University of Twente,
Human Media Interaction group

MSc thesis

Multi-user interaction with molecular visualizations on a multi-touch table

Author:
Jeroen Logtenberg

Supervisors:
Paul van der Vet
Wim Fikkert
Jack Leunissen

August 11, 2009

Contents

Chapter 1

Introduction

In biomedical research, molecules play an important role. By examining complex macromolecules such as proteins and nucleic acids, structural biologists can explore and explain phenomena of nature, for instance the origin of diseases. Using molecular visualization software such as Yasara [17], Jmol [12] or PyMOL [2], structural biologists study the composition and 3D structure of these molecules. If biologists want to examine molecular structures together, there are limited options. Because the visualization software is run on a PC, the input is traditionally provided by a mouse and keyboard. This means that usually one person has control of the software, while others are left to watch. The goal of this research is to provide a more interactive and comfortable way of examining molecular visualizations together. The ideal choice for such a way is a single display groupware (SDG,[38]) system, which is designed to bring people together in person to collaborate using an interactive digital environment. In this research, a multi-touch table was used for this purpose. Because multi-touch tables have a large horizontal surface, they are extremely suited for people to stand or sit around. This makes it possible for users to interact with the table and with the people around it at the same time. The multi-touch surface provides a way for everyone to interact with the system, not just the user controlling the mouse and keyboard. This choice brings us to the main research question:

> How can several structural biologists effectively interact with complex molecular visualizations using a multi-touch table?

In this question, the direct interaction between the users around the table is not included, because the limited number of test users available makes it hard to investigate this area. However, some form of order has to be achieved when several people simultaneously want to operate the multi-touch table. In the context of this study, effective interaction consists of two major components: gestures and multi-user support. The analysis, implementation and testing of these two components resulted in a multi-touch application called TouchMol.

The first objective was to design a multi-touch gesture set that allows individual users around the table easy-to-use access to the functionality of the molecular visualization software. This task was answered using the following sub-questions:

1. Which functionality of the molecular visualization software is used by structural biologists when examining molecules?
2. Which gestures are suited for structural biologists to access this functionality?

With control for individual users established, the second objective was to add turn-taking support to the system. With multiple people around the table and only one molecule to examine, control conflicts are bound to occur. This leads to the following sub-questions:

1. What are the usage scenarios and control conflicts that occur with multiple users?

2. Which protocols can be applied to accommodate these usage scenarios and solve the control conflicts?

3. What is the opinion of the users about these protocols?

The remainder of this report is organized as follows. This chapter is concluded by a description and motivation of the hard- and software setup used in this research. Chapter 2 describes the work related to the topics of this research: interaction with molecular visualizations, multi-touch gestures and multi-user interaction using digital systems. Chapter 3 describes the design, implementation and testing of the multi-touch gesture set for interacting with TouchMol. The analysis of multi-touch turn-taking is covered in chapter 4, its implementation and testing in chapter 5. The research as a whole is concluded and discussed in chapter 6.

1.1 Hard- and software used

The hardware setup of this research is shown in figure 1.1. The DiamondTouch table is the input device for the users. Using an ordinary desktop PC, the TouchMol Java application is top-projected onto the DiamondTouch using a projector suspended from a stand. The remainder of this section describes and motivates the use of the main hardware and software components in this setup. Section 1.1.1 describes the DiamondTouch multi-touch table; section 1.1.2 describes Jmol, the molecular visualization component of TouchMol.

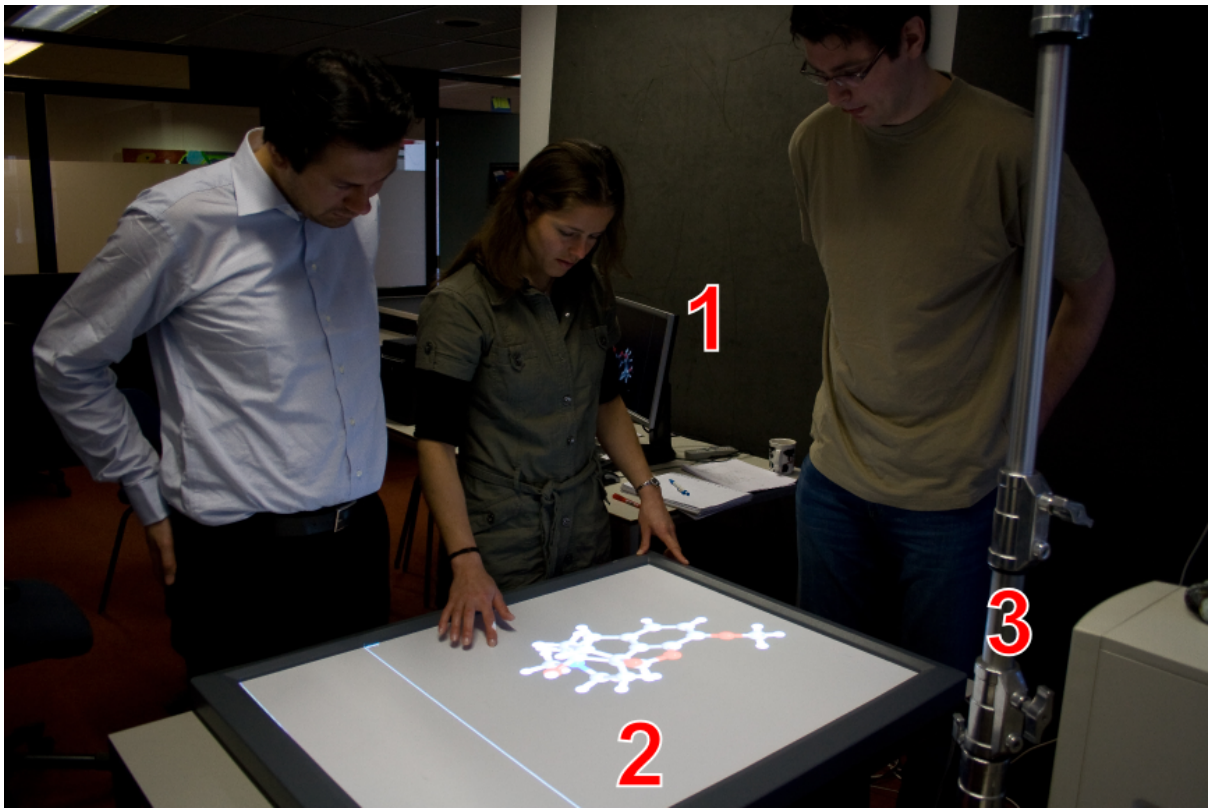

Figure 1.1: Hardware setup of TouchMol. Taken during a user test, this figure shows the PC (1) running TouchMol, which is projected onto the DiamondTouch table (2) using the projector suspended from a stand (3).

1.1.1 DiamondTouch table

The DiamondTouch multi-touch table [3] is essentially a top-projected surface with a large antenna array under the surface to register touch, which sends its input signals to a PC using USB. Alternative techniques for multi-touch are FTIR (Frustrated Total Internal Reflection) and Diffuse Illumination, which use reflection of infrared light into a camera to register touch. More information about these techniques is described by, for example, Hakvoort [9]. The reason why the DiamondTouch was chosen over a FTIR table was because the DiamondTouch is the only table that can detect and differentiate multiple users, something that is not possible using FTIR. When using a DiamondTouch table, the user stands or sits on one of four receiver pads. When the user touches the table, a weak current runs from the antennae through the user's body to the receiver pad (Capacitive Coupling). Because every user has a personal receiver pad, up to four users can be distinguished. Using interpolation, the DT107 model that was used can reach a touch resolution of 2752x2064 on an area of 86x65 cm.

To filter out electrical interference from the environment, the DiamondTouch driver applies a threshold to its input signals. These thresholds can be manually set to a certain level, or an adaptive setting can be chosen. In this research, the adaptive threshold was used. When an input signal is strong enough to pass the threshold, the DiamondTouch table sends out input signals to the PC up to a frequency of 30Hz. On the PC, the DiamondTouch driver converts these signals to input frames in either C or Java. Each input frame contains input of a single user. Among other things, these input frames contain a bounding box of the current touch, as well as the X and Y segments where touch is occurring. An important thing to notice is the fact that the X and Y segments are not synchronized. For example: a frame with two X and two X finger-sized segments can be interpreted as anything between two and four touch points as is shown in figure 1.2. Example interpretations in this picture are {`[X1,Y1],[X2,Y2]`} or {`[X2,Y1],[X1,Y2]`} or {`[X1,Y1],[X1,Y2],[X2,Y2]`}. This ambiguity makes it hard to design multi-finger gestures for the DiamondTouch table, something that is easier to do with FTIR tables.

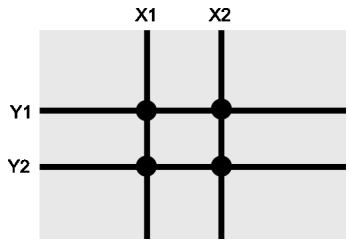

Figure 1.2: DiamondTouch ambiguity.

A general limitation for multi-touch tables is that the finger is the input method with the highest precision. Unlike for instance a mouse or stylus, the finger is not quite suited for precise selection. A possible solution for this problem is given by Olwal et al. [29], who present multi-touch gestures for precise selection by combining pointing and zooming into a single gesture.

1.1.2 Jmol

Jmol [12] is a Java based open-source molecule viewer. Along with the DiamondTouch gesture processing, this application is the most important component in TouchMol. Despite looking less fancy than for instance VMD [14], YASARA [17] and PyMOL [2], Jmol features high rendering quality and speed. The main reason for developing with Jmol besides its rendering quality lies in its accessability. Because Jmol is developed primarily for integration in websites, every action is performed using scripts. The main reasons for choosing Jmol over YASARA and PyMOL is that it is open source. VMD is also an open source application, but is written in C as opposed to Java. Jmol was chosen over VMD, because it was easier to integrate Jmol in a multi-touch application. VMD offers so many option windows to tweak and adjust the display of the molecule, that dialog windows are unavoidable. Jmol does not offer so many options, and nearly all of these options can be selected from its right-click menu. This makes Jmol more suitable than VMD for gesture-based interaction from all sides of the table.

Because Jmol is an application designed for normal PC use, only one user can interact with it simultaneously. With the mouse and the keyboard, the user can adjust the 3D orientation of the molecule using rotation, translation or zooming. These tasks are performed using the mouse, sometimes modified by pressing the Control, Alt or Shift keys simultaneously. the other interactions in Jmol are performed using the right-click menu. By selecting parts of the displayed molecule, the user can also change the display style of the molecule. Often used display styles are cartoon, CPK spacefill and ball and stick. Changes in display style can help the user better understand the composition and function of the molecule. Jmol

can display the names of individual atoms or atom groups, and can also perform angle and distance measurements between atoms.

Chapter 2

Related work

2.1 Molecular visualization

In 2003, Marchese et al. [23] have built a system that allows teleconferencing between two persons using Jmol. Communication between the users happens through a chat window. They present no solutions for turn-taking or input conflicts.

A recent project by Forlines and Lilien [5] described a system that also uses a gesture interface to operate Jmol on a DiamondTouch table. This system is not focused on just the multi-touch table, but also uses wall displays and a tabletPC to operate the more advanced options of the application. Multi-user support is present only in a first-come first-serve protocol. In their project, no user studies have been reported.

Another application for collaborative molecular visualization examination was described by Chastine et al. [1]. They have built a virtual environment, in which multiple remote users can examine visualizations together. This environment is operated using a data glove and viewed with a head-mounted display. The molecules are visualized with the AMMP [13] application.

Van Liere et al. [43] have researched how depth cues should be used to support the study of 3D molecules. They tested their solutions on two systems that both use eye tracking and object tracking to control the molecule. The user holds a small cube that is orientation-tracked to rotate the molecule on the screen. Using a pen that is location-tracked, the user can select or trace parts of the molecules. To increase the 3D experience, the eyes of the user are tracked so the user can truly look past an object. Similar to van Liere's research, Luo et al. [21] present a system that uses a haptic tool with six degrees of freedom to control a stereographic projection of a molecule visualization. While only one person can use the control device, the large size of the projection allows six to ten people to view it simultaneously.

2.2 Multi-touch gestures

In addition to the work by Forlines and Lilien, the developers of the DiamondTouch table at the Mitsubishi Electric Research Laboratories (MERL,[24]) have published numerous other papers about their table. For instance, they have published an article informing about the issues, challenges and opportunities in the design of multi-touch tables [36]. In the article the authors conclude that the multi-touch interface design is still immature, and that there are many questions that need answering to find out how the tabletop form factor can change or contribute to user task performance. Another study is concerned with observations about the behaviour and experiences of people walking up to and using the table in the wild [32]. They conclude that ordinary PC input is not always suitable for multi-touch use. For instance, interface elements that are designed for a mouse do not take text occlusion from hands into account. Yet another study, by Forlines et al. [6] examines performance differences between direct-touch and mouse input with both unimanual and bimanual docking tasks on tabletop displays. The results of Forlines' study indicate that mouse input is faster and more precise for the unimanual interaction, while direct-touch hand gestures perform better for the bimanual task.

A study by Moscovich and Hughes [28] also compares one-handed and two-handed gestures while performing tasks on 2D objects, such as aligning and docking. The core of their study is a kinematic analysis that determines the amount of control and the constraints that are present in the physical (multi-)finger movement. Results indicate that two-touch interaction using two hands is compatible with control of two points, while two-touch interaction using one hand is compatible with control of a position or orientation. An evaluation of several gestures and widgets for orientation control of 2D objects is done by Hancock et al. [11]. They conclude that different situations require different solutions, for instance depending on the amount of precision needed in a task. Unlike most other studies, Malik and Laszlo [22] use two cameras to register gestures on a flat surface, or just above it. With their Visual Touchpad technique, pure 2D gestures can be enhanced using the 3D orientation of the user's hands or fingers.

Wu and Balakrishnan [48] have developed *RoomPlanner*, an interior planning tool which is operated by a wide range of gestures. This is one of the first studies to include hand gestures, not just interaction with fingers. In 2006, a large part of these gestures have been included by Wu et al. [49] in a study about gesture registration, relaxation and reuse. These three techniques are studied to provide a more easy, relaxed and uniform way for users to learn and perform gestures. Wobbrock et al. [47] have performed a study to find out which gestures the users make when confronted with a task, such as copying or deleting an item. They not only collected a large number of gestures, but also classified them to find out which kinds of gestures are performed most often. This resulted in a set of gestures that can be used to control an operating system, with actions such as move, select, drag, minimize/maximize or copy/paste.

Hancock, Carpendale and Cockburn [10] have studied gesture interaction with shallow-depth 3D objects. Shallow-depth means that there is no depth in the environment, apart from the depth of the 3D object itself. They studied gestures to control a small cube with one, two and three fingers. This is one of the few studies to examine 2D gesture interaction with 3D objects. In his PhD thesis [27], Tomer Moscovich also proposes a scheme for interaction with 3D objects, using three fingers to control rotation in all three dimensions. User studies have not been reported, however.

