1-1-2008

Advanced human-robot interaction system based on human communication recognition

Adrian Bulzacki
Ryerson University

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

Part of the [Electrical and Computer Engineering Commons](#)

Recommended Citation

Bulzacki, Adrian, "Advanced human-robot interaction system based on human communication recognition" (2008). *Theses and dissertations*. Paper 294.

This Thesis Project 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.

TJ
21.415
B85
2008

ADVANCED HUMAN-ROBOT INTERACTION SYSTEM BASED ON HUMAN COMMUNICATION RECOGNITION

by

Adrian Bulzacki
B. Sc., DeVry University, United States of America, 2005

A project
presented to Ryerson University
in partial fulfillment of the
requirements for the degree of
Master of Engineering
in the Program of
Electrical and Computer Engineering

Toronto, Ontario, Canada, 2008
© Adrian Bulzacki 2008

PROPERTY OF
RYERSON UNIVERSITY LIBRARY

Author's Declaration

I hereby declare that I am the sole author of this thesis or dissertation.

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

I further authorize Ryerson University to reproduce this thesis or dissertation 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.

Borrower's Page

Ryerson University requires the signatures of all persons using or photocopying this thesis. Please sign below, and give address and date.

[illegible]

ABSTRACT

ADVANCED HUMAN-ROBOT INTERACTION SYSTEM BASED ON HUMAN COMMUNICATION RECOGNITION

© Adrian Bulzacki 2008

Master of Engineering
Department of Electrical and Computer Engineering
Ryerson University

This work describes the implementation of various human-robot interaction systems in a functioning mobile robot. This project is the result of integrating a tracking system for human faces and objects, face recognition, gesture recognition, body tracking, stereo-vision, speech synthesis, and voice recognition. The majority of these systems are custom designed for this particular project, these systems and how they were designed are explained in detail throughout this report. A unique vector-based approach is used for gesture recognition. There is minimal focus on the mechanics and electronics of the human-robot interaction system, but rather on the information processing of the robot. Using combinations of many information processing systems will allow robots to interact with human users more naturally, and will provide a natural conduit for future cooperative human-robot efforts. This project lays the groundwork for what will be a large collaborative effort aimed at creating possibly one of the most advanced human interactive robot in the world.

Acknowledgments

I would like to thank my two supervisors, Prof. Ling Guan and Prof. Lian Zhao for all the support and help. Prof. Ling Guan's trust in allowing me to perform work on such exclusive and exciting equipment. He saw potential in me when I even doubted myself.

I would like to thank my parents and grandmother for always looking out for my best interest. They supported me, and were patient during my struggles and mistakes.

Lastly, my friends Alisdiar, Louville, and Paul who understood when I needed to focus on studying, and were always there to cheer me up.

Table of Contents

Chapter 1 Introduction	1
Chapter 2 Original Robot Specifications	3
2.1 Camera System.....	3
2.2 E100	6
2.2.1 Robot Information System.....	7
2.2.2 Map Display System.....	10
2.2.3 Control Panel System	11
2.2.4 Power Management Control System	12
2.2.5 Manual Control System.....	13
2.2.6 Constellation Control System.....	13
2.2.7 Audio Control System	14
2.3 I90.....	15
Chapter 3 Face Recognition.....	19
3.1 Static Face Recognition.....	19
3.2 Dynamic Face Recognition	23
3.3 Emotion Recognition.....	23
Chapter 4 Surveillance Body Tracking.....	25
4.1 Blob Detection and Tracking	26
Chapter 5 Stereo-Triangulation.....	27
Chapter 6 Speech Synthesis and Recognition.....	33
6.1 Speech Synthesis	33
6.1.1 Multilanguage Robotic Communication System.....	34
6.2 Speech Recognition.....	35
Chapter 7 Gesture Recognition.....	37

7.1 Defining Gestures.....	37
7.1.1 Functional Roles of Human Gesture	38
7.1.2 System Responses.....	38
7.2 Gesture Continuum	39
7.3 How to Represent Gestures	41
7.4 Tracker-Based Gesture Recognition (Tracker-Based Vision)	42
7.4.1 Hand Gesture Recognition using Trackers	43
7.4.2 Body Gesture Tracking.....	45
7.5 Vision-Based Gesture Recognition (None-Tracker-Based Vision)	45
7.5.1 Static Gesture Recognition System	46
7.5.2 Dynamic Gesture Recognition System.....	46
7.6 Hidden Markov Model	47
7.6.1 Forward-Backward Algorithm	49
7.6.2 Baum-Welch Algorithm	50
7.6.3 Viterbi Algorithm	51
7.7 Hand and Arm Gesture.....	52
7.8 Dynamic Vector-Based Gesture Recognition	54
7.8.1 Optimized Collins Approach	59
Chapter 8 Recommendations and Conclusion	61
8.1 Recommendations and Future Work.....	61
8.2 Conclusion.....	61
Appendix I – E100 Network & Wireless Connection Configuration	63
Appendix II – E100 Robot PC Server Programs	64
Appendix III – Stereo Vision Code	65
Appendix IV – Speech Synthesis Tree	66

Appendix V – Gesture Recognition Tree..... 67

Appendix VI – Collins Method..... 68

Appendix VII - E100 System Specifications 70

Appendix VIII - E100 Vision System Specifications and Docking 71

References..... 73

List of Figures

Figure 1: Camera Used in Both Robots	4
Figure 2: Screen Shot of Dual Camera View.....	4
Figure 3: Screen Shot of Internet Explorer Camera Control	5
Figure 4: Movement Detection System	5
Figure 5: Dr. Robot E100.....	6
Figure 6: Remote Computer Interface System for E100	7
Figure 7: Screen Shot of E100 Robot Information System	8
Figure 8: Screen Shot Raw Data for E90 Information System.....	9
Figure 9: Screen Shot E90 Temperature Status Display.....	9
Figure 10: Screen Shot E90 Map System	10
Figure 11: Screen Shot E90 Robot Control Tab	11
Figure 12: Screen Shot E90 Human Detection System	12
Figure 13: Screen Shot E90 Power Management Tab	12
Figure 14: Screen Shot E100 Manual Control Tab.....	13
Figure 15: Constellation Positioning System.....	14
Figure 16: Screen Shot E90 Constellation Control Tab.....	14
Figure 17: Screen Shot E90 Audio Control Tab.....	15
Figure 18: Dr. Robot I90.....	16
Figure 19: I90 PC Control System.....	17
Figure 20: High-Speed Face Recognition Flow Chart.....	20
Figure 21: Haar Object Detection Possible Features	21
Figure 22: Example of Haar Features	21
Figure 23: Static Face Recognition.....	22
Figure 24: Static Face Recognition Downside.....	22
Figure 25: Dynamic Face Tracking and Recognition.....	23
Figure 26: W4 Surveillance Body Gesture Tracking.....	25
Figure 27: Flow Chart for Surveillance Body Tracking.....	26
Figure 28: Doctored Demonstration of Surveillance Body Tracking.....	26
Figure 29: Pinhole Camera Model.....	28

Figure 30: Mono-Camera Model	29
Figure 31: Parallel Stereo-Camera Model	30
Figure 32: Non-Parallel Stereo-Camera Model	31
Figure 33: Text-To-Speech Engine.....	34
Figure 34: Speech Recognition Engine.....	36
Figure 35: Example of HMM-based Speech Recognition.....	36
Figure 36: Kendon's Gesture Continuum.....	39
Figure 37: Motion Capture Illustration	44
Figure 38: Face Gesture Recognition using Trackers.....	44
Figure 39: Pattern Recognition System (Ideal for Static Gesture Recognition).....	46
Figure 40: Dynamic Gesture Recognition System.....	47
Figure 41: Architecture of a HMM.....	48
Figure 42: Probabilistic Parameters of a HMM.....	48
Figure 43: Forward Algorithm on HMM.....	49
Figure 44: Backward Algorithm on HMM	50
Figure 45: Baum-Welch Algorithm on HMM.....	51
Figure 46: Viterbi Algorithm on HMM.....	52
Figure 47: Conversion Algorithm Tweaking.....	55
Figure 48: Remove Right Angles (0-100%)	55
Figure 49: Remove Jagged Edges (0-100%)	56
Figure 50: Remove Strangles (0-100%)	56
Figure 51: Fit Curves (0-100%).....	56
Figure 52: Peak Reduction (0-100%)	57
Figure 53: Raster to Vector Conversion	58
Figure 54: Vector File from above Revealing Nodes for Vector Nodal-Network Analysis	58
Figure 55: E100 Vector-Based 3D Vision.....	59
Figure 56: Vector-Based Gesture Recognition Virtual Hand.....	59
Figure 57: Further Optimized Vector File From Above.....	60
Figure 58: Differential Evolutionary Algorithm Scheme	60

Chapter 1

Introduction

By 2020, South Korea aims to have 100% robot market penetration, meaning that a robot will be in every South Korean home [1]. Regardless if South Korea meets this goal, it is irrefutable that robots and robotics will play a critical role in human society in the coming years. As the mechanical and electronic components required for reliable robotics becomes more practical, and as it becomes ever clearer that robots will become a part of our daily lives in the not too distant future, it is important now to focus on how humans will interact with these robots, and in turn how these robots will interact with us. For robots to truly integrate into human life and human society they need to be able to interact naturally with people and their surroundings. Currently, the biggest drawback with present day robotics is that robots are nearly entirely oblivious to the human presence. Humans cannot, as of yet, interact with robots through the same communication methods that we use with each other, we are forced to conform to the machines system by pressing buttons or typing in simple commands. This report will attempt to layout a system of human-to-robot communication that utilizes convention human communication methods. These systems are face tracking and recognition, gesture recognition, body tracking, stereo-vision, speech synthesis, and voice recognition.

