Perspective corrected view on (touch-)table surfaces

Depth perception on flat surfaces using head tracking

Harold Bruintjes
h.y.bruintjes@student.utwente.nl

ABSTRACT
Adding an illusion of depth to touch table surfaces may improve the way users interact with them. Using head tracking and perspective correction an illusion of depth can be created, which may allow more intuitive interaction where the user can use both touch and head movement to use the system. Currently, various solutions for touch tables exist and head tracking systems can be made relatively cheap. The main question the research aims to answer is whether adding depth perception influences the user experience in a beneficial way. The research will consist of implementing a prototype to answer the technical aspects of this problem, and a user interview for the user related part.

Keywords
Head tracking, Touch table, Wiimote

1. INTRODUCTION
When working with a touch table, being able to literally use your head as a means of interaction can be a valuable addition. By using head tracking to add a perception of depth, manipulating 3 dimensional data becomes more intuitive as the view matches what the user would see in the real world. Traditional (touch) table displays only show 2 dimensional interfaces and have no support for depth perception in 3 dimensional interfaces. Adding 3 dimensional interfaces with depth perception allows the user to view the elements of the interface in various angles, freeing the user from having to manually adjust the view to get a different overview. This allows the user to perform other manual tasks simultaneously, possibly decreasing the time needed to perform the tasks the user wants to perform.

Touch tables are an intuitive way for both displaying information and accepting input from the user, by showing content on the surface of a table and allowing users to touch that surface to interact with that content. Multiple users can use a touch table at the same time, allowing for collaborative work. Head tracking allows a system to keep track of the location of the user’s head. This allows the system to check whether the user is viewing a display and, if so, from what angle. Based on this information adjustments in the systems state, such as the display of the interface, can be made. A practical use for head tracking can be found in the display of 3 dimensional images. Instead of displaying it from a fixed perspective, updating the view according to the position of the user’s head allows for a more realistic display.

The main goal is to explore the possible benefits of adding depth perception to touch table surfaces. Recent developments in consumer grade electronics have lead to a cheap method for head tracking [1]. This system uses a Wiimote that is capable of tracking the location of infrared light sources. Having users wear these light sources on their head, the Wiimote is capable of tracking it. This allows for the creation of a head tracking setup with a touch table, that can be used to determine if the addition of depth depth perception has any benefits over traditional touch table interfaces.

1.1 Problem Statement
Touch table interfaces often consist of a 2 dimensional surface on which various elements can be positioned and modified. However, it is possible for these elements to overlap each other, or simply take up all the available space, especially in environments with little screen estate. A possible solution for these problems is the application of a 3 dimensional user interface, in which elements can be viewed from multiple angles. Overlapping objects can easily be identified by rotating the view towards the underlying object, and zooming in or out can provide extra space for element placement [2]. However, 3 dimensional interfaces often require extra input from the user to adjust to the required view [3]. A method to do this automatically based on the location of the user could alleviate this issue. Currently, head tracking systems are becoming more accessible and can even be constructed from consumer grade products [1]. Most computer systems allow for very advanced 3 dimensional rendering, so a relatively simple interface is easily achievable. The main question is whether the addition of depth perception to touch tables has any benefits in addition to traditional 2 dimensional interfaces. Underlying issues are how to integrate depth perception into a touch table interface, and how to determine what the benefits of depth perception are, if any.

1.2 Goal
This study investigates the feasibility and usefulness of the integration of depth perception into touch table software, determining if there are any benefits doing this. The results can aid in the development of new interfaces for touch tables. Developers of new user interfaces could make decisions on implementing depth perception based on the outcome of this study.

2. RELATED WORK
This study is based on the work done by J. Lee on using the Wiimote for head tracking purposes [1]. His work shows how to use a Wiimote to enable a FishTank VR environment on a regular desktop computer.

Much of the research in FishTank VR solutions is still being done now, but a lot still needs to be done. For example, Mulder and Liere [4] devised a method that improves the depth perception of object that are partially clipped by the edges of the viewing screen. Other work investigates the usefulness of such systems. According to a study done by Demiralp et al., FishTank VR solutions appear to be more effective than CAVE [5] solutions “for applications in which the task occurs outside the user’s reference frame, the user views and manipulates the virtual world from the outside in, and the size of the virtual object that the user interacts
with is smaller than the user’s body and fits into the FishTank VR display [9].

