Wiimote 3D Interaction in VR Assisted Assembly Simulation

MIKAEL RINGMAN OLSSON

KTH Computer Science and Communication

Master of Science Thesis
Stockholm, Sweden 2011
Wiimote 3D Interaction in VR Assisted Assembly Simulation

MIKAEL RINGMAN OLSSON

Master’s Thesis in Computer Science (30 ECTS credits) at the School of Computer Science and Engineering, Royal Institute of Technology, year 2011.
Supervisor at CSC was Lars Kjelldahl.
Examiner was Lars Kjelldahl.

TRITA-CSC-E 2011:072
ISRN-KTH/CSC/E--11/072--SE
ISSN-1653-5715

Royal Institute of Technology
School of Computer Science and Communication

KTH CSC
SE-100 44 Stockholm, Sweden

URL: www.kth.se/csc
Wiimote 3D Interaction in VR Assisted Assembly Simulation

The report describes the design and implementation of a 3D interaction solution for the Nintendo Wii Remote (Wiimote). Its intended use is as a part of an educational Virtual Reality software environment with a stereoscopic display system. During the preparations for the Virtual Reality Lab (VRLab) project installation at ASU-GARDS of Ain Shams University in Cairo, Egypt, it was concluded that further 3D interaction options should be considered. The required interaction tasks mostly involved basic 3D positioning, rotation, selection and manipulation, but a few specific requirements of the project's Mechanical Assembly Assessment application were also noted. Usability and cost were important considerations, which made the Wiimote an interesting prospect given its simple design and potential user familiarity. The use of the device as a direct 3D input device for the VRLab installation is evaluated and presented with a proof-of-concept implementation and a set of 3D interaction design conclusions. The solution was shown to be useable under certain conditions when tested on site at Ain Shams University.

Wiimote som 3D-interaktionslösning för monteringsimulering i VR

# Table of Contents

1 Introduction.......................................................................................................................................... 1

2 Background...........................................................................................................................................3  
   2.1 VR and 3D Interaction............................................................................................................3  
   2.2 VR and Education.....................................................................................................................4  
   2.3 Nintendo Wii...............................................................................................................................6  
   Previous Work..............................................................................................................................7  
   Considerations...............................................................................................................................7

3 Design and Implementation..................................................................................................................11  
   3.1 Premises and Assumptions...............................................................................................11  
   The Mechanical Assembly Assessment Application.............................................................11  
   The Wiimote.................................................................................................................................12  
   The Work Area..............................................................................................................................14  
   3.2 The 3D Interface.....................................................................................................................15  
   Tasks........................................................................................................................................15  
   Preliminary 3D Interaction Model..............................................................................................17  
   Testing and Feedback.....................................................................................................................19

4 Wiimote 3D Interaction Evaluation.................................................................................................22  
   4.1 Interaction Issues..................................................................................................................22  
   4.2 Wiimote Issues........................................................................................................................26  
   4.3 VRLab Issues............................................................................................................................28  
   4.4 Reliability..................................................................................................................................30

5 Technical Details...................................................................................................................................32  
   5.1 WorldViz Vizard......................................................................................................................32  
   5.2 Implementation Details......................................................................................................33  
   Distance and the IR Camera........................................................................................................33  
   Depth and Calibration..................................................................................................................34  
   Position and the Display Screen.................................................................................................37  
   Filtering and Adjustments............................................................................................................38

6 Conclusions...........................................................................................................................................41  
   6.1 The Wiimote and VRLab.....................................................................................................41  
   6.2 VR and 3D Interaction..........................................................................................................41

References..................................................................................................................................................44
1 Introduction

Virtual Reality (VR) as a term is typically associated with cutting edge technology and advanced 3D computer applications. The ability to employ such solutions has technically been in place for a couple of decades, but the concept itself is ancient by industry standards [1]. Its practical uses have largely remained limited [2][3]. One cause for this is the design overhead and costs associated with deploying even simple VR solutions, especially if greater levels of immersion are desired – immersion here describing the amount of realistic and intuitive connection between system and user. Such obstacles are being lessened however; as the associated 3D equipment costs are reduced and application concepts mature, we are coming closer to the threshold where VR equipment and its use may become as commonplace as was once envisioned. Although this breakthrough would likely not occur in a way many would have imagined it even 10 or 15 years ago. As more mundane manifestations of the technology becomes commonplace, rather than some expected future of Head Mounted Displays and Force Gloves for everyone, we are starting to see what general use of VR will actually look like. A concrete example of this increased contemporary use would be the introduction of 3D display applications and devices, in the form of cinematic productions and venues, or game systems such as the Nintendo 3DS [4].

As for the more classical and interactive VR solutions mentioned earlier, improved performance and complexity of virtual environments is now also possible thanks to hardware advances, technology that is currently available for most ranges of home consumer computer solutions [5]. This has allowed the common use of applications with graphical 3D environments, although mostly in the form of games and other entertainment software. However such 3D applications do not necessarily offer VR experiences, the difference between them basically being a matter of interaction and display equipment. Traditional interaction devices like a mouse and keyboard, or TV console controllers – all typically associated with 2D traditional desktop and console applications – still constitute the main means of interacting with these 3D applications [6][7], and 2D displays are still the standard in most setups. Some type of 3D interaction and visualization equipment is thus needed for the level of immersion that VR is expected to provide. There are reasons to suspect that the preference for 2D-centric interaction hardware is largely a matter of legacy and inertia, that properly designed 3D devices would often be preferred and used for many 3D applications if
they were somehow made available.

Interaction devices developed for regular 3D environments could perhaps still be of use for VR installations though, in the sense that they may provide some measure of 3D interaction and immersion qualities, while also offering the advantage of reduced overhead costs as well as user familiarity. This report will consider the use of consumer level hand held devices for basic 3D and VR interaction, for the specific use of an educational exercise application as well as 3D applications in general.
2 Background

Virtual Reality can be thought of as the simulation of reality through the use of 3D computer graphics hardware and software, in a way that offers some sense of spatial awareness and immersion [8]. A traditional 3D application – in the sense that is does not offer VR – will try to work around the limitations of classical human-computer interaction through indirect interaction and perception mapping via abstraction and suspense of disbelief. VR on the other hand is about attempting to bridge part of that "suspension" by providing immersive information to the user in a way that mimics real world senses and perception [8]. Commonly this is achieved by providing an impression of depth through visual information beyond that of a classical 2D display, but the concept can be taken much further by also affecting the senses of hearing, touch or even smell [8][9]. Of equal importance is the provision of 3D input capabilities that allow the user to affect the virtual environment in a spatially intuitive way.

2.1 VR and 3D Interaction

Virtual models or objects in a computer simulation can conceptually be dealt with in terms of 2D and 3D (or any number of dimensions for that matter), and that distinction also applies to the interaction devices and displays they are presented through [6][7]. Most 3D applications, especially games, still use regular mouse and key-board setups in the case of computers, or regular hand held controllers for console systems. Both solutions can be considered strictly 2 dimensional interaction systems, as they are limited to 2 degrees of spatial freedom (DoF), the number of continuous interaction dimensions that are accessible through the interaction device for positioning and rotation [6][8]. This is as opposed to the 3 or more DoF desirable for a 3D interface in a VR environment. These 2D interface systems require some kind of conceptual 2D to 3D input mapping to take place as part of their application's interaction model, much like the visual 3D to 2D mapping that occurs when a regular
computer screen shows a 3D environment. The need for such conceptual perception and interaction mapping is a major detriment to the quality of immersion in virtual computer environments [6][7]. VR and 3D interaction solutions represent one way of amending that lack of immersion, with the complexity and functionality of the equipment used having a great effect on the results.

3D displays (figures 1 and 2) present a similar issue, in that they are instrumental in providing depth perception and visual immersion to users in VR systems. Of course, there are many ways of achieving this depth perception, certainly ones of less complexity and expense than the classical Head Mounted Display concept that VR still is associated with. Stereoscopic glasses present one such popular option. Depending on the particular version employed (color occlusion, shutter glasses etc) an installation of that type can be quite economical even for applications involving several simultaneous users. There are still issues of course, mainly related to the need for head-tracking for proper perspective, although such problems are also application dependent.

As for VR in general, it is known that usability is an important aspect of software application design, and Virtual Reality is no different[2][8]. VR applications and 3D interfaces are however still relatively inadequate when it comes to usability and interaction design, although the area is also the subject of some groundbreaking work on the research side [2][5][6][10][11]. VR systems in particular rely heavily on immersion and on the subjective perceptions of users [8], and should likely require even more consideration in this area. Human-Computer Interaction and usability however is still not a major focus of 3D/VR development [6].

2.2 VR and Education

A common use for recent and contemporary Virtual Reality solutions are realism or visualization focused simulations for research and educational purposes. This tendency has a relatively long history, and ambitions to adapt VR to various spatial and intuitive perception exercises for learning have fueled much of the research in the field [12][13][14][15]. Some have looked into collaborative simulations and efforts [16]
how to augment abstract theoretic presentations or more specific work with actual simulations for practical research or education purposes. The Virtual Reality Lab project (VRLab) at ASU-GARDS (figure 3), of Ain Shams University, is another project along these lines. It is hoped to eventually provide several local faculties as well as external interests with facilities for research, presentations, exercises and experiments – all assisted by Virtual Reality equipment and software solutions. The 3D display equipment will consist of stereoscopic displays of varying size and complexity, with accompanying software applications as well as custom designed work areas. The systems are augmented by a set of interaction devices, including regular 2D solutions like a mouse and keyboard setup, but also devices like 3D-mice, head tracking solutions and potentially other future expansions.

