SELECTING AND COMMANDING GROUPS OF ROBOTS USING A VISION-BASED NATURAL USER INTERFACE

by

Brian Milligan

B.Sc., University of Toronto, 2008

A Thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science
in the School
of
Computing Science

SIMON FRASER UNIVERSITY
Summer 2012

APPROVAL

Name: Brian Milligan

Degree: Master of Science

Title of Thesis: Selecting and Commanding Groups of Robots using a Vision-based Natural User Interface

Examining Committee: Dr. Mark Drew
Chair

Dr. Greg Mori, Co-Supervisor

Dr. Richard Vaughan, Co-Supervisor

Dr. Ze-Nian Li, SFU Examiner

Date Approved: _____

Abstract

This thesis presents a novel method of selecting groups of robots from a population. Using a vision-based gestural user interface a user can select a group by circling robots with his or her hand. A robot equipped with a single camera can determine if it was selected by tracking the user's face and a hand. Tracing the user's hand through the circling of robots forms an ellipse in each robots view. We show that by testing if the face point is within that ellipse the robot can infer if it was within the area circled on the floor. This method is then implemented using four robots and demonstrated as a proof of concept.

To everyone who has helped me get to this place in my life, especially my wicked wife, my brother and most of all Mom who has done so much to help me along the way and making me the person I am today. Thanks to my entire awesome family, great friends, my Supervisors and all the teachers and collegues that have helped me through this journey.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Overview

This thesis contributes a unique method of selecting groups of robots from a population using a vision based approach. This is the first demonstration of how a user can select a subset of one or more robots from a population and command those selected robots to perform a task. This method has been demonstrated and published as a peer reviewed video [42] which won the best video award at the HRI2011 International Conference on Human-Robot Interaction [1].

1.1.1 Description of Set up - Multiple Robots and user

The scenario this work considers is shown in Figure 1.1 and described as follows. There is a population of robots, which gather around a user. The user wants to assign tasks to a group of robots. The user needs a way to bind a group of robots together that he or she wants to perform each task. Two options for grouping the robots are to name all the robots the user wants to command or group the robots by their spatial location. The former option may not be feasible if the robots are uniform in appearance, as the user cannot differentiate the robots to name them, or the user could simply not know or forget the names. Further this method would not scale well as one would need to call out a longer and longer list for larger groupings. Our method focuses on the latter option, as it can be performed without the need to know robot labels a priori. The challenge is to find a way for a human to select groups of robots spatially in a way that the robots can understand.

Figure 1.1: A population of robots that are ready to serve the user. The user needs a way to declare which robots he or she wants to perform a particular task.

1.1.2 Using only hand motions to control multiple robots

It is desirable to enable a user to control the robots without requiring an extra device to manipulate. This allows anyone to be able to control the robots without needing to possess an external controller, which would either need to be passed from user to user or be supplied to each potential user. In some cases the user may not even plan to meet the robots so would not be carrying a controller, for example if the user was found by search and rescue robots. Also by not using a device for control, the user can operate the robots with his or her hands free. This can be very important in certain work scenarios, for example when working in an operating room if the user touches a non-sanitized object he or she may be required to re-wash his or hands after every interaction.

Using one's hands to signal towards a location is also natural and intuitive to a user. Humans commonly use their hands to reference people, places and things. These types of gestures include pointing, waving to acknowledge a person, or moving one's hands in a direction to indicate where to go, Figure 1.2 shows illustrative examples of this. If robots can interpret the type of gestures that people make when naturally dealing with other people,

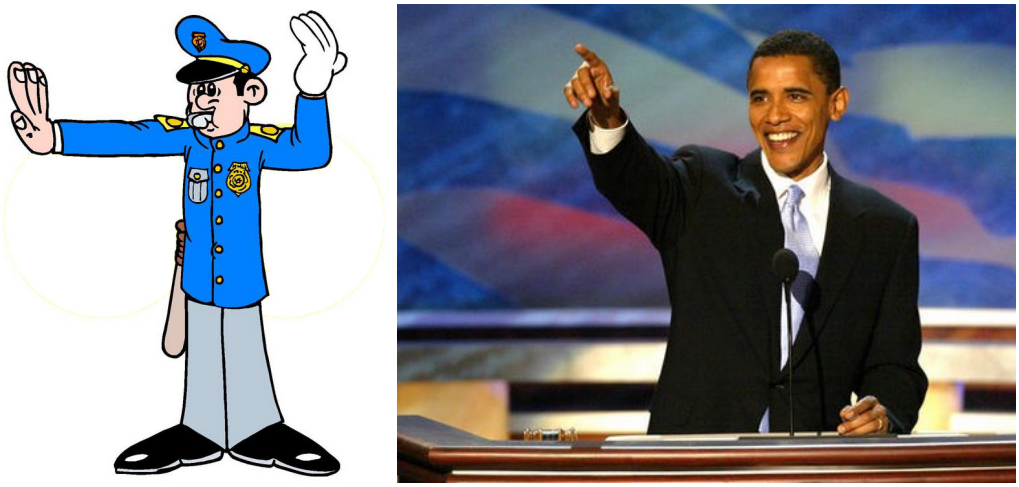

Figure 1.2: Examples of gesturing towards a spatial area. A traffic officer signals drivers in one area to stop while indicating to drivers in another location to drive forward. President Obama points to an individual at a press conference to acknowledge them.

there is less that the user needs to learn in order to control robots. Please note that this work does not include, nor is intended to be a user study. We provide this as an informal motivational argument and look to future directions to show how this work can tie closer to a principled usability argument.

1.2 Motivation

1.2.1 The Future of Robots

Robotics research has developed rapidly in recent years with more real life applications coming to fruition. A future where robots are commonly at our command may not be far away. The Roomba by iRobot is a mobile robot that acts as a vacuum cleaner in many homes, iRobot has claimed to have sold over six million units as of 2010 [4] and furthermore claims that one in three vacuums sold in Spain was a robot in their last quarterly report [5]. The American military has deployed over 2,000 robots to Afghanistan to work alongside soldiers for bomb disposal, surveillance, and control of dangerous entry points in raids [38]. Robots are being used in hospitals as therapeutic companions for patients[62], assistants for doctors and nurses[23], tools for surgeons [55] and for doctors to visit patients by telepresence[57]. As robots become ubiquitous there is greater need to be able to control

them and instruct them in an unencumbering manner.

1.2.2 Interaction with intelligent objects

While the minimal requirement to call something a *robot* can be vague, uninstrumented selection of objects can be applied to any object that shares space with the user. With the cost of cameras and processing steadily decreasing, the concept of having cameras and computing found commonly throughout the home is becoming possible. Early concepts of such homes include being able to turn on and off lights and appliances by pointing to them [64], being able to control groups of objects about the house could offer more user convenience.

Such selection could also be applied not only to physical objects but also to virtual objects. With three-dimensional selection of objects there could also be interaction with holographic displays or 3D images in a head mounted display. Figure 1.3 shows some hollywood concepts of users interacting with a 3D user interface (UI). The Microsoft Kinect and Xbox game console uses a natural user interface to select menu items and interact with game elements on a 2D screen, a more immersive 3D experience in the future is not hard to imagine.

1.2.3 Reading Body Language

The importance of non-verbal communication has been well researched and documented in the field of Psychology [32]. Non-verbal communication is often unconscious and can be easily be taken for granted, yet we use non-verbal cues all the time. People infer other's emotional mindstate from facial expressions and body posture, or can infer where someone's attention is focused from gaze direction or through gestures like pointing. Furthermore knowing that others can read one's body language allows one to use his or her body language to lead others. Examples of this include how one can direct other people's attention by pointing or by turning his or her head to look at something. Non-verbal communication has been found particularly important when communicating to groups of people [58], for example when selecting someone from a crowd or leading a group when walking. These non-verbal cues are subtle and are often processed subconsciously, can be very complicated and culturely contextual [6]. If robots can observe and decipher non-verbal cues they can understand human's intent better[10], and can be used to better follow natural human

Figure 1.3: Concepts from film of users manipulating 3D user interface. Top from Minority Report (all rights reserved to 20th Century Fox) bottom from Ironman (all rights reserved to Paramount Pictures).

dialogue[36].

By trying to model interpretation of non-verbal cues we may also learn more about how human's decipher body language. For example by making a robot that can figure out where a person is pointing we may gain insight into which features human's use to accomplish the same task.

1.3 Challenges

1.3.1 Computer/Robot Vision

Working on real world projects utilizing vision from a camera can present some difficult challenges. Using a natural user interface with vision requires identifying human relevant features in a scene and be able to interpret those features into an intelligible input method. When using a digital camera, the computer doing the vision processing only has a two-dimensional (2D) array of numbers which represent colour values. From this 2D array features from the frame must be extracted and over multiple frames motion features can be extracted. There is a wide variation of movements people can make, for example the motions of a hand can move in any direction and a wide range of speeds. To make sense of the image, features must be found that are invariant enough that they can be reliably discerned from randomness (false positives) yet not so strict as to obscure being able to detect the feature (missed positives).

Another challenge in working with human actions is correctly inferring intent. People can use the same action to mean different things in different contexts. People can also make unintentional movements that are identical to an intentional gesture in context. Holding a hand up can mean "stop" in one context or "hi" in another.

Not only are the motions made by the same person variable, but the variation between users can also make for challenges when interpreting human activity. People can range in height and shape considerably, and trying to use the same method of capturing an action on two very different bodies might not work. Finding users of all different body types to experiment with during development can be challenging to test if the method works for a given body type.

It can be difficult to get precise and always accurate data when using a natural user interface. In Chapter 4 of this thesis hand tracking is used in an implementation of the method described in Chapter 3. With hand tracking the hand may become occluded and

the location of the hand must be inferred. When tracking a hand there may be a loss of precision without further definition, for example is the hand located at the palm, thumb, wrist? Natural user interfaces must deal with this imprecision and be able to deal with potentialy erroneous data.

Working with camera sensors has many challenges for computer vision. Most cameras have a limited view, so interpretation of the scene is limited to a particular area and angle. Camera data is dependent on how light hits the sensor, and colours look very different with changes of illumination. In low light colours in the camera are all closer to black, and the numbers in the 2D array of the camera are harder to differentiate, in the extreme case of a pitch black dark room the camera reports nothing but black and there is nothing to differentiate in the view. On the other side bright light can bleach out colour values and leave the pixel value all closer to white. There is also a degree of noise found on any sensor, so the colour constancy may not even hold with constant illumination.

Single lens cameras also reduce a scene from 3D down to 2D, and any depth information must be inferred from the image. In human vision typically people have two eyes which the brain can use to process depth in a scene. Using two images depth can be inferred from the different locations things are found between the two eyes, a process called binocular disparity. This can be similarly done with two cameras, but increases the cost (by having two cameras) and increases processing by now doubling the amount of image data that is being handled and the disparity must be processed too. Depth cues can be found in single images by making certain assumptions about the world, as will be done in this work, but in general a 2D image does not have enough information for us to know where something in the image is in world space. This is especially challenging when you need to find vectors as at least two 3D points must be found and errors will propagate if making an inference to another point along the vector.

Working with images can be computationally expensive. A two megapixel camera supplies 2 million pixel values for the robot to process with every frame. This requires that vision processing must be done highly efficiently when there are hardware limitations. If the hardware is unable to process the image fast enough, the image processing will either lag (fall behind real-time), or frames will be dropped (reducing frame rate). When tracking motions, decreasing frame rate increases the risk of losing motion features between frames.

1.3.2 Robots

Working with robots presents its own unique challenges. Aside from the camera sensor challenges mentioned above, robots have a limitation on how much of the world they can represent and parse from their internal sensors. Communications between the user and the robot can be limited by the robot's effectors, such as if it has an animatronic face, speakers, lights or other methods of communicating back to the user. The number of sensor and communication devices onboard the robot affects how much the robot costs, and in many applications robots may need to be plentiful and possibly disposable. Cost can also limit the computing power that a robot can utilize, making the above vision processing challenge even more stringent. The nature of robots working in the real world require real time processing, so any delayed or missed processing can put the robot in danger or inoperable.

When working with multiple robots at once, robots can interfere with each other. Particularly they spatially interfere with each other, as they get in each other's way, can block each other's view, or even damage each other if there is a collision. Working with multiple robots can also limit what kind of sensing can be done. Many active sensors that send signals into the environment, like laser depth sensors, can interfere with similar sensors on other robots. With multiple robots, the robots may also require coordination amongst each other, therefore must find a method of communicating from robot to robot.

1.4 Contributions

1. **Chapter 2** A survey of foundational and related work is provided with a new taxonomy provided to distinguish different methods of human controls over robots.

2. **Chapter 3** A novel method of selecting groups of robots to command is detailed that uses vision-based natural user input. This is the first concept of its kind for human robot interaction. This method contributes to the control of multiple robot systems with a user present, an area that had very little exploration as is shown in the survey in Chapter 2. This contributes to machine vision with how it deals with many of the challenges presented in Section 1.3. The method utilizes properties found in the literature on machine interpretation of human pointing gestures reviewed in Chapter 2 and shows how these properties can be used to refer to an extended area rather than just a single point.

3. **Chapter 4** The method described in Chapter 3 is implemented and tested providing evidence that the described method is implementable and works. This offers a potential selection method that can be further investigated for usability and potential real world utility.

