Institut für Informatik
der Technischen Universität
München

Augmented Chemical Reactions – Research on 3D Selection and Confirmation Methods

Patrick Julian Ludwig Maier

Institut für Informatik
der Technischen Universität
München
Fachgebiet Augmented Reality

Augmented Chemical Reactions – Research on 3D Selection and Confirmation Methods

Patrick Julian Ludwig Maier

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:		Univ.-Prof. Dr.-Ing. Nils Thürey
Prüfer der Dissertation:		
	1.	Univ.-Prof. Gudrun J. Klinker, Ph.D.
	2.	Prof. Arthur J. Olson, Ph.D., University of California / USA

Die Dissertation wurde am 15.05.2014 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 13.08.2014 angenommen.

Abstract

User interfaces are important components of a computer system with the user being in control. It is very difficult to build a user interface that fits the needs of a task and most important which is easy to use, understandable and effective. Thus, lots of work goes into research of new and better user interfaces. Whereas the 2D user interfaces of computers have already been there for a long time, 3D user interfaces are still quite new and most of them still lack some of the previous mentioned properties. In my thesis I have investigated and implemented new ways for communicating with the computer when being in a 3D virtual or augmented reality environment.

Evaluations on specific parts of a new user interface always bear the problem of being distorted by other factors surrounding the user interface. I therefore spent much effort on building a flexible system that has an easy to use interface itself which does not distort the results of the evaluations.

This thesis was based on a collaborating project between the Fachgebiet Molecular Catalysis (MolCat Laboratory) of the Chemistry Department and the Fachgebiet Augmented Reality (FAR) of the Computer Science Department here at the Technische Universität München. In this collaboration I built the application *Augmented Chemical Reactions* that lets the user visualize molecules as well as watch how molecules deform in their spatial structure while interacting with them. I have researched how a user can do *Selection* and *Confirmation* while at the same time controlling the pose of the virtual molecules in the 3D space.

An evaluation at a school showed that inspecting different molecules with a direct control of the pose, led to a better spatial understanding of the molecules than controlling the position and orientation of the molecules by using the mouse and keyboard.

Zusammenfassung

Benutzerschnittstellen sind wichtige Komponenten eines Systems, in dem der Benutzer die Kontrolle hat. Es ist sehr schwierig eine Benutzerschnittstelle zu erstellen, welche den Bedürfnissen einer Aufgabe entspricht und – am wichtigsten – einfach zu bedienen, verständlich und effektiv ist. Dies ist der Grund, wieso viel Aufwand in die Erforschung von neuen und besseren Benutzerschnittstellen gesteckt wird. Während 2D Benutzerschnittstellen bereits lange Zeit existieren, sind 3D Benutzerschnittstellen noch relativ neu. Zudem fehlen den meisten von ihnen noch einige der zuvor genannten Eigenschaften. In meiner Arbeit habe ich neue Möglichkeiten erforscht, um mit dem Computer möglichst intuitiv in einer virtuellen oder augmentierten 3D Umgebung zu interagieren.

Studien zu bestimmten Elementen einer neuen Benutzerschnittstelle haben oftmals das Problem, dass die Ergebnisse durch externe Faktoren verfälscht werden, die nicht direkt das Design der Benutzerschnittstelle betreffen. Ich habe daher großes Augenmerk auf die Entwicklung eines flexiblen Systems gerichtet, welches eine einfach zu benutzende Schnittstelle bietet, um die Resultate meiner Evaluationen nicht zu verfälschen.

Diese Doktorarbeit basiert auf einem Kooperationsprojekt zwischen dem Fachgebiet Molecular Catalysis (MolCat Laboratory) des Instituts für Chemie und dem Fachgebiet Augmented Reality (FAR) des Instituts für Informatik der Technischen Universität München. In dieser Kooperation habe ich das Programm *Augmented Chemical Reactions* entwickelt, mit dem Benutzer Moleküle visualisieren können. Weiterhin ist es möglich die Veränderung der räumlichen Struktur von Molekülen zu betrachten, während man mit ihnen interagiert. Ich habe untersucht, wie Benutzer etwas *selektieren* und *bestätigen* können, während sie zur selben Zeit die Position und Ausrichtung virtueller Moleküle im 3D Raum kontrollieren.

Eine Studie an einer Schule hat gezeigt, dass das Betrachten verschiedener Moleküle mit einer direkten Steuerung der Position und Orientierung zu einem besseren räumlichen Verständnis führte, als das Betrachten der Moleküle mit einer indirekten Steuerung mit Maus und Tastatur.

Acknowledgment

Many thanks to numerous great people who supported me in my research as well as in this thesis.

At first I would like to thank Gudrun for having been such a great supervisor of my PhD thesis, as well as for the encouraging words, helpful advice and the comfortable working atmosphere in her lab.

Many thanks also to all of my colleagues Andi, Tayfur, Amal, Marcus, Christian, Manuel, Frieder, Yuta and Manfred of the „Fachgebiet Augmented Reality" who were great to work with.

I would also like to thank Professor Olson from the Molecular Graphics Lab and his colleagues, who gave me a wonderful three month stay abroad in San Diego.

Big thanks go to my parents Geralda and Winfried as well as to Robert and Alex who greatly supported me during this time.

Thank you all so much!

Contents

List of Figures

List of Tables

1 Introduction

With the power of graphics cards and modern computers, three-dimensional (3D) environments become more and more popular. Virtual 3D environments are mostly known from 3D computer games or from animated films. But also research fields such as *Virtual Reality* (VR) and *Augmented Reality* (AR) are based on 3D environments that the user can interact with. To interact with such a three-dimensional environment there is the need of 3D user interfaces. The classical way is to use the well-established mouse and keyboard interfaces to interact with parts of the 3D environment. But as it is not really a natural user interface (NUI) it can become a bit cumbersome sometimes. NUIs which incorporate direct interactions with the environment can be learned and used faster but are not yet commonly available. There still exists a lot of potential to extend, improve and design new 3D user interfaces. Bowman et al. presented an overview of research on 3D user interfaces and its definition [12, 13]. The research on such 3D user interfaces (especially on *selection* and *confirmation*) is the main focus in this thesis. To test and evaluate such 3D user interfaces, I worked in the area of chemistry and developed the application *Augmented Chemical Reactions* (ACR).

Parts of this chapter have already been published in papers [54] and [56] presented at IJAS 2009 and ICCE 2009.

1.1 3D User Interfaces for Chemistry

Designing new molecules is a complex and time consuming task which requires a spatial understanding of the molecules. When scientists create a new molecule like a catalyst, they generally first create a model of the desired molecule in 2D which is then transferred to a 3D model in a computer. The 3D computer representation of the molecule is examined to determine whether the desired structure can be implemented and whether it provides the desired abilities or whether a redesign is necessary. When the design process has settled, the molecule is synthesized in the laboratory and tested for the desired attributes.

Especially the design, the development and the examination of the spatial 3D structure still bears several issues. Even if the structure of the molecule is represented in 3D, examining the molecule from different perspectives requires the designer to interact with conventional computer input devices. Such devices abstract 3D motions to keys on a keyboard or to the 2D space of a computer mouse. This way a mapping from the keys or the two dimensions of freedom of the computer mouse to the six dimensions of freedom (three dimensions for the position and three dimensions for the orientation) of the molecules has to be made. This mapping bears a relatively high mental load and can slow down or distract the user from the real task.

A second issue lies in the changes of the energy distribution of the molecule during its development. The spatial structure changes every time a new atom, ligand or radical is added to the existing molecule structure. Without the help of a computer system that continuously calculates the new structure and properties of the molecules, it becomes hard to predict the change in the structure during assembly.

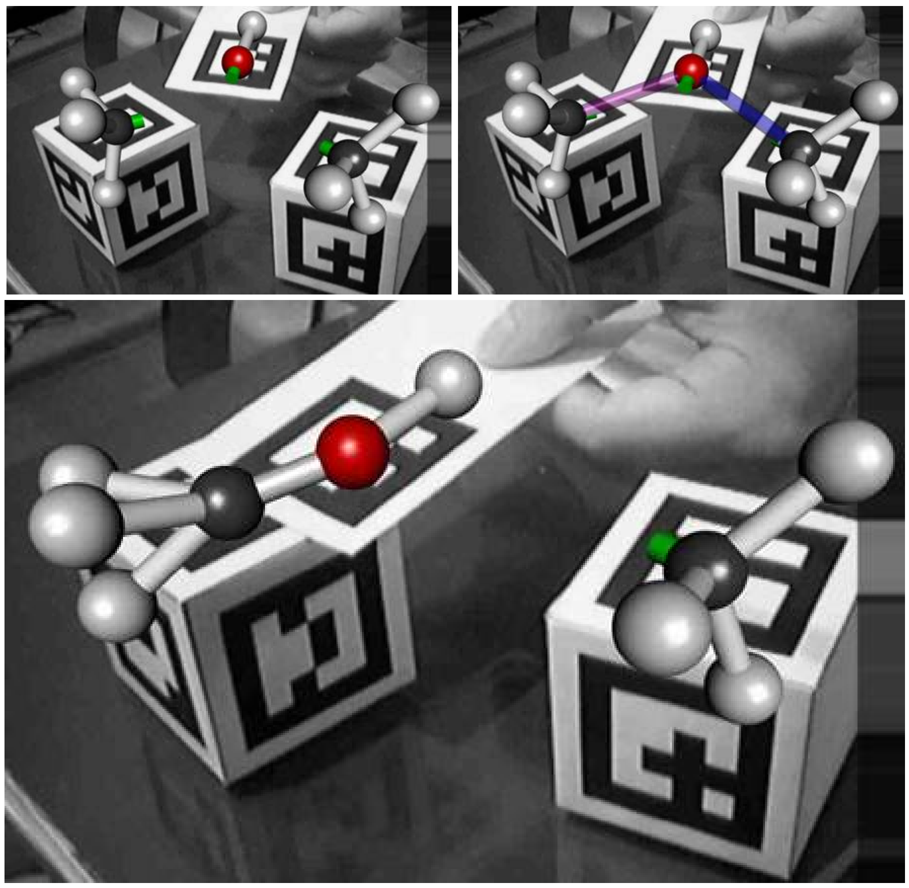


Figure 1.1: AR-based visualization of a dynamic adaptation of a molecule structure to forces from atoms of other molecules.

To support this development process, I implemented a 3D visualization and interaction system which helps users to easily visualize chemical molecules as tangible ball-and-stick models on pre-defined markers. It also simulates the dynamic change of the spatial structure while reacting with other molecules or atoms and visualizes it in an intuitive way (see Fig. 1.1). The system incorporates the paradigm of Augmented Reality (AR) where users can manipulate virtual 3D objects in a simple and intuitive way by moving real 3D objects. With this natural user interface (NUI), users can interactively explore the influence of different molecular properties on a reaction, such as the change of the spatial structure of the molecules during their manipulation, the rigidity of the molecules as well as steric clashes, resulting, for example, in chiral biases. A real-time simulation calculates the attracting and repulsive forces between the reactants of a molecule and can be steered by manipulating the associated tracked, physical objects („tangibles") in the real 3D space. The motion of those tangible placeholders for reactants leads to a dynamic deformation in the structure of the possible molecule product.

With this method, scientists can examine design issues at the early stage of molecular design. The opportunity to also change the viewpoint on the molecule in a 3D interaction space concurrently by using the same tangible handlers, extends the capabilities of the system in comparison to conventional approaches that use mice or keyboards for navigation.

In such „hands-on" explorations and visualizations of chemical behaviors, 3D positioning and timing of user gestures are an essential part for the simulation results. Users need to move molecules to the right place and keep them there while the next steps of a reaction take place. Depending on the number of molecules involved, this may require one or more hands or support structures to hold the tangibles – possibly even a team of researchers, analog to puppetry.

Even though the direct manipulation via tangibles can control some very important aspects of a chemical simulation, many more parameters exist – and even those that are under direct positional control may be susceptible to imprecise user gestures. For example, many bonds between two molecules may be theoretically possible (see Fig. 1.2). If only one is to be selected, which one will it be? If it is selected based on the distance between the atoms involved (Fig. 1.1), this may require users to have a high level of dexterity and the ability to hold very still when non-trivial molecules are involved.

A number of solutions are possible to provide such system control commands, e.g. to select a particular bond out of a list of bonding options, alongside with direct manipulation. I will present and discuss some of them in this thesis.

If one hand can be freed from direct tangible interaction with the molecules, it can be used to control the system via regular mouse or keyboard. It can also be used to interact with widgets that are embedded into the 3D environment [52, 80].

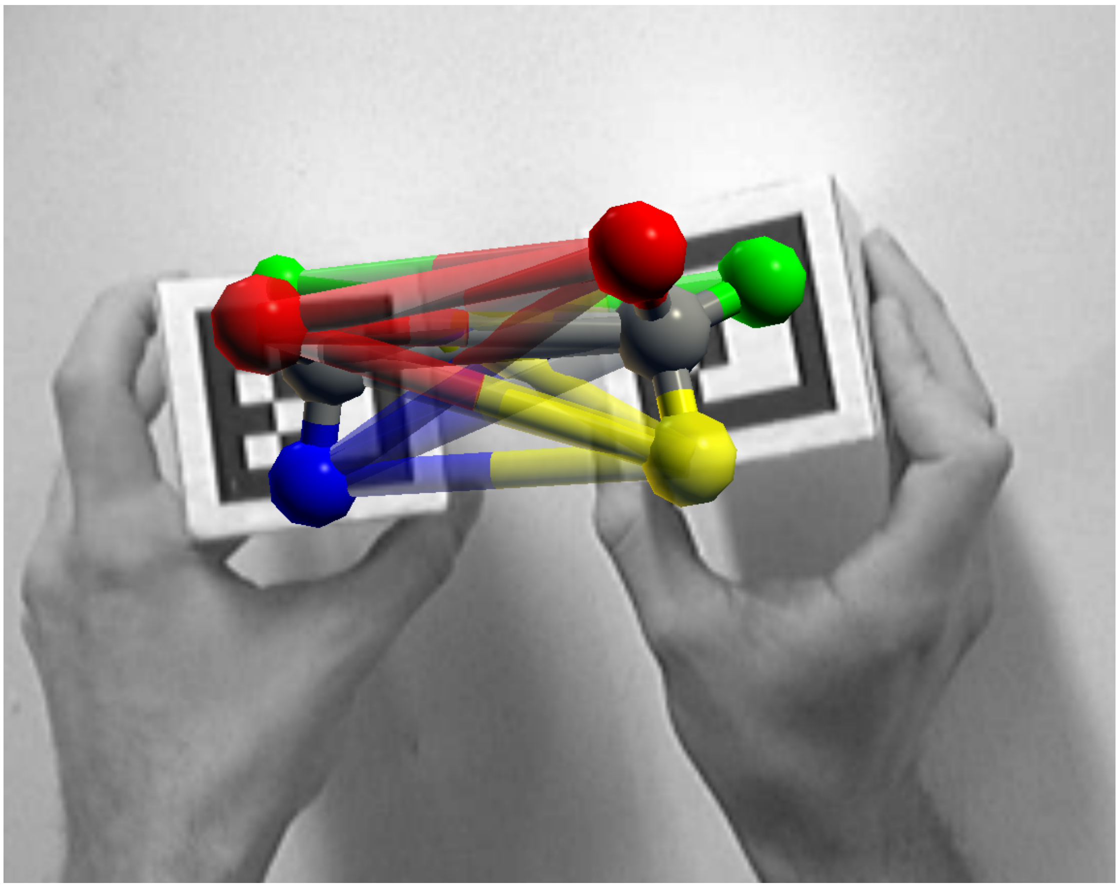


Figure 1.2: Expemplary display of all potential bonds between pairs of atoms on two molecules. The possible bonds are visualized as semi-transparent cylinders between the atoms of the two molecules.

However, if both hands are regularly involved in the direct manipulation of the molecules, system control gestures must be integrated more deeply. To this end, tangibles may be equipped with extra sensors or special buttons – requiring a specialized environmental setting. If no such scene modifications are acceptable, interactive (un)clutching metaphors must be provided that allow users to switch between gestures for direct manipulation (in the true sense of Augmented Reality) and gestures for system control while temporarily „freezing" the manipulated objects. Another alternative to (un)clutching metaphors is to integrate the gestures for system control in the direct manipulation task. Here it becomes very difficult to differentiate between the direct manipulation movements and the movements for the system control gestures. One possible option is using gestures at very different speeds: slow motions for direct manipulation and fast (meta) motions or no movement (See section 5.2.2) for system control. Without (un)clutching metaphors it might still be possible to perform system control along with direct manipulation, but it reduces the options for suitable gestures.

Voice-based input or the use of foot pedals are further interactive options. Yet, it may be hard for users to describe a specific bond concisely with spoken words or with their feet.

To support the users with an easy and understandable user interface, I had to think of input techniques other than the classic keyboard and mouse based user interfaces.

The structure of the following chapters is as follows:

In Chapter 2, I will introduce the state of the art in the field of education to help students in learning as well as teachers in teaching with the help of Augmented Reality. Further I will describe some related work which introduces molecular dynamics simulations in chemistry as well as other applications that visualize molecules on a computer system. As the application (*Augmented Chemical Reactions*) that was used to develop and evaluate all the 3D user interface metaphors, uses a two-handed interaction, section 2.3 will examine work on two-handed interactions. The last section in the background chapter lays its focus on fast user interactions such as shaking.

Chapter 3 Giving an overview of the application *Augmented Chemical Reactions* that was developed and used to investigate the different 3D user interfaces, this chapter describes the basics of Augmented Reality, the integration of the molecular dynamics simulations, the system architecture which provides the application its flexibility and at the end of the chapter the configuration interface.

Chapter 4 To support 3D interaction, there is the need of input devices which deliver 3D poses for the application. Several systems exist ranging from expensive and stationary systems such as the ART tracking system [1] to low cost and portable systems such as the marker tracker from the UbiTrack library of our chair [39] with a webcam. In this chapter I will introduce the different systems and explain another possible input device: TISCH [20].

Chapter 5 is the main chapter and describes the different aspects of 3D user interfaces as well as the research and evaluation on new gestures for selecting and manipulating in the 3D space. For *Selection*, I developed two methods (*Selection by Enumerative Toggling* and *Selection by Proximity*) where both have their advantages and drawbacks. As an improvement I combined both methods into one that has the advantages of each single method while eliminating their drawbacks. The next logical step after selection is the confirmation of a selection. For this task I investigated two methods (*Confirmation by Holding Still* and *Confirmation by Performing a Back&Forth Gesture*) where the first method turned out to be the better solution not only for the interaction with virtual molecules but also for several other fields of application. Especially for the selection and confirmation in a reference-less 3D environment, it is best suited.

Chapter 6 To determine the advantages of the direct manipulation 3D user interface of *Augmented Chemical Reactions* compared to the normal 3D user interface of commonly used molecular visualization applications that use the mouse and keyboard, I conducted a user study with students at a school. It turned out that the direct manipulation user interface helps the students in remembering the spatial structure of the molecules.

Chapter 7 The last chapter wraps up the thesis and gives a higher level conclusion.

2 Background

Augmented Reality applications can support people in many ways. As my focus lies on AR-Applications that support students in learning, as well as support scientists with their work, I will introduce some of the work that has already been done there.

Parts of this chapter have already been published in papers [54] and [56] presented at IJAS 2009 and ICCE 2009.

2.1 AR-Applications for Education and Research

In education, there are still not many AR-Applications which support the students in learning. But there are some that I would like to mention.

In mechanics education, Hannes Kaufmann et al. presented an augmented reality application named PhysicsPlayground [48]. It uses a physics engine originally developed for PC games to simulate real time physical experiments. Here students can build their own experiments and analyze them in a 3D environment. Several properties and parameters of the simulation can be changed to analyze the effect on forces, mass and their paths in real time. By using this immersive virtual environment, new teaching content can be conveyed to the students.

To increase the learning effectiveness in chemistry education, Fjeld et al. have created an AR-toolkit-based chemical education system for children in secondary school [24, 25]. It uses a number of specialized tangible and augmented objects to create and incrementally extend a molecule: an augmented magic book with each page presenting a different chemical element in 3D, special markers to modify the visualization and interaction mode, and a tracked gripper with a button to grab augmented elements from the book and add them to the molecule. When the user pushes the button, the grabbed new element bonds with the closest binding place on the molecule. The molecule floats in mid-air above a workspace that is defined by a movable marker on the table. The marker can be moved, but it does not need to be continuously held in the hand. The molecule can be rotated by another tangible object – a cube – whenever this is shown to the camera. Thus, users' hands are not directly attached to the molecule. Most of the time, they are free to bring in or manipulate any of a number of markers, and also to use the keyboard and the mouse to issue system control commands.

In a later publication [26], Fjeld et al. compared learning effectiveness and user acceptance of Augmented Chemistry (AC) versus the more traditional ball-and-stick model. They reported that users found it hard to mentally switch back and forth between tangible interactions in the augmented workplace and typing on the keyboard, and that users made many mistakes when adding atoms to or removing them from the molecule with the gripper. User preference and rankings, using NASA-TLX [65] and SUMI [87], showed stronger differences and they decided to focus mainly on improving these aspects in a re-design of the AC system. For an enhanced interaction, keyboard-free system configuration, and internal/external database access, a graphical user interface was incorporated into the tangible user interface of the redesigned application. Three-dimensional rendering has also been improved using shadows and related effects, thereby enhancing depth perception. The redesigned Augmented Chemistry system was then compared to the old system by a small qualitative user study. This user study showed an improvement in subjective opinions on the system's ease of use and ease of learning.

Suzanne Weghorst described in a report [90] the progress on an at that time ongoing joint research project between the HIT Lab at the University of Washington and the Molecular Graphics Lab at The SCRIPPS Research Institute (TSRI). They wanted to find new interface tools for teaching and doing research on molecular biology. TSRI's python molecular viewer (PMV) integrated these new interfaces. Complex molecules were printed with a 3D printer and augmented with mixed reality graphics, sound, voice interaction, and haptics. With the help of these tools, new classes were developed to be tested in both high school and college level classrooms. This system was intended to directly create physical 3D models of molecules, which were overlaid with additional information such as electrostatic fields.

In this research at TSRI, Gillet et al. have further developed the Molecular Biology application that combines the use of auto-fabricated (3D-printed) tangible models of biological molecules with AR-toolkit-based [47] Augmented Reality [29]. It uses the previously mentioned in-house chemical visualization and rendering system (PMV) to overlay tracked physical models with different pre-recorded molecular structure representations, with textual chemical information or different representations which can be changed easily by the user. The structural representations are rigid, i.e. they do not change shape when molecules approach each other. A basic animation facility is provided with which users can see the virtual augmentations of the molecules in a number of different, pre-recorded molecular structure representations. Users can use tangible interaction only for the 3D manipulation (translation, rotation) of the molecules.

In this context, further studies have been made by Eliana Medina et al. with these static molecules to help students learn biochemistry [62]. The Molecular Graphics Lab at TSRI also researches on several mobile augmented reality applications [64] to visualize several molecular aspects.

2.2 Dynamics in Chemistry Visualizations

Molecular modeling requires very sophisticated and experienced understanding of the dynamics of the underlying chemical processes. Molecular structures have to be understood and imagined both in their three-dimensional spatial extent and in their dynamic behavior. When planning for chemical reactions between molecules (e.g. when designing a catalyst), chemists have to understand whether the desired result is sterically achievable, i.e. whether there is sufficient space for the molecules to form bonds between targeted sets of atoms on each side. In this respect, it is not enough to consider molecules to be rigid 3D structures. Rather, the forces between atoms need to be taken into account, thereby requiring a complex understanding of the dynamic behavior of all atoms involved in a reaction. Angular relationships between atoms within a molecule are no longer static, but rather depend on the impinging force fields from neighboring atoms from the same molecule, as well as from other molecules during a chemical reaction.

A large number of tools have been developed to help chemists and students visualize molecular structures. We all know the ball-and-stick models that chemistry teachers bring to chemistry classes to help students gain a basic understanding. They are very intuitive because they are real: students can touch and manipulate them. They can move them around and view them from all sides. However, such ball-and-stick models are inflexible. They are unable to change the angular relationships of the bonds within a molecule when new bonds occur during a chemical reaction.

In contrast, various computer-based chemical simulation and visualization tools (e.g., GAMESS [30], TURBOMOLE [86], ADF [77], TINKER [74], HyperChem [40], GaussView [27], Molden [75], Platon [11], Molekel [18], Rasmol [43], and JMol [63]), are able to support chemists in mentally enacting and understanding chemical reactions while taking the influence of inter-atomic forces into account. Yet, the manipulation of the molecules is generally tedious. The results are typically shown as 2D or 3D visualizations. Creation of and interaction with the individual molecules occurs via WIMP[1]-based user interfaces: users use menus, scroll bars and direct 2D manipulation to select molecules, atoms and potential bonds. In comparison to real ball-and-stick models, these simulated visualizations are harder to manipulate since they don't provide 3D handles.

As the work by Fjeld et al. helps in understanding how the structure of molecules are, there is no dynamics of the molecules. To have dynamics in a molecular presentation system, there is the need of an application that simulates the physical properties of the molecules. For this reason there exist molecular dynamics simulations. Those molecular dynamics simulations can simulate the physics of the chemical parts in var-

[1]WIMP stands for **W**indows, **I**cons, **M**enus, **P**ointers in human-computer interaction

ious ways. The most correct way is to use a simulation which implements the electron dense theory [70]. But as those calculations are really expensive in means of computing time at the moment, they cannot be done in real-time, even not on a super computer. A good approximation delivers the mass-spring-model of molecules. Such simulations run in real-time and can also give a good understanding of the dynamics of molecules.

During my stay at the Molecular Graphics Lab [81] of Professor Olson at the SCRIPPS Research Institute in La Jolla, I learned about several molecular dynamics simulation applications. NAMD [71] is a good molecular dynamics simulation application that can run with the mass-spring-model. For my work I used the less complex version Mindy [83] which I extended to be a network server and to be controlled by an AR application.

2.3 Approaches towards Two-Handed Interaction

The paradigm of two-handed interaction gains interest in a wide range of application areas. Not having the need to tell another one which action exactly to perform, reduces communication time and can speed up processes. Often, a second person might not be available, either due to available manpower or due to the increasing demand for keeping project costs low. Investigations on two-handed interaction were made with respect to two classes. The first class distinguishes between the roles which each hand has in a task. Either one hand is the dominant hand and the other is non-dominant or both hands have equal roles. The second class addresses the dimensionality of the space available for task execution, either on 2D surfaces or in a 3D space.

Concerning the differences between the roles of the two hands, Guiard developed a model [31] for the asymmetric division of labor in bi-manual actions. In his *kinematic chain model*, the non-dominant hand is used to coarsely define a spatial reference frame, followed by actions of the dominant hand within this reference frame at higher precision.

Dominant-to-Non-Dominant Spatial Reference The non-dominant hand sets the frame of reference relative to which the dominant hand performs its motions.

Asymmetric Scales of Motion The two hands operate in asymmetric spatial-temporal scales of motion. For instance, when writing on a piece of paper, the motion of the non-dominant hand controlling the position of the paper is of lower temporal and spatial frequency than the writing movements of the dominant hand which nonetheless depends on the non-dominant hand's movement for spatial reference.

Precedence of the Non-Dominant Hand Contribution of the non-dominant hand to a co-operative bi-manual task starts earlier than the dominant hand. In the handwriting example, the dominant hand starts writing after the paper has been oriented and positioned by the non-dominant hand.

This model has been validated, among various others, by Hinckley et al. [36] and Xia et al. [94]. Balakrishnan et al. [9] in contrast determined deviations to Guiard's theory. They built a system for digital tape drawing on a vertical 2D surface. Both hands each held a knob with which the user could define start and end points, and directions on the board. The buttons were used to change the state of the system for drawing lines or curves and for defining the gestures to finally tape the drawing to the board. They tested their system with a set of designers who were easily capable of transferring their working principles to the new interface. In contrast to Guiard's model, the designers used their dominant right hand to define the frame of reference. Also the non-dominant left hand operated at a higher spatial frequency than the right hand. Balakrishnan [9] pointed out, that more analysis and refinements are required to adequately explain human bi-manual interaction.

The setup used for my Augmented Chemical Reactions (ACR) application uses symmetric bi-manual interactions rather than an asymmetric setting. Both hands have equal functionality: 3D manipulation of the position and orientation of two molecular structures with respect to one another.

Casalta et al. [15] investigated differences between asymmetric and symmetric division of labor in bi-manual interaction. They set up a 2D rectangle editing task and found that their test participants revealed better performance and a higher degree of bi-manual parallelism with the symmetrical than the asymmetrical option. They left the question whether Guiard's model still holds for symmetric interaction.

Balakrishnan and Hinckley also investigated symmetric bi-manual interactions [8]. Test users had to track a pair of targets, each controlled with one hand while forcing them to divide attention between the two targets through putting them further apart. They also investigated visual connections between the two targets. Concerning the setup of Augmented Chemical Reactions, these lines can be compared to the possible bonds between two molecule structures. They found that the degree of parallelism is affected by distance and by visual cues.

The systems presented thus far use 2D interaction surfaces. Other systems offer a 3D space for bi-manual interaction.

The work of Pierce et al. [72] uses asymmetric two handed interaction in a virtual environment. The „Voodoo Dolls" system provides facilities to concurrently control the working context of a handled object and its parameters for object manipulation. Grabbed by the non-dominant hand, an object is seamlessly scaled to a useful size for operation. The viewing context is adjusted according to how the two hands are held relative to each other when two objects are held. They founded their setup on the work of Guiard [31]. Evaluations showed that after a phase of familiarization, the test participants had little to no difficulties to arrange objects in a room.

2.4 Approaches using Fast User Interaction (Shaking)

For the symmetric bi-manual interaction in Augmented Chemical Reactions, I distinguish between no, slow and fast user hand motions. Slow motions are attributed to direct molecule manipulation whereas fast motions are used as meta motions to perform system control tasks such as toggling through a selection (5.1) or unclutching a hand from a molecule, i.e. „freezing" the tracking operation. Such unclutching has been used in a number of AR applications [32]. In the context of this thesis, it is more important how such unclutching is achieved than what it is used for. In this case, no hands are free to push a „freeze" button. A lazy implementation in many optical marker tracking algorithms can be used to unclutch the virtual object from the marker: when the tracker loses track of a marker, the virtual object remains at the last known position. Thus, by quickly hiding a marker, a molecule can be suspended somewhere in mid-air. The freed hand can then be used for other interactions that are not linked to the molecule. However, it is doubtful whether ordinary users can be expected to be aware of and exploit the shortcomings of a tracking algorithm when using the system.

Instead, I developed a shake-based method to allow users to temporarily perform actions on a system control level. Shaking gestures have already been used in other AR-based contexts, most recently by White et al. [91, 92] to activate and deactivate menus. In contrast to the shake-based toggling through several options used in this thesis, White et al. used shaking only to pop up a menu. Item selection is not performed by shaking but rather by targeted marker alignment with a menu item – something that cannot be done here since the primary purpose of the tracked objects is the manipulation of the molecules as a whole. White also points out that shaking is becoming a common gesture on mobile phones with built in accelerometers [10, 76]. This indicates that our toggle-oriented use case may be a suitable addition to a growing list of use cases.

When implementing a shaking gesture, it first had to be analyzed how such a movement of the controlled object (i.e. the marker) can be described. White et al. have presented a technique for recognizing a shake gesture by following the path of the tracked marker and transforming it into as sequence of directional units (up, down, left, right, front, back) [91]. They parse the directional information. When they detect four continuous movements in opposing directions, they accept a shaking gesture. Here in contrast, a statistically-based approach (see Section 5.1.1) is used to detect a shake gesture.

3 Augmented Chemical Reactions

As mentioned earlier, I designed and built an application which should facilitate the work of chemists as well as support chemistry students in learning chemistry. To be helpful, the user interface of the application should be designed in a user centric way. Not the user should adapt to the application, but the application should be designed to fulfill the user expectations of the interface.

In addition, the application was used to do research on the 3D user interfaces. In this chapter I will give a short introduction to Augmented Reality and the design and functionality of the application. Parts of this chapter have already been published in paper [56] presented at ICCE 2009.

3.1 Augmented Reality in ACR

Augmented Reality is one of the main techniques used in my application. In general, Augmented Reality interactively adds virtual objects or information to the real environment in real-time to give the impression as if those virtual objects are part of the real world (Definition by Ronald Azuma [6]). In a typical AR system, there exist one or more sensors (trackers) that can determine the position and orientation of physical, real world objects and a camera or human viewer relative to them. With this data, it is possible to overlay the real camera image with virtual objects, rendered by a virtual camera with the same visual properties as the real camera. The most common tracking systems are so called marker trackers. Markers are – for example – black and white patterns with an encoded marker id. With the use of image processing, the marker tracking algorithm determines the position and orientation of the pattern relative to the camera (Figure 3.1). With the computed position and orientation of the marker in the coordinate system of the camera, it is possible to augment the camera image with a virtual object. It further enables the use of the tracked marker as a 3D input device.

In my application, I use the data to render virtual molecules on top of the markers and thus control their position and orientation. Users can easily inspect the spatial structure of virtual molecules by just moving and rotating the markers in an intuitive way. It furthermore provides the opportunity to steer a molecular dynamics simulation which is attached to the visualization system to give further understanding of the chemical behaviors.

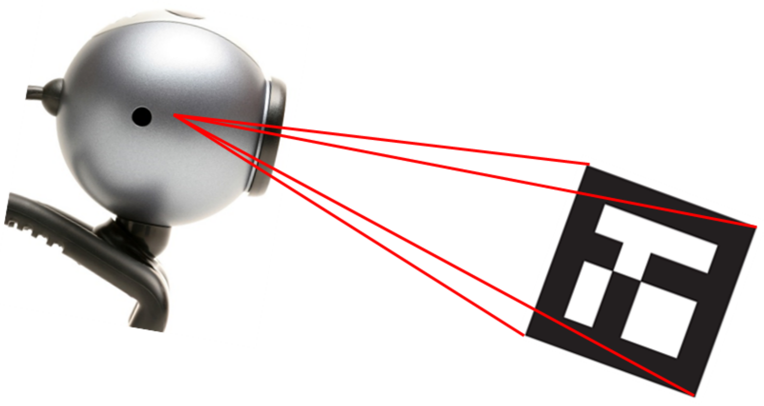


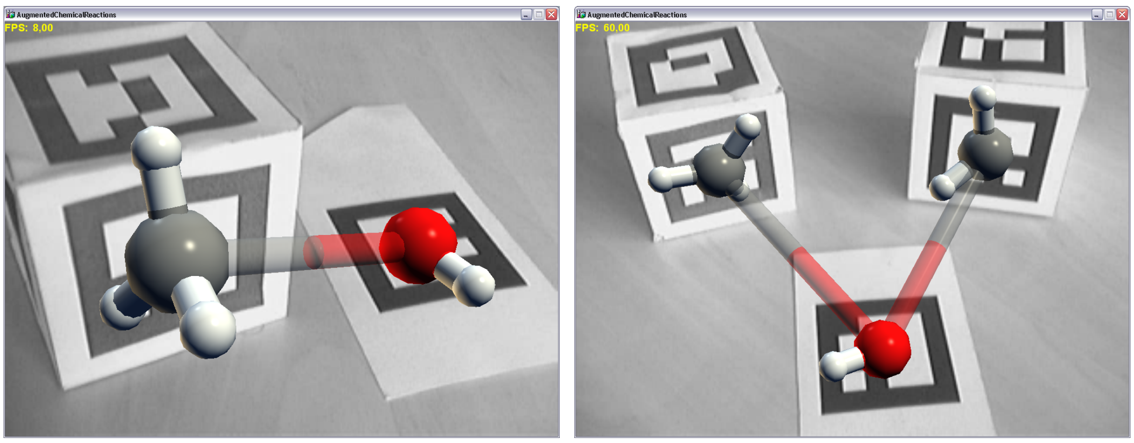
Figure 3.1: Webcam observes a black and white pattern (marker). A marker tracker computes the position and orientation of the marker in the coordinate system of the camera.

3.2 Molecular Dynamics Simulation

The dynamic behavior of the spatial structure of molecules while interacting with them, could be important for the understanding of chemical reactions as well as for validating the desired attributes of designed molecules. Every time a new atom or structure is bound to a molecule, its internal structure changes. Incorporating the energetic influence of the new atom, all atoms in the molecule get into another spatial setting. The energy level of the whole molecule thus gets into the lowest possible level. When designing a molecule, the designer has to check whether the existing structure can accommodate a new group and whether the deformation of the bonding angles keeps the reactive parts of the molecule accessible. Conventional modeling software for chemistry usually computes the new molecule structure at the point in time when a new atom is attached and then renders the new layout on screen. With this principle, the 3D model changes instantly. To determine the structural changes, the designer has to memorize the previous setup, examine the new structure and compare both.

Having a system that shows the deformation already while a new element is approaching, could reduce the effort a developer has to invest for building a new structure. In fact, it might not be useful to show how the whole reaction occurs in detail, because this could confuse the user. Instead, a visualization only showing how the molecule structure bends according to the distance of the two binding partners might be useful. Such visualization could have the potential to facilitate the understanding of the energetic effects of chemical reactions.

A developer of a molecule could move a new atom towards the molecule existing



(a) (b)

Figure 3.2: Possible bonds between molecule parts. The positions of the gray and red atom is defined by the two respective markers, whereas the other atom positions are calculated by an energy optimizer of the tool TINKER.

thus far. When the distance from an atom in one molecule to an atom in another molecule is smaller than a given value (here 5Å) and the atoms have the ability to bind, a possible bond is drawn, represented by a transparent cylinder connecting those two atoms. The molecules are starting to deform due to the forces between the molecules (Figure 3.2a). The dynamic transformation of the molecules can be done by an energy optimizer out of the TINKER package [74] that optimizes the structure of a molecule by minimizing the energy of the molecule. Simulating only one molecule allows the inspection of the molecule in its right spatial confirmation. When interacting with more than one molecule not bonded to each other at the same time, the optimization program would take the whole system into account. As a result, the molecules would be separated in the simulation, due to the repulsive forces between them. To prevent this behavior and give the feeling of holding the molecules in the hands of the user, there is the need to freeze the distance and position of the center atoms of the molecules to the distance and position of the markers. This leads to a deformation of the spatial structure of the molecules because of the attracting and repulsive forces. This interaction is the intuitive correspondence to holding ball and stick models of atoms at their center atoms, allowing to deform the surrounding atoms. The simulation calculates in real-time the optimized version of the whole system with the fixed distance of the center atoms, defined by the distance of the markers.

Figure 3.2b shows additional transparent cylinders drawn between the molecules, when there are more potential bonds. The designer must not only be able to see the next possible bond at a time, but also which other possible bonds are available. Possible bonds could be visualized in different ways. Transparent cylinders between the linking atoms, as well as dashed lines or curves are possible. When there is more than one possibility, where molecules can react, multiple possible bonds have to be displayed to the users. The selection of one particular possible bond out of these can be supported by highlighting the possible bond that has the strongest binding forces. The section 5.1 *Selection* explains further methods for selecting a specific bond out of a list of multiple possible bonds.