Finding appropriate interaction and interface solutions for these applications is an essential part of the project. For example choosing appropriate 3D input devices is crucial, as their technical specifications and functionality requirements have a direct impact on VR interaction possibilities; price, availability, familiarity, ease of use as well as the nature of the application itself are equally important aspects. Of interest when looking for ways to expand the 3D interaction tool set for the VRLab installation was any device that could offer some measure of efficient, intuitive and immersive direct +3DoF input capabilities at a reasonable price, while still providing simple and usable handling. Several options were considered, ranging from motion tracking setups to haptic equipment and data/force gloves, though most of them were quite costly. A very interesting possibility given the premises was the controller associated with the Nintendo Wii game system of late 2006, the Wii Remote Controller (Wiimote). At its release it differed greatly from contemporary game controllers by providing more than the classic set of buttons and directional pad (D-pad). Lately similar offerings from various companies have become available, such as Microsoft's Kinect and Sony's Move controllers, but at the time it was quite unique. The Wiimote promisingly carries a set of accelerometers capable of roughly determining the movement direction of the device, as well as an infrared camera sensor. These characteristics were originally intended for various gaming and entertainment purposes, but they have also become known for allowing a certain amount of spatial tracking capabilities, both with the Wiimote in an active and in a passive role. At this point in time many studies have been done regarding the uses of the Wiimote for non Wii Game System hardware, such as for PC applications. These include limited 3D positioning and rotation uses. The Wiimote's low cost compared to other 3D devices, its great availability and general user familiarity made it a worthwhile prospect for
this investigative project. It may perhaps not be able to satisfy all the stated VRLab requirements fully, yet as long as it provides some measure of effective 3D input functionality it should prove potentially useful for VRLab.

One planned application in particular at the VRLab installation involves the assessment of student exercises on the subject of mechanical assembly. Assembly in this case means the act of physically putting the parts of mechanical components together; for example with the purpose of making a water pump or engine complete and operational given a set of interconnecting parts. There are already such exercises in place at ASU-GARDS, partly for the purpose of teaching design methods, and partly for making the students familiar with the assembly part of the manufacturing process. These exercises are however not computer assisted, and the hope is that a properly designed VR environment, coupled with an appropriate 3D interaction solution, could greatly improve the impact of these exercises. This study of the Wiimote as a 3D input device will be using the Mechanical Assembly Assessment application test case for its implementation and evaluation phase, with the hope that the resulting conclusions and design may relevant to further VR applications, both as parts of VRLab and in other projects.

2.3 Nintendo Wii

The Nintendo Electronics Corporation has a long tradition in the console gaming industry, and has been at the conceptual and technological forefront for much of its lifetime. The 2006 Wii Console System (figure 4) still represents its major product line presently in 2011. Using an unconventional controller design and interaction scheme, together with the custom game software to take advantage of it, the product was mainly targeted at a casual or mainstream, typically less gaming oriented audience [24][25].

The standard Wii System controller is called the Wiimote (Wiimote), and besides the conventional game controller basics it also offers some original functionality: it comes equipped with a set of accelerometers for detecting snap movement along three axes, as well as an IR camera used mainly for pointing purposes. The novelty of the
control setup itself was to be considered one of its main features at release, although the actual gains in interaction capabilities varied greatly depending on the gaming software it was used with [25]. Eventually additional extension devices were offered, perhaps most interestingly the MotionPlus device which provided a gyroscope which greatly increased the potential accuracy of the provided accelerometer readings provided by the Wiimote.

**Previous Work**

There has been a great deal of research done on the Wiimote and its potential for non-gaming applications, including work in various 3D and VR environments. A common theme seems to be the study of the Wiimote’s IR camera for motion tracking from a stationary position [24][29][30][31], or for active input as a hand held device [24][32][33][34][35][36][37]. The IR camera itself is of sufficient quality for tracking real time motion at medium distances, and at its price range the only other options are web cameras and similar devices, none of which provide equal resolution or updating frequencies [24]. The low cost of the Wiimote’s hardware and peripherals in particular is one of the major reasons for the interest in its use in this context [33]. Additionally, the likely common user familiarity with the device and its operation makes it potentially useful for deployment and testing in non-controlled environments and casual situations [8].

Much work has also been done on how to further utilize the functionality that the Wiimote provides when used in accordance with its original hand-held-device design. Most interestingly, the IR camera, in concert with two or more IR emitter points, can be used to triangulate a relative (and variably exact) distance value. This particular functionality has been used in several studies [6][24][33][34], using different kinds of display hardware solutions. Through this method the Wiimote can be used either as a passive 3D tracking device or as an active input device with 3 to 6 degrees of freedom (DoF).

**Considerations**

For this project the use of the Wiimote as a +3DoF input device in a desktop (or with a large screen display solution such as a projector) environment is the main point of interest. The VRLab Mechanical Assembly Assessment (MAA) application will involve the active selection, manipulation, positioning and ideally rotation of objects in a virtual 3D environment. The overall goals is to use the Wiimote for augmenting this and possibly other related VRLab applications. Based on background research and
what is known of the Wiimote and VRLab, these following items constitute the immediate issues and facts that should be considered in preparation for the design and development phase:

**The Wii Remote:** This device is essentially a hand held IR Camera and accelerometer with a set of buttons attached. It can be connected to a PC and can interface with third party software, although it apparently does not fully follow the Windows HID Driver standard, and can have issues when handshaking (connecting) with certain Bluetooth hardware and software solutions. The IR Camera has a 100Hz updating frequency and records information at a 1024x768 resolution. The viewing cone spans a horizontal angle of roughly 45 degrees, and a vertical angle of roughly 38 degrees. This IR information is usually provided by a standard Sensor Bar by using two IR diode clusters separated by about 20cm. These metrics have consequences for the work space setup if the device is to be used with a PC system, whether the Wiimote is used as a passive recording device or as an active hand held controller, especially if only standard equipment is used. The ideal operating distance interval under these circumstance is a range of one half to a bit over three meters from the Sensor Bar, although a range of one to three meters seem appropriate for error margin reasons. The accelerometer in turn provides acceleration data in the range of +/-3g along three axes at a 100Hz [24][34]. The data from the device can be error prone, especially if it is not used under optimal conditions, with causes varying from light artifacts confusing the camera, to built in imperfections, battery levels or even the user having a shaky hand. This might be compensated for in most cases, either by some simple sampling and averaging over time, or by some filtering solution.

**Active or Passive Usage:** As can be gathered from the previous work associated with the Wiimote there is a fundamental choice available when it comes to using its IR Camera functionality. The Wiimote can here be simply used in its standard active mode as a hand held device, but the most common approach in 3D interaction solutions seems to be to use it as a passive infrared motion tracker. The advantages of this passive mode is that one can make more useful assumptions about the device’s location as well as gain further precision when it comes to 3D positioning [24]. The active mode however enjoys the advantage of extra input options in the form of buttons as well as the accelerometer, while also being easier to install and manage. There is of course also a fundamental difference in interaction paradigms between the hand/motion tracking interface and the pointer-like behavior of using the Wiimote actively. Additionally the passive mode requires extra customization efforts and work space considerations. The premises of VRLab and the MAA application suggested that
an active hand held interaction mode would be the most appropriate subject of study.

**Positioning:** Determining a virtual 3D position representing the Wiimote in the VR environment is a fundamental requirement of this project. While a reasonably accurate real world distance between the Wiimote and the Sensor Bar can be triangulated, the concern is rather the nature of the resulting virtual 3D position and how it should relate to the distance value and to the virtual Camera View (defined by the visual cone visible through the display screen) of the VR environment. Two fundamentally different ways of dealing with hand controlled movement in VR could be considered here [38]. The first option is the ray-casting projection method which determines a 3D position by projecting a ray/vector into the scene via a screen coordinate. The length of the ray and the direction of the Camera View then determines the resulting 3D scene position. This solution would produce an interaction experience akin to reaching through the screen into the virtual scene with the Wiimote, and while likely efficient it is probably the less immersive option as it would essentially mean using the Wiimote as a sort of 3D pointing device. The second option would be a variation of the arm extension method, where the user and the Camera View are using a shared frame of reference. The virtual 3D position is here determined through a vector offset locally relative to the Camera View. The dimensions of the vector would then be relative to the Wiimote’s deduced position relative to the user. Physical movements would here cause proportional changes in the virtual 3D position, giving the impression of controlling a virtual hand directly with your own. This is possibly the option with the best simulation potential, although compromises due to movement constraints and visibility would likely be necessary. It should however allow for the user to feel like he reaches into the scene from below the Camera View, with the movement closely matching the motions of the physical Wiimote.

**Interaction and Usability:** The usability of the Wiimote as a 3D interaction device is one of the more important aspects of this project. The setup of the work space itself is equally relevant – that includes issues such as operating distance and screen positioning. As noted the FoV of the IR camera and the positioning of the Sensor Bar puts some serious constraints on the amount of hand movement – in any direction – that can be allowed while providing full IR tracking functionality and 2D/3D positioning. This is less of an issue when using a large screen display solution with the user at the ideal 1-3 meters from the Sensor Bar, but is definitely a problem when using a desktop work station, which is the case in the current plans for the MAA application. Fundamental 3D interface tasks such as object selection and manipulation
suggested by Bowman, LaViola and Poupyrev[8] seems to suggest a useful approach for analyzing this aspect of design and evaluation.

The background information above and the related considerations should be sufficient for moving on to the design an outline of a preliminary 3D interaction concept appropriate for the Wiimote and the MAA application. Given an implementation and testing phase this should result in a functional implementation based on justifiable design conclusions. The first step to this end is to determine the fundamental and practical usage and interface requirements, and then suggest solutions and implementation options. Chapter 3 will cover that, with Chapter 4 presenting the resulting implementation details, including related conclusions and evaluation references. The technical details of the implementation can be found in Chapter 5.
3 Design and Implementation

For development purposes the main concern now is how to use Wiimote as a 3D input device to augment a stereoscopic display system. Once a practical solution is found, conclusions regarding general interaction and the Wiimote specifically can be made. The focus of the project is on direct interaction aspects of 3D input and interaction such as selection and manipulation, the reasons being partly the requirements of VRLab and the MAA application which will be covered later on. The Wiimote is a limited device in many ways, which likely will keep it from assuming all the necessary interaction roles of any specific 3D application interface. However assuming a proper interaction design it should be possible to adapt it to a specific subset of 3D interface requirements.

