Design and Implementation of a Hand Tracking Interface using the Nintendo Wii Remote

Michal Piotr Wronski

A dissertation submitted to the Department of Electrical Engineering, University of Cape Town, in partial fulfilment of the requirements for the degree of Bachelor of Science in Computer and Electrical Engineering.

Cape Town, October 2008

Declaration

I declare that this dissertation is my own, unaided work. It is being submitted for the degree of Bachelor of Science in Computer and Electrical Engineering in the University of Cape Town. It has not been submitted before for any degree or examination in any other university.

Signature of Author .....

Cape Town

20 October 2008

Abstract

This project designs and implements a 3D hand tracking interface for aiding molecular visualisation applications. It uses the infrared optical sensor in the Nintendo Wii remote to track infrared LED light sources attached to a user's hands.

A review of the Wiimote's capabilities, sensors and interfaces is presented. Several conceptual models are discussed, and the most appropriate solution is chosen and implemented. A theory is presented that allows for very fast 3D tracking at a slight expense of accuracy. A 6 degree of freedom hand tracking interface is built that is less computationally intensive compared to proprietary tracking hardware, at a fraction of the cost.

It is concluded that 3D hand tracking is a novel, yet acceptable method of interfacing with a computer. The implementation allows for easier and more intuitive visualisation of complex molecules, and can easily be extended and adapted to other Computer Aided Design (CAD) applications.

Acknowledgements

I would like to acknowledge valuable advice and contributions from the following individuals:

- **Professor Mike Inggs**, for his supervision and assistance during the design and write up of this thesis. His suggestions and oversight proved invaluable, and he addressed my concerns quickly and professionally.
- **Sebastian Wyngaard,** for his insight and recommendations for the implementation of the hand tracking interface, and for providing me with an excellent starting point from which I could begin my research.
- My family, including **Mirek Wronski**, **Sonja Wronski**, **Eva Wronski** and **Jan Nel**. Without your love and support over the years, I would never have accomplished what I have.
- **Vladimir Mtei**, for helping me formulate my ideas and for the numerous technical discussions we shared that broadened our perspective and knowledge.

I would like to thank the **Centre for High Performance Computing** for providing me with testing equipment and a computer for the duration of my thesis, and for allowing me to use their facilities.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background to Investigation

The Nintendo Wii Remote (Wiimote) is an extremely versatile device that incorporates both a high precision accelerometer that can capture motion with 3 degrees of freedom, as well as an infrared (IR) camera that can detect sources of IR light. The Wiimote communicates with a host controller, such as the Wii console or a PC, via wireless Bluetooth technology. Using this wireless connection, it can transmit accelerometer data and IR camera pixel coordinates at frequencies of 100 Hz.

The components of the Wiimote allow it to be used in a variety of ways, but it is typically used as the hardware interface for games played on the Wii console. The user holds the Wiimote and moves it around, and the motion data is sent to the host controller. Connected to the console is a sensor bar, which features a set of 5 IR Light Emitting Diodes (LEDs) on each side. The console subsequently calculates the relative 3D location of the Wiimote via triangulation, based on the relative positions of the IR LED sources detected by the IR camera.

Figure 1.1: Wii console with the Wiimote (left).

The Wiimote need not only track the IR light being emitted from the sensor bar. It can be utilised as a tracking device for practically any source of IR light, including candles,

IR pens and even natural daylight. Using this idea, the Wiimote can be mounted on a stationary object, and any sources of IR light within the camera's field of view will be accurately tracked. In this particular setup, the accelerometer will not be used since the Wiimote remains stationary.

Using the concept of tracking any source of IR light, a set of IR LEDs can be mounted on a user's hands and detected by one or more Wiimotes. This data can then be used to reconstruct the approximate positions of the user's hands in space, which can in turn be used as the basis of a software interface of a Computer Aided Design (CAD) package or for navigation in an arbitrary software program. The application in this project focuses on the effective visualisation and transformation of molecules utilizing hand tracking.

1.2 Objectives of the Investigation

This project investigates the design, performance and accuracies of tracking one's fingers using the technique outlined. The objective is to successfully build and test a working prototype for use in a molecular visualisation interface at the Centre for High Performance Computing (CHPC) in Cape Town. Several conceptual models are outlayed, and the most appropriate design is selected and implemented.

The potential of the Wiimote as a tracking device in an engineering context will be analysed, with emphasis on:

- The underlying hardware functionality and the communication with relevant software Application Programming Interfaces (APIs);
- The stability of the system over an extended usage period and with different sources of IR light;
- The dynamic range, sensitivities and sensing limits of the optical sensor;
- The performance characteristics of the IR camera, such as visible pixel range, field of view and tracking distance;

The factors influencing the above list are examined and quantitavely analysed.

A hand tracking algorithm is constructed based on the selected design, and is analysed with respect to its efficiency, viability and execution time. The algorithm is coded and used in conjunction with a system of LEDs and Wiimotes to perform hand tracking. The system is analysed for its speed, accuracy, cost and viability and conclusions from these results are drawn.

An appropriate open source software Wiimote API is examined and modified as necessary for use in this project. Performance modifications are programmed if possible, in order to more efficiently interact with the software's Graphical User Interface (GUI).

1.3 Scope and Limitations of this Investigation

This project only deals with the theory and application of hand tracking using the IR camera on the Wiimote. Several available software APIs will be analysed and only the most appropriate package will be chosen for the application. The application of the API will be tested in a molecular visualisation package only, although it will be designed to easily connect to any higher level software applications through a modularized, layered software model. The limitation of the project design to only one application is primarily due to the time constraints of an undergraduate thesis. However, useful insight is gained from studying the implementation of the device in a high performance computing environment.

1.4 Plan of Development

The scope of the remainder of this document is outlined as follows:

- A detailed literature review is presented, providing details of the underlying hardware of the Wiimote and its various hardware extensions, as well as an in depth analysis of the software API that will be used
- An explanation of the environment in which the system operates is given, followed by information regarding the layered approach to how the hardware, software driver and software GUI communicate in the system.
- A list of the most viable conceptual models is presented, giving their advantages and disadvantages.
- A hand tracking algorithm is created and described based on the selected model.
- A set of results is given describing the functionality and technical details of the constructed system, as well as a description of its accuracies and limitations.
- Conclusions are drawn from these results, and recommendations for further use of the finger tracking system are given.

Chapter 2

Literature Review

In this chapter, an overview of the Wii console and its peripherals is given. This provides some details of the overall system hardware and software interfaces that are used with the Wii system. Since the Wiimote is the only necessary device in the Wii system for this project, a seperate section is given to providing an in-depth review of the Wiimote's capabilities. An overview of the available software APIs is presented, as well as an analysis of the library selected for this project.

2.1 Console and Peripherals Overview

The Wii is a video game console produced by Nintendo. Nintendo has attempted to keep detailed hardware information concerning their console to a minimum. However, several individuals have reverse engineered the Wii hardware and the technical details are fairly well known already.[1]

The core Wii CPU is a PowerPC based processor codenamed "Broadway", clocked at 729 Mhz. The graphics processor is a custom made GPU from ATI, running at 243 Mhz. It should be evident that these hardware specifications are geared towards the mid-level market, since Nintendo games are typically not as computationally or graphically intensive as their competitor's products, such as the Microsoft XBox 360 and the Sony Playstation 3.

The Wii has support for up to four Wii Remote controllers, connected via Bluetooth[2]. This allows up to four players to connect simultaneously in order to play a game or access the Wii console functionality. Connected to the Wii console is a sensor bar. The sensor bar contains 10 IR LEDs, with 5 positioned on each side. The LEDs are angled outwards in order to maximise the sensing range of the Wiimote's IR camera. Despite its name, the sensor bar does not perform any sensing functions. It is merely a set of LEDs that is powered through a proprietary connector from the Wii console. No data is read from or written to the sensor bar, and the light intensity is not modulated in any way.

[1] http://www.wiire.org/

[2] For an overview of the Bluetooth protocol, consult Appendix A.

Figure 2.1: Wii Sensor Bar.

More specific details regarding the console are tabulated below:

Name	Description
Memory	24 MB onboard, 64 MB GDDR3, 3 MB GPU texture memory
Storage	512 MB flash memory
Ports	SD memory card slot, AV multi-port, 2 USB ports, 4 Gamecube controller ports, 2 Gamecube memory card ports
Video	Up to 480p NTSC or PAL/SECAM
Audio	Dolby Pro Logic II-capable
Other	802.11b/g Wi-Fi capability, bcm4318 compatible

The Wii console is roughly 2x as powerful as its predecessor, the Nintendo DS. However, it is the least powerful mainstream consumer gaming console in its generation, competing with far more powerful devices. However, it features several novel interfaces and extensions that have made it exceedingly popular, with 3,629,361 consoles having been sold in 2007.[3] At the time of writing, a single Wii console can be purchased for R3400 in South Africa, and a single Wiimote costs R650.[4]

Several peripheral extensions for the Wii console have been manufactured, and are described briefly.

2.1.1 Nunchuk

The Nunchuk controller is a peripheral that is connected to the Wiimote via a cable. It contains a 3D accelerometer to detect acceleration with 3 degrees of freedom, as well as an analogue stick and two buttons on the front end of the controller. Data is sent to and received from the Nunchuk by writing to and reading from a certain address range on the Wiimote's flash memory. The Wiimote in turn relays the data to the host controller via Bluetooth. The constant calibration data for the Nunchuk's accelerometer, which is used to determine the correct balance of forces during force measurements, is stored in its internal flash memory. The calibration data is determined after manufacture and manually

[3] http://www.nintendo.com/whatsnew/detail/UQkQDsY6UcUbiHdjvIEXMrbwcYk2sbpx
[4] http://www.incredible.co.za

uploaded to the device. These values are added to the accelerometer readings on each axis, in order to accurately compensate for the effect of gravity on the accelerometer and minor manufacturing defects.

Figure 2.2: Nunchuk connected to Wiimote.

Classic Controller

This peripheral connects to the Wiimote in a similar fashion to the Nunchuk. It contains two analogue sticks and several buttons for input, but unlike the Wiimote and the Nunchuk, it contains no accelerometer. However, it has proprietary connecting slots on its back side in order to directly hook onto a Wiimote, in order to take advantage of the Wiimote motor rumble and accelerometer functions.

Anascape Ltd filed a lawsuit against Nintendo claiming that the Classic Controller and other Nintendo devices violated Anascape's "six degrees of freedom" interface device patent. In July 2008 the verdict ruled in favor of Anascape and Nintendo was ordered to stop selling the Classic Controller in the United States until further notice.[5]

Balance Board

The Balance Board is another peripheral available for the Wii and has a similar shape to a body scale, featuring a flat rectangular design. The board contains four pressure sensors situated at each corner. These are used to determine the user's centre of balance and body mass index. However, for fully accurate sensing, the board must be positioned on a flat, hard surface. The board can be powered for up to 60 hours with four AA batteries, and communicates via Bluetooth with the host controller. It is thus not dependant on the Wiimote for communications, unlike the Nunchuk peripheral.

[5] http://www.bloomberg.com/apps/news?pid=newsarchive&sid=aO_ucYxT3eNw

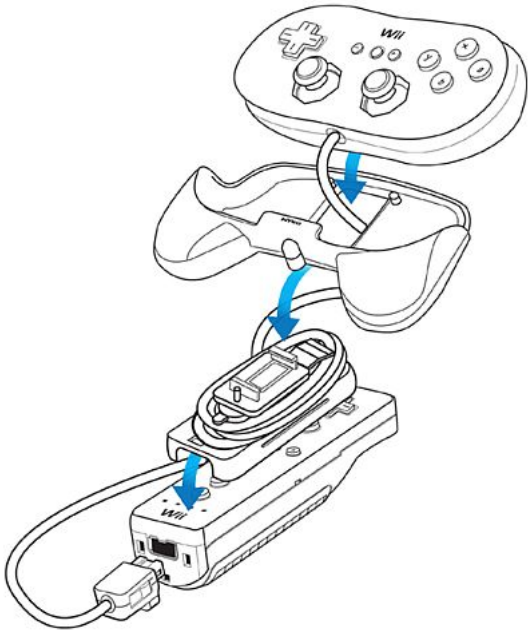

Figure 2.3: Classic Controller connected to Wiimote.

The board was closely developed with the Wii Fit game, which utilizes the technology. Wii Fit contains several training programs where a user can select from yoga, strength training, aerobics and balance games. The Wii Fit proved to be a huge success, selling over 1,000,000 copies in its first week.[6]

In an interview conducted by gaming web site IGN, Shigeru Miyamoto stated that the Balance Board's ability to measure weight is probably more accurate than that of a typical bathroom scale.[7] The Balance Board can withstand a mass of up to 300 kg, although the pressure sensors do not read any values above 136 kg in the Japanese version, and 160 kg in the "Western" version.

