first frame or the  $n^{th}$  frame, is a necessity for any tracking method. A general approach to detecting an object is to use frame



differencing – seeing what has changed in consecutive frames. [25] discusses several object detection method types: *Point Detectors*, *Background Subtraction*, *Segmentation*, and *Supervised Learning*. Point detectors are used to find interest points in images. Interest points are features contained in an image that are detectable regardless of the image’s scale, noise, or illumination. They can be locations in an image where the signal changes two dimensionally, such as corners [49]. Commonly used interest point detectors include Moravec’s interest operator [50], Harris interest point detector [51], KLT detector [52], SIFT detector [53], and FAST detector [54] [55]. [56] compares some of these detector methods in more detail. Background subtraction is used to find any significant changes in two consecutive frames. The area where the change occurred is further processed. Work done by [57] [58] [59] [60] [61] explain more about background subtraction. The segmentation method partitions the image into “perceptually similar regions” [25]. Some segmentation techniques used for tracking include mean-shift clustering [33] [62], graph-cuts segmentation [63] [64], and active contours [45] [65] [66]. Object detection using supervised learning works by learning different object views automatically from a set of examples by means of a supervised learning mechanism [25]. Learning methods include neural networks [67], adaptive boosting [68], decision trees [69], and support vector machines [70].

## **2.4 Tracking an Object**

As mentioned above, the three main categories of tracking an object are *Point*, *Kernel*, and *Silhouette* tracking. This section will go over what has been accomplished in these categories thus far.

### 2.4.1 Point Tracking

Detecting the same point of an image across multiple frames falls into two categories: *deterministic* and *statistical* [25]. The authors from [39] explain that deterministic methods use “qualitative motion heuristics”, which means that certain constraints are applied to the tracking in order to form an optimal tracking set. These constraints include *proximity*, which assumes the location of the object will not change notably from frame to frame, *maximum velocity*, which defines an upper bound on the object velocity, *small velocity change*, assumes a smooth motion of the object to be tracked, *common motion* which limits the velocity of objects near the object to be similar to the velocity of the object to be tracked, *rigidity*, which assumes that objects in the 3D world are rigid, therefore the distance between any two points on the actual object will remain unchanged, and *proximal uniformity*, which is the combination of proximity and the small velocity change [25].

The work done in [71] proposes an iterative algorithm that uses the small velocity change and rigidity constraints to track a point across multiple frames. Their method cannot handle occlusions, entries, or exits. The authors from [72] use the proximity constraint to track an object. They find the initial correspondence by computing optical flow in the first two frames. Their method does not handle entries or exits of objects, and if the number of detected points decreases, chance of occlusion or misdetection is increased. In [73], the authors modify the work done by [72] to handle entries and exits by using background subtraction. The work done by [39] introduces a new constraint, called the *common motion* constraint, which handles occlusion and midsection, but it does not allow for entries or exits. This constraint is only suitable for the tracking of points that lie on the same object.

The second category of point tracking is the statistical method. This method assumes an object's path is not necessarily linear, therefore it expects uncertainties and noise across frames. It uses the state space approach to model properties of the object such as position, velocity, and acceleration [25]. These properties are defined by a sequence of states, as in Equation (2.1), where  $W^t: t = 1, 2, \dots$  is white noise.

$$X^t = f^t(X^{t-1}) + W^t \quad (2.1)$$

The relationship between the measurement properties and the state is specified by Equation (2.2), where  $N^t$  is the white noise and is independent of  $W^t$ .

$$Z^t = h^t(X^t, N^t) \quad (2.2)$$

In order to track an object, the state  $X^t$  needs to be estimated given all the measurements up to that moment, or the Probability Density Function (*PDF*) needs to be constructed (Equation (2.3)).

$$p(X^t | Z^{1, \dots, t}) \quad (2.3)$$

In scenarios where there is a single object to be tracked, and the noise involved has a Gaussian distribution, then *Kalman Filters* are used. This filter is made up of two steps, prediction and correction. The authors of [74] use Kalman filters for tracking an object across a sequence of noisy images. The work done in [75] uses this filter for predicting the object's position and speed. For states that do not follow a Gaussian distribution, it is better to use *Particle Filters* [76]. The PDF at time  $t$  is represented by a set of samples (or particles) as in Equation (2.4), where the weights define the importance of a sample [35].

$$\{s_t^{(n)}: n = 1, \dots, N\} \quad (2.4)$$

Particle filters are comprised of three steps: selection, prediction, and correction [25]. Its algorithm will be further explained in Chapter 3. The authors of [77] use particle filters along with Haar-like classifiers to track an object. They are able to track an object well through enter and exit states, as well as occlusions, and illumination changes. They however rely on finding Haar-like features on an object before tracking is done, and further rely on using Gentle AdaBoost, a machine learning algorithm. In [78], the authors combine color-based image features with a particle filter. The problem with their tracking method is if an object's neighbor has a similar color, the tracking of the original object could be skewed. The work done in [79] uses optical flow with particle filter to track an object.

### 2.4.2 Kernel Tracking

Being able to compute the motion of the object from frame to frame is known as kernel tracking. *Template matching* is useful to track single objects using a description of the region of the object in an image, such as image intensity or color features. It is a brute force method of searching the image for a region similar to a template of the object, which was found in the previous frame. The new position of the object is calculated by a simple cross correlation equation. The problem with template matching is that it is computationally expensive, as it performs a brute force search across each frame. In [46], the authors find the mean color of pixels in a rectangular region, and use this as a template for matching. They reduce the computing requirements by only searching for the object in eight neighboring locations. The position which has the highest similarity to the mean color is deemed the correct location of the object. The work done in [62] uses a weighted

histogram and a mean-shift tracker method, whilst [80] extends on this by using a joint spatial-color histogram instead of just a color histogram. Using a mean shift tracker takes away the computation power necessary for a brute force search, however it requires that a portion of the object has to be inside the region of interest upon initialization [25]. One of the more popular tracking methods that fall under the kernel tracking category, is the use of *optical flow*. It works by generating dense flow fields by computing the flow vector of each pixel under the brightness constancy constraint [81]. The KLT Tracker [82] [83] tracks an object very well, however motion of an object is necessary. If the motion stops or slows down enough, the object will less likely be tracked. [84] makes use of optical flow combined with rigidity constraints to track vehicles. In [34], the authors are able to track objects using a layering method. They use the background as one layer and all other objects as a different layer. Based on the objects previous motion and shape, tracking is computed.

### 2.4.3 Silhouette Tracking

Silhouette tracking is useful for tracking objects that do not necessarily have primitive shapes. To track an object, previous frames are used to generate an object model, and based on that model an object region is found in the current frame. Silhouette tracking can fall under two categories: *shape matching*, and *contour tracking* [25]. Shape matching looks for a match of the object silhouette in the current frame to the initial frame. Only translation (movement in a constant distance in one direction) of the object is assumed to take place in this method. The main difference between point matching and silhouette matching is the object representations and the object models used – silhouette matching makes use of an objects appearance features, whereas point matching uses only motion and position-based features [25]. Normally edge-

maps, or edge-based representations, of the objects, are used to find the silhouette of the object [37] [85] and background subtraction is generally used to track the silhouette. Other methods to track the silhouette include the use of histograms [38] and optical flow [86]. Contour tracking iteratively evolves an initial contour in the previous frame to its new position in the current frame [25]. Because of the constant iterations, part of the object in the current frame must overlap with the object region in the previous frame. Contour tracking can be accomplished by either using state space models such as the contour shape and contour motion, or using *temporal image gradients* by computing the optical flow of the contour. Kalman or particle filters [87], Hidden Markov Model (*HMM*) [88], optical flow constraints [89] [47] [90] are used in contour tracking. Silhouette tracking is great for when tracking of the complete region of the object is necessary. It is especially useful for handling a large variety of shapes. Its deficiencies are its poor handling of occlusions, and dealing with silhouette-objects that are whole at the beginning of a frame, but are split further down a frame. For example, the authors in [8] show that a person carrying an object is considered as one silhouette, but when the person drops that object, the silhouette is split.

## **2.5 Social Interaction**

The authors of [91] classify the field of human-robot interaction into four “interaction paradigms” regarding the use of robots: used as a tool, a cyborg extension, an avatar, and a sociable partner. In the first, the robot is used to simply perform a task for the human. The second sees the robot physically merged with the human, i.e. robotic leg or arm. In the third paradigm, the human uses the robot as its face – the person projects themselves through the robot in order to communicate with another from far away. The last paradigm involves natural

interaction to exist between human and robots that mimics the interactions found between humans. Social interaction is the way people communicate and correspond with one another. It can be defined as actions taken by an individual that relate to the behavior of others, and therefore dictate the individual's further reactions and directions [12]. It also can be described as a focused or unfocused interaction between two people [92], where focused interactions are considered to be obvious states of initial focus, such as eye contact, or voice tones [8]. After this initial focus stage, interaction is moved into an engagement phase which involves proximity of one speaker to the other, gestures, and mutual glances, and then onwards to a communication phase. Social interactions between robots and humans should mimic these definitions. As mentioned in the introduction of this paper, for robots to be socially intelligent, their behaviour and actions must fully meet a person's social model for it, whilst remaining active in the entirety of the humans environment [17]. For robots to exist in this social model, they must follow four distinct modes of interaction, as explained in the introduction: they must be socially evocative, socially communicative, socially responsive, and simply sociable [17]. In [93], the authors point out that for social interaction to exist coherently between humans and robots, several questions need answering: What modes of communications should the robots employ? What is the role of the robot's physical embodiment? How to integrate the *a priori* knowledge about the users into the robot system? How can the human-robot interaction design ensure safety? And how are friendly, familiar, usable, and effective interaction models with a robot to be designed? The work done by [94] reinforces these questions by expressing that robots should show such characteristics as expressing or perceiving emotions, communicating with high-level dialogue, learning or recognizing models of other agents, establishing and maintaining social relationships, using natural cues such as gazes and gestures, exhibiting distinctive personalities and characters,

and learning or developing social competencies. In [91], the authors list three advantages of having socially interactive robots: people would find working with them more enjoyable, and would therefore feel more competent doing so; communicating to robots would be easier; and teaching the robots “new tricks” through imitation or emulation, would be easier.

## **2.6 Applications of Social Robotics**

The authors of [94] describe several components of a socially interactive robot: embodiment, emotion, dialog, personality, human-oriented perception, user modelling, socially situated learning, and intentionality. In [12], the authors add several more properties to this list regarding categories that socially interactive robots fall in, as well as the tasks they should carry out. In particular: the elderly, the disabled, those in special care (rehabilitation), and those in needs of friendships. The authors of that paper also express that SAR should be task driven – especially in the areas of tutoring, physical therapy, daily life assistance, and emotional expression. SAR should also be interactive in the sense they are capable of such things as speech, gestures, and/or direct input. Some examples of interactive robots include robotic shopping carts [95], robotic health-care aides [94] [96] (Figure 2.3), walker-aides [97], and robotic manipulators [98].



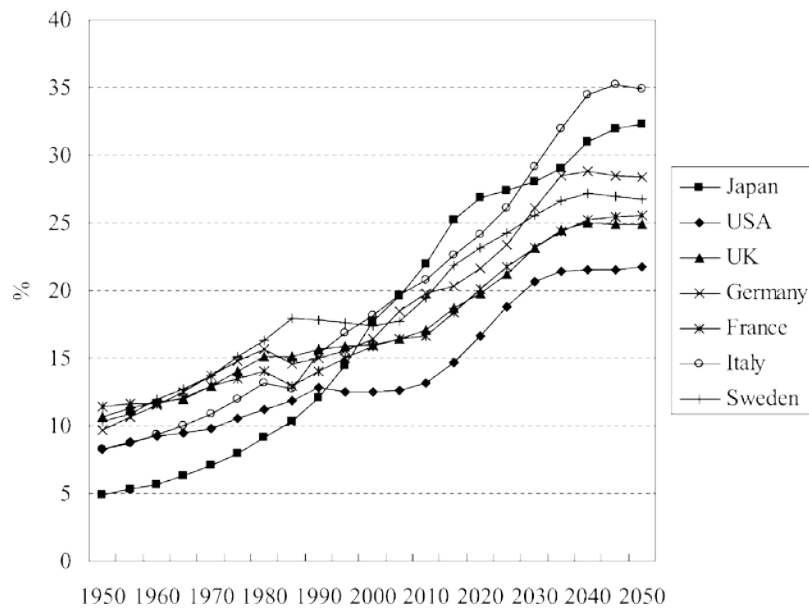


**Figure 2.3 - Pearl the nursebot [99]**

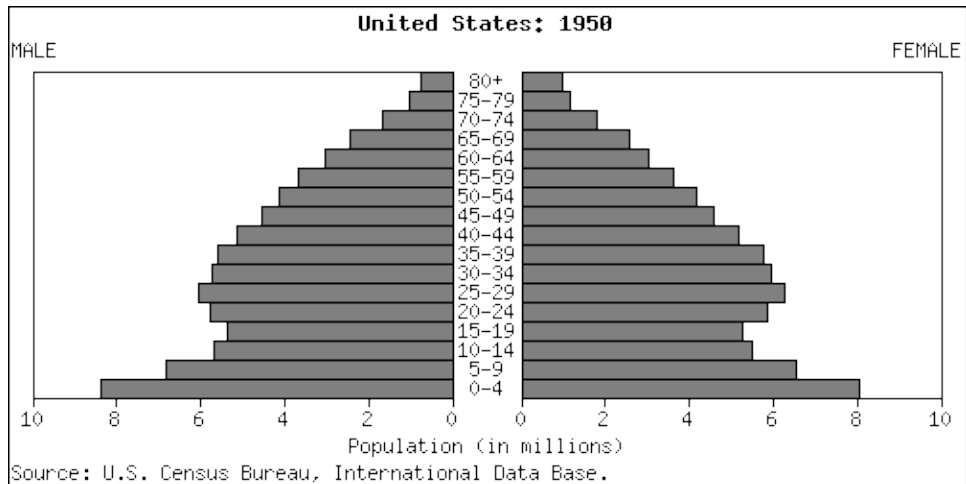
### **2.6.1 Elderly**

Figure 2.4 shows the changing proportions in several countries. According to the United Nations, the proportion of people 64 years and older in the population of a country exceeding 7%, indicates an aging society, with the proportion exceeding 14% indicating an aged society [100]. Figure 2.5 - Figure 2.8 shows the increase in number of people aged 60 and older in the United States. By 2050, it is projected that the population of the elderly will outnumber the younger population; seniors will account for 7.2% of the entire population of the United States. Assistive technologies can help seniors to “age in place” – it will allow them to stay in their homes for longer periods of times [101]. In fact, previous research has shown that senior Americans prefer to remain independent in their own house for as long as possible [102]. The health care costs associated with seniors are high – in America, Medicaid/Medicare pays for nearly 60% of the \$132 billion annual nursing home bill [101]. Having technology that helps seniors at home can improve their Quality of Life (*QOL*) as well as help save money.

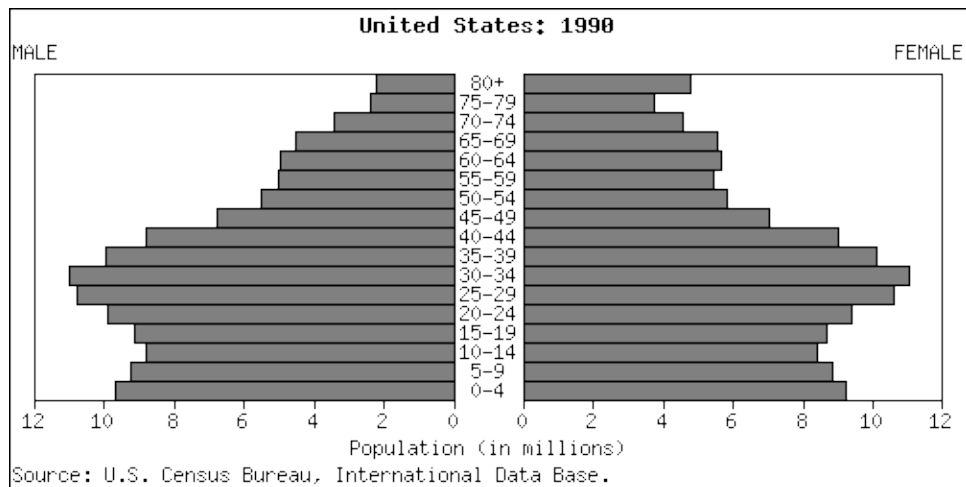
As mobility becomes a problem for the elderly, it is easy for them to fall into depression or become stressed [103]. Robots may act as companion robots to these people. They can communicate and/or simply spend time with, and be active, around the seniors. This will help them exercise their mind, thus lowering depression and stress levels. Many elderly people require significant amount of time in order to physically get from point A to point B. Instead of other humans walking alongside these seniors at a slow pace (thus holding them back from other tasks), a robot “walker” can be put into place.



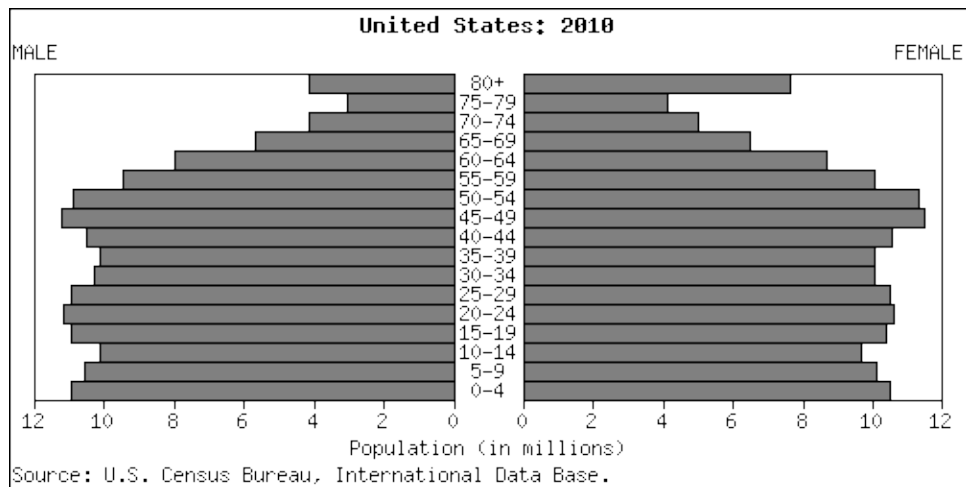
**Figure 2.4 - Ratio of people 65 years old and over to the total population of the most advanced countries [100]**



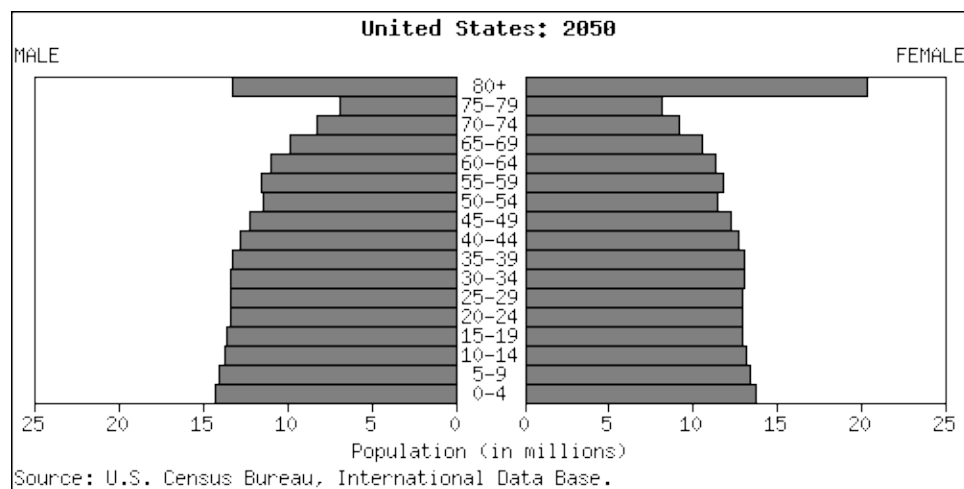
**Figure 2.5 - Population Pyramid for the United States in 1950 [104]**



**Figure 2.6 - Population Pyramid for the United States in 1990 [104]**



**Figure 2.7 - Population Pyramid for the United States in 2010 [104]**



**Figure 2.8 - Estimated Population Pyramid for the United States in 2050 [104]**

[96] shows that it is possible and successful to place interactive robots into an environment where elderly people reside. Since mobility is a problem for these seniors, it is harder for them to travel to health care facilities. There is an added benefit in that these robots can act as virtual bodies for health care workers – they can act as an interface between the medical industry and the elderly, and thus improve the frequency of care and oversight [105]. One of the most positive interaction models is animal-human interaction. The literature demonstrates that people who had animals as companions were better off in health and survival [106] [107] [108]. There are three main effects that animal assisted therapy has on people: psychological (relaxation, motivation), physiological (improvement of vital signs), and a social effect [109]. Paro (Figure 2.9), the seal robot, was used among the elderly at a day service center and a health service facility over several weeks. The researchers found that interaction with Paro increased the moods of the elderly, making them more active and communicative with each other and caregivers, and reduced their stress levels [109]. Paro can act both in a proactive and reactive manner. It can react to sudden simulation such as sudden loud sounds. It is equipped with a sensor system that

imitates the four primary senses: sight (light sensor), sound (speech recognition, sound direction), balance, and a tactile sense.



**Figure 2.9 - Paro, the seal robot [110]**

AIBO [111] (Figure 2.10) is another pet-like robot, in the form of a dog. It is able to learn and mature based on its experiences with its owner. It is also equipped with a similar sensor system as the Paro. Research has found that the elderly were able to touch and care for the dog, just like they would with a real one [112]. This was particular in cases where the AIBO was outfitted in a furry suit. The system called *PAMM*, Personal Aid for Mobility and Monitoring provides physical support and guidance to a senior, whilst monitoring the person's basic vital signs [113].

