tCAD:
A 3D Modeling Application on a Depth Enhanced Tabletop Computer

DISSERTAÇÃO DE MESTRADO

Paulo Alexandre Câmara Bala
MESTRADO EM ENGENHARIA INFORMÁTICA

UNIVERSIDADE da MADEIRA
A Nossa Universidade
www.uma.pt

Setembro | 2012

tCAD:
A 3D Modeling Application on a Depth Enhanced Tabletop Computer

DISSERTAÇÃO DE MESTRADO

Paulo Alexandre Câmara Bala
MESTRADO EM ENGENHARIA INFORMÁTICA

ORIENTAÇÃO
Ian Oakley

CO-ORIENTAÇÃO
Augusto Esteves

ABSTRACT

Tabletop computers featuring multi-touch input and object tracking are a common platform for research on Tangible User Interfaces (also know as Tangible Interaction). However, such systems are confined to sensing activity on the tabletop surface, disregarding the rich and relatively unexplored interaction canvas above the tabletop. This dissertation contributes with tCAD, a 3D modeling tool combining fiducial marker tracking, finger tracking and depth sensing in a single system. This dissertation presents the technical details of how these features were integrated, attesting to its viability through the design, development and early evaluation of the tCAD application. A key aspect of this work is a description of the interaction techniques enabled by merging tracked objects with direct user input on and above a table surface.

Keywords: Depth-sensing, Tangible User Interfaces, 3D object manipulation.

RESUMO

Computadores *Tabletop* com input multi-toque e reconhecimento de objectos são uma plataforma comum para pesquisa em Interfaces de Utilizador Tangíveis (também conhecida como Interacção Tangível). Contudo, tais sistemas tão confinados a detectar actividade na superfície, ignorando a rica e relativamente inexplorada tela de interacção acima do *tabletop*. Esta dissertação contribui com tCAD, uma ferramenta de modelação 3D combinando reconhecimento de marcadores fiduciais, reconhecimento de dedos e detecção de profundidade num único sistema. Esta dissertação apresenta os detalhes técnicos de como estas características foram integradas, atestando a sua viabilidade através do design, desenvolvimento e avaliação precoce da aplicação tCAD. Um aspecto chave deste trabalho é a descrição das técnicas de interacção permitidas pela fusão do reconhecimento de objectos com input directo do utilizador na e acima da superfície da mesa.

Palavras chave: Detecção de Profundidade, Interfaces de Utilizador Tangíveis, Manipulação de Objectos 3D.

ACKNOWLEDGMENTS

Firstly I would like to express my gratitude to my advisor Dr. Ian Oakley, for his wise guidance and willingness to quench my questions and concerns. I would also like to thank my co-advisor Augusto Esteves, for helping me expand and clarify my inspiration into a working Tangible User Interface.

Secondly, I would like to thank my family for their love and support at the times I needed it the most.

Lastly, I need to thank my friends for motivating me and distracting me in equal parts. Sorry for the times I pretended to be listening to you but was actually zoned out thinking about code…

For all those who helped in my dissertation, I can only repeat William Shakespeare and say "I can no other answer make, but, thanks, and thanks".

CONTENTS

Figures

Tables

1 INTRODUCTION

Tangible User Interfaces (TUI), also known as Tangible Interaction, is a growing field of study in Human Computer Interaction (HCI). Its focus on systems that allow users to experience tangible representations of digital information via physical graspable controls [UI00], promotes the use of real world skills, leverages the natural affordances of objects and establishes a strong basis for collaborative work, memorization and learning [JGH+08]. Gartner's 2010 Hype Cycle for Emerging Technologies [URL1] (see Figure 1) places Tangible User Interfaces in the "Technology Trigger" phase, aiming for mainstream adoption in the next 10 years. This Hype Cycle justifies this position by stating that TUIs are bounded to research projects and lacking in commercial viability. Nonetheless, this should change in the next decade as research rethinks the user experience brought upon by this technology and as development toolsets are refined [URL1].

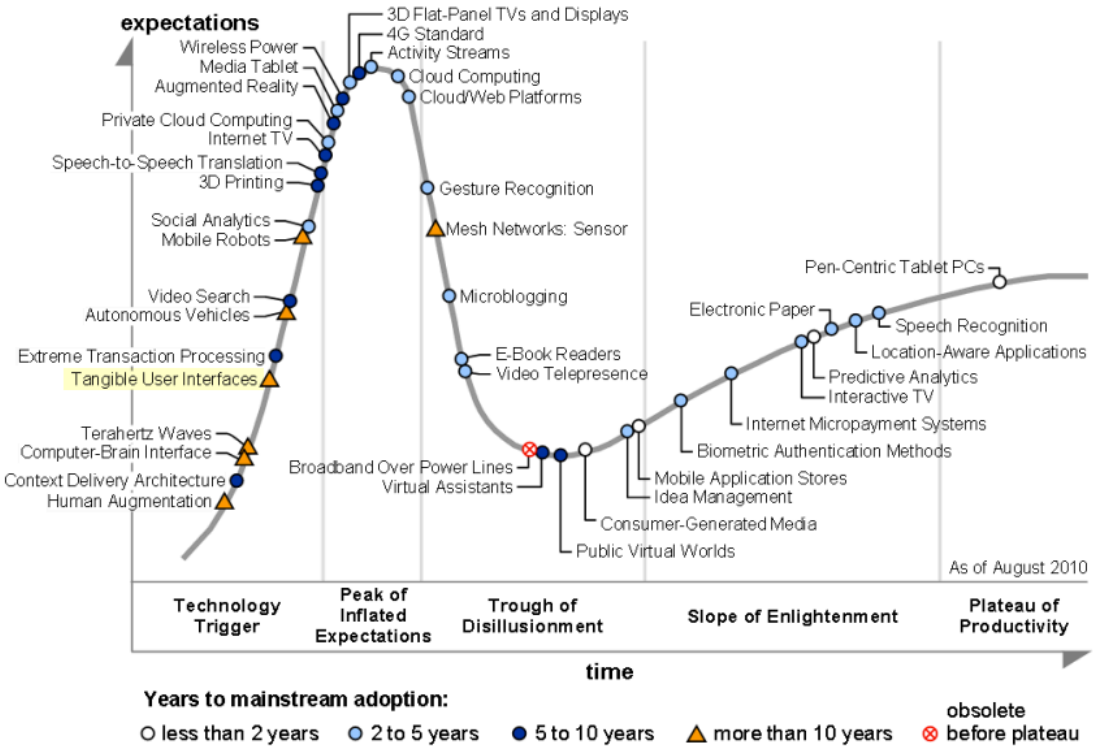


Figure 1 Hype Cycle for Emerging Technologies [URL1]. Tangible User Interfaces, highlighted in yellow, is at the "Technology Trigger" phase.

Arguably the dominant paradigm for work on TUIs is via tracking of fiducial markers and touch on tabletop computers [KB07]. Due to a combination of the relatively low cost, high performance, simple construction, flexibility and reliability of such systems [MF12], numerous works have emerged that explore tangibility in tabletops.

Muller and Fjeld's [MF12] analysis of the Hype Cycle of Tabletop Technology (see Figure 2) reveals that the peak of visibility and expectations for tabletop technology is coincident with popularization of fiducial tracking (triggered by reacTable [JGA+07, J08]) and with the paradigm shift from single touch to multi-touch and tangibility. This leads to the unavoidable question: "Why aren't tabletops a part of our day to day?". After a peak period (characterized with extensive research, publications and media attention), commercial solutions became available but were cost prohibitive. Stunted by unaffordable solutions, tabletop technology visibility began to suffer. However, the unveiling of new technologies (capacitive screens, Surface 2.0, etc.), offering better performance at affordable prices, promises an increase in visibility and research. Tabletop technology is standing on the edge of enlightenment and is in the cusp of mainstream adoption.

How can TUIs ride the incoming wave of Tabletop Technology?

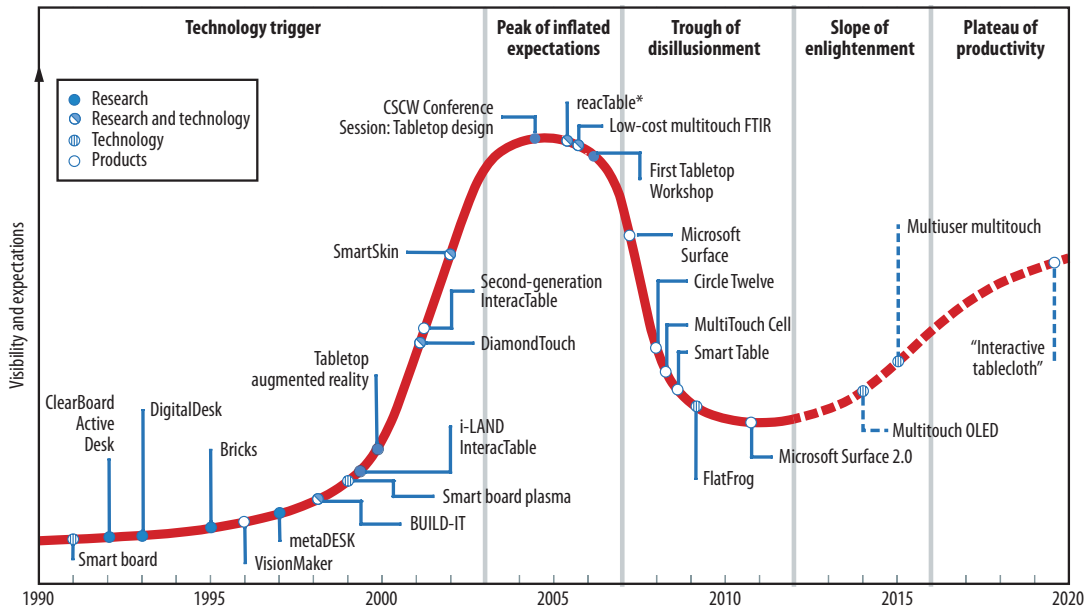


Figure 2 Hype Cycle for Tabletop Technology [MF12]. Current technology is at the start of the "Slope of enlightenment" phase.

1.1 THESIS GOALS AND MOTIVATION

Considering this expected boost in tabletop research, this thesis will defend that tabletop technology improvement can benefit Tangible User Interface research, by dealing with some of its limitations. Additionally, performance improvement and enhancement of previous tabletop technology promotes the idea that tabletops are not merely destined for lab research, but also for commercial purposes, as they can be upgraded and extended to deal with new interaction paradigms.

Specifically, this thesis will delve into the limitation of fiducial tracking and touch on tabletop computers, an emerging trend in tabletop research. Since fiducial tracking and touch events are detected directly on the tabletop surface, the richness of interactions, manipulations and events that take place above the surface are ignored. This has the potential to introduce significant disconnects between users' expectations of interacting with objects freely in the real world and the system's ability to sense simple planar manipulations [H12].

Consequently, a developing trend in tabletop research is in extending the interaction space to encompass the volume above the table surface. Past approaches include sensing the height and configuration of objects placed on a table [BBR10] or directly sensing user movements to support gesture recognition and designing for proxemics [AGW+11]. A key concept in these efforts is the notion of a Continuous Interaction Space (CIS), where the tabletop surface and the volume above it are seen as a unified whole [JMG+11]. While promising work on this topic is at an early stage, much attention is on the features and viability of sensing setups; many authors focus on the creation of complex bespoke hardware setups. This thesis will argue that the challenges of creating and replicating these sensing systems is impeding research into the potential interactions enabled by the notion of a Continuous Interaction Space [JMG+11].

1.2 THESIS CONTRIBUTION

This thesis aims to contribute by tackling the issue of complex bespoke hardware setups. Therefore, this thesis explores the practicalities of how a consumer depth camera (Microsoft Kinect [URL2]) can be combined with a standard tabletop computer to create a powerful and expressive Continuous Interaction Space [JMG+11] simply and at low cost, describing the iterative process of development.

Secondly, this thesis contributes to Tangible User Interface Application Domain by developing an application that explores the interaction possibilities of the depth

tracking setup. The feasibility of this setup is demonstrated through the design and implementation of tCAD, a simple tabletop application for the manipulation of 3D virtual objects. tCAD is enriched with a number of novel CIS interaction techniques for common tasks such as issuing commands or adjusting parameters. This thesis also presents a discussion of the design (and brief formative evaluation) of these techniques.

1.3 THESIS STRUCTURE

The remainder of this dissertation is structures as follows:

- Comprehensive review of existent works correlated to the thesis domain and goals. Considering fiducial tracking and touch on tabletops (tangibility on tabletops) to be the merging of two broad areas of research, Tangible User Interfaces and Tabletop Technology, each topic is dealt with separately, before delving into how users behave on tabletops and how tabletops can be enhanced to better transform user behavior into interaction. Finally, considering the application domain of 3D Manipulation, related works exploring 3D Manipulation are looked upon, paying particular attention to the use of novel manipulation interactions.
- The incorporation of depth sensing into a tabletop setup is described, along with other relevant technical details for the replication in other tabletop setups.
- The design and implementation of TUI prototype tCAD is documented, through Unified Modeling Language (UML) and Tangible User Interface Modeling Language (TUIML).
- The interactions implemented in tCAD, taking advantage of the Continuous Interaction Space given by the physical setup, are described in more detail, as well, as a brief formative evaluation of these interaction techniques.
- Finally, the work is explored in terms of limitations and how it affects the research community.

2 STATE OF ART

2.1 TANGIBLE USER INTERFACES

Tangible User Interfaces (TUI) have emerged in recent years as a growing research field [URL1] and a valid alternative to post-Wimp (Windows, Icons, Menus and pointers) Graphical User Interfaces (GUI) [JGH+08]. Although not as popular as the latter, TUI development is rapidly growing as technology becomes easier to use and more affordable and as the field mixes with other fields (like Ubiquitous Computing and Augmented Computing), adopting concepts and integrating them into the field's framework [SH10]. Unlike GUIs, TUIs allows users to explore virtual data through physical objects in such a way, that they function as an interface, an interaction device and an interaction object [HB06]. Though TUIs offer advantages against traditional GUIs, they also present several restrictions that confine the user experience.

In the following subsections, a brief history of TUIs will be presented, followed by a subsection on model, frameworks and taxonomies important to comprehend the theoretical underpinnings of TUIs. Succeeding the more conceptual area of TUIs, a subsection on Application Domains will be presented, with previous implementations. Carrying the practical exploration of TUIs, a subsection on supporting technologies and a subsection on strengths and limitations of TUIs. Finally, a subsection on research directions will be presented in order to clarify future expectations for this field.

2.1.1 HISTORY

Beginning in the 1970's, several prototypes were created that explored the core concepts of Tangible Interaction and worked as precursors to TUIs. One of the first prototypes was Perlman's Slot Machine [P76], where physical cards representing language constructs (an action, number, variable or condition) were used to program the Logo Turtle. To write a program, a sequence of plastic cards is introduced into one of three racks. Stacking cards upon each other, created complex commands, while adding colored cards invoked a procedure call for a specific colored rack. A button next to the racks caused the turtle to execute the commands specified.

In another iconic prototype, the Marble Answering Machine [P95] (see Figure 3, left) developed in 1995, incoming calls are represented by color marbles that roll into a bowl entrenched in the machine. Placing a marble onto a specific indentation, playbacks the message left or calls the original number of the caller. This prototype introduced novel

ideas such as object mappings (assigning meaning to objects, turning them into pointers to something else, into containers of data and references to other objects) and spatial mappings (deriving meaning from the context of an action, for example, the indentation where a marble is placed)[SH10].

Lastly, Aish [A79, AN84] and Frazer's work [F95, FF80, FF82] concerning 3D Modeling used physical objects as input devices for Computer Aided Design (CAD) systems. A computer analyzed a group of physical objects (meant to be the construction), deduced location, orientation and types of digital component, in order to create a digital representation, offering suggestions and additional information like floor space and water piping, permitting the user to print out the plans once satisfied.

Although these prototypes contained key concepts, none of them actually presented themselves as a new form of interaction interface. In 1993, the Communication ACM issue "Back to the Real World" [WMG93] defended that desktop computers and virtual reality were counterintuitive to humans, distancing them from their natural environment. By promoting the idea of "humans are of and in the everyday world" [W93], systems should be oriented to retain the richness and situatedness of physical interaction, incorporating computing seamlessly into our environment and encouraging the use of human practices and actions, in order to blur the transition from the digital world to the real world. This concept is fundamental in the Tangible User Interface field as it accentuates the importance of physicality.

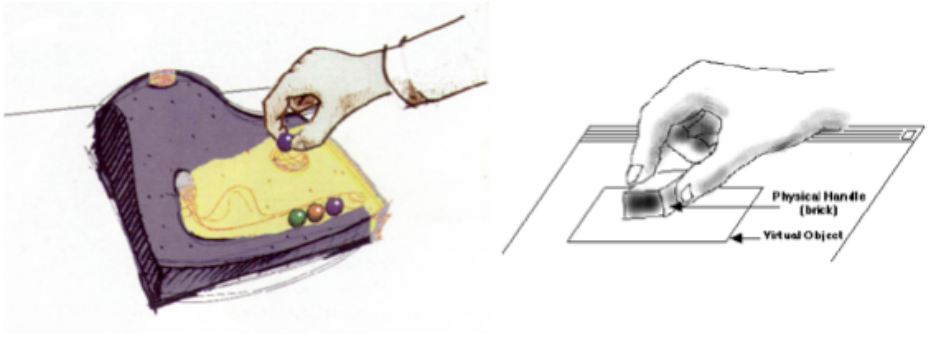


Figure 3 From left to right: Marble Answering Machine [P95], prototype where marbles representative of voice messages are played by placing them in indentations and Bricks [FIB95], prototype where wooden blocks (bricks) are used to manipulate a virtual object.

In 1995, Fitzmaurice et al [FIB95], gathered the ideas presented in "Back to the Real World" [WMG93] and introduced the notion of "Graspable Interface" (see Figure 3, right), using wooden blocks as graspable handles to manipulate digital objects (in this case, graphical objects). Rotating and moving a block anchored to a graphical object,

corresponded to a synchronous rotation or movement of the graphical object. Placing two blocks over a graphical object and moving them in opposite directions zoomed in the graphical object (an action that evolved into the present two-fingered interactions in multi-touch surfaces [SH10]). Graspable user interfaces were defined as having a "physical handle to a virtual function where the physical handle serves as a dedicated functional manipulator" [F96]. One of the properties of graspable user interfaces introduced is space multiplexing. If a system only has one input device, it is time multiplexed, as a user has to repeatedly select and deselect objects and functions [SH10]. By having several input devices, input and output are divided through space, permitting every object or function to have its time and space (allowing for selection by reaching for its physical handle) [SH10]. This separation of function allows simultaneous selection of objects (as objects are independent from each other) and permits tangible planning (a user may grab a few objects and plan their use as a reminder sequence). Space multiplexing is beneficial [FB97] as it reduces switching cost, exploits innate motor skills and hand-eye coordination. Other properties of graspable user interfaces described by Fitzmaurice were [SH10] concurrent access and manipulation, the use of strong-specific devices (usage of iconic and non-general objects), spatial awareness of the devices (the system should track objects in order to make the objects aware of each other and able to communicate) and spatial reconfigurability (users should be able to configure the space to achieve maximum task efficiency). The notion of Graspable User Interfaces differs from that of TUIs due to its focus on the physical as interaction devices, while the latter also includes physical as interface display [SH10].

