Abstraction of Interaction Techniques and Devices

Vicky Pagnaer

promotor :
Prof. dr. Chris RAYMAEKERS

co-promotor :
Prof. dr. Karin CONINX

Universiteit Maastricht

Eindverhandeling voorgedragen tot het bekomen van de graad
Master in de informatica human-computer interaction

universiteit hasselt

Abstract

As three-dimensional virtual environments gain in popularity, more and more devices and techniques are being developed to interact with these environments. Developers have no fixed method for choosing the devices and techniques to use in their applications. This thesis presents a tool, ABITID or Abstraction of Interaction Techniques and Interaction Devices, to propose the best matching devices and techniques depending on the developer's requirements.

Ideally, a single choice of device or technique would give a single answer of the best match possible. However, as choosing techniques and devices for an application depends on so many factors, many of which aren't easily measured, ABITID rather returns a list of possibilities that most closely match the requirements for the application.

ABITID is based on the abstraction of interaction techniques and devices into taxonomies. Abstracting devices and techniques gives a better understanding of how they can be matched. From this understanding we propose an algorithm, which matches a particular device with the most appropriate techniques and vice versa. This algorithm is implemented in ABITID, which works upon a self-made database of common interaction techniques and devices.

Acknowledgements

Mijn thesis zou nooit tot een goed einde gebracht zijn zonder de hulp van enkele belangrijke personen. Via deze weg wil ik hen graag bedanken.

Eerst en vooral de mensen die me begeleid hebben tijdens het maken van deze thesis, mijn promotor Prof. Dr. Chris Raymaekers, mijn co-promotor Prof. Dr. Karin Coninx en mijn begeleider Tom de Weyer. Ze hebben me voorzien van tips en suggesties gedurende de loop van mijn thesis.

Verder wil ik ook graag mijn vriend Lode bedanken voor het nalezen van mijn tekst en zijn morele steun; hij was ook altijd bereid me te helpen indien er problemen waren.

En als allerlaatste natuurlijk mijn ouders, want zonder hun zou ik nooit deze opleiding hebben kunnen volgen en succesvol afronden.

Vicky Pagnaer

Samenvatting

Drie-dimensionale virtuele omgevingen (3D VO's) spreken nog steeds tot de verbeelding. Waar er origineel vooral sprake was van onderzoek naar grafische verbeteringen, zijn er nu tal van domeinen in betrokken. Zo ook het domein van de Mens-Machine Interactie, dat zich op het vlak van 3D VO's bezighoudt met de manier waarop gebruikers met deze omgevingen kunnen werken.

De interactie tussen gebruikers en 3D VO's bestaat uit verschillende delen. Ten eerste zijn er de apparaten die een gebruiker hanteert. Hiervan zijn er talrijke voorbeelden, vaak speciaal ontworpen voor 3D VO's. Deze apparaten sturen invoer door naar het computersysteem. De invoer wordt daar omgezet naar virtuele acties op basis van interactietechnieken, die bepalen hoe de invoer geïnterpreteerd wordt. Ook van interactietechnieken zijn er talrijke voorbeelden.

Omdat er tegenwoordig zoveel mogelijke apparaten en technieken bestaan, wordt het voor ontwerpers van 3D VO's steeds moeilijker om hieruit efficiënte en gebruiksvriendelijke keuzes te maken. Bovendien moeten de gekozen apparaten en technieken ook goed samenwerken. Het doel van deze thesis is dan ook om voor de 3D VO-ontwerper een methode te vinden die deze keuze gemakkelijker maakt.

Hiervoor worden eerst interactietechnieken en apparaten bestudeerd. Uit dit onderzoek worden essentiële eigenschappen geabstraheerd. De belangrijkste eigenschappen worden in een taxonomie gezet, terwijl andere eigenschappen apart beschouwd worden.

Voor interactietechnieken wordt er eerst gekeken naar de taak waarop ze van toepassing zijn. Als een gebruiker interageert met een 3D VO, heeft hij of zij namelijk altijd een doel voor ogen. In 3D VO's komen enkele taken altijd terug, zoals navigatie, object selectie, object manipulatie en applicatie- of systeemcontrole. Dit is de basis voor de taxonomie. Deze wordt verder onderverdeeld op basis van andere eigenschappen. Voor navigatie wordt er bijvoorbeeld gekeken naar het soort navigatietaak dat een gebruiker kan uit-

voeren. Misschien wil hij gewoon wat rondkijken, misschien is hij actief op zoek naar een plaats of object in de 3D VO, of misschien wil hij een specifiek standpunt innemen.

Andere eigenschappen van interactietechnieken zijn bijvoorbeeld het soort feedback dat gegeven wordt (visueel, auditief...), of of de techniek relatief of absoluut beweegt ten opzichte van de reële omgeving.

Apparaten kunnen niet gestructureerd worden in een taxonomie op basis van interactietaken. Veel apparaten zijn namelijk bruikbaar voor meer dan één taak. Apparaten zijn ook veel complexer, met meer eigenschappen. Daarom wordt er voor apparaten een bestaande taxonomie gekozen, die verschillende eigenschappen in beschouwing neemt, zoals de bewegingsvrijheid en het soort invoer (beweging, kracht, draaiing...). Verdere eigenschappen zijn onder meer de samenhang van de bewegingsvrijheid (*separability and integrality*) en of een apparaat relatief of absoluut beweegt.

De geabstraheerde eigenschappen van interactietechnieken en apparaten worden dan vergeleken, om te bepalen of ze een positieve invloed hebben op elkaar. Als een techniek een apparaat positief beïnvloedt of omgekeerd, betekent dit immers dat die techniek en dat apparaat bij elkaar passen (of in het geval van negatieve invloed, zeker niet bij elkaar passen).

Op basis van dit principe kunnen we voor een gegeven techniek een lijst van apparaten evalueren, of voor een gegeven apparaat een lijst van technieken evalueren. Het resultaat van deze evaluatie is een lijst van apparaten (of technieken) die gequoteerd zijn op hun compatibiliteit met de gegeven techniek (of het gegeven apparaat).

We leggen het algoritme uit voor een gegeven techniek en een lijst van apparaten. Voor elke eigenschap waarvan bepaald is dat deze een invloed heeft op de compatibiliteit tussen technieken en apparaten, wordt een scorelijst opgesteld. Deze scorelijst bevat alle mogelijke waardes voor de eigenschap en een score die de compatibiliteit reflecteert. Dus, als de waarde van de eigenschap een positieve invloed heeft, wordt er een hoge score gegeven. In het andere geval, een lage score. Door hoge scores te geven aan waardes die de compatibiliteit tussen een techniek en een apparaat verhoogt, wordt het apparaat hoger geëvalueerd.

Zo worden voor alle apparaten alle eigenschappen gequoteerd en het apparaat dat het beste bij de gegeven techniek past, op basis van de scorelijsten, zal de hoogste totaalscore hebben. Om rekening te houden met de wensen van de 3D VO-ontwerper, worden verstelbare gewichten toegekend aan de scores, zodat de ontwerper kan beslissen welke eigenschappen hij belangrij-

ker vindt.

Tot nu toe hebben we ons voornamelijk beziggehouden met invoerapparaten. Deze zijn namelijk de basis voor interactie met een 3D VO. Uitvoerapparaten zijn echter ook van belang, vooral omdat ze de gebruiker de resultaten van zijn acties laten merken. Een 3D VO-ontwerper moet natuurlijk ook uitvoerapparaten kiezen. Om hier ook bij te helpen is er een tweede, gelijkaardig algoritme opgesteld, dat uitvoerapparaten evalueert op basis van de resultaten van het eerste algoritme.

Bij deze algoritmes zijn er enkele belangrijke kwesties die beschouwd moeten worden. De belangrijkste hiervan is het gebrek aan aangepast onderzoek. Er is nog maar weinig onderzoek gedaan naar de specifieke omstandigheden die vereist zijn voor de algoritmes. Weinig is geweten over waarom apparaten en technieken goed of minder goed samen passen. Ook belangrijk is de mening van de 3D VO gebruikers, die echter moeilijk gemeten kan worden.

De twee besproken algoritmes zijn geïmplementeerd in een toepassing, ABITID. Deze toepassing werkt op basis van een database, waarin alle informatie over technieken en apparaten en hun eigenschappen wordt bijgehouden.

Deze toepassing laat de 3D VO-ontwerper toe om voor verschillende technieken of apparaten en met verschillende gewichten het algoritme toe te passen en de resultaten ervan op te slaan. Ook zijn verschillende database types ondersteund. Uit de informele evaluatie blijkt dat het algoritme en het database schema goed werken met de tot nu toe gevonden eigenschappen, maar dat verder onderzoek zeker nodig is om hier meer duidelijke en precieze informatie over te verkrijgen. Ook blijkt dat de toepassing te weinig flexibel is om nieuwe informatie over eigenschappen of over het algoritme gemakkelijk te kunnen opnemen.

We kunnen voor deze thesis concluderen dat het zeker handig is voor 3D VO-ontwerpers om verschillende apparaten en technieken te kunnen te vergelijken op basis van hoe goed ze bij een andere techniek of apparaat passen. Er is ook nog veel nood aan verder onderzoek naar de verschillende elementen die het algoritme bepalen, zoals de eigenschappen van technieken en apparaten. Ook kan er onderzoek gedaan worden naar de bijeenpassendheid van verschillende technieken onderling of verschillende invoerapparaten onderling, om te bepalen welke combinaties nodig zijn om een hele 3D VO te voorzien van de nodige interactiemogelijkheden.

Toekomstig werk zal zeker verder onderzoek inhouden, maar ook aanpassingen aan de implementatie zijn nodig. Zo kan deze meer flexibel gemaakt worden, om aanpassingen in het algoritme gemakkelijk te kunnen ondersteu-

nen. Ook kunnen extra opties toegevoegd worden om meer informatie en mogelijkheden te verschaffen aan de 3D VO-ontwerpers. We hopen dat dit onderzoek zal bijdragen tot het ontwerpen en meer toegankelijk maken van 3D Virtuele Omgevingen.

Contents

List of Figures

Chapter 1

Introduction

When three-dimensional virtual environments (3D VEs) first became popular, research was focused mostly on graphics. The aims were then to provide better image quality and faster rendering. Throughout the years, 3D VEs have evolved to amazingly realistic environments, as evidenced most clearly in computer games and life-or-death applications. Over the last few decades, the focus of 3D research has shifted to include the search for optimal user interaction. As in most research domains, human-computer interaction has proved to be of as much importance. After all, what good is a realistic environment if you can't interact with it?

Research into 3D interaction and user interfaces has so far turned out many interesting results, among which a variety of input and output devices, interaction techniques and design guidelines. As these lists of possibilities grow, it becomes more and more difficult for developers to make appropriate choices for their own 3D applications.

1.1 Thesis Aim

Ideally, a developer should be able to choose either an interaction technique or a device and find the best matching device or technique to go with it.

However, techniques and devices don't match perfectly. Some techniques can be implemented with many devices while others were made for a specific device. On the other hand, devices may be used for many different techniques, but may fit better with some than with others.

The aim of this thesis is to present a tool for matching a device with a technique or vice versa. The process for matching techniques and devices isn't clear cut, however, and often not a single solution will be given, but a list of possibilities that most closely match the requirements of the 3D

VE developer.

Furthermore, it isn't sufficient to just work on a case-to-case basis, as many devices match each other closely or can perform similar actions. Likewise plenty of techniques differ only in minor points and could be grouped together for part of the matching process. Therefore this thesis also considers the abstraction of interaction techniques and devices. A solid understanding of an interaction technique as an abstract concept can improve the process of matching a particular technique with a device.

Thus, the aim is to provide certain levels of abstraction to improve the matching process. These levels of abstraction will be combined in two taxonomies, one for both devices and techniques.

1.2 Thesis Overview

In this thesis, we will first consider the abstraction of interaction techniques (see Chapter 2) and interaction devices (see Chapter 3). With the information resulting from these chapters, we will then propose a matching algorithm (see Chapter 4).

In Chapter 5 we will discuss the implemented tool for this thesis: ABITID. This tool will be based upon both the taxonomies from Chapters 2 and 3 as well as the algorithm from Chapter 4. Chapter 6 will present the conclusions reached from this thesis.

Chapter 2

Interaction Techniques

2.1 Introduction

Computing systems would not be of much use without the ability to interact with them. From simple text commands in the early years of computers to WIMP (Window, Icon, Menu, Pointing device) [Art] in our current desktop environments, interaction techniques have always been needed to make these systems work for us.

As 3D Virtual Environments (3D VEs) have been gaining importance, so has the need for interaction techniques that suit them. Research and experience have shown that existing techniques cannot simply be used in a completely different environment.

In this chapter, we will first explore the term interaction technique and how it applies to 3D VEs. We will also endeavor to explain how new techniques originate.

After these explanations, a comprehensive taxonomy will be assembled from both known taxonomies and requirements pertaining to the matching tool presented later in this thesis. Other properties and issues concerning interaction techniques will be discussed in section 2.3.

We will end with an overview of the most commonly known 3D interaction techniques, their properties and their placement in the newly created taxonomy.

2.1.1 What is an Interaction Technique?

According to [BKLJP04] an interaction technique is:

> *A method allowing a user to accomplish a task via the user in-*

> *terface.*

Thus, interaction techniques are the means to achieve our ends on a computing system. More specifically, the tasks which need to be accomplished by 3D VE users are usually complex and need a succession of techniques that each complete a certain subtask. Furthermore, each interaction technique is most often comprised of a series of actions which need to be performed in a certain order and with certain conditions to be completed correctly.

For example, the task of creating a new text document requires the user to first execute the subtask of selecting the word processing program. This selection task in a 2D desktop environment is accomplished by the point-and-click interaction technique, which—as it says—consists of two actions: pointing the cursor to the correct place on the desktop and clicking the mouse to confirm the selection.

The definition of [BKLJP04] also includes the following:

> *An interaction technique includes both hardware and software components.*

This implies that separating input and output devices from the actual techniques is not immediately obvious. Indeed, some interaction techniques were made with specific devices in mind. Furthermore, this also indicates that certain techniques fit better with some devices than with others. In this chapter, however, we will concentrate purely on the techniques themselves. We will discuss how interaction devices and techniques fit together in Chapter 4.

As a final note in this explanation, it is important to realize the difference between an interaction technique and an interaction metaphor. An interaction metaphor is the conceptual idea behind the technique, and is usually a comparison to some concept of the real world.

For example, the Virtual Hand interaction technique (see figure 2.1), which allows movement of a hand-shaped cursor in six degrees of freedom, is based on the metaphor of a human hand which can move in the same manner.

2.1.2 Designing Interaction Techniques

It is not in the scope of this thesis to fully explain how interaction techniques are designed. However, there are a few common methods to obtain new techniques.

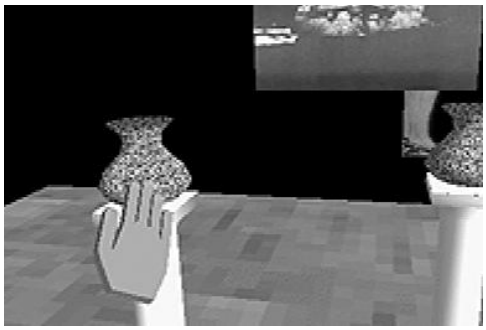


Figure 2.1: Virtual Hand [PWBI98].

A lot of techniques are based on a real-life metaphors. On one hand, they are usually easier for the user to understand, because they are based on an already familiar concept. On the other, real life concepts can have limitations that are much more pronounced in 3D VEs.

As an example, consider the metaphor of walking to navigate through a 3D VE. Walking is easily understood, as most of us move around in this manner everyday. However, in a large environment, navigating by walking can be slow and tedious.

Another way to obtain techniques is by magical metaphors. The possibilities here are virtually endless. The advantage of basing interaction techniques on magical concepts is that with a little imagination, most real-life limitations can be easily overcome. On the other hand, this might complicate the interaction. These concepts can be more difficult to grasp and some users might be opposed to actions that are impossible in the real world.

Teleportation is such a technique, based on popular fiction. It provides fast and easy navigation, but it can disorient the user upon arrival in a suddenly different scene. As mentioned, magical techniques can be easily adapted to overcome drawbacks. In this case, the user could first be provided with an overview of the area around the new scene, while slowly zooming in, giving the sensation of appearing above the destination and falling into place.

Finally, 3D interaction techniques can also be found by adapting existing techniques from other areas. In this case it is important to consider the original application of the technique and how it differs from a 3D VE. It might be possible, for example, to adapt a two-dimensional technique to function in a 3D VE. However, it is important to provide for the additional dimension and the extra degrees of freedom.