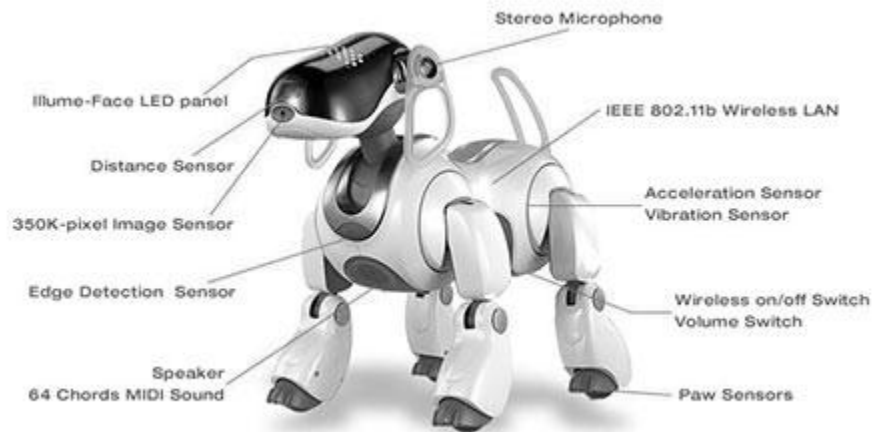


Figure 2.10 - AIBO, the robotic dog [114]

## 2.6.2 Disabled and Rehabilitation

In 2008, 19% of the United States population was found to have a disability. 11 million of those people, 6 years of age and older, need personal assistance with everyday activities [115]. Studies show that there are in excess of 730,000 strokes per year in the United States alone [116] [12]. Rehabilitation robots [117] - [118] can help those who are disabled or those in convalescent care. They address problems that people may have with motor, hearing, speech, visual, and cognitive impairment [119]. They can help the severely disabled, whose movements are restricted, become mobile. Control inputs on the face (i.e. joysticks controlled by the chin, or muscle movement in the face) can move a wheelchair [120]. Those with developmental disabilities (cognitive disorders), benefit from mobile robots because they are predictable, simple, and easy to comprehend [121]. The AURORA project showed how robots were used to teach children with autism basic interaction skills [122]. This project showed increased attention (visual contact, physical proximity) when children were paired with a robot as compared to children paired with a human mediator. The Handy 1 robotic system is capable of assisting severely

disabled people with functions such as eating, drinking, washing, teeth cleaning, shaving, and makeup applications [123].

### **2.6.3 Human Friendships**

The application regarding human friendships is quickly rising in popularity – it is one that helps form relationships with others. [124] proposes a model for estimating human friendships in the presence of a humanoid robot. This is a very important idea for socially interactive robots, as it allows the robot to understand and recognize human friendships and relationships. The authors of this paper conducted a study involving children in a classroom. By using only visual, non-verbal data, they found several important factors for friendship estimation: touch, gaze, co-presence, and distance. Using a robot called Robovie (Figure 2.11), interaction behaviours between robot and the children were based on three principles [124]: 1. Calling children's names. 2. Pseudo development – the robots interaction with a child increases proportionately to the child's interaction with it. 3. Secret sharing – as a child's interaction with the robot increases, the robot lets the child in on secrets. This experiment resulted in interaction increasing between the children themselves. The secrets children received became popular, and seemed to be sort of an anchor in building relationships. This kind of social robotics can be extended to the problem of bullying. As a robot learns about a child through interaction, it can act as a mediator in such places as the playground, or it can even inform a teacher, or parents, about possible bullying. [125] developed a storytelling robot for children. Kids can teach the robot to act out emotions such as sad, happy, excited, and then write stories using a storytelling software and include these emotions in the story. The story is then played out by the robot. This helps motivate the

children and reach their therapy goals by exercising muscles or joints, or by reflecting on the expression of emotions.



Figure 2.11 - Robovie the communication robot [126]

## 2.7 Designing Assistive Robots

The basics of designing a robot, whether it be social or otherwise, still exist – cost, reliability, robustness, and availability are all important factors [127]. However there are added factors that should be applied towards social robots. When it comes to designing a socially interactive robot, we can program them in two ways: biologically inspired, or functionally designed [94]. Designing a robot using the first method essentially molds the cognitive, behavioral, motivational, motor, and perceptual systems of a robot, to mimic the same characteristics in humans and other living things. The view of the second method is to design a robot that simply appears to be socially intelligent. This means that its internals aren't wired to be intelligent, i.e. through natural, human, or animalistic means. Rather, it puts on a mask, and appears to be socially intelligent.



Designing the physical structure of the robot is important also. That is not to say that the robot should look exactly like a human. Rather, the robot should be able to emulate a living thing that a human can relate to, i.e. think of a human's relationship with animals, like a chimpanzee. This is because characteristics of humans and animals, i.e. sight, hearing, should be a constant factor in the physical design. [94] discusses several types of embodiment for interactive robots: zoomorphic, caricatured, and functional. Zoomorphic robots are designed to imitate living creatures. This is meant to create robotic "companions". Caricatured robots are basically robots that look like an exaggerated or distorted version of an object, human, or other living thing. It can give the relationship a lighter feeling, i.e. it can help the user focus or distract attention from certain features of the robot, or even the environment. Functional robots are those that are built to reflect the environment they are situated in. For example, wheelchair robots can have larger seat space, and cargo space for users to ride in. These embodiment types can also be extended to personality types [94]: tool-like – robots that perform specific services, and pet-like – robots that exhibit entertainment like traits, i.e. Paro, AIBO.

The programmatic design of interactive robots is an important step. One of the most important features of assistive robots should be that they should have a (fast) real time response to events and actions that happen in an environment. If a reaction by a robot is slow, even by a few seconds, it will throw off any possible interaction with a human. [127] believes that low latency of an interactive social robot should be a primary requirement for social robots. In addition to its latency, the robot must be equipped to exist in a dynamically changing environment. This environment should be one that is comparable to an environment that a human is capable of being and working in. It involves various, dynamically changing, objects, paths, lighting

conditions, and physical environment sizes. In order to handle such an active setting, sensory systems must be endowed on the robot. As a human being relies on the five primary senses, so must a robot have a system that can mimic visual, vestibular, auditory, and tactile sense [128]. Some common sensor systems include: an odometry system, infrared/sonar receptors and emitters, laser range finders, imaging systems (i.e. camera), temperature sensors, and accelerometers and gyroscopes [129].

The overall system should not just be a receptive system; rather it should give an output, i.e. feedback, based on the various inputs. Given a human's interaction, such as social cues or gestures, the robot should use its sensory system to understand those cues and gestures, and then output a meaningful response back to the human. This response should be one that is understandable to the human, i.e. a nod, or an auditory "yes" or "no". [130] classifies two types of systems involving the role of a robot's appearance and speech: a closed-loop, and an open-loop system. The closed-loop system receives feedback about the state of the system (environment), whereas the open-loop system does not use feedback, but instead relies entirely on the system model. [127] expresses that social robotic systems must respond based on "hidden" states of goal, desire, and intent, rather than explicit actions. This means that the designs of these systems should ideally respond to the intent of the human rather than the human's explicit actions. This definition forms the basis for any socially interactive or assistive robot. [131] is a good example of this kind of system – it is an auditory dialogue system for a nursing home assistant. A robot named Kismet [17] (Figure 2.12), is designed to be a robotic creature that can interact physically, affectively, and socially with humans in order to ultimately learn from them. Kismet is equipped with skills and mechanisms to exist in even a complex

environment – it can tune its responses to a human, and give the human social cues, so that he/she is better able to tune him/herself to Kismet. For example, Kismet has the ability to direct its attention to a reference by the human; it has the ability to give readable, expressive feedback to the human, the ability to recognize expressive feedback such as praise and prohibition, the ability to take turns to structure the learning episodes, and the ability to regulate interaction to establish a suitable learning environments [132] - [133].

While responding to the user's intent, the robot must be careful to minimize the stress to the user. One such way of doing this is to evaluate comfort levels of a human using a Galvanic Skin Response (*GSR*) [130]. The *GSR* is a type of biofeedback technique that basically measures the emotional states of a human. Using techniques such as this allows the (emotional) interactions between humans and robots to be mapped, which results in designing the robot to be more aware and receptive of a person's emotions, thus fine-tuning its responses.



Figure 2.12 - Kismet, a robot that mimics and responds to human emotions [134]

## 2.8 Assistive Robotic Wheelchairs

As mentioned in the introduction of this paper, between 1994-2005, there has been an increase of 1.5 million people who are dependent on wheelchairs. As the baby boomers reach the age of 65, the dependence on assistive tools such as wheelchair will grow higher. Social and economic means are motivators of research into assistive robotics [135]. Social and economic forces can be related to the aging of society and the presence of physical impairment. As a person's emotions deteriorate, their need for companionship and social activity may increase. Assistive robotic wheelchairs may help relieve those needs as well as the mobility constraints and stress-related problems associated with a user's mobile tasks [135]. Economically, with the increased use of interactive wheelchairs, the dependence on health care, and the costs associated with them, will decrease. A wheelchair that is robotic and assistive does not mean that the wheelchair must be fully autonomous. Rather, the human user can share control with the robot wheelchair. Using shared control, the strengths of humans and machines can be combined to reduce their weaknesses [135]. Extending the notion of interactive robots responding towards intent rather than explicit actions, is the fact that robot wheelchairs should infer the user's plan from their actions and from the environment itself. "Intent estimation" [136] [137] can be defined as inferring the actions taken by an actor into a goal, whilst organizing those actions in some sort of planned structure.

An important concept that relates to this definition would be the behaviour of robots to naturally follow a person. A study by [138] found that participants believed that a direction-following behaviour was significantly more human-like and more natural than a path-following behaviour. The direction-following behaviour means that the robot infers the individual's direction based on

his/her actions, and thus reacts to follow in that direction. A path-following type of robot does not care about the actions that happen in the environment, rather simply follows the path taken by the individual. For example, every step, acceleration or deceleration, or turn, is mimicked exactly by the robot. The experiments run in the same study [138], show that direction-following robots were able to follow a person through doorways and around corners without collisions and turned the corners smoothly. This is in contrast to the path-following robots – these robots had trouble turning corners, since they tried to use the exact same path as humans. If a human made a tight turn, which would be easy enough for him/her, the robot would try to follow this same path, and evidently would have trouble doing so. Wheellesley [139], is a robotic wheelchair that can provide users with driving assistance. It is equipped with an array of infrared and sonar sensors that are used to sense the environment. If an object is in the way of the wheelchair, the system will not allow the chair to continue moving in that direction. Work done by [140] (Figure 2.13) and [141] uses a vision system to direct the wheelchairs path. What is interesting about this system is that the camera used is pointed at the rider, whilst the rider is driving it. By using face recognition and facial directions, the wheelchair can move accordingly to the user's intent. In [140] when the user does get off the wheelchair, the user can aim the camera outwards to the scene ahead – the system will then look for the users face, and move accordingly to gestures provided by the user. The problem with this method is primarily that the user has to face the wheelchair in order for it to follow him/her, and second, that in a cluttered environment, this method may not perform accurately. [142] presents a wheelchair that is capable of navigating semi-autonomously within its workspace. It functions by using sonars to find depth distance, and a 360 degree field of view camera. This system uses a tri-histogram matching method to follow the person, by comparing the histograms of the head, torso, and legs,

to a pre-modelled histogram found in a database. This method worked well, however it was not real time, but ran at 3 Hz. It also failed in cases when moving persons wore the same colour clothes as the background. MAid [143] (Figure 2.14), Mobility Aid for Elderly and Disabled People, is an intelligent robotic wheelchair whose task is to support and transport people with limited motion skills. It is equipped with sonar and infrared cameras, as well as a 2D laser range-finder, which was used to estimate speed and direction of objects. Based on the sensor data, MAid can avoid objects ahead of time, while navigating through a crowded area and adjusting its velocity. Due to the range-finder, MAid is able to detect objects in approximately 70 ms. When the length of a sensor observation (the recording of the range image) is included, the cycle time increases to 0.3s. This can cause an object to be misdetected, and for a collision to be imminent. Other range-finder/sensor systems exist also, such as SENA [144] (Figure 2.15) and [145].



Figure 2.13 - A semi-autonomous robotic wheelchair [140]



Figure 2.14 - MAid, the robotic wheelchair [143]



Figure 2.15 - SENA, a robotic wheelchair [144]

## 2.9 Summary

This chapter provided the background information related to the fields of tracking an object and socially interactive robots. It gave the reader a clearer understanding of representing and detecting an object, and the types of tracking methods that are commonly used. The paradigm of social interaction was discussed, and its applications relating to the elderly, the disabled and those in rehabilitation, as well as friendships, were talked about. Existing works done with assistive and socially robotic systems were described in length.

## **Chapter 3. Methodology**

To adapt a transport system like an electric wheelchair to autonomously follow a person, various techniques were put together to form an algorithm. This chapter will explain these techniques in detail, and will show the methods used to make the wheelchair socially interactive.

### **3.1 Architecture of SIGHT**

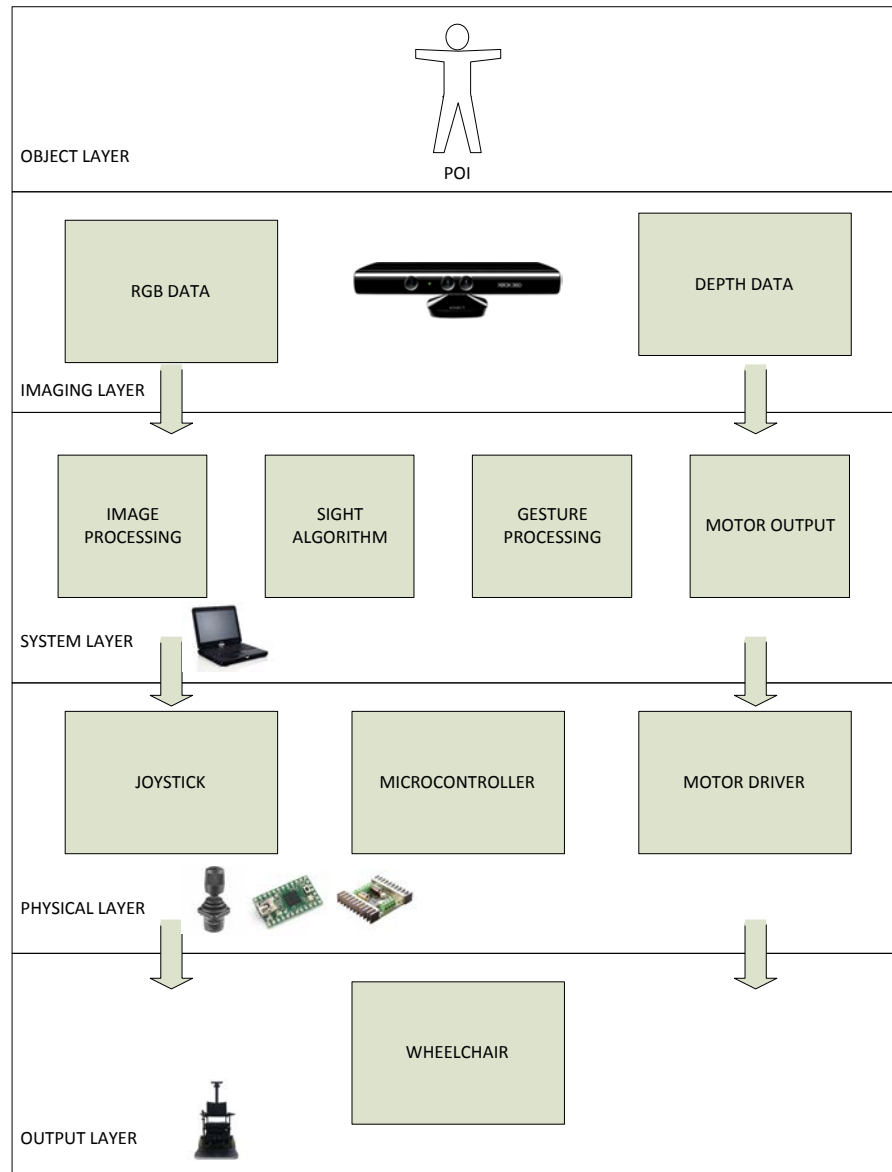
Figure 3.1 depicts a high level layout of what the SIGHT system looks like. The system is comprised of five main layers: the object, imaging, system, physical, and output layers. The object layer decides on an OOI or POI to track/follow. The imaging layer takes camera data as an input, processes it into something meaningful, and then sends that data into the system layer. The system layer correspondingly parses this data and runs it through the main SIGHT and gesture processing algorithms. The algorithm then outputs motor control information to the physical layer, and subsequently the wheelchair, the output layer, is controlled. These layers will be further discussed below.

### **3.2 Object Layer**

SIGHT allows for an object (or person) of interest, to be selected manually or via the gesture system. At the start of the system, during the initialization of the software, the person or object to be tracked should be at a comfortable tracking distance away from the transport. This initial distance is important because it will be used by the transport as an acceptable distance to maintain following. This means that if the person is 3 metres away from the transport at the time of initialization, the transport will always maintain a distance of 3 metres to the person, regardless of whether that person speeds up or slows down. This distance will be referred to as

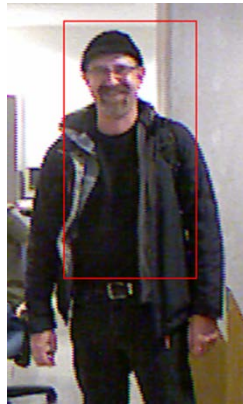


the Optimal Subjective Proximity (*OSP*). Edward Hall's *Proxemics* [146] gives four categories of interpersonal distances: *intimate* (0-0.46 metres), *personal* (0.46-1.22 metres), *social* (1.22-3.66 metres), and *public* (>3.66 metres) [147]. The *OSP* in the SIGHT system should fall in Hall's social distance category, however the theoretical distance limits for SIGHT are between 1.5 – 6 metres.



**Figure 3.1 - Architecture of SIGHT**

There are two ways of initially setting an object to track – manually, via the computer’s mouse, or interactively, via the gesturing engine. In both cases, it is important for the user or object to stand at their OSP. If the manual method of choosing an OOI is used, then the system takes a snapshot of the object (the object must be at the OSP by this time) upon initialization, and then on that snapshot, the user creates a box, or region of interest (*ROI*), around a region of the object that is to be tracked. In general, if the object is a person, a good region to select would be from the top of their head, to the mid region of their body, i.e. their belly button. Figure 3.2 shows what an ROI manual selection looks like on a snapshot image of the POI.



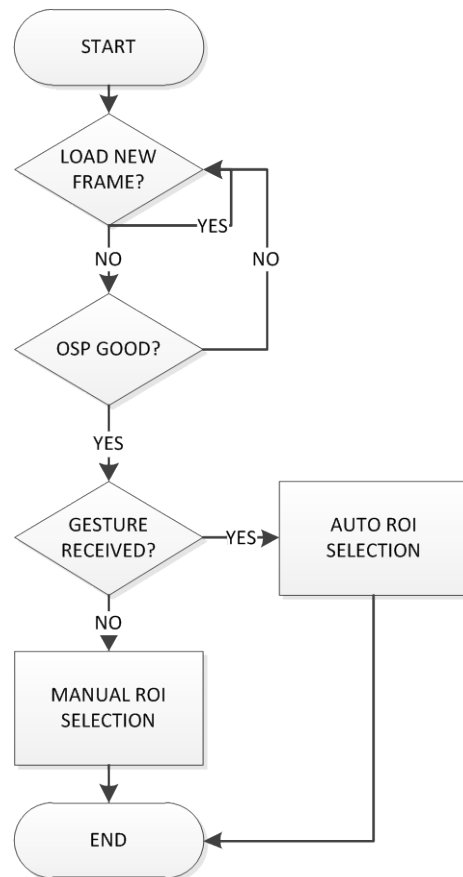
**Figure 3.2 - An ROI selected on a POI <sup>1</sup>**

If the gesture engine is to be used, then calibration of the user using a third party application is necessary. Once the calibration is complete, the user, standing at their OSP, does a pre-defined gesture, e.g. a hand wave motion, which would automatically create a ROI on the person. The gesture that the person wants to use is set at the time of calibration. Regardless of which method (manual or automatic), the outcome will look like Figure 3.2.

---

<sup>1</sup> The ROI was taken of the front of the person’s body, however the same ROI can be taken of the back of the POI. This image was taken to easily demonstrate the selection of an ROI

Since this system can be run in a real-time, clutter-independent, environment, the ROI can be selected without much restriction. However, since part of the algorithm is based on color representations of the OOI, it is important that the colors the POI is wearing differs from the background. This will ensure that the SIGHT tracking algorithm operates at a smooth and accurate rate. Figure 3.3 shows a flowchart of the object layer subroutine.



**Figure 3.3 - The object layer subroutine**

### **3.3 Imaging Layer**

One of SIGHT's advantages is its adaptability to various cameras and vision devices. One of its primary goals is to be able to use a fairly cheap, common, off-the-shelf camera for tracking. Two cameras used in particular were the Sony PlayStation 3 (PS3) Eye [148] (Figure 3.4), and

the Microsoft Xbox Kinect [149] (Figure 3.5). The PlayStation Eye was first used because of its high frame rate capabilities and the open source libraries that existed for it. It could reach a frame rate of 60 Hz at a resolution of 640x480 pixels, and 120 Hz at a resolution of 320x240 pixels. This worked out great for real-time (around 25Hz or higher) usage – the PS3 Eye was able to track a selected object, in real-time, with few errors. The main problem with it however, was its inability to perceive distance. For distance to be known, complex mathematical calculations would have had to be made. This slowed down the SIGHT algorithm enough that the tracking was no longer in real time, but lagged noticeably.