Figure 2.4: Wii Balance Board.

[6]http://www.gamesindustry.biz/content_page.php?aid=31293
[7]http://uk.media.wii.ign.com/articles/804/804464/vids_1.html

2.2 Wiimote Technical Specification

2.2.1 Hardware Details

The Wiimote is a wireless communications device that transmits and receives data via a Bluetooth link. It has an on-board Broadcom 2042 Bluetooth driver chip that facilitates this purpose. The chip is a class 2 Bluetooth device that fully supports the Bluetooth human interface device (HID) standard and comes complete with a Bluetooth stack, a 16 bit 8051 microprocessor and on-board RAM/ROM memory.[8] The Wiimote does not use any of the authentication or encryption features of the Bluetooth standard. Appendix A gives an overview of Bluetooth communications technology.

The Wii Remote contains a 16 KB EEPROM chip from which a section of 6 kilobytes can be freely read from and written to by the host controller. This allows for easy storage and transportation of custom application data, given the wireless nature of the Wiimote. Certain hardware and software calibration settings, as well as interface options, can be stored in the Wiimote in this fashion and can be accessed anywhere by an appropriate host.

Perhaps the most relevant hardware component of the Wiimote for this project is the IR camera, located at the front end of the remote. On May 12th, 2006, PixArt announced a strategic relationship with Nintendo to provide object tracking technology exclusively for the Nintendo Wii Remote.[9] The camera features a resolution of 1024x768 with built-in hardware for IR "blob" tracking of up to 4 points at 100Hz. The camera contains a Charge Coupled Device (CCD) with an IR filter attached that senses light within the near-infrared spectrum, at wavelengths of greater than 800 nm. There are reports that the camera operates at 128x96 resolution, and uses 8x8 sub-pixel analysis to create a virtual resolution of 1024x768. This, however, is abstracted from the higher level user interface and only a constant pixel resolution of 1024x768 is made available to software APIs. Due to Bluetooth transmission bottlenecks, it would be highly inefficient to transmit all pixel values of the camera at its operating frequency, and only information concerning the detected IR blobs are transmitted. Blobs are essentially IR hot spots that the camera senses, from which pixel coordinates and relative blob sizes are extracted and transmitted across the wireless link. Any IR light source can be tracked to some degree, including incandescent lamps, torches, candles and daylight. IR LEDs are available that can be tracked accurately and efficiently, due to their specially matched IR wavelengths, high luminescence and small active areas, making them ideal for use in applications using the Wiimote's optical sensor. According to recent reports, 940 nm IR sources are detected with approximately twice the intensity of equivalent 850 nm sources, but are not resolved as well at close distances. If the IR filter is removed from the camera, it can track any

[8] http://www.broadcom.com/collateral/pb/2042-PB03-R.pdf

[9] http://www.pixart.com.tw/investor.asp?sort=4&learname=level03

bright object.[10]

The motion of the remote is sensed by a 3-axis linear accelerometer located towards the front of the remote. The integrated accelerometer circuit is the ADXL330[11], manufactured by Analog Devices. This device is physically rated to measure accelerations over a range of at most +/- 3g with 10% sensitivity. Inside the chip is a mechanical structure which is supported by springs built out of silicon. Differential capacitance measurements allow the net displacement of the tiny mass to be converted to a voltage, which is then digitized. The voltage reading is proportional to the acceleration of the tiny mass inside the circuit. It should be noted that acceleration is not directly measured with this technique, but the force that the mass exerts on the test springs attached to the capacitors is measured. The digitized values are added to the internally stored calibration constants, which are used in a similar nature to those of the Nunchuk. When at rest, the Wiimote reports a vertical force reading of 1 g (9.81 m/s^2) due to gravitational pull on the internal mass. When dropped, the Wiimote reports an acceleration of approximately 0. Manufacturing and calibration defects do cause some intrinsic zero offsets, but the chip is self-calibrated during manufacture to produce offset values for any measurement inaccuracies present. These calibration values are stored near the start of the Wiimote's flash RAM.

Figure 2.5: Wiimote viewed from different angles.

There are 12 buttons on the Wiimote. Four of them are arranged into a directional pad as shown above, and the rest are spread across the controller. The proprietary connector port for the Nunchuk and classic controller is shown on the left of the above image. The IR camera housing is shown on the right side of the image, which is the front of the Wiimote. The bottom edge of the remote body contains 4 blue LEDs. These LEDs are used to indicate that the Wiimote is in discoverable mode by blinking, and in normal game play are used to show battery level. This functionality can be overridden via a Wiimote API on a PC.

In order for a host to interface with the Wiimote, it must be put into discoverable mode by pressing the 1 and 2 buttons simultaneously. Once in this mode, the 4 Wiimote LEDs will flash on and off and a 20 second period will be allocated in order to connect to a Bluetooth

[10] http://wiibrew.org/wiki/Wiimote#IR_Camera

[11] http://www.analog.com/static/imported-files/data_sheets/ADXL330.pdf

HID driver on the host. If connection is not established within this period, the Wiimote will automatically switch off. A Bluetooth dongle connected to a PC with an appropriate Wiimote API installed is sufficient to establish a complete full duplex connection to the Wiimote.

The Wiimote also contains a small low-quality speaker, used for short sound effects during game play. The sound is streamed directly from the host, and the speaker has some adjustable parameters. At the time of writing, the speaker functionality has not been completely reverse engineered yet and produces very low quality sound output with libraries that support its functionality.

2.2.2 Bluetooth Communication

The Wiimote is recognized as an Human Interface Device (HID) class peripheral, and the Bluetooth HID standard allows devices to be self-describing using an HID descriptor block. This block includes an enumeration of reports that is transmitted to the host via the wireless link at the beginning of communication. The HID descriptor lists report IDs and payload sizes for the duplex communication link between the host and the controller.

The Service Discovery Protocol (SDP) is the basis for discovery of services on all Bluetooth devices. Using the SDP device information, services and the characteristics of the services can be queried and, subsequently, a connection between two or more Bluetooth devices may be established. Like all Bluetooth HID devices, the Wiimote reports its HID descriptor block when queried using the SDP protocol.[3] When queried with the SDP, the Wiimote reports the following information shown below. A simple query can be made in Linux using the sdptool command.

Attribute	Data
Name	Nintendo RVL-CNT-01
Vendor ID	0x057e
Product ID	0x0306

The values in the above table are unique to the Wiimote and identify it among the range of all registered Bluetooth devices. All Wiimotes also have a built in mac address inside of the Broadcom communications chip that allow for unique hardware referencing during Bluetooth communication. The following table shows the HID descriptor block for all communication originating from the host:

Report ID	Payload Size	Known functions
0x11	1	Player LEDs
0x12	2	Force Feedback
0x13	1	IR Sensor Enable
0x14	1	Enable speaker
0x15	1	Controller status
0x16	31	Write data
0x17	6	Read data
0x18	21	Speaker data
0x19	1	Mute speaker
0x1a	1	IR Sensor Enable 2

All Wiimote APIs use the above information to transmit data using the Bluetooth link. For example, in order to enable the Wiimote's IR camera, a packet needs to be sent from the host containing the hexadecimal ID 0x13 and a positive integer in the 1 byte payload field.

2.2.3 Limitations

Unfortunately much information about the Wiimote has been garnered from reverse engineering of the PCB and hardware integrated circuits, since Nintendo has been rather secretive with releasing detailed technical information for the Wii and its peripherals. Several details of the design are not yet fully understood, and access to certain functions are still restricted. At the time of writing, the following details of the Wiimote's design have not yet been successfully reverse engineered:[12]

- Enabling the classic controller connected to the Wiimote, and reading events from it such as button presses and joystick movements;
- Accessing the Wiimote's power button functionality;
- Successfully playing sound on the speaker without distortion;
- The format of half rate IR reports from the camera;
- Gaining complete control of the Wiimote's expansion port;
- Executing arbitrary code on the 8051 processor;
- A full map of all the available registers is not yet known.

[12] http://www.wiili.org/index.php/Wiimote_Reverse_Engineering

2.3 Software Selection

2.3.1 Operating System

An Ubuntu Linux system was chosen for the basis of this project, due to its vast array of development tools and easy access to the source code of all relevant libraries. Thus, only Wiimote interface libraries for Linux are considered.

The BlueZ library for Linux facilitates Bluetooth communications. All of the considered Linux Wii libraries interface with BlueZ for access to the underlying Bluetooth protocol. It provides a complete modular implementation of the Bluetooth stack and functionality. It also supports multithreaded data processing and symmetric multiprocessing, while allowing access to multiple Bluetooth devices simultaneously.[13]

2.3.2 Wiimote API

At the time of writing, there is a vast array of different Application Programming Interfaces (APIs) for Linux that interface with the Wiimote. The selection for this project is based on the library that provides the following characteristic features:

- Clean, simple and efficient codebase that can easily be extended.
- Appropriate documentation of the library's functionality and internal workings.
- Access to most, if not all, of the Wiimote's functionality.
- Access to multiple Wiimotes simultaneously.

Six packages were analysed for their appropriateness and conformity to the above requirements, and the library that matched the specifications of this project most optimally was cwiid[14], a lightweight event based API. This library has been written in C for faster processing and easier access to underlying hardware mechanisms.

Message packets are read from the Wiimote via a callback function that is registered once connection to the Wiimote has been established. All received packets are decoded and then further processed in the customized callback function. Each received message contains the following information:

- A message type, which indicates the context of the message data. This is used to distinguish between the available input readings, including accelerometer data, IR tracking data, button presses and battery level.

[13] http://www.bluez.org/about/
[14] http://abstrakraft.org/cwiid/

- A timestamp, which indicates the exact time of message generation. This is useful in time based processing, although typically a first-come-first-served approach is used when processing packets.
- Message payload, which contains the specific data values relating to the message type.

Various functions of the Wiimote, such as IR tracking, LED control and motion capture can be selectively enabled and disabled by toggling bits in a binary sequence, known as a report mode. The bit sequence, which is represented by an unsigned character type in C, allows 8 bits to be independently changed simultaneously. The updated report mode is passed to a library function which updates the corresponding parameters on the Wiimote.

Accelerometer readings and button presses of a connected Nunchuk are also processed by the library and sent through to the callback function for further processing. Much of the underlying implementation, such as low level hardware communications and packet processing, is abstracted from the front end application. This enables easy rapid development with simple, but almost fully complete, access to the Wiimote's functionality.

2.3.3 Molecular Visualisation Software

There are several packages that facilitate 3D visualisation of molecules in Linux. Avogadro[15] was selected for this project. It is a stable cross-platform, open source package that has been in development since late 2006.

Avogadro is an application that has been developed with the idea of extensibility in mind. Plugins are easily created and installed in Avogadro, providing full access to its underlying functions and interfaces. It includes interactive tools, such as an embedded Python interpreter, to run commands and Python scripts in real time that access various widgets in Avogadro. It can quickly process and display complex molecules using OpenGL.

This project aims to create an intuitive hand tracking interface to visualize complex molecules in Avogadro. The molecular viewport, where the molecules are rendered in Avogadro, is accessed using its internal camera widget. This allows for easy perspective transformations that are directly available to plugins, such as scaling, rotation and translation of molecules. A plugin can be compiled with the Avogadro package, and is later accessed via Avogadro's main menu. Thereafter, a plugin can register a "redraw" function which is called per frame update of the molecular viewport. Once the frame rate is measured, it is possible to extract frame timing information in order to standardize the transformations of molecules at constant velocities, independent of frame rate.

An in-depth analysis of conceptual models for the hand tracking application is presented in Chapter 3.