2.3 Collaboration and turn-taking

When there are multiple users around the table interacting with the same object, turn-taking becomes an issue. The most intuitive way of turn-taking happens in conversation. Sacks et al. [33] analyzed turn-taking in natural conversation. By analyzing recorded conversations they identified a model about how turns are constructed (what is said in a turn), how turns are allocated (who determines when a turn transfers), and what the rules are for turn construction, turn allocation and the actual turn transfer. Related to conversation analysis is speech addressing. Van Turnhout et al. [44] have used eye gaze, utterance length and dialog events as non-verbal features to classify the addressee in mixed human-human and human-computer interaction. Their research shows that this classification is very difficult, but can be improved by including more features such as pitch and loudness. An interesting point the authors raise is that as communicative capabilities of computer systems increase, the classification of the addressee will become more and more difficult.

Turn-taking in computer systems was first introduced in *shared view systems*, the predecessors of single display groupware. Pioneered by Douglas Engelbart in 1968 [4], these systems allowed multiple remote people to view the same screen. Typically, one person had control of the screen while others could watch. A historic overview of these systems can be found in a paper by Greenberg in 1990 [7], who also describes the different components and challenges in making a single user application available to multiple remote people. One of these components is the *chair manager*. This component is responsible for enforcing turn-taking protocols and converting the input signals from multi-user to single user. Greenberg mentions a number of floor control or turn-taking mechanisms for this component. With concepts like time-slicing, timeouts, round robin and random access, these mechanisms seem to originate from computer science and network protocols rather than structured human conversation. Greenberg also mentions that no research has been done up to that point in the area of floor control. A year later, Greenberg presents research about roles and group differences in the use of groupware [8]. He discusses these issues in combination with a number of turn-taking mechanisms that have already been applied in groupware systems. These mechanisms will be discussed in chapter 4 of this report. In 1999, Stewart et al. [38] created the Single

Display Groupware (SDG) model, specifically aimed at co-located instead of remote collaboration: *"We define Single Display Groupware to be computer programs that enable co-present users to collaborate via a shared computer with a single shared display and simultaneous use of multiple input devices."* In their research, they mention several benefits and negative effects caused the possibility for turn-taking in SDG, and conclude that these effects should be carefully balanced in order to build a successful SDG application.

The multi-touch table is an example of such an SDG system. A lot of research that is done about multi-touch tables concerns turn-taking and collaboration. Scott et al. [34] present guidelines for designing collaborative work on tabletops. In their work not just the table is taken into account, but also the users, physical objects and environment around it. The design guidelines include advice that tabletop technology should support natural interpersonal interaction, as well as simulaneous user interactions. Another study by Rogers et al. [31] that combines physical objects with a touch table in the collaborative process, shows that extending the table with physical objects can improve collaboration in planning applications by providing two different perspectives on the same problem space. In his PhD research [39] about multimodal co-located interaction, Edward Tse adapted a number of existing single user applications for use on a multi-touch table [41, 42], such as *Warcraft 3*, *Google Earth* and *The Sims*. These applications are operated using a combination of gestures and speech, aimed at collaborative interaction. A focus of the study was to find out how people interact with each other and with the application. The results show that selection tasks become more effective and accurate, and that multimodal commands increase activity awareness between users, especially when working in parallel. Morris et al. [25] have studied collaboration and user preferences when working with centralized or replicated (personal) controls in a photo-sharing application. They report that users like to reserve the center of the table for sharing objects with the group, and prefer not to have controls there. A completely different application area for multi-touch collaboration is shown by Piper et al. [30]. They present SIDES, a game for adolescents with Asperger's Syndrome in which the participants have to cooperate to win. The game is aimed at teaching the students to improve their group skills and to build confidence in social interaction. The game was played with no rules, human-enforced rules or computer-enforced rules. The results of their experiment showed that the game appeared beneficial for the social behaviour of the students, and that the group therapist was able to identify problems in social communication more easily.

When people stand around a table, text or other elements with a directional component will not be displayed correctly for every person around the table. Wigdor and Balakrishnan [46] have performed a study about the readability of text at different angles, and when text should be rotated towards users. As opposed to the general opinion that text should always be rotated towards a user, they found that this is not the case. If maximum reading speed is desired, text rotation is needed. When the text that is displayed consists of short recognizable terms, rotation is not necessarily needed. Kruger et al. [18] argue that orientation of objects on the screen is critical in the coordination of collaborative action. Their observational study of collaborative activities on a traditional table shows that people should have the option to rotate and reorient objects on it. For the DiamondTouch table, Shen et al. [37, 35] have developed DiamondSpin and CoR^{2}Ds (or Context-Rooted Rotatable Draggables), applications that allow users to rotate and translate menus and other directional windows towards them. These systems have the potential to be automated, if the positions of the users around the table are known. A pilot study by Whalen [45] tries to apply board game properties to digital tabletop interaction. In her experiment the users were asked to match photos to their location on a large map of Europe which was only oriented in one direction. The only way to change the orientation was to rotate the entire display. Results show that users are unwilling to rotate the display, as that would hinder the actions of the person across the table. This suggests that rotating a display is not desired for tasks performed concurrently.

Chapter 3

Gestures

3.1 Introduction

The first objective of this research was to define a gesture set to operate Jmol. This set did not have to be built from scratch; in earlier research we already designed a possible gesture set [19]. This set was designed independently, but has remarkable similarity to the one made by Forlines and Lilien [5]. Our set was not verified by users, but rather was based on comparison of literature. In practice it was not a matter of simply evaluating these theoretically motivated gestures. In this chapter, the practical gesture design is documented. The development of the gestures was not an arbitrary process. The gestures had to satisfy three major constraints:

1. The possibilities and limitations of multi-touch tables, specifically the DiamondTouch table.
2. The possibilities and limitations of Jmol.
3. The wishes and requirements of the users.

The first two constraints have already been explored in section 1.1, and are used throughout the design process. The user requirement constraint is perhaps the most important constraint for the research. Throughout the gesture design process, there were three moments where the users could explicitly state their wishes and give feedback, and gestures were always designed with usability in mind.

This chapter describes how the three contraints have been applied in the gesture design process. First the method of the design is explained in section 3.2. This iterative process consisted of a preliminary test, a test for the basic implementation and a second implementation test. Then the results of this method are described in section 3.3, which include a final gesture set. Finally, the results are concluded and discussed in section 3.4.

3.2 Method

3.2.1 Preliminary test

In order to build a good gesture set, the first thing that needed to be done was to find out which tasks structural biologists perform when examining molecules. Because it is hard to correctly classify multi-finger gestures on the DiamondTouch properly (as explained in section 1.1.1), the features that were used most frequently had priority for easy gestures. Features that are not used at all or only on rare occasions were not prioritized. The preliminary test was done by interviewing and observing two structural biology PhD students while they were working with Jmol. To find out about their intuitions and preferences, the biologists were also asked to suggest possible gestures for the functionality they used. This test was performed in an empty computer room, with a beamer present to allow for multi-user viewing of the laptop running Jmol.

3.2.2 Implementation

With the combined input of the structural biologists and that of earlier research, the gestures were implemented to work on the multi-touch table. While designing the gestures, the design philosophy was to use as few buttons or widgets as possible. The main reason for this was that these interface items have to be placed at a specific location, while most gestures can be performed anywhere on the table. Keeping in mind that users can be positioned all around the table, the amount of interface clutter caused by placing buttons and widgets all around the edges would not benefit users. The downside of this choice is that the users have a slightly steeper learning curve. A possible solution for this problem is to integrate a gesture demonstration mechanism. Kirk et al. [16] for instance have demonstrated that projection of a gesture made by a remote or virtual helper can significantly improve task performance.

Using the jdt[1] and DTEvents libraries, the multi-touch table is queried for touch input. With this input, the gestures are processed. The gestures are first classified according to their input type (the number of fingers or non-fingers on the table). This is done using the X and Y segments received from the multi-touch table. With this input type, the current user is entered into a specific mode, for instance rotation or translation mode. While the user makes the gesture in this mode, changes in the gesture type (for instance adding or releasing fingers) are monitored and acted upon. The state diagrams for structuring the processing procedure are documented in appendix A. Because Jmol is a single-user application that uses a mouse and a keyboard for input, the gestures were implemented using a `Robot` from the `java.awt` package. The robot takes over regular keyboard and mouse input to control the application. The mouse pointer can only be present at one location at a time, as opposed to multi-touch input. When multi-touch gestures are used to perform a task that is normally done with the mouse, the input area of the gesture needs to be mapped to a single point. In these situations, the location for the mouse pointer is determined by calculating the center of the bounding box of the current touch frame.

3.2.3 User test

At two stages during the implementation, users were invited to the HMI lab to test the system. Two structural biologists were invited for each of the two sessions. The test of the basic implementation was performed by a scientific programmer and a biology postdoc, who both did not use molecular visualization often (a few times per month). The two PhD students from the preliminary test were invited for the second implementation test. They used molecular visualization tools on a daily basis. To have some form of a turn-taking protocol, a simple single-user limitation was used. This meant that while a user touched the table, no other user was able to make a gesture.

In the first part of the test session the gestures were demonstrated in a working prototype to the biologists, after which they were asked to try them out for themselves and to discuss the individual gestures. If there was more than one gesture for a particular action (such as zooming), the differences between the gestures were discussed with the subjects. The subjects were encouraged to think of ways to improve the gestures. During the test session, the subjects were asked to think aloud. They were also asked questions about the ease of use, comfort and effectiveness of the individual gestures. Observations and followup inquiries were made about errors that occurred, both on human and on computer sides.

The second part of the test session was more focused on observation. In this part, the biologists were asked to examine one or two large macromolecular structures[2] to find out what their functions and properties are. During this time all of the previously learned gestures were used, which provided a good tool for usability feedback. Observations were made as to what the subjects were doing exactly, which gestures they made and whether any mistakes occurred. Another attention point during this part was the occurrence of turn-taking between the two biologists. This information was used in the second part of the research.

After the test session with the multi-touch table, the biologists were asked to fill out a usability questionnaire. This questionnaire in Dutch consisted largely of questions adapted and translated from the USE questionnaire [20]. This is a subjective usability measure that has questions about usefulness, ease of use,

[1] Java DiamondTouch

[2] Molecular structures retrieved from the Research Collaboratory for Structural Bioinformatics Protein Databank (www.rcsb.org): PDB id's `2F8S`, `1AMK` and `1A3N`

ease of learning and satisfaction. Some questions were added to make the questionnaire more specific to the multi-touch table, others were left out. The English version of the questionnaire can be found in appendix B. Because there were only four test subjects, the questionnaire produced no statistically significant user data. Even so, it was used to get an impression of what the subjects thought about the system as a whole. By quickly studying the answers to the questionnaire, extra feedback questions could be posed to the biologists.

3.3 Results

In this section the results of the preliminary test, the implementation and the user tests are presented. We start by showing in section 3.3.1 what the application looks like, and which areas it is composed of. When the general picture is clear, the individual areas of the application are described in detail, including the gestures that can be performed on them. Most of the gestures are also shown as pictures, using the legend shown in figure 3.1. The subsections describe all the gestures that have been tested by the user, and nearly all of them are included in the final gesture set. If a gesture is not included in the final set, this is explicitly mentioned and motivated. The section is concluded by general comments and observations made during the user tests.

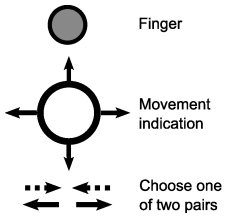

Figure 3.1: Gesture figure legend.

3.3.1 Application composition

Originally, the application was designed to only show a Jmol viewer window that allows the user to manipulate a single molecule. However, the users suggested that storing and recalling the orientation and visual style of a molecule would help them in presentations and examinations of molecular structures. They liked to be able to prepare for presentations by selecting some interesting viewpoints on a molecule, or to store interesting orientations for further investigation during examination. This functionality will be called *bookmarking* in the rest of the report. The bookmarking functionality was added to the application by means of a panel to the side of the Jmol window, as shown in figure 3.2.

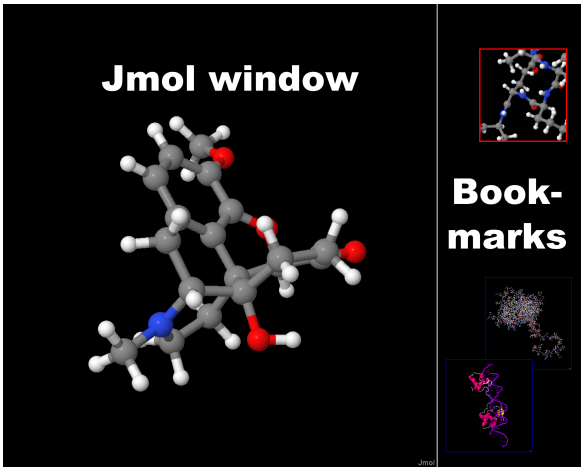

Figure 3.2: Application composition.

3.3.2 Jmol area

From the results of the preliminary test, the key functionality of Jmol was determined. For these parts of the functionality gestures were designed. The main functionality the users used when examining molecules in Jmol was navigation: rotation, translation and zooming. They also used more advanced functionality such as atom group selection and display style changes of these selections. In order to limit the complexity of the gestures, these advanced features did not receive individual gestures. Instead, the right-click menu in Jmol which contains these options was bound to a gesture. This section describes the gestures that give access to this set of functionality. The individual subsections first discuss the design of the gestures, followed by the results of the user tests.

3.3.2.1 Orientation reset

When unexplained or unwanted orientation changes happened during the user tests, the participants needed to have an easy way of undoing the change. This was provided by the orientation reset gesture.

This gesture was not designed based on user feedback, but originated instead from the implementation process of the other gestures. To perform the gesture, the users had to make a fist on the table, which resets the rotation and zoom level of the molecule. The fist gesture is detected on the DiamondTouch by checking for a single large segment in both X and Y directions.

The orientation reset gesture was welcomed by the users. Not only did it provide them with an easy way out if unexplained things happened, it also helped them to get back to an overview of the entire molecule after detailed inspection at a lower level.

3.3.2.2 ArcBall rotation

Often the first thing that users do when examining a molecule, is rotation. Just viewing all sides of a molecule can already give an indication of its properties. In Jmol and many other molecular visualization tools, ArcBall rotation is used. This means that the mouse gesture always rotates the front of the molecule to the back and vice versa. While this provides the potential to view the molecule from all sides, the mouse gesture does not allow for Z rotation only. Z rotation means that the molecule rotates around the axis that is perpendicular to the screen, like a carousel seen from above. In Jmol, this action is performed by either holding down the shift key while dragging sideways or by using the animation or spin functionality. While the initial theoretical idea was to develop a rotation gesture for all three axes at once (as argued in [19]), the user feedback from the preliminary test indicated that this would be too large a step from the application's mouse gestures. Therefore, the two ways of rotation are described in separate sections.