The purpose of this Chapter and the following subsections is to outline such a preliminarily 3D interaction design for the Wiimote with a focus on direct 3D input tasks such as selection and manipulation. The assumption is that other 3D interface tasks and components are satisfied in some manner by the target application, through the use of unused Wiimote buttons or other input devices. That could mean an interface solution contingent on the use of a keyboard and mouse separately for non 3D purposes, or one using a separate hand held device, or even one using the Wiimote in a secondary mode of interaction. In either case the specifics of how to use the Wiimote for these 3D input tasks must be determined. The first step is to cover the fundamentals of the Wiimote, the Mechanical Assembly work study application as well as the intended physical work space.

3.1 Premises and Assumptions

The Mechanical Assembly Assessment Application

This application (shortened MAA), being one of several at VRLab installation, will be the focus as well as the main test case of this project. The testing phase should ideally make use of some version of it, and as stated the resulting Wiimote implementation is intended to augment it. At the same time it should help to validate the sought Wiimote 3D interaction design from a more general perspective. The goal of the application itself is to provide a timed (or otherwise constrained) assessment exercise for
mechanical engineering students in which they may try assembling pieces of a machine or device in a correct and efficient manner. Objects are presented to the user on a virtual table surface. It is meant to mimic an actual assembly situation, with the presented goal of putting the pieces together in their correct order by positioning them properly next to each other and in the right order. These connection points provide some snap-on behavior which actually allows for some margin of positioning error in the technical Wiimote implementation. There is some measure of realism and simulation expected of the application as well, and this in turn puts certain demands on its interface and the accompanying input devices. Providing such an interaction experience given the Wiimote’s inherent constraints is the major purpose of this project.

The application’s basic functional demands on 3D input are essentially the actions of object grabbing and release with position and rotation control. The virtual Camera View is also likely to be fixed in the standard mode. The current plan is for students to use the application at desktop workstations equipped with stereoscopic display solutions with accompanying head tracking functionality. A large screen multiple viewer stereoscopic display setup for demonstrations is another possible environment.

The Wiimote

The constraints of the interaction design will be represented mostly by the specifics of the Wiimote (figure 5). The device is basically an infrared (IR) camera with a set of buttons attached. While it also comes equipped with a three-dimensional acceleration measurement device (the accelerometer), the IR functionality constitutes our main area of interest. The data produced by these components are made available for external access via an integrated and a semi-standardized wireless Bluetooth connection, which can be connected to most PC systems with the same functionality. In its standard mode of operation, as defined by the way the Wiimote is used with the regular Wii System and a home TV setup, its camera records IR points for orientation purposes, and can thus operate for example as a pointing device by using the observed position of the Wii Sensor Bar and its IR emitters. In practice any set of 1 to 4 IR points/clusters (as that is the number the Wiimote simultaneously can keep track of) can be used for this purpose, but we will initially only be dealing with a standard Sensor Bar and its 2 IR points. The accelerometers in turn give gravity readings in meters per square second along three spatial directions from the Wiimote itself: front-back, left-right and up-down. These can for example be used to to approximately
determine either the static or moving direction of the Wiimote. The IR camera itself has roughly a 40-45 degree Field of View (FoV) [24].

The Sensor Bar (figure 6) mentioned earlier is simple a passive IR emission device which provides two IR diode clusters, which are perceived by the Wiimote as two IR points at distances over 0.5-1 meters. Given that the real world distance between these points is known to be about 20 cm in the standard case, it is possible to triangulate the distance in meters between the Wiimote and the Sensor Bar assuming an orthogonal angle. For non orthogonal angles the value is merely approximate, however it is still likely adequate for our purposes for reasons that will be covered in Chapters 4 and 5. In either case the distance value can then be used together with the perceived Sensor Bar 2D position to determine a 3D position representation of the Wiimote. As noted in Chapter 2 the nature of this position can be defined in several ways, although they all require some type of shared frame of reference between the Wiimote, the User and the virtual Camera View. More on this issue in chapters 4 and 5 as well.

The distance- and angular intervals that allow for proper 3D position determination are defined by the size dimensions of the IR emitter (a standard Sensor Bar in this case, on which the IR clusters are separated by about 20 cm) and the FoV angle of the camera. Taken together they significantly constrain the allowable range of movement and minimum distance possible while maintaining the position calculations. As long as they are satisfied it is possible to provide the basic data needed for the Wiimote to function as a 3D input device. What
needs to be determined is if that is enough for it to function as an efficient and reliable 3D input device for a VR application. These movement and usage constraints would likely have a significant effect in such a case, so the next step is to consider the work area and to identify any major related issues and concerns.

The Work Area

A Virtual Reality installation usually comes with a large set of peripherals together with their related demands on work space and useability. A Head Mounted Display (HMD) setup for example has a certain set of usage and equipment requirements, and that also applies to a slightly simpler solution such as an active stereoscopic display setup. The VRLab installation uses such a system for its work stations, one which is capable of providing users with depth perception in a 3D computer environment. It is accomplished by displaying appropriate simultaneous visual data to both of the user's eyes. Depending on the hardware available the technology can be used both for desktop and large multiple viewer screen setups, such as a 3D projector screen. Actual testing will be mainly performed at desktop workstations.

Large Screen Display Setup: This is in a sense the standard case for Wiimote interaction, with the user standing or sitting at some distance (1-3 meters) from the display, with the Sensor Bar positioned above, below or next to it. Using the Wiimote at a home TV set is one variation of this. In this situation the Wiimote can be used in a relatively free and intuitive manner, with no particularly obtrusive limitations on movement and handling as long as the device is pointed roughly at the Sensor Bar.

Desktop Setup: The standard Sensor Bar could theoretically be as close as half a meter while maintaining the positioning calculations. It is a matter of keeping the reference IR points within the camera’s FoV without the IR Clusters being separated into IR diodes from the IR Camera perspective at close range, or them becoming too weak at larger ranges. This can be hard to manage in your typical desktop seating arrangement, forcing the sensor bar to be put for example on the floor or to the side of the display. One option is perhaps to use a non standard reference IR emitter that allows for the Wiimote to be used closer than half a meter. Another solution would be to have the user standing by the work station at a slightly larger range than originally intended, although that puts further demands on the work space accommodations.
3.2 The 3D Interface

The background and premises of the project have been covered, including the MAA application requirements and the limitations of the Wiimote. But before implementation and testing can start an actual interaction model must be outlined. To begin with some definitions are required. Interaction in three dimensions (3D) means perceiving and performing actions on virtual data objects. An “Object” in this context could for example be a sphere, an icon or a block of text, as they all have positional coordinates and virtual volume or area in the spatial sense. The difference compared to 2D is simply the order of virtual/spatial dimensions involved, although as covered in Chapter 2 many applications using 3D environments tend to deal with only 2 degrees of freedom by assuming a 2D interaction design. That means that they rely on the regular 2D screen display projection – the one that the user perceives for example on the computer screen – to interact with the environment. The purpose of this project is however to use the Wiimote to interact fully in three dimensions. This section will present the discussion and justifications behind a basic interaction model for a 3D interface for use with the Wiimote.

Tasks

A useful basis for the design of a 3D interface would be a set of interaction components – or tasks – together constituting a sort of basic requirements list. As covered in the background chapter one proposed structure of a 3D interface involves separating it into such separate tasks [8]: selection and manipulation, travel and wayfinding (navigation) and system control. In terms of actions in a 3D interface the list could be simplified to: selection, manipulation, navigation and control, the details of which will be detailed here:

Selection: The task allowing the user to indicate one or more objects of interest, within the context of the application. In the typical 2D interaction design this task is conceptually simple, and most users will be very familiar with the idea of for example using a mouse pointer to select an icon or a block of text by clicking or dragging the cursor accordingly. There are still other ways to go about it of course: using something like the TAB button to switch between available objects could be an excellent selection system in some situations, even for 3D applications. Furthermore, some 3D applications barely make use of actual selection, or even like some games simply use the viewpoint and navigation together with symbolic input in order to sometimes activate objects of interest. The Wiimote is fundamentally a pointing device, which is
likely to be useful in also in a 3D interface. For selection purposes that means translating its mouse-pointer like behavior to 3D terms, including how objects are indicated.

**Manipulation:** Manipulation can be similar to selection in certain situations, at least for spatial positioning, which is why the tasks are fundamentally linked in many 3D applications. Moving a pointer around a 3D scene may not necessarily be all that different from moving (manipulating) a proper object in the same manner. The key difference is that the properties of the selected object will be kept or modified, but not replaced. In other words an object will typically have properties besides the ones being currently manipulated, and the user should be provided feedback in order to make it clear what exactly is being changed during the manipulation process. If one were to use a 2D interface in a 3D application, this manipulation could be accomplished by changing the object’s screen position and depth position separately via different inputs (mouse for the screen, keyboard for the depth for example). For 3D interfaces the task might be performed similarly depending on the input device characteristics; a 3D mouse would move or rotate the object along the three dimensions directly, while a Force Glove would use hand movements to control position and rotation. Rotation is in fact the other major aspect of the manipulation task, and an important question for the Wiimote interface design is if rotation and positioning should or even could be performed simultaneously, or if they should be kept separate in different interaction modes.

**Navigation:** The position, orientation and related properties of the 3D Camera View has a great effect on selection and manipulation in most interfaces. Navigation in this context means the way in which the interface is used to control these properties of the Camera View (travel), and also how the user perceives the virtual environment and chooses how to path and move within it (wayfinding). Travel could here involve simply switching between a number of views with the press of a button, perhaps only being allowed to zoom the view along its viewing axis, just rotating it, or changing its position and orientation freely in 3D space. Many 3D games for example allow something like the last option along a floor or surface but with boundaries and restraints on rotation, for understandable reason – major view changes always carry the risk of disorienting the user. The nature Navigation in virtual 3D space thus depends on the type of application one is working with. This causes some issues when designing a general interaction design or interface, as one either ends up with a very basic and adaptable design, or one which only really works with a small subset of applications (like the MAA application for example). The goal would here be to find a
way to intuitively control the camera using the Wiimote, or at the very least identify which solutions are inadvisable.