Humans use many approaches to communication; a robot needs to be able to understand and respond to all these communication methods simultaneously. The approach to this project is to integrate similar methodologies for various recognition systems, which up till now have been implemented only discretely. An important byproduct of this integrated approach is the potential for increased accuracy through the sharing of variables across the subsystems. The objective of this project is to see if it is possible to integrate all these subsystems together with the ultimate goal of producing more natural human-computer interaction for increased accuracy and effectiveness.

The systems will be implemented on a mobile security patrol robot (E100). A second robot will also be discussed (I90) in this project, this secondary robot, is much smaller and will have a “micro” version of these systems. Chapter 2 will cover original systems in the robots,

while all other sections will cover systems that have been designed and implemented into the robots. Chapter 3 covers the face recognition system implemented, and hints to future work. Chapter 4 provides an example of the body tracking system that has been added to the robot, which can be later used for security patrol. Chapter 5 explains in detail stereo-vision and triangulation. Chapter 6 provides an explanation on how speech synthesis and recognition work. Chapter 7 explain various approaches to gesture recognition and details the vector-based system that was implemented. Chapter 8 ends this report with known issues and conclusions.

Chapter 2

Original Robot Specifications

We are working with two robots the E100 and the I90. Both of the robots have similar platforms, and mainly differ by size and only a few component differences. This is advantageous because the large contrast in size and mobility between the two systems will help us refine future designs, also any unique components in each robot can easily be copied over to the other's systems. Also, the E100 has two cameras, while the I90 has one.

2.1 Camera System

Figure 1 shows the cameras used in both robots, they are network camera (have an IP address) and have easy to control motors for the x-axis and y-axis. The cameras have an optical zoom of ten times, traditional computer cameras do not have an optical zoom, but simply a digital zoom (this camera has digital zoom as well.) This camera is ideal because it is more of a security camera rather than a webcam, making this robot well suited for security patrol. The camera can easily see users at great distances (e.g., the camera can easily zoom into the whites of someone's eyes at a distance of 10 meters.) The camera has a pan of 270 degrees and a tilt of 135 degrees. Figure 2 shows the multi-camera interface of the robot.

Since these cameras are network cameras with IP addresses, one could log into the camera from anywhere in the world to view what the robot is seeing. This would be desirable if security to a building was outsourced or if we only wanted the security guard to monitor the cameras but not the robot control system. Because of the security risk affiliated with having a security camera open to the internet, this robot should be kept off the internet if possible; if it is necessary to connect this robot to the internet, it should only be done by someone very familiar with the robot's systems as well as an advanced knowledge of internet security. Hacking network security cameras is extremely easy and can be achieved with minimal knowledge of computer systems. Figure 3 shows the Internet Explorer camera view.

Figure 4 shows a motion detection system that can be used to detect movement, not just throughout the whole image, but in specific section of the image. This would allow the robot to perform a “guard mode” whereby the robot could stare at a wall of doors and only respond when a specific door is opened (presuming no one walks in front of the door).

Figure 1: Camera Used in Both Robots

Figure 2: Screen Shot of Dual Camera View

Figure 3: Screen Shot of Internet Explorer Camera Control

Figure 4: Movement Detection System

2.2 E100

The E100 robot is a mobile robot that is largely intended for the applications of security patrol, safety, and guiding tours. Figure 5 shows the E100. This robot is capable of self docking and operating without the need for supervision. It has GPS capabilities, and ultrasonic transmitters and receiver for the purpose of positioning itself on a predefined map and 3D space. The E100 has two cameras which are positioned at a height comparable to that of the average human; the neck is height adjustable which allows the robot to be adapted to various tasks and environments. The robot has proximity sensors at the base, neck and head, which prevent the robot from colliding with obstacles of varying heights. For example, if the robot encountered a table, the robot would collide with it if sensors were not positioned on the neck. This combination of three levels of proximity sensors greatly reduces the possibility of the robot bumping into objects. The majority of the robots processing is on a remote computer that transmits through a WIFI network (no wires). Figure 6 shows the remote computer interface system for the E100 which will be explained in the next section. Appendix I shows the E100 network connection configuration chart, while Appendix II shows the E100's PC Server Programs; both of these charts need to be understood before performing any implementation to the robots systems.

Figure 5: Dr. Robot E100

Figure 6: Remote Computer Interface System for E100

2.2.1 Robot Information System

Figure 7 shows the robot information screen which shows most of the critical robot status data(e.g., motor heat level.) Firstly, we can see that the battery charge and charging status is sent in real-time. The robots location in reference to the charging station is displayed. The green circles indicate that the system in question is correctly operating. The robot is also able to distinguish if it is being lifted. Figure 8 shows the raw data for the robot information system. Figure 9 shows the temperature status data for the E100.

Figure 7: Screen Shot of E100 Robot Information System

Figure 8: Screen Shot Raw Data for E90 Information System

Figure 9: Screen Shot E90 Temperature Status Display

2.2.2 Map Display System

Figure 10 is the map screen shot; this shows two of the robots on a pre defined map of the walls and critical locations. This is used to create a patrol area for the robot. Current day robots like the iRobot Romba are capable of creating maps of their surrounds, but currently both the E100 and I90 require pre-specified map data. We are working on adding a navigation system capable of creating map data after the robot “hugs” all the walls and obstacles.

Figure 10: Screen Shot E90-Map System

2.2.3 Control Panel System

Figure 11 shows the robot control panel tab, which contains all the single click robot movements. The red button refers to the current state of the robot. Idle state will turn off motion controllers and cameras. Stop means the robot is currently not moving. Manual mode allows for control through a wireless joystick, or by manually clicking the direction through the keyboard. Wander allows the robot to roam around a room with its collision detection system running (preventing the robot from colliding with objects.) Patrol state makes the robot follow a pre-selected path. GoCHARGE request the robot to return to its charging station (charging station is the home dock for the robot.) A button not included in Figure 11 is the Human Detect button, this system is copied over from the I90; Figure 12 shows the human detection system. The red blocks are the human sensor data, while the blue blocks are the IR range sensor data. The program will by default detect the closest human, and will turn to face and detect the person. This system will combine with the recognition systems discussed later.

Figure 11: Screen Shot E90 Robot Control Tab

Figure 12: Screen Shot E90 Human Detection System

2.2.4 Power Management Control System

Figure 13 shows the E90's power management system; this mainly specifies how much the robot needs to charge the battery and how much power should be left before the robot has to do an emergency charge. This system will later be upgraded to calculate the robots distance from the charge station and to automatically return if it is too far away from the charging station to make it back on its available battery power. The RescuePower is the emergency battery backup system.

Figure 13: Screen Shot E90 Power Management Tab

2.2.5 Manual Control System

The manual control tab shown in Figure 14 is the only way to access the manual motor controls of the robot. The user can click the buttons on the screen, use the keyboard, or use a wireless joystick to control the robot. The right most bar is the accelerator for the motors, this must have a value or the robot will not move, regardless if the forward button has been clicked.

Figure 14: Screen Shot E100 Manual Control Tab

2.2.6 Constellation Control System

The E100 robot sends encoded signals for a pair of system transponders; the corresponding transponders issue a response signal which is acquisitioned by the two receiver components on the robot. Depending on the time interval between the robot's pulse and reception of the signals from the two transponders, the robot derives its spatial location and orientation. Figure 15 shows an illustration of this process.

The Constellation Control System shown in Figure 16 shows how the E100 can coordinate based on ultrasonic transmitters placed on the ceilings of the RML libratory. As long as values are coming in, the system is usually working properly, if not, the robot is out of range or the transmitter is not transmitting.

Figure 15: Constellation Positioning System

ID1:	255	ID2:	255	ID3:	255	ID4:	255	Set	UnSet
Constellation System									
Receive ID:	7	8	1	2	☒				
#1 Receiver(L):	1.72	1.67	-1	2.82	☐				
#2 Receiver(R):	1.68	1.7	-1	2.6	☐				
Compass Dir:	204.6		RecSeq:	0					
Cmd Seq:	4		21029		0				
☒ GPS Scan									

Figure 16: Screen Shot E90 Constellation Control Tab

2.2.7 Audio Control System

Figure 17 shows the audio control tab, which simply allows a user to type in text to be converted into speech (outputted through robot), or input a text file to read aloud. This section has been upgraded and will be discussed later in this paper.