Work by Dorfmüller and Wirth [7] shows a system were two infrared cameras are used to get a 3 dimensional position of the user’s head and hands. Their work suits similar types of applications as this might (see section 4.2 (Applications)), although their approach is based on a setup with two cameras that requires extra software to post-process the image data acquired from these cameras.

Other work aims at finding low cost solutions, such as the Virtual Window system created by Penner and Parker [8]. This system was designed to be a relatively inexpensive platform to develop applications for “Head Coupled Displays (HCD)” or an inexpensive end user HCD. It introduces a few techniques that enhance the experience in conditions that would normally degrade the effect of immersion.

3. RESEARCH QUESTIONS

The main question to be answered is “Does depth perception improve the user’s interaction with a touch table?” To answer this question, a few subquestions need to be addressed first:

- How should the perspective be adjusted according to the position of the user’s head?
- How can the position of the user’s head relative to the table be determined?
- What type of interface/application for touch tables allows depth perception to be of additive value?
- How can improvements in user interaction, due to adding depth perception to touch table interaction, be measured?

The former two subquestions aim to answer the technical details of the main research question, whereas the latter two will answer how the user will perceive the system.

4. BACKGROUND

4.1 Headtracking

Most head tracking solutions work on an optical based approach. One or more cameras take images of the current scene and user which are then processed and analyzed to determine the locations of the user’s head. In most cases, the analysis algorithm searches for a special pattern in the image generated by the camera. This can be a pattern that is based on a clearly recognizable shape, such as a dot pattern, or, more complex, the shape of a face. The former is more easily implemented, but the latter prevents the need of the user wearing special head mounted gear for the camera to see.

Implementations can be based on visible light or infrared [7]. Visible light based solutions can make use of regular consumer grade cameras, which allows for very low cost solutions. However, they only work correctly in properly lit conditions and detecting the object to be tracked from the image can be very difficult. When using an IR based system, separate IR light sources are required in order for the camera pick up any signal. This can be done by illuminating the working area or by making use of IR beacons. In the former case, it is required that the object reflects enough of the IR light whereas the latter case requires the beacons to be attached to the object.

4.2 Touch tables

Touch tables generally consist of a flat surface on which an image is projected and that allows the detection of touch. Various solutions exist for both image display and touch detection. The image can be displayed using a screen on the tables surface, such as the landaTable [9], or by using a projector above or below the table surface. Touch detection for touch tables can be based on various techniques [10], of which the most common will be listed here.

4.2.1 Optical recognition

With optical recognition, a camera is mounted near the table facing the surface. The camera can detect if any object is near or on the surface using various techniques, of which possibilities are:

- Frustrated total internal reflection (FTIR) [11]: Using an IR light that under normal circumstances completely reflects inside the surface. When an object is placed on the surface, the internal reflection is frustrated and the light hits the object, which will then reflect that light towards the camera, allowing it to register the object.
- Diffused Illumination (DI): Using an IR light that is placed below the surface of the table that illuminates all objects above the table. The camera can then register those objects.
- Active Recognition: Using objects that emit a special light pattern themselves to be recognized by the camera, as is used by the lambdaTable setup [9] and the Wiimote whiteboard project [1].
- Another approach is to place various IR sensors at the edges of the table, with IR lasers aimed parallel to the surface. Any object placed on the surface will create a shadow, that the sensors can use to triangulate the position.

4.2.2 Capacitive coupling

Capacitive coupling makes use of conductive properties of objects that are in touch with the table. When a conductive object is placed on the table, a current can flow from transmitters on the table to receivers. The transmitters and receivers that allow the current to flow determine the location of the touch. Two possible implementations are:

- SmartSkin [12]: The SmartSkin technology uses a grid of vertically placed transmitters and horizontally placed receivers. The transmitters are sequentially activated, resulting in a time-divided input from each combination of transmitter with receivers. This allows for an accurate reading of multiple touches.
- DiamondTouch [13]: The DiamondTouch technology has only horizontally and vertically placed transmitters on the table. The receivers are mats where the user can sit or stand on. As each user is in contact with a different mat, a distinction can be made between them allowing for multi-user interaction. The downside is that multiple touches from the same user can generate an ambiguity as they run through a single transmitter.