As suggested by the users in the preliminary test, the mouse gesture for ArcBall rotation was chosen as simply a single finger drag (figure 3.3) anywhere in the Jmol area. This mimics the rotation made by the mouse, which is also the easiest way of selecting coordinates on the two axes. There are essentially two ways in which a molecule is rotated. The first is to rotate the entire molecule around the center of its bounding box, the second is to rotate around a specific atom. In order to select which of these two ways to use for rotation, additional gestures were needed. As suggested by the users, the way to select an atom to rotate around was to double tap that atom. When the application detects a double tap, it activates Jmol's center picking mode for a moment, selects an atom and disables center picking again. If the gesture is performed correctly, Jmol shows an animation which moves the selected atom to the center of the screen. Subsequent rotations happen around this atom now, again operated by the single finger drag. Other atoms to rotate around can be selected by again double tapping them. This is slightly different from standard Jmol behaviour. When center picking is enabled there, every single click or tap centers the molecule on the selected atom. Because it is easy to make an accidental tap on the multi-touch table, normal Jmol functionality can cause frustration. The double tap gesture is implemented to be accident proof, while still providing consistency in operation. When the molecule is centered on an atom, Jmol does not easily change back to rotation around the bounding box. For this to happen, the users had to use the orientation reset gesture.

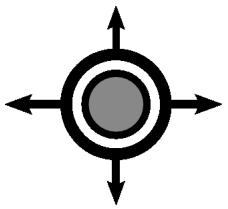

Figure 3.3: Arcball rotation.

The user test confirmed that the single finger drag gesture was easily understood and performed by the users, as it related to the already known mouse drag. This is a good result, because rotation is often the first action that is performed when users load a molecule. However, a small flaw in the gesture occurred when the users examined a zoomed in view of a large molecule. In this situation, a minor finger movement already caused a large enough rotation to cause disorientation to the users. This can be remedied by lowering the speed of rotation when the zoom level becomes bigger. The double tap gesture to center rotation on an atom required a bit of practice, because the taps work best when performed in a relaxed and slow way, unlike the double click on a mouse. In other cases when the gesture was not performed successfully, the main cause was the lack of precision. Especially in larger molecules, selecting a single atom twice in a row proved to be hard. Another problem with this gesture was that the Jmol measurement mode activated when the

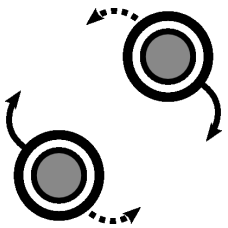

Figure 3.4: Z rotation, twisting.

double tap occurred. This is because the default Jmol functionality is to activate this mode when the user double clicks. This problem can possibly be solved by reimplementation. By querying Jmol for the selected atom and centering on that atom using a script, only one mouseclick should be needed. This requires only one of the two taps to actually hit the targeted atom, resulting in better precision. It would also relieve the user of the measurement mode activation.

3.3.2.3 Z rotation

Z rotation did not have a large priority, as it is an operation that is mostly decorative. This rotation mode is only useful when zoomed out, because when zoomed in on a molecular structure, the notion of direction disappears completely. Because the users in the first user test wanted to perform this gesture and were disappointed when they could not, the decision was made to implement the gesture for the second user test.

The initial idea for the Z rotation gesture was to twist two fingers on the table (figure 3.4), but this proved to be too hard to implement with the restrictions of the DiamondTouch. Instead the gesture was chosen to be two fingers moving on one horizontal line as seen from the user (figure 3.5). This resembles the operation of a horizontal slider.

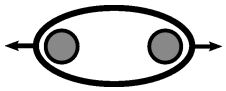

Figure 3.5: Z rotation, sliding.

The Z rotation gesture was only tested by two users, as it was implemented after the first user test. Even so, those two users could not once successfully perform the gesture. For the gesture to succeed, the user had to keep his fingers on a strict line, which proved to be too difficult. The users commented: *"I don't think it's very intuitive to be honest"* and *"It's too similar to the menu gesture"*. The menu gesture (section 3.3.2.6) was used more often during the test, as opposed to Z rotation. The users did not have ideas for another gesture to perform Z rotation, therefore Z rotation was not included in the final gesture set.

3.3.2.4 Translation

After ArcBall rotation, translation along with zooming are most often used by the users. Translation is most often used when the bounds of a molecule exceed the edges of the window when the user zooms in. The translation gesture only translates the molecule in the plane of the table surface, not depth (which is zooming). The first gesture the users in the preliminary test thought of to translate a molecule was to drag the molecule with a flat hand (figure 3.6) or five finger touch (figure 3.7). Another possibility is to 'pick up' the molecule with a five finger touch and place the molecule where you want it with another five finger touch. The latter gesture would not work as well as the first, because the table cannot track the position of a hand while it is off the table. Lack of feedback would then result in confusion with the user. This problem can be solved, but would require additional 3D tracking equipment which is beyond the scope of this research.

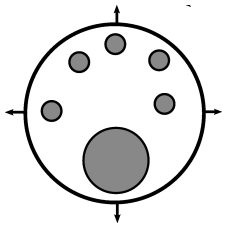

Figure 3.6: Flat hand translation.

The translation gesture can be performed in two ways: dragging with more than two fingers, or sliding a flat hand across the table. The application counts the possible number of finger-sized segments on the table, and activates translation mode when there could be more than two fingers on the table. The flat hand gesture is distinguished by checking for a single large segment in both the X and Y direction, accompanied by multiple finger-sized segments. As long as more than two fingers are touching the table, the translation gesture continues. Releasing or adding of fingers is handled gracefully by letting the robot release the mouse button before calculating the new bounding box center, and picking up the gesture from there. Especially when holding a flat hand in place, the DiamondTouch input becomes unpredictable. Because the signal strength at the edges of contact is very weak, the bounding box often significantly reduces in size for a single frame after which it snaps back.

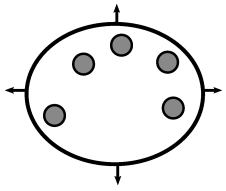

Figure 3.7: Five finger translation.

In order to reduce jittery behaviour when this occurred, a threshold for size and position changes of the bounding box center was introduced.

The users did not like the flat hand gesture as much as the multiple finger translation gesture. The problem they had with the flat hand was the fact that it caused too much friction with the table to perform it gracefully. The multiple finger gesture displayed this problem in a lesser way, specifically when translating away from the user. Sweaty hands and the fairly unfamiliar movement caused the fingers to loosen contact with the table, which sometimes caused the input signal to drop below the table's input threshold. Moving the molecule towards the user did not exhibit this problem. The flat hand gesture also required the users to lean over, since they were standing at the table. This was not comfortable for them, and could also result in occlusion of the beamer projection. The users also found that the application threshold for minimal movement was too high. They often wanted to move the molecule slowly or just a little bit, which resulted in shocking movement of the molecule. This effect can also be explained by the amount of pressure applied to the fingers during the gesture. Because the electrical signal from the table is spread over all the fingers on the table, the signal strength per finger reduces. When too little pressure is applied, fingers can then cease to be recognized.

3.3.2.5 Zooming

Even though the overview of a molecule can tell a lot, the users agreed that zooming is a relatively important action during molecule examination. The examination of small regions with atypical or otherwise interesting atoms or groups can provide key insights into the workings of a molecule.

To zoom in or out on a molecule, the basic gesture is to increase or decrease the distance between two finger-sized touch points. There is no restriction on minimum or maximum distances in the gesture, only in the maximum zoom levels of Jmol. This means that the gesture can be performed with fingers from two hands, or with fingers from the same hand. Results from the preliminary and first user test showed that this gesture is made in practice in three different variations. The first variation is to move both fingers in order to alter the distance between them (figure 3.8). The second variation leaves one finger in the same position, while the other finger alters the distance (figure 3.9). The motion of the moving finger can be repeated (stroking), which results in the third variation. The users in the preliminary test also suggested zooming with the sides of hands (figure 3.10). The problem with this gesture however was similar to that of the flat hand translation gesture: the input signal was fluctuating all the time. The zooming gestures were detected by the application by counting the number of input segments. With two fingers, the touch points can be determined despite the ambiguity of the DiamondTouch checking for three cases: Two fingers next to each other (two X and one Y segments), two fingers above each other (one X and two Y segments) and two fingers at an angle (two X and two Y segments). With the knowledge of the location of the two fingers, the distance between the two fingers can be calculated. Using the `java.awt.Robot` the mouse cursor is placed between the two fingers, the shift key is pressed, and by calculating the difference between the finger distance of the current and previous frame, the mouse can be moved accordingly to accomplish the zooming on-screen.

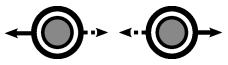

Figure 3.8: Zooming, first variation.

The two finger zooming gesture caused some confusion in the first user test. The intuition of the subjects was not to make the gap between their fingers smaller or larger, but instead tried to zoom by making a movement similar to operating a slider (moving both fingers the same way). After explaining the gesture, they agreed that manipulating the size of the gap was more logical. After trying several times, it was obvious the subjects found it hard to place two fingers on the table at the same time. With the implementation at the time, the zooming gesture did not register in this situation, and nothing happened. This caused some frustration with the subjects. The subjects also often released one finger during the gesture, and expected the gesture to resume when they put it down again. The lack of reactivation also caused some frustration. Because of this, the activation and deactivation of the gesture was relaxed for the second user test. When releasing one of the two fingers, the application returns to rotation mode, because there is one finger on the table. When the second finger is placed again, the application switches back to zooming mode, something that was not done previously. After this change, the mentioned problems did not recur.

Figure 3.9: Zooming, second variation.

In the second user test, the two finger zooming gestures were easily performed. The users commented that when they performed the gesture close to their body, the second and third variation of the gesture were more intuitive. This was because the gesture was made in empty space, regardless of the molecule position. When they zoomed by placing their fingers around the molecule, the first variation was more intuitive for them. In both user tests, it became obvious that the sides of hands gesture was not liked by the users. Because the users were standing during the test, this gesture required them to lean over in order to make it, which required more effort and sometimes obscured the beamer projection. Another problem with it was that the gesture did not offer enough precision to zoom the exact amount the users wanted: *"I'm more accurate when using fingers"*. This was partly caused by the jittering of the zoom level when the gesture was held stationary. The sides of hands gesture was therefore not included in the final gesture set.

3.3.2.6 Right-click menu

In the preliminary test, the users commented that they did not often use the more advanced features of their molecule visualization software, but that they mostly changed the display style of the molecule (for example ball and stick, CPK or cartoon). During the course of testing the application however, it became obvious that nearly all functionality in the right-click menu was used. Selection of specific parts of the molecule was used in combination with display styles to provide a better view of the important parts, while coloring and labeling modes were also used. The right-click menu is also the only place where surfaces can be shown and hidden. This shows that the menu is an essential and frequently used tool in examining molecules, so the priority for a easy and simple gesture for this functionality is high.

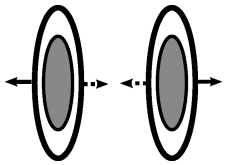

Figure 3.10: Two hands zooming.

In the preliminary test, no real suggestions were made for a gesture to activate the right-click menu in Jmol. The initial gesture to call the menu was taken from DTMouse, the standard mouse emulator for the DiamondTouch. To perform this gesture the user has to touch the location where the click has to be performed, and tap with a second finger next to it. After the first user test, a second gesture was designed by me to be easier and more descriptive of the task. This gesture is performed by dragging two fingers in parallel towards the user (figure 3.11). The idea behind it is that the user gestures where the right-click menu has to be opened. Keeping in mind that users can be at all sides of the table, the menu can be rotated to match the gesture made. Although the functionality for the rotation of the menu was not ready, the gesture was tested anyway. After completing one of the gestures, the right-click menu popped up towards the bottom of the screen, and remained popped up until an option was selected or the user performed a new gesture in the space outside the menu.

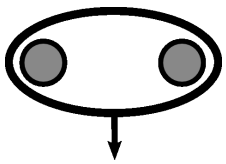

Figure 3.11: Right-click.

During the first user test, it quickly became obvious that the first gesture was not suited for use in TouchMol. When navigating with a molecule, the users often accidentally activated this gesture, resulting in frustration. A quote from one of the test users aptly summarized the gesture: *"When I don't want the menu I get it, and when I want the menu I can't get it"*. The reason why the users could not perform the gesture even when they wanted to was that the gesture did not allow for dragging of the fingers, something that proved to be difficult. Because the gesture did not perform its function properly, this gesture was not included in the final gesture set. Instead, the second gesture was introduced for the second user test. This gesture was easily picked up by the users, and they were able to make it successfully when they wanted. However, the gesture was still performed accidentally during other gestures. This was caused by imprecise activation constraints in the implementation. In both user tests, the users were afraid the menu disappeared when they released their fingers after making the gesture. This resulted in them keeping one finger in contact with the table, dragging it across the menu to select an option. After explaining that the menu would not disappear on release and that they could tap options and submenus, most users were able to operate the menu in a more relaxed way. The size of the options in the menu was large enough to select them without problems. The only times when there were problems with menu item selection were when the table had moved slightly, resulting in a slightly squinted projection. When this happened the cursor would not be placed directly under the

user's finger, which caused some of their taps to miss the intended target. After this happened for one of the users, she then dragged the cursor over the menu to select items. This technique helped her to select the correct options anyway, with the help of the menu option highlighter.

3.3.3 Bookmark panel

The bookmark panel was introduced to give the users the ability to store and recall entire Jmol scenes. This can be used for instance to prepare certain interesting viewpoints to show in a presentation, or to store an interesting detail of a molecule to come back to at a later time. A bookmark is represented by a small screenshot of the scene in question, as shown in figure 3.2.

Because the bookmark panel was only introduced after the first user test, there was no user input on which gestures they wanted to make in it. Therefore, simple gestures were implemented for four different bookmark actions: creating, deleting, moving and restoring. The gesture to create a bookmark was to make a fist in an empty space of the panel. This gesture makes a picture of the current scene, and stores it along with a dump of Jmol's current state. To delete, the user had to make a fist on one or more bookmarks. Moving a bookmark was done by simply dragging it. With a double tap, the state of the bookmark was restored to the Jmol window. The users were able to organize the bookmarks by dragging them, confined to the bounds of the sidepanel.

The user test showed that the gestures were easy to make, though they were not intuitive: *"It's not what I would have guessed, but this is more logical"*. The users had expected drag and drop to work in order to restore or delete bookmarks. However, after showing them to delete bookmarks by planting a fist on the table, they found this gesture fun to make. They also commented that the double tap gesture to restore a bookmark worked like a button. Because one of the users stood quite far away from the bookmark area, she sometimes had difficulties when trying to make a bookmark. It is not clear what caused the gesture to fail, but she had to lean over the table and maybe lost full contact with the mat on the floor in the process. Reaching over could also influence the amount of pressure applied to the gesture, which combined with the lessened contact with the mat could cause the failure. When the two users switched places, the problem was not present anymore. The subjects both thought the bookmark area was a valuable addition to the program, especially for presentations.