Figure 17: Screen Shot E90 Audio Control Tab

2.3 I90

The I90 robot is the leaner, and more mobile version of the E100, it is more like a scout compared to the E100 tank. The I90 only has one camera, so generally vision based systems will be simpler on this unit, and more confined to static algorithms. The I90 does not have an onboard computer like the E100; the I90 simply has a PDA capable of displaying basic information sent through the network. Generally, the systems on the I90 are very similar, the robot does have collision detection and human detection systems, however, the constellation position and self docking system are not currently installed. Figure 18 shows the I90, while Figure 19 shows the computer control system for the I90.

Figure 18: Dr. Robot I90

Figure 19: I90 PC Control System

Chapter 3

Face Recognition

Face recognition is the process of being able to recognize a human face as a form of identification. This process can also be used for expression and emotion recognition. This is desirable because the robot will need to know who is in its presence (e.g., intruder, engineer, officer.) Using the human detection system in the robot combined with a face recognition system makes the robot a very strong identification system, which can customize responses to the user. The other reasoning for adding face recognition is because it can be built up to perform emotion recognition, which allows for the robot to respond accordingly to human reactions. An example is if the robot provides a response that is completely wrong, the user will look confused and the robot can correct the response. Face recognition assist other forms of gesture recognition and provides a more complete human computer interaction system with the robot. Figure 20 shows the flow chart for a high-speed face recognition system, it is up to the engineer to decide how many characteristics are required based on the necessary accuracy level. The system bases its decision on the cumulative accuracy values of the different checked details of the face.

3.1 Static Face Recognition

The approach we will use for our system is the “Haar-Like Object Detectors.” Using HAAR-like objects is a well established method for face recognition and is reasonably simple to implement [2][3]. Haar-like feature resemble the Haar wavelets, hence the name. In the past, object and face recognition required intensive processing power because every pixel of an image was compared. In the Haar-like feature approach, feature sets are considered as rectangular regions of the image and pixels in the region are summed up. The values calculated are used to categorize the images. Figure 21 shows the Haar-like features, and Figure 22 shows an example of the features on a human face. The face tracking process is simplified by using the OpenCV development tools. The implemented head tracking code is based on OpenCV, the input can be provided from a video file (Adrian_Test1.avi in the example) or in real-time. Our static face recognition system works in real time. Figure 23 shows the system in action, and we can see the face is very easy to detect. The left most bar on the right shows how well the system is detecting

a face, and the right most bar shows how well the face recognition for that particular user is working. As we can see, any type of rotation or change in the facial characteristics lowers the recognition ability. Systems like this can be fooled by images, Figure 24 shows that a student identification card for the user registers as a face, but does not pass the recognition phase. Using a higher quality image would fool this system.

Figure 20: High-Speed Face Recognition Flow Chart

Figure 21: Haar Object Detection Possible Features

Figure 22: Example of Haar Features

Figure 23: Static Face Recognition

Figure 24: Static Face Recognition Downside

3.2 Dynamic Face Recognition

Figure 25 shows a dynamic face tracking and recognition system, this is far more complex than the static system, and resembles other dynamic systems discussed later in this report. The way this system operates is as follows, a virtual model of a face is overlaid and fitted to the user, then as the user moves around the mask continues to follow him in 3D space. This system is a commercial piece of software, but from our research, this is one of the top facial tracking and recognition systems available today.

Figure 25: Dynamic Face Tracking and Recognition

3.3 Emotion Recognition

Facial expression can reveal emotion, control social manners, and reveal brain function. Psychology research has shown that a minimum of six emotions are commonly related to dissimilar facial expressions. Though several other emotions and combinations of emotions have been documented, they are not generally distinguishable. The six core emotions are anger, disgust, fear, happiness, sadness, and surprise. Commonly, research has been conducted on

“mug-shot” style pictures, and this allows the detection of static cues, position, and shape of the facial features. There is of great interest in the study of facial expression recognition, particularly in the field of artificial intelligence. The hope is that a system could better understand the needs and intent of a user through facial expressions. Generally there are two types of systems that can automatically categorize facial expressions: systems that identify prototypic facial expression representing emotions, and systems that recognize facial movements.

Most commonly, facial expression recognition system make assumptions; they assume the face is in the scene and that global locations are known to the system in advance. Most systems require an unvarying background so as to identify the face more easily. This methodology does cause segmentation problems quite commonly in systems that make these assumptions.

Chapter 4

Surveillance Body Tracking

What is often done in this field is to create a sharp silhouette around the human body and apply gesture recognition algorithms based on the silhouette. The problem with this method is that segmentation is quite common in this process. One very interesting paper in on this was done by Haritaoglu, Harwood, and Davis (W4 System.) [3] They chose to use a monochrome surveillance camera and managed to track and monitor people in an outdoor environment. They have ignored all colour cues and simply use a combination of shape analysis and tracking to identify people and their parts. The system runs at 20 Hz and at a resolution of 320X240 using a simply a 200 MHz computer (all unusually low requirements for this otherwise impressive system). Foreground is detected throughout all frames using a combination of background analysis and basic low level processing of the binary image. This system is capable of tracking multiple users (up to 4 at a time) and the system is capable of describing what the person is doing (i.e. walking or standing). This paper describes a very promising path for further research. Figure 26 shows the W4 system as it tracks four gesturers.

Figure 26: W4 Surveillance Body Gesture Tracking

4.1 Blob Detection and Tracking

Blob detection refers to an approach whereby points are detected through brighter or darker qualities versus their surroundings. There are two approaches to blob detections, the first being a differential method based on derivative expressions, and the second being a local min-max of landscape intensity. Figure 27 is a flow chart of a blob detection system that will be used for surveillance body tracking. Figure 28 shows a doctored demonstration of the system. The system can track multiple users, and assign variable to each of them. This system has been implemented into the robot.

Figure 27: Flow Chart for Surveillance Body Tracking

Figure 28: Doctored Demonstration of Surveillance Body Tracking

Chapter 5

Stereo-Triangulation

Stereo-vision is a method for acquiring 3D locations of objects in a scene by merging the data of two or more cameras. One easy method to do this is to have some type of proximity sensor (laser or acoustic) to aid in calculation of the Z-axis, but we will presume that this system can only use cameras. Though this system can be used for all forms of recognition, let us use a simple example to better explain the system. Suppose that the E100 robot systems identified a hand (point $P(X,Y)$) and the robot wanted to relocate to its position. Let us first consider the camera itself, Figure 29 shows an over simplified pinhole camera model. "P" is the location of the hand and "p" is where the hand appears in the photo-receptors. The image plane is where the photo-receptors are situated. Focal distance(f) is the distance between the lens and the photo-receptors (this is unique to the camera.) The camera in the robot has a focal distance of 37.8mm. The optical axis (Z) is the direction that the camera is facing. Figure 30 shows the more mathematical version of the single camera system. Appendix III.I shows the code for a single camera system.

Figure 29: Pinhole Camera Model

Figure 30: Mono-Camera Model

Figure 31 shows two parallel faced cameras performing the same task. “L” is representing the left camera, and “R” represents the right camera, we are defining the system from the perspective of the robot. The optical axis is still “Z” and we now have the variable “B” to consider, which represents the distance between the two cameras. “f” remains to be the focal length. The code for this type of parallel stereo vision is available in Appendix III.II.

Figure 31: Parallel Stereo-Camera Model

The camera in the E100 is not consistently parallel so we have to be able to calculate our problem, from the perspective of a non-parallel camera. Figure 32 presumes the cameras are pointing in different non-parallel directions. We should now note that the Z-axis (optical axis) is the line of sight for the right camera, while the Z_o -axis is the optical axis for the left camera. Both of the cameras are on the XZ plane, but the right camera is rotated by an unknown angle (Φ .) The point where both optical axes meet $(0,0,Z_o)$ will be the fixation point. Appendix III.III has the code for this system. Since our camera cannot rotate on the Z-axis, we have laid out the various camera situations possible. Though currently we are using a two camera system, groundwork for a three or potentially four camera system has already been created. The equations for this system would be largely similar, though we are currently completing an algorithm to fully utilized three cameras as an error compensation system for ensuring the

highest degree of accuracy for vision data (trying to get as close to tracker based system accuracy as possible.)