In 1997, Ishii et al. [IU97] presented the notion of "Tangible Bits" and their vision of turning the physical world into an interface by involving objects and surfaces with digital data, making bits (data) directly accessible and manipulable, using the real world not only as a display, but also as a mean of manipulation (real world objects as tools for the manipulation of digital content). A working expansion of the tangible bits concept are Ambient Displays, which represent information through sound, light, air or water movement. As an example, Jeremijenko's work LiveWire [KHT06], where network traffic was presented as movement of a piece of string hanging from the ceiling.

One of the first TUI prototypes developed by Ishii and using the concepts presented in "Tangible Bits" paper [IU97], was Tangible Geospace [IU97] (see Figure 4, left), an interactive map with physical icons (unlike the normally generic blocks). Placing an phyicon (representing a building) on the table, would reposition the map to make the

building location visible. Adding additional icons would reposition the map to include several buildings, while movable monitors were used as magic lens/viewports to a 3D representation of the underlying area. Later prototype Urp [UI99] (see Figure 4, right), an urban planning application with tokens that represent data and served as manipulators, tried to distinguish themselves from the strong Graphic User Interface (GUI) influence that was present in the Tangible Geospace [IU97] prototype.

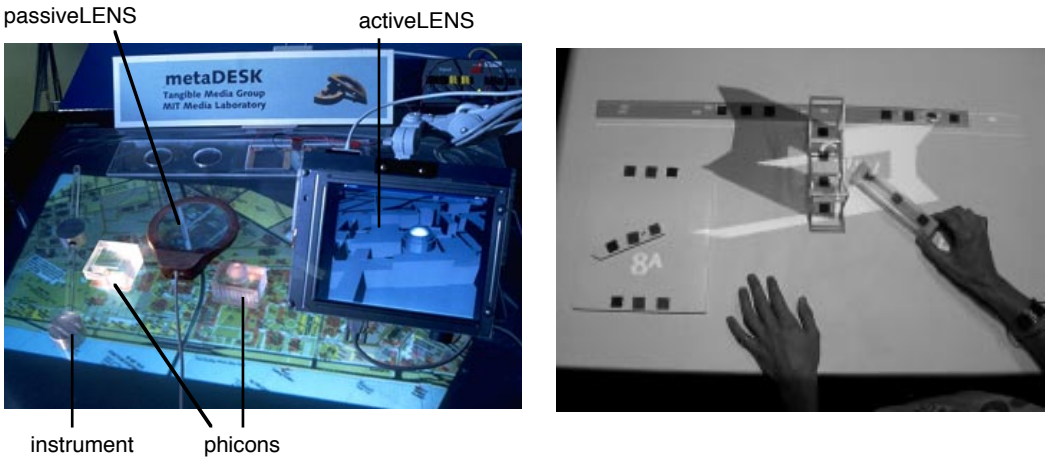


Figure 4 From left to right: Tangible Geospace on metaDesk [IU97] and Urp [UI99].

At around the same time as Ishii's work [IU97], other research teams concentrated on the development of domain specific applications that employed the core idea of physical interaction. Among these studies a few standouts are Wendy Mackay's work [MF99] on the use of flight strips in air traffic control and on augmented paper in video storyboarding and German Real Reality [SBB97] for the building of virtual/digital and real models. Grounded on the work of Fitzmaurice et al [FIB95], Build-IT [RFK+98] was an augmented reality tabletop planning tool that used graspable hands, utilizing input mechanisms and additional augmented reality visualizations in order to properly recreate architectural planning tasks. Other notable prototypes included AlgoBlock [SK93, SK95] (TUI supporting programming for children) and LogJam [CWP99] (TUI supporting video logging and coding). All these prototypes are relevant as they explore basic TUI concepts in diverging application domains, therefore confirming the usefulness and importance of TUIs.

All these studies and projects indicate that Tangible User Interfaces is emerging as a new interface and interaction style, quickly gaining popularity. After an initial period for proof of concept (researchers studied technical possibilities), research is now focused on more mature subjects like conceptual design, user and field tests, critical reflection, theory and identification of guidelines and rules towards the building of design

knowledge [SH10]. Technical advancements such as the development of toolkits, has lowered the development time for TUIs, and therefore increased research into the field [SH10].

2.1.2 Models, Frameworks and Taxonomies

As the field of Tangible User Interfaces evolves, researchers have developed several frameworks (conceptual structures for thinking through a problem or application) and taxonomies (type of frameworks that classify entities according to their properties) that help developers analyze and compare TUI instances and suggest new directions to explore [SH10].

Inspired by the Model View Control (MVC) model of traditional GUI, Ulmer and Ishii [UI00] proposed the MCRit model (see Figure 5), an interaction model distinguishable from the MVC model by the union of the graphical representation and control, eliminating the distinction between input and output devices. This means that tangible objects embody both the means of representation and the means of manipulating data.

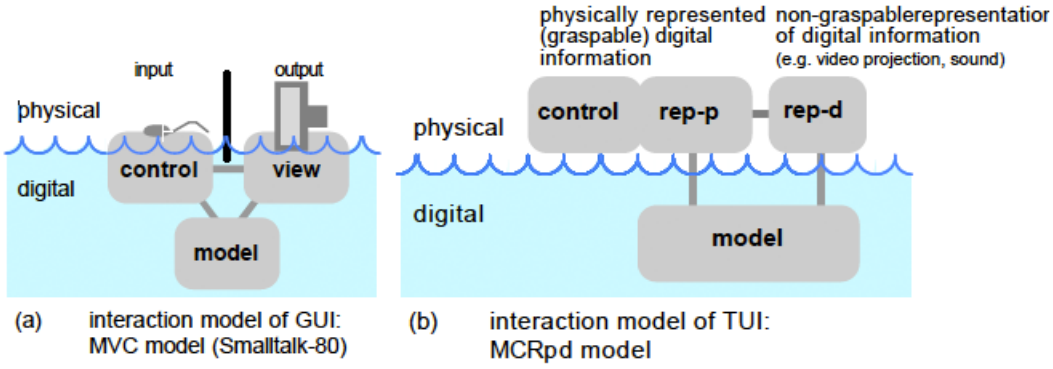


Figure 5 From left to right: GUI and TUI interaction models [UI00].

Ishii and Ulmer's work [UI00] accentuated the existence of core properties concerning representation and control of tangible objects. Tangible objects (the physical representation) are coupled via computerized functionality with digital data, embodying means of interactive control and representation (e.g. audio or visual). The state of tangible objects is representative of the system's state, which means that there should be a representational significance (even if the system is inactive, the system is understandable or the objects are partially functional).

Mappings are one of the crucial pillars of Tangible Interaction. Ulmer and Ishii [UI00] identified several types of digital information that could be associated to physical representations, namely static digital media (images and 3D models), dynamic digital

media (live video and dynamic graphics), digital attributes (color or material properties), computational operations and applications, simple data structures (lists or trees of media objects), complex data structures (combinations of data, operations and attributes), remote people, places and things [SH10]. Another noteworthy concept presented was static binding (specified by the system's designer and unchangeable) vs. dynamic binding (specified by the system's user) [SH10].

Alternatively, Holmquist et al. [HRL99] introduced a different taxonomy composed of containers (generic objects that are coupled to any type of digital information), tokens (physical objects similar to the information they are coupled to) and tools (manipulators of digital information).

Ullmer et al. [UIJ05], upon studying existent TUIs, created a classification taxonomy based on the physical model of the system which separates TUIs into three types (see Figure 6). The first type, Interactive Surfaces, concerns TUIs where tangible objects are placed on planar surfaces (having meaning arise from their organization and/or relations). Constructive Assembly systems, the second type, are composed of modular elements connectable between each other having meaning arise from their organization or order of placement. Finally, Token + Constraints systems are composed of two types of physical objects: tokens and constraints (structures that limit the position and movement of tokens, supplying haptic cues to users and expressing interaction syntax). Due to current complexity of TUIs, there are some TUIs that show characteristics from two or three of the previous styles (for example, having constraints in an interactive surface) [SH10].

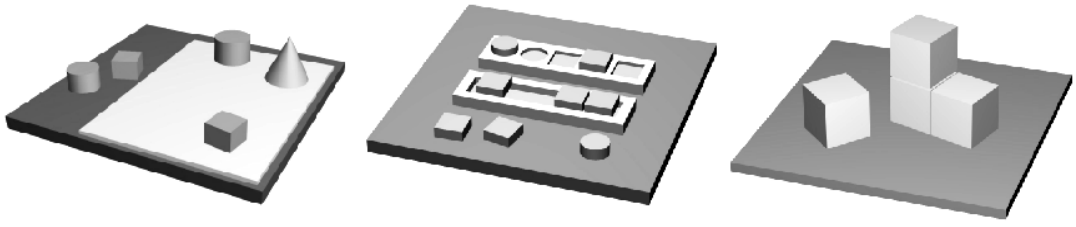


Figure 6 Illustrations of interactive surface, token+constraint and constructive assembly approaches [UIJ05].

As a curiosity, Sharlin et al. [SWK+04], Van der Hoven and Eggen [HE04] and Edge and Blackwell [EB09] have offered a different TUI classification based on the concept of spatial mapping, therefore differencing between spatial TUIs (take in to account shape, space and structure for its interactions) and non-spatial TUIs (tokens are not analyzed according to their spatial positioning or the relational positioning to each other).

Based on Ulmer's definition of Tokens + Constraints [UIJ05], the TAC paradigm [SLC+04] serves to describe the structure of a TUI as a group of relationships between physical representations and digital information, based on four core elements (pyfo, tokens, constraints and TAC). Pyfos are physical object that can be a token and/or constraint. While tokens are graspable object coupled to digital information or computational action, constraints are objects limiting behavior of a token. Limiting is achieved by suggesting (through the constraints physical properties) how to use a token, by restraining interaction space of a token or by serving as reference for the interpretation of a token. The last element concerns TACs, relationships between a token and one or more constraints.

Hornecker and Buur's [EB06] framework, unlike other sensor based interactions like Belloti et al. [BBE+02], Benford et al. [BSK+05] and Rogers and Muller [RM06], is focused on user behavior and experience, rather than on the visible interface. This framework accentuates concepts like haptic direct manipulation (material qualities and manipulability of an interface; how does a interface feel and work?), spatial interaction (spatial qualities like whole-body interaction), embodied facilitation (how physical and digital properties affects user patterns of use?) and expressive representation (expressiveness and representational function of the interface; is the interface understandable and consistent to its function?). This framework is particularly useful when designing TUIs as most frameworks and taxonomies focuses on the physical aspect of the system and not on how the user interacts with the physical aspect.

2.1.3 Application Domains

After reviewing theoretical concepts applied in TUIs, the following section concerns itself with more practical aspects of TUI development, by analyzing TUI prototypes in several application domains (learning, problem solving and planning, programming, music, entertainment, social communication and reminders). Having TUIs in a wide scope of domains not only serves to underpin TUI as a legitimate field of study, but also is helpful to determine how TUIs can be improved to more accurately reflect real life activities.

TUIs aimed at learning have a large representation on TUI research studies due to several reasons. Psychological studies [OF04] have show that concepts entrenched in Tangible User Interfaces, such as physical movement and embodiment, are crucial for learning. Since the target audience of a TUI aimed at learning is formed by beginners, applications can be simplified, surpassing the usual cumbersome problems like screen

estate or counterintuitive mappings, therefore making them appropriate for quick TUI research.

One approach to making TUIs for learning focuses on modifying existing toys, since these already present a physical shape and most users already have a mental model of how they work. MIT's Digital Manipulatives [RMB+98] are computationally enhanced versions of traditional children's toys like blocks, beads, balls and badges, aimed at exploring specific complex system concepts like emergence and feedback. Another approach method focuses on creating construction kits marketed as toys, like Lego's MindStorms [URL3] and Pico Cricket [URL4] (kit composed of sensors, actuators and robotic parts, aimed at the creation of scientific experiences for young children).

Constructive Assembly systems are a common sight in TUIs aimed at learning since these perfectly explore organization and placement of elements, a process commonly found in learning. Smart Blocks [GSH+07] is composed of augmented manipulatives for exploration of mathematical concepts like volume and area through the mix and match of blocks. Other examples of constructive assembly systems include Curlybot [FSM+00] (robotic ball capable of recording movement and replaying the recorded movement) and Topobo [RPI04] (connectible moveable parts capable of recording individual movements between joints and evolving them into more complex movements; see Figure 7, left).

Figure 7 TUIs aimed at learning, from left to right: Topobo [RPI04], a Constructive Assembly system for understanding movement and Storymat [RC99], a play carpet using RFID-tagged objects.

TUIs aimed at learning can be developed to cater to different age brackets, making them challenging enough to motivate learning. For example, Storymat [RC99] (see Figure 7, right) is a TUI aimed at children learning to read, composed of play carpet capable of recording and playing children's stories according to RFID-tagged objects present. This TUI does not offer interaction elements that would be challenging to its indented age group. WebKit [STR+04] is a system aimed at teaching rhetoric to children

through the use of cards ordered in the sequence of their argument. Since the target audience is older, a more complex interaction has been adopted to make the system engaging.

In a similar fashion to TUIs for learning, Embodied Interaction offers some benefits (such as epistemic actions, physical constraints and physical representations) advantageous to the process of problem solving and planning [SH10].

One of most commonly cited systems is Urp [UI99], a luminous-tangible workbench for urban planning and design, used to determine the effects (shadows, wind, reflections and distance) of a site configuration onto the surrounding environment. Since this simulation is done on the fly, the system is very useful as the user is capable of changing the configuration of the site and receiving immediate feedback (not achievable by the traditional way of building physical models). Physical objects (clocks, rulers, etc.) are used to change parameters and present additional information. Similarly, MouseHaus Table [H04] is an urban planning and design environment for the simulation of pedestrian behavior, by the addition of objects as obstacles, while Sandscape [I08] was designed for the study of landscapes, through the manipulation of real sand and the projection of information on the work area. Interfaces concerned with urban planning and design such as the ones discussed are especially beneficial for problem solving and planning, as they allow the user to immerse himself into environments that in the real world are hard to understand due to their scale.

TUIs aimed at problem solving and planning can be extended to several fields as proved by Mementos [EO10] and EcoPlanner [EO11]. The former is a TUI aimed at tourism and travel, which offers *in situ* planning (by offering collaborative plan creation and revision). Through the mix and match of tokens, the user can create simple queries to determine destinations. The latter is a TUI intended for energy conservation at home, as it works as a tabletop system for scheduling sustainable actions (creation and visualization of daily routines).

Another notable system is Physical Interaction in Computational Organization (PICO) [PI07] where the TAC paradigm can be clearly observed. PICO is an interactive surface that supports solving complex spatial layout problems through improvisation, allowing physical objects as mechanical constraints in the problem space. The use of rubber bands as physical constraints limits the physical affordance of objects and therefore, limits the solution space.

TUIs, especially TUIs that use tabletops or interactive surfaces, offer certain properties that make them an excellent way to visualize information, namely two handed input and clearly visible interactions appropriate for collaborating environments [SH10].

Props-based Interface for 3D Neurosurgical Visualization [HPG+94] (see Figure 8, left) is a perfect example of a TUI aimed at information visualization, as it allows for the exploration of a 3D MRI scan of a patient's brain in the pre-surgery phase. A doll head is used as a tangible representation of the scans, allowing the surgeon to use a prop as a plan cutter and a prop as trajectory selector. The use of two-handed physical props in free space is easily understandable as they are part of the day to day of a surgeon, therefore diminishing training time.

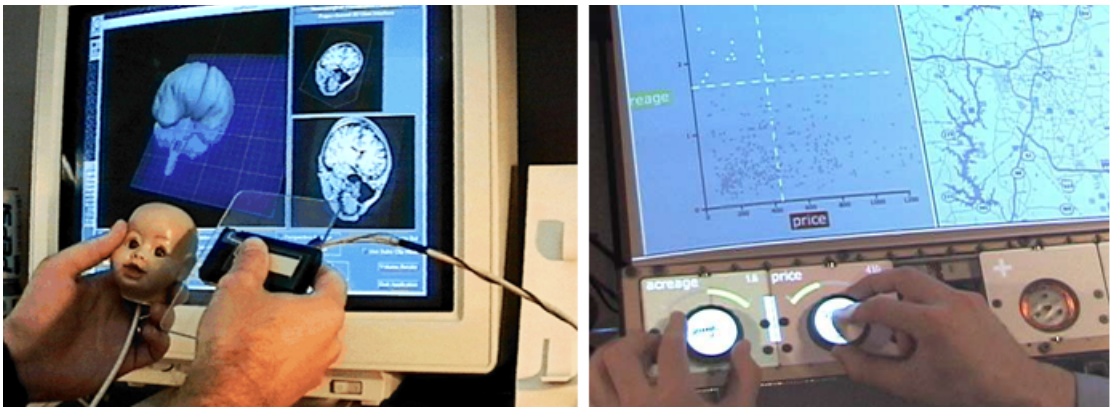


Figure 8 TUIs aimed at information visualization, from left to right: Props-based Interface for 3D Neurosurgical Visualization [HPG+94], TUI for visualization of 3D MRI scan using objects and Tangible query interfaces [UIJ05], TUI for database exploration using parameter wheels.

In an alternative field, GeoTUI [CRR08] is a TUI for the exploration of terrain by geophysicists. A geographical map projected upon a surface is subjected to a ruler prop for the selection of planes. In a more common and somewhat unimaginative implementation, Tangible query interfaces [UIJ05] (see Figure 8, right) is a TUI aimed at the exploration of a database through the construction of simple queries as manipulation of parameter wheels.

The concept of tangible programming has been around since the previously mentioned Perlman's Slot machine [P76] but was only coined in 1993 with the Algoblocks system [SK93, SK95]. Algoblocks [SK93, SK95] used big blocks as constructs for the language Logo. With the goal of guiding a submarine through a maze, blocks were attached to create complex constructs and during execution, the child could observe runtime behavior through the block's flashing LEDs.

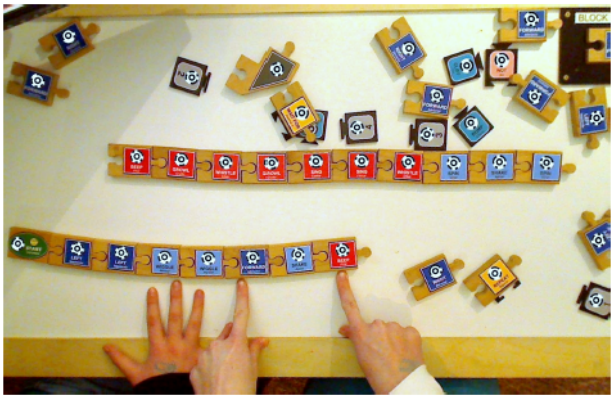


Figure 9 Tern [HSJ08], TUI for programming where blocks are attached to each other to construct commands.

Another example of TUIs aimed at tangible programming is Tern [HSJ08] (see Figure 9) where physical constraints were used to guarantee correct language syntax. Tern is composed of blocks, either representing commands or variables. The shape of the block limited the type and the number of blocks it could attach to, in order to form complex constructs, serving as example of a pyfo that works as both a token and a constraint.

TUIs are fairly popular in the entertainment field as they posses a novelty factor which makes it attractive to people. The best example of a TUI in this field is the Wii [URL5], which has grown to be the most popular gaming system and proves the economic validity of TUIs.