Chapter 2

Background and Related Work

With the growth of robotics one of the major challenges is how users can interact with them in the real world. The field of human robot interaction (HRI) [22] deals with how people can interact with robots in the most user friendly manner. HRI is as old as the concept of robotics, in Issac Asimov's early robot books from the 1940's, particularly in his collection of the short stories "I, Robot" [7], the robots operated around laws that all concerned how they affected humans. Recently the field of HRI has been growing rapidly with an annual conference [3] that began in 2005 and dedicated journal that will begin publication in 2012. HRI is a cross disciplinary field that includes the engineering of the robot, artificial intelligence and social sciences working together to find efficient ways for users to engage with robots. Natural user interfaces (NUI) are especially appealing for HRI. NUI use gestures and speech to interact, and do not require the users to manipulate a device to communicate with the robot.

An important feature of robots is that they are embodied and situated in the same world as the people they are interacting with. This has its advantages and challenges. One of the major challenges in robotics has been that the world must be sensed and interpreted, and it can be very difficult to represent all the intricacies of context and consequence. One of the advantages though is that we can refer to common objects, as in a person can refer to objects using deictic references, as in pointing to "that object" rather than needing to refer to everything as a pre-labeled object. Two ways people often reference other people, places and things is through by gaze and hand gestures. In this section I review the use of deictic references in HRI to reference objects, robots and locations.

2.1 Vision-based Gesture Recognition

Gestures are movements, usually in reference to human movements, that convey meaning to another person, animal or machine. Gestures have been used in computer vision [43] as a way to communicate with a computer or robot. One of the current application of gestures as a user input system has been used by Microsoft's Kinect Sensor. The Kinect sensor uses a depth sensor and standard RGB camera, this sensor delivers the image and depth content to a connected game console or PC. Software in the receiving computer can articulate a skeleton composed of located body parts and used for human-pose estimation [53]. With this skeleton the computer can read the user's movements to control a game or navigate a menu system on a screen. Figure 2.1 shows two users playing a Kinect game.

Figure 2.1: Promotional pictures from Microsoft's Kinect Sports 2 [56], a game that utilizes the Kinect. The users are able to play a game of american football by moving parts of their body and pantomiming actions in the game.

Two types of gestures include symbolic and deictic gestures. Symbolic gestures are movements that have a predefined meaning. An example is sign language, where pose or motions of a hand can represent words and sentences. Many researchers work on making machines able to interpret different types of sign language [15] as a way to instruct a computer or robot. Others have worked on making new vocabularies and have simple gestures that a user can learn to converse with a robot [34].

Deictic gestures are gestures that are contextually relevant, particularly here they refer to something spatially relevant to someone / something's body. An example would be if

two people were in a room with five objects on the floor, if one person said to the other person "can you pick that up", it would be ambiguous which object they were referring to. If the asking person instead points to an object, or is staring at it and asks the other person "can you pick that up", the other person will have a good chance of knowing which object to pick up. The act of pointing or staring to bring a shared attention to the object is a deictic gesture. Deictic gestures have been found to be an important aspect to human communication and comprehension [14], as well as in child development [44]. These gestures help us understand what people are acting on without having them explicitly clarify what "that" is. Being able to understand deictic references has been recognized as crucial to achieving good HRI, Brooks and Brazeal found it such an important aspect with interacting with the robots around the MIT media lab that they created a spatial reasoning framework particularly for tracking and maintaining deictic references [11]. Other researchers have highlighted the essential aspects of deictic referencing for teaching robots about new objects and tasks [8] and others have used interdisciplinary approaches to finding what make good features for deictic gestures from infant development and modeled them on robots [45].

Two ways people commonly use deictic references are through gaze and pointing. With gaze humans are remarkably good at inferring what other people are looking at or if someone is looking at them, this provides cues for shared attention [31]. The other common use of deictic referencing is by pointing, which is particularly relevant to this thesis and discussed in detail next.

2.1.1 Pointing

Pointing is a common deictic gesture people use to bring attention to places or things in our environment. Pointing here is defined as the act of referencing an object or location based on the location and orientation of a person's hand. Human infants learn to interpret other's pointing references and typically can understand and point themselves at about 9-12 months [63], yet most other animals cannot interpret pointing. Even chimpanzees, which can be taught to follow symbolic gestures as complicated as American Sign Language[21] typically cannot learn how to reference what or where a person is pointing [17]. Interestingly one of the few animals that can reference a place or thing by a person's pointing gesture are dogs [54], which were bred as human social companions.

Human-computer interaction with pointing has been done since 1980, with Richard A Bolt's seminal paper "Put-That-There" [51] in which he used pointing to select icons on a

screen. The direction of pointing was found using magnetic sensors, one magnetic cube was mounted on the arms of a chair and another magnetic coil was strapped onto the users arm, as can be seen in Figure 2.2. When sitting in the chair the direction of the arm could be determined by the displacement from the magnetic sensor. Selected objects could then be manipulated using voice commands.

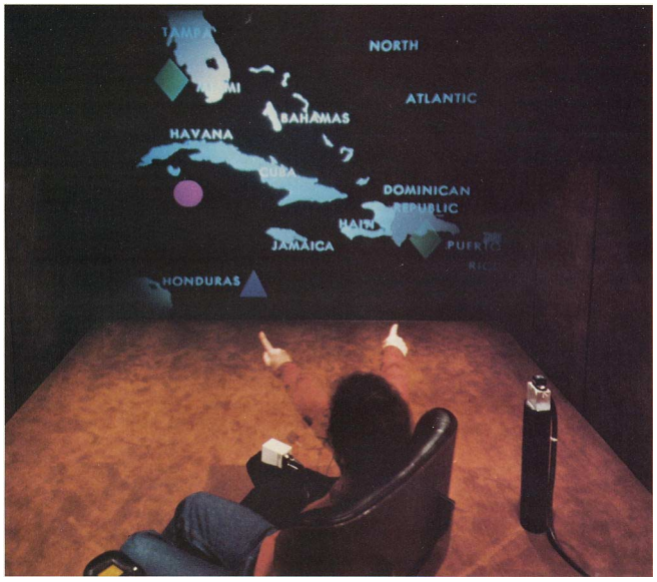

Figure 2.2: Image from Bolt's early work on using pointing to interface with a computer [51].

Some of the first uninstrumented pointing with cameras was done by Cipolla et al. [13]. In their work the user shapes their hand like a gun, with the index finger pointing to the target and thumb pointing up. The line from thumb joint to the end of the finger was found from this pose and projected across the image. Using two cameras for stereo vision they could find a point on the floor to which the user was pointing(see Figure 2.2).

Some of the first work involving recognition and estimation of pointing involving multiple parts of the body is by Jojic et al.[25]. In their work the authors use disparity maps from two cameras and and extract a human's arm vector from the hand to the shoulder, and used this arm vector to drive an on screen mouse icon.

An important aspect of pointing was discovered by Nickel and Stiefelhagen [47]. In this paper the authors compared targeted pointing interpreted by a robot with a stereo camera.

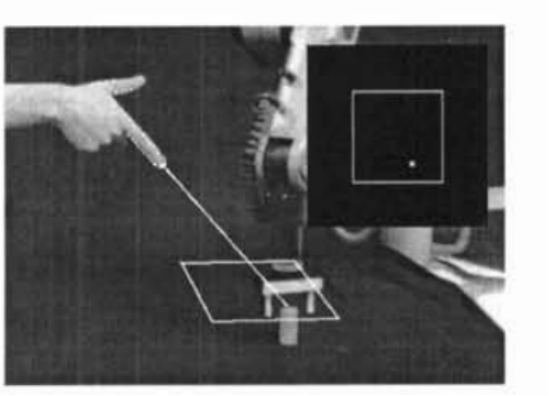

Figure 2.3: Image from Cipolla et al.'s early work on pointing [13].

They then compared the accuracy of using the direction of the arm with the vector made up of the user's head and the outstretched hand. They found that the hand-head oriented vector provided significantly better results than the arm vector. They verified these results with motion capture technology and a head apparatus that provided the user's viewing direction, see Figure 2.4 for a summary of the results and clarity on the pointing lines compared. These results have since been confirmed in subsequent independent studies [52] [18] [48].

Droeschel et al. recently found that while the hand-eye line is the best single estimator of a person's pointing target, pointing is more sophisticated and they showed that they could get better results than the hand-eye line using machine learning that used the hand-eye, hand-elbow and hand-shoulder lines [18].

While the above methods for estimating pointing involve extracting body part locations and constructing a 3D model, Richarz et al.[52] and Martin et al.[39] have extracted pointing targets from 2D images from single low cost webcameras. They used neural networks that take images of the user's head and sides of the body and run Gabour filters over the images to extract orientations from the network.

2.2 Human-Robot Control

Control of robots has several important factors to define for discussion, which is provided here as a simple contribution and illustrated in table 2.1. Control comes in various degrees and is not necessarily counter to autonomy. An example is with a dog, you can teach a

	without head-orientation	with head-orientation
Detection rate (recall)	78%	87%
Precision	83%	86%
Avg. error angle	37°	28°
Targets identified	65%	83%

Figure 2.4: Image from Nickel and Stiefelhagen's comparison of the head-eye line, arm-elbow line and gaze direction and experimental results [47].

dog to obey and do tricks on command. A good owner may be said to have good control over their dog, yet it is difficult to say that a dog is not an autonomous animal. While not wishing to invoke a free will argument, for discussion levels of control will be qualified as direct vs indirect control on a continuum. Direct control is where the user is controlling effectors, such as driving a car where they control how fast the car moves, such a robot relies on the user's senses to make decisions rather than the robot's own sensors. Indirect control is where the robot controls more of the low level aspects of the task, such as the actual path that the robot drives and the user issues a higher level command, such as telling the robot a location to drive to. Another factor that will be discussed through the rest of this chapter is one of presence. Fully present control will be defined where the user is in the presence of the robots they are controlling, the user perceives the state of the robots using their eyes and the robots act on the user present. The opposite of fully present is remotely controlled. Fully remote controlled HRI does not depend on the spatial proximity between the user and the robots in concept (range of communications may be a factor in practice though). In fully remote HRI the user must depend on an interface separate from the robots to perceive

[33][12]	**Present**	[47][52][39] [18] [11][60]*[37][16]*
Direct		**Indirect**
[28][24][57][50]	**Remote**	*[40][41]*

Table 2.1: Two axis description of robot control types with references of examples used in this chapter, italics denote multi-robot work. On one axis (illustrated on the vertical axis) is the Present vs Remote continuum, of whether the user is present in the control of the robots, where he or she is using their own senses to guide the robots, as opposed to remote control where the user is in another location and uses the robot's or other external sensors to guide the robot. On the other axis (illustrated on a horizontal axis) is the continuum of Direct vs Indirect control, which is a distinction of how much the user controls on the robot, direct control has the user manipulating effectors directly where indirect control has the robot doing more to interpret the user's commands.

the robot's state. A special type of remote presence is telepresence, where the user sees and acts through the body of the robot.

2.2.1 Single Robot

Remote direct controls on a single robot are similar to controlling a character in a video game, often from a first person perspective the user controls the movements and actions of the character, only instead of being in a virtual world the robot moves in the real world. Users controlling robots that are directly controlled by telepresence are essentially using the robots as an avatar of themselves. One application is where the robot can act as a character for entertainment with the user controlling them from as distance. The user can animate and talk through the robot giving the impression that the robot is "alive" and autonomous, such as mall guides tested in Japan [28]. In other cases the robots may act to allow the user to extend themselves into a dangerous situation, for example with bomb defusing robots used by the American military [24], a user can operate on a bomb without putting their own body at risk. In other cases the robots can extend a user's presence towards others. An example is found in hospitals with robots that are controlled by a patient's familiar doctor [57]. The doctor's face can be seen on a screen and their voice projected to make the patient comfortable without the busy doctor needing to travel to the hospital.

Some robot controls can have an optional amount of presence. An example is found the Parrot AR.Drone (AR standing for augmented reality) [59] [50]. The AR.Drone is a quadracopter flying robot which promotes itself as "the flying video game" [59]. The

Figure 2.5: Images of the AR.Drone being controlled fully present (left from [59]) and to the right an image of the controls which displays the robots view and allows the robot to be remotely controlled.

AR.Drone can be controlled from a touch screen cell phone or tablet, the view from the robot is shown on screen as well as touch based control (the robot can also be controlled by tilting the controller). The user can either use only the screen and watching the view from the robot, or look at the robot and control it. Similarly medical robots used in surgery can have the surgeon looking at both the patient directly and through cameras inside the body where the robot is exploring [55].

Fully present and direct control of a robot can be done by having the robot imitate the user's movements exactly. Koenemann et al. used a Kinect sensor and mapped a user's body movements onto a humanoid robot [33], making the robot do whatever the user did. Less direct control can be used similarly with non-humanoid robots which must interpret human actions into something it can map on to its non-humanoid body (an example [12]), figuring out this mapping from one body to another is termed the correspondence problem [46].