5. METHOD

To answer the technical questions, a prototype is implemented with a 3 dimensional interface. It is also both able to perform head tracking and able to adjust the perspective of the interface to allow depth perception to work. A user study consisting of performing a test with the prototype and giving an interview will answer the user related questions. The test with the prototype will give insight on how well the depth perception using head tracking performs compared to an interface without head tracking. The
the size of the sensorbar currently on the screen, we have

5.1 Head tracking using the Wiimote

The Wiimote contains an IR camera with a virtual resolution of 1024x768 and a field of view (FOV) of approximately 45 degrees in horizontal and vertical direction. The camera is able to register up to 4 distinct IR light points, which are then translated by the Wiimote into coordinates in the virtual screen plane. In order to determine the positions of the user’s head, the Wiimote was used in combination with a head mounted sensorbar. The sensorbar is a device that contains two IR light sources separated from each other at a fixed distance. The sensorbar is attached to a baseball cap, allowing the user to wear it while using the interface. The Wiimote is connected to a PC using a bluetooth interface. Once a connection is established, the Wiimote can be polled for any changes in its current state, including a change in the location of the registered IR points.

To determine the location of the sensorbar relative to the camera, the locations of two IR points on the virtual screen have to be read and analyzed. The distance between two points is an indication of the distance between the sensorbar and the Wiimote. The position of the two dots determines the angle on both the X and Y plane of the sensorbar, which, combined with the given distance, allows the calculation of the X, Y and Z position relative to the Wiimote. Taking the offset and orientation of the Wiimote relative to the screen of the touch table into account, the absolute X, Y and Z position of the user is easily calculated.

Given this approach, if \( size \) is the size of the sensorbar in units of the Wiimote virtual screen size at a distance of 1 meter, and \( x \) is the size of the sensorbar currently on the screen, we have

\[
distance = \frac{size}{x}
\]

To determine the position on the x-plane, given the center of the two points \( center \) (in units of the Wiimotes virtual screen), we use

\[
angleX = \frac{center - 0.5}{screenWidth}
\]

\[
positionX = \sin(angleX) \times distance
\]

(all positions being in meters). The calculations for the y-plane are similar.

This implies that the sensorbar always has to be aligned parallel with the screen plane of the camera of the Wiimote, because otherwise the rotation will cause the perceived distance of the two light sources to change. A solution would be to use a device with at least 4 non-planar IR lights, as this allows to track the rotation as well. However, this requires a custom build IR beacon and is not a necessity for this study.

This answers the question on how to track the location of the user’s head. When the sensorbar is kept at the same location as the user’s head, the Wiimote can use the above method to determine the position of the user’s head, simply by finding the position of the sensorbar. In order for the sensorbar to follow the user’s head, it can be attached to head wear such as a baseball cap, which the user can then wear when using the system.

5.2 Depth perception

In order for the interface to generate an illusion of depth, the viewing parameters have to be adjusted given the information about the position of the user’s head. In most 3d rendering interfaces, a notion of a virtual screen plane and a virtual camera exists. The camera and the plane define a bounding volume, which is often referred to as the viewing frustum.

When rendering the image, the 3 dimensional data has to be displayed on a 2 dimensional screen. In order to achieve this, a projection has to be applied. Under normal circumstances, this projection will ensure that objects at a greater distance will appear smaller on the screen and vice versa. Two parameters that control this projection are the position of the camera relative to the screen, and the size of the screen itself. By adjusting any of these parameters, the angle at which objects are visible, called the field of view (FOV), can be changed. This is effectively the same as a person moving closer to a window, or moving away from it. When moving closer to the window, more of the outside will be visible, which means that the person’s FOV has increased. To increase the FOV, similar as with the window, either the camera can be moved closer to the screen plane, or the screen plane can be enlarged (see also figure 1). In this case the first method is used. This ensures that objects that are located at the exact position of the screen will remain the same size independent of the position of the camera.

By positioning the camera at the location of the user’s head in the virtual world, and the screen plane at the location of the touch table, a perspective corrected image is generated that matches the user’s current view. This is known as FishTank VR.

This immediately answers the question on how to perform the perspective correction. By simply adjusting the position of the virtual camera and the virtual screen plane, the correct perspective can be generated that gives an added illusion of depth.

When drawing objects in the 3 dimensional world, it is also possible to place them in front of the screen. The perspective correction will then act as if the object were floating above the table surface. In this study the choice is made not to do this, as the depth illusion of these objects is harder to maintain (though techniques exist to improve on this, see [8]) and interaction using a
table surface would become less intuitive. This is because all the interaction with these ‘floating’ objects occurs at the surface of the table, meaning interactions with the surface cause the user to reach through these objects, and positioning the cursor behind the object will still display as if it were on top of it. The result is conflicting depth cues, as the closer object is occluded by the surface that has a greater distance.

