Multimodal Selection in Virtual Environments:
Enhancing the User Experience and Facilitating Development
Saying “Thank You” to someone does not necessarily mean a lot, but at the end of my journey towards my PhD, it is the only way of showing my gratitude to a multitude of people.

While writing this section, I also reflected on the last four years and could only conclude that I would like to thank myself for starting a PhD. In the past four years, I have experienced many things which I probably would have never experienced otherwise: performing the research/work I’m interested in doing, writing papers about it, going to conferences, visiting foreign places, meeting a lot of nice people, and probably many other things which I forgot…

I would not be writing this word of thanks, if it would not be for two persons in particular, first and foremost my promotor prof. dr. Karin Coninx. She allowed me to explore my own interests in the topic of multimodal interaction and supported me with guidance when necessary. This dissertation especially, would not be in today’s state without her. Second, without prof. dr. Chris Raymaekers, who supervised me during my MSc. thesis and asked me if I would be interested in pursuing a PhD, I might have never started one. Next to boosting my interests in obtaining a PhD, he also guided me during my research.

One important colleague researcher also deserves a word of gratitude: Tovi Grossman. The experience of collaborating with him was truly unique. It made me think differently and become even more critical towards details during user experiments. Without this collaboration, it is impossible to predict what would have come from my PhD as a very substantial part of the first part of it, is our collaborative work.

These four years, I have had the pleasure of working in a lab with many fun colleagues, the list is rather extensive due to regular place switching: Anastasiia Beznosyk, Carl Bruninx, Maarten Cardinaels, Erwin Cuppens, Joan De Bocck, Tom De Weyer, Sofie Notelaers, Johanna Renny Octavia, Tim Tutenel, Davy Vanacken and Geert
Vanderhulst. Furthermore, I would like to thank the rest of the HCI group and other colleagues and ex-colleagues for the experiences we shared while working together. A special word of thanks goes to Jo Vermeulen who is always open for discussion and has the correct mindset to perform research.

Without the EDM, I would never have been able to finish my PhD. Therefore I want to thank the EDM and its management Prof. Dr. Eddy Flerackers and Prof. dr. Frank Van Reeth, together with all the other professors which are involved in making the EDM what it is today.

I am also grateful to the EDM secretariat, Ingrid Konings and Roger Claes, for all the things they arranged for me during the past four years; they made a lot of administrative work easy.

Finally, I would like to thank my parents for giving me as many opportunities as possible during my life. They gave me a lot of freedom in my decisions and when necessary always supported and guided me. Furthermore, I would like to thank my girlfriend Vicky for always being there for me and for answering all my questions with regard to English. And of course, I would like to thank my brother and my friends for cheering me up when necessary and providing me with many pleasant diversions.
Abstract

The usage of virtual environments is manifold, from visualisation and interaction with molecular data to medical applications. Virtual environments are represented in three dimensions and display technology to visualise such environments has significantly improved. Besides visual feedback, other types of feedback in virtual environments, such as haptics and audio, are gaining importance. We explore the possible benefits of these types of feedback. In this dissertation we focus on one of the four primary interaction techniques which must be supported for interactive virtual environments, namely selection. Our efforts during this dissertation are twofold.

Firstly, we perform research into selection techniques augmented with multimodal feedback. Several newly designed selection techniques are compared taking into account new factors such as the environment density and occlusion. Results show that our techniques could manage these factors and that multimodal feedback provides small non-significant improvements. In our experiments the force feedback was designed using a pilot test or a trial-and-error approach, therefore we try to formulate a guideline to design force feedback. We perform a series of multidirectional point-select experiments which show that the definite integral of the force profile can be used as a guideline to know when a certain force feedback value will cause the performance of the user to deteriorate during his movement.

Secondly, we propose approaches to integrate semantic and contextual knowledge in a model-based user interface design process for virtual environments. Semantic information is added through the addition of a data type to the interaction description model (NiMMiT) and the automatic generation of a speech grammar. With regard to context, we propose a context system based on the ‘Event-Condition-Action’ paradigm which is realised at the dialog level instead of the usual task level. Several case studies show that these additions can be useful during the design of interaction techniques including selection techniques.
# Contents

1 Introduction 1
   1.1 Challenges and Aims ........................................ 2
   1.2 Overview ...................................................... 3

I Multimodal Selection Techniques 5

2 An Overview of Selection and Multimodal Feedback 9
   2.1 Introduction .................................................... 9
   2.2 3D Selection Techniques ....................................... 10
      2.2.1 Virtual Hand ............................................. 12
      2.2.2 Virtual Pointer .......................................... 13
   2.3 Multimodal Feedback .......................................... 14
      2.3.1 Haptics ................................................. 16
      2.3.2 Audio ................................................... 18
   2.4 Conclusion .................................................... 20

3 Multimodal Feedback for Selection Techniques 21
   3.1 Introduction .................................................... 21
   3.2 Selection Techniques .......................................... 22
      3.2.1 Virtual Hand ............................................. 22
3.2.2 Aperture ........................................ 23
3.2.3 The 2D Bubble Cursor .......................... 24
3.3 Force and Audio Feedback for Selection Metaphors .... 26
  3.3.1 Force Feedback ................................. 26
  3.3.2 Audio Feedback ................................. 27
3.4 Multimodal Feedback ............................... 27
  3.4.1 Force Feedback for the Virtual Hand ............ 27
  3.4.2 Force Feedback for an Area Cursor .............. 30
  3.4.3 Audio Feedback ................................. 31
3.5 Multimodal Feedback Experiment ........................ 31
  3.5.1 Apparatus ...................................... 32
  3.5.2 Participants ................................... 32
  3.5.3 Procedure ..................................... 32
  3.5.4 Independent Variables .......................... 33
  3.5.5 Design ......................................... 34
  3.5.6 Results ........................................ 35
3.6 Conclusion ........................................ 40

4 Influence of Density and Occlusion on Selection 43
  4.1 Introduction ...................................... 43
  4.2 Related Work ..................................... 45
    4.2.1 Selection Techniques for Dense Environments .. 45
    4.2.2 Overcoming Target Occlusion ................... 47
    4.2.3 Adding Multimodal feedback .................... 47
    4.2.4 Summary ..................................... 47
  4.3 Design Guidelines and Strategies .................... 48
    4.3.1 Design Guidelines ............................ 48
    4.3.2 Increased and Unambiguous Activation Areas .. 49
    4.3.3 Integrated Visual Enhancements for Occluded Targets ... 49
5.2.2 Comparison of Haptic Devices .................................. 85
5.3 Experiment 1: Force Amplitude and Direction .................. 86
  5.3.1 Participants ................................................. 86
  5.3.2 Apparatus .................................................. 86
  5.3.3 Procedure .................................................. 87
  5.3.4 Independent Variables .................................... 89
  5.3.5 Design ...................................................... 89
  5.3.6 Results ..................................................... 90
5.4 Experiment 2: Force Shape and Duration ....................... 94
  5.4.1 Participants ................................................. 95
  5.4.2 Apparatus and Procedure .................................. 95
  5.4.3 Independent Variables .................................... 95
  5.4.4 Design ...................................................... 96
  5.4.5 Results ..................................................... 97
5.5 Experiment 3: Subjective Preference ............................ 102
  5.5.1 Apparatus and Procedure .................................. 103
  5.5.2 Participants ................................................. 103
  5.5.3 Stimuli and Design ....................................... 103
  5.5.4 Results ..................................................... 104
5.6 Investigation of the Device Influence ........................... 104
  5.6.1 PHANToM versus Falcon ................................... 105
5.7 Experiment 4: Objective Device Comparison .................... 107
  5.7.1 Apparatus and Procedure .................................. 108
  5.7.2 Participants ................................................. 109
  5.7.3 Independent Variables .................................... 109
  5.7.4 Design ...................................................... 109
  5.7.5 Results ..................................................... 110
5.8 Experiment 5: Subjective Device Comparison .................... 113
  5.8.1 Apparatus, Procedure and Participants .................... 114
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.5.1</td>
<td>Process</td>
<td></td>
<td>159</td>
</tr>
<tr>
<td>7.5.2</td>
<td>Generating the Speech Grammar</td>
<td></td>
<td>159</td>
</tr>
<tr>
<td>7.5.3</td>
<td>Reference Resolution</td>
<td></td>
<td>161</td>
</tr>
<tr>
<td>7.5.4</td>
<td>Case Study: a “City Park Designer”</td>
<td></td>
<td>162</td>
</tr>
<tr>
<td>7.6</td>
<td>Conclusion</td>
<td></td>
<td>163</td>
</tr>
<tr>
<td>8</td>
<td>Extending a dialog model with Contextual Knowledge</td>
<td>165</td>
<td></td>
</tr>
<tr>
<td>8.1</td>
<td>Introduction</td>
<td></td>
<td>165</td>
</tr>
<tr>
<td>8.2</td>
<td>Context and NiMMiT</td>
<td></td>
<td>167</td>
</tr>
<tr>
<td>8.2.1</td>
<td>Context in Task and Dialog Modelling</td>
<td></td>
<td>167</td>
</tr>
<tr>
<td>8.2.2</td>
<td>Modality Constraints in Task Modelling</td>
<td></td>
<td>168</td>
</tr>
<tr>
<td>8.2.3</td>
<td>Context-aware Modality Selection at the Dialog Level</td>
<td></td>
<td>169</td>
</tr>
<tr>
<td>8.2.4</td>
<td>Context-aware Task Selection at the Dialog Level</td>
<td></td>
<td>170</td>
</tr>
<tr>
<td>8.3</td>
<td>Case Study</td>
<td></td>
<td>173</td>
</tr>
<tr>
<td>8.3.1</td>
<td>Setup</td>
<td></td>
<td>173</td>
</tr>
<tr>
<td>8.3.2</td>
<td>Creation</td>
<td></td>
<td>174</td>
</tr>
<tr>
<td>8.3.3</td>
<td>Context Integration</td>
<td></td>
<td>176</td>
</tr>
<tr>
<td>8.4</td>
<td>Conclusion</td>
<td></td>
<td>177</td>
</tr>
<tr>
<td>9</td>
<td>Designing Context-Aware Multimodal Virtual Environments</td>
<td>179</td>
<td></td>
</tr>
<tr>
<td>9.1</td>
<td>Introduction</td>
<td></td>
<td>179</td>
</tr>
<tr>
<td>9.2</td>
<td>Related Work</td>
<td></td>
<td>180</td>
</tr>
<tr>
<td>9.2.1</td>
<td>Multimodal User Interface Frameworks</td>
<td></td>
<td>180</td>
</tr>
<tr>
<td>9.2.2</td>
<td>Context Modelling</td>
<td></td>
<td>181</td>
</tr>
<tr>
<td>9.2.3</td>
<td>Context in Virtual Environments</td>
<td></td>
<td>182</td>
</tr>
<tr>
<td>9.3</td>
<td>Defining a Context System</td>
<td></td>
<td>182</td>
</tr>
<tr>
<td>9.3.1</td>
<td>Context Detection and Switching</td>
<td></td>
<td>182</td>
</tr>
<tr>
<td>9.3.2</td>
<td>Context Detection and Switching Using NiMMiT</td>
<td></td>
<td>183</td>
</tr>
<tr>
<td>9.4</td>
<td>Tool Support</td>
<td></td>
<td>187</td>
</tr>
<tr>
<td>9.5</td>
<td>Case Studies</td>
<td></td>
<td>189</td>
</tr>
</tbody>
</table>
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.5.1 Case Study 1: Pose-Aware Interaction</td>
<td>189</td>
</tr>
<tr>
<td>9.5.2 Case Study 2: Preference-Aware Selection Techniques</td>
<td>192</td>
</tr>
<tr>
<td>9.6 Discussion</td>
<td>195</td>
</tr>
<tr>
<td>9.7 Conclusion</td>
<td>197</td>
</tr>
<tr>
<td><strong>10 Conclusions and Future Research Directions</strong></td>
<td>201</td>
</tr>
<tr>
<td>10.1 Conclusions</td>
<td>201</td>
</tr>
<tr>
<td>10.2 Future Research Directions</td>
<td>202</td>
</tr>
<tr>
<td>10.3 Scientific Contributions and Publications</td>
<td>204</td>
</tr>
<tr>
<td><strong>A Nederlandstalige Samenvatting (Dutch Summary)</strong></td>
<td>209</td>
</tr>
<tr>
<td><strong>B An Automatic Generated Speech Grammar</strong></td>
<td>213</td>
</tr>
<tr>
<td><strong>C User Experiment Questionnaires</strong></td>
<td>217</td>
</tr>
</tbody>
</table>
List of Figures

2.1 Interaction in virtual environments. (Poupyrev and Ichikawa, 1999) . 10
2.2 Classification of selection techniques by task decomposition. (Bow- 11
man et al., 2004) ............................................
2.3 Classification of selection techniques by metaphor. (Poupyrey et al., 12
1998) ..........................................................
2.4 The virtual hand metaphor. (Poupyrev et al., 1996) ............... 13
2.5 Two ray casting metaphors. .......................................... 14
2.6 Commonly used definitions related to the term haptics. (Oakley et al., 17
2000) ..........................................................
2.7 Haptic devices. ...................................................... 18

3.1 The Selection Metaphors that were used in this research. ....... 23
3.2 The aperture technique illustrated. (Forsberg et al., 1996) ... 23
3.3 A 2D Voronoi diagram. ............................................. 25
3.4 An abstract 2D representation of two gravity wells with labels point- 29
ing to the different possible parameters of a gravity well. ........
3.5 The scenes of the pilot test experiment in order to define the gravity 30
well parameters. ................................................
3.6 An overview of the audio feedback during the selection process. 31
3.7 The experiment apparatus. ......................................... 33
3.8 The scene being used during the experiment. The figure was altered 34
for a better representation: the original background colour was black.
LIST OF FIGURES

3.9 Average trial completion, reach and reaction time for all conditions.
A legend of all the conditions can be found in Table 3.1.

4.1 (a) The 3D bubble cursor is rendered as a semi-transparent sphere
which dynamically resizes such that it only captures the closest target,
here highlighted yellow. (b) When necessary, a second sphere
is rendered around the captured target so that it always appears to be
completely contained by the 3D bubble cursor.

4.2 (a) Targets in close proximity to the 3D bubble cursor become semi-
transparent. (b) As the cursor approaches an occluded goal target (red
cube) it becomes visible and can be selected.

4.3 (a) The depth ray selects the intersected target which is closest to the
depth marker. (b) The depth marker position can be controlled by
moving the hand forwards or backwards.

4.4 (a) The 3D bubble cursor divides the environment into 3D Voronoi
regions. The effective width of each target is defined by its corre-
sponding Voronoi region. (b) When using the depth ray, the Voronoi
region is based only on the intersected targets. (c) Changing the po-
sition of the ray can change a target’s effective width.

4.5 The experiment apparatus.

4.6 The experiment environment.

4.7 Six distractor targets are carefully positioned around the goal target,
creating a cube-shaped Voronoi region. The distance between these
targets and goal target is the density spacing ($DS$).

4.8 Technique completion times by density.

4.9 Technique completion times by visibility condition.

4.10 Technique completion times by target size.

4.11 3D Bubble cursor times by size and density spacing.

4.12 Learning effects for each of the techniques.

4.13 An overview of the multimodal feedback for the 3D bubble cursor.

4.14 The experiment apparatus (stereo is disabled for illustrative purposes).

4.15 Technique completion times by multimodal feedback.

4.16 Completion times by density and multimodal feedback.
LIST OF FIGURES

4.17 Technique completion times by density. 72
4.18 Technique completion times by visibility. 73
4.19 Technique completion times by target size. 74
4.20 3D Bubble cursor times by density and target size. 75
4.21 Questionnaire scores from the subjective feedback (1 = Strongly Disagree, 5 = Strongly Agree). 77

5.1 An illustration of the force evolution of a sinusoidal haptic bump over time. 83
5.2 The PHANToM and the forcemeter being used to calibrate it. 87
5.3 Setup of Experiments 1, 2 and 3: PHANToM haptic device and the ISO 9241-9 tapping circle on the screen. 88
5.4 The measures used during motion path analysis. (MacKenzie et al., 2001) 90
5.5 Learning effect of the experiment. 91
5.6 Pair wise post hoc comparisons of the force integral levels for trial completion time, all significant differences (p < .05) are shaded. 92
5.7 Trial completion times for force integral conditions. Note that the y-axis does not start at zero to better show the differences between FI conditions. 93
5.8 Force integral values by shape. 98
5.9 Velocity Profiles of some “typical” movements: Upper graphs contain the lateral velocity (orthogonal with moving direction), lower graphs contain the longitudinal velocity (parallel with moving direction). 99
5.10 Transfer functions according to the x-axis for both devices. 106
5.11 Setup of Experiments 4 and 5: PHANToM and Falcon haptic device with the ISO 9241-9 tapping circle on the screen. 108
5.12 Force integral values by input device. Note that the y-axis does not start at zero to better show the differences between FI conditions. 111
5.13 Force integral values by shape. 111
5.14 Velocity Profile of the PHANToM and Falcon. The topmost graph is the lateral velocity, the bottommost graph represents the longitudinal velocity. 112
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1</td>
<td>The VR-DeMo Approach. (Coninx et al., 2006b)</td>
<td>130</td>
</tr>
<tr>
<td>6.2</td>
<td>The process used to model the virtual world. (Coninx et al., 2006b)</td>
<td>131</td>
</tr>
<tr>
<td>6.3</td>
<td>An excerpt of an ontology.</td>
<td>131</td>
</tr>
<tr>
<td>6.4</td>
<td>The model-based user interface design process used in VR-DeMo and CoGenIVE.</td>
<td>133</td>
</tr>
<tr>
<td>6.5</td>
<td>An example of a NiMMiT diagram representing an interaction technique in which the user can select and move an object.</td>
<td>136</td>
</tr>
<tr>
<td>6.6</td>
<td>Designing the Dialog Model.</td>
<td>139</td>
</tr>
<tr>
<td>6.7</td>
<td>Designing the Interaction Model using the NiMMiT notation.</td>
<td>140</td>
</tr>
<tr>
<td>7.1</td>
<td>The Concept data type.</td>
<td>147</td>
</tr>
<tr>
<td>7.2</td>
<td>The concept dialog in CoGenIVE. with the current values of the constant label (left) and the concepts of the current project (right).</td>
<td>148</td>
</tr>
<tr>
<td>7.3</td>
<td>The newly introduced predefined tasks.</td>
<td>149</td>
</tr>
<tr>
<td>7.4</td>
<td>The selection task which only selects objects of certain concept(s).</td>
<td>150</td>
</tr>
<tr>
<td>7.5</td>
<td>The setup of the driving simulator.</td>
<td>152</td>
</tr>
<tr>
<td>7.6</td>
<td>The driving interaction technique.</td>
<td>154</td>
</tr>
<tr>
<td>7.7</td>
<td>A semantic manipulation technique.</td>
<td>157</td>
</tr>
<tr>
<td>7.8</td>
<td>All steps performed during speech input.</td>
<td>160</td>
</tr>
<tr>
<td>7.9</td>
<td>Parts of the generated speech grammar.</td>
<td>161</td>
</tr>
<tr>
<td>7.10</td>
<td>SPARQL queries: (a) query structure: [concept] [object relation] [concept]/[instance] (b) example of (a): “bench left of fountain”. (c) Query structure: [data-property-value]* [concept] (d) example of (c): “Brown obelisk”.</td>
<td>162</td>
</tr>
<tr>
<td>7.11</td>
<td>Impression of the virtual city park.</td>
<td>163</td>
</tr>
<tr>
<td>8.1</td>
<td>(a) Combining modality constraints with the decision task notation (task model and dialog model). (Clerckx et al., 2007) (b) Merged dialog models.</td>
<td>168</td>
</tr>
<tr>
<td>8.2</td>
<td>(a) An abstract NiMMiT example. (b) The context view of an abstract NiMMiT Diagram, ‘EVENT1’ and ‘EVENT3’ were added to a specific context.</td>
<td>170</td>
</tr>
</tbody>
</table>
### LIST OF FIGURES

8.3 Task model in which different selection tasks (bubble cursor and depth ray) can be chosen depending on the context. ............... 171

8.4 (a) Two dialog models generated from the task model. (b) The merged dialog model, containing the ‘Selection’ decision task. (c) NiMMiT diagram without context arrows, representing the ‘Selection’ decision task. (d) NiMMiT diagram with context arrows, assigned to the corresponding context (bubble cursor or depth ray), representing the ‘Selection’ decision task. .................................................. 172

8.5 Setup of the case study: a wall projection combined with a tracked glove, speech and a desktop setup with mouse and keyboard. ........ 173

8.6 The ConcurTaskTree of the case study. ................................. 174

8.7 Active events in the Select interaction. ................................. 175

8.8 NiMMiT Diagram of the Select Interaction with the context arrows. 175

8.9 NiMMiT Diagram of the Select Interaction in the Wall context. .... 176

9.1 The context detection diagram from the first case study (pose-aware interaction) with the ‘GESTURE’ interface. .................. 184

9.2 Context Handling. ................................................................. 186

9.3 CoGenIVE dialog to define context units and their values. ........ 187

9.4 Events Pane in CoGenIVE, including context events. .............. 188

9.5 The setup used in the pose-aware interaction case study. ........... 189

9.6 The context detection diagram from the pose-aware interaction case study using a head tracker. ................................. 190

9.7 The context switching diagram from the pose-aware interaction case study. .................................................. 192

9.8 The context detection and switching diagram from the preference-aware selection techniques case study. .................. 194

9.9 Part of a NiMMiT diagram to change the users preference with regard to the force strength using a label. .................. 196

C.1 A (translated from Dutch) copy of the questionnaire administered after the user experiment described in Chapter 3. ............... 218

C.2 A copy of the questionnaire administered after the first user experiment described in Chapter 4 in Section 4.5. .................. 219
C.3 A copy of the questionnaire administered after the second user experiment described in Chapter 4 in Section 4.6. . . . . . . . . . . . . . 220
C.4 A copy of the questionnaire administered after the first user experiment described in Chapter 5 in Section 5.3. . . . . . . . . . . . . . 222
C.5 A copy of the questionnaire administered after the second user experiment described in Chapter 5 in Section 5.4. . . . . . . . . . . . . . 223
C.6 A copy of the questionnaire administered after the third user experiment described in Chapter 5 in Section 5.8. . . . . . . . . . . . . . 224
List of Tables

3.1 Overview of all the different selection conditions being used during the experiment. .................................................. 35

3.2 Post hoc comparison table of different conditions. A dash means that that particular modality was not taken into account. A legend of all the conditions can be found in Table 3.1. . . . . . . . . . . . . . . . . 37

5.1 Equivalent force amplitudes calculated using the force integral value and the definite integral. . . . . . . . . . . . . . . . . . . . 96

5.2 Identified parameters (mass \( m \) and damping \( b \)) of both the PHAN-ToM and the Falcon. . . . . . . . . . . . . . . . . . . . . . . . 107
Chapter 1

Introduction

3D Virtual Environments (VEs) have a lot of different application areas. They are used, for instance, to visualise or interact with 3D data such as molecules or medical scans. They are also used to design and model new prototypes. A virtual environment is a computer-simulated environment. Such an environment can be completely generated by the computer or from other media such as digital cameras. These environments can resemble the real world but are not restricted by it. All virtual environments represent the virtual world in three dimensions, this enables users to interact with the virtual world as if they were there. Although the world is represented in three dimensions, other parts of the user interface, such as menus, could be in two dimensions.

It is important to perceive virtual environments realistically. A lot of attention has already been paid to the visual aspect of virtual environments. Different solutions exist to visually experience them; three dimensional display technologies, such as immersive Virtual Reality (VR) systems (Buxton and Fitzmaurice, 1998), or non-immersive fish-tank VR systems using LCD shutter stereo-glasses (Ware et al., 1993), have significantly improved in display quality. Experimental evaluations have also shown that these displays can improve the user’s ability to perceive virtual 3D scenes (Ware and Franck, 1996), making them a beneficial alternative for 3D applications on 2D display technologies. In this dissertation the virtual environments will be using non-immersive stereoscopic visual feedback. Besides visual feedback, other types of feedback in virtual environments, such as haptics and audio, are gaining importance. These other types of feedback can provide the user with extra information during his
interactions with the virtual environment. We will explore the possible benefits of these types of feedback.

One of the primary techniques which must be supported in any interactive application is object selection. Researchers of 3D virtual environments have often categorised selection as one of the four basic interactions (along with navigation, manipulation, and data input) (Mine, 1995; Bowman et al., 2004). Selection allows users to specify an object, which they wish to manipulate or with which they want to interact. Without being able to specify an object, it would therefore be impossible to interact in the virtual environment which makes selection an essential task. However, when selecting objects in a 3D environment, other options than the traditional standard 2D mouse should be explored, as the targets will be positioned in a virtual world with three dimensions. As such, it is important for researchers to consider new selection techniques, specifically designed for 3D environments. Because it is such a critical task, there has been a wide variety of research looking into various techniques for supporting selection in a virtual environment. But not all factors of the environment, such as density and/or occlusion, have been fully explored.

Apart from research into selection techniques themselves, it is also important to consider the realisation of such techniques. The implementation of interaction, in our case selection techniques, is usually complex (Wingrave, 2008). Besides the implementation of such techniques, there is usually no possibility to reuse them in other applications or they are difficult to reproduce in another context of use. Furthermore, extra available information such as semantic or contextual knowledge is interesting to integrate during the design phase of the virtual environment. Not much attention is paid to how techniques are developed or integrated in applications; this is also an important aspect researchers have to take into account.

In this dissertation, through formal experiments, we will investigate different selection techniques augmented with multimodal feedback. The focus will be on non-moving abstract objects (e.g. a box or sphere) of a certain size inside a 3D virtual world. We will also explore ways to design and integrate selection in virtual environments using semantic and contextual knowledge. In Section 1.2 we will provide an overview of this dissertation, but first we will list the challenges and aims.

1.1 Challenges and Aims

The challenges and aims of this dissertation are:
1.2 Overview

- To compare several selection techniques and evaluate them in formal experiments.
- To investigate the influence of the environment density and occlusion on selection techniques.
- To explore the addition of multimodal feedback to several selection techniques.
- To investigate which force feedback magnitudes do not deteriorate the performance of the user’s movements during selection.
- To study the addition of semantic information to a model-based design process for virtual environments and more in particular how it can benefit the realisation and integration of selection.
- To introduce context at the dialog level instead of, as is usual, at the task level.
- To present a context system which can aid in designing context-aware virtual environments.

1.2 Overview

This dissertation will consist of two parts, of which the first part will discuss our formal experiments with regard to selection techniques augmented with multimodal feedback and the second part will investigate how we can use semantic and contextual knowledge during the design and integration of selection techniques.

The first chapter of Part I will introduce and discuss important related work. In Chapter 3 a first experiment will be performed comparing several selection techniques augmented with multimodal feedback such as force feedback and audio. Of these selection techniques the 2D bubble cursor performed the best overall but some issues were identified. Therefore Chapter 4 will investigate the influence of environment density and target occlusion. We successfully designed a 3D bubble cursor and depth ray that can manage these factors with only a minimal overhead. To these designed selection techniques, we again add multimodal feedback, which shows small but non-significant improvements. In these experiments, the force feedback design was decided by a small pilot test or trial-and-error design. As these ways of designing force feedback are non-sufficient, Chapter 5 investigates which force feedback magnitudes do not deteriorate the performance of the user’s movements during selection and tries to find a guideline which can be used to design force feedback.
In Part II we change the topic from designing and investigating new selection techniques to development approaches for the integration and creation of selection techniques in virtual environments. First, Chapter 6 describes the VR-DeMo process, the model-based user interface design process which we will be using to prototype virtual environments. In Chapter 7 we add semantic information to this process by augmenting NiMMiT, a notation to define interaction, and by creating an automatic speech grammar. Validation is performed through case studies. Contextual knowledge is another type of information which we will be introducing in Chapter 8 and 9. First, we will discuss how context can be introduced at the dialog level instead of at the task level. Thereafter we propose a context system to design context-aware virtual environments based on the ‘Event-Condition-Action’ paradigm. Validation of the approach is once again performed using case studies.

Finally, to conclude this dissertation we provide a conclusion and future research directions in Chapter 10.
Part I

Multimodal Selection Techniques
In this part, selection techniques and multimodal feedback strategies for virtual environments will be introduced and designed, after which we will evaluate our proposed solutions using formal evaluation (Dix et al., 1997).

Firstly, the current state of the art will be discussed with regard to selection metaphors and multimodal feedback for selection techniques. Secondly, in Chapter 3 a first experiment was performed in which several selection metaphors, augmented with multimodal feedback, were compared. From this experiment we saw that an image-plane technique, the 2D bubble cursor (Grossman and Balakrishnan, 2005), outperformed the traditional techniques. A downside of image-plane techniques is the fact that they are 2D techniques and therefore don’t perform well with overlapping objects. For this reason, further experiments will investigate two new factors which haven’t yet been investigated thoroughly: density and occlusion. As 2D techniques have problems with these factors and do not always allow selection in every situation without extra actions being involved, the 3D version of the 2D bubble cursor will be explored.

The first of these factors, the density of the environment, was varied in a second experiment discussed in Chapter 4. The second factor, occlusion, was realised by possibly occluding the goal target, of which the users knew the position, from their viewpoint. The 3D bubble cursor was compared with the traditional virtual hand and a newly introduced selection technique for volumetric displays, the 3D depth ray (Grossman and Balakrishnan, 2006). It was found that a higher density has a larger impact on the 3D bubble cursor than on the 3D depth ray and that occluded objects could be selected with only a minor 1 s overhead using transparency. In this experiment we noticed that users did not take full advantage of the selection metaphors and therefore we designed multimodal feedback for these selection techniques. This extra feedback could enhance the performance of the user, unfortunately in another experiment only a small non-significant performance gain was found when this extra feedback was present.
During these selection experiments we found that multimodal feedback provided a small objective improvement and was subjectively preferred. But an important question arose during the design of the force feedback, namely which force parameters should be chosen. In Chapter 3, before we performed our first experiment, we did a pilot test to identify the parameters of our force feedback. While in Chapter 4 we performed trial-and-error design until we were satisfied with the force feedback. In order to alleviate this problem and aid designers in designing force feedback we investigate which force parameters (force profile, duration and amplitude) are safe to use and won’t deteriorate the performance of the user. In order to be able to guarantee this, we deduced a guideline which allows to calculate the force parameters, using an integral. These force parameters then should not deteriorate the performance of the users during their movements. This guideline will be derived through a series of objective and subjective experiments in Chapter 5.
Chapter 2

An Overview of Selection and Multimodal Feedback

Contents

| 2.1  | Introduction | 9 |
| 2.2  | 3D Selection Techniques | 10 |
| 2.3  | Multimodal Feedback | 14 |
| 2.4  | Conclusion | 20 |

2.1 Introduction

Considerable potential of virtual environments has already been shown for providing intuitive interfaces. In order to enhance the experience in virtual environments, it is important to optimise the most basic interaction tasks. In this part of the dissertation we will focus on the selection task, as this is the basic interaction task we will mainly be investigating. Researchers in 3D virtual environments have often categorised selection as one of the four basic interactions (along with navigation, manipulation, and data input) (Bowman et al., 2004; Mine, 1995). Selection allows users to specify an object, which they wish to manipulate or interact with. Because it is such a frequent task, there has been a wide variety of research looking into several techniques for supporting it. In this chapter, we first provide a review of selection techniques for 3D environments. Besides exploring different selection techniques, we will also investigate the addition of multimodal feedback, such as audio and force feedback, to selection techniques. Therefore we also discuss the use of multimodal feedback, and its application to selection techniques.
2.2 3D Selection Techniques

Before we start discussing selection techniques, it is important to introduce the general concept of interaction in virtual environments (Poupyrev and Ichikawa, 1999). The interaction with the virtual environment application is a closed-loop system (see Figure 2.1). In order to perform actions and commands, the user input, captured by devices—such as the position of the hand or voice commands—is translated using interaction techniques. Thus, for a selection task, the user input, usually the user’s hand, is transformed through a selection technique, to be able to indicate objects.

![Figure 2.1: Interaction in virtual environments. (Poupyrev and Ichikawa, 1999)](image)

A selection task is comprised of three sub-tasks: indication of object, feedback, and confirmation of selection (Bowman et al., 2004). All these sub-tasks can be realised in different ways. Some possible choices are depicted in Figure 2.2.

Not all possibilities are presented in this figure and some areas allow other possibilities to be added. The figure shows that potentially a lot of different combinations are possible and the combinations will depend on the available hardware (input and output devices) and the application. For instance force feedback can only be provided through a force feedback device and indirect voice selection can only be performed through a microphone.

In this part, we will focus on the indication of an object and the possible feedback during the selection task. We will not focus on the confirmation of selection, which will be performed by pressing a button.

Besides task decomposition, it is also possible to classify tasks using metaphors. Poupyrey et al. (1998) proposed a taxonomy for the classification of different manipulation metaphors which is also commonly used to classify selection metaphors (Poupyrev and Ichikawa, 1999; Bowman et al., 2004). It is used for selection due to the fact that manipulation is tightly coupled to the selection task, in that the former follows the latter and that usually very similar metaphors are used for manipulation.
2.2 3D Selection Techniques

and selection. After having selected an object, the users will usually switch to manipulating the selected object. This is easiest if both techniques and devices for these two tasks are similar if not the same.

Figure 2.3 shows this taxonomy, in a first step they are classified as exocentric and egocentric techniques. Exocentric selection techniques (also known as god’s-eye viewpoint) are those performed from outside the virtual world (e.g. World-in-Miniature (Stoakley et al., 1995; Wingrave et al., 2006)), while egocentric selection techniques are those performed from inside the virtual world. We will limit ourselves to egocentric selection techniques since the interaction we investigate is performed from a first-person point-of-view in non-immersive setups. The egocentric selection techniques are divided in two subclasses (virtual hand and virtual pointer), each of which will be introduced separately in this chapter. Further on in this dissertation new techniques in these subclasses will be investigated and discussed.
2.2.1 Virtual Hand

The class of virtual hand metaphors represents techniques in which the user controls the X, Y, and Z coordinates of a 3D cursor (Hinckley et al., 1994; Mine, 1995; Poupyrev et al., 1996). The class of virtual hand metaphors is also commonly called the class of hand extension metaphors.

Mine (1995) states that in local interactions, a direct mapping from the user’s hand to a 3D “virtual cursor or drone” could be used to select an object (see Figure 2.4). For distant objects, the go-go technique explores the use of nonlinear mappings between the user’s hand and a 3D cursor, extending the range which the cursor can cover (Poupyrev et al., 1996). One drawback of the hand extension metaphor is that the selections are constrained by three dimensions (as the targets are 3D volumes), usually resulting in longer selection times (Poupyrev et al., 1998; Bowman et al., 1999; Grossman and Balakrishnan, 2004; De Boeck et al., 2006a; Grossman and Balakrishnan, 2006). As an alternative to a simple 3D point cursor, Zhai et al. (1994) developed the silk cursor, which is a semi-transparent 3D volume cursor. Using a volume increases the activation area of the cursor, and it was shown to increase performance in a 3D target tracking task. While a volume cursor could reduce target
acquisition times, its efficiency is reduced when interacting in dense target environments, as multiple targets may fall within the boundaries of the cursor’s volume.

![Image of virtual hand metaphor](image)

**Figure 2.4: The virtual hand metaphor. (Poupyrev et al., 1996)**

It is important to note that many virtual hand implementations (Poupyrev et al., 1998; Bowman et al., 1999; De Boeck et al., 2006a) represent the user’s hand through a selection volume. This makes it hard to compare different experiments as it is unclear how large the representation of the hand is. Therefore the exact influence of the selection volume is unclear, especially if the volume is not rotation invariant, such as the virtual hand used in Figure 2.4.

### 2.2.2 Virtual Pointer

On the other side of the egocentric metaphors is the virtual pointer metaphor, which is the most common alternative to the virtual hand metaphor. The user has to point to objects in order to select them. Different metaphors or techniques exist to translate this pointing action into a selected object.

One of the earliest implementations of selection for 3D environments was Liang and Green’s (1994) “laser gun” ray casting technique. With this technique, a ray is emitted from the user’s hand, which gives control over the origin and trajectory of the ray, much like using a physical laser pointer (see Figure 2.5a). One observed problem with this technique is the difficulty in selecting distant and small objects due to the required angular accuracy. To overcome this, they introduced a technique called “spotlight selection” or “flashlight”. Instead of emitting a ray, the user emits a conic selection area, originating from the user’s hand (see Figure 2.5b). With a conic selection area, instead of a ray, the selection volume increases further along the cone. Therefore this selection area allows to select distant and small objects more easily.
Since this original work, there have been a number of iterations on the ray casting metaphor in the 3D research community, such as aperture based selection (Forsberg et al., 1996) and 2D image plane selection (Pierce et al., 1997).

With traditional ray casting, only the first target to be intersected is selected. Because of this, a general problem is that it can be difficult to select a target that is behind a dense surrounding of targets. Allowing multiple targets to be intersected by the ray, or increasing the activation area to a conic selection area are possible solutions, however these create an ambiguity of the intended target.

In this part we will experiment with selection techniques (both virtual hand and virtual pointer metaphors), augmented with multimodal feedback, in order to gain new insights for their use and future designs. In the following section we will introduce multimodal feedback.

### 2.3 Multimodal Feedback

Before we discuss multimodal feedback, we need to define *multimodal*. An interface is multimodal when it makes use of multiple senses for input and/or output. Interaction in our daily lives is multimodal by nature and we are capable of seamlessly integrating information provided by different modalities. A *modality* can be a sense through which the human can receive output of the computer or a sensor or device through which the computer can receive input from a user. Usage of multimodal interaction in computer interfaces that aim to approach real life interaction...
2.3 Multimodal Feedback

is a research area that still has a lot of potential for development (Jaimes and Sebe, 2005). We won’t further elaborate on the definition of multimodal and its different technicalities; this definition is sufficient for the purpose of this dissertation.

The most important interface nowadays is the desktop WIMP (Window, Icon, Menu, Pointer) interface (van Dam, 1997), it uses a combination of keyboard and mouse for input and the screen for output, but it is usually not considered as a multimodal system. Since the system introduced by Bolt (1980), researchers are trying to investigate and understand combinations of multimodal input such as speech, pen or gesture input (Cohen et al., 1997; Oviatt, 1999; Tse et al., 2006) which can be considered as more genuine multimodal interfaces (Dix et al., 1997). Besides different combinations of input modalities, research also focuses on a combination of output modalities such as haptics, audio, olfactory and taste. Less research is being conducted into taste (Stoffregen and Bardy, 2001) and olfactory (Mori et al., 1999) as it is impossible to have input/output with the same bandwidth as in other modalities (Kaye, 2004).

Because modalities can be combined in a lot of different combinations, several theoretical frameworks exist which allow designers to reason about these combinations (Coutaz et al., 1995; Nigay and Coutaz, 1997; Martin, 1998; Sturm et al., 2002). These frameworks mainly provide mechanisms to describe how the modalities relate with regard to expression power and temporal constraints. Nigay and Coutaz (1997) define properties which characterise four types of relationships between modalities, the CARE properties: Complementarity, Assignment, Redundancy and Equivalence. Complementarity means that all modalities are necessary in order to be able to complete the task and each modality provides only a part of the information. The assignment relationship defines that a specific modality is assigned to the task and that there is no other modality to execute the task. Redundancy is when modalities have the same expressive power and are used in the same temporal window, meaning that the system or user receives multiple information conveying the same message. Finally, equivalence is the relationship in which modalities have the same expressive power but with no obligation in which one to use. Note that other frameworks, such as the TYCOON framework by Martin (1998), are able to express the same or similar relationships.

Besides theoretical frameworks which allow designers to express different ways to combine modalities, there also exist theoretical frameworks to reason about the perception of a user when different modalities are combined. When modalities are combined, it is important to also take into account how information is processed by humans. One such example is the multiple resource theory from Wickens (2008), describing how multiple resources of information, in our case modalities, can be handled by humans when those are presented to them.
A selection of an object typically lasts a few seconds and if we want to be able to improve the selection time, we need to augment it with modalities which are able to provide direct and fast feedback. For this reason, with regard to multimodal interfaces, this dissertation focuses on output modalities and more specifically on haptics and audio, which can both be useful in the feedback sub-task of the selection task (see Figure 2.2). As the feedback sub-task allows to give extra information about which object is currently selected during the selection, haptics and audio feedback might be interesting additions to visual feedback. These modalities, in comparison to the other ones, taste and olfactory, provide the user with direct instantaneous feedback, which is important for selection.

During the remainder of this dissertation both the haptic and audio feedback will be used for the same purpose, i.e. give extra information during the feedback sub-task. Because of this, they will be used as redundant modalities with regard to the CARE properties, as they will be equivalent in meaning and will be expressed during the same temporal window. In the remainder of this section we will give an overview of haptics and audio feedback.

### 2.3.1 Haptics

The word *haptic* is derived from the Greek word ‘haptethai’, meaning ‘to touch’ or ‘to grasp’, and pertains to the sense of touch. The sense of touch is performed by different systems of the human body, but research hasn’t yet reached the point where these different systems and their parameters are fully understood. Oakley et al. (2000) give an overview of the different systems and of definitions related to the term haptics, which are represented in Figure 2.6.