Fully present and more indirect control can be found when a user is standing in front of a robot and the robot interprets what the human wants. An example of what would be fully present highly indirect control would be robots that reacted to a user falling and helping them up. More commonly seen are robots that interpret gestures and perform commands on what they interpret from the gesture. Several of the works described in section 2.1.1 above all use pointing gestures that are interpreted by a robot as a control scheme [47] [52] [39] [18] [48] [11]. Van Den Bergh et al. used pointing gestures to pass a robot along to other users [60]. With their robot interaction a robot with a Kinect sensor mounted on it drove around a real university environment, when the robot detects a human (by face detection) it

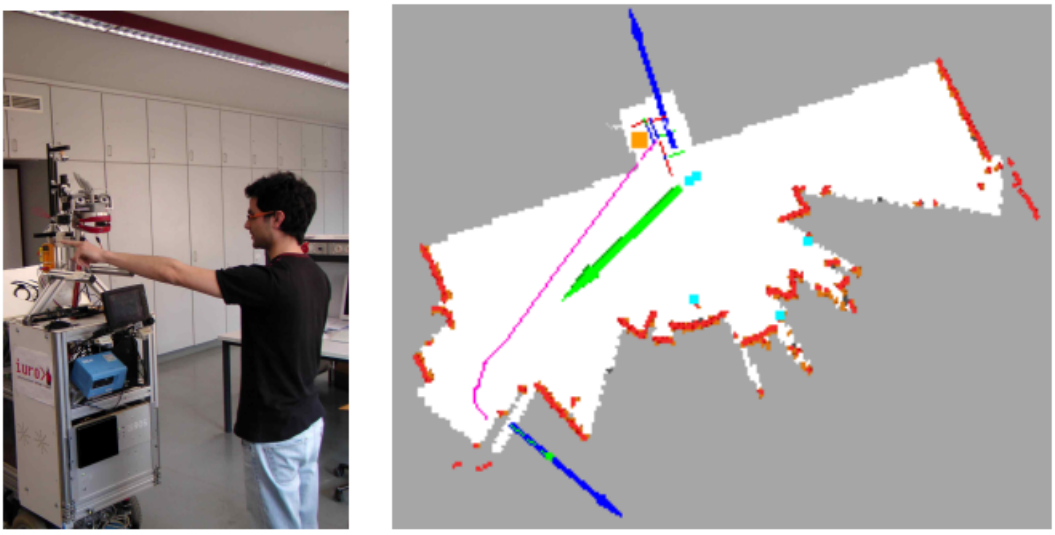

Figure 2.6: Images from Van Den Bergh et al.'s paper [60]. Robots are directed by a user by pointing, the robot interprets the pointing gesture to decide which location to travel to. Left a user can be seen interacting with the robot. Right a map showing the users pointing direction (the green arrow) the potential locations to explore (the blue arrows) and the robots tracks (the magenta line).

looks to the human for directions. Directions consisted of hand poses, particularly pointing gestures. With the Kinect they were able to extract the hand region and classified its pose, a closed hand with extended index finger indicated a pointing gesture. The direction the user was pointing was estimated using the vector made up of the hand to wrist, and the robot travelled to the next location nearest to that pointing direction seeking out the next human.

2.2.2 Multiple Robots

Direct control of multiple robots at once is much more difficult than with a single robot. Zheng et al. [65] did a study using multiple guide robots controlled by a single user, where the user had to drive each robot and carry out conversations on the robots behalf, and found that users could not maintain performance when operating more than one robot at a time. More ordinarily multiple robots may be overseen by a single supervisor that can take control of a robot at any given time, while they drive autonomously otherwise. Chen et al. [26] has looked at attention switching in the case of a supervisor overseeing surveillance

from multiple robots where the user would take control in case of the robot being in danger or a significant event. They found the design of alerts to be key to keeping high robot to user ratios, particularly that alerts must not be too many to overwhelm the user while not being so few that important events go unreported.

If remotely controlled methods for single robots are analogous to first person video games, remotely controlled methods for multiple robots are analogous to real-time strategy games like StarCraft [20] with a third person interface. In StarCraft the user views the action from a birds eye view and selects one or more items and/or characters on the screen using a mouse and then controls them with subsequent mouse clicks to activate items or send characters to locations. A similar method was implemented by McLurkin et al. to control robots [41] that they call "SwarmCraft". With the SwarmCraft system the user sees a representation of robot locations on a screen from a similarly overhead view, and can click on robots to view the robots state and can issue commands to a single robot from a swarm. Kato et al. implemented a similar system with overhead view control. In their system a multitouch interface displays the robots and their environment from an overhead camera system, robots are controlled by manipulating a vector field with the multitouch table that control how fast the robots move through the vector field [30]. McCann et al. designed a multi-robot multi-user control system that could control individual or groups of robots using the Microsoft Surface and touch screen tablets [40], their system works with a fully detailed robot view (simulated) and can move camera controls around the virtual environment, drop beacons to attract or repel robots from a location or even take control of a single robot and drive it with direct control [40].

Very little work has been done with present control with multiple robots. Payton in his work with Pheromone robots [49] had a control where a user could issue commands to an individual or to an entire population of robots with an IR LED remote. The remote would activate an omni-directional beam to send signals to entire group of robots, and a narrow beam to communicate with an individual. This control method was not published but shown to us through personal communication. Alex Couture-Beil et al. from the SFU Autonomy lab pioneered work on selecting an individual robot from a population and assigning it a command using a natural user interface [16]. In this demonstration a user could select an individual robot from a population by focusing their gaze (inferred from frontal face detection) on the robot that the user wanted to select. Once selected the user could issue a command to that robot using a hand gesture as can be seen in Figure 2.8. Similar recent

work by Litchenstern et al. uses four quadrocopters flying robots and a Kinect to perform a selection and commanding task [37]. In this work three of the robots are selectable and the fourth holds the Kinect to observe the user using skeleton tracking. Robots are selected by the user pointing to robots with his or her right hand and then tapping his or her right arm with his or her left hand to confirm the selection. The selected quadracopters are then controlled by moving relative to the motions of the right hand.

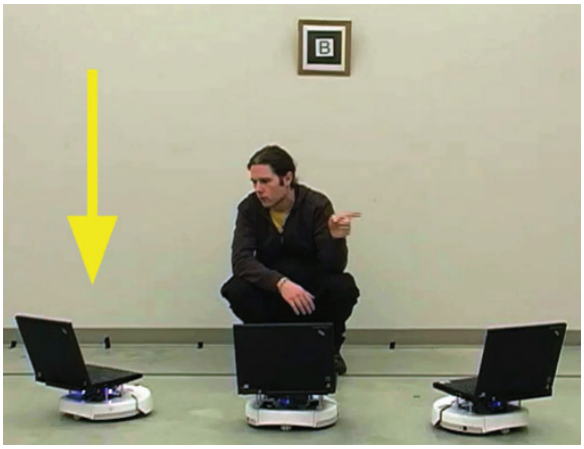

Figure 2.7: Alex Couture-Beil selecting a robot by looking at it and commanding it with a hand gesture. [16]

2.2.3 Relevance

The work in this thesis contributes to the little explored present indirect control of multiple robots. It continues from the work done by Couture-Beil et al. [16] but goes farther by allowing the user to select and command groups of one or more robots from a population rather commanding one robot at a time. The only other work with selecting multiple robots and controlling the selected group was done by Lichtenstern et al. which followed the publication of the video describing this work, and sites it as such in their paper [37].

We also follow from the pointing literature. Using Nickel and Stiefelhagen [47] discovery that the head-hand vector is the best single estimator of a user's pointing direction is used as an essential feature to our selection method, and we show how it can be used to reference an area as opposed to just a point.

Chapter 3

Selecting and Commanding

3.1 Overview

In this chapter a new method for selecting groups of robots is described. This method is the first of its kind where a user can select robots without using any instruments external to his or her body. To select the robots the user uses a natural gesture of circling the robots with their hands. Building off the properties of pointing that were discussed in Chapter 2, it will be shown that a robot can determine whether it was circled by the user by tracking only the user's hand and face. From the robot's view, if the user's face can be found inside the shape formed by the hand tracks, it was selected by the user.

This method is dynamic in that the robots are not pre-assigned groups. For example the grouped robots do not all share a common attribute in a database, so we cannot simply call out a command like "blue robots perform task A." Instead the robots are grouped at runtime and the group is then assigned to a task.

3.2 Selection Method

3.2.1 Selecting Groups on Screen

In most graphical user interfaces such as Microsoft Windows, Apple and Gnome desktop, you can select groups of icons by drawing a shape around the group, as seen in Figure 2.1 where a rectangle is used in windows to group together a subset of icons on the desktop, a technique called the "rubber" band graphical user interface control. The same approach

Figure 3.1: Different methods of selecting objects on a screen and the robots we want to select.

is used in real time strategy video games like Starcraft where the user selects a group of soldiers or robots using their mouse to draw a rectangle and sending those troops into battle. A more flexible area selection is used in paint programs such as Adobe Photoshop that use a "lasso" selection where the user draws a shape around an area, selection is completed when the lasso's shape is closed, all pixels within that shape are then selected and can be manipulated as a group. Our method similarly involves drawing a shape on the floor around robots and robots within that drawn shape are selected to be a part of the group.

3.2.2 Drawing a shape around robots

Since we are drawing our shapes on the floor without the user possessing any devices on their body, we must rely on external sensors in the environment to interpret a user's intention. By having the external sensors mounted on the robot we achieve a sensing method that is robust to environmental variation, as there are no special properties to the floor or overhead view. This also provides a method most true to robotics.

To draw the selection shape we have the user point around the area that he or she wishes to select. This creates the perimeter of the selection area, and like the lasso method is completed when the shape is closed.

To interpret where the user is pointing at a given moment, the line from the user's eyes

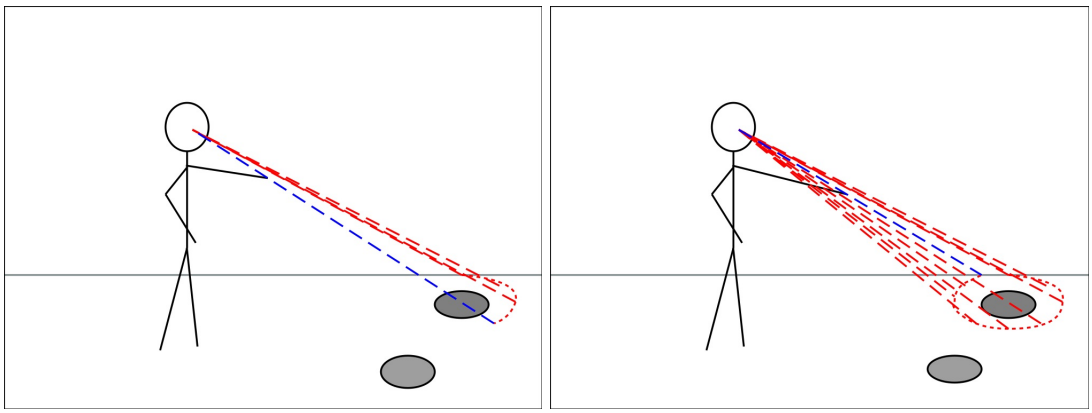

Figure 3.2: In selecting a robot using a circling gesture, they are caught in a "cone" if selected.

to hand is used. Using this line is the single best estimator of a user's intended target [47] (see Chapter 2) and also provides essential properties for how our method is implemented as discussed later in this section. Therefore to implement our method the robot must be able to track a user's hand and the location of their face.

If the reader is to imagine drawing such a selection shape on the floor, extend a hand and look the floor where your finger is. Move your hand to trace the selection shape on the floor, a typical shape would be a circular shape around the desired robots that you want to group. Figure 3.2 illustrates this.

By looking at the user's hand-eye pointing lines that create the 2D shape on the floor, one can see a 3D cone like shape is formed with the drawn shape forming the base and the users face forming the vertex, and the hand motions form the ellipse describing a slice through the cone. Robots selected in the shape drawn on the floor's area are then also contained within the 3D cone like shape's volume.

3.2.3 The Robots' Point of View

For the robots on the floor, one strategy is to construct the cone in a 3D internal representation of the space around them, but we propose a simple strategy that makes this unnecessary. Our method allows the robot to determine if it is selected using a minimal model of the environment, and can be implemented using a single low cost web camera mounted on each robot.

For the robot looking up at the user from a camera it needs to approximate the location

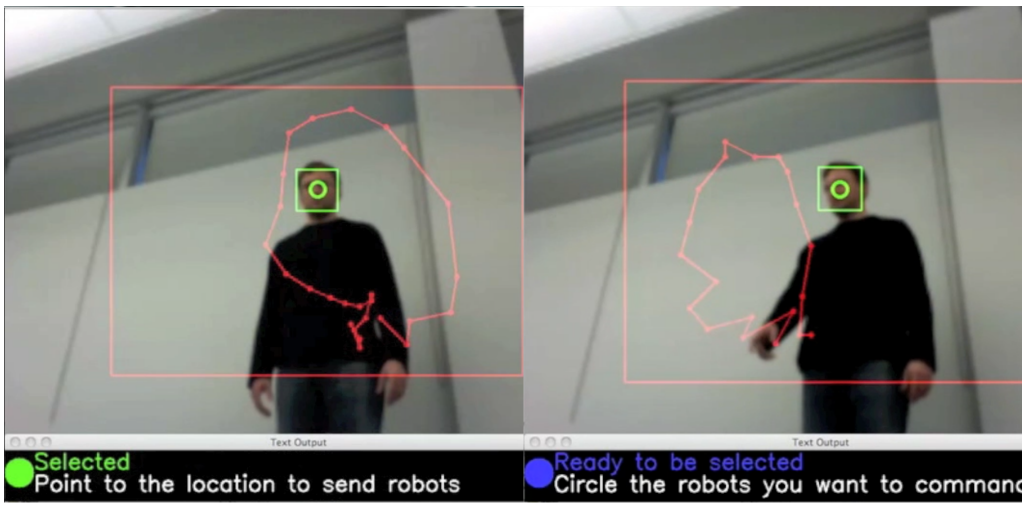

Figure 3.3: An example of a positive selection (left) and a negative selection (right).

of the user's eyes and the location of the user's hand when they are potentially making a selection. When following a moving hand it will watch for a motion that completes a 2D polygon within its view. The hand motion describes a slice of the cone, so the robot should then only have a roughly elliptical shape and a face point in its model of the motion as can be seen in Figure 3.3. Our innovation is that we can infer selection simply by testing if the user's face is found within the polygon drawn by the hand motion without any explicit 3D information required in the camera's view. Therefore we never explicitly use the cone instead just do a test to see if a point (the face) is within a polygon (as formed my the hand motion).