**Input and System Control:** The control task is the interface portion dealing with matters separate from symbolic input, such as direct commands for selection or travel. It is essentially the interface equivalent of dialog frames and buttons in Windows, or the menus of a typical game. In a 3D application, one might attempt to include actual 3D objects, in the form of active symbols or banners, or perhaps simpler solutions like discrete key bindings. While system control is an important integrating aspect of any application interface, the Wiimote 3D interaction design is most likely to have little relevance to an applications general workings. However the Wiimote handling itself may require commands and key combinations, for example when switching between input modes, that would fit under this definition. As mentioned rotation is one task that may need to be kept separate from positioning, the solution of which would certainly constitute a control system issue, albeit with a symbolic component.

An interface is ultimately application specific, and the Wiimote is just intended to constitute a 3D input option in this context, perhaps one of several within a given interface. This means that not all interface aspects are equally relevant for a 3D interaction design. As covered the most important tasks for this project are selection and manipulation, as they represent the the main requirements of the MAA application. Determining how to design those tasks for the Wiimote means deciding on real and virtual interaction patterns and solutions, in short designing an Interaction Model. This model should provide both application specific and generalized conclusions in order to be relevant when discussing general VR issues later on.

**Preliminary 3D Interaction Model**

The interaction design for Wiimote 3D input must basically deal with two issues: firstly the practicalities of the physical Wiimote and how it is to be handled by the user, together with the equivalent virtual responses, usability in short; secondly the actual implementation design and the solutions to issues and required functionality. Most important at this stage is to determine how to satisfy the Selection and the Manipulation interface tasks. The first step to that end is to determine how users are to be expected to handle the physical Wiimote and how to design the corresponding virtual responses. As mentioned in Chapter 2 there are two main options that seem appropriate for the Wiimote in this situation. The first and arguably more intuitive option is the ray-casting solution, being where the user points at the appropriate
screen coordinate and “reels” objects back and forth by adjusting the distance to the Sensor Bar. This is a more detailed explanation of that concept in Wiimote terms:

**Wiimote Handling:** A 2D screen position can simply be determined by the position of the sensor bar as seen from by the IR Camera. This will be referred to as the IR position, with horizontal and vertical components, and it can usually be directly interpreted as a screen coordinate an implementation context. This coordinate is most easily affected by angling the Wiimote along the horizontal or vertical axes, but it can also be done by orthogonally moving the device sideways (which while being less effective has the benefit of maintaining the integrity of the distance calculation). As the IR position is available semi-independently of the Wiimote’s distance from the Sensor Bar, that then leaves the option of using that distance measurement for determining a depth position for the 3D position calculation. Rotation interaction could then be performed as mentioned earlier, by using for example the directional key pad or by switching interaction modes using one of the keys. The basic Wiimote handling will then consist of the user moving it back and forth to change depth position while directing the 3D representation through visual feedback (figure 8).

**Virtual Response:** A pointer representing the current screen position should be rendered during selection. To start with this pointer will be a 2D representation on the virtual screen layer representing only the IR position. When an object is selected for manipulation the pointer should be minimized or hidden as the object itself will then represent the 3D position of the Wiimote. Preliminarily, pressing and releasing the B button while indicating an object should begin and end the manipulation process. In short users will select objects in 2D and then position them in 3D until they are deselected. In this case using the A button for selection could for example be reserved for when objects are intended to be rotated by using the accelerometers, although using the D-pad for 2 DoF rotation during regular selection could be considered a more accessible solution.

The basic interaction concept described above can be considered a “2D-mode” solution, one that makes no particular concessions to immersion or realism in the VR sense, yet still allows for efficient 3D manipulation once objects have been selected. There seems to be two ways of expanding this model for VR and immersion purposes:
the first option is to define a “3D-mode” version of the design above, one which uses 3D for selection as well, essentially performing constant ray-casting while having the user reel a pointer back and forth in order to select and manipulate objects in full 3D. This solution however carries some issues regarding the shared frame of reference between the Wiimote and the virtual View, which will be covered in detail in Chapters 4 and 5. The second option is to try for an arguably more realistic approach, which would involve using the previously mentioned arm extension solution for determining a 3D position – a “VR-mode” so to speak. Rather than using the pointer paradigm the screen position concept will be discarded, and the 3D position will be entirely determined by deducing the position of the Wiimote relative to the user. The distance calculation is fundamentally the same, however the Sensor Bar as seen from the IR camera will instead be used to determine the sideways position of the Wiimote rather than a screen coordinate. In this mode it would likely be preferable if the user moved the Wiimote along the horizontal/vertical rather than turning it, however both ways of handling it should work. The actual selection and manipulation process would be the same as the 3D mode solution, with both employing a catchment box with some amount of leeway when indicating objects.

The Wiimote has the technical specifications and data availability necessary to allow for the implementation and testing of these three interaction modes. However the required calculations will have to be implemented in an adaptable way, allowing for some measure of unpredictable Wiimote handling, all depending on user expectations and the work space setup. The three interaction models will be implemented and evaluated both with regards to the VR specifics of the MAA application as well as general 3D interaction concerns.

**Testing and Feedback**

The purpose of the testing phase is mainly to evaluate and develop the interaction tasks of selection and manipulation as outlined in the preliminary interaction model, in addition to finding some agreeable solutions to the issues of the work area setup. The evaluation must mainly consider and deal with the design issues listed below. They are a mix of practical Wiimote implementation issues and some interaction goals and concerns that needed at least a minimum of user feedback before a solution truly could be considered.

**Immersion and Realism:** Being as this is a 3D interaction design with VR ambitions, it would be advantageous to provide some measure of immersion. As outlined in the
preliminary design there are two main ways of achieving full 3D interaction with the Wiimote, and evaluating them from this perspective seems appropriate. On the other hand the 2D mode solution should be considered from an effectiveness and familiarity perspective, which may in the end prove more important aspects in certain applications even when using VR equipment.

**Accuracy and Precision:** One of the more immediate requirements of the design was the need to give users a sense of precision and control over the Wiimote positioning in the virtual environment. This is an aspect beyond the basic calculations of these values, as filtering and “padding” of the positional values may be needed for users to feel like the pointer/object actually ends up precisely where they want it (though it likely never quite is that exact). A direct linear relationship between the hand held Wiimote and its 3D representation could also run the risk of seeming jumpy and unreliable, and so the options for correcting such issues have to be considered. Possible solutions include introducing a drag component to any position adjustments, or just dampening overall movement. A position difference threshold reducing jumpiness could also permit the user to move the 3D position along straight orthogonal lines.

**Calibration:** As detailed further in Chapters 4 and 5, the 3D position calculation of the Wiimote relies on using a common frame of reference between the real and virtual environments to maintain comfortable yet still responsive physical interaction range, while allowing for a large virtual depth interval. This issue is present for all three of the interaction modes detailed earlier, but is most problematic for the 3D and VR mode solutions. The Wiimote and the Sensor Bar in essence require this common frame of reference as the Wiimote has a forward limit (about half a meter from a standard Sensor Bar) while the virtual scene inversely has a near limit (the View/Camera “lens”). Translating the Wiimote/Sensor Bar distance to a virtual View/3D Object depth value thus requires that the implementation defines a point where the two values are meant to intersect, as well as an appropriate scaling factor. That means performing a calibration, whether manual or automatic. The ways of providing that calibration – whether through automatic, “smart” or manual options – has to be evaluated. The automatic option would for example be possible in cases where a 3D position value could be coupled with the users current Wiimote position during run time. In some cases this can be done seamlessly, such as in the 2D mode where 2D selection can be used to calibrate the frame of reference before 3D manipulation starts. In other cases one can default to some custom setting and constantly adjust the calibration (with the risk of confusing the user). A “smart” option
would involve deducing some intuitive position from the user's hand position and movements and then use that to determine a dynamic calibration point on the fly. Lastly the user could issue manual calibration commands by deciding on a physical position that feels appropriate as a center of movement. Feedback on how far automatic and smart solutions could be pushed is needed, as well as some information on when manual calibration could reasonably be asked for.

The VRLab Mechanical Assembly Assessment (MAA) application should serve as a practical test case for these specific issues, as well as the initial interaction design in the precious section. The user feedback was obtained during the final design phase, on site at Ain Shams University at the VRLab premises. Quantitative ways of testing the use of such an application could have consisted of measurements of the precision of user positioning, or checking the correct order of included objects, or by observing direct assembly timekeeping from start to finish. Another way would be to compare the efficiency of entirely different mechanical designs during a single user session. Assessments like these have been done in similar situations [19][21]. However as the relevant VRLab application was not finished at the time of testing it was elected to settle for a qualitative evaluation of the Wiimote as a 3D input device for the application's rudimentary interface and functionality. That means that only limited validation of the Wiimote in the specific MAA case could be performed, and a mostly generalized interaction evaluation was the result. This effects for example the resulting design in the way that it lacks specific conclusions regarding the application, and that the implementation itself lacks relevant customization. On the other hand conclusions about general 3D interaction principles for simple hand held devices such as the Wiimote should still be relevant.

The testing participants were students and teachers that at least had some basic familiarity with the expected uses of the MAA application and VRLab in general. Everyone had some knowledge of the Wiimote as well, or with game controllers and similar computer input devices. The testing session thus mainly involved determining which variations on the Wiimote 3D interaction theme seemed the most promising, as well as considering solutions and options regarding the practical issues above. Circumstances allowed for implementation and design changes to be done concurrently, and in the end allowed for an interaction design of reasonable usability compared to the regular 3D mouse that was also available. Specific feedback and justifications can be found along the design presentation in the following chapter.
4 Wiimote 3D Interaction Evaluation

This Chapter sets out to provide a set of discussion and testing based design conclusions for using the Wiimote as a 3D input device. The platform mainly used for these tests and the resulting evaluation and design was the Mechanical Assembly Assessment application of the VRLab installation at the University of Cairo. The process of obtaining these results began with the outline of a basic interaction design, and then moved on to the implementation and testing phase, finally resulting in a functional implementation. The structure below is intended to simply order the evaluation of these results and the corresponding conclusions by some measure of general relevance. The first part presents the general direct 3D interaction issues dealt with during development, ones potentially relevant for at least other similar hand-held devices such as positioning and basic interaction tasks; the second part covers the Wiimote specific issues, ones probably of limited relevance for other circumstances, like implementation issues and hardware and data quirks; the third part then goes full circle and deals with the VRLab side of things, discussing how the Wiimote fits into its role as part of an educational VR application.