TUIs are a common presence in museum exhibitions as they attract the audience to interact with the work. Such examples of this are the tangible manipulation of DNA strands in Glasgow Science Museum [SH10] and Todd Machover's Brain Opera Installation [SH10]. TUIs can also be used as museum guides, providing information about exhibitions, as found in Correia et al.'s work [CMN+10].

One approach to TUIs in entertainment is the enhancement of existing toys, in a similar fashion to TUIs for education. The EnterTaible [LBB+07] (see Figure 10, left) is a tabletop gaming platform that augments gameplay, not sacrificing social relations. Leitner et al.'s Comino [LHY+08] is a tabletop game that combines virtual and real objects. Similarly, Wilson [W07] utilized a landscape created by the user as a racetrack for a virtual projected object. Hinkse et al. [HLL08] continued this theme of augmenting toys by adding audio cues to a Playmobile Castle while Oasis [ZGL+11] projects animations on the Lego structures that are being played with.

Figure 10 TUIs in entertainment, from left to right: EnteTaible [LBB+07], augmented game board and Tangible Video Editor [ZHS+07], TUI for video editing.

Another approach to TUIs in entertainment relates to the capture and editing of audio and/or video clips. For example, Tangible Video Editor [ZHS+07] (see Figure 10, right) is a TUI for video editing using tokens that can be mix and matched to achieve the desired sequence. Another notable TUI is I/O Brush [STR+04], a digital drawing tool that captures moving images and allows the child to draw with them.

Musical TUIs are a very common research topic as TUIs focuses on collaboration, real time interaction with real time feedback and complex and expressive interactions [SH10], which are all needed in music. TUIs aimed at music fall generally into one of three groups: instruments, sequencers and sound toys [SH10].

Instruments are fully controllable sound generators and synthesizers being the reacTable [JGA+07, J08] (see Figure 11, left) the most popular example. In the reacTable, physical tokens have a specific function and sound is created through the relation between tokens. Similar systems include AudioCubes [SJ08] and Block Jam [NNG03], both based on the premise of cubes relating to each other in order to create sound.

Sequencers are interfaces for the mixing and playing of audio samples [SH10], in a similar fashion to editors TUIs like Tangible Video Editor [ZHS+07]. The most symbolic examples of this type of interface are the Audiopad system [RI02] (see Figure 11, right) and mixiTUI [PH09].

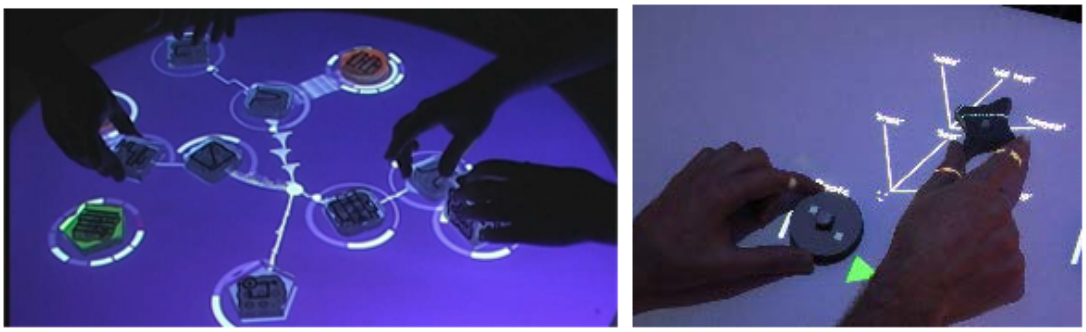


Figure 11 Musical TUIs, from left to right: reacTable [JGA+07, J08] and Audiopad [RI02]. In both TUIs, tokens are used to manipulate sound.

In similar fashion to augmented toys for education and entertainment, toys can also be augmented for music production through its manipulation [SH10]. The most popular example of this endeavor is Squeezables [WG01], physical jello objects with force sensing resistors in the inside to detect squeezing.

One of the most commons uses of TUIs is in ambient awareness, to communicate information in a non-invasive way [SH10]. Somewire [SHS+99] uses tangible figurines to control auditability and direction of sound while in Peek-a-boo surrogate [GK99], a peek-a-boo surrogate (figurine) is moved according to the presence of the person (facing away if not present; facing the cameras if present). In former, the figurine is used as a control, while in the latter, is used as a display. Differencing himself from the Peek-a-boo surrogate [GK99], abstract forms can be used instead of iconic objects as in Light surrogate [EB09] where moving light patterns are displayed according to people's activities.

Other TUIs tackle the issue of intimacy by allowing objects to communicate between each other in response to an action. Examples of this type of TUIs include LumiTouch [CRK01] (two pictures frames communicate when one of them is pressed), Lovers Cup [CLS06] (two glasses communicate when one of them is used), InTouch [BD97] (interconnected rollers capable of transmitting movement) and UnitedPulse [WWH08] (transmission of a partner's pulse between two rings).

TUIs are often used as reminders by having tangible objects act as containers for memory. WebStickers [HRL99] is a system where tangible objects are coupled with webpages. Digital Photo Browser [HE04], Memodules [MRG07] and Mementos [EO10] are all TUIs that incorporate physical souvenirs coupled to digital information (photos).

From the above extensive list, is clearly visible the large scope of Application Domains where TUIs have left their mark.

2.1.4 SUPPORTING TECHNOLOGIES

One of the main factors limiting the growth of the TUI field is existent technology. Whether due to its price, or its limited capacities, technology is the main barrier to accurately portraying human physical actions in TUIs. In this section, several common technologies in TUIs are presented as well as some examples of their usage.

Firstly, Radio Frequency Identification (RFID) [SH10] allows for the recognition of presence and identity of tagged objects within a certain range (according to a tag reader, an antenna). According to its nature of charge, RFID tags can either be active tags (possess a battery and can transmit a signal by itself) or passive tags (do not possess a battery and therefore are dependent on external sources for the transmission of a signal). RFID tags contain a transponder (integrated circuit) for processing and storing information and an antenna for receiving and transmitting a signal. StoryMat [RC99] is an example of a TUI that uses this type of technology to determine presence of objects.

As a common supporting technology, Computer Vision is capable of detecting identity, presence, shape, color, orientation, position, relative position and sequence of objects [SH10]. Computer vision systems are based either on Tags or in Artificial intelligence [SH10].

Systems based on Tags rely on fiducial markers attached to physical objects. Tracking fiducial markers offers information about marker position, identity and orientation in a way that is robust, accurate and cheaper than artificial intelligence systems. Examples of this type of systems include Urp [UI99], Tern [HSJ08] and the reacTable [JGA+07, J08]. For the support in the development of computer vision TUIs, several libraries/toolsets have been created and distributed. For example, ARToolkit [KBI+00], reacTIVision [KB07] and Top Codes [HSJ08] are libraries that support the tracking of fiducial markers.

Systems based on Artificial Intelligence rely on complex algorithms to interpret an image provided by one or more cameras in the system. Examples of this type of systems includes the MouseHaus Table [H04], ColorTable [MPW08] and Designers Outpost [KNF+01]. This type of systems is more prone to error due to inherent complexity of Artificial Intelligence algorithms. The most popular library for computer vision is OpenCV [URL6], which serves as the backbone for several applications.

Lastly, microcontrollers are small computers that act as a gateway for information about the physical world (through sensors), acting upon the physical world by sending information to actuators [SH10]. Sensors are capable of detecting physical properties like light intensity, reflection, noise, level, motion, acceleration, location, proximity, position, touch, altitude, direction, temperature, gas concentration and radiation. Actuators affect the physical world by altering light, sound, motion or haptic feedback. Several TUIs take advantage of microcontrollers, sensors and actuators including the previously mentioned PICO [PI07] (uses actuation to move objects across surface) and Topobo [RPI04] (motorized pieces that can be programmed to perform a movement).

2.1.5 Strengths and Limitations of TUIs

In order to understand how to design TUIs, it's crucial to understand its strengths (to take advantage of) and limitations (to avoid or overcome).

One of the main strengths of TUI is the ability to promote collaboration between users. The use of affordances common in our daily life lowers the difficulty level of user interfaces, promoting discourse and collaboration between users [SH10]. The ability to do simultaneous actions, letting these be observable to others, increases collaboration as it engages all users in a common task [SH10]. Interfaces that support embodied facilitation can guide and restrict user actions to benefit collaboration [SH10]. For example, the circular shape of Jordà's reacTable [JGA+07, J08] creates an environment where all users have an equal right to collaborate. The usage of tangible objects that can be passed among users, nurtures discussion and planning according to the context of the interface [EO10].

Considering the concept of Ubiquitous Computing [W93] and Dourish's work [D01], Situatedness can be identified as another strength of TUIs, as they allow for the meaning of an interface to change according to its context, creating appealing and enticing interfaces [SH10].

The use of body and mind in TUIs emphasizes Tangible Thinking [SH10] as a crucial strength, especially in TUIs aimed at learning, problem solving and planning. Tangible Thinking [SH10] defends that thinking can be achieved by bodily actions (gestures), physical manipulation (epistemic actions) and tangible representations.

The first aspect of Tangible Thinking is gestures [KHT06, SH10]. Gestures are not only a method of communication, but they also reduce cognitive load and help plan sequences. Some TUIs use gesture as input (through symbolic gestures or real life

actions) as they promote the use of kinesthetic memory (ability to sense, store and recall muscle effort, body position and movement) to harness skill [S00]. TUIs that rely on real life actions may also use Experiential Cognition, developed over years but immediate on to the TUI system [SH10].

The second aspect of Tangible Thinking is epistemic actions [SH10]. Unlike pragmatic actions (user manipulates a object to accomplish a task), epistemic actions (user manipulates a object to understand context's task) foster mental work, as they are not oriented in terms of functionality, but rather in changing mental tasks [K09, KHT06]. Epistemic actions (in the vein of pointing at objects, changing arrangements, turning them, occluding them, annotating and counting) reduce cognitive load by relying on external resources [SH10].

The final aspect of Tangible Thinking is tangible representation [SH10]. The immersion into the tangible representation enhances reality-based interactions. In other words, the user interacts with digital information, like he would interact with the real world. The reality-based interactions are based on four concepts [JGH+08]:

- Naïve Physics – common sense knowledge about physical world.
- Body awareness and skills – the knowledge about your body and how to control and coordinate them to achieve a task.
- Environment awareness and skills – the knowledge of their surroundings and how to manipulate and navigate through it.
- Social awareness and skills – the knowledge about people in their environment and the skills to communicate (verbally and non verbally) and to work together for a common goal.

As mentioned previously, one of key advantages of Tangible User Interfaces is spatial multiplexing as different objects represent different functionalities or data, increasing system functionality and reducing complexity of interaction [SH10]. Unlike traditional GUIs, where a mouse click is dependent on the object or function called, the use of a specific object to a specific function or data creates a strong persistent mapping [SH10], taking advantage of spatial memory to speed up task [F96]. Having multiple objects not only helps to speed up the task but also supports eye-free fashion [SH10]. Space multiplexing also allows for tokens to be strong specific and iconic [F96].

While its beneficial to understand the strengths of TUI when designing them, it is also crucial to properly understand its limitations and how to solve them or avoid them to achieve a TUI true to its conceptual essence.

One of the most blatant limitations regards scalability [SH10], namely the representation in applications with large data sets and parameters. Simple applications require few lines of code but industrial projects would be in such a scale that they would seem unfeasible [SH10].

Another issue that TUIs face is that of physical clutter [F96], or as its called in traditional GUI, screen estate. To avoid complex setups with clutter, one would have to increase surface size, forcing the users to walk around it, limiting collaboration and inconveniencing reach for objects. The size of tangible objects is also problematic because seeing that size is static, if one wants to work on two projects, one of the projects must be cleared away [SH10].

Another limitations of TUI include the expressiveness of tokens when issuing complex commands (compared to command line interfaces), the inability to manipulate several objects at the same time and difficulty to implement automatic undo, history function and replay of actions [KST+09]. Favoring physical objects to digital ones brings up the question of versatility and malleability, since physical objects are rigid, static, iconic and specialized unlike digital objects that are malleable, modifiable and easy to create [SH10].

One final limitation of TUIs is relative to its application domain. Most TUIs are limited to a restricted group of application domains and too specialized in a specific set of tasks [SH10]. Designers should aspire to create interface that are reality-based, only sacrificing realism for additional value for the user [SH10].

2.1.6 Research Directions

In this section, several popular research directions in TUIs will be explored: actuation, organic user interfaces, whole body interaction and performative tangible interaction.

Although a concept introduced in the vision for TUIs, actuation has only of late trended as topic of research because of the inherent technical difficulties of introducing actuation to user interfaces [SH10]. Systems with tangibles that push back help the system engage the user more efficiently by making the objects less rigid and lifeless. Previously discussed systems that introduce actuation include the Peek-a-boo surrogate [GK99] and Pico [PI07] (objects on a surface are moved by magnetic forces). Other

systems include Navigational Blocks [CDJ+02] (blocks are attracted and repelled due to electromagnets) and Lumen [PNO07] (13x13 pixel bit map display where each pixel moves up and down individually).

The incorporation of relief into TUIs is a recent research focus in the community as it gives way to a novel way to experience, create and manipulate objects. Ishii et al.'s Recompose [LLD+11, BDL+11] is an actuated surface (a 2.5D Shape Display) capable of interpreting gestures above the display into an approximated object shape.

Another of trends found in TUI research has been Organic User Interfaces, where rigid and discrete objects have been replaced by organic malleable materials [SH10]. One popular material explored is paper in such works as PaperWindows [HVT05] (windows are projected into physical paper, which is then used as an input device and tracked by the Vicom Motion Capturing System) and Paper Devices [SG09] (interactive paper objects). Another popular material choice is the human body as found in HoloWall [MR09] (user can interact with a computerized wall using fingers, hands, their body or inanimate objects).

The concept of whole body interaction although present in the original vision for TUIs, is ignored by most TUIs as they require only object manipulations with upper members [SH10], to simplify the system's physical setup and facilitate development. Whole body interaction happens when the user's body is used as an input device and has become a research trend due to the growth in technologies that support whole body tracking (e.g. Wii [URL5], Kinect [URL2], OpenCV [URL6]). Projects that explored this concept include Squeeze [P07] (multi-person interactive furniture whose movement or pressure on specific areas triggers interaction with photos projected on a wall) and SignalPlay [KWD05] (interactive sound experience where large objects can be moved to change the sound experience).

Lastly, Performance is in general improved by TUIs, when compared to GUIs, because the manipulation of tangible objects is publicly visible and legible to the audience [SH10]. TUIs aimed at music performance such as reacTable [JGA+07, J08], mediacrate [BHO09] and mixiTUI [PH09], all prove that TUIs are very successful in the performance field. TUIs can also be used to actively engage the public in the performance as accomplished by iPoi [SB08], aimed at interactive performances in clubs and music festivals with balls hung by strings that could be manipulated to create light and sound patterns. TUIs can also be employed in untraditional performance fields as it was

achieved with SandCanvas [KCZ11], a sand painting TUI for visual storytelling through the pouring and manipulation of digital sand in an illuminated surface.

2.2 Interactive Tabletops and Surfaces

The field of interactive tabletops and surfaces is a rapidly emerging field, as technology becomes more affordable and consistent and interaction paradigms become more explicit [MF12], bringing tabletops and surfaces from research labs onto public spaces (from small tabletops up to large walls). The following subsections will focus on a general history of tabletops, taxonomy for the classification of tabletops and the application of said taxonomy onto several tabletops, notable through their technical contribution to the field.

2.2.1 History

Tabletops are horizontal displays working as interfaces where users interact directly with digital information, rather than by using a keyboard or mouse [MF10]. The concept of physicality still remains in tabletops as users utilize a mental model of traditional tables [MF10].

The origin of direct touch devices can be traced back to 1971 and Hurst's invention of the Touch Screen [S08], the first computer with a transparent surface sensitive to touch sense. However, only in 1990, large interactive wall displays were created thanks to more reliable touch detection [MF10]. Technical developments and popularization of such technologies like data projectors in late 1990's [MF10], Plasma Display Panels (PDP) in 1999 [MF10], Liquid Crystal Display (LCD) [GO07] and Organic Light Emitting Diodes (OLEDs) [GO07] have changed the landscape of tabletops, introducing several viable and cost-effective alternatives. The increased research in tabletops lead to the eventual commercialization of tabletops, therefore confirming the existence of a rich market for this type of technology [MF10].

2.2.2 Tabletop Classification

According to Kunz and Fjeld [KF10], tabletops can be classified according to their interaction style, their tracking technology and their display properties (see Figure 12). In regard to the latter, their classification differentiates between front projection (user and image source are on the same side of the interaction plane) and back projection (user and image source on opposite sides of the interaction plane).

Although tabletops are displays as well as direct input devices, most tabletops offer additional ways on interacting with the system as to reduce cognitive load of the user-

system interaction [KF10]. In relation to interaction style, their classification focuses on differencing between hand-based (pointing or complex gestures) and device-based systems.

Lastly, their classification according to tracking and detection is debatably the most interesting, as it is the best at showing the variety of tabletops that can be built. Several examples of this classification will be presented in the next section.

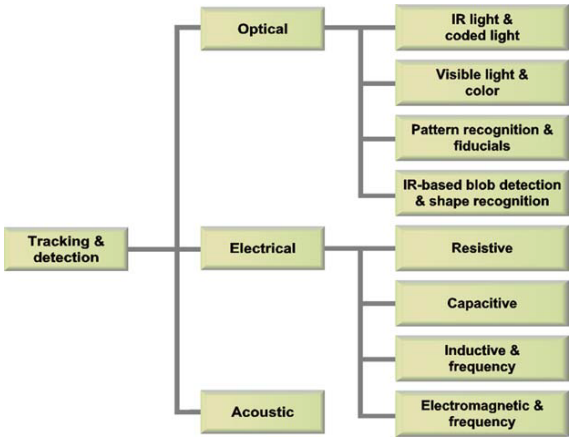


Figure 12 Classification of Tabletops according to tracking and detection [KF10].

2.2.3 NOTABLE TABLETOPS

One of the first and most significant tabletops was DigitalDesk [W91], a system that allowed the manipulation of digital (projected) paper with a bare finger, using a front-projection and front-capture to detect the user's finger and hand. Responsible for the introduction of term "Graspable User Interface" (physical representation of the graphical user interface), ActiveDesk [FIB95] was back projected system where other interaction devices (other than fingers) were used.

With the goal to expand the insufficient interaction ability of ActiveDesk and the non-intuitive nature of objects, metaDesk [UI97] introduced the concept of a TUI, having intuitive objects from the users daily life. MetaDesk used infrared (IR) light (back projected with visible image), tracking the infrared light reflected by the objects with a camera above the tabletop. Like the metaDesk system, BUILD-IT System [RFK+98] relied on reflective IR tracking system but since all the components were located above the table, a user could comfortably use it sitting down.

Up to this point, tracking was done externally to the tabletop; with the creation of SenseTable [PIH+01], a front projected system where pucks (tokens) are associated to virtual objects, this changed as two WACOM Intuos tablets were integrated to the

tabletop as a tracking system. Similar to SenseTable, DiamondTouch [DL01] had a front-projected and integrated tracking system in the tabletop. However, the tracking system is electronic and triggered by a current passed from the user's chair to the user and then finally to the tabletop. This allows for differentiation of touch between users and resting objects in the tabletop without triggering interactions (since the objects are not transferring current). Like DiamondTouch, SmartSkin [J02] relied on capacitive sensing as an integrated tracking system. However, SmartSkin analyzes position of touch blobs to detect gestures like grasping and zooming. Capacitive tags attached to objects, allows for the detection of these objects when a user is touching them (and identification of the object through its shape), making this system one of the first systems to integrate both TUI and touch into a surface.