The psychophysics of touch, the study of perception through touch, plays a vital role in the design and construction of any haptic interface. Many researchers have done (Durlach et al., 1989; Tan et al., 1994) and are currently doing (Tan et al., 2007; Yang et al., 2008a,b) many perception experiments to investigate more thoroughly the different characteristics which influence our sense of touch. This is important for the creation of devices and user interfaces. Neglecting the psychophysical literature when designing a device is likely to result in a mismatch between the human perceptual system and the device capabilities. Such a design will be unable to produce realistic and correct stimuli.

Some interesting characteristics which have to be taken into account are (Burdea, 1996; Stanney, 2002, chap 5.):
2.3 Multimodal Feedback

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Haptic</td>
<td>Relating to the sense of touch.</td>
</tr>
<tr>
<td>Proprioceptive</td>
<td>Relating to sensory information about the state of the body (including</td>
</tr>
<tr>
<td></td>
<td>cutaneous, kinaesthetic, and vestibular sensations).</td>
</tr>
<tr>
<td>Vestibular</td>
<td>Pertaining to the perception of head orientation, acceleration, and</td>
</tr>
<tr>
<td></td>
<td>deceleration.</td>
</tr>
<tr>
<td>Kinaesthetic</td>
<td>Meaning the feeling of motion. Relating to sensations originating in</td>
</tr>
<tr>
<td></td>
<td>muscles, tendons and joints.</td>
</tr>
<tr>
<td>Cutaneous</td>
<td>Pertaining to the skin itself or the skin as a sense organ. Includes</td>
</tr>
<tr>
<td></td>
<td>sensation of pressure, temperature, and pain.</td>
</tr>
<tr>
<td>Tactile</td>
<td>Pertaining to the cutaneous sense but more specifically the sensation of</td>
</tr>
<tr>
<td></td>
<td>pressure rather than temperature or pain.</td>
</tr>
<tr>
<td>Force Feedback</td>
<td>Relating to the mechanical production of information sensed by the</td>
</tr>
<tr>
<td></td>
<td>human kinaesthetic system.</td>
</tr>
</tbody>
</table>

Figure 2.6: Commonly used definitions related to the term haptics. (Oakley et al., 2000)

- The joint resolution, calculated through the just noticeable differences, in the hand is approximately 2.5 degrees, in the wrist and elbow 2 degrees and in the shoulder 0.8 degrees.

- When grasping objects of 1 to 80mm, the fingertip positional resolution ranges between 0.5 and 2.5mm.

- The force control resolution or the force production accuracy, is 0.04N. The maximum controllable force that can be exerted at the fingertip is between 50 and 100N, depending on whether the shoulder muscles are in use. However, typical forces used in manipulation are between 5 and 15N.

These characteristics are important for the design of devices, but they are also of interest to designers creating force feedback. For example if we want users to perceive or apply different force strengths, it is important to know the resolution by which the human system can discriminate or create forces.

Laycock and Day (2003) discuss the recent developments and applications of haptic devices, several commercial force feedback devices exist. Unfortunately no commercial device up until now integrates tactile and force feedback into one device. Although this integration has been attempted in some experimental devices created from commercial devices (Declerck and Lenay, 2006; Magnenat-Thalmann et al., 2007).

Former research projects in our lab have been researching force feedback, these projects focused on the integration of this modality within a 3D virtual environ-
An Overview of Selection and Multimodal Feedback

Because of this acquired knowledge, it is a logical choice to keep this focus using the same force feedback devices. We will be using a PHANToM premium (Massie and Salisburg, 1994) (see Figure 2.7a), which is considered as a high-end device and is one of the oldest and most used devices with good physical parameters (low friction and inertia). Recently, Novint (2009) launched the Novint Falcon (see Figure 2.7b), a device designed for the gaming market. The Novint Falcon is a cheap device (180 dollar) compared to the PHANToM premium, which has prices starting from 24,250 dollar. We will also use the Novint Falcon in this dissertation, in Chapter 5 we will compare the Novint Falcon with the PHANToM for a multidirectional point-select task.

The use of haptics can improve the user interface and shows a lot of potential in many application domains such as teleoperation (Reintsema et al., 2004), virtual prototyping (Chen, 1999), data visualisation (Hanson and Zhang, 2005; Zhang and Hanson, 2007) and medical applications such as surgical simulators (Liu et al., 2003) or rehabilitation (De Boeck et al., 2008a).

Haptic research focusing on selection techniques in 3D virtual environments is rather limited. It can be used to provide the user with extra information during the selection. An overview of related work with regard to force feedback and selection techniques will be discussed in Chapter 3.

2.3.2 Audio

Besides visual and haptic feedback, audio feedback is another option to present information to the user. Usually audio feedback is neglected in 3D user interfaces (Bow-
2.3 Multimodal Feedback

man et al., 2004). Even though different goals for the usage of audio feedback have been investigated: auralization, sonification, ambient effects, sensory substitution, and annotation and help. We will discuss these goals in the following section. Many different hardware-setups and techniques exist to generate realistic 3D sounds: (stereo) headphones, external speakers, sound sampling and synthesis, and auralisation. Depending on the purpose of the audio feedback, different hardware-setups can or have to be used, note that not all goals, for example sensory substitution, need realistic 3D sounds.

An in-depth analysis of audio feedback discussing different setups and techniques and all the different properties characterising a sound (e.g. frequency, intensity, . . . ) has been described by Stanney (2002, chap 4.). In this dissertation we will be employing audio feedback without any special audio hardware. We will also only discuss the audio feedback in 3D user interfaces, as this is where our interests lie.

Usage of Audio Feedback

The following are the common usages of audio in a 3D user interface:

Auralization: The creation of 3D spatial sound adds an extra possible audio depth cue, which can increase the aural sense of the virtual environment and hence the immersion and presence (Slater and Wilbur, 1997) of a user. Another interesting usage, with regard to auralization, is the addition of wayfinding aids to a virtual world. Wayfinding is the cognitive component of navigation (Darken and Sibert, 1996).

Sonification: Information visualisation using audio cues, instead of visual cues. For example Hermann and Ritter (1999) designed a sonification model to reveal information about the clustering of vectorial data. Several sonification techniques exist: auditory icons, earcons, audification and parameter mapping (Dix et al., 1997), these techniques make use of non-speech sound.

Ambient effects: These effects provide a sense of realism in the virtual environment, for example cars passing by and nature sounds such as birds chirping.

Sensory substitution: The process of substituting a sound for another sensory modality, mostly haptics. Possible examples are the sound of a button click, bumping into an object.

Annotation and help: Speech synthesis can be used to instruct users how to use a certain application or task or give extra information about a certain item in the virtual world.
For a selection task, sonification and sensory substitution are the most obvious candidates. Ambient effects and annotation and help won’t be able to help the users improve their efficiency and accuracy because they don’t provide extra information useful to the task. Finally auralization could be of benefit. For example, the depth of the virtual hand could be mapped to a property of sound, for instance the frequency, but continuous sound feedback will probably overload the senses of the user and after a while become annoying instead of helpful.

Sensory substitution can be used when the users have hit or reached a target, to give the impression that the users have touched the target and can finalise their selection. This could lead to a higher sense of presence and a faster reaction time due to the users being alerted that they have reached a target. For the realisation of sensory substitution a sonification technique could be used. This is also the main use of sonification during selection, to provide extra information to the current state of the task.

2.4 Conclusion

An overview of different selection techniques has been given, guided by a classification. Several different techniques were discussed further. These techniques have been thoroughly evaluated and all of them have their specific advantages and disadvantages. We also explained the term multimodal and introduced two feedback modalities, haptics and audio. Both modalities have proven that they can play an important role in certain application domains and might be useful during the feedback sub-task of the selection task.

In this part of the dissertation we will introduce new selection techniques which try to innovate with regard to the indication of object sub-task. We will also investigate if an augmentation with multimodal feedback might give extra enhancements to the feedback sub-task of 3D selection techniques.
Chapter 3

Multimodal Feedback for Selection Techniques

Contents

3.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . 21
3.2 Selection Techniques . . . . . . . . . . . . . . . . . . . . . . 22
3.3 Force and Audio Feedback for Selection Metaphors . . . . . 26
3.4 Multimodal Feedback . . . . . . . . . . . . . . . . . . . . . . 27
3.5 Multimodal Feedback Experiment . . . . . . . . . . . . . . . 31
3.6 Conclusion . . . . . . . . . . . . . . . . . . . . . . . . . . . . 40

3.1 Introduction

In the previous chapter we provided an overview of several selection techniques and defined multimodal feedback. Only a few studies exist which investigate the addition of multimodal feedback to 3D selection techniques and perform a formal evaluation. In this chapter, we look at several ways to enhance selection metaphors with force and audio feedback. We first discuss the selection metaphors which we will investigate. We will choose them based on previous work performed at our lab. Afterwards, we elaborate on the addition of multimodal feedback to these selection metaphors and we discuss our experiment in which the influence of extra modalities is investigated. Finally, we draw some conclusions and indicate topics for our future research on selection techniques.
3.2 Selection Techniques

The taxonomy for egocentric selection metaphors exists of two branches, Virtual Hand and Virtual Pointer (see Figure 2.3). As both have different characteristics, it is important to include selection techniques from both. In order to pick, we will continue from the results found by De Boeck et al. (2006a). They evaluated several selection techniques using the dominant or the non-dominant hand. Virtual hand and aperture showed to be the most promising techniques: virtual hand was preferred by most users, while aperture was the fastest technique.

Further, we elaborate on an interesting new technique which is adapted from 2D Graphical User Interfaces (GUIs), namely the 2D bubble cursor (Grossman and Balakrishnan, 2005). This technique has not been tested in a 3D environment before and we would like to investigate whether this method might be an improvement of aperture.

These three techniques provide us with selection techniques belonging to the Virtual Pointer (aperture and 2D bubble cursor) and Virtual Hand (virtual hand) branches of the selection taxonomy. In the next sections we will introduce the chosen selection techniques more elaborately.

3.2.1 Virtual Hand

The virtual hand metaphor has been discussed earlier in Section 2.2.1. The user’s hand position and possibly orientation are virtually represented in the virtual environment, but this virtual representation does not have to mimic a real hand. In order to realise this, the user’s movements are tracked and mapped to the virtual representation. Figure 3.1a represents our implementation of the virtual hand, it is represented by a sphere. A sphere is used because it is rotation invariant. Therefore the orientation of the input device does not matter, and the user only has to concentrate on moving the cursor without taking into account its orientation. When the virtual representation intersects with an object, the object can be selected by pressing a button.

The main disadvantage of this technique is the limited workspace, which is restricted to our own reach or that of the device. Solutions to this problem have been explored, for example Go-Go (Poupyrev et al., 1996), but for this experiment we assume that all objects are reachable or that the user can navigate easily to their vicinity.
3.2 Selection Techniques

Figure 3.1: The Selection Metaphors that were used in this research.

3.2.2 Aperture

The aperture metaphor (Forsberg et al., 1996) can be seen as a flashlight with an invisible cone. Only the circle of light, the aperture, can be seen on the screen parallel to the projection plane (see Figure 3.1b), this is because the apex of the cone is in the user’s eye point. Figure 3.2 illustrates this, when moving in the image plane (X and/or Y direction), the cone and aperture move along. When moving in the depth (Z direction), the cone becomes smaller, making the aperture smaller as well. The object closest to the centerline of the cone becomes selected. For the user, this is the same as projecting the 3D view onto 2D and selecting the object that falls in the aperture, this technique has a strong resemblance to the area cursors used in 2D GUIs (Worden et al., 1997).

Figure 3.2: The aperture technique illustrated. (Forsberg et al., 1996)

When having multiple objects close together, the aperture will have to decrease in size in order to disambiguate the selection, especially if small objects are involved. In a worst case scenario it has to be made as small as the smallest object. This can be a huge disadvantage, as movement depthwise (along the Z axis) is necessary to decrease the aperture size and this extra overhead will cost the user extra time during selection.
3.2.3 The 2D Bubble Cursor

The 2D bubble cursor originates from research into 2D area cursors (Worden et al., 1997). A single point cursor has a single point hotspot, whereas the area cursor’s hotspot is the size of the area it encompasses. A 2D area cursor can have any shape: a square, a circle, an ellipse, etc. A lot of research on area cursors has been guided by Fitts’ law (Fitts, 1954; Soukoreff and MacKenzie, 2004; Cockburn and Brewster, 2005), which states that the movement time depends on the size of the target and the distance to the target. Thus, an area cursor, due to its area, increases the width of a certain target, which has an obvious benefit. Note that the aperture selection technique is in essence a 2D area cursor, which uses a conic volume to decide which objects need to be selected and therefore there is a clear relation with the 2D bubble cursor.

The design of the 2D bubble cursor addressed two shortcomings of area cursors (Grossman and Balakrishnan, 2005). Typically area cursors are squares. A square does not guarantee that the closest target is captured first by the area covered by the cursor because of the corners in the shape, therefore the 2D bubble cursor is represented by a circle which does not have this problem. The second shortcoming is exactly the same disadvantage as the aperture selection technique has. The size of the area can be too large capturing several targets at the same time, or the area can be too small such that no targets are captured and the cursor is in empty space. This problem is addressed by dynamically changing the size of the area based on the proximity of the targets. The algorithm to define the size at any time is described by Grossman and Balakrishnan (2005). In the experiments of Grossman and Balakrishnan (2005), the 2D bubble cursor appeared faster than a single point cursor in every situation, even when targets were very densely spaced.

Essentially, the selection algorithm of the 2D bubble cursor divides the space in a 2D Voronoi diagram (Aurenhammer, 1991), where every target represents one Voronoi region, and any point within that region is closest to the target of that region (see Figure 3.3). A Voronoi region corresponds to the width of the target which can be used to select it (the effective width). Hence, the Voronoi region the centre of the cursor is in, is the target currently being selected and the visualisation of the dynamic area is used as visual feedback such that users can more easily anticipate on which target will be the next selected target during their movements.

Thus, the 2D bubble cursor is represented by a circle which resizes dynamically such that its area selects only one target at a time. The cursor is also drawn semi-transparent (see Figure 3.1c). It is not straightforward to use the 2D bubble cursor in a 3D environment, compared to 2D GUIs, as the algorithm is based on a 2D Voronoi
3.2 Selection Techniques

Because of this, we use the projection of the virtual objects onto the image plane to calculate the 2D Voronoi diagram.

When comparing the aperture technique and the 2D bubble cursor, we notice that both are based on the same principles, though coming from different backgrounds. The aperture technique can be seen as an area cursor for 2D GUIs and the 2D bubble cursor can be seen as an image plane technique for virtual environments. A difference between both techniques, is how is decided, whether or not a certain object is selected. The aperture uses a conic volume starting at the user’s eye and the size of the conic volume is defined by the depth of the user’s hand, while the 2D bubble cursor uses a 2D Voronoi diagram for its selection. Another important difference is the visual feedback, the aperture has a static area, while the 2D bubble cursor has an area resizing dynamically which clearly indicates the currently selected target. The aperture technique will usually encompass several targets, which is less decisive with regard to visual feedback and might confuse users. Making the aperture smaller will resolve this problem but takes essential time as the user has to move depthwise.

Figure 3.3: A 2D Voronoi diagram.
3.3 Force and Audio Feedback for Selection Metaphors

In Chapter 2 we discussed several selection techniques and gave an overview of multimodal feedback. Multimodal output during object selection has been thoroughly studied in 2D (Akamatsu et al., 1995; Cockburn and Brewster, 2005). For 3D virtual environments, possible benefits for multimodal feedback have been found (Sallnäs and Zhai, 2003; Unger et al., 2002; Wang and MacKenzie, 2000) and its use within traditional selection techniques has been explored (Arsenault and Ware, 2000). Multimodal feedback of collision and contact has been used in many studies (Arsenault and Ware, 2000; Lindeman, 2003; Unger et al., 2002), but it is not clear how such techniques can be directly applied to object selection. However, it is clear when during the selection task they can be applied, namely during the feedback sub-task (see Figure 2.2). We now outline the existing literature related to the addition of force and audio feedback to selection.

3.3.1 Force Feedback

The use of haptics in an application can improve the feeling of immersion and degree of intuition of the interaction to the user (Stone, 2000; Van Erp et al., 2004). A large proportion of force feedback research concentrates on selecting items, such as icons and menu items, in Graphical User Interfaces (GUIs) (Ahlström, 2005; Ahlström et al., 2006; Lécuyer et al., 2004; Oakley et al., 2002; Smyth and Kirkpatrick, 2006). Oakley et al. (2000) investigated the effects of different kinds of force feedback, such as textures, friction, recess and gravity wells. They found that gravity wells, a “snap-to” effect which pulls the user to the centre of the target, seemed best in improving performance time and error reduction. However, follow-up research, which introduced distractor targets to the task, found that when the cursor needed to pass over the distractor targets, the gravity wells became problematic (Ahlström, 2005; Ahlström et al., 2006; Hwang et al., 2003; Keuning, 2003).

In 3D virtual environments, research into the use of haptics for selection is limited. Arsenault and Ware (2000) found that the addition of force feedback in a tapping task improved the user’s performance, probably due to the fact that the cursor bounced off a target actually speeding up its progress back to the other target. Wall et al. (2002) investigated 3D gravity wells for ray casting selection. The force feedback did improve the accuracy, but not the performance time. Magnusson and Rassmus-Gröhn (2005) asked users to test different audio and force feedback in a memory game. Gravity wells were among the conditions with the best subjective results. Finally, Kim and Kwon (2007) showed that force feedback can help the user become
more aware of depth in a 3D environment, using pop-through grid planes placed along the Z-plane.

### 3.3.2 Audio Feedback

The addition of sound and its advantages have also been explored in 2D GUIs (Brewster, 1998a). Many different widgets (menus, buttons, scroll bars, progress bars, etc.) have been augmented with non-speech sound, called earcons, resulting in lower error rates and decreased task completion times (Brewster, 1998a,b). An earcon is a brief, structured sound pattern often used to represent a specific item or event (Blattner et al., 1989).

Audio feedback has also been used for 2D selection, to indicate that the cursor has reached a target (Akamatsu et al., 1995; Cockburn and Brewster, 2005). Akamatsu et al. (1995) found that such audio feedback did not improve overall selection time, but it did reduce the time spent over the target, as the sound made users react faster. Cockburn and Brewster (2005) found that the addition of audio feedback reduced mean selection times by 4.2%, but that combining sound with other feedback modalities, which improve selection time independently, does not assure further improvements. As we are not aware of any literature which tries to augment a selection technique for 3D virtual environments with audio feedback, we experimented with audio feedback during selection.

### 3.4 Multimodal Feedback

In this section we will discuss our multimodal feedback design based on the related work of the previous section. First, we give an overview of the different force feedback designs for the virtual hand and the image plane techniques. Afterwards, we propose our audio feedback which will be similar for all selection techniques.

#### 3.4.1 Force Feedback for the Virtual Hand

Based on the literature on force feedback during selection, gravity wells seem an interesting type of force feedback to further explore. A problem with gravity wells is that they have several different parameters with a lot of possible values. Note that most researchers don’t mention the details of these parameters so therefore it is impossible to compare different experiments and their results. In order to be able to
choose these parameters for our experiment we first define them and perform a small experiment with five users to decide which parameters are most useful.

The following parameters define a gravity well, they are also represented in Figure 3.4:

**Size:** The size of a gravity well is very important. Taking into account Fitts’ law (see Section 3.2.3), the size influences the width, thus the larger the gravity wells, the faster a user can select. On the other hand, larger fields overlap each other, especially when targets are close to each other. This causes oscillation problems and makes it possible to get stuck on the wrong target. Usually, gravity wells have a spherical volume to encompass the target by a rotation invariant volume which leaves more options to solve distractor problems or makes it easier to detect distractors.

**Overlap:** The overlap parameter is not really a value-based parameter but an action which needs to be defined when overlap between gravity wells occurs. The easiest action to take when gravity wells overlap, is doing nothing at all. A second option is to activate the gravity well that is reached first. This can be further enhanced with prediction algorithms. A possible simple algorithm is the following: when one of the overlapping gravity wells is active and if the user moves 25% away from the closest ever distance to the centre of this active gravity well, a switch from active gravity well occurs, to the overlapping gravity well. Another enhancement could be the recalculation of the force strengths and sizes of the gravity wells, using the overlapping percentage. Half of the percentage could be applied such that the overlap disappears.

**Attraction Force:** The attraction force can be either constant or decreasing/increasing towards the centre of the volume representing the gravity well. The magnitude of the maximum force can also differ. Keuning (2003) concluded that users have different preferences regarding these parameters, she found that a higher force makes the users more precise. Therefore it seems best to start with a high force and degrade towards the middle of the gravity well.

**Velocity:** When incorporating the velocity at which the user is moving, we can disable the gravity wells at a certain threshold because the first movements during the selection process are at a high velocity (Meyer et al., 1988; McGuffin and Balakrishnan, 2002). Oakley et al. (2001) already noticed that performance time improved when adjusting the force according to the velocity of the user.

**Shape:** A gravity well can have any shape, but a rotation invariant shape allows the user to approach the target from any direction in a similar manner. Usually
3.4 Multimodal Feedback

spheres are used (Oakley et al., 2000; Wall et al., 2002; Hwang et al., 2003), sometimes rectangles (Ahlström et al., 2006).

**Figure 3.4:** An abstract 2D representation of two gravity wells with labels pointing to the different possible parameters of a gravity well.

A small experiment served as a pilot test to be able to witness trends and identify the gravity well parameters for the virtual hand. We used a strong force (1N) with force degradation to the centre. For the size we used different percentages, namely 120%, 150% and 200% of the bounding sphere of the target. When gravity wells overlap, we test both the proposed prediction algorithm discussed earlier and the prediction algorithm enhanced with the adjustment of sizes and forces. Finally, the gravity wells are disabled when the user moves the cursor with a speed of 2 cm/s, which seems like a valid threshold deduced during informal testing as it was frequently reached during the initial movement. For the test setup, we used a PHANTom in combination with a 17-in monitor. The participants were presented 8 test-scenes (see Figure 3.5a) in which 2-3 small and/or large boxes are present. Figure 3.5b shows one of the 8 test-scenes. The boxes sometimes served as distractors and were close enough to each other to overlap, such that the largest gravity well sizes were not given too much advantage. Every participant performed the experiment with every type of gravity well and the conditions were randomly assigned.

Due to the limited amount of trials and the limited amount of users we did not perform any statistical analysis but performed an evaluation based on the preference of users and the mean/median value of the completion time of every trial. No condition clearly was the best condition. Therefore we decided to take the parameters from the best
condition in which the user performs the fastest. In order to find the fastest condition, we used the median of the time it took to perform the selection. We used the median because it is influenced less by the standard deviation. We believe that our decision is further strengthened due to the fact that the technique was ranked second when using the mean.

The fastest condition had the following parameters: 1N force degrading towards the centre, a 120% enlargement with the simple prediction algorithm and disabling of the gravity wells when the velocity reached 2 cm/s.

### 3.4.2 Force Feedback for an Area Cursor

In Section 3.3.1 we discussed the addition of force feedback for the virtual hand selection metaphor in GUIs and VEs, but in Section 3.2 we also introduced two other metaphors. To our knowledge, no research has been done on combining haptics with image plane techniques, and neither for their equivalents in GUIs, the area cursors. Based on our literature study, we propose two possible force feedback strategies which try to emulate sticky icons as used by Worden et al. (1997). The first force feedback strategy is a constant directional force in the opposite direction of the movement when moving over a target. Another strategy is a viscous drag field with a negative constant, which is also active when moving over a target. These strategies are similar, with the difference that a viscous drag force does not have a constant force output, as it multiplies a constant value with the velocity and the velocity varies continuously during movement.
Informal testing showed that the viscous drag field was preferred. For this reason, the viscous drag field will be used in our experiment.

### 3.4.3 Audio Feedback

As discussed in Section 3.3.2, most research used earcons, but different types of earcons were and can be used. Based on those findings, we decided to use a combination of four earcons during the different phases/sub-tasks of the selection task: when reaching an object, an earcon sounds (F’ on piano for 1 s) to inform the user that a target is highlighted. If the user switches to another target, another earcon is played (C on celesta for 0.8 s). This alerts the user for possible movement mistakes and reinforces the fact that another target right now is highlighted. When the user moves off a target, another earcon sounds (C on soprano sax for 1 s). And finally, when selecting a target (i.e. pressing the button), an earcon is played (B on glocken for 1 s) to confirm the selection of an object. An overview of the audio feedback is illustrated in Figure 3.6.

![Figure 3.6: An overview of the audio feedback during the selection process.](image)

### 3.5 Multimodal Feedback Experiment

We have presented several methods to add force and audio feedback to selection metaphors. This section describes our experiment in which the selection metaphors from Section 3.2 are enhanced with the above proposed multimodal feedback. The main goal of this experiment is to investigate the implications of the integration of multimodal feedback when selecting targets in a virtual environment. Another goal is to see how the 2D bubble cursor will perform compared to both the other selection
techniques and in particular the aperture technique, as the aperture technique and 2D bubble cursor can both be considered as image plane techniques.

The following hypotheses will be investigated in this experiment:

- Adding audio or force feedback makes the selection of an object faster.
- The combination of audio and force feedback does not further enhance the trial completion time.
- The 2D bubble cursor will perform faster than the aperture and virtual hand technique.

3.5.1 Apparatus

The display we used was a monoscopic projection screen. For input a PHANToM premium 1.0 was used with a stylus for 6 DOF input and 3 DOF force feedback. The stylus is equipped with a single button. The force update rate was 1000Hz and the tracker update rate was 120Hz. The input device controlled each of the selection techniques with an absolute 1 to 0.5 mapping. During the experiment participants were seated 75 cm from the display and they wore normal headphones at all times. Figure 3.7 illustrates the experiment apparatus.

3.5.2 Participants

Eleven participants, nine male and two female unpaid volunteers, ranging in age from 21 to 30 with an average of 24, served as participants in this experiment. Participants had a varying range of experience from no experience at all to expert. All participants were right handed and used their right hand to control the input device.

3.5.3 Procedure

A 3D static target acquisition task was used for the experiment. The scene consisted of four possible start targets and 9 possible goal targets. The start targets were rendered as spheres, the goal targets as grey cubes. The grey cubes had a size of 0.4 or 0.8 cm. Figure 3.8 illustrates the visual environment used during the experiment.

To complete the task, participants were gently pulled towards one of the start targets, while the other starting targets were invisible. Once this target was reached the respective starting target disappeared and a sound played to indicate the trial had begun.
3.5 Multimodal Feedback Experiment

Figure 3.7: The experiment apparatus.

In order to indicate which goal target to select during this trial, among the possible goal targets, it was highlighted in green. After selecting any target, either the correct one or a wrong one, the trial ends.

For all techniques the currently captured target was rendered with a yellow solid opaque border.

3.5.4 Independent Variables

In the experiment three selection metaphors are used: virtual hand, aperture and the 2D bubble cursor. Aperture is actually included in the experiment as a baseline technique for comparison with the 2D bubble cursor, as they are based on the same principle. Therefore only the 2D bubble cursor will be enhanced with modalities as we expect the results of the addition to be similar for the aperture technique.

For the virtual hand metaphor we have two options regarding force feedback: touching an object (normal haptics) and the gravity wells as described in Section 3.4.1. The audio feedback will use the approach as proposed in Section 3.4.3. The 2D bubble cursor will use a viscous drag field in combination with the same audio feedback as the virtual hand.
Haptics and sound are tested separately and in combination, i.e. each technique will be used with audio only, with haptics only and with haptics and audio together.

With regard to the virtual scenes we only take object size as an independent variable. Two different sizes were used: 0.4 and 0.8 cm.

### 3.5.5 Design

A within-participant design was used. The independent variables were a combination of the selection technique, audio on/off, force on/off. All these combinations form eleven conditions: six conditions for the virtual hand, because two force feedback strategies are tested; four for the 2D bubble cursor and the baseline comparison technique aperture. In order to counterbalance these eleven conditions a Latin Square design was used of size eleven. An overview of all the conditions can be found in Table 3.1.

Each participant performed the experiment in one session lasting about 20 minutes. The sessions were broken up by the eleven conditions with 20 trials appearing for each condition. A trial consisted of a start position (one of the four spheres) and a box position (i.e. the target to select), these were manually defined before the experiment
3.5 Multimodal Feedback Experiment

<table>
<thead>
<tr>
<th>Selection Technique</th>
<th>Force Feedback</th>
<th>Audio Feedback</th>
<th>Acronym</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Hand</td>
<td>None</td>
<td>None</td>
<td>VH-N-N</td>
</tr>
<tr>
<td>Virtual Hand</td>
<td>None</td>
<td>Earcons</td>
<td>VH-N-S</td>
</tr>
<tr>
<td>Virtual Hand</td>
<td>Classic</td>
<td>None</td>
<td>VH-C-N</td>
</tr>
<tr>
<td>Virtual Hand</td>
<td>Classic</td>
<td>Earcons</td>
<td>VH-C-S</td>
</tr>
<tr>
<td>Virtual Hand</td>
<td>Gravity Wells</td>
<td>None</td>
<td>VH-GW-N</td>
</tr>
<tr>
<td>Virtual Hand</td>
<td>Gravity Wells</td>
<td>Earcons</td>
<td>VH-GW-S</td>
</tr>
<tr>
<td>Aperture</td>
<td>None</td>
<td>None</td>
<td>AP-N-N</td>
</tr>
<tr>
<td>2D Bubble Cursor</td>
<td>None</td>
<td>None</td>
<td>BU-N-N</td>
</tr>
<tr>
<td>2D Bubble Cursor</td>
<td>None</td>
<td>Earcons</td>
<td>BU-N-S</td>
</tr>
<tr>
<td>2D Bubble Cursor</td>
<td>Viscous Drag</td>
<td>None</td>
<td>BU-H-N</td>
</tr>
<tr>
<td>2D Bubble Cursor</td>
<td>Viscous Drag</td>
<td>Earcons</td>
<td>BU-H-S</td>
</tr>
</tbody>
</table>

Table 3.1: Overview of all the different selection conditions being used during the experiment.

and randomly given to the participant. The first 5 trials for every condition were discarded in order to give the user time to get familiar with the condition.

Data was logged after every selection including the total trial completion time. The total trial completion time will be divided in two parts: the time it took to reach the object (Reach Time) and the time it takes to finalise the selection after having reached the desired object (Reaction Time). We also logged whether the correct object was selected and which particular trial had been performed. After the experiment, the users were asked to fill in a questionnaire, querying their subjective impressions.

3.5.6 Results

Trial Completion Time

When analysing the trial completion time, we discarded trials in which errors occurred. A one-way analysis of variance was conducted on the selection techniques, disregarding the combination of modalities. A significant main effect was found for trial completion time ($F_{2,1762} = 194, p < 0.0001$), reach time ($F_{2,1762} = 215, p < 0.0001$) and reaction time ($F_{2,1762} = 9.22, p < 0.001$). Post hoc comparisons for trial completion time and reach time showed that the 2D bubble cursor (2 and 1.2 s) is significantly faster than aperture (2.5 and 1.5 s) and virtual hand (3.1 and 2.3 s), and aperture is significantly faster than virtual hand (all $p < 0.0001$). For the reaction time we did not expect to see a significant difference, but post hoc comparisons showed us
that aperture (0.96 s) was significantly slower than both the bubble cursor (0.83 s) and virtual hand (0.84 s) (p < 0.001). We expect this to be due to the fact that users had to move into the depth, regardlessly of having the correct object highlighted, in order to assure themselves that their selection is correct as the aperture often overlapped other objects.

In Figure 3.9, all the conditions are depicted, with their average trial completion, reach and reaction time. They are ordered using the trial completion time. We now discuss the combination of modalities separately for the 2D bubble cursor and virtual hand, starting with the virtual hand.

A one-way analysis of variance of the virtual hand selection technique with all different modality conditions showed a significant main effect for trial completion time ($F_{5,972} = 10.72$, $p < 0.0001$), reach time ($F_{5,972} = 25.67$, $p < 0.0001$), and reaction time ($F_{5,972} = 50.3$, $p < 0.0001$). Post hoc comparisons showed that there is no significant difference between virtual hand augmented with gravity wells and audio, virtual hand without force feedback and with audio, and virtual hand with classic force feedback and audio feedback. The remaining conditions of the virtual hand have no audio feedback and are significant slower for trial completion times but not always for reach or reaction times. An overview of the post hoc comparisons can be found in Table 3.2.

Virtual hand with gravity wells and audio feedback is significantly faster in trial completion time and reaction time, compared to virtual hand with gravity wells and with-
Table 3.2: Post hoc comparison table of different conditions. A dash means that that particular modality was not taken into account. A legend of all the conditions can be found in Table 3.1.

Out audio. This indicates that the gravity wells only speed up the reach time, not the reaction time, or at least not significantly. The same findings could be seen when we compare virtual hand with gravity wells and without audio feedback with virtual hand without force feedback and audio feedback, where the trial completion time and reach time are significantly faster. Note that the combination of gravity wells and audio feedback did not make the reach time significantly faster, although audio feedback did improve the trial completion, reach and reaction time significantly when compared to virtual hand without sound.

These results are similar to those of Cockburn and Brewster (2005), audio or force feedback improves the trial completion time while the combination provides no extra benefit. We expect that the participants probably used the audio feedback to immediately stop moving, removing overshoot and therefore lowering reach time as well. This contradicts with the results of Akamatsu et al. (1995), which concluded that only the reaction time would be reduced, while we noticed that both the reach time and
reaction time are reduced. A possible explanation could be the extra dimension (2D to 3D), as this enlarges the possible performance deterioration caused by overshoot, an overshoot in 3D is less easy to correct than in 2D. Comparing both types of force feedback, gravity wells are significantly faster ($p < 0.03$).

Taking into consideration the size of objects, we noticed that all previous conclusions still hold except one: when selecting large objects, force feedback had no influence on the trial completion time. For small objects we notice that force feedback did behave differently for trial completion time ($F_{2,345} = 3.16, p < 0.05$). Post hoc comparisons showed that gravity wells are significantly faster than classic force feedback, $p < 0.015$.

For the 2D bubble cursor, similarly, as with the virtual hand, a one-way analysis of variance was performed with all different modality combinations. This analysis of variance showed a main effect for trial completion time ($F_{3,627} = 3.87, p < 0.01$) and reach time ($F_{3,627} = 2.99, p < 0.05$). Reaction time showed no main effect ($F_{3,627} = 1.16, p = 0.33$), which is an unexpected result as audio feedback normally would improve the reaction time (Akamatsu et al., 1995). Post hoc comparisons showed, for both the trial completion and reaction time, that the condition with no modalities at all was the fastest one. However, it was not significantly faster than the 2D bubble cursor with force feedback and audio feedback. But the addition of audio or force feedback on its own made the bubble cursor significantly slower. This is a rather unexpected result. We expected that both modalities on their own would improve the trial completion time but that the combination of modalities could be slower because of an overload of information. Since the results show it is faster than the separate modalities, this is not the case. A more in depth study would be necessary to see if these results are applicable to image plane techniques or area cursors in general.

We are not aware of other related work that investigated the 2D bubble cursor or any other area cursor augmented with modalities, therefore we have no other experiments to compare against. Some clarification might be found in Fitts’ law. The 2D bubble cursor already has a very large effective width, the width of the target which can be used to select it. Note, the effective width is usually larger than the visual width of the target. In case of the bubble cursor, it can select targets from a distance, without having to physically reach them, due to its resizing algorithm: this distance is the effective width. The addition of force feedback, which slows down the movement over the target, increases the effective width even more. This addition could slow down the users because they already see that they have reached the object and need no slowing down. The same holds for audio feedback: the 2D bubble cursor probably gives enough visual feedback to the users. Furthermore the switching earcon had a length of 0.8 s which was probably too long, taking into account the average duration, $\pm 2$ s,
of a selection task. The combination of both force and audio feedback could cancel the effect of too much feedback as the extra audio feedback could make the users less aware of them slowing down. Important to note is that the audio feedback was also not disabled depending on the velocity, this might also be worthwhile exploring.

### Error Rate

For this task, errors were defined as trials in which users selected the wrong target before selecting the goal target. The overall error rate for the experiment was 2.75%. The virtual hand had an error rate of 1.21%. The aperture metaphor had an error rate of 5.45% and the 2D bubble cursor 4.39%. These differences were statistically significant: $\chi^2(2) = 19.9, p < 0.0001$. With regard to the different modalities we found no significant difference. The significant difference between techniques in error rate can be attributed to two possible reasons: the way errors are counted for the virtual hand and two particular box positions for trials.

On the one hand the chance of making an error with the virtual hand is lower due to the fact that only errors are counted when selecting the wrong target and not when selecting empty space (i.e. no target at all). The bubble cursor on the other hand can’t select empty space, as it always has one target selected. The aperture technique also didn’t cover much empty space in our experiment.

On the other hand, analysis of the 20 different trials showed that two specific box positions, the middle left and right box, caused a much higher amount of errors, 7.5% and 4.5% versus ± 2%. Furthermore, nearly all the errors for those box positions were made by the 2D bubble cursor or aperture technique. Both techniques are 2D techniques and these particular box positions are difficult to select in 2D due to high density and overlap with other boxes. This is an inherent disadvantage of 2D selection techniques in 3D environments. For this problem no straightforward solution exists which does not incorporate extra actions. A possible solution would be the tumble and splatter techniques (Ramos et al., 2006). These techniques allow users to first spread out layers of occluding objects in a 2D environment so that they are all visible. But adding extra actions to selection techniques should be avoided as much as possible.

### Subjective Feedback

Users filled out a short questionnaire after the experiment (see Figure C.1 in Appendix C). Subjectively, for the selection techniques in general, almost all users (8 out of 11) preferred the 2D bubble cursor, two preferred aperture and one preferred
the virtual hand. For the virtual hand, with regard to the modalities, answers pointed out that users always preferred the addition of audio feedback. The addition of force feedback is preferred 8 out of 11 times: four users preferred gravity wells, the other four preferred normal force feedback.

The 2D bubble cursor results were similar to the one’s for virtual hand. All users, except one, preferred the audio feedback with 7 of them preferring the combination with force feedback. This means that 3 users preferred only audio feedback. The fact that the users preferred feedback, even though the no feedback condition is the fastest, could have several causes. First of all the addition of modalities is probably experienced as a novelty and participants therefore judged based on this. But looking at the selection technique in particular, another possible explanation would be that when the 2D bubble cursor is used, the target being highlighted changes very fast. This occurs due to the dynamic resizing and the 2D nature of the technique, and therefore always one object is highlighted. The extra multimodal feedback could make the users more comfortable and aware of what is happening exactly, but this higher state of awareness also seemed to make them slower.

### 3.6 Conclusion

In this chapter we presented several selection metaphors for virtual environments and a number of ways to combine them with force and audio feedback. Based on these possible strategies, several combinations of audio and force feedback were implemented and investigated in a user experiment.

The experiment showed that the addition of force and/or audio feedback for the virtual hand metaphor had similar effects as in GUIs. The addition of one modality separately results in a decrease of the trial completion time, but the addition of both force and audio feedback does not decrease the trial completion time.

The results for the 2D bubble cursor were in contradiction to those of the virtual hand or traditional GUI research. The addition of force or audio feedback increased the total selection time. On the contrary, the addition of both modalities sped up the selection, but it was still slower than having only visual feedback.

Overall, the 2D bubble cursor had the best performance during our experiment. However, as little research into image plane techniques or area cursors and multimodality has been conducted, other visual, force or audio strategies might be necessary. A downside of the 2D bubble cursor is it being in 2D, while targets have 3D positions and potentially could overlap or be close to each other with a high density.
In our experiment we noticed that the accuracy of the 2D bubble cursor might break down in such a situation. A high error rate was found for a certain trial in which high density (on the image plane) and occlusion was present. Up until now, not many selection technique experiments investigated the influence of density and occlusion on the selection. Therefore in the next chapter we will investigate how density spacing and occlusion influence selection techniques. We will investigate a 3D variant of the bubble cursor and we will further try to enhance our multimodal feedback, taking into account the results from this experiment.
4.1 Introduction

The experiment of the previous chapter investigated, among the addition of multimodal feedback, a new technique, the 2D bubble cursor, which had recently been introduced in 2D GUIs. It was found to be the best technique of the tested techniques due to the dynamic resizing of the cursor. The major drawback of this technique is the limited degrees of freedom, it is an 2D image plane technique and therefore only allows to select objects visible from the current viewpoint of the user. In very dense
Influence of Density and Occlusion on Selection

scenes where objects overlap each other, other problems might arise. In the previous experiment indications of these problems were found when studying the error rate (see Section 3.5.6). Despite a number of experiments, conducted by ourselves and by other researchers, in which numerous designs and evaluations of selection techniques have been investigated, there are important factors which remain to be fully understood. In this chapter, we focus on two of the factors that require further investigation.

The first factor concerns the density of targets in the environment. The level of density can greatly affect the performance of a selection technique, particularly in a 3D environment. For example, with traditional ray casting techniques, only the first intersected target can be selected, making it difficult to select targets behind dense areas (Liang and Green, 1994). Allowing multiple targets to be intersected, either along the length of the ray (Hinckley et al., 1994), or in some cases, within its conic selection area (Liang and Green, 1994), results in an ambiguity of the user’s intended target. Recently, researchers have attempted to address this problem of ambiguity with new selection techniques (Olwal and Feiner, 2003; Grossman and Balakrishnan, 2006; Wyss et al., 2006), but the effect of target density on such techniques remains to be explored.