Figure 32: Non-Parallel Stereo-Camera Model

Chapter 6

Speech Synthesis and Recognition

6.1 Speech Synthesis

Speech synthesis (text-to-speech) allows the robot to communicate verbally for the purpose of giving instructions, providing warnings, and giving system status information without the need for reading data off of a screen. Speech synthesis is simply the artificial production of human speech based on text inputs. In theory this system can be done either through hardware or software, but for the purpose of this robot, we will be doing this through a software system to allow for more universal integration throughout the robots systems. To achieve simple synthesis of speech, one approach is to store recorded speech in a database; this is very comparable to static systems discussed throughout the paper. The alternative is to have a synthesizer integrate a model of the vocal band and other human voice features to generate a completely synthetic voice output; hence “speech synthesis.” Quality of speech synthesizers is rated by how close to a human voice a synthesiser is capable of reaching. Figure 33 shows a text-to-speech engine, the system first takes raw text (including symbols and abbreviations), and converts this into comparable words. That first process is commonly called “text normalization,” or “tokenization.” The engine then converts the words into phonetics, and then separated the text into prosodic units (e.g., sentences and phrases.) Phonetic and prosodic information make up the symbolic linguistic, which is converted into sound.

Figure 33: Text-To-Speech Engine

Because of the ease of availability for free text-to-speech engines, it is not necessary to create a system from scratch. A very advanced text-to-speech engine is included with the last few versions of Windows. Bell Labs has one of the most famous text-to-speech engines, and it comes quite close to resembling the human voice; however it is the most expensive of the systems available, and requires the most processing power. The Nuance Realspeak system allows for varying dialects, accents, and languages; it is also very compact (can run on a PDA), so it can be installed directly on the robots onboard computer instead of streaming. Currently for simplicity, the robot has been upgraded to a Natural Reader voice system based on the free Microsoft engine, though the robots systems has been configured so that it can be easily upgraded to any kind of speech synthesis engine. Appendix IV shows a tree of possible speech synthesis methodologies.

6.1.1 Multilanguage Robotic Communication System

Using AltaVista's Babel Fish Translation or Google's Translator freeware engine to convert the English commands into other languages can allow the robot to speak many languages with ease. Though to maintain quality pronunciations it is best to change the speech synthesis algorithm respectfully. This is generally a simple process but it is very rare to see in other robot designs. For example, if the robot stops someone, and asks the person to provide some sort of

input, if there is a misunderstanding, the person can select a more suitable language preference, thereby making the robot universal to any country.

6.2 Speech Recognition

Speech recognition is desirable for these robots because verbal communication is a critical component of human interaction. These robots will be able to perform any of their functions through verbal commands. When it comes to speech recognition, two general approaches are common, the first is simply a comparator that compares two audio tracks against each other, and this is only good for very small dictionaries. The other is to break up words into phonemes and use a dynamic system based on hidden Markov model. Phonemes are sounds that come together to make up words, the English language has 40 phonemes, that represent the roughly 500,000 words in the dictionary. Figure 34 shows a speech recognition engine. Using phonemes is ideal for a hidden Markov model, and produces very high quality results; current systems have accuracies rates in the high 90% range. Phonemes are extracted and segmented by putting the waveform through a Fourier transform. Figure 35 shows an excellent example of the phonemes hidden Markov model approach on pronouncing the word “Tomato.” This system is also idea for training accents.

Figure 34: Speech Recognition Engine

Figure 35: Example of HMM-based Speech Recognition

Just like speech synthesis engines, speech recognition engines are quite common and reliable (e.g. almost every cell phone made in the last few years has voice dialing), for this reason a third party system will be used, currently the free Microsoft speech recognition engine has been implemented but it will be upgraded to Dragon's Naturally Speaking.

Chapter 7

Gesture Recognition

Research in gesture recognition is based on the goal of creating a system that can accurately recognize and categorize human gestures conveyable as information or for the purpose of system control. The key is to define what information is critical when viewing gestures from the perspective of a computer system. Gesture research and recognition systems date back to the 1960s, however, most fields of gesture recognition are still in their infancy and are able to perform only very basic functions.

From our research, almost no one else is using a gesture recognition system for more than just detecting one or two simple commands.

7.1 Defining Gestures

“Every gesture is the physical expression of a mental concept[5].” However, there are innumerable gestures with myriad meanings, so gesture is both a very complex and abstract concept. “The notion of gesture is to embrace all kinds of instances where an individual engages in movements whose communicative intent is paramount, manifest, and openly acknowledged[6].” Gestures are often a method of communication between people, and are regularly used to convey information, from saying goodbye with a wave, warning someone of a hazard, or “conveying information about space and temporal characteristics[7].” It has shown that gestures do not just amplify spoken language, but also play a key role in the language production process[8]. Further, gestures can also represent unintentional information about emotions and moods. Generally, for the purpose of gesture recognition as examined in this paper, gestures are expressive and meaningful body motions, like the physical movements of arms, the face, fingers, hands, or the whole human body.

7.1.1 Functional Roles of Human Gesture

There are three functional roles to human gestures[9]:

- Semiotic - this is when gestures communicate some form of meaningful information, usually based on shared cultural experiences, e.g., American Sign Language, gestures for guiding airplanes, and waving goodbye.
- Ergotic - this is when a gesture modifies the environment around it through direct manipulation i.e. picking up objects, or shaping a bowl from clay.
- Epistemic - this is when humans learn from the environment around them through tactile experience, e.g., touching an object to discover what material it is constructed of.

It is important to note that all three functional roles can be amplified through the use of devices, e.g., using a handkerchief to wave goodbye, picking up an object with tweezers, or using a measurement tool to discover the nature of an object.

7.1.2 System Responses

Since gesture recognition is the process by which gestures made by the user are made known to the system[10], it is important to realize why gestures are being used to obtain a system response. Gestures are a primary form of communication between humans; however this is something that is difficult to explain to a computer system. If a computer could understand human gestures, this would allow more natural, intuitive and much faster human-computer interaction.

One of the best examples of this is virtual reality navigation. Virtual reality navigation refers to human movement through a computer environment, whether it is entering text, surfing the web, or playing games. Virtual reality is not very real if a user has to type in and cursor all commands and movements through a keyboard. Even a mouse represents an artificial intermediation between human and computer. In the real world, one would simply use hands, feet, voice, etc.

Virtual reality critically needs gesture recognition as a method, not just to provide commands, but also to interact with a virtual environment. Gesture recognition can not only eliminate peripherals, but also can add a lot more detail to the information being sent to the system and allow for much more detailed interaction than can be achieved with a mouse and keyboard. This means that the user would be able react and interact with the environment in a method that is comfortable and natural for them. This would also allow computers to respond to human needs better and allow for greater usability.

7.2 Gesture Continuum

Figure 36 shows Kendon's Gesture Continuum, which was an early step to understanding and describing the nature of human gesture[11]. Kendon defined five types of gesture, as follows:

1. Gesticulation represents the unprompted actions of the hands and arms which accompany speech.
2. Language-like gestures are movements that are part of speech, and replace spoken words or phrases.
3. Pantomimes are gestures that portray actions or objects, and may or may not be associated with speech.
4. Emblems are culturally specific gestures, such as a "thumbs up" for "good," or holding up the index and middle fingers in a "V" for victory (during World War II) or peace (since the 1970s).
5. Sign language is a formalized language system which uses manual communication, body language and lip patterns instead of sound to convey meaning.

As we move to the right in Figure 36 below, the accompaniment of speech with gesture is reduced, impulsiveness decreases, language properties increase, and social guidelines increase.

Figure 36: Kendon's Gesture Continuum

Going into further depth, McNeill explains four sub-types of gesticulation, as follows[12][13]:

1. The iconic type represents figurative gestures that depict features of the action, object or event that is being described. For example, when trying to describe a missing object, often people will say “Have you seen this thing?” followed by then physically indicating the object’s size.
2. The metaphoric type represents familiar metaphor, rather than describing the action, object or event directly. For example, when you let someone in ahead of you, you would move your hand outward while rolling your wrist, or wave them in.
3. The beat is a brief, unformed gesture that is usually related to the stressing of a word. For example, when someone is emphasising a question, they might extend both hands in a quick chop.
4. The deictic is a gesture that involves pointing at a person or object in relevance to space or time. For example, when you need a tissue you could just point to the box just out of reach.

These types of gesticulation account for 90% of human gestures. People tend to gesture even when on the phone, and it is quite common for the blind to gesture when speaking to each other. Human-computer interaction (HCI) requires technology that can simultaneously understand speech and gestures.

Though it is important to study the elements represented on the left side of Figure 1, most gesture recognition systems focus on those on the right side of this figure. This is because experiments can be more controlled when the gestures are less ambiguous. There is less randomness involved when trying to design a system to read sign language than a system trying to understand gesticulation generally. Understanding sign language is an ideal precursor to understanding gesticulation. Gesture recognition requires a system that can accurately define gestures, and sign language provides a clear test of how well gestures are being converted. Once a high accuracy system exists for recognizing this type of gesture data, an AI system can be created to interpret gesticulation well enough to even predict what seems to be too difficult

today. Creating such a system would allow for a very intuitive HCI system that would seamlessly interact with a user.