During the process of designing interaction techniques, there are also certain issues that must be kept in mind and that are of interest here.

Firstly, a technique must be attuned to the task it is being made to fulfil. In selection tasks, it is often unnecessary to incorporate rotation, as positioning can be sufficient to fulfill the task. The interaction technique should at the very least have the amount of actions necessary to complete the task. It is possible to provide more actions, but although that gives the user more alternatives or extra features, it might also complicate the task.

Secondly, although it is possible to consider a technique purely as a set of actions, we usually also keep in mind the way those actions are performed by the user. A technique has to be executable by the devices available to the user. As already mentioned, we will explore the connection between techniques and devices more in Chapter 4.

How interaction techniques are designed can give us a better perspective on exactly what properties and issues are of importance in choosing an interaction technique for a 3D VE. We will explore this further after we have derived a comprehensive interaction technique taxonomy in the following section.

2.2 Interaction Technique Taxonomies

There have been many researchers already who have attempted to subdivide interaction techniques in different categories, depending on one or other property. In this section, we will view some of the most common taxonomies off of which we will base our own taxonomy to use in ABITID, the matching tool which we present in Chapter 5. With our own taxonomy, we attempt to categorize the design space of interaction techniques to a more manageable whole.

Section 2.3 will give an overview of the properties of interaction techniques that were not included in our taxonomy. These properties would have cluttered the taxonomy and can be more easily seen as constraints or requirements for the matching process.

Most interaction technique taxonomies are based on either the tasks these techniques are used for or on the metaphors upon which the techniques are based. We will first discuss task taxonomies to obtain the first levels of our own interaction technique taxonomy.

2.2.1 Task taxonomies

Often when discussing interaction techniques, tasks such as selection, manipulation and navigation are mentioned, among others. True interaction tasks are usually more complex and depend on the purpose of the environment, but can be accomplished by using some combination of basic recurring tasks. However, are the above three tasks basic?

The three task taxonomies we will discuss are quite different and while they all mention selection, manipulation and navigation, they differ in their interpretation of these tasks.

Note that the idea of categorizing by interaction tasks follows the definition of an interaction technique given in section 2.1.1, which is why it is a commonly accepted way of sorting techniques. For this same reason, the first level of our own taxonomy will also be task-based.

One of the taxonomies is used in [BKLJP04]. The authors consider the following tasks to be at the same basic level:

- manipulation
- navigation (travel - wayfinding)
- system control
- symbolic input
- (modeling)

The second taxonomy is used to give an overview of interaction techniques in [Han97]. Hand recognizes three basic tasks in most—more complex—tasks:

- object manipulation
- viewpoint manipulation
- application control

The last taxonomy is also used to present different interaction techniques, but by [Min95]. Mine mentions four basic tasks and one derivative:

- movement
- selection
- manipulation

- scaling
- (virtual menu and widget interaction)

Although some similarities are immediately visible, so are most differences. It is important to understand what exactly is meant by these tasks to be able to choose which ones to include in our own taxonomy. In this level, we aim for a clear subdivision of interaction techniques, that both makes the subsequent levels more manageable and that will provide the developer with a cognitively easier overview of techniques in our tool.

All three taxonomies mention some form of user movement (*navigation*, *viewpoint manipulation* and *movement*). While Mine and Hand only consider the physical aspect, Bowman et al. also consider a cognitive aspect—wayfinding. However, the 'techniques' they assign to this category don't fit their definition of an interaction technique. Rather, they are design guidelines and environmental aids and additions that ease the travel task and the feeling of immersion.

On the other hand, while both Bowman et al. and Hand agree that user movement is more than just movement and positioning of the viewpoint (they include adjusting zoom factor, field of view size and other factors as part of this task as well), Mine considers only basic user movement. According to him, movement can be solely defined by the speed and direction of motion of the user.

In our own taxonomy, we will consider navigation to be solely the movement of the user, as in Mine's definition. The other factors that Hand and Bowman et al. include can be seen as adjustments to the virtual world or the perception of it, and will thus fall into the category of system control, which we will discuss momentarily. Also, the cognitive task of wayfinding does not fit Mine's basic view of traveling and will not be considered in our taxonomy. Distinctions of the navigation task will be discussed when we compose the second level of our taxonomy.

The second item which is shared is object manipulation (*manipulation*, *object manipulation* and *manipulation, selection and scaling* respectively). Opinions differ greatly here. Hand categorizes all forms of manipulation in a single item, including selection, translation, rotation, scaling, creating, editing, deleting and others. Bowman et al. only consider selection, translation and rotation. Other manipulations are claimed to be accessible through the three basic manipulations upon virtual widgets, for example translating the handles of an object's bounding box will scale that object. Furthermore, Bowman et al. mention modeling as a task, although they haven't included

it in their discussions. The modeling task is described as the creation and modification of geometric objects. According to Hand, however, modeling is a complex task which can be accomplished by simple object manipulations. Mine then, defines manipulation as solely translation and rotation, and considers both selection and scaling as separate tasks.

Manipulation will, in our own taxonomy, only include translation and rotation. Furthermore, we will have a selection task separate from this manipulation definition. Translation and rotation are the most essential and widespread forms of manipulation and have been summarily included in this task. The other forms of manipulation (such as scaling, creating...) were not included as essential to the manipulation task. They can be accomplished by either system control or through a combination of selection, translation and rotation. For example, Hand [Han97] mentions some techniques that use selection, translation and rotation of 3D objects, called 'virtual tools', to accomplish more complex manipulations.

The selection task was separated from the manipulation task because although they are interrelated, the tasks are vastly different. While manipulation changes an object or its position and orientation, selection changes the focus of the application to the object. From a more practical point of view, selection cannot be included with manipulation either, because it is a task that always *precedes* manipulation.

Lastly, there is the task of application or environment control (in the forms of *system control and symbolic input*, *application control* and *virtual menu and widget interaction*). Surprisingly, application control and system control—though defined by different authors—have the same task description, they namely describe interactions which aren't directly part of the 3D Virtual Environment, but change the environment from the outside. This can include a variety of different tasks, such as field-of-view adjustment, object creation... It can be argued here that these tasks, just like more complex manipulations, can be accomplished through the manipulation of 'virtual tools' in the environment. Especially as using virtual objects to control the system instead of 2D techniques should improve the immersion in the 3D VE. Following that logic, the task should not appear in our taxonomy.

However, we should not only consider 'virtual tools' as a means of application control. Bowman et al. included *symbolic input* as a task in their overview. As the focus of research has mostly been on object and viewpoint manipulation, symbolic input doesn't have the well-known techniques the other tasks do. However, as words and numbers play such large roles in both real life and 2D environments, symbolic input will surely be needed in 3D VEs as well. As it is not immediately connected to the 3D environment,

however, it might be considered under application control as well. Thus, we will finish the first level of our taxonomy with the category of application control.

In this subsection, we have reviewed the most common task taxonomies to deduce the first level of our own taxonomy. We chose to start our taxonomy with task categories because not only is it a widely accepted division of interaction techniques, as it follows the definition given in 2.1.1, it will also provide a more easily understood overview in our tool ABITID (see Chapter 5). The final first level of our taxonomy looks as follows:

- navigation
- manipulation
- selection
- application control

In the further levels of our taxonomy, we aim to divide techniques both logically and in relation to the interaction devices they will be matched with. The next sections further specify our first taxonomy level.

2.2.2 Navigation Taxonomy

Bowman et al. [BKLJP04] cover a wide range of properties of the navigation task, which should be considered for use in our taxonomy.

One classification distinguishes between active versus passive techniques and between physical versus virtual techniques. Active techniques require input from the user, while passive techniques are executed by the system. Combinations are possible as well. Physical techniques require physical travel from the user's body, while in virtual techniques the body remains stationary—except to use devices—while the virtual viewpoint moves. Again combinations are possible.

However, this two-by-two classification isn't suitable to our goals. When considering the correct input devices for certain techniques, it requires that these techniques *need* input devices, thus passive techniques aren't considered. In the case of mixed techniques, only the active parts need matching to a device, but for simplicity, we will call those techniques active as well. The other distinction uses input devices to separate techniques, namely whether the user uses his body or not. Either the technique would already be matched to a device (the user's body) or it wouldn't, in which case there still would be no further classification.

A second classification of Bowman et al. is based on metaphors. Again, this isn't suitable to our needs for various reasons. Firstly, the amount of metaphors could be endless, leading to an unnecessarily large taxonomy. Secondly, categorizing by metaphor is a mostly cognitive aid. Metaphors can be indicative to some input device—usually one on which the metaphor is based—but not with certainty the best possible match for each technique that is based upon it.

Bowman et al. also presented two of their own taxonomies, one focusing on the actions in the navigation task, another focusing on the level of user control. The second taxonomy doesn't suit our purposes for the same reason that the classification of active versus passive isn't used. We only consider the techniques or the parts of techniques that need user input, thus it isn't necessary to distinguish whether or not the user controls the movement, we only consider the cases where the user does.

The first taxonomy, based on the actions of which the navigation task consists, namely *target selection*, *acceleration selection* and *input conditions*, isn't used to categorize techniques, but to define them. This corresponds with Mine [Min95], who mentions *speed* and *direction of motion* as the two key parameters to define navigation. Furthermore, the action components with which a technique can be built aren't separated from input and output devices. For example, Bowman et al. mention *gestures* as a possible way to select the acceleration, and Mine suggests using the user's hand as a possible way to select the direction of motion.

Therefore, we consider another subdivision of Bowman et al., which centers around the goal for the navigation task. They distinguish three possible goals:

Exploration where the user doesn't have a specific goal, but gains knowledge of the environment

Search where the user has a specific destination

Maneuvering where the user wishes to perform a task which needs precise positioning (for example inspecting an object)

Mackinlay et al. [MCR90] distinguish four similar types of navigation.

General movement corresponds to Bowman's exploration goal

Targeted movement where the user moves with respect to a specific target, which corresponds to Bowman's search goal

Specified coordinate movement in which the user supplies the exact position and orientation of his destination

Specified trajectory movement in which the user supplies a path and is moved along that path.

The last two types are not addressed directly in Bowman's subdivision of the navigation task. However, there are a number of techniques that fit into those categories.
Along with Bowman et al.'s categories, they present a list of navigation characteristics that may influence the choice or design of a technique as well as the user's goal. These are:

- Distance to be traveled
- Number of turns in the path
- Visibility of the target from the starting location
- DOF (degrees of freedom) required for the movement
- Required accuracy
- Other primary tasks that take place during travel

However, these characteristics can also be joined with the three possible goals above to provide better defined subcategories.

Exploration usually covers longer distances, with many turns, but low accuracy. There is no target, thus no need for target visibility, and the user's goal is exploring, thus his focus shouldn't have to be on traveling.

Maneuvering usually covers short distances, with high accuracy and no elaborate path, thus not many turns. The user's focus is completely on the positioning of his virtual representation, but there may be a target towards which the user's movements are aimed.

The Search subtask isn't as well defined as the previous two. The target can be both far or close by, can be both visible from the start or not and can have both few or many turns. The required accuracy is low: as soon as the target is visible or within reach, the search task has been completed and should high accuracy be required, the user then has a new goal, maneuvering. During the Search task, however, the user's primary focus is on the environment, to be able to find the target.

If we combine these same characteristics with Mackinlay's navigation types,

we get the following results.

General movement is like exploration, and thus covers longer distances (although immediate surroundings might be explored as well). It requires many turns, low accuracy, with no need for target visibility and some other basic task during the traveling (exploring).

Targeted movement corresponds to the search subtask. Whether the target position is known by the user only makes a difference in the time the user needs to reach it and, depending on the environment, the number of turns needed to get there. Targets can be both close by or far off, and visible or occluded.

Specified coordinate movement does not require the user to move around, but lets him specify a certain point in the environment after which he is taken to that point by the system. Both the means of specification and of travel can be varied. Both distance and target visibility can be varied as well and the number of turns is irrelevant. The accuracy of the movement should be high, but it is required of the system, not of the user. As the system moves the user to his target, the user should be free to engage in other tasks.

Specified trajectory movement has the same principle as the previous type. The user specifies a path and the system moves the user along that path. The characteristics for this type have the same possible values as they do for specified coordinate movement.

Clearly, we can combine the last two into a more general type of navigation, *specified movement*. Mackinlay's other two types correspond to Bowman's exploration and search, so a combination of these categorizations can be made. As both the choice of techniques and that of devices are influenced by the user's navigation goal(s), our interaction technique taxonomy now looks as follows:

- navigation
  - exploration
  - search
  - maneuvering
  - specified movement
- selection
- manipulation

- application control

2.2.3 Manipulation Taxonomy

For manipulation techniques, Poupyrev's [PWBI98] taxonomy has always been a leading classification. We present it here briefly:

- Exocentric Techniques (World-in-Miniature, scaling...)
- Egocentric Techniques
  - Virtual Hand Metaphors ("classical", Go-Go...)
  - Virtual Pointer Metaphors (raycasting, aperture...)

However, the concerns that were raised in the previous section considering metaphors are valid here as well. There could be an endless list of metaphors if we'd decide to classify by that factor. Furthermore, the classification isn't exhaustive. Hybrid techniques have no place in it.

On the other hand, most egocentric manipulation techniques seem to be based on the two given metaphors, so with some slight modifications it might be a valid taxonomy to use.

Mine [Min95] presents another taxonomy for manipulation techniques, based on the parameters that define the manipulation task. Again, as mentioned in the previous section, these classifications are more suited for defining and creating techniques, rather than for classifying existing ones. Another similar taxonomy is presented by Bowman, Johnson and Hodges [BJH99]. Even though defining taxonomies aren't suitable in our own taxonomy, they contain a wealth of information about how interaction techniques are pieced together. This information can be useful when discussing constraints and requirements for our matching tool, see section 2.3.

As most papers discussing manipulation techniques either define their own taxonomy, or work with Poupyrev's, we will modify the last one to suit our needs. Because we do not wish an endless list of metaphors, we keep the most common ones available and add an *other metaphor* category. Another category will be added for the combination of metaphors, *hybrid techniques*. Also, we remove the specified techniques from the taxonomy.

We present the final subtaxonomy for manipulation as follows:

- Exocentric Techniques
- Egocentric Techniques

  - Virtual Hand Metaphors
  - Virtual Pointer Metaphors
  - Other Metaphors
- Hybrid Techniques

2.2.4 Selection Taxonomy

Because the selection task is so closely related to the manipulation task, a lot of techniques can be used to fulfil both. For example, the taxonomy by Poupyrev presented in the previous section was used to both test the manipulation and selection tasks.

For this task, it is Bowman that presents a classification based on the parameters that define selection, rather than classify the different techniques for that task. Mine divides selection techniques by distance: *local* and *at-a-distance*, while also presenting some specific techniques.

In De Boeck's paper [DBRC05], selection techniques are separated in four groups:

- ray-casting and cone-casting
- aperture technique
- direct manipulation techniques (Virtual Hand, image plane techniques...)
- speech

Apart from speech, these categories fit right in with the egocentric manipulation techniques of Poupyrev. Speech is not really considered as a technique as such, but as a collection of interpretation techniques that are all paired with a single specific device: the human voice box.

It is for obvious reasons that the manipulation taxonomy is so strongly linked to selection techniques. After all, selection precedes manipulation. The user, after having selected an object, will have to switch to manipulating that object. This is easiest if both techniques and devices for these two tasks are similar if not the same.

Most of the techniques for manipulation can be used for selection as well, perhaps with some modifications. To exploit this link with manipulation techniques, we will use the same categories in our taxonomy. Other factors, such as the *distance* used by Mine, will be discussed in section 2.3.

The selection subtaxonomy:

- Exocentric Techniques
- Egocentric Techniques
  - Virtual Hand Metaphors
  - Virtual Pointer Metaphors
  - Other Metaphors
- Hybrid Techniques

2.2.5 Application Control Taxonomy

As mentioned in subsection 2.2.1, application control concerns those techniques that change the environment itself. Symbolic input was included in this category for several reasons. Firstly, it has no direct link to the other tasks. It might be used in a technique that employs *specified target movement* as seen in subsection 2.2.2, but it still needs separate interaction techniques to be included in other tasks. Secondly, both application control and symbolic input have not garnered much research attention yet. However, they are both widely used in 2D environments and a lot of current 3D VEs use the known 2D techniques for these tasks.

This gives us an immediate set of categories for the application control task. As mentioned when discussing the first level of our taxonomy, application control is often accomplished through manipulation of virtual objects, which is done through 3D techniques. Another possibility is then the use of two-dimensional techniques. Hand [Han97] further mentions the mapping of 2D techniques to three dimensions. And lastly, techniques could be one-dimensional. Mine [Min95] mentions one-dimensional techniques, as a very simple means of controlling a small number of application functions.