The second factor which we investigate in this chapter is the visibility of the goal target. While, in many cases, users’ intended targets are visible, there may be cases where the intended targets are occluded from their view by another object in the scene. Generally, users are required to either rotate the scene or to switch to a different viewing mode, to allow them to select their desired target. These extra steps may be time consuming, requiring the addition of modes and buttons to the interface. It is, therefore of interest to develop techniques which allow users to seamlessly select targets which are occluded from their view, and to understand how the visibility of the target affects selection performance.

In this chapter, we investigate various candidate selection techniques, which may allow for efficient selection of targets in both sparse and dense 3D environments. Furthermore, we augment these techniques with various forms of multimodal feedback to also allow for the selection of targets which are not visible from the user’s viewpoint. In an initial experiment, we augment a depth ray and 3D bubble cursor with only visual feedback, and compare these techniques to a standard 3D point cursor, also augmented with visual feedback to allow for the selection of occluded targets. The results show that both the depth ray and the 3D bubble cursor outperform the 3D point cursor in all conditions, with the depth ray performing best overall. Furthermore, the additional visual feedback which we introduced to all techniques allowed users to adequately select targets which were not visible from their
initial viewpoint. However, our observations indicated that the visual feedback was not sufficient to completely overcome the difficulties associated with selecting targets in dense and occluded environments. This visual feedback stimulates only the (already heavily loaded) human visual system, leaving the powerful human senses of touch and hearing unused. Other forms of feedback have been suggested as a way to improve interaction and reduce the load on any one sense (Bolt, 1980; Oviatt, 2002). Therefore in a second experiment, we augment the selection techniques with audio and force feedback, in addition to slight improvements of the visual feedback used in the first experiment. Results show that while participants did prefer the presence of the multimodal feedback, the quantitative advantages were only slight, and nonsignificant. These results indicate that providing adequate visual feedback is most critical for selections in dense and occluded 3D environments.

4.2 Related Work

In Section 2.2 3D selection techniques are discussed in general, but in this chapter we will design selection techniques which can cope with dense target environments or occluded targets. Therefore in this section, we discuss selection techniques which have been studied in dense target environments. A discussion of techniques for overcoming target occlusions follows. Lastly, we discuss the use of multimodal feedback based on the results from the previous chapter.

4.2.1 Selection Techniques for Dense Environments

For ray casting techniques, generally only the first intersected object will be selected, even though the ray can intersect multiple objects simultaneously. Under this implementation, it may be difficult or even impossible to select objects that are further away, depending on the density of targets. While research has shown that users may choose to navigate to a new position such that the desired object is closer or not occluded (Ware and Lowther, 1997), there may be cases where the target will be occluded regardless of the viewpoint. Furthermore, it may be desirable for the user to have the ability to perform the selection in-place, without having to switch to and perform navigation.

To address this, Liang and Green (1994) developed a metric for the spotlight selection to determine which object would be selected when multiple targets were captured, based on the distance between the target to the apex and central axis of the cone. An interesting extension to spotlight selection is Shadow Cone Selection (Steed and
Parker, 2004), which selects targets by sweeping out an area with a cone selection cursor. Metrics have also been proposed (Steed, 2006), but it is unclear how well they will work in dense target environments, as they have not been formally evaluated.

An alternative to defining these predictive metrics is to provide an explicit mechanism for users to specify their intended target among all those intersected (Hinckley et al., 1994). Grossman et al. (2004) used forwards and backwards hand movements to cycle through intersected objects, however, limited visual feedback was provided to the user. Olwal and Feiner (2003) describe the flexible pointer, which allows users to bend the cursor to avoid intersecting other targets, however it requires two 6 DOF devices to control the cursor. Another technique requiring two input devices is iSith (Wyss et al., 2006), where two rays are intersected to define a target location.

In a recent study, Grossman and Balakrishnan (2006) designed and evaluated several new ray casting techniques which allowed for multiple target disambiguation. Of their tested techniques, they found the depth ray to be most successful. The depth ray augments the ray cursor with a depth marker, which can be moved forwards and backwards along the length of the ray, with similar movements of the hand. The intersected target closest to this depth marker is the one which can be selected. Although the study used a dense environment, the environment was constant throughout the experiment, so they were not able to study the effect of the environment density on the technique.

In a study that did vary target densities, Looser et al. (2007) compared three different selection techniques, including ray casting, for tabletop augmented reality. They found no significant difference between the two densities, although, they argued that this might have been because the difference between the densities which they tested was small.

There has also been work in selecting targets in dense two-dimensional environments. Most notably, the 2D bubble cursor (Grossman and Balakrishnan, 2005) is an area cursor that dynamically changes its size to always capture only the closest target (see Section 3.2.3). The technique was studied in environments of varied density, and found to provide efficient selections in sparse and dense environments. In the previous chapter, we also experimented with this technique in a 3D environment as a 2D image plane technique. We did not study for density but found that the error rate increases in high density where objects overlap. We believe that such a technique could be an interesting alternative to a static 3D volume cursor, as only one target will be captured at all times such that no ambiguity can arise during the selection. We will explore such a technique in this chapter.
4.2 Related Work

4.2.2 Overcoming Target Occlusion

Some of the above techniques for dense environments explore the scenario of multiple objects occluding the ray cursor’s approach to a goal target. However, none explore the issue of a target being completely occluded from the user’s viewpoint. During our literature review we found limited research on this topic. In 3D desktop applications, users generally rotate the scene so that their target of interest becomes visible. This has also been shown to be an effective approach in 3D virtual environments (Flasar and Sochor, 2007). Another approach is to switch to a viewing mode, such as wireframe rendering, so that all targets become visible. Some more specialised viewing modes, such as fill drawing with varied opacity, have also been studied in augmented reality environments (Livingston et al., 2003). Other techniques to reduce occlusions, such as interactively distorting the space (Carpendale et al., 1997; Elmqvist, 2005; McGuffin et al., 2003), or morphing the viewing projection (Elmqvist and Tsigas, 2006), have also been explored. Elmqvist and Tsigas (2008) provide a thorough overview of other possible techniques. However, such techniques are generally independent from the selection mechanisms.

Some relevant techniques in the 2D realm are the tumble and splatter techniques (Ramos et al., 2006). These techniques allow users to first spread out layers of objects which are occluding each other in a 2D environment so that they are all visible. The user can then select and perform interactions with any of these objects.

4.2.3 Adding Multimodal feedback

In the previous chapter we addressed related work with regard to multimodal feedback in Section 3.3 and investigated the addition of multimodal feedback to selection techniques. We augmented a virtual hand and a 2D bubble cursor, used as an image plane selection technique (Forsberg et al., 1996), with haptic feedback. Results have shown that the force feedback resulted in significant speed gains for the virtual hand technique but not for the 2D bubble cursor, possibly due to its dominating visual feedback. We also found that earcons improved reaction time, but not for the 2D bubble cursor. However, the use of multimodal feedback within selection techniques that address target densities and occlusions has not yet been investigated.

4.2.4 Summary

There is large body of research on selection techniques for 3D environments. However, less research has focused on supporting selection in dense target environments,
and even less on the selection of objects which are fully occluded. While some promising techniques do exist, to date there has not been an exploration of how these techniques are affected by the environment density and target visibility. With regard to other modalities of feedback, audio earcons have shown to improve reaction times when visual feedback is inadequate. Force feedback seems to be a viable technique, but evidence suggests that it may encounter difficulties in dense target environments. In the remainder of this chapter we will first introduce design guidelines and strategies before we design and evaluate several selection techniques.

4.3 Design Guidelines and Strategies

Before introducing our modified techniques and their evaluation, we discuss some high level design guidelines which we kept in mind as evaluation criteria for these techniques. Afterwards, we elaborate on our proposed design strategies for satisfying these guidelines. It is important to note that previous research has proposed several guidelines with regard to particular aspects of the design of virtual environments (Kaur, 1998; Gabbard et al., 1999; Bach and Scapin, 2003). We won’t consider guidelines such as “error correction” for the application-level, we will propose design guidelines which are specific to selection interaction techniques.

4.3.1 Design Guidelines

Our literature review of the previous work on selection techniques in 3D environments and the experiment of the previous chapter revealed that the environment density and visibility of the goal target are factors which are not well understood. As such, it is our goal to design and evaluate techniques which can adequately account for these two factors. In addition to some standard design guidelines for 3D selection techniques, this gives us the following six design guidelines:

- Allow for fast selections
- Allow for accurate selections
- Be easy to understand and use
- Produce low levels of fatigue
- Satisfy the above for sparse and dense target environments
4.3 Design Guidelines and Strategies

- Support selections for both visible and occluded targets

While previous work will guide us in satisfying the first four design guidelines, we propose two design strategies for satisfying the last two. We now discuss these two design strategies in detail.

4.3.2 Increased and Unambiguous Activation Areas

For a selection technique to be appropriate for use in an actual interface, it should support efficient selections in both sparse and dense environments. However, supporting efficient selections for both hand extension and ray casting techniques is a difficult task, due to the constraints imposed by the task. For hand extension techniques, the added third dimension imposes both a physical and visual constraint which can slow down selection times (Bowman et al., 1999; Grossman and Balakrishnan, 2004). For ray casting techniques, small angular changes in the input device can result in large movements of the ray (Liang and Green, 1994). Our strategy for reducing the effects of these constraints will be to increase the activation areas of the selection techniques. Established methods for this include: using a selection volume for hand extension techniques and emitting a cone for ray casting techniques (Liang and Green, 1994; Zhai et al., 1994).

Unfortunately, increasing the cursor's activation area means multiple targets can be captured simultaneously, introducing an undesirable ambiguity. This can be especially problematic in dense target environments. We, thus, seek 3D selection techniques which provide a mechanism for disambiguating between potential targets of interest. One strategy would be to use a dynamic activation area (Grossman and Balakrishnan, 2005). Another alternative is to provide a mechanism for disambiguating between multiple captured targets (Hinckley et al., 1994). Such techniques have a better chance of being efficient in both sparse and dense target environments.

4.3.3 Integrated Visual Enhancements for Occluded Targets

It is often the case in 3D environments that users need to select objects which are obscured from their viewpoint. This is generally the case when the target of interest lies behind another object in the scene and is thus occluded. If the environment is densely populated with targets, the chance that such an occlusion exists increases. It is generally impossible for the user to select an invisible target, as there is no visual feedback as to the location of the cursor relative to the intended target.
As discussed in Section 4.2.2, existing techniques for overcoming such occlusion generally require explicit and separate steps on behalf of the user. This means that the actual selection cannot be performed in a single, fluid, interaction. This is a drawback which motivates us to find other strategies.

When a user wishes to select an occluded object, the occlusion generally introduces a problem of access, and not discovery (Elmqvist and Tsigas, 2006). This is because it can usually be assumed that the user has a good conceptual model of the environment, knowing the general area of the intended target. As such, explicitly altering the viewing state of the entire scene may be excessive. We apply a design strategy of integrating visual enhancements into the selection technique itself. This allows occluded targets to be made visible, without the requirement of additional modes or auxiliary devices. The idea is to apply these enhancements only to targets in the vicinity of the selection cursor, almost as if the selection cursor were also controlling a magic lens (Bier et al., 1993). This strategy can be thought of as improving the visual feedback during the selection task. Later, in Section 4.6 of this chapter, we will investigate the use of other feedback modalities.

4.4 Selection Techniques

In this section, we discuss the selection techniques which will be used in our first experiment, developed with the above mentioned design guidelines and strategies in mind. We applied these design strategies to both hand extension and ray casting metaphors, resulting in a 3D bubble cursor and an augmented depth ray.

4.4.1 3D Bubble Cursor

We first apply our design guidelines, an increased and unambiguous activation area and integrated visual enhancements, to the hand extension metaphor. If we simply increase the activation area of such a selection technique we end up with a volume cursor, where the user must capture the intended target inside the cursor volume to select it. However, we also require the activation area to be unambiguous, and with a volume cursor, multiple targets can fall within the cursor’s boundaries.

To alleviate this problem, we implemented a 3D version of the bubble cursor, which dynamically resizes such that only the closest target falls within its boundaries. We
4.4 Selection Techniques

render the 3D bubble cursor as a grey\(^1\) semi-transparent sphere similar to the rendering of the silk cursor (Zhai et al., 1994) (see Figure 4.1a). When necessary, we render a second semi-transparent sphere around the captured target, such that it always appears to be fully contained by the cursor (see Figure 4.1b). For added visual feedback, we highlight captured targets yellow. As with the 2D implementation, we also render a crosshair inside the 3D bubble cursor to mark its centre. The user controls the location of this crosshair by positioning the input device in 3D space. In Section 3.2.3 we discussed how the 2D bubble cursor work. The 3D variant uses an adaptation of the same algorithms for 3D.

![Figure 4.1: (a) The 3D bubble cursor is rendered as a semi-transparent sphere which dynamically resizes such that it only captures the closest target, here highlighted yellow. (b) When necessary, a second sphere is rendered around the captured target so that it always appears to be completely contained by the 3D bubble cursor.](image)

Our hope is that this cursor will allow for efficient selections of targets in both sparse and dense environments. However, as per our design guidelines, we also wish for the technique to support the selection of targets which are occluded from the user’s viewpoint. To overcome such occlusions, we give the 3D bubble cursor magic lens capabilities, such that targets in its vicinity become semi-transparent. To do so, we calculate the distance between the 3D bubble cursor and each target, measured on the 2D image viewing plane. Any target within 4 cm is rendered as semi-transparent, so that targets which may be initially occluded become visible (see Figure 4.2a). This allows users to home in on an occluded goal target as they approach it, assuming they know its general location (see Figure 4.2b).

It is important to note that this localised transparency function is only appropriate when users know the general region of their intended target. This is often the case when users are familiar with the scene that they are interacting with, and is an as-

\(^1\)Note that the colours, used for our implementation, are chosen such that they will suit the environment in an optimal way. Therefore these colours might have to be changed depending on the application.
sumption we make in our research. If the user needed to search for the target, global methods such as rotating the scene or switching to a different viewing mode would be more appropriate.

Figure 4.2: (a) Targets in close proximity to the 3D bubble cursor become semi-transparent. (b) As the cursor approaches an occluded goal target (red cube) it becomes visible and can be selected.

4.4.2 Depth Ray

The 3D bubble cursor results from an application of our design strategies to the hand extension metaphor. Here, we apply these guidelines to the ray casting metaphor.

Without augmentation, ray cursors can already be thought of as having increased activation areas, since they can select any target along the length of the ray. A conic ray cursor further increases this activation area. Thus, multiple targets can simultaneously be captured, and we are again required to provide a disambiguation mechanism.

As previously discussed, the depth ray has been shown to provide an effective mechanism for disambiguating between multiple intersected targets (Grossman and Balakrishnan, 2006). The user controls a depth marker, which exists always along the length of the ray. Moving the hand forwards or backwards will make the depth marker move in the same manner. The object intersected by the ray cursor, which is closest to the depth marker, can be selected. We render the ray as a thin red cylinder, although a conic selection area, with an apex of 1°, originating from the user's hand, is used for the selection test. As with the 3D bubble cursor, the captured target is highlighted yellow and remaining targets intersected by the ray are highlighted green. Figure 4.3 shows a screenshot of our implementation.
4.4 Selection Techniques

![Figure 4.3:](a) The depth ray selects the intersected target which is closest to the depth marker. (b) The depth marker position can be controlled by moving the hand forwards or backwards.

To allow for the selection of occluded targets, we augment the depth ray with a similar transparency function used by the 3D bubble cursor, using the distance between the targets and the ray, measured in the 3D environment.

### 4.4.3 Effective Widths

The effective width of a target can be defined as the size in motor space of a target’s activation boundaries. It has been shown that the effective width of a target plays a larger role than its visual boundaries on selection times (Blanch et al., 2004; Grossman and Balakrishnan, 2005; Elmqvist and Fekete, 2008). While the goal of this work is not to obtain a sound theoretical model of how each technique will perform, understanding the effective widths for each technique will help us form hypotheses on the relative performances of the two techniques.

The dynamic activation area of the 3D bubble cursor divides the 3D environment into regions, such that there is exactly one target inside each region, with that target being closest to any point inside that region. These are also known as Voronoi regions, see Section 3.2.3. The 3D bubble cursor increases the effective width of a target to its surrounding 3D Voronoi region. In other words, to select a target, the centre of the 3D bubble cursor only needs to be positioned inside the target’s associated Voronoi region (see Figure 4.4a).

For the depth ray, two dimensions of the effective width depend on the angular accuracy required to intersect the target. This is determined by the target size, distance, and conic apex of the ray. The third dimension of effective width depends on which surrounding targets are intersected by the ray. This establishes a Voronoi region, which when projected onto the length of the ray, defines the segment of the ray where the depth marker can select the target. It is the length of this segment which can be thought of as the effective width. In some situations, this will result in a similar effec-
tive width to when using the 3D bubble cursor (see Figure 4.4b). However, if the ray is positioned to avoid certain surrounding targets, the effective width can be larger (see Figure 4.4c). This could potentially make the depth ray a faster technique than the 3D bubble cursor.

![Figure 4.4](image)

**Figure 4.4:** (a) The 3D bubble cursor divides the environment into 3D Voronoi regions. The effective width of each target is defined by its corresponding Voronoi region. (b) When using the depth ray, the Voronoi region is based only on the intersected targets. (c) Changing the position of the ray can change a target’s effective width.

### 4.5 Experiment 1: The Effects of Density and Occlusion

Applying design strategies to both hand extension and ray casting metaphors, resulted in the 3D bubble cursor and depth ray techniques. That are augmented to allow for the selection of occluded targets. We believe that these techniques will allow for efficient selections in both sparse and dense target environments, and for both visible and occluded targets. Previous work tells us that the depth ray can be efficient in dense environments (Grossman and Balakrishnan, 2006), but it is unknown how exactly the density will affect performance. A better understanding exists for the effect of density on the 2D form of the bubble cursor (Grossman and Balakrishnan, 2005), but no such understanding exists for its 3D counterpart which we have introduced in this chapter.

In this section, we present a formal experiment to evaluate these two techniques, where we manipulate both the environment density and goal target visibility in a 3D virtual environment. One goal of the experiment is, thus, to gain an understanding of how these factors affect selections. Another goal is to compare the relative

---

54 Influence of Density and Occlusion on Selection
performance of these two techniques, and also provide a comparison to a baseline technique.

One of the factors in our study is occlusion, because of this factor we will have to generate environments in which targets will be occluded. Therefore in some of the experimental environments which we will use, it would be impossible to select a target with a naive implementation of the ray cursor. For that reason, we use a 3D point cursor for a baseline comparison technique. The point cursor is rendered as a 3D crosshair, the same as the crosshair drawn in the centre of the 3D bubble cursor, described in Section 4.4.1. The centre of the 3D crosshair must be positioned inside the target to select it. This is an important difference with the classical hand extension technique, the virtual hand. Instead of using the volume of the crosshair, we only use its centre, this allows us to perform a better theoretical and fundamental comparison. Therefore we denominate our baseline comparison technique “3D point cursor” instead of “virtual hand”. To allow for the selection of occluded targets, we augment the 3D point cursor with the same transparency functionality used for the other techniques. A true baseline technique would not have this built-in functionality, and would require users to either rotate the scene or switch viewing modes to select an occluded target. These operations would have to be performed using extra hardware, therefore including the transparency function minimises hardware requirements, and provides for the fairest possible comparison.

The following hypotheses will be investigated in this experiment:

- An increased environment density will slow down selection times for the depth ray and 3D bubble cursor. However, the density should have little effect on a standard 3D point cursor. We hypothesise this because an increased density will decrease effective target widths for the depth ray and 3D bubble cursor and the increased density won’t have an effect on the motor space size of the target’s activation boundaries.

- It will take longer to select targets which are occluded. However, our hope is that our techniques in combination with transparency will reduce the overhead cost when the targets are not initially visible.

- Comparing the three techniques, we hypothesise that the depth ray and 3D bubble cursor will outperform the 3D point cursor. The relative performance of the 3D bubble cursor and depth ray is hard to predict, and will be an interesting result to analyse.
4.5.1 Apparatus

The display we used was a 21-in Hitachi CM813ETPlus monitor combined with StereoGraphics CrystalEyes 3D LCD shutter glasses for stereo viewing. Left and right eye images were provided using quad buffering at a refresh rate of 60Hz per eye which was coordinated through the shutter glasses by an infrared transmitter positioned on top of the monitor.

For input a Polhemus Fastrak 6 DOF magnetic tracker was used, encapsulated in a handheld device. The device was equipped with a single button. The tracker was updated at 120Hz with a precision of less than one millimetre. The input device controlled each of the three cursors with an absolute 1 to 1 mapping. Participants stood during the course of the experiment, and the display was raised to be roughly shoulder level. Figure 4.5 illustrates the experiment apparatus.

4.5.2 Participants

Eleven male and one female unpaid volunteers, ranging in age from 20 to 28 with an average of 24, served as participants in this experiment. Participants were screened using a stereopsis test in which they had to order objects according to their depth. All participants were right handed and used their right hand to control the input device.

![The experiment apparatus.](image)
4.5 Experiment 1: The Effects of Density and Occlusion

4.5.3 Procedure

A 3D static target acquisition task was used for the experiment. The scene consisted of a start target, goal target, and 45 distractor targets. In this experiment, the start target was rendered as a white sphere, the goal target as a red cube, and the distractor targets as blue spheres. To complete the task, participants first selected the start target, centred at the front of the display space. Once selected, this target would disappear, and the participant would then have to select the goal target. The position of the start target was static, and the goal target was positioned in a random location, such that the distance between the start and goal targets was 20 cm in 3D space. The positions of the distractor targets were random, with the constraint that they did not intersect. The radius for the distractor targets were randomly assigned values between 0.75 and 1.5 cm. Figure 4.6 illustrates the experiment environment.

![Figure 4.6: The experiment environment.](image)

Each trial started with a fade-in of the scene. Initially, only the goal target would be visible. After 500 ms the distractor targets would fade in over a duration of 2 s. This would give users an understanding of the general location of the goal target for the occluded conditions. Once all targets appeared, the user could select the start target to begin the trial. Users were instructed to select the goal target as fast as possible while minimising their errors. Users had to successfully select the goal target before the trial could end.
For all techniques the captured target was rendered yellow with a solid opaque border. Additionally, other targets intersected by the depth ray were highlighted green.

4.5.4 Independent Variables

The main factors we study in this experiment are the selection technique, environment density, target visibility, and target size.

For environment density, we were mostly concerned with the immediate surroundings of the goal target, as this would affect the effective width for the 3D bubble cursor and depth ray. To control this variable, we carefully position six distractor targets around the goal target. Two distractor targets were placed along the direction of movement, defined as the vector between the start and goal targets, one before and one after the goal target. Perpendicular to this vector, targets were then placed above, below, and to either side of the goal target, forming a cube shaped Voronoi region. We controlled the size of the resulting Voronoi region by changing the distance between these six distractor targets and the goal target, measured from their closest edge. We call this distance variable density spacing, $DS$ (see Figure 4.7).

![Figure 4.7: Six distractor targets are carefully positioned around the goal target, creating a cube-shaped Voronoi region. The distance between these targets and goal target is the density spacing (DS).](image)

For goal target visibility, we tested fully visible and fully occluded conditions. In both conditions, we ensured that none of the six surrounding targets were occluding the
goal target. To ensure this, in some cases, we rotated the entire Voronoi region. For the visible condition, we ensured that no other distractor targets occluded the goal target. For the occluded condition, we ensured that two distractor targets partially occluded the goal target, such that together the goal target was completely occluded.

4.5.5 Design

A repeated measures within-participant design was used. The independent variables were: cursor type, $CT$ (3D point cursor, 3D bubble cursor, and depth ray); density spacing, $DS$ (1 cm, 2.5 cm, 5 cm); visibility condition, $VC$ (visible, occluded); and target size, $SIZE$ (0.75 cm, 1.5 cm). A fully crossed design resulted in 36 combinations of $CT$, $DS$, $VC$, and $SIZE$.

Each participant performed the experiment in one session lasting about 70 minutes. The sessions were broken up by the three cursor types, with three blocks appearing for each of the techniques. Within each block, the 12 combinations of $DS$, $VC$ and $SIZE$ were repeated three times in a random order, for a total of 36 trials. Prior to the experiment, 36 scenes were randomly generated for these trials, and were the same across all participants. The cursor ordering was fully counterbalanced across the 12 participants, with two participants randomly assigned to each of the six unique orderings. Before each cursor type, participants were given several warm-up trials to familiarise themselves with the selection technique.

4.5.6 Results

Trial Completion Time

In our analysis of trial completion time, we discarded trials in which errors occurred, and removed outliers that were more than three standard deviations from the group mean (1.7% of the data).

Repeated measures analysis of variance showed main effects for $CT$ ($F_{2,22} = 383$), $SIZE$ ($F_{1,11} = 277$), $VC$ ($F_{1,11} = 330$), and $DS$ ($F_{2,22} = 68.7$), all at the $p < 0.0001$ level. Average trial completion times were 4.59 s for the 3D point cursor, 3.10 s for the 3D bubble cursor, and 2.85 s for the depth ray. Post hoc comparisons showed that both the depth ray and 3D bubble cursor were significantly faster than the 3D point cursor ($p < 0.0001$), and that the depth ray was significantly faster than the 3D bubble cursor ($p < 0.005$).

We also found that $CT$ had significant interaction effects with $SIZE$ ($F_{2,22} = 12.8$, $p < 0.0001$), $VC$ ($F_{2,22} = 118$, $p < 0.0001$), and $DS$ ($F_{4,44} = 3.97$, $p < 0.005$), showing
that each of these independent variables affected the three techniques differently. We now provide a discussion on each of these observed interaction effects.

The interaction between $CT$ and $DS$ is illustrated in Figure 4.8. The general trend, as expected, is that the selection times are reduced with increased density spacing. There are, however, two interesting results which we highlight here.

Firstly, the density had a significant effect on movement times for the 3D point cursor ($p < 0.0001$). This is somewhat surprising since in theory it should only be the target size which constrains a selection with the 3D point cursor. A possible explanation is inadequate visual feedback. In the dense conditions, it may have been more difficult to discern when the goal target had been highlighted.

Secondly, it is interesting to note that for the depth ray, increasing the density spacing from 2.5 to 5 cm did not significantly reduce movement times. In fact, the movement times actually increased, although the effect was not significant. In contrast, the 3D bubble cursor completion times decreased significantly for each increase in density spacing ($p < 0.05$). While the data for the depth ray may at first seem counterintuitive, we must recall that the effective width of the goal target when using the depth ray is not completely determined by the density spacing value. It also depends on the angle of approach that the ray takes on (see Figure 4.4), so it may be the case that users were not taking on optimal approach angles for the condition $DS = 5$ cm. This could have been for many reasons, one of which being that the randomly generated scenes were different for each density value.

Figure 4.9 illustrates the interaction between $CT$ and $VC$. As can be seen, the occluded condition increases trial completion times for all three cursors ($p < 0.0001$),
4.5 Experiment 1: The Effects of Density and Occlusion

Figure 4.9: Technique completion times by visibility condition.

with an average increase in completion times of 1.01 s. From this figure, we can also see that the depth ray was the fastest technique overall because of its superiority in the visible condition. Post hoc multiple means comparison shows that in this condition, the depth ray is significantly faster than the 3D bubble cursor (p < 0.0001), whereas in the occluded condition, there is no difference between the two techniques. It may be the case that the overhead cost introduced by the occluded condition outweighs the differences between the two techniques, so their differences are not apparent.

Finally, the interaction effect between $CT$ and $SIZE$ is illustrated in Figure 4.10. As can be seen, the size had its most dramatic impact on the 3D point cursor, which is the only technique constrained by the dimensions of the actual target. However, the size also had a significant effect on the 3D bubble cursor (p < 0.0001), with completion times of 3.4 s for $SIZE = 0.75$ cm and 2.8 s for $SIZE = 1.5$ cm. This is somewhat surprising, since selections with this technique are constrained by the proximity of the surrounding targets, and not the actual target size. This indicates that users were being drawn towards the visible borders of the targets, not taking full advantage of the target’s effective width. This result is consistent with observations of the 2D bubble cursor, which showed a similar result, although not as strong. The effect may be increased in our 3D task because the visual feedback is not as clear, as multiple objects could be layered over the cursor.

In Grossman’s original bubble cursor study (Grossman and Balakrishnan, 2005), it was shown that the size of the target will have more effect on selection times when the surrounding targets are closer. To determine if this was the cause for the effect we were observing, we looked at the interaction between $SIZE$ and $DS$ for the 3D
bubble cursor condition. Indeed, the interaction effect was significant ($F_{2,22} = 17$, $p < 0.0001$), illustrated in Figure 4.11. It can be seen that the effect of size increased when surrounding targets were closer to the goal target. Post hoc comparisons show that size had a significant effect for $DS = 1$ cm and $DS = 2.5$ cm ($p < 0.0001$), but not for $DS = 5$ cm.

Figure 4.10: Technique completion times by target size.

Figure 4.11: 3D Bubble cursor times by size and density spacing.

As for the depth ray, target size had no significant effect, indicating the visual feedback of when targets were captured may have been better than for the 3D bubble cursor, reducing unnecessary movements towards the boundaries of the goal target.
4.5 Experiment 1: The Effects of Density and Occlusion

Learning

A learning effect was observed in our experiment, with the block number significantly affecting trial completion times ($F_{2,22} = 23.8, p < 0.0001$). Performance improved over time, with average completion times of 3.79, 3.43, and 3.33 s for blocks 1, 2, and 3, respectively. Post hoc comparisons show that Block 1 was significantly slower than Blocks 2 and 3 ($p < 0.0001$), but Blocks 2 and 3 were not significantly different. There was also a significant interaction between the block number and $CT$ ($F_{4,44} = 2.54, p < 0.05$). The effects, illustrated in Figure 4.12, indicate that there is little overhead cost involved with learning the techniques. It can be seen that the most learning occurs with the 3D point cursor. It is also interesting to note that the 3D bubble cursor and depth ray are almost equivalent in the first block, and differ significantly only in the last block ($p < 0.05$). This may indicate that users learned to take optimal approaches to the goal target using the depth ray, to maximise the effective width.

![Figure 4.12: Learning effects for each of the techniques.](image)

Input Device Footprint

The input device footprint is defined as the length of the total path which the input device travelled to complete each trial. The cursor type had a significant effect on the input device footprint ($F_{2,22} = 210, p < 0.0001$). The footprints were 31.2 cm for the 3D point cursor, 26.9 cm for the 3D bubble cursor, and 22.4 cm for the depth ray, all significantly different ($p < 0.0001$). The depth ray likely had the lowest value because, for this technique, users are not required to move their hand towards a specific 3D location.
Error Rates

For this task, errors were defined as trials in which users selected the wrong target before selecting the goal target. The overall error rate for the experiment was 2%. For the 3D point cursor no errors were reported. This shows that users were able to use the visual feedback provided to perceive when the goal target was highlighted. The 3D bubble cursor had an error rate of 2.2% and the depth ray 3.4%, for which there was no statistical difference.

Subjective Feedback

In a questionnaire administered after the user experiment (see Figure C.2 in Appendix C) we asked for the user’s preferences. Overall, participants found the 3D bubble cursor and depth ray easy to learn and use, preferring these techniques over the 3D point cursor. On a scale of 1 to 5, with 1 as strongly disagree and 5 as strongly agree, they respectively scored on average 4.25 (SD = 0.75), 3.83 (SD = 1.03) and 3.5 (SD = 1.09). With regard to the user’s opinion on the speed of the selections techniques, the depth ray was rated best on the same scale with a score of 4.25 (SD = 0.75), followed by the 3D bubble cursor 4 (SD = 0.74) and the 3D point cursor 1.59 (SD = 0.51).

Participants who preferred the depth ray noted that less movement was required, and that they could aim from any angle. All 12 participants responded that they liked the transparency function. Some commented that it would be better than letting the targets disappear and that it might also be useful to find other objects in the environment. Two participants complained that the transparency made their depth perception worse.

4.6 Multimodal Feedback

The results from our first experiment indicate that while the provided visual feedback works adequately, it does not completely alleviate the difficulties imposed by selections within dense and occluded target environments. For example, users did not take full advantage of the effective target width when using the 3D bubble cursor, and relied on the visual boundaries of the target. Furthermore, with all techniques, there was about a 1 s overhead cost when selecting targets which were occluded from the user’s viewpoint.
4.6 Multimodal Feedback

These shortcomings motivate us to investigate other forms of feedback, to complement the visual feedback which we provided, so that users will have a better overall understanding during the selection task. In a follow-up experiment which we will describe in the next section, we investigate if other forms of feedback, more specifically audio and force, could further aid users during selections within dense and occluded target environments.

We will only investigate extending the ray casting and 3D bubble cursor selection techniques with these forms of feedback. The 3D point cursor is not considered here, because it was found to be less efficient, and it is unlikely that multimodal feedback would provide it with enough gain in performance to surpass the other techniques. Furthermore, multimodal feedback techniques for the traditional point cursor have already been investigated in both 2D (Cockburn and Brewster, 2005) and 3D environments, see Chapter 3. In contrast, the use of multimodal feedback for 3D selection techniques with dynamic activation areas, such as the 3D bubble cursor and depth ray, to our knowledge, has never been thoroughly investigated.

Because the selection techniques we are using have dynamic activation areas, the main purpose of the additional feedback is to make the user more aware of the targets’ activation boundaries. This means that the feedback needs to be activated when the user switches between targets. To prevent the user from being overwhelmed with feedback during the initial ballistic movement of selection (Meyer et al., 1988; Rosenbaum, 1991), we disable the feedback during fast movements. We use both a velocity and acceleration threshold (3.0 cm/s and 1.0 cm/s², respectively). When either threshold is crossed, we disable the additional feedback (Oakley et al., 2001). We now discuss the additional feedback which we will apply to our selection techniques.

4.6.1 Force Feedback

It has been shown that attraction forces, commonly used when haptic feedback is used during selection, can be detrimental, especially in dense target environments (Ahlström, 2005; Ahlström et al., 2006; Hwang et al., 2003; Keuning, 2003). Since our motivation is to find techniques which work well in such environments, we will diverge from these magnetism techniques. Instead, we propose a short and subtle haptic response during selection. It is best described as a quick bump in the movement with a low force to make sure that the user’s movement is not disturbed. For the 3D bubble cursor, the direction of the force feedback is parallel to a Voronoi edge with a duration of 25 ms and maximum strength of 1N with a sine wave profile (see Figure 4.13). For the depth ray, a bump, with the same properties, is felt in the direc-
Influence of Density and Occlusion on Selection

tion perpendicular to the ray. This bump is felt any time the movement of the depth marker causes a new target to be captured.

Figure 4.13: An overview of the multimodal feedback for the 3D bubble cursor.

4.6.2 Audio Feedback

To realise audio feedback, we make use of the earcons technique (Akamatsu et al., 1995; Cockburn and Brewster, 2005). Whenever a new target is hit, an earcon is played for a short duration. We do not play earcons with a long duration (Cockburn and Brewster, 2005) or continuously while a target is being hit (Akamatsu et al., 1995), because with our techniques of interest a target is always selected. This would result in continuous, and thus meaningless, audio feedback. Any time a new target is captured, an audio earcon (C on celesta for 0.2 s) was sounded (see Figure 4.13). Compared to the earcons used in the experiment of the previous chapter (see Section 3.4.3), it is important to mention that the one’s used here are much shorter. The earcon, to indicate a target switch in the previous chapter, was 0.8 s long, which is too long with respect to the total trial completion time of 2 to 3 s. Therefore in this experiment, we use a short, 0.2 s, but clearly audible earcon.

4.6.3 Visual Feedback

In addition to the force and audio feedback, we also changed some properties of the visual feedback provided during the experiment. These modifications were made based on observations and user feedback from Experiment 1, discussed earlier in this chapter in Section 4.5.3. In the first experiment, we used a discrete transparency
4.6 Multimodal Feedback

function, which was either enabled or disabled for each target based on whether that target was within a threshold distance of the cursor. We modified this to use a continuous function instead. With this function the percentage of transparency depends on the relative distance, which prevents otherwise sudden and distracting changes in target transparencies.

We also changed the visual appearance of the 3D bubble cursor. We decided not to render the bubble at all, and only render the crosshair representing its centre. We hoped this would make it easier to see which target was highlighted and thus captured.

4.6.4 The Haptic Lock Ray

We have augmented our earlier designed techniques, the 3D bubble cursor and depth ray, with new forms of multimodal feedback. During our initial tests with the depth ray we found that too many parameters were changing simultaneously (such as the trajectory of the ray, location of the depth marker, proximity to targets, etc) for the multimodal feedback to be intuitive for users. The lock ray (Grossman and Balakrishnan, 2006), a technique closely related to the depth ray, splits the selection task into explicit selection and disambiguation phases, and would thus be potentially more appropriate with the presence of multimodal feedback. With this technique, the depth marker only becomes active once the user presses the input device button. At this point the users can control the depth marker, as they would with the depth ray, but the position of the ray itself is locked. The user confirms the selection by releasing the button. We slightly modified the original lock ray design by initially placing the depth marker at the midpoint of all intersected targets to minimise its expected travel distance. Furthermore, we render the depth marker when it is inactive, so the users will know exactly where it is before they begin to use it. When inactive the marker is grayed out. The disambiguation phase, when the user is controlling the location of the depth marker and the position of the ray is locked, is ideal for multimodal feedback design. We used the same audio feedback as with the other selection techniques, except that an additional earcon (F on piano for 0.2 s) signaled that the disambiguation phase had started. During the disambiguation phase, the haptic input device was constrained to move only along the vector of the ray. In addition the users felt similar bumps as with the depth ray, every time a new target was selected as a result of moving the depth marker. We slightly increased the force to 1.5N since there would likely be less targets to pass over using this technique. This greater bump force created a subtle “pop-through” effect (Smyth and Kirkpatrick, 2006). If the depth marker reached the first or last intersected target, a force wall would pre-
vent any further movements of the depth marker, such that potential error corrections during the disambiguation phase would be faster.

### 4.7 Experiment 2: The Effects of Multimodal Feedback

In this section, we present a second experiment to determine if the additional multimodal feedback, can improve users’ ability to select targets in dense and occluded 3D environments. We will compare and evaluate the 3D bubble cursor, depth ray, and lock ray, both with and without the new multimodal feedback. We omit a comparison with our previous baseline technique, the 3D point cursor, because it proved to be much slower than both the 3D bubble cursor and depth ray. The procedure of the experiment was identical to Experiment 1, discussed in this chapter.

The following hypotheses will be investigated in this experiment:

- The addition of multimodal feedback will enhance the perception of the effective width of a target and therefore speed up the selection times.
- Users judge the multimodal feedback as helpful.
- It is harder to predict how the lock ray will perform compared to the 3D bubble cursor and the depth ray, a previous experiment indicated that the depth ray outperformed the lock ray (Grossman and Balakrishnan, 2006), which could be changed through the addition of multimodal feedback.

#### 4.7.1 Apparatus

The apparatus differed from Experiment 1 mainly because a haptic input device was required. The display was a 2.4m × 1.8m polarisation projection screen with passive stereo using two DLP projectors. For input a PHANTom premium 1.0 was used with a stylus for 6 DOF input and 3 DOF force feedback, this was also the device used in Chapter 3. The stylus is equipped with a single button. The force update rate was 1000Hz and the tracker update rate was 120Hz. The input device controlled each of the three cursors with an absolute 1 to 0.35 mapping. Participants were seated during the experiment 3m from the display and wore normal headphones at all times. Figure 4.14 illustrates the experiment apparatus.
4.7 Experiment 2: The Effects of Multimodal Feedback

Figure 4.14: The experiment apparatus (stereo is disabled for illustrative purposes).

4.7.2 Participants

Eleven male and one female unpaid volunteers, ranging in age from 21 to 30 with an average of 25.5, served as participants in this experiment. Participants had not taken part in Experiment 1 and were screened using a stereopsis test in which they had to order objects according to their depth. All participants were right handed and used their right hand to control the input device.

4.7.3 Design

A repeated measures within-participant design was used. The independent variables were: cursor type, $CT$ (3D bubble cursor, depth ray, and lock ray); multimodal feedback, $MM$ (on, off); density spacing, $DS$ (1 and 5 cm); visibility condition, $VC$ (visible, occluded); and target size $SIZE$ (0.75 and 1.5 cm). A fully crossed design resulted in 48 combinations of $CT$, $MM$, $DS$, $VC$, and $SIZE$. The $DS$, $VC$, and $SIZE$ variables behaved exactly the same as in Experiment 1. The multimodal feedback variable indicated the presence of both force and audio feedback. The same visual feedback was used in all conditions.

Each participant performed the experiment in one session lasting about 70 minutes. The session was broken up by the three cursor types. Trials for each cursor were further split between the two values of $MM$. For each $CT$ and $MM$ combination, three
blocks of trials were performed. Within each block, the six combinations of DS, VC and SIZE were repeated three times in a random order, for a total of 18 trials per block. In our previous experiment, 36 scenes were randomly generated for these trials, and were the same across all participants. We reused 18 of those 36 scenes which fit the conditions of this experiment. The cursor ordering was fully counterbalanced across the 12 participants, with two participants randomly assigned to each of the six unique orderings. The multimodal feedback was first off and then on for each cursor for the first group of six participants, and this order was swapped (MM = on, MM = off) for the second group of six participants. Before each cursor, participants were given several warm-up trials to familiarise themselves with the selection technique.