7.3 How to Represent Gestures

Breaking gestures down into their component parts, or segmenting them, is required to correctly identify natural and uninterrupted gestures. Segmenting gestures automatically is very difficult, however, so that this step is often skipped in gesture recognition systems. As a workaround, a start position in time and/or space is required.

Another common complexity is how to differentiate between a random movement and an intended gesture. Commonly, gesture recognition systems share similar and common configurations. Either a gesture can be dynamic mode (defined by movement) or static mode (user is in a special pose). McNeill also lays out three stages of dynamic gestures, the pre-stroke, stroke, and post-stroke. When it comes to sign language recognition, it is important to apply both static and dynamic elements to the system for accurate results. It is common that when creating gestures continuously, gestures are affected by the last gesture. Hummels [14] lays out four characteristics for a gesture that help identify its significance, as follows:

1. Spatial information is the location of the gesture in question.
2. Pathic information is based on the path of the gesture.
3. Symbolic information represents the sign or symbol made by the gesture.
4. Affective information represents the emotional value of the gesture.

In order to capture the gesture fully, we have to be aware of the gesturer's position, movement and configuration. Usually this is done with a data suit, sensors, or special gloves being worn by the gesturer, though using computer vision systems seems to be a good indirect approach. Each method has its strength and weaknesses which will be discussed later in this paper.

Gestures tend to vary from person to person, so it is critical to capture the invariant gesture properties to keep a system universal. A database of invariant gestures is necessary that

can be applied to varying gesturers. Just as in speech recognition systems, the higher the accuracy required, the lower that universal compatibility, since the system becomes more and more gesturer-specific. Ideally, this system would be trained in advance through a learning process, in order to improve both accuracy and usability.

7.4 Tracker-Based Gesture Recognition (Tracker-Based Vision)

Trackers are any type of device that would need to be mounted or attached to a user to monitor gestures. This technique is most commonly used in the film industry because it provides a very accurate description of the movements of the body. There are many types of trackers available. Most commonly optic or ultrasonic trackers are used in gesture recognition, though optical sensors are preferred because of their higher degree of accuracy and much smaller size, enabling detailed face tracking (Figure 37). When it comes to ultrasonic trackers, a gesturer would wear ultrasonic transmitters on critical joints of the area in question, then a set of stationary ultrasonic receivers would position the transmitters in three dimensions, and real-time gesture data would be obtained. The more receivers there are, the higher the accuracy. It is also possible to use magnetic sensors, though this will not be covered in this paper.

Optical tracking systems position markers in three dimensions for one or more cameras, which work together and provide overlapping data (Figure 38). Tracking many markers requires multiple cameras and would require a fixed capture area because the cameras are stationary. Markers allow for data to have three degrees of freedom from each marker. To capture rotational information, three or more markers are required. There are 4 types of optical markers available, as follows:

1. Passive Marker – a marker which is coated with a retro-reflective material that allows light near the camera lens to be reflected back. Adjusting the camera's sensitivity allows most cameras to narrow the viewable footage to just bright markers and ignore skin and clothing. Centroids of each marker are estimated in a two-dimensional image. Greyscale values of each pixel can provide sub-pixel precision.
2. Active Marker – a marker made from LEDs. Triangulation of markers is done by illuminating one LED at a time very quickly. Multiple LEDs can be illuminated

simultaneously but complex software is required to identify each marker to its relative position. Unlike passive markers, the LEDs do not reflect light but are powered so they generate the light necessary. This type of system is better suited for larger capture areas, because the LEDs can generate light more visible from greater distances than passive markers. However identifying each marker requires additional processing.

3. Time Modulated Active Marker – Uses a quicker method of strobing one marker, or when considering multiple markers, the markers are tracked over time, and by modulating the pulse width the marker ID can be identified. LEDs are connected to onboard processing and wirelessly synchronized. This system can be applied outdoors, unlike passive markers
4. Semi-Passive Imperceptible Marker - Instead of using retro-reflective or LED markers, this system uses photosensitive marker tags to decode the optical signals. By using photo-sensors, the tags can calculate illumination, location, orientation, and reflectance

Though each type of tracker has a different approach, the idea is consistently the same, if you can model the gesture in 3D, you can identify and categorize it.

7.4.1 Hand Gesture Recognition using Trackers

The hands are a critical part of the human body to map since they are very expressive and are used for many important gestures. The human hand is capable of 29 degrees of movement, which make it a critical but difficult part of the body to digitally capture. To use trackers, a data glove (similar to an exoskeleton) is required to capture all the movements of the fingers and wrist. There are many benefits to using this approach: it is easy to use, no line-of-sight is required, it provides a high sampling rate, and generates a lot of detail. The drawbacks are that an accurate system is expensive, the glove is invasive and awkward to use, calibration can be difficult, and if the glove is tethered it can reduce range. Despite these drawbacks many experiments have used data gloves to create complex gesture based interfaces. Table 1 lays out various data glove experiments.

Ref.	Author	Task	Method
[15]	Latoschik	Pointing for multimedia apps.	Multi-agent architecture
[16]	Väänänen	Teaching static gestures	Neural network
[17]	Böhm	Teaching dynamic gestures	Kohonen feature map to reduce data
[18]	Baudel	Gesture input to presentation	Interaction model and notation for gestures
[19]	Fels	Hand gestures to speech	Adaptive neural network
[20]	Takahashi	Japanese Kana manual alphabet	Real-time 3D model
[21]	Lee	Recognize new gestures	Hidden Markov models

Table 1: Research on Hand Gestures using Tracking

There is a large following of Nintendo Power Glove enthusiasts who have hacked the glove to be used as a gesture recognition system using the REND386 3D rendering package[22]. Also, the University of Washington created a C++ library called GloveGRASP, which allows developers to submit gesture recognition functions to the SGI systems[23].

Figure 37: Motion Capture Illustration

Figure 38: Face Gesture Recognition using Trackers

7.4.2 Body Gesture Tracking

Generally speaking, body gesture recognition is a simple process when using trackers. It has been successfully done for decades in film-making. Nonetheless, it is important to explore how gesture data can be properly associated with a gesture. Bobick [24] has laid out a classification system to better understand motions, movement, activity, and action. The first level is movement; it represents any minuscule characteristic of motion. Activity is a chain of motions, or static configurations. Action represents more complex, higher-level detail of what is occurring in a situation or background. The majority of research does not go past the first two levels. Table 2 briefly outlines research and experiments conducted on body gesture recognition with the aid of tracking

Ref.	Author	Task	Method
[25]	Boyd	Evaluation of rehabilitation	Shape-of-motion gait analysis
[26]	Davis	Capture history of motion	Temporal templates

Table 2: Research on Body Gestures using Tracking

Though optical, ultrasonic and magnet tracking systems are burdensome, which conflicts with the need to create a more natural interface, it is likely that advances will be made in sensor technology that will enable innovation in such devices, e.g., gloves, rings, and watches. Also, body tracking suits may be replaced by sensors in belts, glasses, shoes, shirts, and even pants. [27][28][29]

7.5 Vision-Based Gesture Recognition (None-Tracker-Based Vision)

Though tracker-based systems can render virtual 3D data on a gesture very accurately, they are not realistic for general users, and the usability of such a system is very poor. For the purpose of this report “vision-based” will refer to any camera-based gesture recognition system that does not use trackers. One of the most important hurdles in solving the gesture recognition problem is how to get tracker quality data on a gesture without the need for a physical tracker. The only solution to this is to explore vision-based systems and find a method using visual processing to get as close as possible to tracker accuracy. Let us first explore the two types of vision-based gesture recognition, static gestures, and dynamic gestures.

7.5.1 Static Gesture Recognition System

Static gesture recognition is simply when a particular hand configuration and pose is represented by a single image. When it comes to static gestures, the recognition system can be accomplished through a process illustrated in Figure 39, by way of a neural network, template matching or traditional pattern recognition methods. This system is generally ideal if the system being created is only intended to identify a small handful of different gestures. Static system can usually only be used for the simplest of tasks and will tend to be very restrictive on user.

Figure 39: Pattern Recognition System (Ideal for Static Gesture Recognition)

7.5.2 Dynamic Gesture Recognition System

Dynamic gesture recognition is a moving gesture represented by a sequence of images. Figure 40 shows a dynamic system. As we can see, stereo gesture data is important to the system, unlike static systems which can rely simply on a 2D image; dynamic systems are indented to capture as much data on the gesture as possible. Similar to a static system, background data and image effects are done to the gesture to help separate the important data from the interference. Dynamic systems require the use of tracking to constantly track the movement of the body. When designing dynamic gesture recognition systems, one must consider temporal events through either Hidden Markov models (HMMs), Bayesian networks, or dynamic time warping. We can see that the system needs to know if this is first frame, this is relevant when we consider how HMM requires past knowledge to help recognise gestures.