Real-Time Interaction with TINKER: To achieve a real-time optimization of the atoms in the visualization, TINKER was closely integrated into the system. The energy optimizer of TINKER runs as a standalone application, thus reads from a non optimized molecular definition file, calculates the optimal positions of the atoms in the molecule to get to the lowest local energy level and writes the results back to an optimized molecular definition file. As the optimization process with this tool is very time expensive, it cannot be used for a real-time optimization of the structures. When calling the optimization each time in the render loop of the main application, it would be unusable. A trick helped to still interact with the system in real-time while TINKER optimizes the molecule structure: At first all the atom positions of the molecule are written to the non-optimized molecule definition file, marking one atom as fixed atom which should be directly controlled with the marker. A background thread is started where the TINKER optimizer does its calculations, keeping the position of the fixed atom the same. During the optimization, the user can normally interact with the molecule and change its position and orientation. When the optimization process is done, the visualization reads the new updated positions from the output file of TINKER and updates the atom positions of the visualization relative to the fixed atom.

It becomes a bit more complicated when controlling more than just one molecule at the same time. Each molecule is connected to its own marker to be controlled independently. In this case, we also write the positions of all atoms to the input file of TINKER and mark those atoms as fixed atoms which are directly connected to the respective markers. When the optimization process is done, we cannot just write the optimized new atom positions back to the visualization. Because TINKER does not know anything about the movements and rotations of the markers done by the user in the meantime, this would revert those interactions because the user-modified positions are just overwritten. Instead, the molecules of the optimized output file are separated, translated and rotated by the respective translations and rotations done by the user during the optimization time. This gives the impression of a real-time optimization while still interacting with the system.

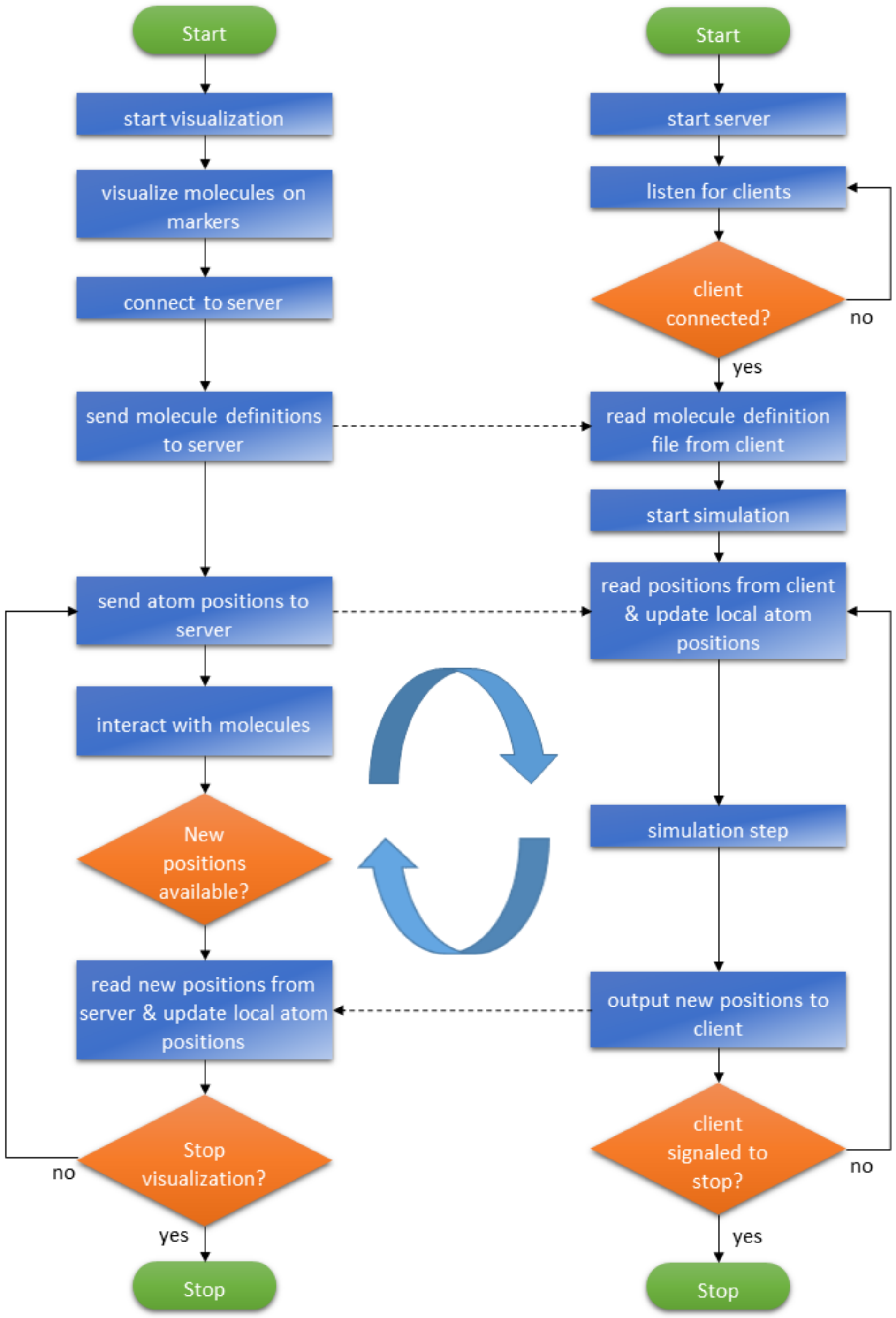

Start
start visualization
visualize molecules on markers
connect to server
send molecule definitions to server
send atom positions to server
interact with molecules
New positions available?
read new positions from server & update local atom positions
Stop visualization?
no
yes
Stop
Start
start server
listen for clients
client connected?
no
yes
read molecule definition file from client
start simulation
read positions from client & update local atom positions
simulation step
output new positions to client
client signaled to stop?
no
yes
Stop

Figure 3.3: This flowchart explains the process of simulating and interacting with the molecules at the same time.

Molecular Dynamics Simulation with Mindy: To get an even more realistic impression on the molecular dynamics, there exists another molecular dynamics simulation NAMD [71]. NAMD is a scalable molecular dynamics application for parallel and high-performance simulations. Like the TINKER package, NAMD is a stand-alone application. For the real-time simulation in Augmented Chemical Reactions, there was the need to implement a molecular dynamics simulation server application which reads and writes all the data via a network connection. Because the source code of NAMD is too hard to convert it to a network server, I instead used the much simpler *Mindy* implementation of NAMD [83]. Mindy is a minimal molecular dynamics simulation based on NAMD. I added a network server part to Mindy to supply all the necessary data like configuration files over the network. To be as flexible as possible, I implemented the *Interactive Molecular Dynamics* Protocol (IMD). With this, also other applications (for example at Molecular Graphics Lab at TSRI [81]) were able to use the new Mindy implementation. In contrast to TINKER, Mindy runs as a server and is initialized for the first time when the simulation in Augmented Chemical Reactions starts. ACR sends the parameters as well as the molecule definition data to the Mindy server which then starts the simulation (Figure 3.3). There is also a thread started within ACR which sends the current atom positions to the simulation. The simulation alters the physical properties like position, speed and acceleration of the atoms in the molecules while the user is simultaneously interacting with the system and sends just the new positions back to the visualization. As with TINKER, the visualization splits up the data into its molecule parts and applies the pose changes of the markers done by the user in the meantime to the results of the simulation. The new position data is then sent back to the simulation to calculate the new properties. This way it is possible to interact with and steer the molecular dynamics simulation.

When running the simulation the user sees how the atoms in the molecules move around due to their kinetic energy (heat). When moving two molecules close enough towards each other, the effects of the attracting and repulsive forces can be observed.

This gives a closer look at the molecular behaviors and can help students to understand better what is going on in a molecule.

3.3 System Architecture

The application „Augmented Chemical Reactions" is written in the high level language C# with the use of the Microsoft® .NET Framework. For drawing the 3D content, the application makes use of the Microsoft® XNA® Framework. In the previous versions of my application I tried to use OpenSceneGraph [82], but as it is written in C++, I had to use a wrapper to managed code to use it in C#. However, this wrapper slowed down the visualization. As it is vital for this application to run at a high responsiveness to allow precise and useful results in my research, the wrapper made it unusable for this project.

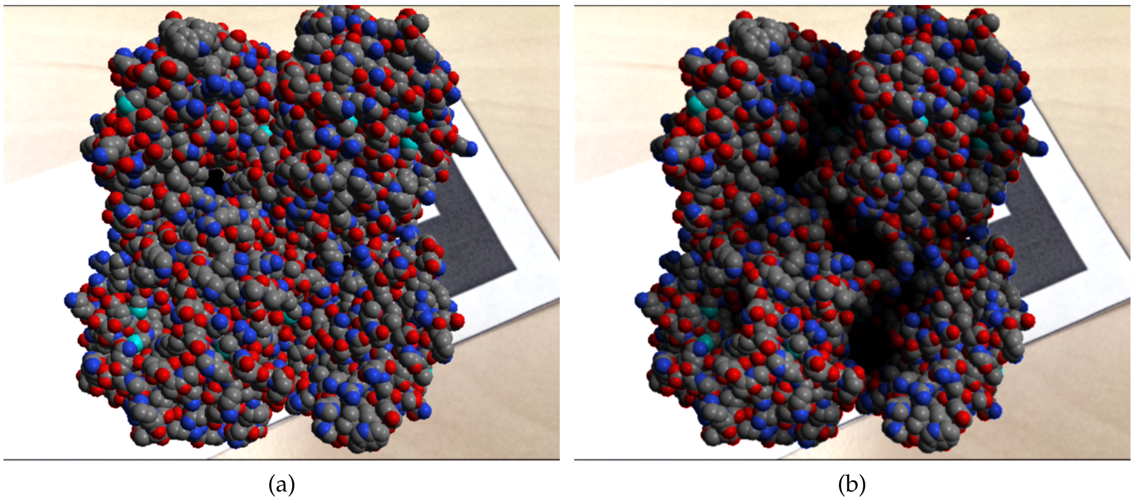


Figure 3.4: Screen Space Ambient Occlusion adds more plasticity to the 3D model. Without SSAO (a) it is difficult to see the spatial structure, whereas with SSAO turned on (b) it can be easily seen where the atoms come out of the molecule and where holes are.

The XNA® Framework came to the rescue as it is written in .NET. But it came with the drawback that it does not contain the functionality of scene graphs. That is why I implemented my own scene graph library for the XNA Framework that supports basic primitives like spheres, cubes, cylinders and much more. I used a basic implementation by GfxStorm [61] of a scene graph for XNA, rewrote and extended it with all the functionality that was needed to fulfill the needs of this application. The extension also included the functionality to render in a stereographic way to be used in head-mounted displays and added a shader which supports „Screen Space Ambient Occlusion" [45]. Screen Space Ambient Occlusion adds another important property to the visualization as it provides a better depth impression to the virtual object/molecule (Figure 3.4).

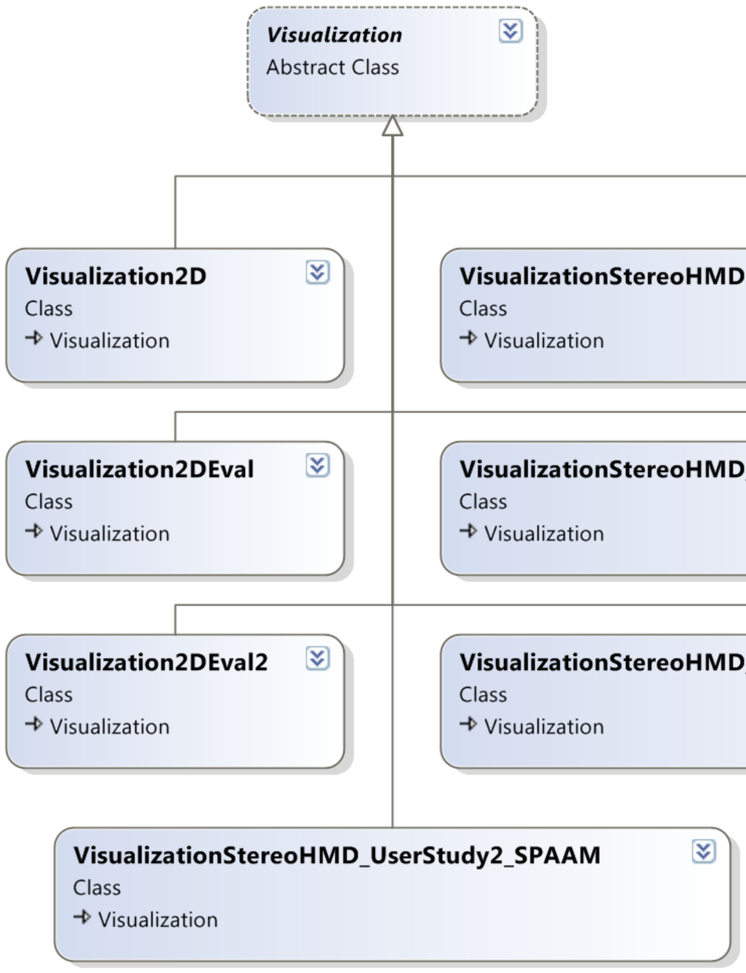


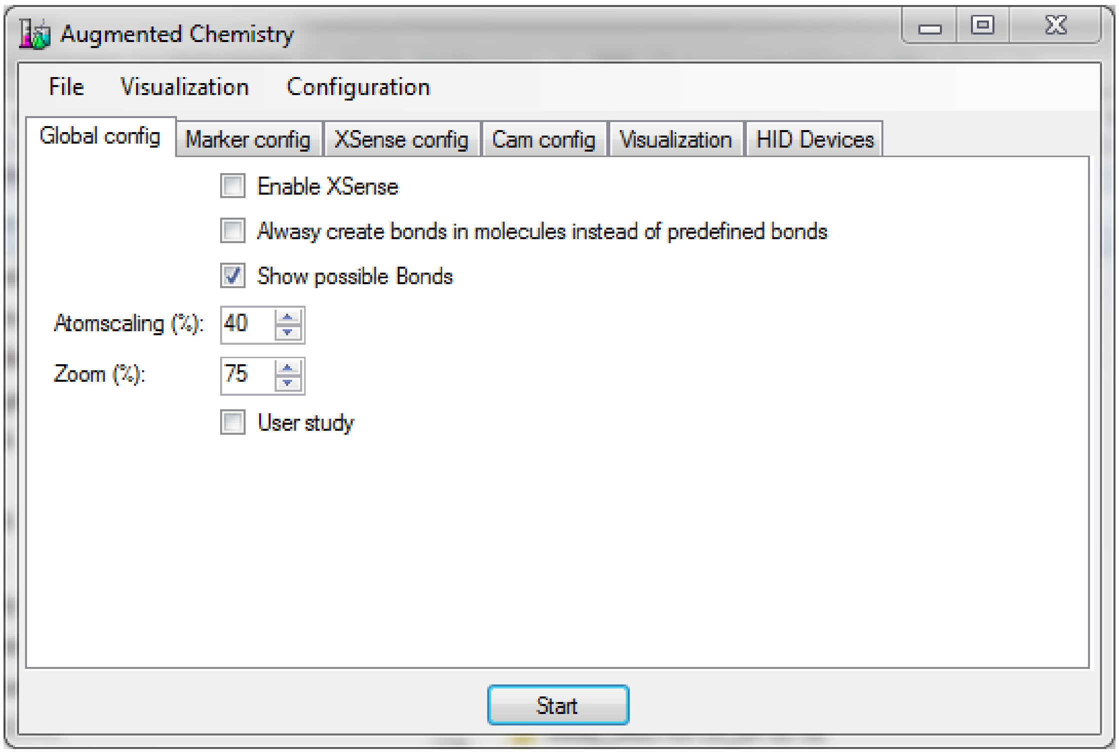
Figure 3.5: All different visualizations inherit from the abstract `Visualization` class.

To provide much flexibility to the application that can be used to do expressive research on, there has been a lot of work in designing the application as usable as possible.

As there exist many ways to retrieve input data for interactions, I used the UbiTrack [39] library which is developed at this chair (Fachgebiet Augmented Reality at the Technische Universität München). It provides the functionality to use several tracking systems to combine them and to deliver the data on a well-defined interface to the Augmented Chemical Reactions Application. This way, it is possible to exchange the tracking systems without changing and recompiling the source code by just changing the configuration file for UbiTrack. The Augmented Chemical Reactions Application just alters the configuration file of UbiTrack according to the configurations made in the graphical user interface. The relevant configurations for UbiTrack are among others the configurations of the markers and the webcam for the marker tracking.

By designing the application in a modular way, it added even more flexibility. As I will mention in the next section, all the different kind of visualizations inherit from the `Visualization` class to allow polymorphism (Figure 3.5). This way the application searches for all implementations of the `Visualization` class and displays their name in the configuration tab of the visualizations (see Figure 3.12). To add another visualization, one just has to implement and extend the `Visualization` class. The same modularity comes with the molecular simulations. As there exist multiple ways for conducting molecular dynamics simulations, the application also supports an abstract `Simulation` class from which the different molecular simulations can inherit.

3.4 Configuration Interface



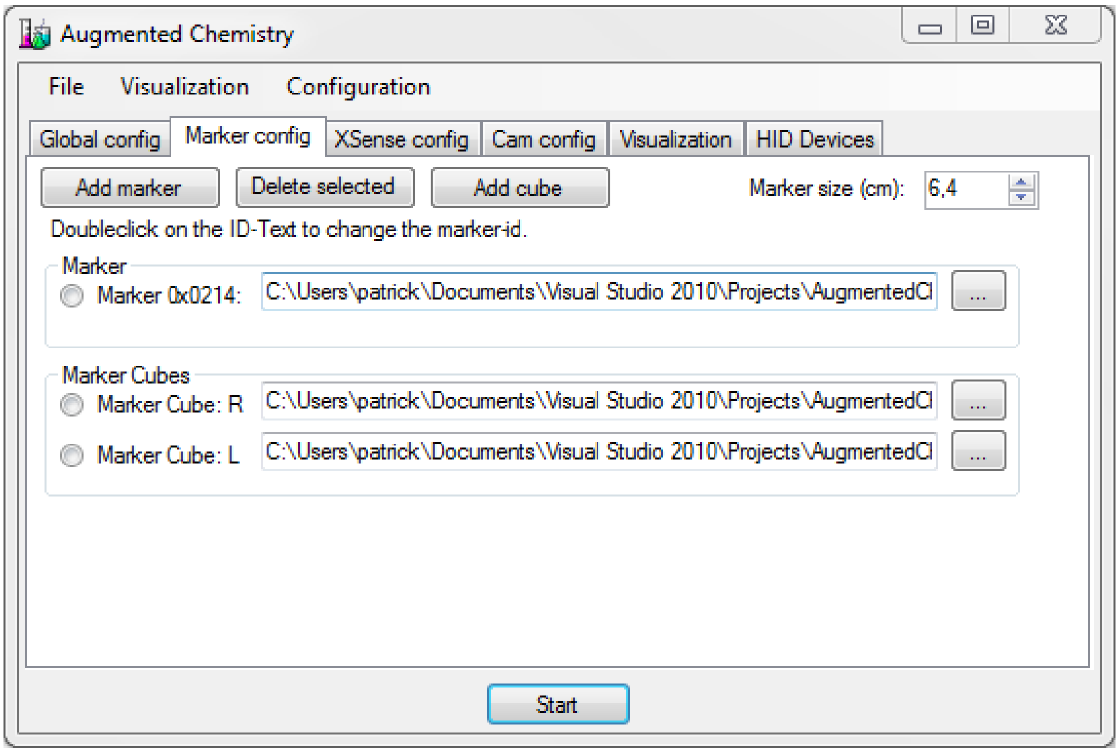
Enable XSense: ☐
Alwasy create bonds in molecules instead of predefined bonds: ☐
Show possible Bonds: ☑
Atomscaling (%): 40
Zoom (%): 75
User study: ☐

Figure 3.6: Configuration Interface

The application starts with a normal WIMP-based configuration interface. Here the user can easily change the configuration and as a result the behavior of the application. The interface consists of several tabs to configure the different areas of the application.

Global Config: In the *Global Config* tab (Figure 3.6), the user can change application wide parameters. By checking the „Enable XSens" checkbox, a hardware gyroscope from the company XSens [95] can be used to manipulate the orientation of a molecule. It can be further configured on the tab labeled *XSens Config*. The next checkbox can be used to tell the system to always create and calculate the bonds by itself, rather than use the bond definitions in the file if they exist. By activating „Show possible Bonds", you can define whether there are possible bonds between two or more molecules when they are able to bind. With the next two elements, there exists the possibility to scale the atom radius and to generally enlarge the 3D environment.

Marker Config: This tab contains the most important configurations – the mapping between a molecule definition file and a marker (Figure 3.7). Here the user can decide which molecule they want to control with which marker.



Marker size (cm): 6,4

Doubleclick on the ID-Text to change the marker-id.

Marker

Marker 0x0214: C:\Users\patrick\Documents\Visual Studio 2010\Projects\AugmentedCl

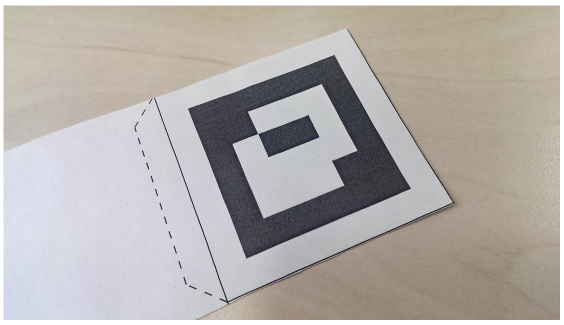
Marker Cubes

Marker Cube: R C:\Users\patrick\Documents\Visual Studio 2010\Projects\AugmentedCl

Marker Cube: L C:\Users\patrick\Documents\Visual Studio 2010\Projects\AugmentedCl

Figure 3.7: Molecule definition files can be associated with markers

There exist two types of markers here: ordinary 4x4 black and white markers (see fig. 3.8a) as used by the UbiTrack [39] marker tracker and marker cubes (see fig. 3.8b) where five sides of the cube have an ordinary marker.


(a)

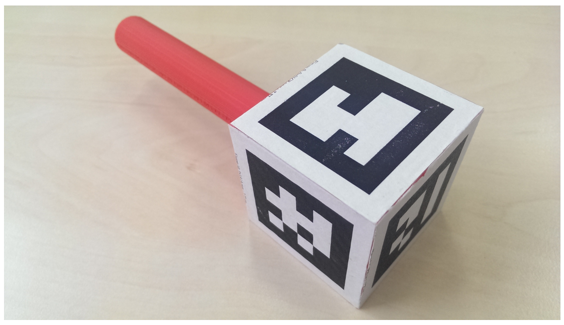

(b)

Figure 3.8: There are two types of markers: A simple marker (a) and a marker cube (b).

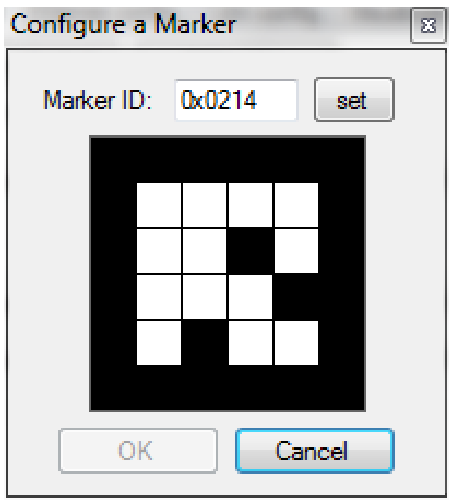
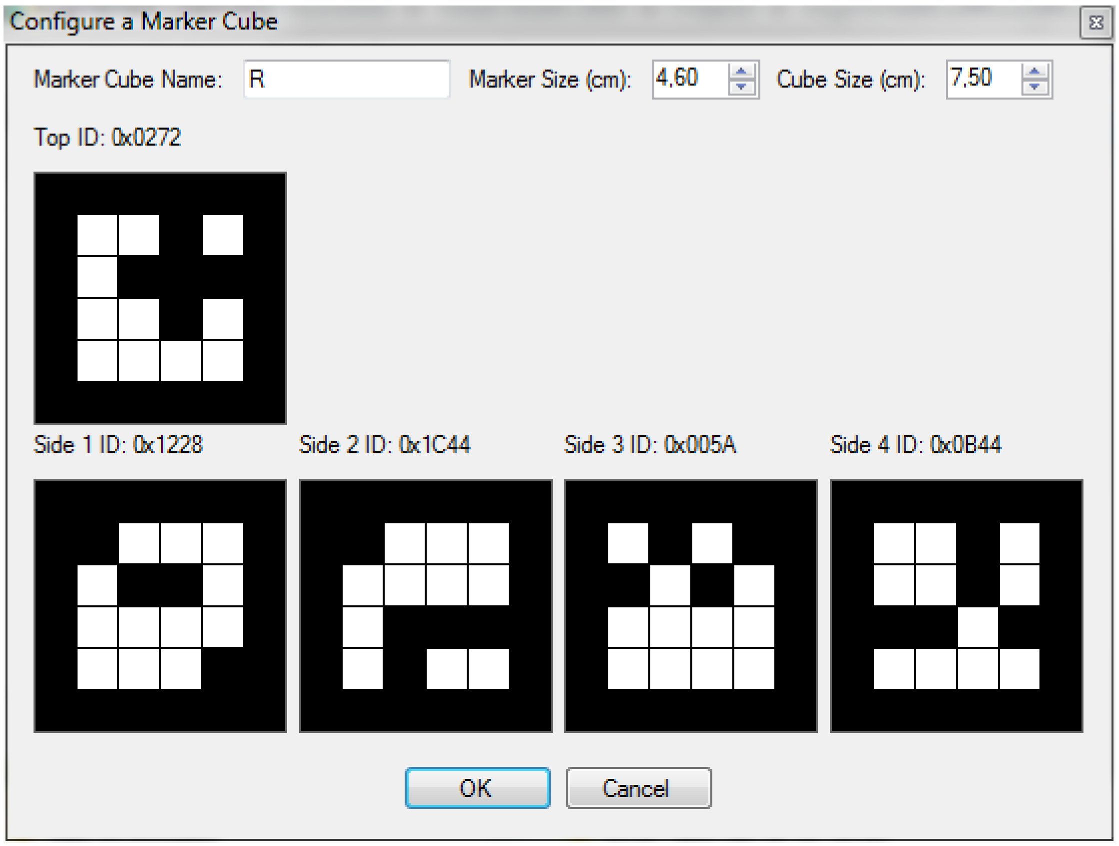


Figure 3.9: By clicking in the 4x4 matrix of the pattern, the user simply can define the marker id.

To add and configure a new simple marker, the user has to click on „Add marker". A new unconfigured entry appears in the list. The user can select a molecule definition file (either in *.xyz or *.pdb format) by clicking on the button next to the text field. The path and the selected filename will appear in the text field. To configure the correct id of the marker on which the molecule should appear, the user has to double click on the id left of the text field. A new window appears (figure 3.9) where for the sake of simplicity the user easily can copy the pattern by clicking the black parts in the marker on and off. The correct id is automatically calculated. So there is not anymore the need to calculate the id of a marker prior to configure the marker tracker. The application also checks if the pattern is a valid marker or if it is ambiguous in the rotation. Those ambiguous markers are not allowed and thus the OK button is disabled. The user also has the opportunity to directly input the id in the text field.

To add and configure a new marker cube, the user has to click on „Add cube". As described above, a new line for the marker cubes appears and the user can choose a molecule definition file for this marker cube. By double clicking on the left side of the text field, a new window appears (Figure 3.10) where the user can define the patterns and ids of the markers on the five sides of the cube. In this new window, the user can configure the size of the markers and the size of the cube, as well as the name of the cube. The five sides of the cube can be configured the same comfortable way as the simple markers described above. Here the marker in the first row describes the top marker. The second row describes the markers which are on the sides of the cube. They have to be oriented and ordered exactly the same way as on the real cube. The collocation is like the unfolded cube correspondence.



Marker Cube Name: R

Marker Size (cm): 4,60

Cube Size (cm): 7,50

Top ID: 0x0272

Side 1 ID: 0x1228

Side 2 ID: 0x1C44

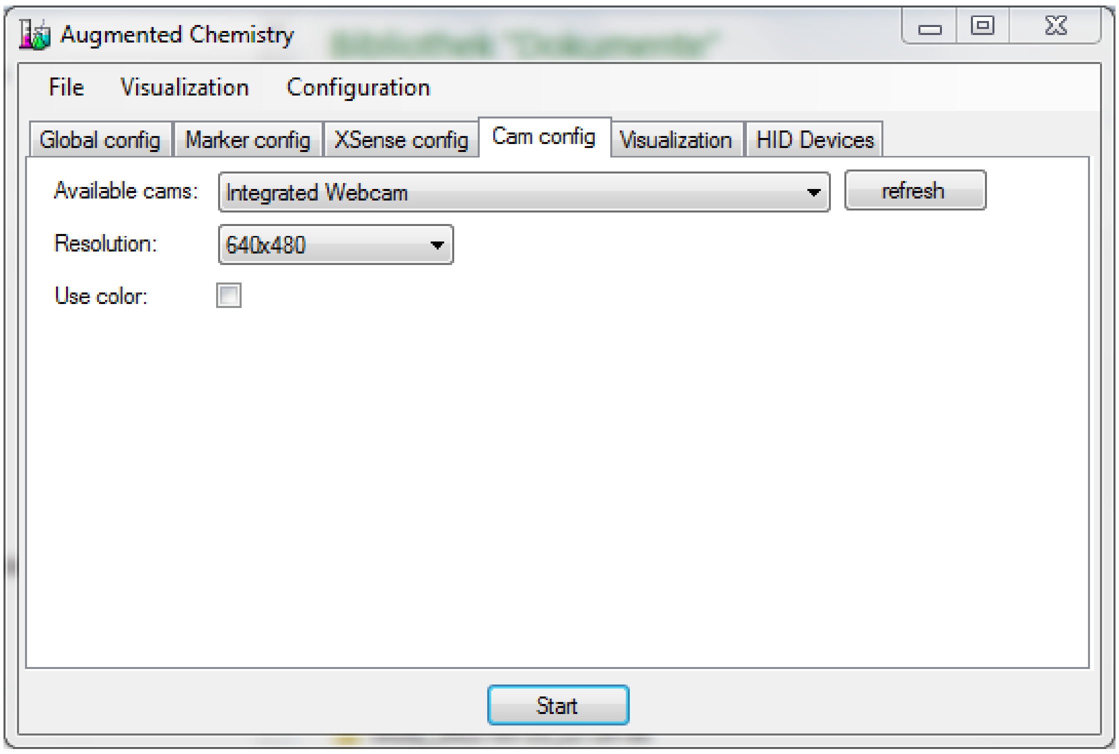
Side 3 ID: 0x005A

Side 4 ID: 0x0B44

Figure 3.10: As with the simple marker the user can define the pattern and id of each side of the cube by clicking in the 4x4 matrix of the pattern of each marker.

XSens Config: The *XSens Config* tab is for setting up the serial communication to the gyroscope device. All the COM ports can be probed for a correct XSens device. The desired device can be selected from a drop down list.

Cam Config: On the *Cam Config* tab, the user is presented a list of all connected camera devices from which the desired one can be selected (Figure 3.11). The camera is used by the marker tracker to track the black and white square markers and is also used to provide the background image in the application for the augmentation. The resolution as well as the color mode (color / black and white) can be configured.



Available cams: Integrated Webcam

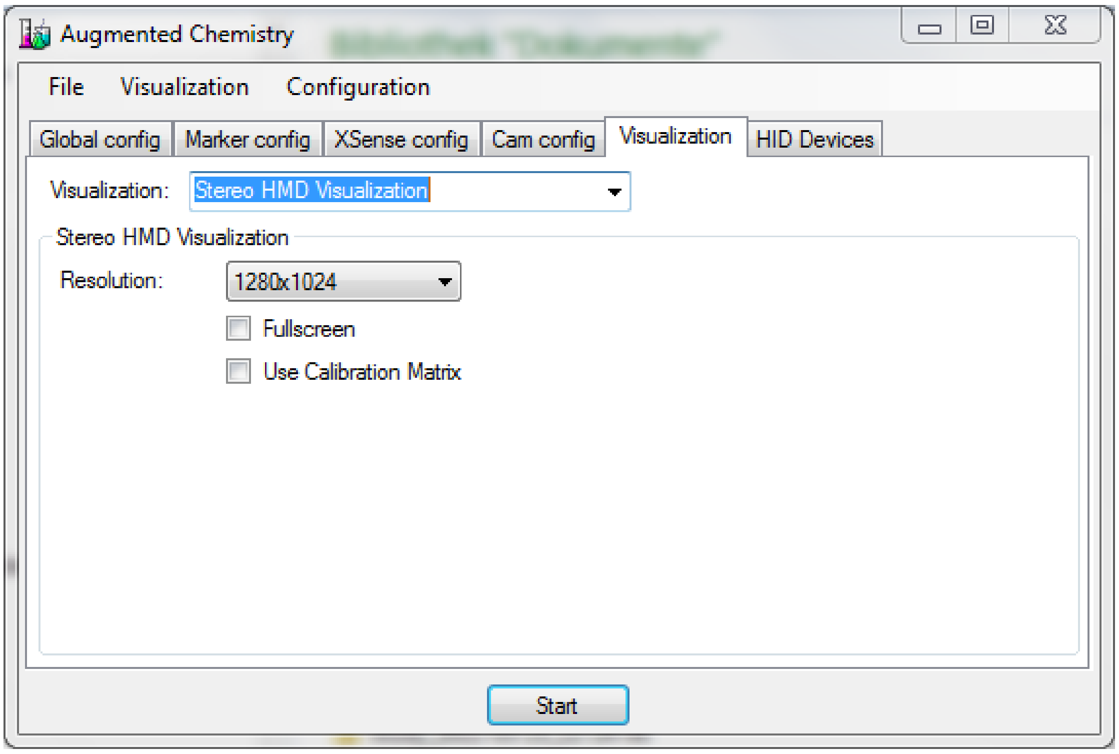
Resolution: 640x480

Use color: ☐

Figure 3.11: The program automatically searches for all available camera devices. The resolution as well as the color mode can be configured.

Visualization: There exist several ways to visualize and display the molecules with augmented reality. As I have already described earlier in the previous section 3.3, the application has a modular design. There can be several ways for displaying the augmentation on the markers. For each scenario there is an entry in the drop down list (see Figure 3.12). When a visualization is selected, the user also can change some parameters associated to this visualization in the group box below. There can be the possibilities to set the resolution of the visualization and to run it in full-screen mode or not.

For example there is a 2D visualization, where the user can use the computer monitor as the display. There also is a „Stereo HMD Visualization" where the augmentation is rendered stereoscopically on two screens to support a stereo head-mounted display. With this option a user can use a stereo optical see-through HMD for a immersive impression. There are also other visualizations possible as well as other programmatic behaviors. For user studies, I copied existing visualizations and altered the way the user can interact with the system as well as changed the application logic for doing the evaluation.



Visualization: Stereo HMD Visualization

Stereo HMD Visualization

Resolution: 1280x1024

☐ Fullscreen

☐ Use Calibration Matrix

Figure 3.12: Several different visualizations can be selected and configured in this configuration tab.

HID Devices: On the tab for *HID Devices* exists the possibility to scan for the wireless rumble motor used in a demo for „The 2nd Experiment@ International Conference" [88]. The University of Porto in Portugal [89] created a marker cube with a built-in vibration motor. This motor was used to raise the level of immersion and give additional information and feedback when the user brings together two virtual molecules. Each time a possible bond is created or destroyed between the two molecules, there is a short rumble. The users liked the physical feedback when the possible bonds changed. There is also the possibility to start the vibration motor when a possible bond is created and amplify the vibration with the inverse length of the possible bond. This should give an idea about how close the two molecules are together.

4 3D Input Devices

Every user interface needs some kind of input device (for example buttons, keyboards, switches, sliders, touch surfaces, etc.). Those user interfaces are the connection between the user and the devices in general. On electronic devices, buttons, switches or sliders are the input devices for the user interface. With PCs, the common hardware input devices are the mouse and the keyboard. Those are used to deliver input for the WIMP[1]-based user interfaces.

For the application Augmented Chemical Reactions, I investigated several input devices to control the position and orientation of the molecules as well as to perform system control with gestures. The input device is normally tracked in the 3D space to deliver 3D position and orientation data which then can be processed by the user interface of the application.

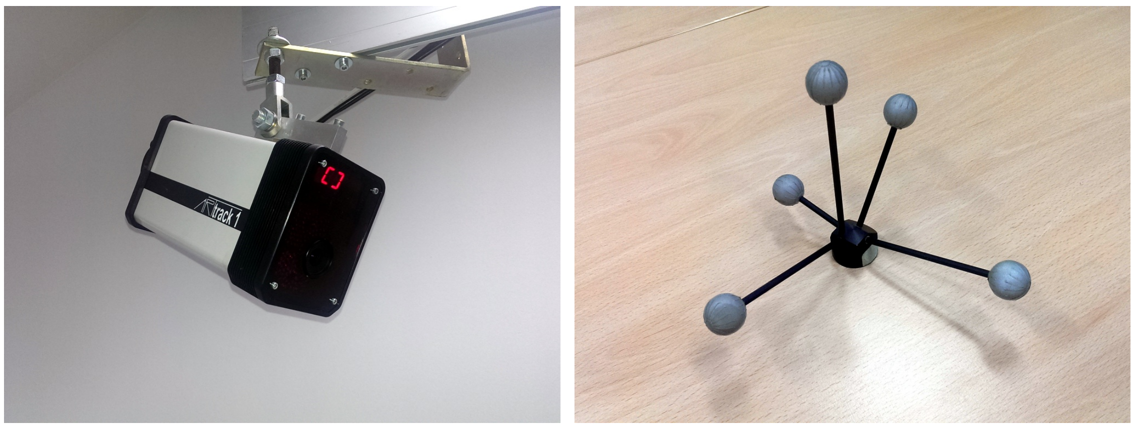
The UbiTrack [39] library that is developed at our chair serves as a tracking library and supports several different input devices. Its power lies in combining the different input devices (sensor fusion) for better tracking data and to achieve a high flexibility. With UbiTrack it becomes possible to uncouple the input devices from the application. By using UbiTrack, my application does not need to know anything about the input devices. The virtual molecules in the application can easily be linked to any input device of UbiTrack. No matter which input devices or trackers are used, the UbiTrack library delivers the position and orientation of the molecules.

In my research I tested several different input devices at which we will now have a closer look.