4.7.4 Results

Trial Completion Time

In our analysis of trial completion time, we discarded trials in which errors occurred, and removed outliers that were more than three standard deviations from the group mean (1.6% of the data). Repeated measures analysis of variance showed main effects for CT (F_{2,22} = 9.08, p < 0.001), SIZE (F_{1,11} = 215, p < 0.0001), VC (F_{1,11} = 167, p < 0.0001), and DS (F_{1,11} = 126, p < 0.0001). No effect was found for the MM condition (F_{1,11} = 0.018, p = 0.896). Average trial completion times were 2.34 s for the 3D bubble cursor, 2.44 s for the depth ray, and 2.76 s for the lock ray.

Post hoc comparisons showed that both the 3D bubble cursor and depth ray were significantly faster than the lock ray (p < 0.005) and the difference between the 3D bubble cursor and the depth ray was not significant (p = 0.382). Despite this non-significance, the fact that the 3D bubble cursor times were now faster than the depth ray overall, in contrast to Experiment 1, where they were significantly slower, indicates that the adjusted visual feedback, used for the 3D bubble cursor in this experiment, was effective. The finding that the lock ray was slower than the depth ray is consistent with the prior literature (Grossman and Balakrishnan, 2006), and indicates that the addition of multimodal feedback is not enough to overcome their differences.

Another important result is that the multimodal feedback did not have any significant effect or interaction with the technique (CT), as illustrated in Figure 4.15. This was somewhat disappointing; we had hoped that the multimodal feedback would further improve the users’ awareness during the selection task. But as a positive note, this indicates that the visual feedback provided with the techniques, alone, is sufficient. The only effect of interest related to the multimodal feedback was that MM had a weak but significant interaction with DS (F_{1,11} = 6.39, p = 0.028). Figure 4.16 shows
4.7 Experiment 2: The Effects of Multimodal Feedback

Figure 4.15: Technique completion times by multimodal feedback.

that this effect is barely apparent, but it does indicate that the multimodal feedback could be more useful in dense target environments, when it becomes harder to discern the targets visually. We further investigated the effect of MM by looking at reaction times, as this is a metric that multimodal feedback has been observed to have an effect on in past studies (Akamatsu et al., 1995). It is defined as the time taken to select an object with a button press once it has been captured with the cursor. However, a repeated measures analysis of variance showed no main effect of MM on reaction time either ($F_{1,11} = 0.430, p = 0.525$). This result is similar to the results of our experiment described in Chapter 3 in which the 2D bubble cursor also did not see a significant improvement in reaction time, while the virtual hand did see an improvement as found in previous literature.

We now discuss how the other variables – target size, visibility, and density – influenced the techniques. We found that CT had significant interaction effects with each of these variables: SIZE ($F_{2,22} = 12.8, p < 0.0001$), VC ($F_{2,22} = 118, p < 0.0001$), and DS ($F_{2,22} = 16.6, p < 0.0001$). The interaction between CT and DS is illustrated in Figure 4.17. The general trend, as expected, is that the selection times are reduced with increased density spacing. It is interesting that the 3D bubble cursor is significantly faster than the depth ray and lock ray for density spacing of 5 cm ($p < 0.008$ with bonferroni corrections), while for density spacing of 1 cm there is no significant difference between any of the three cursors. This is in contrast to Experiment 1, where the 3D bubble cursor was slower than the depth ray at $DS = 1$ cm, and similar at $DS = 5$ cm. This again indicates that the new visual feedback for the 3D bubble cursor was effective.
Figure 4.16: Completion times by density and multimodal feedback.

Figure 4.17: Technique completion times by density.
Next, we discuss the interaction between CT and VC. In Figure 4.18 we can see that the occluded condition is slower for each cursor. Post hoc comparisons with bonferroni correction show that it is significant for all cursors (p < 0.001). Most notable is that the 3D bubble cursor seems much less affected by the visibility condition than the ray casting techniques. This shows that the 3D bubble cursor, with the new visual feedback which we introduced, is an effective selection technique when targets may be visible or occluded. We cannot be certain as to why the ray casting techniques were less efficient in the occluded conditions. One possible explanation is that the continuous transparency function was less intuitive, or more difficult to take advantage of, when the metric was based on a distance to a ray, rather than a distance to the centre of the 3D bubble cursor. Another possible explanation is the change of input device. Users might have had more problems using a ray casting technique with the PHANToM as more pointing was done by the wrist in a smaller workspace compared to the larger allowed arm movements with a magnetic tracker. Similar problems have been noted in another selection experiment (De Boeck et al., 2006a).

Finally, Figure 4.19 shows the interaction effect between CT and SIZE. It is interesting to note that size has an effect on each of the three techniques (p < 0.0001), even though size plays no role in the effective width, or motor space activation boundaries. Thus, as in Experiment 1, we are seeing that users are not taking full advantage of the increased activation boundaries created by the techniques. As in Experiment 1, users were more reliant on the visual boundaries of the targets when the targets were closer together. Figure 4.20 illustrates this effect with the 3D bubble cursor, which is almost identical to Figure 4.11 from Experiment 1. We had hoped that the addition of the multimodal feedback would compensate for this effect, improving the
users understanding of the motor space activation boundaries, but this was not the case. This indicates that in dense and occluded environments, to optimise the user’s understanding of the activation boundaries of a desired target, feedback techniques – visual, haptic and audio – even combined are not enough.

The $CT \times SIZE$ interaction effect seems to be a result of the higher times for the lock ray for $SIZE = 0.75$ cm. This may be due to the fact that if users do rely on the visual boundaries of the target, then they have to aim at these smaller targets twice with the lock ray, once with the ray in the selection phase, and once with the depth marker in the disambiguation phase. The original study of the lock ray (Grossman and Balakrishnan, 2006) did not vary size, so this is an interesting result.

### Learning

A learning effect was observed in our experiment, with the block number significantly affecting trial completion times ($F_{2,22} = 16.2$, $p < 0.0001$). Performance improved over time, with average completion times of 2.75, 2.59, and 2.46 s for blocks 1, 2, and 3, respectively. Post hoc comparisons shows that Block 1 was significantly slower than Blocks 2 ($p < 0.05$) and 3 ($p < 0.001$), and Block 2 was also significantly slower than Block 3 ($p < 0.0001$). There was no significant effect between block number and $CT$, or block number and $MM$, indicating the techniques were just as easy to learn with or without the multimodal feedback.
Input Device Footprint

Because selection with a six degree-of-freedom input device requires a user to hold the device in mid-air, it is important that the selection techniques minimize arm fatigue. With the device used in this experiment, a PHANToM, the user rests one arm using the elbow or forearm depending on what is most comfortable for the user. As in Experiment 1, we can quantify potential fatigue by measuring the input device footprint, defined as the length of the total path which the input device traveled to complete each trial. The cursor type had a significant effect on the input device footprint ($F_{2,22} = 45.6, p < 0.0001$). The footprints were 21.2 cm for the 3D bubble cursor, 15.8 cm for the depth ray, and 8.7 cm for the lock ray, all significantly different ($p < 0.005$). The lock ray may have had the lowest value because when the users completed the selection phase, the depth marker was positioned at an optimal location, in the middle of all targets being intersected. Furthermore, movements during the disambiguation phase of the lock ray were constrained to the direction of ray, which may have further reduced the footprint. As with the previous experiment, the depth ray had a lower footprint than the 3D bubble cursor, probably because users did not need to move towards a specific 3D location.

An interesting observation was that the $CT \times VC$ interaction effect was significant ($F_{2,22} = 12.75, p < 0.0001$). For both the depth ray and lock ray, footprints increased significantly for $VC = \text{occluded}$ ($p < 0.001$). However, $VC$ had no effect at all on
the footprints for the 3D bubble cursor. This further supports our understanding of why the 3D bubble cursor was a better technique for occluded targets. The constant device footprint indicates that users were better at using the transparency function with the 3D bubble cursor than with the depth or lock ray to locate the occluded targets. The increased footprint for the ray casting techniques indicates that users moved the cursors more to use the transparency function to locate the target.

Error Rates

As in Experiment 1, errors were defined as trials in which users selected the wrong target before selecting the goal target. The overall error rate for the experiment was 2.89%. The CT did have a significant effect on error rate ($\chi^2(2) = 6.51, p < 0.05$), however the overall error rates were comparable: the 3D bubble cursor had an error rate of 1.9%, the depth ray 3.12% and the lock ray 3.64%. Post hoc analysis showed a significant difference between the 3D bubble cursor and both the depth ray and lock ray ($p < 0.05$). The MM condition did not have a significant effect on the error rate ($\chi^2(1) = 2.99, p = 0.22$).

Subjective Feedback

Users filled out a short questionnaire after each cursor had been tested (see Figure C.3 in Appendix C). An overview of the results can be found in Figure 4.21, with the results represented by median values. We see that the overall results were stronger for the 3D bubble cursor and depth ray. This observation again indicates that even with multimodal feedback, the lock ray is not a preferable technique. Users found that the 3D bubble cursor and depth ray were easy to learn, and allowed for fast selections. It is also interesting to note that the results indicate that users did feel that the multimodal feedback was helpful and preferred, even though our quantitative analysis revealed that the multimodal feedback provided no advantage.

In post-experiment discussions, users reported that the MM condition helped them to better understand when a new target was selected and that it was nice to have the extra feedback. Some persons also commented that they were only relying on either force or audio feedback, but that having both did not seem beneficial. One user did feel that neither form of feedback was helpful. For both the depth ray and 3D bubble cursor users gave general remarks that the forces were sometimes disturbing during general movement. For the lock ray users found the force feedback intuitive and understood

---

2 Due to the low amount of users, we did not perform any statistical analysis.
it well, but some users commented that the boundary force to switch between objects was too strong.

4.8 Comparison between Experiments

In this section, we compare the results of the two experiments which we have presented, with the aim to investigate possible important differences between both. In order to perform this analysis, we have removed certain data from both experiments, such that the conditions which are compared are equivalent. Because the 3D point cursor was only used in the first experiment, and the lock ray was only used in the second, we have removed the data from both techniques. Furthermore, the level of $DS = 2.5$ cm, which only appeared in the first experiment, and the condition $MM = \text{on}$, included only in the second experiment, are both removed. These changes result in a repeated measures analysis of variance with a between-participant factor $EXP$, representing the experiment number.

A repeated measures analysis of variance showed a main effect for $EXP$ ($F_{2,22} = 24.9, P < 0.0001$). Comparing the average completion times showed that the first experiment (3.1 s) was slower than the second experiment (2.4 s). We believe the second experiment was faster because of the different hardware setup. Most notably,
the control gain was altered, and a larger display was used. The faster times may have also been a result of the improved visual feedback used for the 3D bubble cursor.

Aside from this overall difference in completion times, the only significant two-way interaction effect across both experiments was \( \text{EXP} \times \text{VC} \) (\( F_{2,22} = 46.3, P < 0.0001 \)). This interaction occurred because the effect of \( \text{VC} \) was much greater in Experiment 1. This was likely another result of the visual changes which were made between the experiments.

This analysis shows that while our change in setup had an overall effect across all conditions, the results which we have obtained in both experiments are consistent with each other.

4.9 Discussion

The results of our studies have some important implications to user interfaces for virtual environments. We have discussed the design of two beneficial techniques for object selection. The poor performance of the 3D point cursor in Experiment 1 validates our design of new techniques, as well as our discussed design guidelines. While the depth ray was an existing technique for volumetric displays, we have provided, as far as we are aware, its first virtual environment implementation. Similarly we have provided the first implementation of a 3D bubble cursor. Furthermore, these techniques are both original in that they are augmented to allow for the selection of occluded targets.

In the first experiment we found that the depth ray seemed to perform best overall, and that users were not taking full advantage of the increased activation areas created by the 3D bubble cursor. We felt this may have been due to the difficulties in providing adequate visual feedback during selection in dense and occluded targets, which led to some changes in the techniques’ visual feedback. For the 3D bubble cursor we removed the rendering of the outer bubble. For all techniques we switched to a continuous, rather than discrete (on or off) transparency function. This seemed to have an influence in our second experiment. Overall, we felt the 3D bubble cursor performed best in this experiment. The average completion times were not significantly different from the depth ray, but the technique was much less affected by the occlusion condition, and the error rates were lower. However, a benefit of the depth ray, found in both experiments, was that it lowered the input device footprint, which could minimise arm fatigue over extended usage.

Unfortunately, our added multimodal feedback did not provide any observable significant advantages. We expected that this feedback could mitigate the difficulties
associated with only providing visual feedback. Mainly, users seemed to rely on the visual bounds of targets, not taking full advantage of the increased motor activation boundaries. The result that the multimodal feedback did not provide an advantage, while unfortunate, is consistent with prior art, which shows that effects of multimodal feedback during selections can be minimal (Akamatsu et al., 1995; Cockburn and Brewster, 2005; Wall et al., 2002). The problem may be that providing feedback, regardless of the form, only helps the users to become conscious of the fact that they have captured a target. It does not help them plan how to capture it. Or, in other words, the users may not understand the motor activation boundaries until the target has already been captured. It would, thus, be interesting in the future to consider “feedforward” techniques, which give the user an understanding of the activation boundaries before the selection task even begins. For example, the starburst technique renders each target’s surrounding activation area, which are modified Voronoi regions (Baudisch et al., 2008). It would be interesting, to adapt this strategy to 3D, but potentially difficult, to realise without making the environment too visually cluttered. In general, it is worthwhile to continue investigating improvements to the visuals provided during the selection task, as our results show that this plays a significant role in the user’s task performance. Furthermore, understanding the motor activation boundaries by the users might be a reason for improved reaction times when using multimodal feedback as was found in previous literature and in Chapter 3 for the virtual hand selection technique.

It is not surprising that our results showed that for all techniques, occluded targets took longer to select. However, it is important to note that our transparency function did enable users to minimise the overhead cost for such tasks to just one second. If a more traditional approach were used for making the target visible, such as using a hotkey to switch viewing modes or rotating the scene, we would expect to see a similar, if not greater, overhead cost. The more traditional approach would also have an added drawback of extra buttons or input devices being required. Besides, this transparency function could also be used for higher level tasks such as the exploration of dense and complex environments.

While the addition of multimodal feedback had no significant influence, users did generally like the extra feedback. If the user’s task did not otherwise require a haptic device, it would probably not be worth introducing this device just to provide the feedback, given its limited impact. However, the audio feedback could be incorporated for user’s who preferred it, without any overhead costs.
4.10 Conclusion

We have presented an in-depth evaluation of techniques which support selections in dense target environments, and of targets which are fully obscured from the user’s viewpoint. In particular, we found two techniques, the depth ray and 3D bubble cursor, which outperformed a baseline 3D point cursor in our experimental task, providing faster selection times and lower device footprints. In a second experiment, we found that the addition of multimodal feedback to these techniques had small effects, although it was received well by users. The results of the second experiment also showed that our iteration on the visual feedback provided during the selections was helpful. These results indicate that 3D user interface designers should pay close attention to the visual stimuli provided in their designs, and if done properly, should not have to rely on specialised hardware to provide other forms of feedback.

The force feedback used in the second experiment was designed using a trial-and-error approach. It is unclear if another force amplitude, profile and duration would have given better results, and even more important, would another combination of force amplitude, profile and duration have deteriorated the performance of the user? In the next chapter we will try to answer this question.
Chapter 5

Investigating the Force Feedback Magnitude Effects on User’s Performance during Target Acquisition

Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Introduction</td>
<td>81</td>
</tr>
<tr>
<td>5.2</td>
<td>Related Work</td>
<td>83</td>
</tr>
<tr>
<td>5.3</td>
<td>Experiment 1: Force Amplitude and Direction</td>
<td>86</td>
</tr>
<tr>
<td>5.4</td>
<td>Experiment 2: Force Shape and Duration</td>
<td>94</td>
</tr>
<tr>
<td>5.5</td>
<td>Experiment 3: Subjective Preference</td>
<td>102</td>
</tr>
<tr>
<td>5.6</td>
<td>Investigation of the Device Influence</td>
<td>104</td>
</tr>
<tr>
<td>5.7</td>
<td>Experiment 4: Objective Device Comparison</td>
<td>107</td>
</tr>
<tr>
<td>5.8</td>
<td>Experiment 5: Subjective Device Comparison</td>
<td>113</td>
</tr>
<tr>
<td>5.9</td>
<td>Discussion and Implications</td>
<td>115</td>
</tr>
<tr>
<td>5.10</td>
<td>Conclusion</td>
<td>118</td>
</tr>
</tbody>
</table>

5.1 Introduction

Practical applications of force feedback often try to render the generated forces realistically, trying to provide the user with a natural sensation. Several simulation applications may serve as examples of this approach (Stone, 2000). Research has also focused on the extra information channel provided by haptic feedback, independent of the realism of the generated forces. This extra feedback provides the users with additional information during their interaction such that their experience or their performance can be improved. Examples of this kind of feedback include the force
feedback used in the previous chapters, Tactons (Brewster and Brown, 2004) or the use of a haptic belt for navigation at sea (Van Erp et al., 2004).

Similarly this abstract force feedback can be applied to support pointing tasks in a desktop application or in a virtual environment. This can be achieved either by assisting the user e.g. using gravity wells (a “snap-to” effect that pulls the user’s pointer to the centre of the target) (Hwang et al., 2003; Magnusson and Rassmus-Gröhn, 2005; Oakley et al., 2000; Wall et al., 2002; Van Mensvoort et al., 2008) or by giving haptic feedback in the form of a bump or a vibration as soon as an event occurs (Akamatsu et al., 1995; Cockburn and Brewster, 2005; Forlines and Balakrishnan, 2008). In the previous two chapters we experimented with both such types of feedback. In Chapter 3 we performed an extra pilot experiment, discussed in Section 3.4.1, to identify possibly beneficial parameters for the selection technique, while in Chapter 4, we designed the force feedback using trial-and-error until suitable values were found. Most of our or other researchers’ experiments focusing on user’s performance when applying force feedback found only small improvements, even not always statistically significant. In this context, the fundamental question may arise what kind of forces may or may not be applied in order to support (or at least not disturb) the user. To be able to deduce guidelines we decided to search for a point of performance deterioration, rather than directly investigating statistically significant improvements.

It is exactly this challenge that we take up in this chapter. We study how the amplitude, as well as the force shape and the duration of a force bump may influence the user’s performance. We define the force bump as a short force in a given direction, with given duration and amplitude. The amplitude of the force over time may follow a mathematical pattern such as a sine or a step function, which we define as the force shape. Figure 5.1 illustrates a sinusoidal bump which may occur in any direction, for example in a lateral or longitudinal direction with respect to the user’s movement. The grey shaded area in the figure is the definite integral, which in this chapter we refer to as the force integral (FI).

There are many possible degrees of freedom: device, bump shape, duration, amplitude, direction, etc. In a first experiment we measure for a constant bump shape and duration if certain amplitudes cause significant reduction of the user’s performance. This gives us a baseline measurement, which can be generalised in the next experiment. Based on the behaviour of the physical system (user-device), we formulate a hypothesis using the definite integral of the force bump (see Figure 5.1). This allows us to predict the influence of other force bumps with other amplitudes, shapes and durations. In a second experiment, we verify the validity of this hypothesis. A third experiment probes for the user’s subjective impression when ‘disturbing’ and
‘non-disturbing’ forces occur during a selection task. Finally, in a fourth and fifth experiment we also check if different devices with different mechanical parameters behave differently as these parameters may have an influence on the performance of the device itself and on the user’s experience.

These experiments are valuable for designers who want to apply force feedback for a targeting, pointing or selection task. They can use the results as a guideline to know what force values must not be exceeded. Before describing the experiments and discussing the results, in the next section we first introduce several relevant topics with regard to the conducted experiments.

\[ \text{Figure 5.1: An illustration of the force evolution of a sinusoidal haptic bump over time.} \]

5.2 Related Work

In this section we give an overview of tactile and force feedback during target acquisition and existing frameworks and methods to compare haptic devices.

5.2.1 Haptics Supporting a Target Acquisition

An important topic in the domain of haptic feedback research focuses on the integration of tactile or force feedback in Graphical User Interfaces (GUIs), e.g. for selecting
items, such as icons and menu items. Similarly, in the domain of 3D virtual environments a vast amount of research focuses on haptic feedback as well (Stone, 2000; Van Erp et al., 2004). However, research particularly focusing on the use of haptics supporting a selection task, is rather limited as we discussed earlier in Section 3.3.1. We will again discuss some of the most relevant literature with a focus on the force design and its parameters.

Akamatsu et al. (1995) and Cockburn and Brewster (2005) used tactile vibrations to indicate that the mouse cursor hovered over a target. They both found that tactile vibration could improve the performance in certain situations although they reported that the vibration could make the user miss small targets. However, they did not investigate what type of vibration could be most useful or in such a way distracting that users deviated from their motion path. Oakley et al. (2000) investigated different kinds of force feedback: texture, friction, recess and gravity wells. They found that the texture and friction effect gave much worse results due to the fact that the user was not constrained to the target and the user’s movements were perturbed. It is not clear if an analysis about the strength of the effects would have made it possible to improve the perturbation effects. Likewise, Forlines and Balakrishnan (2008) evaluated tactile feedback with direct or indirect stylus input and found that a small amount of tactile feedback is beneficial.

Our research discussed in both previous chapters investigated the addition of force feedback for an object selection task in a virtual environment. Especially the design of the force feedback of Chapter 4, a small sine-wave force of a short duration to indicate when a user had switched between indicated targets, argues that more fundamental research into force feedback design has to be performed. A small non-significant improvement was found, but the influence of the feedback on the user’s movement was not clear. Sparked by the aforementioned research, the work presented in this chapter investigates what force strengths, shapes and durations may or may not be useful in a user interface supporting force feedback.

The influence of the force on the user’s movement can be investigated with motion path analysis. This may for instance be useful when motion impaired users are involved (Langdon et al., 2002; Hwang et al., 2003). Performing motion path analysis is tedious as every trial would have to be analysed, which is a brute force approach (Hwang et al., 2004). A more refined approach with similar results makes use of the accuracy measures for evaluating computer pointing devices, proposed by MacKenzie et al. (2001). Amongst others, these measures indicate how many times the users changed their direction or diverted from the ideal path. In our experiments, we will use these measures to determine the influence of the applied forces on the performance.
5.2 Related Work

5.2.2 Comparison of Haptic Devices

In addition to the first three experiments, in the final two experiments of this chapter, we will be comparing two haptic devices. Therefore we will give an overview of existing frameworks and methods to compare haptic devices.

Kirkpatrick and Douglas (2002) defined a haptic mode as a distinct style of using haptic perception for a purposeful activity and proposed a taxonomy representing different haptic modes for goal-oriented applications. This taxonomy allows researchers to categorise their experiments into a haptic mode which can be used by designers in order to choose the correct haptic device for their applications. These haptic modes were used by Samur et al. (2007) to create a testbed for evaluating haptic interfaces. They proposed to express information transfer in bits (index of performance) for several different tasks in a virtual environment. This uniform representation allows to compare different experiments with different haptic devices. In this paper we perform a comparison in the motor control - target acquisition haptic mode. Our experiments have a static Fitts’ task index of difficulty (Fitts, 1954) and focus on different force magnitudes, which do not allow us to compute the information transfer as more indices are needed.

Only few experiments exist which have compared different haptic devices. Harders et al. (2006) compared three different haptic devices in a 3D assembly task. They saw small but non-significant differences with regard to completion time and subjective preference. Therefore they could not deduce which device was clearly superior for this particular task. Another comparison was performed by Yu and Brewster (2002); they compared a PHANToM and a Logitech Wingman Force for multimodal graph rendering, but similarly as Harders et al. they did not see any significant difference in the multimodal condition.

Hayward and Astley (1996) presented standard metrics for evaluating and comparing haptic devices: Motion Range, Ground Device Inertia, Precision, etc. Similarly, Guerraz et al. (2003) argued to use the physical parameters of the device to have a better evaluation of the benefits of the applied force feedback. They discuss the usage of the position, velocity and force, but also mechanical parameters such as stiffness, viscosity and inertia. In our case, we will calculate the relevant mechanical parameters by means of the device’s frequency response, as described by Pintelon et al. (1994).
5.3 Experiment 1: Force Amplitude and Direction

As we initially have no accurate idea of what kind of force bumps and what magnitudes may or may not cause a significant deterioration of the selection task, the first experiment measures some baseline values. Because there are many independent variables in this experiment (device, bump shape, bump duration, force direction, force amplitude, ...), we only take into account the force direction and magnitude for a given device, duration and shape.

The following hypotheses will be investigated in this experiment:

- From a certain force magnitude the users’ performance (trial completion time and error rate) will deteriorate.
- Different force directions will influence the users differently with regard to their path during movement, with regard to their performance it is hard to predict.

5.3.1 Participants

Nine male and one female unpaid volunteers, ranging in age from 22 to 30 with an average of 25, served as participants in this experiment. They were selected among co-workers. All participants were right-handed and used their dominant hand during the experiment.

5.3.2 Apparatus

The display we used was a 19-in (18” viewable) ViewSonic P90f with a resolution of 1280 by 1024. The input device was a PHANToM premium 1.0 with a control display gain of 1, which means that 1 cm on the screen represents 1 cm in the physical space. For validity purposes of the experiment we took extra care in calibrating the input device. Using a “forcemeter” (see Figure 5.2), we measured the influence of the gravity, unbalanced weight (Massie and Salisburg, 1994), on the stylus of the PHANToM, as well as the gain factor between the force requested in the software (CHAI3D API (Conti et al., 2005)) and the final result at the device. We found that a gravity compensation of only 0.08N downwards was required and forces provided by the software were nearly equal to the final forces measured at the device. Another important factor that had to be taken into account was the inertia and the internal friction of the device. Obviously, we desire values that are as low as possible as higher
frictions and higher inertia may interfere with or “smooth out” the haptic bump. As the PHANToM premium is designed to keep these values as small as possible, this device suits our needs.

Figure 5.2: The PHANToM and the forcemeter being used to calibrate it.

5.3.3 Procedure

A simple multidirectional point-select task, as described in ISO 9241-9 (ISO, International Organisation for Standardisation, 1998), was used for this experiment. Ten targets are placed in a circle on the screen (see Figure 5.3). The diameter of the circle is determined at 6 cm and the size of a target is 0.7 cm (we use physical measures rather than pixels, since pixel sizes vary from display to display). This task has a Fitts’ index of difficulty of 3.26 bits. This is a measure typically used in Fitts’ law experiments to indicate the difficulty of the task (Fitts, 1954). The value is chosen to be comparable to the task difficulty of a typical icon selection task (Douglas et al., 1999; Soukoreff and MacKenzie, 2004).

We also had to take into account the implications of the movement scale; the limb segments of the user involved in the task depend on the physical distance that has to be covered (Langolf et al., 1976; Balakrishnan and MacKenzie, 1997). Usually, the operation of desktop haptic devices is situated in the range of the wrist and fingers. Therefore a 6 cm distance appeared to be a good value (Accot and Zhai, 2001), as it will adhere to a typical movement to be expected with the device.
During the test, the ten targets were highlighted one after the other and users were requested to select the highlighted target “as efficient” (fast and accurate) as possible, by pointing and clicking. Highlighting is altered between opposite sides of the circle so that it requires the user to make movements equally distributed among all directions with a maximum distance between the targets.

As the task to perform was a 2D selection task and the haptic device we used is a 3D input device, a vertical guiding plane restricted the task to two dimensions. In order to make sure that users did not use the guiding plane as extra support, such that the forces had less impact on their movement, we provided them with extra visual feedback about their position inside the guiding plane. The background colour was completely black within a certain offset of the guiding plane and interpolated to white the more the user pushed into the plane. Users were instructed to avoid having a continuous grey/white background.

Finally, force feedback appearing in the form of a force bump with given shape, duration and amplitude was activated when exactly half-way in the path to the next target. Half-way the path is calculated using a simple Euclidian distance. Note that this activation strategy serves as a distractor without any purpose of being beneficial. We hope this experimental approach allows us to find a cutoff point which shows us what bumps may and may not be disturbing. Alternatively, if we would focus on the
5.3 Experiment 1: Force Amplitude and Direction

usage of the force feedback in a beneficial manner, it would further complicate our findings in an environment where already many degrees of freedom exist (amplitude, shape, duration, . . . ).

5.3.4 Independent Variables

We expect that the direction of the applied force bump may have an influence on the outcome. Three different directions ($D$) are considered: the force can be lateral to the moving direction ($d_{lat}$), longitudinal in the direction of the current movement ($d_{long}^+$) and longitudinal opposite to the direction of the current movement ($d_{long}^-$). The force bump (Shape $S$) for this experiment is defined as a $\sin(x)$-function over the interval $[0, \pi]$ ($S=\sin(0,\pi)$). The total duration of the interval ($T$) is determined at $T=75$ ms for this experiment. The amplitude ($A$) is chosen from the following values: ($A=0.0N, 0.2N, 0.4N, 0.6N, 0.8N, 1.0N, 1.2N, 1.4N, 1.6N, 1.8N, and 2.0N$). Due to the fact that we have a constant force shape and duration, the only independent variable is the force amplitude. Combining however the different parameters, we can also represent the values by the definite integral of the shape function. From here on, we call these values the force integral ($FI$) (see Figure 5.1), defined as $\int_0^T \text{abs}(A \cdot f(x))dx$. The following values represent the integral values ($FI = 0.0, 9.55, 19.10, 28.65, 38.20, 47.75, 57.30, 66.85, 76.39, 85.94, and 95.49$) used in this experiment and which will also serve as the independent variable during the design and analysis.

5.3.5 Design

A repeated measures within-participant design was used. The independent variables were: force direction $D$ ($d_{lat}$, $d_{long}^+$, and $d_{long}^-$); and force integral $FI$ ($0.0, 9.55, 19.10, 28.65, 38.20, 47.75, 57.30, 66.85, 76.39, 85.94, and 95.49$). A fully crossed design resulted in 33 combinations of $FI$ and $D$.

Each participant performed the experiment in one session lasting about 20 minutes. The session consisted of five blocks with each block containing the 33 combinations (11 $FIs$ and 3 $Ds$) repeated three times in a random order. For a total of 99 trials per block, this resulted in 495 trials per participant. Between each block, users were obliged to take a break of at least 15 seconds to minimise fatigue during the test. Before the experiment, participants were given all 33 conditions in random order to familiarise them with the task.

For each selected target, the time it took to select a target was recorded, furthermore all parameters were logged at 1000Hz and after low-pass filtering and downsampling
90 Investigating Force Feedback Magnitude Effects during Target Acquisition

to 200Hz stored in a database. Parameters include the actual position of the cursor, velocity, the exerted force bump and the number of button presses before a successful selection. Additionally the Movement Direction Change (MDC), Orthogonal direction change (ODC), Movement Variability (MV), Movement Offset (MO) and Movement Error (ME) were calculated according to the recommendations of MacKenzie et al. (2001). In order to have an intuition for these measures, an overview is provided in Figure 5.4.

(a) Movement Direction Change: amount of changes relative to the task axis.
(b) Orthogonal Direction Change: amount of changes along the axis orthogonal to the task axis.

(c) An overview and comparison of Movement Variability, Movement Error and Movement Offset.

Figure 5.4: The measures used during motion path analysis. (MacKenzie et al., 2001)

5.3.6 Results

We first investigated the general learning effect during the experiment, comparing the results of the different blocks. Repeated measures analysis of variance showed a main effect for Block \(F_{4,36} = 5.3, p < .003\). Post hoc comparisons showed that block 1 and 2 were slower than block 3, 4 and 5 \((p < 0.05)\), while between the last three blocks no significant differences occurred. These findings can be seen in Figure 5.5, as well. From now on, as we want to eliminate the results of any learning effect in our analysis, we will discard the first two blocks during the analysis. Further analysis of the data revealed two outlying measures, which were five times the standard deviation away from the mean. Those values were excluded as well.
Figure 5.5: Learning effect of the experiment.

**Trial Completion Time**

A repeated measures analysis of variance of the faultless trials, showed main effects for $D$ ($F_{2,18} = 5.42, p < .02$) and $FI$ ($F_{10,90} = 12.6, p < .0001$). The average trial completion times for direction are 901.5 ms for $d_{long}^+$, 916.2 ms for $d_{long}^-$ and 905.5 ms for $d_{lat}$. Although these differences are small with respect to practical use, post hoc comparisons showed that $d_{long}^+$ is significantly faster than $d_{long}^-$ ($p < 0.01$), which is a result we could expect as $d_{long}^-$ always opposes the user’s movement while $d_{long}^+$ stimulates the user’s movement until overshoot occurs in the final $FI$ conditions. Between $d_{lat}$ and $d_{long}^-$ no significant difference was found, which was not surprising as both force directions are less ideal with relation to the movement direction. No significant difference was found between $d_{lat}$ and $d_{long}^+$. Post hoc comparisons on the $FI$ condition showed many significant results. Figure 5.6 represents these differences ($p < .05$) darkly shaded. Comparing the no-force condition with all other $FI$’s (first column), a significant trial completion time deterioration, all $p < .002$, can be seen from $FI = 66.85$ (1.4N force amplitude). Figure 5.6 shows that this deterioration is also confirmed by the comparison of the other $FI$ conditions, and Figure 5.7 depicts all the average completion times for all $FI$ values in a more traditional graph.
The analysis also showed an interaction effect $D \times FI$ ($F_{20,180} = 1.96, p < .02$). Further analysis of this interaction effect did not show anything interesting. We could presume that different force directions and different force strengths may interfere with the particular distances and target sizes of the task. For instance, it can be seen that the $d_{long}^{+}$ condition for certain forces, for certain users, assists them by bringing the pointer closer to the target (without overshoot). However, as this “assist” is highly dependent on the force, the distance and the target size, but also on the mass and the stiffness of the particular user’s hand holding the device, it is difficult to derive general conclusions from this.

**Other Measures**

In this paragraph we shortly discuss the parameters logged for motion path analysis. Starting with MO, it did not show a significant effect with $D$ and $FI$. This can be expected as the forces did not cause the users to divert from their path: obviously $d_{long}^{+}$ and $d_{long}^{-}$ exert a force in the movement direction, but also with $d_{lat}$, the original movement path is restored relatively quickly. The lateral velocity profiles depicted in Figure 5.9b and 5.9f (upper graph), illustrate this.

The other parameters (MV, ME, MDC and ODC) showed main effects for $D$ ($F_{2,18}$, all $p < .01$) and $FI$ ($F_{10,90}$, all $p < .01$). Post hoc comparison taught us that MV for $d_{lat}$ is significantly higher than for $d_{long}^{+}$ and $d_{long}^{-}$ ($p < .001$). Obviously, a lateral force causes a higher variability with regard to the task axis than longitudinal forces which are mainly in the direction of the task axis.

The significance levels for $FI$ showed very similar patterns compared to the results of the trial completion time: from $FI=66.85$ (or 1.4N force amplitude) a significant
difference occurs, confirming the trial completion time deterioration we concluded in the previous section.

With regard to ME, the results didn’t follow any expected behaviour, possibly due to the fact that movement error is not mainly influenced by the force feedback, but by the user’s movement instead.

The direction changes (MDC and ODC) showed no surprising results either: the \( d_{\text{lat}} \) condition shows significantly more movement direction changes than \( d_{\text{long}}^+ \) (\( p < 0.001 \)) and \( d_{\text{long}}^- \) (\( p < 0.05 \)). Alternatively, the orthogonal direction change for \( d_{\text{long}}^+ \) was higher than for \( d_{\text{long}}^- \) and \( d_{\text{lat}} \) (\( p < 0.001 \)), probably due to the large overshoot for the strong force conditions. Finally, \( d_{\text{long}}^- \) has a higher ODC than \( d_{\text{lat}} \) (\( p = 0.055 \)).

**Error Rates**

An error for this selection task was recorded when the user clicked the button without having the correct target underneath the pointer. The overall error rate for the experiment was 114 errors or 2.3%. The direction had no significant effect on error rate \( D \) (\( F_{2,18} = .668, \ p = .525 \)). However, the force integral values did show a significant
Investigating Force Feedback Magnitude Effects during Target Acquisition

The effect $FI$ ($F_{10,90} = 2.62, p < .01$). Post hoc comparisons showed that the no-force condition (0.7%) has a significant difference with the last two $FI$ conditions (85.94 and 95.49, respectively 5.2% and 5.6%). This indicates that a higher amount of errors occurred for a higher $FI$ condition in comparison to when the deterioration of the user with regard to trial completion time occurred. In the scope of this finding, we might refer to the remarks of several users reporting involuntary miss-clicks due to the large forces, and due to the forces diverting them from the anticipated target click.

Subjective Feedback

In a post-experiment questionnaire (see Figure C.4 in Appendix C) all users indicated that some of the forces were too strong, which confirms with our findings that there is a performance penalty for some forces. We also asked the participants to estimate the amount of different forces they felt which resulted in an average of 3.1 different force strengths ($SD = 0.69$). Although no validation with an adaptive staircase method (Leek, 2001) was performed, this result is in line with previous findings (Cholewiak et al., 2008; Tan et al., 2007), although (Yang et al., 2008a) found that force magnitude perception deteriorated during hand movement. Another interesting result is that 8 out of 10 participants were able to discriminate the three different force directions, which confirmed our expectations based on existing research (Yang et al., 2008b).

An important remark, which most users made was that the strongest forces made them accidently click the button of the device, increasing the error rates, as discussed in the previous section.

5.4 Experiment 2: Force Shape and Duration

In a second experiment described in this chapter, we focus on the force shape and the duration of the force. The previous experiment demonstrated that for a given force shape and duration, a significant performance penalty occurs when the force amplitude passes a certain value. To provide a more convenient calculation of the different parameters, we already proposed the force integral ($FI$) as the independent variable, as illustrated in Figure 5.1.

From the physical relationship between force, velocity and position, we may suppose that the resulting effect of the applied force is related to the (double) integral of the force shape. Keeping this in mind, we formulate the hypothesis that the force integral
can serve as a convenient guideline that allows us to predict whether or not a given force bump will deteriorate the user’s performance. In this section we try to verify this assumption by measuring the results of haptic bumps with different force shapes, different durations and different amplitudes, but with the same $F/I$, and compare how they relate to each other.

The following hypothesis will be investigated in this experiment:

- The force integral ($F/I$) can be used as a guideline which allows us to predict whether or not a given force bump will deteriorate the user’s performance. Thus, different shapes and durations (with the same force integral) won’t influence the performance differently.

### 5.4.1 Participants

Eight male and two female unpaid volunteers, ranging in age from 25 to 31 with an average of 27, served as participants in this experiment. They were selected among co-workers which did not participate in the previous experiment. All participants except one were right-handed and used their dominant hand during the experiment.

### 5.4.2 Apparatus and Procedure

For this experiment, we used the same apparatus and applied the same experimental procedure as described in Experiment 1.

### 5.4.3 Independent Variables

In order to focus on a minimum of independent variables, we are convinced that we can use $d_{lat}$ as the only direction for this second experiment. We motivate our choice by the fact that $d_{lat}$ was the only direction that did not differ significantly from the other directions in Experiment 1. Moreover, both $d_{long}^+$ and $d_{long}^-$ are more subject to undershoot or overshoot in the case of the strong force conditions.

The following shapes $S$ are considered, a visual representation can be found in Figure 5.9:

- The same sine as in Experiment 1 ($S=sin_{[0,\pi]}, T=75\text{ ms}$)
- A step function ($S=sqr, T=40\text{ ms}$)
Investigating Force Feedback Magnitude Effects during Target Acquisition

<table>
<thead>
<tr>
<th>Force Integral</th>
<th>$\sin([0, \pi])$ (75 ms)</th>
<th>$\text{sqr}$ (40 ms)</th>
<th>$\sin([0, \pi])$ (110 ms)</th>
<th>$\sin([0, 2\pi])$ (75 ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.0N</td>
<td>0.0N</td>
<td>0.0N</td>
<td>0.0N</td>
</tr>
<tr>
<td>9.55</td>
<td>0.2N</td>
<td>0.24N</td>
<td>0.14N</td>
<td>0.2N</td>
</tr>
<tr>
<td>19.10</td>
<td>0.4N</td>
<td>0.48N</td>
<td>0.27N</td>
<td>0.4N</td>
</tr>
<tr>
<td>28.65</td>
<td>0.6N</td>
<td>0.72N</td>
<td>0.41N</td>
<td>0.6N</td>
</tr>
<tr>
<td>38.20</td>
<td>0.8N</td>
<td>0.96N</td>
<td>0.55N</td>
<td>0.8N</td>
</tr>
<tr>
<td>47.75</td>
<td>1.0N</td>
<td>1.19N</td>
<td>0.68N</td>
<td>1.0N</td>
</tr>
<tr>
<td>57.30</td>
<td>1.2N</td>
<td>1.43N</td>
<td>0.82N</td>
<td>1.2N</td>
</tr>
<tr>
<td>66.85</td>
<td>1.4N</td>
<td>1.67N</td>
<td>0.95N</td>
<td>1.4N</td>
</tr>
<tr>
<td>76.39</td>
<td>1.6N</td>
<td>1.91N</td>
<td>1.09N</td>
<td>1.6N</td>
</tr>
<tr>
<td>85.94</td>
<td>1.8N</td>
<td>2.15N</td>
<td>1.23N</td>
<td>1.8N</td>
</tr>
<tr>
<td>95.49</td>
<td>2.0N</td>
<td>2.39N</td>
<td>1.36N</td>
<td>2.0N</td>
</tr>
</tbody>
</table>

Table 5.1: Equivalent force amplitudes calculated using the force integral value and the definite integral.