[15] http://avogadro.openmolecules.net/wiki/Main_Page

Chapter 3

System Design

3.1 System Layout

A modular approach is taken when designing the system. The overall system layout is fairly complex, and the author has chosen to take a layered approach to providing an adequate solution to the given problem. The overall system layout is presented as follows:

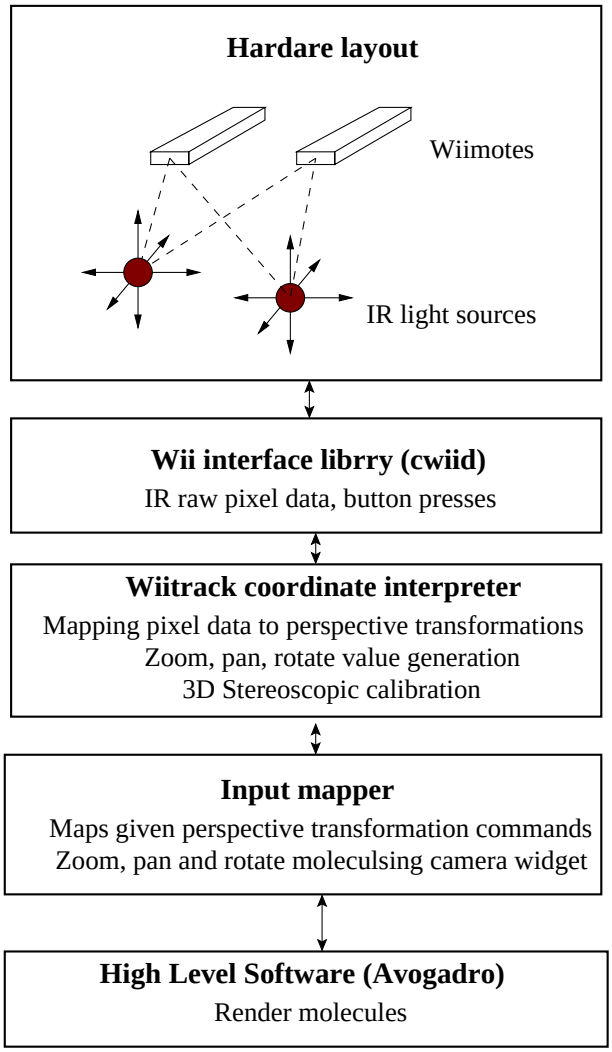

Figure 3.1: Layered system development diagram.

The above system is developed using a spiral methodology. The spiral methodology reflects the relationship of tasks with rapid prototyping, increased parallelism, and concurrency in design and build activities. The spiral method should still be planned methodically, with tasks and deliverables identified for each step in the spiral.[1]

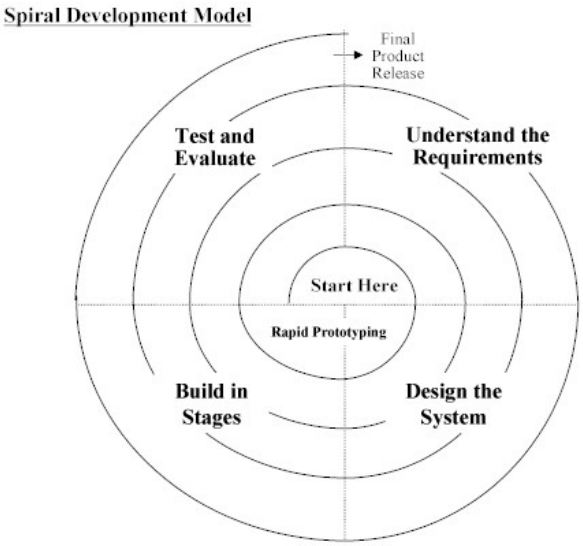

Figure 3.2: Spiral software methodology.

In order to implement the following proposed algorithms effectively, certain intrinsic variables of the camera need to be known, including the fields of view in both the horizontal and vertical directions. The perspective distortion of the lens is not taken into consideration in the proposed algorithms, due to the assumption that the camera represents a linear homogeneous system in this project.

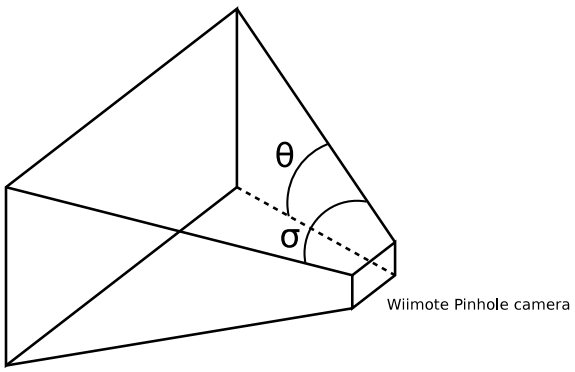

Figure 3.3: Field of view symbols for the Wiimote's camera.

One or more IR LEDs are connected to the periphery of a user's hands, and sensed by one or more Wiimote cameras. These LEDs are in turn sensed as IR hotspots by the Wiimote system. Regardless of the design selected, the molecular viewport updates are performed on a per-frame basis. After each successive frame, the differences in the Wiimote camera's recorded IR blob positions between the current frame and the previous frame, measured in pixels, are calculated. These differences are multiplied by the time

[1] http://www.hyperthot.com/pm_sdm.htm

elapsed between the successive frames in order to create a time-relative velocity vector, thus standardizing the transformation rates and make the molecule's motion independent of frame rate. The values are also multiplied by pre-determined constants to allow for smooth molecular transformations, according to what is intuitively the optimal transformation rate of the molecule. After the values required for translation, rotation and scaling have been established, Avogadro's camera widget is accessed in order to apply the perspective transformations.

Several design concepts are presented in this chapter. The most appropriate design is selected and built, as well as tested according to the testing methodology given afterwards. The conceptual designs are presented in increasing orders of complexity, implementation and execution cost, and degrees of freedom (DOF).

3.1.1 Minimal Tracking with 2 DOF

A fairly basic, albeit crude, design can be constructed with a single LED light source attached to one hand, and a single Wiimote as shown.

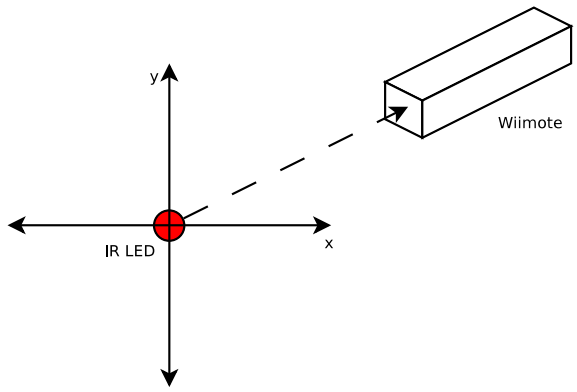

Figure 3.4: Tracking system with 2 degrees of freedom.

The Wiimote detects the single IR light source, and reports its (x, y) position to the host. The host uses the rate of change of x and y coordinates as the basis for tracking a hand with 2 degrees of freedom as the subject is able to pan vertically and horizontally. It is important to note that the camera's measured x coordinates need to be reversed, since the camera reports its values from its own reference point. Thus if the the user moves left, the camera reports the LED light source as moving to the right, and vice-versa.

This design is very simple, understandable and fast to implement. No computationally intensive post processing is required on the data. However, since there is no depth tracking, the sensitivity of the camera changes as the subject moves closer and further away. Thus, if the subject is further away he will need to move his hand in greater steps of distance to compensate for the camera's larger viewing range at that point. More complex molecular transformations, such as rotation and zooming, are not available with this technique.

3.1.2 Basic 4 DOF Tracking

Basic 4 DOF Tracking, as demonstrated in a head tracking video made by Johnny Chung Lee[2], can be implemented to track a person's hand. This design involves using two sources of IR light of a fixed known distance apart, and a single Wiimote. A user's distance from the camera can be calculated by using the distance between the two measured light sources from the camera. If the camera reports two IR sources (x_1, y_1) and (x_2, y_2), a relative z coordinate can be calculated by using the inverse of the Pythagorean distance measured between the two points. The further the subject is from the camera, the closer the two points become due to the camera's perspective. The relationship between the subject's distance from the camera and the pixel distance between the points is linear, but due to non-linear perspective distortion of the camera's lens, the greatest change in measured pixel distance occurs when the subject is closest to the camera.

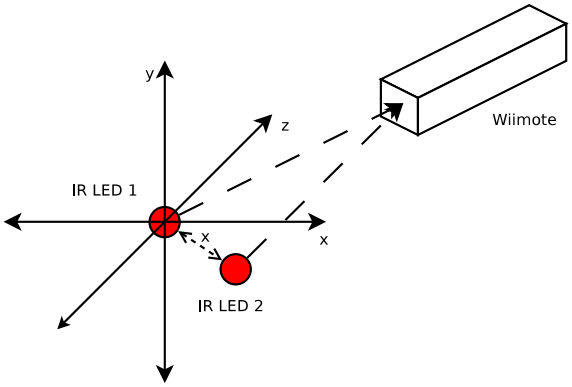

Figure 3.5: Tracking system with 4 degrees of freedom.

If the camera's horizontal field of view is represented as the symbol σ, where x is the fixed distance between the two points, and d is the subject's distance from the camera, the basic camera model may be represented as shown.

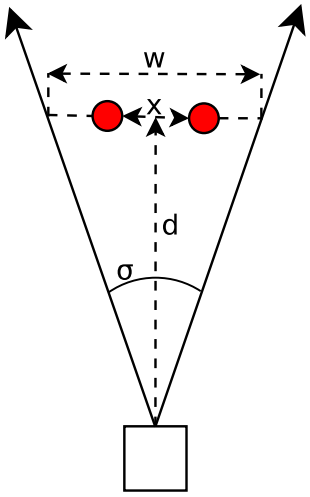

Figure 3.6: Camera model showing triangulation principle.

[2]http://www.cs.cmu.edu/~johnny/projects/wii/

The width of the horizontal viewing range at a distance d from the camera can be shown as follows.

$w = (2\tan(\sigma)).d$

If the two sources of light are not rotated in any way, and are kept horizontally parallel to the ground, then an algorithm can be constructed based on the ratio of the measured pixel distance between the light sources r, compared to the overall horizontal viewport pixels. The ratio between r and its maximum value of 1024 can be equated to the ratio between the physical distance between the LEDs, x, and the horizontal viewing range w at a distance d from the camera. The concepts are mathematically formulated as follows.

$\frac{x}{w}.1024 = r$

$\frac{x}{[2\tan(\sigma)].d}.1024 = r$

The above equation shows linearity between distance from the camera d and the corresponding horizontal pixel distance r. If the subject moves away from the camera, a constant proportional amount of measured pixel distance will be decreased. The symbols x and σ are system parameter constants.

This design is fast and simple, and is similar to the method used in the Wii console to measure the player's relative location. However, LEDs must face directly ahead and may not be arbitrarily rotated, and lens distortion does have an impact at excessively large and small distances from the camera. It is a simple technique of implementing 3 DOF tracking. Since the LEDs can be rotated along the z axis without distorting the measured pixel distance between them, the measured angle between the pixels allows for another degree of freedom. The system thus essentially tracks an object with 4 DOF, including the 3 translational degrees of freedom as well as the angular roll of the object. Pitch and yaw are not tracked using this system.

3.1.3 Highly Accurate 3 DOF Tracking

The concept presented previously may be modified somewhat to allow for more accurate tracking of a subject. The model presented requires two Wiimotes, but only one point source of light.

The two Wiimotes are placed parallel to each other and at a fixed distance apart. The respective 2D pixel coordinates of the same light source in both cameras are stereoscopically resolved into a 3D coordinate, which represents the IR Light source's true position in 3D space. The basic system layout is shown as follows:

Overall, this system is both financially and computationally more expensive than the previously presented models. It does not contain any rotational degrees of freedom since the rotation of a single point source of light cannot be measured. However, with an additional light source being added to the system, a full 6 DOF can be established by analysing the angular and positional differences between the two light sources in 3D.

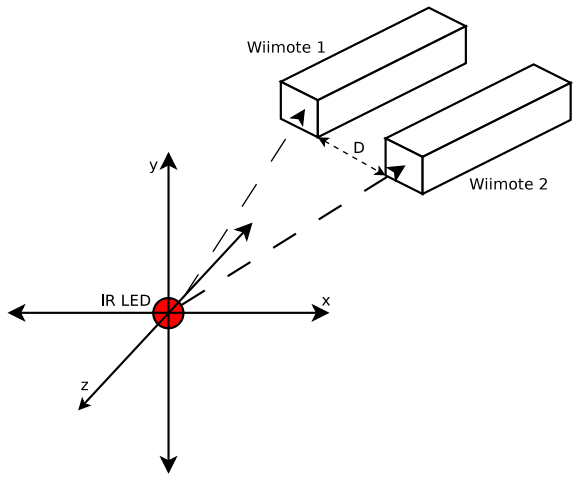

Figure 3.7: System showing implementation of accurate 3 DOF tracking.

Chapter 5 provides an explanation of the underlying details of implementation using this method. It will be shown that this technique is effective, powerful and of adequate accuracy for this project's implementation.

3.1.4 6 DOF Tracking

In order to implement complete tracking of a subject with a full six degrees of freedom - that is, movement along all 3 axes with independent rotation along each axis, an extra point source of light needs to be added to the previously presented conceptual model. This is illustrated below.

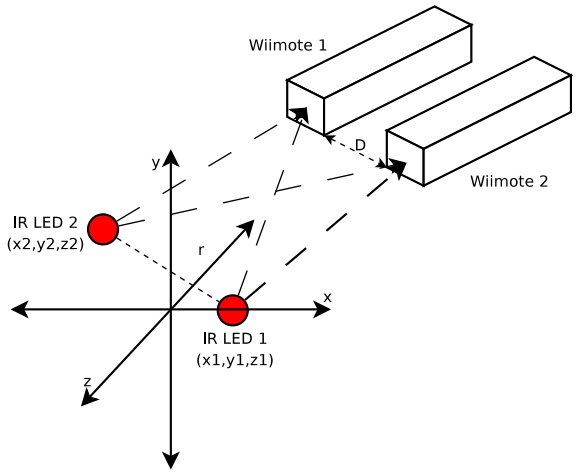

Figure 3.8: System showing two light sources for 6 DOF tracking.