Bowman et al. [BKLJP04] give a classification for application control techniques. However, this classification is very specific and doesn't separate techniques from devices. The examples they provide, though, are sorted by dimension as well.

Since dimension is the most distinctive property in other application control research, we conclude our application control subtaxonomy as follows:

- 1D techniques
- 2D techniques
- 3D techniques (excluding the manipulation of virtual objects)
- Other techniques

2.2.6 Proposed Interaction Technique Taxonomy

We have discussed many different classifications for the wealth of interaction techniques available for 3D VEs. For the major tasks, we have proposed taxonomies to ease the use of our tool, ABITID. We finish this section by giving a final overview of our complete taxonomy (see figure 2.2).

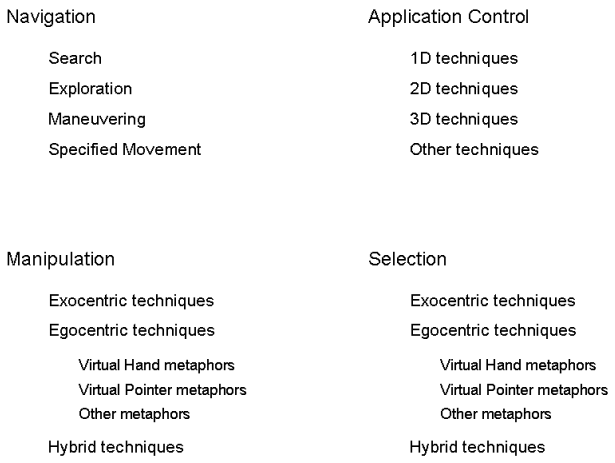


Figure 2.2: Complete interaction technique taxonomy.

2.3 Other Properties of Interaction Techniques

In the previous section, we have assembled a taxonomy for categorizing interaction techniques. The aim of this taxonomy was to assign a task and further categories to interaction techniques, so that they may be more easily matched to possible interaction devices (see Chapter 4).

However, this taxonomy alone can not completely describe all the facets and peculiarities of an interaction technique, or how and why it best fits with a certain device. Therefore we must look further, and discern other properties of interaction techniques. We do this separately from our taxonomy, because firstly, we do not wish to clutter the taxonomy with too many categories and secondly, because further properties of interaction techniques aren't clearly defined.

We can derive some other properties from the taxonomies discussed in the

previous section, but not much research has gone into an abstract and theoretical definition of an interaction technique. Furthermore, any properties we consider must bear an influence on the interaction devices they can or cannot be used with, otherwise the property will not be of use for the purpose of this thesis.

In this section, we will attempt to assemble a list of properties of interaction techniques, while in Chapter 4 we will consider whether or not the properties are useful to our matching goal.

2.3.1 Properties

As we had already deduced from section 2.1.1, one property of an interaction technique is the amount of actions it consists of. Another property is the degrees of freedom (DOF) an interaction technique uses. For example, Virtual Hand manipulation [PWBI98] consists of three actions—entering manipulation mode, manipulating an object and exiting manipulation mode—and works with 6DOF, while selection by raycasting [PWBI98] uses only 5DOF (rotation around the ray itself is not included) and two actions.

A next property is the types of feedback a technique provides. This can be related to output devices rather than input devices and will be further discussed in Chapter 4.

Another property to distinguish between techniques is whether the technique works relative or absolute to the input device used. This is an already established link between interaction techniques and devices. For example, in Augmented Reality, it is mostly so that the user's real position and orientation are also the user's virtual position and orientation, as the virtual environment overlaps the reality.

In the same vain, we can consider whether the received movement input is translated to virtual movement with a constant or variable function. As an example, consider moving a mouse forward to start moving forward in a virtual environment. In the first case, a constant function would force us to keep moving the mouse to keep going forward at the same speed. With a variable function, one possibility would be to move the mouse further forward, the faster you'd want to move in the virtual environment.

Finally, we can also consider as a property the range of an interaction technique, the distance at which a technique can be efficiently used. Some techniques work better at a distance, while some work better close. Consider for example the Image Plane Head Crusher technique (see figure 2.5). This

technique works better for selecting objects at a distance, because the user doesn't have to spread his fingers very wide to indicate the chosen object, which is smaller the further away it is.

2.4 Overview of Common Techniques

This section will present an overview of well-known techniques that fit into each task category.

2.4.1 Navigation Techniques

Walking

The simplest method of moving around in a virtual environment is Walking. It is a standard technique of movement which can be adapted to many input devices and environment circumstances. The environment view moves and turns at eye-level according to user input. In its most basic form, this technique requires only two degrees of freedom: turn left and right and move forwards and backwards.

Additional DOF can be used to turn the view up and down, which represents the *head* metaphor, or for moving up and down, to make use of more than only the ground plane in the environment. Moving left and right can even be added for more precise positioning.

On the other hand, this technique can even be implemented with only two half degrees of freedom: turn left and move forward only. Of course, this representation is limited and lacks ease of use and efficiency. However, this proves the technique's flexibility.

A disadvantage of this technique is that walking isn't always feasible in 3D VEs. Just as in real life, the distances can be too large to navigate efficiently with this technique. This can be overcome by adapting the technique to allow for variable speeds. Another problem is the occasional need for an overview of the environment. Again, multiple solutions are possible, although there are other techniques that already incorporate some of these.

Speed-coupled Flying

One of the techniques that solves both problems presented by the Walking technique is Speed-coupled Flying [TRC01]. It uses the same two degrees of freedom (walking forward and backward and turning left and right) to navigate the environment. However, the further the user indicates to travel, the faster the camera moves. Furthermore, the camera height and viewing

angle are adjusted as well, causing the effect of launching of the ground and flying. By indicating a nearby traveling point, the camera lowers and regains its original viewing angle while losing speed at the same time.

This technique is useful for both long-distance travel and for getting an overview of the environment. In [BKLJP04], wayfinding is considered the cognitive process that takes place during navigation. This cognitive process enables the user to establish spatial awareness of the 3D VE. The Speed-Coupled Flying technique helps the wayfinding process by relating the environment overview to the user's position.

Map-based Travel

A completely different way of traveling involves *cross-task* techniques. Bowman et al. [BKLJP04] use this term to describe techniques that are used for a different task than the ones they were originally meant for. One such example is Map-based Travel [BJH99, BKLJP04] (see figure 2.3). The user is given a miniaturized representation of the environment and uses either a selection or a manipulation technique to choose his destination. The system then moves the user towards the destination.

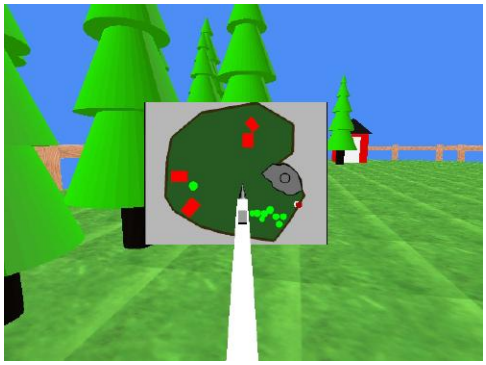


Figure 2.3: Map-based travel [BJH99].

Many variations of this technique are available. Firstly, the representation of the environment can differ. It might be a simple 2D map, possibly enhanced with visualization techniques, but it is also possible to use a smaller 3D view of the environment or WIM (World In Miniature, [SCP95]). Secondly, various selection or manipulation techniques are possible. We will discuss a few common techniques in sections 2.4.2 and 2.4.3. The difference in this context is that the user either has to simply select his destination or manipulate a representation of himself on the map towards the destination.

A last variation point is the execution of the user movement. The simplest solution would be to just instantaneously change the viewpoint to the destination, however, such a sudden change is usually found to be disorienting and nefast to the cognitive wayfinding process. A better option is to move the viewpoint along a route from the original position to the destination. Again many different solutions are possible. It is, however, generally accepted that once the user has chosen his destination on the map, that the system handles the actual travel.

2.4.2 Selection Techniques

Raycasting

Selection techniques are commonly divided, as we have discussed in 2.2.4, in virtual hand and virtual pointer techniques. Raycasting is the most basic of the virtual pointer techniques. It features a pointer or a hand from which a line—the ray—moves out in the direction the user is pointing in (see figure 2.4).

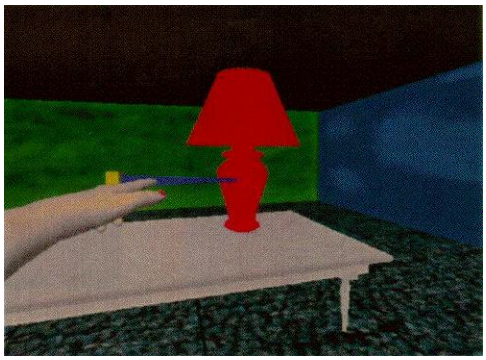


Figure 2.4: Raycasting [PWBI98].

This task only uses five degrees of freedom—three for translation, two for rotation—as rotation around the ray's axis doesn't change the direction of the ray. It is usually supplemented with an extra action that indicates the actual selection of the object which the ray intersects, but another possibility is the immediate selection of the object when the ray is directed at it.

Raycasting is a strong and fast technique for selection, except for very high precision selection or in cluttered environments [PWBI98]. Again, techniques based on raycasting have tried to improve upon its drawbacks. Most notable are flashlight techniques, which change the ray to a conic volume instead of a cylindrical. This allows easier selection of far away objects, but introduces a new problem: disambiguation of multiple close objects [LG94].

Disambiguation rules have been introduced to solve this problem. Further advancements, such as the aperture technique [FHZ96] have tackled other drawbacks.

Image Plane

A completely different set of selection techniques fall under the header of Image Plane Techniques [PFC+97]. These also include several manipulation and navigation techniques, though selection techniques are most prominent. The main feature of these techniques is that they work with the projection of the 3D environment on a 2D surface, the image plane (see figure 2.5).

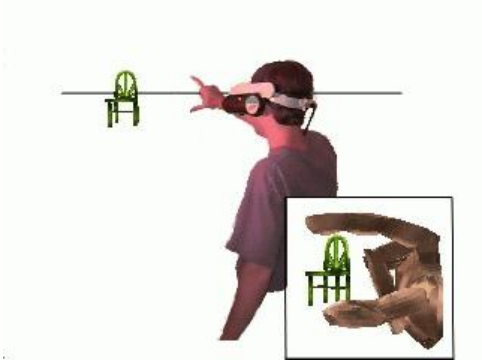


Figure 2.5: The Head Crusher technique [PFC+97].

The user moves one or two hands in certain patterns to select an object. One of these patterns uses an index finger and thumb as if sizing an object, the Head Crusher Technique. The actual selection is still done by raycasting, from the user's eye through the midpoint between thumb and index finger. It is followed by a selection command. Disambiguation in this case is solved by comparing orientation of an object with the orientation of the user's hand or fingers.

2.4.3 Manipulation Techniques

Virtual Hand

As with raycasting, Virtual Hand is the most basic technique in its own category. With the Virtual Hand technique the user controls a representation of a hand or some other pointer, which can be moved in six degrees of freedom (see figure 2.6). The selected object will translate and rotate along

with the Virtual Hand until the selection is released [PWBI98].

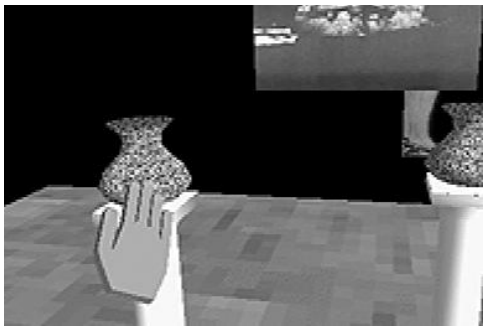


Figure 2.6: Virtual Hand [PWBI98].

The most basic virtual hand technique maps the exact movements of the user, but direct mapping is disadvantageous for translating large distances. Improvements include the Go-Go Technique [PWBI98], which uses an exponential mapping after a certain distance and stretches the Virtual Hand to work with longer distances.

The Virtual Hand technique is advantageous for manipulation, while the Raycasting technique works well for selection. This has prompted the creation of HOMER [BH97], which combines these two techniques into a single working technique that addresses both the selection and manipulation tasks. Selection occurs through raycasting, after which the virtual hand is attached to the object and manipulation can occur. This technique falls into the category of Hybrid Techniques, because it combines the Virtual Hand and Virtual Pointer metaphors.

WIM

WIM or World-In-Miniature [SCP95] is an indirect technique for manipulation, where the 3D VE is copied and scaled down, after which the user can interact with the miniature representation of an object to manipulate the actual object (see figure 2.7).

As mentioned before, WIM works for navigation as well, which makes it an all-round technique. The drawback situates around the scaled down copy of the 3D VE, which could be very large. A copy of a large environment that still fits in the user's view might be on such a small scale that interacting with individual objects is no longer viable. A possible solution would be a selective scaling of only a part of the environment [WHB06]. This would limit

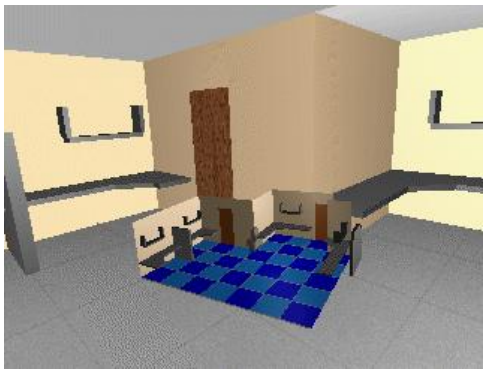


Figure 2.7: World-In-Miniature [SCP95].

navigation and long-distance translation, but would enhance the selection and precise manipulation of miniature objects.

2.4.4 Application Control Techniques

Menu's

In 2D environments, the most common method of application control has always been through menu's. It is therefore only logical, that in developing 3D VEs, menu's have been given a lot of attention as well.

Solely taking over 2D menu's from the original 2D environments, however, has not proved the most excellent solution. The addition of a third dimension complicates matters. Firstly, regular menu's—those that in reality aren't an actual part of the environment—don't promote the immersion that we wish to achieve in designing 3D VEs. Secondly, using 2D menu's inside the environment is still impeded by having to interact with a flat plane where depth is present.

There are many menu techniques that solve these problems, for example by constraining the user's movements. An example of this is the ring menu [LG94], a 1DOF technique, which presents menu items in a circle around the user's hand (see figure 2.8). Therefore, the user only has to use rotation around one axis to be able to select an item. Another advantage of this technique is that the menu is always easily accessible, because of its placement near the user's hand.

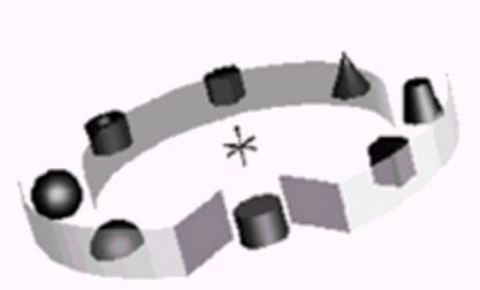


Figure 2.8: Ring Menu [LG94].

Tools

A different way of providing application control is through tools or widgets that the user can manipulate. These tools can be both real or virtual, but real tools can also be considered input devices, so they fall outside the scope of this chapter.

Through manipulation of virtual objects, the application control task can be completed. These techniques could therefore fall under the category of manipulation as well. An example of such a tool is the 'color picker widget' [Mat93] (see figure 2.9). By manipulating the spheres on three axes representing the RGB spectrum, the user can specify a color.

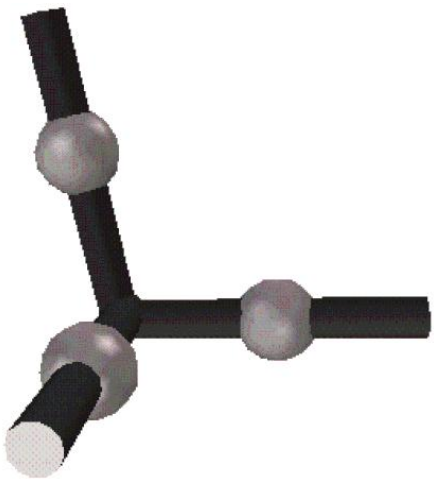


Figure 2.9: Color Picker Widget [Mat93].

Other examples can be found in [Min95], [Han97] and [DH07].