Frustated Total Internal Reflection (FTIR) [H05] was a technology introduced in 2005. Similarly to the reacTable [JGA+07, J08] (back projected system where all the components are under the table and capable of detecting objects through tracking of fiducial markers and touch through reflected infrared light), FTIR had all components located under the table but, unlike the previous, has an additional acrylic overlay, where the light (introduced by peripheral LEDs) is internally reflected until a touch causes the reflection of the light out of the acrylic and into a camera with an IR sensitive filter.

With the commercialization of LCD displays, systems like MightyTrace [HKK08] began to appear. In this system, a synchronization flash triggers devices into emitting infrared light onto the tabletop. Underneath a LC-screen, a set of IR sensors detects the emitted infrared light (barely influenced by the content shown on the LC-screen). This system has a disadvantage of only allowing augmented devices like a stylus or the interaction brick. This hurdle was overcome with FLATIR [HNK09], an adaptation of the FTIR technology to a LC-screen.

In order to avoid complex pattern recognition techniques or augmentation of objects with electronics or fiducial markers to detect presence and manipulation of objects, SurfaceFusion [OW08] joins computer vision (for detection of shape and movement) with RFID tags (to avoid problematic and computationally expensive computer vision algorithms, by incorporating data from RFID tags to recover position, shape and identification).

The previous list of tabletop technology accentuates that over time, technology has been geared to improve tracking quality of touches and objects, a direction that is expected to remain constant in the future.

2.3 USER BEHAVIOR ON TABLETOPS

Several studies have been carried out focusing on user behavior on tabletops in order to establish guidelines and design trade-offs. These studies permits to observe aspects where tabletops excel at (such as collaboration) and help determine how that level of excellence is achieved. In the following subsections, several studies will be presented that show how users interact with tabletops and among themselves.

2.3.1 INDIVIDUAL, GROUP BEHAVIOR AND COLLABORATION

One of the areas that tabletops naturally excels at is collaboration; however, choosing a certain interaction technique can drastically change the results if the interaction technique is more appropriate for individual work rather than work group. Nacenta et al. [NPG+10] identified several criteria for the analysis of interaction techniques for individual work and group work. Regarding individual work, the main criteria are performance (speed and accuracy at which a user accomplishes a task) and power (type of actions that the technique makes possible). Regarding group criteria, the main criteria are group awareness (how people perceive and understand others actions), interference (actions that interfere in other user's performance) and space use (spatial patterns of use). An additional criterion relates to user preference, while other studies [MHM+08] count equality of participation as a valid criterion.

Nacenta et al. [NPS+01] studied the relation between type of interaction and coordination in a group setting by comparing 5 interaction techniques in 2 settings that required collaboration (and not competiveness). This study concluded that having input and output in the same location (direct interaction) is beneficial for performance while indirect interaction is too cumbersome but beneficial if the user wants to interact with objects that are too distant. Although direct interaction makes the interaction aware to other users, if it is used primarily in the local user space, other users lose awareness and more conflicts arise, diminishing amount of collaboration (because the user is less focused in the group setting). Also, direct input techniques are limited by reach [NPG+10]. Another study by Nacenta et al. [PNG+08] focused on users embodiment (physical or virtual) in order to determine awareness of other users. Using physical embodiments provided better awareness than virtual ones because are not limited to the 2D plane of virtual embodiments, becoming more visible to users. Lastly, Nacenta et al. [PBN+09] also studied control policies by assigning control levels for each user, in order to determine possession and stealing behaviors of objects from users with higher control levels; either the computer or the location of the object determined the control

level. The main conclusion that this study arrived at is that users are willing to restrict access to the table in order to protect objects that they are using.

2.3.2 TERRITORIALITY AND REACH

Another common approach to studying user behavior on tabletops concerns studying territoriality in order to achieve task and group facilitation [SC10]. Tang [T91] in his study on collaborative tabletop design, noticed that workspace partitioning is a crucial element to tabletop activity, as did Kruger et al. [KCS+03]. Both studies implied that orientation and proximity are used to establish personal and group spaces in collaboration environments. Scott and Carpendale [SC10] evidenced that users tend to create three types of territories in tabletops: personal, group and storage. The first type is an area that the user creates for individual activities (dependent or independent from the main group task). The personal areas can also be surveyed by other users to monitor activity and items or detect situations where users can help with individual tasks. Group territories are areas that are common to all users and are concerned with the group task. This area is normally located in the center of the table, extending into the areas between users. Lastly, storage territories are areas for storing resources relevant to the task.

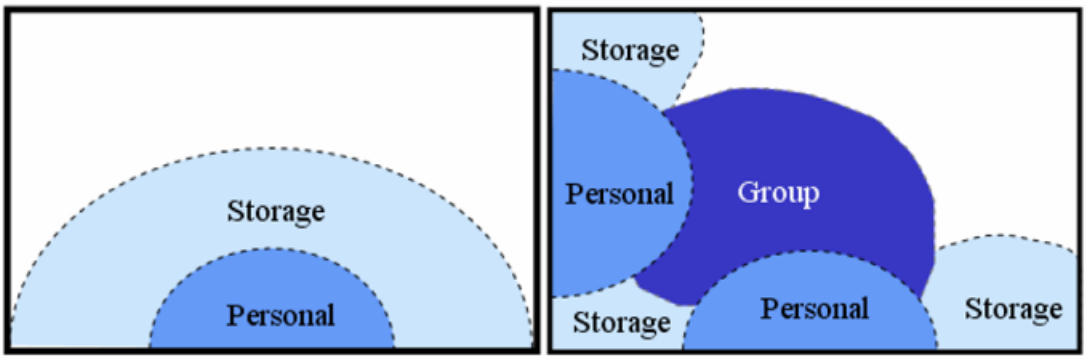


Figure 13 From left to right: Tabletop territories for individual users and groups [TT06B].

Based on Scott and Carpendale [SC10] division of territories, Toney e Thomas [TT06, TT06B] (see Figure 13) studied reach (maximum extension of the arm at which the user can perform a manipulation) as a formal way to predict workspace segmentation. In individual scenarios, reach allows for the observation of the storage area and personal, since no group area is required. In group settings, regions of overlapping reach between users are group areas. Using Zone of Convenient Reach (ZCR), reach conditions can be studied according to table size, shape and number of users, in order to have areas that are easily accessible to several users.

2.4 Enhancement of Tabletop Interactions

The main focus of research of interactive tabletops and surfaces has revolved around the enhancement of interactions. The following subsections emphasis four types of improvements of user interactions. The first type is centered on physical improvement by adding tangible controls. The second type of improvement revolves around improving current touch interactions in order to make them more natural to users. The third type of improvement concerns interactions above the tabletop, breaking the 2D interaction barrier into 3D interaction. Finally, the fourth type concerns the exploration of Ubiquitous Computing's concepts on tabletops.

2.4.1 Tangible Controls

In multi-touch surfaces, objects can be manipulated by directly touching and dragging them, allowing interaction between several hands and several users (collaboration), not depending on a keyboard or mouse. Tabletops are effective in the presentation of information between several users, moving away from desktop computers into systems that conceal technology and accentuate direct natural interaction [WHB10].

As tabletops technology becomes more widespread and users become more accustomed to interaction with touch sensing technologies (like smartphones), the transfer of traditional desktop applications to tabletops becomes a main area of concern [WHB10]. This transfer is problematic as traditional desktop applications and operating systems are not designed with multiple touch input or gestural interaction in mind. Virtual controls don't offer haptic feedback and need visual attention during usage. The usage of controls without visual attention leads to errors due to lack of control borders or indication of state [WHB10].

TUIs offer interface components with natural haptic feedback during interaction. However, these components are normally specific to the domain of the application and not general enough for usage in a wider range of applications [WHB10]. Therefore, the incorporation of traditional and conventional interface components like buttons, sliders and keyboards into tabletop applications is an interesting area of research as these components offer physical affordances, guide user action, enable tactile feedback and can be used without visual attention [WHB10]. Unlike their virtual counterparts, they are expensive, have a fixed visual experience and clutter tabletops [WHB10].

Figure 14 From left to right: DataTiles [RUI01] and SLAP Widgets [WWJ+09]. Both TUIs promote the use of tangible controls in tabletops environment.

Among TUIs that explore the concept of tangible controls: VoodooSketch [BHG+08], DataTiles [RUI01] (see Figure 14, left) and SLAP widgets [WWJ+09] (see Figure 14, right). VoodooSketch [BHG+08] is a system with customizable interactive palettes where users can plug real physical controls while DataTiles [RUI01] are acrylic tiles that are placed on a tray, detected by RFID technology and back projected with information about its content or function. Similarly to DataTiles, Silicone iLluminated Active Peripherals (SLAP) widgets [WWJ+09] are a group of physical controls made of silicone or acrylic destined for user input in tabletop interaction. They are inexpensive, battery free and untethered, combining the flexibility of virtual objects with tangible affordances of physical objects.

2.4.2 Atomic User Interactions

One of the main challenges when transferring traditional desktop GUI interfaces to tabletop environments is the lack of support for basic/atomic user interactions like pointing, selecting, scrolling and menu navigation as there are no equivalent interaction techniques [ASA10]. The adaptation of atomic user actions to tabletop environment is problematic as tabletop systems offer richer interaction possibilities and the size and position of tabletops affects large reaching distances and ranges, display occlusion by users hands and physical objects, lack of precision, need to orientate objects, the need for sharing and the lack of tactile feedback [ASA10]. The need to support atomic user interactions has oriented several studies towards the exploration of several user interactions.

Selection is the process of highlighting a target object on the screen [ASA10]. This action is commonly studied with pointing, as they are similar. Most touch based system tabletops are either two state devices (“out of range” and “tracking” states) or three state devices (“out of range”, “dragging” and “tracking” states) [ASA10].

The two-state issue (when the input is fingers, hands or other parts of the forearm) is addressed by the techniques such as Take-off [PWS88] (selection on finger down-up transition within target areas), Double tap [MO98] (selection on finger down-up-down-up within target area) and SmartSkin mouse press [J02] (selection on hand-surface distance; distance between surface and palm is measured to distinguish press and released states), among others [ASA10].

All of the previous solutions do not take into account the problem with occlusion, namely, fingers cover the object (or part of) challenging the selection of small objects or objects that are closely located. Offset techniques such as Cursor offset [BWB06] (cursor is presented above the finger, allowing the selection of small targets without covering them), Dual finger stretch [BWB06] (area of interest is stretched, permitting for easier selection) and Shift [VB07] (callout is presented with information about the occluded content), among others [ASA10], have been used to resolve this problem.

In case the input method is tangible objects, selection is achieved by placing the object over the virtual object. To deselect the virtual object, the tangible must be removed, covered or partially covered by the hand (if the tangible tracking is above) [ASA10].

Rotating is the ability to reposition (translate) and reorient (rotate) objects [ASA10]. In tabletop scenarios with several users, rotating is crucial as its necessary for all the users to have access to information even if they to not have same perspective. Rotating also indicates if an object is private or public if it is oriented towards the user or away from the user respectively.

There are three main approaches to resolve the rotating issue. The first involves the rotation of the entire workspace while the second involves automatic rotation of objects according to the position of the user. Lastly, a user could manually rotate objects. One simple way to resolve the rotation issue is by opting for iconic objects, which do not need text labels. However, since having a tabletop application without any text is very rare, several techniques have been developed to overcome this hurdle [ASA10]. Among these techniques of note, Corner-to-rotate [ASA10] (touch a corner of the virtual object and then turn it around a central axis), Rotate 'N Translate [KCS+05] (a virtual object is divided into a central circle and periphery and pressing the central circle allows for translation of the object, while pressing the periphery allows for rotation and translation), Turn and Translate (TnT) [LPS+06] (surrogate physical object is placed on

top of the virtual object) and Two Finger rotation [HMT03] (using two fingers to establish moving, rotating and scaling).

Pointing is the process of moving a cursor from an initial position to a target object on screen [ASA10]. Techniques are divided according to the area where they take place into direct pointing techniques (interaction with objects happen on location of that object even if it's considered group space) and indirect pointing techniques (interaction with objects happens on personal area) [ASA10].

Direct Pointing techniques include Drag-and-drop [RM00] (user selects a job by touching it with a stylus and the deselecting it by lifting the stylus) and Bubble cursor [GB05] (selection of targets by analysis of empty space nearby the input spot, through Verenoi diagrams), among others [ASA10]. In regard to Indirect Pointing, there are several techniques such as Radar views [R97] (miniature view of the workspace is presented in front of the user, where he can use a stylus to select) and Throw and flick techniques [WB03] (simple strokes to slide an object).

Scrolling shifts the viewport to content of interest that currently resides off-screen [ASA10], being especially useful in applications for information visualization (where there is a large volume of data). One of the main factors that determine scrolling is mapping function, the relation between the manipulations of an input device to the scrolling operation. For example, a zero-order position mapping system presents a proportional relation between input device and scrolling operation. As for techniques created, they can be divided according to its use of device as being device independent or pen/touch based.

In relation to device independent techniques, the most popular are Scrollbars (user positions a scrollbar thumb), Panning (ragging the content into the desired position) and Rate-based scrolling (first-order rate mapping technique where the displacement of a variable is converted to scroll movement) [ASA10].

As for pen/touch based scrolling, several techniques have been developed such as Virtual Scroll ring [MH04] (virtual doughnut shaped touchpad where scrolling is accomplished by making circular movements) and Gesture scrolling/panning [J02] (scrolling is activated by a finger slide having other fingers increase the speed of scroll), among others [ASA10].

The large volume of alternatives to support atomic user interactions attests to the richness of interaction possibilities of tabletops.

2.4.3 THREE-DIMENSIONAL INTERACTIONS

Even tough tabletops support physicality correspondent to our mental model of traditional tables, most tasks supported by current tabletops are two dimensional in nature (and not third dimensional as they are in the real world) [GW10]. In the following section, several tabletops attempting to respect our mental model will be analyzed and several techniques to achieve three-dimensional interactions will be explored.

A popular research trend in tabletops relates to the creation of 3D tabletops, systems that have a 3D environment on or above a surface. However, the most limiting factor in the creation of 3D tabletops is the restricted array of technologies available and the inadequacy of these technologies for the creation of a 3D environment.

One of the simplest methods to implement a 3D tabletop relies on using a 2D tabletop display and interacting with 3D data [GW10]. As an example of this, Roomplanner [WB03] is an application for the visualization of floor plans using an orthographic top-view projection. Wilson et al. [WIH+08] worked on bringing 3D physics to 3D tabletops by allowing the pick up of virtual objects and piling them up or tearing digital fabrics.

In order to offer a more realistic 3D experience, stereoscopic technology was used in projects like the ImmersaDesk [CPS+97], a drafting table where shutter glasses were required to provide the user with stereoscopic image considering the users location and viewpoint.

An alternative to stereoscopic technology relies in the realm of augmented and virtual reality with the head mounted augment reality display [FMS93] used in such systems as VITA [BIF04], a tabletop application for the visualization of an archeological dig. This type of systems augments the physical reality by placing virtual imagery on a viewing plane (the surface) [GW10]. Other approaches rely on the augmentation of physical reality with imagery directly projected in the objects as found in Illuminating Clay [PRI02], Sandscape [I08], Urp [UI99] and Shader lamps [RWL+01].

A promising technology for 3D tabletops is three-dimensional Volumetric Displays, where volumetric pixels (“voxels”) are illuminated to recreate truly 3D images. The main drawback of this technology relies on the surface encapsulating the voxels, which makes it impossible for the user to interact directly with the 3D image. A possible workaround to this problem involves using hand and finger gestures above the display surface [GWB04] or 6DoF input devices [GB06].

Grossman and Widgor [GW10] defined a taxonomy for 3D tabletops (founded on existent works) based on three main areas: display properties, input properties and physical properties (see Figure 15).

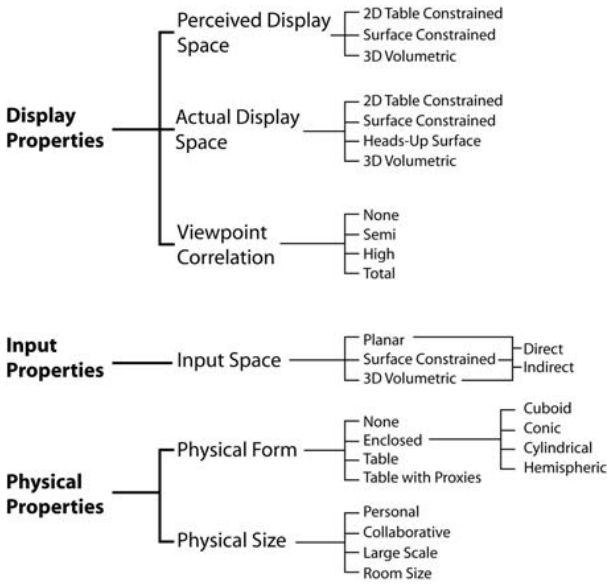


Figure 15 Taxonomy of 3D Tabletops [GW10].

In regard to Display properties, there is a separation according to location of image into Perceived Display Space (spatial location where the image is perceived to be), Actual Display Space (spatial location where the image really is) and Viewpoint correlation. Further clarification about this taxonomy can be found in Grossman and Widgor's work [GW10].

A prevalent research direction regarding tabletops is the introduction of depth, converting a 2D tabletop interaction into 3D interaction. Considering several approaches to this problem, solutions tend to be divided into 3 categories: Depth Sensing, Object Augmentation and Gesture Recognition.

The first approach, Depth Sensing, relies on having electronic equipment to detect depth. Wilson developed Micromotorcross [W07], a tabletop equipped with a depth-sensing camera above. Physical objects (papers, hand, etc.) were placed on the surface to create a virtual 3D racetrack, while a user controlled a projected racecar. As one of the first projects to incorporate depth-sensing technology into tabletops, Micromotorcross was not fully developed but showed the promise of this type of technology. Ziola et al. [ZGL+11] used a depth-sensing camera to recognize object depth, comparing it to an object database in order to achieve object recognition (triggering animation or function selection). As an extension of his work on Micromotorcross, Wilson and Benko [WB10]

used several depth-sensing cameras and projectors to allow interaction on, above and between the tabletop. This system allowed the user to transfer virtual content between surfaces by linking them with his body.

As an alternative to depth cameras, several studies were developed using the equipment below the tabletop to achieve depth sensing. Hilliges et al.'s work [WIH+08, HIW+09] achieved depth sensing information and gesture by analyzing IR reflected light. This information was then used to pick up virtual objects, having virtual shadows shown in the tabletop to be more consistent to our mental model. Similarly, Pyryeskin et al. [PHH11] (see Figure 16) split each infrared frame into layers according to brightness (reflected light) and apply a threshold filter in order to distinguish touch (bright blobs) from hovering (dimmer blobs). Techniques that use reflected light cannot detect objects above a specific height due to deferred dispersion (the further away an object is from the surface, the more the light reflected off the object spreads before its hits the diffuser, making its image on the diffuser appear blurry [BBR10]).