- A sine wave but with a longer duration ($S=\sin([0, \pi]), T=110\text{ ms}$)
- A full sine wave [0, $2\pi$]. It is interesting to see how this shape will behave as it produces positive and negative forces ($S=\sin([0, 2\pi]), T=75\text{ ms}$)

The amplitudes and the force integral values, chosen in the first experiment are considered again, but recalculated for the new shapes according to the definite integral. This results in the amplitudes given in Table 5.1.

5.4.4 Design

A repeated measures within-participant design was used. The independent variables were: force integral $FI$ (0.0, 9.55, 19.10, 28.65, 38.20, 47.75, 57.30, 66.85, 76.39, 85.94, and 95.49); and shape $S$ ($\sin([0, \pi]), T=75\text{ ms}$; $\text{sqr}, T=40\text{ ms}$; $\sin([0, \pi]), T=110\text{ ms}$; and $\sin([0, 2\pi]), T=75\text{ ms}$). A fully crossed design resulted in 44 combinations of $FI$ and $S$.

Each participant performed the experiment in one session lasting about 25 minutes. The session consisted of five blocks with each block containing the 44 combinations (11 $FI$s and 4 $S$s) repeated three times in a random order. For a total of 132 trials per block, this resulted in 660 trials per participant. Between each block, users were obliged to take a break of at least 15 seconds to minimise fatigue during the test.
Before the experiment, participants were given all 44 conditions in random order to familiarise them with the task. Finally, we logged the same parameters as in the first experiment.

### 5.4.5 Results

We first investigated the general learning effect during our experiment by comparing the results of the different blocks. As in our first experiment, the first two blocks were removed to eliminate the results of any learning effect ($\text{Block } (F_{4,36} = 8.7, p < .0001)$).

#### Trial Completion Time

A repeated measures analysis of variance of the faultless selection trials showed no main effect for $S$ ($F_{3,27} = 1.92, p = .151$) which implies that the shapes did not differ significantly from each other in overall trial completion time: 881.6 ms for $\sin_{[0,\pi]}$, $T=75 \text{ ms}$, 893.2 ms for $\text{sqr}$, $T=40 \text{ ms}$, 874.9 ms for $\sin_{[0,\pi]}$, $T=110 \text{ ms}$ and 907.5 ms for $\sin_{[0,2\pi]}$, $T=75 \text{ ms}$. This result was to be expected, as we hypothesised that the $FI$ values would be the most important factor with regard to the trial completion time of the user. Analysis also showed a main effect for $FI$ ($F_{10,90} = 8.6, p < .0001$). Post hoc comparisons showed that, similar to the first experiment, at a certain $FI$ value the trial completion time deteriorates significantly, $FI=76.39$ or 1.6N force amplitude ($p < 0.02$). A similar pattern as in Figure 5.7 was seen but for this experiment the deterioration occurred one $FI$ condition later. We have to guess at the cause of this shift, as it may or may not be caused by coincidence.

Although $S$ did not show a significant main effect, it does show an interaction effect with $FI$ ($F_{30,270} = 1.66, p < .02$). Figure 5.8 shows the interaction: all shapes show a similar pattern with regard to the force integral, except for $\sin_{[0,\pi]}$, $T=110 \text{ ms}$. This shape does not seem to have an equally strong deterioration at the higher $FI$ conditions. Several reasons may cause this effect, but future research is necessary to verify our suppositions: it can be argued that the less pronounced deterioration of $\sin_{[0,\pi]}$, $T=110 \text{ ms}$ is due to the lower amplitude which may be partly masked by the friction of the device. Alternatively, it can also be argued that the longer period of force activations gives more opportunity for the user’s reflexes to counter the deviation and apply a compensation.
Velocity Analysis

As an additional analysis, the study of velocity profiles can provide us with a deeper understanding of the different stadia in the user’s motion. Figure 5.9 shows a typical lateral (top graphs in each figure) and longitudinal (bottom graphs in each figure) velocity profile for the different shapes for different FIs. We have to stress that the graphs shown in this figure are individual selection trials of individual users. It has to be noted that the entire population of all velocity profiles is subject to a large variation. However, the selected graphs give a good representation of the velocity’s behaviour in general.

Figure 5.9a shows a selection without applied force. In the topmost graph, we see a small lateral velocity variance around zero. The longitudinal velocity behaves according to the optimised initial impulse model of Meyer et al. (1988). In a first phase, the ballistic movement (BM in Figure 5.9a), the velocity profile looks like a parabola or a Gaussian. In the depicted example, this first phase is about to finish at sample 120 (or 600 ms). The second phase, the controlled movement phase (CM in Figure 5.9a), follows with a much lower but constant velocity intended to accurately reach the target.

**Figure 5.8:** Force integral values by shape.
5.4 Experiment 2: Force Shape and Duration

Figure 5.9: Velocity Profiles of some “typical” movements: Upper graphs contain the lateral velocity (orthogonal with moving direction), lower graphs contain the longitudinal velocity (parallel with moving direction).
100 Investigating Force Feedback Magnitude Effects during Target Acquisition

Figure 5.9: Velocity Profiles of some “typical” movements: Upper graphs contain the lateral velocity (orthogonal with moving direction), lower graphs contain the longitudinal velocity (parallel with moving direction).

(f) $\sin[0, \pi]$, $T=75$ ms, $FI=95.49$

(g) sqr, $T=40$ ms, $FI=95.49$

(h) $\sin[0, \pi]$, $T=110$ ms, $FI=95.49$

(i) $\sin[0, 2\pi]$, $T=75$ ms, $FI=95.49$
Looking at Figures 5.9b, 5.9c, 5.9d and 5.9e, they represent the velocity profiles when a small force \((FI=38.20)\) for the respective shapes is applied. In all cases, we see a clear influence of the bump on the lateral velocity, in the form of a small oscillation. From the longitudinal velocity profile, we can learn that it is not (or only slightly) affected by the force bump.

Figures 5.9f, 5.9g, 5.9h and 5.9i show the velocities when a large force \((FI=95.49)\) is applied. Obviously, we see a similar but larger effect on the lateral velocity. Surprisingly, however, is the behaviour of the longitudinal velocity, which drops down to zero immediately after the bump. That this dramatic decrease of velocity is not intended as the end of the ballistic movement phase, is confirmed by many profiles that show a new (second) but shorter ballistic movement phase (BM2 in Figure 5.9i). Manually cataloguing the profiles shows that 80% of the trials \((\sin[0,\pi], T=75\, ms, FI=95.49)\) have a clear speed reduction, or even a complete halt. Probably the temporarily ‘halt’ induced by the user will help to dampen the unwanted oscillation, caused by the force bump. We believe that the significant deterioration discussed in the previous sections is mainly caused by this (unconscious) speed reduction, rather than the extra distance physically induced by the oscillation.

From the analysis in the previous section, we found that the longer sine \((\sin[0,2\pi], T=110\, ms)\) behaved in a somewhat different manner. It did not show the significant trial completion time deterioration we should expect for the highest two \(FI\) conditions. From the analysis of the velocity profiles, we found that only 64% of the selection trials had a clear speed reduction or halt (compared to 80% for \(\sin[0,\pi], T=75\, ms\)). Figure 5.9h shows how the longitudinal velocity is less affected, with only a short speed reduction, and no halt, as shown in Figure 5.9g. We cannot prove why the longer sine behaves different, but a possible explanation might be that longer application of a force leaves more time for the user to compensate for the force and therefore 110 ms might reach the boundary in which the force integral can be applied as a guideline.

**Other Measures**

Analysing the other parameters such as ODC, MDC, MV, ME and MO showed a similar pattern for \(FI\) as in the first experiment. For \(S\) we saw no surprising significant effects, therefore we will only shortly describe the most important ones.

The MDC showed a main effect for \(S\) \((F_{3,27} = 5.27, p < .005)\). Post hoc comparisons showed that the full sine’s MDC \((\sin[0,2\pi], T=75\, ms)\) is different from the others with an average exactly 1 higher \((p < .03)\). ME and MV also showed a main effect for \(S\) \((F_{3,27} > 13, p < .001)\) and post hoc comparisons showed similar patterns with
significant differences between all the shapes except the sine shapes. The step shape has significant higher values (mean = 7.4) while the full sine has significantly lower values (mean = 6.1) (all \( p < 0.001 \)).

We believe that the shape could induce a difference in variability with regard to the movement but has no major influence on the performance of a user. The fact that MO and ODC showed no main effects for \( S \) supports this argumentation, up to some extent the user’s offset to the ideal task axis does not differ significantly.

**Error Rates**

Analogous to the first experiment, an error was recorded when the user did not click on the correct target. The overall error rate for this experiment was 139 errors or 2.1% which is similar to the first experiment. The shape had no significant effect on the error rate \( S \) (\( F_{3,27} = .657, p = .586 \)). The force integral values showed no significant effect on the error rate either, although the p-value approached the significance level \( FI \) (\( F_{10,90} = 1.78, p = .076 \)).

Further inspection on this unexpected result, revealed that in this experiment users made the same amount of errors in the low \( FI \) conditions as they did in the high \( FI \) conditions, while the middle \( FI \) conditions showed a more expected behaviour. The overall result is some kind of U-shaped error curve. This differed from the Experiment 1, but the difference with the first experiment only occurred in the low \( FI \) conditions. We are uncertain about the cause of this effect.

**Subjective Feedback**

In a post-experiment questionnaire (see Figure C.5 in Appendix C) all users indicated that some of the forces were too strong which is a similar result as in the first experiment. We also asked them to estimate the amount of different forces they could distinguish. They discriminated on average 3.8 different force strengths (SD = 0.92), which is higher than in the first experiment. We expect this to be a consequence from the fact that different shapes may be felt as different force amplitudes.

### 5.5 Experiment 3: Subjective Preference

In both previous experiments we found that above a certain \( FI \) value the performance of the user deteriorated significantly. What is more, nearly all users indicated that
5.5 Experiment 3: Subjective Preference

some of the forces during the experiment were too strong. In this third experiment we investigate how strong forces can be before users consider them too strong. The following hypothesis will be investigated in this experiment:

- The final step value found as too strong, will be in the same region as the $FI$ value which causes performance deterioration.

5.5.1 Apparatus and Procedure

For this experiment, we used the same apparatus and applied the same experimental procedure as in both Experiment 1 and 2. For the force shape and direction we applied $S=\sin[0,\pi], T=75\text{ ms}$ and $D=d_{lat}$ which were also present in the other experiments.

5.5.2 Participants

Thirteen male and two female unpaid volunteers served as participants in this experiment. Participants were selected among participants of the previous two experiments\(^1\), ranging in age from 22 to 31 with an average of 26. All participants except one were right-handed and used their dominant hand during the experiment.

5.5.3 Stimuli and Design

For this experiment, we applied an adaptive staircase method (Leek, 2001). This is a psychophysical method in which thresholds are adjusted up- or downward depending on if the user discriminated a difference or not. In our case we will ask the users if they considers the current force as too strong. Instead of using a simple up and down staircase, we will use staircases where one of the two answers has to be the same twice. This type of staircase will enhance the precision due to the fact that the user has to twice perform the discrimination correctly.

We simultaneously interleaved two staircases: a one-up two-down starting from a strong force of 2.0N and a two-up one-down starting from a weaker force (0.6N). This design ensures that the user approaches the threshold from two different sides. For the first four reversals\(^2\), the step size was set to 0.3N, thereafter a smaller step size

\(^1\)Five of the twenty participants of Experiments 1 and 2 were not able to participate in this experiment.

\(^2\)A reversal is counted as soon as the user changes the direction of the staircase; e.g. when after one or more answers decreasing the forces, the next answer increases the value again.
Investigating Force Feedback Magnitude Effects during Target Acquisition

for resolution of 0.1N has been taken for the next six reversals. Hence, the experiment ended after a total of ten reversals.

The user had to perform five selections after which we asked the question “Do the forces complicate the execution of the selection task in any way?” The user had to answer ‘yes’ or ‘no’ using the keyboard with the non-dominant hand and depending on the answer and staircase, the force strength was changed accordingly. Each participant performed the experiment in one session lasting about 15 minutes depending on the time necessary to complete the 10 reversals. We logged the current strength value, step size and amount of reversals so far.

5.5.4 Results

For each participant, the mean force strength value for both staircase methods of the adaptive procedure was calculated by averaging the force strengths of the last 6 resolution reversals. Averaging these values we found that the one-up two-down staircase (started at 2.0N) had an average of 1.31N (sd = 0.5) and the two-up one down staircase (coming from 0.6N) an average of 1.04N (sd = 0.44), which results in a overall average of 1.175N (sd = 0.47) ($F(1,56.10)=56.10$).

This value is lower than both $FI$ conditions which we found in the Experiment 1 and 2 to deteriorate the trial completion time significantly, respectively 1.4 and 1.6N. We did see a rather high standard deviation which is due to subjective flavour of the participants with as lowest value 0.338N ($F(1,16.14)=16.14$) and as highest value 2.28N ($F(1,108.86)=108.86$).

5.6 Investigation of the Device Influence

Nowadays, increasingly more commercial haptic devices are becoming available. Although most of these devices are intended for similar purposes, the different devices have different physical constructions and physical parameters (e.g. friction and inertia) (Hayward and Astley, 1996). A force calculated as a result of a simulation, may hence be felt different on different devices, at its turn having implications on the performance or the satisfaction of the user.

In the previous experiments, we investigated the fundamental question what kind of forces may or may not be applied in order to support or at least not disturb the user.

We translated the question to the native language of the users, to make sure no language confusion might have occurred.
5.6 Investigation of the Device Influence

The applied forces were short 'bumps' with a given direction, amplitude and duration. Taking the effect of different device parameters into account however, short force bumps optimal for one device may have a completely different outcome on another device.

In this section, we compare two haptic devices during a target acquisition experiment in which different magnitudes of force feedback are applied. The haptic devices being compared are a PHANToM Premium 1.0 and a Novint Falcon, respectively an expensive and a cheap device with low and high damping (friction).

In the remainder of this section we discuss and deduce the main parameters of the PHANToM and the Falcon that are of an influence in this experiment: mass and damping. Afterwards we perform an objective and subjective experiment, similar to the previous experiments in this chapter.

5.6.1 PHANToM versus Falcon

Both devices are impedance controlled devices (Wen et al., 2008), measuring the user's movement as input and producing a force at the output. On the one hand, the PHANToM 1.0 haptic device, sold by Sensable Technologies is considered a high-end device with prices starting from 24,250 dollar (price from early 2009). We used a model with stylus and gimbal encoders. On the other hand, the Falcon haptic device, sold by Novint is a very low cost haptic device (180 dollar, early 2009) focusing on the consumer market.

The PHANToM premium 1.0 has a workspace of about $25 \times 18 \times 12$ cm (W×H×D). The Falcon’s workspace is somewhat smaller with $10 \times 10 \times 10$ cm. According to the specifications, the maximum applicable force for the PHANToM is 8.5N; the Falcon should be able to produce 10N.

Besides these general specifications, which can be found from the data sheets provided by the manufacturer, for our experiments we would like to know parameters such as the damping (mainly caused by friction) and inertia (apparent mass at the device’s handle). The result of an applied force bump can be different when a device has a higher inertia or a higher friction.

To achieve this, we adopted parametric identification of the transfer function in the frequency domain, as described by Pintelon et al. (1994). For both devices random forces were applied at the motors (input), resulting in a white noise spectrum in the frequency domain. While the device was in open loop (not held by a user), we measured the position of the end-effector (output).
The transfer functions (output/input) in the frequency domain for both devices are given in Figure 5.10. For clarity purposes, the phase diagrams have been omitted, as we don’t use them during the identification process.

![Transfer function of the PHANToM.](image)

![Transfer function of the Falcon.](image)

**Figure 5.10:** Transfer functions according to the x-axis for both devices.

Given that the general transfer function is given by \( \frac{1}{m s^2 + b s} \), we know

\[
Y_{(db)} = \left| 20 \log\left( \frac{1}{m s^2 + b s} \right) \right|
\]

where \( m \) can be identified as the mass, \( b \) as the damping, and \( s \) as the Laplace transform parameter. Two poles can be found, resulting in a transfer function that has a slope of \(-20 \text{db/dec}\) starting from 0Hz, and a slope of \(-40 \text{db/dec}\) after the second pole. Figure 5.10b is a good example for identifying both theoretical lines.

Elaborating the formula towards \( f(s = j\omega = 2\pi jf) \), and solving for \( m \) and \( b \) gives us:

\[
m = \left| \frac{-1}{(2\pi f)^2 \cdot 10^{-\frac{Y_{db}}{20}}} \right|
\]

With \( f \) the frequency (which must be high enough to ignore the influence of the damping), and \( Y \) the amplitude read on the graph (in db).
Table 5.2: Identified parameters (mass \(m\) and damping \(b\)) of both the PHANToM and the Falcon.

\[
\begin{array}{ccc}
\text{Parameter} & \text{PHANToM} & \text{Falcon} \\
\hline
m_x & 0.06 & 0.17 \\
b_x & 0.51 & 8.20 \\
m_y & * & * \\
b_y & * & * \\
m_z & 0.41 & 0.23 \\
b_z & * & 9.28 \\
\end{array}
\]

With \(f\) a frequency which is small enough to minimise the influence of \(m\). \(Y\) is again the amplitude read on the graph in \(db\).

The results of the identification can be found in Table 5.2. The values indicated with an asterisk (*) could not be identified with sufficient accuracy. For both devices, unfortunately the transfer functions for the y-axis were too noisy to make a decent estimation. For the PHANToM, it appears that the damping (mainly caused by the friction) is small compared to the influence of the inertia. Therefore it is very difficult to make good estimations for this parameter. It may attract the attention that the mass of the PHANToM for the z-axis is significantly higher, but this may not be surprising because movements around y and z cause the motors to move. With the Falcon, the friction appears to be the main factor (which can be derived from the graphs), therefore the mass can only be found with less accuracy.

The most important conclusion for us, is the fact that the Falcon appears to have a higher damping compared to the PHANToM. We may expect that this will have an influence on the device’s behaviour. Since the inertia differs not so much, its influence will be less pronounced.

5.7 Experiment 4: Objective Device Comparison

In our previous experiments we found that for the PHANToM, from a certain amount of force (i.e. the force integral \((FI)\) as a combination of shape, amplitude and duration), significant performance deterioration could be measured. We could see that the
deterioration was a result of a stabilising action of the user, reducing the oscillation caused by the force bump. As mass and damping have influence on the oscillation, it may sound obvious that other device parameters may lead to other cutoff values. To investigate this, a similar experiment has been conducted comparing the PHANToM with the Falcon haptic devices.

The following hypothesis will be investigated in this experiment:

- The difference in physical parameters between the the PHANTOM and the Falcon will have an influence on the user’s performance.

### 5.7.1 Apparatus and Procedure

For this experiment, we used the same apparatus as described in Experiment 1 (see Section 5.3.2) except for the input device, besides the PHANToM premium 1.0, we added a Novint Falcon. For the Falcon we also used a “forcemeter”. We found that a multiplication factor of three had to be used to achieve correct output, and the grip required us to compensate the gravity with 0.6N upwards.
5.7 Experiment 4: Objective Device Comparison

With regard to the procedure we applied the same experimental procedure as described in the Experiment 1 (see Section 5.3.3), a simple multidirectional point-select task, as described in ISO 9241-9 (ISO, International Organisation for Standardisation, 1998).

5.7.2 Participants

Seventeen male and one female unpaid volunteers, ranging in age from 20 to 28 with an average of 22.7, served as participants in this experiment. They were recruited among computer science students and had not taken part in any of the previous experiments. All participants were right-handed and used their dominant hand during the experiment.

5.7.3 Independent Variables

The independent variables are almost exactly the same as in the force shape and duration experiment discussed in Section 5.4. A quick overview can be found in Table 5.1. The only difference is the addition of the input device (ID): the PHANToM premium 1.0 and the Novint Falcon.

5.7.4 Design

A mixed design was applied, a repeated measures within-participant design for all independent variables was used except for input device ID, which is a between-participant factor. The independent variables were: force integral FI (0.0, 9.55, 19.10, 28.65, 38.20, 47.75, 57.30, 66.85, 76.39, 85.94, and 95.49); and shape S (sin[0,π], T=75 ms; sqr; T=40 ms; sin[0,π]; T=110 ms; and sin[0,2π], T=75 ms). A fully crossed design resulted in 44 combinations of FI and S.

Each participant was randomly assigned to a device and performed the experiment in one session lasting about 25 minutes. This way nine participants were assigned to each device. The session consisted of five blocks with each block containing the 44 combinations (11 FIs and 4 Ss) repeated three times in a random order. For a total of 132 trials per block, this resulted in 660 trials per participant. Between each block, users were obliged to take a break of at least 15 seconds to minimise fatigue during the test. Before the experiment, participants were given all 44 conditions in random order to familiarise them with the task.

Finally, we logged the same parameters as in Experiment 1. As the hardware driver of the Falcon does not return the velocity as an output parameter, we calculated the
velocity through a moving average of 100 position samples, similar to the velocity estimations of the PHANToM (Çavuşoğlu et al., 2002).

5.7.5 Results

Trial Completion Time

Before discussing the shape $S$, force integral $FI$, and input device $ID$ we will investigate the learning effect. Repeated measures analysis of variance of the faultless trials, showed a main effect for Block ($F_{4,64} = 18.991, p < .0001$), post hoc comparisons showed that the learning effect continued throughout all the blocks: participants were still improving their performance, even in the last block. With regard to the input device $ID$ no interaction effect could be seen with Block ($F_{4,64} = .155, p = .960$), from which we can conclude that both devices had a similar learning curve. As the learning effect did not influence the device and was present in all blocks, we will continue our repeated measures analysis of variance including all blocks. Note that this learning effect is different than from our previous experiments, we believe that this difference was caused by the overall grade of user experience. The participants in this experiment were computer science students while the other experiments were performed by research colleagues who regularly take part in different kinds of user experiments.

Input device $ID$ ($F_{1,16} = .256, p = .620$) did not show a main effect. Therefore we can conclude that there is no significant difference between the devices. However, we saw that the overall trial completion time for the PHANToM was 1029 ms and for the Falcon 1064 ms. Probably, with more participants, this difference will become significant, but such a small difference can be argued to be of less practical use. The interactions $S \times ID$ ($F_{3,48} = .811, p = .494$) and $FI \times ID$ ($F_{3,48} = 1.07, p = .385$) were not significant and showed a very similar trend. The Falcon consequently performs slightly slower. Figure 5.12 illustrates how each device behaves according to the respective force integral values. Not only is the Falcon consequently slower but it appears that for the highest $FI$ conditions the Falcon has a less strong trial completion time deterioration compared to the PHANToM.

Similarly as in Experiment 2 (force shape and duration), shape $S$ ($F_{3,48} = .952, p = .423$) did not show a significant main effect and force integral $FI$ ($F_{10,160} = 8.0, p < .0001$) did show a significant main effect. Post hoc comparisons for $FI$ showed that above a certain force integral value ($FI=85.94$) the trial completion time deteriorated significantly ($p < .01$). This value, 85.94, is again one $FI$ condition higher than in our previous experiments which is probably caused by the learning effect seen in this experiment.
5.7 Experiment 4: Objective Device Comparison

Figure 5.12: Force integral values by input device. Note that the y-axis does not start at zero to better show the differences between FI conditions.

Figure 5.13: Force integral values by shape.
An important difference is the non-significant interaction effect $S \times FI$ ($F_{30,480} = .799, p = .769$), in Experiment 2 we found a significant interaction. Figure 5.13 shows how every shape is influenced by $FI$, the biggest difference with Figure 5.8 is that all shapes follow the same trend much stronger, this is also the reason why there is no interaction effect. We have no real explanation for this statistical difference. But with or without this interaction effect, most important to note is that all shapes, in both figures, are influenced similarly by the force integral.

**Velocity Analysis**

Although we found no statistical significant differences between both devices, it may be interesting to take a look at the velocity behaviour during a selection task. Figure 5.14a and Figure 5.14b depict a typical velocity profile respectively of the PHANToM and the Falcon. The uppermost graph contains the evolution of the lateral velocity over time, the bottommost graph the longitudinal velocity. As with the previous velocity analysis, we have to stress that these graphs are individual selection trials of individual users in the highest amplitude condition ($FI=95.49$). The entire population of all profiles obviously has a large variation. However, the selected graphs give a good representation of the general behaviour.

![Velocity Profile](image-url)

**Figure 5.14:** Velocity Profile of the PHANToM and Falcon. The topmost graph is the lateral velocity, the bottommost graph represents the longitudinal velocity.

For the PHANToM device we can see very similar results, as in Experiment 2 with regard to the velocity. In the uppermost graph in Figure 5.14a, we can see that the force bump causes an oscillation lateral to the movement direction. In the bottommost graph, we see that the ballistic movement is interrupted after the force bump, possibly to tackle the oscillation caused by the force feedback. A second but shorter ballistic
movement is initiated afterwards. Finally, the controlled movement brings the cursor in an accurate way to the target. It can be assumed that the majority of the trial completion time penalty with high forces is caused by the interruption of the ballistic movement.

When considering the velocity behaviour of the Falcon 5.14b, we see a similar effect, but the effect of the higher damping is clearly visible, as well. After the force bump, a more strongly damped oscillation occurs. Here again, the ballistic movement is interrupted, but in general in a less pronounced way. Keeping these figures in mind, may help us to explain the behaviour in Figure 5.12. At the no force condition ($FI=0$) the Falcon appears to be (insignificantly) slower than the PHANToM, which is not so surprising, because of the higher damping causing a lower longitudinal velocity (Average maximum longitudinal velocity of all trials for the PHANToM: $\bar{v}_{max} = 60.12$ mm/s; for the Falcon: $\bar{v}_{max} = 42.03$ mm/s). Alternatively, the high forces cause less oscillation, and hence require a less severe interruption of the ballistic movement phase, which at its turn explains the lower trial completion time deterioration in the higher force conditions ($FI=85.94$ and $FI=95.49$).

**Error Rates**

An error for this selection task was recorded when the user clicked the button without having the correct target underneath the pointer. The overall error rate for the entire experiment was 315 errors or 2.7% of the trials. The device had no significant effect on error rate although the 5% significant level was approached ($F_{1,8} = 4.26$, $p = .073$). The error rate for the PHANToM is 3.3% and the Falcon is 2.0%.

5.8 Experiment 5: Subjective Device Comparison

To further compare different haptic devices, we conducted a second experiment sounding out for the users’ subjective experience with respect to the different devices.

The following hypothesis will be investigated in this experiment:

- The final step value force found for the Falcon will be higher than for the PHANToM as the different physical parameters might have as a consequence that the forces are perceived differently.
5.8.1 Apparatus, Procedure and Participants

For this experiment, we used the same apparatus and applied the same experimental procedure as in Experiment 4 (see Section 5.7). The force bump had a lateral force direction with $S = \sin[0, \pi]$, $T = 75\, ms$, which was also present in the other experiments.

The participants from the previous objective device experiment immediately participated in this experiment, right after they had finished the objective device experiment. This resulted in a between-participants design with nine users per device.

5.8.2 Stimuli and Design

The same stimuli and design are used as in the previous subjective experiment, Experiment 3 discussed in Section 5.5. An adaptive staircase method was used in which the following question was asked: “Do the forces complicate the execution of the selection task in any way?”.

5.8.3 Results

Mean Force Strength

For each device, for each participant, the mean force strength value for both staircase methods of the adaptive procedure was calculated by averaging the force strengths of the last six resolution reversals.

Averaging these values for the PHANToM, we found that the one-up two-down staircase (started at 2.0N) had an average of 1.45N ($sd = 0.48$) and the two-up one down staircase (coming from 0.6N) an average of 1.05N ($sd = 0.35$), which results in an overall average of 1.25N ($sd = 0.41$) ($FI=59.68$).

For the Falcon we found that the one-up two-down staircase (started at 2.0N) had an average of 1.64N ($sd = 0.71$) and the two-up one down staircase (coming from 0.6N) an average of 1.17N ($sd = 0.60$), which results in an overall average of 1.41N ($sd = 0.66$) ($FI=67.32$).

Comparing the average values, the Falcon ends up with a higher value for the mean force strength than the PHANToM, respectively 1.41N and 1.25N. This can possibly be assigned to the fact that the damping of the Falcon is higher, which might have an influence on the perception of different forces. Another indicator confirming this idea is the fact that, on average, the Falcon participants had to answer the question 27% more often than the PHANToM participants before ending the experiment. This
could imply that forces are less distinguishable with the Falcon than with the PHANToM, in order to justify this finding, a force discrimination experiment comparing both devices would be necessary.

In comparison to Experiment 4, both devices have a value lower than the $F_I$ condition we found in the objective device comparison experiment, which is in correspondence with the findings of our first PHANToM-only experiments.

**Subjective User Feedback**

After both experiments we asked participants to take a post-experiment questionnaire (see Figure C.6 in Appendix C). We asked them if they liked the grip of the device they had used.

With regard to the PHANToM three out of nine did not like the stylus grip because they got tired. The six people which liked it commented that this was due to the similarity with a pen and therefore was judged as intuitive. Participants disliked the grip of the Falcon more than the PHANToM. Only two participants liked the grip, because they could hold it in slightly different ways. The other 7 which commented negatively did not only raise the fact that it was tiresome but also mentioned problems with the form factor. The sphere was too small, or they did not like a sphere as a grip as it did not feel so comfortable. Some also mentioned that the Falcon did not feel “smooth”, so some participants did feel that the device has a higher damping factor.

In our experiments we only focused on the physical parameters of the device and not the form factor, but during subjective feedback, many users did not like the grip of the Falcon while the PHANToM’s natural pen-like grip did find personal preference. Besides personal preference the grip design targets different muscles groups which are used during the handling of the device. This could induce different performance, perception and behaviour of the user.

### 5.9 Discussion and Implications

Our experiments lead to a guideline that is useful to improve the design of user interfaces containing force feedback. The large number of degrees of freedom does not allow us to easily cover all possibilities, but important degrees were covered by our experiments.

As could be expected, the shape, time and amplitude of a force bump have their influence on the user. As a guideline we express the “size” of a force bump by means
of the force integral \((FI)\), which can be visualised by the area below the force graph (see Figure 5.1). Even though different force shapes are statistically different in terms of movement direction change, movement error, movement variability, . . . , these force shapes tend to show similar behaviour according to the effect on the user’s completion time. We found that above a certain force integral value, the performance of the user significantly deteriorates. The two objective experiments (Experiment 1 and 2) with the PHANToM resulted in very similar values (66.85 and 76.39) which indicates that designing abstract force feedback with forces below these values will not damage the performance of the user. Based on these results, we conclude that the definite integral of the force function \((FI)\) can be used as a good approximation to predict the effect of a force bump on the user’s performance, as long as the values are within limits. Although, the limits of this guideline must be deduced in future experiments, they can be intuitively understood from the physical parameters of the human-device system. Very short (strong) bumps ultimately will lose their effect because of the inertia, while similarly, very long (but soft) bumps will disappear by the friction of the device or simply will be below the user’s just noticeable difference (Tan et al., 2007).

Considering the amount of errors, we saw that users made more errors at higher \(FI\) values. Important to notice is that those \(FI\) values were higher than the forces that caused performance deterioration. In the subjective experiment (Experiment 3) we observed that a certain amount of force feedback complicates the performance of the selection task. This value, however was lower than the values that cause a performance penalty, and by far lower than the values increasing the number of errors. We advice user interface designers integrating force feedback –used as an extra information channel (similar to Tactons)– not to exceed the force integral values we found in Experiment 1 and 2 (66.85 and 76.39), and when possible keep below the subjective user’s preference (\(FI=56.10\) or 1.175N).

As the device can play an important role in how the haptic feedback is perceived, we also investigated this degree of freedom (Experiment 4). Objectively measured, no significant difference was found with regard to trial completion time between both devices. Note that this is a similar conclusion as supported by previously conducted haptic device comparison experiments, comparing the PHANToM to other haptic devices (Yu and Brewster, 2002; Harders et al., 2006). But more interestingly, we did find that the damping played a role on how the force influenced the velocity profile of the users. The oscillation due to the force bump was dampened, causing a less pronounced interruption of the ballistic movement phase (see Figure 5.14).

The effect of the higher damping could explain the results from the subjective device comparison experiment (Experiment 5): the force amplitude that was considered as
5.9 Discussion and Implications

“not complicating the task” was higher for the Falcon than for the PHANToM. Due to the damping the users probably perceive the force in a different way and probably had more difficulties discriminating the differences between the forces. In this context, it would be interesting to investigate if the Falcon would behave differently in a force discrimination experiment than other haptic devices (Tan et al., 2007; Yang et al., 2008b).

Another interesting point of discussion are the results of the two force shape and duration experiments (Experiment 2 and 4). They show similar results and the device comparison experiment serves as a replication of the previous experiment, strengthening our idea that the force integral can be used as a guideline during the design of abstract force feedback.

In all experiments we could learn from velocity analysis, that for large force integral values the longitudinal velocity dropped detrimentally immediately after the force bump occurred. We believe that this involuntary halt is the main cause of the user’s trial completion time penalty, explaining in some respect the deterioration behaviour explained above. The $FI$ guideline however has its limitations, as for longer and hence less strong forces, or very short but strong spike forces, other parameters such as the user’s compensating ability or the device’s inertia and friction will start to play a more important role.

Our experiments were based on a selection task with a constant Fitts’ index of difficulty. However, as the velocity impact seems to play an important role, we believe that our findings can also be transferred to other types of interfaces in which motion is involved. Examples of interfaces which are mainly motion-oriented are crossing based interfaces (Accot and Zhai, 2002; Forlines and Balakrishnan, 2008) or gesture interfaces (Bau and Mackay, 2008).

With regard to the direction of the force feedback, we did not find any differences that should motivate for a certain choice, such that other arguments are probably more important to consider. For instance, the ideas of Ruiz et al. (2008), can be used to adjust the force direction according to the predicted direction constraint of a possible target. They are able to discriminate between an amplitude/stopping or directional/steering constraint in the first 70% of the movement. If the force direction would change depending on the constraint, the influence of the direction could be less troublesome for the movement and target constraint. For example with a directional/steering constraint a lateral force direction could make the user miss the target.

Finally, it is interesting to compute the $FI$ value of the force feedback used for the bubble cursor in Chapter 5. The $FI$ was 15.92, which is well within the values we found in this chapter. It would be interesting to explore how other ‘safe’ $FI$ values
would perform, maybe a higher value (i.e. stronger force) would make the users more aware of having reached the target and might improve the performance such that its addition might become significant.

5.10 Conclusion

We studied the effect of different magnitudes of force feedback on the user’s performance in a target acquisition task. In order to facilitate the design of such feedback we propose to use the definite integral ($FI$) as a combination of the force parameters: shape, duration and amplitude. From five multidirectional point-select experiments we can deduce the following conclusions: (1) above a certain force magnitude the user’s performance significantly deteriorates; (2) When considering other force shapes and durations we found that the force integral can be considered as a good guideline to predict the user’s performance; (3) Subjectively spoken, users reported complications for force magnitudes which are below the objective deterioration point; and finally, (4) the PHANToM and the Novint Falcon have no significant difference with regard to performance, but subjectively forces are perceived differently and the higher damping of the Novint Falcon influences the resulting effect of the force bump.

Summarising the results of our experiments, the value of this work is to provide user interface designers with a guideline to keep the calculated force integral (based upon the force shape, duration and amplitude) below the force integral values that objectively caused the performance penalty. By preference, it should be even advisable to keep the force values below the subjective threshold.

It is important to note that the results of this investigation do not imply that force feedback below the values found in these experiments is a priori useful. If and when force feedback can be applied, is still up to the designer to decide, and ideally it is investigated in application specific validation scenarios.
Conclusion

In this part of the dissertation we performed several formal experiments with regard to selection techniques in virtual environments and experimented with several ways of multimodal feedback during selection. In order to be able to place the selection techniques in the appropriate context, the most interesting ones are illustrated in the following figure. It shows an updated classification of the taxonomy based on selection metaphors. With regard to multimodal feedback we focused on how the feedback sub-task could be improved.

In the very first experiment three selection techniques were compared and augmented with audio and force feedback. The 2D bubble cursor was found to be faster than the aperture technique and the virtual hand technique. Only, the 2D bubble cursor showed some potential problems in very dense environments. This can occur often on the image plane, where targets could also overlap each other. Therefore we decided to continue our research with 3D techniques and not image plane techniques. The added audio and force feedback showed significant improvements for the virtual hand but not for the 2D bubble cursor, where the visual feedback seemed dominant.

In a second series of experiments regarding selection techniques we investigated the factors density and occlusion. We designed two selection techniques, the 3D bubble cursor and the depth ray, to cope with dense scenes and we also augmented the tested selection techniques with a transparency function to be able to select fully occluded targets. In a first experiment we found that the transparency function allowed users to
select occluded targets with an overhead of about only one second. With regard to the
density we noticed that the depth ray was less influenced by the density and that the
3D bubble cursor was influenced by the density, according to the size of the Voronoi
region of the targets, which was an expected result. But we found that the target size
had an influence on the performance of the 3D bubble cursor while this should only
be the size of the Voronoi regions. This indicated that enhancing the visual feedback
alone is not sufficient and, therefore we investigated the addition of multimodal feed-
back. For audio feedback, earcons were used and for force feedback a short force
bump was used. Unfortunately, only a small non-significant improvement was found
which indicated that the multimodal feedback did not improve the understanding of
the size of the Voronoi regions. A positive note though, is that users did prefer the
multimodal feedback.

The force feedback, which we used in both these selection experiments, was designed
using either a small pilot test or a trial-and-error approach. This intrigued us in trying
to find a guideline which could help designers predict how the force feedback would
behave. We studied the effect of different magnitudes of force feedback on the user’s
performance in a target acquisition task. In order to facilitate the design of such
feedback we propose to use the definite integral (force integral $FI$) as a combination
of the force parameters: shape, duration and amplitude. From five multidirectional
point-select experiments we deduced that the force integral above a certain value, ir-
respective of the shape, causes the user to slow down significantly. We also found that
the force integral guideline is valid for both the PHANToM and the Novint Falcon
although the subjective feeling of the force is different and damping of the Novint
Falcon influences the resulting effect of the force bump.

We found that the $FI$ rule of thumb is a good approximation within bounds. In future
work, the extremes to which the prediction is valid, should be defined. Another
interesting question is whether different Fitts’ indexes of difficulty will have their
influence on the results of these experiments. Of course also other user interfaces, in
which this kind of force feedback can be beneficial, should be explored. It might also
be interesting to investigate if the grip of a haptic device plays a role in the user’s task
performance and force perception. We also argued that the physical parameters such
as the friction are important for the objective and subjective performance of the haptic
device. So, it is possible to try to improve these parameters in the code on application
level. Bernstein et al. (2005) presented a hybrid friction modelling approach which
lowers the friction of the device through a fluent combination of two well known
methods: approximate cancellation and variable-gain low bandwidth force feedback.
Such models were not incorporated in our experiments, but might provide valuable
solutions for some commercial haptic devices.
With regard to selection techniques for virtual environments a lot of formal experiments focus on abstract and simplified virtual worlds. An important next step could be to implement these techniques in applications working with full-fledged virtual worlds. In such an environment a formal evaluation is usually nearly impossible and other types of evaluation such as monitoring and interviews need to be applied.

In the second part of this dissertation we will focus on how we can help the designer and developer during the creation of virtual environments. More particularly we propose development techniques supporting the integration of selection techniques. Note that the selection techniques used in this part, more in particular in Chapter 3 and 4, were developed using the development process which will be used as the basis for our research in the second part of this dissertation.
Part II

Integration of the Selection Task in Virtual Environments
Introduction

Developing virtual environments remains a technical and time-consuming process. Usually, low-level programming languages are used to define the virtual world, the interaction of the user with virtual objects and the objects’ behaviours within the virtual environment. This is in particular true for 3D multimodal interfaces for virtual environments. The process of creating or selecting interaction techniques for such interfaces is not straightforward. A possible solution to explore is conceptual modelling, which aims to ease the development process of virtual environments by defining different aspects of the application on a higher level (Cuppens et al., 2005; Willans and Harrison, 2001; Kulas et al., 2004; De Boeck et al., 2008b; Coninx et al., 2009). This development process can be supported by advanced interactive tools. With respect to the support of virtual environment realisation through conceptual modelling, research has been focusing on scene and interaction development.

In the first part of this dissertation we performed formal experiments with selection techniques, but it is also important to be able to design and integrate selection techniques in applications. In this part of the dissertation we will focus on how model-based development can be augmented with semantic and context information such that it can aid in the implementation of selection techniques. We will start with an introduction about model-based development and discuss in particular the VR-DeMo (Coninx et al., 2006b) process, which has been created to design and prototype virtual environments. More specific, we will focus on the CoGenIVE tool which allows to design interaction using NiMMiT and at the same time prototype the application.