Figure 3.4 - Sony PlayStation 3 Eye [148]



Figure 3.5 - Microsoft Xbox Kinect [149]

The Xbox Kinect served well for the purposes of perceiving distance, as well as being able to maintain a suitable frame rate. The Kinect is able to output the Red, Green, Blue (*RGB*) video at a frame rate of 30 Hz, at a resolution of 640x480 pixels. It uses an infrared laser projector, combined with a monochrome CMOS sensor, to capture video data in 3D. This 3D data, provides 11 bits of depth data, thus providing 2048 levels of sensitivity, which basically means that the depth sensor of the Kinect has a range of about 0.7 – 6 metres [150]. Two open source libraries exist for communicating with the Kinect: libfreenect [150], and OpenNI [151], attached with Primesense [152]. Both these libraries were used and tested with SIGHT. OpenNI contained more features, especially relating to gestures, and was therefore chosen as the main

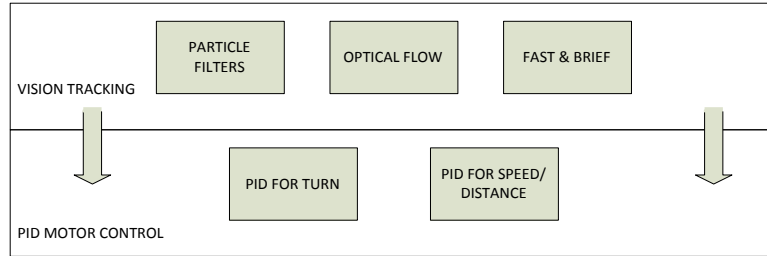
Kinect library to be used with SIGHT. By using both the RGB and depth data from the Kinect, we were able to provide a real-time, lag free, object tracking system.

### 3.4 System Layer

This layer contains the bread and butter of the algorithms in the SIGHT system. The system layer receives camera data from the above imaging layer, processes that data, inputs that into the SIGHT algorithms and gesture processing engine, and finally outputs motor commands. The camera communicates to the computer via USB, and uses the open source libraries, OpenNI and libfreenect, to provide a method of receiving and parsing this data in a trivial manner. The programming language used in SIGHT is a mix of C and C++, with the inclusion of OpenCV [153] libraries. RGB and depth data from the camera are stored into 8-bit matrices for RGB, and 16-bit matrices for depth, of size 640x480x3 for the RGB, and 640x480x2 for the depth. These matrices are converted into OpenCV specific structures for further use. These data structures are sent into the SIGHT algorithms and the gesture processing engine to determine tracking and motor control. The computing system used in this layer is a Fujitsu TH700 [154], equipped with an Intel Core i3-380M Processor, 4 GB DDR3 of memory, an onboard Intel HD Graphics video card, running the Ubuntu 10.10 operating system.

At its highest level, the SIGHT system is two tiered (Figure 3.6) – it consists of a vision tracking algorithm, and a Proportional, Integral, Derivative (*PID*) algorithm to control the motors of the transport. The vision tracking algorithm joins two tracking methods, particle filters and optical flow, and a FAST feature finding method, to form a more robust, adaptive tracking scheme. The

PID algorithm itself is made up of two stages, one to control tuning the turning radius, and another to determine speed relative to distance to the OOI.



**Figure 3.6 - Two-Tiered High Level Architecture of SIGHT**

### 3.4.1 Vision Tracking Algorithm

Particle filters and optical flow techniques are applied and are run jointly to give a more accurate tracking measure. When the tracking methods fail and the object is lost, a method for re-finding the object is presented. This method is called FAST and BRIEF. The following subsections will give a clearer understanding of each technique and how it's used.

#### 3.4.1.1 Particle Filters

Particles filters [35] [155] [156] [157] are used to estimate the position of an object using the distribution of the particles. They are also known as the sequential Monte Carlo method, or a Sampling Importance Resampling (*SIR*) filter [77]. The idea behind them is to continually collect random samples (particles) from a scene, and compare them to a model. To determine if a particle is successful or not would depend on how well that particle matches the model. Each particle has a weight assigned to it, which is proportional to the probability of the particle being at the correct position of the object. Particle filters use an intensity based approach that weighs particles higher based on a simple numerical comparison of its own weight, to a reference weight

of the object to be tracked. This means that the higher the particle weight, the higher probability it has of being at the correct position of where the object is heading. As explained in the previous chapter, particle filters are comprised of three steps: selection (sampling), prediction (importance), and correction (resampling). Mathematically, a distribution (Equation (3.1)) of the state of the tracked object is approximated by maintaining a set of weighted particles (Equation(3.2)) over time, where each particle (Equation (3.3)) consists of its state vector  $x_t^j$  and an importance weight  $\pi_t^j$  [77].

$$p(X) \tag{3.1}$$

$$S_t = \{s_t^j, j \in \{1 \dots J\}\} \tag{3.2}$$

$$s_t^j = (x_t^j, \pi_t^j) \tag{3.3}$$

The set of particles is updated from frame to frame using the same recursive procedure from [77]: First, a new sample set  $S_t$  is drawn with replacement from the previous set  $S_{t-1}$ , where a sample  $s_{t-1}^i$  from the old set is chosen with probability proportional to its weight  $\pi_{t-1}^i$ . Next, for each sample a new state  $x_t^j$  is determined by sampling from the motion model (Equation (3.4)). Finally, the measurement of the new frame  $Z_t$  is integrated by updating the importance weights  $\pi_t^j$  with the likelihood of the observation (Equation (3.5)).

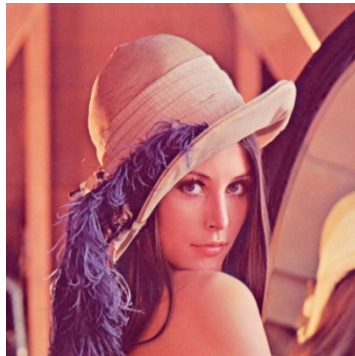
$$p(X_t | X_{t-1} = x_{t-1}^i) \tag{3.4}$$

$$\pi_t^j = p(Z_t | X_t = x_t^j, Z_0, Z_1 \dots Z_{t-1}) \tag{3.5}$$

(For validity of these equations, the reader can look at [77] and [79]). Since we want to track an object visually, it makes sense that the samples used in the particle filter method be of a physical nature, i.e. color representation. In order to represent the colors of an object, color histograms are used. A histogram is a representation of the distribution of data. They are “collected counts of the underlying data organized into a set of predefined bins” [83]. The more bins that are used, the more detail the histogram will give. Histogram’s are relatively invariant with translation and rotation about the viewing axis, and vary only slowly with the angle of view [158] [159]. A color histogram is a representation of the distribution of colors in an image. They are mainly used for comparison purposes, whether it be comparing objects from scene to scene, or comparing each scene to a library of predefined objects. Typically, after the image data is put into the various bins, the entire histogram is normalized, so that each individual bin represents the fraction of the total histogram. In general use, the RGB color space is used to represent colors in an image. Figure 3.7 shows the popular and standardized Lena RGB image. The problem with detecting the RGB values of an object is that it can change depending on the lighting conditions and noise of the environment [160]. The Hue, Saturation, Value (*HSV*) color space is better suited for dynamic environments. The Hue component represents the color’s hue angle (essentially it represents the color itself), the Saturation represents the color purity (how much color), and the Value component represents the brightness of the color. In a dynamic setting, the Value component changes drastically, depending on light, whereas the Hue and Saturation values stay relatively constant. Because of this, the Value component is typically not used in a color histogram. To build an HS (Hue-Saturation) histogram, it’s important to first note the range of values for the HS color space. The Hue values range between 0-180 (units are in degrees) while the Saturation values range between 0-255°. The number of bins commonly



used in an HS histogram would be 30 bins for the Hue values, and 32 for the Saturation. This leaves each bin to hold 6 values of intensity for Hue, and 8 intensity values for the Saturation. Figure 3.8 shows what a Hue-Saturation histogram of the Lena image looks like, using 30 bins for Hue, and 32 bins for Saturation. By comparing the bins of two histograms, similarity or differences between images or frames can be easily found. There are five main histogram comparison methods: correlation, chi-square, intersection, Bhattacharyya, and the Earth Mover's Distance (*EMD*) [83]. Intersection was found to work well for quick matches, chi-square and Bhattacharyya worked best for slower but more accurate matches, and the EMD method gave the most intuitive matches, but was much slower. In this paper, the Bhattacharyya method was used to match histograms across frames. Equation (3.6) shows the mathematical representation of this method [83].



**Figure 3.7 - An RGB image of Lena**



**Figure 3.8 - A Hue-Saturation Histogram of Lena**

$$d_{Bhattacharyya}(H_1, H_2) = \sqrt{1 - \sum_i \frac{\sqrt{H_1(i) \cdot H_2(i)}}{\sqrt{\sum_i H_1(i) \cdot \sum_i H_2(i)}}} \quad (3.6)$$

In order to assign weights to the color histogram samples, a color histogram of the selected ROI is taken and matched against. The following algorithm forms the basis of particle filter tracking using color histogram samples:

Initialization (Selection/Sampling): Create  $N$  particles with an initial state of 0 (Equation (3.7)), where  $x_0^{(n)}$ ,  $w_0^{(n)}$ ,  $h_0^{(n)}$ , and  $d_0^{(n)}$  are the  $n^{th}$  position vectors, weights, histogram values, and dimensions (width and height) respectively.

$$s_0^{(n)} = x_0^{(n)}, w_0^{(n)}, h_0^{(n)}, d_0^{(n)} = 0, n = 1, \dots, N \quad (3.7)$$

*Step 1:* For each particle, set the position and dimension to equal the ROI's position and dimension values (Equation (3.8), Equation (3.9)).

$$x_t^{(n)} = x_{ROI} \quad (3.8)$$

$$d_t^{(n)} = d_{ROI} \quad (3.9)$$

*Step 2:* For each particle, set the histogram value to equal the ROI's histogram value (Equation (3.10)(3.9)).

$$h_t^{(n)} = h_{ROI} \quad (3.10)$$

*Step 3: Prediction/Importance:* Using a uniformly distributed random number, assign a new position for each particle, based on the particles last known position (Equation (3.11)).

$$x_t^{(n)} = rand \% h_{ROI} \quad (3.11)$$

*Step 4:* Reset the weight of each particle (Equation (3.12)).

$$w_t^{(n)} = 0 \quad (3.12)$$

*Step 5: Update:* For each particle, set a region of interest with the ROI width and height are equal to the target ROI's width and height (Equation (3.13)). Find the histogram value of the particle's ROI (Equation (3.14)).

$$d_t^{(n)} = d_{original ROI} \quad (3.13)$$

$$h_t^{(n)} = h_{particle ROI} \quad (3.14)$$

*Step 6:* For each particle match the particles ROI histogram value to the target ROI histogram value using the Bhattacharyya method. Set the weight to this matched value (Equation (3.15)) and then reset the particle's ROI.

$$w_t^{(n)} = \text{Bhattacharyya}(h_t^{(n)}, h_{ROI}) \quad (3.15)$$

*Step 7: Correction/Resampling:* Sort the particles by weight, with the heaviest weight at the beginning of the list.

*Step 8:* Replace the  $N$  particles with the top fifteen percent of the particles.

*Step 9: Estimate position:* Calculate the weighted sum of the particles weights  $w_s$  (Equation (3.16)), position  $x_s$  (Equation (3.17)), and dimensions  $d_s$  (Equation (3.18)), and determine the object's new position  $x_{obj}$  (Equation (3.19)) and dimensions  $d_{obj}$  (Equation (3.20)) based on  $w_s$ .

$$w_s = \sum_{i=0}^N w_t^{(n)} \quad (3.16)$$

$$x_s = \sum_{i=0}^N x_t^{(n)} \quad (3.17)$$

$$d_s = \sum_{i=0}^N d_t^{(n)} \quad (3.18)$$

$$x_{obj} = \frac{x_s}{w_s} \quad (3.19)$$

$$d_{obj} = \frac{d_s}{w_s} \quad (3.20)$$

Appendix A shows a flowchart of the particle filter algorithm.

### 3.4.1.2 Optical Flow

Optical flow is the calculation of the translation of a pixels velocity vectors, based on brightness patterns across two frames. Dense optical flow, as in the Horn-Schunck method [161], computes velocity for every pixel in the entire image. Sparse optical flow, such as the Lucas-Kanade (*LK*) method [82] on the other hand, computes the optical flow on some subset of points, or feature/interest points, of the image. This second tracking method used in SIGHT relies on the LK sparse optical flow method. This method is applicable in the “sparse” sense, because it relies on local information that is derived from some small window surrounding each of the points of interest [83]. In the case where large motions cause the interest points to move outside of that small window, a pyramidal method called, which basically means the size of the window increases with each run.

The LK algorithm is based on three assumptions or directions [83] (Figure 3.9): 1. *Brightness constancy*, 2. *Temporal persistence* or “small movements”, and 3. *Spatial coherence*. The brightness constancy assumption tells us that the brightness of a tracked pixel does not change as it moves from frame to frame, or over time (Equation (3.21), Equation (3.22)) [83]. The temporal persistence argument states that the image motion of a surface patch changes slowly over time. This means that frame to frame, the object does not move drastically – the change is differentially small. The third rule, spatial coherence, declares that neighbouring points in a scene belong to the same surface, and thus have similar motion and projects to nearby points on the image plane.

$$f(x, t) \equiv I(x(t), t) = I(x(t + dt), t + dt) \quad (3.21)$$

$$\frac{\partial f(x)}{\partial t} = 0 \quad (3.22)$$

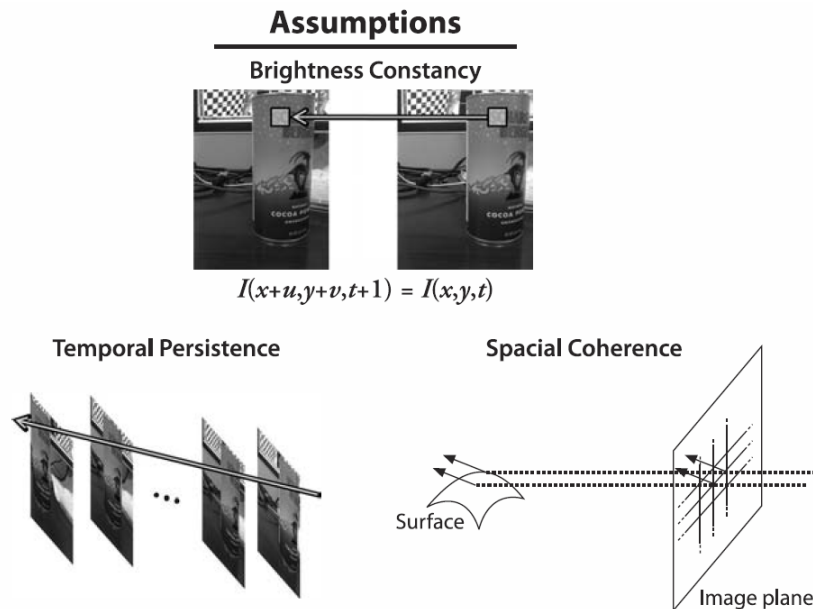


Figure 3.9 - Lucas-Kanade Optical Flow Assumptions [83]

To determine the points of interest needed to use the LK optical flow method, we use points in the image that have corners. To determine corners in an image, we use a method given by Harris [51]: we consider the autocorrelation matrix of the second-order derivatives images over a small window around each point. Corners are places in the image where this matrix has two large eigenvalues. This basically means that a corner is defined by edges going in at least two separate directions centered around a point. Using corners of an image gives us the advantage of not worrying about rotation. Shi and Tomasi extended the Harris corner finding method by stating that good corners are those whose smaller eigenvalue (one of two values) is greater than some minimum threshold [52]. The Shi and Tomasi method is used in conjunction with the Lucas-Kanade sparse optical flow method in SIGHT. The following algorithm forms the basis of the optical flow and corner finding tracking method:

*Initialization:* Create a list of  $M$  points  $(x,y)$  (Equation (3.23)) with an initial state of 0, where  $f_0^m$  is the original feature list position points.

$$f_0^{(m)} = 0, m = 1, \dots, M \quad (3.23)$$

*Step 1:* Upon initial ROI selection, find good features to track on the ROI<sup>2</sup>, and update the feature list to equal the reference feature point positions found (Equation (3.24)).

$$f_t^{(m)} = f_{ref}^m \quad (3.24)$$

---

<sup>2</sup> Finding features in the ROI, and calculating optical flow, was accomplished using the built in functions from OpenCV.

*Step 2: Update:* Upon receiving a new frame, use optical flow to find the new positions the feature points have translated to<sup>2</sup>.

*Step 3:* Calculate the average position of the feature point positions found in the new frame (3.25)).

$$x_t^{(m)} = avg(f_t^m) \quad (3.25)$$

*Step 4:* Set a ROI around the new point, with the ROI's dimensions  $d_t^{(m)}$  (width and height) equal to the original ROI width and height (Equation (3.26)).

$$d_t^{(m)} = d_{ROI} \quad (3.26)$$

*Step 5: Correction:* Remove any outliers (features found outside the ROI) from the updated feature list.

*Step 6:* Find the average position of the updated feature list not including the outliers.

*Step 7: Estimate position:* Set a new ROI around the average position from Step 6. The object's new position is equal to the average position found in step 6 (Equation (3.27)), and its ROI equal to this ROI.

$$x_{obj} = avg(f_{correct}^m_t) \quad (3.27)$$

### 3.4.1.3 Combination of PF and OF

SIGHT uses a combination of both the particle filter and optical flow tracking methods. The reason for doing this is so several common tracking issues such as camera motion, partial occlusions, clutter, and scale rotation and variations, can be dealt with by combining these two

algorithms. The following algorithm shows how the SIGHT system tracks an OOI using both particle filters and optical flow:

*Initialization:* Select an ROI

*Step 1:* Setup LK optical flow variables

*Step 2:* Find and save features found in the ROI into a reference features list

*Step 3:* Calculate and save the histogram of the ROI

*Step 4:* Initialize particles

*Step 5:* Predict particles

*Step 6:* Update and resample particles

*Step 7:* Estimate object position based on particles

*Step 8:* Update features

*Step 9:* Run correction on features

*Step 10:* Estimate object position based on optical flow

*Step 11:* Calculate histogram of ROI from step 7

*Step 12:* Calculate histogram of ROI from step 10

*Step 13:* Make sure both histograms from steps 11-12 are above a threshold

*Step 14:* If the difference of histograms from Step 11-12 is greater than a threshold, then set the smaller histogram value to equal 0, and the larger to equal 1 (analytically compare histograms)

*Step 15:* Set the new object position to equal the average position of steps 7 and 10

*Step 16:* Equate the reference feature list to the updated feature list

*Step 17:* Repeat steps 2-15



Based on the positions found individually by the particle filter and optical flow methods, a more precise object position can be determined. By comparing the histogram values of each of the PF and OF ROIs to a threshold, allows for error checking – making sure the new position definitely is the object’s new position, and not a misread, or mis-tracked position. Also comparing the differences of the histograms for OF and PF allows for further error checking. If the difference is greater than a threshold, then the one that weighs less (less because the smaller the histogram value the greater the chance of a match), is set to zero and used, and the one that weighs more is set to 1 and not used. Finally, equating the reference feature list to a new set of features (step 16), allows for the tracking program to adapt its view of the object over time, across various conditions such as lighting and orientation. Appendix B shows a flowchart of calculating the object’s new position.

#### **3.4.1.4 FAST & BRIEF**

In instances where the tracking scheme fails, SIGHT will try to seek and find the lost object. This is a feature many of the systems discussed in the previous chapter do not have. To do this we use a method called FAST (Features from Accelerated Segment Test) feature detection and matching, created by Edward Rosten [54]. FAST is a corner detection method that works by detecting features at a point  $p$  if the intensities of at least 12 contiguous pixels are all above or all below the intensity of  $p$  by some threshold  $t$ . In the SIGHT system, the FAST corner detection method is run at the beginning, on the reference frame (where the ROI is first selected). The list of the features is run through a keypoint or descriptor, extraction method called BRIEF (Binary Robust Independent Elementary Features) [162]. This extraction method takes the features found using the FAST corner detection, and calculates interest points around each feature. Each

of the BRIEF descriptors is made up of a binary string 64 bytes long which is basically a representation of an image patch around each corner. Although BRIEF is not designed to be rotationally invariant, its speed of detection beats other descriptor methods such as SURF [162]. Both the reference features found by FAST and the keypoints extracted by BRIEF are saved in the beginning for comparison later. When the object is deemed lost by the program, i.e. when the histogram value of either the particle filter or optical flow position is greater than a threshold, FAST corners and keypoints are again extracted from the frame at the time the object was lost, and using a brute force matching method, the positions for where the reference corners and keypoints match the best to the corners and keypoints are the current frame, is used for object re-detection. Upon re-detection of the object, SIGHT continues tracking the object as in the previous subsection. Appendix C shows a clearer picture of how finding a lost object works using FAST and BRIEF. Figure 3.10 shows how features are matched across two frames.



Figure 3.10 - FAST and BRIEF matching a POI

### 3.4.2 PID Motor Control

The second main tier of this system is SIGHT's capabilities to control the transport's motor speeds in relation to what it sees and to where the object is that it is tracking. In order to do this, a PID control system is used. A PID controller is essentially a feedback controller consisting of

proportional (P), integral (I), and derivative (D) elements. These three elements produce outputs with the following nature [163]: ‘P’ element: proportional to the error at the instant  $t$ , which is the “present” error; ‘I’ element: proportional to the integral of the error up to the instant  $t$ , which can be interpreted as the accumulation of the “past” error; and ‘D’ element: proportional to the derivative of the error at the instant  $t$ , which can be interpreted as the prediction of the “future” error. Taken together, this means that a PID controller takes errors in the past, present, and future into consideration [163]. Mathematically Equation (3.28) describes the PID controller (Figure 3.11), where  $K_p$  is a proportional gain constant,  $K_I$  is an integral gain constant,  $K_D$  is a derivative gain constant,  $e$  is the error between the setpoint and the process variable, and  $t$  is the time at the present [164].

$$u(t) = K_p e(t) + K_I \int_0^t e(\tau) d\tau - K_D \frac{d}{dt} y(t) \quad (3.28)$$

The set point is the point at where the control is stable at, and the error is anything subtracted from the set point.