4.1 ART Input Device

The ART tracking system [1] is a optical outside-in tracking system which consists of several infrared cameras (Figure 4.1a) mounted around the tracking volume and some targets (Figure 4.1b) representing the molecules. The cameras emit infrared flashes which get reflected by the retro-reflective surfaces of silver spheres on the targets. With the positions of the spheres in the images of the cameras, the server calculates the real

[1]WIMP stands for **W**indows, **I**cons, **M**enus, **P**ointers in human-computer interaction



(a) (b)

Figure 4.1: ART infrared cameras (a) track several targets (b).

position and orientation of the targets in the tracking volume. The tracking data is then sent to the application to modify the position and orientation of the molecules.

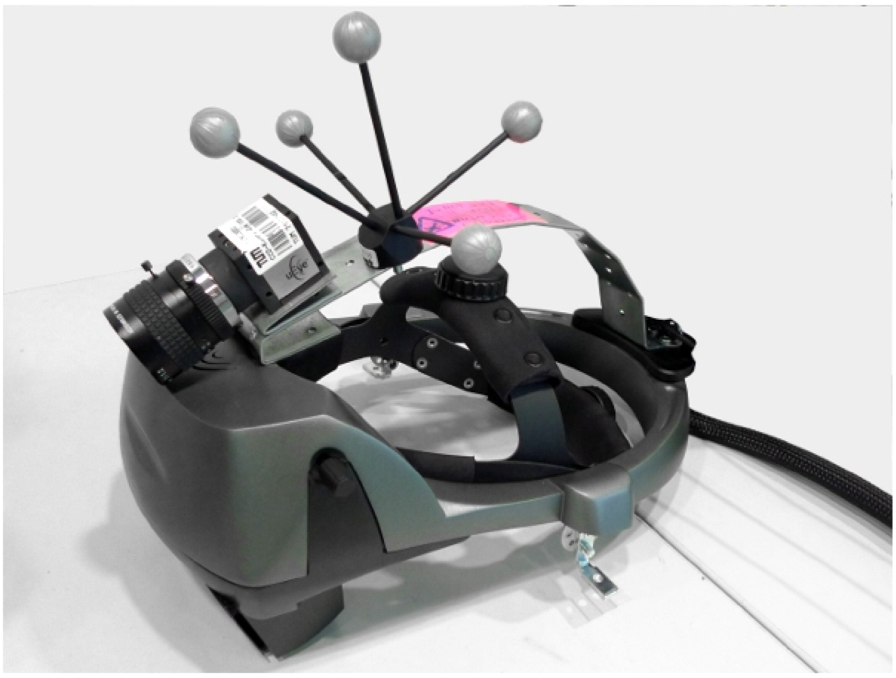


Figure 4.2: A tracked Video/Optical See-Through Head-Mounted Display visualizes the molecules in a stereoscopic way.

As the user moves and rotates the tracked targets, the virtual molecules move and rotate accordingly. With a tracked stereo Head-Mounted Display (Figure 4.2), the molecules can be inspected in a natural way and it appears as if the user interacts directly with the virtual molecules.

4.2 Gyroscope Input Device

For inspecting molecules, the rotation is very important. A very fast input device for delivering low latency rotational data is the gyroscope by XSens [95] (Figure 4.3). To use this gyroscope in UbiTrack, there already exists a driver which can be used to get the rotation data.

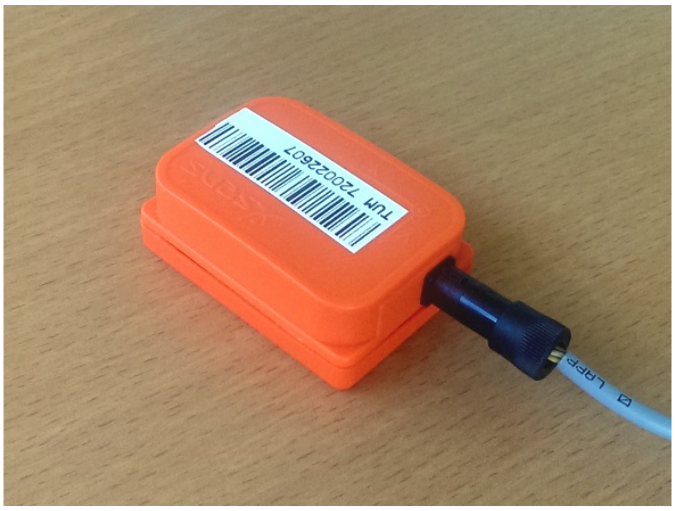


Figure 4.3: XSens gyroscope retrieves accurate orientation data with low latency.

When the use of the XSens gyroscope is configured in the application, the user has to hold the gyroscope and rotate it. The orientation is directly mapped onto the molecule. Thus, by rotating the gyroscope, the molecule gets rotated the same way. This input method has the drawback that it has only three dimensions for the orientation and none for the position. It therefore is not possible to translate the molecule. As a solution for this problem, depth cameras as those described in the next subsection could be used in addition to deliver the position data.

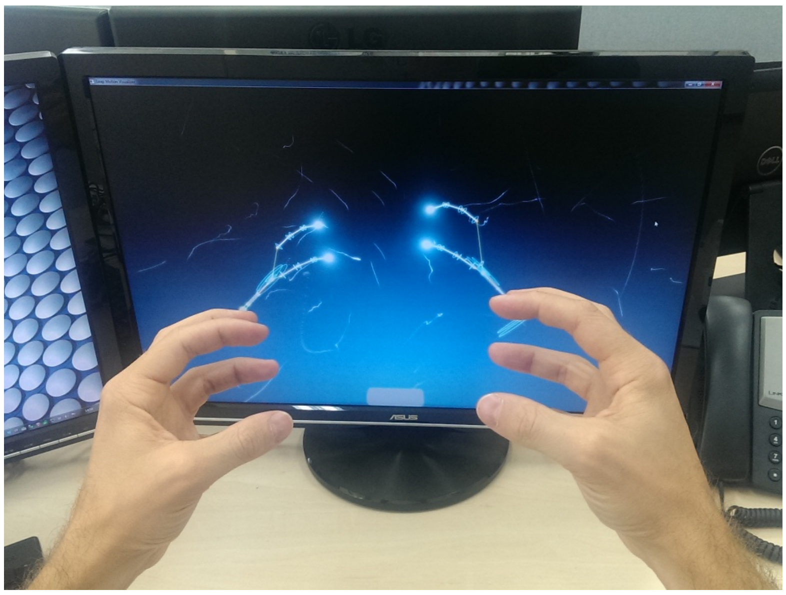


Figure 4.4: Finger tracking of the LEAP Motion gives orientation and position data of the two hands.

4.3 Depth Camera Input Device

As already mentioned in the previous section, there is a need of a positional sensor if we use gyroscopes to retrieve the orientation for the molecules. The Microsoft® Kinect™ or the LEAP Motion [49] can be used to acquire the positional data. The data can be sent through UbiTrack without changing the source code of Augmented Chemical Reactions. In contrast to the Kinect, the LEAP Motion can also deliver all six dimensions for rotation and location (Figure 4.4).

Those depth camera input devices in conjunction with gyroscopes could be usable, but as the devices were not yet available to me at that stage of my thesis, I did not investigate the use of such devices more deeply.

4.4 TISCH Input Device

Another completely different 3D input method for controlling the six dimensions of an object/molecule was tested on our multi-touch table TISCH [20] (Figure 4.5).

In his bachelor thesis [46], Franziskus Karsunke implemented a driver into UbiTrack which integrates our multi-touch table as an input device for 6DOF data. 2D gestures

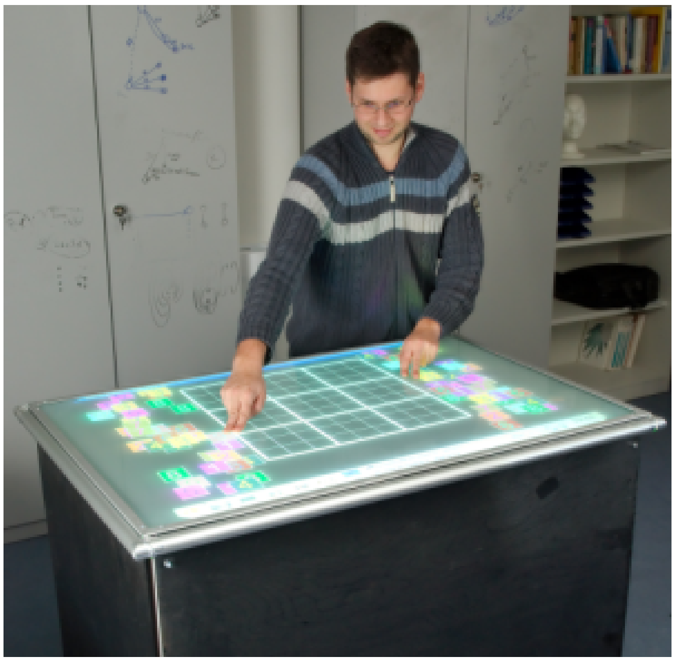


Figure 4.5: Tangible Interaction Surfaces for Collaboration between Humans (TISCH) multi-touch device.

on the touch surface are mapped to 6D pose changes. A normal dragging changed the position of the 3D object on the x-y plane, while a zooming gesture changed the z position. By holding one finger down next to the object and performing a drag gesture with a second finger on the object, it is rotated as you would rotate a sphere by grabbing and dragging on the surface of the sphere. On the multi-touch table the molecules were displayed at the same time.

This input method turned out to be not as usable as the others. This is why it was not further investigated.

4.5 Marker Tracker Input

The easiest and most usable way to retrieve the poses for the molecules is an optical marker tracker. The UbiTrack library also supports this kind of tracker which is besides the ART tracker the most used tracking system. Its advantages lie in the simplicity of the usage and in the cheap components. Users just have to print the black and white pattern and use their favorite webcam to track the patterns.

The camera acquires images of the black and white pattern (marker). On the basis of

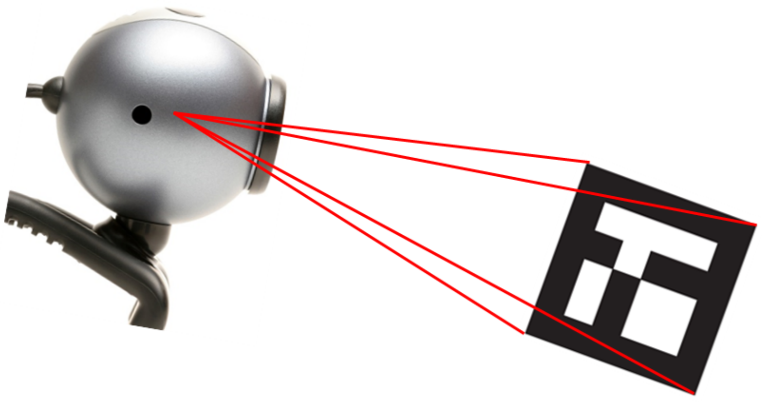


Figure 4.6: Webcam observes a black and white pattern (marker). A marker tracker computes the position and orientation of the marker in the coordinate system of the camera.

the intrinsic parameters of the camera and the deformation of the pattern in the camera image, the tracking algorithm computes the position and orientation of the marker in the coordinate system of the camera. By setting the virtual rendering camera to the same intrinsic parameters as the real camera, and by rendering the molecule at the calculated position and orientation, the augmentation of the marker by the molecule can be achieved.

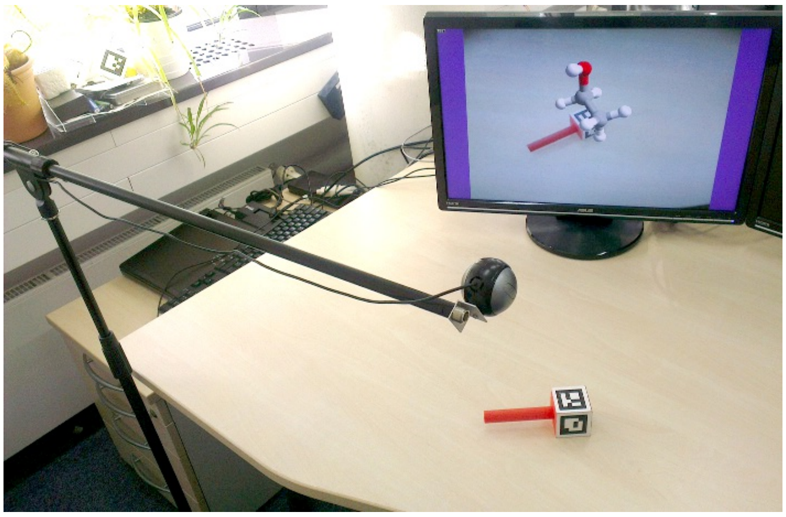


Figure 4.7: To give the user as much space to move, I used a microphone stand with the camera mounted at the end of the arm.

When using the marker tracker in conjunction with the 2D visualization on the computer screen behind the working volume, the webcam has to be placed near the eyes of the user. When doing so, and letting the camera point into the same direction as the user is looking, the user will have no problem moving and rotating the molecules the way they are intended to.

In searching for an easy way to place the camera next to the head of the user without always colliding with a tripod, I designed a special kind of tripod consisting of a microphone stand where the microphone is replaced by the camera (Figure 4.7).

5 3D-UI Concepts

In the process of designing molecule structures from smaller molecule parts, there is the need to tell the system what it should do. Here the user interface is a very important component of the system. Commonly used user interfaces for a 3D environment on the PC are mouse and keyboard input devices. Those input devices operate in two dimensions (mouse on the desk) and therefore need a mapping to the three dimensions of the 3D environment. This mapping is not intuitive and needs a certain amount of training.

Manipulating objects in the 3D world with our hands is a task that we learned since we were children. Therefore it is very natural to us to manipulate, navigate and control objects in a virtual 3D environment with a direct manipulation of tracked real objects. When handling with more than one virtual object, both hands are required for the direct manipulation. As a consequence, there is a need for further 3D user interface concepts.

Parts of this chapter have already been published in papers [58] and [57] presented at 3DUI 2010 and JVRC 2012.

Selection: When directly manipulating virtual objects in the 3D space with a two-handed interaction, the selection of other objects becomes a problem. There is no additional hand available to perform a selection task while already manipulating the virtual objects at the same time. For this reason I investigated different approaches of selection while performing a two-handed direct manipulation task.

Confirmation: After the process of selecting, a selection has to be confirmed. Analogue to the selection problem, there also exists the problem of confirming this selection.

In the following sections, different aspects and possible solutions are described for the selection and confirmation task in the context of Augmented Chemical Reactions. Although there exist several ways for interacting, this thesis concentrates on two-handed interactions in contrast to a one handed direct manipulation with an additional keyboard and mouse interactions.

5.1 Selection

In 3D user interaction, there exist five main tasks: selection, manipulation, system control, navigation (wayfinding, travel) and symbolic input [13]. The selection problem is in my eyes one of the most important problem in 3D user interaction. It gives the answer to the question „What are you referring to?”. This is mostly important for the manipulation where you first have to select the object you would like to manipulate.

There are two selection problems. The first is the selection of virtual objects that are not directly controlled by the user. Various solutions exist for this problem as for example the world-in-miniature by Stoakley et al. [79], the Go-Go by Poupyrev et al. [73] or the ray-cast selection. For the ray-cast selection a user controls a tracked pointing device and tries to select the virtual object by pointing at the virtual object. This is called ray-cast selection because a ray is cast in the virtual world, originating from the tip of a user controlled and tracked object and can hit a remote object, which is then selected.

The second selection problem is a problem that is not so obvious at first. When a user controls tracked objects, how can the user select one controlled object to trigger an action on that object? As the user is controlling that object, the user cannot perform a ray selection with that object itself. There have to be other methods to select the user controlled object.

Both problems can be solved by using gestures. With gestures, the user can select an already controlled object without having to use the keyboard or other input devices.

This can be used during the visualization and the simultaneous control of molecules with a two-handed interaction where users need to combine molecule parts to build larger molecules or simulate a molecular reaction.

To achieve a binding of two molecule parts, a bond has to be created. As molecules can be bonded in several ways, there can exist multiple possible bonds from which one is the desired bond. The user has to select one out of a potentially large list of possible bonds.

In detail a molecule consists of multiple atoms. When two molecules react with each other, they bind together. This binding usually happens between an atom pair between the two molecules. Each molecule has a list of possible binding partners (atoms) $A_{(m,n)}$ where m is the molecule number and n is the index of this possible binding partner in this list. A bond $B_{(k,l)}$ is then created connecting one binding partner $A_{(0,k)}$ from the first molecule with an binding partner $A_{(1,l)}$ of the second molecule.

When the user wants to create such a bond while directly controlling the position and orientation of those two molecules, a new method has to be used. A bond $B_{(k,l)}$ can either be selected by directly selecting it, or by selecting the two binding atoms $A_{(0,k)}$ and $A_{(1,l)}$. There exist many ways to select a particular atom from a large molecule.

As described above, this has traditionally been done in AR and VR by pointing at the object with a tracked pointing device (or finger). Yet, in my system, the user would have to release the molecules in order to grab the pointer. When releasing the molecule, it changes its position, which could cause unwanted effects on the chemical simulation.

I developed and investigated two methods to select a possible bond from such a large list without the need of releasing the molecules.

5.1.1 Selection by Enumerative Toggling

In the first approach to get by without a special pointing device, the system can ask users to toggle through a list of all objects (in this case: atoms $A_{(m,n)}$ for a specific molecule number m) in sequential order. For the bond selection, the user has to toggle through the list of possible binding atoms $A_{(0,k)}$ of the first molecule and through the list of possible binding atoms $A_{(1,l)}$ of the second molecule. To this end, a toggling signal is needed to switch from one binding partner to the next in the list. This can be done by voice or by a gesture. As voice is not always the best choice, especially when using it in a crowded or noisy laboratory environment, I designed a shake-based method to toggle between objects.

Metaphor: This gesture implements selection by shaking. When the user wants to select a possible bond between two molecules, it is possible to cycle on each molecule through the list of possible binding partners (atoms) by shaking the respective molecule. This gesture can be seen as a movement to get rid of the current selection and go one to the next selection. It is like shaking a cup with a die to discard the current value on the die and go to the next value. In contrast to the example of the die, where the next value is a random one, we will select here the next possible binding partner in the list.

Interaction Process: When selecting one out of several atoms from a molecule that have open bonds, these atoms are highlighted one at a time in a sequential order. The user toggles from the currently highlighted atom to the next one by briefly shaking the tracked real object that is associated with the molecule. The system then highlights the next atom and continues through the ring list with every further shake by the user. The underlying expectation is that a brief, sharp shaking gesture is less disruptive to the overall chemical simulation than depositing the physical tracked object on a surface and grabbing another interaction device to select the binding partners. The result depends on the algorithm that recognizes the shaking gesture. But the main problem with this method and a simultaneous simulation persists: The shaking will have an influence not only because of the fast movement but also because the molecules do not have the exact position after the gesture as before.

Recognition Algorithm: The detection of a shaking gesture that is robust enough for this application is not that easy. The algorithm has to distinguish between a slight brief shaking and the jitter of the tracked object induced by the inaccuracy and errors of the tracking system or as well the normal human tremor.

White et al. have presented a technique for recognizing a shake gesture by following the path of the tracked marker and transforming it into as sequence of directional units (up, down, left, right, front, back) [91]. They parse the directional information. When they detect four continuous movements in opposing directions, they interpret this as a shaking gesture. This could cause problems when the user performs a brief shake consisting of less than the four continuous movements in opposing directions. No shaking gesture will also be detected when the user changes the direction of the shake movements during shaking.

To achieve a robust detection, my gesture recognition algorithm investigates two main properties of the tracking data. Those two parameters are the length and the spatial spread of the trajectory of the tracked real object. During the run-time of the detection, we save the tracking data (position of the tracked real object) in a list containing only position data which are not older than a certain amount of time (in this case $0.5sec$).

In contrast to a list containing only a fixed number of position data, this has the advantage that the algorithm does not depend on the sampling rate of the tracking data. For example with a fixed number of tracking data and a higher sampling rate, the trajectory length would become shorter. This is not the case with position data of a fixed time period.

Fig. 5.1 shows two trajectories of the origin of a tracked object over the last half second. The points P_{Ti} represent the sample points measured by the tracking system. The left trajectory (blue lines) comes from a typical non-critical hand movement whereas the right trajectory shows a shaking gesture.

Length of the Trajectory: Having the list of position data of the tracked real object, we can calculate the first parameter for the detection algorithm, the trajectory length. We calculate the length of the trajectory l_T for the last half second by summing up the distances between the sampling points $P_{T1}...P_{T7}$ of the trajectory.

$$l_T = \sum_{i=1}^{n-1} \left| \overrightarrow{P_{Ti}P_{T(i+1)}} \right|$$

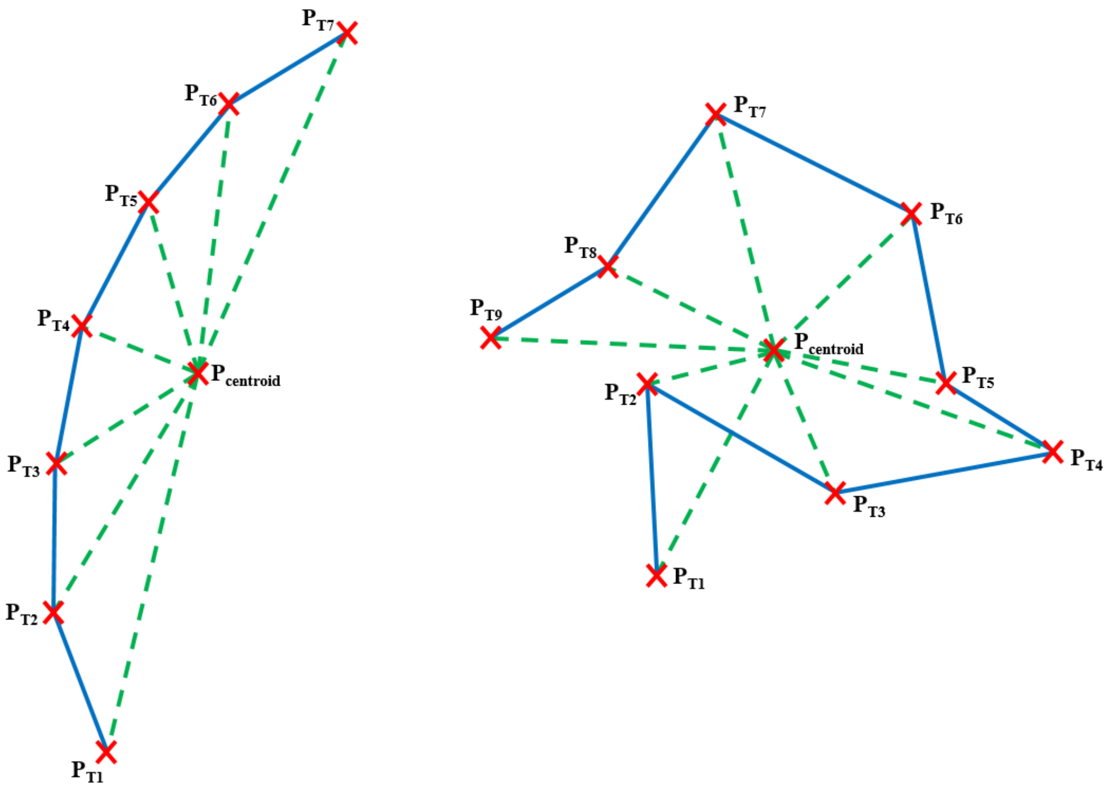


Figure 5.1: Trajectory length l_T and spatial spread s_T calculation during the past $0.5sec$ to classify the shaking gesture. left: normal movement, right: shaking

Spatial Spread of the Trajectory: The second attribute is the spatial spread of this trajectory s_T. The spatial spread is an indicator which shows whether the tracked object is moved mainly around one local position. We calculate the spatial spread by computing the arithmetic mean of all absolute distances from the sampling points P_{Ti} to their centroid $P_{centroid}$.

$$P_{centroid} = \frac{1}{n} \sum_{i=1}^{n} P_{Ti}$$

$$s_T = \frac{1}{n} \sum_{i=1}^{n} \left| \overrightarrow{P_{Ti} P_{centroid}} \right|$$

Alternatively one also could use the variance or the median instead of the mean distance to the centroid. Those have the advantage that outliers produced by tracking errors do not have so much influence on the result. In my algorithm, I used the mean as this was already a really good measure for the spatial spread and gave good results.

Decision Process: Now that we have these two parameters, there has to be a formula which decides when the user is in the process of performing a shaking gesture. To get thresholds for the decision, I collected tracking data for two types of movements. The first movement type was just moving around and inspecting the tracked object. The second was performing shaking gestures in different intensities.

With this data, a scatter plot was made which can be seen on figure 5.2. In this plot you can see one cross for every time-step of the prerecorded tracking data. Here the x-axis represents the length of the trajectory l_T of the last $0.5 sec$, whereas the y-axis represents the spatial spread value s_T within the trajectory.

As one can see, there are mainly two areas where the points of the two types of movements cluster together. Starting from the lower left, going to the upper right are points that result from normal movements like inspecting the hand held tracked object. Here on the upper right, the trajectory is long which is a result from fast movements, and the spatial spread value is high which means that the sample points of the tracked object positions are widely spread (e.g. a straight movement as on the figure 5.1 on the left side).

Starting from the lower left, going to the lower right are points that result from shaking movements. With a low spatial spread value and a high value for the trajectory length, this is an indicator for a shaking gesture, as the shaking is performed fast but in a small local area (Figure 5.1 right side).

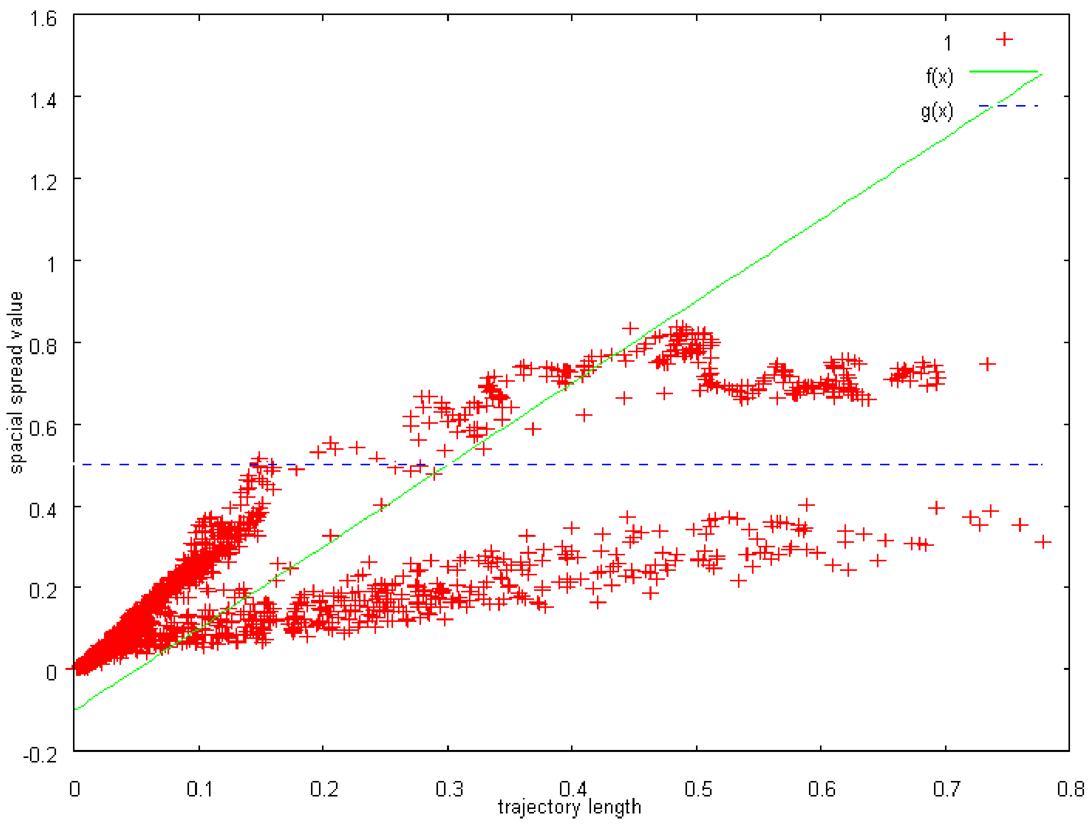


Figure 5.2: Scatter plot of red points, each showing the current gesture detection value of prerecorded movements. The x-axis shows the length of the trajectory of the last 0.5 second, whereas the y-axis represents the spatial spread value. When a point is below the blue and the green line, a shaking gesture is recognized. The points outside that area are normal movements of the tracked object, thus not triggering a shaking gesture.

Having now those two parameters, we can define areas of acceptance in the scatter plot. We defined the area below the green $f(x)$ and blue line $g(x)$ as area of acceptance of the shake gesture.

$$g(x) = 0.5$$
$$f(x) = 2 * x - 0.1$$

This area can be adapted to the special needs or circumstances of the respective setup. This can be achieved by recording and analyzing the movement data as described above and fitting the two lines in a way to separate the clusters of the two movement types. Whenever the currently calculated values of the trajectory length l_T and the spatial spread s_T generate a point in that area, a shaking gesture is in progress. This applies when the following inequations are true:

$$s_T < 0.5$$
$$l_T > \frac{s_T}{2} + 0.05$$

There still can be false positives and false negatives. To minimize those wrong detections and unwanted multiple detections, we have to tune the parameters and the detection algorithm a bit.

Parameter Tuning: When performing a shake gesture, the corresponding data point in figure 5.2 stays below the line $g(x)$ and moves from the lower left side to the right side, crossing the line $f(x)$ and thus triggers the gesture. But when a brief shaking gesture with a change in intensity is executed, it may happen that during one gesture the data point crosses the trigger-line into the area of acceptance multiple times and thus also triggers the detection multiple times.

To suppress those multiple triggers at the border of the area of acceptance, we could use a fixed hysteresis around that border. Hystereses used hitherto apply fixed boundaries. They trigger the event when the value rises above (falls below) the border (threshold). The event then can only trigger again, when the value falls (rises) a specific amount below (above) the threshold value. This hysteresis should prevent from multiple triggering when the value just alters a bit due to noise in the measurement of the value. With such hysteresis it could happen that multiple shaking gestures, rapidly made one after each other, were recognized as only one shaking gesture, because the value did not cross the threshold value during those multiple shaking gestures. To make the gesture triggering even more sophisticated, I introduced a dynamic hysteresis.

Instead of having a fixed hysteresis around the threshold, I incorporated a dynamic hysteresis with the width of 0.05, updating the upper and lower boundaries according to the current value. Thus, whenever the value raises above the upper boundary, the entire interval is raised such that the upper boundary is at the current value. When the value raises above this boundary for the first time and the spatial spread value stays below 0.5 (see inequation from above), the shaking gesture is triggered and a variable is set telling the system not to trigger the gesture again until it has been reset. When the value falls below the lower boundary, the entire interval is lowered accordingly, and the variable is reset. When the value now exceeds the upper boundary again, the next shaking gesture is triggered.

This allows the user to rapidly perform multiple shaking gestures one after the other even when the entire spatial spread value stays above the threshold $c_T < 0.5$. Furthermore it also allows the system to recognize heavy shaking gestures and shyly made gestures with the same setup.

Future Work: This method only uses the positional data for recognizing a shaking gesture. But users also could perform shaking by rotating the tracked objects in quick movements around the same point. Here the algorithm would not detect a shaking gesture as the trajectory length would be below the threshold. An additional check for the amount of rotation would detect such rotation shakes. The algorithm would check, if the spatial spread value s_T is below its threshold, not looking at the trajectory length l_T but at the sum of acceleration of the rotations α_{sum}. α_{sum} is calculated by summing up the rotational accelerations from all the n elements in the queue.

$$\alpha_{sum} = \sum_{i=1}^{n-1} \alpha_i$$

Nevertheless, as already mentioned before, the problem remains that the fast movement for the gesture can disturb a simultaneous molecular simulation. Without a concurrent simulation, this will not be a problem. The next method overcomes this problem by using slow movements.

5.1.2 Selection by Proximity

A second method for selecting a possible bond out of the large set of possible bonds is selection by proximity. In contrast of the previous method of selecting the possible bond by treating both molecules separately, this method only works when we investigate both molecules together.

Metaphor: This method exploits the fact that close objects have some kind of reference to each other. Even more, when having a look at magnetic or electrostatic systems, objects with an opposite feature attract each other. There is the tendency that the two closest objects will connect, because they normally have the strongest attractive forces.

Interaction Process: To select a possible bond between a pair of binding partners from both molecules, the user has to bring the intended binding partners close to each other. The distance between them has to be the shortest among all other binding partners.

Recognition Algorithm: When atoms of two molecules that can build a bond are moved closer than a specific distance d towards each other, the system assumes that a bond $B_{(k,l)}$ between the binding partners $A_{(0,k)}$ and $A_{(1,l)}$ is possible. If no other restriction applied, all possible bonds between all atom pairs candidates that are close enough would show up as seen in figure 5.3.

The algorithm selects the possible bond that has the shortest distance from the binding atom $A_{(0,k)}$ of the first molecule to the binding atom $A_{(1,l)}$ of the second molecule. To select only one possible bond from the previously already reduced set of possible candidates, the users have to simultaneously move and rotate both molecules in their hands in such a way that the desired binding atoms form the closest bond.

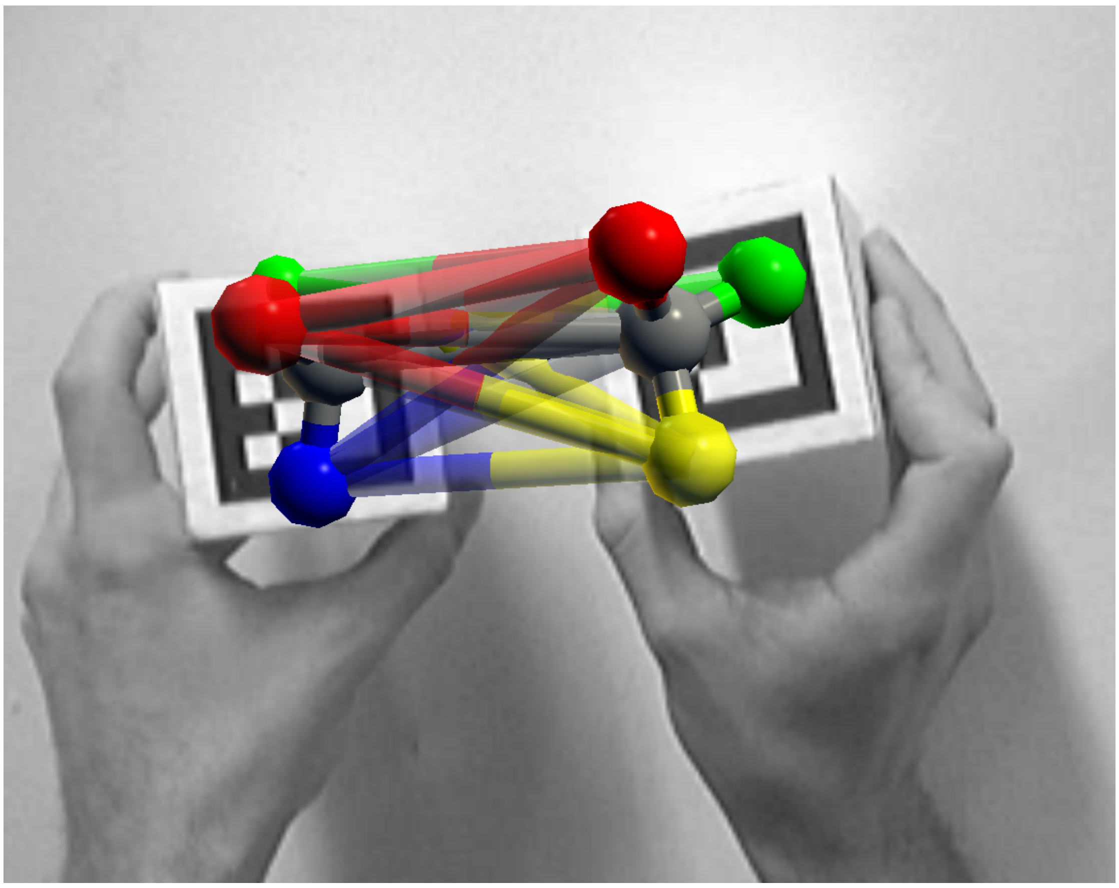


Figure 5.3: Exemplary display of all potential bonds between pairs of atoms on two molecules. Possible connections between two possible binding partners are visualized as semi-transparent cylinders when they are closer than a predefined distance.

```
// Calculate index k and l of the binding partners
int i, j, k, l;

// A[0][] and A[1][] are the Lists of possible binding
// partners in the first and second molecule

// Cut off distance of 10cm
float d = 0.1f;

// Initialize the distance
float tempdist = MAXFLOATNUMBER;

// Loop through all possible combinations
for(int i = 0; i < A[0].count(); i++) {
   for(int j = 0; j < A[1].count(); j++) {
       if (distance(A[0][i], A[1][j]) < MIN(d, tempdist)) {
           tempdist = distance(A[0][i], A[1][j]);
           k = i;
           l = j;
       }
   }
}

PossibleBond = CreateBond(A[0][k], A[1][l]);
```

In the visualization of the Augmented Chemical Reactions application, the shortest bond is shown as a pulsating, semi-transparent cylinder. Optionally, further, longer bonds can also be shown as semi-transparent but not pulsating cylinders to give users an idea of the possible choices.

5.1.3 Further Selection Methods, specific for Augmented Chemical Reactions

As an extension to the proximity-based selection, bonds can also be selected with respect to the strength of the attractive force – which is a function of distance as well as other parameters. With such weighting functions, bonds or other parts can be selected by calculating the values for each selectable part and then select only the part that fits best (minimum, maximum, specific value).

5.1.4 Experimental Evaluation

A user study was conducted to measure the differences in speed, precision, user acceptance as well as to discover the advantages and disadvantages between the shake-based method and the proximity-based method to select a bond.

Task: The user study tried to determine the differences in selection speed, error rate and acceptance by the user. Therefore participants had to select a series of predefined bonds using either the shake-based or the proximity-based method. Details and the test procedure are described below.

Experimental Setup: For the user study 19 people (7 female, 12 male) between the age of 20 and 51 years were asked to participate in the study. The users mainly had no experience with marker based tracking but they had no problems using the system.

Sitting in front of a table, the participants interacted with two cube-shaped markers (labeled „L" and „R") in the workspace in front of them. Both markers were augmented with identical ball-and-stick models of molecules without chemical semantics. Each molecule consisted of a center atom in gray color, surrounded by 4 atoms in red, green, yellow, and blue (see Figures 5.4 and 5.5). A Logitech QuickCam Pro 4000 was mounted above the user's head, on a microphone arm. The webcam was placed as closely as possible to the user's head and oriented towards the working volume (Figure 5.5) such that the users would not get confused regarding the translation of their hand movements in comparison to the image on the screen (as previously mentioned in section 4.5).