3.3.4 General observations

After the test sessions of an hour each, the subjects did not report fatigue in their arms, though standing for such a long time was not very comfortable. In their regular activities, an hour of studying a molecule is not unheard of. But when they work on a molecule that long, even when working on a PC they would take short breaks. The receiver pads of the DiamondTouch can also be used on a chair, so a setting in which the users sit down would also be possible. It is possible that reach becomes a problem in that situation, which could be solved by asking a group member across the table to perform a gesture. The users in the first test were not inclined to sit down: *"It's more practical when you're standing"*.

One of the subjects suggested that for first-time users a window with sliders would make it easier to use the application. A window with sliders for zooming, rotation and translation would be fool proof, but the subjects agreed that with a little more time and effort a gesture interface would be more fun and more efficient to use. This indicates that users are ready for a change from the standard PC environment, if there are fun, efficient and easy alternatives available.

The subjects in the first user test commented that they would not use the multi-touch table when they were examining molecules on their own. They did acknowledge that it had added value when performing this task with two or more people. Neither of the subjects in the second test thought the system would improve their productivity or save them time. They thought the system encouraged them to show more things to others, reducing their productivity. This is not necessarily a bad thing, it is just different from the way they are used to doing their work. This is interesting, because it shows the system is not just a multi-touch adaptation of a single-user application. They commented that the system would be most useful in their educational work.

Jmol area	
ArcBall Rotation	Single finger drag, figure 3.3
Atom center select	Double tap on atom
Translation	Drag with more than two fingers, figure 3.7
	Flat hand drag, figure 3.6
Zooming	Two fingers; both moving away from or towards each other, figure 3.8
	Two fingers; one stationary, one moving closer or away (can also be stroking motion), figure 3.9
Right-click menu	Two fingers next to each other moving in parallel towards the user, figure 3.11
Reset orientation	Fist
Bookmark area	
Create bookmark	Fist in empty area
Delete bookmark	Fist on bookmark
Move bookmark	Single finger drag
Restore bookmark	Double tap on bookmark

Table 3.1: Final gesture set, a legend for the referenced figures can be found in figure 3.1.

3.4 Conclusion and discussion

In this chapter the iterative process of designing, implementing and testing gestures for TouchMol has been described. This process has resulted in a set of gestures, which is summarized in table 3.1. The gestures are used to control both a molecule representation in the Jmol window and the bookmarks in the side panel. The gestures access the functionality that the users looked for when examining molecular visualizations, and augment this with the bookmark functionality. In the user test, the participants were able to execute these gestures to their satisfaction. Observations in the second user test indicated that the system does not just provide an alternative for examining molecular visualizations, but that it could create a new and exciting way of collaboration in this area.

One part of the Jmol functionality was not included in the gesture set: Z rotation. The two presented gestures for this functionality were simply not suited, resulting from difficulties in either the implementation or the usability of the gestures. A possible solution to the Z rotation problem is to introduce another new gesture. Dragging a finger around two stationary ones makes for a gesture that is clearly distinguishable from zooming, though the input ambiguity is a problem there again. From a programming point of view this gesture should be easier to implement, but the question would be if the users still think the gesture is logical enough.

Although the bookmark panel was only introduced in the second user test, it was still welcomed and enjoyed by both users in that test. An improvement to this side panel would be to make its operation function with drag and drop, since that was seemed more intuitive to the test subjects. This would be difficult to implement in Java however, since drag and drop is not supported for all interface elements. Another interesting option is to overlay the bookmarks on the Jmol area, completely removing the need for a separate side panel. This would suddenly make the bookmarks personal, because each user can then store his own bookmarks near his position. This option is also difficult to implement, because the Jmol window itself is built to be always on the foreground. An initial attempt to overlay pictures on the molecule view resulted in the pictures being pushed to the background, removing them from sight. Therefore, this functionality is left for future work.

Chapter 4

Turn-taking analysis

4.1 Introduction

After designing a gesture set, the issue of turn-taking was explored. Because Jmol is built for ordinary PC use, it can be operated by one user (or one keyboard and a mouse) at a time. When used on a multi-touch table, turn-taking problems arise when multiple users are collaborating using the same visualization. In this chapter the goal is to theoretically analyze the issue of turn-taking for single-user applications on multi-touch tabletop surfaces, resulting in a set of turn-taking protocols that can be implemented and tested in practice. Section 4.2 describes how the analysis was performed, followed by the results of the actual analysis in section 4.3. The chapter is concluded in section 4.4 by summarizing and discussing the results of the theoretical side of turn-taking.

4.2 Method

In order to design a suitable set of turn-taking protocols, a thorough analysis was needed. Firstly, we needed to find out what the usage scenarios are for collaborative examination of molecules. The scenarios were conceived using literature and our experiences with the biologists in the earlier gesture tests.

Secondly, we needed to examine the touch conflicts on the multi-touch table that can occur in these scenarios. After all, these conflicts are the reason why turn-taking protocols are needed in the first place. In the literature, only Greenberg [8] and Morris et al. [26] have reported multiple turn-taking solutions in single display groupware systems. Interestingly enough, neither of these two studies describes an analysis of the various touch or control conflicts that can occur in these systems. Morris does categorize the turn-taking solutions in terms of their level of occurrence, e.g. application wide or sub-element conflicts. Forlines and Lilien [5] only briefly touch on the subject, proposing the same solution that was used during the gesture user tests, without motivation. Another multi-user tabletop interaction study by Tse et al. [40] does not even describe how turn-taking conflicts are handled.

With the scenarios and conflicts in place, the existing turn-taking protocols in literature were described and classified. The classification of the protocols included scenario suitability, conflict resolution and stage relevance. Stage relevance describes in which stage of the turn-taking process a protocol applies: turn requesting, turn taking or turn releasing. Finally, the individual protocols are compared and combined to form complete turn-taking solutions.

4.3 Results

This section describes the results of the turn-taking analysis for TouchMol. The section starts off by describing the usage scenarios in section 4.3.1, followed by the touch conflicts in section 4.3.2. In section 4.3.3, the turn-taking protocols in literature are described. Finally, the protocols are compared in

section 4.3.4. Throughout the entire results section, the term *social protocol* is used. This is not a protocol for interacting with the multi-touch table, but the way people interact with one another through conversation. This makes it a protocol for users to specifically interact *without* using the multi-touch table.

4.3.1 Scenarios

The scenarios assume groups of two to four users, as four is the maximum number of users that can interact with a normal DiamondTouch table simultaneously. Observers are not taken into account, since they cannot cause any touch conflicts. To say that observers have no influence in the collaborative process would be incorrect, but examining their interactions is left for future work.

4.3.1.1 Leader

The leader scenario is defined by the presence of an authority figure. This scenario is inspired by Jones et al. [15], in which the authors argue that the use of molecular visualizations in chemistry education is valuable, but only when the students know how to interpret these visualizations properly. Another motivation was provided in the second gesture user test. In this test, a user commented that the system would be beneficial in his teaching activities. A real world occurrence of this scenario is a teacher educating students. In this scenario, the teacher has to be able to control who has the turn. This is an effect of social hierarchy (i.e. respect for the teacher's authority and knowledge) and the ordered structure of the lecture or class.

In a work meeting there is also a leader for the session. This can be a supervisor who calls a meeting with his employees to start a project involving a new molecule. The supervisor assumes the role of meeting chair, and should be able to allocate turns to the participants to keep order in the meeting.

4.3.1.2 Peers

The peers scenario can be used for examining a new molecule without a clear leader among the participants. This scenario has been used in the gesture tests in chapter 3, and we believe it is the scenario that will be the most relevant for molecule examination in a normal scientific work environment. We assume that social protocol is the most important tool for turn-taking in this scenario, as this is more or less an informal meeting. Anecdotal evidence from the first and second user test suggests that this is accurate for two people. The question is if this lack of structure presents a problem when there are three or four people around the table.

4.3.2 Touch conflicts

The turn-taking conflicts that occur on multi-touch tables can be qualified as touch conflicts. The touch conflicts described in this section only address conflicts between two or more users on the same object. In TouchMol this object is the entire screen, which consists of the molecule and bookmark panels. Examples of other objects include for instance a single picture on the screen, or the global state of an application. This section describes the three possible touch conflicts for these kinds of objects.

Simultaneous touch start The first conflict occurs when multiple people start to touch at exactly the same time. This is only a theoretical problem, as the input frames of the DiamondTouch are offered to its listeners in sequence, not in parallel. If the input frames are indeed processed sequentially, this specific conflict does not exist, because the table forces a resolution of the conflict by allowing the frame of one user to be processed first. Therefore, simultaneous touch starts on the DiamondTouch will instead fall in the category of the second conflict, control stealing. On other multi-touch tables without user distinction, simultaneous touch starts could occur.

Control stealing The control stealing conflict occurs when a user starts a touch while another user is already in contact with the table. The second touch can either be accidental (e.g. when pointing something out on the screen), or an attempt to steal the control of the other user. Control stealing is not polite, and a solution for this should be sought in the social interaction between the users rather than being enforced through a turn-taking protocol. Accidental touches should ideally have no influence on the gesture that the other user is already performing. The prevention of this influence is a task for the turn-taking protocol, as social interaction does not apply in this situation.

Short release Sometimes while making a gesture, the user has to momentarily release contact with the table. This occurs when double tapping, or when a user repositions his fingers during a zooming or translation gesture. When a second user touches the table when the touch of the first user is not registered, the short release conflict type occurs. In this situation the turn could be transferred to the second user, depending on the scenario and turn-taking protocol. This should not happen, because the first user is in fact still actively making a gesture. The solution for this conflict would be a turn-taking protocol that does not release the turn automatically when touch is released.

4.3.3 Protocols

There are a number of protocols available to implement turn-taking. The protocols described in this section are proposed by Greenberg [8] and Morris et al. [26]. An important thing to notice is that not all protocols (especially those in the study by Morris et al.) are designed to be a complete solution that addresses every conflict type. To come to a good solution, a combination of multiple protocols is probably the best option. This section describes how the protocols in literature can be adapted for use in TouchMol, and for which scenario or conflict they are a solution. The result of this analysis is summarized in table 4.1.

4.3.3.1 Greenberg

Greenberg [8] describes protocols for single display groupware with remote users. This is essentially about turn-taking with one object, being the entire screen. Besides voice communication and the viewing of the shared screen, no interaction between the users is possible. The situation of turn-taking with one object is actually quite similar to the interaction in TouchMol, where there is only one molecule to interact with. Therefore, the protocols suggested by Greenberg are often complete solutions, without need for combination with other protocols.

Free floor This protocol organizes turn-taking entirely in social protocol, and does not apply any restrictions in the system. Application of this protocol would most likely result in chaos, as both accidental and intentional conflicts could activate unwanted functionality.

Pre-emptive Anyone can take control at any time. This solution would be a little less chaotic than free floor, as the application is only controlled by one user at a time. Even if the user can be changed at any time, the operation will be more orderly. This protocol is suited only for the peers scenario, as there is no way for a leader to enforce turns. This solution only addresses the control stealing conflict.

Explicit release In this protocol, a user has to explicitly release control before someone else can take it. The upside of this solution is that accidental touches of all users except for the active one can be ignored, and they can safely touch the table to point things out. The downside of this solution is that the interface needs additional buttons for the release of turns. The protocol can be combined with other protocols for taking or requesting turns, and can be used in both scenarios. It also addresses all three conflict types.

Pause detection This is essentially implicit release by pausing. The downside of interface clutter of the explicit release solution is not present here. Users can start to interact with the table a short time after the previous user has released the table. Accidental touches can be a problem though, because there can be periods when the turn is unassigned. Similar to explicit release, this protocol can also be combined with other protocols for taking or requesting turns. Pause detection is suited for use in both scenarios, and addresses the control stealing and short release conflicts. A form of control stealing can still be done in this protocol, for instance when a participant is thinking about which option to select in the right-click menu. Another participant can then quickly select something after the pause timer is over. This could be interpreted as impolite, but could also allow for more natural turn-taking interaction during informal meetings. This indicates that it relies more on social protocol than the explicit release solution.

FIFO queue with explicit release Users can request the turn by pressing a button, and are placed in a queue. The user at the head of the queue is assigned the turn, which he has to release explicitly by pressing the button again. In co-located interaction, social protocol could take care of the queue. The leader scenario could make use of the queue however, more so with larger group sizes. In the peers scenario, a queue would probably be too formal and rigid for the scenario. If two people request a turn at the exact same time, the protocol does not provide a solution, although a random function could be used. This protocol does assign turns explicitly, so the control stealing and short release conflicts cannot occur.

Central moderator A central moderator can allocate turns to the participants around the table. This requires extra buttons on the interface. This protocol is suited for the leader scenario, not for peers. Because the turns are explicitly allocated by one person, none of the conflicts can occur. Turn requesting is not included in this protocol, this can be done in social protocol. Alternatively, the FIFO queue protocol can be used for turn requesting. In this case the turn is not assigned automatically, but initiated by the moderator.

4.3.3.2 Morris et al.

Morris et al. [26] introduce multi-user coordination policies for co-located groupware. These protocols are focused on a situation with multiple objects on the table. Global state changes that affect every user are still possible in this situation, so not every protocol is aimed at solving same-object conflicts. Many of the protocols described by Morris are concerned with solving conflicts on a single object, which is the type of protocol that is suitable for adaptation and implementation in TouchMol. However, Morris does describe some protocols that depend on multiple (and often personal) objects on the table, or are otherwise unsuitable for implementation in TouchMol. Therefore, they will not be described in detail or be included in the results table (table 4.1).

No touches This protocol regulates that the turn can only be changed if nobody touches the table. If this is the case, the turn can be taken by either pressing a button (explicit) or just by starting a gesture (implicit). This protocol is suited for the peers scenario only. It only solves the control stealing conflict.

Rank The rank protocol allows participants of higher rank to steal control of lower ranked participants, but not vice versa. This is a solution for all three conflicts, if there are no users with the same rank. However, if a higher ranking person takes over control the previous user could become frustrated. The rank protocol is suited to be implemented for the leader scenario.

Anytime/public These protocols rely heavily on social protocol. They are analogous to free floor and pre-emptive by Greenberg, in the previous section.

	Stage		Scenarios		Conflicts		
Protocol	Request	Release	Leader	Peers	Same time	Control steal	Short release
Free floor		✓					
Pre-emptive		✓		✓		✓	
Explicit release		✓	✓	✓	✓	✓	✓
Pause detection		✓	✓	✓		✓	✓
FIFO queue	✓	✓	✓	✓		✓	✓
Central moderator		✓	✓		✓	✓	✓
No touches				✓		✓	
Rank		✓	✓		✓	✓	✓
Stalemate	✓		✓	✓	✓	✓	
Speed, force	✓	✓			✓	✓	
Dialog	✓	✓				✓	

Table 4.1: This table shows which protocols are used at which stage of turn-taking, which scenarios they apply to and which conflicts they solve.