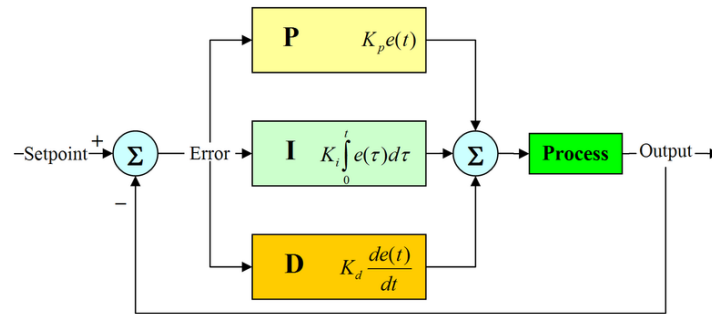


Figure 3.11 - A graphical representation of a PID controller [165]

A general PID motor control algorithm looks like this:

*Initialization:* Set setpoint variable.

*Step 1:* Find the current position.

*Step 2:* Find proportional value (Equation (3.29)).

*Step 3:* Find integral value (Equation (3.30)).

*Step 4:* Find derivative value (Equation (3.31)).

*Step 5:* Save the current proportional value as the previous proportional value (Equation (3.32)).

*Step 5:* Find and limit error value (Equation (3.33)).

*Step 6:* Add error value to motor speeds (Equation (3.34))

$$\text{proportional} = \text{position} - \text{setpoint} \quad (3.29)$$

$$\text{integral} = \text{integral} + \text{proportional} \quad (3.30)$$

$$\text{derivative} = \text{proportional} - \text{prev\_porportional} \quad (3.31)$$

$$\text{prev\_proportional} = \text{proportional} \quad (3.32)$$

$$\text{error\_value} = (\text{proportional} * K_P) + (\text{integral} * K_I) + (\text{derivative} * K_D) \quad (3.33)$$

$$\text{motor\_speed} = \text{max\_speed} + \text{error\_value} \quad (3.34)$$

In order to make the wheelchair keep its pace and course with the target, a two stage PID control scheme was used, one for determining turning control, and another for maintaining distance to the object. This seemed the most natural way of following an object – if one person were to follow another person, it would be logical that the person behind keep the same course as the person in front. For example, if the person ahead were to turn left, so would the person behind. After keeping course, it would make sense for the person behind to preserve its distance to the person ahead. This means that if the person ahead was to walk faster, or to slow down, the person behind would increase or decrease his or her speed in relation to the distance between

him/her and the person in front. The setpoint used in the PID turning method is the tracked objects position (the centre point of the object's ROI), and the position used is half of the frame size x-axis, which is 320 (our frame size is 640x480 pixels, therefore the halfway point on the x-axis would be  $640 / 2 = 320$ ). This method allows for the control to react to all kinds of changes in motion, whether the object moves away from the centre of the frame regardless of its speed. After all the calculations, the error value is found and added to the left and right motors to give motor speeds. This calculated value for the left and right motor speed is then sent into the second PID control loop – the PID distance control. A third input, the depth from the camera to the object, is also added to this control method. The left and right motor speeds from the PID turning control, are used as ratio's to determine speed. The setpoint in this method is the maximum distance, a constant set to define how far away the object is allowed to travel from the transport. The position would be the depth outputted from the camera, identifying the distance from the camera to the OOI. In this method, the error value is only used when the proportional value is less than a predefined constant. This is because if the proportional value is low enough, it infers that the transport needs to make a steep turn, thus it does not need to move much in the z-axis (forward or backwards). If this happens, the left and right motor speeds are multiplied by another predefined constant, in order to make the transport turn faster. If the proportional constant is high enough, then that means the object is far away for the transport to be able to drive forwards, therefore the error value calculated is multiplied by the left and right ratio received by the PID turning control method, and then added to the left and right motors to produce the correct motor speeds. The following algorithm forms the basis of the two stage PID control methods:

#### PID Turn:

*Initialization:* Set the setpoint variable to the tracked objects ROI's x-position (Equation (3.35))

*Step 1:* Set the  $K_P$ ,  $K_I$ ,  $K_D$ , and  $max\_speed$  constants

*Step 2:* Find the proportional value (Equation (3.36)), where  $position = FRAME_{WIDTH}/2$ ,

*Step 3:* Calculate the integral, derivative, prev\_proportional, and error values using Equation (3.30) – Equation (3.33).

*Step 4:* Calculate the left and right motor speeds using Equation (3.36) – Equation (3.37) if error\_value is  $< 0$ , and Equation (3.38) – Equation (3.39) if error\_value  $> 0$ .

$$setpoint = x_{ROI} \quad (3.35)$$

$$right\_speed = max\_speed + error\_value \quad (3.36)$$

$$left\_speed = max\_speed \quad (3.37)$$

$$left\_speed = max\_speed - error\_value \quad (3.38)$$

$$right\_speed = max\_speed \quad (3.39)$$

#### PID Distance:

*Initialization:* Set the  $max\_distance$  constant. Equate the setpoint variable to this constant

*Step 1:* Set the  $K_P$ ,  $K_I$ ,  $K_D$ ,  $turning\_distance$ ,  $offset$  and  $max\_speed$  constants

*Step 2:* Normalize the left and right speeds calculated from PID Turn using Equation (3.40) and Equation (3.41) respectively

*Step 3:* Find the proportional value (Equation (3.29)), where  $position = depth_{obj}$

*Step 4:* Calculate the integral, derivative, prev\_proportional, and error values using Equation (3.30) – Equation (3.33).

*Step 5:* If the proportional value is less than the *turning\_distance* constant, and the left ratio is less than the right ratio, then calculate the left and right motor speeds using Equation (3.42) and Equation (3.43). If the left ratio is greater than the right ratio, then use Equation (3.44) and Equation (3.45) to find the left and right motor speeds respectively. If the proportional value is greater than the *turning\_distance* constant, then calculate the left and right speeds by using Equation (3.46) and Equation (3.47) respectively.

$$left_{ratio} = \frac{leftspeed_{PID-TURN}}{max\_speed} \quad (3.40)$$

$$right_{ratio} = \frac{rightspeed_{PID-TURN}}{max\_speed} \quad (3.41)$$

$$left_{speed} = offset * right_{ratio} \quad (3.42)$$

$$right_{speed} = -offset * right_{ratio} \quad (3.43)$$

$$left_{speed} = -offset * left_{ratio} \quad (3.44)$$

$$right_{speed} = offset * left_{ratio} \quad (3.45)$$

$$left_{speed} = error_{value} * left_{ratio} \quad (3.46)$$

$$right_{speed} = error_{value} * right_{ratio} \quad (3.47)$$

Appendix D shows a complete flowchart for following an object or person.

### **3.4.3 Gesturing Engine**

A big part of SIGHT's features are its ability's to interact with the user. This is done through gesturing with the system. The range of gestures that can be used with SIGHT is limited to any movements of any body parts. If you can move a body part, you can essentially create a gesture. Typical gestures are waving a hand, extending an arm, using your hand in a certain motion like a clicking motion. These gestures could be extended to practically anything that is obvious and uses your body parts. For example, a gesture could be created for putting your right hand on your head. Using gestures, you can do practically anything that you normally do with a computer. Common examples of gesture systems include using your face as a mouse and your hands and fingers as a keyboard. In SIGHT, these gestures can be programmed to the intent of the user. By default, a remote control audio player is implemented. When the user would like to start playing a playlist, he or she simply has to face the wheelchair, and wave at it in order to unlock the gesturing engine. Once unlocked, the user can then start their playlist by extending their arm to the top right, above their head. By extending the right hand from the belly button to as far as the hand goes right (at least past the right shoulder), the track can be skipped forward. Similarly, extending the left hand from the belly button to past the left shoulder, will skip the playlist backwards. Pausing and playing can be controlled by extending the left arm to the top left, above the head. Volume control can be adjusted also. To unlock the volume control, the user must "click" the screen. That means he or she must push their hand from their chest outwards to the screen. Once unlocked, the volume can be tuned by moving their hand to the bottom right, besides the waist, and slowly moving it upwards (to raise volume) and back downwards (to lower volume).



Controlling music is just one of many possible applications that can be used with SIGHT. Other possible applications could be starting up a video or audio chat, playing video files, browsing the internet, or reading documents.

### **3.5 Physical and Output Layer**

The PID controller sends the appropriate motor speeds to the microcontroller, via a usb-serial line. The microcontroller used in SIGHT is a Teensy board [166]. The usb-serial talks to the microcontroller at a baud rate of 19200 bits per second. The motors of the wheelchair are connected to the microcontroller via a motor controller board made by Dimension Engineering [167]. A joystick is attached to the wheelchair for manual control. The outputs from the joystick are also connected via serial to the microcontroller.

## **Chapter 4. Experiments and Results**

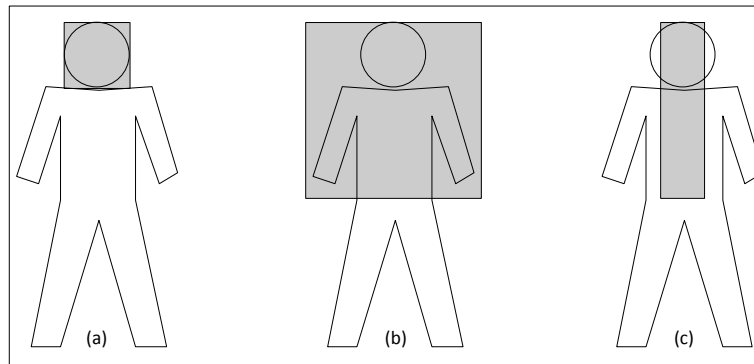
This chapter presents and discusses the experiments that were run as well as the obtained results for the methodologies presented in the previous chapter. In particular, experiments demonstrating the strengths and weaknesses of the person-following algorithm will be presented.

### **4.1 ROI Selection**

In order to track a person of interest, a region of interest must first be set. Early on experiments were run to determine how to best select an ROI. It would make most sense to select a region from the start of the persons head, to about their belly button. This is because visually, the characteristics between these two points are the most unique compared to other persons. The shape of a person's head and shoulders can differ dramatically from person to person. The centre point of the region should be around the mid chest area of the person. This is because the centre point of the ROI is where the distance from the camera to the person is calculated. Early tests showed that selecting only the person's head and neck as the ROI proved insufficient, because although there were numerous distinctive features that worked well for tracking, it was too small an area for the distance to be consistently found. What kept happening was that the centre point was found around the neck, and because the neck is narrower than the face, the centre point kept being found to the left and the right of the neck, effectively meaning that the centre point was not on the person itself, but behind the person. This led to the distances being significantly incorrect. Using just the person's face as the ROI can work, however as the person increases walking speed and moves further away from the transport, the region around the face gets smaller and smaller. Eventually the ROI gets too small, and features from the face are no longer found, thus leading to incorrect tracking and distances being found. Selecting a region

from the top of the head, to about the belly button proved the best for tracking. This is because the centre point of this region rarely gets “misplaced”, meaning it is always on the person. Using this region, the centre point is normally around the mid chest area. Using a narrower region of interest, but keeping the height the same (from the head to the belly button), works even better. This region starts at the top of the head, roughly where the person’s left eye is, and continues down to the belly button. The reason this region works better is because the background doesn’t influence the features found on the person. When using the head-shoulders-belly button region, there can be numerous features found around the persons head, which can completely throw off tracking. Figure 4.1 shows how the various types of ROI selection would look like. In Figure 4.1(a) although the ROI envelops the entire area of the head, it is very small in comparison to the rest of the body, and therefore is very small in comparison to the rest of the background as well. As the person moves, this already small region can get even smaller, and the tracking can easily be skewed and misplaced. Figure 4.1(b) shows what would be an obvious choice for a ROI selection, however it also shows the pitfalls of choosing this region. Notice how much space is around the frame. The software will not differentiate this space from the person itself, therefore whatever is behind the person at the time of the ROI selection will be thought of as part of the person themselves. That means if there is an object that can be seen behind the person in the first frame (where the ROI is selected), then that same object will always be expected to be behind the person across future frames. This will probably not be the case, therefore this choice of ROI selection is not the best. Figure 4.1(c) shows the most ideal way to choose an ROI. By selecting a narrower region of interest from the head to the belly button, there is a minimal chance of misdetection happening in future frames. Whatever is seen in this ROI will be seen across future

frames as well – the background, or objects around the person in the first frame, will have no impact on tracking the person through future frames.



**Figure 4.1 – ROI selections: (a) choosing just the head, (b) choosing the top frame, (c) ideal ROI**

The figures in Table 1 shows what the software actually sees for the three types of ROI's. Notice how in selecting the head and shoulders as an ROI results in many false interest points (besides the head) for both the optical flow features and the particles. The features found using just the head as the ROI results in many positive points, but a very small section of the entire frame (only about 2% of the entire frame). The ideal ROI choice results in all positive detections and points.

Before setting the ROI, the POI must stand at a social distance (see Chapter 3) perpendicularly away from the centre of the camera, somewhere between 1.22 – 3.66 metres. This distance will be maintained by SIGHT throughout the life of the tracking run. The reason for standing perpendicular to the centre of the camera is because for SIGHT to properly follow the POI, it must alter its course based on how far off centre the POI is to itself. For example, if the POI is 0.5 (m) to the left, SIGHT will output commands to the motors to swing left at an appropriate speed, so the POI is at the centre of the frame.










ROI TYPE	ROI	O.F FEATURES	P.F PARTICLES
Head + Shoulders			
Head			
Ideal ROI			

Table 1 - Types of ROI Selections

After standing at an appropriate distance from the chair, the user can set an ROI using one of two ways – either manually or interactively. The main difference between choosing a ROI manually versus interactively is that the manual method allows for the size and position of the ROI to change, whereas by choosing a ROI interactively, the size and position of the ROI remains fixed. At a frame size of 640x480 pixels, the fixed ROI size is 56x215 pixels, and is always at a fixed point of (292, 220). The x-coordinate of the fixed ROI was found using Equation (4.1), and the y-coordinate was used because at the 215 pixel mark, the POI was always comfortably in this fixed ROI, while standing somewhere in the social distance bounds of 1.22 – 3.66 (m). To give the user a better idea of where to stand, this fixed ROI was always displayed on the screen. The user could then adjust his or her position so that they were found in the ROI. When the user felt

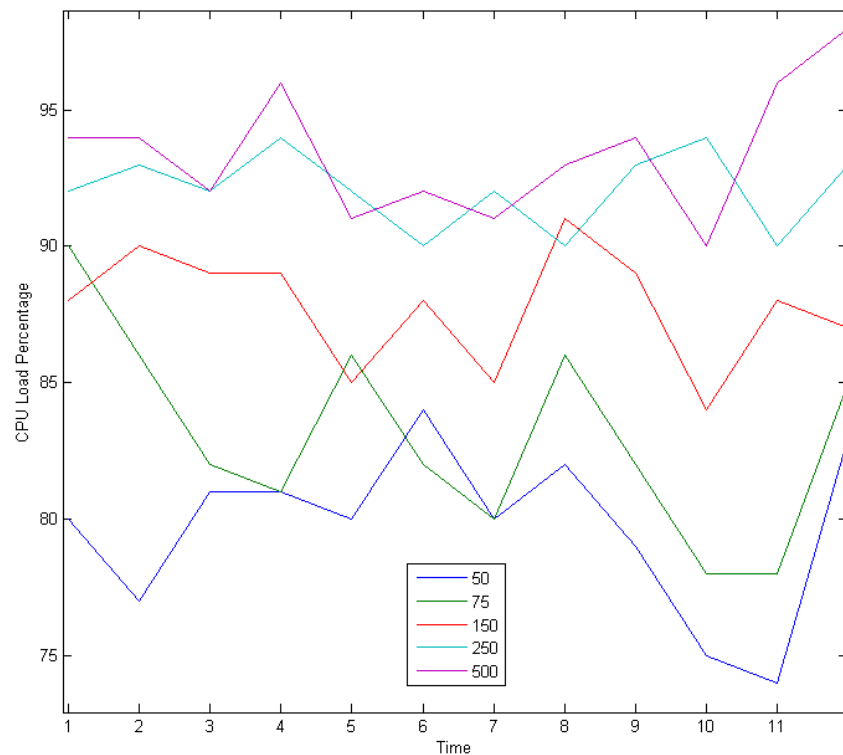
that they were properly identified in the ROI, he or she could then set the ROI by waving their hand to the camera. This would initiate the start of the system.

$$x_{ROI} = \frac{width_{FRAME}}{2} - \frac{width_{ROI}}{2} \quad (4.1)$$

If on the other hand, a manual ROI selection was preferred, then the user manually creates a region of interest on a still frame. An example of where this would be used would be if the user wanted the chair to follow another person, while the user themselves sat on the chair. The person would be instructed to stand in front of the chair, at an appropriate social distance. The user would then be able to load the next frame from the camera, and when happy with the frame, select a region of interest on that person. Pressing enter, or “continue” on the screen, would initiate the system. Regardless of which method is chosen, manual or interactive, the person to be followed must stand at an appropriate social distance away from the centre of the wheelchair. The ideal ROI size (Figure 4.1c) should be used in all cases for following to be the most robust.

SIGHT allows for the user to adjust the number of corners and particles as they see fit. These are set at execution time as program arguments. As the number of corners increases, the number of matches that the optical flow method will try to match will increase. Since the ROI is small compared to the size of the frame, (at the fixed ROI size, it is about 4% of the entire frame), setting the number of corners too high is pointless, as the corners found will eventually just overlap other existing corners, thus skewing future tracking. However increasing the number of particles used can be beneficial, as the higher the number of particles are, the higher chance the particle filter matching method has of finding the POI or OOI. The upper limit of the number of particles really depends on the computing system being used. Since the histogram of each particle must be found, there is a direct correlation between the computing power and the number

of particles: as the number of particles increases, so does requirements for the computing power. For the computing system used in this paper, the Fujitsu TH700, the maximum number of particles that could be used without slowing down the tracking system was found to be 75 particles. Figure 4.2 shows a graphical representation of how increasing the number of particles correspondingly increases the CPU load usage. This graph shows a 12 second run of the program, using the same fixed ROI size of 56x215 pixels, for 5 different sets of particles: 50, 75, 150, 250, and 500 (optical flow usage was turned off for this test). Using 75 particles, an average CPU load of 83% was found. This is high enough that it doesn't cause instability in the tracking system. Using a number of particles higher than 100 caused an average CPU load of 90% and greater, and caused instability in tracking.



**Figure 4.2 - A representation of the CPU load percentage vs. time for various numbers of particles**

## 4.2 Data Logging

In order to run any experiments, data had to be properly collected and analyzed. The Open Sound Control (*OSC*) is a type of messaging protocol widely used in the music and digital industry. It was modified to use with the liblo [168] implementation in order to gather data. What is great about OSC is that messages can be sent across the network (or even through the localhost) to be dealt with by another process. SIGHT uses this method to log data. Using OSC, messages about what is happening is sent through the localhost, port 127.0.0.1, to another program which accepts the messages, and saves them to a file. This is important because it makes sure the process running the SIGHT algorithms are not affected by saving data to a file – there is a separate process running that takes care of this instead. So SIGHT can go on running, without errors due to writing to a file, or parsing messages. If there are errors, then the separate process will take care of them. A typical OSC message consists of a minimum 5 entities: a hostname, port, address, type, and value. The hostname specifies the remote hostname, the port is the port that the hostname is listening on, the address is where the message should be sent to, and the types can consist of the various primitive data types, such as integers (32 or 64 bit), chars, strings, boolean values, or even NIL values. An example of sending an OSC message to the localhost would be: `oscsend localhost 7777 /address f 0.1234`, where `f` is the identifier for a 32 bit floating point number, and `0.1234` is the value of that floating point number. The receiving end of the OSC message listens for what the address is, and sends the message to that specific address, in this case, the actual message would be the float value `0.1234`, and the address would be `/address`. The receiving end that is waiting at this address can take the value and do whatever is necessary with it. In SIGHT, there are several values which are important to use for data analysis. These values, at time  $t$ , are: depth (single float), the optical flow and particle filter



histogram values (two floats), the number of corners (single integer), how long (number of iterations) the POI was lost (single integer), and the motor values (two integers). Using OSC and liblo, these values are sent across the localhost via the addresses /depth, /hists ( $k$  for optical flow histogram, and  $p$  for particle filter histogram), /corners, /lost, and /motors ( $l$  for left, and  $r$  for right) respectively. A timestamp, in the form of “ticks”, or “iterations” is sent along with each message. Using these 5 categories made it easier to see why and where the system failed. Table 2 shows an example of these values look like, while being sent across two frames.

/depth	t	0	f	1.69		
/depth	t	1	f	1.698		
/hists	t	0	k	0.167579	p	0.176756
/hists	t	1	k	0.257731	p	0.189961
/corners	t	0	92			
/corners	t	1	82			
/lost	t	0	0			
/lost	t	1	0			
/motors	t	0	l	62	r	193
/motors	t	1	l	64	r	191

Table 2 - OSC messages used in SIGHT

The receiving end listens for OSC messages, sends them to the correct function for the address, and each respective function parses the message and outputs the result to a file. The results of the graphs found in the rest of this chapter are put together from the sorted output of this file.