We used an Intel Core 2 Duo 2.33 GHz Notebook with 2 GB RAM and a NVIDIA GeForce Go 7900 GTX graphics card driven by Windows XP SP3. The gray scale camera image with a resolution of 320x240 was displayed and augmented on the display with a resolution of 1280x1024 pixels. Markers were glued on top of cubes made of polystyrene. The system is implemented using the Microsoft's® XNA® game developing framework, the UbiTrack library [39] and the TINKER Molecular modeling package [74] for the simulation process.

Test Procedure: The participants had to select a series of predefined bonds using the shake-based or the proximity-based methods. In the shake based method a shaking gesture performed by the left and right hand was used to cycle through the five atoms of the left and right molecule, respectively, to define the binding partners. After the binding partners were defined, the users had to bring both binding partners close to each other, such that the bond between both appeared. In the proximity based approach, the nearest pair of atoms of the two molecules were taken to select the desired bond.

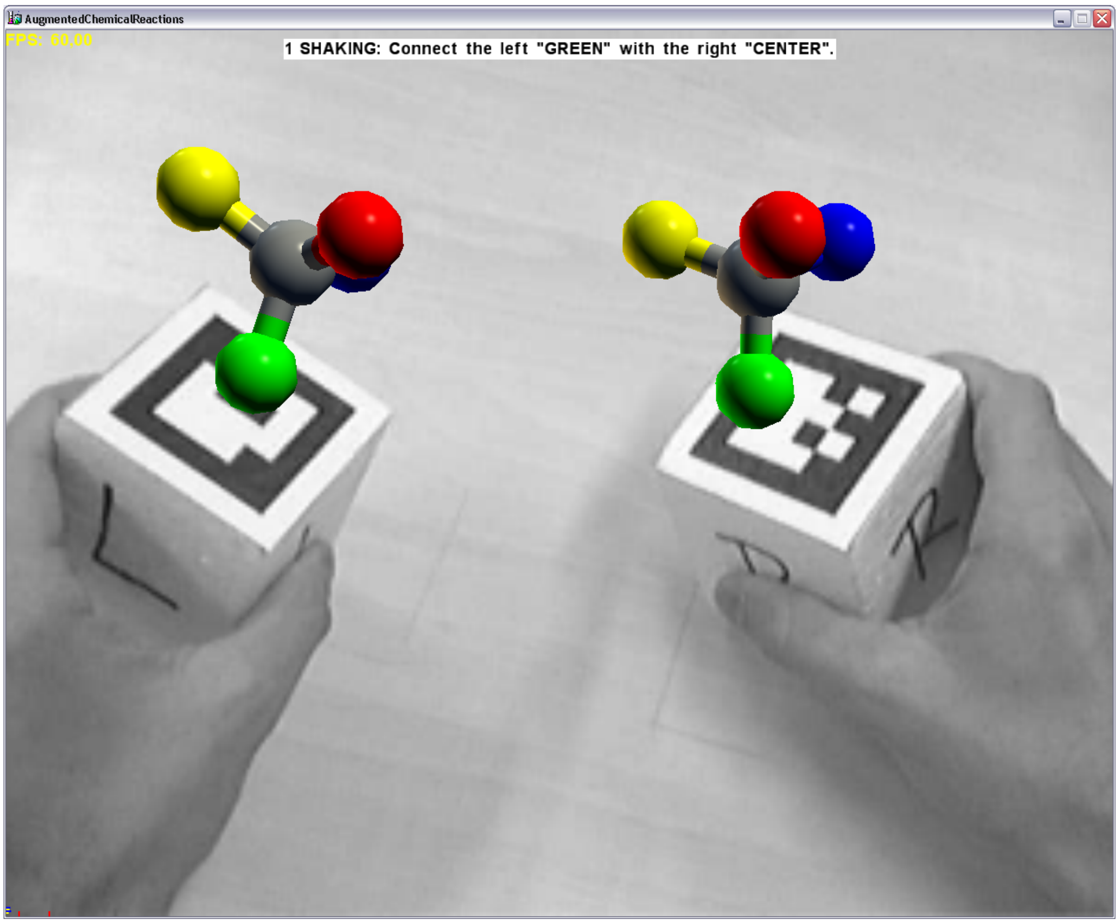

AugmentedChemicalReactions
FPS: 60,00
1 SHAKING: Connect the left "GREEN" with the right "CENTER".
L
R

Figure 5.4: Molecule layout

First, users were given a brief introduction to the topic of selecting bonds between two molecules which are directly controlled by the tangible optical markers. Users then had to fill out a first questionnaire, inquiring about general information, such as age, gender, color blindness and experience with marker tracking. No user had problems with color recognition, so no one had to be excluded from the study.

The participants had to select a specific bond by selecting a pair of atoms from two molecules. The desired combination was shown in an instruction line on the screen in front of their work area together with the augmented image of the camera above their head as described above. With displaying this message the timer was started. Users had to use both methods in the evaluation. After using the first method 24 times, they had to use the other method for further 24 combinations.

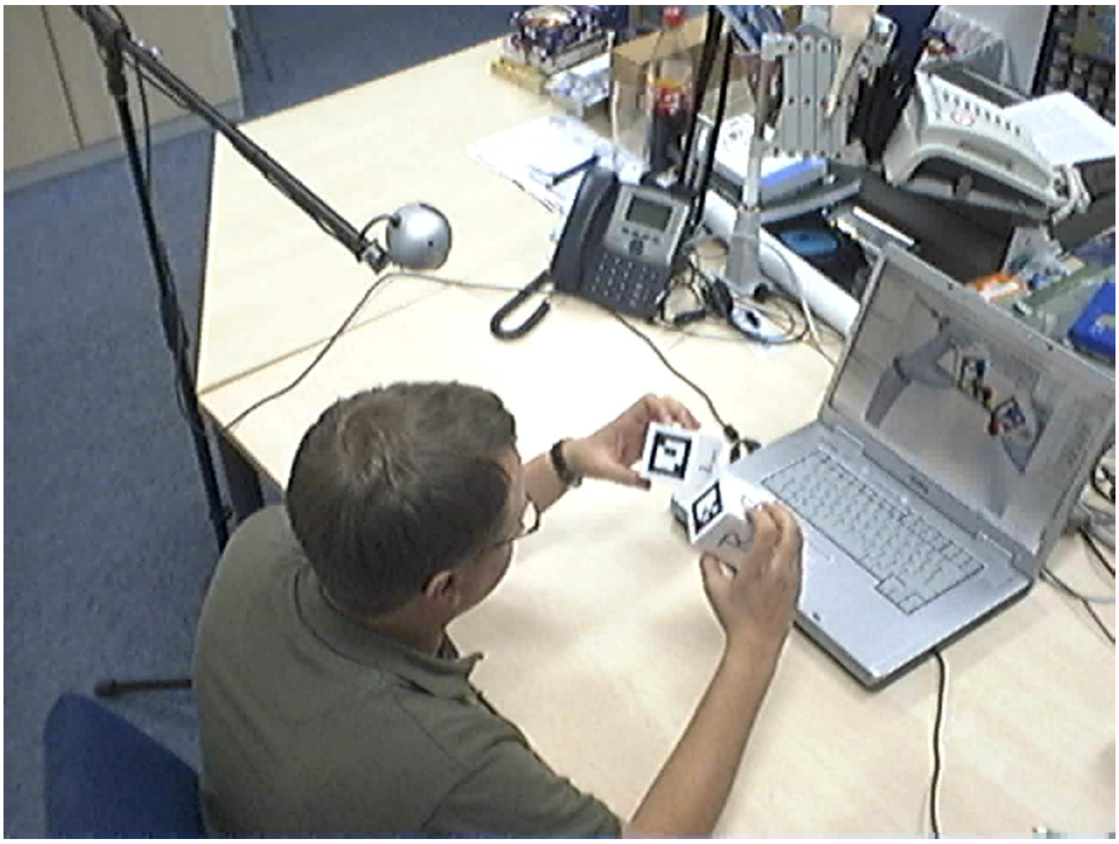


Figure 5.5: Experimental setup

Shake-based Method: For the shake-based method, the participants had to select the correct bond by shaking the markers to cycle through the binding atoms of the molecules. Each shaking gesture resulted in a change of the selected atom of the respective left or right molecule, cycling through the five atoms in the order of RED, GREEN, YELLOW, BLUE and CENTER. The selected atom was highlighted by pulsating its transparency. After selecting the two atoms by shaking, participants had to hold the atoms closer than a specified distance to see the bond, shown as a semi-transparent cylinder. In order to remove confounding factors as much as possible, we used a Wizard-of-Oz technique for users to confirm their selections: they were instructed to say *Done* as soon as they felt that they had correctly specified their selection with the shake-based or proximity-based method. The experimenter then hit a key on the keyboard to confirm the selection. For the performance analysis, the time was taken that the user needed to arrive at the target bond and say *Done*. False combinations were recorded but not taken into account for the analysis of the needed time.

Proximity-based Method: For the proximity-based method, the users were first given the instruction of the desired combination on the screen. The molecules had to be moved toward each other to bring up a connection in the form of a semi-transparent cylinder between the nearest atoms of both molecules. When the users thought that the correct combination was established, they had to say *Done* to complete the task. The time that the participants needed to perform this task was measured as well.

After each variant, the participants had to fill out a SUS questionnaire [14]. At the end of the study, the users completed a questionnaire regarding subjective impressions of each method.

Test Design: A within-subject, repeated measures single-session design was used in this study. Each session lasted about fifteen minutes, including introduction and questionnaires. The session was divided into two parts for both methods. In each part, the participant had to select 24 combinations (5 x 5 atoms, excluding the combination CENTER-CENTER). This set of 24 combinations was permuted for each user and for each method. After 10 participants the order of the methods was switched to suppress dependencies on a confounding learning effect. Before the first set of 24 combinations users could familiarize themselves with each method by selecting two combinations from each method.

The formulated two hypotheses were:

- H1: Proximity based selection will be faster than the shake based approach.
- H2: Shake based selection will be faster than the proximity based approach, when only bonds with the center atom are requested.

Results: To investigate the performance of both methods, both the time the participant needed to select the correct combination and the error rate of both methods were analyzed. The error rate of our shake recognition algorithm was also investigated.

Selection Time and Error Analysis: A two-tailed t-test was used for repeated measures on the mean time the users needed to select the correct bond. There is a significant difference in selection time when looking at all performed bonds for $\alpha = 5\%$ ($t(639.08) = 5.358, p < 0.001$), thereby supporting hypothesis H1. When considering all bonds, the proximity based approach was in mean 1.82 seconds faster than the shake based method. Yet, when looking at only those bonds that involved a CENTER atom, the shake based method was in mean 4.96 seconds faster than the proximity based method ($t(148.85) = 5.804, p < 0.001, \alpha = 5\%$). In conclusion, hypothesis H2 is accepted. On the other hand, when analyzing only bonds between outer atoms, the shake based method was in mean 4.68 seconds slower than the proximity based approach ($t(399.11) = 23.294, p < 0.001, \alpha = 5\%$). (see Fig. 5.6)

This leads to the consequence that a combination of both methods can eliminate those drawbacks and use the advantages of both. This is further explained later in the „Discussion of the Selection Methods" 5.1.5.

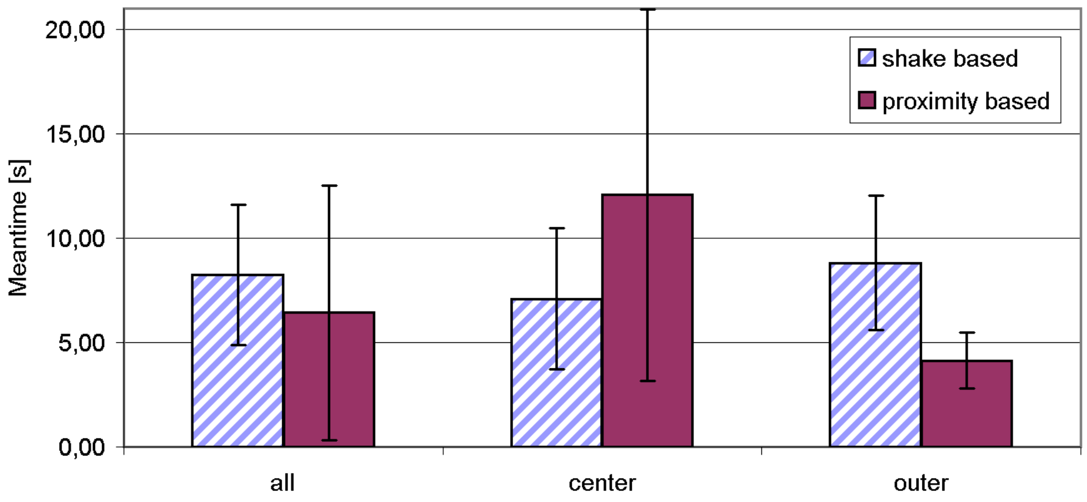


Figure 5.6: Comparison of the mean selection time between shake based and proximity based approaches with standard deviations

To analyze the error rates, a two-tailed t-test was used on the average errors users made for each method. An error was made by the users when they selected a bond between the wrong atoms. When regarding all bonds, there is a significant difference of 1.58 in mean errors per user for $\alpha = 5\%$ ($t(21.75) = 3.067, p < 0.01$), with the shake based method being worse. Both methods also have a significant difference of 1.47 in mean errors per user for $\alpha = 5\%$ ($t(19.33) = 3.236, p < 0.01$) when considering only bonds with center atoms. Only for bonds using only the outer atoms, there was no significant difference of the average errors per user for $\alpha = 5\%$ (see Fig. 5.7). This also shows that it is difficult for the users to correctly select a bond with inner atoms. When outer atoms are to be bonded, there is statistically no significant difference between the preciseness of both methods.

Error Rate Analysis of the Shake Recognition Algorithm: To analyze the robustness of the shake recognition algorithm, the application recorded all the position and rotation values of the user study sessions including the timestamps. With this information the movements of the users could be replayed and analyzed in detail. To calculate

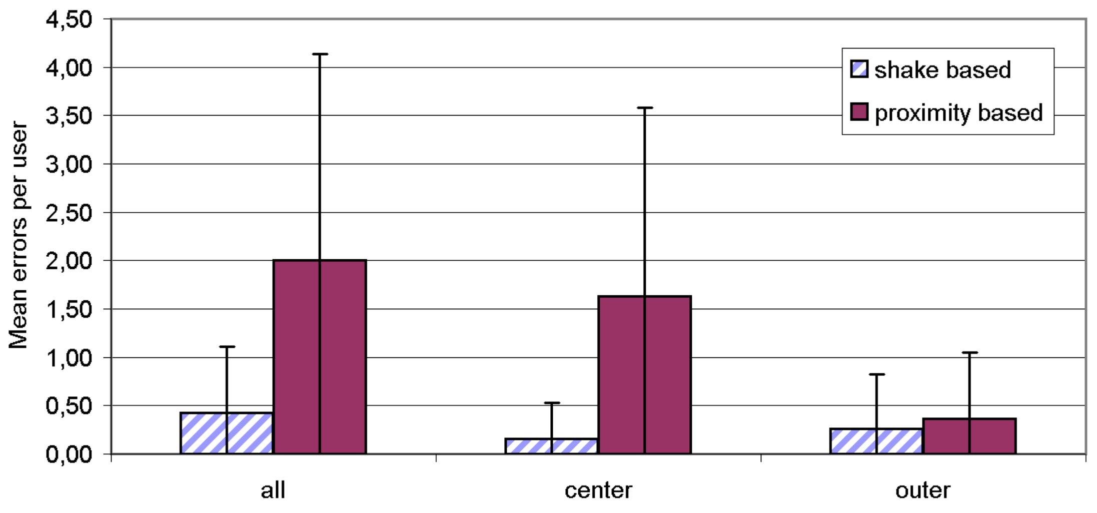


Figure 5.7: Comparison of the average error rates per user between shake based and proximity based approaches with standard deviations

the error rate, all shake gestures performed by the users were counted and compared with the recognized gestures. Unrecognized gestures as well as twice recognized gestures were counted separately. From the entire set of 2265 performed shake gestures, 58 gestures were counted twice and 151 were not recognized. This is an overall error rate of 9.2% – which is high. Yet, it was not irritating or disturbing to the users during the study. On average, a user performed 119.2 shake gestures, with 2.39% (standard deviation of 2.28%) being recognized twice and 6.33% (standard deviation of 6.28%) being not recognized, as seen in figure 5.8. While looking at the recorded values and the recognized gestures, it shows much space for improvement.

Subjective Results: From oral interviews with the test subjects, the thesis was confirmed that with the proximity based method it was very difficult to form a bond of an outer atom with a center atom. The participants also mentioned that the shaking method was a bit slower but they could select the desired combinations more precisely. In the questionnaire, users were asked to give grades for each method on a 6-point Likert scale (with 1=best to 6=worst) for like/dislike, ease of use/difficulty, fast selection/slow selection, accurate selection/inaccurate selection and the difficulty to select combinations with a center atom.

The analysis of the questionnaire shows (Fig. 5.9) that the users liked both methods and thought that both methods were easy to use, although they thought the shake based approach was a bit easier to use. Users stated that they felt that they were on average

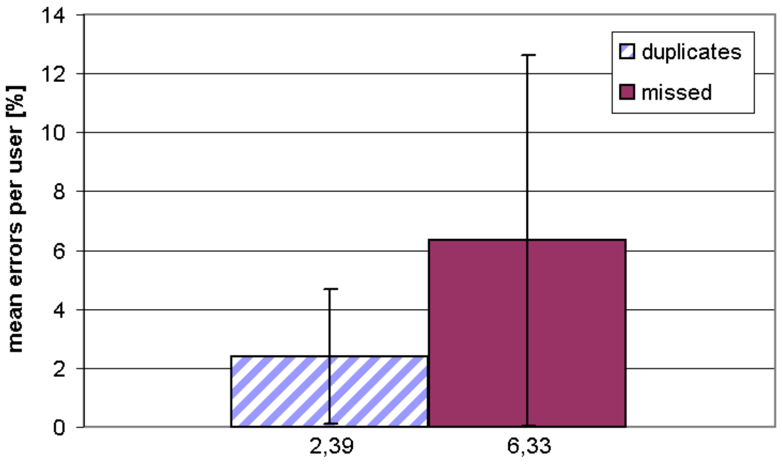


Figure 5.8: Mean errors in % per user in the shake recognition algorithm of double recognized and not recognized shake gestures with standard deviations.

equally fast with both methods. Regarding the accuracy and the difficulty to select a center atom for the bonds, the questionnaire reflects that the shake based approach is more accurate and easier to use for selecting the center atom.

The relatively high SUS-values (Fig. 5.10) show that both systems were accepted by the users.

5.1.5 Discussion of the Selection Methods

In contrast to selecting virtual objects by pointing at them, selecting parts of virtual objects that one already controls with both hands, cannot be done that way. Here there is no additional hand to control the pointing device. There have to be other ways to select those parts. I focused in my thesis on developing and evaluating different gesture based approaches. Other methods not using additional hands could have been voice control, buttons on the tracked real objects or buttons on the floor that can be controlled by the feet.

Combining all results from the experiment, it can be seen that the selection by enumerative toggling (see 5.1.1) and the selection by proximity (see 5.1.2) as described above also have their specific drawbacks. When using the shaking method to cycle through the possible binding partners, it can become quite exhausting when this has to be done more often and for larger sets of possible binding partners. Furthermore, its performance deteriorates rapidly with increasing numbers of atoms.

When binding molecules only with atoms on the outer shell, the proximity based ap-

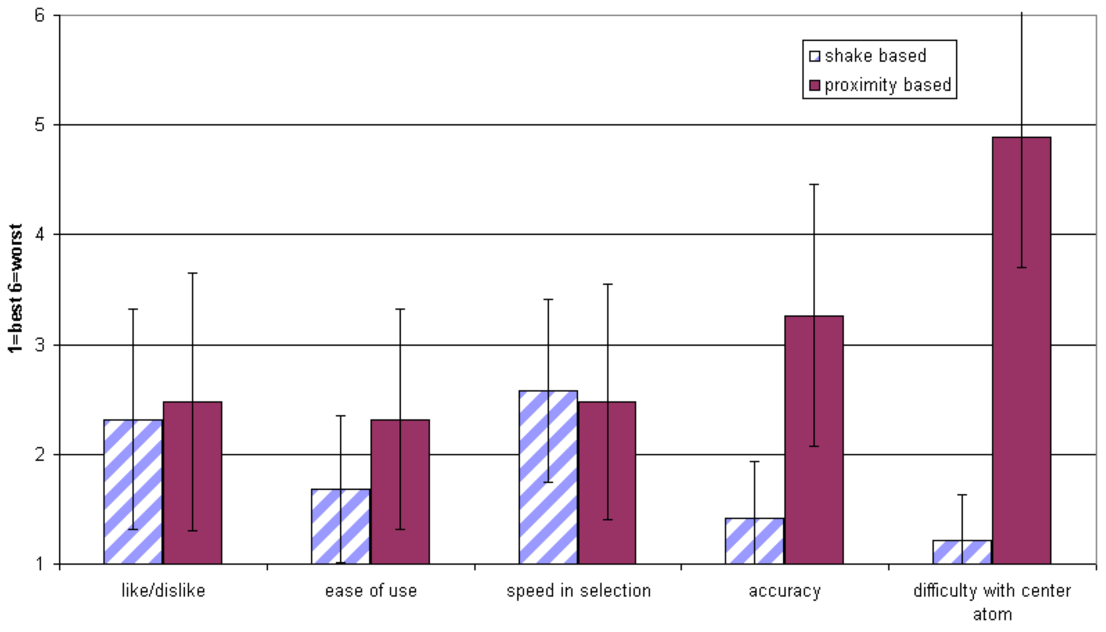


Figure 5.9: Evaluation of the questionnaire regarding the shake based and proximity based approaches with standard deviations

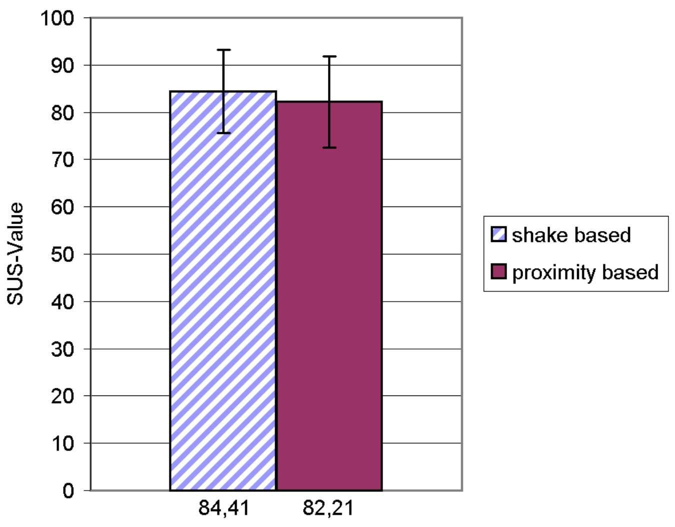


Figure 5.10: Mean SUS value for the shake based and proximity based approaches with standard deviations

proach works very well since the bonding atoms are easy to reach. But this method also has its drawbacks, when trying to bind with atoms that are sheltered by surrounding atoms, the proximity based method deteriorates dramatically since the outer atoms often have a shorter distance to the atoms of the other molecule than to the center atoms. Thus it is very hard to establish a bond with center atoms. In this case it can become difficult to arrange both molecules such that the desired binding partners form the shortest bond between both molecules.

It might be a very good solution to combine the advantages of both methods: the preciseness of the shake based method to toggle between very specific, hard to reach options, and the 3D-immersiveness of the proximity based method to use spatially consistent hand motions to define bonds between the closest atoms. It seems to be a good solution to use the shake based method to switch between task contexts on a higher level, i.e. to switch between the inner and outer part of the molecules (or cycle through more shells if those exist, or between other kinds of substructures), and then using the proximity based approach for detailed, direct selection between atoms in the selected sub set.

When dealing with large or complex molecules where binding partners are surrounded by others, we can introduce layers. Layers can be spherical, arranged in a way like the layers of onions. The shaking gesture cycles through the set of layers and selects the layer which contains the desired binding partner. Using now the proximity method, there is a highly reduced set of binding partners where the possibility of disturbing surrounding binding partners is minimized. Only bonds can be created between binding partners out of the selected layers from both molecules.

Partitioning the set of possible binding partners in smaller sets by using layers is important. It has to be done in advance either by hand or automated. The way how the set is partitioned strongly depends on the weighting function. For proximity, spheres could be the right choice but for other functions this could be completely different. Defining such layers automatically can be a complex task that was not covered in this thesis.

Such selection methods can also be applied outside the field of chemical visualizations. Selection by proximity and shaking are very good techniques which also can be used in several other areas. One can think of selecting parts of a complex structure by using another tracked probe. In this scenario the part of the object which has the closest distance to the probe will be selected. The improvement of introducing higher level structures like shells with subsets of elements of the complex structure can also be applied here and can lead to a higher level of usability.

The importance of this issue has recently been recognized in the 3DUI community and formulates the central problem of the annual 3DUi!2014 contest [41].

5.2 Confirmation

After the desired bond has been selected from the large set of possible bonds with one of the methods described above, the user needs a comfortable way of actually confirming that bond between the two partners. At this point, the challenge is how to press for example a button on the keyboard to confirm the selection, when both hands are already occupied by controlling the physical, tracked objects. This section specifically addresses this issue. It can be generalized to any kind of application using two-handed interaction, especially those that require a non-stationary working environment. Within the context of chemical modeling, two separate methods of confirming a selected bond were developed and investigated, when both hands are already performing a two-handed symmetric interaction task.

The first method for confirming a bond is a waiting method, where the user is to perform no motion on either of the two binding partners. The second method is a back&forth motion gesture where both hands have to execute a dual swinging movement.

The two methods have been evaluated in a user study, showing that the first technique, holding still, outperformed the swinging technique. Results of objective and subjective measures are presented and discussed in detail.

5.2.1 Related Work on Triggering Events

Usually, events are triggered with tools such as the Pinch Gloves^{TM} [21]. With such tools the user can select or move objects by pinching two or more fingers and then moving the tracked hand. Buttons are commonly used on 3D input devices for event triggering while moving the input devices. Geiger and Rattay [28] also use buttons at the ends of their TubeMouse – a flexible two-handed 3D input device to modify flexible virtual objects.

Buttons are not always the only solution. Choumane et al. [17] developed a system that provides selection and pick-release without the use of a physical button and tested this against the use of buttons. They implemented a gesture to trigger the selection or pick-release by analyzing the trajectory when pointing on or moving a virtual object.

Sometimes, a passive approach of event triggering is more applicable for a specific use. For instance, some devices trigger events by detecting that no movement has occurred. An example is the RES.Q mobile device from Swissphone [23], that sends an alert when the device ceases to be moved. This device is meant to be used by employees working alone so that they get immediate help when they cannot move anymore in an emergency.

Another well-known system is the tooltip element used in programming [93]. With this system, the user moves the mouse pointer over an element. When the mouse pointer is not moved for a certain time, an information element appears and provides additional information to the underlying element.

Steed [78] presented an elaborate discussion on mechanisms for the confirmation of selections by dwell time based pointing at objects. Among others, he stated that in the presence of jitter or errors, it is unlikely that the wrong object will be selected because the other object is unlikely to fulfill the dwell time constraint. Such holding still approaches with dwell time are implemented in some applications.

While most examples focus on providing help or information to the user, such methods of event triggering can also be used to activate the transition to the next logical state of a system process flow. In the context of combining virtual objects, the next step after selecting the desired link is the confirmation of the selection.

5.2.2 Confirmation by Holding Still

While inspecting molecules, the user generally moves the molecules back and forth a bit to get a glance from other perspectives. There thus is a continuous flow of motions. This first technique to confirm the creation of a bond makes use of this. In contrast to normal gestures, in this gesture the user has to do nothing.

Metaphor: The gesture implements a gluing metaphor. When users have selected a possible connection, they stop moving the tangible props and hold still for a certain time to confirm the selected connection. The gesture thus is similar to holding two objects together such that the glue between them can dry.

Interaction Process: The confirmation process consists of two levels: In the first level, the system observes the user at any time. When no significant motion occurs in a time interval of the last T_1 seconds, the system hypothesizes that the user does not move the props and that a holding-still gesture is in progress. In the second level, it then indicates this hypothesis – as long as it is valid – by showing a timing-out glyph on the display. This gives the user the chance to negate this hypothesis by making a small motion before the second time-out T_2 is passed. When the hypothesis is still valid after T_2, the confirmation gesture is completed.

Recognition Algorithm: Recognition of a confirmation gesture by holding still, depends on the amount of motion M_t at time t that occurred during the time interval

$[t - T_1, t]$. In the context of Augmented Chemical Reactions, where a connection between two controlled molecules has to be made, the recognition algorithm needs to inspect the positional information of the two tracked props A and B, $\vec{a} = (x_A, y_A, z_A)^T$ and $\vec{b} = (x_B, y_B, z_B)^T$, accumulated over the T_1 seconds of the time interval $[t - T_1, t]$.

The recognition algorithm has to counterbalance two critical factors in order to be intuitively usable.

The first factor is the selection of optimal time-out intervals: On the one hand, the gesture recognition algorithm must not trigger the confirmation too quickly when the user only holds still to inspect the actual layout. On the other hand, the recognition time-out has to be short enough for the user not to become impatient or tired – especially when several selections and confirmations have to be executed in a short time.

The second factor concerns the sensitivity of the recognition algorithm to small motions: In an insensitive setting, relatively large motions during object inspections may still fall below the threshold and thus cause the recognition algorithm to wrongly hypothesize that a confirmation process has started (thus causing the timing-out glyph to appear and forcing the user to repeatedly perform extra jittering motions just to negate the hypothesis). In a too sensitive setting, on the other hand, every small tremble leads to a reset of the confirmation process and as a consequence to a frustrated user.

The algorithm has to inspect the movements of the props to detect the intention of the user. It was a difficult task to find the right attributes and reference systems for the values used in the algorithms. Should the movements of the molecule centers be investigated separately as in *Approach 1*, or should the length between the two connection points (bond length) be analyzed as in *Approach 2*, or – to suppress human trembling or sensor jitter – should only the maximum and minimum length between the connection points (bond length) be analyzed in a time interval, as in *Approach 3*? To obtain a robust algorithm, those three development iterations were needed.

Approach 1: The first implementation of the holding still method calculates M_t by separately measuring the trajectory length of the movements of each of the two props A and B in the time interval of the last T_1 seconds.

$$M_{At} = \sum_{i=t-T_1}^{t} d_{Ai}, \qquad d_{Ai} = ||\vec{a}_i - \vec{a}_{i-1}||$$

$$M_{Bt} = \sum_{i=t-T_1}^{t} d_{Bi}, \qquad d_{Bi} = ||\vec{b}_i - \vec{b}_{i-1}||$$

with t being the actual point in time and d_{Ai} and d_{Bi} being the Euclidean distance between two subsequent measurements of $\vec{a}_i$ and $\vec{b}_i$. Therefore each tracking measurement is stored in a queue with a time stamp of the measurement. Elements in this queue that are older than T_1 seconds are removed from that queue. If the maximum of both measures M_{At} and M_{Bt} is below a specified threshold Len, the system assumes that there is no movement and that a holding still gesture has been started.

$$M_t = \max(M_{At}, M_{Bt})$$

The algorithm continues to calculate the motion value M_t. It triggers the confirmation, if M_t stays below the threshold Len for the second time interval with the length T_2.

The thresholds were determined by having several people hold the props while trying not to move them. Parameter settings $Len = 0.75cm$, and $T_1 = T_2 = 0.5\,\mathrm{sec}$ worked well. Because different tracking systems and different setups induce different jitter or human trembling, those values have to be determined again every time the setup or the tracking systems change.

Approach 2: To improve the detection of no movement, the second implementation computes the distance between the two connection points $\vec{va}$ and $\vec{vb}$ at which the virtual objects on props A and B are to be linked. When users hold two props in front of them, they mainly move them synchronously, because a rotation or movement of the torso affects both arms at the same time with the same amount. As a consequence, the relative distance between the binding partners does not chance so much. The algorithm thus accumulates the changes of this distance at every point in time t for the last T_1 sec.

$$M_t = \sum_{i=t-T_1}^{t} |d_i - d_{i-1}|, \qquad d_i = ||v\vec{a}_i - v\vec{b}_i||$$

To achieve this, the algorithm stores the change of the distance of the connection points with the corresponding time stamp for each time step in a queue and discards elements, that are older than T_1 seconds. If $M_t < Len$ for $Len = 0.75cm$, and $T_1 = 0.5\,\mathrm{sec}$, the system assumes that the props are not moved. The algorithm also continues to evaluate M_t and triggers the confirmation, if M_t stays below the threshold Len for the second time interval with the length T_2.

With this approach, the user can move both probes in parallel without resetting the algorithm, whereas in the first approach, the process would already have been reset.

Approach 3: Approach 1 and 2 have problems with sensor jitter and human trembling. The jitter and trembling can produce large distance changes, $d_i - d_{i-1}$, especially in Approach 2 when the connection points $\vec{va}$ and $\vec{vb}$ are far away from the tracked center of the props or the point at which the user holds the prop. This can cause the accumulated changes to exceed the threshold Len. The third implementation overcomes this problem by inspecting the minimal and maximal link length during the last $T_1 = 0.5\,\text{sec}$ from the actual point in time t. As in the previous implementation, this version also stores the distances of the binding partners in a queue and discards elements, that are older than $T_1 = 0.5\,\text{sec}$. Yet we do not have a look at the sum of the deltas, but rather inspect the lowest and the highest stored value of the distances.

$$M_t = \max(d_{t-T_1}, ..., d_t) - \min(d_{t-T_1}, ..., d_t)$$

When the difference M_t between both extreme values is smaller than $Len = 0.3cm$, the system assumes that the props have not been moved. A jitter or tremble below $0.3cm$ (see Parameter Tuning for details) in its amplitude is therefore not recognized as a movement. Larger outliers, caused by wrong detections of the positions or orientations will disturb the detection process. When such errors occur more often, the tracking setup should be reviewed and improved, because those errors cannot be distinguished from intended sudden movements performed by the users. Here again, the algorithm continues evaluating M_t and it triggers the confirmation, if M_t stays below the threshold Len for the second time interval with the length T_2.

Parameter Tuning: For a robust use in real applications, the parameters of the recognition algorithm, Len, T_1 and T_2 have to be tuned not only to get successful recognition of intended holding still gestures (avoidance of false negatives), but also to carefully distinguish themselves from regular small motions while users are inspecting a potential connection (avoidance of false positives). In an insensitive setting of Len, relatively large motions during object inspections may still fall below the threshold and thus cause the recognition algorithm to wrongly hypothesize that a confirmation process has started (thus causing the timing-out glyph to appear and forcing the user to repeatedly perform extra jittering motions just to negate the hypothesis).

In a too sensitive setting, on the other hand, small trembles lead to a reset of the confirmation process. For optimal settings for the gesture recognition algorithm, T_1 should not be too small, because short rests in the movement then lead to a frequent appearance and disappearance of the timeout glyph. T_2 also must not be too short, not to trigger the confirmation too quickly when the user only holds still for a moment to inspect the actual layout. Yet, the recognition time-outs $T_{1,2}$ have to be short enough for the user not to become impatient or tired – especially when several selections and confirmations have to be executed in a short time.

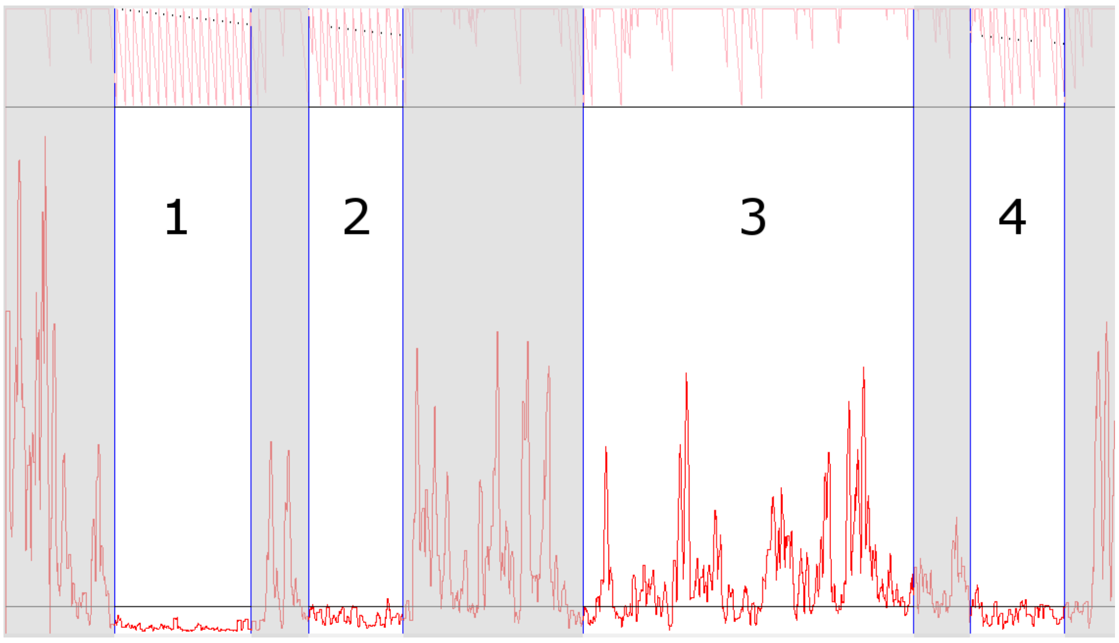


Figure 5.11: Analysis of the amount of movement during the tasks 1 to 4.

To optimize the parameter setting for the threshold Len beyond the first quick study of the previous section, the settings were re-tuned by presenting approach 3 to test persons. The tracking data for two props were recorded while the subjects performed several tasks.

- The first task was to hold the props still for about 15 sec while placing the wrist on the table. The stabilizing table should generate a relatively low movement.
- The next task was to hold the props still for about 15 sec while holding the props in the air without support.
- In the third task, the test persons had to move both virtual objects close to one another to entice a potential connection. While sustaining the potential connection, they had to calmly inspect and count the different colors on the virtual object. This gave an indication of the upper limit of the sensitivity to suppress unwanted confirmations while an inspection task is executed.
- As a last task, the test persons had to repeat the second task and hold the props in the air, again without supporting the arms with any kind of fixation. After executing all previous tasks, a slightly larger movement was expected (compared to second task) due to increased tiring.

Figure 5.11 shows the calculated bond length during the tasks 1 to 4. The lower horizontal line is the threshold value Len which was automatically calculated. The lighter graph at the top represents the progress of the confirmation. The line always starts from the top and proceeds down towards the next horizontal line. When this line is reached (after T_2) the confirmation is triggered. In this example one can see that during the tasks 1, 2 and 4 there are lots of correctly detected confirmations due to „no movement". In the area of task 3 – during inspection of the molecules – there are only some false positives.