2.5 Conclusion

In this chapter, we have aimed to abstract interaction techniques. To do so, we have first provided a definition of the term, after which we assembled a comprehensive taxonomy based on certain interaction technique properties. Properties not included in the taxonomy were discussed afterwards. We ended the chapter by giving the most common examples for each interaction task.

In the next chapter, we will aim to abstract input and output devices. These abstractions, in the form of taxonomies and properties, will then be used to achieve the objective of this thesis: matching interaction techniques and devices.

Chapter 3

Interaction Devices

3.1 Introduction

In the previous chapter, we discussed interaction techniques and possible categorizations and properties that can have an influence on the three-dimensional Virtual Environments (3D VEs) and interaction devices they're used with. As we have previously explained, interaction techniques are necessary to ensure the interactivity of a 3D VE. However, these techniques can only be employed by handling interaction devices, that bridge the gap between user and system.

In this chapter, we will explore interaction devices similarly to our exploration of interaction techniques in Chapter 2. We will first give a general description of the term interaction device, after which we will explore different taxonomies to form a comprehensive taxonomy of our own.

Properties that do not fit within the assembled taxonomy, and issues concerning interaction devices will be discussed in section 3.3. The chapter will end with an overview of common interaction devices, their properties and their placement in the interaction device taxonomy.

3.1.1 What is an Interaction Device?

Bowman et al. [BKLJP04] do not define 'interaction device' generally, but separate the definition for input or output devices:

> *An input device is a physical (hardware) device allowing communication from the user to the computer.*
> *An output device is a physical device allowing communication from the computer to the user.*

This distinction differs only in the direction of the communication, so we can conclude that an interaction device serves as a means of communication

between the user and the system. The direction of the communication then, distinguishes whether the device allows for input, output or possibly both.

As an example, consider a set of headphones with attached microphone (see figure 3.1). This device can be used to interact with a system via speech interaction and auditory feedback and it communicates both ways between user and system. On one hand, input is given through the microphone, on the other hand, output is received through the headphones.

Figure 3.1: Some devices combine both input and output [Aud].

It is clear that this definition by [BKLJP04] is not very specific. Apart from allowing communication between user and system, a device can have any possible combination of properties.

The design of new interaction devices is out of the scope of this thesis, but such papers as [CMR90] and [Frö05] explore the means for obtaining new interaction devices.

To distinguish between these many different possibilities and to aid the choice of interaction device for a certain 3D VE and a chosen interaction technique, we will endeavor to order interaction devices in a taxonomy. The following section will present an overview of existing taxonomies, on which we will base our own.

3.2 Interaction Device Taxonomies

As already mentioned, Bowman et al. [BKLJP04] distinguish between input and output devices. This because interaction with a 3D VE occurs through input, while output is mostly a means for feedback. Nevertheless, both the choice of output and input devices is of importance with 3D VEs and even more so how seamlessly they fit together with the chosen interaction tech-

niques.

In the taxonomy that will be constructed in this chapter, we will make the same distinction. Even though there are devices that can communicate both from the user to the system and back, matching interaction techniques with interaction devices deals mostly with input devices. However, output devices will not be neglected, as their choice can have an influence on the forms of feedback that are incorporated in some techniques.

3.2.1 Output Device Taxonomy

Output devices transfer information from the system to the user. Bowman et al. [BKLJP04] sort output devices by the sense they appeal to. This is only logical, as humans perceive information through their senses. Furthermore, the feedback that techniques provide are usually centered on certain senses. This provides us with a means to link output devices to interaction techniques.

Bowman et al. make the following distinctions in output devices:

- Visual output
- Auditory output
- Haptic output
- Olfactory output

The first two are most widely used, both in 2D and 3D environments. Still, further research is being done to improve existing devices and invent new ones. We will give some examples of visual and auditory output devices in section 3.4.

Haptic output devices can appeal to different facets of our sense of touch. Humans can sense pressure, texture, temperature and so on through their skin, and these effects are simulated with tactile output devices. On the other hand, we are also aware of the position, movement and rotation of our limbs as well as our body, and some haptic devices use kinesthetic cues to simulate this awareness.

Lastly, research has also started focusing on olfactory output devices. There now exist devices that produce smells to further immerse a user in a 3D environment. Much research is still being done in this area, however. Bowman et al. did not include *taste* as a possible output sense, but it is a possible venue for further research.

Of course, not all devices fit mainly into one category. There are some devices that provide more than one output modality. This requires an additional category for multimodal output device. Our final output device taxonomy is as follows:

- Visual output
- Auditory output
- Haptic output
- Olfactory output
- Multimodal output

3.2.2 Input Device Taxonomy

There is no doubt that output devices are important to ensure the interactivity of a 3D VE. However, it is the input devices that actually enable the user to interact with such environments. When choosing interaction techniques and devices, the input devices are the ones that are matched with interaction techniques. Output devices are then added where needed. We will discuss this more in depth in Chapter 4.

It is with this in mind that we discuss input device taxonomies. Previous research has provided us with several input device taxonomies, not all of which are suitable to our needs. A general overview is provided in [BKLJP04], but in this section we will discuss only some taxonomies more in depth, namely those that can contribute to our goal.

The earliest taxonomies, such as in [FW74] or [FWC84], classified devices based on the interaction tasks they could be used for. Devices can, however, often be used for more than one interaction task, making this classification redundant.

Buxton provides us in [Bux83] with a taxonomy for continuous hand-controlled devices. This is only a subsection of possible input devices, yet it takes into account several properties that can be linked to interaction techniques (see Chapter 4).

In figure 3.2, we can see that Buxton considers the dimensions of the device as well as the property that it senses. Furthermore, this taxonomy distinguishes between direct interaction (touch, T) and indirect interaction (mechanical intermediary, M).

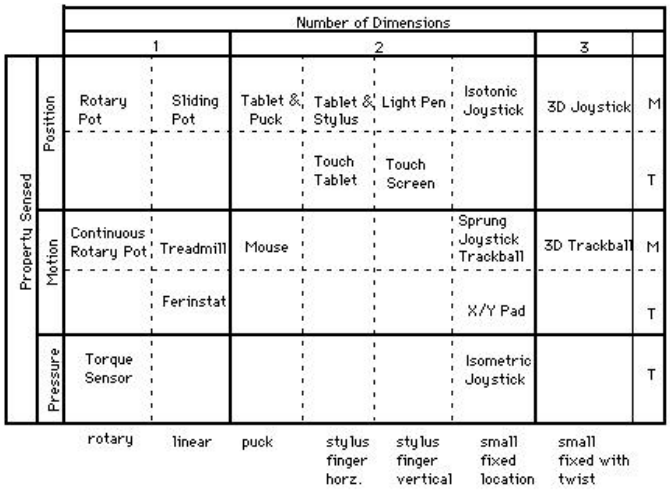


Figure 3.2: Input Device Taxonomy of Buxton [Bux83].

There are some drawbacks to this taxonomy, however. Firstly, as Buxton only considers continuous hand-controlled devices, there is no place for any device outside this category. Furthermore, Buxton obviously makes the assumption that devices sense only one property. There are, however, devices that can sense both position and pressure (such as more recent pressure-sensitive tablets).

In [CMR90], Card et al. solve this problem and more properties have been added to their taxonomy (see figure 3.3). They have also replaced dimensions with linear and rotary directions.

A device in this taxonomy isn't represented within a single cell, but through a combination of different properties and directions. This taxonomy takes not only note of the degrees of freedom of a device, but also *which* degrees of freedom and the range of those degrees. Furthermore, it also indicates which type of input the device provides.

Card et al.'s taxonomy considers many factors and is very flexible, which is why we will use it in this thesis, with the addition of some other properties that can be linked to interaction techniques.

3.3 Other Properties of Interaction Devices

Just as with interaction techniques, we wish to look at other properties of interaction devices as well (see section 2.3). Finding links between interaction techniques and interaction devices will help us to determine which

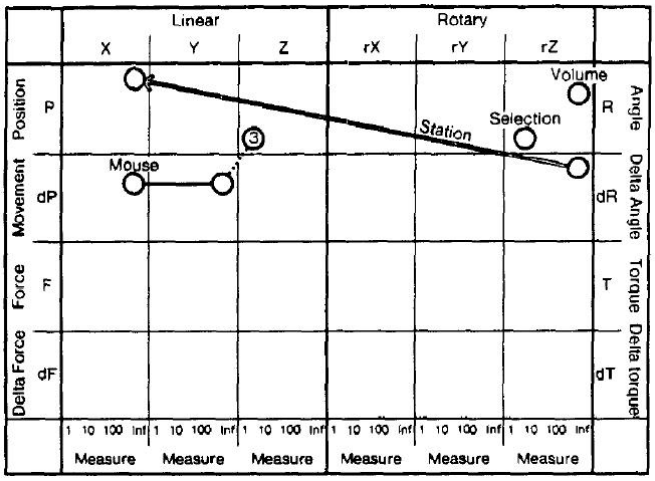


Figure 3.3: Input Device Taxonomy of Card et al. [CMR90].

devices and techniques match well together.

In this section, we will explore some properties of interaction devices (in particular, input devices) that might have an influence on the interaction techniques they are best used with. We will look further into the links between interaction techniques and devices in Chapter 4.

3.3.1 Properties

Interaction devices, just as interaction techniques, have a certain amount of actions and degrees of freedom. However, it is sometimes possible to combine some of these to get even further possibilities. For example, with a common mouse, combining the mouse movement (2DOF) with the available buttons enables us to perform at least two or three extra actions.

Furthermore, as indicated in Jacob et al.'s research [JSMMJ94], the degrees of freedom of a device can often be separated into several integral groups of rotation and movement. Although the degrees of freedom can be read from Card et al.'s taxonomy (see section 3.2.2), there is no indication which can be used simultaneously.

Another property is whether the device works with absolute or relative data [Jac96]. In the case of the former, the device has a fixed origin and measures input according to that origin. With relative devices, the input is measured according to a relative—and flexible—starting point.

We mentioned in section 3.2.2 that a taxonomy based on interaction tasks was an insufficient classification, but it is still possible to determine for an input device which tasks it is or isn't suited to.

3.4 Overview of Common Devices

There exist many devices for 3D interaction, too many to count. A lot of these devices are designed by research facilities to experiment with new forms of input and output. In this section, we aim to provide an overview of the most common 3D devices, both for input and output.

3.4.1 Output Devices

CAVE

A CAVE or Cave Automatic Virtual Environment [CNSD+92] is a set-up that uses multiple screens and projectors to immerse a user in the virtual environment. The idea is to surround the user with six screens, creating a small room. Most often, however, three screens are used, in front and to the sides of the user. Possibly the floor can be used to project on as well (see figure 3.4).

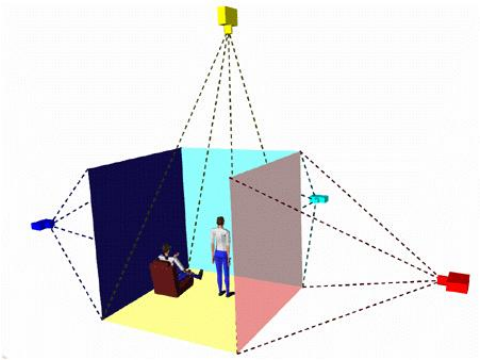


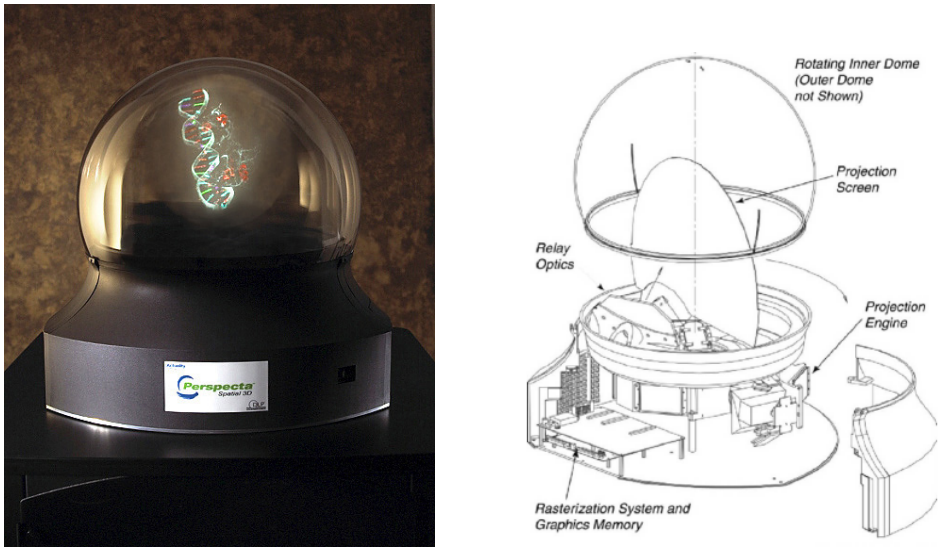
Figure 3.4: A possible CAVE set-up.

A CAVE can be used to create a higher level of immersion, as it uses user-centered projections and leaves the user less aware of the real world outside the room. On the downside, a CAVE set-up is expensive and takes up a lot of space

Volumetric Display

Where 2D displays try to achieve a three-dimensional view through depth cues or by employing 3D glasses, volumetric displays actually create 3D imagery.

These displays consist of a volume, wherein a projection screen rotates at high speed while 2D image *slices* are projected on this screen (see figure 3.5).



(a) Perspecta Volumetric Display (by Actuality Systems, Inc.).

(b) Schematic View.

Figure 3.5: Two views of a volumetric display [Fav02].

Volumetric displays aren't generally used to project 3D VEs and are thus not very immersive, but they provide high quality 3D images and are very suited as visual feedback for object manipulation.

Head-Mounted Display

Head-Mounted Displays (or HMDs) provide users with screens right in front of their eyes [Wik]. A HMD is both an input and output device, as it usually includes head tracking and visual output. Two small displays provide the user with a view of the 3D VE, dependant on his position and orientation, which is determined through head tracking. As the device rests on the user's head, often headphones are included as well.

Early HMDs were heavy and cumbersome, but they have sunken much in size and shape and are now readily available. Another advantage is their

Figure 3.6: A Head-Mounted Display.

high level of immersion, as long as most of the user's field of view is covered by the displays.

Binaural Sky

Not only speakers, headphones or surround systems can be used to present auditory feedback. An example of a different auditory output device is the Binaural Sky [MWTF05]. This set-up combines many different audio techniques to generate 3D sound and employs tracking to localize feedback.

Figure 3.7: The Binaural Sky[MWTF05].

As the Binaural Sky is set up against the ceiling, it doesn't clutter the user's field of view, however, this also limits its working range.

PHANToM

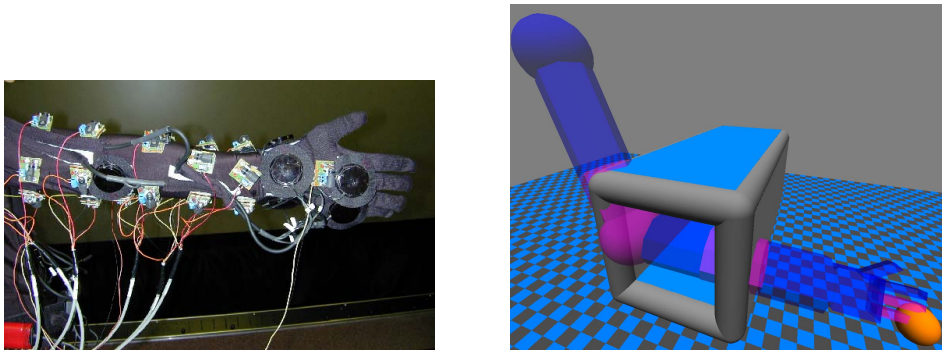
Like the Head-Mounted Display, the PHANToM is both an input and output device. It can track the movement of a finger or pen and it provides force feedback by exerting a force against the user's hand. Force feedback

allows the user to feel the existence of objects in a 3D VE.

Figure 3.8: Phantom Desktop from Sensable[MS94].

Tactor Suit

Another option to allow a user to feel the 3D VE is tactile feedback. One example of this is the Tactor Suit [BB07], a suit that vibrates when the user collides with a virtual object. Variations in the frequency and occurrence area of the vibrations provide the user with different feedback. For example, in the case of deep collisions, the suit would vibrate in the area of collision, with urgent pulses.



(a) Tactor suit with the tactors attached.

(b) The virtual arm.

Figure 3.9: An example of a Tactor suit and its application [BB07].

Tactile feedback allows for both a greater and smaller sense of realism. Feeling the virtual environment makes it more real, but note that tactile feedback does not prevent the user from intersecting virtual body parts with objects, detracting from the realism at the same time.