### 4.3 Static Tracking

Tracking an object while remaining in a fixed or static position is very different than tracking an object while moving positions. This is due to the background changing scenes from frame to frame. Before running trials to see how well the combined particle filter and optical flow

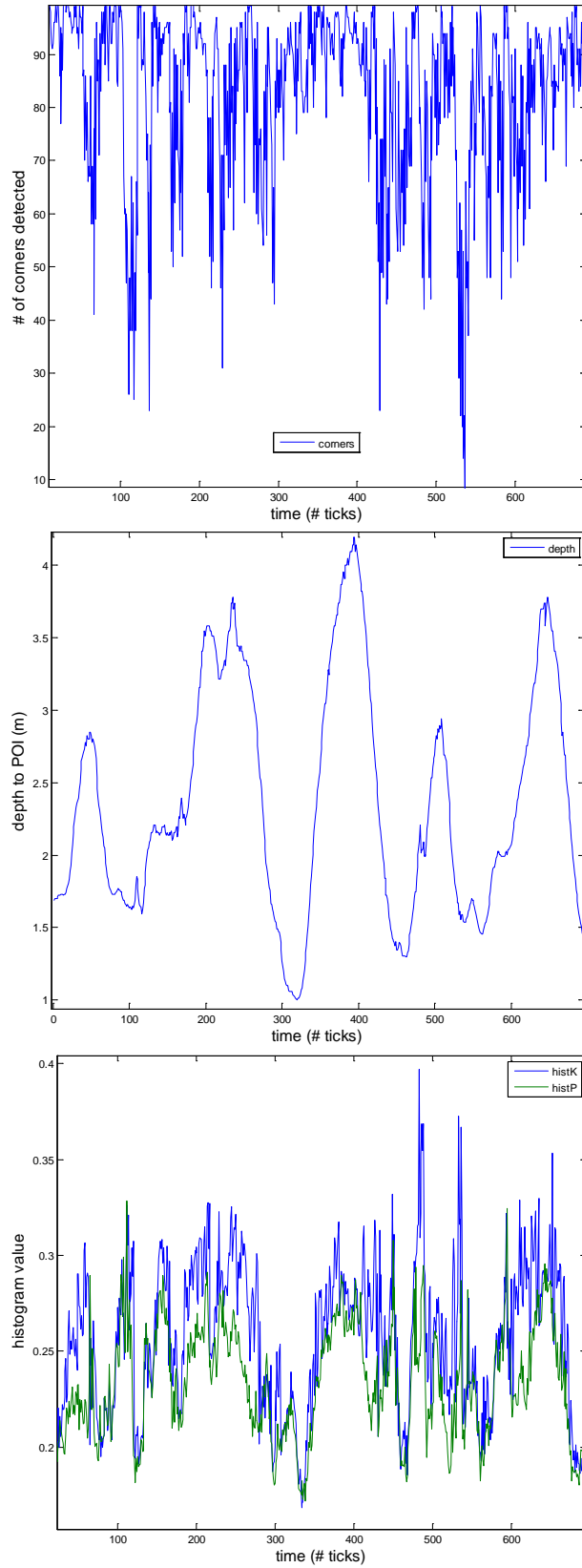
tracking algorithm work in a dynamic environment, it was important to see how it tracks while remaining static. Experiments were run in the NCART computer science lab at Ryerson University to test how well the system tracked a person while in a fixed position.



**Figure 4.3 - NCART Lab**

Figure 4.3 shows how the lab looks like from the point of view of the wheelchair. This area was chosen in particular because of how cluttered the background is. Since there are many objects and various colors, it was an ideal area to test static tracking. In this experiment, a user selected themselves as an ROI using the manual selection method, and then walked in this area in various directions, to as far back as the red box on the floor (to the left of the red chair). The area used was measured to be roughly 4x2 metres. Using just optical flow and particle filters by themselves (without combining), led to many misdetections, with optical flow faring worse than particle filters. Using the same stationary position, experiments were conducted for each method. The misdetections were not found by the FAST-BRIEF lost-object algorithm because the histograms for the optical flow and particle filter positions did not rise above the threshold for losing an object. Due to the fact that optical flow relies on movement, it makes sense that it would fare worse than particle filters. On average, during three experiments roughly 70 seconds

long, using just the optical flow method misdetected the person four times, whereas the particle filter method misdetected the person twice. However combining the two methods together, using parameters of 75 particles and 100 corners, gave near perfect results. The initial distance (the distance from the camera to the person at the time of the ROI selection) was roughly 1.6 m. Each run showed that the person was adequately tracked across frames for the entire time without being lost once. Figure 4.4 shows a graphical representation of one of the runs. The corners found throughout the 70 second run (roughly 700 ticks) had an average value of 82 corners (out of a maximum 100), while the distances logged matched the distances travelled throughout the experiment. The histogram values show the person was tracked very well through the run. The optical flow histogram (histK) had an average of 0.2564, whereas the particle flow histogram (histP), had an average of 0.2335. This is a very good number, as the lower the histogram values are, the more accurate the match is. Analyzing this graph further, we see a steep decline for the corner values around the 530<sup>th</sup> tick. The POI was not lost however, due to the fact that the histogram from the position returned by the particle filter algorithm outweighed the position of the optical flow algorithm. Due to the difference between the histograms being over a threshold of 0.05, the particle filter's histogram, and thus particle filters position, was used and the histogram and position found by the optical flow method was discarded. This allowed the SIGHT algorithm to continue tracking the person at the correct position. This can be seen in the graph by the histP values – as the number of corners dipped, the histK values went up, but the histP values remained low, thus showing that the person was still correctly being tracked.



**Figure 4.4 - Corners, Depth, and Histogram vs. Time in a stationary position**

In reality, there are many objects and obstacles that can exist in an environment where following an object or person occurs. These objects and obstacles can cause the object to be lost by the target tracker. Most tracking algorithms do not allow for re-finding an object after it is lost – once it is lost, either the software has to be reset, or manual interaction or influence is required. One of SIGHT's main features is its ability to find the target object or person by itself, without any external interaction. Using FAST and BRIEF interest and feature points, SIGHT is able to find the object when it is lost. Testing this feature in a static environment was necessary to see how well it re-found a lost object. In this experiment, SIGHT was started up with 75 particles, and 100 corners, and the ROI was selected on the POI roughly 1.6 m away from the centre of the camera. The area used was the same static environment as above. In the first run of the experiment, after initialization the POI left the scene on the left, and entered the scene again from the right. The results showed that upon leaving the scene (on the left), the POI is no longer tracked, and the FAST-BRIEF algorithm starts. A few seconds later the POI enters the scene (on the right) and the algorithm immediately finds the POI again. This run ran for about 21 seconds, and the POI was originally at a distance of 1.75 m from the camera. Figure 4.6 shows a graphical representation of this run. You can see from the graphs that around the 27<sup>th</sup> iteration, SIGHT figured out that the object was no longer in the frame, and thus labeled it lost. Comparing this frame across graphs for corners, depth, and histograms, you can see that the object was indeed lost. The number of corners and the depth detected at that timeframe dropped to zero, and the histogram values for the optical flow and particle filters climbed above the histogram threshold limit. In the graph you can also see that the FAST-BRIEF algorithm spent roughly 60 iterations searching for the object. When the POI was back in the frame, a few seconds later, all three parameter types continued to track the object as if nothing was changed.

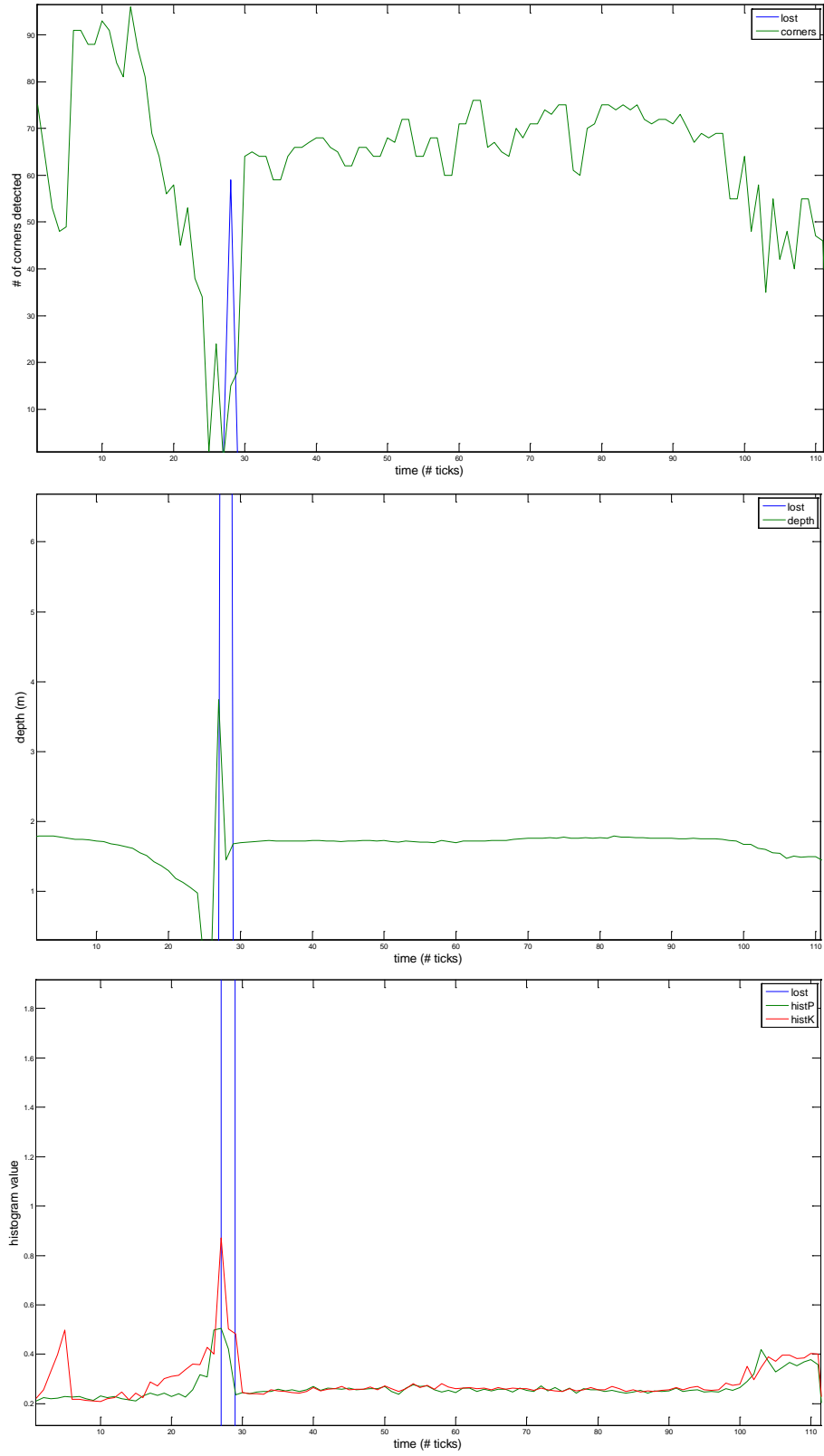
In this run, the object returned to roughly the same depth as where it was lost – you can see this from the depth vs. time graph. As soon as the person was re-found, the depth continued around the 1.8 m mark. The same goes for the histogram and the corner values. Another run tested for re-finding a POI after another object came in between the camera and the POI, so that the POI was completely blocked (occluded) from view. In this run, the same stationary location was used except the wheelchair was centered more, and a chair was placed in the middle of the hallway (Figure 4.5).



**Figure 4.5 - View used for re-finding blocked objects**

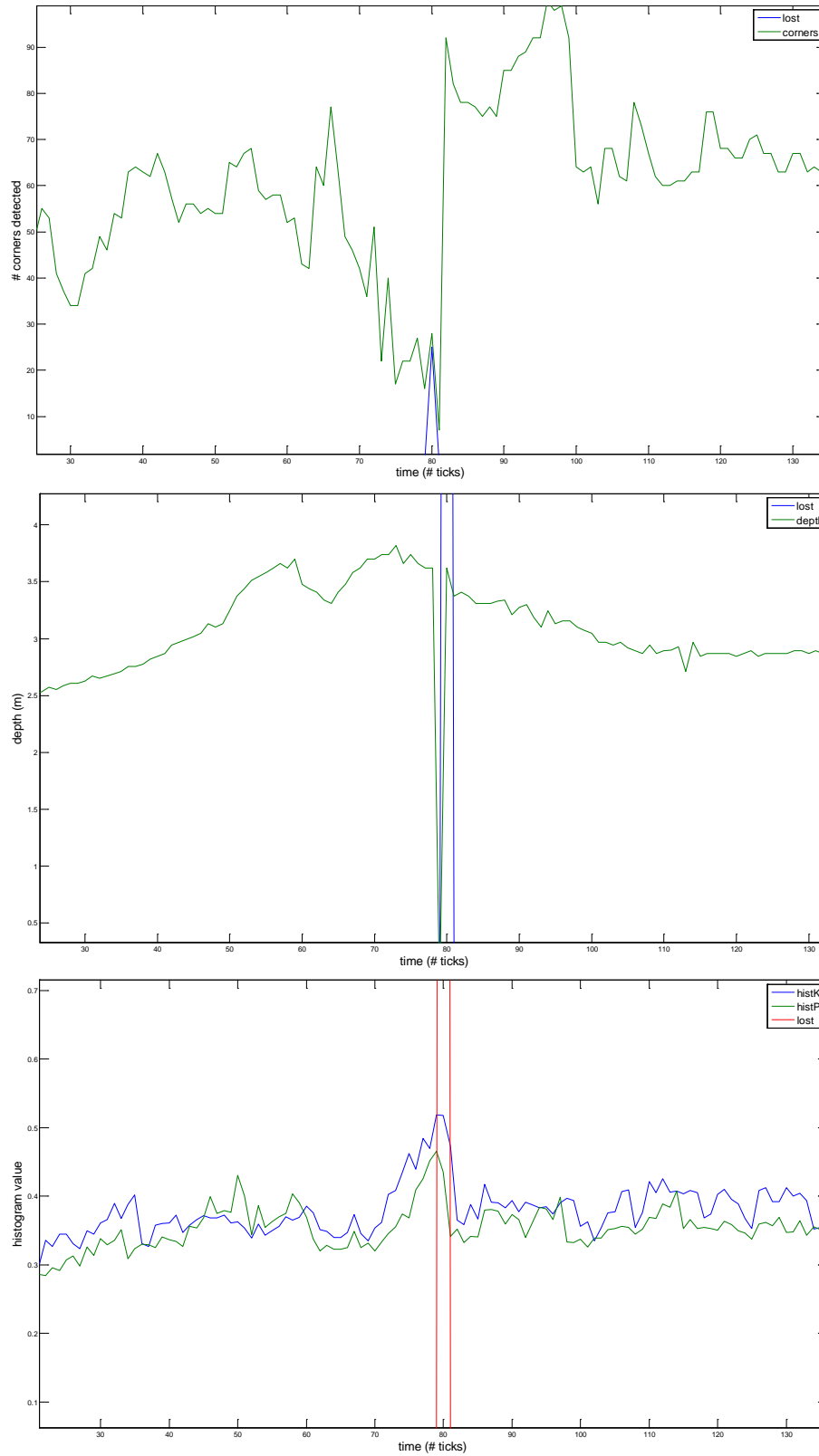
The blue chair placed in the middle of the hall was set about 4 m from the camera. The chair was used as an object that blocked the POI from the view of the camera. This experiment required the POI initialize the software by setting the ROI on them, and then walking behind the chair, and crouching behind the chair, so as to completely hide themselves from the camera, and then after a few seconds, stand back up so he/she was back in the view of the camera. The expected result was that the person would be deemed lost, and the FAST-BRIEF finder would start to look for, and then eventually find the person. In the first run, the POI initialized the system manually by setting the ROI while standing about 2.5 m away. Walking immediately to the left of the chair (from the camera's point of view), the POI then hid behind the chair for about 3-4 seconds. As expected, the person was lost by SIGHT and the FAST-BRIEF re-finding

algorithm started. Figure 4.7 shows a graphical representation of this run. Due to the nature of the BRIEF algorithm not being rotationally invariant, the person was not found right away as he stepped back in view of the camera. The person has to walk about 1 metre in front of the chair for the BRIEF algorithm to find his interest points, and for FAST to match them to the original feature points. The result of walking closer to the camera in order to be re-found was seen across repeated experiments – the POI, upon re-entering the scene after being “lost”, was only “found” after he/she presented themselves to the camera in the same fashion they were in when the reference image was taken. This means that if they were standing in a certain pose when the ROI was first set, they needed to stand in that same pose for the finding algorithm to work. If the POI even maintained a similar pose, and was a reasonable distance off the starting distance ( $< 1$  m from where the ROI was first set), the finding algorithm would eventually find the POI, but it would take longer than if the POI returned to its starting depth, and recreated its pose. In each experiment run, after losing the POI, returning to the original pose and distance resulted in 100% matches. To increase accuracy of re-finding a lost object, it is again important to correctly set a proper ROI. For the best chance of matching, an ROI should be selected on a surface where there are many features, i.e. letters, lines, markings, on a shirt, facial features. In all experiments the ROI position and size was selected like Figure 4.1c. In trials where the POI’s shirt was featureless, i.e. a mono colored shirt, the time it took to re-find the POI after being lost was significantly higher than when the POI’s shirt had features. The “ideal ROI” from Table 1, shows how many features could be found whilst wearing a shirt with many markings. Wearing just a mono colored shirt on the other hand, would result in features only being found on the face which decreases the chance of re-finding the object simply because of the size of the face compared to everything else in the frame.



**Figure 4.6 – Re-finding an object: corners, depth, and histograms, vs. time**





**Figure 4.7 - Re-finding an object after occlusion of the object occurs**

## 4.4 Dynamic Tracking and Following

Experiments run while the wheelchair was in a stationary position showed that SIGHT's tracking algorithm worked well when combining particle filters and optical flow, along with a re-finding method. The next step was to run tests to see how well it maintained tracking while it was moving, thus how well tracking was preserved given that both the background, and the OOI/POI kept changing. The experiments run in this stage were also used to tune the PID controller so that following a person or object was viable.

Before the experiments started, the wheelchair motor speeds were capped at about 1 m/s, and a safety RF remote emergency-off switch was installed. This made sure that in an emergency the wheelchair was able to be stopped right away. A "course" was then set up outside the NCART lab at Ryerson so that testing was uniform. The course consisted of a 32 metre track. Figure 4.8 shows the first half of the track. From 2 m in front of the black tape (bottom of the figure) to just in front of the door at the end of the hall, 5 metre intervals were marked, for a total of a 17 m stretch. At the 17 m mark, a turn was set up going left. From the end of this hall (in track 1), to the end of the hall in track 2, a 15 m track was set up (again in 5 metre intervals). Figure 4.9 shows what the hall looks like from the end of the second hall (facing the end of the first hall). Figure 4.10 shows what the complete track looks like from the top down. There were two primary types of experiments that were run on this course. The first experiment was a straight run, it used only track 1. The wheelchair would follow the person for 17 metres from the start to the finish, the program would be reset, and the wheelchair would follow the person from the 17 m mark back to the start. The second experiment used the complete track. The person and wheelchair would walk the first 17 metres, then make the left turn at the end of the first hall,

continue 15 metres till the end of the second hall, and then while in the same run, make a 360 degree turn, and continue back the way it came from – travelling 15 metres till the end of the first hall, turning right, and then walking 17 metres back to the beginning. Using the complete track resulted in about a 64 metre distance, with one left turn and one right turn.



Prior to running experiments on the full track, the PID controller was tuned by staying within the first 5 metres of the track. Using the data captured by the data logger, the tuning was made accurate so that the wheelchair was able to turn fast enough to keep up with the POI. It was decided that reversing the wheelchair was not a useful feature; instead the wheelchair had the capability of turning around in order to go the other way.

For SIGHT to be used in the real world, it must be able to follow a person's natural walking speed. There were three types of walking speeds that were used in testing, a fast, medium (average), and slow walking speed. Based on a study of pedestrian walking speeds [169], which found that the average walking speed for those younger than 65 was 1.25 (m/s), and 0.97 (m/s) for those over 65, the slow walking speed metric was set to about 0.97 (m/s), the medium to 1.25

m/s, and the fast to 1.5 (m/s). Using these values gave a reasonable idea of how fast different age groups walk.

For the first set of experiments, the straight path (from the start to the 17 (m) mark and back), was used. For each walking speed, the POI initialized the ROI, and then walked straight to the 17 (m) mark. At the 17 (m) mark, the POI would stop and restart the system, reselect a new POI, and go back from the 17 (m) mark to the start. Due to a change in the lighting conditions around the 14 (m) mark, SIGHT lost the POI once out of six times. However in the other five runs, the system tracked and followed the POI perfectly. Table 3 shows more detail of the six runs. The depth is given as a metric as to give the reader an idea of the speed of the wheelchair – the further from the target the faster the speed of the wheelchair, and vice versa – the closer the target, the slower the wheelchair.

<b>SPEED</b>	<b>Start -17m POI (sec)</b>	<b>17m - start POI (sec)</b>	<b>Start -17m WC (# ticks)</b>	<b>17m - start WC (#ticks)</b>	<b>Figure</b>
<b>Fast</b>	12.4	13.0	~300	~200	Figure 4.11, Figure 4.12
<b>Medium</b>	15.8	16.0	~240	~220	Figure 4.13, Figure 4.14
<b>Slow</b>	22.3	21.2	~290	~280	Figure 4.15, Figure 4.16

**Table 3 - Track 1 Testing**

Following the POI on a straight course worked well. Even in the first case where the POI was lost, once the POI returned back in the view of the camera, it was re-found by the finding algorithm and following continued. Based on the data, it is obvious that the wheelchair was able to keep up with the POI, more so in the medium and slow speed cases. From the graphs, it can be seen that the initial acceleration was the greatest change of speed – this acceleration was necessary so the wheelchair could maintain the OSP distance throughout the run. After this

initial jump of speed, the wheelchair kept its speed and distance relatively still, until the POI reached the end, at which time the wheelchair decelerated appropriately until it was a safe distance from the POI. Notice how in the fast speed case the speed was steep for a longer period of time compared to the medium and slow cases. This makes sense, as the wheelchair had to move faster in order to maintain the OSP distance. The graph for the slow speed infers that the best tracking occurred here. This makes sense - as the chair is moving slower, the camera is moving less, and therefore the optical flow and particle filter algorithms have more accurate readings. The results from this experiment showed that even though the background is changing, the combined tracking algorithms that SIGHT uses, still works.