Figure 40: Dynamic Gesture Recognition System

7.6 Hidden Markov Model

Before we can understand a HMM we must first understand a traditional Markov Process. A Markov Process is a type of random process where one is determining future probabilities based on the information of a current and recent value (state). However with HMM, the current and future state are unknown, but need to be somehow predicted, therefore, HMM uses the only values it might know, old values, and indirect variables. HMM is a learning algorithm. HMM is most commonly used in speech recognition algorithms, because the system needs to conform to individual users, it is easy to see how speech recognition approaches can be adapted to vision-based gesture recognition; HMM has also been used in handwriting recognition. Figure 41 shows the architecture of a HMM, traditionally the “ x_1 ”, “ x_2 ”, “ y_1 ”, and “ y_2 ” (y’s are indirect variables) variables are not included in textbook or publication examples but it is important to understand that any states from the past can be useful when trying to predict the state of x or $x(t+1)$; this of particular importance to know especially if an initial training phase was necessary in a recognition algorithm. Figure 42 shows the probabilistic parameters of a HMM, “ x ”

represents the states, “y” represents the potential observations (indirect variables), “a” is the state transition probabilities, and “b” is the output probabilities.

Figure 41: Architecture of a HMM

Figure 42: Probabilistic Parameters of a HMM

To solve HMM the following three methods are traditionally used, Baum-Welch, Forward-Backward, and Viterbi algorithm

7.6.1 Forward-Backward Algorithm

Forward-Backward algorithm is used to solve HMM when the parameters of the model are known, but it is necessary to calculate the probability of a particular output sequence, and the probabilities of the hidden state values, based on that particular output sequence. Forward-backward algorithm can be done in brute force approach to find the solution of this type of problem by generating all the possible sequences of the observed events, and hidden states with their probabilities using two transition matrices. The combined probability of the two sequences, with a given model, is derived by multiplying the related probabilities. Figure 43 shows the forward method while Figure 44 shows the backward.

Figure 43: Forward Algorithm on HMM

Figure 44: Backward Algorithm on HMM

7.6.2 Baum-Welch Algorithm

Baum-Welch algorithm is a generalized expectation-maximization algorithm which calculates the maximum likelihood estimates for HMM, when only provided with emissions (transition states and output probabilities) as training data. Baum-Welch algorithm is used to determine the parameters of the HMM that has a known dataset of sequences. Another way of perceiving this is by finding the most likely set of state transitions and output probabilities if we knew the output sequence or a simply a set of such sequences. Baum-Welch has two steps, first calculating the forward and backward probability of each HMM state. Secondly, based on the first step, derive the frequency of the transition-emission pair values and dividing that by the probability of the entire string (calculating the expected count of the particular transition-emission pair.) For every time a specific transition is identified, the value of the quotient of the

transition divided by the probability of the whole string increases, this value becomes the new string. Baum-Welch Algorithm is more desirable than Backward Forward because it allows for the calculation of one additional state. **Error! Reference source not found.** shows the algorithm.

Figure 45: Baum-Welch Algorithm on HMM

7.6.3 Viterbi Algorithm

Viterbi algorithm is used when trying to find the most likely sequence of hidden states (viterbi path) that may have generated a known output sequence, if the parameters of the model were known. Viterbi algorithm makes a few conditions; firstly both the observed events and hidden events are in sequence. Secondly, those two sequences are aligned, and indirect variables need to correspond to one hidden event. These conditions are all satisfied in a first-order HMM. Figure 46 shows the algorithm.

Figure 46: Viterbi Algorithm on HMM

7.7 Hand and Arm Gesture

Vision-based analysis of hand gestures is the most acceptable and natural method for obtaining gestures data, and is a better approach than glove-based analysis. Unfortunately, due to limitations in current computer vision systems, vision-based hand gestures are very difficult to execute with a high accuracy level. Several methods have been attempted, though commonly a video camera is placed in front of a user and gesture data is attempted to be extracted. There are several difficulties with this approach: segmentation of the hands and the environment, complexities analysing hand motion, tracking hands in the environment, and identifying hand posture. The biggest limitation is that hand gestures are complex and require 3D data that is very complicated to extract from a 2D image. One possibility is to approach the hand gesture problem by creating a 3D virtual model of the human hand and trying to keep that synchronized with a high frequency multi-camera system. Dounnton and Drouet[38] created a system that tracks

limbs through an “articulated generalized cylindrical human model,” which shows promising results. Though they only used a single monochrome camera, matching was done through a perspective projection of the cylindrical model. The output results of this system were kinematic specifications of the joint angles that were capable of being used for sign language analysis. A generalized cylinder was also used by Etoh[39], using stereo cameras and an algorithm that partitions objects through a hierarchical set of approximated cylinders. Both of these were the basis for DigitEyes[48], which was stereo-optic, and was a complete hand gesture recognition system using Gauss-Newton minimization of feature errors. Lee[46] used a 3D hand skeleton model with 27 degrees of freedom and used five restrictions on the human hand kinematics to narrow search space. Creating constraints is an interesting approach because it helps localize and narrow system results. Even though it really limits the capabilities of the system, most commercial products use constraints to keep systems simple. Kuch [45] allowed for 26 degrees of freedom and six constraints; the system was adapted to specific users and would gather gestures out of a long sequence of single-camera images. It is also quite common to see marked glove gesture recognition systems[37][47][52]. Though these still use a vision system, this approach is too much like a low cost solution to using trackers.

Table 3 is an updated table of Hang and Pavlovic’s[53] vision based hand gesture models, though an excellent table is available at [54].

Ref.	Author	Task	Method	Camera	Speed(fps)
[30]	Ahmad	Hand Tracking	Fingertip detection model	Mono	10-30
[31]	Ahmad	Posture Recognition	Gaussian NN	Mono	N/A
[32]	Bulzacki	Gesture Recognition	Vector nodal network	Mono	40
[33]	Cipolla	Gesture Recognition	Marked fingertips	Mono	25
[34]	Cho	Posture Recognition	Local shape property learning	Mono	N/A
[35]	Darrell	Gesture Recognition	Set-of-views model	Mono	10
[36]	Davis	Hand Tracking	Fingertip detection model	Mono	N/A
[37]	Davis	Gesture Recognition	Marked fingertips	Mono	4
[38]	Downton	Limb Tracking	3D cylindrical limb model	Mono	N/A
[39]	Etoh	Hand Tracking	3D cylindrical hand model	Stereo	N/A
[40]	Fukumoto	Pointing	Finger detection and virtual projection origin	Stereo	Real-Time
[41]	Freeman	Gesture Recognition	Orientation histograms	Mono	Real-Time
[42]	Kervrann	Hand Tracking	Stochastic deformable model	Mono	N/A
[43]	Kjeldsen	Gesture Recognition	ALVINN	Mono	N/A
[44]	Krueger	Object Manipulation	Silhouette	Mono	30
[45]	Kuch	Gesture Recognition	3D NURBS hand model	Mono	3-30
[46]	Lee	Gesture Recognition	3D hand skeleton model	Mono	0.6-1.33
[47]	Maggioni	Gesture Recognition	Marked glove	Mono	25
[48]	Rehg	Gesture Recognition	3D cylindrical hand model	Stereo	10
[49]	Schlenzig	Gesture Recognition	Zernike moments	Mono	0.5
[50]	Segen	Gesture Recognition	Silhouette edges	N/A	Real-Time
[51]	Starner	Gesture Recognition	HMM geometric parameters	Mono	5
[52]	Torige	Pointing	Marked glove	Stereo	30

Table 3: Index of Vision Based Hand Gesture Models

7.8 Dynamic Vector-Based Gesture Recognition

An algorithm has been developing algorithm (based on Collins Algorithm [55], Appendix VI) to covert regular pixel based video into vector video (made of geometric shapes and points) as a platform for a gesture recognition system. Figure 47 shows a screenshot of the software used for the conversion example. Simply, the system is using a combination of MATLAB features, and a program coded in Visual Basic to allow for the use of Macromedia components and algorithms. The algorithm used allows for a human factor customize the results. The human factor is necessary because all conversions are not the same and may require certain adjustments to be made for optimal results. Figure 48 illustrates the human choice for removing of right angles; the image on the left represents 0% while the image on the right represents 100%. Figure 49 illustrates the human choice of removing jagged edges; the image on the left represents 0% while the image on the right represents 100%. Figure 50 illustrates the removal of stragglers; the image on the left represents 0% while the image on the right represents 100%. Figure 51 illustrates the fitting of curves (0-100%); higher thresholds will cause a looser curve fitting.

Figure 52 illustrates the peak reduction (0-100%); peak reduction threshold determines how many pixels are allowed for the peak of a fitted curve above the line used as the basis for curve fitting. This approach needs be optimized specifically for gesture recognition.