All six degrees of freedom can be effectively independently resolved using 3D tracking with 2 points sources of light. The respective angles and distances used in the tracking algorithm are shown below:

In this project, the vertical angle α between the two points in 3D is mapped to the angular roll of the rendered molecule on the screen. The horizontal angle β determines the yaw of the molecule. The distance d between the light sources is mapped proportionally to

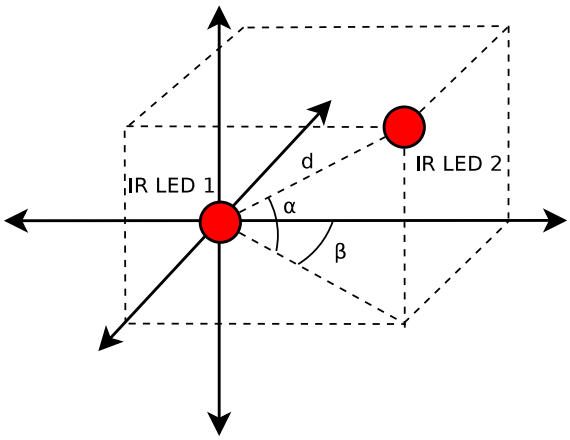

Figure 3.9: Variables in the 6 DOF tracking model.

the zoom of the molecule. The central position between the two light sources gives an indication as to where the average location of the LEDs are in 3D space. The respective x and y values of this central coordinate are used for horizontal and vertical panning of the molecule. The z coordinate, that is the subject's distance from the camera, is mapped to the pitch of the molecule. All of this information is further explained in Chapter 5.

This system is more complex and computationally costly compared to the previous algorithms, but can effectively track a subject with all of its mechanical degrees of freedom. However, the LEDs must always remain in the cameras' viewing ranges and need to face the cameras at all times in order for the 3D tracking algorithm to be effective. Thus there are two important restrictions that are considered:

- The subject needs to move within both the cameras' fields of view simultaneously in order for proper system functioning.
- Both cameras need to capture all point sources of light continuously and thus the lights need to face the cameras at all times.

The best approach to improve the restrictions given above would be to move the cameras further apart and point them inwards slightly, resulting in improved area of the resulting combined viewport. This is illustrated as follows:

In the figure, the rotated Wiimote layout shown on the right has less overall viewing area compared to the parallel Wiimote orientation. However, as the arrows indicate, the shape of the combined area is more intuitive in allowing for more natural movement without crossing the boundaries of the camera's viewports, as may be the case in the initial implementation. It also allows for more flexible rotation of the actual LEDs, which require having a line of sight to the cameras. This would be the ideal setup in a Computer Aided Design (CAD) implementation of this technology, where the user would not move his/her hands too far away from their resting positions, and it would allow for more accurate tracking. In this project's implementation however, the camera's are stationed parallel for improved measuring accuracy.

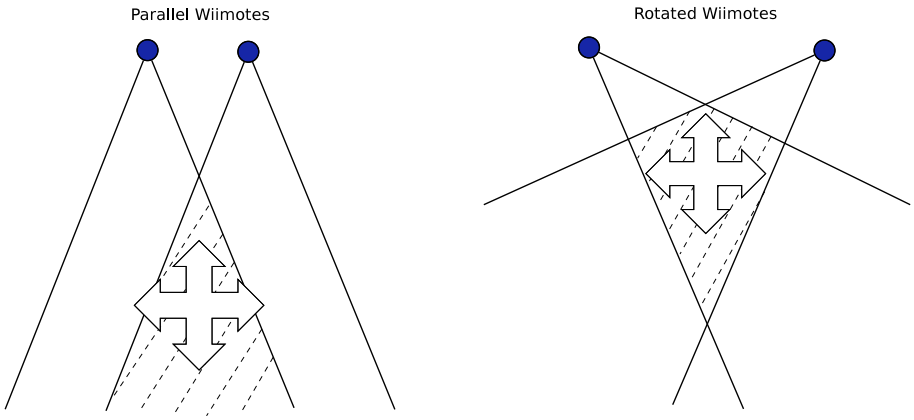

Figure 3.10: Two possible Wiimote system layouts.

The 6 DOF tracking presented in this section is selected as the design for this project. The author's 3D stereoscopic tracking algorithm is explained in chapter 4, and implementation details are presented in Chapter 5.

Chapter 4

3D Stereoscopic Tracking

4.1 Overview

The human visual system operates with a complex neural feedback mechanism that resolves each image captured by the eyes into a 3D landscape. Effectively, two or more cameras can be used to track 3D motion of an object. This project involves the tracking of point sources of light, and so the burden in calibrating the cameras and extracting important pixel data is much reduced because only individual points of known coordinates need to be tracked. However, the fundamental concepts remain the same.

In order to track an object in 3D space with 2 cameras, the intrinsic parameters of each camera need to be known.[7] These intrinsic parameters are typically stored as a matrix, and include data such as the focal length, image center and size of the image sensor of the CCD. Typically a distortion skew matrix is also used to account for lens distortion that exists in all lens cameras. However, in this project the Wiimote camera is assumed to be a linear homogeneous system and only two important properties of the camera are needed for internal intrinsic calibration. These values are the horizontal and vertical fields of view of the camera.

In order for both cameras to be referenced using the same coordinate system, the extrinsic parameters of the camera system need to be resolved.[8] These include the translation and rotation vectors of one camera in relation to the other. In the constructed model, the cameras are pointed parallel outward and are located a fixed distance apart from each other, resulting in zero effective rotation and a simple translation in one axis. This is a proof of concept project and the model can easily be extended to incorporate more advanced transformations. The left-most Wiimote, from the reference point of the cameras, is identified and selected as the origin. All vector geometry is constructed with reference to the left-most Wiimote's position as the origin. The right-most Wiimote lies along the x-axis at a known distance from the origin, as manually measured when the system is constructed.

4.2 Implementation

Once the camera system's intrinsic and extrinsic parameters are known, it is fairly quick to resolve a point source of light into a 3D coordinate.[4] The method presented here is a technique using vector geometry and is taylor made for use in the Wiimote system.

It is important to note that every pixel in the camera can be mapped to a unique ray in 3D space, which propagates outward from the pinhole camera's own origin.[5] As an example, if a recorded pixel of light from the Wiimote camera has the coordinates $(0,0)$, then a ray may be constructed that propagates along the bottom left boundary of the camera's viewport and extends outwards. The exact direction of the ray depends on the camera's intrinsic parameters, namely the horizontal and vertical fields of view. This concept is illustrated in figure 4.1, with Ray 0 being the ray mapped to the pixel coordinates $(0,0)$, and a ray (x,y,z) being constructed from an arbitrary coordinate from a detected IR light source. The values of σ and θ are the camera's horizontal and vertical fields of view, respectively.

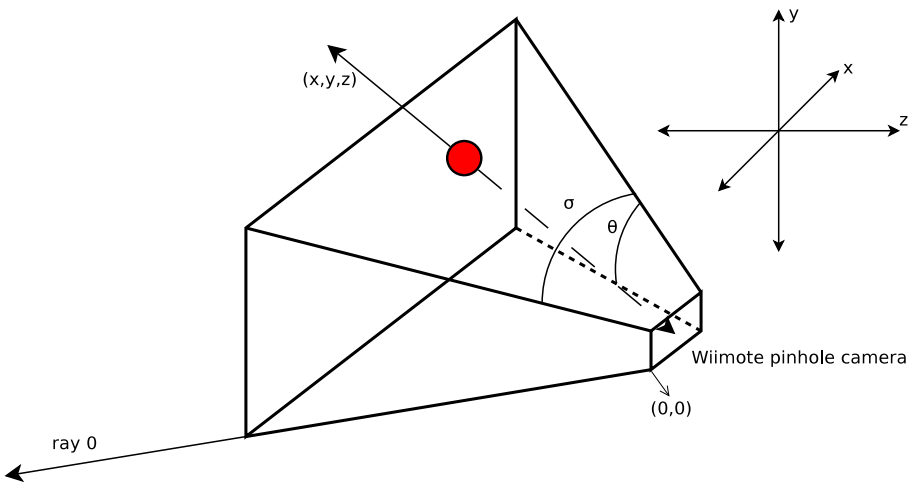

Figure 4.1: An example of mapping pixel coordinates to 3D rays.

As shown in figure 4.1, any sensed light source has a pixel value associated with it, and will lie on any point in the constructed ray. The mathematical approach taken in this project involves calculating the normalized pixel values $(\hat{x},\hat{y})$ such that they lie within the range $[-1;1]$, scaled from the normal pixel range of $[0;1024]$ and $[0;768]$. The normalizing equation is given below.

$\hat{x} = 2\frac{x}{1024} - 1$

$\hat{y} = 2\frac{y}{768} - 1$

As explained, any pixel with coordinates (x,y) can be mapped to a ray originating from the origin with parameters $\begin{matrix} x' & y' & z' \end{matrix}^T$. The Wiimote is assumed to point in the direction $(0,0,-1)$, straight down the z axis, in its own coordinate system.

$$\begin{matrix} x' \\ y' \\ z' \end{matrix} = \begin{matrix} \hat{x}.tan(\sigma/2) \\ \hat{y}.tan(\theta/2) \\ 1 \end{matrix}$$

With a single camera's measured values, a ray is the greatest amount of information one can extract from the 3D location of the light source. If another camera is added to the system that senses the same light source, the setup appears as follows:

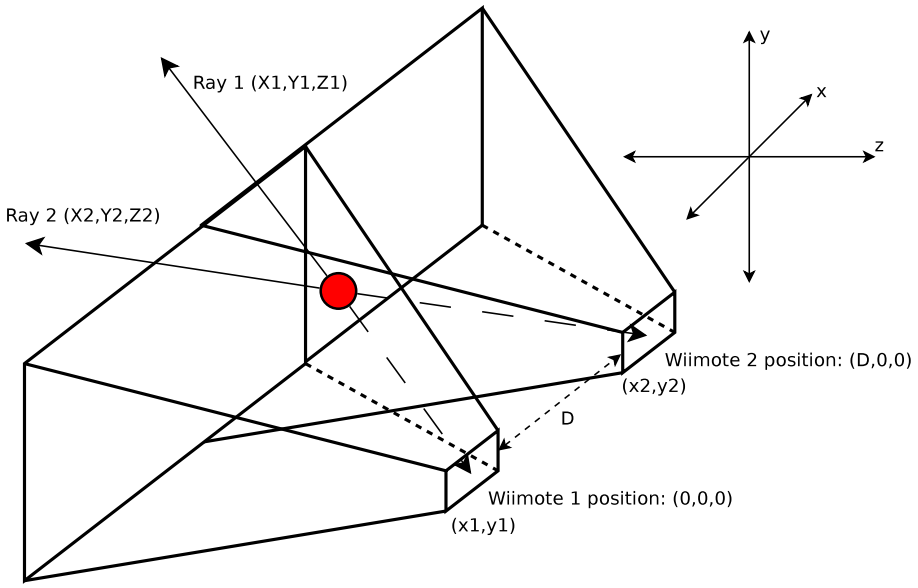

Figure 4.2: Ray constructions using two parallel Wiimotes.

As shown, with two cameras two separate rays are constructed. Since the orientation of one camera to another is known, a plane is formed between the locations of the two pinhole cameras and the 3D point. The 3D point is resolved by the intersection of the two known rays $\mathbf{r_1} = (x_1, y_1, z_1)$ originating from the origin, and $\mathbf{r_2} = (x_2, y_2, z_2)$ originating at the point $(D, 0, 0)$. The units of D determine the units of the final 3D calculated coordinate, and in this project all real world units are measured in centimetres.

If the cameras are pointed inwards for a more efficient viewing angle as shown in figure 3.10, the only underlying difference is the rotation of both constructed rays (x_1, y_1, z_1) and (x_2, y_2, z_2) around their y axes by the angles they are rotated from their normal rest orientation. If the angle between the Wiimotes is γ, and they are assumed to be rotated by the same angle from their origins, the left Wiimote needs to have its constructed rays rotated by $\frac{\gamma}{2}$ around the y axis, and the right Wiimote needs its rays rotated by $-\frac{\gamma}{2}$. This optional addition to the system can be implemented using the following yaw rotation matrix, where λ is the angle of rotation about the y axis:

$$M(\lambda) = \begin{matrix} cos(\lambda) & 0 & -sin(\lambda) \\ 0 & 1 & 0 \\ sin(\lambda) & 0 & cos(\lambda) \end{matrix}$$

The new left Wiimote ray is constructed with the product $M(\frac{\gamma}{2}).\mathbf{r}_1$ and the right ray, by symmetry, is constructed with the product $M(-\frac{\gamma}{2}).\mathbf{r}_2$.