Olfactory Display

As using smell as an output device is still in its early research stages, there are not yet any well known and readily available devices to discuss. There are, however, some experimental devices available. One of these has been designed by Nakamoto and Minh [NM07] and uses solenoid valves to distribute odors (see figure 3.10). Bottles with artificial odors are connected to these solenoid valves and the opening and closing of the valves releases the odors in the appropriate ratios.

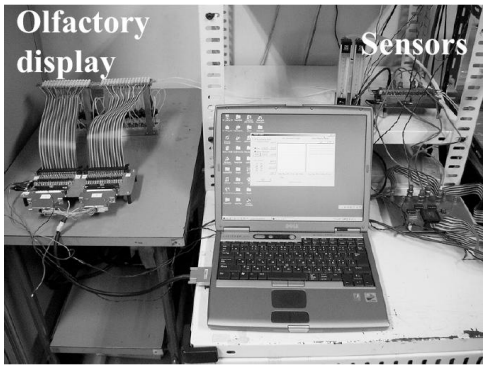


Figure 3.10: A 32-component olfactory display[NM07].

3.4.2 Input Devices

Magnetic and Optical Tracking

As mentioned earlier, some output devices use tracking to localize their output. Tracking is used to determine the position and/or orientation of a user or of an object. There are two common ways of tracking, one of which is magnetic tracking. Magnetic tracking is based on the transmission of a magnetic field and the use of sensors that react to the magnetic field. An example of this is the Polhemus FASTRAK [Pol07] (see figure 3.11).

Another way to determine motion and orientation is through optical tracking. This set-up uses cameras and markers to infer spatial information. Multiple cameras take images from different angles. Different types of camera require different processing to filter out anything but the markers from the images. The position and orientation of the markers is then calculated. An example of optical tracking which uses infrared cameras and reflective markers is Vicon [Vic07] (see figure 3.12).

Magnetic and optical tracking offset each other in advantages and disad-

Figure 3.11: Polhemus FASTRAK[Pol07].

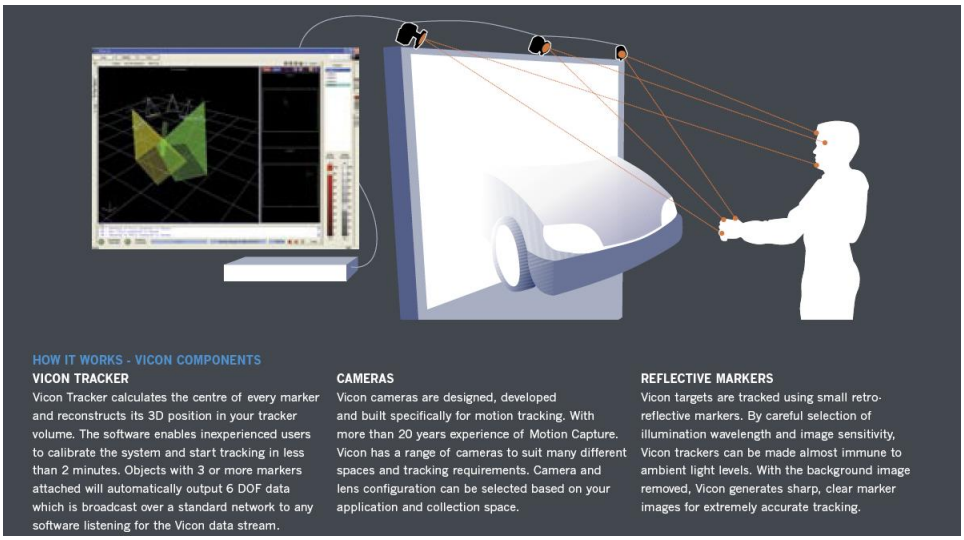


Figure 3.12: An overview of how the Vicon tracking system works[Vic07].

vantages. While the former can have interference issues from conductive materials or magnetic items, the latter may have problems when something obscures the view of the markers. Other types of tracking include acoustic, electromagnetic and inertial.

Data Gloves and Pinch Gloves

Another common type of 3D input device are gloves, which allow a user to interact with the 3D VE with his hands. One type of gloves are data gloves, which are embedded with sensors to determine the movement of the fingers and some movements of the hand. An example of this type of gloves can be seen in figure 3.13.

Another type of gloves are Pinch Gloves [Sys01] (see figure 3.14, which uses sensors on the finger tips to determine which fingers are *pinched* together.

Figure 3.13: 5DT Data Glove 14 Ultra[5DT07].

These can be combined with a regular tracking device. Both types of gloves are used to recognize hand gestures that can be applied to many interactions.

Figure 3.14: Pinch Gloves[Sys01].

Cubic Mouse and SpaceMouse

As the 2D mouse has proven an indispensable device for desktop applications, efforts to produce a likewise device for 3D environments have given us several 3D mouses. One example of this is the Cubic Mouse [FP00]. This device is cubic in shape, with moveable rods that can be used for translations and rotations and several buttons (see figure 3.15). Furthermore, the position and rotation of the device itself is tracked, providing even further input possibilities.

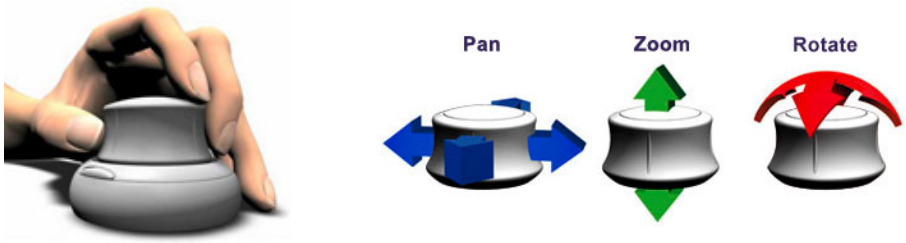
Another 3D mouse is the SpaceMouse [3Dca] from 3Dconnexion. This device is shaped almost like a trackball mouse, but the moving unit on top

Figure 3.15: The Cubic Mouse[FP00].

has six DOF, instead of two (see figure 3.16). Unlike the Cubic Mouse, the SpaceMouse isn't tracked; it is meant as a device for desktop 3D interaction.

Figure 3.16: The Space Mouse Plus[3Dca].

Recently, 3Dconnexion has redesigned the 3D mouse concept and come up with the SpaceNavigator [3Dcb] (see figure 3.17). This device too is meant for desktop 3D interaction and although desktop 3D interaction detracts from the level of immersion into 3D VEs, these devices make 3D environments much more accessible.



(a) The Space Navigator. (b) 3D movement with the Space Navigator.

Figure 3.17: The Space Navigator and its 3D movement. [3Dcb].

3.5 Conclusion

In this chapter, we have aimed to abstract input and output devices. This was done by first defining the terms, after which we provided taxonomies for both input and output devices. Then properties were discussed to provide a further theoretical framework of input devices. The properties of output devices will be discussed in the next chapter.

The next chapter will compare the taxonomies and properties of interaction techniques and input devices to determine how these can be matched. An algorithm will be proposed to score the compatibility of devices with a technique, or of techniques with a device. Furthermore, a second algorithm will be discussed which will suggest possible output devices to be used as well.

Chapter 4

Matching Techniques and Devices

4.1 Introduction

After our discussion of interaction techniques (see Chapter 2) and interaction devices (see Chapter 3), we are now ready to propose an algorithm for matching techniques with devices and vice versa.

To accomplish our goal of aiding the designer of a 3D Virtual Environment in his choices, we will implement this algorithm in a matching tool, ABITID (see Chapter 5).

In this chapter, we will first search for links between techniques and devices, which can help us during the matching process. This will be done by looking at the influence of the properties discussed in sections 2.3 and 3.3.

In a next step, we will formulate an algorithm that uses the links found in the first step to find the best possible matching input devices for a chosen technique and vice versa.

Then, we will discuss how, based on the matching between interaction techniques and devices, we can provide suggestions for output devices as well.

Lastly, we will discuss some ideas and issues concerning this topic.

4.2 Linking Interaction Techniques and Devices

In Chapters 2 and 3 we proposed a taxonomy for both interaction techniques and devices, as well as listing several other properties that might bear an

influence on the matching process.

The taxonomy for interaction techniques is based on interaction tasks, and while it would be possible to assign interaction devices to tasks as well, we determined in Chapter 3 that this wasn't a very good basis for a taxonomy. However, this does not mean that there is no possible link. Certainly, determining which tasks a device is suited to—or not suited to—can help us match devices and techniques.

The taxonomy for interaction devices contains many of their properties already: DOF, input type, amount of actions. These properties will be linked with interaction techniques in the following section.

4.2.1 Relative vs. Absolute

For both interaction techniques and devices we can determine whether they are relative or absolute. In light of the matching process, however, we have to determine if these properties correlate. This is certainly an area where further research would give us more information to work with, but in this thesis we will use reasoning to determine their influence.

Logically, absolute devices and techniques are better suited to each other than to relative devices and techniques, because firstly, it is easier for the designer to program the interface, and secondly, because we believe that in most cases, using a relative device with an absolute technique—or vice versa—will be less instinctual to the user. We will discuss this further in section 4.4.

4.2.2 Range

As with the previous property, it is not clear whether comparing ranges will produce better technique-device matches. This seems to be a property where there is no link between technique and device. After all, the Image Plane Head Crusher technique [PFC+97] (see section 2.4.2), which works well for objects at a distance, has been implemented successfully with data gloves (see section 3.4.2).

4.2.3 Input Type and Interaction Task

As we mentioned earlier, the chosen taxonomies for interaction techniques and input devices also represent certain properties. One option is to assign interaction tasks to input devices, however, as mentioned in section 3.2.2, devices can often be used for multiple interaction tasks. Nevertheless, it may prove a distinguishing factor in the algorithm.

Another option is to consider the types of input against the interaction tasks. As shown in the chosen taxonomy, reprinted here (see figure 4.1), Card considers eight types of input data. It is possible to match these types with interaction tasks instead of devices themselves. If a certain input device then produces the type of input data that corresponds with the interaction task of a certain technique, it will indicate a higher compatibility.

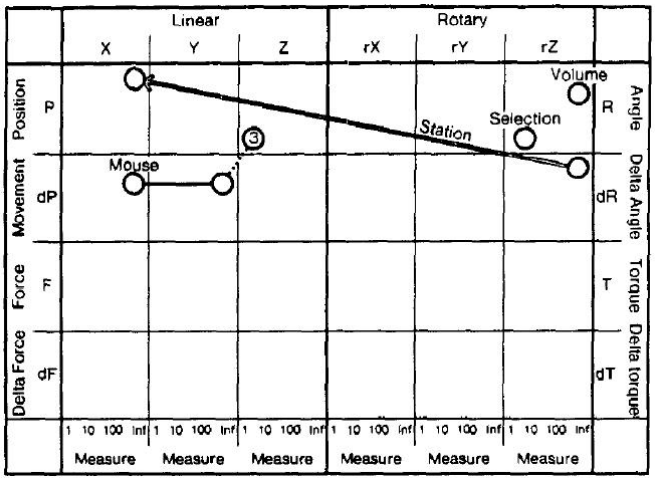


Figure 4.1: Input Device Taxonomy of Card et al. [CMR90].

As an example, consider the selection task, which only needs movement data. A device that provides this data can then be used for this task. In the examples provided by Card (see figure 4.1), we can see that the mouse generates movement data. Of course, we also immediately see that the mouse does not generate movement data for all three linear dimensions, which brings us to the next property.

4.2.4 Actions and DOF

The actions and degrees of freedom provided by both interaction techniques and devices are the most complex properties in our matching process. Before we determine how these properties can be linked to determine the compatibility of a technique and a device, we will first go into deeper detail as to their definition.

Degrees of Freedom

Degrees of freedom (or DOF) represent the motions an object can perform in 3D space. Generally there are six degrees of freedom in 3D space: three translational or linear degrees and three rotational or rotary degrees. For

interaction techniques, six degrees of freedom describe all possible operations. The walking navigation technique, as described in section 2.4.1, uses two degrees of freedom, while the Virtual Hand manipulation technique (see section 2.4.3) uses six.

Input devices, on the other hand, can provide more than six degrees of freedom [HSK+05, Frö05]. There are devices that can be used for two interaction tasks at the same time, such as the Cubic Mouse (see section 3.4.2), that can be used for both viewpoint and object manipulation.

Another factor that is inherent to input devices is how the degrees of freedom are combined. In their research, Jacob et al. [JSMMJ94] speak of separability and integrality. This means that in a device, some degrees of freedom can be altered simultaneously—in an integral manner, while others are separated and require different movements from the user.

Using the Cubic Mouse as an example again, we can determine that the tracking is completely integral, where moving and turning the cube in six degrees of freedom can be done simultaneously. Manipulation is done by moving and turning three rods. Firstly, this is separate from the navigation, and secondly, the manipulation of each rod is separate from the others as well. However, moving and turning a rod can be done at the same time, so again, this is integral. This gives us a more detailed view of the degrees of freedom of the Cubic Mouse, namely: 6+2+2+2 or 6+(3*2).

Research proves that using the more detailed information about degrees of freedom—to determine which tasks to use a device for— yields better results [JSMMJ94]. There has been no research, however, about linking this data to the degrees of freedom of an interaction technique. There are several issues which we must consider.

First, it has been proven that integral devices work better for integral tasks and separable devices work better for separable tasks [JSMMJ94]. Since we know the interaction task for a certain interaction technique, we can determine for two interaction devices with different DOF *formulas*, which of them is better suited to the interaction task and thus, to the interaction technique that performs that task.

A more difficult issue is whether we can assume that a device with the same degrees of freedom as a technique is more compatible with that technique than a device with more or less degrees of freedom. In case that a device has less DOF than a technique, it might still be possible to perform the task by employing modes, but it seems only logical that this is less preferable than a device with the correct DOF. This is, for example, the reason why a 2D

mouse is not as suited to 3D operations.

The issue is less clear when the input device has more DOF than warranted for the interaction technique. It is sometimes argued that too many DOF can confuse the user or make the interaction more difficult. However, this is not necessarily the case. We can at least conclude that it is preferable over too few DOF.

A last issue is the actual comparison of the DOF formula of a device with the degrees of freedom of a technique. This is were our first issue is not very clear. The research of [JSMMJ94] proves that integral devices work better for integral tasks, but a lot of devices are not simply integral. They are a separable combination of several integral parts. It is nearly impossible to determine which input device DOF formulas work best for an interaction task, even more so when we consider that these formulas don't mention which degrees of freedom are separable or integral.

Actions

Interaction techniques and input devices often have actions that accompany the degrees of freedom. In the case of interaction techniques, there is need for more than movement or rotation, such as an indication to select an object, or an indication to switch from navigation to manipulation, if that is not automatically done.

In the case of input device, there are often buttons that provide extra input possibilities, but are not included in the named DOF of a device. As a note, buttons or other input actions can be represented on Card et al.'s taxonomy, within the position region (see figure 4.1 for an example).

The amount of actions of a technique, just as with the types of feedback (see section 4.4), is often dependent on the exact implementation of a technique. Nevertheless, it is clear that an input device should have at least the same amount of actions available, if not more, to be able to used for that technique.

Note that input device actions can be used to provide more DOF, through modes, though as mentioned earlier, this is generally less preferable.

4.2.5 Linked Techniques and Devices

There is one final link between techniques and devices that must be considered. It is possible that a certain technique has already been matched with a device, or vice versa, perhaps even from its design phase. For example,

consider the Image Plane techniques (see section 2.4.2), which have been designed with hand and finger tracking in mind, for example through Data Gloves (see section 3.4.2).

In such cases, these techniques or devices should score better, unless the 3D VE designer is, for example, trying to find new matches and thus doesn't consider this property as important.

4.3 Matching Algorithms

To reach our goal of matching techniques and devices when designing 3D Virtual Environments, it was necessary to research why certain techniques and devices match better together than others. To determine this, we have given an in depth review of interaction techniques and devices in the previous two Chapters.

In those Chapters, we tried to form a more abstract view of techniques and devices, by discussing classifications and properties that are used to define their specifications.

Then we established exactly how interaction techniques and devices influence each other, by determining which properties interact. There might be other properties or influences, which have not yet been researched. For certain, what users consider a good match between a technique and a device is difficult to exactly define.

In the next section, we will propose an algorithm to use the previously gained knowledge to match techniques with devices or vice versa.

4.3.1 An Algorithm for Matching Techniques and Devices

The whole basis for the matching process lies in the links between techniques and devices that we have determined in section 4.2. The matching process should then consist of examining these links for the possible matches and scoring the matches accordingly.