### 5.3 Link between Wiimote and Interface

In order to map the real world data of the user’s position with the interface, a fixed coordinate system is used. In this coordinate system, the height of the user’s head above the table determines the z-position. Positive z means the user’s head is above the table, negative z means its below the table. The position left or right relative to the table is denoted by the x-axis (respectively negative and positive), and the distance from the table is denoted by the y-axis (closer to the table is positive).

Because the virtual screen of the Wiimote can be oriented on a different axis than the table, a translation may have to be made. When the Wiimote is oriented as in figure the z-axis of the Wiimote maps to the negative y-axis of the screen, and the y-axis of the Wiimote maps the z-axis of the screen. This is trivial to implement in the prototype. When required, it is instructed to swap the y and z values, where the Wiimote’s z value is negated.

### 5.4 Experiment

![Diagram of Wiimote setup](image)

**Figure 2: Setup of the test**

The set-up consists of a touch table, a computer running the test application, a Wiimote and a head-mounted infrared source, as can be seen in figure. The test application itself is kept as simple as possible in order to answer the research questions. For example, only mouse-driven input is used. The Wiimote will be positioned behind the table in such a manner that the user (or more specific, the IR light source) will stay within view when moving around the table. The head mounted IR light source should always be pointed towards the Wiimote so it will stay visible. This means that the user can only sidestep or lean sideways in order to move his/her head sideways. The computer will then determine the user’s location relative to the table by analyzing the IR data recorded by the Wiimote and adjust the view projected on the table accordingly. To move the mouse cursor, the user is given a wireless keyboard with a built-in trackball. The touch sensing of the table will remain unused.

### 5.4.1 Application

The application consists of a small GUI to control the timing and a main interface that the user has to interact with. The interface consists of twelve flat squares (objects) that are clearly numbered. Each object can be selected by positioning the mouse over it, and pressing the left mouse button. An object can only be selected in order, meaning that every object with a value lower than the object to be select must already have been selected. A selected object can be distinguished from another object by a different color. Objects can be moved around the screen along the screen surface, but cannot be brought in closer or moved further away. To move an object, a user has to select it using the mouse cursor followed by moving the mouse.

All objects were positioned inside a box the size of the viewing screen and with a fixed depth. Objects are positioned with independent depth values, allowing some objects to obfuscate others.

On top of all the objects (to be more precise, at exactly the position of the screen plane) a transparent plane is placed that covers exactly half of the screen. This plane prevents objects from being selected that are under it, but allows the user to still see those objects. This will enforce users to make use of the head tracking in order to interact, preventing them from solely using the mouse. The plane can be moved around the screen based on the test currently run. Users can do this by either moving their head, the depth perception will allow them to ‘look around’ the plane, or by moving the plane using their mouse.

For each test four different distributions of objects were used. The objects are placed at random positions, but each user has the same random distributions. Each test consists of three different phases, each phase using a different method of interaction. One phase does not involve any head tracking, and will serve as the control. It will be as similar as possible with the other phases, but instead of using head tracking to adjust the view, complete mouse driven input is used. The two other phases use a form of head tracking and will serve as the actual test. When a phase is completed, the time taken to complete it will be recorded for later analysis. Each phase consists of the user having to select each object in order. When the timer is started, the user is allowed to select the first object, and when the last object has been selected, the timer will automatically stop.

The test phases are performed in random order and grouped by each dataset. This ensures that any learning effect does not solely affect a single testing phase, but is distributed among them. During a test, a small icon is displayed informing the user what type of test is currently being run and after a test what type the next test will be. As the ordering of the test phases is random, the icon will help users determine exactly what they are expected to do.

In the first phase, free head tracking, the view is freely adjusted to the position of the user’s head in all directions, but the obstructing plane will be fixed in the middle of the screen. This allows the user to select objects under the obstructing plane by ‘looking around it’. The icon associated with this phase consisted of two double-headed arrows placed crosswise over each other.