Algorithms for the intersection of two rays[1] are well documented, and the final solution is presented here.

[1] http://www.realtimerendering.com/intersections.html

With two parallel Wiimotes, the two constructed rays from the two pinhole cameras to the sensed IR light position may be parametrized as follows:

$\mathbf{R_1} = \mathbf{o_1} + \mathbf{d_1}.t_1$

$\mathbf{R_2} = \mathbf{o_2} + \mathbf{d_2}.t_2$

where

$$\mathbf{o_1} = \begin{matrix} 0 \\ 0 \\ 0 \end{matrix}$$

$$\mathbf{o_2} = \begin{matrix} D \\ 0 \\ 0 \end{matrix}$$

$$\mathbf{d_1} = \begin{matrix} x_1 \\ y_1 \\ z_1 \end{matrix}$$

$$\mathbf{d_2} = \begin{matrix} x_2 \\ y_2 \\ z_2 \end{matrix}$$

$t_1, t_2 \in \mathbb{R} > 0$

The values of t_1 and t_2 that provide the points of intersection of the rays $\mathbf{R_1}$and $\mathbf{R_2}$ are as follows:

$$\hat{t_1} = \frac{det(\ \mathbf{o_2}-\mathbf{o_1} \quad \mathbf{d_2} \quad \mathbf{d_1}x\mathbf{d_2}\ )}{|\mathbf{d_1}x\mathbf{d_2}|^2}$$

$$\hat{t_2} = \frac{det(\ \mathbf{o_2}-\mathbf{o_1} \quad \mathbf{d_1} \quad \mathbf{d_1}x\mathbf{d_2}\ )}{|\mathbf{d_1}x\mathbf{d_2}|^2}$$

Due to the camera setup being a physical system, there will be measuring inaccuracies and errors. The two Wiimotes will never be perfectly parallel and the two rays will realistically almost never cross perfectly, but the values of $\hat{t_1}$and $\hat{t_2}$ give the points on each ray that are the closest to each other, irrespective of whether the rays actually intersect. The final 3D coordinate $\mathbf{x}$ is calculated as the average of the two calculated 3D points as follows:

$$\mathbf{x} = \frac{\mathbf{o_1}+\mathbf{d_1}.\hat{t_1}+\mathbf{o_2}+\mathbf{d_2}.\hat{t_2}}{2}$$

The presented algorithm can be extended with additional Wiimotes. If the extrinsic variables of all n cameras in the system are accurately measured, a 3D point $\mathbf{x_i}$can be created from each unique pairing of two Wiimotes. With 3 Wiimotes (namely W_1,W_2,W_3) in the system, the algorithm is executed 3 times. The 3D location of the same point source can be calculated using the constructed rays between W_1 and W_2, W_2 and W_3, and W_1 and W_3. The points are aggregated and averaged to form an accurate representation of the 3D light source. The only requirement is that the light source remains in all of the cameras' fields of view at all times. The averaging equation simply calculates the mean coordinate of all calculated 3D points:

$$\mathbf{x} = \frac{\sum \mathbf{x_i}}{n}$$

4.3 Extensions

Much research is being conducted into automatic calibration of the intrinsic and extrinsic parameters of camera systems for use in 3D vision. In this project, all parameters are measured manually beforehand, and are used as constants in the programming code. However, potential problems with this system may include:

- Unusual changes in the system that may change the Wiimotes' orientation or viewing angles, such as bumping one of the Wiimotes out of place.
- Restriction of only 2 Wiimotes in the system, leading to less measurement accuracy.
- No consideration of lens distortion or skewing.

Due to the time constraints of an undergraduate thesis project, the above problems could not be suitably addressed in the model. However, research into possible solutions has been conducted and is made available in the final chapter on future work.

Chapter 5

System Implementation

5.1 Hardware Interface

Two Wiimotes are used to reconstruct the 3D coordinates of 2 respective IR LEDs in space. A single IR LED is attached to the tip of each index finger, along with a small Lithium coin cell, resistor and connecting wires. The circuit is small enough to be attached to the tip of the index finger of a glove which can be easily equipped on the user's hand.

Figure 5.1: Photo of the LED system mounted on an index finger.

The two Wiimotes are placed parallel to one another, 20 cm apart. Both Wiimotes sense IR light data from both of the LEDs at the same time, which is in turn sent to the host computer and, subsequently, the software interface. In order to distinguish between the 2 LEDs detected, an algorithm is used which assumes the user's hands do not cross one another when navigating through the interface. Thus, it is guaranteed that the left most IR light as measured by both Wiimotes corresponds to the user's right hand, and vice versa.

5.2 Software Interface

Avogadro is a molecular visualisation package. It has been designed to be easily extendible with an efficient plugin architecture that has direct access to underlying functionality. A plugin is constructed for this project that receives IR tracking data from two

pre-calibrated Wiimotes, and modifies the molecular viewport accordingly using translation and rotation functions. The Avogadro package includes direct access to its internal camera widget, which plugins can use to access functions for transforming the viewport.

In the hand tracking system, two 3D points are constructed using the formulas presented in the previous chapter. The 3D points are constructed once per rendered frame of the molecular viewport, and not with every camera update. The cameras update information at 100 Hz but the rendering frame rate may be significantly lower than this. Inter-frame updates are largely irrelevant in this application and are thus discarded. If the time elapsed between each rendered frame is measured and previous 3D coordinates of the light sources are stored, accurate velocity and even acceleration vectors can be constructed to record the exact motion of both hands. Although acceleration isn't explicitly used in this project, velocity vectors of the user's hands form the basis of the interface. If the time between rendered frames is δ_t, and the 3D coordinates of a single light source in two successive rendered frames are stored as $\mathbf{x_{new}}$and $\mathbf{x_{old}}$, an example of how a single velocity vector $\frac{\delta \mathbf{x}}{\delta t}$ is constructed is shown below:

$\frac{\delta \mathbf{x}}{\delta t} = \frac{\mathbf{x_{new}} - \mathbf{x_{old}}}{\delta_t}$

The interface incorporates 6 degrees of freedom, and the hand-transform mappings are presented as follows:

Hand Movement	Viewport Transformation
Moving hands closer together	Zooming into molecule
Moving both hands up,down,left or right simultaneously	Panning horizontally and vertically
Moving left hand down and right hand up, or vice-versa	Rotating molecule along z axis (roll)
Moving left hand forward and right hand backward, or vice-versa	Rotating molecule along y axis (yaw)
Moving both hands forward or backward simultaneously	Rotating molecule along x axis (pitch)

Velocity vectors are used to construct a model of the hand movement in all of the cases shown above.

5.3 Underlying Details

The 3D point reconstruction system is developed using the theory outlined earlier. However, there are certain nuances relating specifically to this application that are considered.

5.3.1 Components

Aside from the two Wiimote cameras used for sensing IR Light, the constructed LED system is made as simple as possible in order to maximise usability, battery life and

aesthetics.

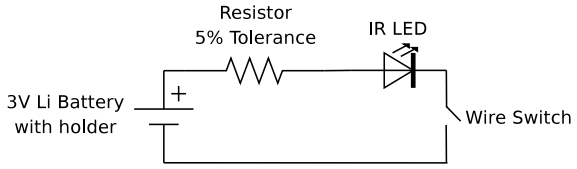

Figure 5.2: Circuit diagram of LED system.

Each circuit is attached to the tip of the index finger of a glove which can be worn by the user. The resistor limits the current and its value is chosen such that the IR LED emits light at its maximum rated current and power when switched on. The LED selected for the final design is the Siemens SFH484 IR LED[1], but several alternate light sources are considered, as described in section 5.4.2.

5.3.2 Noise and External Interference

Daylight, incandescent lighting and similar light sources can be needlessly detected by the Wiimote. The Wiimote senses these sources of noise as random IR points that may flicker and move around the camera's field of view sporadically. There are several methods for filtering noise out, and an effective noise reduction algorithm is used in this project that is described as follows.

The Wiimotes are designed to sense up to 4 sources of IR light. With this project only 2 IR LEDs are used, leaving up to 2 additional IR sources to be sensed. If there are more than 2 measured sources of light, the two most likely positions of where the IR LEDs are selected based on the last successful data capture of the Wiimote. Given the relatively slow nature of one's movements when compared to a 100 Hz camera, the new measured positions of the IR LEDs will be very close to the old values. With these nearest values being selected as the LED positions, all other IR sources are assumed to be noise and are discarded.

5.3.3 Insufficient IR Sources

It sometimes occurs that one or more LEDs are blocked from view temporarily. This could be as a result of the user rotating or moving the LEDs outside of either of the cameras' fields of view. With the absence of noise, this results in fewer than the 2 required IR sources captured by the Wiimote. In this situation, the plugin ignores the Wiimote's data capture for that frame, effectively rendering it non-existent. Once the IR LEDs come back into view again, data processing continues normally. Thus, when there are insufficient IR sources, the sensor data is ignored and the molecular viewport is not modified.

[1] http://harrier.itc.if.ua/doc/opto/irda/siemens/sfh484.pdf

5.3.4 Camera Orientation

The plugin establishes Bluetooth links with 2 Wiimotes during initialization, but initially has no information regarding where each remote is located in the system. In order for the presented algorithm to be effective, it needs to differentiate between the left camera and the right camera in space, because the left camera is mapped to the origin of the global coordinate system used. This problem is easily solved by capturing a sequence of IR light locations from both cameras. If the cameras track the same sources of light as in this project, it logically follows that the left-most Wiimote will capture light data towards its right hand side, and the right-most Wiimote will correspondingly capture IR blobs to the left of the same measured light sources in the left-most Wiimote. The leftmost Wiimote is selected as the one with the largest sum of all captured x coordinates of all measured light sources. This concept is illustrated as follows:

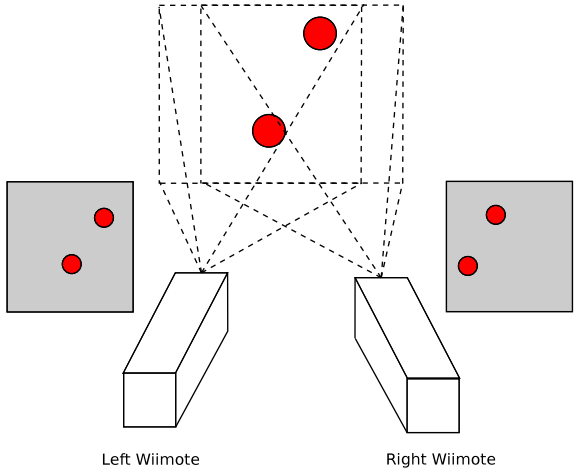

Figure 5.3: Relative IR data images for both Wiimotes.

5.4 System Testing

The Wiimote is extensively tested in this project, and accurate quantitative attributes are measured. Once sensing boundaries, accuracies and sensitivities of a single Wiimote are established, the stereoscopic hand tracking system is assembled and extensively tested. Details of all tests conducted are presented here.

5.4.1 Time Dynamics

A Wiimote is left unchanging over an extended period of time in order to establish whether the passage of time has any effect whatsoever on the measurement accuracies or stability of the system. The effect of extended usage on battery life is analysed in order to determine the system's maximum operational time between battery changes. This includes the maximum power-on time and discharge rates of both Wiimotes, and of the LED system attached to the user.

5.4.2 Alternative Light Sources

An analysis of several IR light sources is conducted, including the following:

1. Siemens SFH484 IR LED[2]
2. Siemens SFH487 IR LED[3]
3. NTE3027 IR LED[4]
4. Candle light
5. Mini Maglight® incandescent bulb

These light sources are tested for their viability for use in 3D tracking.

5.4.3 Camera Limitations

The camera reports IR blobs at a 1024x768 resolution. However, distance and angle measurement limitations during IR blob tracking are unknown. These include any minimum or maximum measurement distances for effective camera operation, and any non-linear changes in the field of view as the user moves further away from the camera. These changes may or may not be significant, and are attributed to the focal distortion of the lens. The tests are outlined below.

1. Minimum effective sensing distance from camera with several light sources.
2. Maximum sensing range for different IR lights.
3. Horizontal and vertical fields of view at measured distances from the camera.