Figure 47: Conversion Algorithm Tweaking

Figure 48: Remove Right Angles (0-100%)

Figure 49: Remove Jagged Edges (0-100%)

Figure 50: Remove Strangles (0-100%)

Figure 51: Fit Curves (0-100%)

Figure 52: Peak Reduction (0-100%)

The goal of this system is to deploy either one to three cameras and convert sign language gestures into a fully functioning real time, high accuracy, gesture-recognition system. This system, once complete, could potentially revolutionize the gesture recognition field and open the door not only for sign language recognition, but also for many other gesture based systems, including a complete replacement for current day input peripherals. Web browsing and all general computer operation and navigation could be controlled with a much more human intuitive system where images, for instance, could be sorted and viewed simply by virtually dragging images by a user's hands and not requiring any additional wearable equipment. Figure 53 shows an *.avi file of my hands, and after applying a conversion algorithm, a vector format video is created that has all the required information for gesture recognition. Figure 54 shows the nodes that represent outlines of all the vector shapes used to make up this video. What is important to note is that these nodes contain the necessary outline data of the hands and can not only track [56] but when using special equations based on a vector nodal-network, it can be used to interpret the formation of the fingers and hands. Figure 57 is an optimized version of the vector conversion algorithm which only outlines the hands with vector nodes and lines. The system is currently being implemented on the E100 vision system, Figure 55 shows how the stereo camera system uses multiple vector images of a gesture to gather 3D gesture data. The easiest way to visualize this system is to imagine the vector nodes creating a galaxy (stars represent the nodes), these nodes would create a virtual glove that a virtual hand could fit inside (Figure 56.) As the nodes move in 3D space, they will force the movement of the virtual hand. A HDD will be used to optimize the real-time computation of such a system.

Figure 53: Raster to Vector Conversion

Figure 54: Vector File from above Revealing Nodes for Vector Nodal-Network Analysis

Figure 55: E100 Vector-Based 3D Vision

Figure 56: Vector-Based Gesture Recognition Virtual Hand

7.8.1 Optimized Collins Approach

Figure 57 shows a partially optimized algorithm based on Collins method, what is important to notice is how the video simply shows just the outline of a human hand. This new algorithm is loosely based on differential evolution which is a type of genetic algorithm. The differential evolution is a population-based optimization algorithm. The problem formulation is

for an object function, and the algorithm optimizes float (real) parameters for real functions. Figure 58 shows a basic differential evolution algorithm scheme.

Figure 57: Further Optimized Vector File From Above

Figure 58: Differential Evolutionary Algorithm Scheme

The new vector outlining algorithm starts by being provided proximity (explained in stereo-camera section) and based on that value, a cut-off is created which helps the differential evolutionary based Collins method to decide between only two vector colours instead of the full colour library. The decision has a catch however, the algorithm makes its decision presuming only two shapes exist (background and hand), and one of the shapes will be inside the other. This is critical because it forces the algorithm to isolate the two possible parts of the image. Once that process is done the foreground and background each have separate colours, however, this algorithm has been further optimized for convenience. Instead of the two sections being separated by a difference in colour, another algorithm comes in and just borders off the two sections (explaining why the hand is just an outline.) This is just a convenience for future work. Appendix V is a tree showing the various approaches to gesture recognition.

Chapter 8

Recommendations and Conclusion

8.1 Recommendations and Future Work

A new navigation system capable of creating map data after the robot “hugs” all the walls and obstacles is still being developed. A system like this would allow the robot to conform to its surrounding and require minimal configuration time between locations. Because of all the simultaneous systems running on the robot, it is best not only to add a powerful onboard computer(will reduce battery power) to the E100, but also to program all the various systems in parallel to fully utilize current and upcoming multi-core technology. It is recommended to supply each camera on the E100 with its own independent wifi transmitter because currently so many processing tasks are being performed remotely. Also, it would be useful to compare the multi-camera system discussed versus a specialty stereo-camera. The microphone should be relocated to the head of the E100, currently there is a lot of interference because of its proximity to the speaker. The mobility of the E100 is quite slow and would benefit from weight reduction and hardware simplification. All systems currently implement are only the basic versions of what they potentially will become, no current system on these robots is intended to be a uniquely revolutionary, but the revolution is, how these systems will act together. Appendix VII is a rundown of sensor systems on the E100, while Appendix VIII shows the vision subsystems and the docking components.

8.2 Conclusion

This project has shown how this robot can navigate, maintain its own power, recognize human faces, identify user movements, recognize human gestures, perceive three dimensions, understand speech, and speak. It has also shown how to integrate similar methodologies for various recognition systems, which up till now have been implemented only discretely. It has been demonstrated that it is possible to integrate all these subsystems successfully. All these subsystems work together to create a more natural interaction between humans and robots than an individual system only. As more dynamic systems are incorporated into the robot, the robot will begin to become more responsive to human actions and behaviours, which will allow robots

to anticipate human actions and respond faster and more accurately. This report is part of a larger project to incorporate complex algorithms for practical applications into the robot platform described in this report.

Appendix I – E100 Network & Wireless Connection Configuration

Appendix II – E100 Robot PC Server Programs

Appendix III – Stereo Vision Code

III.I Single Camera:

$$x_camR = focal_length * (X_actual / Z_actual)$$

$$y_camR = focal_length * (Y_actual / Z_actual)$$

III.II Parallel Cameras:

$$Z_actual = (b * focal_length) / (x_camL - x_camR)$$

$$X_actual = x_camL * (Z_actual / focal_length)$$

$$Y_actual = y_camL * (Z_actual / focal_length)$$

III.III Non-Parallel Cameras:

$$Z_o = b / \tan(\phi)$$

$$Z_actual = (b * focal_length) / (x_camL - x_camR + focal_length * b / Z_o)$$

$$X_actual = x_camL * Z_actual / focal_length$$

$$Y_actual = y_camL * Z_actual / focal_length$$

---With Rotation on X-axis---

$$Z_actual = (b * focal_length) / (x_1 - x_2)$$

$$X_actual = x_camL * Z_actual / focal_length$$

$$Y_actual = y_camL * Z_actual / focal_length + \tan(\phi) * Z$$

Appendix IV – Speech Synthesis Tree

Appendix V – Gesture Recognition Tree

Appendix VI – Collins Method

F:1 D:100

F:2 D:100

F:3 D:100

F:4 D:100

F:5 D:100

F:6 D:100

F:7 D:100

F:8 D:100

F:9 D:100

F:10 D:100

F:11 D:100

F:12 D:100

F:13 D:100

F:14 D:100

F:15 D:100

F:16 D:100

F:17 D:100

F:18 D:100

F:19 D:100

F:20 D:100

F:21 D:100

F:22 D:100

F:23 D:100

F:24 D:100

F:25 D:100

F:26 D:100

F:27 D:100

F:28 D:100

F:29 D:100

F:30 D:100

F:31 D:100

F:32 D:100

F:33 D:100

F:34 D:100

F:35 D:100

F:36 D:100

F:37 D:100

F:38 D:100

F:39 D:100

F:40 D:100

F:41 D:100

F:42 D:200

F:43 D:200

Appendix VII - E100 System Specifications

Appendix VIII - E100 Vision System Specifications and Docking

References

- [1] S. Lovgren, "A Robot in Every Home by 2020, South Korea Says," *National Geographic News*, September 6, 2006.
- [2] T. Mita, T. Kaneko, and O. Hori, "Joint Haar-like Features for Face Detection," *Tenth IEEE International Conference on Computer Vision*, vol. 2, pp. 1619-1626, 2005.
- [3] K. Hong, J. Min, W. Lee and J. Kim, "Real-Time Face Detection and Recognition System Using Haar-Like Feature/HMM in Ubiquitous Network Environments," *ICCSA 2005*, pp. 1154-1161, May 2005.
- [4] I. Haritaoglu, D. Harwood, and L. Davis, "W4: Who? When? Where? What? A Real Time System for Detecting and Tracking People," *Proc. Third International Conference on Automatic Face and Gesture Recognition*, Nara, Japan, Apr. 1998.
- [5] S. Thieffry, "Hand Gestures," *In the Hand*, pp. 488-492, Philadelphia, PA: Sanders, 1981.
- [6] J. Nespoulous, P. Perron, and A. Lecours, "The Biological Foundations of Gestures: Motor and Semiotic Aspects," *Lawrence Erlbaum Associates*, Hillsdale, 1986.
- [7] A. Kendon, "Conducting Interaction: Patterns of behavior in focused encounters," *Cambridge University Press*, Cambridge, 1990.
- [8] D. McNeill and E. Levy, "Conceptual Representations in Language Activity and Gesture," *John Wiley and Sons Ltd*, pages 271-295, 1982.
- [9] C. Cadoz, "Les réalités virtuelles", *Dominos*, Flammarion, 1994.
- [10] G. Barattoff and D. Searles, "Gesture Recognition," <http://www.hitl.washington.edu/scivw/EVE/I.D.2.b.GestureRecognition.html>.
- [11] A. Kendon, "Some relationships between body motion and speech," in A. W. Siegman and B. Pope (eds.), *Studies in Dyadic Communication*, New York, Pergamon Press, 1972.
- [12] D. McNeill, "Hand and Mind: What Gestures Reveal about Thought," *Chicago: University of Chicago Press*, 1992.
- [13] D. McNeill, "So you think gestures are nonverbal?" *Psychological Review*, Vol 92(3) 350-371, 1985.
- [14] C. Hummels and P. Stappers, "Meaningful gestures for human computer interaction: beyond hand gestures," *Proc. Third International Conference on Automatic Face and Gesture Recognition*, Nara, Japan, Apr. 1998.