Stalemate This protocol entails that nothing happens on a direct conflict, and all interaction is blocked until resolved. A positive effect of this solution is that no unexpected things can happen when a simultaneous touch occurs. The user who had control before the conflict would become frustrated though, as his interaction is interrupted. This protocol could be applied in both scenarios, and solves the simultaneous touch start and control stealing conflicts.

Speed, force These protocols use physical force measurements such as drag speed or finger pressure to determine which user wins a conflict. They do address the simultaneous touch start and control stealing conflict types, but are not suited for any of the scenarios.

Dialog On a conflict, this protocol allows the initial user to stop the interrupter by opening a popup dialog box. In this dialog the user can approve or decline the change of the turn. Only control stealing is solved in this solution, as the second touching user can be allowed or denied to take over the turn. This could of course lead to frustration, similar to the stalemate protocol. This solution would not be appropriate for TouchMol, as the use of dialogs and windows is avoided.

Unsuitable protocols *Voting* is used by Morris to allow every participant to have influence on global state changes of the program. There are not many things that require consensus in TouchMol, apart from the recalling of a bookmark. We do not think that voting on screen is necessary, as objections about the action can be handled in conversation. The *private* protocol is meant to prevent users to steal each others objects on the table, and is meant to bring order into systems with many objects on the table. Since this does not apply in TouchMol, the protocol cannot be implemented. The *sharing* protocol allows the owner of an object on the table to designate it as either public or private, again used in systems with many objects on the table. In the *duplicate* and *tear* protocols, the item that is touched by multiple users splits into copies or halves. This is not suited for TouchMol, as Jmol does not allow for this kind of action.

4.3.4 Protocol comparison

The result of the analysis in the previous section is shown in table 4.1. The first column section in the table shows at which stage of the turn-taking process the solutions are used. Turn request and turn release are the two stages included in the table. Turn allocation is a third stage, but this is omitted from the table since every solution addresses this stage. The next column section shows which of the two scenarios the protocols are suited for. The third and last column section shows if the protocols solve the three conflicts in turn-taking.

The criteria for a good protocol in this situation are that it is suited for the scenario, and that it addresses at least the control steal and short release conflicts. These restrictions leave the following options for the leader scenario: Explicit release, pause detection, FIFO queue, central moderator and rank. The peers scenario has the following options: Explicit release, pause detection, FIFO queue and no touches.

The leader scenario would probably work best using the central moderator scenario, since the moderator role is so similar to the leader role. For turn requesting, both social protocol and request queuing can be used. Request queuing reduces the need for communication between the participants of the meeting, but still leaves the turn control with the central moderator. The rank protocol is also suited for a leader scenario, but this solution requires different ranks for all participants to work properly. The explicit release and pause detection solutions also qualify for the leader scenario, but lack control by one user. These solutions are instead used for the more formal approach to the peers scenario. This leaves two solutions for the leader scenario: the central moderator protocol that uses social protocol for turn requesting, and the queue moderator protocol that uses request queuing as an extension of the central moderator protocol.

There are two options for the peers scenario. The first option, pause detection, uses the implicit no touches protocol for turn taking and pause detection for turn release. This option relies on social protocol for turn requesting, but still addresses control stealing during short release. No extra interface elements are needed in this solution, leaving more room for the molecule window. The second option, explicit take and release, uses explicit no touches taking and explicit release of the turn. Social protocol still has to be used to request a turn if someone else is in control. Accidental touches from users without the turn are ruled out in this solution, something that is not always the case in the first option. The FIFO queue is not present in the peers scenario, because this solution works best when there is a leader to manage it.

4.4 Conclusion and discussion

In this chapter, the issue of turn-taking in TouchMol has been analyzed. Comparison of protocols from literature has resulted in four complete protocol solutions; two for the leader scenario and two for the peers scenario. These solutions address the touch conflicts relevant to TouchMol: control stealing and short release.

The pause detection protocol is suited for the peers scenario, and is the most informal protocol of the four. This protocol uses social protocol for turn requesting, implicit no touches protocol for turn taking and pause detection for turn release. The explicit take and release protocol is a more formal solution for the peers scenario. It uses social protocol for turn requesting, explicit no touches protocol for turn taking and explicit release for turn releasing.

The central moderator protocol is the least formal solution for the leader scenario. Social protocol is used for turn requesting, the central moderator protocol (by Greenberg) is used for turn taking and releasing. The most formal solution for the leader scenario is the queue moderator protocol. Both request queuing (FIFO queue) and social protocol can be used for turn requesting, the central moderator protocol by Greenberg is again used for turn taking and releasing.

The expectation is that the central moderator protocol works well in general, but that there will also be some situations in which the protocol is not efficient. An example of such a situation is people helping each other with for instance a menu option. Depending on how important these situations are according to the users, the protocol would have to be changed. The queue moderator is very formal, maybe too formal for the users. This protocol will also require quite a lot of space on the screen for its bars and buttons, reducing the size of the molecule window. The molecule window is expected to be the most important part of the screen to the participants, so they would like it to be as large as possible.

The pause detection protocol in peers scenario is expected to work well with two people. This scenario is similar to that of the user tests in the gestures chapter, and these have shown that conflicts rarely occur. In peer meetings of three and four participants, more conflicts are to be expected, since there is more than one other participant involved in the cooperation. This means that subgroups or subconversations can occur in the interaction, which could cause conflicts when different subgroups want to see different

parts of the molecule. This concern was expressed by one of the users in the previous user tests. If these conflicts do occur, the solutions in the leader scenario are expected to be more popular, since a moderator can actively contain fragmentation of groups or conversations.

Chapter 5

Turn taking implementation and testing

5.1 Introduction

With the results of the theoretical turn-taking analysis, the protocols were ready to be implemented for testing in practice. Contrary to the gesture tests, the application had to be capable of simultaneously receiving input in the turn-taking areas as well as the molecule and bookmark windows. Additionally, a test had to be prepared that allowed the test participants to evaluate the scenarios and compare the protocols. This chapter begins by describing the method of the implementation and the design of the test in section 5.2, followed by their results in section 5.3. Finally, the practical process is concluded and discussed in section 5.4.

5.2 Method

5.2.1 Implementation

Of the four protocols for turn-taking, three of the solutions need additional buttons in the interface to support them. These buttons are placed in panels along the edges of the screen. Because turn allocation and requesting should be possible simultaneous with the control of the molecule and bookmark sections, the mouse robot cannot be used for the control of the panels. Instead, the touches in the panels are processed by calculating which interface elements are touched. First, each panel is checked to find out if the center of touch lies within its bounds. The resulting panel then checks if there is a user button on the location of the touch, and handles the touch of this button accordingly. The details of the different sorts of panels are discussed in the results section.

An abstract class diagram of TouchMol is shown in figure 5.1. Each turn-taking solution has its own subclass of the main TouchMol class, because the layout and functionality of each solution are different. All these subclasses can customize the layout of the application, and they also implement methods to take, release or request the turn. These last three methods implement the functional side of the turn-taking, and they also instruct the individual panels to act accordingly. The turn-taking panels all implement the take, release and request methods as well. These methods are used to visualize the changes in turn. The functionality for the main panels (molecule and bookmark windows) remain in the main TouchMol class.

5.2.2 User test

The goal of the turn-taking user test was three-fold. Firstly, the scenarios had to be verified by the users. A set of protocols is only of use when the usage scenario actually occurs in practice, so the scenarios

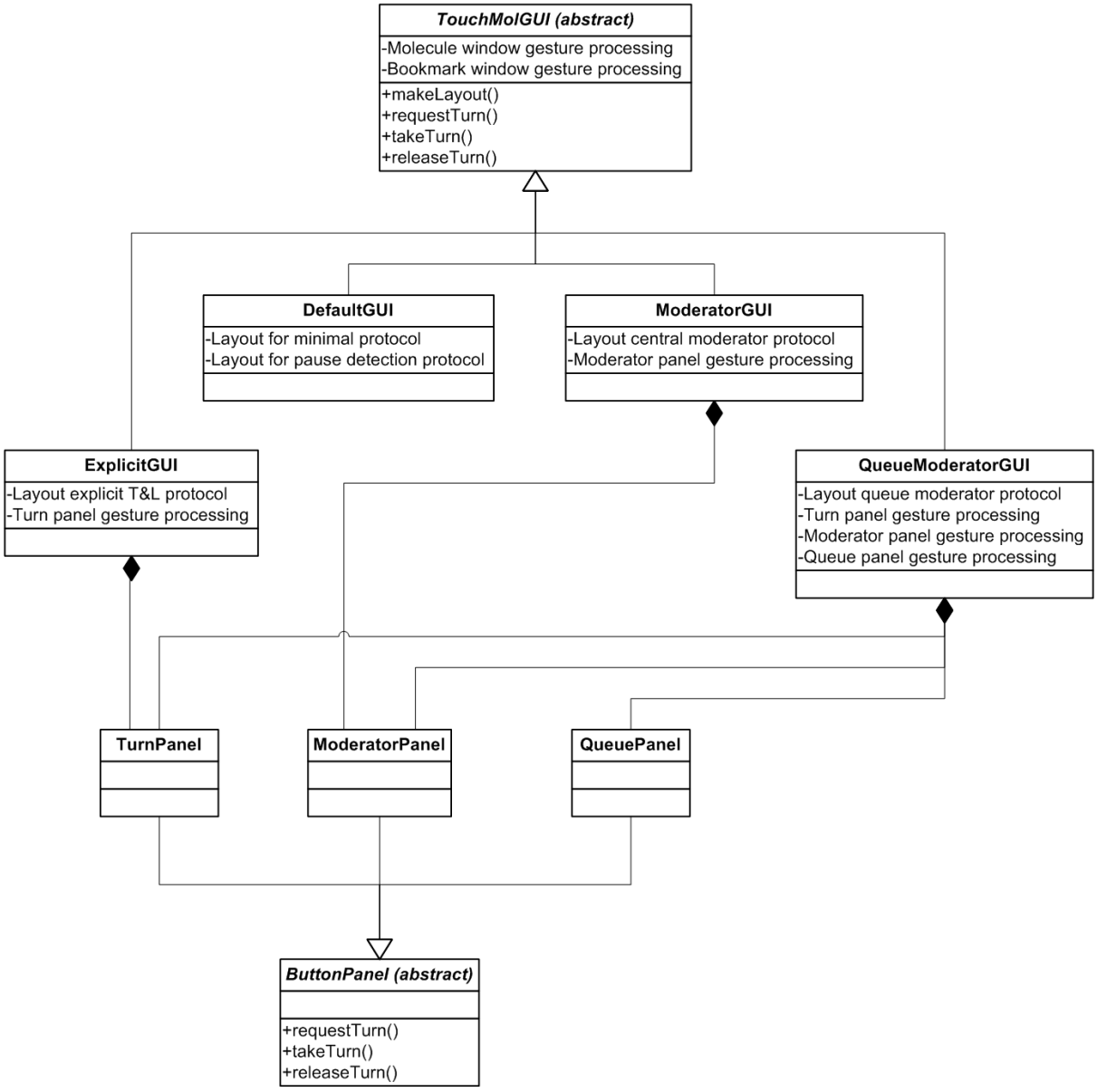

Figure 5.1: Abstract class diagram for TouchMol.

had to be verified. Secondly, the protocols had to be compared in order to find out which of them is best for which scenario. Thirdly, the design and functionality of the new interface elements had to be evaluated. While this is not an evaluation of theoretical results, the application should still be able to function properly, free of bugs and errors.

The turn-taking test was performed with three biologists, two of who were also the participants of the first gesture user test, and all of who worked together on a daily basis. Since none of these users was familiar with the final gesture set, the test was initiated by teaching them these gestures similar to the gesture user tests. This was done to eliminate gesture problems during the rest of the test. Because of the limited availability of test subjects and time, the test was only performed once.

The turn-taking part of the test was performed in two sessions; one for each scenario. In the peers scenario session the users first tested the pause detection protocol, followed by explicit take and release. In the leader scenario session the central moderator protocol was tested first, followed by the queue moderator protocol. For each protocol, a different molecule from the Protein DataBank was used to provide a fresh conversation and analysis start for each protocol: PDB id's `1A3N`, `1AMK`, `6PAX` and `2MRN` respectively. The first two PDB molecules were used earlier in a gesture user test in which the users in this test were not participating. In the leader scenario session, one user was asked to act as the moderator in a meeting. The moderator was also asked to suggest a topic for the meeting scenario test. He chose to examine transcription factors, and requested the `6PAX` and `2MRN` molecules. During the test, observations were made concerning the group behaviour and their turn-taking activity. If unexpected behaviour occurred (both from the participants and the application), the participants were questioned about it immediately.

Both sessions were concluded with a questionnaire, specific to the scenario. These questionnaires are included in appendix C. The questionnaires were not based on previous work, and were focused at finding answers to the research questions of the test. Because of the small number of test participants, the results of this questionnaire are not statistically significant. The questionnaires started by asking the users if the scenarios were appropriately tested. Instead, the questionnaire is used to gain an impression of individual user experiences. The next questionnaire sections concerned the usability and interface aspects of each protocol. In the last section, both protocols were compared to find out which of the two the users liked best. The questionnaires contain both open questions and closed questions on a seven-point scale.

5.3 Results

In this section, the results of the implementation and testing of the turn-taking protocols are discussed. The results are presented according to the design of the test: for each scenario its two protocols are addressed, followed by a comparison. Each protocol is addressed by first describing its implementation and functionality, followed by the observations, user reactions and questionnaire results from the user test.

5.3.1 Peers scenario

5.3.1.1 Pause detection

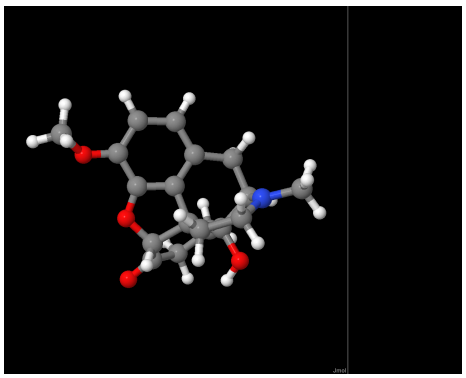

Figure 5.2: Pause detection layout.

Pause detection is nearly identical to the minimal turn-taking protocol used in the previous user tests. While someone is touching the table, nobody can interrupt him. New in this protocol is that the user retains his turn for a predetermined time after touch is released. After this time, the table is free for everyone to claim. If the current user releases the table and retouches it before the pause is over, his control is resumed. This protocol does not have extra interface elements, and looks the same the application used in the gesture tests, as shown in figure 5.2. In this test a pause of one second was used. This length

was chosen to be long enough for a user to perform a double tap without being interrupted, but short enough so users can assist each other easily in selecting a menu option.