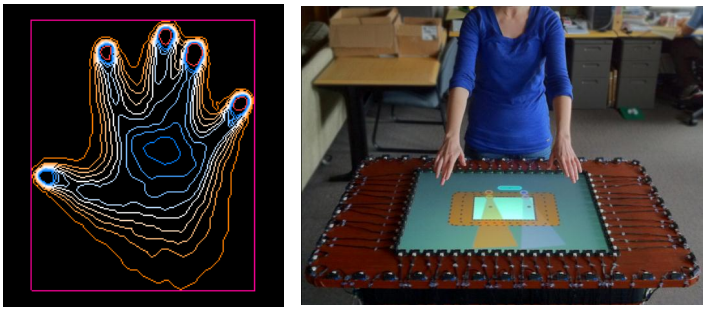


Figure 16 From left to right: Interactions in HoverSpace [PHH11], through the decomposition of reflected light into layers and Medusa [AGW+11], a tabletop enhanced with rings of proximity sensors to detect user presence.

As an alternative to depth sensing and reflected IR light, Medusa [AGW+11] (see Figure 16) is a proximity aware multi-touch table with 138 proximity sensors able to detect a user's presence and location, determining body and arm locations, distinguishing between right and left arms and map touch points to a specific user and specific hands. The sensors were distributed in 3 rings (outward, outer and inner rings) as to determine and distinguish between several actions. However, as depth sensors are not located directly on the table (but on the surroundings), hand position is calculated and not actually sensed. Plus, the complicated setup is incompatible with existent tabletops.

The second approach to achieve depth on tabletops concerns augmenting objects in such a way that their height is relevant to the system. The main challenge about the

detection of structures on tabletops is deferred dispersion. Lumino [BBR10] (see Figure 17, left) attempted to resolve this problem by having blocks with fiberglass fibers inside, which when stacked prevented light dispersion. However this system is limited to a five-block height as transition between block causes some light dispersion. Chan et al. describe CapStones [CMR+12] (see Figure 17, right), a conceptually similar system based on capacitive technology.

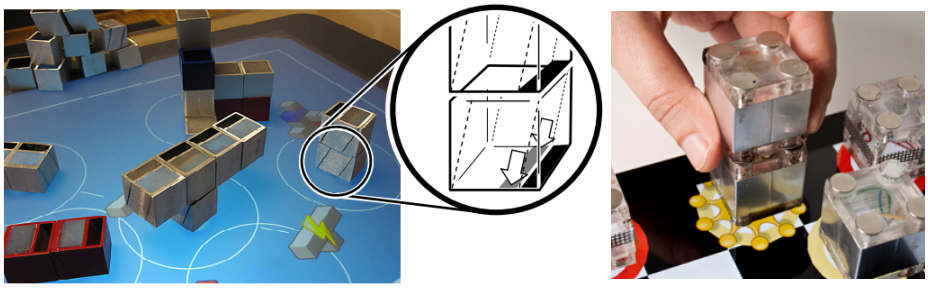


Figure 17 Object augmentation for depth, from left to right: Lumino [BBR10], tangible blocks of fiber glass and CapStones [CMR+12], tangible blocks for capacitive screens.

Finally, the third approach relies on gesture recognition above the table, using computer vision algorithms. One such example of this is Recompose [LLD+11, BDL+11] (see Figure 18, left), where gestures for selection, translation, rotation and scale are recognized. Klompmaker et al. [KNF12] (see Figure 18, right) presented a depth-camera based framework supporting multi-touch input and tangible interaction. The system supports detection and stacking of objects but is limited to unmarked rectangular objects – discrimination is possible only by their size. While compelling, this approach is not currently integrated with the powerful object recognition provided by fiducial marker systems [KB07].

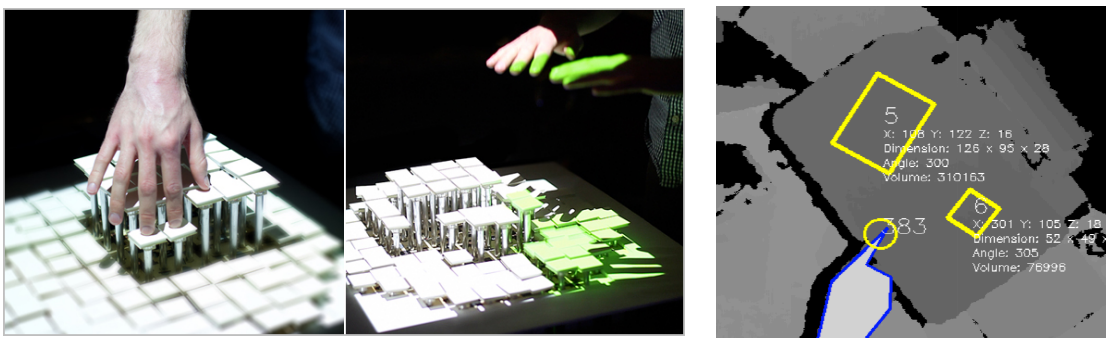
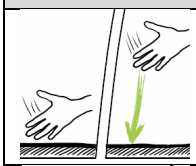
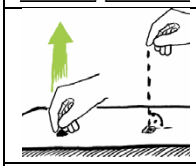
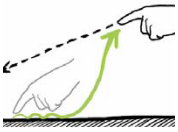
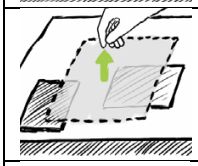
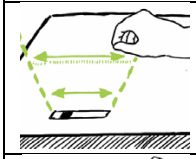
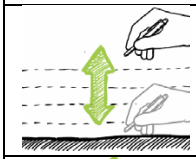
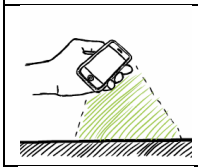
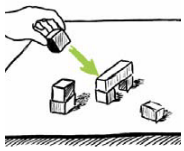


Figure 18 Gesture recognition, from left to right: Recompose [BDL+11], actuated TUI allowing for direct manipulation and gesture manipulation and dSensingNI [KNF12], framework for object and hand recognition.

In 2011, Jota et al. [JMG+11] opposing the split between interaction on the surface (hands or objects) and interaction above the surface (gestures), proposed the concept of

Continuous Interaction Space, as a unification of interaction on and above the surface (considering space above as a extension of the surface). As proof of concept, Jota et Al. [JMG+11] presented several categories of interaction gestures (see Table 1) that capitalized on the notion of Continuous Interaction Space.

Illustration	Gesture
	Mirrored Gestures – pair of gestures (similar or different) that trigger the same function, providing redundancy.
	Continuous Gestures to Avoid Occlusion – gesture started on the surface and extended to the surface above.
	Extended Reach/Raycasting Gestures – extend the users reach limitation to remote surface locations.
	Lifting Gestures to Reveal Objects – lift virtual content to reveal content underneath.
	Lifting to Adjust Scale Precision – lift to increase precision.
	Interaction with Discrete Layers – based on Subramanian's work [SAL06], the space above the surface can be split into layers where function is different.
	Stacks of Digital Objects – a tablet computer shows the content of a stack of digital objects according to tablet depth.
	Magic Lenses and View Ports – based on Bier et al. [BSP+93], a tablet or other device shows an alternative content to the one shown on the tabletop.
	Stacking of Physical Objects – meaning is achieved by a structure of physical objects as in Lumino [BBR10].
	6-DOF Manipulation – a virtual object is manipulated through the user's hand (3D position, yawn, roll and pitch).

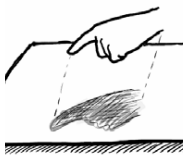
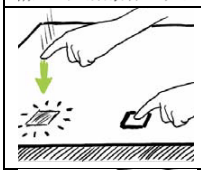
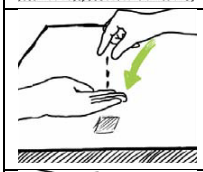
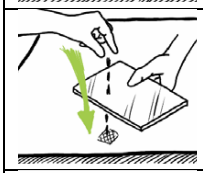
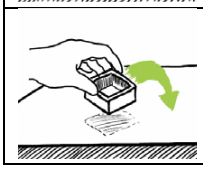
	Feedback of Hand and Object Actions by Shadows – system shows shadows as a feedback mechanism as in Hilliges et al.'s work [WIH+08, HIW+09].
	Feedback of Possible Actions by Hovering – inspired by PreSense [RIS+03], as a user hovers above an area, feedback about possible actions is shown.
	Picking and Dropping Gestures – manipulation of digital objects that follow real world physics (physics based interaction) as in Hilliges et al.'s work [WIH+08, HIW+09].
	Picking and Dropping Objects Through Filters – variation of physics based interaction, where the passage of a virtual object through a physical object (posing as a filter) alters its properties.
	Pouring Gesture – physical object functions as a container for virtual objects and its pouring, results on the virtual objects fall.

Table 1 Interaction gestures categories in Continuous Interaction Space [JMG+11].

2.4.4 PROXEMIC INTERACTION

A field quickly gaining popularity in the field of Ubiquitous Computing, is Proxemics, the study of spatial relationships namely, about how people perceive, interpret and use distance, posture and orientation to mediate relations to other people, and to the fixed (immobile) and semi-fixed (movable) features in their environment [H66]. This information can be used to mediate people's interactions with surrounding digital devices, so therefore, tabletops can also benefit from the incorporation of proxemics information caption and analysis, albeit implicit information (user does not show intention) or explicit information (user shows direct information) [BMG10]. For example, a tabletop exhibition in a museum capable of sensing presence of users would be more enticing that traditional exhibition. There are five dimensions to proxemics relationships [MDB+11]:

- Distance – A continuous measure or a discrete measure as Vogel and Balakrishnan's [VB04] zones of interaction.
- Orientation – A continuous measure (the pitch/roll/yaw angle of one object relative to another) or discrete (facing toward, somewhat toward, or away from the other object). Orientation allows determining if a user is looking at a device.
- Movement – Distance and orientation over a period of time.

- Identity – Identifies an entity in space from the specific (“this is book X”), to category (“this is a book”), to roughly ("non digital object"), to affiliation to a group ("visitor) [BMG10].
- Location – Physical context where the system is. For example, passing a threshold (a door) could trigger a certain action.

In order to simplify development of proxemic aware systems, Marquardt et al. [MDB+11] created a Proximity Toolkit for rapid prototyping (see Figure 19), also allowing for visual monitoring of proxemics interaction in a 3D space.

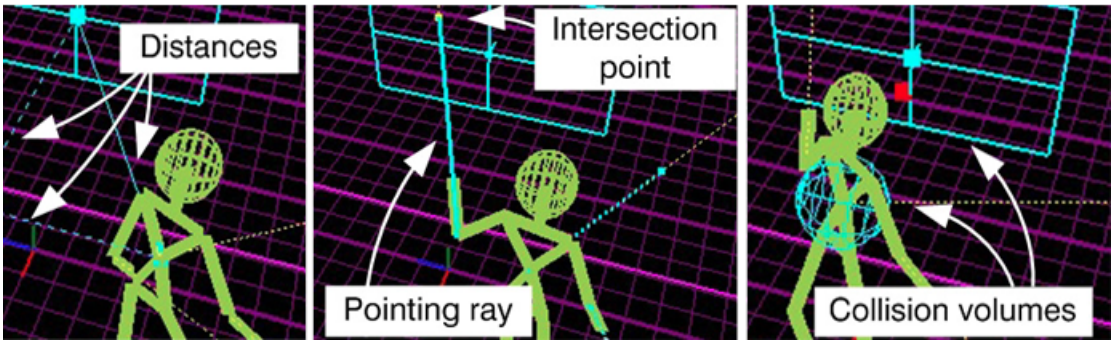


Figure 19 Proxemics relationships in the Proximity Toolkit [MDB+11], from left to right: distance, pointing and collision.

The development of Proxemic Media Player [MDB+11, BMG10] allows the identification of several characteristics that could be applied to tabletops systems. A proxemics aware system allows for the management of relationships between people. For example, if a user is facing another user instead of the surface or talking to the user, the system stopped playback. This type of information could be useful in collaboration environments. For example, if a user was looking at collaborator’s workspace, the collaborator could be notified that he was being watched. In order to distinguish between gestures for communication between users and gestures indented for the system, mobile tokens (physical objects) were used (for example, a cellphone was used as a pointer). Sensing new users, triggered actions like screen splitting or appearance of information that could be useful in tabletop collaboration environments or tabletop exhibitions.

2.5 Application Domain

The domain of 3D Object Manipulation, is an area that has attracted prior attention in the design of experimental and novel tabletop interactions, not only because it presents a vast group of features/functionalities, but also because it a rich and complex field to explore interactions, specially interactions based on gestures or object manipulation.

Au et al. [ATF12] (see Figure 20, left) discusses constrained transformations of 3D objects on touch surfaces, proposing a small set of gestures grounded on visible axes. This redesign of transformation widgets simplifies the manipulation process in touch surfaces. However, by being only based on touch surfaces, this work does not gain from the benefits brought forward by TUIs. Similarly, Song et al. [SGH12] (see Figure 20, right) present a manipulation metaphor based on handle bar behavior for mid-air interaction (through tracking gestures). This metaphor although understandable for users, also is not pertinent for TUIs. Finally, Wang et al. [WPP11] accomplish 3D manipulation by hand tracking and pose estimation via queries to a pre-computed database of hand silhouettes. While this metaphor is closer to how users use objects, it does not involve real objects.

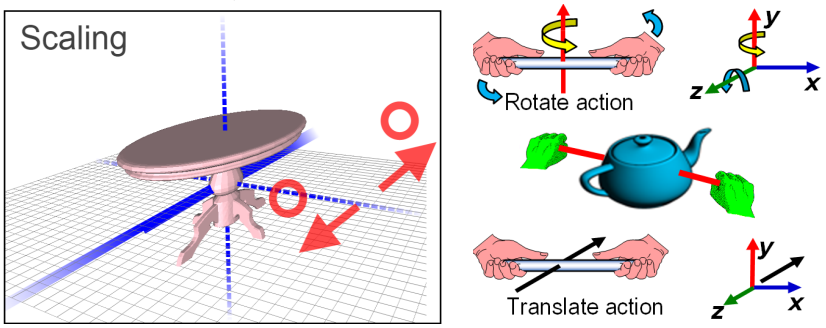


Figure 20 Manipulation Metaphors, from left to right: axis constrained scaling in touchscreens [ATF12] and handle-bar metaphor for rotation and translation of virtual objects [SGH12].

Tangible User Interfaces is applicable to 3D object manipulation as it is easy to apply and create clear mappings between the physical object manipulation and the virtual object manipulation. For instance, TangiCAD [AD07] (see Figure 21, left), a tangible/virtual construction kit, allows virtual models to be manipulated via commands issued by moving physical cubes. This manipulation interface is based on flip-the-box movements and combining different cubes. This interface presents a very weak mapping to how objects are actually used, merely presenting the cubes as selection tools for modes. 3D Manipulation has also been explored in Tangible User Interfaces in a variety of its sub domains. For example, Aguerreche et al. [ADL10] focused on virtual 3D object

manipulation through the manipulation of a Constructive Assembly system, while Follmer and Ishii [FI12] (see Figure 21, right) explore the digital remixing of shapes through a deformable gel surface.

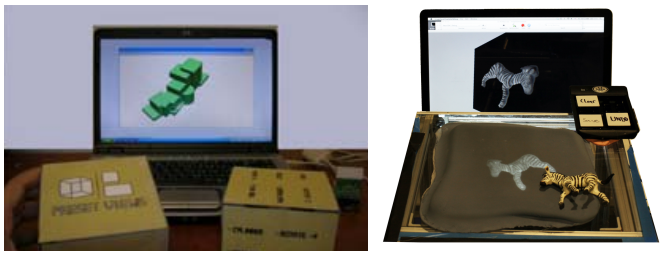


Figure 21 TUIs in 3D Modeling, from left to right: TangiCAD [AD07], TUI for architecture using cube as mode selectors and kidCAD [FI12], TUI for toy re-mixing through a deformable gel surface.

3 System Development

This section delves into the exploration of the physical setup of the tabletop, namely how depth sensing was successfully incorporated in a tabletop, through a description of the iterative process taken to achieve this.

3.1 Physical Setup

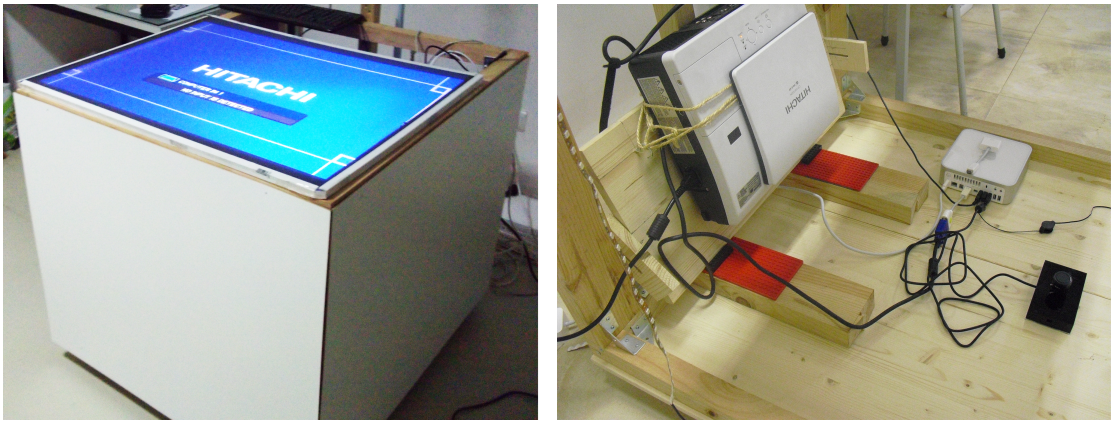


Figure 22 Physical Setup, from left to right: exterior and interior of tabletop.

The physical setup (see Figure 22) uses the Diffused Surface Illumination (DSI) method. A wooden frame encloses a Hitachi CP-AW100N projector addressing its top surface, 78 by 57 centimeters in size and comprised of a diffusing layer (5mm thick Evonik ACRYLITE 7D006) on top of a sensing layer (10mm thick Evonik Endlighten). The sensing layer is wrapped with a string of 850nm IR diodes. Although the DSI setup allows for even light and fiducial tracking, it presents less contrast compared to normal DI setups (as the plexiglass also redirects the IR towards the camera) and potentially causes problems with ambient IR (because of less contrast)[URL7] (see Figure 23).

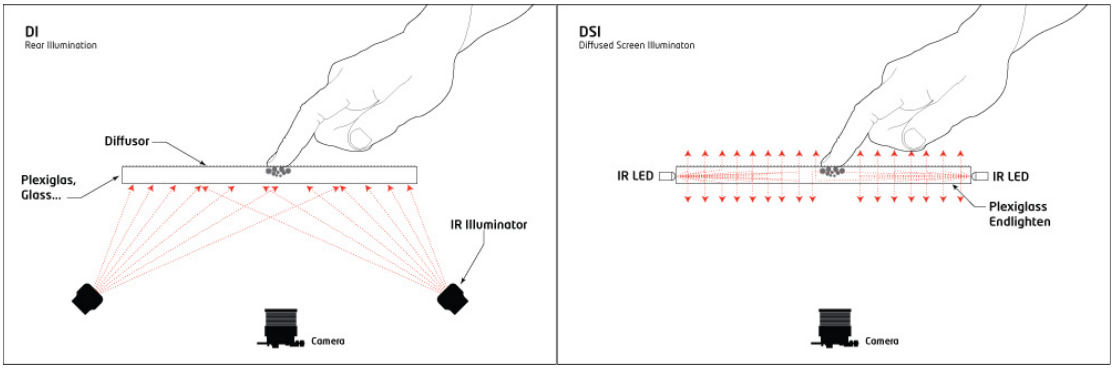


Figure 23 Lighting setup, from left to right: Rear Illumination (DI) and Diffused Screen Illumination (DSI) setups [URL7]. Setups differ on how IR light is emitted and distributed on the sheet of plexiglass.