While this method does not require depth information, it should be noted the errors that would be seen in constructing the cone projection would also affect this method. Errors in the location of the face or hand will propagate along the head-hand line that would affect the polygon and face point and would be more evident in far range selections.

3.2.4 Geometry

Geometrically what we want to show is that we can determine if a point, (in this case the focal point of the robot's camera) is within the volume of the cone using only a two dimensional representation of the vertex of the cone and an arbitrarity slice parallel to the camera's frustum.

We begin with showing how to determine if a point is within a 3D volume if we had the entire 3D model available. If we describe the 3D shape as a cone, the shape of the hand motions would be a conic section that projects the shape of the cone to the floor. Since it is assumed that the robot is above the floor, it is unnecessary to test if the robot is between the vertex and base, instead only if it is inside the surface of the cone's vertex-boundary lines.

Take any plane that would include the test point and would intersect all of the cone's boundary lines and we would have an ellipse and the test point on that plane. We now do a 2D test on that plane to see if the point is within the ellipse to arrive at a result for the test.

Now consider testing if the line formed by the test point and the vertex is surrounded by the cone's surface. This becomes the same test at the test point, but also requires that all points along the vertex-test line also are within the cone's surface. Since all lines connect at the vertex point, they cannot intersect at any other point, therefore all points on the test line will be on the same side of all the points of the surface boundary lines. This means all points on the test line should be within any slice of the conic section boundary lines when we perform the 2D ellipse test. From the camera 2D perspective then, we can test the vertex itself (the face) with any conic section (the hand movements), making our face within the hand motion polygon a valid test if the robot's camera center is inside the volume of the cone. As mentioned above, this includes whether the robot is in the shape drawn on the floor around the robots.

3.3 Commanding

With some subset of robots selected, a command can be issued that instructs all of the selected robots at once. This command could be issued by voice or another gesture maintaining a purely natural user interface. Important to the role of the group though is that all robots in the group agree to what that command is, or the group may operate in conflicting ways. The command can be interpreted by a lead robot that conveys to the rest of the group or the command can be decided by the group through some form of election. Unselected robots can also participate in interpreting the command, and in fact may take a counter action as an unselected robot. The important point towards implementation is that while selection can be decided by each individual in the population (am I in the group or not),

deciding on the common goal of the group requires some form of communication between the robots.

In implementation we used pointing as the command input. After selection the user points to one of two target zones and selected robots go to the target zone that was pointed to, as will be discussed next alongside the full implementation of the method in Chapter 4.

Chapter 4

Demonstration

4.1 Overview

To demonstrate and validate the method of robot selection described in Chapter 3, we implemented the method using four mobile robots and recorded the results. In the demonstration robots are selected into a group by a user by circling the desired robots. The selected robots are then given a task by the user by pointing to one of two locations and the robots travel to that location and return to the location where they were commanded from.

4.2 System and Setup

4.2.1 Robots

We used four iRobot Creates, which have custom-built extended processing via an onboard Gumstix computer that controls the robot's driving, collision avoidance and LED lights. A Netbook was attached on top of each robot (Asus EeePC 1201N, with a Intel Atom 330 1.6Ghz dual core processor inside) and a webcam (Logitech 2mp Webcam C600) to together handle image processing.

4.2.2 Arena

Our robot experiments were all carried out in the Autonomy Lab at Simon Fraser University. This lab features a large open area that serves as our robot arena, with three walls and a divider to keep the robots from escaping. An overhead camera system reports and broadcasts

Figure 4.1: One of the robots used for the demonstration.

the locations of robots in the arena, which is used for robot localization. This is accomplished through six overhead cameras that track fiducials using the ARToolkit Open Source library [29], then translates the location of the fiducials in the camera space to arena coordinates and updates a central database. The database can then be queried by the robot to localize themselves. In a more real world scenario we would imagine localization and mapping would to be accomplished by an established method such as SLAM [19], but since localization is not our focus for this demonstration we used the overhead camera localization as a proxy.

4.2.3 Communications

General communication is done over WiFi, the netbooks and gumstix board on each robot have 802.11g WiFi connections and the lab/arena have WiFi coverage via router, that also connects to the tracking database. Passing of messages is all done through Redis database commands over WiFi [2] which connect to a central Redis database running on the tracking system.

Figure 4.2: A screen showing how the program identifies and scores a face.

4.3 Selecting and Commanding Implementation

Selection is done in two discrete steps. Firstly the robot must locate the user's face, once the face is located the robot looks for a moving hand, if the hand completes a closed shape while keeping above a minimum hand speed, the robots then test if the face is within the hand motion's shape to decide if they belong in the selected group or not. After at least one robot is selected the group will then listen for a command.

4.3.1 Finding User's Face

Face finding/tracking was implemented through modifying the face tracking example found in the Open Source Computer Vision library (OpenCV)[9], which implements the Viola-Jones face detection algorithm [61]. The OpenCV function returns a score around an area where it detects a face, this score is based on the number of overlapping face detections it found in the area. We get higher scores for better face detections because the detector finds faces at different scales and pixel translations. The robot chooses the face with the highest face score within its view as long as that face score is above a minimum threshold. If there are two people in the frame the closer person would most likely be chosen, but this demo assumes there would only be one person in the scene at a time. To adapt this to with multiple potential users, an enrollment technique such as waving to take focus could be used.

Once a user's face has been found, the program takes a sample of the user's skin tone from his or her face, which is used for hand tracking.

Challenges with Face Detection

Face finding was the most challenging and troublesome part of the demonstration. While the face finding would almost always work it was very slow, and finding the right minimum face score threshold was difficult to balance the time it took to find a good face and avoiding false positives. We believe this is mainly due to the robot's perspective from the floor. The training set used in OpenCV consists of pictures of front facing faces taken from a level roughly the same height as the person's face. This could be seen when testing the program on a desk rather than the floor, on a desk level the face was found with in a second or so when the user looked at the camera, from the floor it could take 10-20 seconds (both using the same netbook for processing). This could be improved by creating a new training data base of pictures of faces taken of people standing and the camera near the floor. Furthermore using a faster computer also alleviated this delay, as my 2009 Macbook Pro (Intel duo 2.66ghz) was able to detect faces using the program within a few seconds.

4.3.2 Tracking User's Hand

Hand tracking was done using a combination of colour filtering (using the user's face colour) and optical flow tracking. Once the face was detected and the skin patch taken, a filter is applied to the video frame removing all pixels outside of a range of the median of the face colouration. This can be seen in the top right window of Figure 4.3. The filtered skin patch image is then passed to a motion filter, the motion filter compares the last frame skin patch frame with the current skin patch frame and utilizes the OpenCV implementation of the Lucas-Kanade optical flow algorithm [27]. The optical flow procedure returns a 2D velocity for each of the current skin pixels, and ranks the fastest moving pixel on the screen. All of the skin pixels' velocity values are then divided by the fastest value, and the program weighs all of the skin-pixels by their velocity and chooses the hand location to be the fastest moving skin patch area. The bottom right window in Figure 4.3 visualizes this, inside the red circle are white specs that represent the fastest moving skin-pixels, and the large colour window to the left shows where it placed the moving hand.

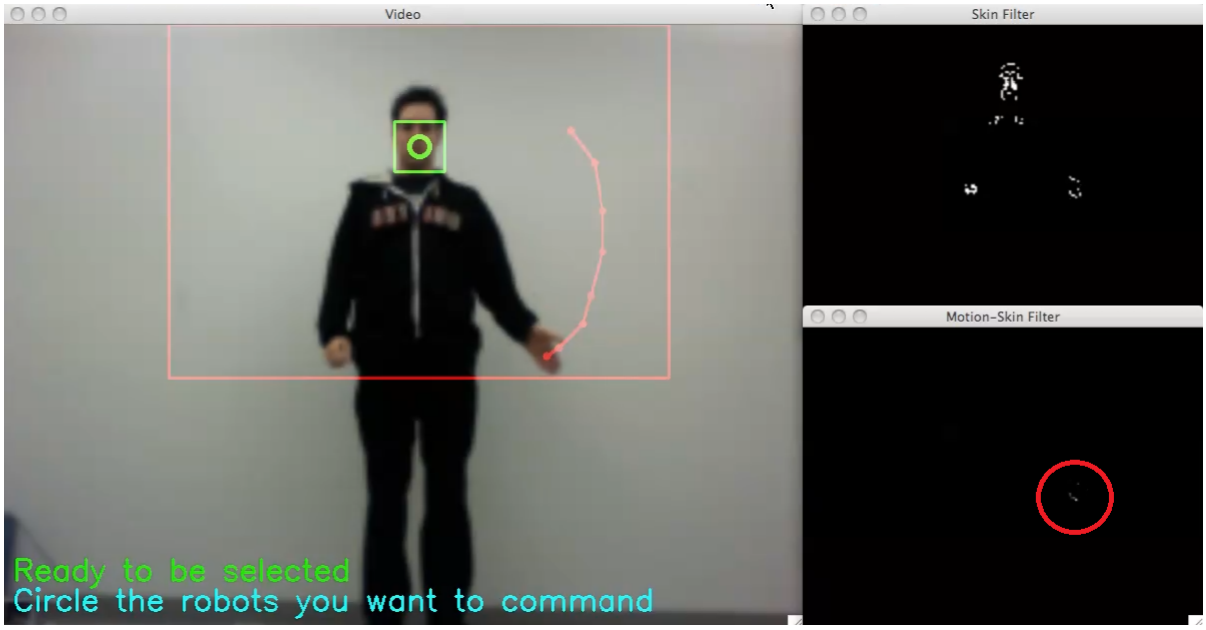

Figure 4.3: Visualization of hand tracking filters. Left is the resulting hand track, top right is the frame after filtering out pixels that are not the users skin tone, bottom right shows the remaining pixels after the motion filter is applied to the skin pixels.

Challenges and Limitations

The hand tracker works quite well, but required a lot of tweaking for performance as the netbooks were too slow to handle the image processing at the full resolution (2MP) of the web cameras. This was overcome by scaling the images down and cropping to a useable area around the face (the red rectangle in Figure 4.3), resulting in a 50x50 pixels frame that maintained accuracy and good frame processing frequency. Choosing the range of the skin filter was all done empirically using the hardware described above, and further tweaking may be required if the program were running on different cameras.

This method also suffers from issues of dependence on colour filtering by skin. The user is required to wear long sleeves and wear clothes with colours salient from their skin tone. Also backgrounds too close to the skin tone will also degrade performance. Also it is assumed that the user's hands are a similar tone to the face, which is not always the case. This being said, our intention is to show the potential of our selection method, not the robustness of our hand tracker, and any hand tracking method can be used in place of this one.

4.3.3 Gesture Detection

Once the face and hand tracking positions became available, gesture detection interprets the position sequence. A gesture in the program is considered a sequence of frames where a hand is moving above a speed threshold. A gesture begins when the hand moves fast enough, and ends when the hand stops moving faster than the lower threshold. To avoid throwing out a gesture on a bad frame, the hand must be below the lower threshold for a particular number of frames.

Following a gesture sequence, the program determines if a selection gesture was made. In earlier iterations of the project, the program would use machine learning to classify different gestures based on the hand motion sequence, but at the time of preparing the demonstration this step was removed as it did not contribute to the selection demonstration. In testing some simple hand crafted rules were able to classify a selection gesture.

1. Was the gesture long enough? A gesture that only lasted a few frames was most likely noise or incidental movement

2. Did the hand move far enough? A motion that was only in a small area could not sweep enough ground to make a selection

3. Did the motions close a shape? The selection gesture requires circling an area, if the shape drawn was not closed then selection cannot be performed

Thresholds for each of these criteria were chosen by trial and error from recorded video of several users making different hand gestures.

The command gesture involved pointing to a target on the left or right side of the user. To infer this gesture the program simply watched for a hand motion that travelled far enough laterally to the left or right side of the user's face.

4.3.4 Gesture Interpretation

Once the robot recognized a selection gesture, it would test if the location of the face was inside the shape of the hand gesture, Figure 3.3 shows a positive and negative selection. Each robot makes this determination independently and reports to a central server whether it was selected or not, more details can be found in Section 4.4.6.