The integration of semantic information in virtual environment interaction is mostly still ad-hoc. In our case semantic information could for example indicate that a certain 3D object is a “House” or a “Hotel”. Usually, the framework incorporating semantic information requires that the semantic information is expressed in a specific manner. We introduce a model-based user interface approach which introduces se-
mantic information, represented using ontologies, during the modelling phase. This semantic information itself is created during the design of the virtual world. The approach we propose is system independent and enables the semantic information content to be chosen and adapted in complete freedom without considering the underlying framework. In Chapter 7 the addition of semantic information to selection will be discussed. Two different strategies will be explored. The first strategy will add semantic information to NiMMiT as a new data type which allows the designer to have more flexibility with regard to how can be indicated which object should be selectable or not. Another strategy which we will explore, is the automatic generation of a speech grammar based on semantic information. Speech interfaces are becoming more and more popular as a means to interact with virtual environments. However, the development and integration of these interfaces is usually still ad-hoc, especially the speech grammar creation of the speech interface is a process commonly performed by hand. Using semantic information makes certain that the grammar contains most words which the user will utter during interaction. This speech grammar can be used for several purposes, it could be used as a selection technique on its own (e.g. “Select the bus left of the bus shelter”) or as a filter to the selection (e.g. “Filter, only chairs”).

Apart from semantic information, specific to the application domain, context information could be of use during the design of virtual environments. In this dissertation, the context information will be used in case studies to allow more flexibility for the user with regard to the interaction with the application (e.g. possibility to change input devices or interaction techniques). Depending on the current context situation the user will have the interaction enabled, best situated for that situation. The context information has to be incorporated in our model-based development process such that the developer or designer can exploit this context information. We will propose a context system consisting of three different parts: Context Detection, Context Switching and Context Handling. Context detection is the process for detecting changes in context, while context switching brings the system in the new state that needs to be supported. Finally, context handling adapts the interaction possibilities to the current context. In our solution, we propose an approach for context detection and switching for virtual environments that is based on the ‘Event-Condition-Action’ paradigm. But before we discuss our context system in Chapter 9, in Chapter 8 we will elaborate on how the context handling will operate, with regard to the traditional model-based process, at the dialog level, in comparison to the task level, where it is traditionally performed (Clerckx, 2007).
Chapter 6

Model-Based User Interface Design

Contents

6.1 Introduction .......................................................... 127
6.2 VR-DeMo ............................................................... 129
6.3 NiMMiT ................................................................. 134
6.4 Tool Support: CoGenIVE .............................................. 138
6.5 Conclusion ............................................................. 141

6.1 Introduction

Model-Based User Interface Design (MBUID) intends to design the user interface from a higher level while traditionally programming code is used. Through the usage of high level models, the final user interface is incrementally derived, which should facilitate the development cycle of the interface.

Over the last years, the principles of model-based user interface design have been thoroughly investigated for traditional form-based desktop user interfaces (Vanderdonckt, 2005). The recent need for flexible development of contemporary interactive applications, even has raised the attention for this approach. Indeed, mobile applications (Mori et al., 2002), context-sensitive multi-device user interfaces (Calvary et al., 2003) and distributed and migratable user interfaces (Clerckx, 2007) are more complex and versatile. These application types are in need of easier ways to develop them than using a regular programming environment.
Most model-based processes have several properties in common (Calvary et al., 2003; Clerckx et al., 2004; Luyten et al., 2003; Mori et al., 2004). In most MBUID processes several models are involved expressing different perspectives on the application needs and its behaviour. In many cases this is a gradual process, converting one model into the next one, by means of an automatic transformation (through mapping rules) or by a manual adaptation by the user. For instance a typical process may start at the level of a task model, moving over several other models, towards the final user interface.

The design and implementation of an intuitive user interface in a 3D interactive virtual environment is a time consuming and hence expensive process. Especially the process of finding and creating suitable interaction paradigms is not straightforward. Interaction in an interactive virtual environment is often required to be multimodal, supporting gestures, direct manipulation, speech input, etc. More often than not, the creation of such a rich user interface results in an iterative process in which a solution is defined, implemented and evaluated multiple times. Another important factor for virtual environments applications is the virtual world itself. Without a virtual world, the user would have nothing to interact with, because of this the design of the virtual world is equally important and also has to be taken into account during the model-based design process.

Multimodality, including speech input, voice output and pen-based interaction, is a central topic in many research projects. However, most of the contemporary research activities in the area of MBUID still concentrate on 2D applications, in which interaction is done in two dimensions with traditional or pen-based input, even when working with 3D scenes or data. Only recently, the application of MBUID approaches has been broadened in its scope, and research initiatives in this area pay increased attention to the design of highly-interactive applications, such as virtual environments and mixed reality applications (Dubois et al., 2004). Although MBUID has proven its value in dialog and web-based interface generation, none of the existing approaches seems directly and entirely usable for the design of an interactive virtual environment. They all lack the possibility to describe direct manipulation (directly interacting with the virtual world instead of clicking buttons and menus) and metaphors using a single rich modality of interaction or multiple modalities. Some approaches such as ICO (Navarre et al., 2005) specifically focus on the interaction techniques, but do not describe an entire process. A good MBUID process should therefore consider the task model, the user interface widgets (such as dialogs and menus) as well as the description of possible interaction techniques for direct manipulation supporting multimodal input and output (such as speech, gestures and haptics).
In the next section, we describe how a model-based user interface design process can be applied to the design of an interactive virtual environment. More in particular we will introduce the VR-DeMo process developed in an IWT SBO project (IWT 030248) for basic strategic research. In the rest of this part of the dissertation we will extend models used in the VR-DeMo process with semantic and contextual knowledge.

6.2 VR-DeMo

The VR-DeMo project was carried out in our HCI lab at Hasselt University - EDM from 2003 - 2007 in collaboration with the VUB-WISE lab, and its results inspired further research. The main aim of the VR-DeMo project (Coninx et al., 2006b) is to ease the development process of virtual environments by high level definitions of as much aspects of the application as possible. The focus is on conceptual models and descriptions for the virtual world and the interaction, which are partly generated into source code and partly interpreted by the resulting applications, for an overview see Figure 6.1.

This process overview can be divided into two parts, the part on the left side (realised by VUB-WISE) serves as a scene generator and is also concerned with the behaviour of the objects within the virtual world. The right side (realised by our HCI lab), the interaction generator, defines the user interaction. Note that the interaction generator could also use scenes which already exist or were generated using another process. In the next sections we will give an overview of both parts.

6.2.1 Scene Generator

In a first step the designer creates an ontology, formulated in OWL (2009), containing domain knowledge specific to the virtual environment application. This domain knowledge is immediately used as the underlying representation formalism to model the virtual world. Two steps have to be performed, namely the specification step and the mapping step (see Figure 6.2). In the specification step the designers specify the virtual world at a high level using application domain knowledge and without taking implementation details into account. Two levels are considered namely the domain level and the world specification level. At the domain level, the designer will specify the different concepts of the domain of the virtual environment. The world specification level deals with the actual conceptual description of the virtual world, this is done by instantiating concepts described at the domain level. Finally, the mapping
step will specify how concepts and instances are represented visually in the virtual world.

This information is not only useful to express the structure of the virtual scene, but also to carry semantic information concerning the virtual environment application. In the VR-DeMo project a tool, OntoWorld (Bille et al., 2004), is being developed for this purpose by VUB-WISE. Therefore in this dissertation, we will not discuss how this semantic information can be attained, but focus on how it can be used. It is this semantic information that will be used by our approach presented in Chapter 7.

An excerpt of such an ontology, representing the semantic information, can be seen in Figure 6.3. It represents the domain level information: two concepts and two instances with their properties and relations. The domain concepts are represented as classes, properties and the possible relations between those concepts, while the mapping provides the instances of the classes, properties and their relations. For example a concept would be a hotel and its instance could be the Hilton.
Figure 6.2: The process used to model the virtual world. (Coninx et al., 2006b)

Figure 6.3: An excerpt of an ontology.
6.2.2 Interaction Generator

In this section we will give a brief overview of the model-based process used to design the interaction part of the virtual environment (De Boeck et al., 2006b; Coninx et al., 2009), the process is illustrated in Figure 6.4. This process is supported by CoGenIVE (Code Generation for Interactive Virtual Environments), a tool developed for the VR-DeMo project.

In a first and optional phase in our model-based user interface design process, the tasks that the user can perform in the application and the tasks that the computer must execute accordingly are modelled in the task model. For this purpose, we for example use the ConcurTaskTrees notation (Paterno, 1999) in CoGenIVE. This notation orders these tasks in a hierarchical tree with time dependencies. Based on temporal relationships between the tasks, a dialog model is derived from this task model using the algorithm of Clerckx et al. (2004). The notation allows to use different types of tasks: abstraction task, user task, interaction task, and application task. The result of this transformation is a collection of Enabled Task Sets (ETS) (Paterno, 1999), where an ETS groups all tasks that can be executed at a particular moment in time. These different ETSs then correspond to the states in the dialog model. In our process the task model is optional, which means that alternatively the designer directly can start modelling the application by creating a dialog model.

The dialog model plays a central role in the presented approach. Either directly created by the user, or imported from a task model, the dialog model is implemented as a state transition diagram with each state implicitly containing the tasks that can be performed in a given situation (expressed in a certain ETS). For instance, when the users have chosen to manipulate a given object (and thus are in a given ‘state’ of the application), they can only move or rotate an object, and are for instance unable to create a new object. Some tasks explicitly perform a state transition to another state, enabling other tasks.

To support menu-based and form-based interaction with the virtual environment in addition to direct manipulation, the dialog model is annotated with a presentation model, describing the user interface widgets (menus, dialogs, . . .). As in many MBUID approaches, the presentation model describes an abstract user interface, which means that it does not define how the interface will exactly look like, but only what functionality is provided through specific widgets. For the final user interface, hybrid 2D/3D user interface elements such as 2D menus or dialogs positioned in 3D are used (Coninx et al., 1997; Raymaekers and Coninx, 2001). The presentation model is described using VRXML, an XML-based user interface description language, suitable for 2D/3D hybrid menus (Cuppens et al., 2004).
Figure 6.4: The model-based user interface design process used in VR-DeMo and CoGenIVE. (De Boeck et al., 2006b)
From the point of view of interaction, the interaction description model as shown in the process overview (see Figure 6.4) is a central artefact in order to enhance a MBUID approach for use in the context of virtual environments. The interaction model, which is in fact an annotation of the dialog model, describes the direct manipulation and multimodal interaction from the user with the virtual objects. As explained earlier, most traditional MBUID approaches lack the support for direct manipulation and multimodal interaction. Therefore NiMMiT, Notation for MultiModal Interaction Techniques, has been developed. NiMMiT is developed to describe interaction techniques at a higher level than by writing code. An interaction technique can be seen as a complex ensemble of multimodal information that is merged and applied in order to execute a compound task which consists of several sub-tasks. A good example may be ‘touching an object to push it away’. NiMMiT is a graphical notation, inheriting the formalism of a state-chart in order to describe the (multimodal) interaction within the virtual environment. Furthermore, it also supports data flow which is important in the user interaction, as well. We will elaborate in more detail on NiMMiT in Section 6.3.

The interconnection of the presentation model and the interaction description with the dialog model is a manual process, in which the designer has to indicate which events, directly coming from a device or (indirectly) produced by a user interface element, correspond to a given task. A task then can be a simple atomic task, as described in the task model, or it can be a more complex action described using NiMMiT.

After annotating the dialog model, an application prototype is generated, which can be executed immediately. The prototype also contains the application code and some metadata containing the contents of the models. If necessary, a programming specialist can still tweak the code. This prototyping phase can be considered as an iterative process, which means that the interaction description model, the presentation model, and the final annotation of the dialog model, can be altered, while possible changes in the programming code are preserved.

After explaining NiMMiT in the following section, we will discuss CoGenIVE, the tool supporting the model-based development approach in Section 6.4.

### 6.3 NiMMiT

In this section we will discuss NiMMiT (Notation for MultiModal interaction Techniques), a diagram based notation intended to describe multimodal interaction between a human and a computer with the intention to automatically execute the designed diagrams. The purpose of NiMMiT is to be able to describe actions such as...
6.3 NiMMiT

“When the cursor hits an object, the object sticks to the cursor and follows the cursor until the user speaks the words -release object-.”

In the remainder of this section, we shortly describe the primitives of our notation and will discuss a simple example. For a more detailed description of NiMMiT, we refer to (Vanacken et al., 2006a; De Boeck et al., 2007). NiMMiT has also been extended in order to be able to perform interaction evaluation (Coninx et al., 2006a).

6.3.1 NiMMiT Primitives

In NiMMiT, interaction is considered event-driven: users initiate an (inter)action by their behaviour, which invokes events into the system. These events can be triggered by different modalities, such as speech recognition, an action with a pointing device, or a gesture. Interaction is also state-driven, which means that not in all cases the system responds to all events. The response to an event can bring the interaction in another state, responding to other events. Being data-driven is another important property of the notation. It is possible that data needs to be shared between several states of the interaction. For example, a sub-task of the interaction can provide data, which has to be used in a later phase of the interaction (e.g. touching an object to push it). Finally, an interaction technique can consist of several smaller building blocks, which can be considered as interaction techniques themselves. Therefore hierarchical reuse is possible within the notation.

When investigating the aforementioned considerations, we encountered other notations which can be divided in several categories. Some of them are state-driven based on state charts (Harel, 1987) or petri-nets (Petri, 1962), examples are ICO (Navarre et al., 2005), Interaction Object Graphs (Carr, 1997) and CHASM (Wingrave and Bowman, 2008)). Others, use a data flow architecture (InTml (Figueroa et al., 2002) and ICon (Dragicevic and Fekete, 2004)). All the example models focus on interaction but only ICO, InTml and CHASM have a similar goal as NiMMiT. They are also oriented towards interaction in (multimodal) VEs. A comparison of several existing solutions with NiMMiT has been discussed by De Boeck et al. (2006c).

None of the aforementioned notations entirely fulfills the above listed requirements, while being easy to learn and ultimately supporting automatic execution of the diagram. Figure 6.5 shows an example of a simple NiMMiT diagram, describing an interaction task that allows the user to select an object using ray casting (Liang and Green, 1994) after which the object can be moved around. Referring to the figure, we explain the basic building blocks of NiMMiT and provide some additional comments related to the diagram.
Figure 6.5: An example of a NiMMiT diagram representing an interaction technique in which the user can select and move an object.
6.3 NiMMiT

**State:** A state is depicted as a circle. The interaction technique starts in the start-state (double-bordered circle), and ends with the end-state (bold circle). A state defines a set of events to which the system responds.

**Event:** An event is generated by the framework, based upon the user’s input. A combination of events can be multimodal, containing actions such as speech recognition, gestures, pointing device events and button clicks. A single event or a specific combination always triggers the execution of a task chain. Events can be related to each other with 'and' or 'or' relations. Figure 6.5 shows an 'or'-relation between a button-press event and a speech event, which both can be used to release the selected object. As events coming from the user will never occur at exactly the same time, Nigay’s Melting Pot principle is used to resolve the synchronisation problem (Nigay and Coutaz, 1995; Coutaz et al., 1995). Note that these possible relations between events in combination with the Melting Pot principle allows designers to implement all possible relationships expressed in the CARE properties (De Boeck, 2007).

**Task Chain:** A task chain is a linear succession of tasks, which will be executed one after the other.

**Task:** A task is a basic building block of the actual execution of the interaction technique. Typically, tasks access or alter the internal state of the application. For example when running in a typical 3D environment, a task can be ‘collision detection’, ‘moving objects’, ‘playing audio feedback’, etc. Tasks can be predefined by the system, but designers can define their own custom tasks, as well. All tasks can have input and output ports, on which they receive or send parameters or result values. Input ports are required or optional, indicated by a square or circle input port respectively. The port’s colour, as well as a small letter inside, indicates its data type.

**Labels:** To share data between tasks in different task chains, or to store data for later reuse, we provide high level variables in the form of labels. The content of a label is maintained as long as the NiMMiT diagram is operational, and its scope is the entire diagram.

**State Transitions:** Finally, when a task chain has been executed completely, a state transition moves the diagram into the next state, note that this can also be the same state. A choice between multiple state transitions is also possible, based upon the value of a certain label.

**Hierarchical Task:** An interaction technique can hierarchically be reused as a task in a task chain since the interfaces of atomic tasks in a task chain and of in-
interaction techniques as a whole are similar. When such a task is activated, the execution of the current interaction technique is temporarily suspended waiting for the inner interaction technique to finish.

Referring to Figure 6.5, we recognise most NiMMiT primitives. The diagram starts in the ‘Start’-state, waiting for an ‘IDLE’-event, which by definition occurs always. We can hence assume that after starting the NiMMiT diagram, the first task chain (‘Selection’) is executed immediately. This task chain contains a hierarchical task that suspends the current diagram in order to perform a ray casting selection technique, described in another NiMMiT diagram. After the selection has been completed successfully, the result (obviously the selected object) is stored in the label ‘selectedobject’. Thereafter, we move on to the ‘Manipulation’-state, where three events are available. When the pointer device moves, the ‘Move Object’-task chain is executed, getting the current position of the pointer and moving the selected object. When either a button is pressed, or the speech command to release the object is recognised, the ‘Deselect’-task chain is executed, deselecting the selected object. After this has successfully been done, the ‘End’-state is activated, closing the current interaction technique.

With regard to the work presented in this part of the dissertation, we will primarily add extensions to the NiMMiT notation. A new data type and event type will make it possible to respectively use semantic and contextual knowledge inside NiMMiT diagrams. In Chapter 7 and 8 these additions will be introduced and discussed.

6.4 Tool Support: CoGenIVE

In this section, we briefly illustrate how the aforementioned process is supported by CoGenIVE (Code Generation for Interactive Virtual Environments) (De Boeck et al., 2008b). We emphasise how the tool is used throughout the process for the different models.

A task model can be the start of the process but is optional. As a consequence and because good tools exist for ConcurTaskTrees, CoGenIVE does not contain an editor for the task model. Instead a task model may be imported, in which case it is directly converted into a dialog model. Thus, the dialog model can be a result of a ConcurTaskTree conversion, or the designer may have chosen to design the dialog model from scratch, by dragging the states onto the canvas (top window pane of Figure 6.6).

Interactive virtual environments strongly rely on the user’s input. As a consequence we have to define how the user can interact with the system. In Figure 6.6, the window
6.4 Tool Supppport: CoGenIVE

The presentation model describes menus, dialog boxes and so on. For these common widgets, we want to fall back on familiar design activities. To design these widgets, CoGenIVE supports ‘drag and drop’ while filling out the requested properties. Typically, each user interface element and each item must have a name. Together they define the final event that will be fired when activation occurs. The presentation model in CoGenIVE is represented using an abstract visualisation, because the appearance of menus and dialog boxes may be slightly different depending on the rendering engine and/or the platform.
An editor was created for the interaction model which allows to design NiMMiT diagrams (see Figure 6.7). The editor allows the basic building blocks to be dragged onto the canvas. The tasks, forming a task chain, are picked from a list in the bottom right window pane. For each element, the properties can be altered: for instance the name of a state, or the event associated with a call of a task chain. At its turn each NiMMiT diagram is considered a new task, and hence appears in the bottom right pane for hierarchical reuse.

The NiMMiT editor ensures that the created diagrams are semantically and syntactically correct. For instance, it ensures that each state transition arrow ends in a new state, and that labels of a correct type are connected to the input ports of a task. This results in semantically correct meta-files that can be executed directly by the NiMMiT interpreter.

An application prototype can be generated when all models (dialog model, presentation model and interaction model) are created and mutually interconnected. This prototype contains automatically generated programming code, together with the (XML-based) description of the models. The generated programming code is an application based on a VR framework containing the NiMMiT interpreter. The application prototype is compiled immediately resulting in an executable prototype. When desired,
the code can be tweaked before compilation when specialised features need to be added. Changed code (within designated areas) is preserved when a new iteration of the process is done.

6.5 Conclusion

We introduced the general idea of a model-based user interface design process and more in particular discussed the model-based design process used in the VR-DeMo project.

The VR-DeMo project lifts the design of an interactive virtual environment to a higher level of abstraction. It is split up in two parts, a scene and interaction generator. In the scene generator, the virtual world is created using domain knowledge which is represented through ontologies. These ontologies not only define the virtual world, but they also provide semantic information which will be used to enhance the user’s interaction (e.g. selection). The interaction generator is based on a traditional model-based user interface design process with an addition of NiMMiT at the dialog level. NiMMiT is a notation which allows to design user interaction using a graphical notation.

In the next chapters we describe our extensions to NiMMiT such that it can be used with semantic and contextual knowledge. First, we will discuss the integration of semantic information in NiMMiT. Afterwards, we will add contextual knowledge to NiMMiT and also use NiMMiT to detect and switch between different contexts of use. In order to evaluate our additions we will use case studies.
Model-Based User Interface Design
Chapter 7

Augmenting Selection with Semantic Information

Contents

7.1 Introduction ........................................... 143
7.2 Related Work ............................................. 144
7.3 Adding Semantic Information to NiMMiT ................. 146
7.4 Using Semantic Relations Between Concepts .......... 155
7.5 Automatic Speech Grammar Generation ................. 158
7.6 Conclusion .............................................. 163

7.1 Introduction

Using the model-based development approach described in Chapter 6, based on conceptual modelling for the construction of the virtual world, semantic knowledge can become available to interaction designers. The modelling approach proposed in Section 6.2.1 could be used, but note that other modelling approaches are also possible (García-Rojas et al., 2008). The usage of semantic knowledge is increasing in popularity due to the semantic web. Although the semantic web itself has not yet been fully realised, the supporting technologies (RDF (2009), OWL (2009), SPARQL (2009), . . . ) are mature enough to be used for other applications. These technologies are nowadays finding their way to other applications such as traditional WIMP interfaces or virtual environments (Otto, 2005; McCorkle and Bryden, 2007; García-Rojas et al., 2008).
Here, semantic knowledge will mainly indicate what type of object is represented in the virtual environment and which relations it has with other objects in the virtual environment. This information can be used during the creation of interaction either by automatically generating actions or content based on this information or by taking into account this information during the interaction such that the interaction becomes less application-dependent. In this chapter we will introduce the usage of semantic information during conceptual modelling of interaction for virtual environments. In Sections 7.3 and 7.4 we will discuss our enhancements to NiMMiT in order to incorporate semantic information. We will discuss several case studies in order to proof the flexibility of the integration. In one of the case studies a selection task is designed which can only select objects from a certain semantic type.

With respect to the support of virtual environment realisation through conceptual modelling, research has been focusing on scene and interaction development. Speech interfaces, more specifically command languages using speech grammars, have found their way as an interaction technique, but facilitating their development has not yet received much attention up to now. Besides the addition of semantic information to NiMMiT, in Section 7.5 we will explain our approach of using conceptual modelling in order to automatically generate a speech grammar. This speech grammar can be used as a selection technique on its own, as filter to selection techniques or in any other combination with other interaction techniques. Speech grammar generation thus becomes part of the conceptual design phase of the virtual environment development. Before we will discuss our integration of semantic information, we will discuss related work.

7.2 Related Work

7.2.1 Semantic Integration with Virtual Environments

Several attempts have already been made to incorporate semantic information in some way or another with virtual environments. Some research initiatives focus on system integration while others focus on interaction.

Irawati et al. (2005, 2006) use semantic virtual environment information which is divided into domain-dependent and domain-independent information, and which is represented through ontologies. The authors claim that in their solution domain-dependent information is specific to the application while their domain-independent information remains static. However, in our opinion their domain-independent information depends on the domain of the application as it contains semantic information.
such as “wall” or “ball”. The exact coupling of the semantic information with the application framework is not discussed, but we expect it to be strictly coupled with the domain-independent information.

In order to incorporate semantic information Otto (2005) introduced a framework which combines the W3C Resource Description Framework (RDF) (2009), with his world model which has a similar structure as VRML or X3D. We will be using a similar notation which is based on RDF, namely OWL (2009). The solutions from Irawati et al. and Otto have as drawback that their semantic information is tied to the system (or the system tied to the semantic information), our approach will avoid this.

A first attempt at semantic interaction is the work of Kallmann and Thalmann (1999) in which a framework, designed for simplicity, was created for Smart Objects. These objects have pre-programmed interaction behaviours and are self-aware of how and where the user could interact with them. Abacý et al. (2005) integrate action semantics, expressed using rules, with these Smart Objects such that these actions can be used for action planning purposes by intelligent agents trying to achieve a goal. Easy integration with their previous work or in a design tool was left as future work.

Martínez (2004) proposed an augmentation of existing open file formats (VRML, X3D) with metadata such that all objects have unique identifiers. Using this technique Martinez was able to use them as semantic information combined with fuzzy logic for reference resolution, the resolving of the referent, which can be used during interaction. The main problem of fuzzy logic is that the system has to contain correct fuzzy sets. Another reference resolution approach has been proposed by Pfeiffer and Latoschik (2004), they incorporate several parameters such as a common ground, features, naming and spatial references which are similar to semantic information represented in ontologies.

Semantic reflection, a term introduced by Latoschik and Frölich (2007), represents a similar construction as the one we will present here. Current extensible, module based, architectures try to decouple specific application content from the internals of the framework. This is a challenging task due to the close coupling between data and control flow but distinct data representations between the various modules. For this problem, semantic reflection can form a solution. It is an extension to the existing reflection paradigm from object-oriented programming languages. Their approach is mainly oriented towards simplifying bidirectional communication for independent modules, such as graphics, physics and haptics, while our work focuses on how to integrate semantic information during interaction, and more specific during interaction modelling in NiMMiT.
Finally, Gutiérrez et al. (2005) present a system accompanied with a tool which allows for real-time configuration of multimodal virtual environments. Devices and interaction itself are represented using xml-based descriptors and coupled using a tool to define which device modality is activated by which interaction modality.

### 7.2.2 Semantic Speech Interfaces

Speech interfaces are often used in combination with direct manipulation because speech alleviates some of the disadvantages of direct manipulation such as the difficulty to express quantities (Cohen, 1992). Quickset (Cohen et al., 1997), FUSS (Gorniak and Roy, 2005) and Goubran et al. (1996) are examples of such direct manipulation systems in which speech is combined with a pen-based interface. For virtual environments some examples were presented by McGlashan (1995); Muller et al. (1998); Cernak and Sannier (2002); Kaiser et al. (2003). These are usually combined with some form of direct manipulation (e.g. gestures). All these speech interfaces use a speech grammar which is hand-made. Therefore depending on the application, a new speech grammar has to be created. The other frameworks, discussed in the previous section, usually do not incorporate speech or don’t generate the speech based on the semantic information.

Some frameworks do exist which incorporate ontological knowledge while speech processing. Usually, they try to detect the correct words being recognised based on this knowledge (Porzel and Gurevych, 2003) or for efficient control of the voice-based interaction (Kopsa et al., 2005).

Conti et al. (2006), created a semi-automatic tool which makes it possible for the application to understand the speech input of the user. During the coding process of the application the coder adds extra semantic tags to the code which are processed during compilation such that they can be used by the framework for understanding the user. This approach is very useful for speech input but it is hard to use these semantic tags for other goals, because a speech grammar is produced in a post-processing step. Some kind of middleware layer would have to interpret these semantic tags at runtime in order to use them during interaction.

### 7.3 Adding Semantic Information to NiMMiT

A problem that frequently arises when using NiMMiT is the fact that part of the interaction depends on the application being developed. This often results in custom-made tasks, hence increasing the time needed to develop the interaction. An example
7.3 Adding Semantic Information to NiMMiT

of this is selection: often some objects should be selectable, but others not. For example, in a room design application, the user should be able to select and move objects, such as tables and chairs. However, this is not the case for the walls and windows.

As these possibilities and restrictions can often be derived from semantic information, it is a logical step to also use this information within NiMMiT. In order to realise this, we will introduce a new data type in NiMMiT to be able to express the concepts of the semantic information (see Section 6.2.1). Of course, this means that we also will have to extend CoGenIVE to deal with this new data type.

Finally, we will realise some new predefined tasks, which use the semantic information or take it into account. These tasks have as advantage that they are application-independent and therefore NiMMiT diagrams using them can be reused.

7.3.1 A New Data Type: Concept

A task in NiMMiT can have several input or output ports, these ports have predefined data types: integer, double, boolean, string (red), position, rotation (green) and object (blue). A port can either be required to have input for the task to be executed or it can be defined as optional which will let the task execute without incoming data. In order to be able to represent semantic information we introduce a new data type: Concept. A concept is directly mapped to a concept of the ontology (class). As there is a clear relation to objects we gave the concept data type the same colour coding as an object (see Figure 7.1).

![Figure 7.1: The Concept data type.](image)

Besides introducing a new data type we need to be able to specify values if a constant label is being used, while other variable labels receive values during the interaction at runtime. To help the user choose the concept values a dialog has been added to CoGenIVE (see Figure 7.2). This dialog shows the ontology (see Section 6.2.1) in a treeview and through drag and drop values can be added and removed. We would also like to remark that the tree-structure of an ontology offers very high flexibility. It is for example possible to define all residences by taking the parent in the treeview
of all residences (Residence) while it would also be possible to define only certain types of residence, for example Hotel and Apartment.

![Concept dialog in CoGenIVE](image-url)

**Figure 7.2:** The concept dialog in CoGenIVE, with the current values of the constant label (left) and the concepts of the current project (right).

### 7.3.2 Use of the Concept Data Type

The concept data type was not defined in previous versions of NiMMiT. Therefore no other tasks exist which use this data type, and hence we will introduce several new predefined tasks such that the designer can use these during the modelling phase. Note that it is always possible to define a custom task if the designer needs to use the semantic information in another way.

Usually semantic information will be used during interaction, to check if an object which is being interacted or is going to be interacted with has a special feature for this specific type of interaction (e.g. haptic snapping or chairs under a table). Therefore we decided that the following tasks would be sufficient: ‘GetObjects’ and ‘IsOfConcept’ (see Figure 7.3).

The ‘GetObjects’-task outputs objects of certain concept(s) in the ‘objects’ output port. It has a required input port ‘concepts’ which represents all concept(s) which are valid and an optional input port ‘objects’. When this port contains no input data the task will search for all object(s) of the given concept(s), otherwise it will filter
7.3 Adding Semantic Information to NiMMiT

Figure 7.3: The newly introduced predefined tasks.

the input object(s) for the given concept(s). In order to check if object(s) belong to the concept(s), the semantic world is queried for the concept(s) of the object(s). This query is performed using SPARQL queries which are sent to the OWL file (ontology) representing the virtual world and its concepts. In Section 7.5 this will be further clarified, see also Figures 7.8 and 7.10.

‘IsOfConcept’ has a similar purpose, it simply checks if object(s) are of a certain concept and puts the result in the boolean output port: ‘bool’.

7.3.3 Case Study

We will illustrate our approach to integrate semantic information during interaction modelling using two case studies. Afterwards we will discuss our findings.

Selecting Objects

In most virtual environments not all objects are dynamic and interactive. To make only a certain subset of all objects interactive, a lot of ad-hoc, hard-coded solutions exist in which code is written such that the application knows which objects are interactive. In most situations the user first needs to select objects to interact with. Therefore the easiest solution for the designer is to make only the interactive objects selectable.

In Figure 7.4a, a NiMMiT diagram is presented which only supports selection of objects which are of concept(s) represented by the ‘SELECTABLE’-label. The diagram works in the following manner, the ‘Select’-state in the diagram responds to two events: POINTINGDEVICE.MOVE and POINTINGDEVICE.BUTTON_PRESSED.
Augmenting Selection with Semantic Information

(a) With a constant label (‘SELECTABLE’) providing the selectable items.

(b) With a concept interaction technique input port (‘selectable’) providing the selectable items.

Figure 7.4: The selection task which only selects objects of certain concept(s).
When the pointing device fires a ‘MOVE’-event (i.e. it moves) the right-hand task chain will be invoked and all tasks within the chain are executed. First the task ‘UnhighlightObjects’ is executed. Next the collided objects are detected and semantically filtered. Finally, the ‘HighlightObjects’-task will highlight the remaining objects and store these objects in the ‘highlighted’-label. If the chain has been fully evaluated, the diagram returns to the ‘Start’-state. In order to select the highlighted objects the left-hand task chain should be executed. Therefore the user should press button 1 on the pointing device. Once the ‘SelectObjects’-task is executed the diagram gets to the ‘End’-state and the interaction technique finishes.

As shown in Section 7.3.2 the concept selector dialog can be used to indicate the values for the ‘SELECTABLE’-label. Independent of the design, it is easy to allow all objects to be selected by attributing the concept value ‘Thing’ to the ‘SELECTABLE’-label. Remember that it is also possible to pick a parent-concept instead of all children concepts individually. Also, note that this allows for the ontology to not be specifically designed for interaction. Compared to the approach of Irawati et al. (2006), they introduce concepts as ‘Interactive_Object’ from which all other interactive concepts are derived which would allow the system to search for only objects of the concept ‘Interactive_Object’. Our approach gives more flexibility as any concept(s) can be selected at runtime without having to belong to a certain type of objects which has been decided upon during the coding of the system.

A disadvantage of the above mentioned interaction technique, is that it is not hierarchically re-usable. In the case where we would like to use this task in another interaction technique the same concept(s) would be defined in the ‘SELECTABLE’-label. An easy solution is to give the interaction technique an input port which defines the concept(s) which are then passed onto the input port of the ‘GetObjects’-task. The resulting NiMMiT diagram, with an input port of type concept, can be seen in Figure 7.4b. This way the semantic selection technique can be used in all applications and thus is application-independent.

An important remark is that when multiple objects are highlighted, this task highlights those that belong to the correct concept(s). If we would like the task to be more restrictive or exclusive and only let it allow to complete successfully when all objects being collided with belong to the concept(s) indicated in the ‘selectable’-label, then we need to use the ‘IsOfConcept’-task and add a new state and task chain. These small differences of course depend on the application, the designer’s preference and which option is the least confusing for users.

For example, this NiMMiT diagram could perfectly be used in the dynamic selection system of Tse et al. (2007) where, depending on the user input, only objects with specific colours are selectable.
Driving Simulator

Another case study is a driving simulator in which a steering wheel (Logitech G25) is integrated. An overview of the setup can be seen in Figure 7.5. In this case study, we need to know the surface which we are driving on such that realistic force feedback can be given through the steering wheel. The objects to collide with and checks to determine which object, such as a road or grass, we are driving on, was hard-coded during the initial design of the driving interaction.

There are two possibilities which can be used to integrate the semantic information into the existing interaction diagram. A first option would be to integrate one of the predefined tasks we introduced earlier. The positive effect of these predefined tasks (e.g. ‘GetObjects’) is the fact that the designer would have to create less custom or scripted tasks, but the downside is that it would introduce more design complexity, as extra states and task chains are necessary to create a working schema. We have opted for integrating the semantic information in the existing custom tasks and removing the existing hard-coded code.
For readability, the schema in Figure 7.6 is a shortened version of the driving interaction. In our setup we are using a steering wheel which needs special drivers and therefore a first custom task ‘GetInputDeviceParameters’ is used to get the parameters from the device. The event firing the task chain is the ‘IDLE’-event, which is always fired meaning the task chain is always executed. The second task updates the physical vehicle model and receives an optional concept label ‘road type’ such that the physical model can take the road type into account. The third task checks what type of road we are driving on and the slope of the road. This information is also placed in the ‘road type’-label. Note that in the first iteration the values of the second task are empty and are filled out in the third task. The ‘UpdateWorld’-task updates the position of the car and finally the ‘SendHaptics’-task sends the haptics to the steering wheel according to the ‘road type’ we are driving on which has been determined earlier by the ‘GetWorldParameters’-task.

The integration of the concept values into the custom tasks makes the solution more tied to the current interaction diagram and the addition of new concepts would require coding. This unfortunately means that some of the semantic information is coupled to the system, although in a less strict way as only this part of the application is tied to the system. Another fact stimulating loose coupling is that custom tasks are application specific and still make use of the underlying system to address the semantic information.

Discussion

We have created two case studies with regard to semantic interaction. In both case studies we have shown that we prevent hard-coded semantic information into the interaction through the usage of the new concept data type. The semantic information being used is also system-independent, because it only has to contain a link to the virtual world. This prerequisite is guaranteed in our particular model-based design process by the conceptual modelling phase early in the development cycle.

In the first case study we provided solutions which use the newly introduced predefined tasks, while in the second case study it seemed better to add the semantic information to the custom tasks which were present. We could see there exists a trade-off between custom task complexity and state transition complexity. It is not straightforward to say which would be the best option, having more predefined tasks makes the design much more complex but tries to eliminate programming. We believe that it is better to minimise state transitions which depend on boolean-values as it is usually much easier replaced by a custom task performing the if-test and consequent task(s). Having too many states could cause a state explosion during the modelling process.
Figure 7.6: The driving interaction technique.
7.4 Using Semantic Relations Between Concepts

A possible solution to overcome state explosion could be to use preconditions for an interaction technique, such that interactions are only performed (at runtime) if the precondition is satisfied. For example, when trying to help the user during interaction with objects we add haptics to the (multimodal) interaction. If a designer would like a certain haptic effect to be present during interaction with certain objects belonging to certain concept(s), this haptic effect could be performed in a separate NiMMiT diagram. The NiMMiT diagram is always running, but has the precondition that it only is active if objects of a certain concept are currently selected. Another possibility is the approach used in CHASM (Wingrave and Bowman, 2005, 2008). Each different type of behaviour (a certain state depending on a value) has response relationships. If the behaviour changes value this is propagated to another behaviour which can now take into account the new value. In the situation of Section 7.3.3 one such a behaviour would be road type, another one would be haptics, and both would then be interrelated. Note though, CHASM is directly linked to programming the solution and predefines some programming structures and therefore comes very close to a visual programming language. This is something we try to avoid with NiMMiT, being a more conceptually oriented modelling notation.

7.4 Using Semantic Relations Between Concepts

During the conceptual modelling phase of interactive virtual environments it is also interesting to use relationships between concepts. For example, when user wants to place lampposts alongside a road in a certain virtual environment, one could try to use the relationship between the concepts lamppost and road (namely ‘alongside’). This relationship indicates that a lamppost resides alongside a road (wrt. the current application) and we could help the user in placing a lamppost alongside a road by giving multimodal feedback (e.g. force enabled snapping locations). Note that other relationships could be exploited in this context, for example there is a ‘on’ relation between the concepts lamppost and ground.

It is not straightforward to introduce relations during the interaction modelling such that the solution is generic. In this section, we will first introduce a straightforward approach and afterwards discuss problems which will have to be solved in the future to achieve an as generic as possible solution.

In a first approach we try to create a manipulation technique, that similar to the selection technique from Section 7.3.3, is application-independent. Generally, a designer currently employs two possible strategies to create a manipulation technique. One approach is to create a manipulation technique per set of concepts that need
different handling of the manipulation. Another possibility is to create one manipulation technique that changes its manipulation style depending on the concepts to be manipulated. The first possibility makes the designer move the enabling of the manipulation technique to a higher level (the dialog model of the model-based design approach) and might overcomplicate this. The second one does not contaminate the dialog model but might become unmanageable to design or maintain. We believe that different types of manipulation can be directly linked to the different relations, so therefore using these relations inside NiMMiT and the manipulation technique could form a more generic solution.

In Figure 7.7, an application-independent manipulation technique is depicted. The input port ‘selected’ contains the object which is currently selected and thus will be manipulated. This data is stored in the ‘selected’-label and will be used during manipulation. If the user moves the pointing device an event ‘MOVE’ will be fired and the corresponding task chain (‘Manipulate’) will be executed. The first task, ‘GetPointerPosOriDelta’ returns the delta movement which will be used in a new task ‘MoveUsingRelations’. This task will check the semantic world for possible relations between the currently selected object and other objects and execute the corresponding relations. These relations can be defined through programming code or LUA scripting. Using this new task ‘MoveUsingRelations’ it is possible to use one manipulation task which autonomously implements the manipulation by moving the possible manipulation to the relations. With good tool support this will help the designer not to forget about certain possible relationships between concepts which are important for the application. Finally, if the user decides to stop manipulating the object, a button press will fire another task chain (‘Drop Object’) which drops (deselects) the object and ends the manipulation technique.

During the design of the relations, the designer can use the hierarchical representation of an ontology. The designer can first implement a certain relation, which can hold between all the concepts in the hierarchy for a parent concept. If a specialisation is needed the designers needs to implement it for a certain child concept. For example, consider the following hierarchy: Object - Furniture - Table - (RegularTable, DesignTable, . . .). The designer can first implement the relation ‘on’ for Furniture and use that design for the relation for a Table and if necessary further specify it for other types of tables, such as DesignTable which might have a completely different structure than a general table represented by the parent concept. Note that a general table will be represented by a RegularTable because of the fact that we can not specify this in a property of Table as RegularTable or DesignTable could also have children.
7.4 Using Semantic Relations Between Concepts

We have not yet formalised completely how we will define relations to use them during manipulations, in the following section we will give an overview of issues which we are investigating.