We will discuss the algorithm where the 3D VE designer has chosen a technique and wishes to match this with a device. The process is reversible for matching a selected device with a technique. We will follow up our discussion with the algorithm in pseudo code and an example. The algorithm consists of three major steps:

Step 1: Listing the possible matches

Having selected an interaction technique, it is necessary to make a list of all input devices from the database that can be matched with this technique. In assembling this list, the matching score of each device is set to zero.

Step 2: Scoring each device

For each device in the list from step one, the links or influences discussed in section 4.2 are all assessed, to determine how well the chosen technique and the current device match together.

Step 2-1: Scoring each property

For each property, a score list is available to determine which values deserve which scores. These scores are percentages and the score list can be as simple as an all-or-nothing score or can contain a whole range of possibilities. The actual scores in these score lists can be up for debate, however, see section 4.4. In the example following this explanation, different score lists will be shown.

Step 2-2: Weighing each property

The scores obtained in the previous step will be weighed according to their importance. Different 3D VEs will have different design conditions and thus, the weighing values can be set by the designer.

The total scores of each property (p) will be added to form the total score of each device (d), as in the following equation:

$$score_d = \sum_{p=1}^{n} (score_p * weight_p) \tag{4.1}$$

This final score will be converted to a percentage as well, for easier comparison between devices. This can be done by dividing the total weighted score by the maximum weighted score.

Step 3: Presenting results

After each device has been scored, the device will be sorted, highest score first, and presented to the designer. The highest score will represent the best matching device, according to the weights stated by the designer and the scoring list for different property values.

Pseudo code

The algorithm is here summarized in pseudo code:

Algorithm 1 MATCHING(technique)

```
1: for all devices do
2:    for all properties do
3:       give basic score for property value
4:       multiply with user-defined weight
5:       add score to total device score
6:    end for
7: end for
8: list devices, highest score first
```

Example

To illustrate the algorithm, and more specifically the scoring process, we will give an example with one technique and one device. The technique which we will use is the Image Plane Head Crusher technique, the data of which is available in section 5.2. As an input device, we will consider Head Tracking, of which the data is given in the same section.

To start comparing Head Tracking with the Head Crusher technique, we must first gather the score list of each comparable property. The properties considered are:

- interaction task
- absolute vs. relative
- amount of actions
- total DOF
- integral and separable DOF
- DOF types
- suggestions

The first two properties are all-or-nothing properties. Either the input device can accomplish the interaction task, or it can't. Either both input device and interaction technique are both absolute or both relative, or they aren't. This gives us the score lists in figure 4.2.

Note that of course, the values given aren't actually thus represented in the algorithm. The value *available* in the first score list is determined by first considering the top level task category of the interaction technique and then searching if this category is mentioned as a possible interaction task in the input device data.

Interaction Task	
Value	Score
available	100
unavailable	0

Absolute vs. Relative	
Value	Score
equal	100
unequal	0

Figure 4.2: Score lists for Interaction Task and Absolute vs. Relative.

For the amount of actions and total DOF we can distinguish three possibilities. Either they are the same for both technique and device, either the device has more actions or DOF, or the device has less actions or DOF. As explained in section 4.2, the first option is preferable over the second and the second option is preferable over the third. The score lists are then as shown in figure 4.3.

Amount of Actions	
Value	Score
equal	100
more	66
less	33

Total DOF	
Value	Score
equal	100
more	66
less	33

Figure 4.3: Score lists for Amount of Actions and Total DOF.

It is possible for these properties to distinguish more possible values and thus provide more detailed scoring. This can be done by considering exactly how much more or less actions or DOF the input device has. Having many more DOF than the interaction technique might be less preferable than just having one or two more.

Comparing the integrality and separability of an interaction technique and an input device is done in steps. First we look at the number of integral DOF groups, which we will call the separability amount. This is an all-or-nothing property, but unlike the first two score lists, we will let the score reflect that unequal separability amounts are not inadvisable, but just not as preferable as equal separability amounts (see section 4.4).

To compare integrality, we try to match the integral DOF groups of the technique with those of the device. Scoring is done based on the total amount of integral DOF groups and the amount of them that match, which gives us the score list shown in figure 4.4.

Separability	
Value	Score
equal	100
unequal	50

Integrality	
Value	Score
n groups	100
n-1 groups	(n-1/n)*100
...	...
1 group	(1/n)*100

Figure 4.4: Score lists for Separability and Integrality.

The next property to compare is the DOF types. Here we check whether the DOF types of the input device match the DOF types of the interaction technique. Scoring is based on whether the DOF types of the interaction technique are all available in the device. If they aren't, the technique will probably not be executable with that device, in which case the score should be zero. It is, however, possible for the input device to have more DOF types than the interaction technique (as one of the previous properties has evaluated). The score list is shown in figure 4.5.

The last property to match is suggestions given for possible technique-device combinations. This is again an all-or-nothing score, also shown in figure 4.5.

DOF Types	
Value	Score
all available	100
not all available	0

Suggestions	
Value	Score
available	100
unavailable	0

Figure 4.5: Score list for DOF types and Suggestions.

With these score lists, it is possible to determine the base scores for our example as shown in figure 4.6:

The next step of our algorithm consists of calculating the weighted scores. As mentioned earlier, if the DOF types of the technique aren't available in the device, this most often indicates that the technique can't be executed with this device. Therefore, this property holds a lot of weight. For this example, we have chosen our own weights, but as mentioned before, the weights are adjustable by the 3D VE designer. After the weighted scores have been calculated, these can be added to achieve the total score. This is used to calculate the final percentage, by dividing the total weighted score by the maximum weighted score. The results are shown in figure 4.7.
As in this example, a similar score is calculated for all other devices. The final results depend a lot on the weights assigned to the properties. For

Base Score Table	
Property	Score
Interaction Task	100
Absolute vs. Relative	100
Amount of Actions	33
Total DOF	100
Separability	100
Integrality	100
DOF Types	0
Suggestions	0

Figure 4.6: Base score list.

Score Table			
Property	Score	Weight	Weighted Score
Interaction Task	100	6	600
Absolute vs. Relative	100	6	600
Amount of Actions	33	5	165
Total DOF	100	8	800
Separability	100	10	1000
Integrality	100	10	1000
DOF Types	0	50	0
Suggestions	0	5	0
		Total	4165
		Percentage	**41,65%**

Figure 4.7: Resulting score table.

example, giving each property the same weight results in a final score of 66,6%, or assigning a weight of 60 to the DOF Type property would lower the final score to 31,7%. Thus, the results are influenced by the importance the 3D VE designer attaches to the properties.

4.3.2 An Algorithm for Suggesting Output Devices

A 3D Virtual Environment isn't achieved with just input devices and interaction techniques. However, they are the main focus of this thesis as they signify the interaction between the 3D VE and the user. To not neglect output devices, we will propose an algorithm for this as well.

This algorithm works on the same principles as the previous one, namely the properties that link output devices with interaction techniques and input devices.

These properties have not yet been discussed, as they are fewer and lim-

ited to this algorithm. We will go over them in this section, followed by the proposed algorithm.

Properties

Output devices are characterized by their type, as in the taxonomy we proposed in section 3.2.1. We can link this to interaction techniques by looking at the feedback that these techniques incorporate. Furthermore, precedence is given to these feedback types in the following order:

1. Visual output
2. Auditory output
3. Haptic output
4. Olfactory output

This precedence is based on both the availability and the level of immersion of certain output types. Multimodal output is looked at separately. It will generally be considered a better output device than those with only a single mode, but that would depend on the modes included. Therefore, multimodal output is given a higher precedence only when multimodal output with lower modes includes higher modes as well. Thus, an output device that generates both auditory and haptic output would not be given the highest precedence, because it would not include visual output.

Another property of output devices is the level of immersion the device provides. This is not explicitly linked to either input devices or interaction techniques, but does influence the design of a 3D VE, so it will be included.

Lastly, it is difficult to link input devices with output devices as not much research has been done on the use of output devices, especially concerning the available or used input devices. However, for those devices that have both input or output, or are often available together, it is possible to include their influence in the algorithm.

Algorithm

This second algorithm to suggest output devices can be employed only after the first algorithm has executed. It uses the results of the first algorithm to prune inadvisable output devices and score the remaining ones according to the properties discussed in the previous section. We will discuss this second algorithm in steps as well.

The first step for suggesting output devices is removing those output devices that certainly shouldn't be suggested. The algorithm looks first at the resulting interaction techniques of the previous matching process. In the case where the first algorithm was employed to match an interaction technique, there will only be one technique to consider. In this step, the algorithm will compare the feedback types of the interaction techniques with the feedback types of each output device. If an output device does not have the same type as one of the feedback types of the interaction techniques, it will no longer be considered. Issues concerning this step will be discussed in section 4.4.

Secondly, the algorithm will consider the precedence discussed in the previous section about output device properties. Output devices will be scored according to their precedence. The higher the precedence, the higher the score. As with the first algorithm, weights can be given to all scores, to let the 3D VE designer decide on the importance of certain properties.

In a third step, we will consider the list of input devices from the first algorithm (again, possibly only one input device could be considered). In the first part of this step, output devices will be scored if they also provide input and if this input device, then, is in the aforementioned list. In the second part, output devices will be scored if they are compatible with an input device on the list. If any devices scored on the first part, they will no longer score on the second part. However, unless the 3D VE designer thinks differently, the first score will have a heavier weight than the second. Again, issues concerning this step will be discussed in section 4.4.

Lastly, a score will be given to the output devices according to their level of immersion. As with the first algorithm, a final score will be calculated to rate the output devices.

Pseudo code

The algorithm is here summarized in pseudo code:

4.4 Issues and Ideas

Many issues have arisen while designing algorithms to match interaction techniques and devices. Some of these can be solved through further research, while others are more fundamental problems that keep arising in the domain of Computer-Human Interaction. In this section we will attempt to address these issues and where appropriate, suggest possible solutions.

A first issue concerns the whole concept underlying the algorithms. It should

Algorithm 2 SUGGESTION(techniques,devices)

```
1: for all interaction techniques do
2:    collect all types of feedback
3: end for
4: for all output devices do
5:    if output type is not in the list of feedback types then
6:       remove output device
7:    else
8:       for all properties do
9:          give basic score for property value
10:      end for
11:      multiply with user-defined weight
12:      add score to total output device score
13:   end if
14: end for
15: list output devices, highest score first
```

be stated that a lot more research is needed to corroborate the ideas presented in this chapter. This thesis has operated on some hypotheses that only further research can substantiate. The properties and their effects on the compatibility of techniques and devices have not been verified.

A second issue concerns the human factor in the whole matching process. The goal of this thesis is to enable a 3D VE designer to choose matching interaction techniques and devices to provide an all-round better interaction for the user. It is then, of course, important to take into consideration what exactly the user considers good interaction, and which devices and techniques the user prefers, not only separately but even more so together. However, it is near impossible to take a human factor into consideration in the algorithm. To do that, the user opinion should be as objective as possible, and that can only be achieved by testing all interaction techniques and devices under the same conditions; an insurmountable task.

There are also a few more specific concerns regarding the algorithms. For one, as mentioned in section 4.2, the detailed DOF formulas concerning separability and integrality of input devices can't easily be compared with the degrees of freedom of an interaction technique. However, perhaps integrating this information into the input device taxonomy by Card et al. [CMR90] might enable a better comparison than currently proposed. Knowing which DOF are separable and integral might make it possible to research how they are best applied to interaction tasks, which in turn would enable us to evaluate the compatibility of the device with a technique.

Secondly, there are certain properties specific to interaction devices with a more technical nature, such as range, frequency etc. These don't directly correlate with interaction techniques, but are linked more closely with the virtual environment (portability, augmented reality...). If we consider the full scope of 3D VE design (and not just choosing interaction techniques and devices), these properties gain importance.
Another issue concerning the algorithms is the use of score lists. This works very well, but only if the scores have been accorded properly. But exactly what scores can be considered correct? This is certainly a point of debate. In this thesis, we have considered the amount of values, n, and scored the worst matching value 100*1/n, the second worst 100*2/n, and so on until the best matching receives 100*n/n, or 100%. Note that for all-or-nothing properties, the same formula can be used, giving a score of either 100% or 50%, which indicates that the worst matching value isn't necessarily bad, it just isn't as good as the best matching value.

Furthermore, we have found that the DOF Type Property is the most crucial and important to the matching process. It would have been advisable to make the comparison of this property a pruning step in the algorithm, but for the fact that there are cases were it is possible to use differing input DOF Types to execute an interaction technique. However, the algorithm results are skewed the more the property weights are equally distributed.

As mentioned for both the amount of actions in an interaction techniques and the feedback types it provides, these properties often depend on the exact implementation of the technique. After all, techniques are much easier to come by than devices. This means that for the algorithm, all possible values for these properties can apply. To appropriately select devices, however, it will be assumed that the data on which the algorithm relies will reflect the reality of the available techniques and their feedback mechanisms. Therefore, the 3D VE designer should adjust the underlying data in order for the algorithm to provide optimal results.

In case of the second algorithm, if the list of interaction techniques from the first algorithm is very large, possibly every type of feedback will be available and no pruning would take place. In that case, only the first few interaction techniques could be considered. This threshold should be adjustable. If still every type of feedback occurs, the algorithm should move on to the next step. The same can be applied to the list of input devices, for some of the other properties.

About the property of immersion level of output devices, this can be considered a subjective factor. As mentioned in section 4.3.2, it also isn't linked

to interaction techniques or input devices. In the opinion of the author, the weight of this factor should be less than for the other properties of the algorithm.

Lastly, in the second algorithm it might be possible that the scoring will cause an output device to be the best suggestion while the corresponding interaction technique or most compatible input device is rated lower in the original algorithm. This is a side effect of taking into account multiple input devices or interaction techniques into the second algorithm (depending on the direction of the first algorithm). In this case the threshold proposed in a previous issue could be used to only consider the best rating result from the first algorithm.

4.5 Conclusion

In this chapter, we have compared the taxonomies and properties of both interaction techniques and devices to determine how their properties influence each other. Based on these influences, we have proposed an algorithm that calculates compatibility scores to *match* a technique with devices or vice versa.

Another algorithm was defined, based on the same principles as the first algorithm and using the results of that algorithm to suggest output devices for the combination of interaction techniques and input devices from the first algorithm.

After discussing the influencing factors between interaction techniques and devices, and discussing the algorithms, several issues were raised concerning the topic. For some issues, solutions have been proposed, but in general it was concluded that a lot of research is still necessary to gain further knowledge of the principles applied in this chapter.

In the next chapter, we will discuss the tool implementation and the underlying database that give a practical demonstration of the algorithms in this chapter.

Chapter 5

Tool Implementation: ABITID

5.1 Introduction

Based on the taxonomies derived in the first chapters and the algorithms proposed in Chapter 4, a tool has been implemented to test the matching of interaction techniques and devices.

Underlying this tool a database has been set up to contain the data for interaction techniques and devices and their properties.

In this chapter, we will first present the database scheme that has been implemented, and secondly the tool that implements the algorithms from Chapter 4.

5.2 Database Setup

As a basis for the matching tool, a database scheme has been set up to contain the needed data. This scheme can be used for any database type. In this thesis, we have implemented the scheme in both PostgreSQL [Pos] and MySQL [MyS].

In figure 5.1, an overview of the database scheme is given. The properties of interaction techniques and devices have been implemented over several tables. We will give an overview of the database tables in the following sections.

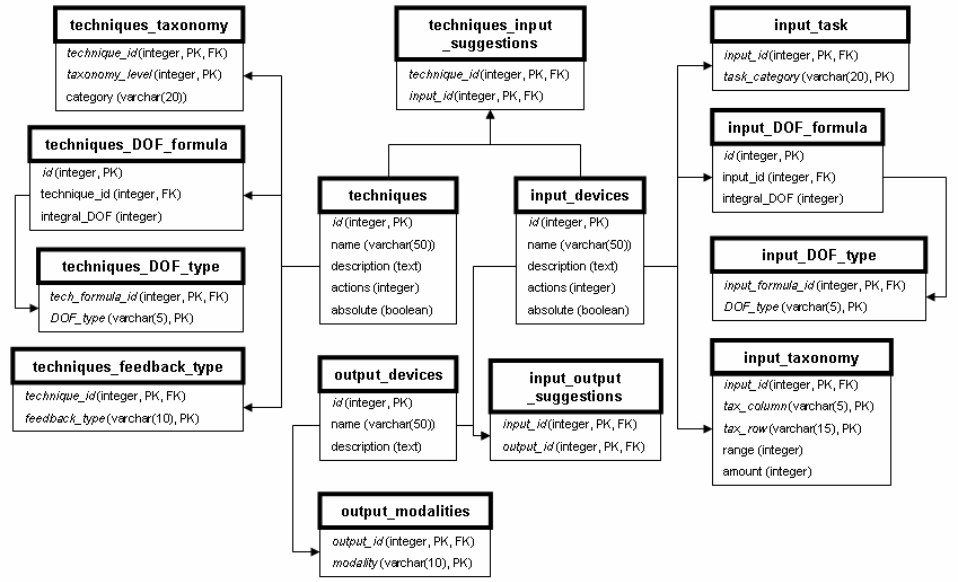


Figure 5.1: Database Scheme.