The command gesture is scored by how far the robot saw the hand move laterally from the face and reported to the central database. All robots that registered that a selection

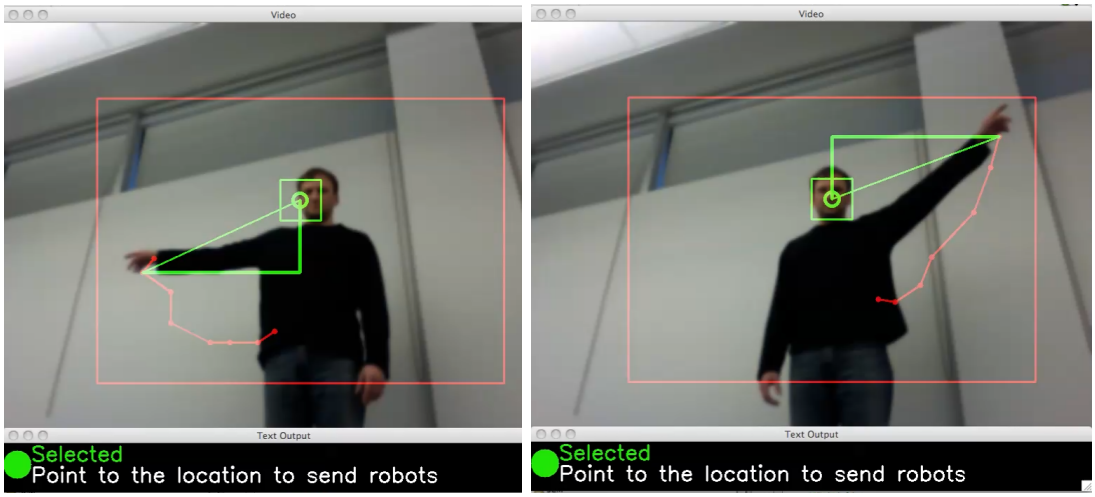

Figure 4.4: Examples of how the pointing gesture is measured as the distance the hand travelled from the face.

gesture was made contribute to reading the command gesture, whether they we selected or not.

4.3.5 Human Robot Interaction

There were two methods used for communicating the robot's state with the user. The first was with the netbooks open and displaying commands and states to the user on screen with large coloured text. The screen could also optionally show visualizations of how the robot was reading the user's movements, as has been shown Figure 3.3, 4.2 and 4.3, which was useful for debugging. The problem with using the on screen data was to make the robots move around in the environment, we needed the laptops shut to display the fiducial on top for the overhead tracking system. To communicate to the user with the netbooks closed, we used the light emitting diodes (LEDs) on the robots to display their current state (see Figure 4.7 for the robot's state diagram). The sequence the robots and the users interacted were as follows

1. **Face Finding**: In this state the robot is trying to find the user's face. On screen it displays a message "Searching for Face, Look at me", the LEDs flash Amber on and off.

2. **Ready to be Selected**: In this state the robot has found the user's face and is watching for a selection gesture. On screen it displays a message "Ready to be selected, Circle the robots you want to command", the LEDs are lit up a solid orange.

3. **Selected / Not Selected State**: One of the robots has been selected, the robot is waiting for a command gesture. If this robot was selected it displays "Selected, point to the location to send robots" and the LEDs are lit up a solid green, or if this robot was not selected it displays "Another selected, point to the location to send robots" and the LEDs are lit up a solid red.

4. **Work State**: After a selected robot receives its command, it displays "Working, going to target on my left/right" and the LEDs are lit up a solid blue.

Figure 4.5: The LEDs on the robots indicate what state they are in

4.3.6 Multi-Robot State Synchronization

The robots must also communicate with each other in this implementation to communicate when a robot was selected, and to come to a team decision on what command they are executing.

1. **User/face Finding state**: This state is independent in each robot and they do not communicate when in that state. Future implementations may want to tell other robots where they found the user.

2. **Selection State**: In this state the robot is watching for a selection to be made. If another robot is selected it will send a message to the central server that will begin a

timer interval. If this robot did not read a positive selection by the time the interval expires, it goes into "another selected - Read Command" state, and watches for a command gesture to help the other robots to interpret it.

3. **Selected State**: If the robot is selected it informs the central server and waits for other robots to declare themselves selected during the timer interval. After the interval expires it goes into "Selected - Read Command State"

4. **Read Command State**: The robot is watching for a command gesture (lateral hand motion.) When it reads a command it sends a message to the central server saying it read a command and how far the hand moved later to the user's face and a timer interval is started

5. **Command interval expires**: When the command interval is completed, the robots all look at the database to read what each of the other robots saw for the command. A decision is made using the collected weighted distances observed from the user's face, and the robots agree that the selected robots should go to the left or right target. The motivation for using the weighted distances is that a robot with the most orthogonal view to the user's pointing direction should have the best estimate of where the hand traveled to. Also this prevents two unsure robots, that maybe only saw the hand go slightly outside the shoulder, from overruling a robot that saw someone point to the far left side.

6. **Work state**: Selected robots then perform the task, and the unselected robots return to the User/face Finding state.

While we used a central database to pass messages between the robots for convenience, this could have been done only using the robots. This could have been accomplished by having one of the robots host the data base or messages could have been coordinated by using a distributed leader algorithm such as [35].

4.4 Robot Work task

Upon command the robots travel to one of two locations (either to the left or right of the user) then travel back to the location that they originated from in front of the user. The robots used the overhead tracking system to localize themselves and orient themselves

towards their target location. When the robot arrives within a certain radius of the target zone, it drives to a waypoint (to reduce traffic between coming and going robots) and then returns to the location it originated from. The robots each have an array of infrared sensors that can detect if another robot or obstacle is near it, if an obstacle gets too close the robot will go into an obstacle avoidance mode and steer away from the obstacle, once the obstacle it far enough away it will return to driving towards the target destination. Figure 4.10-4.18 shows tracks of where the robots travelled in response to a selection command (illustrated as a blue circle) and a command (illustrated with an arrow to the left or right of the stick figure that represents the user), Figure 4.9 describes what is represented in these figures.

4.5 Trials

4.5.1 Selection

To test performance of the selection system, four robots were arranged around a user and selected the user selected between 1-4 robots in ten permutations. Naming the robots as robot 1-4 with robot 1 being the robot to farthest to the left of the user and increasing to four on the robot to the farthest right of the user, the permutations were select one robot, 1,2,3,4, select 2 robots, 1 + 2, 2 + 3, 3 + 4, select 3 robots 1 + 2 + 3, 2 + 3 + 4, and select all four robots.

In 5 trials through all of the permutations (50 total selections) 41/50 were completely correct selections, while 9 selections either missed one robot from the selection or a robot was incorrectly selected. These results are presented in Table 4.1. Most of the mistakes occurred on the robots near an outside boundary of selection, either being missed selection or a false positive selection. Two of the errors where an interior robot was missed in selection could be seen to be hand tracking failures where the hand was lost near the user's face.

R1	R2	R3	R4	correct	Errors Made
1	0	0	0	3/5	1100, 0100
0	1	0	0	4/5	1100
0	0	1	0	5/5	
0	0	0	1	4/5	0011
1	1	0	0	5/5	
0	1	1	0	5/5	
0	0	1	1	5/5	
1	1	1	0	2/5	1100*2, 1010, 0110
0	1	1	1	5/5	
1	1	1	1	4/5	1101

Table 4.1: Results of selection trials. The four columns to the left represent the intended targets for selection, with R1 representing robot 1, the robot farthest to the left of the user, and the robot numbers counting up to R4 with each robot to the right incrementing by one. This set up that can be seen in Figure 1.1. Columns with 1s in them represent intended selection targets and 0s represent robots that were not supposed to be selected. The "correct" column shows how many trials had all of the robots selected correctly, and the "Errors Made" column shows what was selected/unselected when there was an incorrect selection made, in the same format as the "R" columns.

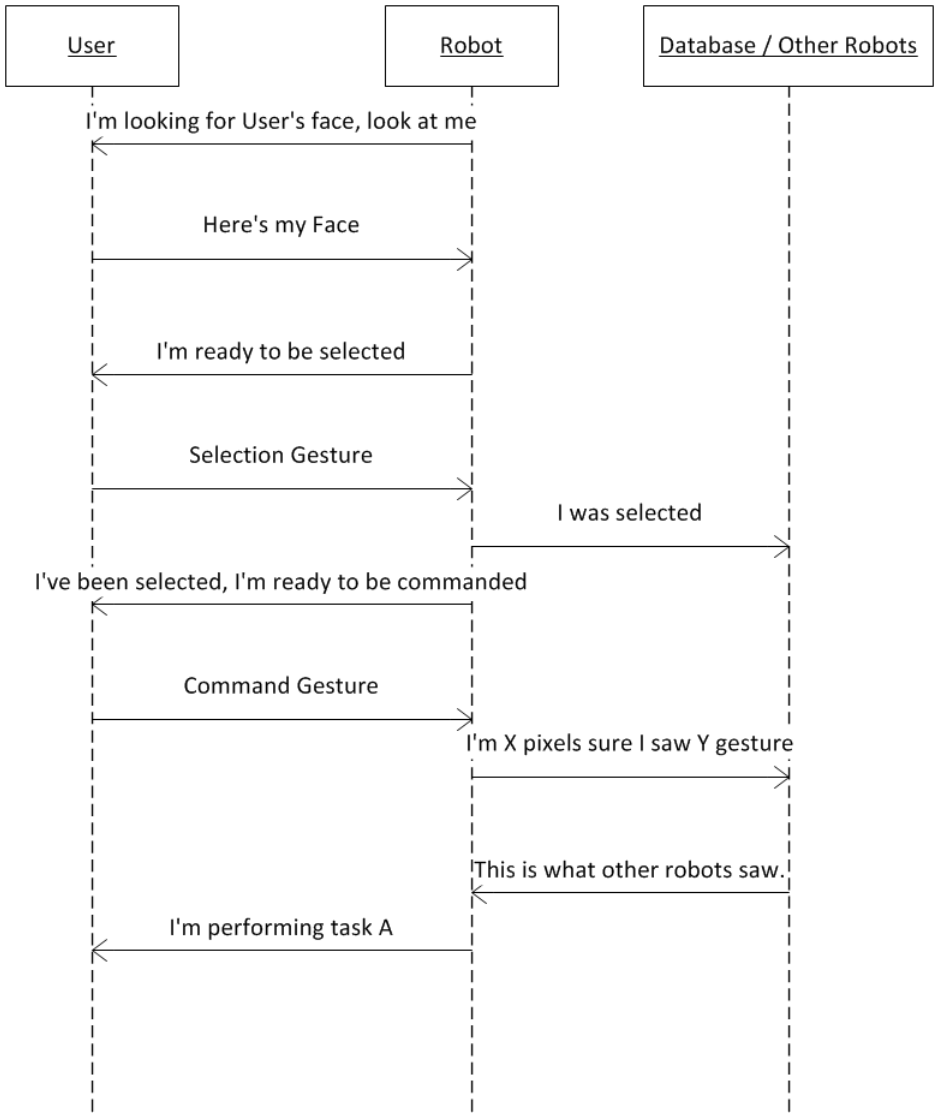

Figure 4.6: UML diagram of communication between the user and robot and the robot and the other robots.

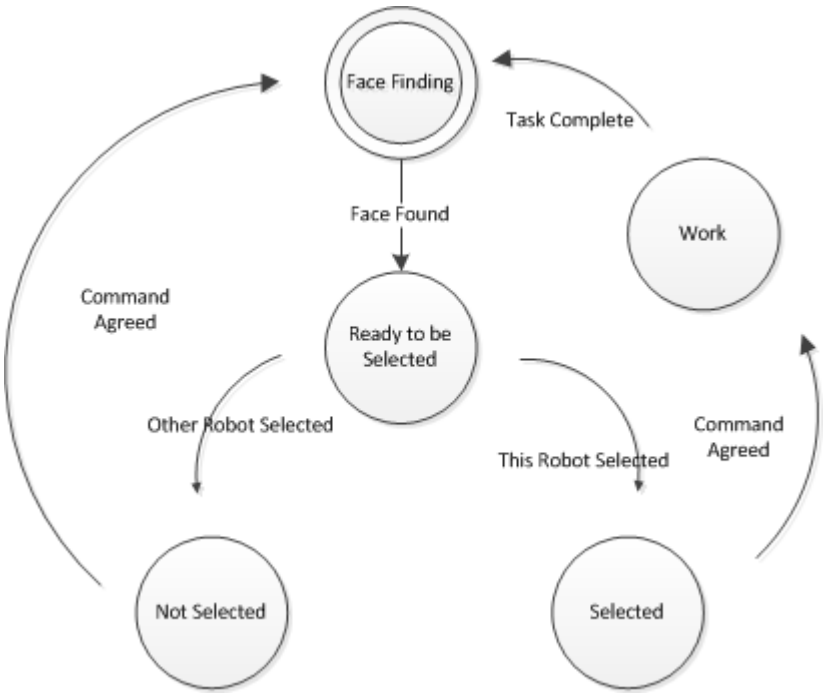

Figure 4.7: State Diagram that one robot goes through during a selection and command of a robot

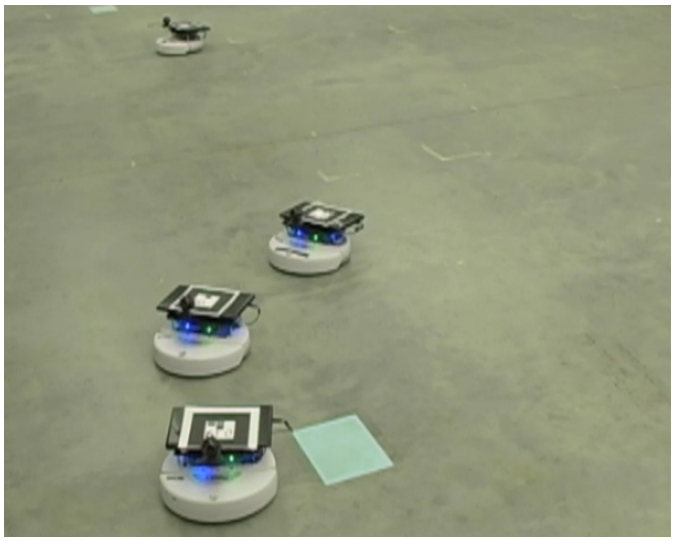

Figure 4.8: Three selected robots driving to the target location (the blue paper), while an unselected robot stays behind (top)