The next experiment made use of the entire track, from the start to the 32 (m) mark, and back to the start. As mentioned above, there are two turns involved, one left turn at the 17 (m) mark, and a right turn coming back at the same corner. This experiment was interesting because it tested SIGHT's capabilities in a more real-world environment. In the second track, the lighting conditions really showed the drawbacks of this system. The quick changes in lighting threw off the camera data, thus throwing off the tracking scheme. The FAST-BRIEF finding algorithm is not able to re-find the object when lost, because the lighting conditions changed so drastically – the features and interest points that were stored about the POI were no longer applicable with the new light. However once the chair was moved away from the light into an environment which better suited the original ROI lighting, the object was re-found and the tracking was continued. Of the three runs that were used in the complete track, only one run made it the entire way. Both the other runs (one going at a fast speed, and one at a slow speed), got lost at the end of the second hall. The main reason for it getting lost, besides the change in lighting, was due to the

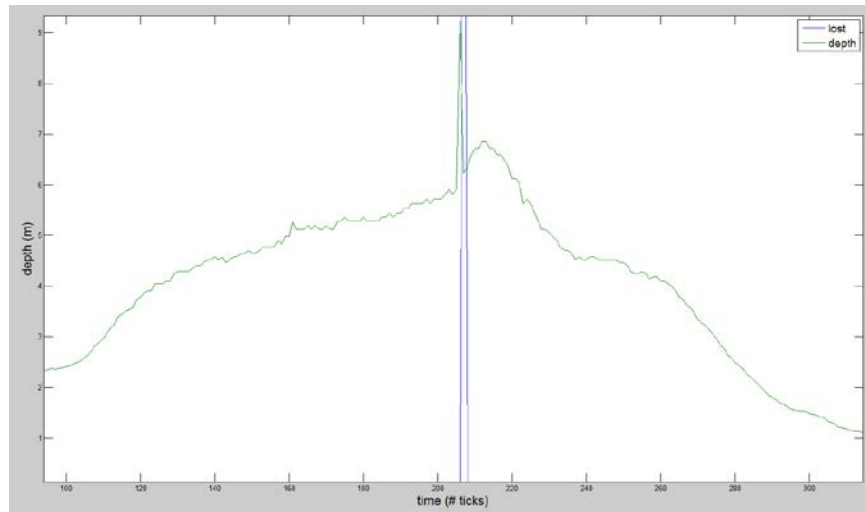
person slowing down too much at the end and then making the turn around to head back. Because of the slowdown and the turn, the conditions for tracking deteriorated and the object was lost. In the one run that worked well (at a medium speed), the speed was continued throughout the entire run, including the end of the second hall and the turn. Because the speed was continuous, the lighting condition did not play as big a role as it did in the other two runs. Table 4 shows the metrics from this test. You can tell from the graph (Figure 4.17) for the medium run, that around the 400<sup>th</sup> tick, the POI was making the turn at the 32 m mark in order to come back. Figure 4.18 shows one of the runs that got lost at the end of the hall, just before the 180 degree turn back. Figure 4.19 shows the histogram values recorded on that run, you can see how at the end of the second hall, the values suddenly increased – showing that even though the object was in front of the wheelchair, the lighting conditions made it seem like it wasn't.

<b>SPEED</b>	<b>Start – 64m Finish POI (sec)</b>	<b>Start - 64m Finish WC (# ticks)</b>	<b>Figure</b>
<b>Medium</b>	75.0	~600	Figure 4.17

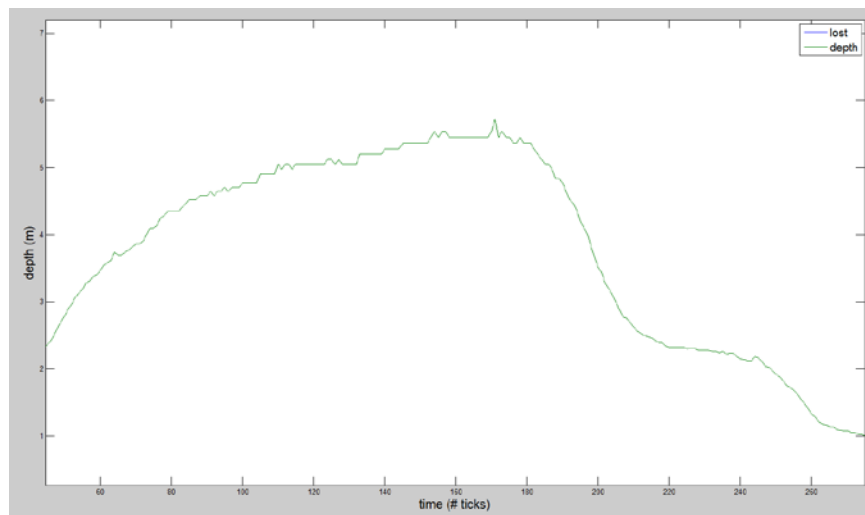
**Table 4 - Complete Track testing**

The experiments that were run showed us that lighting plays a vital role in SIGHT's ability to track and follow a OOI/POI. In conditions where the entire route of the POI's path has stable lighting, SIGHT was able to track with a very low "lost" rate. However, in a route where the lighting changes were volatile, SIGHT did not perform as well. From the experiments run, it can be seen that travelling at a speed of 1.25 (m/s) or slower, resulted in the best tracking. Increasing the speed to about 1.5 (m/s) and higher does have a higher "lost" rate. The FAST-BRIEF finding algorithm does work well when the lighting conditions do not change much from the

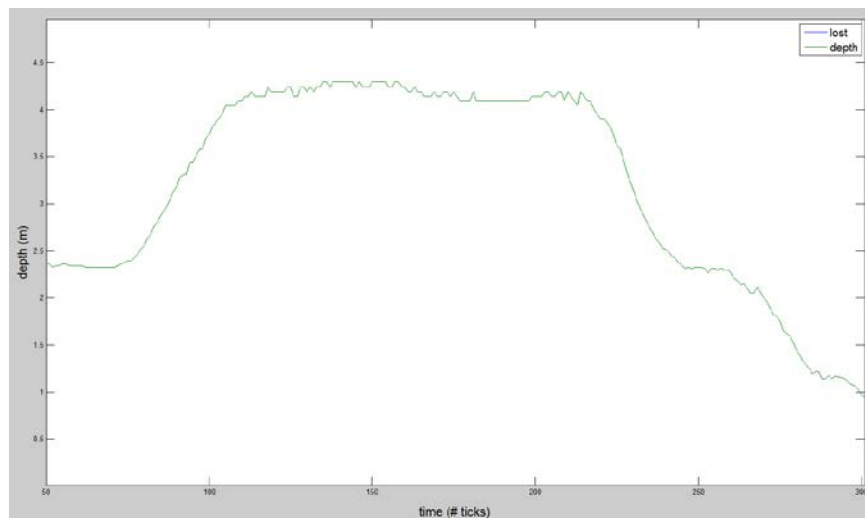
environment where the ROI was taken, and when the POI is able to maintain a similar pose to the original ROI.



**Figure 4.11 - Start to 17m, fast speed**

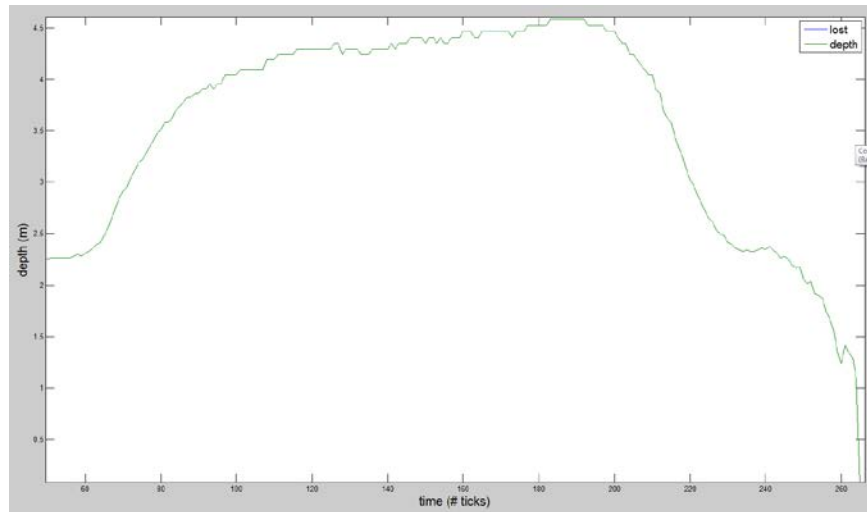


**Figure 4.12 - 17m to start, fast speed**

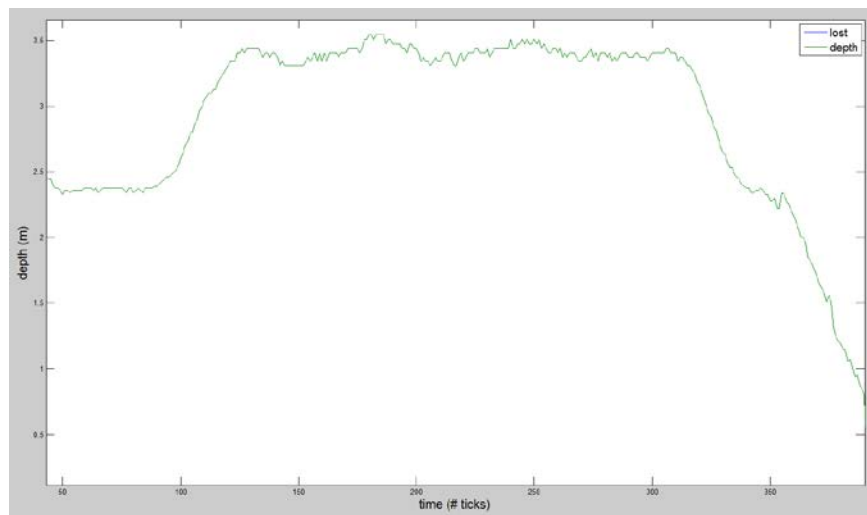


**Figure 4.13 - Start to 17m, medium speed**

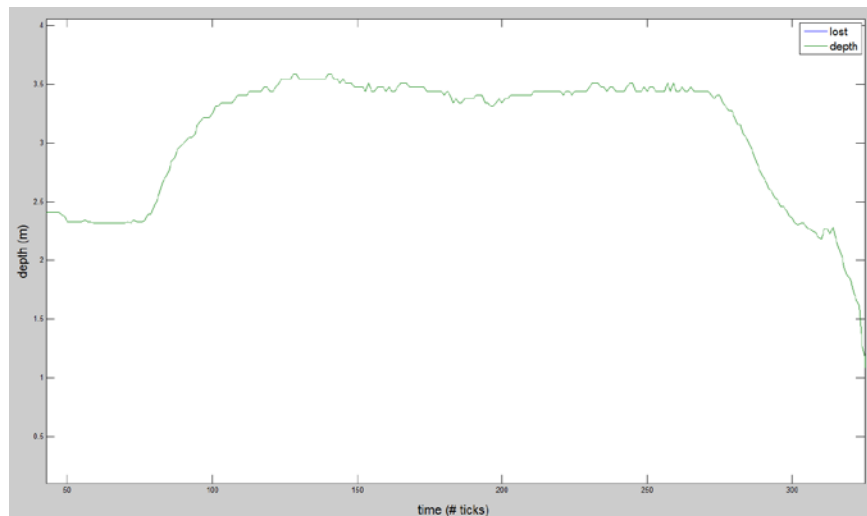




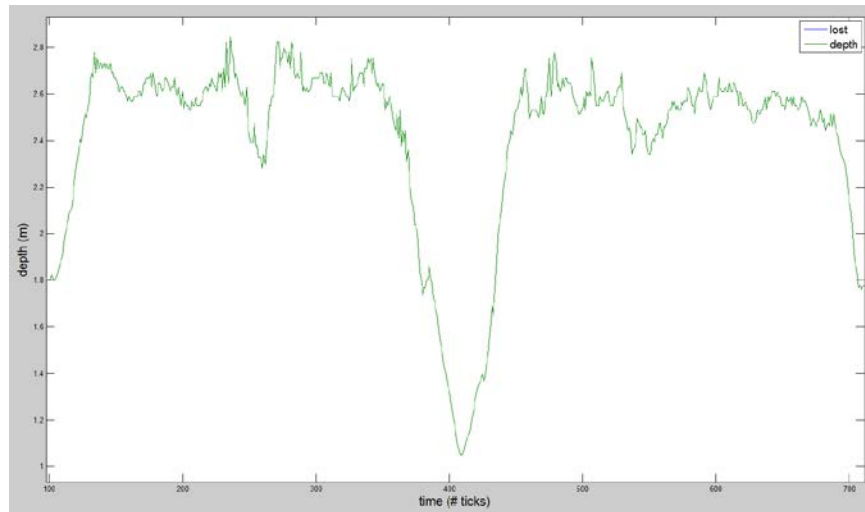
**Figure 4.14 - 17m to start, medium speed**



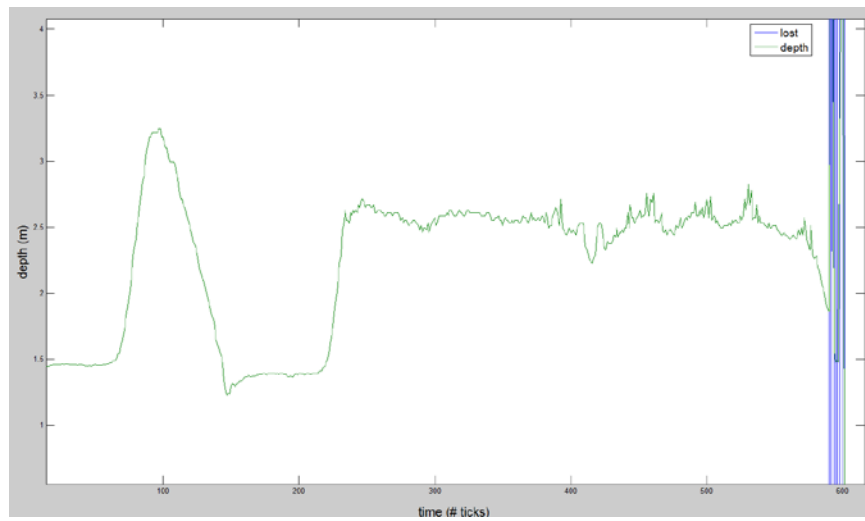
**Figure 4.15 - Start to 17m, slow speed**



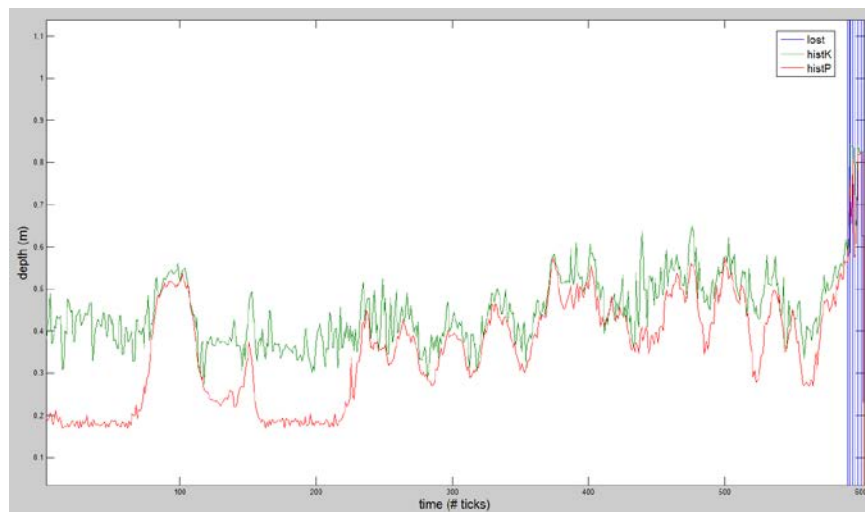
**Figure 4.16 - 17m to start, slow speed**



**Figure 4.17 - Start to finish, medium speed**



**Figure 4.18 - Start to finish - lost at the 32 m mark: depth values**



**Figure 4.19 - Start to finish – lost at the 32 m mark: histogram values**

## Chapter 5. Conclusion and Future Work

This thesis had a two-part objective – to demonstrate that it was possible to modify an existing wheelchair to allow it to follow a person and their path through a human-occupied environment, and to demonstrate social practicality for such a system. We created a system called SIGHT, a Socially Interactive, Gesture-aware, Human-following, Transport. What makes SIGHT special is that by using cheap off the shelf components, we were able to transform an electric wheelchair into a person following transport, for under \$300 (not including the laptop). Compare this to the robotic wheelchairs mentioned in the literature. Most of those had expensive equipment and were not able to run in real time.

The main contribution of this thesis is to demonstrate that two common tracking methods, particle filters and optical flow, could be successfully combined to form a more robust tracking method. When these tracking methods failed and the object was deemed lost, SIGHT's use of the FAST and BRIEF algorithms allowed it to re-search and possibly re-find the object.

Social interaction, in a general sense, is the way people communicate and correspond with one another. It can be defined as actions taken by an individual that relate to the behavior of others, and therefore dictate the individual's further reactions and directions [12]. SIGHT is social in the sense that it uses gestures and coordination between the robot and its (human) user [11]. Because of the gestures, a channel of communication is open between the user and the chair. With this functionality, companionship can exist also between the user and the chair. This can lead to a sense of immediacy and intimacy, awareness, and connectedness that the user may feel towards the wheelchair. Immediacy, which is a measure of the psychological distance, such as

smiling and nodding, can also be detected by the gesture engine we implemented. The behaviours are used to create and maintain intimacy and social awareness. Connectedness is the emotional experience felt by another's presence. The chair, acting as a social outlet to the user, can help the user feel more connected.

## **5.1 Limitations**

Experiments were run to show how effective the SIGHT following system is. Its major weakness is tracking an object or person under variable lighting conditions. Due to the infrared nature of the camera being used in SIGHT, lighting changes from frame to frame could completely throw off the tracking, even though to the human eye, not much has changed between frames. Re-finding the object will also not work because of the light variations.

Another limitation is re-finding a lost object, regardless of the lighting. In most trials, when the object was lost, he or she had to return to a similar distance and pose as to how they were when the ROI was first set. Although in most cases, returning to a similar pose will result in a "find" by the algorithm, it can still be inconvenient for the POI to do so.

Speed was a third factor which restricted SIGHT's ability to follow a person. If the person walked too fast, especially at the start of the system, the chances of losing the person increased.

The fourth factor is the appearance of the person that is to be followed. If there are other people or objects that are similar to the POI's appearance (shirt color, features), SIGHT will become confused, and chances of losing the POI will increase.

A final limitation would be the colors of the POI. Because the system relies on the histogram values of what colors it is seeing, colors that blend in with the background will fare poorly compared to colors that stand out. The same goes with features on the person or object.

## 5.2 Advantages

Although limitations do exist, SIGHT can still work robustly in many scenarios. The target audience for SIGHT is seniors who rely on having a wheelchair nearby, but can still manage to walk themselves slowly. A senior's pace would be around 0.97 m/sec [169] – at this pace SIGHT would find it much easier to maintain following. Furthermore, SIGHT has the advantage of finding lost objects, a feature that many tracking robots do not have. It deals well with occlusion because of this. SIGHT also gives the user a stronger sense of control – whether the user wants to drive, or have the chair follow them, they can always be in control. The gesture system allows for this to happen. Although currently the gesture system is set up for a remote control media player, it can be modified to many other applications. Other following robots do not have this feature. Figure 5.1 shows how the wheelchair used in this thesis looks like with the attached camera.



Figure 5.1 - SIGHT, an autonomous people-following gesture aware robotic wheelchair

### 5.3 Future Research

The limitations mentioned above are a primary research focus for future work. Lighting conditions is an important factor that needs to be remedied. A better method for finding lost objects needs to be found. This method should be rotationally variant, but yet not much more computationally costly.

Since the primary target of this system are meant for seniors, more on board sensors on the wheelchair may be beneficial. Bodily measures such as heart rate, blood pressure, or even oxygen levels, could be monitored directly from the chair. The chair could then communicate autonomously with a hospital or health care worker. This would be useful in emergency cases, where the user experiences a medical problem, and is not able to call for help themselves. The chair will be able to detect a problem, and correspondingly call for help [170].

We would also like to see a brain-wheelchair relationship. This means that we would like the wheelchair to correspond to specific brain waves. I.e., the chair would follow you while you are in a relaxed state of mind, and would stop following you while you are in a more stressed state of mind. In fact, we have been awarded an NSERC grant in order to look more into this. We look forward to making this an actual feature.

What is probably the most important future feature is for the chair to be able to get from point A to point B based on a gesture or a social cue from the user. Instead of following the user, the chair could instead learn over time where certain destinations are, so in the future if the user gives a cue that they want to go to a certain place, the wheelchair could take that person there by

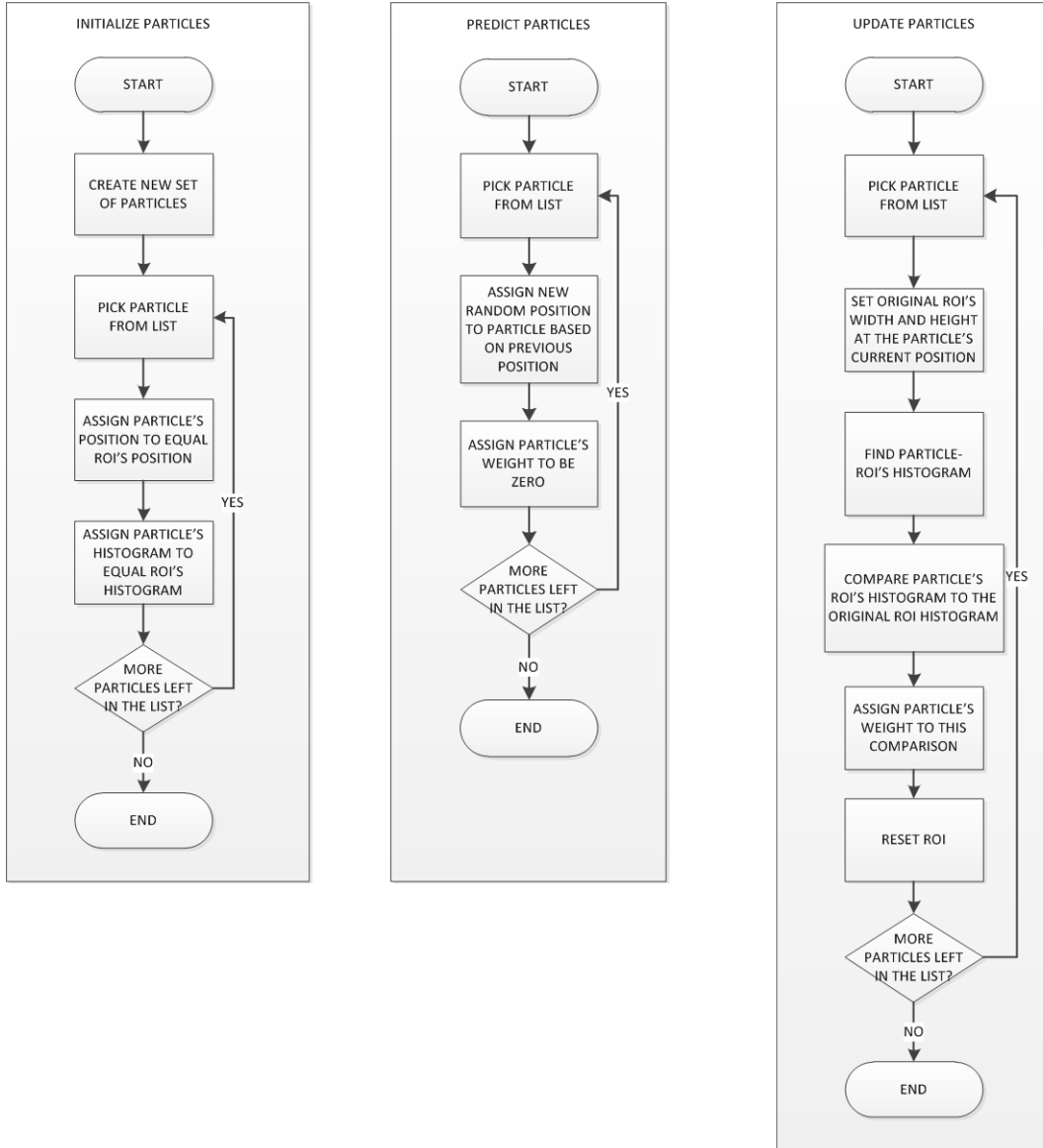
itself, without any direction or interference by the user. An example of this would be to learn where the kitchen or washroom is, and to take the user there upon receiving a signal that he or she wants to go there.