Interaction Techniques

Five tables have been used to describe interaction techniques. The main table, *techniques*, contains the technique name and possibly a description. Note that the technique name should be as specific as possible, as some techniques have many variations. Furthermore, it specifies the amount of actions of a technique and whether a technique is relative or absolute.

To account for separate and integral degrees of freedom and whether these DOF are translational or rotational, two different tables describe the DOF formula, the *techniques_DOF_formula* table and the *techniques_DOF_type* table. The first table lists each integral DOF group, while the second determines for each DOF in such a group, exactly which type it is.

The fourth table, *techniques_taxonomy*, places a technique in the taxonomy, by giving the correct category for each taxonomy level. Techniques that can be used for different tasks will be considered variations, which should be reflected in the *techniques* table. The Virtual Hand technique [PWBI98], for example, can be used for both selection and manipulation, and will thus appear in the *techniques* table twice, once as the Virtual Hand Manipulation technique, and once as the Virtual Hand Selection technique.

The last table, *techniques_feedback_type*, lists the different types of feedback an interaction technique provides. As mentioned in section 4.4, this is de-

pendent on the technique implementation and the 3D VE designer should adjust these lists to reflect the implementation available to him. In figure 5.2, the tables concerning interaction techniques are shown:

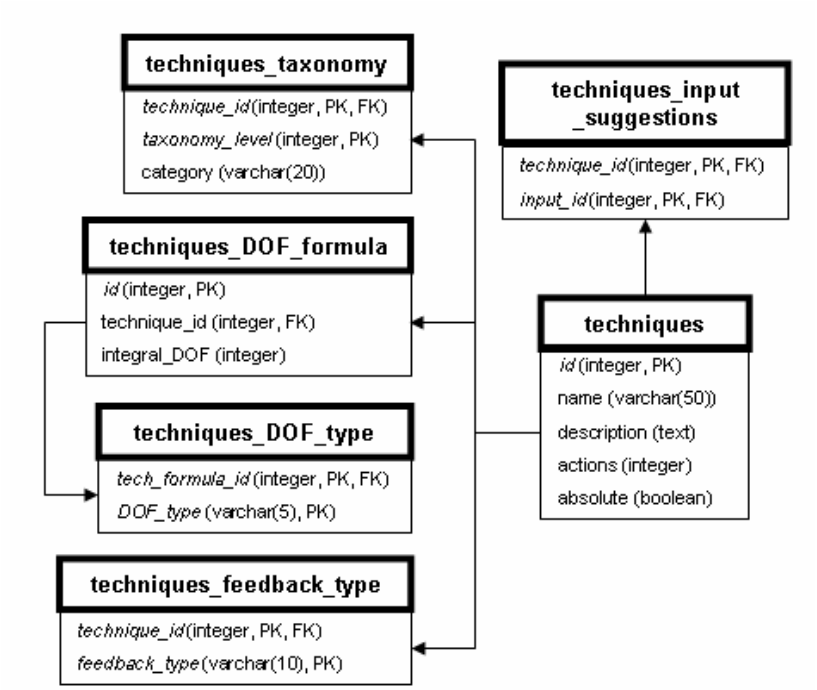


Figure 5.2: Database scheme for interaction technique data.

There is one table in figure 5.2 that hasn't been mentioned yet. The *techniques_input_suggestions* table links interaction techniques and input devices, by considering the techniques and devices that already have been matched, as explained in section 4.2.

In figure 5.3 we give a complete example of the Image Plane Head Crusher selection technique.

Input Devices

Input Devices are represented by six tables. Again, the main table, *input_devices*, stores the device name, a description, the amount of actions and whether the device is relative or absolute.

The DOF formula is represented with two tables, *input_DOF_formula* and *input_DOF_type*,which mirror the two similar tables discussed for interaction

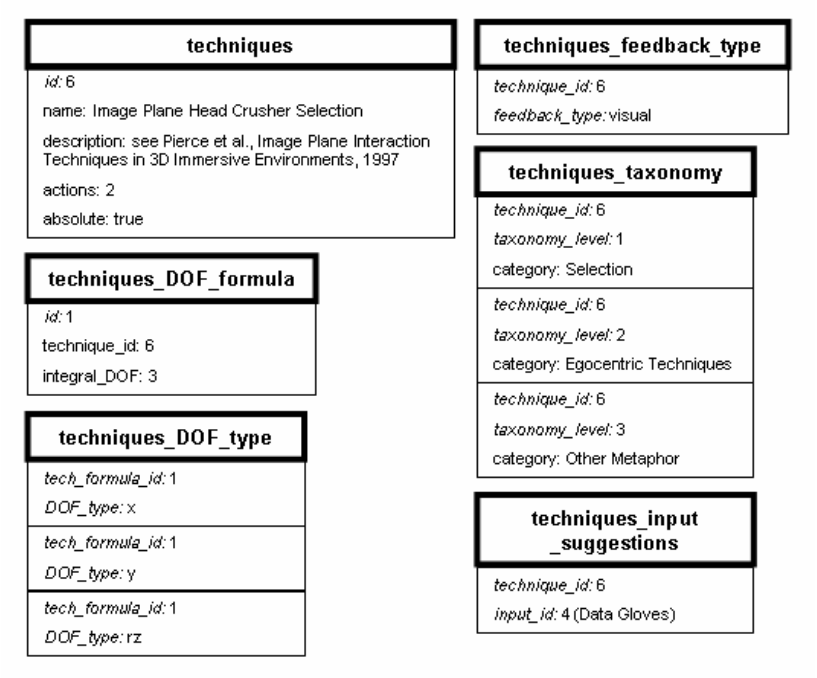


Figure 5.3: Data example: Head Crusher technique.

techniques.

The interaction tasks with which input devices can be coupled are represented in a third table, *input_task*, because one input device can be used for multiple tasks.

The fifth table to represent input devices, *input_taxonomy*, is used to place these devices into the chosen taxonomy. Each device is represented in the taxonomy by values in different cells. We implement the cells into the database based by designating the column (degrees of freedom) and row (input data type) and the value or range of each cell. To incorporate buttons or other actions of an input device, it is also possible to specify the amount of times that a certain cell is occupied by the device.

In a final table, *input_output_suggestions*, we store the suggestions for input-output device combinations, used in the second algorithm. In figure 5.4, the tables concerning interaction techniques are shown:

In figure 5.5 we give a complete example of head tracking, such as employed on Head-Mounted Displays.

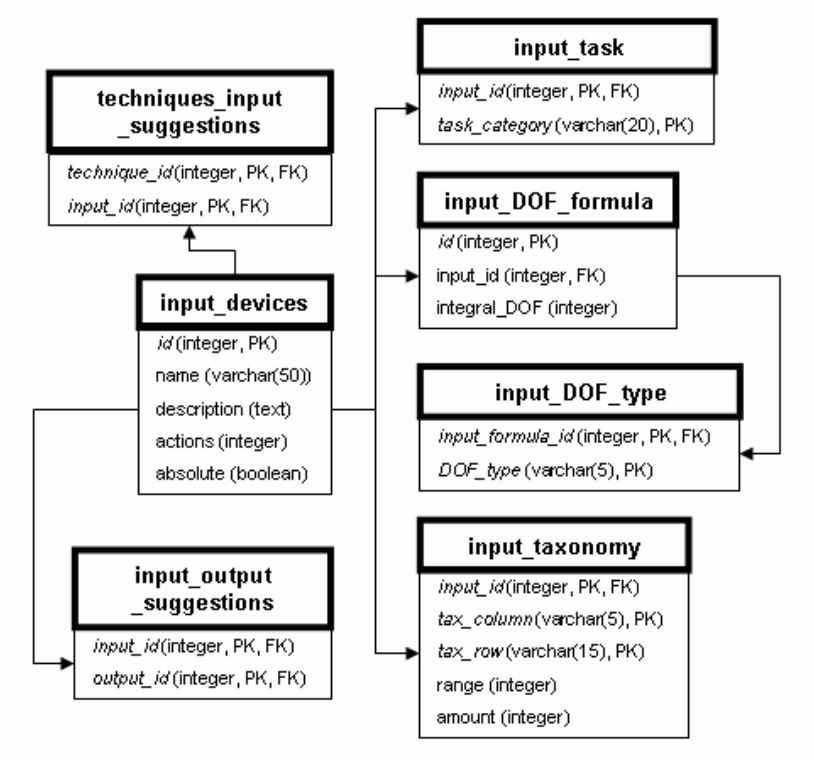


Figure 5.4: Database scheme for input device data.

Output Devices

For output devices, only two tables are needed. The main table, *output_devices*, holds the name and description of the output devices, while the other table, *output_modalities*, holds the taxonomy category. This has been moved to a separate table to account for the possibility of multimodal output, in which case it is necessary to know exactly which modalities the output device uses.

Of course, the *input_output_suggestions* table mentioned in the previous section also holds output device data. Note that devices that provide both input and output are considered as separate devices, and are linked through this table. In figure 5.6, the tables concerning interaction techniques are shown:

We complete our example from the previous section by providing the output data of a Head-Mounted Display (see figure 5.7).

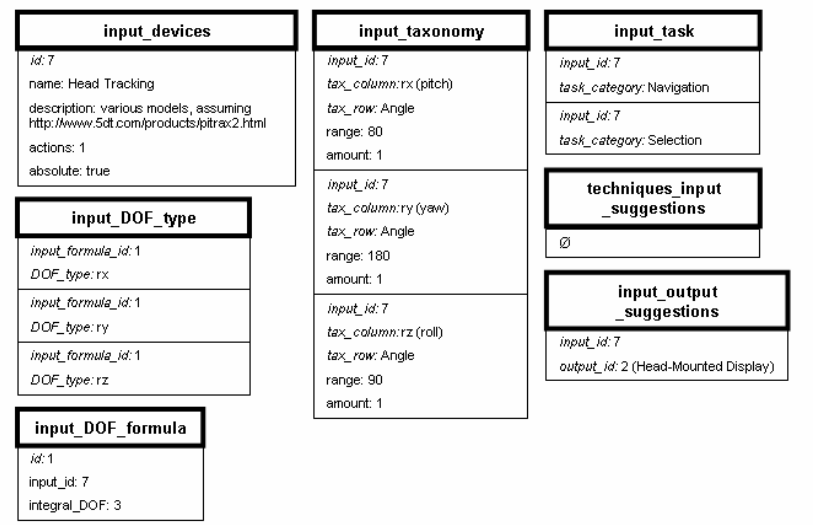


Figure 5.5: Data example: Head Tracking.

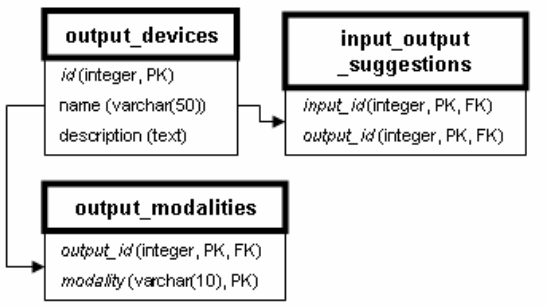


Figure 5.6: Database scheme for output device data.

5.3 ABITID

ABITID (ABstraction of Interaction Techniques and Interaction Devices) is a tool which has been implemented to test the practical application of the algorithms discussed in Chapter 4. In this section, we will present this tool through screenshots and descriptions.

ABITID has been implemented in C++ and Qt [Tro]. When the tool is launched, a database connection has to be made before the matching process can begin. The tool user can choose which database type to connect with. Database access is accomplished through the Qt SQL interfaces. A tool tip shows the possible database type codes. The connection dialog is shown in figure 5.8(a). The buttons in the start-up screen are then enabled,

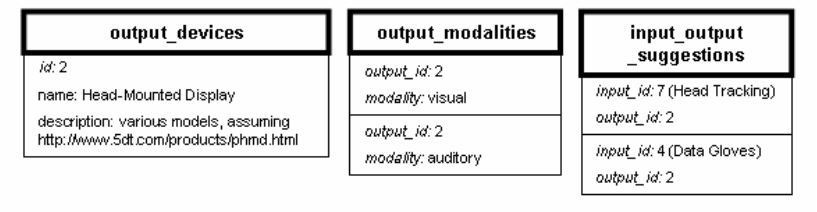
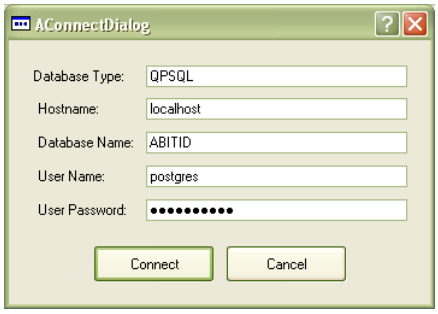


Figure 5.7: Data example: Head-Mounted Display.

as shown in figure 5.8(b).



(a) Dialog with database connection data.

(b) Start-up screen when the database connection has been made.

Figure 5.8: Screenshots of the ABITID tool implementation.

To start the matching algorithm, the tool user needs to choose either a beginning technique or device and a corresponding dialog will appear to adjust the algorithm parameters. As shown in figure 5.9, the tool user can browse the possible techniques or devices from a tree view. For techniques, the tree view is identical to the taxonomy. Devices are sorted based on the input data types they can provide (as in the input device taxonomy by Card et al. [CMR90]: position, movement, angle, delta angle...). As many devices have more than one input data type, they are sorted by the least common type first, to spread the devices over the taxonomy. Thus instead of sorting top-to-bottom left-to-right (as positioned in the taxonomy table, see figure 3.3), which would put most devices with *position* or *angle*, we sort bottom-to-top right-to-left.

As the tool user browses through the techniques or devices, the most pertinent information is shown on the right. At the bottom of the dialog, the tool user can set the weight values for the matching properties. The total weight should amount to 100, so when the user reaches this amount, the values can no longer be raised.

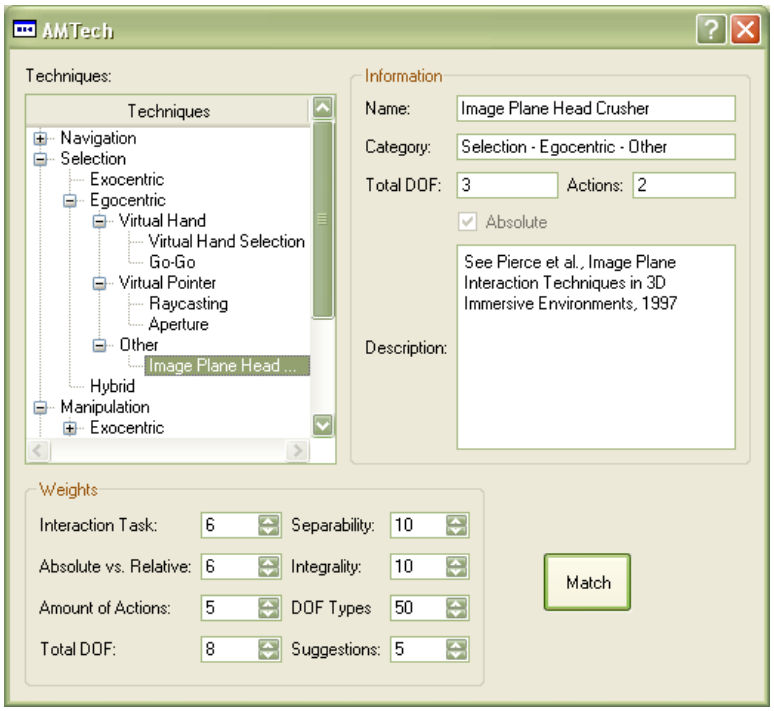


Figure 5.9: Matching Algorithm set-up screen.

When the parameters for the algorithm have been adjusted, the tool user starts the algorithm by clicking the button. A result screen, such as shown in figure 5.10 is shown.

The top part of this dialog shows the list of devices or techniques and their final scores. These are sorted according to score, but this is rearrangeable by the user. The user can save the results or return to the previous screen by clicking the appropriate buttons. Algorithm results are saved to .csv files (comma separated values), which can be opened by a spreadsheet application or loaded into a database table.