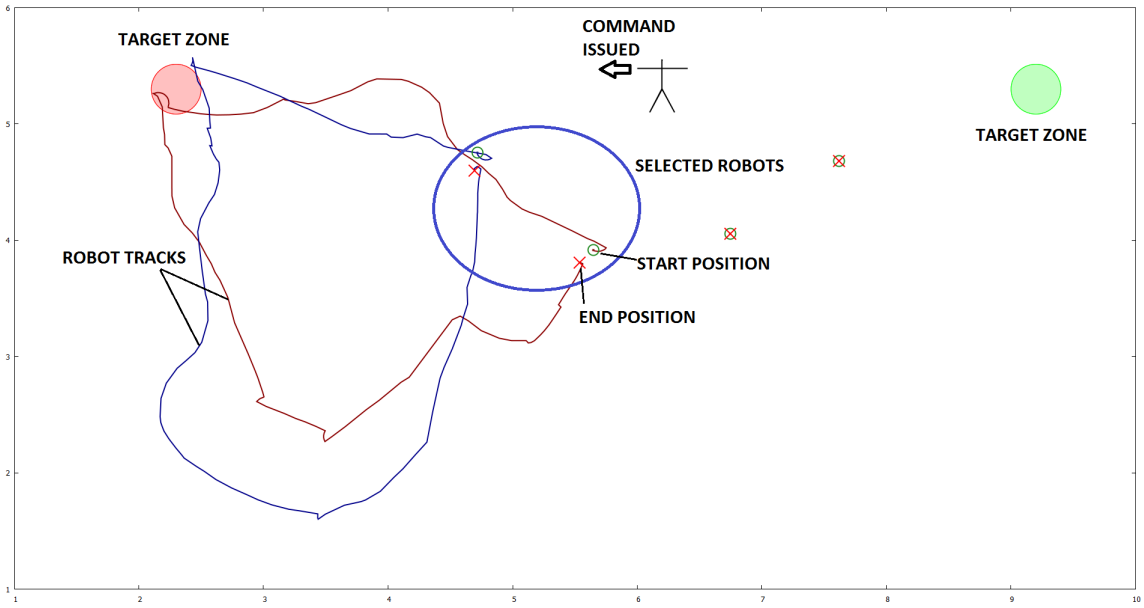

Figure 4.9: Key to the track diagrams. Selected robots are inside the blue ellipse. The arrow next to the stick figure shows which command the user issued. A small green circle represents where a robot started in the scene, a red x represents where they finished. Two circles in the top corners represent the target zones that the robots are to travel to when instructed. Coloured lines show the robots tracks (where they travelled during the scene)

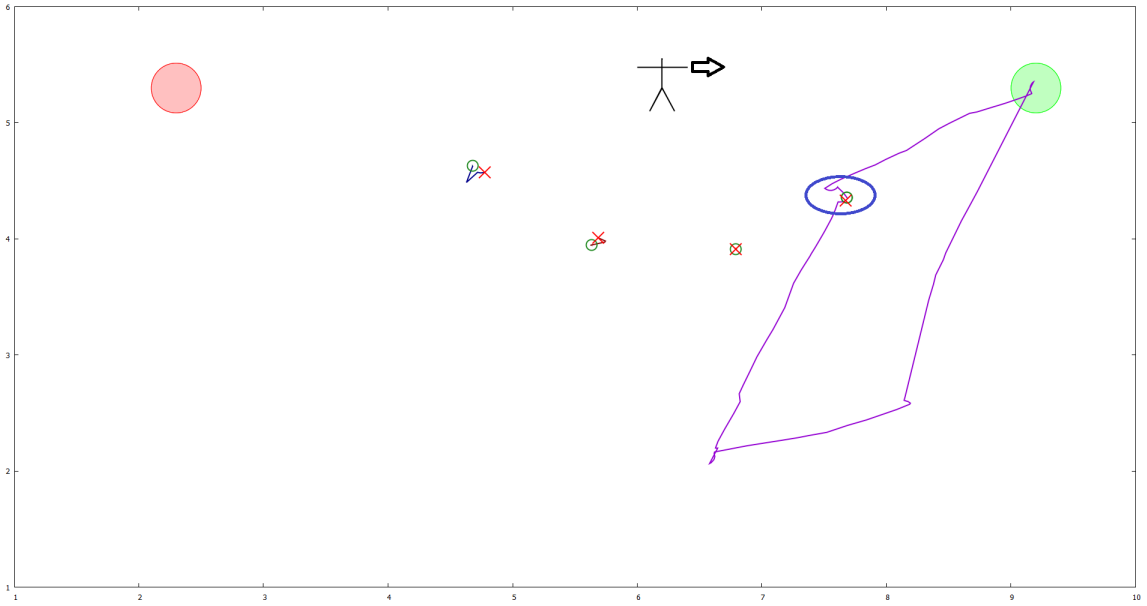

Figure 4.10: User selects Robot 1, then commands it to go to the location to the user's left (Green Target). The robot travels to goal green target and returns to its original location

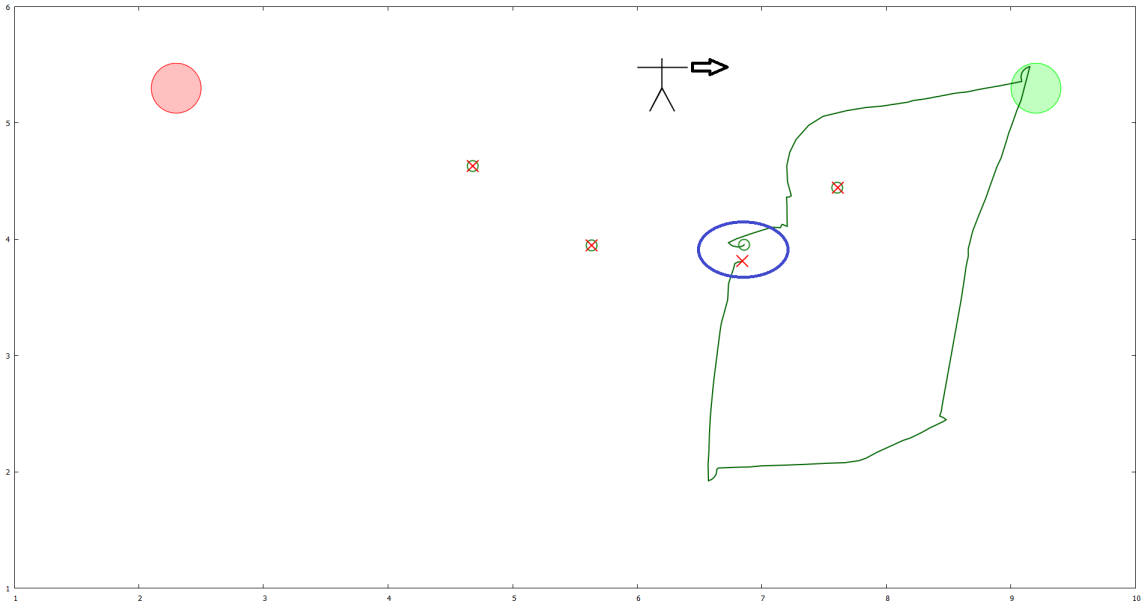

Figure 4.11: User selects Robot 2, then commands it to go to the location to the user's left (Green Target). The robot travels to goal green target and returns to its original location

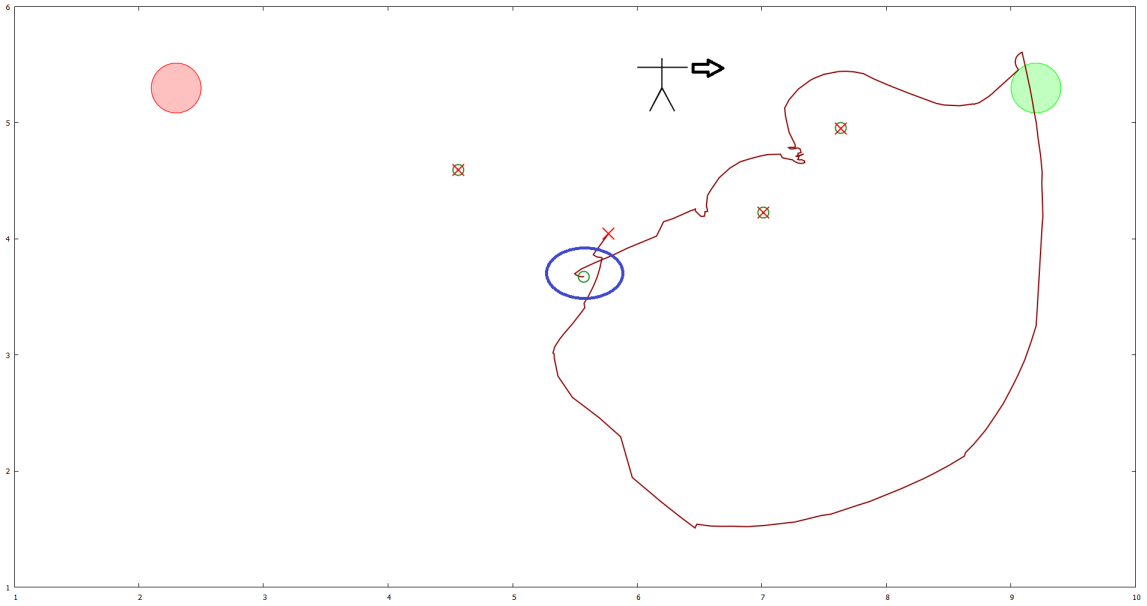

Figure 4.12: User selects Robot 3, then commands it to go to the location to the user's left (Green Target). The robot travels to goal green target and returns to its original location

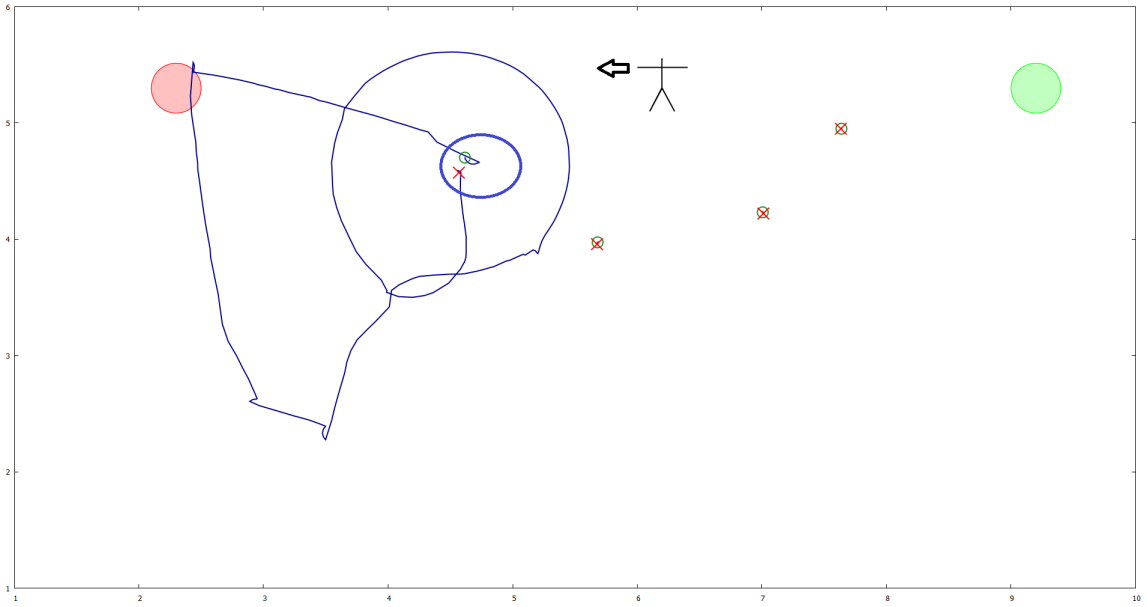

Figure 4.13: User selects Robot 4, then commands it to go to the location to the user's right (Red Target). The robot travels to goal red target and returns to its original location

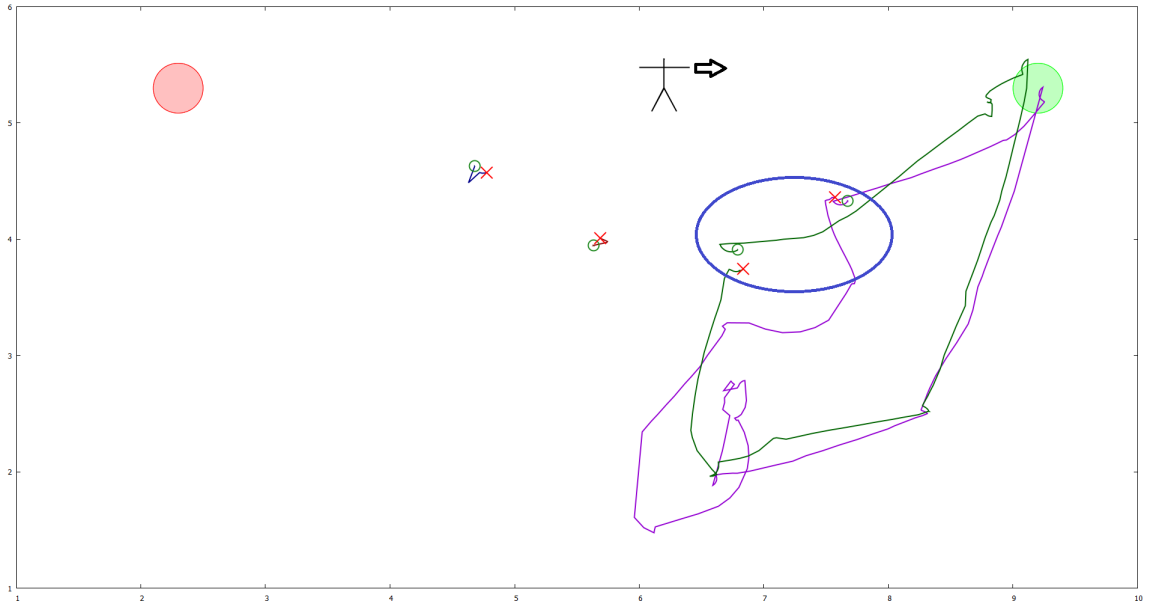

Figure 4.14: User selects Robot 1 and 2, then commands them to go to the location to the user's left (Green Target). The robots travel to goal green target and returns to its original location