- [15] M. Latoschik and I. Wachsmuth, "Exploiting Distant Pointing Gestures for Object Selection in a Virtual Environment," *Gesture and Sign Language in Human-Computer Interaction, Proc. International Gesture Workshop*, Sept. 1997.
- [16] K. Väänänen and K. Böhm, "Gesture Driven Interaction as a Human Factor in Virtual Environments – an Approach with Neural Networks," *Proc. Virtual Reality Systems, British Computer Society*, 1992.
- [17] K. Böhm, W. Broll, and M. Solokewicz, "Dynamic Gesture Recognition Using Neural Networks; a Fundament for Advanced Interaction Construction," *Stereoscopic Displays and Virtual Reality Systems, SPIE Conference on Electronic Imaging Science and Technology*, Vol. 2177, 1994.
- [18] T. Baudel and M. Beaudouin-Lafon, "CHARADE: Remote Control of Objects Using Free-hand Gestures," *Communications of the ACM*, vol. 36, no. 7, p. 28-35, 1993.
- [19] S. Fels and G. Hinton, "Glove-TalkII: An Adaptive Gesture-to-Formant Interface," *CHI'95*, Denver, CO, May 1995.
- [20] T. Takahashi and F. Kishino, "Gesture Coding Based in Experiments with a Hand Gesture Interface Device," *SIGCHI Bulletin*, Vol. 23(2) pp. 67-73, April 1991.
- [21] C. Lee, Y. Xu, "Online, Interactive Learning of Gestures for Human/Robot Interfaces." *IEEE International Conference on Robotics and Automation*, vol. 4, pp 2982-2987, 1996.
- [22] Power Glove Page, 2004. <http://www.angelfire.com/ok2/stepinto/PowerGlovePage.html>
- [23] GloveGRASP Gesture Recognition, 2001.
<http://www.hitl.washington.edu/projects/multimodal/GestureGRASP.html>
- [24] A. Bobick, "Movement, Activity, and Action: the Role of Knowledge in the Perception of Motion," *Royal Society Workshop on Knowledge-based Vision in Man and Machine*, Feb., 1997.
- [25] J. Boyd and J. Little, "Shape of Motion and the Perception of Human Gaits," *IEEE Workshop on Empirical Evaluation Methods in Computer Vision*, June 1998.
- [26] J. Davis and A. Bobick, "The Representation and Recognition of Human Movement Using Temporal Trajectories," *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Puerto Rico, June 1997.
- [27] J. Smith, T. White, C. Dodge, D. Allport, J. Paradiso, and N. Gershenfeld, "Electric Field Sensing for Graphical Interfaces," *IEEE Computer Graphics and Applications*, Vol. 18, No. 3, pp. 54-60, June 1998.

- [28] J. Paradiso, E. Hu, and K. Hsiao, "The CyberShoe: A Wireless Multisensor Interface for a Dancer's Feet," *Proc. International Dance and Technology 99*, Feb., pp 26-28, 1999.
- [29] R. Fletcher, J. Levitan, J. Rosenerg, and N. Gershenfeld, "Application of Smart Materials to Wireless ID Tags and Remote Sensors," *Materials Research Society Fall Meeting*, 1996.
- [30] S. Ahmad, "A Usable Real-Time 3D Handtracker," *Proc. of the 28th Asilomar Conference on Signals, Systems, and Computer*, 1994.
- [31] T. E. Starner and A. Pentland, "Visual Recognition of American Sign Language Using Hidden Markov Models," *International Workshop on Automatic Face- and Gesture-Recognition IWAFIGR95*, June 1995.
- [32] A. Bulzacki, L. Guan, and L. Zhao, "Introduction to Gesture Recognition through Vectorization," *Asia-Pacific Workshop on Visual Information Processing*, pp. 83-88, Dec. 2007
- [33] R. Cipolla, Y. Okamoto, and Y. Kuno, "Robust Structure from Motion Using Motion Parallax," *International Conference on Computer Vision*, pp. 374-382, 1993.
- [34] K. Cho and S. M. Dunn, "Learning Shapeclasses," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, pp. 882-888, Sept. 1994.
- [35] T. Darrell and A. Pentland, "Space-time Gestures," *Computer Vision and Pattern Recognition Conference*, 1993.
- [36] J. Davis and M. Shah, "Gesture Recognition," *Tech. Rep. CS-TR-93-11*, 1993.
- [37] J. Davis and M. Shah, "Determining 3D Handmotion," *28th Asilomar Conference on Signals, Systems, and Computer*, 1994.
- [38] A. C. Downton and H. Drouet, "Image Analysis for Model-based Sign Language Coding," *Progress in Image Analysis and Processing II: Proc. of the 6th International Conference on Image Analysis and Processing*, pp. 637-644, 1991.
- [39] M. Etoh, A. Tomono, and F. Kishino, "Stereo-based Description by Generalized Cylinder Complexes from Occluding Contours," *Systems and Computers in Japan*, vol. 22, no. 12, pp. 79-89, 1991.
- [40] M. Fukumoto, Y. Suenaga, and K. Mase, "Finger-pointer: Pointing interface by Image Processing," *Computers and Graphics*, vol. 18, no. 5, pp. 633-642, 1994.
- [41] W. T. Freeman and M. Roth, "Orientation Histograms for Hand Gesture Recognition," *International Workshop on Automatic Face and Gesture-Recognition*, June 1995.

- [42] C. Kervrann and F. Heitz, "Learning Structure and Deformation Modes of Nonrigid Objects in Long Image Sequences," *International Workshop on Automatic Face and Gesture-Recognition*, June 1995.
- [43] R. Kjeldsen, "Visual Hand Gesture Interpretation," *IEEE Computer Society Workshop on Non-Rigid and Articulate Motion*, Nov. 1994.
- [44] M. W. Krueger, "Environmental Technology: Making the Real World Virtual," *Communications of the ACM*, vol. 36, pp. 36-37, July 1993.
- [45] J. J. Kuch and T. S. Huang, "Vision Based Hand Modeling and Tracking," *International Conference on Computer Vision*, June 1995.
- [46] J. Lee and T. L. Kunii, "Constraint-based Hand Animation," *Models and Techniques in Computer Animation*, pp. 110-127, 1993.
- [47] C. Maggioni, "A Novel Gestural Input Device for Virtual Reality," *IEEE Annual Virtual Reality International Symposium*, pp. 118-124, 1993.
- [48] J. M. Rehg and T. Kanade, "Digiteyes: Vision-based Human Hand Tracking," *Tech. Rep. CMU-CS-93-220*, 1993.
- [49] J. Schlenzig, E. Hunter, and R. Jain, "Visionbased hand gesture interpretation using recursive estimation," *28th Asilomar Conference on Signals, Systems, and Computer*, 1994.
- [50] J. Segen, "Controlling Computers with Gloveless Gestures," *Proc. of Virtual Reality Systems*, April 1993.
- [51] T. E. Starner and A. Pentland, "Visual Recognition of American Sign Language using Hidden Markov Models," *International Workshop on Automatic Face and Gesture Recognition*, June, 1995.
- [52] A. Torige and T. Kono, "Human-Interface by Recognition of Human Gestures with Image Pro-cessing Recognition of Gesture to Specify Moving Directions," *IEEE International Workshop on Robot and Human Communication*, pp. 105-110, 1992.
- [53] T. Huang and V. Pavlovic, "Hand gesture modeling, analysis, and synthesis," *Proc. International Workshop on Automatic Face and Gesture-Recognition*, Zurich, June 1995.
- [54] Vision Based Hand Gesture Recognition Systems.
<http://ls7-www.cs.uni-dortmund.de/research/gesture/vbgr-table.html>.
- [55] R. Collins, "Drawing Pixmap to Vector Conversion," United States Patent, US 6,173,075 B1, Jan. 2001.

[56] C. Manresa, J. Varona, R. Mas and F. Perales, "Hand Tracking and Gesture Recognition for Human-Computer Interaction," *Electronic Letters on Computer Vision and Image Analysis*, vol. 5, pp. 96-104, 2005.

[57] Hand Gesture Machine Control Systems, Patent Number 5594469, Jan.14, 1997.

[58] "Handbook Outline," <http://vehand.engr.ucf.edu/revised2.htm>.

[59] C. M. Bishop, "Pattern Recognition and Machine Learning," *Springer*, Oct. 1, 2007.

© BL-10-20