A Fire-i Digital Board Camera [URL8] at the base of the structure captures the underneath of the surface where the tokens are placed. An alternative camera (modified PS3 Eye Cam [URL9]) was tested but did not offer benefits (such as quality of image, resolution or noise level) when compared to the Fire-i Digital Board Camera.

Two computer vision solutions for tracking of fiducial markers and touch were considered: Community Core Vision (CCV) [URL10] and reacTIVision [KB07]. Both applications utilize a video input stream and outputs tracking data and events, through TUIO/OSC protocol, to external applications. Although, CCV is capable of tracking fiducial markers and touches (the latter more efficiently than reacTIVision), these features are only present in the Windows version, while in the OSX version, only touch is capable of being tracked. Therefore, considering the importance of using fiducial markers, the reacTIVision software is preferred to recognize fiducial markers.

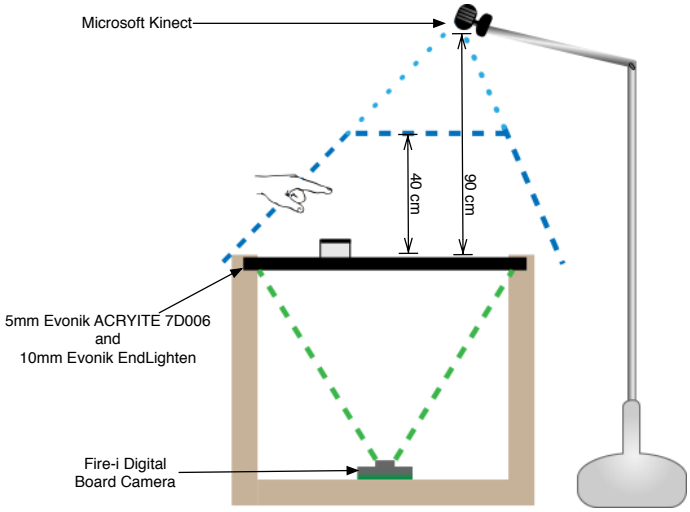


Figure 24 Detailed setup scheme.

Using a modified garden umbrella stand, a Microsoft Kinect [URL2] was mounted 90 centimeters above the surface. It was capable of sensing activity up to 40 centimeters above the surface (as the Kinect has an range of 50 centimeters to 5 meters).

In order to reduce the Kinect depth range, a Zoom (range reduction lens for Kinect) [URL11] was mounted on the Kinect. Although the new lens reduced the Kinect range allowing for closer interaction, the lens introduced a fish eye effect, which reduced our area of interest (the tabletop) but allowed for tracking users more efficiently around the tabletop (making it recommendable if the main development goal was the exploration of proxemics relations around the tabletop).

Figure 25 From left to right: depth and real images of a Kinect augmented with Zoom lenses [URL11] when IR diodes are turned off.

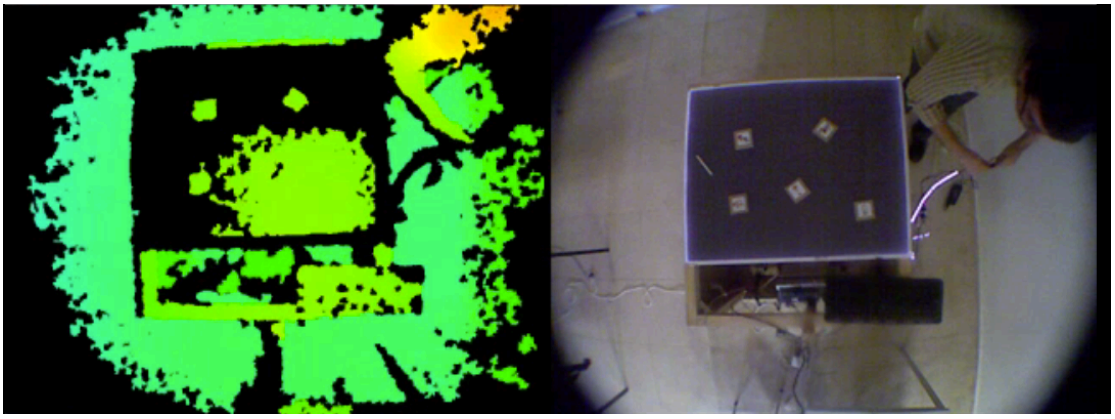


Figure 26 From left to right: depth and real images of a Kinect augmented with Zoom lenses [URL11] when IR diodes are turned on. The IR interference is clearly observed in the edges of the tabletop.

The use of Zoom lenses brought to the forefront the crucial issue of IR interference as these lenses, are more sensitive to IR interference (although this issue is also present in the non augmented Kinect). The IR light interference on the Kinect can be observed (see Figure 26) in the existence of areas of infinite depth (black areas) on the tabletop, especially around the edges of the tabletop due to the strong IR diodes.

IR light interference from the Kinect prevented reacTIVision from detecting finger touches to the tabletop. As observable in the image below (see Figure 27), IR from the Kinect creates a noise pattern in the free areas of the tabletop. This noise obscured the touches, only allowing touches to be recognized if they were sheltered from the Kinect IR; in this case, the hand shelters one finger. Markers are also subjected to the noise pattern but are only affected if the marker's border is too small. This is due to internal reacTIVision procedure where the thresholded input stream image is divided into rectangles (for rapid processing of the image). If a rectangle sector presented part of the marker and noise, it would influence the marker's identification process. Therefore,

markers with a border wide enough won't suffer from being in quadrant with noise. Fiducial makers with a black border of one centimeter around their rim were easily detectable. Consequently, tokens in the system were designed to be seven by seven centimeters (and four centimeters high) with five centimeters printed marker in their center.

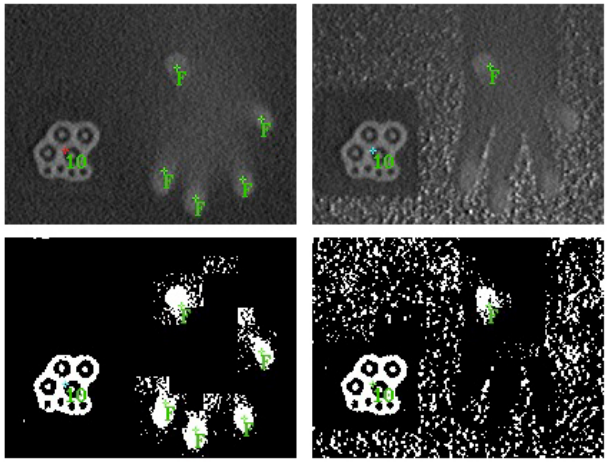


Figure 27 IR light interference. Top left image corresponds to camera image when the Kinect is turned off. Bottom left image is the previous image when thresholded to detect fiducial markers and fingers (note that all fingers and fiducial marker are identified in green). Top right image corresponds to camera image when the Kinect is turned on. Bottom right image is the previous image when thresholded to detect fiducial markers and fingers (note that most fingers are not identified due to IR light Interference).

3.2 ITERATIVE DEVELOPMENT

Based on the physical setup and the goal of creating a Continuous Interaction Space [JMG+11], certain technical requirements were identified as crucial for the development of the application:

- Plane detection is necessary for the system as knowing the plane representing the tabletop surface allows for the correct interpretation of depth data above the surface, the creation of layered interactions and the identification of objects on the surface.
- Tabletop corner detection is required for the system since there is a need to isolate what happens above the tabletop, from what happens outside the tabletop (for example, presence of users).
- Finger tracking is compulsory in the system, since IR interference disables reacTIVision touch identification (see Figure 27). Unlike reacTIVision's touch tracking which is two dimensional, finger tracking needs to be three dimensional as to be able to detect touching, hovering over objects and layered interaction.

- Due to differences in position of the Kinect, Fire-i Digital Board Camera and the tabletop, a Calibration process is necessary to unify events and allow for more precise interaction.
- To counter instabilities in the finger and ReacTIVision marker tracking (e.g. fast movements, transient tokens), a pipeline for fiducial and finger tracking is necessary to achieve a consistent state.

After identifying these five areas of technical necessity, a technical exploration process was embarked in order to identify possible solutions, iterating over time to achieve a stable technical setup. Following is a brief rundown of technical paths explored.

3.2.1 TABLETOP CORNER DETECTION

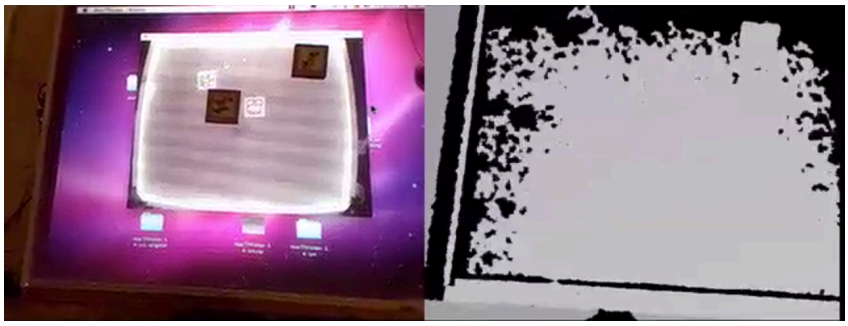


Figure 28 From left to right: real image and depth image from Kinect.

One of the first focuses of exploration was about the quality of Kinect depth data. As visible from Kinect depth image, in Figure 28, there are large patches of infinite depth (black) due to IR interference or mere occlusion. Desire to solve or improve these patches was driven by the rationale that the more information about the physical scene, the better, as subsequent steps would have a better quality base to draw upon. Solving this problem was approached from two different angles: "Do we solve the black patches only in the depth image or do we solve it by addressing the actual depth values?".

Solving the black patches issue in the depth image involved using openCV [URL6] interpolation and inpainting (reconstruction of areas according to nearby pixels) functions. In this process, after obtaining the depth image and founding maximum and minimum depths, a linear interpolation (to correct values) was used before inpainting with a mask of the unknown pixels (patches with infinite depth). Although it successfully covered the black patches, the inpainting procedure destroyed edges. Alternatively, the black patches issue in the depth map was solved using a smooth filter. This filter used a matrix (with a inner and outer band) around the missing pixel and

chose a value for this pixel using the dominant depth value of the matrix. This process allowed for the preservation of edges but was extremely cost intensive (causing missed frames).

Rather than considering the existence of black patches as a detrimental to Kinect depth quality, embracing it allowed for automated tabletop corner detection process. Although this process could be done by choosing the corner points in an image, the existence of a ring of infinite depth (due to the high intensity of IR diodes lighting) allows for the detection of tabletop borders in a manner that is easy and protected against human error. After experimenting with openCv's Canny edge detector and Sobel edge detector, better results were encountered using openCv's Hough transform. This function is able to extract features like straight lines, unlike Canny and Sobel, which are geared for edge detection. Using the Kinect depth image, openCv's Hough Lines detection is capable of distinguishing easily detectable straight lines. The lines are then organized according to their orientation and onscreen position as being top, bottom, right or left lines. The closest lines to the center of the image (the tabletops lines) are chosen and consequent intersections are used to identify tabletop corners.

3.2.2 Perspective Correction

After identify tabletop corners, the technical exploration delve into how to use the table corners to limit information on or above the tabletop from information around the tabletop. The most evident way of achieving this is to correct the Kinect perspective. To this purpose, perspective correction was attempted using Iterative Closest Point (ICP) [BM92], an algorithm to minimize the difference between two sets of points. However, both the ICP implementation from the Point Cloud Library [URL12] and independent implementation of the ICP algorithm, did not achieve desired results. Considering the rectangular shape of the tabletop, the best method found to correct the Kinect perspective is to use openCV's warpPerspective. This function not only allows the correction of the depth image, but also allows for correction of depth values (as long as they are ordered in a matrix).

3.2.3 Plane Detection

After the plane perspective correction, focus was turned on the plane detection issue. The goal of plane detection is to represent the table surface by a normal (x, y, z) and a distance, allowing for depth values from the Kinect to be converted to distance above the plane. This procedure is not as simple as it seems due to noise inherent in to the Kinect. Wilson [W10] experienced first hand this influence, stating that depth image

noise is not normal, nor is the same in every pixel location. This was confirmed by visual inspection of a 3D visualization of the depth point cloud, were points presented an irregular movement. Wilson [W10] dealt with this irregularity by taking an initial snapshot of the scene and computing a 16-bit histogram with a small range of deviations for each pixel location. Believing this to be cost intensive, the solution chosen corrects the plane given by the depth values instead of correcting the depth values.

When trying to detect a plane, a group of points on the free table surface (points within a central circle in the Kinect image) are converted to world coordinates and passed to a function that calculates the best-fit plane for those points. Using openCV's Kalman Filter (also know as, linear quadratic estimation), the normal of the new best-fit plane are corrected according to past normal values. Although first attempts of plane detection present irregular planes due to Kinect noise, the continuous corrections based on past values stabilize the plane into a plane that better represents the tabletop surface. To achieve a better-calibrated plane, a second plane detection is executed. In this procedure, three columns of equal height are placed on the tabletop surface. A parallel plane to the one calculated by the best-fit plane and closer to the Kinect (therefore, above the tabletop) is continuously lowered. Every time the plane is lowered, a depth image is combined with a mask that only allows pixels above or on the plane. A blob finder, in search of the three blobs caused by the three equal height columns, then analyzes this image. When the three blobs are found, points inside the blobs are converted to real world coordinates and a best-fit plane is found for these blobs. The user can then manually specify the distance for the plane above the surface and the distance for the plane above the objects.

3.2.4 FINGER TRACKING

After the detection of the two planes, these can be used to create depth images that are masked to only allow pixels that are on or above the corresponding plane. This results in an image where only objects and limbs can be observed (the tabletop is below the plane) and in an image where only limbs can be observed (the tabletop and the objects are below the plane), as seen in Figure 102 in Annex E. These black and white images are then used for the finger detection process. In this process, the black and white image is analyzed for contours. Contours with a small area are immediately discarded, as they are not big enough to represent limbs. A simplified contour is then converted to a convex hull in order to distinguish finger extremities from finger base. Convex hull points that are close to image boundaries are ignored as they mostly likely represent points of limb entry into the image. For each point of the convex hull that is valid, the

angle between a vector from its predecessor point and the point and a vector from is successor point and the point is calculated and is used to define if the point corresponds to a finger extremity. In conclusion, this process (see Figure 29) identifies finger extremities (x, y and z in both planes), finger base (x and y) and limbs (x and y).

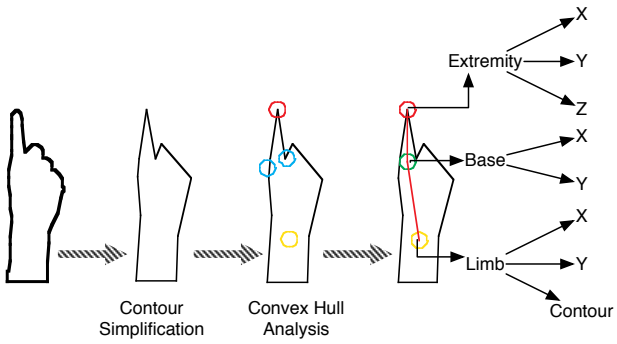


Figure 29 Finger Tracking. After contour simplification and convex hull analysis, the system is capable of detecting extremity (x, y, z), base of extremity (x, y) and limb (x, y, contour).

The process of finger detection has to be executed twice: in the depth image masked with plane above the tabletop surface and in the depth image masked with the plane above the objects. This is a required step, as if only the first image was used, fingers above object would go undetected (the masked image would mesh together the object pixels with hand pixels); however, limbs above the tabletop would be detected in both images. This issue is resolved by mixing the finger detections result from both depth images and eliminating duplicate results.

At this point, the system is capable of detecting fiducial markers and fingers. However, due to instabilities in the finger and ReacTIVision marker tracking (e.g. fast movements, transient tokens), fingers and tokens occasionally experienced abnormal behavior. As a way to counter this, a pipeline was created that registered fingers and markers only after 0.5 seconds of stable observance and maintained these 0.5 seconds after non-observance. This served to remove transient token or finger disappearances due to sensor error. The pipeline was also responsible for "stitching" current fingers with past fingers. This process of "stitching" is not necessary for tokens as reacTIVision already accomplishes it internally.

3.2.5 CALIBRATION

The final area of technical exploration left to be address is calibration. Due to differences between the projection, finger tracking and marker tracking, a calibration process is necessary to tie them all in a single solution. This procedure was based on CCV [URL10] calibration process for calibrating multi-touch events. In this process, a configurable

grid is shown on the tabletop and each grid point is iterated over when a touch event is sensed. After the calibration is finished, new touches are corrected according to the sector they belong to. Based on CCV source code for calibration [URL10], two separate calibration processes for calibration were created for the calibration of fingers to the tabletop image and the calibration of markers to the tabletop image. The calibrations results are saved to Extensible Markup Language (XML) files as to not require a calibration process every time the tabletop is used.

3.3 FINAL SETUP

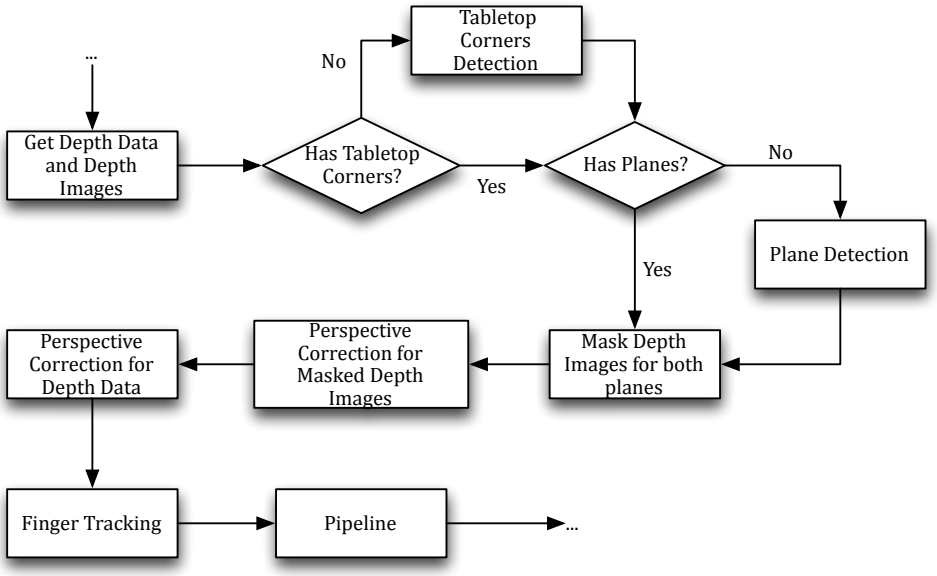


Figure 30 Software flow concerning technical issues.

The figure above summarizes the flow of technical processes necessary for converting depth information from the Kinect to knowledge about the interaction space above the tabletop.

Tabletop corner detection and plane detection are processes triggered by the user when setting up the working environment (see Figure 100 in Annex E). Alternatively, these processes also produce a loadable XML file with relevant information as to avoid the configuration process every time the tabletop is used.