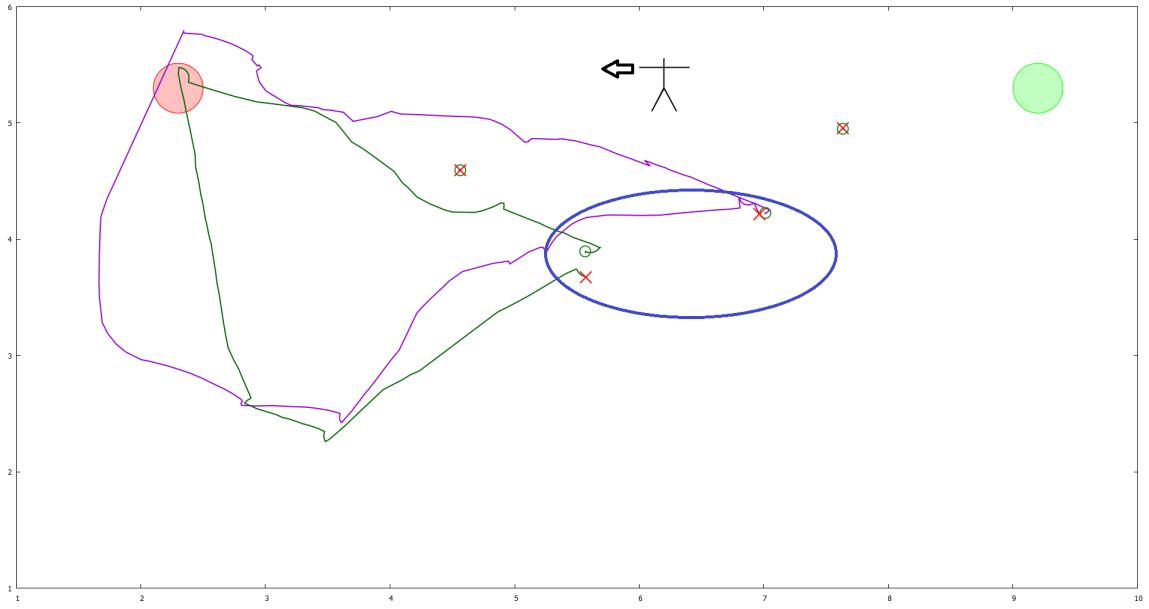

Figure 4.15: User selects Robot 2 and 3, then commands them to go to the location to the user's right (Red Target). The robot travels to goal red target and returns to its original location

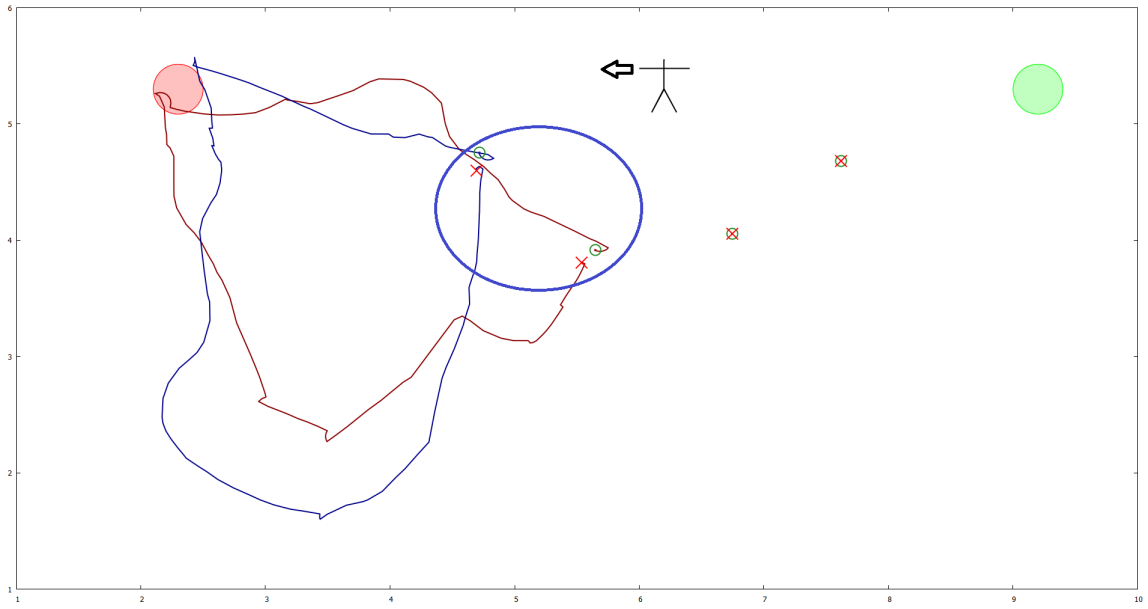

Figure 4.16: User selects Robot 2 and 3, then commands them to go to the location to the user's right (Red Target). The robot travels to goal red target and returns to its original location

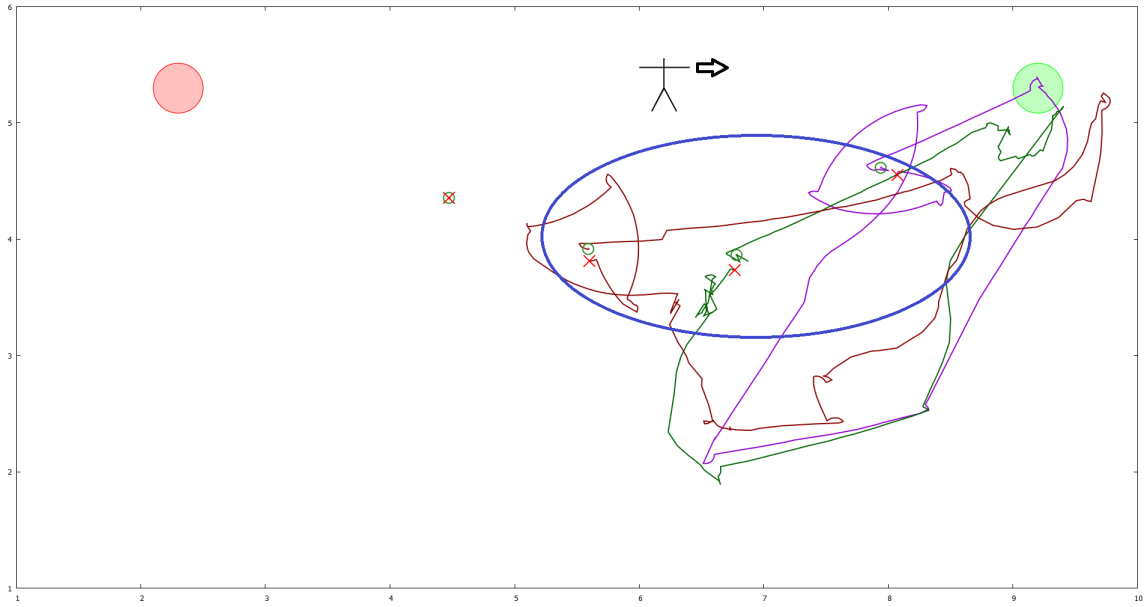

Figure 4.17: User selects Robot 2 and 3, then commands them to go to the location to the user's right (Green Target). The robots travel to goal green target and returns to its original location

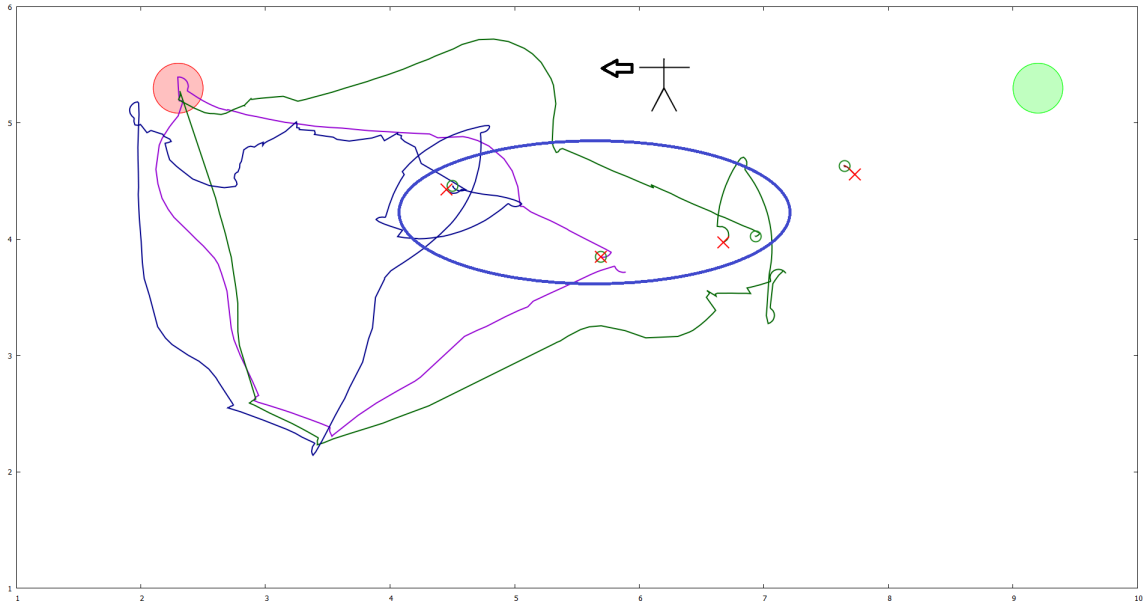

Figure 4.18: User selects Robot 2, 3 and 4, then commands them to go to the location to the user's right (Red Target). The robot travels to goal red target and returns to its original location

Chapter 5

Conclusions

In this thesis the first method of selecting and commanding groups of robots from a population has been detailed, implemented, demonstrated and tested. In Chapter 3 we proposed using a circling gesture to select one or more robots. This gesture allows the robots to use a simple method of determining if they were selected by checking if the user's face can be found inside a sequence of hand positions. In Chapter 4 we showed that the features of this circling method lend themselves to be implementable on basic hardware, namely a web camera and low cost netbook. Also in Chapter 4 we showed in the proof of concept demonstration that robots can be successful selected as a group and commanded using this method. Our method of selecting robots has its basis in the human-robot interaction research on pointing which was detailed in Chapter 2, and contributes a method of indexing not only a point location, but an extended area that is drawn on the floor. This serves to break new ground in the control of multiple robots, which has had limited previous work as discussed in Chapter 2.

5.1 Method

The method described in Chapter 3 has unique qualities that lend themselves well to implementation, and helped to meet some of the challenges in computer vision, natural user interfaces and robotics. Since the hand motions and face location test did not require 3D information, the dimensional reduction problems were not a factor. Using only the hand and face positions saved us from needing to construct a detailed body model, this saved computing power, it also reduces the reliance on finding difficult joint positions, such as

elbows which can compound in error.

The method the user uses to select does not require an external instrument and requires very little instruction. Using one's hand to draw an imaginary circle on the floor is a simple and natural gesture. By using a deictic area relative to the user, the method scales well as selecting 1 or 100 robots depends only on one motion, and does not require much more effort other than drawing a bigger circle.

5.2 Implementation

In implementation some of the challenges as just described were solved implicitly. In implementation colour constancy issues were partially solved by using skin sampling from the users face before selection and using a optic flow to track the hand.

5.3 Future Directions

Other challenges remain for bringing this demonstration to becoming a viable product. Issues of view continue to be problem, in our demo the robots begin with the user in view, and it is assumed the user (as in their face) will remain in the same location after their face is found until they make their selections. A future system where robots can seek out a user, who may be moving about the room would be a challenging and exciting project.

While the hardware used for the demonstration was relatively moderate, it would be interesting to see an optimized version that could implement this method on a minimally priced and/or sized platform. This would open up testing in a larger populations and allow it to be combined with more interesting multi-robot tasking scenarios.

Finally this system has not been tested for usability asides from some informal proof of concept tests done by myself. If tested against controls with full body skeletons it can be determined how well the hand-eye line works for selection compared to other lines such as the forearm vector, or look at how much better a machine learning approach could perform such as [18] did with pointing.

5.4 Summary

This was the first work that dealt with choosing groups of robots from a population using a vision based approach, and its predecessor work [16] was the first to select an individual from a population using a vision based approach. This is an important and underlooked area of HRI. It is hoped that this work will encourage new discussion and provoke competitive ideas in this space. The future of interfaces such as this is an burgeoning field that will contribute to an exciting future full of robots that react intelligently to the mere waving of our hands.