4.1 Interaction Issues

These items cover the aspects of the design work that are of general relevance from a 3D interaction design perspective. While some specific conclusions and their background reasoning may be Wiimote centric, these issues - such as manipulation or general positioning as well as precision - and the considerations behind them carry at least some potential use outside the scope of this particular project. Given that the basic principles discussed here would have been part of this paper whether the Wiimote had been used or not, this seems to be an appropriate ordering. Each item begins with a discussion and background section followed by a concluding design statement.

2D Mode. 2D mode is the case in which the depth component of the 3D input device position is disregarded by the interface, and only the 2D positioning information is used. The depth value chosen as a constant here can either be none (the screen pane is used) or some arbitrary scene depth. Testers in Cairo appeared to consider the 2D interaction mode a fundamentally familiar experience, likely because of its similarity to how one would use regular 2D mouse input. It seems prudent to provide that
functionality for application interfaces when using input devices of this sort, especially if the design is intended to accommodate a range of 3D or VR application types. Selection and manipulation along the abstract plane defined by the screen and the default/current depth seems to be well understood and intuitive mechanics with little need for instructions as well. Using 2D selection together with 3D manipulation also seemed to be an effective and intuitive combination that takes advantage of in this case the Wiimote's fundamental pointer-like design while at the same time correcting somewhat for its disadvantages (the need for calibration, the sometimes awkward depth positioning).

Conclusion: 2D positioning is performed by indicating screen positions during selection. After selection and during manipulation positioning is performed by ray-casting into the scene to a depth relative to the selection depth by an application dependent scaling factor. Selection is performed by hovering the 2D pointer over of the intended object, and then indicating it by pressing and holding the selection button. The object will assume the current 2D screen position the pointer, however the current depth position of pointer will be set to that of the object. Manipulation is then performed by changing the 3D position directly.

3D Mode. 3D mode is when the euclidean 3D position of the input device is used to interact with the virtual environment. Providing 3D selection and manipulation was a fundamental requirement of the design independently of the actual device used, and this is satisfied by this mode as well as the VR Mode. It should be likened with the 3D manipulation process of the 2D Mode. The particulars of the design might vary of course, but the description above offers the bulk of it. Requiring the user to match the depth position before selecting an object was found to be awkward when lacking stereoscopic equipment (or some equivalent), and even then the effectiveness of 3D selection relies heavily on proper calibration. The 3D depth manipulation was found to be unintuitive for some, and it depended on well tuned depth interval settings and on the range of hand movement available to the user at any particular time. The selection, catchment and manipulation specifics did however work as well as expected.

Conclusion: Positioning is performed by ray-casting into the scene into an application determined depth interval. Selection is performed by moving a 3D pointer close enough to an object (depending on the catchment area properties and object size) and then pressing and holding the selection button. The object will snap to match the current pointer position and can then be manipulated.
**VR Mode.** The suggested VR mode is the case where the 3D position is determined by using a position relative to the user, ending up at an equivalent virtual position relative to the View. The idea is to provide a more realistic and immersive full 3D interaction option more akin to controlling a virtual arm than a pointing device. There are issues to consider however. For example assuming direct relationship and a static Camera View many positions mapped from the user into virtual space would be outside the View cone, and the reach of the virtual arm would be limited unless some non-linear reach scaling was used. Evidently these are issues of navigation, which means that this “realistic” mode also requires “realistic” View handling in order to function properly in the general case. The Camera View would have to move around to change the users virtual position, and it would have to rotate in order to visualize all the possible positions the user could reach at a given position. An extension of this concept is however to rather offer an approximation of realism by allowing the reachable virtual area to be arbitrarily defined by scaling and offsets in the form of a bounding box. This allows the mode to provide realistic or semi-realistic interaction depending on application preferences within a section of the view cone. In general the main problem with this mode is that the position of the pointer on screen is decoupled from the pointing direction of the Wiimote, which proves unintuitive unless the user is familiar with the exercise.

Conclusion: Positioning is performed through arm reaching mapped into an application defined bounding box positioned relative to the View's position and direction. A depth interval is used in the same way as in 3D mode. Selection and manipulation is also performed as in 3D mode, but with slightly different movement characteristics given the different movement scaling.

**Rotation.** Rotation is not the strongest aspect of the Wiimote interaction package, at least for VR purposes when used in combination with positioning. This is partly by design and partly because of the inherent properties of a hand held device with fewer than 6 DoF. Testing in Cairo verified this, which was unfortunate as that severely limited the simulation and immersion capabilities of the Wiimote as a full 3D input device. Separation of these inputs appeared to be the most effective solution, by simply allowing the Wiimote's accelerometers to do their work independently of the IR Camera. For non-simulation situations that mode of operation appeared to be fairly satisfactory however, and not overly detrimental to the device's immersive qualities, small as they are. When using a full 6DoF device this issue becomes less pressing as it would typically provide rotation input by default, however lesser devices like the Wiimote must be adapted to work properly in this context. This is still of general
relevance as any device with less than 6 DoF would likely face this issue.

Conclusion: Manipulation of an object’s rotation should be performed in a separate selection mode when not intrinsically supported by the input device used, such as is the case with the Wiimote. This can be accomplished in two ways: by using keys such as the Wiimote’s keypad to rotate objects during normal manipulation, or by other available means such as accelerometers which can determine 3D rotation readings. Regular positioning would for example obviously not be concurrently available in the latter case.

**View Navigation.** Considering how application dependent Navigation is, it is assumed that appropriate interface functionality would be implemented separately, in ways that may or may not use the the 3D input device. The testing that was done shows that navigation and traveling are very different tasks compared to the selection and manipulation concerns mostly dealt with up until now. The Wiimote certainly appears limited in that regard, and any similar hand held device would likely fare the same unless the entire interaction scheme was redesigned to revolve around that part of the interface.

Conclusion: Navigation control is performed by a separate mode, much like rotation. The same button that one would use to select an object for rotation could instead optionally allow for View manipulation if no object is available for selection.

**Precision and Reliability.** Merely determining the actual 2D or 3D position is not enough when providing an interface. Using for example a quadratic drag component to slow smaller movements while allowing larger changes in position to have a more immediate effect proved to be an almost entirely positive adjustment. Testing showed that the alternative tended to aggravate users. In fact most users just rejected the basic 3D position input concept until these basic quality improvements were implemented. As for the increased depth sensitivity, it is mostly an issue when using an application that allows for full depth interaction ranges of for example 1-100. Assuming a static camera view, and assuming that the user would want to operate at this entire range without moving his body, then using a simple linear model was just impractical. An intuitive and practical solution was to use some non-linear function to provide continuous yet adaptable scaling. Quadratic, exponential and inverse functions were considered, although one would likely want to retain the linear behavior in VR Mode while using non-linear scaling in the 2D and 3D modes.

Conclusion: The implementation should provide features for improving pointer precision in both 2D and 3D modes. A drag component of variable strength together
with data filtering are used to accomplish this. Applications will be able to decide which of the filtering levels and position adjustments are to be used. Depth movement sensitivity should be capable of both linear and non-linear scaling depending on application preferences.

4.2 Wiimote Issues

These items cover concerns and conclusions that are likely unique to the Wiimote and any design or implementation that attempts to deal with the device. They are fairly application independent in the sense that they apply when using the Wiimote for direct 3D input purposes, and for the same reason do not concern the platform of software environment used.

Handling and Positioning. Users appeared to find it intuitive to move the Wiimote sideways as well as back and forth to affect 3D positioning as long as the precision, accuracy and calibration issues were not too noticeable. While testing this in Cairo it was found that modifying sensitivity settings and movement limit parameters allowed for a fairly large range of adaptability for different applications at least in the 2D and 3D modes, thus the actual Wiimote handling design for direct 3D positioning for selection and manipulation tasks could be generalized rather than application specific. Conclusion: Positioning in 2D and 3D mode is accomplished by moving the Wiimote back and forth and by turning or moving it horizontally and vertically with the IR Camera facing the Sensor Bar withing an acceptable angle. Moving a pointer or object to a specific point on the screen is done by pointing the Wiimote accordingly, and changing the depth component of the 3D position is done by moving the Wiimote closer or further away from the Sensor Bar (or equivalent). In VR mode the device is ideally moved rather than turned sideways and up/down moved while being kept orthogonal relative to the sensor bar, although either option works in practice.

Calibration. Letting users have control over the calibration process proved to work well enough with our particular testers, however this relied on providing basic automatic calibration that the users then could correct slightly as needed. The manual calibration itself also requires some technical awareness and understanding and thus somewhat limits casual usage of the 3D and VR modes of interaction. It might be that the 2D mode is the most practical casual interaction solution for the Wiimote in this context, as the transition to 3D manipulation allows for a fairly seamless automatic calibration. Both the 3D and VR modes in the end rely on the smart calibration for decent usability.
Conclusion: Calibration is performed by connecting the application’s preferred interaction depth interval to an appropriate physical Wiimote and Sensor Bar distance value, and its ideal interaction range. Manual Calibration can be performed by user command at appropriate times by pressing the near, middle of far calibration buttons while directing the Wiimote towards the Sensor Bar and holding it at the preferred physical interaction position. Automatic Calibration is performed during 2D mode Selection. “Smart” calibration occurs at any time when the depth value of the current Wiimote distance is outside the defined usage range. It then adjusts the calibration point to capture the current position if it is relatively close.