The recorded data was analyzed to find the optimal distance threshold Len between the extremal values of detection algorithm 3 (described above). I also implemented an algorithm to automatically calculate this distance value. The algorithm measured the time where no movement is detected and the total time interval in which the user tried not to move. The algorithm was built on the assumption that it is more annoying for the user when the detection of the waiting gesture resets often if the user wants to hold the props still, than starting and displaying the detection of the gesture when the user only inspects the layout of the virtual objects. In the case of the false positive detection of a waiting gesture, the user can apply an extra jittering movement, to prevent the detection of the gesture. Whereas we observed some frustration when the gesture detection resets due to a small jitter or trembling.

Because of these facts, it is more important that the algorithm correctly detects no movement than a wrong start of the confirmation timeout while inspecting. In a discussion, it was agreed that the correct detection of „no movement" should be four times more important than the correct detection of „movement", in order not to frustrate the user by resetting the gesture often.

To easily calculate and find the distance value Len where the importance ratio 1:4 is met, I weighted the value of the correct detection of „movement" by a factor of four (See Fig. 5.12). The point where both graphs cross each other is the point where the time of correct detection of „no movement" is four times higher than the correct detection of „movement". This gives the desired threshold value for the Len.

To calculate the desired optimal distance value Len for the detection algorithm, I first measured the percentage of the time for which the algorithm correctly detected no movement. Next the percentage of the time was measured for which the algorithm correctly detected movement while inspecting. The values are computed for each distance value between $0cm$ and $1cm$. The resulting values for the percentages of the correct detection of no movement (red) and the correct detection of movement (blue), according to the distance value, are scaled with respect to the importance factor of four (as discussed above) and shown in fig. 5.12. Here the blue graph, representing the correct detection of movement, is scaled by the factor four in the direction to 100% the top because of it is of lower importance.

The optimal distance value for the gesture recognition is found by searching the inter-

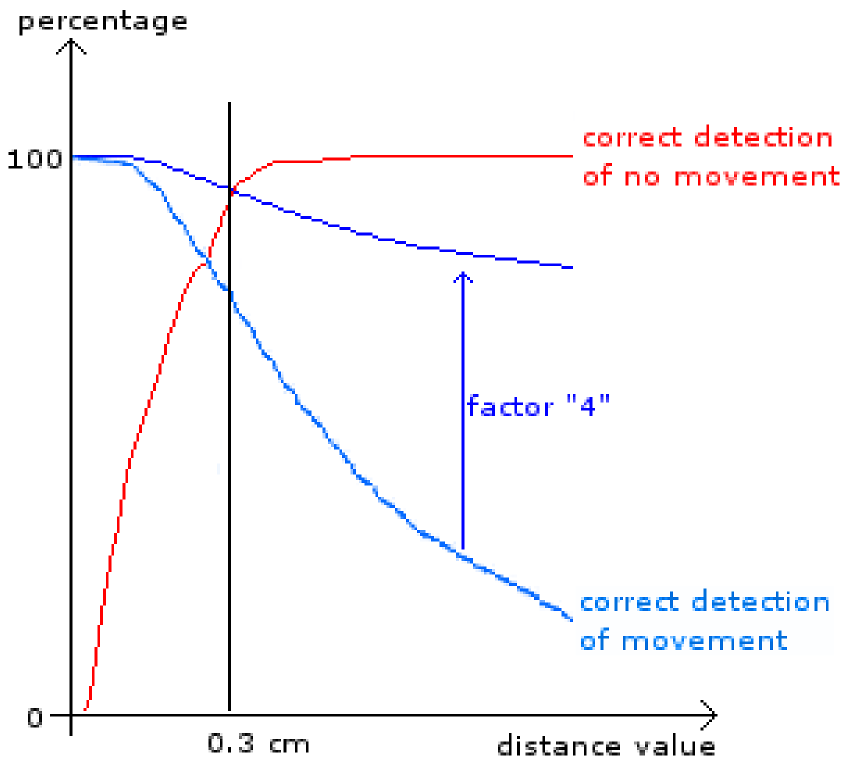


Figure 5.12: Visualization of the percentages of the correct detection of no movement (red) and the correct detection of movement (blue) according to the distance value

section point of the two graphs (Fig. 5.12). The determined factor is the suitable tradeoff between unwanted recognitions of „no movement" while inspecting and wrong recognition of „movement" while holding still.

The analysis of the recorded data for the three test users gave the optimal values of $Len_1 = 0.28cm$, $Len_2 = 0.19cm$ and $Len_3 = 0.22cm$ for the distance of the extremal values in the detection algorithm and thus matched the even more insensitive value $Len = 0.3cm$ that was used in the initial implementation of the algorithm. An insensitive setting is not as frustrating as a too sensitive setting when the user cannot complete a confirmation.

After investigating these three approaches separately, the variants were presented to an expert group to determine which one was better to use for a wider range of people. The last implementation (*Approach 3*) of this waiting method was the most satisfactory one and therefore was used for the user study.

Another option to detect no movement could have been to take the two parameters from the detection of the shaking in the previous section (Selection 5.1). When the values of the spatial spread and the length of the trajectory lay in the lower left part of the scatter plot (Fig. 5.2), the algorithm should detect no movement.

5.2.3 Confirmation by Performing a Back&Forth Gesture

For the second method, I developed a swinging gesture. To confirm a selection, the user had to perform a back and forth movement as visualized in figure 5.13.

Metaphor: This gesture implements some kind of a pointing metaphor. When the users want to say that their selected possible bond should be confirmed, they somehow like to say: *I like THIS bond to be created.* This back & forth movement, where the length of the selected possible bond is repeatedly shortened and elongated, stands for a pointing gesture, moving the binding partners towards each other.

Interaction Process: The user has to move the connection points (points where both objects are to be linked) twice towards and apart from each other. It does not matter if the connection points are first moved towards or apart from each other. This gesture therefore is called the back&forth method.

Recognition Algorithm: To detect this gesture, the system observes the last $2\,\mathrm{sec}$ of the movements. The gesture thus has to be finished within these $2\,\mathrm{sec}$. $2\,\mathrm{sec}$ for an upper limit to complete the gesture, prevents unwanted triggers due to continuous

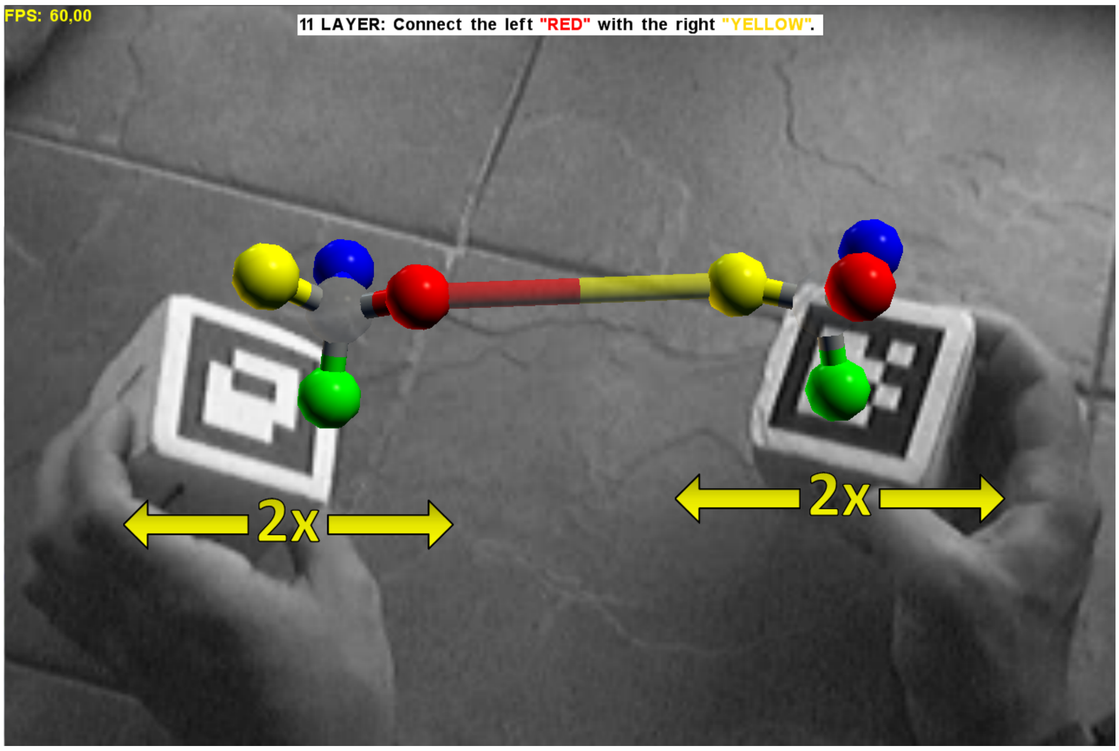


Figure 5.13: Visualization of the back&forth movement

slow movements by the users. If the user is too slow, the first directional motion of the movement is discarded and the user has to continue moving the props back or forth to trigger the gesture as long as he stays in the defined time interval. To prevent unintended confirmations by moving too quickly while rearranging the props in the hand, gestures that are executed faster than $1\,\mathrm{sec}$ are discarded. Those timings were chosen based on observations during tests with colleagues. After some testing with colleagues, the minimal length of each of the four motions was set to $1.5cm$, which appeared to be a reasonable distance.

When the desired possible connection is selected by the user, the detection algorithm stores the initial distance value of the object to be connected (see Fig. 5.14 (a)). The user can now start the gesture by moving the connection points in either direction. As a result, the distance value increases (decreases) when the props are moved apart from (towards) each other. When the absolute change in the distance value exceeds $1.5cm$ (see Fig. 5.14 (b)), the user has completed the first of the four necessary steps to complete the gesture. The props now have to be moved in the opposite direction (step two). The starting position for the next calculation is the turning point of the movement (see Fig. 5.14 (c)). From this turning point, the user has to change the distance of the objects again by at least $1.5cm$ (see Fig. 5.14 (d)). This back&forth procedure has to be performed twice to complete the gesture (step three and four).

After performing the gesture, the selected link is confirmed and the objects are connected.

Parameter Tuning: To find an optimal value for the back&forth gesture, I asked an expert group to perform the gesture explained above. They first had to select a connection between the two objects and then had to confirm this link three times by performing the back&forth gesture with short breaks in between. To analyze it, the movement data was again recorded. An optimal value for the minimal distance for a user to move before alternating the direction during the gesture should be smaller than the smallest amplitude measured in the recorded data. The value should also be larger than the amplitudes of any back&forth movement made during a normal movement or inspection of the object. The smallest measured amplitude in the recorded data was $1.9cm$, so the chosen value of $1.5cm$ in the algorithm is small enough to detect all gestures. The chosen parameters were good enough not to trigger the gesture accidentally with this test and also succeeded with the data recorded for the waiting method.

5.2.4 Experimental Evaluation

Both methods for confirmation were investigated in a user study to find out which method fits best with respect to performance and user acceptance.

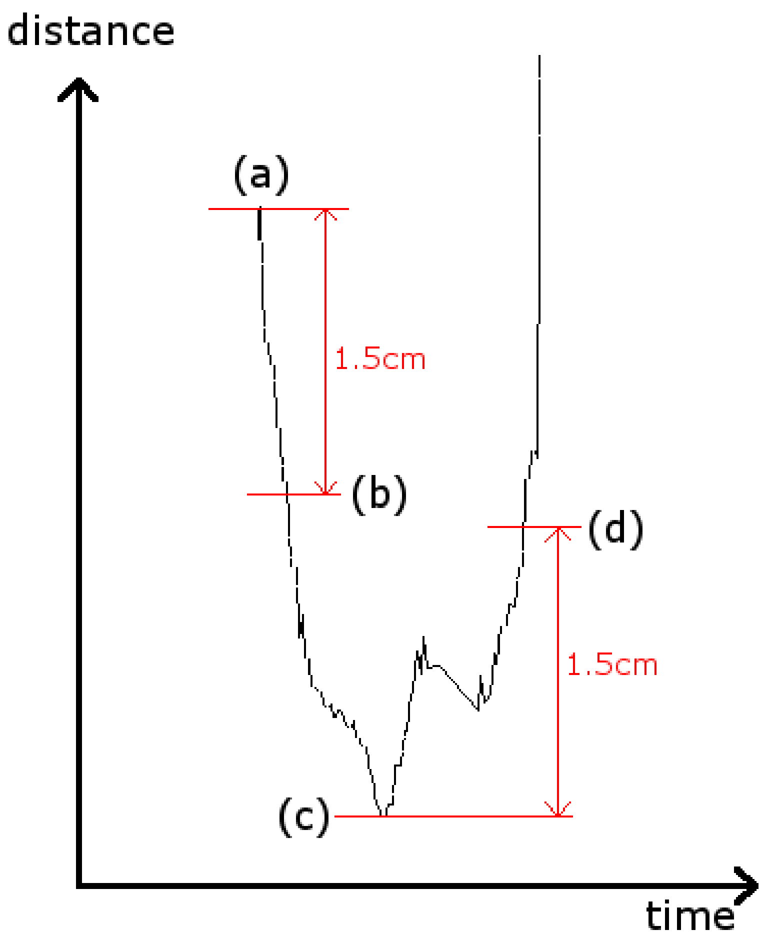


Figure 5.14: Visualization of the back&forth algorithm. The movement starts at point (a). The length of the bond is the vertical value (distance). The User has to change the length at least by $1.5cm$ (b). The starting point for the next calculation is the turning point (c). The user has to change the length of the bond again by at least $1.5cm$ (d).

Task: The user study tried to determine the differences in confirmation speed, error rate and acceptance by the user. Therefore the task was to first select a specific bond between two molecules and then to confirm these bonds by performing either the waiting or the back&forth method. Details and the test procedure are described below. For each session, the participants had to select 24 combinations using the waiting method and 24 combinations using the back&forth method.

Experimental Setup: For the user study, 20 persons were asked (8 female, 12 male) between the age of 17 and 64 which fits the target group for the Augmented Chemical Reactions application. Half the participants had „none" to „little" experience using 3D systems and the other half had „much" to „very much" experience. Even when the users mainly had no experience with marker based tracking, they had no problem in using the system.

Sitting in front of a table, the users interacted with two marker-labeled cubes (named „L" and „R") as shown in figure 5.15. A webcam next to the user's head pointing towards the working area captured the markers. The camera image was augmented by the virtual molecules and displayed on a notebook screen in front of the working area.

The two identical ball-and-stick models of molecules (as described in the „Experimental Evaluation" 5.1.4 of section „Selection" 5.1) were displayed above the markers. They had no chemical semantics such that the user could focus on the user interface itself. Each molecule consisted of a center gray-colored atom, surrounded by 4 atoms in the colors red, green, yellow, and blue (see Figure 5.16). Based on results of the previous user study about the selection methods (see section 5.1.4), I improved the selection algorithm by introducing layers in the molecules. Each molecule consisted of two layers, an inner layer with the center atom and an outer layer with the four surrounding atoms.

An Intel Core 2 Duo 2.33 GHz Notebook was used with 2 GB RAM and a NVIDIA GeForce Go 7900 GTX graphics card driven by Windows 7. A Logitech QuickCam Pro 4000 was mounted on a microphone arm close to the user's head. A gray scale camera image with a resolution of 320x240 was displayed and augmented with the colored molecules on the display running a resolution of 1280x1024.

Test Procedure: At the beginning of each session, the user was given a detailed description of the physical setup as well as a short overview of the methods to select and confirm bonds between two marker-controlled virtual molecules. After the introduction, the participants were asked to fill out the first questionnaire which included questions on demographic data, experience with 3D systems, marker tracking and further questions on color blindness and general constitution.

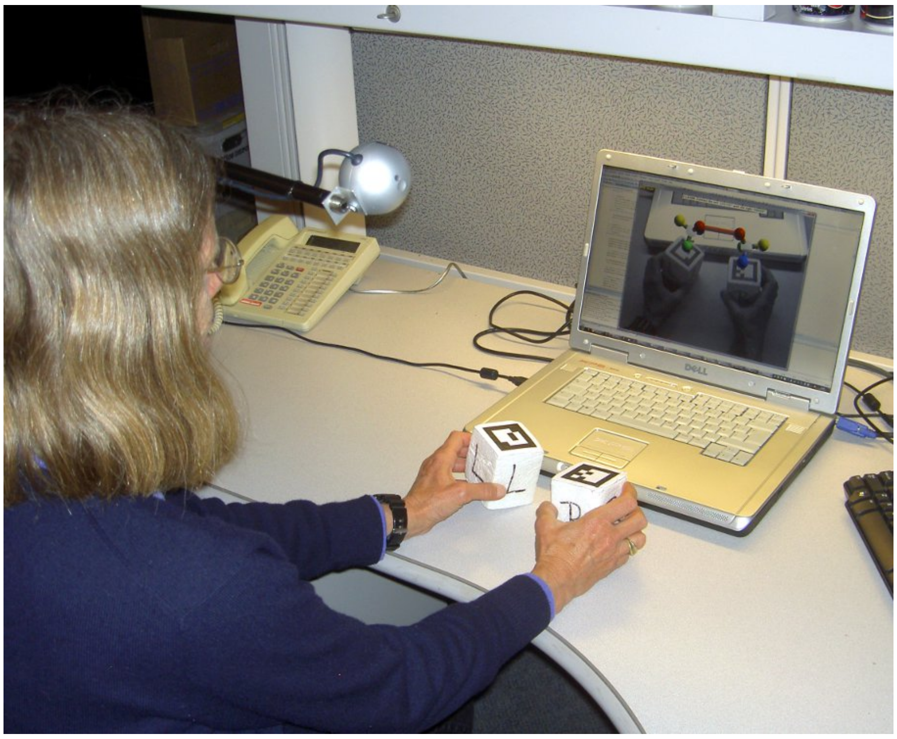


Figure 5.15: Experimental setup of the user study

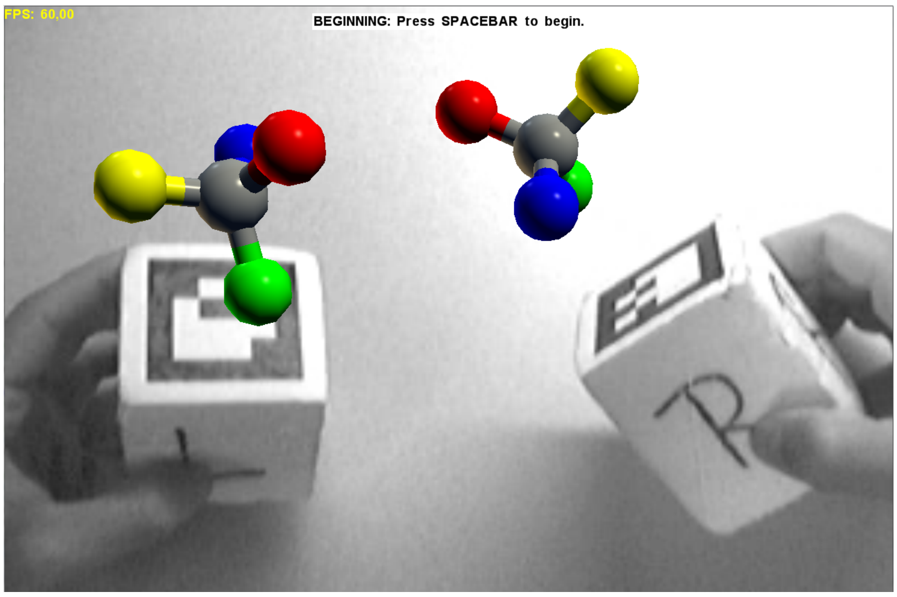


Figure 5.16: Markers with the augmented molecules for the user study.

The user study required each participant to first select a possible bond, predefined by the system and communicated by a text at the top of the display, such as „Connect the left „BLUE” with the right „CENTER””. They then had to confirm this bond by using the two described methods. The participants had to select and confirm 24 combinations by the waiting method and 24 combinations by the back&forth method.

For the waiting method, the participants first had to select the desired combination of atoms of both molecules by using a proximity based method in combination with the layer selection by shaking. After selecting the desired bond they had to keep the markers still to confirm their selection. This had to be repeated in a total of 24 combinations.

For the back&forth method, the users selected the desired bond as in the other method and then performed the back&forth gesture. For this gesture the users had to move the binding partners twice towards and apart from each other. This also had to be repeated for 24 combinations.

To analyze the performance of each method, the time interval starting with selecting the right combination of atoms was measured until the selection was confirmed. Although false combinations were recorded, they were not taken into account for the analysis of the time needed for the confirmation task.

After using each confirmation method, participants were asked to complete a SUS questionnaire [14].

At the end of the study, the users completed a questionnaire to provide their subjective impressions of the two methods.

Test Design: For this evaluation a within-subject, repeated measures single-session test design was chosen. Each session lasted about 15 to 25 minutes, including the introduction and the completion of the questionnaires. The session was divided into two parts, one for the waiting method and the other for the back&forth method. In each part, the participant had to select 24 combinations (5 x 5 atoms, excluding the combination CENTER-CENTER). This set of 24 combinations was randomly shuffled for each user and for each method. After 10 participants, the order of the methods was switched to counter-balance and suppress dependencies on a confounding learning effect. Before the first set of 24 combinations, users were shown both methods and provided time to familiarize themselves with each method by selecting two combinations from each method.

Results: The main goal of this user study was not the performance of the methods, because the waiting method is highly dependent on the timeout value and the back&forth method is limited in speed because the time to execute the gesture has to be in minimum 1 sec to suppress false positives. Yet since the timeout value for the waiting

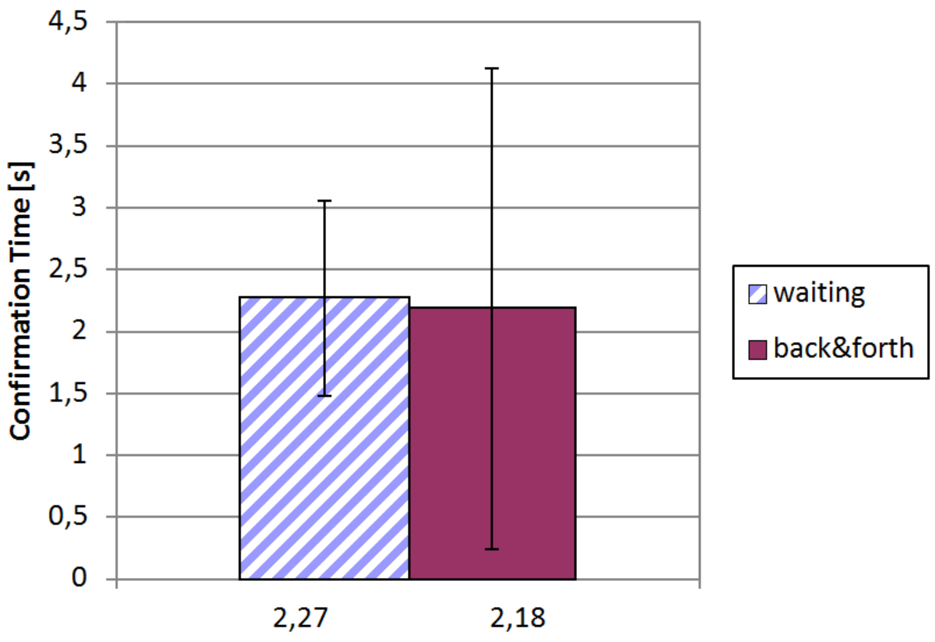


Figure 5.17: Comparison of the mean confirmation time between waiting and back&forth methods with standard deviations

method was the same as the minimum time requirement for the back&forth method, they still can be compared. Instead, the main goal of this empirical analysis was to determine which confirmation method is superior in terms of perceived confirmation speed and general preference.

Selection Time and Error Analysis: A two-tailed t-test for repeated measures on the mean time the users needed to confirm the selected bond was used. The analysis showed no significant difference for the confirmation time for $\alpha = 5\%$ ($t(631) = 0.911, p = 0.363$) (see Fig. 5.17).

As mentioned, it is difficult to compare both methods by time only, because both implementations are time dependent. It depends on the application area, how long the timeout value has to be to get a comfortable system when using the waiting method. The longer the timeout, the longer the entire confirmation process is. The confirmation time of the back&forth method depends on the time frame given to the user to complete the gesture. When the minimum time length of the gesture is increased to reduce false positives (e.g. when shaking), the entire confirmation time will increase as well.

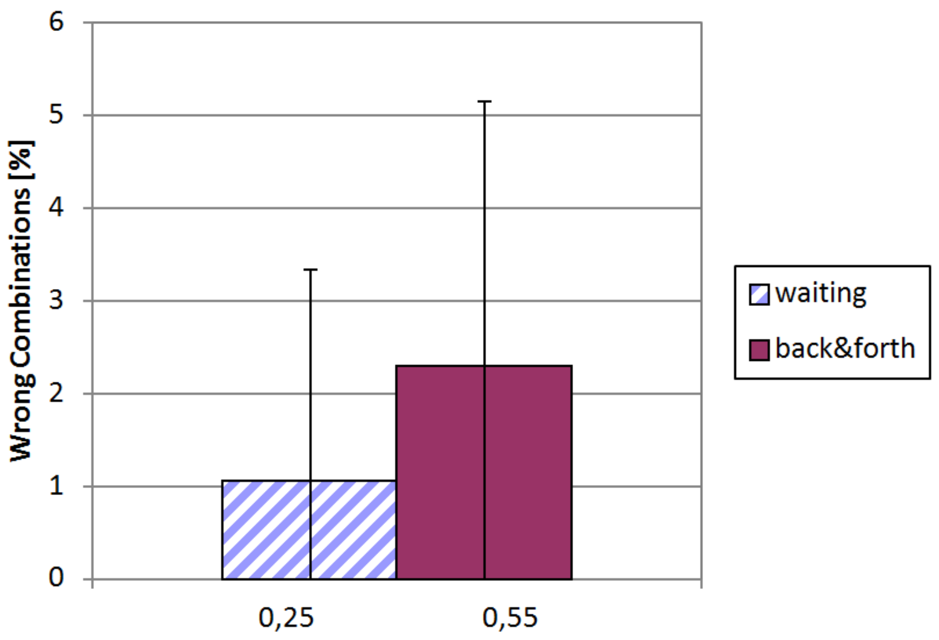


Figure 5.18: Wrong confirmed combinations for each method

Looking at the errors (wrong bond selections) which were made with both implementations (Fig. 5.18), we can see a slightly higher value for the back&forth method. Users made five errors on average with the waiting method and eleven errors on average with the back&forth method.

Subjective Results: Short interviews held after each session revealed that the waiting method was much more convenient for users than the back&forth method. Some participants told that they were lazy and did not want to move when they had the option of not moving. The waiting method seemed to be well integrated, so that the users had no problems with unwanted connections. They also stated that the waiting method is better suited for the purpose of connecting objects, which, as some said, was like holding objects together to let the glue dry. The interviews showed here that the test participants clearly got the concept of the metaphor. It was also suggested that the back&forth method might be better suited for the purpose of disconnecting objects. The back&forth method did not generate a perfect match with the metaphor. The users had the impression to shake something off of the markers instead of connecting the molecules.

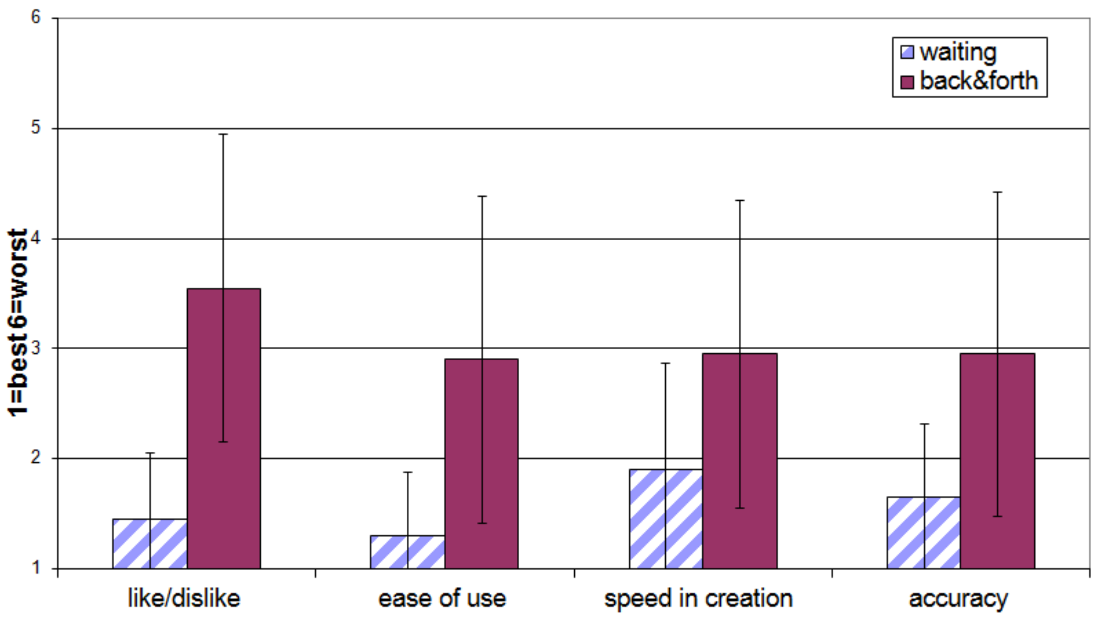


Figure 5.19: Evaluation of the questionnaire regarding the waiting and the back&forth methods with standard deviations

Regarding the questionnaire filled out at the end of each session, the participants were asked to grade each method on a 6-point Likert scale (with 1=best to 6=worst) for like/dislike, easy to use/difficult, fast selection/slow selection, accurate selection/inaccurate selection.

As shown in the Figure 5.19, which presents the results of the questionnaire, the participants had a strong preference for the waiting method. A Mann-Whitney-U-Test was used to compare both methods according to the answers. The participants liked the waiting method ($\mu = 1.5$, $\sigma = 0.6$) significantly more ($\alpha = 5\%$, $p < 0.001$) than the back&forth method ($\mu = 3.6$, $\sigma = 1.4$). They also found that the waiting method ($\mu = 1.3$, $\sigma = 0.6$) was significantly easier ($\alpha = 5\%$, $p < 0.001$) than the back&forth method ($\mu = 2.9$, $\sigma = 1.5$) in use. According to the subjectively perceived time, users thought that they were significantly faster ($\alpha = 5\%$, $p < 0.011$) with the waiting method ($\mu = 1.9$, $\sigma = 1.0$) than with the back&forth method ($\mu = 3.0$, $\sigma = 1.4$). They thought that they were significantly more accurate in selection ($\alpha = 5\%$, $p < 0.001$) with the waiting method ($\mu = 1.7$, $\sigma = 0.7$) than with the back&forth method ($\mu = 3.0$, $\sigma = 1.5$).

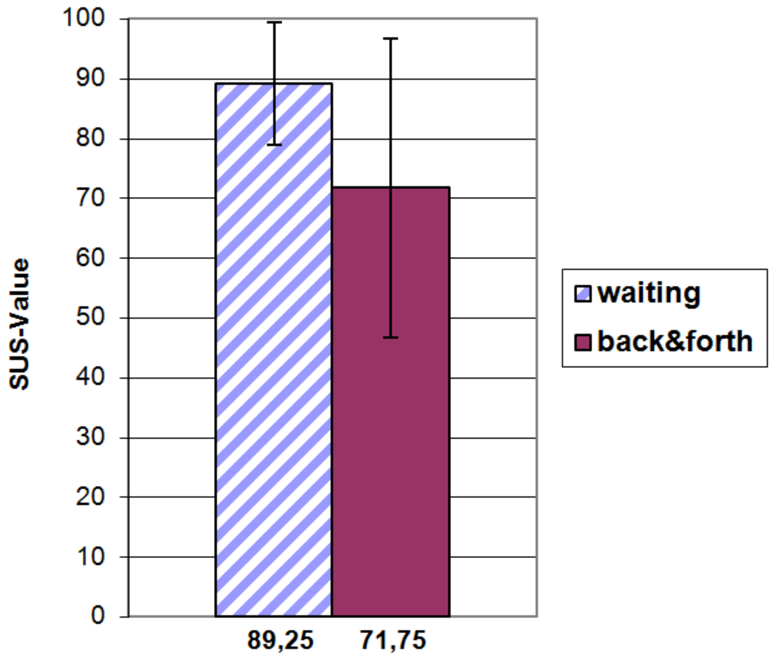


Figure 5.20: Mean SUS value for the waiting and back&forth methods with standard deviations

Looking at the SUS-values (Figure 5.20), a significant difference was found using a Mann-Whitney-U-Test ($\alpha = 5\%$, $p < 0.005$) in both values. The waiting method ($\mu = 89.3$) for confirming the selected bond was preferred over the back&forth gesture ($\mu = 71.8$).

Discussion: Comparing the results of the perceived time that participants needed to confirm the selection with both methods, and the measured time, one can see a large difference. While the measured time — users needed to confirm a bond with both methods — is statistically not significant different (Fig. 5.17), the participants perceived the time with the waiting method faster than with the back&forth method (Fig. 5.19).

This different perception of time can be explained by the different mental workload and the different preference for a method. When a task is easy and comfortable for the user, the time passes faster. When the user on the other hand has to perform a cumbersome or complex task that he dislikes, the perception of time appears longer (as stated by Hancock [34]). The participants could use the waiting method easily, whereas they perceived the back&forth movement as a bit more complex than the waiting gesture. This could explain that the users perceived the back&forth method as the slower method.

Looking at the different error rates of both methods, there can be several reasons for that. The first reason for such errors in both methods was that the participants switched left and right although they stated that they did not have problems in distinguishing left and right. By confusing left and right, wrong atoms were bound whenever two differently colored atoms were to be bound. Due to a probably higher level of uncertainty in performing the back&forth task than the waiting method, the participants generated more errors with the back&forth method. When the participants were asked why they swapped left and right, they stated that they did not spend as much attention to the order of the colors, because they were distracted by moving the markers correctly. Another reason for the higher error level can be that sometimes the gesture was triggered when it was not desired. This was the case in rare situations when the users tried to rearrange the marker cubes in their hand while the two molecules were too close to each other and already had a possible bond shown. This movement caused a shortening and stretch of the shown possible bond and thus was interpreted as a back&forth movement.

When the order of the two methods in the user study was switched after ten participants, participants judged the back&forth method better than when it was the first method (Fig. 5.21). As already mentioned, users rated the back&forth method worse when they had to use that method after having used the waiting method. The reason for that can also be psychological: the users did not accept another solution when they already had a satisfying method.

5.2.5 Automatic Adaptation of Gesture Recognitions

Users might tend to change their motion behavior according to the size of the objects they hold, as well as different users also have different motion behaviors. As the detection of gestures is quite difficult and needs precisely tuned values, nearly perfect recognitions can only be achieved per user and per situation. So there is the need to automatically adjust the parameters for the detection algorithms to adapt to the respective situation.

For the waiting method (see *Parameter Tuning* of section 5.2.2) an automatic adaptation of the threshold value Len is needed while using the system. The algorithm starts with a higher value for the distance of the extreme values. With such large values, the system recognizes easily when the user does not move the markers but this also leads to a high rate of false positives. The value can then be adapted to the motion behavior of the user: When no motion is detected by mistake, the user suddenly moves the hand-held props to prevent the triggering. The system recognizes this sudden move and decreases the distance value according to the last movement data. When the user

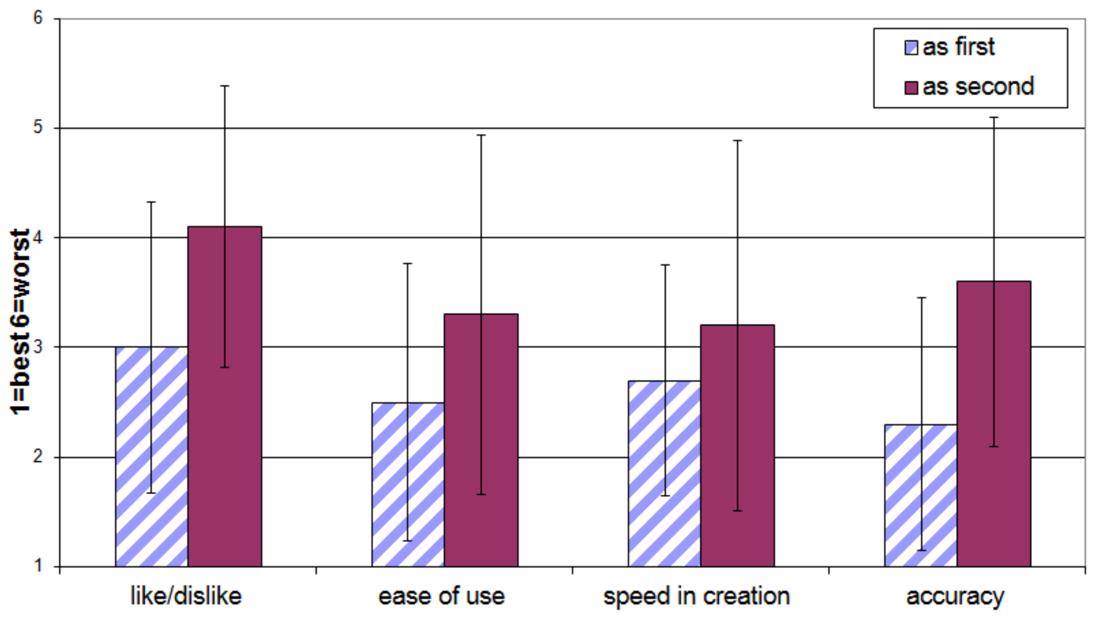


Figure 5.21: Rating of the back&forth method as the first and as the second method in the user study. It has better ratings, when the method was done first.

undoes an unintended confirmation, the system could also decrease the value to prevent this in the future. Whereas, after a successful confirmation, the system could adapt the value according to the recorded movement data, because it knows that the confirmation was intended. However, if the user wants to trigger the confirmation and the distance value is too small, the system unintentionally aborts the confirmation process. When this happens and the user did not made a sudden movement, the system knows that this was not intended and increases the value according to the last movement data. With this described adaption of the parameters, the detection algorithm has the potential to be more robust for different users and different situations.