The pause detection protocol was easily understood by the users when first presented with it. After a short period of stress testing the protocol in which the users tried to steal each others turn, they began examining the molecule presented to them. While they did this, surprisingly little turn-taking conflicts occurred. When the turn shifted, this was usually done using social protocol. Sometimes however, the turn was stolen by someone selecting an option in the menu that the previous user had opened. This sometimes prompted reactions that the pause duration was too short. On other occasions though, the users claimed that the pause was too long. One of the users commented: *"I think it depends on what kind of group you're in. It also went well for us without the one second [delay]"* and *"In the heat of battle a second might be too short, and if you're calmly discussing something it's not even needed."* In this situation though, the protocol did not interfere with the examination of the molecule, witnessed by the quote: *"I believe we've got more questions about Jmol than about how this works."* Accidental touches were not really a problem during the protocol test, users simply hovered their fingers above the table when they wanted to point something out. Concluding the protocol test, a user mentioned: *"The protocol works fine."*

The questionnaire revealed that the users were mostly positive about the protocol, and liked to use it. The only point of concern was the length of the pause. The users unanimously mentioned this as the worst thing about the protocol; sometimes it was too long, sometimes too short. The best thing about the protocol was that it was easy to collaborate and help each other, according to all three users.

5.3.1.2 Explicit take and release

On startup in this protocol, no user has the turn. The user can take the turn by pressing his user button, located in front of him on a sidebar. This gives the user control of the molecule and bookmark panels, visualized by a white border along the button. If the user is done, another press of the button releases the turn. No other user can gain control of the table while a turn is taken. This protocol uses sidebars along three edges, as can be seen in figure 5.3.

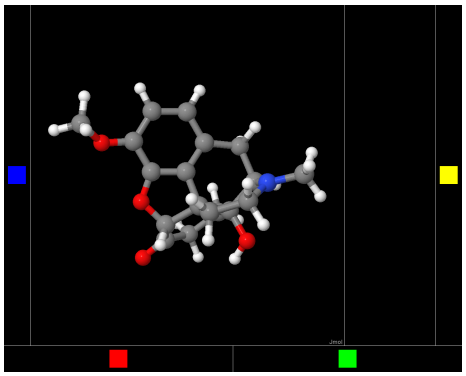

Figure 5.3: Explicit take and release layout.

The additions to the interface of this protocol took the users a bit more time to learn than the simple pause detection interface. The buttons in front of the users quickly led to playful control stealing: by touching another user with one hand and his user button with the other, the turn can be released without the other user's consent. This behaviour was not displayed afterwards. During the examination of the large molecule it was observed that the turn did not change often. The observers asked the user with the turn to show certain things, instead of them asking for the turn themselves. The users commented that they did not feel the need to ask for the turn, as long as the current turn holder knew what he was doing. When turn changes were made, it often occurred when the current turn holder was in conversation instead of operating the table. This caused the turn switch to slow down, as the turn holder often forgot to release the turn when he stopped interacting with the table. One of the users suggested that the turn should be automatically released after a certain pause, for instance ten seconds. Another thing that bothered the users was the reduced size of the molecule window because of the new buttons: *"I think the relative loss of space because of one button is a real pity."*

The questionnaire showed mixed opinions about the protocol. One user remained largely positive, the other two less so. Their main points of criticism were the speed of turn-taking in the protocol and the protocol's unsuitability for improving communication between participants. All three users found the protocol more formal than casual, suggesting that it does not fit too closely with the casual angle of the scenario. Two users mentioned the problematic release of the turn as the worst thing about the protocol, one user disliked the actions and time it took to change the turn in general. The best thing about the

protocol was different for every user. One user liked that it was very clear who had the turn, another liked clearly and explicitly being in charge when he had the turn. The last user liked keeping his turn until he released it, without having to worry about others interrupting.

5.3.1.3 Comparison

The questionnaire clearly showed that pause detection was the user's favourite. The explicit take and release protocol was too formal and problematic with the change of turns, while the pause detection protocol was natural and intuitive. This was supported by a user during the second protocol test: *"I don't think the buttons are needed, I like it the way it was [in the pause detection protocol]."* All three users also agreed with the statement that a larger molecule window is more important than bars and buttons to support turn-taking. One user proposed a general improvement in the turn-taking process; he suggested that a bookmark should be automatically made each time the turn changes.

5.3.2 Leader scenario

5.3.2.1 Central moderator

In the central moderator protocol, one user controls the allocation of turns. This is done using a moderator panel, which contains buttons for every user around the table (figure 5.4). On startup, the blue moderator user has the turn. A press of a user button revokes the turn of the previous user, and allocates it to the new one. The turn is visualized again by the white border around the button of the user that has the turn. The users themselves can only request a turn in social protocol, so they have no buttons available. For the test, the moderator panel was placed on the side of the screen opposite to the bookmark panel.

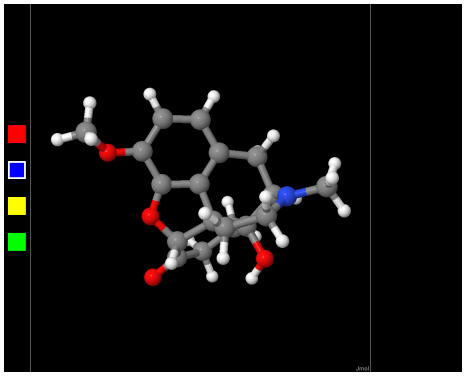

Figure 5.4: Central moderator layout.

The first thing the moderator user did was to start manipulating the PAX-6 molecule that was shown. After a while he asked: *"Why am I the one that has to do everything?"*, after which he was reminded that he had the power to allocate the turn to someone else. The users continued to casually examine the molecule, despite the formal angle of the scenario. They did try to test the protocol, but clearly they were not used to examining molecules in a formal setting. However, the moderator panel did not encourage them to steal control like they did in the explicit take and release protocol. This indicates that the authority of the leader is still being recognized. The placement of the moderator panel caused some discussion; the users thought that the panel could also be placed at the bottom of the bookmark panel. In this way the leader would be on the correct side of the table to be able to see the menu in the correct orientation, as well as operate both the bookmarks and the molecule efficiently. The placement of the moderator panel did depend on the exact role of the moderator, according to the users. They thought that a moderator who merely moderated the turns would be appropriate on the short side of the table, while a moderator with a more hands-on role would be better suited on the long side near the bookmark panel.

In the questionnaire, two users indicated that they did not really simulate a meeting during the test. All users did however think the central moderator protocol was suited for the leader scenario. They also thought the protocol was good for keeping order, and that the leader had enough influence and control. The users were negative about the usage speed of the protocol, and did not think the protocol improved the communication between participants. Two of the users disliked using the protocol (including the moderator user), the other user was neutral about it. None of the users had any complaints about the clarity and functionality of the interface, although the size of the molecule window concerned them. Two users (one of them the moderator) thought the worst thing about was that the moderator really had to pay attention for the protocol to work. This is not necessarily a flaw in the protocol, but is more likely to be caused by the informal atmosphere around the table. In a real meeting scenario, we expect

a moderator to be paying attention at all times. The other user thought that the worst thing about the protocol was the amount of overhead, this was too much for normal use. He thought the best thing about the protocol was the fact that it was very clear who had the turn. The first two users liked the amount of order and the possibility for strict control by the moderator.

5.3.2.2 Queue moderator

The queue moderator protocol allows users to enter a queue, which appears below the moderator panel (figure 5.5). The button of a user in the queue is bordered gray in the panel near the user. In the queue panel, the button is bordered gray with a number in the border to represent the place in the queue. If the user touches his own button again, he is removed from the queue. The moderator can allocate turns both from the moderator panel as well as from the queue panel. Allocating a turn always removes the user from the queue. When this happens, the queue panel is redrawn to show the new state of the queue. This implementation uses two bars to represent the queue and the moderator panel. A possible other implementation is to merge these two bars into one. If a user enters the queue his button on the moderator panel could be changed to reflect this, by adding the grey numbered border. This would eliminate the need for a queue panel, and leave more space for the molecule window. This alternative was not implemented due to lack of time.

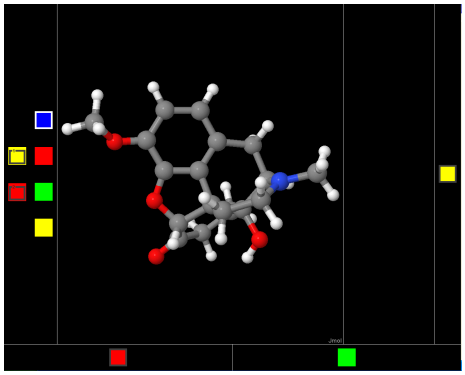

Figure 5.5: Queue moderator layout.

The large amount of extra buttons and functionality in the interface took some time to sink in with the users. The moderator user was surprised that he could pass the turn to a user that was not at the front of the queue, after which it was explained to him that the queue only worked as a suggestion. During the examination of the molecule, the users did not queue often for a turn. Even though their examination was again too casual for the scenario, they agreed that a queue was not really a necessary addition with a maximum of four users around the table. One user commented: *"This is very childish."*

The queue moderator questionnaire showed much the same results as the central moderator version. Users thought that the amount of order and the control and influence of the moderator was good, but the speed of use and the communication between participants left much to be desired. Two users thought the molecule window size was too small in this protocol. One user disliked that the moderator had to pay attention all the time, another user thought again that the protocol was too formal and had too much overhead. The last user thought the worst thing about the protocol was the amount of space the moderator and queue panels consumed, and would have liked to see them at the bottom of the bookmark panel. The users liked the amount of organization and the clarity of turn requesting and assigning best.

5.3.2.3 Comparison

All three users indicated that the central moderator had their preference for the leader scenario. One user stated that the queue moderator protocol was too childish. The other two users liked the fact that the central moderator protocol was simpler and had less overhead, but was still good enough for keeping a tight reign on the meeting. Two users liked social protocol better than a queuing system for requesting a turn, the third user was neutral about it. All three users indicated again that the molecule window size was more important than the presence of bars and buttons for turn-taking. Despite this, one user would have liked a button to release the turn in the central moderator protocol, so he could point things out on the table without hovering.

5.4 Conclusion and discussion

This chapter described the implementation and testing of the theoretical turn-taking protocols resulting from chapter 4. Despite the limited amount of test results, some interesting conclusions can be drawn.

All three participants liked the pause detection protocol best in the peers scenario, and they clearly indicated that the ease of use and the opportunity for helping others are its strong points. The explicit take and release protocol had its strong points, for instance the guarantee of a turn, but was too formal in the end. If a way could be found to incorporate buttons in the interface in a less space consuming way, maybe some of the good points of this protocol could be incorporated in pause detection. The one thing critical to the success of the pause detection protocol is the length of the pause. The test participants sometimes found the pause length too short, and too long on other occasions. This suggests that a possible improvement for the protocol is model with a variable pause length. When the right click menu is activated for instance, the pause does not need to be long when someone is trying to help select a menu item. If another person wants to rotate the visualization while the right click menu is activated, the pause should be longer. Determining the pause lengths for the entire application is a time-consuming process, because every potential turn-taking action needs to be examined in detail. Even so, this change to the pause length model will most likely result in a usability improvement.

In the leader scenario the central moderator was clearly the participants' favourite. It had the easiest controls, and was the least childish of the two protocols. The queue moderator protocol was not necessarily bad, but it would come to rights better in a larger tabletop environment with more participants. The conclusions of the leader scenario can only be seen as indications however, since the participants did not really enact a formal meeting. Perhaps a user test in a more formal business or education environment would yield different results.

An observation that should be considered is that the participants in the test got along well as a group. More formal or controlled turn-taking protocols could be rated better by groups where this is not the case. Testing with groups made up from random participants would be a good way to test this.

The implementation of the turn-taking protocols was quite a difficult task, since the protocols had to be implemented to extend the existing application that did not support simultaneous input (figure 5.1). A better approach to this problem would be to make input handlers in each panel class, including the bookmark and molecule panels. This means that the main TouchMol class is left to deal with directing the input to the proper panels, instead of doing a bit of everything. These changes would result in a more consistent distribution of tasks among the classes.

In the process of implementing the turn-taking protocols, several other paths were explored. One of these paths was an attempt to reduce the impact on the molecule window size when new interface elements were introduced. This would be accomplished by overlaying the bookmarks and buttons on the molecule window. This would leave the molecule window to span the entire surface of the screen, while the extra interface elements would be present in a more integrated way. The reason why this design failed was the molecule window itself. The contents of the window are redrawn every frame, resulting in the overlay panels being pushed to the background when this happened. No solution to this problem could be found, but the idea of such an integrated interface remains appealing.

Chapter 6

Conclusions and future work

6.1 Conclusions

In this report, the objective is to design a way for several structural biologists to effectively examine molecular visualizations together. By combining Jmol, a molecular visualization examination application, and a DiamondTouch multi-touch sensitive tabletop, a multi-user multi-touch Java application called TouchMol was built to achieve this objective. To ensure that the application fulfilled its purpose for individual users, a set of gestures was implemented and tested to control the application. This set featured controls for basic visualization operations such as rotation, translation, zooming and rotation center picking. To enable access to the advanced features of Jmol, a gesture for its right-click menu was added. Jmol's functionality was extended with a bookmark section at the right side of the screen, where molecule orientations can be stored, organized, restored and deleted using gestures. These gestures are summarized in table 3.1. The user tests showed that the test participants were able to control TouchMol with ease using this gesture set.

The next step in the design process was to extend the application to enable smooth operation by multiple users. Since the current Jmol implementation can only be controlled by a mouse and keyboard or a scripting queue, multiple users cannot manipulate visualizations simultaneously. Therefore, a turn-taking solution had to be designed. Analysis of the turn-taking problem and current turn-taking research resulted in two usage scenarios and four sets of turn-taking protocols, two for each scenario. The scenarios are distinguished by means of the presence or lack of a leader. The protocol sets vary in the amount of interface elements they have, as well as the amount of communication needed between the users to switch turns. After implementing these protocols, a user test was performed to compare the protocols for each scenario. In the peers scenario which focused on informal collaboration, the pause detection protocol was the best according to the users. The pause detection protocol lacks buttons for turn-taking, relying almost completely on social protocol for turn-taking. In the pause detection protocol, simultaneous access to the application is disabled and the turn of a user is retained for one second after release of the table. In the leader scenario, which focuses on organized and formal collaboration, the central moderator protocol was rated best by the test participants. This protocol features a moderator panel, which enables the session leader to allocate turns. Social protocol has to be used by the other participants in the session to request a turn, which was not a problem. The leader scenario was not tested in an appropriate session though, so the results of this test are not guaranteed to be correct.

6.2 Discussion

The most important factor in a user-centered research is of course the user. The main problem in this research was the limited number of users available to test the application. A test on a larger scale would preferably be conducted at a university that offers courses in bioinformatics or structural biology. Because of the limited amount of testers, it was not possible to statistically compare the results of the user

tests. Even so, we feel confident that other structural biologists will also be able to operate TouchMol with ease, once the gestures are explained to them.

Besides the users, the DiamondTouch table also played an important role in the design process. Its ability to individually identify each user allowed for easy and straightforward implementation of turn-taking protocols, whereas its limitations in touch recognition imposed a restriction on the gesture design. Without these limitations a wider range of gestures would have been available to implement, but the possibility for user identification more than made up for the touch recognition limitations.