At the bottom of the dialog, a new set of weight is shown, should the user wish to continue with the second algorithm. The same constraints apply as in the previous screen. When these have been set, clicking the button on the right will start the second algorithm.

The last dialog shows the results of the second algorithm, which matches output devices with the results of the first algorithm. An example of this dialog is shown in figure 5.11. This dialog is similar to the previous one. Results are shown, ordered by final score, and can be saved to .csv file.

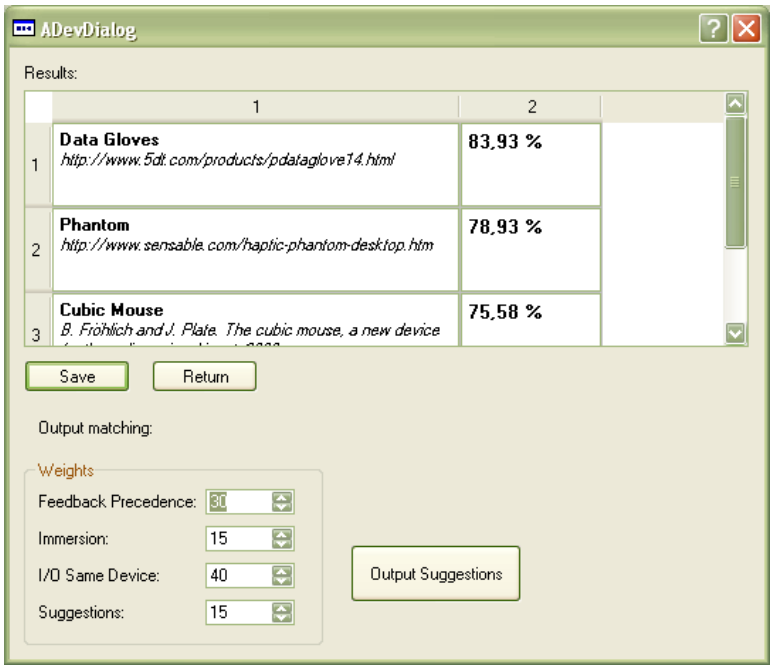


Figure 5.10: Matching Algorithm result screen.

5.4 Evaluation

As with all research, a new design should be evaluated to ascertain its use and whether it fulfills the starting hypotheses. To evaluate this tool however, it would be necessary to assemble a representative group of 3D VE designers, as that is the target audience for ABITID. A formal evaluation is therefore not feasible in the scope of this thesis.

However, we can evaluate some aspects of the tool informally. The algorithms are solid, but for the fact that they neglect user opinion and the human evaluation process which is mostly subconscious.

The database information is succinctly represented and provides all the information needed for the algorithms.

The one point which needs improvement, however, is the flexibility of the tool. Should further research uncover other links between properties, it should be possible to easily insert them into both the database and the algorithm. This is currently not at all possible. Perhaps complete flexibility is unfeasible, but certainly improvements can be made.

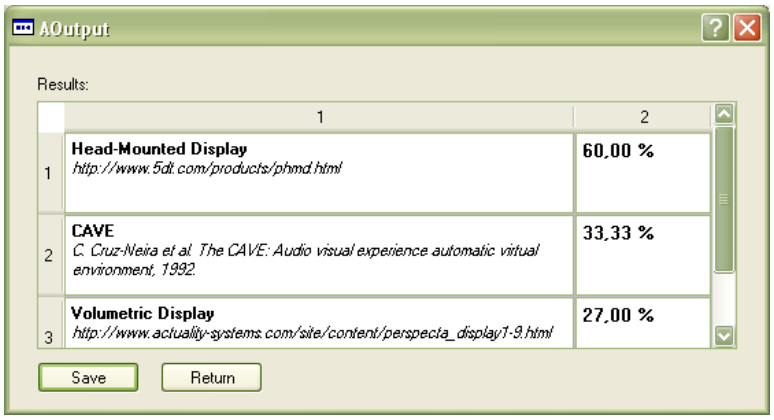


Figure 5.11: Result screen for the Output Device Suggestion Algorithm.

5.5 Conclusion

This chapter has shown the practical application of the subjects discussed in the previous chapters. A database scheme was implemented to hold all the needed information about interaction techniques and input and output devices.

Then, a tool was designed to enable 3D VE designers to apply the two matching algorithms discussed in Chapter 4 with adjustable parameters. The tool enables matching of techniques with devices and vice versa, can work with several database types, allows the 3D VE designer to set scoring weights and can save the results of the algorithms.

An informal evaluation has shown that the algorithms do not consider all factors of the matching process, partly because research into this subject is sparse, and partly because some factors are unable to be taken into account. However, the algorithms work well with the properties they now consider and the database scheme presents the data efficiently.

The one main disadvantage of the tool is its inflexibility. Should further research uncover other aspects of the matching process or links between properties, this would be difficult to incorporate in the tool.

Chapter 6

Conclusions

6.1 Introduction

This thesis aimed to help 3D VE designers find the best matching interaction techniques or devices for their 3D Virtual Environment. To reach this goal, we have first provided a solid theoretical background of both interaction techniques and input and output devices. We have analyzed their properties in Chapters 2 and 3.

We have then attempted to match these properties by determining if they had influences on each other. Several links between interaction techniques and input and output devices were thus found. Based on these links, we proposed two algorithms. The first algorithm matches a technique with possible devices or vice versa. The second algorithm matches the results of the first with possible output devices (see Chapter 4).

These algorithms have then been implemented in a tool, ABITID, with an underlying database scheme, discussed in Chapter 5. The tool enables the 3D VE designer to adjust the parameters needed for the algorithms, perform the matching process and save the results. We will now present our conclusions.

6.2 Conclusions

Several issues concerning the algorithms were discussed in section 4.4 and section 5.4. Among these was the need for human opinion to be calculated into the algorithms, however, to do so objectively would be infeasible. Secondly, this thesis remains undecided as to the best way to assign scores to properties. The score lists now discussed are very general and evenly distributed, but this may not be the best option. Another issue is the exact implementation or design of interaction techniques and devices. For one

technique or device, the varying designs can be very different. To include all possibilities in the database would again be unfeasible.

The one main point which can be concluded, however, is the lack of specific research into this subject. We have presented the algorithms based on certain properties, but we have had to make many assumptions about specifics. On many points, this thesis could be improved if more research had been available.

6.3 Future Work

There are certain venues of future work which can be pursued to improve this thesis.

Firstly, as already has been concluded, more research is needed to corroborate the assumptions taken in this thesis. Specific areas of research concern mostly the properties and links between interaction techniques and devices. Another possible venue of research is in the complete subject of 3D VE design. This thesis presents after all only an aspect of this subject, but questions as to the combination of different techniques and devices remain unanswered.

Another item on the to-do list is the flexibility of the tool. If further research uncovers new links between interaction techniques and devices or fundamental flaws in the algorithms, the tool should be easily adaptable. This is currently not the case.

Lastly, there are certain options that can be incorporated into the tool to provide more information to the 3D VE designer. These include the use of thresholds to selectively browse the algorithm results, providing resulting score lists for individual techniques or devices, and possibly using artificial intelligence to incorporate a learning curve into the algorithm, based on the tool user's opinion of previous algorithm results.

We hope that this thesis will contribute to the design, interaction and accessibility issues of 3D Virtual Environments.

Bibliography

[3Dca] 3Dconnexion. Spacemouse. `http://www.vrlogic.de/html/3dconnexion/space_mouse.html`.

[3Dcb] 3Dconnexion. Spacenavigator, the navigation-for-everyone solution. `http://www.3dconnexion.com/products/3a1d.php`.

[5DT07] 5DT. 5DT, fifth dimension technologies. `http://www.5dt.com`, 2007.

[Art] Wikipedia Article. Window, icon, menu, pointing device.

[Aud] LTB Audio. Ltb audio, listen to believe. `http://www.ltbaudio.com`.

[BB07] Aaron Bloomfield and Norman I. Badler. Collision awareness using vibrotactile arrays. In *Virtual Reality Conference, 2007. VR '07*, pages 163–170, Charlotte, NC, March 2007.

[BH97] Doug A. Bowman and Larry F. Hodges. An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In *Symposium on Interactive 3D Graphics*, pages 35–38, 182, 1997.

[BJH99] Doug A. Bowman, Donald B. Johnson, and Larry F. Hodges. Testbed evaluation of virtual environment interaction techniques. *VRST 99*, 1999.

[BKLJP04] Doug A. Bowman, Ernst Kruijff, Joseph J. LaViola Jr, and Ivan Poupyrev. *3D User Interfaces: Theory And Practice*. Addison-Wesley, 2004.

[Bux83] William Buxton. Lexical and pragmatic considerations of input structures. *SIGGRAPH Comput. Graph.*, 17(1):31–37, 1983.

[CMR90] Stuart K. Card, Jock D. Mackinlay, and George G. Robertson. The design space of input devices. *CHI '90*, 1990.

[CNSD+92] C. Cruz-Neira, D. Sandin, T. DeFanti, R. Kenyon, and J. Hart. The cave®: Audio visual experience automatic virtual environment. *Communications of the ACM*, 35(6):65–72, 1992.

[DBRC05] Joan De Boeck, Chris Raymaekers, and Karin Coninx. Are existing metaphors in virtual environments suitable for haptic interaction. In *Proc. of VRIC 2005*, pages 261–268, April 2005.

[DH07] Raimund Dachselt and Anett Hübner. Virtual environments: Three-dimensional menus: A survey and taxonomy. *Comput. Graph.*, 31(1):53–65, 2007.

[Fav02] G.E. et al. Favalora. 100 million-voxel volumetric display. In *Cockpit Displays IX: Displays for Defense Applications*, volume 4712, pages 300–312. SPIE - Int'l Soc. for Optical Eng., 2002.

[FHZ96] A. Forsberg, K. Herndon, and R. Zeleznik. Aperture based selection for immersive virtual environments. In *Proc. of UIST '96*, pages 95–96, 1996.

[FP00] Bernd Fröhlich and John Plate. The cubic mouse, a new device for three-dimensional input. In *Proceedings ACM CHI 2000*, pages 526–531, April 2000.

[Frö05] Bernd Fröhlich. The quest for intuitive 3d input devices. *Virtual Reality International*, 2005.

[FW74] J. D. Foley and V. L. Wallace. The art of natural graphic man-machine conversation. *SIGGRAPH Comput. Graph.*, 8(3):87–87, 1974.

[FWC84] James D. Foley, Victor L. Wallace, and Peggy Chan. The human factors of computer graphics interaction techniques. *IEEE Comput. Graph. Appl.*, 4(11):13–48, 1984.

[Han97] Chris Hand. A survey of 3d interaction techniques. *Computer Graphics Forum*, 16(5):269–281, 1997.

[HSK+05] A. Huckauf, A. Speed, A. Kunert, J. Hochstrate, and B. Fröhlich. Evaluation of 12-DOF input devices for navigation and manipulation in virtual environments. In *INTERACT 2005*, pages 601–614, 2005.

[Jac96] Robert J. K. Jacob. Human-computer interaction: Input devices. *ACM Computing Surveys*, 28(1), March 1996.

[JSMMJ94] Robert J. K. Jacob, Linda E. Sibert, Daniel C. McFarlane, and M. Preston Mullen Jr. Integrality and separability of input

devices. *ACM Transactions on Computer-Human Interaction*, 1(1), 1994.

[LG94] J. Liang and M. Green. Jdcad: A highly interactive 3d modeling system. *Computers and Graphics*, 18(4):499–506, 1994.

[Mat93] Larisa Matejic. Log: Building 3d user interface widgets by demonstration. Technical report, Providence, RI, USA, 1993.

[MCR90] Jock D. Mackinlay, Stuart K. Card, and George G. Robertson. Rapid controlled movement through a virtual 3d workspace. *SIGGRAPH*, pages 171–176, 1990.

[Min95] Mark R. Mine. Virtual environment interaction techniques. *Technical Report, University of North Carolina*, 1995.

[MS94] Thomas H. Massie and J. Kenneth Salisburg. The PHANToM haptic interface: A device for probing virtual objects. In *Proceedings of the 1994 ASME International Mechanical Engineering Congress and Exhibition*, volume DSC 55-1, pages 295–302, Chicago, IL, USA, November 1994.

[MWTF05] Daniel Menzel, Helmut Wittek, Gunther Theile, and Hugo Fast. The binaural sky: A virtual headphone for binaural room synthesis. In *International Tonmeister Symposium Schloss Hohenkammer*, Germany, 2005.

[MyS] MySQL. Mysql 4.1.10a. `http://www.mysql.com`.

[NM07] Takamichi Nakamoto and Hai Pham Dinh Minh. Improvement of olfactory display using solenoid valves. In *Virtual Reality Conference, 2007. VR '07*, pages 179–186, Charlotte, NC, March 2007.

[PFC+97] Jeffrey S. Pierce, Andrew Forsberg, Matthew J. Conway, Seung Hong, Robert Zeleznik, and Mark R. Mine. Image plane interaction techniques in 3d immersive environments. *1997 Symposium on Interactive 3D Graphics*, 1997.

[Pol07] Polhemus. Polhemus, first in the third dimension. `http://www.polhemus.com/?page=Motion_Fastrak`, 2007.

[Pos] PostgreSQL. Postgresql 8.1. `http://www.postgresql.org`.

[PWBI98] I. Poupyrev, S. Weghorst, M. Billinghurst, and T. Ichikawa. Egocentric object manipulation in virtual environments: Empirical evaluation of interaction techniques. *EUROGRAPHICS '98*, 17(3), 1998.

[SCP95] Richard Stoakley, Matthew J. Conway, and Randy Pausch. Virtual reality on a wim: Interactive worlds in miniature. *SIGCHI*, pages 265–272, 1995.

[Sys01] Fakespace Systems. Pinch, the simple effective way to get your hands in a virtual world. `http://www.fakespacesystems.com/pinch.htm`, 2001.

[TRC01] Desney S. Tan, George G. Robertson, and Mary Czerwinski. Exploring 3d navigation: Combining speed-coupled flying with orbiting. *SIGCHI*, pages 418–425, 2001.

[Tro] Trolltech. Qt 4.1. `http://trolltech.com/`.

[Vic07] Vicon. Vicon tracker real time 3d optical tracking system. `http://www.vicon.com/downloads/Vicon_Tracker_lr.pdf`, 2007.

[WHB06] Chadwick A. Wingrave, Yonca Haciahmetoglu, and Doug A. Bowman. Overcoming world in miniature limitations by a scaled and scrolling wim. In *3DUI '06: Proceedings of the 3D User Interfaces (3DUI'06)*, pages 11–16, Washington, DC, USA, 2006. IEEE Computer Society.

[Wik] Wikipedia. Head-mounted display. `http://en.wikipedia.org/wiki/Head-mounted_display`.

Auteursrechterlijke overeenkomst

Opdat de Universiteit Hasselt uw eindverhandeling wereldwijd kan reproduceren, vertalen en distribueren is uw akkoord voor deze overeenkomst noodzakelijk. Gelieve de tijd te nemen om deze overeenkomst door te nemen, de gevraagde informatie in te vullen (en de overeenkomst te ondertekenen en af te geven).

Ik/wij verlenen het wereldwijde auteursrecht voor de ingediende eindverhandeling:
Abstraction of Interaction Techniques and Devices
Richting: **Master in de informatica** Jaar: **2007**
in alle mogelijke mediaformaten, - bestaande en in de toekomst te ontwikkelen - , aan de Universiteit Hasselt.

Niet tegenstaand deze toekenning van het auteursrecht aan de Universiteit Hasselt behoud ik als auteur het recht om de eindverhandeling, - in zijn geheel of gedeeltelijk -, vrij te reproduceren, (her)publiceren of distribueren zonder de toelating te moeten verkrijgen van de Universiteit Hasselt.

Ik bevestig dat de eindverhandeling mijn origineel werk is, en dat ik het recht heb om de rechten te verlenen die in deze overeenkomst worden beschreven. Ik verklaar tevens dat de eindverhandeling, naar mijn weten, het auteursrecht van anderen niet overtreedt.

Ik verklaar tevens dat ik voor het materiaal in de eindverhandeling dat beschermd wordt door het auteursrecht, de nodige toelatingen heb verkregen zodat ik deze ook aan de Universiteit Hasselt kan overdragen en dat dit duidelijk in de tekst en inhoud van de eindverhandeling werd genotificeerd.

Universiteit Hasselt zal mij als auteur(s) van de eindverhandeling identificeren en zal geen wijzigingen aanbrengen aan de eindverhandeling, uitgezonderd deze toegelaten door deze overeenkomst.

Ik ga akkoord,

Vicky Pagnaer

Datum: **24.08.2007**