5.4.4 Lens Distortion

Overview

Camera lens distortion is measured using a custom method the author names a pixel variance ray trace. In this technique, a different measure of lens distortion is created by analysing how the Wiimote senses an IR light source's position as it is back-projected along a ray extending from the camera's location outwards. For an ideal IR camera with no distortion, the measured pixel value should not change as the light source moves along the ray. A graph is plotted showing the variation from the mean pixel value with distance. It is appropriate to use this kind of measure in this application, because it is necessary

[2]http://harrier.itc.if.ua/doc/opto/irda/siemens/sfh484.pdf
[3]http://harrier.itc.if.ua/doc/opto/irda/siemens/sfh487.pdf
[4]http://www.mantech.co.za/Datasheets/Products/NTE3027.pdf

to analyse how one's hand is sensed as it moves back and forth directly in line with one Wiimote.

In order to implement this method, a string is attached to the bottom base of the camera, essentially forming the camera's origin. The other end of the string is attached at an arbitrary angle and distance away from the camera and is pulled taut so as to implement a "ray" from the camera's origin outwards. An IR LED is connected next to the string, and moved from the base of the camera outwards to the opposite end of the string. All pixel coordinates of the IR LED as measured by the Wiimote are stored as it moves along the string outwards.

The pixel variance at a distance d from the camera's origin is the Euclidean pixel distance between the average pixel coordinate for the entire series, and the camera's coordinate sensed at the given distance. The exact direction of the ray is not needed because it is assumed that distortion is constant along all rays originating from the camera. However, the ray direction can easily be parametrized by its horizontal and vertical angles to a constant reference ray if needed.

Theory

A set of n measured pixel coordinates are stored as follows:

$p_0(x,y,d) \dots p_n(x,y,d)$

where x and y are the camera's measured coordinates of the IR Light at a distance d from the origin. For every distance value d, there is an associated pixel coordinate which is referenced as $p_d(x,y)$ for convenience. The mean value is calculated as the mathematical average of all pixel coordinates:

$\bar{p} = \sum \frac{p_i}{n}$

The pixel variance $V(d)$ at a distance d determined the distance between the measured pixel at d and the mean pixel value $\bar{p}$.

$V(d) = \sqrt{(p_{dx} - \overline{p_x})^2 + (p_{dy} - \overline{p_y})^2}$

The 1D pixel variance function is then plotted as a function of distance. If details concerning the ray's exact parameters are required, the pixel variance function can be given as$V(d,\alpha,\beta)$ where α and β are horizontal and vertical deviation angles from a constant reference ray, respectively. The reference ray can be selected as the ray parallel to the camera, extending along the z axis. However, it is assumed in this project that lens distortion is approximately uniform, independent of the selected ray parameters α and β.

5.4.5 Camera sensitivity

Tests are performed with respect to the measuring accuracies of the Wiimote camera. These tests include:

1. Camera pixels changed per centimetre that the subject moves horizontally at a distance from the camera;

2. Change in IR blob size with distance for various light sources;

3. Sensitivity to noise from other sources of IR light, such as natural daylight and indoor lighting.

5.4.6 Algorithm Efficiency

The 3D point reconstruction algorithm used for the hand tracking interface will be analysed for efficiency in terms of its execution time and big-O notation. This provides insight into possible future implementations for embedded applications.

5.4.7 Accuracy of 3D Algorithm

In order to test the accuracy of the presented algorithm, the best approach is to compare the calculated 3D coordinates of IR light sources to their actual 3D positions in space. A measuring tape is used to measure the (x, y, z) coordinates of arbitrary locations of IR light sources in the cameras' field of view. These measured values are then compared with the coordinates calculated by the algorithm.

Chapter 6

Results

6.1 Camera Details

6.1.1 Camera Range

The horizontal and vertical sensing ranges of the camera were measured at various distances from the camera. These ranges are attributed to the camera's field of view at those distances. An ideal camera has no distortion of its field of view. The Wiimote's camera range plot shows an almost linear curve, as shown below:

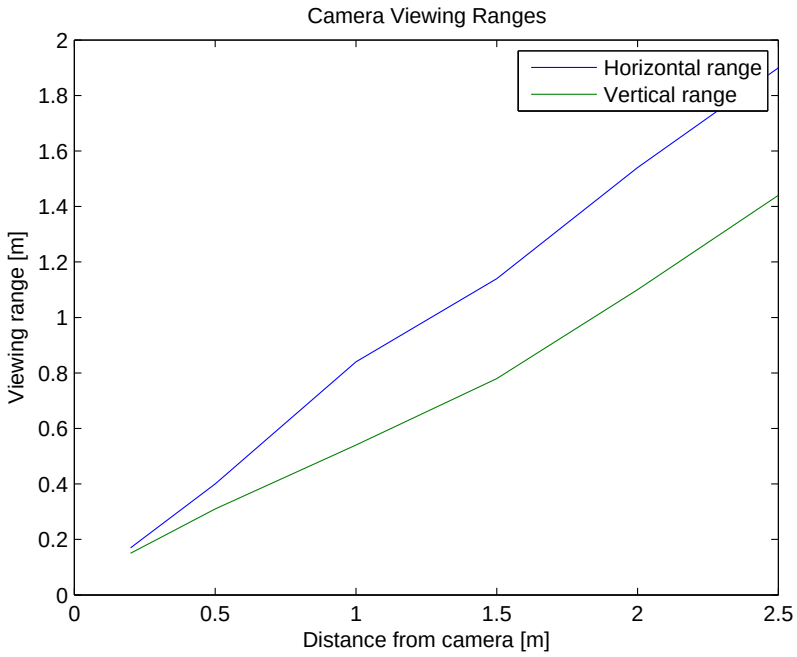

Figure 6.1: Viewing ranges at given distances from the Wiimote.

With the above information, an average field of view is calculated. From this data, the Wiimote's average field of view is calculated as 41° horizontally and 31° vertically.

It is established that any light sources closer than 10 cm to the camera's lens are not accurately tracked. This is most likely due to blurring of the IR light data stored in the

CCD, resulting in difficulties in resolving the central pixel location of the light.

The camera's maximum sensing range depends largely on the size, luminescence, and output direction of the IR light source. Experiments were conducted with several light sources, and incandescent bulbs in torches are accurately sensed up to 10 m away. IR LEDs, however, are not reliably sensed further than 3 m. IR LEDs have significantly less power output per steradian than a focused incandescent light beam has, in addition to a smaller dimension of the active illuminated area. IR LEDs in general have very small output half-angles, typically ranging from ±18° to ±30°. Essentially, if the LEDs are rotated more than their half-angle value, the camera is ineffective at sensing the emitted IR light.

6.1.2 Camera Sensitivity

The sensitivity of the camera was determined, with respect to the number of unique pixels measured per centimetre moved horizontally at a given distance from the camera. These values were measured in 10 cm increments perpendicular to the camera, and subsequently averaged. The results are summarised in the following figure:

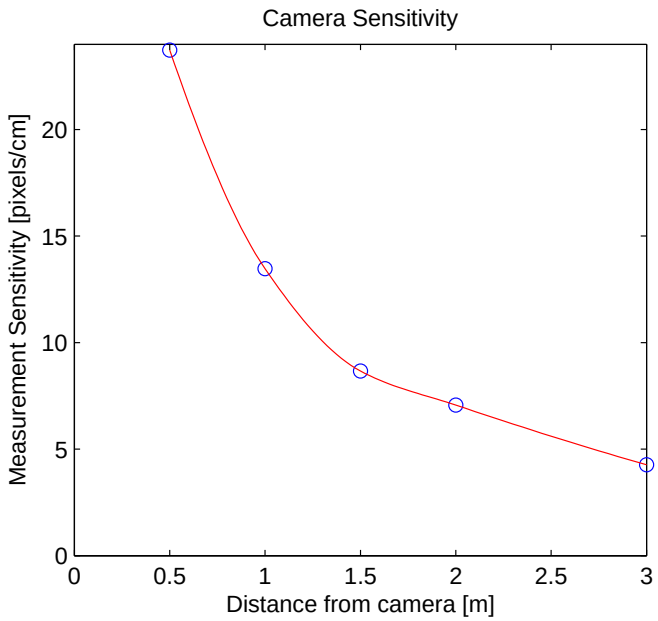

Figure 6.2: Meauring sensitivity at distances from Wiimote.

The camera sensitivity function can be modeled as a hyperbolic curve of the nature $xy = k$ where x is the distance from the camera, in metres, and y is the number of unique pixels per centimetre. Calculation of the mean k value for the above graph results in a sensitivity function of $xy = 13.0533$.

6.1.3 Blob Sizes

Several experiments were conducted to ascertain the variation of measured blob size with distance. Unfortunately, in all experiments the readings proved to be widely varying and sporadic. The measured size of an IR blob is highly dependant of the light sources' angles of inclination to the camera.

Incandescent bulbs and candles proved to have the most sporadic size readings, randomly fluctuating between 1 and 4 pixels at any given distance. IR LEDs had more stable readings, but no results were scientifically usable. IR LED blob sizes typically ranged between 1 and 2 pixels at distances greater than 1 m, and fluctuated greatly at closer distances.

The only way in which blob sizes could be reliably used for any engineering application is to construct an array of LEDs connected next to each other, and mounted in such a way that they constantly face the camera directly. Using this technique, it is possible to construct a crude algorithm for approximating the light source's distance to the camera. More information is provided in the final chapter regarding future work.

6.1.4 Pixel Variance Ray Trace

The pixel variance ray trace, as described in the previous chapter in some detail, shows the variances between measured pixel coordinates and the mean pixel coordinate as a single light source moves along a ray from the camera outwards.

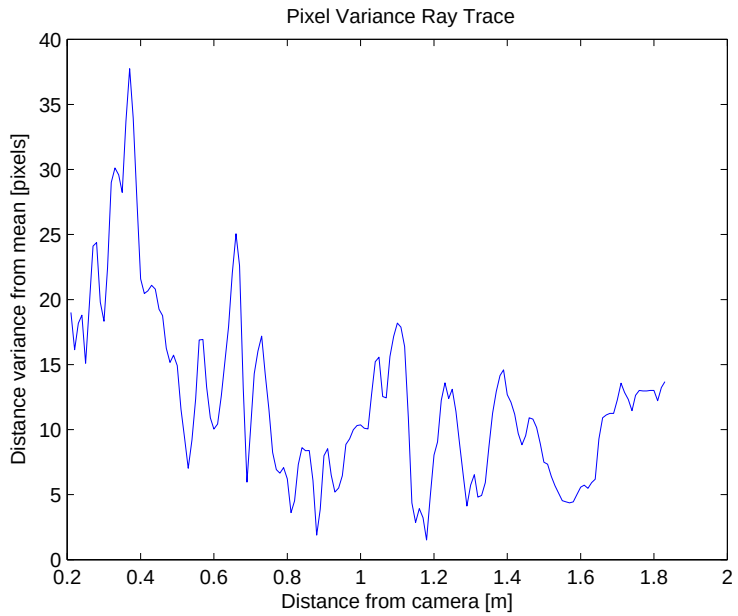

Figure 6.3: Pixe variance ray trace function.

All of the coordinates captured along the ray can be shown as a single scatter plot with no distance information:

Essentially what these graphs indicate is that there is some distortion in measurements along a ray from the camera. The distortion decreases somewhat further away from the

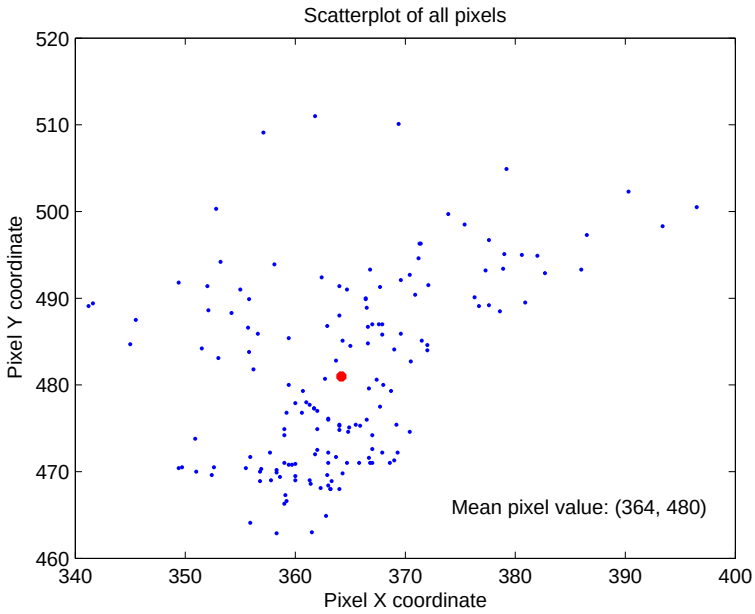

Figure 6.4: Scatterplot of pixel coordinates.

camera, due to lower sensitivity values at these distances. In general, the pixel variance is smaller than 15 pixels from the mean along the ray. This is certainly an acceptable figure for the given application, resulting in relative distortions of less than 1.46% along the x axis with 1024 pixels, and less than 1.95% along the y axis with 768 pixels with a distortion of 15 pixels.