## **5.4 Concluding Remarks**

The hope of this thesis is to ease the lives of people who are dependent on transportation vehicles, whether they are dependent on the transports to get them from point A to point B, or simply if it they depend on them to help in their day to day lives. In particular, this thesis hopes that senior citizens who are alone and helpless can find comfort in this kind of technology. Our work was demonstrated at the Ontario Science Centre's "Robots Rule" event, in November 2011, and was also featured in an article in the "Torontoist" [171]. We are optimistic that this sort of technology is heading in the right direction, and look forward to this research being continued and improved on in the future.

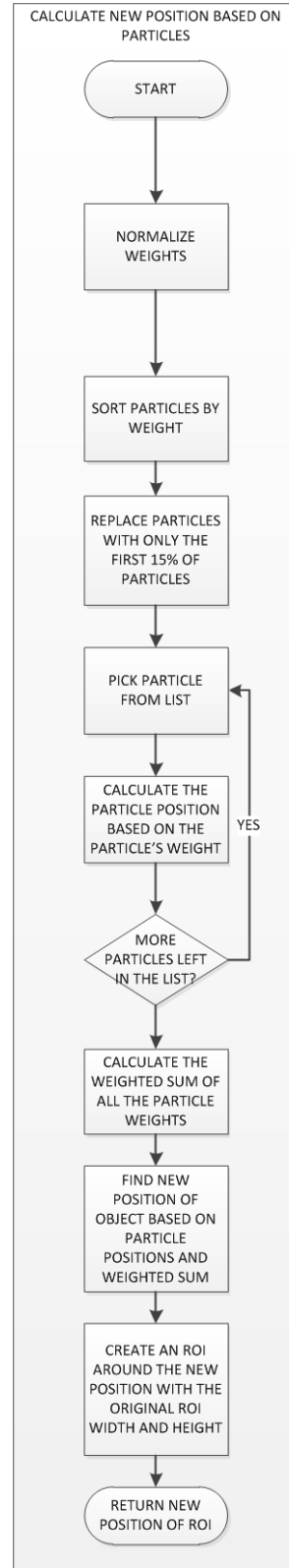
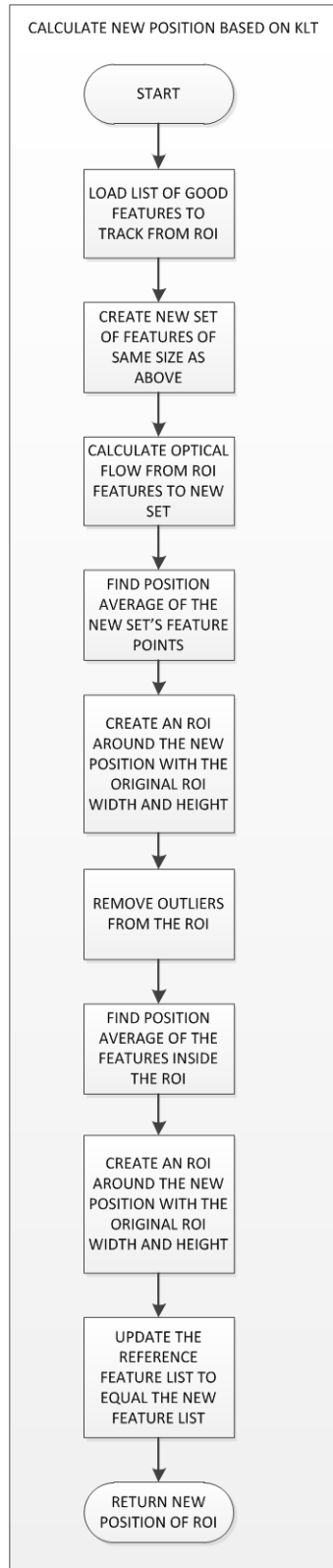
# Appendices

## A. Particle Filter Flowchart

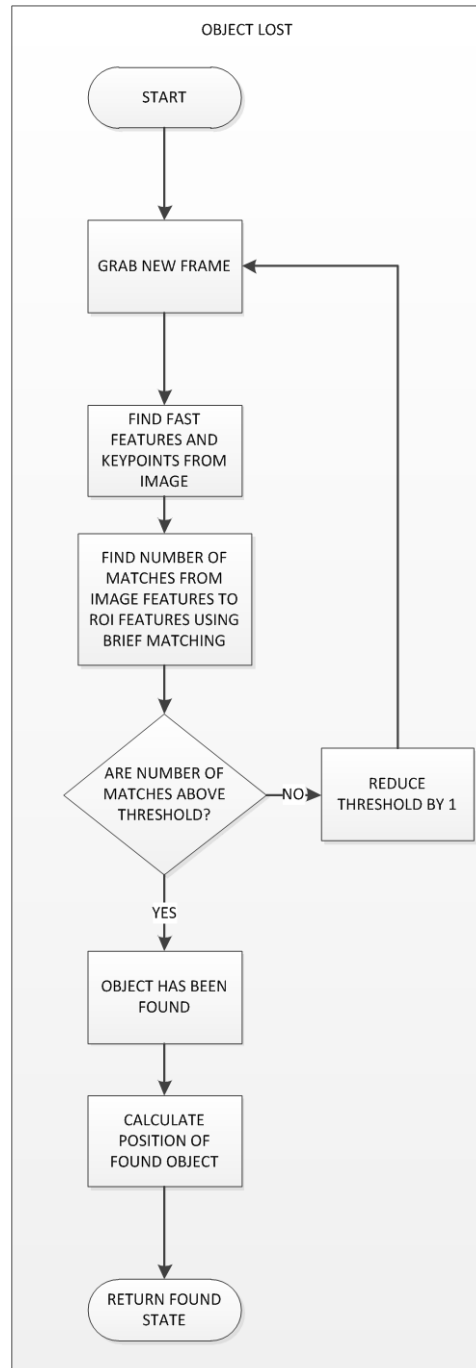




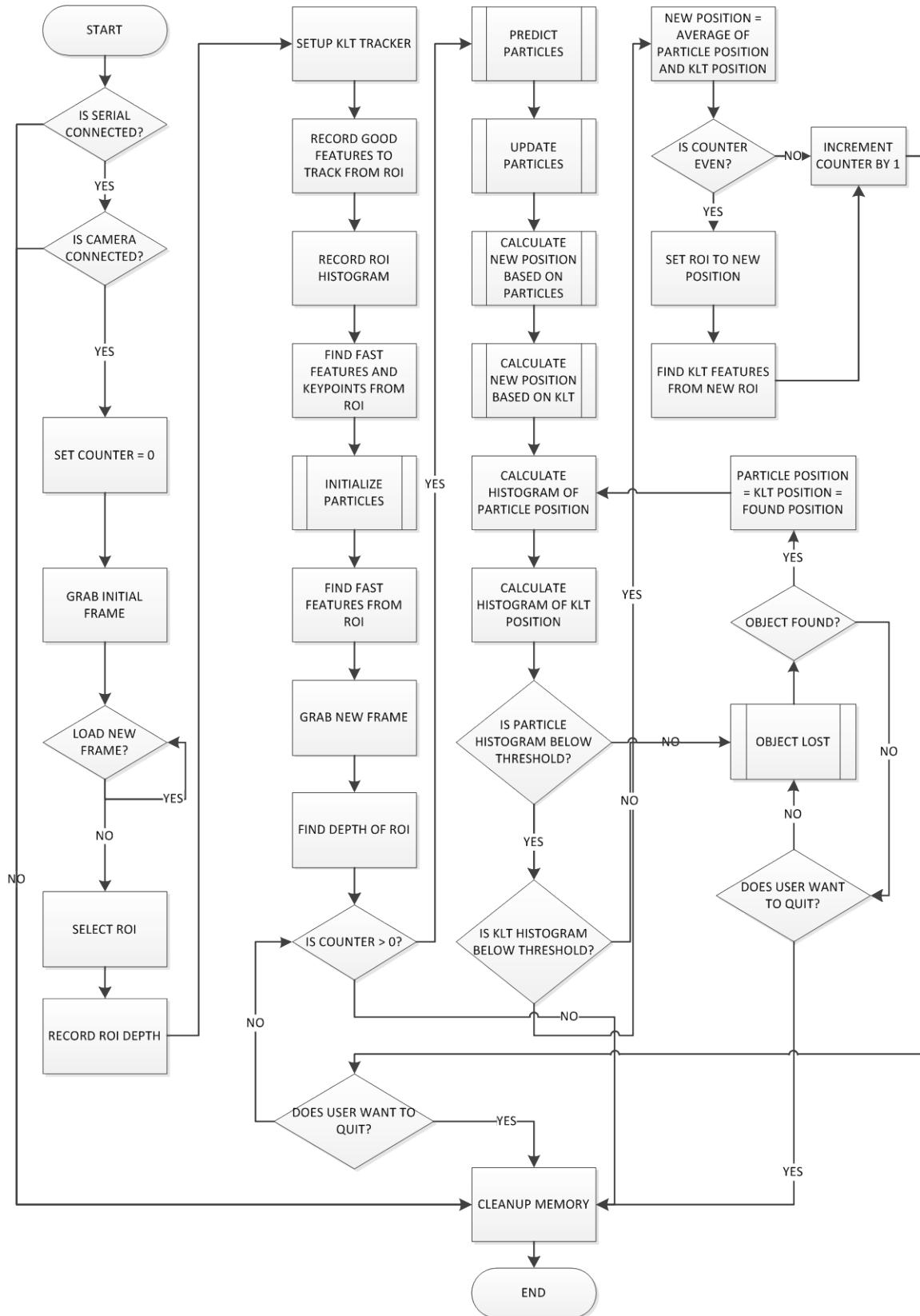
## B. Calculate New Position Flowchart



### C. Lost Object Flowchart



## D. Complete SIGHT Flowchart



## Bibliography

- [1] N. Sharkey, "The ethical frontiers of robotics," *Computer Science*, vol. 322, no. 5909, pp. 1800-1801, 2008.
- [2] K. Kuroi and H. Ishihara, "The four-leg locomotion robot for heavy load transportation," in *Intelligent Mechanical Systems Engineering*, Luoyang, Henan, pp. 221-224, 2006.
- [3] K. Kawamura, D. Wilkes, T. Pack, M. Bishay and J. Barile, "Humanoids: Future robots for home and factory," in *Proceedings of the First International Symposium on Humanoid Robots*, Tokyo, pp. 53-62, 1996.
- [4] iRobot, 2012. [Online]. Available: [http://store.irobot.com/family/index.jsp?ab=CMS\\_IRBT\\_Storefront\\_062209\\_iwantroomba&categoryId=2501652&cp=2804605&s=A-ProductAge](http://store.irobot.com/family/index.jsp?ab=CMS_IRBT_Storefront_062209_iwantroomba&categoryId=2501652&cp=2804605&s=A-ProductAge).
- [5] iRobot, 2012. [Online]. Available: <http://store.irobot.com/category/index.jsp?categoryId=3334444&cp=2804605>.
- [6] iRobot. [Online]. Available: <http://store.irobot.com/category/index.jsp?categoryId=3334470&cp=2804605>.
- [7] M. Fujita, K. Sabe, Y. Kuroki, T. Ishida and T. T. Doi, "SDR-4X II: A small humanoid as an entertainer in home environment," in *Robotics Research*, vol. 15, Springer Berlin / Heidelberg, 2005, pp. 355-364.
- [8] C. Bartneck and J. Forlizzi, "A design-centred framework for social human-robot interaction," in *Proceedings of the Ro-Man*, Kurashiki, pp. 591-594, 2004.
- [9] United Nations, "United Nations and the international federation of robotics," in *Proceedings of the World Robotics*, New York, 2002.
- [10] K. Morioka, J.-H. Lee and H. Hashimoto, "Human-following mobile robot in a distributed intelligent sensor network," *IEEE Trans on Industrial Electronics*, vol. 51, no. 1, pp. 229-237, February 2004.
- [11] E. Prassler, D. Bank and B. Kluge, "Motion coordination between a human and a robotic wheelchair," in *IEEE International Workshop on Robot and Human Interactive Communication*, Bordeaux, Paris, pp. 412-417, 2001.
- [12] D. Fiel-Seifer and M. J. Mataric, "Defining Socially Assitive Robotics," in *2005 IEEE 9th International Conference on Rehabilitation Robotics*, Chicago, IL, USA, pp. 465-468, 2005.
- [13] M. Wiener and A. Mehrabian, *Language within language: immediacy, a channel in verbal communication*, New York: Appleton-Century-Crofts, 1968.
- [14] M. Argyle and J. Dean, "Eye-contact, distance and affiliation," *Sociometry*, vol. 28, no. 3, pp. 289-304, 1965.
- [15] R. Rettie, "Connectedness, awareness and social presence," in *6th Annual International Workshop on Presence*, Aalborg, Denmark, pp. 6-8, 2003.
- [16] P. Dourish and S. Bly, "Portholes: Supporting awareness in a distributed work," in *Proceedings of ACM CHI conference on Human factors in computing systems*, Monterey, California, pp. 541-547, 1992.
- [17] C. Breazeal, "Toward sociable robots," *Robotics and Autonomous Systems*, vol. 42, no. 3-

- 4, pp. 167-175, March 2003.
- [18] M. Liem, A. Visser and F. Groen, "A hybrid algorithm for tracking and following people using a robotic dog," in *HRI '08 Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, Amsterdam, pp. 185-192, 2008.
  - [19] C. Helmlinger, "A growing physical workload threatens nurses' health," *American Journal of Nursing*, vol. 97, no. 4, pp. 64-66, April 1997.
  - [20] J. M. McNeil, "Americans With Disabilities: 1994-95," Washington, 1997.
  - [21] E. Steinmetz, "Americans With Disabilities: 2002," Washington, 2006.
  - [22] M. W. Brault, "Americans With Disabilities: 2005," Washington, 2008.
  - [23] F. Hobbs and N. Stoops, "Demographic trends in the 20th century," 2002.
  - [24] US Department of Health and Human Services, "Health and aging chartbook," United States, 1999.
  - [25] A. Yilmaz, O. Jayed and M. Shah, "Object tracking: A survey," *ACM Computing Surveys (CSUR)*, vol. 38, no. 4, pp. 1-45, December 2006.
  - [26] I. Tena Ruiz, Y. Petillot, D. Lane and J. Bell, "Tracking objects in underwater multibeam sonar images," in *IEE Colloquium on Motion Analysis and Tracking (Ref. No. 1999/103)*, London, pp. (11) 1-7, 1999.
  - [27] P. Bahl and V. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system," in *IEEE Proceedings INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, Tel Aviv, pp. (2) 775-784, 2000.
  - [28] A. Yilmaz and K. Shafique, "Target tracking in airborne forward looking imagery," *Image and Vision Computing*, vol. 21, pp. 623-635, 2003.
  - [29] Braitenberg and Valentino, *Vehicles: experiments in synthetic psychology*, REP edition ed., A Bradford Book, 1986.
  - [30] C. Yang, R. Duraiswami and L. Davis, "Fast multiple object tracking via a hierarchical particle filter," *Tenth IEEE International Conference on Computer Vision*, vol. 1, pp. 212-219, October 2005.
  - [31] V. Salari and I. Sethi, "Feature point correspondence in the presence of occlusion," *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 12, no. 1, pp. 87-91, January 1990.
  - [32] R. Streit and T. E. Luginbuhl, "Maximum likelihood method for probabilistic multihypothesis tracking," in *Proceedings of the International Society for Optical Engineering (SPIE.)*, pp. 394-405, 1994.
  - [33] D. Comaniciu, R. Visvanathan and P. Meer, "Kernel-based object tracking," *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564-577, 2003.
  - [34] H. Tao, H. Sawhney and R. Kumar, "Object tracking with Bayesian estimation of dynamic layer representations," *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 24, no. 1, pp. 75-89, 2002.
  - [35] M. Isard and A. Blake, "Condensation - conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5-28, 1998.
  - [36] R. Ronfard, "Region based strategies for active contour models," *International Journal of*

- Computer Vision*, vol. 13, no. 2, pp. 229-251, 1997.
- [37] D. Huttenlocher, J. Noh and W. Rucklidge, "Tracking nonrigid objects in complex scenes," in *Fourth International Conference on Computer Vision*, Berlin, Germany, pp. 93-101, 1993.
  - [38] J. Kang, I. Cohen and G. Medioni, "Object reacquisition using geometric invariant appearance," in *ICPR '04 Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04)*, pp. 759-762, 2004.
  - [39] C. Veenman, M. Reinders and E. Backer, "Resolving motion correspondence for densely moving points," *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 23, no. 1, pp. 54-72, 2001.
  - [40] D. Serby and L. V. Gool, "Probabilistic object tracking using multiple features," in *IEEE International Conference of Pattern Recognition (ICPR)*, pp. 184-187, 2004.
  - [41] D. H. Ballard and C. M. Brown, *Computer Vision*, Prentice Hall, 1982.
  - [42] A. Ali and J. Aggarwal, "Segmentation and recognition of continuous human activity," in *IEEE Workshop on Detection and Recognition of Events in Video*, Vancouver, pp. 28-35, 2001.
  - [43] I. Haritaoglu, D. Harwood and L. S. Davis, "W4: real-time surveillance of people and their activities," *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 809-830, 2000.
  - [44] S. Zhu, T. Lee and A. Yuille, "Region competition: unifying snakes, region growing, and bayes/mdl for multiband image segmentation," in *Fifth International Conference on Computer Vision*, Cambridge, pp. 416-423, 1995.
  - [45] N. Paragios and R. Deriche, "Geodesic active regions and level set methods for supervised texture segmentation," *International Journal of Computer Vision*, vol. 46, no. 3, pp. 223-247, 2002.
  - [46] P. Fieguth and D. Terzopoulos, "Color-based tracking of heads and other mobile objects at video frame rates," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Juan, pp. 21-27, 1997.
  - [47] D. Cremers and C. Schnorr, "Statistical shape knowledge in variational motion segmentation," *Image and Vision Computing*, vol. 21, no. 1, pp. 77-86, 2003.
  - [48] A. D. Jepson, D. J. Fleet and T. F. El-Maraghi, "Robust online appearance models for visual tracking," *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1296-1311, 2003.
  - [49] C. Schmid, R. Mohr and C. Bauckhage, "Comparing and evaluating interest points," in *Sixth International Conference on Computer Vision*, Bombay, pp. 230-235, 1998.
  - [50] H. P. Moravec, "Visual mapping by a robot rover," in *IJCAI'79 Proceedings of the 6th international joint conference on Artificial intelligence*, Tokyo, Japan, pp. 598-600, 1979.
  - [51] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of The Fourth Alvey Vision Conference*, pp. 147-151, 1988.
  - [52] J. Shi and C. Tomasi, "Good features to track," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 593-600, 1994.
  - [53] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, November 2004.