![Diagram](image)

**Figure 7.7:** A semantic manipulation technique.

7.4.1 Usage Problems

Earlier in this chapter we successfully introduced concepts during interaction modelling inside our model-based design approach. But the introduction of relations, during interaction modelling, brings out some points for discussion.

An ideal relation definition would be generic to such a level that it is application-independent and can be used in any newly designed application. For example defining the relation ‘onTopOf’ might seem simple at first, to constrain an object such that it remains on top of another object. But what is the top of an object and how do we define this? Do we always want that ‘onTopOf’ implies that an object remains on top of another object? Is it also possible that this relationship can end causing the object not to be on top of another object anymore? This also brings us to the problem when do we want a relationship to exist? All these questions being raised demonstrate that
it is difficult to implement a relationship in such a way that it becomes application-independent.

Imagine that we would succeed in realising a mechanism to define a relation in an application-independent manner. Then we would also have to take into account several other relationships which might exist at the same time, and how they would work together. In some cases we could end up in a local minimum where one constraint cannot be completely satisfied, because another limitation strives for another optimal value.

A problem with not being able to define relations application-independent might induce a complex web of relation definitions. If a certain relation has different meanings between concepts from the same application, this would oblige the designer to define the relation on a concept-relation-concept level instead of only the relation level. Of course without extensive testing in the form of case studies this is hard to foresee.

Besides using relations for manipulation techniques they could also be used for behaviour modelling, such that objects can behave autonomously according to the existing relations similar to the work of Lugrin and Cavazza (2007). In this work action representations, grounding and common sense are used to make a knowledge based system work together with computer graphics systems.

Semantic information can serve several purposes, besides the addition to NiMMiT, the following section will propose an approach to automatically generate a speech grammar based on the semantic information.

### 7.5 Automatic Speech Grammar Generation

Speech interfaces are increasingly being used in virtual environments since this way of interacting allows for more flexible and natural forms of interaction within a virtual environment. These interfaces have to recognise what a user utters and have to interpret these utterances in order to perform an action. As speech recognition is still limited, speech grammars are designed such that they can be used in a specific application with limited task domains, such as banking and travel services (Cohen, 1992). Given the fact that the system can recognise user utterances expressed in a restricted language, the user input needs to be interpreted to determine its effect on the virtual world. Therefore the system has to perform reference resolution for which the required knowledge can be divided into ontological, linguistic and contextual knowledge (McGlashan, 1995). In our case ontological and linguistic knowledge
7.5 Automatic Speech Grammar Generation

will be generated by the designer when creating virtual environments using conceptual modelling. Contextual knowledge, on the other hand, is gathered at runtime by the user when interacting with the application. Therefore contextual knowledge can be used as an addition to ontological and linguistic knowledge when building the speech grammar, but it is not a necessity and it is currently not considered in our approach.

In this section we will discuss our general process used to generate a speech grammar based on semantic information. The grammar will have a focus towards system commands which can be performed on certain objects identified through speech. Note that the selection task is such a task, another example would be the delete task. In order to show the flexibility, we also provide the possibility to ask the virtual environment application the value of a certain property of an object.

7.5.1 Process

Our approach to incorporate speech grammar generation in the conceptual design phase of the virtual environment consists of several steps: (1) the virtual world is modelled conceptually by which semantic information is generated (see Section 6.2.1); (2) the semantic information is used to automatically generate a speech grammar; (3) this speech grammar is further annotated with synonyms using a lexical database of English: WordNet (2009). After generation, the speech grammar contains all pronounceable utterances specific to the virtual environment application. When spoken, these utterances still have to be resolved to an interaction or command. In order to interpret the user’s speech we need to perform reference resolution, in our case such an utterance will be translated to a query (SPARQL) which can be resolved using the semantic information generated earlier in OWL. The resulting information consists of the names of those virtual objects that satisfy the query, and these can consequently be passed on as input for an interaction in the virtual environment, such as selecting an object. An overview of the process can be found in Figure 7.8.

7.5.2 Generating the Speech Grammar

The process which generates the speech grammar receives as input the semantic information which has been generated during the conceptual modelling phase of the virtual world. The semantic information is represented as an ontology in OWL format, in Figure 6.3 an excerpt is illustrated. The generation of the speech grammar consists of several steps, each step uses different data of the ontology to finally become a pronounceable grammar.
The resulting speech grammar has the following structure:

- \(<\text{command}>\) \(<\text{query}>\)
- \(<\text{o}>\) What/How/Which is \(<\text{o}>\) [data-property] \(<\text{o}>\) of \(<\text{o}>\) \(<\text{query}>\)
- \(<\text{query}>\)
  * \(<\text{o}>\) all \(<\text{o}>\) [concept]/[instance]
  * [concept] [object relation] [concept]/[instance]
  * [data-property-value]* [concept]

Here \(<\text{command}>\) stands for a command for the application (e.g. “select”), \(<\text{o}>\) means optional and \(<\text{query}>\) stands for the part which indicates the object the user refers to, this part needs to be resolved in a later stage. In order to show the possibilities we added one grammar rule which could be used in dialog systems, namely a question which can be asked to the application. All items between “[[]” are types of semantic information modelled in the ontology. [concept] Stands for a concept or class in the ontology (e.g. “hotel”), [instance] is an instance of such a concept (e.g. “Hilton”), [object relation] interprets as a relation between concepts (e.g. “left of”), [data-property] is a data property of a concept (e.g. “colour”) and finally [data-property-value] is the instance of a data property (e.g. for colour “red”) and which can also be repeated (indicated by “*”). In Figure 7.9 a part of the final resulting
7.5 Automatic Speech Grammar Generation

speech grammar for Microsoft Speech SDK 5.1 is illustrated, it has been created from the ontology in Figure 6.3. The fully generated speech grammar can be found in Appendix B.

For the structure of the speech grammar we based ourselves on other speech grammars used in related work and on a Wizard of Oz experiment performed by Corradini and Cohen (2002). We adopted an \(<\text{action}\>\ <\text{object}\>)\ structure (Clay and Wilhelms, 1996; Sharma et al., 2000), which is respectively command and query. During each generation step we add WordNet synonyms to the speech grammar such that, if the user uses such a synonym, the system would still recognise the correct concept. Note that we could have added other “-nyms” to the speech grammar such as hyper-, hypo-, holo- or meronyms, but we decided not to add any such extra words. First of all, it would enlarge the speech grammar and could make recognition worse. Secondly, if the designer would like the user to be able to use words which belong to any of those categories, he should have incorporated this during the conceptual modelling phase of the virtual world. For example a bike has a steering wheel or peddles. These are meronyms (X is a meronym of Y if Xs are parts of Y(s), or X is a meronym of Y if Xs are members of Y(s)) but if not added by the designer they are probably of no use to the application, i.e. they cannot be manipulated or selected, and should therefore not be included.

7.5.3 Reference Resolution

If a speech command, uttered by the user, is recognised and thus is a valid construction in the grammar, then the final step has to be performed: reference resolution. Because our speech grammar has a consistent structure we can use this structure to perform the reference resolution. As in our case the semantic information is repre-
Augmenting Selection with Semantic Information

sent in an OWL ontology, we need some mechanism to query this ontology. We are using SPARQL (2009), a query language for RDF, which can be used to pose queries at OWL ontologies. In order to easily perform these queries we use the w2p library (Vanderhulst et al., 2007; W2P, 2009).

The <query> part (see Section 7.5.2) of the speech grammar is the only part transformed to a query; this transformation is relatively straightforward and consistent for all types of queries. In Figure 7.10 several transformations from the <query> part can be seen, other transformations are performed similarly.

The answer of the query is the result of the reference resolution and is used as input for the command which is the only remaining unprocessed part of the spoken utterance. Note that at the moment we do not incorporate contextual knowledge, meaning that we do not take into account the history of previous utterances or the position of the user in the virtual environment. Adding such knowledge could filter or interpret the results of the query even further. For example if similar virtual objects are found at different locations, the object at the same location, where the user is, will more likely be the object referenced to.

![Figure 7.10: SPARQL queries: (a) query structure: [concept] [object relation] [concept]/[instance] (b) example of (a): “bench left of fountain”. (c) Query structure: [data-property-value]* [concept] (d) example of (c): “Brown obelisk”.

### 7.5.4 Case Study: a “City Park Designer”

In order to assess the validity of our approach for generating a speech grammar, we augmented a conceptually modelled “City Park Designer” application (Coninx et al.,
2006b) with speech input as an interaction technique, and generated the speech grammar automatically. Using the interactive “City Park Designer”, it is possible to design the park by positioning objects, such as buildings and statues. Furthermore, the designer can simulate behaviours, such as moving cars and shadows projected by the changing position of the sun. The park designer can use the automatically generated speech input commands to select/move/delete/add an object during the design of the city park. An example of such a command would be “select the bus left of the bus shelter” to refer to the object to be selected in the virtual scene. If the bus is of a specific brand, the user would also have been able to use this brand name. Besides using interspatial relations, objects and object-names, the user could also ask the application for the value of a data property of a certain object. The question “What is the colour of the bench left of the fountain?” is answered by the application with: “The colour is White”. Figure 7.11 gives an impression of the virtual city park. Informal validation tests confirm our approach, because they reveal that users mainly use words represented in the semantic information, and therefore also words which are incorporated in the automatically generated speech grammar. Throughout this chapter most of the examples used to illustrate the approach were generated from this case study.

Figure 7.11: Impression of the virtual city park.

7.6 Conclusion

We introduced our integration of semantic information during the conceptual modelling phase of virtual environments. The semantic information is represented using
an OWL ontology and queried using SPARQL for reference resolution. We incorporated semantic information into our interaction description language: NiMMiT. This integration was realised through the introduction of a new data type called concept, which represents a concept known by the system through ontology data. In order to ease the use of this newly introduced approach some predefined tasks using this new data type have been added. To illustrate and validate our approach we presented two case studies, namely selecting certain objects and a driving simulator. These case studies showed that it is easy for designers and users to integrate semantic information independently from the system in existing and new applications but they also unveiled new problems such as possible state explosions if the designers try to model everything.

Next, we discussed our attempt at introducing relationships defined in the semantic information of the interactive virtual environment application. A generic manipulation technique using relationships was introduced and remaining problems we currently encounter were discussed.

Finally, we have introduced an approach to automatically generate a speech grammar using semantic information. We discussed the structure of the speech grammar and the conversion of this structure to the necessary SPARQL queries. The approach was validated by a case study in which we augmented a conceptually modelled “City Park Designer” with our approach. From this case study we could conclude that users used mainly words contained by the speech grammar during interaction with the application.

We feel that the addition of semantic information has shown potential in both approaches and is worth considering to be used in other aspects of virtual environments. Besides adding semantic information to the model-based interface design process, it is also common that other types of information such as contextual knowledge are added. In the next two chapters we will investigate how context can be added to our model-based design process.
Chapter 8

Extending a dialog model with Contextual Knowledge

Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1 Introduction</td>
<td>165</td>
</tr>
<tr>
<td>8.2 Context and NiMMiT</td>
<td>167</td>
</tr>
<tr>
<td>8.3 Case Study</td>
<td>173</td>
</tr>
<tr>
<td>8.4 Conclusion</td>
<td>177</td>
</tr>
</tbody>
</table>

8.1 Introduction

The previous chapter introduced the usage of semantic information during interaction with virtual environments, designed using a MBUID process. In this chapter and the next one, we will explore context in the virtual environment application domain. We will use context to enhance the user’s flexibility with regard to interaction techniques and input devices used by the application. In some situations, the interaction with an application may vary depending on the context of use in which it is applied. This is usually true for mobile applications that can be used on different platforms or in a variety of situations causing some features to be available only in some situations. In the domain of interactive virtual environments, different contexts of use may have an influence on the available interaction.

The context in a virtual environment is often defined by the available input and output devices, external parameters such as the experience level of the user, whether or not there are collaborative partners in the environment, or even the pose of the user.
Extending a dialog model with Contextual Knowledge

(sitting or standing). Without the support for different contexts of use directly from within the MBUID process, the integration may often result in a lot of ad-hoc code, which is difficult and expensive to maintain.

Although no consent exists on the definition of ‘Context’ (Dey, 2000), in this work we consider context as an artefact influenced by different factors (Preuveneers et al., 2004) such as the user, the environment, platform and others. A context is then a multidimensional vector of these factors (sometimes also called observables (Coutaz and Rey, 2002)). For convenience in the user interface of the proposed tool support, we will call these factors ‘Context Units’ and the values available for each unit ‘Unit Values’.

Dey’s definition of context (Dey, 2000) states that context is only relevant when it has an influence on the user’s task. Given the HCI perspective of Dey’s definition it is applicable for virtual environments in the scope of this dissertation. Concerning the influence of context on the interaction with the system we can distinguish several distinct levels (Clerckx et al., 2006).

Two of these are important for the remainder of this chapter:

- **Task Level**: context influences the tasks that are enabled in a certain state of the user interface. A change of context may imply a change of active tasks.

- **Dialog Level**: context influences which state is currently active in the dialog model. Thus, dialog level influence of context may cause a transition to another state of the user interface.

Context-awareness is usually introduced at the task level and not at the dialog level (Priebeau et al., 2001; Van den Bergh and Coninx, 2004; Paternò and Santoro, 2002; Clerckx et al., 2004). We will show that this might cause an explosion in the amount of dialog states in situations where context-aware multimodal interaction is used in one and the same task. We will also introduce an approach to switch between different tasks at the dialog level. Another reason to model context at the dialog level is the fact that from our experience, with the VR-DeMo process, the task model is less detailed and fine grain because we continue our interaction modelling using NiMMeT. More concretely in this chapter we propose an approach which attempts to integrate contextual knowledge in the dialog level where transitions are chosen upon context information. In the next chapter we will discuss the complete context system, as supported by our model-based process.

First, we will elaborate on our approach. Afterwards, validation of our approach through a case study will be discussed in Section 8.3. This case study allows the user...
to position some crates. The user can navigate through the environment and select, move or rotate these crates. How the interaction with the environment occurs depends on the setup the user is working with. While using a desktop computer, interaction is done by means of keyboard and mouse input but when the user stands in front of a large projection screen a tracking glove is used in combination with voice input in order to manipulate the scene.

8.2 Context and NiMMiT

In this section we briefly discuss previous results of research performed at the EDM in the area of model-based design (e.g. in mobile or multi-device developments (Clercckx, 2007)) that have inspired the approach we use here: (1) incorporating context in task and dialog modelling and (2) adding modality constraints to tasks. We discuss these matters in order to introduce a combination of these two approaches enabling context-aware selection of modalities used to perform exactly the same task. Afterwards, we use this combination to allow switching between different tasks. Here, these tasks will usually be different interaction techniques (e.g. switching between different selection techniques depending on the context).

8.2.1 Context in Task and Dialog Modelling

In a first step, we aim to use context information in order to select the appropriate modality to perform a certain task. Several approaches already incorporate context information at the task level (Pribeanu et al., 2001; Van den Bergh and Coninx, 2004; Paternò and Santoro, 2002; Clercckx et al., 2004). In the approach, of our colleagues, inspired by Pribeanu et al. (2001), the decision task (Clercckx et al., 2004) is defined as follows:

Definition 1. A decision task \( t \) denotes a junction in the task model (see Section 6.2.2) where each subtree describes the sub-tasks of \( t \) relevant to the execution of \( t \) according to the status of the context. The iconic representation of the decision task is \( D \).

Thus at runtime exactly one subtree of the decision task is active. Clercckx et al. (2004) discussed an algorithm to deduct a corresponding dialog model for each distinct context of use. After the automatic deduction of the dialog models, the designer makes connections between the dialog models to describe when exactly a change of context can introduce a switch to another dialog model.
8.2.2 Modality Constraints in Task Modelling

In other previous work at the EDM research institute Clerckx et al. (2007) have introduced a way to link constraints to the leaf tasks in the task model to specify which modalities are the most desirable to perform the corresponding task. Therefore the designer has to select one or more modality categories per task and relate the selected categories with a CARE relation (see Section 2.3). This selected category enables a runtime selection of a suitable modality with respect to the available interaction techniques surrounding the user at a certain moment in time.

However this is not enough, we would like to take into account more information than the devices populating the virtual environment (application context) to select the appropriate modality. For example in our case study (which will be presented in more detail in Section 8.3) we have 2 different setups (external context) in which we would like to interact with the virtual environment and both setups require other modalities/devices to be used.

One way to overcome this problem is to use the approach we have discussed in Section 8.2.1. This is illustrated in Figure 8.1a. In this example task $t_2$ is divided into two distinct tasks $t_{2a}$ and $t_{2b}$. In this way the designer can attach distinct constraints, $m_1$ and $m_2$ to the two tasks. As a result at runtime the task that will be active is chosen with respect to the context status (as shown in the corresponding dialog model in Figure 8.1a).

The above described approach works well when just a few tasks require a context-aware selection of the appropriate modality. However when a lot of leaf tasks require a context-aware modality a lot of dialog models are generated and used to describe the same interaction flow. Suppose a task model has got $n$ leaf tasks where a context-
aware selection of the appropriate modality is desired and each task is divided into two tasks by means of a decision node. When the dialog models are extracted from the task specification, all possibilities of context statuses are taken into account resulting in \( \binom{n}{2} \) dialog models. It is obvious that \( n \) should not be that high to result in an impractical amount of dialog models. This is because the actual purpose of the decision task was to specify different tasks in different context statuses. However in this case, the tasks remain the same and for this situation we propose context at the dialog level as a more efficient approach. Note, however, that this way of working can still be combined with context modelling at the task level. This is for instance useful when really different interaction metaphors are offered to the user that do not rely on highly similar task chains. Usually such interaction metaphors are represented as leaf nodes in the task tree, and are modelled with separate NiMMiT diagrams. In Section 8.2.4 we will show how it is also possible to use the approach presented here to switch between different tasks, allowing to model context completely at the dialog level. The possibility to model context completely at the dialog level allows the designers themselves to decide when to stop modelling at the task level and when to switch to the dialog level, depending on where they feel most comfortable with.

A solution to overcome the above mentioned problem of an exploding number of dialog models is to combine the approach of making a distinction between tasks at the task level and the approach of taking care of context at the dialog level. Clerckx et al. (2006) showed how transitions in the dialog model can be executed by a change of context information. A combination of the two distinct approaches of context influence at the two levels can be seen as follows. Instead of having two distinct dialog models, we can merge these two together, and make only a distinction between where a difference is made by a context status. This is illustrated in Figure 8.1b. The two states containing \( t_1 \) are merged into one state in the same dialog model, but a choice is made which state will be reached by means of the context status. In this way the decision at the task level is modelled at the dialog level. In the next section we introduce this concept in the NiMMiT notation.

8.2.3 Context-aware Modality Selection at the Dialog Level

In Figure 8.2a an abstract NiMMiT diagram is presented. Here, in the ‘Start’-state several different events (modalities) could trigger the execution of ‘Taskchain1’. In our new approach we would like to be able to attach a certain context to a certain event or modality such that depending on this context only those events belonging to that context will be taken into account when evaluating a state in the diagram. If for example ‘EVENT1’ is meant for the expert-users of the application one can attach the
Extending a dialog model with Contextual Knowledge

Figure 8.2: (a) An abstract NiMMiT example. (b) The context view of an abstract NiMMiT Diagram. ‘EVENT1’ and ‘EVENT3’ were added to a specific context.

‘expert’-context to the arrow containing ‘EVENT1’, because here this event is fired by a device such as a spacemouse which is difficult to handle. Similarly ‘EVENT3’ could be used in the ‘beginner’-context which is coupled to an easier device such as a keyboard.

Adding this contextual knowledge to events transforms the view of the diagram depending onto which context of the diagram we are viewing. The resulting diagram containing the context arrow is shown in Figure 8.2b.

8.2.4 Context-aware Task Selection at the Dialog Level

The approach to introduce context at the dialog level and switch between different modalities for the same task, can also be used to switch between different tasks.

For example, the selection techniques evaluated in Part I are sometimes better suited for a certain context. In order to be able to exploit this, it needs to be possible to switch between different tasks. In Figure 8.3 a task model is presented in which the selection technique can be changed depending on the context. Transforming this task model to the corresponding dialog model will give a similar model as in Fig-
In order to avoid state explosion we can fully make use of our dialog level as it consists of three models: NiMMiT, state transition network and VRXML (see Figure 6.4).

**Figure 8.3:** Task model in which different selection tasks (bubble cursor and depth ray) can be chosen depending on the context.

In the previous section we used NiMMiT and context arrows to be able to differ between modalities depending on the context. The same approach can also be used to select between tasks depending on the context. NiMMiT makes it possible to re-use a NiMMiT diagram (task) using hierarchical tasks. These hierarchical tasks will represent the different context decisions. The decision task itself becomes a NiMMiT diagram containing ‘IDLE’-event arrows to all different context situations. These ‘IDLE’-events are then assigned to the correct context, allowing the correct ‘IDLE’-event to be enabled for the correct decision. It is this decision task that is then used in the dialog model, in our case the state transition network. Because of this mechanism we don’t need multiple state transition networks, one extra NiMMiT diagram is enough to support switching between tasks.

This approach, applied to task model from Figure 8.3 can be seen in Figure 8.4. Instead of having two dialog models (see Figure 8.4a), one is used containing the ‘Selection’ decision task (see Figure 8.4b). This ‘Selection’-task is represented through a NiMMiT diagram with ‘IDLE’-event arrows to the possible options (see Figure 8.4c). In order to switch to the correct decision depending on the context, these ‘IDLE’-event arrows are assigned to the bubble cursor or depth ray context (see Figure 8.4d).

This strategy is less intuitive to be performed by a designer. But if the designer is aware of this possibility, it actually is an easy way to switch between different tasks depending on the context.
Figure 8.4: (a) Two dialog models generated from the task model. (b) The merged dialog model, containing the ‘Selection’ decision task. (c) NiMMiT diagram without context arrows, representing the ‘Selection’ decision task. (d) NiMMiT diagram with context arrows, assigned to the corresponding context (bubble cursor or depth ray), representing the ‘Selection’ decision task.
In the following section we will discuss our case study in more detail and illustrate the context integration in a concrete example. Note that this example will not contain different tasks which are context-dependent, in Chapter 9 we will discuss a case study in which this approach is used.

8.3 Case Study

Figure 8.5: Setup of the case study: a wall projection combined with a tracked glove, speech and a desktop setup with mouse and keyboard.

8.3.1 Setup

As mentioned earlier we will illustrate our approach through a case study in which a simple scene can be manipulated. In the constructed virtual environment it is possible to select, move and rotate some crates onto a plane. To validate our context integration we created two setups for this application between which the user can switch at runtime. On one side we have a desktop environment in which the user can interact by means of a keyboard and a mouse, and on the other side we have a large wall
Extending a dialog model with Contextual Knowledge

projection in which interaction is done using a tracking glove and voice input. The complete setup is depicted in Figure 8.5.

8.3.2 Creation

The scene modelling application has been created with CoGenIVE (see Section 6.4). The process starts with the creation of a ConcurTaskTree (CTT) describing the different tasks that are available within the application (see Figure 8.6). In this case, some initialisation is done in the ‘Load’-task and consequently the ‘World Mode’-task becomes enabled. In this task, the user can navigate through the world and manipulate (select, move and rotate) the objects within the environment.

The next step is to define the leaf-tasks of the CTT. The application tasks can be mapped onto system tasks but we have experienced that the user interaction can better be expressed by means of a NiMMiT diagram. Since only the selection and manipulation tasks are context sensitive in this case study, we will focus on these tasks in the remainder of this chapter. More specific, we will use the select task to illustrate our approach. The NiMMiT diagram of the ‘Select’-task will be briefly explained in the remainder of this section and the next section will further clarify how context is integrated into the notation.

As shown in the diagrams in Figure 8.7 the ‘Start Select’-state responds to two events (KEYBOARD.MOVE and KEYBOARD.BUTTON_PRESSED.0) for the desktop setup (see Figure 8.7a) and two events (GLOVE.MOVE and SPEECH.SELECT) for the wall setup (see Figure 8.7b). The bottom part of both diagrams is the same and can be seen in Figure 8.8. When either the keyboard or the glove fires a ‘MOVE’-event

Figure 8.6: The ConcurTaskTree of the case study.
8.3 Case Study

(a) Events active in the Desktop setup.

(b) Events active in the Wall setup.

Figure 8.7: Active events in the Select interaction.

Figure 8.8: NiMMiT Diagram of the Select Interaction with the context arrows.
the right-hand task chain will be invoked and all tasks within the chain are executed: first the ‘UnhighlightObjects’-task is executed, then the newly collided crates are detected and finally the ‘HighlightObjects’-task will highlight the found objects and store these objects in the ‘selected’-label. If the chain has been fully evaluated, the diagram returns to the ‘Start Select’-state.

In order to select the highlighted objects the left-hand task chain should be executed. Therefore the user should press a key on the keyboard (in the desktop setup) or issue the speech-command (in the wall setup). Once the ‘SelectObjects’-task is executed the diagram gets to the ‘End Select’-state and the interaction technique finishes.

8.3.3 Context Integration

![NiMMiT Diagram of the Select Interaction in the Wall context.](image)

As indicated in the previous section, for each context a NiMMiT diagram is necessary resulting in $n$ nearly similar diagrams for each task ($n$ being the number of possible contexts of use). In order to solve this problem we use CoGenIVE to add context information to the NiMMiT diagrams. The possible contexts of use are pro-
vided through an XML-file as defined by Clerckx (2007) and loaded into CoGenIVE. Next, the diagrams in Figure 8.7 are merged, this results in Equivalence (Coutaz et al., 1995). But in our case we would like to enforce a certain modality in a specific context, therefore a context specification is added to each of the arrows. Within CoGenIVE, arrows with context specifications are automatically replaced by a context arrow as shown in Figure 8.8.

In order to interpret these context arrows, the NiMMiT interpreter in our runtime environment has been extended with a simple ContextManagementSystem to indicate the context and a ContextInterpreter to evaluate it and replace the context arrows by the activation events that are specified for the current context.

Furthermore, we extended CoGenIVE with a context simulation feature in order to allow the application designer to have a clear view of the NiMMiT diagrams for each context. To do so the user can choose the desired context in a combobox, resulting in a replacement of the context arrows with the activation events that are specified for that context. Figure 8.9 illustrates the wall setup being chosen and the two Wall context arrows that are shown in the diagram. Note that a context arrow does not indicate to which context it belongs and therefore the combo box has to be used in order to be able to see which arrows belong to which context. It would be worthwhile to search for more intuitive visualisations, for example instead of visualising context arrows, the original event arrows could be visualised with extra information indicating to which context it is assigned.

8.4 Conclusion

We have shown that context-awareness for high level notations used within multimodal interaction is not always appropriate at the task level where it is usually introduced. As a solution we have presented an approach which attempts to incorporate contextual knowledge into the dialog level for those situations where the task chains keep the same structure in different contexts of use. We augmented our own high level notation NiMMiT with contextual knowledge. We also showed that this approach makes it possible to switch between different tasks instead of only switch between task chains. Our approach was illustrated using a case study in which a simple scene can be manipulated. We learned that our approach works simple and effective and allows designers to use the same interaction descriptions in different contexts of use.

Even though we introduced the possibility to use context at the dialog level in our model-based user interface design process, we did not provide mechanism or tech-
niques to be able to detect the current context or switch to a new context. In our case study from this chapter we assumed that the context was detected and switched by something unknown. In the next chapter we will propose a complete context system, consisting of three parts, to create context-aware virtual environments.
Chapter 9

Designing Context-Aware Multimodal Virtual Environments

Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.1 Introduction</td>
<td>179</td>
</tr>
<tr>
<td>9.2 Related Work</td>
<td>180</td>
</tr>
<tr>
<td>9.3 Defining a Context System</td>
<td>182</td>
</tr>
<tr>
<td>9.4 Tool Support</td>
<td>187</td>
</tr>
<tr>
<td>9.5 Case Studies</td>
<td>189</td>
</tr>
<tr>
<td>9.6 Discussion</td>
<td>195</td>
</tr>
<tr>
<td>9.7 Conclusion</td>
<td>197</td>
</tr>
</tbody>
</table>

9.1 Introduction

In the previous chapter we integrated context on the dialog level of our model-based design process. This context integration is one important part of a context system, as it allows the application to act adapted to the current context. But besides having a mechanism to handle different context situations, we need a mechanism to be able to detect an active context and switch to that state. We will base our approach on the 'Event-Condition-Action' paradigm (Beer et al., 2003; Etter et al., 2006).

In this chapter, we propose an approach to integrate the use of context-awareness in a model-based user interface design process for the generation of an interactive virtual environment. We will apply the VR-DeMo model-based process and the accompanying tool support CoGenIVE discussed in Chapter 6. The validation of our approach will be studied through two case studies in Section 9.5.
9.2 Related Work

In this section, we discuss several relevant topics within the scope of this chapter. First, we discuss current frameworks for the creation of multimodal user interfaces in general and multimodal interfaces for virtual environments in particular. It is impossible to give a complete overview of all existing frameworks, therefore we will discuss those which we feel are most relevant to the topic discussed in this chapter. We especially look into their support with regard to the handling of context information. As, we will introduce an approach to realize context-aware interactive virtual environments using model-based user interface design, we also have to discuss how context can be modelled in general. Finally, we discuss some frameworks for virtual environments that, to some extent, provide support for contextual information.

9.2.1 Multimodal User Interface Frameworks

For the creation of general 2D multimodal user interfaces, Flippo et al. (2003) present a framework for the rapid development of those interfaces. They focus on re-usable patterns to implement their interfaces. With regard to the use of context, the only aspect being present in this framework is the ability to perform reference resolution for a fusion mechanism of the input modalities. According to our definition of context, this type of context is not considered by our system because modality fusion is achieved by the interaction notation, NiMMiT, as has been described in Section 6.3.

More applicable when creating virtual environments is the COTERIE framework by MacIntyre and Feiner (1996). This framework facilitates the design and separation of application semantics from low-level input handling. Alternatively, Irawati et al. (2008) present the VARU framework, a rapid prototyping framework for virtual reality, augmented reality and ubiquitous computing integrated in one single platform which gives the possibility to explore different types of collaboration across the different spaces. Three other toolkits for the creation of tangible and multimodal interfaces were proposed by Dumas et al. (2008). The SCS system they discuss has several similarities with a MBUID process: it uses the combination of a task model, dialog model and interaction model in a single state machine. However, important to emphasise within the scope of this chapter is that the toolkits described above have no genuine support for context-awareness, and the possible integration of context is usually not considered.

Alternatively, Molina (2008) proposed the TRES-D methodology which allows designers to create virtual environments in several design steps. In one of these steps the designers are offered a guidance tool to decide which interaction techniques are in
9.2 Related Work

general best suited for their application domain. This information could be contextualized and used at runtime such that in a certain situation another technique would be preferred, this technique would be activated. Similarly, Trevisan et al. (2005) propose a design space (DeSMiR), which is an abstract tool for systematically exploring several design alternatives at an early stage of interaction design, without being biased towards a particular modality or technology.

9.2.2 Context Modelling

A context model is necessary in order to be able to apply context-awareness. Many different models exist (Baldauf et al., 2007; Coutaz and Rey, 2002; Li and Willis, 2006; Serrano et al., 2005). A complete discussion of all these models falls beyond the scope of this dissertation. Instead we will limit our discussion to research initiatives focusing on modelling context which have tool support and where the modelling stage is embedded in the model-based user interface design process.

ICap (Sohn and Dey, 2003) supports rapid prototyping of context-aware applications. Their focus is to support quick prototype design of a context-aware application without the production of source code.

Rousseau et al. (2005) present a multimodal output specification and a simulation platform. Both platforms are supported by tools to facilitate the design of multimodal output. A design process is described in which interaction context is taken into account. This is implemented in such a way that the designers know the different contexts of use that can occur and how these contexts of use might influence the multimodal output. The different contexts of use are described in a specification tool. The evaluation of election rules defines which output has to be presented depending on the context. MobiLife (Kernchen et al., 2005) aims at exploiting the synergetic use of multimodal user interface technology and contextual information processing. A ‘Device and Modality Function’ provides functionalities to abstract the interaction between the devices/modalities and the application. Depending on the context, other devices/modalities are chosen or combined. A ‘Device Gateway Function’ makes it easier to provide these functionalities.

In the domain of MBUID, the use of context also has been studied thoroughly during the past years. Clerckx (2007) presents DynaMo-AID which allows context-aware application development for mobile applications. The primary focus of this work is on the integration of context at the task level, but as discussed in the previous chapter, the context system presented in this dissertation, will mainly reside at the dialog level.
9.2.3 Context in Virtual Environments

Applying contextual knowledge to the domain of virtual environments, Lee et al. (2006) present a framework to converge context-awareness and augmented reality. It is separated in three layers one of which is a context layer used to communicate between augmented reality and the ubiquitous services by means of a broker system. In the adapted universal data model nodes, representing a person, place, task or service, are associated with each other and events are fired to notify which tasks need to be executed when a context change occurs. In this framework, no tool support is available to aid developers or designers. It mostly aids developers during low level programming development.

In this chapter, we describe a context system, for detecting, switching and handling contexts of use at runtime. The proposed solution is built upon the existing VRDeMo MBUID process (see Chapter 6). The context models can be defined from within CoGenIVE, the accompanying tool support, and have their influence on the interaction description model (NiMMiT). In Section 9.5, we will motivate our approach by means of case studies.

9.3 Defining a Context System

The context system we will propose in this section consists of three parts: Context Detection, Context Switching and Context Handling. Context detection is the process for detecting changes in context, while context switching brings the system in the new state that needs to be supported. Finally, context handling adapts the interaction possibilities to the current context. In our solution, we use an approach for context detection and switching for virtual environments that is based on the ‘Event-Condition-Action’ paradigm.

In Section 9.4, we finally show how our approach is suitable for the automatic generation of template diagrams for the runtime support of the context-awareness diagrams, and how all of this is integrated in the existing tool CoGenIVE.

9.3.1 Context Detection and Switching

A possible approach to define a context system is by adopting ‘Event-Condition-Action’ rules (Beer et al., 2003; Etter et al., 2006). A certain event or combination of events can signal a context switch. After the event has been recognised, certain conditions have to be met before switching the context. When these conditions are ful-
9.3 Defining a Context System

filled, it might be necessary to first perform some actions before finalising the context switch. For instance, a user may stand up from the chair (event). Before executing a context switch, we must ensure that the user wears the tracked gloves (condition). If this condition is met, we disable the toolbars that are needed in the desktop setup and connect the cursor to make the glove visible (action). Note that the designer has the freedom to decide which actions are defined as events and which parameters are checked as a condition. In the example above, we assumed that ‘standing up’ is recognised as an event, but if this should appear technically difficult to implement, it is also possible to listen to a more general event, e.g. the movement of a tracker mounted to the user’s head (event), and assert for the tracker’s position in order to decide whether the user is seated or standing (condition).

Beer et al. (2003) suggested that it is advisable to have a visual builder tool supporting the creation of context rules, or even the creation of whole interaction scenarios without programming. This approach of visually modelling context through ‘Event-Condition-Action’ rules allows non-technical people to control smart environments and devices without the knowledge of the entire complexity of a system. In the next section, we propose to use NiMMiT as a graphical notation and CoGenIVE as the accompanying tool support to model context using ‘Event-Condition-Action’ rules.

9.3.2 Context Detection and Switching Using NiMMiT

For the implementation of the ‘Event-Condition-Action’ rules using NiMMiT, we have chosen to split the context system in three separate parts. This must keep the design as modular as possible. One part, driven by one NiMMiT diagram is responsible for the Context Detection while another diagram handles the Context Switch. According to the ‘Event-Condition-Action’ paradigm, the context detection part identifies the events that may cause a context switch, checks whether or not the conditions are fulfilled and finally triggers a ‘context switch event’. This event is at its turn recognised by the context switching part. The last and third part is the Context Handling part which makes sure the correct modalities or tasks are enabled depending on the context.

Context Detection Diagram

In the context detection diagram (see Figure 9.1) a given context is typically represented by a state. Hence, the relevant events that can evoke a context switch are available for each context. In general, these events activate a task chain, which checks the condition using one or more tasks. When the condition is not met, the original
state must be restored. Otherwise, before moving on to a new state reflecting the new context, the task chain has to fire a ‘context switch event’, which is handled in the context switching diagram (see Section 9.3.2).

The implementation of the condition check can be achieved in different ways, using different structures in NiMMiT, such as conditional state transitions. However, we prefer to exploit NiMMiT’s exception handling primitives in order to break a task chain and return to the original state. We believe that this approach will result in the most readable diagrams.

Figure 9.1: The context detection diagram from the first case study (pose-aware interaction) with the ‘GESTURE’ interface.
The tasks that are responsible for checking the condition, must collect their result in a single boolean label (see Figure 9.1). This label is passed on to a predefined task ‘ContinueTaskChain’, which checks whether or not the condition is fulfilled. When the boolean contains the value ‘true’, nothing happens and the task chain is continued. When the value is ‘false’ however, this tasks throws an exception and activates the exception handling mechanism of the NiMMiT task chain. In this particular situation, the simplest form of error handling, interrupting the current task chain and restoring the previous state, suits the needs. This means that when the condition is not met, the original state is restored and no context switching event is generated.

We want to stress that, although the designer may decide to adopt other patterns in order to interrupt the state transition, the proposed approach exploiting NiMMiT’s exception handling, keeps the diagrams simple with no superfluous structures. The approach results in patterns that are uniform across different projects, which is a requirement for the tool support generating template diagrams.

**Context Switching Diagram**

A second NiMMiT diagram, responsible for the action (see Figure 9.7), consists of only one state which contains the ‘context events’ representing the possible context switches. The occurrence of a ‘context event’ activates a task chain that contains the code that has to be executed before the context switch can effectively be performed. This code may include altering objects or controls in the world, enabling or disabling certain devices, etc. The last action in this task chain must explicitly change the context, in order to notify other running NiMMiT diagrams defining the actual user interaction to adapt to the new context (see Section 9.3.2).

Independent of the nature or the number of contexts of use, the NiMMiT diagrams for context switching share a similar pattern, as well, which makes the approach suitable for the generation of template diagrams. Obviously, for specific purposes the designer is free to alter the generated diagrams, e.g. if the designer wants to restrict possible context switches.

**Context Handling**

The third and final part of the context system has been described in the previous chapter. The context system needs to be able to create context-aware NiMMiT diagrams that represent the interaction techniques. In this section, we will shortly give an overview of the approach using an example containing context-aware modality selection.
In Figure 9.2, an example of this approach is depicted. Figure 9.2a shows that in the ‘Start Select’-state several different events (modalities) could trigger the execution of the task chains. Using context information, there is the ability to attach a context to a certain event or modality in such a way that, depending on the context only the according events are active. If for example ‘GLOVE.MOVE’ is intended to be used in the immersive setup, one can attach the ‘immersive’-context to the event-arrow ‘GLOVE.MOVE’. Similarly, the event ‘KEYBOARD.MOVE’ can be used in the ‘desktop’-context.

(a) Events active in two different contexts of use.

(b) Events were attached to context arrows.

Figure 9.2: Context Handling.
As mentioned earlier, it is important to note that if there was no support to couple events to a context the same diagram should be created twice with different events (as in Figure 9.2a) which obviously would make maintenance much harder.

Adding this contextual knowledge to NiMMiT, transforms the view of the diagram according to the context. A part of the resulting diagram containing context arrows is shown in Figure 9.2b.

### 9.4 Tool Support

The aforementioned approach has been integrated in CoGenIVE, the tool supporting the adopted VR-DeMo model-based user interface design process (see Section 6.2.2). The dialog supporting the definition of context units and unit values is shown in Figure 9.3. In Section 8.1, we defined ‘Context’ as a multidimensional vector of observables, called ‘Context Units’. It can be seen how context units and possible values for each unit can be defined. The combination of a particular value for each context unit then defines the context. Hence the maximum number of available contexts of use is the cross product of all unit values, grouped per context unit.

As the context switching diagram is triggered by context events, for each context, a matching event is created and made available in the ‘events’-pane in the interface (see Figure 9.4). Here the events can be selected by the user in order to be used in a NiMMiT diagram.

![Figure 9.3: CoGenIVE dialog to define context units and their values.](image)

From within the ‘add and remove context’ dialog, a template context detection and context switching diagram can be generated. The template for the context detection diagram defines a state for each context. Each destination context or state also gets a task chain already containing a task throwing the according context event. This is the last task in each task chain. Finally, each context has an (empty) event arrow to each task chain, excluding a transition to itself. The designer now only has to add the
desired events to the event arrows and eventually add extra conditions into the task chains, resulting in a diagram similar to Figure 9.1 and Figure 9.6.

The template for the context switching diagram is fairly simple. There is only one state. Next, for each context switch, there is an event arrow pointing to a task chain. This task chain already contains a task to explicitly switch the context (typically the last task). Finally a state transition to the (only) state is performed. In this diagram, the designer only has to add the custom actions that have to be performed before the context switch takes place. This results in a diagram similar to Figure 9.7.

In summary, for a situation with \( n \) different contexts of use, a typical context detection template contains \( n \) states, \( n \) task chains with \( n \) state transitions and finally \( (n - 1) \cdot n \) event arrows. The context switching template contains only one state with \( n \) context events and \( n \) task chains. It may be clear that, with regard to the generation of these templates, automatic layout management in their visual representation is needed but currently falls out of the scope of this dissertation.