5.3 Reference-less Selection and Confirmation in a 3D-Environment

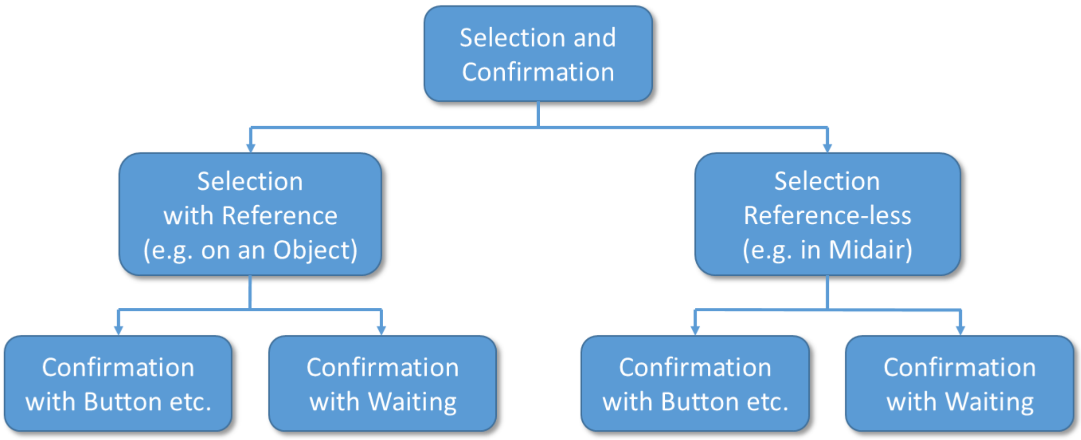

Selection and Confirmation
Selection with Reference (e.g. on an Object)
Selection Reference-less (e.g. in Midair)
Confirmation with Button etc.
Confirmation with Waiting
Confirmation with Button etc.
Confirmation with Waiting

Figure 5.22: Classification of different types of Selection and Confirmation.

Selection and confirmation are crucial parts of user interfaces. In WIMP based user interfaces like the classical desktop environment on PCs for example, parts of the user interface can be selected by pointing at the elements with a mouse and confirmed by pressing a mouse button (leftmost path of figure 5.22). On touch interfaces which are commonly used nowadays on smartphones and tablets, the selection and confirmation normally takes place at the same time when the user taps on the surface. In a 3D user interface of a virtual or augmented reality environment, there also exist several kinds of pointing devices to select objects in a 3D world.

All those scenarios have one thing in common: The selection and the following confirmation of the selection is always applied on objects or elements (*Selection with reference*). For example, a 3D object is selected by pointing at it with a pointing device and confirmed by pressing a button on the device or by applying some kind of gesture based confirmation.

But when there is the need to place an object somewhere in *midair*, confirming the selected position and orientation could be done in several ways (*Selection reference-less*). One way could be to press some kind of button to confirm the selection (third path of figure 5.22). Gestures can also be used to confirm this selection. Waiting gestures are normally used when there is a reference to objects or elements (second path of figure 5.22). For example selecting an object by pointing at it and confirming the selection when the object is selected for a specific amount of time with the pointing device. Those waiting methods – to confirm a selection – always have objects or elements to which they refer.

But what can we do if we want to confirm something that has no reference to an object or element (rightmost path of figure 5.22)? For example if one would like to place an object somewhere in mid-air without using an extra button. A normal timeout cannot be applied here since there is no trigger to start the timeout. There has to be some algorithm to detect when to start the timeout.

I have developed a waiting gesture (see section *Confirmation by Holding Still* 5.2.2) that can be used without having a reference to an object or element, just by observing the movement of the pointer. With this method it is possible to select and confirm something anywhere in the 3D space.

Another advantage of this waiting gesture is the low positioning error. To investigate the usage and accuracy of this waiting method, a user study was performed [57]. Thanks to A. Dey who helped me in the evaluation and took care of the statistical analysis. This study was also done to find out which kind of confirmation method induces the least error in confirming correspondence points for a head-mounted display calibration using SPAAM [84].

5.3.1 Evaluation of Confirmation Methods for Optical See-Through Head-Mounted Display Calibration

A big challenge in Augmented Reality (AR) is to achieve a seamless integration of virtual objects into the real world. In an AR system a camera image or the view of the user is overlaid with the virtual objects. To achieve that the virtual objects appear at the desired position, the virtual camera which renders the virtual objects must have the same internal properties (e.g. field of view) as the real camera or the display-human-eye combination when optical see-through head mounted displays (OSTHMDs) are used. Those OSTHMDs have half transparent displays where the user can see the real world as well as the virtual world at the same time. A so-called display calibration is needed for this.

Irrespective of the display technology used, a display calibration is always required in an AR system. Video see-through head mounted displays (VSTHMDs) augment the video, captured by a camera which is fixed to the head mounted display. The user sees the augmented video on displays in front of his eyes.

For display calibrations, the correct properties of the real camera or display-human-eye combination have to be determined.

To retrieve the needed parameters of VSTHMDs, the computer has to know how a 3D point in the reality appears on the 2D display. The 3D point is seen at the same position as the 2D point on the screen. This is called a 3D-2D correspondence. 3D-2D correspondences have to be collected to estimate a projection matrix that projects 3D points onto the 2D display. Computer vision algorithms are used to find those 3D-2D correspondences for VSTHMDs, requiring a minimum of user interaction.

In contrast, OSTHMDs require a higher level of user interaction for calibration. Here the user has to repeatedly align a 2D and a corresponding 3D point manually to define proper correspondences. Such correspondence data is used in calibration mechanisms for OSTHMDs such as the Single Point Active Alignment Method (SPAAM) [84] and the Display Relative Calibration (DRC) [68]. While most of the parameters required for SPAAM calibration are captured by human users, DRC is based on a two phase calibration method where only in the second phase (using option 4 of the DRC paper [68]) requires user interaction to calibrate the display system for a specific placement on the user's head. SPAAM is therefore more vulnerable to the human error during the calibration process than DRC.

A detailed overview of head-mounted display calibrations is given in the PhD thesis of Zhou [97].

After the alignment of a 2D point on the display with a 3D point in the real world, users usually confirm the correspondence by pressing a key on the keyboard. The assumption is that this confirmation method plays a major role in human performance during the process. The confirmation action requires some degree of hand coordination forcing a misalignment of the 2D point with the 3D point, increasing the inaccuracy of the captured data. It is to be expected that creating an interface that minimizes the required locomotion during confirmation could lead to a more precise acquisition of calibration data, increasing the augmentation accuracy of the system.

In this study, three confirmation methods were evaluated, *Hand-held* (pressing a hand-held button), *Voice* (verbally reporting) and *Waiting* (detecting nearly no movement of the head for at least 0.5 second) with the most often used method *Keyboard* (pressing a key on the keyboard). The study aimed at investigating the influence of the confirmation methods on the quality of the measurements.

To compare the different confirmation methods, the misalignment of the 2D and 3D point by the user had to be measured. Collecting quantitative data was enabled by using a VSTHMD. A computer vision algorithm applied to the video stream calculated the misalignment of the user specified 2D points to the corresponding 3D points in pixels. Procedures as done by Navab et al. [66] where the calibration quality of an OSTHMD is evaluated in a quantitative manner on-line, cannot be applied here: further correspondence point selections which are necessary to evaluate the calibration on-line, additionally induce new user-generated errors. In this case they would have distorted the results.

Using a VSTHMD instead of an OSTHMD appears feasible because the focus lies on the minimization of head motion during the confirmation process; and head motion is mainly dependent on the confirmation method not on the type of HMD. Hence, confirmation methods which reduce the head motion during confirmation will also reduce the error for any type of HMD. In general, the alignment error on both types of HMD mainly consists of an accuracy error generated by the user (even if the user would not tremble or had enough time to do the alignment) and the error induced by the confirmation method. Using an OSTHMD, an additional error is induced by the eye point which is not fixed according to the HMD. By minimizing the error induced by the confirmation method, the error for both types of HMD is reduced. Because of this, a VSTHMD instead of an OSTHMD can be used to analyze the errors induced by the confirmation methods. A subjective expert test at the end of section 5.3.6 also shows the same trends by using an OSTHMD.

The study revealed that the *Waiting* method outperformed all three other confirmation methods including the currently most-used keyboard-based method (see Fig. 5.27).

Furthermore, we investigated the averaging of the data in different time frames to yield further improved results. We also observed that test participants tended to slightly shake their head or directly proceeded to the next point right after an acknowledgment. So there was a decent amount of movement at and after the time of confirmation. Time frames of varying length ($0.1\,\mathrm{sec}$ to $2\,\mathrm{sec}$) were inspected, all ending at the time of confirmation $[t]$ (Used time frames were:$[t-2.0\,\mathrm{sec}, t]...[t-0.1\,\mathrm{sec}, t]$). In addition, the symmetric time frame around the time of confirmation $[t-0.25\,\mathrm{sec}; t+0.25\,\mathrm{sec}]$ was inspected, verifying the previous assumption that the error in this time frame was too high, because of movements caused by the input method and an already ongoing movement to the next correspondence point. To analyze the different time frames, the tracking data during the whole alignment and confirmation process was recorded.

The time frame of $[t-0.6\,\mathrm{sec}, t]$ resulted in the most accurate alignment among the above mentioned time frames. The current approach of calculating the residual error just at the point of acknowledgment $[t]$ in comparison resulted in the worst accuracy. Time windows that end before the time of confirmation in the form of $[t-a, t-b]$ could also be analyzed in the future.

5.3.2 Related Work on Optical See-Through Head Mounted Display Calibration

Since the publication of calibration methods for HMDs, work on improving the calibration mainly investigated numerical aspects, focusing on the computation of perspective and transformational parameters, minimization of errors and on target placement. Human factors have been investigated to a lesser degree, mostly as a side effect of target placement and aiming.

Numerical Aspects: In the first publications of the SPAAM algorithm Tuceryan and Navab [84, 85] noticed that the larger a tracker volume is covered by moving the user's head, the more possible systematic errors in the tracker measurements will be taken into account in the optimization process. They encourage the user to move the head around as much as possible while the calibration process is executed, but not during confirmation. But they relativize this argument due to tracker issues in their setup. The tracker also induces lag in the tracker data at the point of collection. If the button is clicked too quickly, the tracker data read may not correspond to where the user's head is. Newer tracking systems may have a lower lag, but the question remains when and how to capture the data.

Axholt deeply investigated distribution of correspondence points [3] and found that a wider distribution in depth is an influential parameter towards good calibration precision. The addition of more correspondence points can lead towards good calibration precision, but consumes more time and (depending on the calibration exercise) can potentially increase the precision, but also exhausts the user to the point where alignments are not as good as they could be. It therefore appears more useful to consider how to distribute correspondence points in depth.

Further work by Axholt examined this distribution of correspondence points in depth [4]. Three different correspondence point distributions were investigated. Additional simulation results showing improvement factors versus number of correspondence points are presented. Distribution of correspondence points in depth affects the variance of the estimated eye point. Axholt found that almost all parameters of the pinhole camera model depend on the variance of the correspondence point distribution, except for orientation appearing to be more dependent on the number of correspondence points. He also found that a lesser number of correspondence points seem to be necessary when using a random distribution.

Human Aspects: Demer et al. [19] investigated the unwanted head movements of users with telescopic spectacles and without. They observed that the subjects always made a certain head movement with different magnitudes depending on magnification and different qualities of vision. Other works investigate aspects of calibration procedures that are related to human behavior. The user's inability to maintain a stable pose [50] is the most relevant parameter when collecting point correspondences as stated by Tuceryan et al. [84] (p8 last paragraph, p12 section 5.1 last sentence).

McGarrity et al. [59] stated that the user must be factored in when calibrating an HMD. Errors are induced by users because calibration procedures involve manual steps. McGarrity et al. mention factors that might appear to only have small effect, such as facial muscle contractions, e.g., talking, raising the brow, but obstruct calibration quality. In addition they request that simplicity is a must for any calibration algorithm. If users have to do some difficult action, it is likely that they will inject errors into the system. Finally, other factors are listed that may cause errors during calibration and evaluation. For example poor lighting may make target alignment less accurate since the user has difficulties perceiving the target through the darkening display.

Earlier work by Axholt et al. [5] investigated postural stability during the calibration process. They studied alignment performance with head-mounted displays at different levels of azimuth and elevation. While the results show that the viewing direction has a statistically significant effect, the effect can be neglected. In practical settings this effect can be approximated by a circular distribution with standard deviation in sub-angular magnitude.

The already mentioned work by Axholt et al. [4] also discussed user motion during the calibration process. In the experimental design, the subjects were required to move a lot, probably having induced equipment slippage and thus induced a higher error rate during the collection of correspondence points. In our setup, the users sat on a chair while the 3D point was moved forwards and backwards to cover the depth in the calibration procedure.

Livingston et al. [51] investigated issues arising after display calibration. Vertical disparity between the images of both eyes can lead to problems for the user in fusing both images into a coherent picture of the 3D world. Their approach adds a final step to the calibration procedure by requesting the user to align nonius lines. With this correction step, some alignment errors can be corrected, but a preceding display calibration is still required.

Waiting as Input: In 2D user interfaces, waiting as a confirmation method is well established. For example on touch screens in mobile devices, the user touches a spot on the screen and keeps the finger at that place to bring up a menu. In this case the system stores the position of the finger on the screen when the finger touches the display. When the position of the finger does not go further than a predefined radius in a certain time, the system triggers the event. Another well-known system is the tooltip element as already mentioned in section 5.2.1, which is used in programming for mouse based systems where the tooltip shows up when the pointer is over an element for a specific time (no matter how fast the user is moving the pointer on the element).

Some applications implement such waiting approaches with dwell time as discussed by Steed [78]. For instance, Feiner et al. [22] implemented such an approach in a tourist guide system to confirm labels of points of interest. Focusing such a label for at least a second in the center of the head-worn display shows further information about the object of concern. Ha and Woo [33] used a similar dwell time approach. Here the user has to point on an object, having generated a virtual collision for at least $500ms$ to select the object. The work by Axholt et al. [4] lets users aim at a correlation point for at least 2 sec after activation of the corresponding crosshairs. All data was collected over this time period. The calibration procedure aimed at collecting the 2 sec interval of correspondence points with low differences in the HMD rotation. If the rotation of the HMD changed more than $0.19°$ per sample, the data was discarded.

All those waiting methods have a reference to an object or pixels where the user waits in/on a context (button, object etc.) or waits a given time when the system is in a specific state. The introduced waiting method is reference-less. The algorithm has to determine everywhere and to every time when the user stops moving. A detailed description of this method can be found in the next subsection at the waiting method.

5.3.3 Experimental Evaluation

The primary design goal of this experiment aimed at evaluating the different acknowledgment methods for the OSTHMD calibration. The experiment investigated how different confirmation methods influence the accuracy of the input data (correspondence points) for the calibration process.

Acknowledgement Methods: The user has to align a 3D point of the real world at an appropriate depth with the crosshairs displayed on the VSTHMD. To acknowledge the correct overlay of the correspondence points, the user has to use one of the four methods described below. The crosshairs are presented in a predefined order in various positions on the VSTHMD where the center of the crosshairs is the 2D display point. Once the user thinks that the 3D target point and the 2D display point are successfully aligned, some acknowledgment method has to be used to acknowledge the correspondence points. The currently available acknowledgment method is pressing a key on the keyboard [84]. This method forces the user to move the hand, and in consequence a part of the body, which may result in a misalignment while acknowledging. Three additional acknowledgment methods have been implemented to minimize the complexity of the process and to achieve higher accuracy.

Hand-held: The user is equipped with a hand-held button (Fig. 5.23). The user acknowledges the correspondence points by pressing the button. This process requires only a minimal finger movement decreasing the amount of locomotion. The user can hold the device in a comfortable way which could prevent unwanted movements due to strain.

Voice: Users verbally confirmed by uttering „ok" or „k" when they completed the alignment successfully. This study used a wizard-of-oz technique to compensate for possible failures of voice detection systems. This process required no additional device and only required a minimal movement of facial muscles.

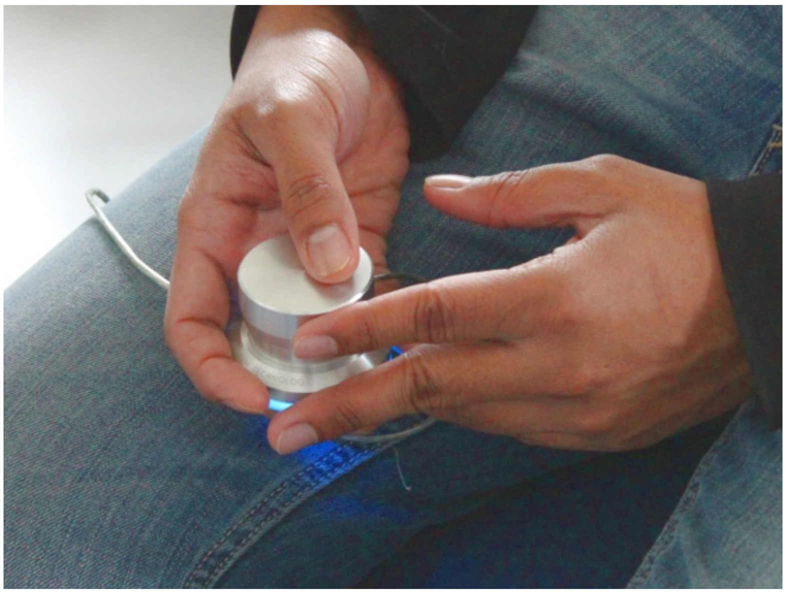


Figure 5.23: A hand-held button was used to confirm a selection by pressing on it.

Waiting: The last implemented method requires users to keep their head steady at the point and direction of alignment and wait for at least 0.5 seconds. Therefore, this interaction technique is called *Waiting* method [55]. The steadiness of the head was measured by inspecting the normalized vector from the HMD to the target sphere in the HMD coordinate system over the alignment process. To analyze the amount of head movement, the implementation calculates two different values w.r.t. the data of the last $0.5sec$ (as described in more detail in subsection 5.1.1 figure 5.2). The first value is the trajectory length of the normalized vector tip. The second value is the compactness of this trajectory. The compactness is the mean distance of the trajectory points to their centroid. It is considered that the head is steady when first, the trajectory length stays below a threshold value of $1.5cm$ and second, when the compactness value is below $0.15cm$ for the last $0.5sec$. This method requires no additional device to carry and enables users to acknowledge without body movement. In fact, the method forces users to reduce body movement and as a result could lead to a lower aiming error.

Hypotheses: The following hypotheses were postulated before executing the experiment.
[H1] *Waiting* and *Hand-held* methods will result in the most accurate data. *Voice* and *Keyboard* methods will produce worse accuracy among the methods.
[H2] Averaging over time frames will result in better calibration than taking only the correspondence points at the time of confirmation.

Experimental Setup: To measure differences in the confirmation methods, a scenario was set up where the users should collect 2D-3D point correspondences. An infrared reflecting rigid body was used as aiming target for the 3D point where the left most silver sphere was clearly indicated (see Figure 5.24b). For the 2D point, nine crosshairs with a surrounding circle were displayed to the user in the HMD one after the other.

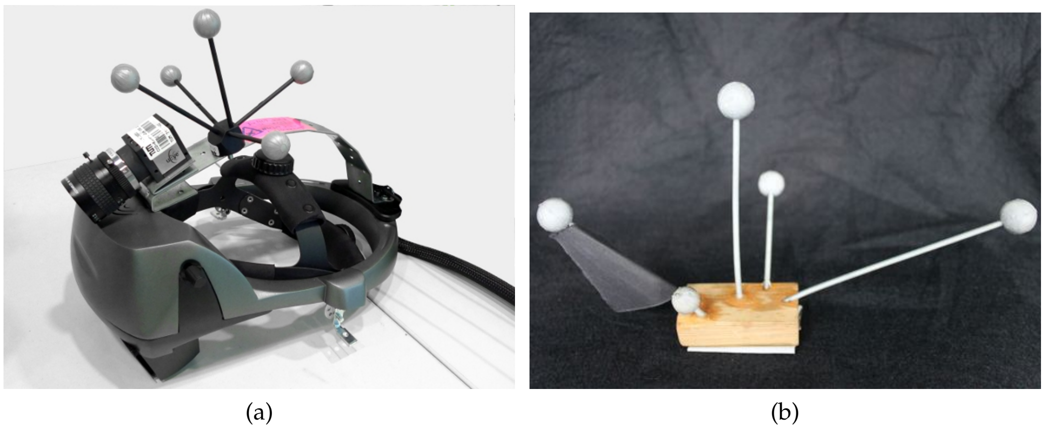


Figure 5.24: A VSTHMD was used to calculate the alignment error (a); participants had to align the crosshairs appearing on the HMD with the center of the left most silver sphere in (b).

To understand human behavior in more detail as well as being able to calculate aiming errors w.r.t. reference data, an OSTHMD (nVisor ST60) was converted to a VSTHMD. As the aiming error of the users had to be measured, we somehow had to see what the users sees. This is not possible with OSTHMD, as we cannot get the image the human eye sees. To overcome this problem, a VSTHMD was used here. This replacement can be done, because as we look at the induced head movement during confirmation, a VSTHMD moves the same amount as an OSTHMD. When the induced movement is reduced for a VSTHMD, the movement will also be reduced for an OSTHMD.

With the image data from the camera, an image recognition algorithm can calculate the ground truth data. This represents an optimal 2D-3D correspondence collection.

For the conversion to a VSTHMD, a camera was mounted on top of the display (see Figure 5.24a). The image of the camera was shown in the display of the dominant eye. The video feed from the camera was also used later to compute the reference data. Users sat comfortably on a chair, looking in the direction of the 3D target. The 3D target was placed at two different positions, either in $1m$ (near) or $2m$ (far) in front of the user at a height of $0.8m$. Users thus could hold their head in a comfortable position when looking at the marker target straight ahead. An infrared tracking system by A.R.T. GmbH [1] was used for tracking of the HMD and the target. The head movements of the participants were tracked using a rigid body fixed to the HMD. To easier analyze the aiming error, a black background was used to detect marker balls in recorded video feeds via image processing methods in a robust way. The center positions of the marker spheres are estimated using a circle detection algorithm. Estimating the centers of these marker balls in an automatic way provided reliable information about the 2D display positions at which the users should have aimed. This reference data was used to calculate the residual error in comparison to the user acknowledged 2D position.

Experimental Task: The experimental task in this experiment is similar to the tasks performed to calibrate an OSTHMD and previously described by [84] and [68]. Participants had to align the crosshairs with the silver target sphere by moving the head and body to get to an alignment. Once this alignment was established, they had to use one of the four methods to acknowledge the alignment. Crosshairs were presented one after the other in nine different positions on the display. Users had to repeat this set of eighteen alignments in near and far distance ten times generating 180 correspondence point pairs. Participants were allowed to take a rest after each set. After task completion, participants had to fill out a subjective questionnaire. One set of 18 correspondence points took about 2-3 minutes and after 5 sets of about 2-5 minutes, users normally wanted a break. The experiment took an average of 45 minutes per user.

There were 24 participants from the student and research population of the university, aged between 23 to 53 years (M=28.9, SD=6.05). Nearly none of the participants had prior experience with the OSTHMD calibration process; some of them had experienced AR applications before.

Dependent Variables: The residual error in the calibration process was calculated as a dependent variable. The entire experiment was based on 4 (methods) $\times$ 21 (time frames) $\times$ 10 (repetitions) $\times$ 6 (participants per method) $\times$ 18 (correspondence points per repetition) = 90720 data points.

Independent Variables:

- **Acknowledgment Method** $\in$ {Keyboard, Hand-held, Voice, Waiting} *between subjects*
  In this experiment, the 24 participants were randomly distributed into 4 different groups having 6 participants in each group. The participants of each group did the task using only one acknowledgment method described in section 5.3.3.
- **Correspondence Points:** $\in$ {1 to 18} *within subjects*
  The experimental setup had two different distance layers – near and far – having 9 correspondence points in each layer. The 9 correspondence points were distributed in a 3×3 squared orientation on each of the two layers.
- **Repetition** $\in$ {1 to 10} *within subjects*
  Alignments of the 18 correspondence points were performed in one repetition. The participants performed the experimental task 10 times.
- **Time frame:** $\in \{[t - 2.0sec, t], ..., [t - 0.1sec, t], [t]\}$ *within subjects*
  The vision of the users through the HMD was recorded at 30fps. In post-process, the mean residual error was calculated in 21 different time frames relative to the time of acknowledgment $[t]$.

5.3.4 Results

To statistically analyze the collected data, the residual errors for every time step were calculated. The residual error is the Euclidean distance between the red crosshairs on the display and the projected center of the silver target sphere in the display image. To calculate the center of the silver target sphere, a computer vision algorithm was used on the recorded screen data. The more users misaligned the crosshairs with the sphere, the more they were away from the image of the target sphere on the display and the greater the residual error was. As measurement for the aiming error, not only residual errors were examined at the time of confirmation t but also the average residual error in the remaining 20 time frames (Figure 5.26). The effect of the various methods and time frames were analyzed by a $4 \times (21 \times 10)$ mixed-factorial ANOVA using the statistical package SPSS.

Effect of Acknowledgment Methods: ANOVA reported a significant main effect of the acknowledgment methods $F(3, 428) = 22.07; p < .001; {\eta_p}^2 = .13$ (regarding all time frames). A Tukey's HSD post-hoc test revealed that among all methods, the *Waiting* method outperformed all other methods and the *Keyboard* method was worst. The *Waiting* method was significantly ($p < .01$) more accurate and the *Keyboard* method was significantly ($p < .001$) less accurate than all other methods. This partly supports the hypothesis *[H1]* for the *Waiting* and *Keyboard* method.

Effect of Time Frames: ANOVA found a significant main effect of the time frames $F(1.087, 465.047) = 127.14; p < .001; {\eta_p}^2 = .23$ on the residual error. As the data did not meet the assumption of sphericity, Greenhouse-Geisser adjustments [7] were used. After performing a pair-wise comparison with Bonferroni adjustments, it could be seen that the time frame $[t-0.6sec, t]$ was overall the best and a significant better time frame than all other time frames except the $[t-0.5sec, t]$ and $[t-0.4sec, t]$ time frames. The error calculated at the point of acknowledgment t is significantly ($p < .001$) worse than the time frames ranging from $[t-0.9sec, t]$ to $[t-0.1sec, t]$ which supports the hypothesis *[H2]*. This can be seen in figure 5.26 where the error of all methods at the time of confirmation (most right data) is higher than the errors of the time frames $[t-0.9sec, t]$ to $[t-0.1sec, t]$.

There was a significant $TimeFrame \times Method$ interaction effect $F(3.260, 465.047) = 26.42; p < .001; {\eta_p}^2 = .16$. While the difference between the mean residual error was very small for the *Waiting* method across all time frames, for other methods the differences were much bigger. However, for the individual methods the best time frames were different. $[t-0.4sec, t]$ and $[t-0.6sec, t]$ were the best time frames for *Keyboard* and *Hand-held* respectively. For *Voice* the best time frame was $[t-0.8sec, t]$; and $[t-1.7sec, t]$ for *Waiting*.

Interestingly, analyzing the differences between the best time frames of each method with a one-way ANOVA $F(3, 4316) = 7.52; p < .001$ and Tukey's post-hoc test revealed that the *Waiting* method was significantly better than all other methods *including* the *Keyboard* method (Fig. 5.27). The *Voice* method was the worst method among these four methods. However, these differences were not significant with the *Keyboard* and the *Hand-held* methods.

The effect of time frames on each individual participant was investigated, as the OSTHMD calibration is indeed dependent on individual skills. Expectedly, the error at $[t]$ was high for any participant in the experiment (which supports *[H2]*). $[t-0.6sec, t]$ was the best time frame for 7 participants (1 - *Keyboard*, *Voice*, and *Waiting*; 4 - *Hand-held*) and $[t-0.4sec, t]$ was the best time frame for 3 (*Keyboard*) participants. Similarly, $[t-0.7sec, t]$ was the best time frame for 3 participants (1 each for *Keyboard*, *Hand-held*, and *Voice*). $[t-0.3sec, t]$ was the best time frame for 2 participants (1 each for *Keyboard* and *Waiting*). Interestingly, participants with the *Waiting* method had longer best time frames such as $[t-1.1sec, t]$, $[t-1.7sec, t]$, $[t-1.8sec, t]$, and $[t-2.0sec, t]$. A similar trend is found for the *Voice* method as well with $[t-0.9sec, t]$, $[t-1.0sec, t]$ (2 participants), and $[t-1.5sec, t]$ being the best time frame.

These results are consistent with the hypotheses. The *Waiting* method produces higher accuracy than keyboard based methods (*[H1]*). OSTHMD calibration data must be collected in a longer time frame to gain accuracy. Collecting data just at the time of confirmation is an inaccurate practice (*[H2]*).

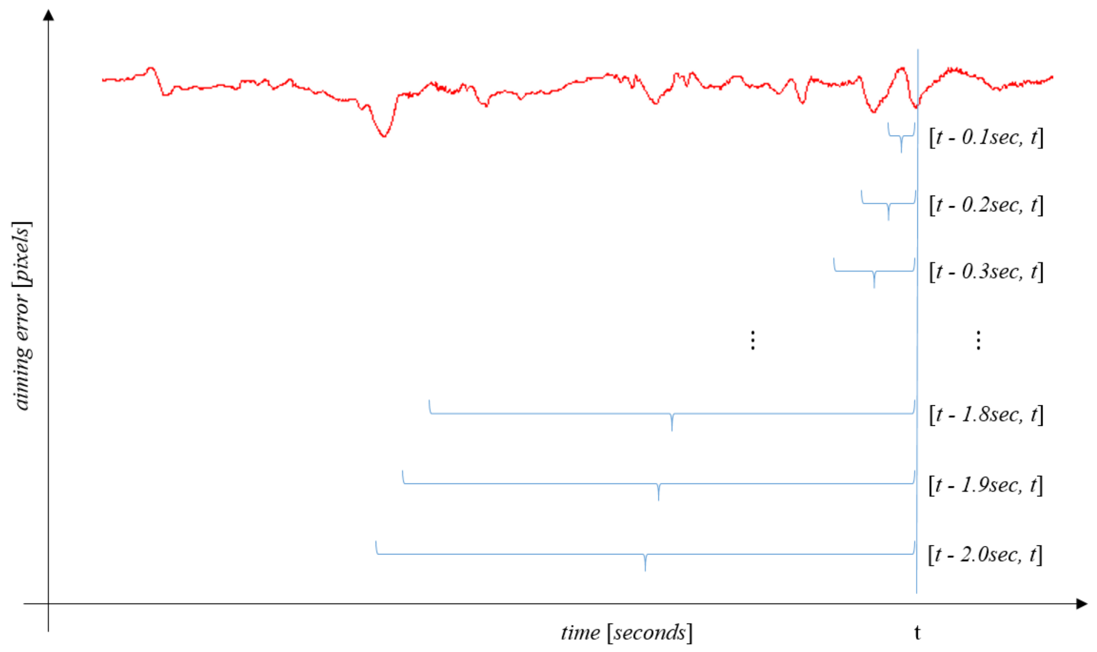


Figure 5.25: Illustration of the time frame calculation. Each time frame ends at the time of confirmation t the length of the time frame range from $2.0\,\text{sec}$ to $0.1\,\text{sec}$. The mean of the aiming error from each time frame is one data point in the figure 5.26.

5.3.5 Discussion

Averaging the collected data of time frames results in better calibrations. This behavior can be explained by the movement of the user's head while aiming at the target. While aiming, the user oscillates over the target spot with the inability to precisely target this spot. Averaging over those points results in a point that lies closer to the target spot. The longer the user concentrates on the target spot, the better the averaged result should be. But the calibration process should also not be too long; otherwise users get tired and will make mistakes. In the study the users did the confirmations in their desired speed in order not to be in a hurry. This gave a good precondition to find the optimal time frames for the different methods.

The good results of the *Waiting* method can be explained in the following way. The users had to concentrate on the target spot trying to keep as still as possible. This reduces the ability to do sloppy confirmations and also calms users down. Whereas with the other methods, users kind of rushed over the correspondence points leading to a higher error than the *Waiting* method. Those other methods also required the user

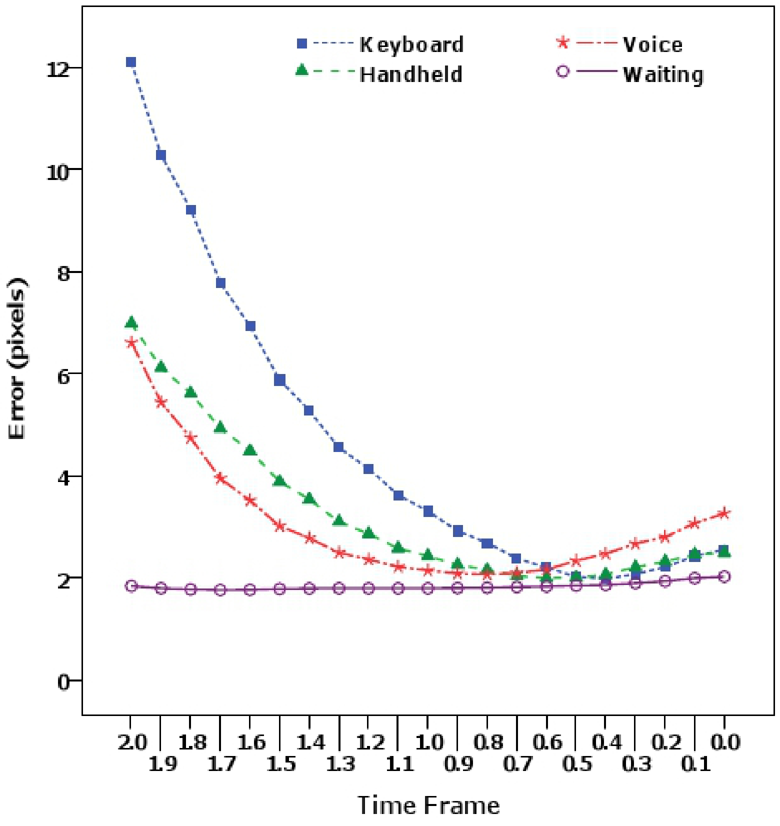


Figure 5.26: The x-axis represents the different time frames $[t-x, t]$. The x values are the lengths of the time frames in seconds. $[t - 0.6sec, t]$ was the most accurate time frame. The error was consistently low for the *Waiting* method across the time frames.

to perform an extra activity which also led to an extra error.

Participants reported their experience with the calibration methods through a subjective questionnaire. As expected, most of the participants reported the heavy weight of the HMD gave them trouble in performing the alignment task. Interestingly, two participants reported for the *Waiting* method that they found aligning the points near to the lower border of the display to be harder than the others, as the front heavy HMD was „falling down". We asked participants to judge the movement of their head *at the point of* acknowledgment according to their perception. All of the participants of the *Waiting* method used terms like „very little", „minimal", „1mm." etc.; which indicates their high confidence with the task. Whereas in the case of other methods (particularly in *Voice*) participants used terms like „a bit", „1-2 cm.", „some pixels". Overall, it fur-

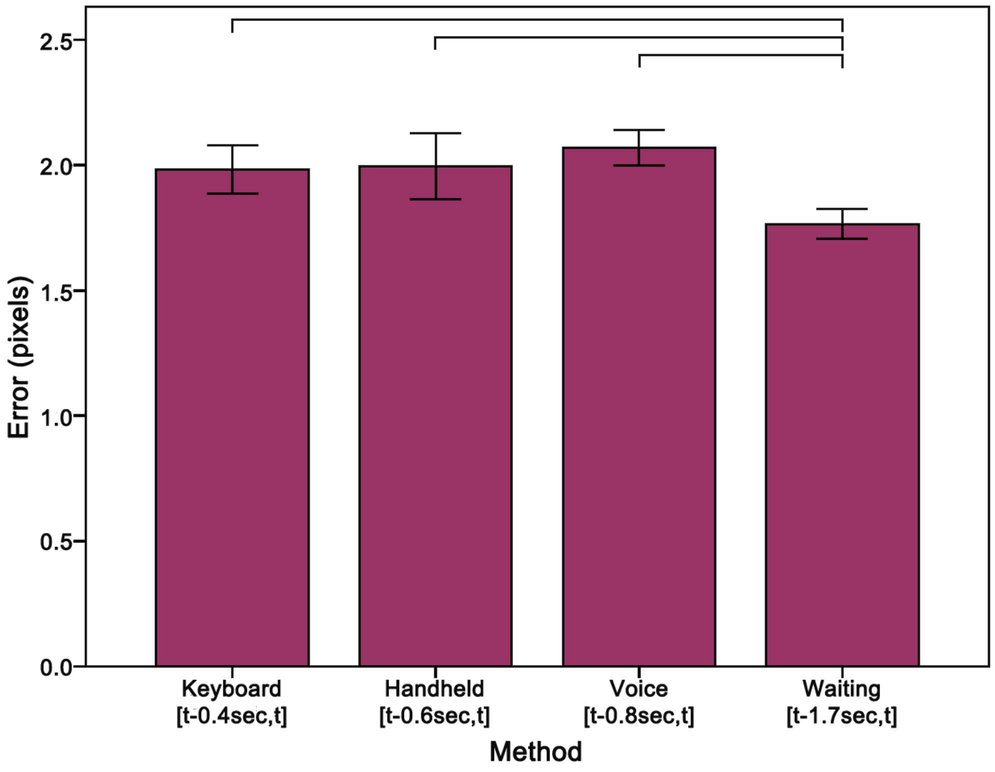


Figure 5.27: The best time frame of the *Waiting* method is significantly better than the best results of all other methods. Whiskers represent $\pm 95\%$ confidence intervals.

ther indicates the acceptance of the *Waiting* method over other methods. Subjectively, participants did not like the *Voice* method. All of the participants reported a neck fatigue, four participants reported eye fatigue and one participant of the *Voice* method reported the task too stressful.

5.3.6 Validation of Results

In the previous study and in its analysis, the dependent variable for judging the different methods was the residual error of the 2D display points and the correspondence point of the 3D target on the display. These findings so far leave the question whether those results can also be transferred to the quality of the resulting projection matrices and thus to the overall augmentation.

Numerical Validation: To validate the gathered results, a numerical analysis of the user generated calibrations with the ground truth calibration, generated from the computer vision data of the previous experiment was used. For this numerical analysis, a SPAAM calibration that uses the recorded target 3D points and the calculated corresponding 2D image point from the computer vision algorithm [67] created the optimal projection matrices. The projection matrices of the user data were computed using SPAAM with the crosshairs' 2D positions and the corresponding target 3D point. Averaging the positions using the 21 time frames generated the corresponding 3D positions. This way 210 different projection matrices were generated for each user and the optimal projection matrix.

The dimensions of the grid with the 3D points were $2 \times 2 \times 2$ meters with a point every 10 centimeters in each dimension, producing an total number of 9261 discrete values. These points were placed at a distance starting at 2 meters to the HMD such that the farthest point was 4 meters away.

Those 3D points were projected onto the image plane using the 210 user generated projection matrices. For each projection matrix the residual errors – compared to the points which were projected using the ground truth projection matrix – were calculated. As a value for comparison, the mean residual error was calculated for all visible points on the image plane.

A main effect of confirmation methods on the error $F(3, 236) = 14.05; p < .001$ was detected. A post-hoc test showed the *Waiting* method was significantly better than the *Keyboard* ($p < .001$) and *Voice* ($p = .025$) methods (see Figure 5.28). Consistent with the previous findings, the *Keyboard* method was the significantly worst method and the *Waiting* method was the best method among the four experimental methods. There was also a main effect of time frame $F(1.36, 320.39) = 54.13; p < .001; {\eta_p}^2 = .19$. Overall, $[t - 0.6sec, t]$ was the best time frame.