In the second phase, limited head tracking, the view will only adjust the the height of the user’s head and the x position. This means that moving away from the table or walking closer to it will have no effect on the rendered image. The obstructing plane is still fixed to the center of the screen. The icon of this phase consisted of a single double-headed arrow in a horizontal orientation. This phase was specifically designed to overcome the problem of objects moving out of reach when the user moves towards the ta-
ble to interact with it. As the user moves forward, the objects will move forward as well, which is most often not what was intended by the user. By disabling movement along this direction, this problem is solved.

In the third phase, no head tracking is involved and all objects will be shown from a fixed perspective. This time however, the obstructing plane may be moved left and right on the screen similar to moving objects. This allows the user to select objects behind the plane by moving the plane to a different location, freeing the object for selection. For this phase, an icon was used that represented a cursor, indicating only mouse movement was allowed.

At the start of each test phase, a timer will be started that keeps track on how long it takes to select all 12 objects in order. The timer automatically stops when the last object is selected, and store the results for later retrieval.

A screenshot of the application as it was presented to the users is given in figure 3. It shows the blocking plane in the center, the timer and icon in the top right, and each of the 12 selectable objects, of which 2 are currently accessible (3 and 10) and 3 are selected (1,2 and 3). The icon shown is for the free head tracking phase.

![Figure 3: Screenshot of the test application](image)

5.4.2 Head tracking
The head tracking is performed using a Wiimote and a head mounted sensorbar. The Wiimote registers the locations of the IR beacons in the sensorbar and translates those to their position on the view screen of the Wiimote’s camera. This information can then be used by the application to determine the 3D position of the sensorbar. Because the Wiimote has a limited accuracy, the position that the application will use is the mean of the position reported in the current frame and in the previous frame to smoothen out sudden ‘jumps’ in the reported position.

5.4.3 Touch table
For this experiment, a touch table is used that features a 88x66cm screen with a resolution of 1280x800 pixels. It is based on optical recognition using IR lights that illuminate objects above on the surface, but this feature is unused. To display the interface, it uses a projector that was mounted beneath the surface of the table, using a mirror to reflect the projected image onto the surface. A camera that could sense IR light was also mounted underneath the surface, and two IR lights were placed below the surface, one on each side.

However, the touch sensing mechanism of this table at its current state proved to be too unreliable to get consistent test results. Either a touch would go unnoticed, or it would be mapped to the wrong location. As such, an alternative means of input was used.

5.4.4 Input
The input was done using a wireless keyboard that features a trackball style mouse pointer device, and two mouse buttons in the top left and -right corners.

5.5 User Test
Before the start of each test, all users are given an explanation of the system, and the test they were about to undergo. Any questions they have will be answered before the test is started.

The tests consists of a user having to select 12 objects displayed on the screen in order. This has to be done for each combination of testing phase and dataset, meaning that in total 12 tests are performed (3 phases and 4 datasets). The user is instructed to perform the task as quickly as possible, in order to determine if a method of input is more efficient for completing the task than another.

Before the actual times are recorded, users are allowed to get used to the application first by performing a few try-out tests. They can use the application as in a real test, but the resulting times are not recorded. This is done to minimize any learning effect because of unfamiliarity with the input methods of the interface.

5.6 Interview
When the test of the application is finished, users are presented with an interview that should help determine how the users experienced the head tracking and how they would like to see it implemented in future applications. It also allows them to give feedback on the test, in order to determine if there were unforeseen issues that might affect the results or simply allow them to state their opinion or give any comments.

The interview started with three closed questions that would profile their experience with various aspects of this type of system, with answers ranging from ‘none’ to ‘throughout the day’. It was then followed by 9 open questions about their personal experience with the system.

6. RESULTS
In total, six users performed the test and answered the interview. The user group consisted of five males and one female, of which none had any experience with head tracking applications. All users used a graphical user interface on at least a daily basis, and five out of six users had a daily experience with 3D applications (one user had experience on a weekly basis).

Most users indicated that they quickly knew how to use the head tracking, but had difficulties adjusting to the trackball. All of them indicated that they had little or none experience with such an input method.

The results for the free head tracking are a mean time of 46.25 seconds, with an standard deviation of 8.33 seconds. For limited head tracking phase these are a mean of 36.86 seconds and a standard deviation of 2.32, and for the fixed perspective phase a mean of 35.28 seconds and a standard deviation of 5.08 seconds. These results are also shown in table 6 (Phase 1,2 and 3 are the
free head tracking, limited head tracking and fixed perspective phase respectively. Dataset 1 through 4 are the 4 distributions of objects.