**Work Space.** Several configuration options were considered and tested, and the smallest distance deemed practical with a standard Sensor Bar was around 1 meter, which is as expected given the information from previous work. Pointing the Wiimote in a different direction than the display location worked acceptably assuming the Sensor Bar was positioned appropriately. Most users appear to make the necessary spatial connection directly, which makes sense considering the level of abstraction already involved – merely consider the way one would use a regular mouse to control a 2D pointer for a similar feat of abstract coordination. The ergonomic disadvantages hardly need to be stressed however, and it is likely unsuitable for prolonged or frequent use for that reason alone. There are some possible variations on these solutions, but in essence they all involve attempts at enabling the user to sit close to the screen while still maintaining the working parameters of the Wiimote for 3D input purposes. As for using the Wiimote at greater distances, it seems to be an appropriate solution and should be used whenever possible. The interesting aspects rather come into play when one attempts to use non-standardized IR emitters. That means for example a smaller device that could potentially be used closer to the Wiimote. No practical tests were done on such solutions however. One further observation from the tests is that care must be taken that there are no additional IR sources not taken into account by the setup – like the IR synchronization communications of a stereoscopic display system, or just a common IR output of a laptop for that matter. Any such source greatly interferes with the Wiimote IR Camera if it is within its FoV.

Conclusion: Desktop work areas can mainly be used with this interaction design in two ways: in the first case the Sensor bar is located above or below the screen, and the user employs very slight movements along all axes for position interaction purposes. The sensitivity setting can be modified accordingly. In the second case the Sensor Bar is positioned 2-3 meters to the left or right of the user depending on handedness. The Work Area for large screens or TV equivalent installations can be used in the standard
Wii System fashion. The Sensor Bar or a similar device should be placed close to the screen, and the Wiimote can be used as it was originally designed to.

**Dual Wiimote Interaction.** The fundamental implementation design allows for this interaction mode, but it was found that the usability and interaction problems were significant. Keeping track of two visual indicators seemed to be a bit confusing in the 2D and 3D mode unless some visual feedback was provided for indicating which pointer represented which arm. It worked slightly better in VR Mode, mostly because the interaction bounding boxes could be kept slightly separated which helped with the visual feedback. The purpose of using the Wiimote as a 3D device is one of familiarity to the users, simplicity and cost effectiveness, which should be considered when attempting to expand its use in this manner. The simulation “advantages” offered by this solution could be debated as well, but it should be provided for cases like the MAA application which requires input from both hands.

Conclusion: The design allows for user applications to initiate two or more Wiimotes for interaction purposes. The interaction design is intended to support for example using one in either hand as long as the initialization and other usage aspects are properly dealt with.

**Non-Standard IR Emitters.** While the testing phase used various work space setups to evaluate the design, it was found that the available solutions for setting these up were limited by the dimensions and properties of the standard Sensor Bar. A smaller one (or some functional equivalent) will less powerful diodes could for example allow the Wiimote to be used at much closer distances, and the inverse would hold true for larger distances although that is less of a practical problem. An implementation should for that reason support non standard IR solutions as the setups covered in this report are merely a subset constrained by those practical limitations.

Conclusion: The implementation should allow for custom solutions to the IR emitter requirement of the Wiimote. This means that smaller or larger dimensions of the Sensor Bar equivalent are taken into account assuming that the specifics are registered in the implementation.

### 4.3 VRLab Issues

All of the above presented design issues and corresponding conclusions are intended to define a functional and effective partial 3D interface for the Wiimote as a part of the VRLab installation at Cairo University. While the testing did take place there, the specific application originally considered for this project was not fully operational,
hence the fairly "general" principles of the design and testing work presented thus far. Still, the specifics of the application were known, and due to the testing location and the knowledge of the testers themselves it is hoped that the results are still relevant. The virtual reality application is supposed to simplify the exercises related to courses and lab exercises in that particular field, with ideally providing a full VR environment for true immersion and simulation as a part of the VRLab installation. In practice however such fully immersive VR equipment was not available, hence this project and the interest in the Wiimote as a potentially useful 3D input device.

**Immersion, Realism and the Work Space.** The specifics of the Wiimote and its limited realism simulation potential really does have an effect on the MAA application in this case. Ideally the application was supposed to present two input devices to the user, preferably allowing realistic full range movement within the virtual environment in order to allow for quantitative analysis and assessment of the assembly process. The available equipment limits the ideal situation however. The Wiimote while being an efficient positioning device given certain allowances does suffer from several significant constraints when it comes to simulation performance: users must face a certain direction, maintain a proper distance and continuously point the Wiimote in the correct direction. This makes a fully immersive VR solution using the device difficult to achieve under the given premises. Some have attempted solutions involving several Wiimotes or Sensor Bars to get around this issue, but a more complicated solution would defeat the purpose of considering the device for VRLab in the first place. The work stations for the MAA application itself are not designed for fully immersive VR in the first place, hence the viability of using the Wiimote as is. Given the testing and the information on the intended work stations the Wiimote should definitely be useful for VRLab in either 2D, 3D or VR mode as detailed earlier. This however assumes that accommodation can be found for the Sensor Bar distance requirements. Either by using some custom version allowing for smaller distances, or by having work stations and surroundings that allow for the intended 1-3 meters of operation distance. Such modifications should not be overly onerous, but do require that some extra free space is available.

Conclusion: The Wiimote should ideally be used with the MAA application or some variation of it that allows for an appropriate work station setup, although desktop work stations can work adequately with some adaptations. The VR mode solution appears to best satisfy the basic requirements of the application, however the 2D and 3D modes have their advantages when it comes to intuitive and efficient 3D positioning.
4.4 Reliability

Despite the many practical aspects of this project, the potential errors here are mostly of an abstract or theoretical nature. There are certainly calculations and assumptions involved that could count as error sources, but as they have been fairly simple to check for accuracy (for example the distance calculations, being what they are, were easy enough to validate with confidence), and one could similarly reasonably expect that the code implementation does what it is intended to do. One could reasonably ask for quantification of the data noise issue which forces filtering, and for example determine how much of it is caused by user hand vibrations. But that would be more of factor if one used a filter that took noise distribution and such into account. The approximation error of the distance attribute could however be a more relevant issue. It is never quantified, although given its nature it is hard for users to notice it due to the slight distance changes incurred by default when angling the device. One could certainly also argue that the code itself should have been more thoroughly evaluated, for example by testing it with more than one application.

In the end it is rather more likely that the interaction design itself would suffer from fundamental problems. The probable causes would include the relatively narrow range of testers, a lack of thorough usability theory considerations and perhaps that some assumptions regarding the best uses of the Wiimote were made quite early. Especially the last issue was often ignored during development, as it was insisted right from the start that the active interaction paradigm for the Wiimote was to be used. It may have been the most suitable given the initial information of course, but as other studies have shown the Wiimote can easily be used for a passive role for motion or gesture tracking, which may possibly have been suitable for the MAA application or some other VRLab application given enough development time.

On the practical side the Wiimote presents some issues that did not really fit into the previous sections, but which should be considered during deployment. Some suggestions have been made that the FoV and IR Camera specs could vary somewhat between models and individual devices, which is certainly of interest but unlikely to greatly affect interaction due to the constant run time calibration and heavy filtering. Additionally the battery levels of both the Sensor Bar and the Wiimote can have a significant effect on data reliability without making the Wiimote strictly inoperable. For example a weak Sensor Bar battery appears to confuse the Wiimote into seeing the separate diodes as distinct IR points – likely due to the overall cluster strength weakening. This slightly gradual signal degradation can be confusing and elusive, and
should be considered as a possible error source. This situation can be avoided entirely by using a corded Sensor Bar, for example a standard one modified to be used with a USB slot.

As for the interaction design itself, while it likely is appropriate for use with the Wiimote as an input device in a 3D environment, it demands much in the way of accommodation from the work space setup and the willingness of users to work within its parameters. That is partially the fault of the Wiimote itself, as it is not a PC device, but also because certain issues probably could have been ironed out given more testing and development. Especially the calibration issues feel like they should be improvable.
5 Technical Details

5.1 WorldViz Vizard

The VRLab project installation uses a specific development and deployment environment: WorldViz Vizard (figure 9). It provides development support for a full 3D environment, as well as drivers for a wide range of peripherals such as interface devices and display systems. It offers an extensive software platform, providing a framework for creating, controlling and interacting with 3D environments among other aspects. For development purposes it offers a framework for interacting both with the various projects in the VRLab project, but also for dealing with the Wiimote itself – Vizard conveniently provides a basic software driver that allows for a connection to the device via the local PC Bluetooth stack. This driver is however useful mainly for connecting the software environment with a Wiimote device and for accessing its data. Beyond that custom calculations and adaptations have to be performed. This is preferably done using the supported Python scripting language, which provides a simple syntax and a partially object oriented design. For interaction applications specifically, it makes several classes and interfaces available, some fully useable and others merely prepared for extension. In this case, the extension sensor code interface will be used to provide an implementation in the form of a Python class meant to be interchangeable with other device interface implementations within the framework. The idea is that the default interaction mode that it provides should allow any application to make use of basic Wiimote support for 2D and 3D positioning without the need for any particular code changes. A seamless and direct interaction design should then make it useable with most types of Vizard 3D applications in this manner without the need for major interface modifications. This should also hold true for potential implementations along the same design outside of the Vizard environment.

The bulk of the code itself will be implemented in that same extension sensor class, which will be instantiated as an object within the Vizard user application. The class
will initiate a number of variables, and each instantiation will try to connect to its own unique Wiimote device on the local Bluetooth stack by using the Vizard Wiimote driver. At that point it will start polling the Wiimote registers for data at 100 Hz, continuously updating the current positioning variables by using a slightly filtered composite of the most recent set of data points. Changes to the data will fire sensor events for applications that make use of them, but they may optionally simply access the data via the provided functions.

5.2 Implementation Details

The code implementation is meant to obtain data from the Wiimote, perform the necessary positioning calculations and then provide them to user applications in a standardized format. The gathering of Wiimote data is solved by the Vizard plug-in, and the extension sensor interface provides the standardization format. These details are provided in the form of a step-by-step process starting with the basic Wiimote data and ending with a functional positioning interface. The included calculations will cover the fundamental mathematics of the overall algorithm.