Regarding the context handling, CoGenIVE provides the possibility for the designer to indicate in a NiMMiT diagram, defining the user interaction, when certain events explicitly belong to a given context. To simplify the visual representation, the events are collapsed in a ‘context arrow’, as can be seen in Figure 9.2b. The tool support allows the designer to view the resulting diagram for each particular context.
9.5 Case Studies

In this section we will present two case studies which show some of the possibilities of the context system proposed in this chapter. The first case study will use external context based on the pose of the user, i.e. either sitting or standing, while the second case study will use the environment condition of the virtual world itself in combination with user preferences as context.

9.5.1 Case Study 1: Pose-Aware Interaction

In order to evaluate and motivate our approach, we elaborate upon a first practical case study. The result is an interactive virtual environment, for designing a city park (see Section 7.5.4), represented by a 3D stereo projection. A user can interact either seated using a 3D mouse and a PHANToM haptic device, or standing using two pinch-gloves which are tracked with a Nest of Birds magnetic tracker. The first alternative offers the user the experience of having force feedback, but in a very limited workspace. The other alternative offers an intuitive two-handed interaction paradigm with a much larger workspace, but without force feedback. Some useful interaction techniques have been implemented using the NiMMiT notation: Two-Handed Scale, Flying Vehicle Navigation and Virtual Hand Object Selection and Manipulation (Bowman et al., 2004). In this case study, we consider changing from either
seated to standing or vice versa as a context switch, enabling or disabling the appropriate devices. Figure 9.5 gives an impression of the described setup.

**Context Detection**

The context detection diagram is illustrated in Figure 9.1. In a first step the initial context is queried and an initial switch to that context is executed (‘GetContext’ and ‘FireContextEvent’). This additional state and task chain are used only to perform some kind of a ‘virtual context switch’ in order to execute the context switching tasks (action) in the initialisation phase. Next, the diagram performs a state transition to the state of the initial context.

![Diagram of context detection](image)

**Figure 9.6:** The context detection diagram from the pose-aware interaction case study using a head tracker.

The events that indicate the possible context switches are ‘GESTURE.SITTING’ and ‘GESTURE.STANDING’. Whenever these events occur the necessary conditions are
checked using the respective task chains (‘Condition Standing’ and ‘Condition Sitting’). For instance, in Figure 9.1 it is checked whether the user is wearing the gloves (‘CheckGlovesOn’). Note that this task is a custom task, defined and implemented by the designer. For this particular proof of concept implementation, we detect if the gloves have been moved over the past 2 seconds. If there has been a movement, we can assume that the user has put on the gloves. Finally, when the condition is not met, the ‘ContinueTaskChain’-task interrupts the task chain, breaking the context switch. Otherwise, the appropriate context switching event is fired. Finally, a state transition to the other context-state is performed.

A possible problem with the proposed solution is the gesture interface which is supposed to be present in order to recognise the ‘sitting’ or ‘standing’ event. This approach assumes either an advanced gesture recognition engine, or it requires again a lot of ad-hoc code, implementing the gesture.

If, however, we do not have the convenience of a decent gesture interface, similar functionality can be implemented in the context detection diagram, by using a more ‘low level’ event. Figure 9.6 shows an alternative version of the same context detection diagram, but now listening to the raw movements of a head tracker, instead of the detection of an event. The position of the head can reveal very easy if a user is either sitting or standing by looking at the height (Y-axis). Therefore an extra condition has to be added in the condition part, checking for the head position.

**Context Switching**

The context event generated in the context detection diagram is detected by the context switching diagram. The diagram, shown in Figure 9.7, contains one state, listening to all available context events. When the user stands up, the event ‘CONTEXT.STANDING’ is recognised (generated by the context detection diagram), and the left task chain is executed. In this case study, the required actions are relatively straightforward: showing and hiding respective pointers for the new context. For instance, while the user is standing and wearing the gloves, the cursors belonging to the virtual representation of the gloves have to be visible while the spacemouse and PHANToM have to be invisible and inactive.

Finally, the last task in the context switching diagram explicitly changes the current context, so that the interaction description diagrams can handle the newly activated context.
The actual handling of the context is performed as described in Section 8.2.3. In this case study, the interaction diagrams listen to the appropriate events, according to the devices that are enabled in the current context. For instance, for the virtual hand selection technique, the ‘MOVE’-events of the PHANToM haptic device are disabled when the user is standing, avoiding unwanted response from this device. Alternatively, when the user is sitting, the ‘MOVE’-events from the right glove are disabled. Disable, here, means that those events, with regard to the virtual hand selection technique, are not attached to the current active context.

9.5.2 Case Study 2: Preference-Aware Selection Techniques

To show how our approach for context-aware virtual environments can be used for multiple scenarios we will discuss a second case study. In this case study we will present a virtual environment in which user adaptation is performed with regard to interaction techniques. Our approach for creating context-aware virtual environments
can be used because adaptations provided to the users can be seen as different context situations.

In Part I, we learned that several aspects of the virtual world influence the performance of selection techniques differently. This means that there is not one technique which is the best technique in all situations. Thus, this information can be used to indicate preferences and provide users with the 'best' selection technique at a right time. However, it should not be surprising that almost all users will slightly deviate from the expected performance for a certain selection technique in a certain environment condition. Because of this, adaptation seems to be a viable solution to explore. The user currently working in the virtual environment could be taken into account and specific for that user different selection techniques could be better in certain circumstances than those techniques found during a user experiment.

The information obtained from our previous user experiments could for example be used to create a template for a user model of a specific user. A user model, is a model containing specific information about the characteristics (e.g. his preferences) of a user (Rich, 1999; Jameson and Wittig, 2001). This can then be further specified into a specific user model either during a special training session with the application or during interaction with the system. It is beyond the purpose of this dissertation to give a detailed description of the user models and how they are created. We refer the interested reader to the work of Rich (1999) and Jameson and Wittig (2001). For the remainder of this section, we will presume we have a user model which the application can access.

In order to start with a limited number of different interaction techniques, we will only use the 3D bubble cursor and depth ray in this case study. They are especially interesting to use, as we can use the user model created by Octavia et al. (2009). They tested both selection techniques in similar density and occlusion conditions as in Chapter 4. We will use the user model in combination with the current viewpoint of the user on the virtual world. From this viewpoint we will derive the current environment condition (e.g. very sparse scene and almost no objects occluding each other). Based on that current environment condition, we will provide the user with the selection technique which is best suited for him in this condition. Different from the previous case study, we will combine the detection and switching diagram into one diagram. As there are no additional actions to perform before the actual switch there is no reason to have a separate switching diagram.
Context Detection and Switching

As in the previous case study, a first initialisation step is performed (see Figure 9.8). This step queries for the initial context and an initial switch to that context is executed. As there is no switching diagram, we also immediately need to finalise the context switch by executing the ‘SetContext’-task. Due to the fact that the different contexts (i.e., different selection techniques) are chosen by evaluating the same conditions in the ‘CheckViewpoint’-task, we only use one state instead of one state per context situation.

We want to provide the user with different selection techniques depending on the viewpoint, therefore the ‘CAMERA.MOVE’-event, the event fired when the viewpoint changes, indicates a possible context switch. If this event occurs the viewpoint will be checked for the current environment condition. We will check for the amount of objects in the current viewpoint in order to assess the density of the objects. In combination with the density, we calculate how many objects overlap each other such that the possibility can be calculated that a user would want to select an occluded target. If we know the current environment condition we can simply use
9.6 Discussion

the user model of the user currently interacting to decide which selection technique should be chosen.

The difference with the previous case study is that we only have one state and one task chain which controls the context changes. Thus, if the ‘CAMERA.MOVE’-event occurs and the viewpoint is checked (‘CheckViewpoint’), we calculate the current context, but in order to only fire context change events when the context has changed, we add a construction which checks if the context has changed (‘ContextChanged’). If the context has changed, we output ‘true’ and the context switches, else we output ‘false’ and the task chain breaks as explained earlier. The ‘ContextChanged’-task can be seen as an extra condition to check before a new context is detected.

Context Handling

In this case study we need to switch between different interaction techniques and therefore will use the approach described in Section 8.2.4. The NiMMiT diagram presented in Figure 8.4d is the diagram which we will use in this case study, depending on the current context, the correct ‘IDLE’-event arrow will be active such that the correct selection technique will be activated.

The solution presented here possibly contains a problem, what if the context changes as a certain selection technique is already active? In the solution presented here, the selection technique currently active will not be aborted, only after the selection is performed the other selection technique can become active. Depending on the preferences of the application this might be the expected behaviour, but it is also possible to abort the current selection technique and enable the other selection technique. To realise this, an action would need to be added which changes a label that informs the current running selection technique that it needs to stop immediately. The main drawback is that this requires to change the selection technique itself by incorporating this aborting-check using the label.

9.6 Discussion

The aforementioned approach describes a well-structured solution in order to support context detection, switching and handling using a single graphical notation, NiMMiT. Since we define context as a n-dimensional vector of observables, called ‘Context Units’, very complex context systems, with several orthogonal units can be built. As the maximum number of contexts of use is the result of the cross product of all unit values, grouped per context unit, the number of contexts may grow rapidly with
the number of units and values. This implies a possible explosion of the number of states in the diagrams, inevitably resulting in complex diagrams, even in spite of the proposed separation of the diagrams. We believe, however, that the amount of different context units and their values will remain limited in practical situations with regard to virtual environments, such that the complexity of the diagrams won’t be catastrophic.

Alternatively, the other extremum can occur as well. One could argue that for very simple context systems, in which two contexts of use exist, where no specific conditions have to be checked, and no additional actions before the actual switch have to be taken, the discussed approach may be ‘overkill’. In those very simple cases, however, both the context detection and the context switching diagram can be merged into a single diagram, containing both the basic events (coming from the devices), as well as the conditions and the actions. This approach was used in the preference-aware selection techniques case study.

![Image](image.png)

**Figure 9.9:** Part of a NiMMiT diagram to change the users preference with regard to the force strength using a label.

Finally, a last point of discussion is the fact that the proposed approach can be used to define quite powerful context handling. Indeed, in the first case study, we showed how the context switching diagram has been used to perform simple visible actions inside the virtual environment such as enabling or disabling devices. Alternatively, it is also possible to assign values to a label (variable) in the context switching diagram, which we briefly mentioned in the second case study. Using input and output labels of the NiMMiT diagrams, these values can be exchanged between the different diagrams running in parallel. Imagine for instance an interaction diagram that has to use the position values for both the dominant and non-dominant hand. According to the
9.7 Conclusion

handedness of the user, the function of both devices may be altered. An easy solution would be is to use a label indicating which device is assigned to what hand. The mechanism of changing label values in the context handling diagram depending on the context, can be very useful for many types of contexts of use such as general user preferences (e.g. strong or soft force feedback). An example is illustrated in Figure 9.9.

9.7 Conclusion

In the previous chapter we proposed an approach in which context was introduced at the dialog level, no support was provided to detect or switch between different contexts of use. This shortcoming was the topic of this chapter. In this chapter we presented an approach to design context-aware multimodal virtual environments using a model-based user interface design process.

We applied NiMMiT, a graphical notation for the description of the user interaction in a virtual environment to create a context system based on an ‘Event-Condition-Action’ paradigm. For the context system, two NiMMiT diagrams are used, one for detecting the context and checking the conditions, and a second diagram for performing the context switch. Using this approach, the complexity of both diagrams is reduced, they are more modular and better maintainable. In order to validate our approach, in this chapter, we described two case studies illustrating the proposed context modelling system. We believe that the proposed context system provides the designer with a powerful tool to create context-aware virtual environments while applying a model-based approach, but further validation on a larger scale is necessary.
Conclusion

The creation and integration of selection techniques for virtual environments in combination with semantic and contextual knowledge has been the focus of this part of the dissertation.

Chapter 7 first looked at how semantic information can be integrated in the VR-DeMo model-based design process which we presented in Chapter 6. In order to be able to use this semantic information, created during the virtual world modelling, we incorporated it inside NiMMiT as a new data type called concept. To support the new data type, which makes it possible to use the semantic information, two additional predefined tasks were integrated. Our systems is not tied to a certain semantic description, which provides flexibility to the semantic information. It was found that these predefined tasks can easily be used to create a selection task which only allows object of a specific type to be selected. If more special needs are required, it is also possible to integrate the new data type in custom tasks. In the future, the semantic information will also be added to other parts of the modelling phase, such as VRiXML, the presentation model in VR-DeMo. The items in the menu and possible values are good candidates to be be enhanced with semantics.

Besides the integration of semantic information in NiMMiT we also used the semantic information to automatically generate a speech grammar. We proposed to use an object-action structure as a grammar and the recognised speech is converted to SPARQL queries. From a case study we could conclude that users used mainly words contained by the speech grammar during interaction with the application. For future work it would be good to incorporate contextual knowledge, such as the current location of the user inside the virtual environment, besides the ontological and linguistic knowledge which both come “for free” during the conceptual modelling phase. In order to further validate the approach a more formal user study will give more insights into the strengths and weaknesses of our approach.
Alongside semantic information, we incorporated contextual knowledge. In Chapter 8 we first proposed an approach to introduce context at the dialog level instead of at the task level. By introducing context arrows in NiMMiT, which allows to tie specific events to a specific context, we were able to avoid an explosion of dialog models. Afterwards we proposed a context system incorporating this mechanism such that interaction depending on the context of use can be designed. The context system consisted of two other parts, a detection and switching part, which are used to detect the current context and aid the application to switch to the correct context. The two parts are based on the ‘Event-Condition-Action’ paradigm and are realised through two separate NiMMiT diagrams. This separation allows for a simpler, more modular and better maintainable solution.

We tested our approach using two case studies in which external context was used and one case study in which the environment condition of the virtual world in combination with the user preferences provided the context. In the first case study the user could choose between a desktop or wall setup depending on his personal preference. In the second case study if either force feedback or a larger working area were more optimal for fluent interaction, the user could work seated using a PHANToM, or standing using gloves and finally, in the third case study, the selection technique was changed between the 3D bubble cursor or depth ray depending on the current viewpoint. These case studies showed us that our approach allowed to create context-aware virtual environments. In the future, several other scenarios using the proposed approach will be further explored. Such scenarios will for example take into account the user’s experience, which can be modelled using context.
Chapter 10

Conclusions and Future Research Directions

10.1 Conclusions

In this dissertation we investigated multimodal selection techniques for virtual environments. Our efforts were twofold. We performed research into selection techniques augmented with multimodal feedback on the one hand. On the other hand, we proposed approaches to integrate semantic and contextual knowledge in a model-based user interface design process for virtual environments.

With regard to the selection techniques, we found that multimodal feedback used during the feedback sub-task, in the form of gravity wells and earcons for the traditional virtual hand technique, showed significant improvements. But for the 2D bubble cursor, which heavily outperformed the virtual hand technique with or without multimodal feedback, no improvement through multimodal feedback was found. Based on these results, we designed selection techniques for dense and occluded environments. We found that the 3D bubble cursor and depth ray performed equally well and were able to cope with dense environments. For occlusion, an augmentation with a transparency function showed that there was only a minimal overhead of one second compared to selecting visible targets. The multimodal feedback showed small non-significant improvements, indicating that primarily good visual feedback is very important.

In both selection experiments the design of the force feedback was performed using a pilot test or a trial-and-error approach, therefore we tried to formulate a guideline to design force feedback. A series of multidirectional point-select experiments showed
that the definite integral of the force profile can be used as a guideline to know when a certain force feedback value will cause the performance of the user to deteriorate during his movement.

Apart from formal experiments for selection techniques, we discussed how semantic and contextual knowledge has been added to the VR-DeMo model-based user interface design process. Semantic information has been added through the addition of a data type to the interaction description model (NiMMiT) and the automatic generation of a speech grammar. In order to add contextual knowledge we first introduced context at the dialog level of our model-based user interface design process through the enhancement of NiMMiT with context arrows. This allowed us to realise context-dependent interaction in virtual environments. But in order to be able to also detect the new context and switch to that context, we proposed a context system based on the ‘Event-Condition-Action’ paradigm. The detection and switching parts are each realised using a NiMMiT diagram. Several case studies showed that these additions can be useful during the design of interaction techniques, including techniques for the selection task. We feel that the addition of semantic and contextual knowledge has shown potential and is worth considering to be used during the design of virtual environments.

We hope that the research performed in the context of this dissertation will both facilitate designers and developers. For designers we gained new insights with regard to multimodal selection techniques through a series of formal studies. For developers, our work opens new perspectives with regard to the creation of virtual environments. This in turn influences designers, offering possibilities for inspiring new interaction techniques. These new insights or interaction techniques lead to better user interfaces which in turn lead to more satisfied users.

10.2 Future Research Directions

Based on the results of this dissertation several directions for future research are possible.

The environments used in the selection experiments contained abstract scenes without any particular final application in mind. One important direction is to see how our selection techniques would perform in real applications. Small adjustments might be needed for different application domains. For example, a virtual world builder application might have complex objects, but they can mostly be handled through simple bounding volumes, very similar to the objects used in our experiments and therefore our techniques probably won’t need many adjustments. On the other hand,
in a very dynamic environment such as flow visualisation applications, it is not so straightforward to predict how the dynamic behaviour of objects would influence our techniques.

In our experiments we always focused on the selection of single objects, it is also interesting to see how multiple object selections could be performed. We are currently exploring the possibility of selecting all/multiple objects of a certain type. In order to be able to identify types of objects, the semantic information from Chapter 7 will be used. Several strategies will be explored on how the users can indicate which type(s) of objects they want to select, for example using a speech interface, some form of menu or a selection which indicates that the user only wants to select that type of objects. Furthermore, after having specified the object type, the selection itself still needs to be performed. An approach using two hands in which both hands in parallel can perform selections or in which both hands form a selection volume (e.g. a box or sphere) could be interesting.

With regard to the selection techniques research should focus on how to maximise the effective width of a target. One possibly strategy worthwhile to investigate is to visually resize the target for better visual feedback (e.g. enlarge it through scaling). For 2D GUIs this has successfully been applied (McGuffin and Balakrishnan, 2005), but for 3D interfaces no real success has yet been reached (Argelaguet and Andújar, 2008). Another possibility, instead of maximising the effective width of a target, is to utilise the intrinsic effective width of a target as efficient as possible, similar to using the Voronoi regions for the bubble cursor.

Our research on context-aware virtual environments primarily involved external context, it is important to thoroughly study other types of context as well. Currently in our group research is being conducted on how adaptation and personalisation can be used in virtual environments to enhance the experience of the user during interaction. The second case study of Chapter 9 showed preliminary steps on how our context system can be used to switch between selection techniques depending on a certain environment condition. Further work is necessary to see how useful adaptation and personalisation can be and if users prefer these types of context-awareness.

With regard to the force feedback experiments of Chapter 5 several more experiments are necessary to further improve and validate our guideline to use the definite integral of the force profile. For example, the bounds within which the definite integral can be used should be investigated, probably very strong and short bumps or very weak and long bumps won’t adhere to our guideline. Another interesting question is whether different Fitts’ indexes of difficulty will influence the results of these experiments. Apart from further investigating this type of force feedback, it might be possible that other types of force feedback, such as gravity wells, can be subjected to a similar
Conclusions and Future Research Directions

guideline. On the long term it might be interesting to see if this guideline could be transferred to other types of user interfaces such as tactile mobile phones or pen-based user interfaces.

The integration of semantic information in interaction with virtual environments seems promising, but an important aspect of this information is the availability. In the research presented in this dissertation virtual worlds were created using a conceptual modelling approach which built the world using the semantic information as a basis. However, almost all currently existing virtual worlds have not been realised in such a way and therefore do not contain semantic information. For the further possible adoption of semantic information, existing virtual worlds need to be annotated, which is a rather unexplored area of research. Furthermore, existing modelling tools for virtual worlds do not incorporate semantic information. Besides supporting the use of semantic information in such a modelling tool, the impact on the designers and possible improvements in their approach and development process have to be explored.

We hope that the research presented in this dissertation provides valuable insights for other researchers and inspires future research in this field, whether stated in this section or not.

10.3 Scientific Contributions and Publications

This dissertation reports about research that was performed during approximately four years. Along the road, several publications disseminated part of the results. In this section we provide an overview of the publications and highlight the scientific contributions.

- We compared several selection techniques and evaluated them in formal experiments (Chapters 3 and 4). Besides the addition of multimodal feedback to these selection techniques, the influence of the environment density and occlusion was studied.

10.3 Scientific Contributions and Publications


- We investigated which force feedback magnitudes do not deteriorate the performance of the user’s movements during selection (Chapter 5). We deduced a guideline using the force integral of the force bump.


- We studied the addition of semantic information to a model-based design process for virtual environments and more in particular how it can benefit the realisation and integration of selection (Chapter 7).


Conclusions and Future Research Directions


- Context was introduced at the dialog level instead of the regular task level (Chapter 8).


- We presented a context system which can design context-aware virtual environments (Chapter 9).


Appendices
Driedimensionale (3D) virtuele omgevingen hebben veel verschillende toepassingsdomeinen. Ze kunnen gebruikt worden voor het visualiseren en interageren met 3D data zoals moleculaire data en medische scans alsook voor het ontwerpen van nieuwe prototypes.

Het is belangrijk deze virtuele omgevingen zo realistisch mogelijk te ervaren. Vooral veel aandacht is al uitgegaan naar het visuele aspect van virtuele omgevingen. Er bestaan veel verschillende manieren om virtuele omgevingen visueel te ervaren. Driedimensionale beeldtechnologieën zoals immersieve of niet-immersieve virtuele realiteitsystemen gebruikmakende van LCD shutter stereo-brillen, zijn veel verbeterd in beeldkwaliteit. In deze thesis wordt er gebruik gemaakt van niet-immersieve stereoscopische beeldtechnologie. Naast het visuele zijn er nog andere zintuigen (modaliteiten) beschikbaar om te gebruiken, zoals het gevoel (haptics) en het gehoor (audio).

De laatste jaren wordt er meer en meer aandacht gegeven aan het toevoegen en combineren van deze modaliteiten (multimodal feedback). Deze andere zintuigen kunnen de gebruiker extra informatie verschaffen tijdens de interactie in de virtuele omgeving. Gedurende deze thesis, zullen we de mogelijke bijdrages van haptics en audio onderzoeken.

Een van de belangrijkste taken die moet ondersteund worden in eender welke interactieve applicatie, en dus ook in 3D virtuele omgevingen, is object selectie. Voor 3D virtuele omgevingen wordt selectie als een van de 4 basistaken aanzien, naast navigatie, manipulatie en invoer van data. Selectie maakt het mogelijk dat gebruikers een object aanduiden dat ze willen manipuleren of waarmee ze willen interageren.
Zonder de mogelijkheid om een object te specificeren wordt het bijna onmogelijk om te interageren met een virtuele omgeving.

In deze thesis hebben we dus onderzoek gedaan naar multimodale selectie voor virtuele omgevingen. Onze bijdragen zijn tweevoudig. Enerzijds zullen we met behulp van formele gebruikersexperimenten verschillende multimodale selectietechnieken onderzoeken om zo nieuwe inzichten te kunnen opdoen. Anderzijds zullen we manieren uitdiepen waarop semantische en contextuele informatie geïntegreerd kan worden in een model-gebaseerd user interface ontwerpproces voor virtuele omgevingen. De praktische uitwerkingen hebben vooral betrekking op selectietechnieken.

Aangezien in een 3D omgeving objecten gepositioneerd zijn in drie dimensies moeten andere strategieën onderzocht worden dan de traditionele 2D muis. Er is reeds heel wat onderzoek naar selectie gebeurd, maar er is nog enig onderzoek gegeven aan manieren om haptics en audio toe te voegen tijdens de selectietoets. In deze thesis ligt de focus op niet-bewegende abstracte objecten zoals dozen of bollen in een 3D virtuele wereld. Daarnaast zijn nog niet alle factoren die de omgeving beïnvloeden, zoals de dichtheid en/of zichtbaarheid, onderzocht. De extra modaliteiten worden altijd toegevoegd op zulk een manier dat ze proberen de gebruiker te helpen tijdens het aanduidproces van de selectietoets en bijvoorbeeld niet bij de manier waarop de selectie wordt bevestigd. Aangezien de snelheid en foutloosheid van de selectietoets het belangrijkst zijn, zijn dit de parameters waarop we zullen testen.

In het eerste gebruikersexperiment vonden we dat multimodale feedback, in de vorm van gravity wells en earcons, significante verbeteringen bracht voor de traditionele virtual hand techniek. Maar voor de 2D bubble cursor, die veel beter dan de virtual hand (met of zonder multimodale feedback) presteerde, zagen we geen verbetering wanneer multimodale feedback aanwezig was. Gebaseerd op deze resultaten hebben we selectietechnieken ontworpen voor zeer dichte omgevingen en omgevingen met verborgen objecten. In volgende gebruikersexperimenten vonden we dat de 3D bubble cursor en depth ray gelijkwaardig presteerden en goed overweg konden met dichte omgevingen. Voor verborgen objecten werd er een transparantiefunctie gebruikt die er voor zorgde dat er slechts een overhead was van een seconde wanneer er niet-zichtbare objecten geselecteerd moesten worden. De additie van multimodale feedback toonde kleine niet-significante verbeteringen, aangezien dat vooral goede visuele feedback zeer belangrijk is.

In de voorgaande experimenten was het ontwerp van de krachterugkoppeling (haptics) gebeurd aan de hand van manueel uitproberen totdat er waardes gevonden waren die het beste bleken te zijn. Om deze reden hebben we geprobeerd om een richtlijn op te stellen om krachterugkoppeling te ontwerpen. Een reeks van multidirectionele selectieexperimenten toonde aan dat de bepaalde integraal van het krachtprofiel, de
functie die het verloop van de krachten beschrijft, gebruikt kan worden als een richtlijn om te weten wanneer een bepaalde kracht de performantie van de bewegingen van de gebruiker zal verslechteren.

Naast onderzoek naar de selectietechnieken zelf, is het ook belangrijk dat er gekeken wordt naar hoe deze technieken gerealiseerd worden. De implementatie van interactietechnieken, in ons geval selectietechnieken, is meestal redelijk complex. Vooral aan herbruikbaarheid in andere applicaties of andere situaties wordt weinig aandacht gegeven. Daarnaast is het ook interessant om extra beschikbare informatie, zoals contextuele en semantische informatie, te integreren tijdens de ontwerpfase van de virtuele omgeving.

We bespreken ook hoe semantische en contextuele informatie toegevoegd is aan het VR-DeMo model-gebaseerd user interface ontwerpproces. Semantische informatie is toegevoegd door een nieuw datatype te introduceren in het interactiebeschrijvingsmodel (NiMMiT) en door de automatische generatie van een spraakgrammatica. Om contextuele informatie te kunnen gebruiken hebben we eerst context geïntroduceerd op het dialoog niveau van ons model-gebaseerd user interface ontwerpproces door NiMMiT te voorzien van contextpijlen. Op deze manier kregen we de mogelijkheid om context-gevoelige interactie in virtuele omgevingen te realiseren, maar ook om de huidige context te detecteren en er naar te veranderen. Hierna hebben we een contextsysteem voorgesteld gebaseerd op het ‘Event-Condition-Action’ paradigma. Het detectie- en verandergedeelte worden gerealiseerd door een NiMMiT-diagram. Verscheidene case studies toonden aan dat onze toevoegingen nuttig kunnen zijn tijdens het ontwerpen van interactietechnieken.

We hopen dat het onderzoek uitgevoerd in de context van deze thesis zowel het werk van designers als ontwikkelaars kan vergemakkelijken. Voor designers hebben we nieuwe inzichten verworven met betrekking tot multimodale selectietechnieken door middel van een aantal formele experimenten. Voor ontwikkelaars biedt ons werk nieuwe perspectieven met betrekking tot de realisatie van virtuele omgevingen, wat ook de designers beïnvloedt. Zij krijgen meer mogelijkheden om met nieuwe interactietechnieken te experimenteren. Deze nieuwe inzichten of interactietechnieken leiden tot betere user interfaces die uiteindelijk leiden tot meer tevreden gebruikers.
Appendix B

An Automatic Generated Speech Grammar

This appendix contains the automatically generated speech grammar (Microsoft Speech SDK 5.1) by the process discussed in Section 7.5. The addition of synonyms is removed for clarity.

<GRAMMAR>
  <RULE NAME="CLASSES" TOPLEVEL="INACTIVE">
    <L PROPNAME="CLASS">
      <P VALSTR="Stop">stop</P>
      <P VALSTR="Car">car</P>
      <P VALSTR="Yield">yield</P>
      <P VALSTR="Residence">residence</P>
      <P VALSTR="Bus">bus</P>
      <P VALSTR="Monument">monument</P>
      <P VALSTR="Fountain">fountain</P>
      <P VALSTR="Signs">signs</P>
      <P VALSTR="Statue">statue</P>
      <P VALSTR="Hotel">hotel</P>
      <P VALSTR="Road">road</P>
      <P VALSTR="Bench">bench</P>
      <P VALSTR="Tree">tree</P>
      <P VALSTR="Vehicle">vehicle</P>
      <P VALSTR="House">house</P>
      <P VALSTR="Apartment">apartment</P>
      <P VALSTR="Obelisk">obelisk</P>
    </L>
  </RULE>
</GRAMMAR>
<RULE NAME="PROPERTIES" TOLEVEL="INACTIVE">
  <L PROPNAME="PROPERTY">
  <P>behind</P>
  <P>front</P>
  <P>below</P>
  <P>above</P>
  <P>left</P>
  <P>right</P>
  </L>
</RULE>

<RULE NAME="INSTANCES" TOLEVEL="INACTIVE">
  <L PROPNAME="INSTANCE">
  <P VALSTR="Egypt,Monument">Egypt</P>
  <P VALSTR="Hilton,Residence">Hilton</P>
  <P VALSTR="BenchFrontHilton,Bench">BenchFrontHilton</P>
  <P VALSTR="Mercedes,Vehicle">Mercedes</P>
  </L>
</RULE>

<RULE NAME="DATAPROPERTIESVALUES" TOLEVEL="INACTIVE">
  <L PROPNAME="DATAPROPERTYVALUE">
  <P VALSTR="brand,BMW">BMW</P>
  <P VALSTR="brand,Mercedes">Mercedes</P>
  <P VALSTR="owner,DeLijn">DeLijn</P>
  <P VALSTR="owner,Hilton">Hilton</P>
  <P VALSTR="owner,Lode Vanacken">Lode Vanacken</P>
  <P VALSTR="owner,Sultan">Sultan</P>
  <P VALSTR="year,2004">2004</P>
  <P VALSTR="year,1970">1970</P>
  <P VALSTR="year,2006">2006</P>
  <P VALSTR="year,2001">2001</P>
  <P VALSTR="colour,Orange">Orange</P>
  <P VALSTR="colour,Yellow">Yellow</P>
  <P VALSTR="colour,Black">Black</P>
  <P VALSTR="colour,white">white</P>
  <P VALSTR="colour,Brown">Brown</P>
  </L>
</RULE>

<RULE NAME="DATAPROPERTIES" TOLEVEL="INACTIVE">
  <L PROPNAME="DATAPROPERTY">
  <P VALSTR="year">year</P>
  <P VALSTR="pumpStrength">pumpStrength</P>
  </L>
</RULE>
<P VALSTR="slope">slope</P>
<P VALSTR="brand">brand</P>
<P VALSTR="owner">owner</P>
<P VALSTR="colour">colour</P>
</L>
</RULE>

<RULE NAME="ALLQUERIES" TOLEVEL="INACTIVE">
 <P VALSTR="QUERYTYPE_CLASS">
  <O PROPNAME="ALL" VALSTR="All">...</O>
  <L>
   <P>
    <RULEREF NAME="CLASSES"/>
   </P>
   <P>
    <RULEREF NAME="INSTANCES"/>
   </P>
  </L>
 </P>
 <P VALSTR="QUERYTYPE_CLASS_RELATION">
  <L>
   <P>
    <RULEREF NAME="CLASSES"/>
   </P>
   <P>
    <RULEREF NAME="INSTANCES"/>
   </P>
  </L>
  <O>...</O>
  <RULEREF NAME="PROPERTIES"/>
  <O>...</O>
  <L>
   <P>
    <RULEREF NAME="CLASSES"/>
   </P>
   <P>
    <RULEREF NAME="INSTANCES"/>
   </P>
  </L>
 </P>
 <P VALSTR="QUERYTYPE_CLASS_DATAPROPERTY">
  <P MAX="INF">
   <RULEREF NAME="DATAPROPERTIESVALUES"/>
  </P>
 </P>
An Automatic Generated Speech Grammar

<P>
<RULEREF NAME="CLASSES"/>
</P>
</RULE>

<RULE NAME="SPEECH.SELECTEVENT" TOLEVEL="ACTIVE">
<L PROPNAME="QUERYTYPE">
<P>
<RULEREF NAME="ALLQUERIES" PROPNAME="QUERYTYPE"/>
</P>
<P VALSTR="QUERYTYPE_DATAPROPERTY">
<L>
<P>What</P>
<P>Which</P>
<P>How</P>
</L>
</P>
</RULE>
</GRAMMAR>
This appendix contains copies of the questionnaires administered during the user experiments.
Figure C.1: A (translated from Dutch) copy of the questionnaire administered after the user experiment described in Chapter 3.

<table>
<thead>
<tr>
<th>Test Person: ..........</th>
<th>Experiment ModalitySelectionTest</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sex:</td>
<td>□ male   □ female</td>
</tr>
<tr>
<td>Age:</td>
<td>______</td>
</tr>
<tr>
<td>1. How much experience do you have with 3D Virtual Environments (similar systems, 3D shooters, . . .)?</td>
<td></td>
</tr>
<tr>
<td>□ None</td>
<td>□ Very Low □ Low □ High □ Very High</td>
</tr>
<tr>
<td>2. Order the 3 selection methods using a number (1 = the best one, 3 = the worst one)</td>
<td></td>
</tr>
<tr>
<td>___ Virtual Hand</td>
<td>___ Aperture</td>
</tr>
<tr>
<td>___ Bubble Cursor</td>
<td></td>
</tr>
<tr>
<td>3. Put a cross at the modality combinations of which you found that they provided benefits during the selection.</td>
<td></td>
</tr>
<tr>
<td>Virtual Hand</td>
<td></td>
</tr>
<tr>
<td>Nothing</td>
<td></td>
</tr>
<tr>
<td>Sound</td>
<td></td>
</tr>
<tr>
<td>Haptics (collision)</td>
<td></td>
</tr>
<tr>
<td>Haptics (attraction)</td>
<td></td>
</tr>
<tr>
<td>Sound en Haptics (collision)</td>
<td></td>
</tr>
<tr>
<td>Sound en Haptics (attraction)</td>
<td></td>
</tr>
<tr>
<td>Bubble Cursor</td>
<td></td>
</tr>
<tr>
<td>Nothing</td>
<td></td>
</tr>
<tr>
<td>Sound</td>
<td></td>
</tr>
<tr>
<td>Haptics</td>
<td></td>
</tr>
<tr>
<td>Sound en Haptics</td>
<td></td>
</tr>
</tbody>
</table>

4. Give your opinion on how sound was added to selection (feel free to express another approach of adding sound): 

5. Give your opinion on how haptics was added to selection (feel free to express another approach of adding haptics): 

6. Other general remarks: 

This concludes the test. May we ask you to not give any information to other test persons; this could influence the results of the research. Thank you for your cooperation!
Figure C.2: A copy of the questionnaire administered after the first user experiment described in Chapter 4 in Section 4.5.
Figure C.3: A copy of the questionnaire administered after the second user experiment described in Chapter 4 in Section 4.6.

Test person: ............. Feedback Experiment

Sex: □ male □ female

Age: _____

How much experience do you have with 3D Virtual Environments (similar systems, 3D shooters,…)  
□ None □ Very Low □ Low □ High □ Very High

Other General Remarks

This concludes the test. May we ask you to not give any information to other test persons; this could influence the results of the research.

Thank you for your cooperation!
Technique: ……

Classify the selection technique using a number (1 = Strongly Disagree, 5 = Strongly Agree)

The technique allowed for fast selections

Strongly Disagree 1 2 3 4 5 Strongly Agree

The technique was easy to learn

Strongly Disagree 1 2 3 4 5 Strongly Agree

The technique allowed for accurate selections

Strongly Disagree 1 2 3 4 5 Strongly Agree

The multimodal feedback improved my ability to complete the task quickly

Strongly Disagree 1 2 3 4 5 Strongly Agree

The multimodal feedback improved my ability to complete the task accurately

Strongly Disagree 1 2 3 4 5 Strongly Agree

The multimodal feedback made the technique easier to use

Strongly Disagree 1 2 3 4 5 Strongly Agree

The multimodal feedback improved my ability to understand when the target could be selected

Strongly Disagree 1 2 3 4 5 Strongly Agree

I preferred the technique when the multimodal feedback was enabled

Strongly Disagree 1 2 3 4 5 Strongly Agree

Overall I found this technique an effective mechanism for the selection task

Strongly Disagree 1 2 3 4 5 Strongly Agree

- The multi modal feedback during the selection technique, could you shortly describe it (both force and audio feedback) and give your opinion on it?
  
  **Force:**

  **Audio:**
Figure C.4: A copy of the questionnaire administered after the first user experiment described in Chapter 5 in Section 5.3.

<table>
<thead>
<tr>
<th>Test person: ..........</th>
<th>Baseline Force Shape Experiment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sex:</td>
<td>male □ female □ □ right □ left</td>
</tr>
<tr>
<td>Age:</td>
<td>_____</td>
</tr>
<tr>
<td>Have you ever performed such a selection experiment with circle targets inside a circle?</td>
<td>□ Yes □ No</td>
</tr>
<tr>
<td>What is your experience with the input device being used?</td>
<td>□ None □ Very Low □ Low □ High □ Very High</td>
</tr>
<tr>
<td>In how many directions along your movement did you feel forces?</td>
<td>_____</td>
</tr>
<tr>
<td>Did you always feel forces?</td>
<td>_____</td>
</tr>
<tr>
<td>How many different force strengths did you identify (don’t include direction)?</td>
<td>_____</td>
</tr>
<tr>
<td>Were some of the forces that you felt, too strong?</td>
<td>□ Yes □ No</td>
</tr>
</tbody>
</table>

Please explain your answer:

Do you think that these types of forces (strong or weak) may be useful in any type of user interface, a traditional or futuristic one? □ Yes □ No

Please explain your answer:

Other General Remarks:

This concludes the test. May we ask you not to give any information to other test persons; this could influence the results of the research. Thank you for your cooperation!
Figure C.5: A copy of the questionnaire administered after the second user experiment described in Chapter 5 in Section 5.4.

<table>
<thead>
<tr>
<th>Test person: .............</th>
<th>Integral Experiment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sex: □ male □ female</td>
<td>Handedness: □ right □ left</td>
</tr>
<tr>
<td>Age: _____</td>
<td></td>
</tr>
<tr>
<td>Have you ever performed such a selection experiment with circle targets inside a circle?</td>
<td>□ Yes □ No</td>
</tr>
<tr>
<td>What is your experience with the input device being used?</td>
<td>□ None □ Very Low □ Low □ High □ Very High</td>
</tr>
<tr>
<td>Did you always feel forces?</td>
<td>□ Yes □ No</td>
</tr>
<tr>
<td>How many different force strengths did you identify?</td>
<td>□</td>
</tr>
<tr>
<td>Were some of the forces that you felt, too strong?</td>
<td>□ Yes □ No</td>
</tr>
<tr>
<td>Please explain your answer:</td>
<td></td>
</tr>
</tbody>
</table>

Other General Remarks:

This concludes the test. May we ask you not to give any information to other test persons; this could influence the results of the research. Thank you for your cooperation!
Figure C.6: A copy of the questionnaire administered after the third user experiment described in Chapter 5 in Section 5.8.

<table>
<thead>
<tr>
<th>Test person: .............</th>
<th>Device Subjective Experiment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sex: □ male □ female</td>
<td>Handedness: □ right □ left</td>
</tr>
<tr>
<td>Device: □ Phantom □ Falcon</td>
<td></td>
</tr>
<tr>
<td>Age: _____</td>
<td></td>
</tr>
<tr>
<td>Have you ever performed such a selection experiment with circle targets inside a circle?</td>
<td></td>
</tr>
<tr>
<td>□ Yes □ No</td>
<td></td>
</tr>
<tr>
<td>Have you ever used a haptic device?</td>
<td></td>
</tr>
<tr>
<td>□ Yes, this one □ Yes, another one □ No</td>
<td></td>
</tr>
<tr>
<td>What is your experience with the input device being used?</td>
<td></td>
</tr>
<tr>
<td>□ None □ Very Low □ Low □ High □ Very High</td>
<td></td>
</tr>
<tr>
<td>Did you like the handle (the grip) of the device?</td>
<td></td>
</tr>
<tr>
<td>□ Yes □ No</td>
<td></td>
</tr>
<tr>
<td>Please explain your answer:</td>
<td></td>
</tr>
</tbody>
</table>

Other General Remarks:

This concludes the test. May we ask you not to give any information to other test persons; this could influence the results of the research. Thank you for your cooperation!


nual ACM symposium on User interface software and technology, pages 3–12, Montreux, Switzerland.


6th international ACM SIGACCESS conference on Computers and accessibility, pages 102–109, Atlanta, GA, USA.