- [54] E. Rosten and T. Drummond, "Fusing points and lines for high performance tracking," in *IEEE International Conference on Computer Vision*, Beijing, pp. 1508-1511, 2005.
- [55] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Computer Vision – ECCV 2006*, Springer Berlin / Heidelberg, 2006, pp. 430-443.
- [56] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1615-1630, October 2005.
- [57] C. Wren, A. Azarbayejani, T. Darrell and A. Pentland, "Pffinder: Real-time tracking of the human body," in *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition*, pp. 51-56, 1996.
- [58] X. Gao, T. Boult, F. Coetzee and V. Ramesh, "Error analysis of background adaption," in *IEEE Conference on Computer Vision and Pattern Recognition*, Hilton Head Island, pp. 503-510, 2000.
- [59] C. Stauffer and W. Grimson, "Learning patterns of activity using real time tracking," *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 747-757, 2000.
- [60] A. Elgammal, D. Harwood and L. Davis, "Non-parametric model for background subtraction," in *European Conference on Computer Vision (ECCV)*, pp. 751-767, 2000.
- [61] K. Toyama, J. Krumm, B. Brumitt and B. Meyers, "Wallflower: Principles and practices of background maintenance," in *Seventh International Conference on Computer Vision (ICCV)*, Kerkyra, Greece, pp. 255-261, 1999.
- [62] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603-619, 2002.
- [63] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888-905, 2000.
- [64] N. Xu and N. Ahuja, "Object contour tracking using graph cuts based active contours," in *International Conference on Image Processing*, pp. III-277 - III-280, 2002.
- [65] S. C. Zhu and A. Yuille, "Region competition: unifying snakes, region growing, and bayes/mdl for multiband image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 9, pp. 884-900, 1996.
- [66] A. Yilmaz, X. Li and M. Shah, "Contour based object tracking with occlusion handling in video acquired using mobile cameras," *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1531-1536, 2004.
- [67] H. Rowley, S. Baluja and T. Kanade, "Neural network-based face detection," *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 23-38, 1998.
- [68] P. Viola, M. J. Jones and D. Snow, "Detecting pedestrians using patterns of motion and appearance," *International Journal of Computer Vision*, vol. 63, no. 2, pp. 153-161, 2005.
- [69] L. Grewe and A. Kak, "Interactive learning of a multi-attribute hash table classifier for fast object recognition," *Computer Vision Image Understand*, vol. 61, no. 3, pp. 387-416, 1995.
- [70] C. P. Papageorgiou, M. Oren and T. Poggio, "A general framework for object detection,"

- in *Sixth International Conference on Computer Vision (ICCV'98)*, Bombay, India, pp. 555-562, 1998.
- [71] I. K. Sethi and R. Jain, "Finding trajectories of feature points in a monocular image sequence," *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 9, no. 1, pp. 56-73, 1987.
  - [72] K. Rangarajan and M. Shah, "Establishing motion correspondence," *Computer Vision Graphics Image Processing: Image Understanding*, vol. 54, no. 1, pp. 56-73, July 1991.
  - [73] S. Intille, J. David and A. Bobick, "Real-time closed-world tracking," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 697-703, 1997.
  - [74] T. J. Broida and R. Chellappa, "Estimation of object motion parameters from noisy images," *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 8, no. 1, pp. 90-99, 1986.
  - [75] D. Beymer and K. Konolige, "Real-time tracking of multiple people using continuous detection," in *ICCV Frame-Rate Workshop*, 1999.
  - [76] G. Kitagawa, "Non-gaussian state-space modeling of nonstationary time series," *Journal of the American Statistical Association*, vol. 82, no. 400, pp. 1032-1041, December 1987.
  - [77] D. Klein, D. Schulz, S. Frintrop and A. Cremers, "Adaptive real-time video-tracking for arbitrary objects," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Taipei, pp. 772-777, 2010.
  - [78] K. Nummiaro, E. Koller-Meier and L. V. Gool, "An adaptive color-based particle filter," *Image and Vision Computing*, vol. 21, no. 1, pp. 99-110, January 2003.
  - [79] T. Kodama, T. Yamaguchi and H. Harada, "A Method of Object Tracking Based on particle filter and optical flow to avoid degeneration problem," in *Proceedings of SICE Annual Conference 2010*, Taipei, pp. 1529-1533, 2010.
  - [80] D. Comaniciu, "Bayesian kernel tracking," in *Annual Conference of the German Society for Pattern Recognition*, pp. 438-445, 2002.
  - [81] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, no. 1-3, pp. 185-203, August 1981.
  - [82] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *International Joint Conference on Artificial Intelligence*, pp. 674-679, 1981.
  - [83] G. Bradski and A. Kaehler, *Learning OpenCV*, O'Reilly Media, 2008.
  - [84] M. Haag and H.-H. Nagel, "Combination of edge element and optical flow estimates for 3d-model-based vehicle tracking in traffic image sequences," *International Journal of Computer Vision*, vol. 35, no. 3, pp. 295-319, 1999.
  - [85] B. Li, R. Chellappa, Q. Zheng and S. Der, "Model-based temporal object verification using video," *IEEE Transactions on Image Processing*, vol. 10, no. 6, pp. 897-908, June 2001.
  - [86] K. Sato and J. Aggarwal, "Temporal spatio-velocity transform and its application to tracking and interaction," in *Comput. Vision Image Understand*, pp. 100-128, 2004.
  - [87] J. MacCormick and A. Blake, "Probabilistic exclusion and partitioned sampling for multiple object," in *The Proceedings of the Seventh IEEE International Conference on Computer Vision*, Kerkyra, pp. 572-578, 1999.



- [88] Y. Chen, Y. Rui and T. S. Huang, "Jpdaf based hmm for real-time contour tracking," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Kauai, pp. 543-550, 2008.
- [89] M. Bertalmio, G. Sapiro and G. Randall, "Morphing active contours," *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 22, no. 7, pp. 733-737, 2000.
- [90] A.-R. Mansouri, "Region tracking via level set pdes without motion computation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 947-961, July 2002.
- [91] C. Breazeal, "Social interactions in HRI: the robot view," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 34, no. 2, pp. 181-186, May 2004.
- [92] E. Goffman, "Facial Engagements," Boston, Macmillian, 1963, pp. 83-111.
- [93] A. Tapus and M. J. Mataric, "Guest editorial: special issue on socially assistive robotics," *Autonomous Robots*, vol. 24, no. 2, pp. 121-122, February 2008.
- [94] T. Fong, I. Nourbakhsh and K. Dautenhahn, "A survey of socially interactive robots," *Robotics and Autonomous Systems*, vol. 42, no. 3-4, pp. 143-166, March 2003.
- [95] V. A. Kulyukin and C. Gharpure, "Ergonomics-for-one in a robotic shopping cart for the blind," in *HRI '06 Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, New York, NY, pp. 142-149, 2006.
- [96] M. Montemerlo, J. Pineau, N. Roy, S. Thrun and V. Verma, "Experiences with a mobile robotic guide for the elderly," in *Proceeding of the Eighteenth national conference on Artificial intelligence*, Menlo Park, CA, pp. 587-592, 2002.
- [97] H. Yu, M. Spenko and S. Dubowsky, "An adaptive shared control system for an intelligent mobility aid for the elderly," *Autonomous Robots*, vol. 15, no. 1, pp. 53-66, July 2003.
- [98] C. Martens, N. Ruchel, O. Lang, O. Ivlev and A. Graser, "A FRIEND for assisting handicapped people," *Robotics & Automation Magazine, IEEE*, vol. 8, no. 1, pp. 57-65, March 2001.
- [99] K. Andreyo, "Nursebot project: robotic assistance for the elderly," [Online]. Available: <http://www-2.cs.cmu.edu/~nursebot/>.
- [100] K. Wada, T. Shibata, T. Saito and K. Tanie, "Effects of robot-assisted activity for elderly people and nurses at a day service center," *Proceedings of the IEEE*, vol. 92, no. 11, pp. 1780-1788, November 2004.
- [101] M. E. Pollack, "Intelligent technology for an aging population: The use of AI to assist elders with cognitive impairment," *AI Magazine*, vol. 26, no. 2, pp. 9-24, 2005.
- [102] T. Hareven, "Historical perspectives on aging and family relations," in *Handbook of Aging and the Social Sciences*, 5th ed., R. Binstock and L. George, Eds., New York, Elsevier Science, 2001.
- [103] S. Pin, E. Guilley, D. Spini and C. Lalive d'Epinay, "The impact of social relationships on the maintenance of independence in advanced old age: findings of a Swiss longitudinal study," *Zeitschrift für Gerontologie und Geriatrie*, vol. 38, no. 3, pp. 203-209, June 2005.
- [104] U. C. Bureau, "International Data Base - International Programs," June 2011. [Online].

- Available: <http://www.census.gov/population/international/data/idb/country.php>.
- [105] P. Deegan, R. Grupen, A. Hanson, E. Horrell, S. Ou, E. Riseman, S. Sen, B. Thibodeau, A. Williams and D. Xie, "Mobile manipulators for assisted living in residential settings," *Autonomous Robots*, vol. 24, no. 2, pp. 179-192, February 2008.
  - [106] E. Friedmann, A. Katcher and S. Thomas, "Animal companions and one-year survival of patients after discharge from a coronary care unit," *Public Health Report*, vol. 95, no. 4, pp. 307-312, 1980.
  - [107] M. M. Baun, N. Bergstrom, N. F. Langston and L. Thoma, "Physiological effects of human/companion animal bonding," *Nursing Research*, vol. 33, no. 3, pp. 126-129, May 1984.
  - [108] T. F. Garrity, L. Stallones, M. B. Marx and T. P. Johnson, "Pet ownership and attachment as supportive factors in the health of the elderly," *Anthrozoos*, vol. 3, no. 1, pp. 35-44, 1989.
  - [109] K. Wada, T. Shibata, T. Saito, K. Sakamoto and K. Tanie, "Psychological and social effects of one year robot assisted activity on elderly people at a health service facility for the aged," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 2785-2790, 2005.
  - [110] P. Martin, August 2009. [Online]. Available: <http://www.procprblog.com/new-robots-help-humans-cope-with-illness-alzheimers-disease>.
  - [111] M. Fujita, "AIBO: Toward the era of digital creatures," *The International Journal of Robotics Research*, vol. 20, no. 10, pp. 781-794, 2001.
  - [112] T. Tamura, S. Yonemitsu, A. Itoh, D. Oikawa, A. Kawakami, Y. Higashi, T. Fujimooto and K. Nakajima, "Is an entertainment robot useful in the care of elderly people with severe dementia?," *The Journals of Gerontology Series A: Medical Sciences*, vol. 59, no. 1, pp. 83-85, June 2004.
  - [113] S. Dubowsky, F. Genot, S. Godding, H. Kozono, A. Skwersky, H. Yu and L. S. Yu, "PAMM - a robotic aid to the elderly for mobility assistance and monitoring: a "helping-hand" for the elderly," in *IEEE International Conference on Robotics and Automation*, San Francisco, pp. 570-576, 2000.
  - [114] September 2005. [Online]. Available: <http://calorielab.com/news/wp-images/post-images/aibo-features.jpg>.
  - [115] May 2010. [Online]. Available: [http://www.census.gov/newsroom/releases/archives/facts\\_for\\_features\\_special\\_editions/cb10-ff13.html](http://www.census.gov/newsroom/releases/archives/facts_for_features_special_editions/cb10-ff13.html).
  - [116] C. J. Winstein, P. J. Miller, S. Blanto, E. Taub, G. Uswatte, D. Morris, D. Nichols and S. Wolf, "Methods for a multisite randomized trial to investigate the effect of constraint-induced movement therapy in improving upper extremity function among adults recovering from a cerebrovascular stroke," *Neurorehabil Neural Repair*, vol. 17, no. 3, pp. 137-152, September 2003.
  - [117] C. Burgar, P. Lum, P. Shor and H. Van der Loos, "Development of robots for rehabilitation therapy: The palo alto va/standford experience," *Journal of Rehabilitation Research and Development*, vol. 37, no. 6, pp. 663-673, November 2002.
  - [118] R. Mahoney, H. Can der Loos, P. Lum and C. Burgar, "Robotic stroke therapy assistant,"

- Robotica*, vol. 21, no. 1, pp. 33-44, January 2003.
- [119] R. Jackson, "Robotics and its role in helping disabled people," *Engineering Science and Education Journal*, vol. 2, no. 6, p. 267, December 1993.
  - [120] T. Felzer and B. Freisleben, "HaWCoS: The "hands-free" wheelchair control system," in *ASSETS 2002 - Proceedings of the Fifth International ACM SIGCAPH Conference on Assistive Technologies*, Edinburgh, pp. 127-134, 2002.
  - [121] J. Nadel, A. Revel, P. Andy and P. Gaussier, "Toward communication, first imitations in infants, low-functioning children with autism and robots," *Interaction Studies*, vol. 5, no. 1, pp. 45-74, 2004.
  - [122] I. Werry, K. Dautenhahn, B. Ogden and W. Harwin, "Can social interaction skills be taught by a social agent? The role of a robotic mediator in autism therapy," in *CT '01 Proceedings of the 4th International Conference on Cognitive Technology: Instruments of Mind*, London, pp. 57-74, 2001.
  - [123] M. Topping, "An overview of the development of handy 1, a rehabilitation robot to assist the severely disabled," *Journal of Intelligent and Robotic Systems*, vol. 34, no. 1, pp. 253-263, 2002.
  - [124] T. Kanda, S. Nabe, K. Hiraki, H. Ishiguro and N. Hagita, "Human friendship estimation model for communication robots," *Autonomous Robots*, vol. 24, no. 2, pp. 135-145, February 2008.
  - [125] C. Plaisant, A. Druin, C. Lathan, K. Dakhane, K. Edwards, J. M. Vice and J. Montemayor, "A storytelling robot for pediatric rehabilitation," in *Proceedings of the fourth international ACM conference on Assistive technologies*, New York, pp. 50-55, 2000.
  - [126] H. Ishiguro, T. Ono, M. Imai and T. Kanda, "Development of an interactive humanoid robot "Robovie" - an interdisciplinary approach," *Robotics Research*, vol. 6, no. 1, pp. 179-191, 2003 2003.
  - [127] B. M. Scassellati, "Foundations for a theory of mind for a humanoid robot," Cambridge, 2001.
  - [128] R. A. Brooks, C. Breazeal, M. Marjanovic, B. Scassellati and M. M. Williamson, "The cog project: building a humoid robot," in *Computation for metaphors, analogy, and agents*, C. L. Nehaniv, Ed., Berlin, Springer-Verlag, 1999, pp. 52-87.
  - [129] F. Matia, R. Sanz and E. Puente, "Increasing intelligence in autonomous wheelchairs," *Journal of Intelligent and Robotic Systems*, vol. 22, no. 3-4, pp. 211-232, 1998.
  - [130] E. Meisner, V. Isler and J. Trinkle, "Controller design for human-robot interaction," *Autonomous Robots*, vol. 24, no. 2, pp. 123-134, February 2008.
  - [131] N. Roy, J. Pineau and S. Thrun, "Spoken dialogue management using probabilistic reasoning," in *ACL '00 Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, Stroudsburg, PA, pp. 93-100, 2000.
  - [132] C. Breazeal, "Emotion and sociable humanoid robots," *International Journal of Human-Computer Studies*, vol. 59, no. 1-2, pp. 119-155, July 2003.
  - [133] C. Breazeal, A. Edsinger, P. Fitzpatrick and B. Scassellati, "Active vision for sociable robots," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 31, no. 5, pp. 443-453, September 2001.

- [134] C. Breazeal, "Kismet, the robot," [Online]. Available: <http://www.ai.mit.edu/projects/sociable/baby-bits.html>.
- [135] E. Demeester, A. Huntemann, D. Vanhooydonck, G. Vanacker, H. Van Brussel and M. Nuttin, "User-adapted plan recognition and user-adapted shared control: A Bayesian approach to semi-autonomous wheelchair driving," *Autonomous Robots*, vol. 24, no. 2, pp. 193-211, February 2008.
- [136] C. Schmidt, N. Sridharan and J. Goodson, "The plan recognition problem: An intersection of psychology and artificial intelligence," *Artificial Intelligence*, vol. 11, no. 1-2, pp. 45-83, August 1978.
- [137] S. Carberry, "Techniques for plan recognition. User Modeling," *User Modeling and User-Adapted Interaction*, vol. 11, no. 1, pp. 31-48, March 2001.
- [138] R. Gockley, J. Forlizzi and R. Simmons, "Natural person-following behavior for social robots," in *HRI '07 Proceedings of the ACM/IEEE international conference on Human-robot interaction*, New York, NY, pp. 17-24, 2007.
- [139] H. A. Yanco, "Wheelesley: A robotic wheelchair system: Indoor navigation and user interface," *Assistive Technology And Artificial Intelligence*, vol. 1458, pp. 256-268, 1998.
- [140] Y. Kuno, N. Shimada and Y. Shirai, "Look where you're going [robotic wheelchair]," *IEEE Robotics & Automation Magazine*, vol. 10, no. 1, pp. 26-34, March 2003.
- [141] Y. Matsumoto, T. Ino and T. Ogasawara, "Development of intelligent wheelchair system with face and gaze based interface," in *IEEE International Workshop on Robot and Human Interactive Communication*, Paris, pp. 262-267, 2001.
- [142] A. Argyros, P. Georgiadis, P. Trahanias and D. Tsakiris, "Semi-autonomous navigation of a robotic wheelchair," *Journal of Intelligent & Robotic Systems*, vol. 34, no. 3, pp. 315-329, 2002.
- [143] E. Prassler, J. Scholz and P. Fiorini, "A robotics wheelchair for crowded public environment," *IEEE Robotics & Automation Magazine*, vol. 8, no. 1, pp. 38-45, March 2001.
- [144] J. Gonzalez, A. Muaoz, C. Galindo, J. Fernandez-Madriral and J. Blanco, "A description of the SENA robotic wheelchair," in *IEEE Electrotechnical Conference*, Malaga, pp. 437-440, 2006.
- [145] B. Kluge, C. Kohler and E. Prassler, "Fast and robust tracking of multiple moving objects with a laser range finder," in *IEEE International Conference on Robotics and Automation*, pp. 1683-1688, 2001.
- [146] E. T. Hall, *The Hidden Dimension: Man's Use Of Space in Public and Private*, London: The Bodley Head Ltd, 1966.
- [147] H. Huettenrauch, K. Eklundh, A. Green and E. Topp, "Investigating spatial relationships in human-robot interaction," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, pp. 5052-5059, 2006.
- [148] S. C. E. A. LLC, 2011. [Online]. Available: <http://us.playstation.com/ps3/accessories/playstation-eye-camera-ps3.html>.
- [149] Microsoft, 2011. [Online]. Available: <http://www.xbox.com/en-CA/Kinect>.
- [150] 2011. [Online]. Available: [http://openkinect.org/wiki/Main\\_Page](http://openkinect.org/wiki/Main_Page).
- [151] [Online]. Available: <http://www.openni.org/>.

- [152] 2011. [Online]. Available: <http://www.primesense.com/>.
- [153] 2011. [Online]. Available: <http://opencv.willowgarage.com/wiki/>.
- [154] Fujitsu. [Online]. Available: [http://store.shopfujitsu.com/ca/EcomCA/pagerender.do?conpg=products\notebooks\tech\\_specs\fc\th700\\_ts](http://store.shopfujitsu.com/ca/EcomCA/pagerender.do?conpg=products\notebooks\tech_specs\fc\th700_ts).
- [155] N. Gordon, D. Salmond and C. Ewing, "Bayesian state estimation for tracking and guidance using the bootstrap filter," *Journal of Guidance Control and Dynamics*, vol. 18, no. 6, pp. 1434-1443, 1995.
- [156] M. Isard and A. Blake, "Contour tracking by stochastic propagation of conditional density," in *European Conference on Computer Vision*, pp. 343-356, 1996.
- [157] G. Kitagawa, "Monte carlo filter and smoother for non-gaussian nonlinear state space models," *Journal of Computational and Graphical Statistics*, vol. 5, no. 1, pp. 1-25, 1996.
- [158] L. G. Shapiro and G. C. Stockman, *Computer Vision*, Prentice Hall, 2003.
- [159] January 2011. [Online]. Available: [http://en.wikipedia.org/wiki/Color\\_histogram](http://en.wikipedia.org/wiki/Color_histogram).
- [160] K. Song, J. Kittler and M. Petrou, "Defect Detection in Random Colour Textures," *Image and Vision Computing*, vol. 14, pp. 667-683, 1996.
- [161] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, no. 1-3, pp. 185-203, August 1981.
- [162] M. Calonder, V. Lepetit, C. Strecha and P. Fua, "BRIEF: Binary Robust Independent Elementary Features," *Computer Vision – ECCV 2010 Lecture Notes in Computer Science*, vol. 6314, pp. 778-792, 2010.
- [163] M. Araki, "PID control," *Control Systems, Robotics and Automation*, vol. 2, no. 1, pp. 1-23, 1995.
- [164] K. Ang, G. Chong and Y. Li, "PID control system analysis, design, and technology," *IEEE Transactions on Control Systems Technology*, vol. 13, no. 4, pp. 559-576, 2005.
- [165] September 2007. [Online]. Available: <http://en.wikipedia.org/wiki/File:Pid-feedback-nct-int-correct.png>.
- [166] P. J. Stoffregen and R. C. Coon, 2011. [Online]. Available: <http://www.pjrc.com/teensy/>.
- [167] D. Engineering. [Online]. Available: <http://www.dimensionengineering.com/Sabertooth2X25.htm>.
- [168] [Online]. Available: <http://liblo.sourceforge.net/>.
- [169] R. L. Knoblauch, M. T. Pietrucha and M. Nitzburg, "Field studies of pedestrian walking speed and start-up time," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1538, no. 1, pp. 27-38, 1996.
- [170] A. Ferworn, W. Lu, A. Arora, W. Shiu, D. Ostrom and J. Pham, "Telebot control of a powered-wheelchair across the WWW — NEPWAK," in *The 2nd International Conference on Mechatronics and Information Technology (ICMIT)*, Jecheon, Korea, 2003.
- [171] S.-J. Battersby, November 2011. [Online]. Available: <http://torontoist.com/2011/11/preparing-for-the-robot-invasion-one-lego-block-at-a-time/>.

## **Nomenclature**

**BRIEF** - Binary Robust Independent Elementary Features

**COTS**: Common Off The Shelf

**EMD** - Earth Mover's Distance

**FAST** - Features from Accelerated Segment Test

**GSR** - Galvanic Skin Response

**HMM** - Hidden Markov Model

**HSV** - Hue, Saturation, Value Color Space

**LK** - Lucas-Kanade Optical Flow Method

**MAid** - Mobility Aid for Elderly and Disabled People

**OOI**: Object Of Interest

**OSC** - Open Sound Control

**OSP** - Optimal Subjective Proximity

**PAMM** - Personal Aid for Mobility and Monitoring

**PDF** - Probability Density Function

**PID** - Proportional, Integral, Derivative

**POI**: Person of Interest

**QOL** - Quality of Life

**RGB** - Red, Green, Blue

**ROI** - Region of Interest

**SAR**: Socially Assistive Robots

**SIGHT** - Socially Interactive, Gesture-Aware, Human-Following, Transport

**SIR** - Sampling Importance Resampling

**SIR:** Socially Interactive Robots