6.1.5 Time Dynamics

A single Wiimote was connected to a PC and allowed to continuously capture IR blob sources. Over a 1 hour period, the batteries flattened by 8% as measured by the Wiimote's battery status report. This translates to approximately 12.5 hours of usage time with a new pair of AA batteries. During this time period, there was no fluctuation or change in the location readings of the IR light sources, indicating that the system is time invariant.

6.2 Analysis of 3D Algorithm and System

6.2.1 Accuracy

The 3D tracking algorithm was tested by measuring coordinates of several arbitrary locations from the camera system's origin, and comparing it to calculated values. The results are presented in the table below, and all values are given in cm:

Actual Location	Calculated Location	Absolute Error	Distance Error
(15,3,100)	(15.7,2.4,83.4)	(0.7,0.6,16.6)	16.62
(0,10,80)	(0.1,7.4,67.3)	(0.1,2.6,12.7)	12.96
(10,3,50)	(7.65,4.09,49.9)	(2.35,1.09,0.1)	2.59
(-25,20,140)	(-21.4,16.5,117)	(3.6,3.5,23)	23.54

The x and y coordinate values are adequately accurate for all of the locations. However, there is significant error in the z coordinate at larger distances from the camera system. The nature of the algorithm requires the Wiimotes to be exactly parallel in order to accurately sense the z coordinate. The Wiimotes were placed as parallel as practically possible, but it was discovered that a rotation of one Wiimote by a mere 5° created errors in the z coordinate calculation of up to 40 cm at 2 metres away.

6.2.2 Execution Time

The 3D tracking algorithm's execution time was measured over 100000 iterations and averaged. The stereoscopic resolution code executed in an average of 4.5 μs on an AMD 2.4 GHz 3800+ CPU. This results in approximately 10800 instructions, mainly due to the usage of an external vector and matrix algebra library, namely Eigen. This value can be reduced with the implementation of custom algebra routines optimized for this application. The execution time, however, is almost negligible when compared to the 10 ms refresh time of the Wiimote's camera, at 100 Hz.

The algorithm's execution time remains constant irrespective of the position of an IR light source, and thus has a Big-O rating of O(1).

6.2.3 IR Light System

The LED system was designed with priorities given to simplicity and size metrics. As such, the usage time of the LED system is significantly less than the camera system's usage time. The 3V CR2032 Lithium coin cell used to power the LEDs provides 229mAH of current usage. The high power SFH484 IR LED chosen for the final model is driven with 100 mA of continuous forward current, resulting in an operation time of slightly over 2 hours. It is important to note that a slightly larger Lithium cell can be purchased which provides significantly more energy. An alternative is the CR2450 Lithium coin cell, providing 600 mAH of current usage.

Incandescent lamps and candles have a further sensing range from the camera due to their higher power output. Due to the flickering of candles, the IR camera randomly loses the IR signal. The incandescent lamp provides a good IR tracking source, but due to its concentratated reflector design, should point to the camera system for full effect. Incandescent lamps and torches are also larger than necessary for engineering applications, where is it most effective to use an IR LED.

The SFH484 and SFH487 LEDs both have the same current rating and power output, but the SFS484 provides a higher half-angle output than its counterpart, and was selected for that reason. The NTE3027 IR LED was also tested, and because of its extremely high current sink of 150 mA and forward voltage drop of 1.7 V, its power usage is undesirably high for use in this application, despite its favourable half-angle output beam.

The system works fairly accurately and allows easy visualisation of a molecule at any angle and position with minimal effort. However, a problem experienced with the light system is that the half angles of most IR LEDs are fairly small. This results in a user needing to concentrate on pointing the IR LEDs directly at the camera system at all times, which can be fairly frustrating. As one moves their hands around to effectively transform the molecule in the viewport, it may be difficult at times making sure that both light sources are pointed at the camera system.

6.2.4 Freedom of Movement

An important factor considered in this project's application is the exact amount of freedom a user has to move around. It was measured that the selected IR LEDs are not reliably sensed further than 2 m away from the camera. With two parallel Wiimotes of a known distance apart, this allows one to construct a freedom of movement equation which gives the maximum leeway available to a user in the horizontal and vertical directions. The freedom of movement for the horizontal plane at 2 m away is illustrated below:

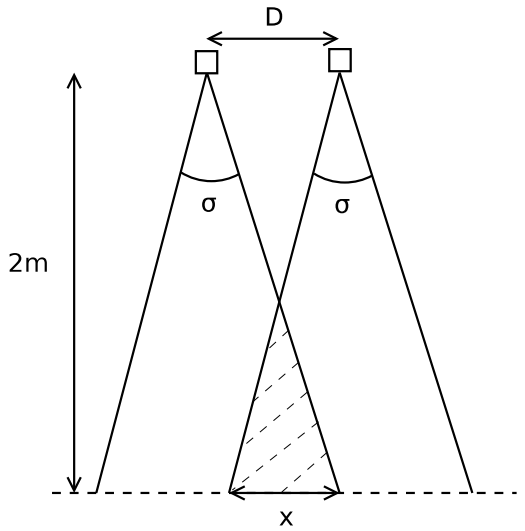

Figure 6.5: Diagram showing freedom of movement.

The result is summarised with the following equation, where all variables are given in metres:

$2\tan(\frac{\sigma}{2}).2 - D = x$

The result is that two Wiimotes 25 cm apart allows the user's hands 124 cm of movement in the horizontal direction, at 2 m away from the camera system. If the horizontal field of

view angle σ is replaced with the vertical field of view angle θ, it is calculated that a user has 86 cm of space to move in the vertical direction.

6.2.5 Cost

The system proved to be remarkably cost efficient for tracking 3D positions of objects in real time. The cost breakdown of the core system components are given below:

Component	Cost
Wiimote x2	R1300
Bluetooth dongle	R250
SFH484 IR LED system	R25
Total	R1575

The cost of the PC used to execute the code was not taken into account, as any Bluetooth-linked processor can be used to execute the 3D tracking code at varying speeds of computation. However, a basic PC system costing approximately R2000 can be used for this setup, bringing the total system cost to R3575.

Chapter 7

Conclusions

Several tracking design concepts were presented, and the most appropriate design was selected and implemented as the basis for this project. The system proved to have numerous advantages over traditional interfaces as well as proprietary 3D tracking systems, essentially combining the best of both worlds to create a novel, intuitive and easy way to interface with virtually any sort of CAD-like environment on the PC.

7.1 Fast 3D Tracking with Low Cost Hardware

The Wii Remote camera is a remarkably cheap and accurate solution for sensing and tracking objects emitting infrared light. With the camera update frequency of 100 Hz, the system can successfully track most slower moving objects, such as a person's hands. A set of two or more Wiimotes can be used to construct a true 3D coordinate of an IR light source from its pixel positions sensed by the camera system. More Wiimotes can be added into the system easily, linearly enhancing the calculation accuracy. The 3D tracking algorithm uses vector geometry, and executes very quickly since some techniques relating to normal stereoscopic algorithms can be bypassed, due to the key pixel locations being already extracted from the image.

The 3D tracking algorithm as presented is not accurate enough to track the exact 3D positions of light sources due to the difficulty in practically calibrating the system manually. However, this is not the aim of this application because the relative positions of a user's hands are successfully tracked and implemented into a molecular visualisation interface. An automated calibration solution is given in the following chapter which counteracts the accuracy problem, allowing the system to perform accurate, fast and cost effective 3D tracking that rivals similar systems that are several times more expensive.

The approximate system price of R3575 is a very cost effective and easily implementable solution to what is a newly emerging and financially expensive field of research.

7.2 Adequacy of Light System

The SFH484 LED system constructed is adequate for the given application. Despite the relatively low half-angle power output of this light source, the LEDs are almost always pointed towards the camera system, allowing for continuous sensing of all light sources. The LEDs are attached to the end of the user's index fingers, which naturally point forward towards the camera lenses when interfacing with the computer.

There are IR LEDs available with significantly greater half angles who's luminescence is captured by the camera system at more extreme angles of rotation, resulting in greater flexibility. One such IR LED is the KM-4457F3C Side Look LED[1], featuring 3 mW/steradian output power with a 150° viewing angle. This would be ideal for an object that rotates sufficiently far for the camera to have problems sensing normal IR LEDs.

7.3 Improvement over Traditional Interfaces

The traditional interfaces for molecular visualisation use a combination of the keyboard and mouse to facilitate molecular transformations. These systems typically do not support multiple simultaneous transformations, resulting in difficulties in moving around and zooming into complex molecules efficiently and quickly. In Avogadro, the mouse can be used in conjunction with its buttons to implement all 6 degrees of freedom for transforming a molecule in the viewport. However, only one transformation can be performed at a time, resulting in slower, bulkier HCI interaction.

With the Wiimote plugin in use, one can use both of their hands to independently translate, rotate and scale the molecule. These transformations can be performed at the same time, and are mentally intuitive to the application. This results in virtually no time taken to learn the novel interface, and much faster and less frustrating navigation around the molecule. This results in a superior interface design to traditional interfaces. The only real disadvantage to this system is the cost of purchasing new components and the time taken to implement and configure the system. These cost factors are performed once only, and are trivial when compared to the budgets of larger institutions and corporations. The 3D tracking using the presented algorithm is not ideal for all engineering tracking purposes, but it is very adequate for the given hand tracking application.

7.4 Viability in Other Applications

The concept of 3D tracking has widespread application possibilities. Using the techniques presented in this thesis, it is possible to track any object with an attached IR light source, provided that the object remains within the fields of view of at least two cameras in the

[1] http://www.datasheetcatalog.org/datasheets2/82/82962_1.pdf

system at all times. With additional cameras being added to the system, a more accurate 3D re-composition is possible.

It is possible to execute the algorithm on a Bluetooth enabled microprocessor for 3D tracking in embedded applications. The possibilities here are virtually endless, ranging from robotic vision and collision detection, to environment mapping and 3D shape reconstruction. Several changes and enhancements would need to be made to enable such functionality, but it is certainly possible. The extensible nature of the system and tracking algorithm provides an important stepping stone to future applications utilizing this technology. Given the 10 ms refresh times of the cameras, even a modest processing time of 1 ms for an algorithm constituting 10800 instructions results in a minimum required processor speed of 10.8 MHz.

The hand tracking concept need not only apply to molecular visualisation, since any CAD related work can be enhanced by using hand tracking. This would aid in the navigation and construction of complex 3D objects in fields ranging from 3D graphical design, architecture and even gaming.

A novel operating system interface can be created by extending the concept to involve full finger tracking. However, the restriction of 4 tracked IR hot spots by the Wiimote does impede the possibilities here. It is important to note that a typical CCD with an IR filter and hot spot detection code can effectively mimic the Wiimote's functionality without the restriction of only 4 IR light sources. With a full finger tracking interface, each finger will have its own unique degree of freedom, allowing for a virtual keyboard interface to the PC. If used in conjunction with a reference picture of a keyboard, it is possible to move one's fingers over the reference picture's keys in order to detect which keys have been "pressed" by sensing the positions of the IR sources located on each finger.

Chapter 8

Future Work

8.1 Automatic Camera Calibration

8.1.1 Research

Zhengyou Zhang has presented a method that involves presenting a checkered pattern such as a chessboard in several orientations to both cameras[7]. The technique only requires the camera to observe a planar pattern from a few (at least two) different orientations, and correspondingly generates the intrinsic system matrices. The algorithm uses maximum likelihood estimation to generate the most accurate system parameters, both intrinsic and extrinsic, in a closed form solution.[8] The algorithm takes into account radial lens distortion in order to accurately resolve 3D objects. The application of this algorithm for the Wiimote has been implemented by the Digital Technology Group at Cambridge University[2]. This method presents a board with 4 IR LEDs at each corner to both Wiimotes, and the Camera Calibration Toolbox for Matlab[1] is used to construct 3D coordinates.

8.1.2 Proposed Algorithm

An automatic camera calibration algorithm is proposed that automatically resolves the extrinsic parameters of the camera system, given *a priori* knowledge of the Wiimote's intrinsic horizontal and vertical fields of view.