Distance and the IR Camera

The necessary calculations for determining a virtual 3D position depend on the availability of certain data, and that a couple of premises and assumptions hold. The first step of the calculation process is to determine the approximate distance between the Wiimote and the IR emitter that is used for the occasion. For that the following data is required:

- The Field of View (FoV) angle of the Wiimote IR Camera. The 45 degree assumption will be used.
- The Width of the IR emitter, or rather the distance between the IR diode clusters that make out its IR points. On a standardized Sensor Bar equivalent that distance is about 20cm.

Assuming that the Wiimote is continuously facing the Sensor Bar roughly straight ahead, and that no “extra” IR sources are present to distract the IR Camera, this is
enough to perform the approximation. Given that basic trigonometry can be used to determine the distance between the Wiimote and the IR emitter. Figure 10 describes this relationship and the basic calculation. The Wiimote data in this case is represented by the \( P \) value which is obtained by using the normalized camera coordinates of the set of IR points that the device is perceiving that specific moment. These are recorded in normalized IR Camera coordinates, which is to say that the distance between two of them (assuming just those two are visible) will be on the range of 0 to 1. The FoV angles differ slightly on the horizontal and vertical axes however, and it can be taken into account for example by adjusting the \( P \) value accordingly. As the Sensor Bar will appear larger on-camera along the vertical axis given the same distance (due to its smaller FoV), that component differential is here adjusted downward by the camera resolution ratio.

\[
D = \frac{\frac{W}{2}}{\tan\left(\frac{\alpha}{2}\right)} \quad (1)
\]

\( \alpha = 45 \text{ degrees} \quad (2) \)

\( W = \frac{S}{P} \quad (3) \)

\( S = 0.2 \text{ meters} \quad (4) \)

\( P = \left\| (x_1 - x_2, r(y_1 - y_2)) \right\| \quad (5) \)

\( (x_1, x_2), (y_1, y_2) : \text{the normalized camera coordinates of the IR clusters} \quad (6) \)

\( r = \frac{768}{1024} \quad (7) \)

**Depth and Calibration**

Translating the resulting distance value to some kind of 3D position is in truth the more complicated exercise in this process. Mostly because one must determine what virtual point of reference the Wiimote should use for this mapping. The two basic options considered in Chapter 2 have been called ray-casting and arm reaching. In this case the first uses the IR position to determine an appropriate screen coordinate, and the distance value to calculate the ray length, or depth as we will call it. The second uses the IR position and the distance value to determine the position of the Wiimote relative to the user. In fact the major difference is in how they use the 2D IR position of the Sensor Bar, as the distance to depth conversion represents essentially the same
problem. Where ray-casting wants a depth value to determine the ray length, arm reaching wants a depth value to determine an offset into the View's direction. Both depth values can in practice be determined through the same algorithm, with the main difference being the choice of effective horizontal and vertical screen positioning.

The major issue when determining a 3D position is then the practical problems associated with obtaining a relevant depth value based on the Wiimote/IR emitter distance. As is evident they have an inverse relationship relative to the Wiimote's movements: positioning it closer to the IR emitter reduces the distance value, but should consequently increase the depth value. Furthermore there will always be a difference in movement scaling depending on the application and its scale. The solution: firstly, the application must have a range of movement defined, stating the virtual depth interval within which the user's physical movements should map; positions outside of that range may or may not be permitted, but the scaling and calibration process uses that interval as a base. This interval consists of a Far value $F$ and a Near value $N$ as illustrated in figure 11. These values can then be used to calculate an appropriate depth calibration value $C_{\text{depth}}$ which is then calibrated to the distance calibration value $C_{\text{distance}}$ so that $U(C_{\text{distance}}) = C_{\text{depth}}$

where $U(D)$ (def. 4) is depth as a function of distance. A corresponding physical movement range for the Wiimote is also defined, to in this case 40cm between the Far and Near depth values. This is interval is defined by the distance calibration value and an offset delta maximum value $R_{\text{distance}}$ in meters such that

$U(C_{\text{distance}} - R_{\text{distance}}) = F$ and $U(C_{\text{distance}} + R_{\text{distance}}) = N$. Whenever calibration is performed by coupling a “real” distance to a “virtual” depth value the depth function $U(D)$ based on distance is redefined. A generalized definition of the depth function was established:

$$U(D) = f(D - C_{\text{distance}}) \cdot S + O$$

(8)
\[ O = f^{-1}(C_{\text{depth}}) \]  \hspace{1cm} (9)

\[ S = - \frac{f^{-1}(R_{\text{depth}} + C_{\text{depth}}) - O}{R_{\text{distance}}} \]  \hspace{1cm} (10)

As can be noted the actual scaling function \( f(x) \) can be freely replaced assuming appropriate boundary constraints are used. Definitions 8 through 10 depend on the following:

\[ f(x) : \text{for example } x, \ x^2, \ \exp(x) \text{ or } \frac{1}{x} \]  \hspace{1cm} (11)

\[ N : \text{application defined} \]  \hspace{1cm} (12)

\[ F : \text{application defined} \]  \hspace{1cm} (13)

\[ R_{\text{distance}} : \text{custom setting, defaults to 0.2 meters} \]  \hspace{1cm} (14)

\[ C_{\text{depth}} = f\left(\frac{f^{-1}(N) + f^{-1}(F)}{2}\right) \]  \hspace{1cm} (15)

\[ R_{\text{depth}} = F - C_{\text{depth}} \]  \hspace{1cm} (16)

Given definitions 8 through 16 the value of \( C_{\text{distance}} \) can be calibrated to \( C_{\text{depth}} \) by using the the current run-time values of \( U \) and \( D \):

\[ C_{\text{distance}} = D - \frac{f^{-1}(U) - O}{R_{\text{distance}}} \]  \hspace{1cm} (17)

This calibration can be either automatic or manual, the specifics of which are fundamentally a useability concern, but it still has technical consequences. In the original and simple interface solution one would use 2D selection in conjunction with 3D manipulation to solve this: each time a virtual object is selected its current depth is coupled with the current distance value at that moment, and one would then use the set sensitivity factor to determine a depth offset directly from the distance offset during subsequent positioning until the manipulation ended. This however did not work well in a full 3D interaction environment, which is why definitions like the virtual movement interval with Far and Near values and physical movement range had to be introduced. So, in the current implementation the calibration can indeed take place during selection, but may (must) as well be initiated during run-time, either manually or through some “smart” deduction.

The manual calibration occurs on user command. When the appropriate button is pressed the current distance value will simply be defined as the new center of the current physical interaction interval. If the distance function returns a value that is
slightly outside the physical movement range interval then a recalibration will be performed. A large difference will cause a complete recalibration according to the current distance average. It should be noted that this particular solution provides a fairly seamless initial calibration for these modes as well, which otherwise was an issue encountered for casual use of the Wiimote in this context.

**Position and the Display Screen**

The specifics of the virtual 3D position can now be defined, given that we have access to a correct depth value as well as an appropriate screen coordinate. What is needed is essentially an absolute euclidean 3D position representing the Wiimote. For ray-casting this functionality is provided natively by the Wiimote Vizard plug-in based on its own interpretation of the visuals of the IR Camera and the Sensor Bar, combined with the depth attribute obtained previously. The resulting position is an absolute coordinate on the \((x, y, z)\) form, representing where one ends up in the scene when projecting the 2D position and depth value into it using the Camera View, as the point of origin as shown in figure 12.

For the arm reaching 3D position the depth value can be directly translated into an euclidean offset relative to the Camera View and its direction. The issue here is how to interpret the horizontal and vertical components of this vector, and we do it by using the IR position to determine a position within a predetermined bounding box. Up/Down and Left/Right boundaries together with the preexisting Far/Near values allows an application to define a bounding box for the arm reaching model. This bounding box is relative to the position and direction of the Camera View, and its size and position will determine its visibility and possibly "realistic" scaling compared to user movements.

In addition to the positional values the plug-in provides the Wiimote accelerometer readings interpreted as a rotational Euler value on the \((\text{yaw}, \text{pitch}, \text{roll})\) form, leaving a
total of three functions on a standardized format appropriate for the Vizard 3D environment and any application that may make use of them. These functions also just happen to provide the basic required information necessary to provide a 2D/3D extension sensor device in the Vizard framework, which thus satisfies the basic requirements of Wiimote 3D input implementation.

Filtering and Adjustments

The sections above describe the necessary deductions and calculations for providing baseline 2D, 3D and rotation parameters to the Wiimote extension sensor class. However given the nature of the Wiimote data and the random IR noise evident from testing, merely providing data straight from the source is not a viable solution. The values provided by the extension sensor will rather be the result of an average of a set of data points, with some filtering and adjustments performed. The key is to identify and adjust for the large number of strictly erroneous IR point readings that tend to show up, while keeping movement from becoming sluggish due to overactive filtering. For this implementation it was accomplished by storing a buffer of position samples, and averaging the the differences between them extending back a few tenths of a second. This delta value could then be applied to the last filtered value in order to form an extrapolated estimate of what the current real time reading should be. The estimate and the actual reading are then weighted together to form the next filtered value. This kind of algorithm has the advantage of capping new readings that differ greatly from previous ones, while allowing for large changes during time intervals of extended differences. The algorithm used is defined here, with the value \( w \) being a custom weight which is set to 5 in this implementation.

\[
f_t = \frac{1}{w} u_t + \frac{w-1}{w} e_t
\]  

(18)

\[
u_t : \text{actual measurement at time } t
\]  

(19)

\[
e_t = f_{t-1} + \frac{(f_{t-1} - f_{t-2}) + \ldots + (f_{t-(n-1)} - f_{t-n})}{n-1}
\]  

(20)

\[
n : \text{number of logged values used for the estimate } e_t
\]  

(21)

\[
w : \text{composition weight}
\]  

(22)

The current filtered value \( f_t \) is determined by using a composite of the current measurement \( u_t \) together with an estimate \( e_t \) based on the last few filtered values \( f_{t-1} \) through \( f_{t-n} \).
In addition to this fairly uncontroversial data filtering, there are a few more interaction and usability focused adjustments performed on the positional data. They relate to the accuracy and immersion aspects of the design covered in Chapter 4, and can all be enabled or disabled separately. The first adjustment addresses the accuracy issue, which is the perceived lack of precision experienced by users in the standard mode of operation, even after filtering. A working solution was to implement a function that dampened small movement changes in order to allow for precise positioning (both screen and depth), yet allowed large movements to pass through faster. In practice this was implemented by letting the position values trail the “true” position, with an adjustment rate that scales quadratically depending on the current distance delta between the true and current position values. Effectively a pointer or an object will adjust slowly to accommodate small differences yet quickly increase the movement speed to adjust for large changes. The calculation works by determining a factor by which to close the difference between the current and true position in the plug-in records. This factor is in its simplest form determined by taking the normalized coordinate differentials to the power of 2 (each axis is dealt with separately), then adjusting them for the plug-in's update frequency, before being applied to the current position differential. The result is that each update run is self contained, and the resulting effect on the perceived accuracy of both screen and depth positioning is quite noticeable. All of this could surely have been solved in a number of different ways, however the obvious need for this adjustment had to be addressed.