These timings can be used to determine the efficiency of the head tracking input methods compared to that of the fixed perspective method. A significant difference in mean times (the difference is larger than the standard deviation) means that one method clearly performs better or worse than the other. Better performance is defined by a lower mean time to complete the task.

These results are the answer to the question “How can improvements in user interaction, due to adding depth perception to touch table interaction, be measured?” By determining the efficiency of the input methods the improvement can be directly measured. The answers from the interview can then augment this data.

### 6.1 Comparing between head tracking methods

The results show that there is a difference between free head tracking and limited head tracking in favor of limited head tracking. The difference falls outside the standard deviation, indicating there is a decrease in the completion time when using limited head tracking as opposed to free head tracking.

There are possibly two causes for this effect:

- The Wiimote has only limited accuracy when reading depth values, since the accuracy is inversely proportional to the distance. Since the user is more than two meters away, the reduced accuracy causes a slight change in distance (for example, when the user moves towards the table for a better view) to be translated in a jump on the perceived distance, causing ‘jerky’ behavior. This makes it harder for the user to position the cursor on the object to be selected, as it will constantly shift across the screen.

- When moving sideways, the user will not always move exactly parallel to the table, causing a increased or decreased (perceived) distance from the Wiimote. This prevents the user from correctly predicting the movement of the object due to changed perspective. Predictive cursor movements may therefore be more inaccurate since the expected movement (purely parallel) does not match the effective movement.

### 6.2 Comparing head tracking to fixed perspective

When comparing between head tracking based input and fixed perspective input, it is clear that the results from the limited head tracking mode are similar to those of the fixed perspective mode. This leads to believe that although the fixed perspective mode is faster than the free head tracking mode, there is no difference between limited head tracking mode and fixed perspective mode.

When asked, the opinion of the users about the performance of both systems differed. Four out of six users said that they did not think that the use of head tracking improved the time taken to complete the task. However, three of those four did think that the performance would have been better if the system would have made use of the touch capabilities of the table instead of having to use a trackball.

### 6.3 Applications

Using the results from the interviews, combined with some literature on the subject, we can find the answer to the question: “What type of interface/application for touch tables allows depth perception to be of additive value?”

When asking the users what kind of application is suitable for head tracking, all of them answer computer games. The system indeed appears to be suitable for this purpose, as is shown in a demo that was presented at the PyCon 2008 conference [14]. Though no other answers were given, this does not have to mean there are no other uses.

Some users indicated that they felt that using head tracking improved their efficiency using the prototype, but they did not think that the prototype would be a useful application.

Applications that are more closely related to this research exist as well, although they currently use other ways to perform their head tracking. Possibilities are medical applications or design applications where 3 dimensional data is available [15].

### 7. CONCLUSION

In this paper, a setup was presented that combined head tracking using a Wiimote and a head mounted sensorbar with a touch table that has a horizontal surface. By tracking the position of the head and adjusting the perspective of the image projected on the table, a FishTank VR environment was created that allowed the user to interact with the system by moving around to adjust the view.

Based on the results of the user test and questionnaire, it can be said that interaction through head tracking is at least comparable to mouse driven input and given the right conditions can improve the user interaction. Although it does not appear to be very suitable for an object based interface such as in the prototype, it can be of valuable addition to other types of interfaces, such as described in the Applications section (section 6.3).

Therefore, based on the results of this research and given the answers of the sub questions, the main question can be answered:

“Does depth perception improve the user’s interaction with a touch table?”

Yes, given the right conditions head tracking can improve the user’s interaction with touch tables. Using the limited head tracking approach from the prototype, results were measured that were
equivalent to those of the fixed perspective interaction. In addition, a majority of the users indicated that they either experienced better performance, or thought it would be better if the application had used the touch interaction capabilities of the touch table.

### 7.1 Summary

As a result of this study we have determined that:

- The Wiimote combined with the sensorbar allows us to create a working head tracking system for use on a touch table (section 5.4.2).
- Head tracking can be of valuable addition to touch tables, being comparable to traditional mouse driven input.

### 7.2 Future work

As was indicated by the users, including touch support into the application may affect the efficiency in performing the tasks. Most users had trouble getting used to the trackball, and touch interaction may be a much more intuitive approach to selecting the various objects.

When working with free head tracking enabled, users often experienced ‘jerky’ behavior when updating the perspective because of the Wiimote’s limited accuracy. Using a filter such as a Kalman filter would most likely produce smoother results, allowing for a better experience. Future work could investigate how good such a filter would work.