The dual Wiimote system is presented in figure 8.1. If an arbitrary Wiimote is chosen as the reference point, at $(0,0,0,0)$, then the second Wiimote's location and orientation is parametrized as $(x,0,z,\alpha)$. Only 3 variables are required, due to the assumption that both Wiimotes lie on the same horizontal plane (such as a table top), and the only rotation is the yaw of the Wiimotes along their y-axes. Maximum possible values x_{max} and z_{max} are pre-selected for x and z, given the nature of the application. If it is certain that the

[1] http://www.vision.caltech.edu/bouguetj/calib_doc/index.html

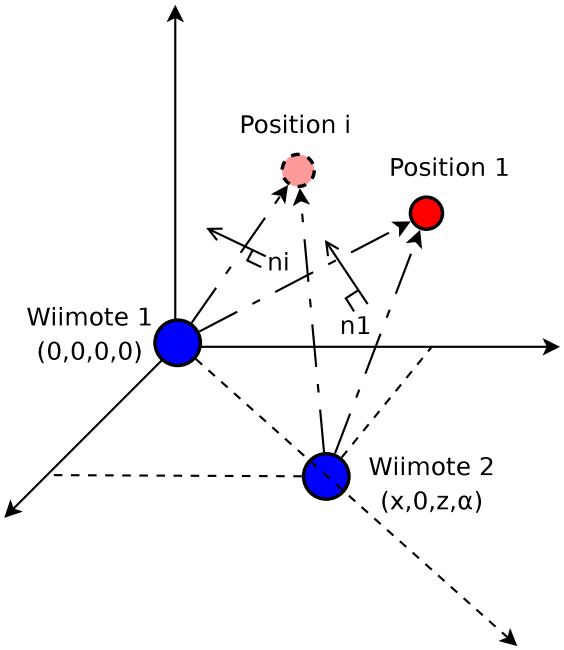

Figure 8.1: Automatic calibration using several generated planes.

Wiimotes are placed within 2 metres of each other, x_{max} and $z_{max}$are assigned the values of 200 cm. Since α is an angle, its maximum value of α_{max} is 360°.

A single LED light source is added to the system and tracked by both Wiimotes simultaneously. Each recorded pixel value is stored and resolved into a 3D ray originating from the Wiimote's reference point, as explained in previous chapters. The LED is allowed to move around, with several different rays being calculated for each set of pixel values over time.

It is described in section 4.2 that the resolved 3D coordinates along each ray, parametrized by $\hat{t}_1$and $\hat{t}_2$, give the points along the rays closest to each other. Given these two points, the distance between them can be used as a gauge for the error in resolving the 3D position. A 3 dimensional binary search[1] is thus conducted for x,z and α that rapidly converges to the best-fitting values of these variables. Every stored ray pair from the Wiimotes is used to create two 3D coordinates using the algorithm described previously, with the variables of x,z and α in the current loop. The distance error between the two points is used to determine the accuracy of the variables within the current loop. This distance error is calculated, squared and aggregated over all stored rays, giving an absolute error value ε for the currently selected values of x,z and α. The binary search continues until the error becomes less than a predetermined amount ε_{min}, whereby the search stops and the values of x,z and α corresponding to the smallest distance error are selected and used in the Wiimote system for future 3D tracking.

Although this algorithm is computationally intensive, it is performed during system initialization only, and need not be used again until a system parameter changes. For example, if $x_{max} = 200$ and the algorithm is assumed to stop when the values are within 1 cm of their real values, the required number of binary iterations for each variable is shown:

$\frac{\log \frac{1}{200}}{\log \frac{1}{2}} \approx 8$

With 3 variables being analysed, 8^3 iterations are required to converge all values accordingly. If the LED is allowed to be tracked for 5 seconds at 100 Hz, 500 pairs of rays will be calculated. Given 8^3 iterations for each pair of rays and an execution time of 4.5 μs for each iteration as timed in this project, the processing time in this algorithm is approximately:

$t = 8^3 . \frac{4.5}{10^{-6}} .500 = 1.15$

The algorithm executes in 1.15 seconds, automatically and accurately resolving the extrinsic parameters $(x, 0, z, \alpha)$ of this system. With additional degrees of freedom in the translation and rotation of the Wiimotes, a higher level binary search can be performed, but greatly degrades algorithmic performance.

8.2 Tracking Multiple IR Points

The Wiimote is capable of tracking up to 4 IR point sources of light. This project tracks two points in 3D with a simple algorithm for differentiating between the 2 light sources as discussed. The algorithm would need to be extended if the light sources were allowed to cross over each other in the cameras' viewports, or if more point sources were to be added to the system.

Once the camera calibration is complete and a single point can be tracked in 3D, up to 4 points can be tracked as long as the system keeps track of which pixel points are associated with which IR LEDs in space. Several methods are presented to keep track of this information.

Distance Differentials

The IR camera creates pixel frame reports at 100 Hz. Under the assumption that the motion of the LEDs is acceptably slow, such as in hand movements, each LED can be tracked using the distance it moved from the camera's previous frame. Each LED's corresponding pixel position will be very near to its position in the previous frame. If each pixel is connected to its nearest neighbour in the previous frame, the mapping is accurate.

As an example, if a pair of IR point light sources are in motion 1 metre away from the camera system, the measuring accuracy at this range is 13 px/cm, as presented in the results. If the LEDs are guaranteed to remain further than 1 cm away from each other, this corresponds to a minimum range of 13 pixels between the LEDs. The LEDs would need to be moving faster than half of this distance between the frames, that is $\frac{13}{2}$ px/frame, in order for the algorithm to map to the wrong point sources, as shown in the diagram.

It is important to note that at the speed of $\frac{13}{2}$ px/frame, the LED's actual movement speed is:

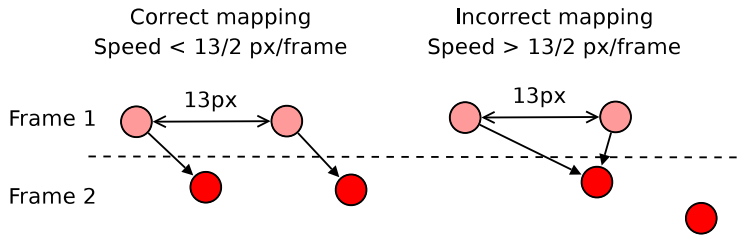

Figure 8.2: Distance differential mapping.

$$\frac{13}{2}\frac{pixels}{frame}\cdot\frac{100}{1}\frac{frames}{sec}\cdot\frac{1}{13}\frac{cm}{pixel}=50\frac{cm}{sec}$$

Thus for the algorithm to remain effective, the LEDs located 1m away and a minimum of 1 cm apart need to travel faster than 50 cm/sec. With a higher minimum distance between the LEDs, the maximum movement speed increases linearly.

Light Size Differentials

The Wiimote is capable of measuring the relative sizes of IR light sources. An approach can be considered where the size of the light sources are used to differentiate between the various measured points. Sets of LEDs with different sizes may be used which will in turn register different IR blob size readings by the camera. Alternatively, a tightly packed array of several LEDs can be used as this will be sensed as one large IR blob by the camera.

In this approach, the distance of all light sources from the camera need to remain fairly constant. This is because the further away the light source moves, the less accurate the measurement of size is which can result in algorithm inaccuracies.

LED Light Modulation

In an LED system, each LED can be selectively modulated at a unique frequency or pulse width. If a sufficient number of sequential pixel frames captured by each Wiimote are saved and analysed, it is possible to identify each IR point with its corresponding LED. An algorithm may examine each successive frame and examine the modulation frequencies of each sensed light source, and match them to the actual modulation frequencies of the corresponding IR LEDs.

This technique is augmented with the distance differential algorithm for a more robust tracking method that is less prone to errors.

Appendix A

Bluetooth Communications

Bluetooth is a short range wireless communications technology that utilises frequencies in the 2.4 GHz radio frequency (RF) band. Class 2 Bluetooth devices, such as the Wiimote's internal Broadcom 2042 chip, emit a maximum power of 2.5 mW (4 dBm). This translates to an approximate maximum range of 10 metres. The Nintendo Wii typically acts as the host for Bluetooth communications from the Wiimote, but in this project a Bluetooth enabled PC is used as the host. In this project, Bluetooth packets sent to and received from the controller are processed by the BlueZ Bluetooth library in Linux.

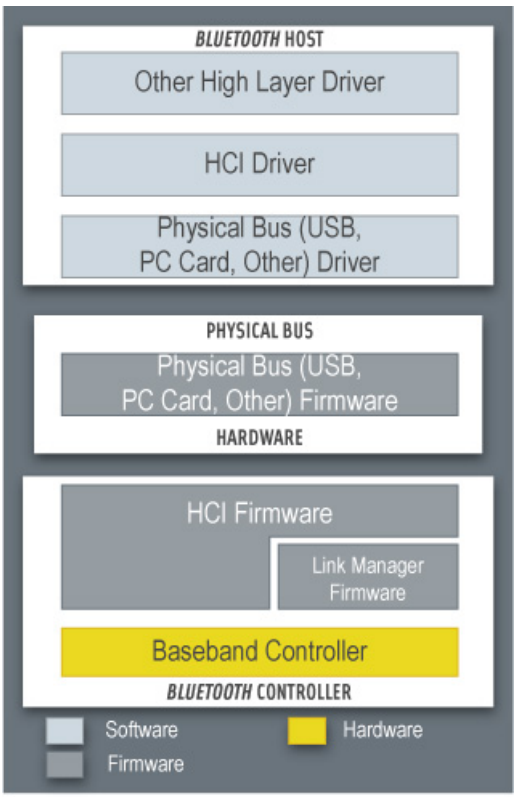

Figure A.1: Bluetooth layer stack.

The figure provides an overview of the lower software layers. The HCI firmware implements the HCI Commands for the Bluetooth hardware by accessing baseband commands, link manager commands, hardware status registers, control registers and event registers.

Several layers may exist between the HCI driver on the host system and the HCI firmware in the Bluetooth hardware. These intermediate layers, the Host Controller Transport Layer, provide the ability to transfer data without intimate knowledge of the data.

The HCI driver on the Host exchanges data and commands with the HCI firmware on the Bluetooth hardware. The Host Control Transport Layer (i.e. physical bus) driver provides both HCI layers with the ability to exchange information with each other.

The Host will receive asynchronous notifications of HCI events independent of which Host Controller Transport Layer is used. HCI events are used for notifying the Host when something occurs. When the Host discovers that an event has occurred it will then parse the received event packet to determine which event occurred.[1]

The Bluetooth HID profile defines the protocols, procedures and features to be used by Bluetooth HID such as keyboards, pointing devices, gaming devices and remote monitoring devices.The HID defines two roles, that of a Human Interface Device (HID) and a Host:

- Human Interface Device (HID) – The device providing the service of human data input and output to and from the host.
- Host – The device using or requesting the services of a Human Interface Device.

The HID profile uses the universal serial bus (USB) definition of a HID device in order to leverage the existing class drivers for USB HID devices. The HID profile describes how to use the USB HID protocol to discover a HID class device's feature set and how a Bluetooth enabled device can support HID services using the L2CAP layer. The HID profile is designed to enable initialization and control self-describing devices as well as provide a low latency link with low power requirements.[2] The outputs of the HID descriptor block facilitating full duplex communication in the Wiimote is presented in section 2.2.2. This is used to differentiate between the types of data that can be read from and written to the Wiimote, contained in the sent and received Bluetooth packets.

[1] http://www.bluetooth.com

[2] http://www.bluetooth.com/Bluetooth/Technology/Works/HID.htm

Appendix B

CD Contents

A CD is accompanied with this thesis, containing the following data:

1. **readme.txt** - provides a guide to installing Avogadro, cwiid and the Wiimote plugin on a Linux PC, as well as information regarding system setup.
2. **cwiid-0.6.00.tar.gz** - the latest version of the cwiid library at the time of writing.
3. **avogadro-0.8.1-wii.tar.gz** - the latest version of the Avogadro molecular editing software, with the Wiimote plugin included to automatically compile and install with the package.
4. **wrnmic002_ug_thesis.tar.gz** - Lyx files, images and references used in the creation of this document.
5. **wrnmic002_ug_thesis.pdf** - electronic version of this document.

Bibliography

[1] R. O. Eyal Kushilevitz and Y. Rabani. Efficient search for approximate nearest neighbor in high dimensional spaces. Technical report, Annual ACM Symposium on Theory of Computing, 1998.

[2] S. H. Joseph Newman, Robert Harle. Optical tracking using commodity hardware. Technical report, Digital Technology Group, Computer Laboratory, University of Cambridge, 2008.

[3] S. Rathi. Blue tooth protocol architecture. Technical report, Microwave Systems, 1999.

[4] C. Sun. A fast stereo matching method. Technical report, CSIRO Mathematical and Information Science, 1997.

[5] C. Sun. Uncalibrated three-view image rectification. Technical report, CSIRO Mathematical and Information Science, 2003.

[6] R. Y. Tsai. A versatile camera calibration techniaue for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. Technical report, IEEE Journal of Robotics and Automation, 1997.

[7] Z. Zhang. A flexible new technique for camera calibration. Technical report, Microsoft Corporation, 1998.

[8] Z. Zhang. Flexible camera calibration by viewing a plane from unknown orientations. Technical report, Microsoft Corporation, 1999.