Validation using an Optical See-Trough Head Mounted Display: The same results were observed when comparing the four confirmation methods on an OSTHMD. A similar hardware setup as in the experiment from before was used, except for using an unmodified OSTHMD this time. Five experienced users did the same SPAAM calibration procedure using the four different acknowledgment methods as described earlier. The users also confirmed 18 2D-3D correspondence points. The previously calculated best time frames (as shown in Figure 5.27) were used for each confirmation method. For each expert user the four projection matrices were calculated. With each projection matrix a chessboard like board was augmented with its complement pattern (see figure 5.29). The pattern was colored in a different color for each projection matrix. Each of the four augmentations was shown in a loop one after the other to investigate user's subjective preference. Each user had to judge the overlay accuracy and had to rank

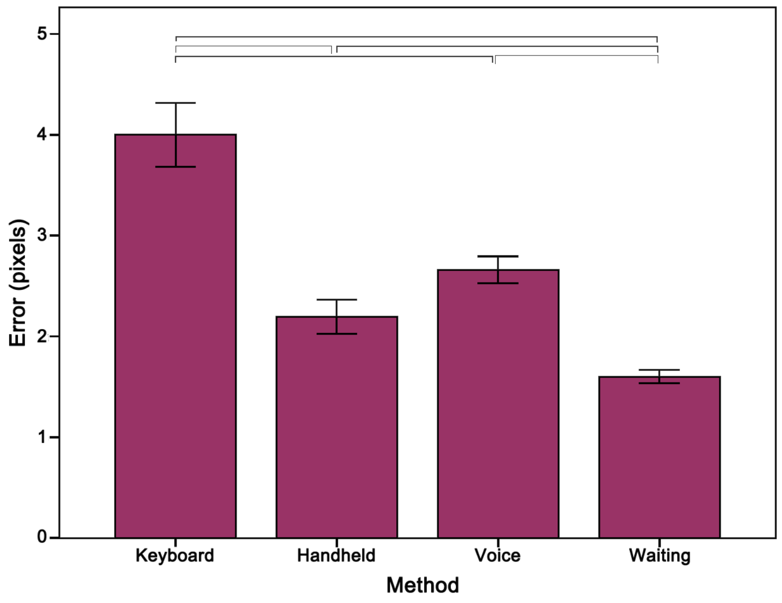


Figure 5.28: A comparison with the ground truth projection matrices confirms that *Waiting* is the best confirmation method and supports the validity of the approach. Whiskers represent ±95% confidence interval.

them according to the estimated quality. The user feedback showed that *Waiting* was the best method for everyone. *Voice* was the worst for three users and two other users reported *Keyboard* and *Hand-held* to be the worst methods. Though, five users do not allow for a deeper statistical analysis, *Waiting* is apparently the best method as found in the previous analysis of Section 5.3.4.

5.3.7 Conclusions

In contrast to the numerical improvements in OSTHMD calibration in most related work, the focus in this work lay on the human factor in collecting more accurate input data for the calibration process. Different confirmation methods for Optical See-Through Head-Mounted Display Calibration were developed and investigated. It was found that different confirmation methods do have an influence on the quality of the calibration. The *Waiting* method as input for the display calibration resulted in the most accurate correspondence point data. It was further found that using averaged correspondence point data of a time frame is always better than only using the correspondence point at the time of confirmation. Each individual method has its own optimal time frame being different from those for other methods. The findings were

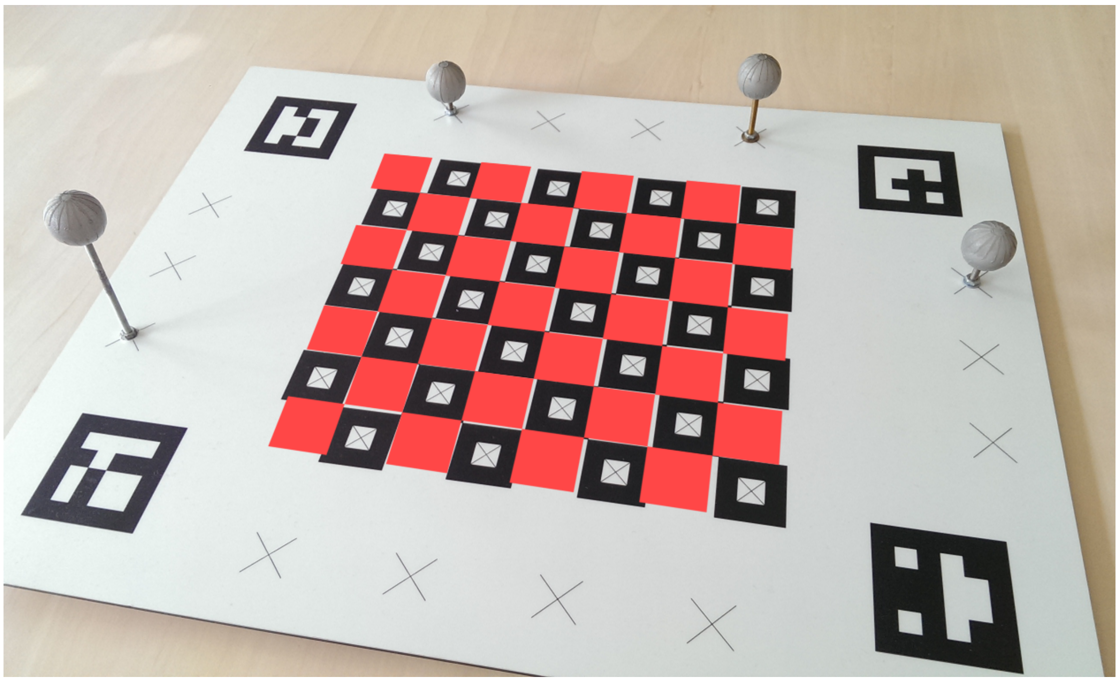


Figure 5.29: Example of a chessboard augmentation. The inverse red pattern is augmented on the chessboard using the user generated calibration. The alignment accuracy had to be judged by the user.

verified with a numerical analysis of different calibrations. Tries with an OSTHMD also showed the same results.

Implementing the presented methods for a calibration process for OSTHMDs will improve the quality of object alignment in AR systems. However, further optimization of user dependent factors might be possible. Here the analysis of the optimal time frames used only time frames which end at the time of confirmation and start at different, earlier points in time. To find the best settings for the time frames, sliding time frames could be an alternative for further examination. With sliding time frames, the length of the time frame varies by letting the end point float in time in the interval between the starting point and the time of acknowledgment. By moving the end point to an earlier point in time than the time of acknowledgment, the possibility increases to further suppress effects in locomotion of the user. This is especially the case with the *Voice* method, where an additional delay has to be subtracted because of the processing time for speech recognition.

In this study, there was the need to select and confirm correspondence points in a reference-less 3D environment. This was done with the help of some standard techniques as with buttons or the keyboard, but also with the waiting method. The waiting method is novel in this context as it has to deeply analyze the movements and behaviors of the user to detect the intention of the user to confirm the selection. The waiting method is a powerful gesture for situations where the standard confirmation methods are either cumbersome or even not possible to use.

6 Evaluation of the 3D User Interface of Augmented Chemical Reactions

To support students in learning chemistry, teachers have to help them understand the spatial structure of molecules. Knowing the 3D structure of molecules is important to understand the chemical behavior and properties of the molecules. Learning the 3D structure of molecules just by looking at the 2D drawings or formulas of textbooks is not the best method.

Hardware representations that the students can touch have been a well-established method in chemistry education for a long time. Yet, such hardware models are not always available, and it is time consuming to build them for each student and for each molecule. Furthermore, such hardware representations are mostly not flexible or dynamic in their structure. Normally they are rigid and the students can only move and turn them. They cannot tear or squish them in order to obtain an understanding of the rigidity and flexibility of individual bonds.

Interactive 3D representations of the molecules are able to provide a better way for students to inspect and understand the 3D structures. Applications can show complex molecules and animations that the students can inspect. But there is a drawback in the classical 3D presentation programs: The molecules can only be rotated and moved by using the mouse and the keyboard. This indirect mapping of mouse movements to the 3D model is not always intuitive. Students have to learn the mapping of 2D mouse movements and keystrokes to 3D object manipulations involving six degrees of freedom. As a result, some of the students' focus may be diverted from the molecules to the user interface, and the structure of the molecules may not be made completely clear.

To combine the benefits of the physical molecule representations with the power of computers, I used the direct manipulation 3D user interface of my application *Augmented Chemical Reactions* (ACR) using tracked real objects to control the position and orientation of the augmented virtual molecules (Section 3).

In this section, I report on my work [53] on evaluating this Augmented Reality based 3D user interface of the application *Augmented Chemical Reactions* against the mouse and keyboard based user interface of the application *Jmol* [63]. The evaluation took place at a secondary school with a class of 14-15 year old students of the 8th grade. In the next subsection, I will take a look at previous work in this field and describe both types of 3D user interfaces. Parts of this chapter have already been published in paper [53] presented at CSEDU 2013.

6.1 Background

Many schemes to support the learning progress of students have been developed. To support the students in understanding the spatial structure of molecules, a number of methodologies have been developed. A number of research efforts have shown that using physical models and therefore using direct manipulation helps students explore and understand the spatial structure of objects [2, 35, 38]. It has also been shown that a direct manipulation interface for rotation via a sensor with 3 degrees of freedom (3 DoF) yields better performance without lacking accuracy, compared to 3D rotation via a mouse [37]. Work at the IBM Almaden Research Center investigated the user performance of different 3D input devices [96].

Desktop-based User Interfaces with Mouse and Keyboard: There are many applications that help users understand the spatial structure and also the resulting dynamics of molecules [44, 63, 69]. Yet, the commonly used user interface to rotate and move virtual objects is still a combination of mouse and keyboard [16].

3D Augmented Reality-based User Interfaces: To combine the advantages of the physical direct manipulation with showing complex structures, *Augmented Chemistry* [26] and *Augmented Chemical Reactions* [54, 56] have been introduced. Both systems use Augmented Reality to deliver a direct manipulation 3D user interface to control the position and orientation of virtual molecules.

As described in more detail in section 3, *Augmented Chemical Reactions* employs a physical cube with a handle that is textured on all sides with black and white patterns (Figure 6.1). In a typical setup, a student holds the cube by the handle and manipulates it while sitting at a desk in front of a monitor. A webcam records the scene with the cube, and *Augmented Chemical Reactions* uses the marker tracking algorithm of UbiTrack [39] to detect and recognize the currently visible patterns on the cube. According to the size and deformation of the patterns, the algorithm calculates their positions and orientations relative to the webcam – and thus the pose of the cube and handle. With this information, the virtual molecule can be drawn on top of the webcam image, leading to the illusion, that the molecule is attached to the cube (Figure 6.2). The virtual molecule moves in unison with the physical cube. This is a three-dimensional direct manipulation user interface.

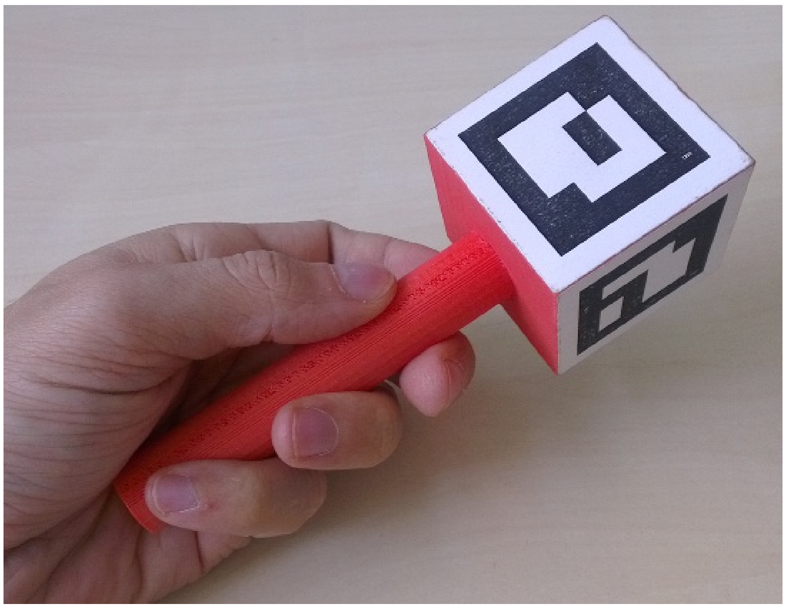


Figure 6.1: This device is tracked by a computer with a webcam. It is used as a 3D input device for direct manipulation.

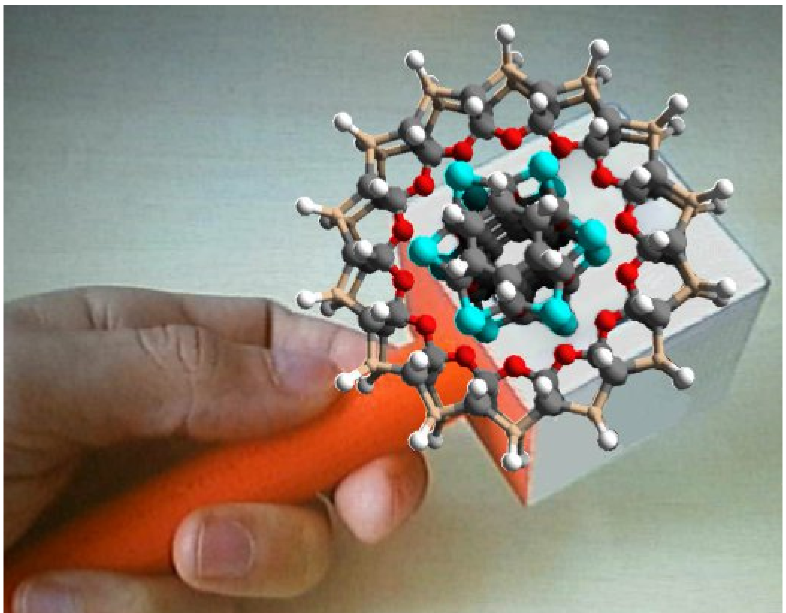


Figure 6.2: Augmentation of a molecule on top of the marker cube.

In comparison, Fjeld et al. [25] do not directly manipulate the molecule. Their molecule is displayed on a marker placed on the table and rotated by a second, tracked input device. There is no direct manipulation of the position and orientation of the molecule with only one device.

6.2 Evaluation

A user study was conducted with 14-15 year old students of a German gymnasium (secondary school) to determine whether direct manipulation of the position and orientation of virtual molecules leads to a better spatial understanding of virtual molecules than input via a standard mouse and keyboard. A class in the 8th grade was selected – just at the time when they had been taught the basic concept of what a molecule is, but they had not learned yet about the spatial structure of molecules. Therefore they were ideally suited for a study investigating which of the two user interfaces leads to a better 3D understanding of molecules.

None of them already had experience with *Augmented Chemical Reactions* or the *Jmol* application. Most of the students had lots of experience with playing 3D games, but nearly none of them had already used a 3D chemical modeling application or another 3D design application. Only one student stated to have already good knowledge of the chemical structures of molecules.

The students were split into two groups, one working with the *Jmol* application and the other one working with the *Augmented Chemical Reactions (ACR)* application, in an between-subject evaluation arrangement.

6.2.1 Experimental Setup

In cooperation with a chemistry teacher, ten different molecules were selected to be inspected in this study. Those molecules were (1) Sulfur S_8, (2) Methane CH_4, (3) Ethanol C_2H_5OH, (4) Acetic acid $C_2H_4O_2$, (5) Benzene C_6H_6, (6) Hydrogen sulfide H_2S, (7) Phosphor P_4, (8) Phosphorus trifluoride PF_3, (9) Hexane C_6H_{14} and (10) Carbon tetrabromide CBr_4.

We had two computer rooms, one for the *Jmol* application and one for the *ACR* application. Each student desk was equipped with the respective computer equipment.

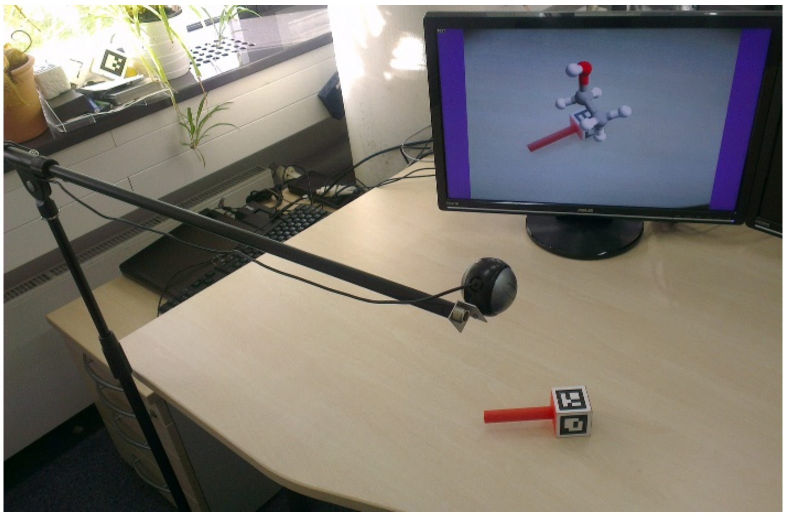


Figure 6.3: Computer setup for the ACR application, using a webcam, a physical cube on a handle and a monitor. The keyboard that is required to cycle through a set of molecules is not shown.

Computer setup for *ACR*: The first computer room was set up to run the *ACR* application with a 3D direct manipulation user interface. Here a monitor, a keyboard, and a marker cube with a handle were placed on each student desk. A webcam on a microphone stand in eye level of a sitting person faced towards the student desk and the marker handle. To control the position and the orientation of the virtual molecule, the students had to hold the marker cube into the field of view of the webcam. The video stream, augmented with the currently selected molecule, was shown on the monitor in front of the student, as shown in Figure 6.3. The students could cycle through the set of molecules by pressing the space-bar on the keyboard.

With a well-aligned arrangement of the camera, the user, the hand-held handle and the monitor, the AR illusion via a directly manipulated physical object can be maintained with minimal strain on the hand-eye coordination. A more immersive, perfectly aligned setup can be achieved when the monitor-based arrangement is replaced with a video-see-through or optical-see-through head-mounted display. Yet, such arrangements are costly and thus currently not deployable in classrooms. For this reason, the current test setup was based on webcams and monitors on student desks.

Computer setup for *Jmol*: The second room was set up to run the *Jmol* application [63], using a classical mouse and keyboard interface to manipulate virtual 3D molecular structures on the screen. To this end, a monitor, a mouse and a keyboard were placed on each student desk. When started, *Jmol* showed the first of the ten molecules,

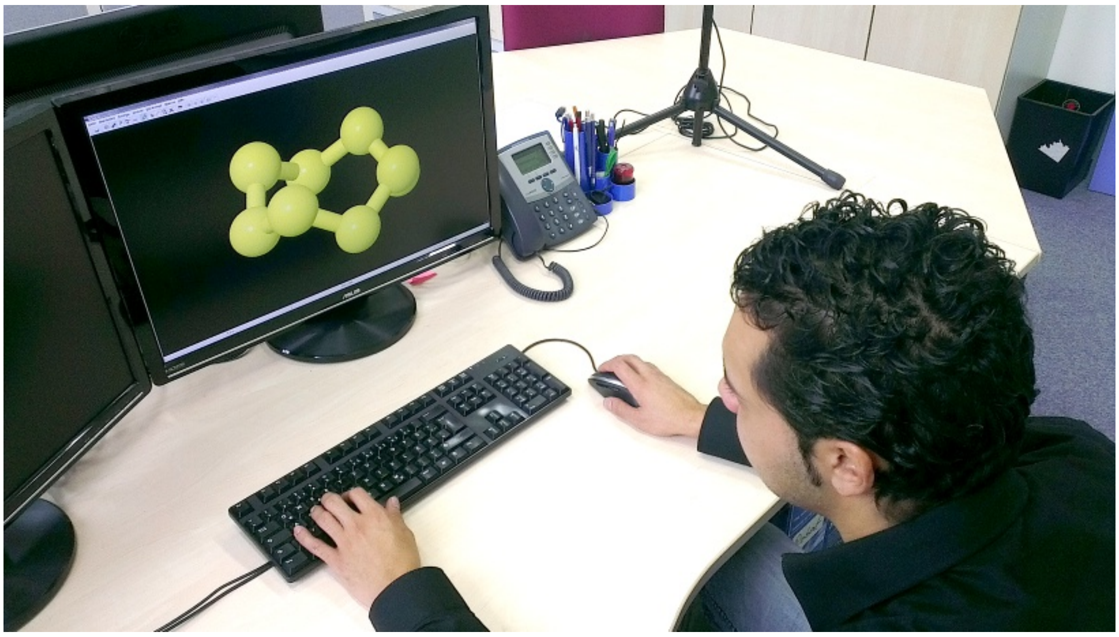
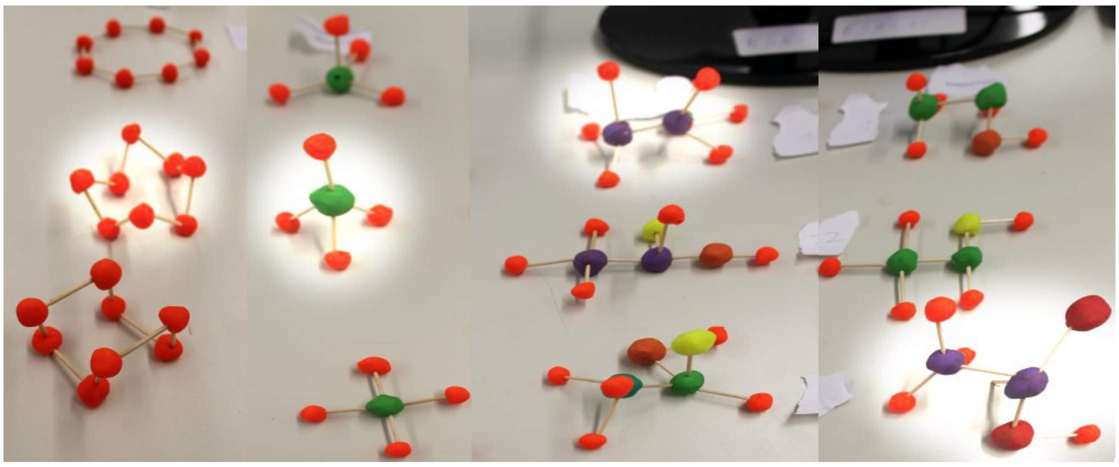


Figure 6.4: Typical setup to inspect and manipulate molecules on a monitor via keyboard and mouse in the *Jmol* application.

centered in the middle of the screen. By moving the mouse upwards or downwards, the molecule rotated around its horizontal axis. By moving the mouse leftwards or rightwards, the molecule rotated around its vertical axis. Schemes for translating and zooming molecules do exist in *Jmol*, but they were not required in the current setup. The students could view and explore the molecule from all sides before switching to the next molecule by pressing a button in the application. Figure 6.4 shows the *Jmol* setup that uses the mouse and the keyboard to manipulate the position and orientation of the molecule.

Further physical setups for further student tasks: In addition to the computer setup, the student desks carried papers and pencils and modeling clay with tooth picks. Furthermore, two to three clay models of each molecule were laid out on a table in a separate area in one of the rooms. Only one of these clay models of each molecule was correct. The other two were wrong with different degrees of spatial inappropriateness. Figure 6.5 shows the set of molecule versions for the first eight molecules. This is described further in task 5 of the next section.


(a) Molecules #1 to #4

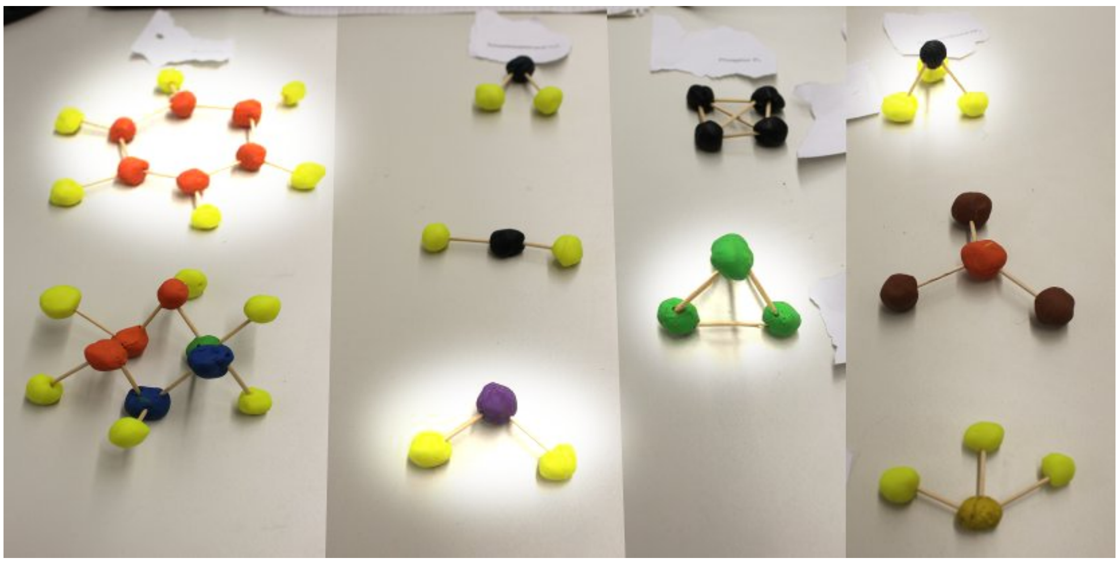

(b) Molecules #5 to #8

Figure 6.5: This is the set of versions of the first eight molecules. The correct versions are highlighted.

6.2.2 Evaluation Design

A between-subjects design was used, consisting of two separate groups. The first group, *Group ACR* consisted of 12 students, the second group, *Group Jmol* had 11 students. With the help of their teacher, the students were grouped to form a similar distribution of grades to ensure that both groups had the same knowledge.

The entire evaluation consisted of an introduction phase, five tasks including use of one of the two molecular visualization programs, and a brief subjective interview at the end.

Introduction phase (5 minutes): At the beginning, all students were in the same room. They received an exercise sheet with printed-out molecular formulas of all ten molecules. The paper also contained a short introduction to the topic and explained what the students were asked to do in this evaluation. We additionally explained the topic and the following tasks to the students.

Potentially confounding factors.

- The students were asked to work by themselves and not to copy from fellow students (cheat), due to the negative consequences to the evaluation. Yet, the potential for such an influence on the evaluation cannot be completely discarded.
- Since this is a between-subject design, learning effects are not crucial. For didactic reasons, we use the same, well-defined sequence through the set of molecules rather than a randomized order. If learning effects occur, they affect both conditions in a similar way and can thus be clearly identified. Yet, the test design consists of a large number of sequentially executed tasks, each involving all eight molecules, and required carry-over experiences between the tasks. Thus, learning can be seen as an omnipresent aspect over time.

Task 1: Drawings of all 10 molecular structures (15 minutes): As their first task, the students were asked to draw the LEWIS structure [60] for all molecules of the exercise sheet next to the molecular formulas – a topic that had been covered in class during the week before the evaluation. They previously were taught by their teacher how to draw this kind of structures. This should give the students a basic understanding of the connections of the atoms in the molecule. Figure 6.6 shows an example of a good (correct) and a bad (wrong) drawing.

After this first task the students were divided into the two groups and went to their respective computer rooms.

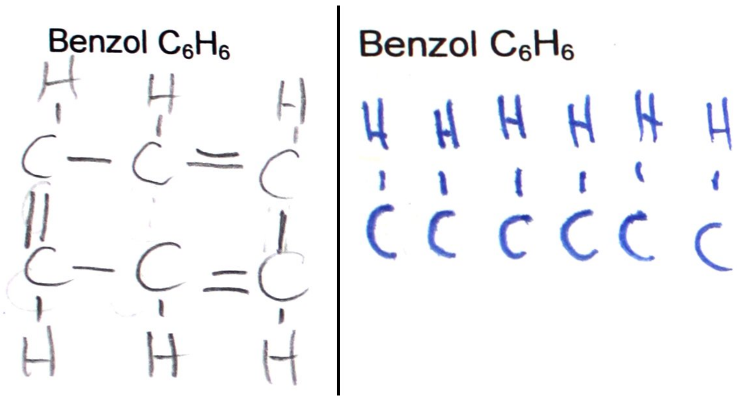


Figure 6.6: Example of a good and a bad drawing of the chemical formula of molecule #5 as LEWIS structure for Task 1.

Task 2: Uninformed modeling of all 10 spatial molecular structures (20 minutes): At their desks, students were asked to build models of the ten molecular formulas with clay and toothpicks, according to their current knowledge on how such a 3D structure could look like. They had not yet received any theoretical training on such 3D structures. We requested students to build these models in order to have reference data on the students' understanding of spatial molecular structures prior to using the computer-based chemical visualization applications. The students had 20 minutes to model up to 10 molecules with clay. We accepted that not everyone would complete this task. For the analysis, only the finished models were therefore taken into account. Figure 6.7 shows a model of the fourth molecule, acetic acid ($C_2H_4O_2$), built by a student.

Task 3: Explore 3D molecular structures with the respective visualization application (20 minutes): Each group was then asked to use their assigned visualization software to inspect all ten molecules. With *ACR*, the students sat in front of the display with the webcam above their shoulder and the marker cube in their hand. On the screen they saw the captured image plus their controlled virtual molecule rendered on top of the marker cube. The students could cycle through the set of molecules by pressing the space-bar on the keyboard. Figure 6.8 shows a part of the classroom with students working on the ACR version. With *Jmol*, the students used the mouse and the keyboard to rotate and move the molecules. To switch to the next molecule, they had to click a button in the software.

The students did not receive initial tutoring for either of the two software systems. Rather, they started immediately with the given molecular assignments. None were observed to have difficulties using the user interfaces.

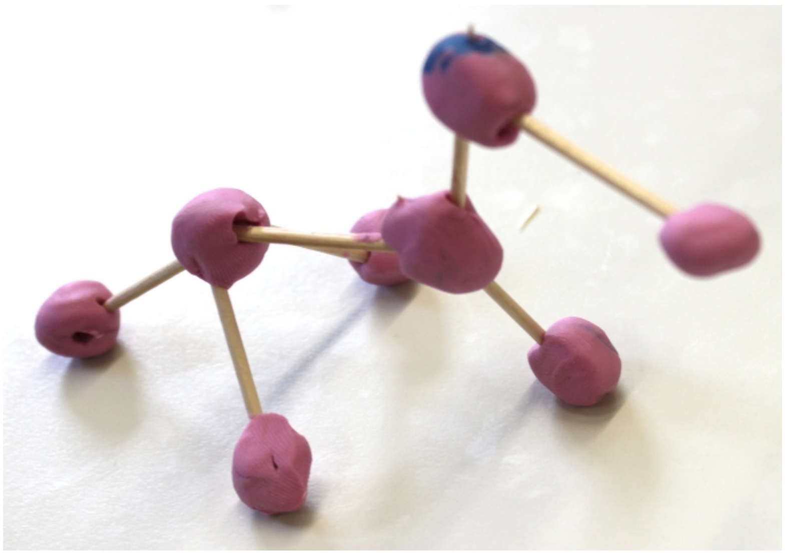


Figure 6.7: Model of acetic acid $C_2H_4O_2$ (molecule # 4), built by a student for Task 2. Students used modeling clay and toothpicks.

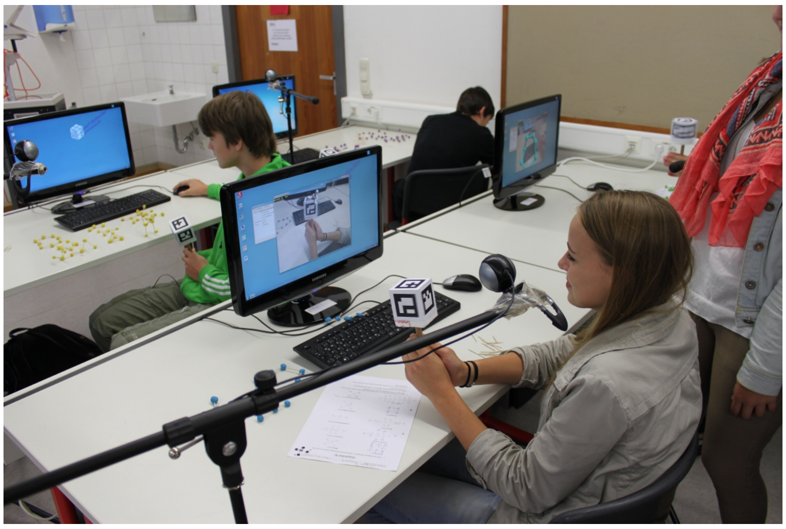


Figure 6.8: Classroom with students using the ACR for Task 3.

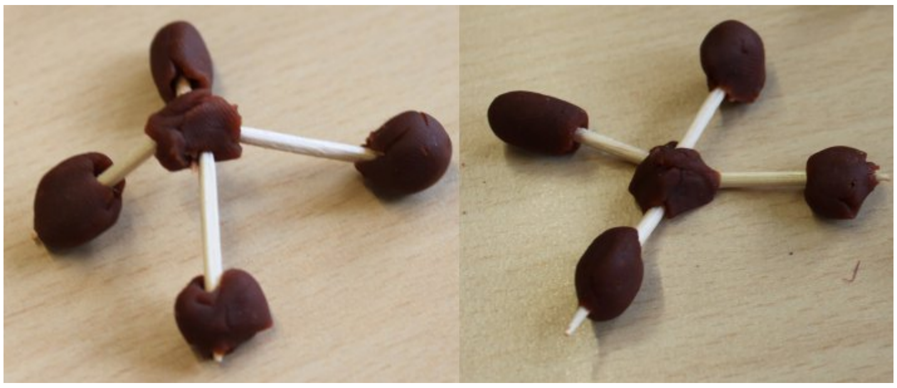
The students had 20 minutes to inspect all ten molecules. After this time, the applications were stopped. In the meantime I took photos of the molecules built for task 2.

Task 4: Informed modeling of all 10 spatial molecular structures (10 minutes): With their new knowledge of the spatial structure of the molecules, the students were asked to improve the molecular models they had built in task 2.

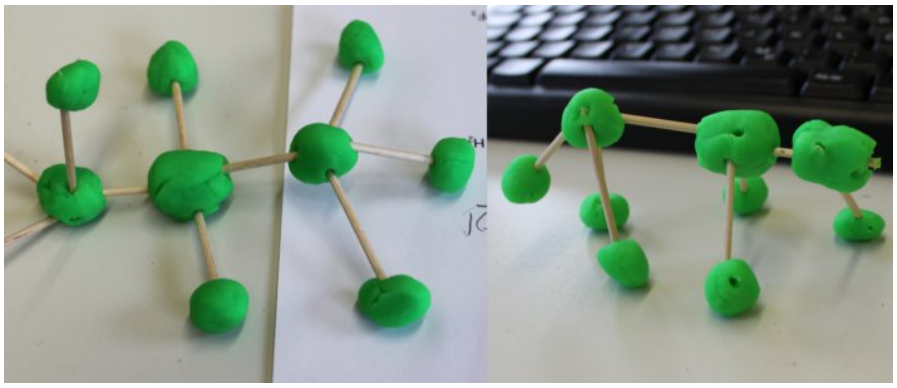
To measure how the 3D understanding of the spatial structure of the molecules changed, the initial version of the molecules were compared with the new version. Figure 6.9 shows typical clay models before and after using the software.

Task 5: Selection between several pre-built clay models of each molecule: To also get an objective measurement of how both 3D user interfaces improved the spatial understanding, we confronted the students with 2-3 pre-built clay models of the first eight molecular structures (see Figure 6.5). For the first eight molecules, we had built one clay model version that was the correct solution, one that was completely wrong and a third one that was almost correct, but still noticeably different from the correct one. Here, it is possible to evaluate to what extent the students had gained a spatial understanding of molecular structures. Figure 6.5 shows the alternative clay models for the first eight molecules.

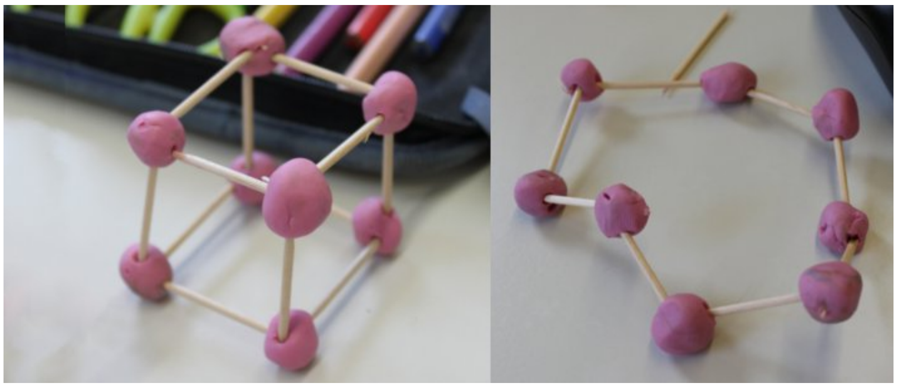
Closing phase: Informal interview and questionnaire: At the end of the evaluation a short joint informal interview was held in front of the whole class. A questionnaire was handed out to learn a bit about the students' prior knowledge. The students were asked what they liked and what they did not like. The students stated that they liked the idea of learning the molecule structure using a computer application. Especially the group using *ACR* told that they had a lot of fun using the user interface. Where the group with *Jmol* liked the general idea of using an application to show the 3D structure, the *ACR* users were fascinated about the user interface with the marker cube. All stated that they would like to continue using the application in their class to learn even more about the geometry of molecules.



(a) worsened (Mol. #2 Methane CH_4)



(b) improved (Mol. #3 Ethanole C_2H_5OH)



(c) strongly improved (Mol. #1 Sulfur S_8)

Figure 6.9: Before-after state of a worsening (a), normal improvement (b) and a large improvement (c) (Task 4)

6.3 Results

The evaluation consisted of two parts – the building and the improvement of the clay molecules and the selection of the right version. Table 6.5 at the end presents the raw absolute scores of all tasks. Empty cells denote that the students did not model the specific molecule or did not select any version in the last task.

The subsequent sections give the results, discuss these scores and suggest interpretations.

6.3.1 Improvements to Students' Clay Models

To measure how the AR-based and the mouse-based user interfaces of *ACR* and *Jmol* affected the spatial understanding of the virtual molecules, The models built in Task 2 were compared with the models changed in Task 4.