Additionally, it may be worthwhile to determine exactly why the free head tracking performed worse than limited head tracking. Though limited head tracking is a simple means to improve performance, it has the cost of a degraded effect of immersion and freedom of movement in the application.

Finally, as the test did not make any use of the touch interaction of the table, it is not known how much the touch interaction would deteriorate the depth perception, as the user is unable to reach for the object located below the table’s surface. Performing this test with touching, it can be determined how well the the depth perception will work when the user interacts with the table’s surface.

### 7.3 Discussion

Originally the test was planned to be performed using the touch capabilities of the touch table, rather than the trackball. However, the touch driver proved to be too unreliable for proper testing (a touch would go undetected or be detected in the wrong location), so an alternative means of input was used. A more reliable touch detection would solve this issue (for example, using a table based on capacitive coupling instead of IR).

Another issue was that when the user went out of range of the Wiimote, sometimes erroneous position data was reported. This could result in a change in perspective causing the objects to appear very small or go out of the view. A possible solution is to ignore position values outside a valid range. However, the most ideal situation would be for the range of the Wiimote to be extended. Since the properties of the Wiimote cannot be changed, this can be done by either increasing the distance of the Wiimote from the user, or by adding multiple Wiimotes to the test setup.

### Acknowledgments

Our thanks go out to Wim Fikkert, Herman Koppelman and Yujia Cao, as well as Tom Brunijes and fellow students and participants in the study.

### REFERENCES


APPENDIX A: USER TEST RESULTS
Phase 1, 2 and 3 are the free head tracking, limited head tracking and fixed perspective phase respectively. Dataset 1 through 4 are the 4 distributions of objects. All values are the time taken to complete the given task (defined by the phase and dataset) in seconds.

Table 2: Data user test 1

<table>
<thead>
<tr>
<th>test results (in seconds)</th>
<th>Dataset 1</th>
<th>Dataset 2</th>
<th>Dataset 3</th>
<th>Dataset 4</th>
<th>mean</th>
<th>stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase 1</td>
<td>38.38</td>
<td>48.17</td>
<td>66.37</td>
<td>51.37</td>
<td>51.07</td>
<td>11.6</td>
</tr>
<tr>
<td>Phase 2</td>
<td>34.12</td>
<td>38.1</td>
<td>48.73</td>
<td>37.23</td>
<td>39.55</td>
<td>6.36</td>
</tr>
<tr>
<td>Phase 3</td>
<td>74.1</td>
<td>31.45</td>
<td>38.73</td>
<td>35.91</td>
<td>45.05</td>
<td>19.6</td>
</tr>
<tr>
<td>mean</td>
<td>48.87</td>
<td>39.24</td>
<td>51.28</td>
<td>41.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>stdev</td>
<td>21.96</td>
<td>8.42</td>
<td>13.99</td>
<td>8.57</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Data user test 2

<table>
<thead>
<tr>
<th>test results (in seconds)</th>
<th>Dataset 1</th>
<th>Dataset 2</th>
<th>Dataset 3</th>
<th>Dataset 4</th>
<th>mean</th>
<th>stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase 1</td>
<td>49.23</td>
<td>51</td>
<td>92.9</td>
<td>41.89</td>
<td>58.76</td>
<td>23.1</td>
</tr>
<tr>
<td>Phase 2</td>
<td>42.79</td>
<td>27.33</td>
<td>36.29</td>
<td>38.74</td>
<td>36.29</td>
<td>6.55</td>
</tr>
<tr>
<td>Phase 3</td>
<td>40.51</td>
<td>27.56</td>
<td>33.21</td>
<td>34.43</td>
<td>33.93</td>
<td>5.31</td>
</tr>
<tr>
<td>mean</td>
<td>44.18</td>
<td>35.3</td>
<td>54.13</td>
<td>38.35</td>
<td></td>
<td></td>
</tr>
<tr>
<td>stdev</td>
<td>4.52</td>
<td>13.6</td>
<td>33.61</td>
<td>3.75</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4: Data user test 3