6.3 Future work

One feature of TouchMol that was discontinued during the implementation was variable orientation of text and menus. To improve the usage speed of the application, menus were to be rotated towards the users so they did not have to read the menu items at an awkward angle. The problem with this idea is that Java components are not suited to be rotated at all. Clearly the computer desktop metaphore does not include the horizontal orientation of physical desks. Attempts were made to rotate a swing component by changing the painting orientation of its `Graphics2D` object, but this solution required too much computing power to be feasible. The DiamondSpin toolkit [37] was also examined to enable rotatable popup menus, but for this toolkit to work the entire application had to be rewritten using DiamondSpin classes. This was not possible without changing the Jmol source code, which is quite a daunting task. However, if a solution to this problem can be found, this feature would be a valuable addition to the application.

This research only used the DiamondTouch multi-touch table. As mentioned before, the user identification feature is the main reason for choosing this table. If other multi-touch tables or technologies would incorporate user identification, TouchMol could be adapted to work on those tables. Switching to different tables with different technologies would present new opportunities for gesture design. With the possibility for more accurate gesture recognition that for instance FTIR tables provide, the gesture set can be expanded to allow access to more complex functionality in Jmol.

In the course of the research, it became obvious that while Jmol is an important component of TouchMol, it is not the only important component. According to the users, the bookmark panel provided a valuable piece of functionality that is not present in standalone Jmol. A direction for future research regarding the effective examination of molecules would be to explore other Jmol-expanding features that can increase TouchMol functionality and usability.

A final consideration regarding future work is the improvement of research about collaboration and cooperation on multi-touch tables or other single-display co-located groupware. Current research into this subject seems largely unfounded, something that will have to change if multi-user interaction on these systems is to have a chance of succeeding. Research in this area is most likely a combination of work from different disciplines; social psychology and user interaction in meetings are examples of relevant fields of work.

6.4 Reflection

One of the things I personally enjoyed and learned a lot from, was to study the collaboration between people. Throughout the entire course program of my education, nearly all studies I performed about human-media interaction were between one human and a medium. After seeing the participants of the user tests learn and enjoy working together on the multi-touch table, I believe that systems for easy and enjoyable collaboration are the way forward. This does cause a conflict with the PC's desktop metaphor though. During the course of the research, the vertical single-user centered layout of a normal PC screen started clashing more and more with the horizontal group-oriented multi-touch principles. Maybe some day in the future, the desktop metaphor can be changed to represent what it is: a flat surface where people can stand, sit and work around.

Something that could have been done better was the implementation structure of the application. Because of the strict distinction between gestures and turn-taking in the research process, the necessary

addition of the turn-taking panels and protocols to the application became a problem late in the research. With an earlier realization of this problem, more time would have been spent trying to neatly and consistently organize the application.

Studying molecular visualizations is a task that can benefit greatly from collaboration. Applications such as TouchMol are therefore a valuable tool to improve the performance of this task. Coupled with the further development of multi-touch tables, these applications have the potential to become the first choice for biologists when it comes to molecular visualization examination.

6.5 Acknowledgements

Firstly, I would like to thank my supervisors Paul van der Vet, Wim Fikkert and Jack Leunissen for their guidance, criticism and feedback during the research. I also would like to thank Betsy van Dijk for helping to shape the usability testing, and Clifton Forlines for providing a kickstart with the DiamondTouch implementation and gesture processing. Last but not least, user test partipants Hanka Venselaar, Robbie Joosten, Harm Nijveen and Aalt-Jan van Dijk receive my thanks for answering my questions without tiring, and for showing me all the wrongs I did not want to see.

Bibliography

[1] Jeffrey W. Chastine, Jeremy C. Brooks, Ying Zhu, G. Scott Owen, Robert W. Harrison, and Irene T. Weber. Ammp-vis: a collaborative virtual environment for molecular modeling. In *VRST '05: Proceedings of the ACM symposium on Virtual reality software and technology*, pages 8–15, New York, NY, USA, 2005. ACM.

[2] Delano Scientific LLC. PyMOL, April 2009. http://www.pymol.org/.

[3] Paul Dietz and Darren Leigh. Diamondtouch: a multi-user touch technology. In *UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology*, pages 219–226, New York, NY, USA, 2001. ACM.

[4] Douglas C. Engelbart and William K. English. A research center for augmenting human intellect. In *AFIPS '68 (Fall, part I): Proceedings of the December 9-11, 1968, fall joint computer conference, part I*, pages 395–410, New York, NY, USA, 1968. ACM.

[5] Clifton Forlines and Ryan Lilien. Adapting a single-user, single-display molecular visualization application for use in a multi-user, multi-display environment. In *AVI '08: Proceedings of the working conference on Advanced visual interfaces*, pages 367–371, New York, NY, USA, 2008. ACM.

[6] Clifton Forlines, Daniel Wigdor, Chia Shen, and Ravin Balakrishnan. Direct-touch vs. mouse input for tabletop displays. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 647–656, New York, NY, USA, 2007. ACM.

[7] Saul Greenberg. Sharing views and interactions with single-user applications. In *Proceedings of the ACM SIGOIS and IEEE CS TC-OA conference on Office information systems*, pages 227–237, New York, NY, USA, 1990. ACM.

[8] Saul Greenberg. Personalizable groupware: accommodating individual roles and group differences. In *ECSCW'91: Proceedings of the second conference on European Conference on Computer-Supported Cooperative Work*, pages 17–31, Norwell, MA, USA, 1991. Kluwer Academic Publishers.

[9] Michiel C. Hakvoort. A unifying input framework for multi-touch tables. 10th Twente Student Conference on IT, January 2009.

[10] Mark Hancock, Sheelagh Carpendale, and Andy Cockburn. Shallow-depth 3d interaction: design and evaluation of one-, two- and three-touch techniques. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 1147–1156, New York, NY, USA, 2007. ACM.

[11] Mark S. Hancock, Sheelagh Carpendale, Frederic D. Vernier, Daniel Wigdor, and Chia Shen. Rotation and translation mechanisms for tabletop interaction. In *TABLETOP '06: Proceedings of the First IEEE International Workshop on Horizontal Interactive Human-Computer Systems*, pages 79–88, Washington, DC, USA, 2006. IEEE Computer Society.

[12] Robert Hanson. Jmol: an open-source java viewer for chemical structures in 3d, September 2008. http://www.jmol.org/.

[13] Robert W. Harrison. Stiffness and energy conservation in molecular dynamics: an improved integrator. *Journal of Computational Chemistry*, 14:1112–1122, 1993.

[14] William Humphrey, Andrew Dalke, and Klaus Schulten. VMD – Visual Molecular Dynamics. *Journal of Molecular Graphics*, 14:33–38, 1996.

[15] Loretta L. Jones, Kenneth D. Jordan, and Neil A. Stillings. Molecular visualization in chemistry education: the role of multidisciplinary collaboration. *The Royal Society of Chemistry: Chemistry Education Research and Practice*, 6(3):136–149, July 2005.

[16] David S. Kirk and Danaë Stanton Fraser. The effects of remote gesturing on distance instruction. In *CSCL '05: Proceedings of th 2005 conference on Computer support for collaborative learning*, pages 301–310. International Society of the Learning Sciences, 2005.

[17] Elmar Krieger. YASARA: Yet Another Scientific Artificial Reality Application, October 2008. http://www.yasara.org/.

[18] Russell Kruger, Sheelagh Carpendale, Stacey D. Scott, and Saul Greenberg. Roles of orientation in tabletop collaboration: Comprehension, coordination and communication. *Comput. Supported Coop. Work*, 13(5-6):501–537, 2004.

[19] Jeroen Logtenberg. Designing interaction with molecule visualizations on a multi-touch table. Research for Capita Selecta, a Twente University course, December 2008.

[20] Arnold M. Lund. Measuring usability with the USE questionnaire. *STC Usability SIG Newsletter*, 8(2), October 2001.

[21] Yanlin Luo, Ping Guo, S. Hasegawa, and M. Sato. An interactive molecular visualization system for education in immersive multi-projection virtual environment. In *Proceedings of the Third International Conference on Image and Graphics*, pages 485–488, December 2004.

[22] Shahzad Malik and Joe Laszlo. Visual touchpad: a two-handed gestural input device. In *ICMI '04: Proceedings of the 6th international conference on Multimodal interfaces*, pages 289–296, New York, NY, USA, 2004. ACM.

[23] F.T. Marchese, J. Mercado, and Yi Pan. Adapting single-user visualization software for collaborative use. In *Proceedings of the Seventh International Conference on Information Visualization*, pages 252–257, July 2003.

[24] Mitsubishi Electric Research Laboratories. DiamondTouch, October 2008. http://www.merl.com/projects/DiamondTouch/.

[25] Meredith Ringel Morris, Andreas Paepcke, Terry Winograd, and Jeannie Stamberger. Teamtag: exploring centralized versus replicated controls for co-located tabletop groupware. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 1273–1282, New York, NY, USA, 2006. ACM.

[26] Meredith Ringel Morris, Kathy Ryall, Chia Shen, Clifton Forlines, and Frederic Vernier. Beyond "social protocols": multi-user coordination policies for co-located groupware. In *CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 262–265, New York, NY, USA, 2004. ACM.

[27] Tomer Moscovich. *Principles and Applications of Multi-touch Interaction*. PhD thesis, Brown University, Providence, Rhode Island, 2007.

[28] Tomer Moscovich and John F. Hughes. Indirect mappings of multi-touch input using one and two hands. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1275–1284, New York, NY, USA, 2008. ACM.

[29] Alex Olwal, Steven Feiner, and Susanna Heyman. Rubbing and tapping for precise and rapid selection on touch-screen displays. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 295–304, New York, NY, USA, 2008. ACM.

[30] Anne Marie Piper, Eileen O'Brien, Meredith Ringel Morris, and Terry Winograd. Sides: a cooperative tabletop computer game for social skills development. In *CSCW '06: Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, pages 1–10, New York, NY, USA, 2006. ACM.

[31] Y. Rogers, Y. Lim, and W. R. Hazlewood. Extending tabletops to support flexible collaborative interactions. In *TABLETOP: Proceedings of the First IEEE international Workshop on Horizontal interactive Human-Computer Systems*, pages 71–78, Washington, DC, USA, January 2006. IEEE Computer Society.

[32] Kathy Ryall, Clifton Forlines, Chia Shen, Meredith Ringel Morris, and Katherine Everitt. Experiences with and observations of direct-touch tabletops. In *TABLETOP '06: Proceedings of the First IEEE International Workshop on Horizontal Interactive Human-Computer Systems*, pages 89–96, Washington, DC, USA, 2006. IEEE Computer Society.

[33] H. Sacks, E. Schegloff, and G. Jefferson. A simplest systematics for the organisation of turn-taking for conversation. *Journal of the linguistic society of America*, 50:696–735, 1974.

[34] Stacey D. Scott, Karen D. Grant, and Regan L. Mandryk. System guidelines for co-located, collaborative work on a tabletop display. In *ECSCW'03: Proceedings of the eighth conference on European Conference on Computer Supported Cooperative Work*, pages 159–178, Norwell, MA, USA, 2003. Kluwer Academic Publishers.

[35] Chia Shen, Mark S. Hancock, Clifton Forlines, and Frédéric D. Vernier. CoR^{2}Ds. In *CHI '05: CHI '05 extended abstracts on Human factors in computing systems*, pages 1781–1784, New York, NY, USA, 2005. ACM.

[36] Chia Shen, K. Ryall, C. Forlines, A. Esenther, F.D. Vernier, K. Everitt, M. Wu, D. Wigdor, M.R. Morris, M. Hancock, and E. Tse. Informing the design of direct-touch tabletops. *Computer Graphics and Applications, IEEE*, 26(5):36–46, Sept.-Oct. 2006.

[37] Chia Shen, Frédéric D. Vernier, Clifton Forlines, and Meredith Ringel. Diamondspin: an extensible toolkit for around-the-table interaction. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 167–174, New York, NY, USA, 2004. ACM.

[38] Jason Stewart, Benjamin B. Bederson, and Allison Druin. Single display groupware: a model for co-present collaboration. In *CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 286–293, New York, NY, USA, 1999. ACM.

[39] Edward Tse. *Multimodal Co-located Interaction*. PhD thesis, The university of Calgary, December 2007.

[40] Edward Tse, Saul Greenberg, Chia Shen, Clifton Forlines, and Ryo Kodama. Exploring true multi-user multimodal interaction over a digital table. In *DIS '08: Proceedings of the 7th ACM conference on Designing interactive systems*, pages 109–118, New York, NY, USA, 2008. ACM.

[41] Edward Tse, Chia Shen, Saul Greenberg, and Clifton Forlines. Enabling interaction with single user applications through speech and gestures on a multi-user tabletop. In *AVI '06: Proceedings of the working conference on Advanced visual interfaces*, pages 336–343, New York, NY, USA, 2006. ACM.

[42] Edward Tse, Chia Shen, Saul Greenberg, and Clifton Forlines. How pairs interact over a multimodal digital table. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 215–218, New York, NY, USA, 2007. ACM.

[43] R. van Liere, A.J.F. Kok, J.B.O.S. Martens, and M. van Tienen. Interacting with molecular structures: user performance versus system complexity. In E. Kjems and R. Blach, editors, *11th Eurographics Workshop on Virtual Enviroments*, pages 147–156. Aalborg University, Denmark, 2005.

[44] Koen van Turnhout, Jacques Terken, Ilse Bakx, and Berry Eggen. Identifying the intended addressee in mixed human-human and human-computer interaction from non-verbal features. In *ICMI '05: Proceedings of the 7th international conference on Multimodal interfaces*, pages 175–182, New York, NY, USA, 2005. ACM.

[45] Tara Whalen. Playing well with others: Applying board game design to tabletop display interfaces. In *ACM Symposium on User Interface Software and Technology*, pages 4–5, 2003.

[46] Daniel Wigdor and Ravin Balakrishnan. Empirical investigation into the effect of orientation on text readability in tabletop displays. In *ECSCW'05: Proceedings of the ninth conference on European Conference on Computer Supported Cooperative Work*, pages 205–224, New York, NY, USA, 2005. Springer-Verlag New York, Inc.

[47] Jacob O. Wobbrock, Meredith Ringel Morris, and Andrew D. Wilson. User-defined gestures for surface computing. In *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, pages 1083–1092, New York, NY, USA, 2009. ACM.

[48] Mike Wu and Ravin Balakrishnan. Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. In *UIST '03: Proceedings of the 16th annual ACM symposium on User interface software and technology*, pages 193–202, New York, NY, USA, 2003. ACM.

[49] Mike Wu, Chia Shen, Kathy Ryall, Clifton Forlines, and Ravin Balakrishnan. Gesture registration, relaxation, and reuse for multi-point direct-touch surfaces. In *TABLETOP '06: Proceedings of the First IEEE International Workshop on Horizontal Interactive Human-Computer Systems*, pages 185–192, Washington, DC, USA, 2006. IEEE Computer Society.