With the help of the chemistry teacher, the molecules were scored (Table 6.5). Table 6.1 presents the interpretation of the students' improvements from Task 2 to Task 4 – due to the use of the chemical education applications (*Jmol*, *ACR*). When the second version (Task 4) of the models was worse than the first version (Task 2), this was scored with -1 point. No improvement of the molecule was scored as 0 points, and an improvement was scored as $+1$ point. When the first version was totally wrong and the second was completely correct, this was scored $+2$ points. When students already provided a perfectly correct solution in the first version (Task 2) and they did not change anything on the molecule in Task 4, it could not be counted – as this does not deliver insight regarding the usefulness of the software system (user interface). The scores are presented in Table 6.1. A Kruskal-Wallis test shows that there is a significant difference in the medians $\chi^2(3, N = 45) = 32.34, p < 0.0001$ at a significance level of 5%. The results show that the direct manipulation 3D user interface of the *ACR* application helped the students significantly better than the keyboard and mouse 3D user interface of *Jmol*.

Table 6.2 and the corresponding graph in Figure 6.10 show how many percent of the molecules were *strongly improved*, *improved*, *unchanged* or *worsened* by students using the *Jmol* the *ACR* program, respectively. The numbers enhance the statistical finding that Group 1 (*ACR*) using the direct manipulation 3D user interface had a significantly better improvement than Group 2 (*Jmol*) using mouse and keyboard. Both the percentages for large improvements and for normal improvements are higher for *ACR* than for *Jmol*, whereas the percentages of no change and of worsening changes are smaller. A deeper analysis of the results shows that students of group 1 (*ACR*) who were wrong in Task 2 improved more in task 4 than students of group 2 (*Jmol*). This also shows that *ACR* with the direct manipulation user interface helps more to understand the spatial structures than using an indirect manipulation user interface with mouse and keyboard (*Jmol*).

Table 6.1: Scores representing students' improvements between uninformed modeling of the molecules in Task 2 and informed modeling in Task 4, i.e., after having visualized the molecules with *Augmented Chemical Reactions (ACR)* or *Jmol* int Task 3.

Group 1 (ACR)								
User \ Mol	1	2	3	4	5	6	7	8
1	-1		1					
2		2	1			2		
3	1		1	0	0			
4	1	1	2	0	0			
5		1	1	0				
6	1	1				0		
7	1	1	1					
8	0	2	1					
9	1	0	0	1	2	0	0	0
10	1	0	0	1		0	2	0
11	2			1			2	
12	0	0	0	0			2	

Group 2 (Jmol)								
User \ Mol	1	2	3	4	5	6	7	8
1	2	2		0		1	0	
2	0	0				0	0	
3	0	0				0	0	
4	1	-1	1					0
5	1	-1	0					
6	2		0	0			0	
7	0				1	1	-1	2
8	1		0	0			1	
9	1		0	0			2	
10	1		0	0			2	
11	1	1	0				1	

6.3.2 Students' Ability to Pick the Correct Clay Model out of a Given Set of Three per Molecule

Since students' dexterous abilities may vary and the quality of some of the students' clay models may thus have depended on that, Task 5 was designed as a test that was independent of the students' own modeling skills and time limitations. We had pre-

Table 6.2: Percentages of the quality the changes that students made to improve their clay models in Task 4.

	ACR	Jmol
Strongly improved	18%	13%
Improved	39%	29%
No change	41%	51%
Worsened	2%	7%

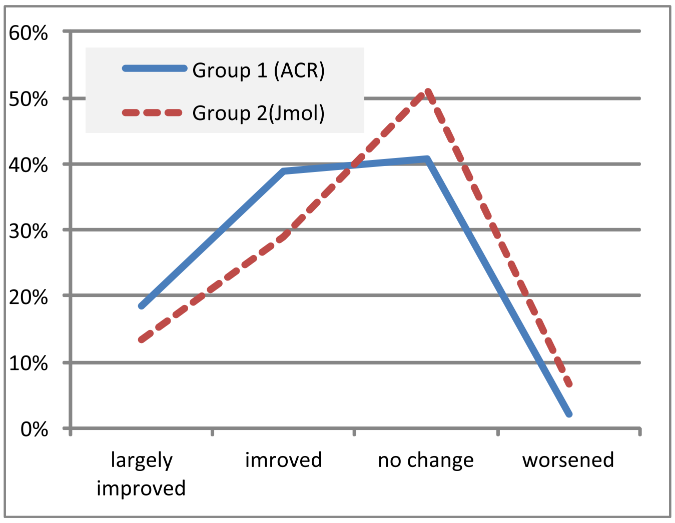


Figure 6.10: Percentages of the quality of the changes that students made to improve their clay models in Task 4. The height of the graph represents the percentage of the molecules which were *strongly improved*, *improved*, *no change* or *worsened*

pared three clay model versions of the first eight molecules, with one being correct, one being slightly wrong and one being completely wrong. The students were asked to indicate for each molecule which one they considered to be the correct model. They received 2 points for the correct answer, 1 point for the nearly correct answer, and 0 points when they selected the completely wrong clay model. Although the students were asked not to copy from the others, it cannot be guaranteed that they did not. Table 6.3 summarizes the score of table 6.5, pertaining to Task 5.

We calculated the average points that students achieved for each molecule. They are shown in Table 6.4 and in Figure 6.11. Interestingly, molecule #1 and #2 had a better result with the user interface of *Jmol*. The variance of these results of the group *Jmol* is unusual small in relation to the other molecules. This leads to the assumption that

Table 6.3: Scores of students' selections of three clay model versions of each molecule in Task 5.

Group 1 (ACR)								
User / Mol	1	2	3	4	5	6	7	8
1	2	2	2	2	2	2	2	2
2	2	2	1	1	2	2	2	2
3	1	2	2	0	2	2	2	1
4	1	2	2	2	0	1	2	2
5	1	1	1	1	0	0	2	2
6	1	0	2	2	2	0	2	1
7	2	0	2	2	2	1	2	2
8	2	2	2	2	2	2	2	2
9	2	0	2	1	2	2	2	2
10	1	2	2	1	2	2	2	1
11	2	2	1	2	2	2	2	2
12	2	2	2	2	2	2	2	2

Group 2 (Jmol)								
User / Mol	1	2	3	4	5	6	7	8
1	2	2	2	1	2	1	2	2
2	2	2	2	1	0	2	2	2
3	2	2	1	2	2	0	2	1
4	2	1	0	2	0	1	2	2
5	2	2	2	2	0	2	2	2
6	1	2	2	2	2	2	2	2
7	2	2	0	1	2	2	2	2
8	2	2	1	1	0	0	2	0
9	2	2	1	1	0	1	2	2
10	2	2	1	1	0	1	2	2
11	2	2	0	1	0	1	2	2

there was something unintended going on (copying from others). Molecule #1 with its crown-like structure can be seen in Figure 6.11. Molecule #2 has a simple tetrahedron structure with the carbon atom in the middle. For molecules #3, #4, #5, and #6, group *ACR* was better than group *Jmol*, while molecules #7 and #8 faired approximately even in both groups. The large difference in the results of Molecule #5 could be explained in the following way: Students using the *Jmol* application could not remember the flat structure of the molecule anymore, so they probably took the more complex looking

Table 6.4: Average number of points students achieved for each molecule in Task 5.

	ACR	Jmol
1	1.58	1.91
2	1.42	1.91
3	1.75	1.09
4	1.50	1.36
5	1.67	0.73
6	1.50	1.18
7	2.00	2.00
8	1.75	1.73

structure, whereas the students with the *ACR* could have remembered the flat structure. Molecule #7 and #8 were so easy that nearly everyone has picked the right version.

On average across all molecules, students of group *ACR* achieved 13.17 points, compared to 11.91 of group *Jmol*.

6.4 Discussion

Although the time for using the software was not very long, there is already a significant difference in the improvement of the spatial understanding of the 3D molecules. I think that by using a direct manipulation 3D user interface, students can grasp the spatial structure both in the literal and the symbolic sense. Whereas with mouse and keyboard, there is a mapping of the movements (2D horizontal movement of the mouse on the table results in a rotation of the virtual molecule on the screen). With this mapping, it seems to be not so easy to concentrate on the spatial structure of the virtual molecules. People are used to direct manipulation from their childhood on. Consequently this user interface is more natural and supports the learning of the spatial structures.

All students also mentioned that they had fun using the application and would like to use it more often in class. Fun is also one of the most important enablers in learning.

This evaluation showed that this assumption seems to be valid. Further investigations with a longer period of the study could investigate this finding in more detail.

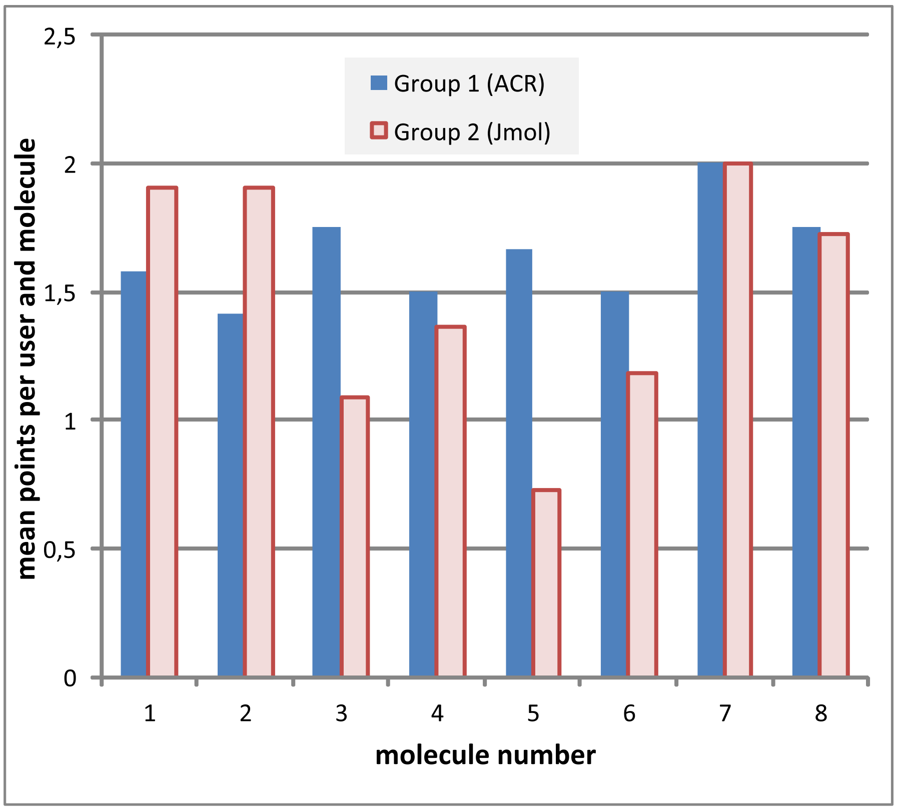


Figure 6.11: This diagram shows the mean points users had for each molecule with Task 5 (Selecting the right version of the molecules).

Table 6.5: Scores of all task assignments, T1, T2, T4 and T5

$Group1(ACR)$																																
$User$	$Mol.1$				$Mol.2$				$Mol.3$				$Mol.4$				$Mol.5$				$Mol.6$				$Mol.7$				$Mol.8$			
	$T1$	$T2$	$T4$	$T5$	$T1$	$T2$	$T4$	$T5$	$T1$	$T2$	$T4$	$T5$	$T1$	$T2$	$T4$	$T5$	$T1$	$T2$	$T4$	$T5$	$T1$	$T2$	$T4$	$T5$	$T1$	$T2$	$T4$	$T5$	$T1$	$T2$	$T4$	$T5$
1	1	2	0	2	2			2	2	1	2	2	0			2	0			2	1			2				2	0			2
2	2	3	3	2	2	1	3	2	2	1	2	1	0			1	2			2	1	0	3	2	1			2	2			2
3	1	0	2	1	2	3	3	2	2	1	2	2	0	1	1	0	1	0	0	2	1	3	3	2	1			2	3			1
4	2	0	2	1	2	2	3	2	0	1	3	2	0	2	2	2	2	1	1	0	1			1	0			2	2			2
5	2			1	2	1	2	1	0	0	1	1	0	0	0	1	1			0	1			0	0			2	2			2
6	1	0	2	1	2	0	1	0	1			2				2				2	1	0	0	0	0			2				1
7	2	0	2	2	2	1	2	0	0	1	2	2	0			2	1			2	1			1	1			2	2			2
8	1	2	2	2	2	1	3	2	0	1	2	2	0			2	0			2	1			2	1			2	2			2
9	1	1	2	2	2	1	1	0	0	1	1	2	0	0	1	1		0	3	2	1	0	0	2	1	0	0	2	2	0	0	2
10	1	0	2	1	2	0	0	2	2	1	1	2	1	0	1	1	2	3	3	2	1	0	0	2	1	0	3	2	2	1	1	1
11	1	0	3	2	2	3	3	2	2			1	2	2	3	2	1			2	2	3	3	2	1	0	3	2	2			2
12	1	0	0	2	2	0	0	2	2	0	0	2	2	0	0	2	1			2	1			2	1	0	3	2	2			2

$Group2(Jmol)$																																
$User$	$Mol.1$				$Mol.2$				$Mol.3$				$Mol.4$				$Mol.5$				$Mol.6$				$Mol.7$				$Mol.8$			
	$T1$	$T2$	$T4$	$T5$	$T1$	$T2$	$T4$	$T5$	$T1$	$T2$	$T4$	$T5$	$T1$	$T2$	$T4$	$T5$	$T1$	$T2$	$T4$	$T5$	$T1$	$T2$	$T4$	$T5$	$T1$	$T2$	$T4$	$T5$	$T1$	$T2$	$T4$	$T5$
1	1	1	3	2	2	1	3	2	2			2	0	0	0	1	0			2	1	0	1	1	1	0	0	2	2	3	3	2
2	1	1	1	2	2	1	1	2	2			2	0			1				0	1	0	0	2	1	1	1	2	2			2
3	1	1	1	2	2	1	1	2	2			1	0			2	0			2	1	0	0	0	1	1	1	2	2			1
4	2	0	2	2	2	3	1	1	0	1	2	0	0			2				0	1	3	3	1	0			2	2	1	1	2
5	0	0	2	2	2	2	0	2	0	1	1	2	0			2	0			0	1	3	3	2	0			2	2			2
6	1	0	3	1	2	3	3	2	2	2	2	2	2	2	2	2	2			2	1	3	3	2	1	2	2	2	2			2
7	2	0	0	2	2	3	3	2	2			0	2			1	0	2	3	2	1	1	2	2	1	1	0	2	2	1	3	2
8	1	0	2	2	2	3	3	2	2	1	1	1	2	1	1	1	2			0	1			0	1	1	2	2	2			0
9	1	0	2	2	2	3	3	2	2	1	1	1	2	1	1	1	2			0	1	3	3	1	1	0	3	2	2			2
10	1	0	2	2	2	3	3	2	2	1	1	1	2	1	1	1	2			0	1	3	3	1	1	0	3	2	2			2
11	1	0	2	2	2	2	3	2	2	2	2	0	1			1	2			0	1	3	3	1	1	1	2	2	2	1	1	2

7 Summary

User interfaces are a very important part in working with computer systems. In the still quite new area of Augmented Reality and Virtual Reality in general, the 3D user interface becomes more and more important. Lots of effort is put into the research of the 3D user interfaces which is also my field of research.

As selection and confirmation are main parts in 3D user interfaces, those were the main parts where I investigated and developed new gestures. Both actions go hand in hand. While most of the existing work relies on pointing devices to perform selections in the 3D environment and thus requires an available hand which is not assigned to another task, it becomes quite difficult when doing a two-handed 3D manipulation task at the same time. I developed and investigated mainly two different selection methods, based on gestures. One selection method uses a shaking gesture to cycle through an available set of selectable objects, where the second method is based on the proximity between two controlled objects. In the application of selecting bonds between two controlled objects, always the shortest possible bond is selected. As selecting the shortest bond between parts of complex structures that reside inside the structures is difficult and shaking through all possible combinations can be exhausting for the user, I combined both methods and introduced a partitioning of the complex structures. By shaking the complex structures, one can cycle through the partitions and only parts of the activated partitions can be selected with the proximity method.

After having selected a specific part in the 3D environment, the selection can be confirmed. In my thesis I also investigated new kinds of confirmation methods. One method is a back and forth gesture which is useful for the confirmation of links between controlled objects. The other method is a waiting method which is related to a gluing gesture. When the controlled objects are considered not to be moving anymore the confirmation is triggered. The trick lies in developing a good method for analyzing the movements of the tracked objects, distinguishing between natural tremor and intended motion.

The waiting gesture turned out to be usable in a lot of other fields where also precision is required. This was for example the case for collecting correspondence points for a head-mounted display calibration (SPAAM). Thus it was included in our UbiTrack library of our chair.

For getting expressive results in evaluations of the different methods, the studies of the 3D user interfaces have to be well designed, not to have any side effects from bad implementations of the surrounding system. I therefore put significant effort in designing all the components in the system as usable as possible. Starting from the quite easy but effective approach of mounting the camera on a microphone stand next to the user's head instead of mounting it on a tripod the user will always collide with. Also the software for doing my research was designed to give as much flexibility for the research as possible as well as a performant system for the user to concentrate on the parts in the user studies which were evaluated.

I also focused on researching new 3D user interface methods aiming for intuitivity. It is my principle that the application should adapt to the users' needs and not vice versa. When designing user interfaces, you always have to think of the user and how the user will intuitively interact with the system. For gesture recognition systems, you have to step back and observe it on a higher level to find out what the essentials of the gestures are.

The research on the 3D user interfaces was integrated in my application *Augmented Chemical Reactions* which should support the design of molecules as well as facilitate teaching chemistry. Students can now examine different binding principles and 3D structures of molecules from various points of view in a natural way. Understanding chemistry depends on understanding the spatial structure of the chemical elements in a molecule. When the spatial structure and the dynamic behavior of chemical molecules is conveyed to the students, chemical processes and chemistry per se have the potential to be understood better.

But as the application was mainly designed for research purposes, it is not ready to be used by everyone. The main configuration interface needs to be changed to be more intuitive to be used as well as the chemical functionality is very limited as it was not the focus of my research. More work can be done in integrating self-adjusting gesture recognition algorithms as described in section 5.2.5 to fit the different behaviors of the users. Additional work also can be done in providing more feedback to the user by integrating vibration motors in the tracked handles as we already did in a prototype in collaboration with the UISPA group [42] in the Institute of Mechanical Engineering of the University of Porto.

My vision is that it will become more easy to use systems in 2D as well as 3D. User interfaces à la *Minority Report* are already possible but still need a lot of research to be really intuitive. I hope that with this thesis I could make my contribution to the realization of this vision.

Bibliography

[1] Advanced Realtime Tracking GmbH. *ART Advanced Realtime Tracking, http://www.ar-tracking.com*. Last accessed, Nov 2013 (cited on pages 5, 29, 88).

[2] Oksana Arnold et al. “Exploring and Understanding the Abstract by Direct Manipulation of the Concrete”. In: *CSEDU (2)*. 2012, pages 100–107 (cited on page 100).

[3] Magnus Axholt et al. “Optical See-Through Head Mounted Display Direct Linear Transformation Calibration Robustness in the Presence of User Alignment Noise”. In: *Proceedings of the Human Factors and Ergonomics Society 54rd Annual Meeting*. 2010 (cited on page 83).

[4] M. Axholt et al. “Parameter Estimation Variance of the Single Point Active Alignment Method in Optical See-Through Head Mounted Display Calibration”. In: *Proceedings of the IEEE Virtual Reality Conference*. 2011 (cited on pages 83, 84).

[5] M. Axholt, S.D. Peterson, S. Ellis, et al. “Visual Alignment Accuracy in Head Mounted Optical See-Through AR Displays: Distribution of Head Orientation Noise”. In: *Human Factors and Ergonomics Society Annual Meeting Proceedings*. Volume 53. 27. 2009, pages 2024–2028 (cited on page 84).

[6] Ronald T Azuma et al. “A survey of augmented reality”. In: *Presence* 6.4 (1997), pages 355–385 (cited on page 13).

[7] T Baguley. *An Introduction to Sphericity, http://homepages.gold.ac.uk/aphome/spheric.html*. Last accessed, May 2004 (cited on page 90).

[8] R. Balakrishnan and K. Hinckley. “Symmetric bimanual interaction”. In: *Proc. of SIGCHI*. ACM New York, NY, USA. 2000, pages 33–40 (cited on page 11).

[9] R. Balakrishnan et al. “Digital tape drawing”. In: *Proc. of Symposium on User Interface Software and Technology*. ACM New York, NY, USA. 1999, pages 161–169 (cited on page 11).

[10] J.F. Bartlett. “Rock ’n’ Scroll Is Here to Stay”. In: *IEEE Computer Graphics and Applications* (May 2000), pages 40–45 (cited on page 12).

[11] Bijvoet Center, Universiteit Utrecht. *PLATON, http://www.cryst.chem.uu.nl/platon*. accessed Nov. 13, 2009 (cited on page 9).

[12] Doug A. Bowman. “3D User Interfaces”. In: *The Encyclopedia of Human-Computer Interaction, 2nd Ed.* Edited by Mads Soegaard and Rikke Friis Dam. Aarhus, Denmark: The Interaction Design Foundation, 2013 (cited on page 1).

[13] Doug A Bowman et al. *3D user interfaces: theory and practice*. Addison-Wesley, 2004 (cited on pages 1, 38).

[14] J. Brooke. “SUS-A quick and dirty usability scale”. In: *Usability evaluation in industry* (1996), pages 189–194 (cited on pages 52, 72).

[15] D. Casalta, Y. Guiard, and M.B. Lafon. “Evaluating two-handed input techniques: Rectangle editing and navigation”. In: *Proc. of SIGCHI*. ACM. 1999, page 237 (cited on page 11).

[16] M. Chen, S.J. Mountford, and A. Sellen. “A study in interactive 3-D rotation using 2-D control devices”. In: *ACM SIGGRAPH Computer Graphics*. Volume 22. 4. ACM. 1988, pages 121–129 (cited on page 100).

[17] A. Choumane, G. Casiez, and L. Grisoni. “Buttonless Clicking: Intuitive select and pick-release through gesture analysis”. In: *Proceedings of the 17th IEEE Conference on Virtual Reality (VR)*. Mar. 2010, pages 67–70 (cited on page 58).

[18] CSSS. *Molekel, http://molekl.csss.ch*. accessed Nov. 13, 2009 (cited on page 9).

[19] JL Demer, J. Goldberg, and FI Porter. “Effect of telescopic spectacles on head stability in normal and low vision.” In: *Journal of vestibular research* 1.2 (1991), page 109 (cited on page 83).

[20] Florian Echtler. “Tangible information displays”. PhD thesis. München, Techn. Univ., Diss., 2009, 2009 (cited on pages 5, 32).

[21] Fakespace Labs, Inc. *The Pinch Gloves, http://www.fakespacelabs.com/tools.html*. Last accessed, Dec 2013 (cited on page 58).

[22] S. Feiner et al. “A touring machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment”. In: *Personal and Ubiquitous Computing* 1.4 (1997), pages 208–217 (cited on page 84).

[23] FireRescue1. “Swissphone Emergency call system triggers an alert when no movement is detected”. In: *FireRescue1, http://www.firerescue1.com/fire-products/communications/press-releases/394524-Swissphone-Emergency-call-system-triggers-an-alert-when-no-movement-is-detected* (Apr. 2008) (cited on page 58).

[24] Morten Fjeld, Patrick Juchli, and Benedikt Voegtli. “Chemistry Education: A Tangible Interaction Approach”. In: *Proceedings of IFIP INTERACT03: Human-Computer Interaction*. IFIP Technical Committee No 13 on Human-Computer Interaction, 2003, page 287 (cited on page 7).

[25] Morten Fjeld and Benedikt M. Voegtli. “Augmented Chemistry: An Interactive Educational Workbench”. In: *ISMAR ’02: Proceedings of the 1st International Symposium on Mixed and Augmented Reality*. Washington, DC, USA: IEEE Computer Society, 2002, page 259. ISBN: 0-7695-1781-1 (cited on pages 7, 102).

[26] M. Fjeld et al. “Tangible user interface for chemistry education: comparative evaluation and re-design”. In: *Proceedings of SIGCHI*. ACM. 2007, page 808 (cited on pages 8, 100).

[27] Gaussian, Inc. *GaussView, http://www.gaussian.com*. accessed Nov. 13, 2009 (cited on page 9).

[28] C. Geiger and O. Rattay. “TubeMouse-A two-handed input device for flexible objects”. In: *IEEE Symposium on 3D User Interfaces, 2008. 3DUI 2008*. Mar. 2008, pages 27–34 (cited on page 58).

[29] A. Gillet et al. “Augmented Reality with Tangible Auto-Fabricated Models for Molecular Biology Applications”. In: *IEEE Visualization*. IEEE Computer Society, 2004, pages 235–241 (cited on page 8).

[30] Gordon Research Group, Iowa State University. *The General Atomic and Molecular Electronic Structure System (GAMESS), http://www.msg.ameslab.gov/GAMESS/GAMESS.html*. accessed Nov. 13, 2009 (cited on page 9).

[31] Y. Guiard. “Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model”. In: *Journal of Motor Behavior* 19 (1987), pages 486–517 (cited on pages 10, 12).

[32] S. Güven, S. Feiner, and O. Oda. “Mobile Augmented Reality Interaction Techniques for Authoring Situated Media On-Site”. In: *Proc. of ISMAR*. Santa Barbara, CA, USA: IEEE, 2006, pages 235–236 (cited on page 12).

[33] T. Ha and W. Woo. “An empirical evaluation of virtual hand techniques for 3D object manipulation in a tangible augmented reality environment”. In: *Proceedings of the IEEE Symposium on 3D User Interfaces*. IEEE. 2010, pages 91–98 (cited on page 84).

[34] P. HANCOCK and M. CHIGNELL. “Mental workload dynamics in adaptive interface design”. In: *IEEE transactions on Systems, Man, and Cybernetics* 18 (1988), pages 647–658 (cited on page 76).

[35] T. Herman et al. “Tactile teaching: Exploring protein structure/function using physical models*”. In: *Biochemistry and Molecular Biology Education* 34.4 (2006), pages 247–254 (cited on page 100).

[36] K. Hinckley et al. “Two-Handed Virtual Manipulation”. In: *ACM Transactions on Computer-Human Interaction* 5.3 (1998), pages 260–302 (cited on page 11).

[37] K. Hinckley et al. “Usability analysis of 3D rotation techniques”. In: *Proceedings of the 10th annual ACM symposium on User interface software and technology*. ACM. 1997, pages 1–10 (cited on page 100).

[38] Nady El Hoyek et al. “Experimental Research Validation for the Use of 3D in Teaching Human Anatomy”. In: *CSEDU (2)*. Edited by Alexander Verbraeck et al. SciTePress, 2011, pages 225–227 (cited on page 100).

[39] Manuel Huber et al. “A System Architecture for Ubiquitous Tracking Environments”. In: *Proceedings of the 6th International Symposium on Mixed and Augmented Reality (ISMAR)*. Nov. 2007, pages 211–214 (cited on pages 5, 20, 23, 29, 49, 100).

[40] HYPERCUBE, Inc. *HyperChem, http://www.hyper.com*. accessed Nov. 13, 2009 (cited on page 9).

[41] IEEE 9th Symposium on 3D User Interfaces. *IEEE 3DUI 2014 Contest Instructions,http://www.3dui.org/contest*. Last accessed, Dec 2013 (cited on page 57).

[42] Institute of Mechanical Engineering, University of Porto. *UISPA || System Integration and Process Automation Unit, http://paginas.fe.up.pt/wwwidmec/uispa/*. Last accessed, Nov 2013 (cited on page 120).

[43] International Union of Crystallography (IUCr). *Rasmol, http://rasmol.org*. accessed Nov. 13, 2009 (cited on page 9).

[44] E. Marcia Johnson et al. “ICT/ eLearning for Developing Visual Spatial Thinking in University Science Teaching”. In: *CSEDU (2)’11*. 2011, pages 73–78 (cited on page 100).

[45] Vladimir Kajalin. “Screen space ambient occlusion”. In: *Shader X 7* (2009), pages 413–24 (cited on page 19).

[46] Franziskus Karsunke. “Controlling 3D objects by using a multitouch surface with gesture recognition”. Bachelor Thesis. Technische Universität München, June 2009 (cited on page 32).

[47] H. Kato and M. Billinghurst. “Marker Tracking and HMD Calibration for a Video-Based Augmented Reality Conferencing System”. In: *Proc. IEEE International Workshop on Augmented Reality (IWAR’99)*. 1999, pages 85–94 (cited on page 8).

[48] Hannes Kaufmann and Bernd Meyer. “Simulating educational physical experiments in augmented reality”. In: *SIGGRAPH Asia ’08: ACM SIGGRAPH ASIA 2008 educators programme*. Singapore: ACM, 2008, pages 1–8. ISBN: 978-1-60558-388-4 (cited on page 7).

[49] Leap Motion, Inc. *LEAPMotion, https://www.leapmotion.com*. Last accessed, Nov 2013 (cited on page 32).

[50] Olaf Lippold. “Physiological Tremor”. In: *Scientific American* 224 (Mar. 1971), pages 65–73 (cited on page 83).

[51] M.A. Livingston et al. "Vertical Vergence Calibration for Augmented Reality Displays". In: (2006) (cited on page 84).

[52] J.C.A. Looser. "AR Magic Lenses: Addressing the Challenge of Focus and Context in Augmented Reality". PhD thesis. University of Canterbury. Computer Science and Software Engineering, 2007 (cited on page 3).

[53] Patrick Maier and Gudrun Klinker. "Evaluation of an Augmented-Reality-based 3D User Interface to Enhance the 3D-Understanding of Molecular Chemistry". In: *Proceedings of the 5th International Conference on Computer Supported Education (CSEDU 2013)*. Edited by Owen Foley et al. SciTePress, 2013, pages 294–302 (cited on pages 99, 100).

[54] Patrick Maier, Marcus Tönnis, and Gudrun Klinker. "Augmented Reality for teaching spatial relations". In: *CD-ROM Proceedings from the Conference of the International Journal of Arts and Sciences*. ISSN: 1943-6114. 2009 (cited on pages 1, 7, 100).

[55] Patrick Maier, Marcus Tönnis, and Gudrun Klinker. "Designing and Comparing Two-Handed Gestures to Confirm Links between User Controlled Objects". In: *9th IEEE and ACM International Symposium on Mixed and Augmented Reality*. 2010 (cited on page 86).

[56] Patrick Maier, Marcus Tönnis, and Gudrun Klinker. "Dynamics in Tangible Chemical Reactions". In: *Proceedings from the International Conference on Chemical Engineering (ICCE 2009)*. ISSN: 2070-3724. 2009 (cited on pages 1, 7, 13, 100).

[57] Patrick Maier et al. "An Empiric Evaluation of Confirmation Methods for Optical See-Through Head-Mounted Display Calibration". In: Oct. 2012, pages 73–80 (cited on pages 37, 80).

[58] Patrick Maier et al. "What do you do when two hands are not enough? Interactive selection of bonds between pairs of tangible molecules". In: *Proceedings of the 5th IEEE Symposium on 3D User Interfaces (3D UI)*. Mar. 2010 (cited on page 37).

[59] E. Mcgarrity et al. "Evaluation of calibration for optical see-through augmented reality systems". In: (2001) (cited on page 83).

[60] A.D. McNaught and A. Wilkinson. "Compendium of chemical terminology". In: volume 1669. Blackwell Science Oxford, UK, 1997. Chapter Lewis formula (electron dot or Lewis structure), page 1135 (cited on page 106).

[61] mdx4ever. *Creating A Scene-Graph in XNA, http://www.madgamedev.com/post/2009/10/28/Article-Creating-A-Scene-Graph-in-XNA.aspx*. Last accessed, Nov 2013 (cited on page 19).

[62] E. Medina, Y. Chen, and S. Weghorst. “Understanding biochemistry with Augmented Reality”. In: *C. Montgomerie & J. Seale (Eds.) Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2007*. Chesapeake, VA: AACE, 2007, pages 4235–4239 (cited on page 8).

[63] Minnesota Supercomputer Center. *JMol, http://jmol.sourceforge.net*. accessed Nov. 13, 2009 (cited on pages 9, 99, 100, 103).

[64] Molecular Graphics Lab. *mobile AR, http://mgl.scripps.edu/projects/tangible_models/mobile-ar*. Last accessed, Apr 2014 (cited on page 8).

[65] NASA. *NASA TLX: Task Load Index, http://humansystems.arc.nasa.gov/groups/TLX*. accessed May, 2009 (cited on page 8).

[66] Nassir Navab et al. “An On-line Evaluation System for Optical See-through Augmented Reality”. In: *Presence: Teleoperators and Virtual Environments*. Volume vol. 11. June 2002, pages (cited on page 81).

[67] J.N. Ouellet and P. Hebert. “A simple operator for very precise estimation of ellipses”. In: *Computer and Robot Vision, 2007. CRV'07. Fourth Canadian Conference on*. IEEE. 2007, pages 21–28 (cited on page 94).

[68] Charles B. Owen et al. “Display-Relative Calibration for Optical See-Through Head-Mounted Displays”. In: *Proceedings of the 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality*. 2004, pages 70–78 (cited on pages 81, 88).

[69] C. G. Panagiotopoulos, G. D. Manolis, and A. Athanatopoulou. “Edusoft Package for Structural Engineering - Web-based Educational Material using JAVA for Structural Dynamics.” In: *CSEDU (2)*. Edited by Markus Helfert, Maria João Martins, and José Cordeiro. SciTePress, 2012, pages 299–303 (cited on page 100).

[70] Robert G Parr and Weitao Yang. *Density-functional theory of atoms and molecules*. Volume 16. Oxford university press, 1989 (cited on page 10).

[71] J.C. Phillips et al. “Scalable molecular dynamics with NAMD”. In: *Journal of Computational Chemistry* 26.16 (2005), page 1781 (cited on pages 10, 18).

[72] J.S. Pierce, B.C. Stearns, and R. Pausch. “Voodoo dolls: seamless interaction at multiple scales in virtual environments”. In: *Proc. of Symposium on Interactive 3D Graphics*. ACM. 1999, pages 141–145 (cited on page 11).

[73] Ivan Poupyrev et al. “The go-go interaction technique: non-linear mapping for direct manipulation in VR”. In: *Proceedings of the 9th annual ACM symposium on User interface software and technology*. ACM. 1996, pages 79–80 (cited on page 38).

[74] R. Paine, Chemistry Dept., Univ. of New Mexico. *TINKER Software Tools for Molecular Design, http://dasher.wustl.edu/tinker*. accessed Nov. 13, 2009 (cited on pages 9, 15, 49).

[75] Radboud University Mijmegen, Centre for Molecular Life Sciences (CMBI), the Netherlands. *MOLDEN, http://www.cmbi.ru.nl/molden*. accessed Nov. 13, 2009 (cited on page 9).

[76] M. Ricknäs. "Gestures Set to Shake up Mobile User Interfaces". In: *PC World, http://www.pcworld.com/article/171535* (Sept. 2009) (cited on page 12).

[77] SCM, Vrije Universiteit Amsterdam, Theoretical Chemistry. *Amsterdam Density Funcitional software (ADF), http://www.scm.com/*. accessed Nov. 13, 2009 (cited on page 9).

[78] A. Steed. "Towards a general model for selection in virtual environments". In: *Proceedings of the IEEE Symposium on 3D User Interfaces*. 2006, pages 103–110 (cited on pages 59, 84).

[79] Richard Stoakley, Matthew J Conway, and Randy Pausch. "Virtual reality on a WIM: interactive worlds in miniature". In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM Press/Addison-Wesley Publishing Co. 1995, pages 265–272 (cited on page 38).

[80] Z. Szalavári and M. Gervautz. "Using the personal interaction panel for 3d interaction". In: *Proc. of Conference on Latest Results in Information Technology*. 1997, page 36 (cited on page 3).

[81] The Molecular Graphics Laboratory. *The Molecular Graphics Laboratory, http://mgl.scripps.edu*. Last accessed, Nov 2013 (cited on pages 10, 18).

[82] The OpenSceneGraph Project. *OpenSceneGraph, http://www.openscenegraph.org*. Last accessed, Nov 2013 (cited on page 19).

[83] Theoretical and Computational Biophysics Group. *Mindy - A 'minimal' molecular dynamics program, http://www.ks.uiuc.edu/Development/MDTools/mindy/*. Last accessed, Nov 2013 (cited on pages 10, 18).

[84] Mihran Tuceryan, Yakup Genc, and Nassir Navab. "Single-Point active alignment method (SPAAM) for optical see-through HMD calibration for augmented reality". In: 11 (3 June 2002), pages 259–276 (cited on pages 80, 81, 83, 85, 88).

[85] Mihran Tuceryan and Nassir Navab. "Single point active alignment method (SPAAM) for optical see-through HMD calibration for AR". In: *Augmented Reality, 2000.(ISAR 2000). Proceedings. IEEE and ACM International Symposium on*. 2000, pages 149–158 (cited on page 83).

[86] TURBOMOLE GmbH, Universität Karlsruhe. *TURBOMOLE, http://www.turbomole-gmbh.com*. accessed Nov. 13, 2009 (cited on page 9).

[87] University College Cork. *Software Usability Measurement Inventory, http://sumi.ucc.ie/*. Last accessed, May 2009 (cited on page 8).

[88] University of Coimbra, University of Porto. *The 2nd Experiment International Conference, http://paginas.fe.up.pt/ expat/*. Last accessed, Nov 2013 (cited on page 27).

[89] University of Porto. *University of Porto, http://www.up.pt*. Last accessed, Nov 2013 (cited on page 27).

[90] Suzanne Weghorst. "Augmenting Tangible Molecular Models". In: *Proceedings of International Conference on Artificial Reality and Telexistence*. Tokyo, Japan: IEEE, 2003, pages 1–6 (cited on page 8).

[91] Sean White, David Feng, and Steven Feiner. "Interaction and Presentation Techniques for Shake Menus in Tangible Augmented Reality". In: *Proceedings of the 8th International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2009, pages 39–48 (cited on pages 12, 40).

[92] Sean White, Levi Lister, and Steven Feiner. "Visual Hints for Tangible Gestures in Augmented Reality". In: *Proceedings of the 6th International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE Computer Society, 2007, pages 1–4 (cited on page 12).

[93] Wikipedia. "Tooltip". In: *Wikipedia, http://en.wikipedia.org/wiki/Tooltip* (Apr. 2010) (cited on page 59).

[94] X. Xia, P. Irani, and J. Wang. "Evaluation of Guiard's Theory of Bimanual Control for Navigation and Selection". In: *Lecture Notes in Computer Science* 4566 (2007), page 368 (cited on page 11).

[95] Xsens Technologies B.V. *Xsens: 3D Motion Tracking, http://www.xsens.com*. Last accessed, Nov 2013 (cited on pages 22, 31).

[96] S. Zhai. "User performance in relation to 3D input device design". In: *ACM Siggraph Computer Graphics* 32.4 (1998), pages 50–54 (cited on page 100).

[97] Ji Zhou. *Calibration of optical see through head mounted displays for augmented reality*. ProQuest, 2007 (cited on page 81).