<table>
<thead>
<tr>
<th>test results (in seconds)</th>
<th>Dataset 1</th>
<th>Dataset 2</th>
<th>Dataset 3</th>
<th>Dataset 4</th>
<th>mean</th>
<th>stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase 1</td>
<td>41.96</td>
<td>42.22</td>
<td>50.48</td>
<td>31.72</td>
<td>41.6</td>
<td>7.68</td>
</tr>
<tr>
<td>Phase 2</td>
<td>37.27</td>
<td>33.91</td>
<td>50.34</td>
<td>51.56</td>
<td>43.27</td>
<td>8.99</td>
</tr>
<tr>
<td>Phase 3</td>
<td>43.07</td>
<td>33.39</td>
<td>41.68</td>
<td>33.29</td>
<td>37.86</td>
<td>5.25</td>
</tr>
<tr>
<td>mean</td>
<td>40.77</td>
<td>36.51</td>
<td>47.5</td>
<td>38.86</td>
<td></td>
<td></td>
</tr>
<tr>
<td>stdev</td>
<td>3.08</td>
<td>4.95</td>
<td>5.04</td>
<td>11.03</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5: Data user test 4

<table>
<thead>
<tr>
<th>test results (in seconds)</th>
<th>Dataset 1</th>
<th>Dataset 2</th>
<th>Dataset 3</th>
<th>Dataset 4</th>
<th>mean</th>
<th>stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase 1</td>
<td>41.2</td>
<td>43.34</td>
<td>46.34</td>
<td>44.59</td>
<td>43.87</td>
<td>2.16</td>
</tr>
<tr>
<td>Phase 2</td>
<td>39.69</td>
<td>37.94</td>
<td>30.59</td>
<td>41.5</td>
<td>37.43</td>
<td>4.79</td>
</tr>
<tr>
<td>Phase 3</td>
<td>27.47</td>
<td>28.87</td>
<td>32.37</td>
<td>27.37</td>
<td>29.02</td>
<td>2.34</td>
</tr>
<tr>
<td>mean</td>
<td>36.12</td>
<td>36.72</td>
<td>36.43</td>
<td>37.82</td>
<td></td>
<td></td>
</tr>
<tr>
<td>stdev</td>
<td>7.53</td>
<td>7.31</td>
<td>8.63</td>
<td>9.18</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Data user test 5

<table>
<thead>
<tr>
<th>test results (in seconds)</th>
<th>Dataset 1</th>
<th>Dataset 2</th>
<th>Dataset 3</th>
<th>Dataset 4</th>
<th>mean</th>
<th>stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase 1</td>
<td>45.79</td>
<td>43.37</td>
<td>38.73</td>
<td>34.37</td>
<td>40.57</td>
<td>5.06</td>
</tr>
<tr>
<td>Phase 2</td>
<td>27.63</td>
<td>33.85</td>
<td>23.95</td>
<td>28.15</td>
<td>28.4</td>
<td>4.09</td>
</tr>
<tr>
<td>Phase 3</td>
<td>35.75</td>
<td>31.31</td>
<td>33.54</td>
<td>32.97</td>
<td>33.39</td>
<td>1.83</td>
</tr>
<tr>
<td>mean</td>
<td>36.39</td>
<td>36.18</td>
<td>32.07</td>
<td>31.83</td>
<td></td>
<td></td>
</tr>
<tr>
<td>stdev</td>
<td>9.1</td>
<td>6.36</td>
<td>7.5</td>
<td>3.26</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7: Data user test 6

<table>
<thead>
<tr>
<th>test results (in seconds)</th>
<th>Dataset 1</th>
<th>Dataset 2</th>
<th>Dataset 3</th>
<th>Dataset 4</th>
<th>mean</th>
<th>stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase 1</td>
<td>35.98</td>
<td>40.31</td>
<td>55.45</td>
<td>34.88</td>
<td>41.66</td>
<td>9.49</td>
</tr>
<tr>
<td>Phase 2</td>
<td>34.4</td>
<td>32.77</td>
<td>41.41</td>
<td>36.4</td>
<td>36.25</td>
<td>3.75</td>
</tr>
<tr>
<td>Phase 3</td>
<td>33.67</td>
<td>34.31</td>
<td>32.49</td>
<td>29.36</td>
<td>32.46</td>
<td>2.2</td>
</tr>
<tr>
<td>mean</td>
<td>34.68</td>
<td>35.8</td>
<td>43.12</td>
<td>33.55</td>
<td></td>
<td></td>
</tr>
<tr>
<td>stdev</td>
<td>1.18</td>
<td>3.98</td>
<td>11.57</td>
<td>3.7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>