After acquiring knowledge of the planes and tabletops corners, software is capable of processing depth information for subsequent use in the finger tracking and pipeline. The depth images from the Kinect suffer a masking process (according to the two established planes) and a perspective correction (according to tabletop corners), while the depth data merely suffers a perspective correction. Next, the depth images are used

to detect fingers and limbs. The corrected depth data is used to deliver the correct distance of extremities according to both planes. Finally, the pipeline is responsible for maintaining a temporal consistency in the finger tracking process. This includes "stitching" together current finger occurrences and past finger occurrences, filtering unstable fingers and applying calibration. The resulting finger occurrences are then classified in accordance to their state (see Figure 35 and 36), allowing for the rest of the software to process them for their function.

4 tCAD Application Development

With the purpose of demonstrating the feasibility of the tabletop setup, a prototype application needed to be designed and implemented. The purpose of this section is to document the conceptual, methodological and technical steps taken during this task.

The following subsections are structured to express the development process undertaken:

- Firstly, the application was explored in a conceptual level, through the elaboration of a description and scenario.
- Subsequently, the application was analyzed in terms of requirements, and how these requirements translated to use cases. The exploration of use cases was done through use case narratives and activity diagrams. Modeling concerns were topped of by the elaboration of class diagrams and state machines. Alternatively to traditional HCI methods, a TUI-specific UML [SJ09] was also used to model the application through a viewpoint of tangibility.
- After dealing with software modeling concerns, focus shifted to visual design (through wireframes and token design) and technical exploration (through the choice of libraries).
- Finally, the application was developed in an incremental and iterative approach, analyzed in the last subsection.

4.1 Concept

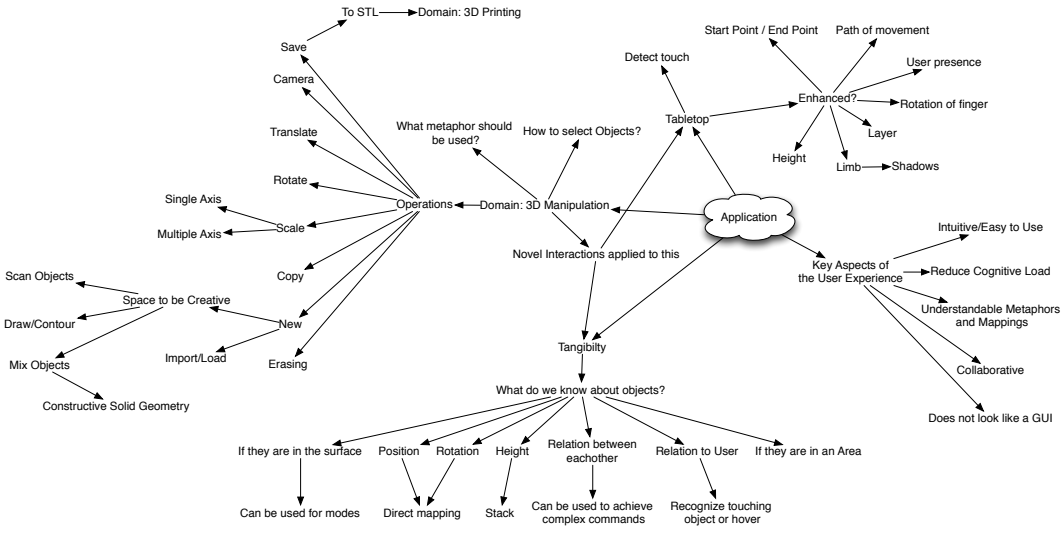


Figure 31 Brainstorming session for tCAD. For an image with bigger resolution, please refer to Annex A.

The concept for the prototype was mainly influenced from the desire to test the physical setup and the various interaction possibilities. Thus, the physical setup ensured some basic requirements to impose on the application domain:

- The application domain should take advantage of 3D interaction offered by the physical setup.
- The application domain should allow the use of tangible objects, in such a way that is not detrimental to the user experience but actually intensifies it.
- The application domain should allow for natural interaction between objects and 3D interaction. This means that interactions should model real life counterparts as to allow users to use their real world skills.

Based on these basic requirements and in line with the literature review into several domains, the realm of 3D Manipulation was chosen, as not only has attracted prior attention in the design of novel interactions (e.g. [AT12],[AD07]), but also is a creative field where the use of tangible objects is rational. Additionally, the vast realm of interaction possibilities is suitable for testing interaction techniques enabled by the Continuous Interaction Space [JMG+11].

In order to expand this concept, a brainstorm session was held (to delineate some key concerns; see Figure 31) and a basic system description was created, along with a scenario/problem statement.

4.1.1 System Description

tCAD is a prototype tabletop application for manipulating 3D Content using physical tokens and gestures in free space.

As expected from traditional 3D modeling packages, tCAD supports basic functionality such as creating, copying, scaling, rotating and translating objects. Nevertheless, tCAD differs from traditional 3D modeling packages, as these commands are done by the manipulation of physical tokens on a tabletop.

One of the core principals of tCAD is the ability of linking physical tokens to virtual objects (dynamic binding [SH10]), treating the digital content as if were physically resting on top of the table surface. In this manner, tokens are effective surrogates for the virtual content. This metaphor is direct and comprehensible as it supports users in thinking and acting, not in a virtual coordinate space, but in the dimensions of the real world.

Taking advantage of the sensing power of the tCAD hardware, novel Content Creation features are available alongside traditional Content Creation (basic shapes and copying existent content). tCAD allows for scanning real objects as 2.5D textured depth maps and using fingers to sketch 2D object contours to serve as templates for extrusion. The application also adopts a constructive solid geometry metaphor, supporting Boolean operations such as intersection, difference, symmetric difference and union.

The application explores a series for interaction possibilities by offering alternative UIs for tokens.

4.1.2 SCENARIO

Task: Making and manipulating virtual shapes to make printable 3D shapes.

Persona: Mary Lamb

It's 11 PM, Mary Lamb has just realized tomorrow is hers best friend Alice's birthday and all the shops are closed. What can she do? Luckily, she remembers her friend's love of chocolate and it seems Mary has all the ingredients to make some chocolate bonbons. Tragedy strikes again; Mary can't find the molds. Mary decides to do some molds with tCAD and then print them with her 3D printer. Things are looking up; the molds that she is going to do can be personalized to Alice's style; much better than those boring store bought molds. She decides to do 3 molds.

Time to start the first mold: a dumbbell to symbolize Alice's love of the gym. Seems like simple to construct: one long skinny cylinder with two spheres on the end. Using the Content Creation Token, Mary choses to put a cylinder in the scene. After adding a Container token, Mary selects the cylinder by pressing it and drags her finger to above the Container; a link forms. Mary needs to stretch the cylinder, so she adds a cube with the planes onto the tabletop and taps on the Container token to get to the scaling option. According to the axis shown on the plane, Mary drags the onscreen scale by the Container, making the cylinder stretch. Rotating the Container token and then moving the token along the tabletop, Mary moves the cylinder on the plane. Time to make the dumbbell ends. Using the Content Creation token, Mary adds 2 spheres. Using a new Container token, Mary links one of the spheres and drags the token until the sphere is on the cylinder. She then connects the cylinder to that Container and taps on the token until she finds the Booleans operations, choosing to unite the sphere and the cylinder. She then passes that Container between two Shredder tokens and the cylinder and sphere disappear, leaving a new shape on the plane (the union of the sphere and the cylinder).

She repeats the procedure to unite that shape with the other sphere. She's very close to finishing: she already has the dumbbell; she just needs to make the mold. After adding a cube and stretching it big enough to enclose half of the dumbbell, Mary makes a new shape by doing the difference between the cube and the dumbbell. She's done, a perfect dumbbell mold. After saving it, Mary sends it to the 3D printer and starts doing the second mold.

For the second mold, Mary wants to a mold of Mary's dog Buster, an English Bull Terrier. She doesn't want to do it from scratch with basic shapes; using the Content Creation token, Mary enters the Contour Mode. She then proceeds to draw Buster's profile on the table, using the Shredder token as a rubber if she makes a mistake. When she's done, Mary saves the contour and a paper-like 3D structure of Buster's profile appears on screen. After adjusting its size with a Container token and adding a cube big enough to enclose half of Buster's profile, she does the Boolean operation resulting in a mold of Busters profile.

One mold left; Mary wants to make a mold of a chess pawn, as Mary loves chess. Using the Content Creation token, Mary enters the Kinect Mode and adds a chess pawn on the tabletop. The 3D Scene shows a texture map of the tabletop with the chess pawn; Mary adjusts the chess pawn until she's satisfied and saves the shape. Using a Container token, Mary adjusts the shape and repeats the same procedure to make the mold as in the previous molds.

After she has printed all the molds, Mary starts making the chocolate bonbons.

4.2 REQUIREMENTS

Based on concept, system description and scenario/problem statement, a list of requirements was elaborated in order to guide the application development.

4.2.1 FUNCTIONAL REQUIREMENTS

FR1 Software supports Camera Control Operations on the 3D Scene.

FR1.1 Camera Control Operations includes rotation and zooming.

FR1.2 Camera Control Operations shall also include a mechanism to define an entry point in space for creation of 3D Content.

FR1.3 Camera Control Operations shall include plane selection.

FR2 Software supports the creation of 3D Content.

FR2.1 Supports 3D Content Creation from established 3D shapes.

FR2.2 Supports 3D Content Creation from Kinect textured depth maps.

FR2.3 Supports 3D Content Creation from freehand Contouring.

FR2.4 Supports 3D Content Creation from existing 3D Content on the scene.

FR3 Software supports the manipulation of 3D Content.

FR3.1 Supports translation of 3D Content.

FR3.2 Supports rotation of 3D Content.

FR3.3 Supports scaling (single scale and multi scale) of 3D Content.

FR4 Software supports advanced editing of 3D Content.

FR4.1 Supports the Boolean operations of Intersection between 3D Content for constructive solid geometry.

FR4.2 Supports the Boolean operations of Union between 3D Content for constructive solid geometry.

FR4.3 Supports the Boolean operations of Difference between 3D Content for constructive solid geometry.

FR4.4 Supports the Boolean operations of Symmetric Difference between 3D Content for constructive solid geometry.

FR5 Software supports 3D Content destruction.

FR6 Software supports saving 3D Content into an exportable file.

4.2.2 NON FUNCTIONAL REQUIREMENTS

NRF1 The software should be written in C++.

NFR2 System should use Open Source Libraries (instead of commercial packages) like the ones provided by OpenFrameworks Libraries and Add-ons.

NFR3 System should save 3D Content as stereography (STL) files, as these are commonly used for 3D printing and compatible with other commercial alternatives for 3D modeling.

NFR4 System should use basics shapes (sphere, cube, tube, cylinder, cone and pyramid) loaded from stereography (STL) files.

NFR5 System should limit the Graphical User Interface to bare minimums.

NFR6 System should incorporate interactions using the Continuous Interaction Space.

NFR7 System should incorporate fiducial marker tracking from ReacTIVision.

NFR8 System should allow for alternative UI for tokens, namely the Content Creation tokens and the Container tokens.

4.3 USE CASES

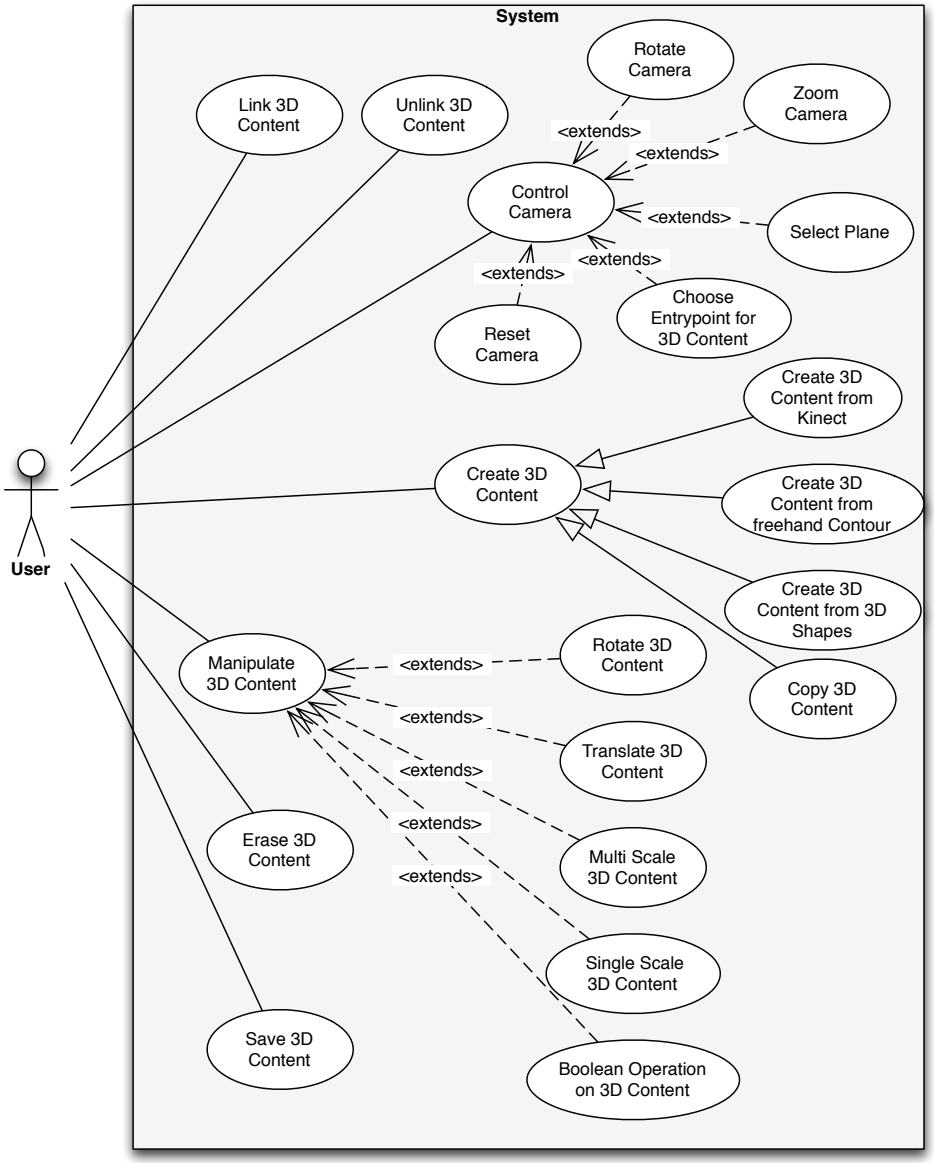


Figure 32 Use Case Diagram for tCAD.

Based on tCAD system description, problem statement/scenario and requirement definition, an use case diagram was constructed following these steps:

- Understanding context of target system.
- Identifying the actors.
- Identifying the use cases.
- Identifying relationships between actors and use cases.
- Refinement of actors and use cases.
- Identification of <<include>> dependencies, <<extends>> dependencies and generalization.

Following the construction of the use case diagram (see Figure 32), use cases were refined textually by the description of use case narratives (see Table 2 or Annex B) and visually by the construction of activity diagrams (see Figure 33 or Annex B).

Name	Link 3D Content
Assumptions	
Pre-conditions	
Use Case Initiation	This use case is initiated by demand (when the user starts a new touch event on the table surface, over the digital representation of 3D Content).
Dialog	The system verifies with the camera, if there are any 3D shapes represented in the position of the user touch. If no 3D shapes are found, stop. Selected 3D shapes change appearance and dashed lines between the shapes and the finger are drawn.
Use Case Termination	The user can hover over the Container token (Normal Termination). The user can remove the finger from the scene (Cancel).
Post-conditions	Normal Termination: Selected 3D shapes change appearance; Selected 3D shapes are added to the Container token; Solid lines between the shapes and the Container token are drawn. Cancel: Selected 3D shapes change appearance.

Table 2 Use Case Narrative for use case "Link 3D Content". For the remaining use case narratives, please refer to Annex B.

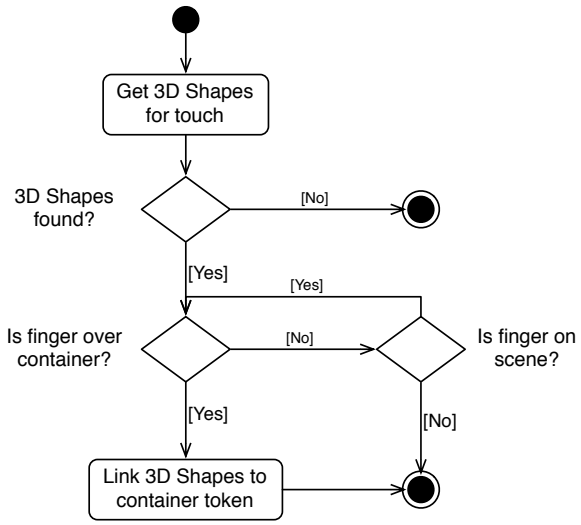


Figure 33 Activity Diagram for use case "Link 3D Content". For the remaining activity diagrams, please refer to Annex B.

4.4 State Machines

A common approach to describing input in interaction design is through state machines like Buxton's Three State Model [B90]. This approach allows for a quick specification of interactions for devices such as a joystick or a mouse. While this approach is suitable for describing token interaction (see Figure 34), the complexity of describing a Continuous Interaction Space is problematic when trying to describe interactions.

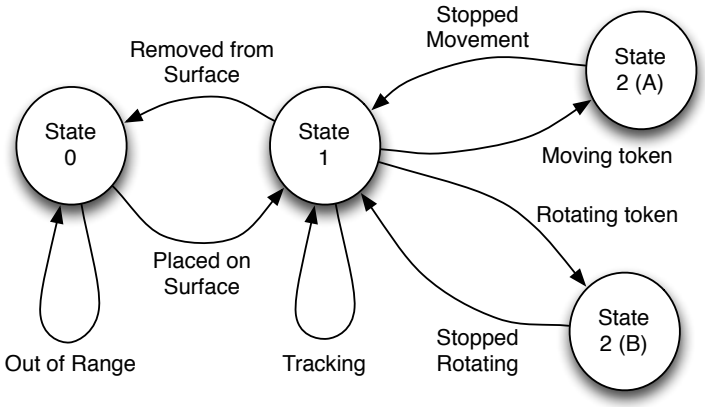


Figure 34 State Machine for tokens.

The physical setup allows for tracking finger in 5 levels: on the surface, above the surface, in the interaction ceiling (camera control layer), on the object and above the object. Some actions are triggered by dwelling on a level or by the transition between levels. Complexity is added when considering application restrictions and features; some actions are limited by graphical representations on the tabletop (menus, 3D

Content for selection, etc.) or current Mode (for example, the Contour Mode in which finger movement on the surface is used for the creation of a contour).

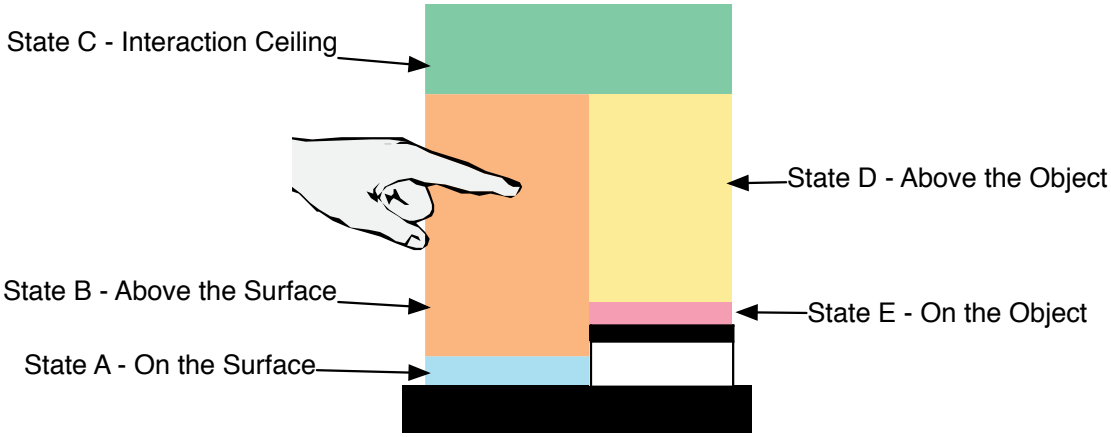


Figure 35 Interaction levels.