The second adjustment of note relates to the vaunted immersion factor, and as was described in Chapter 4 one way of accomplishing it with a Wiimote is to try and move away from the pointer-like behavior of its basic interaction premise. One way of accomplishing that is through the suggested VR mode, however the 3D mode could arguably be augmented to provide a similar experience. The key is to adjust horizontal and vertical position sensitivity depending on the screen depth the user is operating at, in addition to using linear depth scaling. The desired effect is the impression that positioning close to the View is quicker and easier, while movement is supposed to more sluggish at larger depths. The effect does not have to be truly realistic, but rather just convey the notion of difference in movement speed as a function of depth. In this implementation it is solved by simply optionally scaling the resident screen sensitivity factor by the current depth value, and by using a linear depth function.

This and the previous sections cover the details of the Wiimote 3D input implementation as well as the specifics of some important conclusions covered in Chapter 4. The majority of the Wiimote Extension Sensor Interface code directly
corresponds to the descriptions above, and is provided to applications in the form of extension sensor interface functions and variables. The general implementation can thus reasonably be assumed to be compatible with any Vizard application assuming it uses some standard conventions.
6 Conclusions

6.1 The Wiimote and VRLab

As covered in the background chapter the Wiimote is a specialized piece of game control equipment, and 3D interaction as well as other “expanded” uses of it were likely going to face issues no matter the development effort expended. This project was essentially about investigating an expansion of the device’s regular interaction mode into the third/depth dimension. Firstly the results should help by showing the merits of using that capability in settings besides the conventional large-screen-at-range one exemplified by for instance a home TV setup. Secondly these results also point out the inherent issues of the Wiimote when used like this. It was never intended as a PC device, and neither for use at a desktop or in any kind of application requiring simulation, precision or advanced 3D input for that matter. The VRLab applications on the other hand were just of that type, which produces something of a design conflict, in addition to the practical issues of PC connectivity, battery consumption and so on. Still, assuming that some further adaptations could be made regarding a suitable IR emitter as well as some modifications to the overall interaction design, it is possible that the Wiimote could become a proper part of a desktop focused application. As it stands however the implementation and interaction design presented in Chapter 4 is more a proof-of-concept regarding the potential uses of simple 3D interaction devices in a semi-immersive educational environment. This rather than being a strict solution for Wiimote use in the the MAA application specifically. The conclusion must be that the Wiimote could be an excellent low cost alternative for providing efficient 3D input and control in demonstrations and applications given the correct work space setup, or in situations permitting some interaction distance between the user and the display. This assumes that the reduced depth range resolution is acceptable, and that the interface itself is suited for the kind of interaction that the Wiimote offers in this context.

6.2 VR and 3D Interaction

This project was essentially a study of 3D interaction with a special focus on the Wiimote, but with definite undercurrent of generalized interaction principles. While the reasons for selecting that particular device for consideration have been covered, it should be noted that any potentially similar device could ultimately have been chosen.
– in the sense that the subsequent discussion relating it to the 3D and VR interaction concepts would probably have looked much alike. In the same way many of the interaction principles and solutions entertained could have applied if for example a passive tracking solution had been chosen. The reasons for using the Wiimote were simply due to needs such as functionality, simplicity and low cost. These are fundamental for VR applications and installations that do not enjoy substantial financial backing, and ideally a cheap device actually designed for the purpose would have been preferable.

The true issue facing the MAA application, VRLab in general as well as many other VR or 3D applications – the concern that initially caused this evaluation of the Wiimote in the first place – was the lean availability of simple and affordable 3D interaction devices. The particulars of this project and its results constitute just one aspect of that larger issue, one which is basically about a lack of market penetration and demand for VR interaction technology. Even after all these years, academic and certain commercial VR and 3D applications are still left with limited hardware options; such devices are actually starting to be relevant for uses one would expect to be of fairly commonly deployed by now considering the hardware and software developments of the last couple of decades. However contemporary consumer and mainstream commercial 3D software is still very much reliant on 2D display and interaction solutions. This in turn greatly limits the profitability margin for introducing 3D interaction devices at this time. The glaring exceptions are the console platforms which can afford to tie such devices to software and platform offerings, and we will surely see some advances on that front eventually.

Regarding display devices however, as mentioned at the beginning, progress is being made in several areas, exemplified by for example the increasing number of 3D motion picture productions, or home solutions for TVs, PCs and portable devices. This increased availability of 3D display technology does suggest a growing potential for true 3D interfaces and VR concepts at the consumer level, and that prospect should include effective 3D input devices at affordable prices at some point, perhaps even ones designed with usability and simplicity in mind. The Wiimote has been shown capable of providing arguably useable levels of direct 3D interaction although it was never designed for that particular purpose. It and similar devices from Nintendo's competitors show how a simple design that is accessible to users can provide VR input functionality without imposing or complicated interfaces which more professional devices are in some sense required to provide. A device designed along similar principles but with native 3D interaction support could for the same reason prove
viable in the near future. The specifics of such a device would of course depend on its intended market, but one for home use could probably co-opt the game controller paradigm to great advantage for the same reason that the Wiimote concept worked out: user familiarity with its overall design as well as with the simple and symbolic way input itself is handled. The Wiimote made use of the pointing device behavior concept for example, in addition to having users treat it like a handle or a stick when employing the accelerometers. Such design considerations seem sensible in the general case as well.

A generalized 3D interaction device for VR, games and professional applications could of course be hard to design with the same approach that the Wii System used. However the 3D pointing and hand tracking positioning concepts seem to be very strong interaction solutions, and might stay commonplace as 3D interaction solutions find their way into people's homes.
References

[1] I.E. Sutherland
"The ultimate display"

"New directions in 3D user interfaces"

[3] H. Godbersen
"Virtual environments for anyone"

[4] Nintendo
"Nintendo 3DS"
http://www.nintendo.com/3ds/ (2011-03-10)

"Bringing VR and spatial 3D interaction to the masses through video games"

"Swordplay: Innovating game development through VR"

"3D input for 3D worlds"

"3D User Interfaces: Theory and Practice"

[9] L. Valboma, A. Marcosa
"WAVE: Sound and music in an immersive environment"

[10] C.A. Wingrave, D.A. Bowman and N. Ramakrishnan
"Towards preference in virtual environment interfaces"

"High-performance wide-area optical tracking: the HiBall tracking system"

"Immersive training systems: Virtual reality and education and training"

"Curricular modules: 3D and immersive visualization tools for learning"

"Understanding the Costs and Benefits of Using 3D Visualization Hardware in an Undergraduate Mechanics-Statics Course"

"Virtual Reality for Education?"
Proc. of VRST 2000

"The NICE Project: Learning Together in a Virtual World"
Proc. of VRAIS 1998
[17] Ch. Bouras, G. Hornig
"Architectures supporting e-Learning through Collaborative Virtual Environments: The case of INVITE"

[18] C. Johns
"Spatial Learning: Cognitive Mapping in Abstract Virtual Environments"
Proc. of the 2nd international conference on Computer graphics, virtual Reality, visualisation and interaction in Africa.

"Virtual Reality-based Spatial Skills Assessment and Its Role in Computer Graphics Education"
Proc. of SIGGRAPH 2006

"Sciencespace: Virtual Realities for Learning Complex and Abstract Scientific Concepts"
Proc. of VRAIS 1996

"A virtual reality-based experimentation environment for the verification of human-related factors in assembly processes"

[22] T.S. Mujber, T. Szeczi, M.S.J. Hashmi
"Virtual reality applications in manufacturing process simulation"

[23] M. Limniou, D. Roberts, N. Papadopoulos
"Full immersive virtual environment CAVE in chemistry education"

"Hacking the Nintendo Wii Remote"

[25] K.Hall
"The Big Ideas Behind Nintendo’s Wii"

[26] Nintendo
"Wii: technical Details"

[27] Microsoft
"Xbox Kinect"
http://xbox.com/Kinect/ (2011-03-10)

[28] Sony
"PlayStation Move"

"Optical tracking using commodity hardware"

"Gesture recognition with a Wii controller"

"Low-cost, at-home assessment system with Wii Remote based motion capture"

[32] Y.W. Chow
"The Wii Remote as an input device for 3D interaction in immersive head-mounted display virtual reality"

[33] Y.W. Chow
"3D Spatial Interaction with the Wii Remote for Head-Mounted Display Virtual Reality"
[34] T. Schou and H. J. Gardner
“A Wii Remote, a game engine, five sensor bars and a virtual reality theatre”

[35] L. Deligiannidis and J. Larkin
“Navigating inexpensively and wirelessly”

“3D interaction with volumetric medical data: experiencing the Wiimote”
Proc. of the 1st International Conference on Ambient Media and Systems, Quebec, Canada, 2008, article no. 14.

“The Wiimote and Beyond: Spatially Convenient Devices for 3D User Interfaces”

[38] D. A. Bowman and L. F. Hodges
“An Evaluation of Techniques for Grabbing and Manipulating Remote Objects in Immersive Virtual Environments”