Appendix A

Gesture processing diagrams

A.1 Molecule area, main gestures

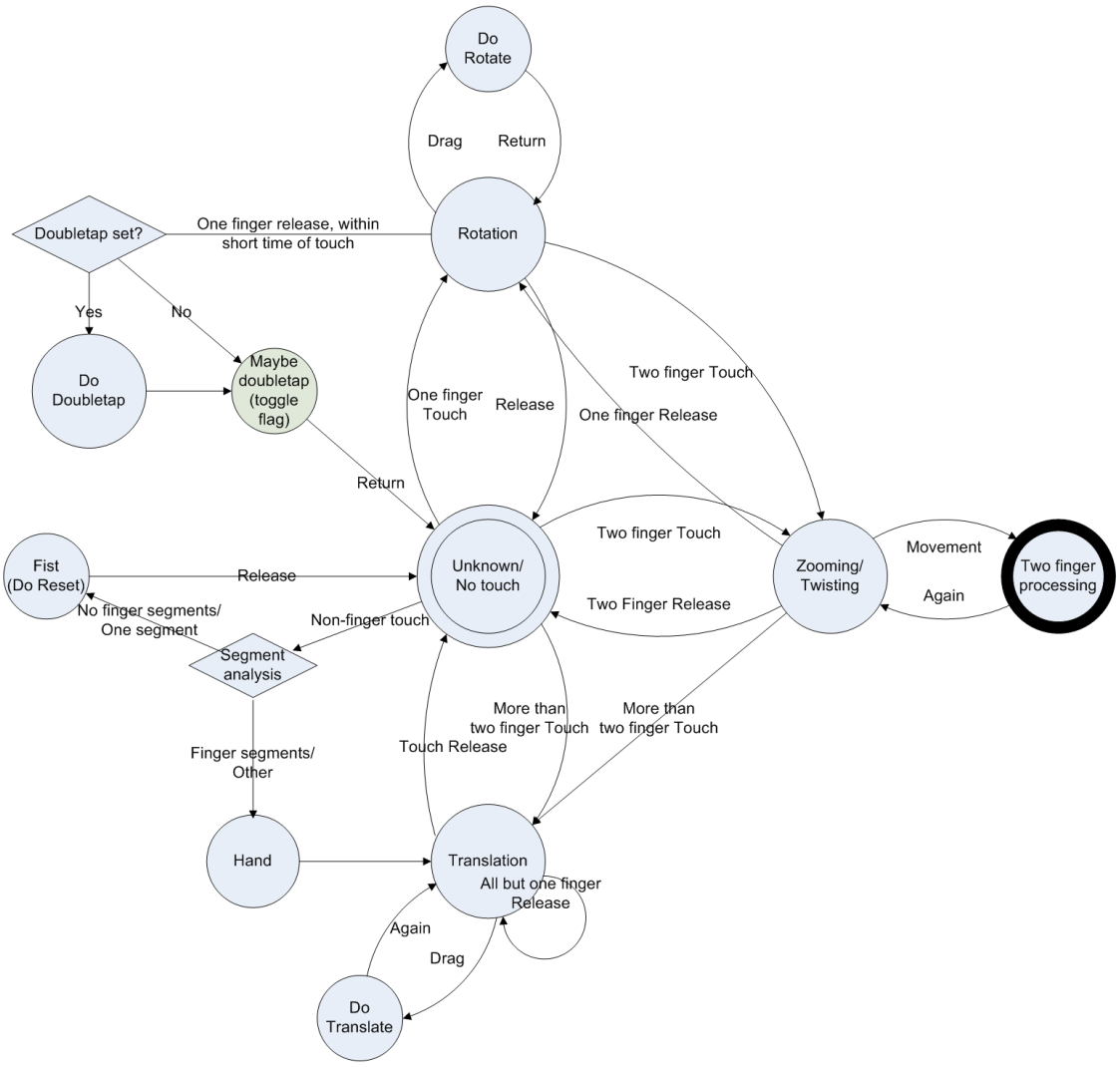

Figure A.1: Main gesture processing.

A.2 Molecule area, two finger gestures

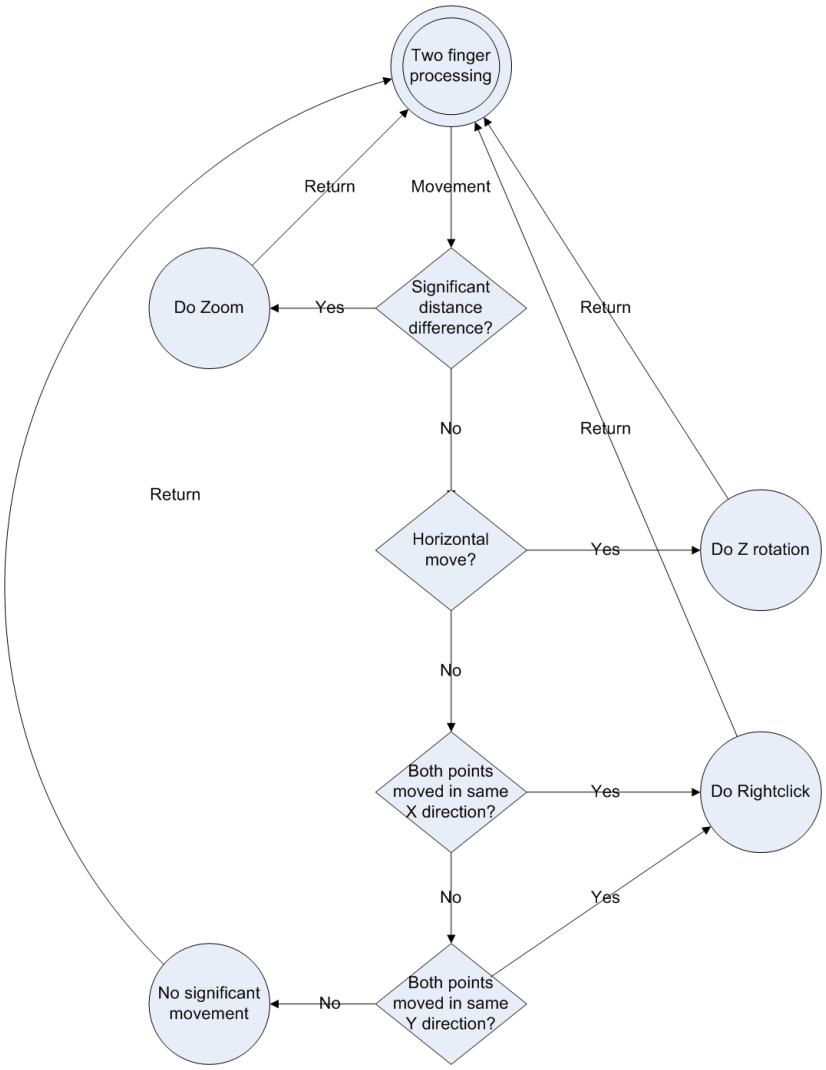

Figure A.2: Two finger gesture processing.

A.3 Bookmark area

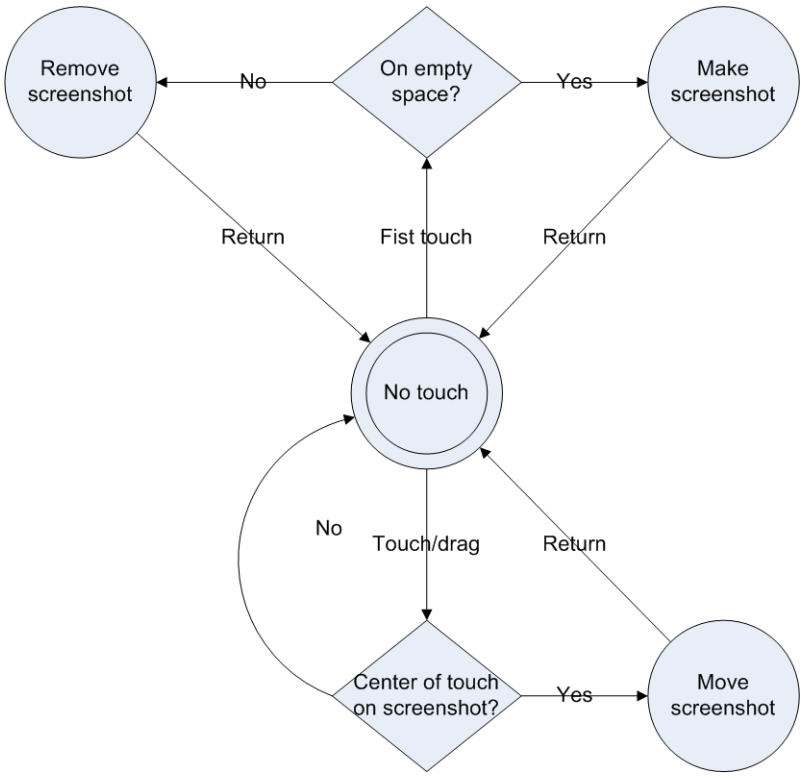

Figure A.3: Bookmark gesture processing.

Appendix B

Questionnaire gesture test

All questions apart from the general and open questions are answered using a seven-point scale: Disagree 1–7 Agree.

General

1. How often do you use molecule visualization tools? (daily/weekly/monthly)
2. How long do you work with molecule visualization tools at a time?
3. Which molecule visualization tools do you use?

Usefulness

4. I can work more effectively with the system.
5. The system makes me more productive.
6. I can achieve things easier with the system.
7. The system saves me time when I use it.
8. The system satisfies my needs when viewing molecular visualizations.
9. The system does everything I would expect it to.
10. I think the bookmark functionality is useful.

Ease of use

11. The gestures in the molecule area are easy to use.
12. The gestures in the bookmark area are easy to use.
13. I don't mind the difference between the gestures in the two areas.
14. I would rather operate a molecular visualization with the multi-touch table than with a mouse.
15. I can quickly achieve what I want with the gestures.
16. The gestures are flexible.
17. I can use the gestures effortlessly.
18. I can use the gestures without written instructions.
19. No strange things occur when I make a gesture.
20. Both occasional and regular users would like the system.
21. I can recover from mistakes quickly and easily.
22. I always perform the gestures successfully.
23. The context menu is organized in an efficient way.
24. The context menu contains the options that I use regularly.
25. Using the system is tiring.

Ease of learning

26. I learned how to operate the system quickly.
27. I can easily remember what the gestures were.
28. It's easy to learn how to operate the system.
29. I quickly became skilfull with the system.

Satisfaction

30. I'm satisfied with the system.
31. I would reccommend the system to a friend or colleague.
32. The system is fun to use.
33. The system works the way I want it to.
34. The system is fantastic!
35. I would like to have this system.
36. The system is is pleasant to use.

Open questions

37. Name three positive points about the system.
38. Name three negative points about the system.
39. Are there things missing in the system.

Appendix C

Questionnaires turn-taking test

Most questions are answered using a seven-point scale: Disagree 1–7 Agree. The other questions are either yes / no or open questions, marked by `(yes / no)` and `(open)`.

C.1 Peers scenario

Scenario

1. The peers scenario is something that happens in practice. `(yes / no)`
2. The tasks that I performed were suited for the peers scenario.

Pause detection protocol

3. The pause detection protocol is suited for the peers scenario.
4. The pause detection protocol improves communication between participants.
5. The pause detection protocol is easy to use.
6. The pause detection protocol is fast in use.
7. The pause detection protocol helps me to retain my turn.
8. The pause detection protocol is casual, not formal.
9. I like using the pause detection protocol.

Pause detection interface

10. The length of the pause is adequate.
11. The lack of buttons in the interface is a good thing.
12. I always know who has the turn.
13. The size of the molecule window is big enough.
14. What is the best thing about the pause detection protocol? `(open)`
15. What is the worst thing about the pause detection protocol? `(open)`

Explicit take & release protocol

16. The explicit take & release protocol is suited for the peers scenario.
17. The explicit take & release protocol improves communication between participants.
18. The explicit take & release protocol is easy to use.
19. The explicit take & release protocol is fast in use.
20. The explicit take & release protocol helps me to retain my turn.
21. The explicit take & release protocol is casual, not formal.
22. I like using the explicit take & release protocol.

Explicit take & release interface

23. The presence of buttons for turn taking and releasing is a good thing.
24. The presence of the buttons does not cause reach problems on other areas of the table.
25. I always know who has the turn.
26. The size of the molecule window is big enough.
27. The size of the turn buttons is good.
28. The turn buttons function properly.
29. What is the best thing about the explicit take & release protocol? `(open)`
30. What is the worst thing about the explicit take & release protocol? `(open)`

Comparison

31. Which protocol do you like best for the peers scenario, and why? `(open)`
32. A bigger molecule display is more important than bars and buttons to support turn-taking.
33. Can you think of a better way of handling turn-taking in the peers scenario? `(open)`

C.2 Leader scenario

Scenario

1. The leader scenario is something that happens in practice. `(yes / no)`
2. The tasks that I performed were suited for the leader scenario.

Central moderator protocol

3. The central moderator protocol is suited for the leader scenario.
4. The central moderator protocol improves communication between participants.
5. The central moderator protocol is easy to use.
6. The central moderator protocol is fast in use.
7. The central moderator protocol helps me to retain my turn.
8. The central moderator protocol is a good solution to keep order in a meeting.
9. The leader had enough influence and control in the central moderator protocol.
10. Social communication is a good solution to request turns.
11. The central moderator protocol is formal, not casual.
12. I like using the central moderator protocol.

Central moderator interface

13. I always know who has the turn.
14. The moderator buttons are on the right edge of the table.
15. The size of the molecule window is big enough.
16. The moderator buttons function properly.
17. What is the best thing about the central moderator protocol? `(open)`
18. What is the worst thing about the central moderator protocol? `(open)`

Queue moderator protocol

19. The queue moderator protocol is suited for the leader scenario.
20. The queue moderator protocol improves communication between participants.
21. The queue moderator protocol is easy to use.
22. The queue moderator protocol is fast in use.
23. The queue moderator protocol helps me to retain my turn.

24. The queue moderator protocol is a good solution to keep order in a meeting.
25. The leader had enough influence and control in the queue moderator protocol.
26. The queue is a good solution to request turns.
27. The queue moderator protocol is formal, not casual.
28. I like using the queue moderator protocol.

Queue moderator interface

29. The presence of buttons for turn requesting, allocating and releasing is a good thing.
30. The presence of the buttons does not cause reach problems on other areas of the table.
31. The moderator buttons are on the right edge of the table.
32. I always know who has the turn.
33. The size of the molecule window is big enough.
34. The size of the buttons is good.
35. The moderator buttons function properly.
36. The turn buttons function properly.
37. What is the best thing about the queue moderator protocol? `(open)`
38. What is the worst thing about the queue moderator protocol? `(open)`

Comparison

39. Which protocol do you like best for the leader scenario, and why? `(open)`
40. Social communication is better than a queuing system for requesting a turn.
41. A bigger molecule display is more important than bars and buttons to support turn-taking.
42. Can you think of a better way of handling turn-taking in the leader scenario? `(open)`