Bibliography

[1] *HRI '11: Proceedings of the 6th international conference on Human-robot interaction*, New York, NY, USA, 2011. ACM. 609114.

[2] Documentation - redis. `http://redis.io/documentation`, Retrieved April 30 2012.

[3] *HRI '12: Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, New York, NY, USA, 2012. ACM. 609124.

[4] C. Angle. iRobot ceo discusses q4 2010 results - earning call transcript. Yahoo Finance Website(Seeking Alpha), `http://seekingalpha.com/article/526661-irobot-s-ceo-discusses-q1-2012-results-earnings-call-transcript?source=yahoo`, February 2011, Retrieved April 30 2012.

[5] C. Angle. iRobot's ceo discusses q1 2012 results - earnings call transcript. Yahoo Finance Website(Seeking Alpha), `http://seekingalpha.com/article/526661-irobot-s-ceo-discusses-q1-2012-results-earnings-call-transcript?part=qanda`, April 2012, Retrieved April 30 2012.

[6] D. Archer. Unspoken diversity: Cultural differences in gestures. *Qualitative Sociology*, 20:79–105, 1997.

[7] I. Asimov. *I, Robot*. Signet, New York, NY, USA, June 1956.

[8] M. Berlin, J. Gray, A.L. Thomaz, and C. Breazeal. Perspective taking: an organizing principle for learning in human-robot interaction. In *proceedings of the 21st national conference on Artificial intelligence - Volume 2*, AAAI'06, pages 1444–1450. AAAI Press, 2006.

[9] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[10] C. Breazeal, C.D. Kidd, A.L. Thomaz, G. Hoffman, and M. Berlin. Effects of nonverbal communication on efficiency and robustness in human-robot teamwork. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005. (IROS 2005). 2005*, pages 708 – 713, August 2005.

[11] A.G. Brooks and C. Breazeal. Working with robots and objects: revisiting deictic reference for achieving spatial common ground. In M.A. Goodrich, A.C. Schultz, and D.J. Bruemmer, editors, *HRI*, pages 297–304. ACM, 2006.

[12] C.A. Calderon and H. Hu. Robot imitation from human body movements. In *In Proceeding AISB05 Third International Symposium on Imitation in Animals and Artifacts*, 2005.

[13] R. Cipolla, P.A. Hadfield, and N.J. Hollinghurst. Uncalibrated stereo vision with pointing for a manmachine interface. In *IAPR Workshop on Machine Vision Applications*, pages 163–166, 1994.

[14] H.H. Clark and C.R. Marshall. Definite reference and mutual knowledge. In Aravind K. Joshi, Bonnie Lynn Webber, and Ivan Sag, editors, *Elements of Discourse Understanding*, pages 10–63. Cambridge University Press, Cambridge, 1981.

[15] H. Cooper, B. Holt, and R. Bowden. Sign language recognition. In T.B. Moeslund, A. Hilton, V. Krger, and L. Sigal, editors, *Visual Analysis of Humans*, pages 539–562. Springer London, 2011.

[16] A. Couture-Beil, R.T. Vaughan, and G. Mori. Selecting and commanding individual robots in a multi-robot system. In *Computer and Robot Vision*, pages 159–166. IEEE, 2010.

[17] T. W. Deacon. *The Symbolic Species: The Co-evolution of Language and the Brain.* W.W. Norton, New York, NY, USA, 1997.

[18] D. Droeschel, J. Stückler, and S. Behnke. Learning to interpret pointing gestures with a time-of-flight camera. In *Proceedings of the 6th international conference on Human-robot interaction*, HRI '11, pages 481–488, New York, NY, USA, 2011. ACM.

[19] J.J. Leonard & H.F. Durrant-Whyte. Simultaneous map building and localization for an autonomous mobile robot. In *IEEE/RSJ International Workshop on Intelligent Robots and Systems*, pages 1442–1447, November 1991.

[20] Blizzard Entertainment. Starcraft. `http://us.blizzard.com/en-us/games/sc/`, Released June 2000, Retrieved April 30 2012.

[21] R A. Gardner, B. Gardner, and T. E. Van Cantford. *Teaching sign language to chimpanzees*, volume 12. State University of New York Press, 1969.

[22] M.A. Goodrich and A.C. Schultz. Human-robot interaction: a survey. *Foundational Trends in Human-Computer Interaction*, 1(3):203–275, January 2007.

[23] T. Hay. The robots are coming to hospitals. The Wall Street Journal Online `http://online.wsj.com/article/SB10001424052702304459804577281350525870934.html`, March 2012, Retrieved April 30 2012.

[24] J. Hollingum. Robots for the dangerous tasks. *Industrial Robot: An International Journal*, 26(3):178–183, 1999.

[25] N. Jojic, B. Brumitt, B. Meyers, S. Harris, and T. Huang. Detection and estimation of pointing gestures in dense disparity maps. In *In The fourth International Conference on Automatic Face- and Gesture-Recognition*, pages 468–475, 2000.

[26] M.J. Barnes J.Y.C. Chen and C. Kenny. Effects of unreliable automation and individual differences on supervisory control of multiple ground robots. In *Proceedings of the 6th international conference on Human-robot interaction*, HRI '11, pages 371–378, New York, NY, USA, 2011. ACM.

[27] B.D. Lucas & T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th international joint conference on Artificial intelligence - Volume 2*, IJCAI'81, pages 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.

[28] T. Kanda, M. Shiomi, Z. Miyashita, H. Ishiguro, and N. Hagita. An affective guide robot in a shopping mall. In *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, HRI '09, pages 173–180, New York, NY, USA, 2009. ACM.

[29] H. Kato and M. Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proceedings of the 2nd International Workshop on Augmented Reality (IWAR 99)*, San Francisco, USA, October 1999.

[30] J. Kato, D. Sakamoto, M. Inami, and T. Igarashi. Multi-touch interface for controlling multiple mobile robots. In *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, CHI EA '09, pages 3443–3448, New York, NY, USA, 2009. ACM.

[31] A. Kendon. Some functions of gaze-direction in social interaction. *Acta Psychologica*, 26(1):22–63, 1967.

[32] M.L. Knapp and J.A. Hall. *Nonverbal communication in human interaction*. R. Holt and W. Holt, 1972.

[33] J. Koenemann and M. Bennewitz. Whole-body imitation of human motions with a nao humanoid. In *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, HRI '12, pages 425–426, New York, NY, USA, 2012. ACM.

[34] K.R. Konda, A. Königs, H. Schulz, and D. Schulz. Real time interaction with mobile robots using hand gestures. In *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, HRI '12, pages 177–178, New York, NY, USA, 2012. ACM.

[35] E. Korac, S. Kutten, and S. Moran. A modular technique for the design of efficient distributed leader finding algorithms. *ACM Trans. Program. Lang. Syst.*, 12(1):84–101, January 1990.

[36] L. Li, X. Yu, and G. Wang J. Li, J. Shi, Y.K. Tan, and H. Li. Vision-based attention estimation and selection for social robot to perform natural interaction in the open world. In *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, HRI '12, pages 183–184, New York, NY, USA, 2012. ACM.

[37] M. Lichtenstern, M. Frassl, B. Perun, and M. Angermann. A prototyping environment for interaction between a human and a robotic multi-agent system. In *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, HRI '12, pages 185–186, New York, NY, USA, 2012. ACM.

[38] S. Magnuson. Robot army in afghanistan surges past 2,000 units. `http://www.nationaldefensemagazine.org/blog/Lists/Posts/Post.aspx?List=7c996cd7-cbb4-4018-baf8-8825eada7aa2&ID=300`, February 2011, Retrieved April 30 2012.

[39] C. Martin, F.F. Steege, and H.M. Gross. Estimation of pointing poses for visually instructing mobile robots under real world conditions. *Robotics and Autonomous Systems*, 58(2):174 – 185, 2010. Selected papers from the 2007 European Conference on Mobile Robots (ECMR 07).

[40] E. McCann, S. McSheehy, and H. Yanco. Multi-user multi-touch multi-robot command and control of multiple simulated robots. In *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, HRI '12, pages 413–414, New York, NY, USA, 2012. ACM.

[41] J. McLurkin, J. Smith, J. Frankel, D. Sotkowitz, D. Blau, and B. Schmidt. Speaking swarmish: Human-robot interface design for large swarms of autonomous mobile robots. In *AAAI Spring Symposium: To Boldly Go Where No Human-Robot Team Has Gone Before*, pages 72–75. AAAI, 2006.

[42] B. Milligan, G. Mori, and R. T. Vaughan. Selecting and commanding groups in a multi-robot vision based system. In *Proceedings of the 6th international conference on Human-robot interaction*, HRI '11, pages 415–416, New York, NY, USA, 2011. ACM.

[43] S. Mitra and T. Acharya. Gesture recognition: A survey. *IEEE TRANSACTIONS ON SYSTEMS, MAN AND CYBERNETICS - PART C*, 37(3):311–324, 2007.

[44] C. Moore and P.J. Dunham. *Joint attention: its origins and role in development*. Routledge, Hillsdale, New Jersey Hove, UK, 1995.

[45] Y. Nagai. Learning to comprehend deictic gestures in robots and human infants. In *Robot and Human Interactive Communication, 2005. ROMAN 2005. IEEE International Workshop on*, pages 217 – 222, aug. 2005.

[46] C.L. Nehaniv and K. Dautenhahn. Imitation in animals and artifacts. chapter The correspondence problem, pages 41–61. MIT Press, Cambridge, MA, USA, 2002.

[47] K. Nickel and R. Stiefelhagen. Pointing gesture recognition based on 3d-tracking of face, hands and head orientation. In *In Workshop on Perceptive User Interfaces*, pages 140–146. ACM Press, 2003.

[48] C.B. Park, M.C. Roh, and S.W. Lee. Real-time 3d pointing gesture recognition in mobile space. In *8th IEEE International Conference on Automatic Face Gesture Recognition, 2008. FG '08.*, pages 1 –6, sept. 2008.

[49] D.W. Payton, R. Estkowski, and M. Howard. Pheromone robotics and the logic of virtual pheromones. In E. Sahin and W.M. Spears, editors, *Swarm Robotics*, volume 3342 of *Lecture Notes in Computer Science*, pages 45–57. Springer, 2004.

[50] D. Vissiere N. Petit P.J. Bristeau, F. Callou. The navigation and control technology inside the ar.drone micro uav. In *18th IFAC World Congress*, pages 1477–1484, Milano, Italy, 2011.

[51] R.A.Bolt. "Put-that-there": Voice and gesture at the graphics interface. *SIGGRAPH Comput. Graph.*, 14(3):262–270, July 1980.

[52] J. Richarz, C. Martin, A. Scheidig, and H.M. Gross. There you go! - estimating pointing gestures in monocular images for mobile robot instruction. In *Robot and Human Interactive Communication, 2006. ROMAN 2006. The 15th IEEE International Symposium on*, pages 546 –551, sept. 2006.

[53] J. Shotton, A.W. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, Richard Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *Computer Vision and Pattern Recognition*, pages 1297–1304. IEEE, 2011.

[54] A. Miklosi & K. Soproni. A comparative analysis of animals' understanding of the human pointing gesture. *Animal Cognition*, 9:81–93, 2006.

[55] C. Staub, A. Knoll, T. Osa, and R. Bauernschmitt. Autonomous high precision positioning of surgical instruments in robot-assisted minimally invasive surgery under visual guidance. *International Conference on Autonomic and Autonomous Systems*, 0:64–69, 2010.

[56] Microsoft Studios. Kinect sports 2. `http://marketplace.xbox.com/en-US/Product/Kinect-Sports-Season-Two/66acd000-77fe-1000-9115-d8024d5309d6`, Released Novemember 2011, Retrieved April 30 2012.

[57] J.F. Sucher, S.R. Todd, S.L. Jones, T. Throckmorton, K.L. Turner, and F.A. Moore. Robotic telepresence: a helpful adjunct that is viewed favorably by critically ill surgical patients. *The American Journal of Surgery*, 202(6):843 – 847, 2011.

[58] E. Tarnow. Body language is of particular importance in large groups. Retrieved April 30 2012.

[59] Parrot USA. Ar.drone parrot - the flying video game. `http://ardrone.parrot.com`, Released June 2010, Retrieved April 30 2012.

[60] M. Van den Bergh, D. Carton, R. De Nijs, N. Mitsou, C. Landsiedel, K. Kuehnlenz, D. Wollherr, L. Van Gool, and M. Buss. Real-time 3d hand gesture interaction with a robot for understanding directions from humans. In *20th IEEE International Symposium on Robot and Human Interactive Communication*, pages 357 –362, August 2011.

[61] P. Viola and M.J. Jones. Robust real-time face detection. *Int. J. Comput. Vision*, 57(2):137–154, May 2004.

[62] K. Wada and T. Shibata. Robot therapy in a care house - its sociopsychological and physiological effects on the residents. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 3966 –3971, May 2006.

[63] A.L. Woodward and J.J. Guajardo. Infants' understanding of the point gesture as an object-directed action. *Cognitive Development*, 83:1–24, 2002.

[64] Y. Yamamoto, I. Yoda, and K. Sakaue. Arm-pointing gesture interface using surrounded stereo cameras system. *International Conference on Pattern Recognition*, 4:965–970, 2004.

[65] K. Zheng, D.F. Glas, T. Kanda, H. Ishiguro, and N. Hagita. How many social robots can one operator control? In A. Billard, P.H. Kahn Jr., J.A. Adams, and J.G. Trafton, editors, *HRI*, pages 379–386. ACM, 2011.