Considering the complexity in describing interactions restricted by 3D planes, the strategy chosen was to label a finger occurrence with two types. The first type corresponds to the interaction level; the second type corresponds to function.

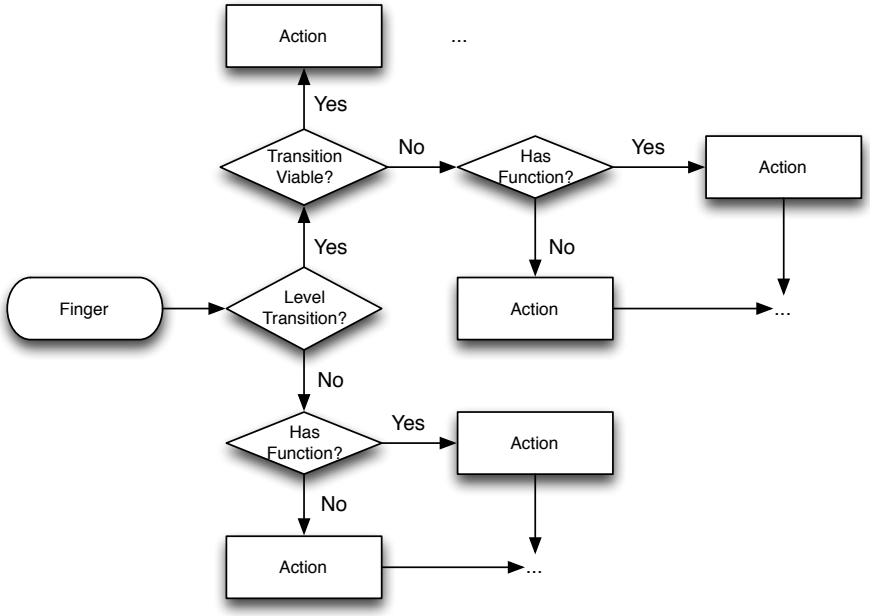


Figure 36 Software flow concerning finger processing.

A new finger occurrence is labeled with a level; according to their level, the finger may be labeled with a function. For example, a user may press on the tabletop (this finger is labeled as being in State A – On the Surface); the place were the user pressed is analyzed to determine the function for the finger press; the finger may be placed over a menu, over a 3D Shape, over a relevant graphical element (like the line connecting

shapes to Containers) or may simply be unrelated to any type of function. The label attribution of a function may trigger an action (for example, a user presses a button in a menu and an action is executed). Alternatively, actions may be triggered when a finger labeled to a certain function passes from a level to another (for example, when a finger marked as "selecting objects" passes from above the surface to above the object, the objects are linked to the Container).

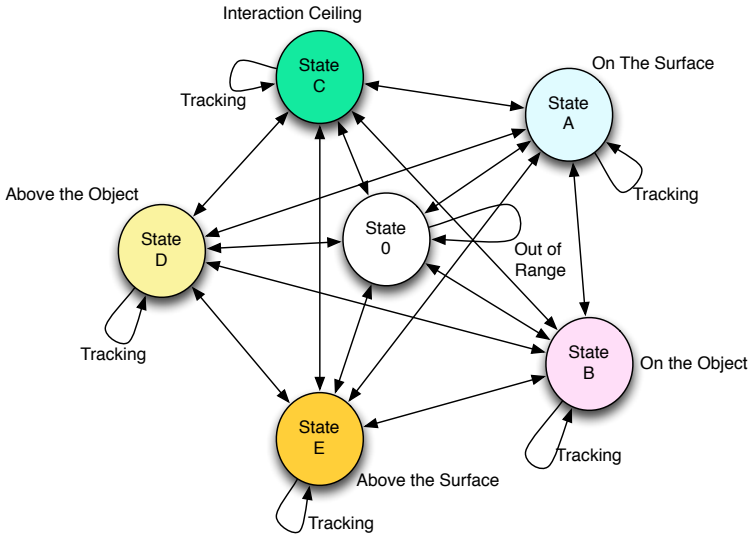


Figure 37 State Machine for fingers. State A to E corresponds to interaction levels (see Figure 35).

The state machine above expresses the interaction of a finger considering interaction levels. Transitions between levels are expressed by connections between States A to E, while the "tracking" arrow expresses dwelling in a level. Elements like modes, graphical elements and token restrictions are abstracted from the state machine, as they would introduce unnecessary complexity to the model.

4.5 TUIML

Current tools supporting the development of TUIs like sketches, diagrams, low fidelity prototypes, storyboards and functional prototyping using toolkits offer several strengths but also, bring upon limitations; sketches and diagrams focus on exploration of high-level concepts in favor of dynamic behavior and the underlying software structure; low fidelity prototype capture the main scenarios disregarding alternative scenarios; storyboards capture physical surroundings, user motivation and emotion but fail to express continuous and parallel interactions; functional prototypes require coding time and fail to present a comprehensive set of abstractions to discuss Tangible User Interfaces [SJ09].

Considering the aforementioned limitations and contemplating the TAC framework, researchers analyzed previous TUIs to define Tangible User Interface Modeling Language (TUIML), a high level User Interface Description Language (UIDL) specialized in the description of structure and behavior of TUIs [SJ09]. In TUIML, the TAC paradigm [SLC+04] is utilized to define relationships between tokens (graspable physical tokens that represent digital information) and constraints (physical objects that limit the behavior of tokens).

The TUIML process starts off by describing the set of physical objects used in the TUI by name, visual representation and list of properties. With this list of objects, the developer can match them to define TACs and extend the documentation with the TAC palette, a table with all possible relationships for the TAC, defining a grammar for the combination of objects in order to achieve meaningful relationships (representation, association and manipulation). From this point, TUI behavior can be described in terms of a dialogue tier (overview of tangible interaction dialogue) and interaction tier (detailed view of each user interaction) [SJ09].

For tCAD, this process was replicated in order to present a system definition based on Tangible User Interfaces. Firstly, tokens and constraints were defined and combined to identify TACs (see Annex C). Following, TACs were refined by creating a TAC palette (extract of TAC palette is shown in Table 3 while the full table is present in Annex C), where TAC relationships were described in terms of representation, association and manipulation.

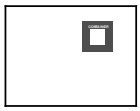
TAC	Representation			Association	Manipulation	
	Variable	Token	Constraint	TAC Graphics	Action	Response
1	3D Content	Container token	Surface		Add	Displays link between Container token and 3D Content
					Move	Update link between Container token and 3D Content
					Remove	Remove link between Container token and 3D Content

Table 3 Extract of tCAD TAC palette. For the full TAC palette, please refer to Annex C.

After the TAC palette, the following dialogue tier (see Figure 38) was constructed and interactions were portrayed by a series of interaction diagrams (see Figure 39 and Annex C for a complete listing).

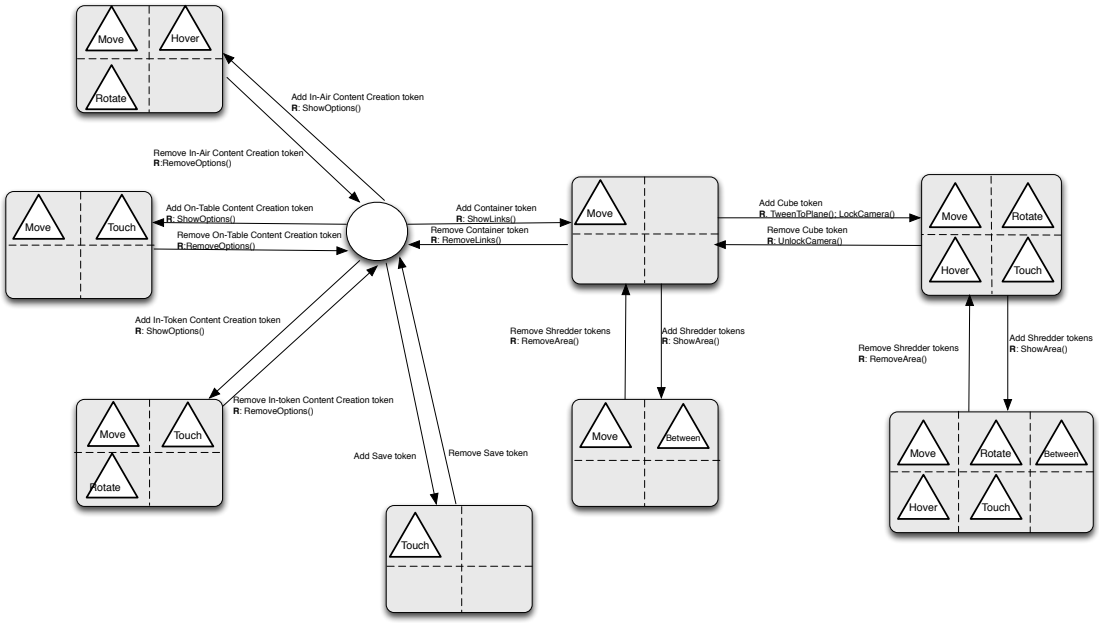


Figure 38 Dialogue Tier for tCAD. For an image with a higher resolution, please refer to Annex C.

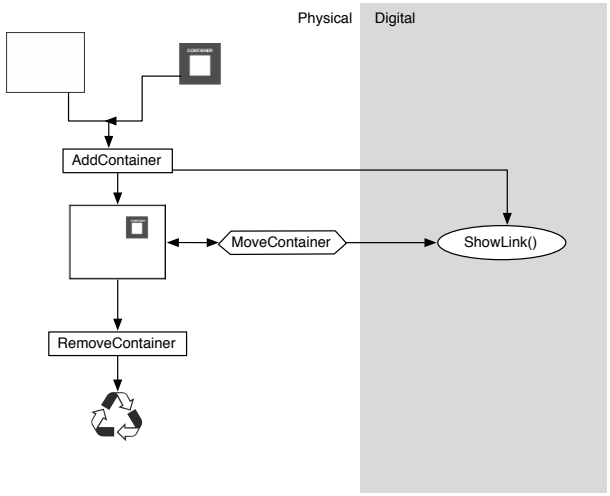


Figure 39 Interaction diagram for TAC 1 of tCAD. For all interaction diagrams for TACs, please refer to Annex C.

4.6 CLASS DIAGRAMS

The previous modeling practices are crucial parts of the process necessary to describe the Class Model; use cases are helpful in identifying objects used by the system; use case narratives and activity diagrams allows the description of object behavior and logical operations; state machines and TUIML focus on input devices and object interaction.

Taking all these modeling practices, a general class diagram was created (see Figure 40). Despite being a very abstract representation of the system, the following diagram expresses some of its main classes.

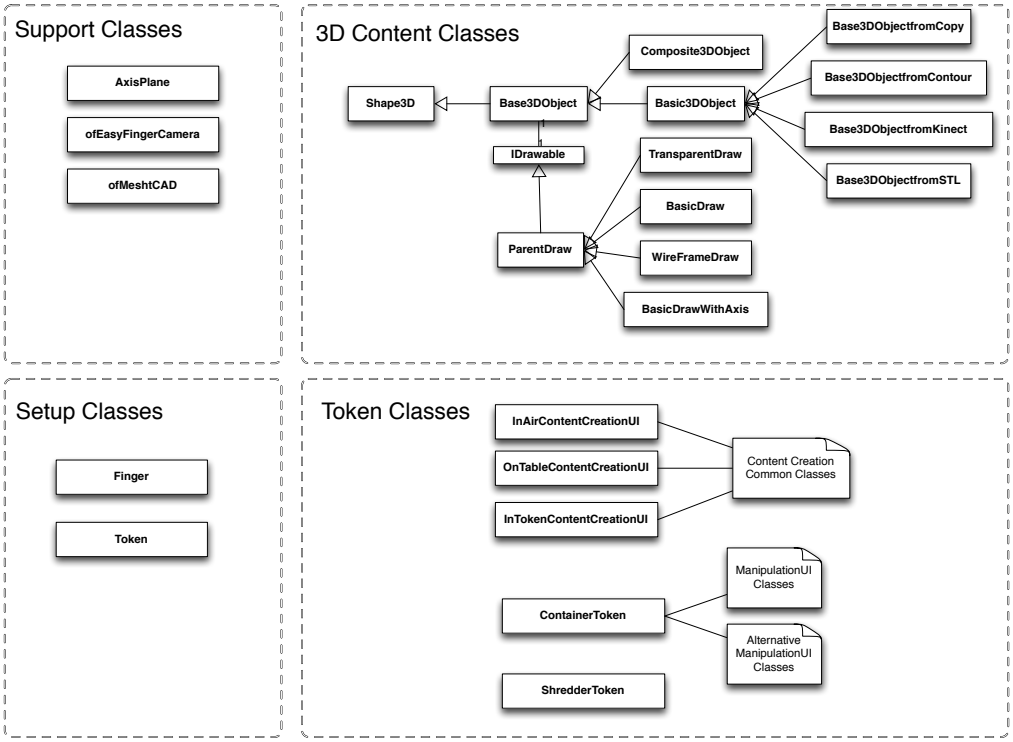


Figure 40 Class Diagram. Although this abstract diagram does not represent all classes, there is a clear separation of classes into 4 categories: Support, 3D Content, Setup and Token classes.

The diagram denotes the division of classes into one of four categories: 3D Content Classes, Token Classes, Support Classes and Setup Classes. The first category regards classes for the creation and representation of 3D Content. Token Classes are classes representing physical objects used in tCAD (tokens) and their User Interfaces (for example, the Manipulation UI and Content Creation UI). Support Classes are classes that assist other classes either by defining visual elements (dashed lines, 3D grids, etc.), representing data (plane, axis, etc.) or useful utilities (3D Camera, etc.). Lastly, Setup Classes refer to classes used in the physical setup of the system (Finger Pipeline, Calibration, etc.)

The final class diagram (see Figure 99, in Annex D), obtained by reverse engineering of the source code, is presented online alongside the source code [URL13].

4.7 VISUAL DESIGN

After focusing on the software modeling of tCAD, emphasis shifted to visual design, in order to focus on Interaction Design.

4.7.1 WIREFRAMES

Initially, the HCI method of Wireframing was used to create low-fidelity prototypes of application's key points. The following wireframes were used to structure the final

application's interactions. In Wireframe #1 (see Figure 41), the user can deal with technical issues of the physical setup, namely, loading and saving of xml configuration files, corner detection, plane detection and planes specification (distance for plane above surface and plane above objects). The user can visualize the Kinect depth (see Figure 42), either by a Point Cloud View (3D representation of the depth data as a mesh made of points) or a Depth Image View (variety of depth images with and without masks). The Point Cloud View is recommend for the plane detection process, while the Depth Image View is recommended for the planes distance specification process.

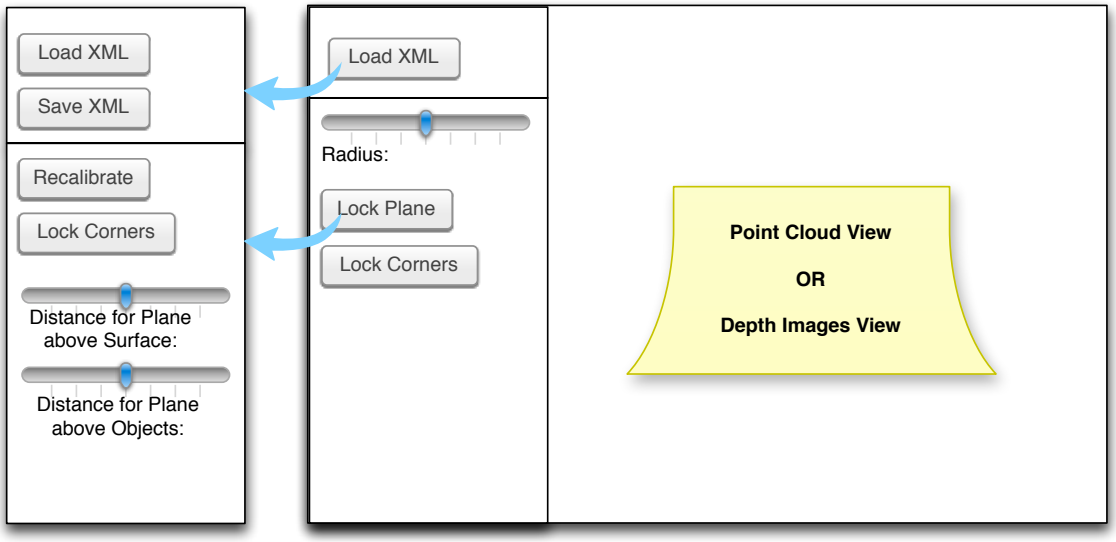


Figure 41 Wireframe 1# for Plane Detection Screen. The user can proceed to the plane detection procedure ("Lock Plane"), tabletop corner procedure ("Lock Corner") or load an external XML file with information about planes and corners. Depth data is visualized according to a Point Cloud View or Depth Images View (see Figure 42).

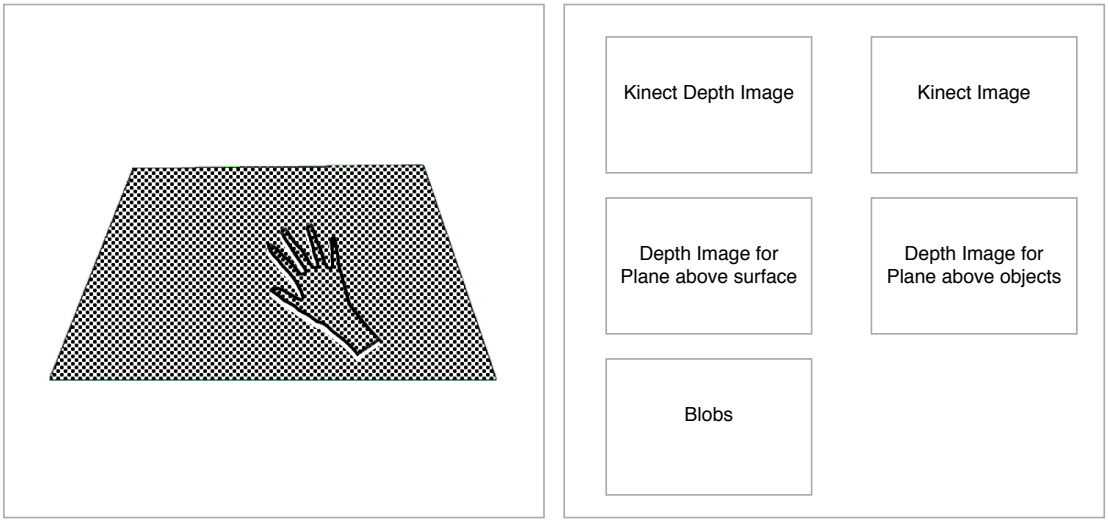


Figure 42 From left to right: Point Cloud View and Depth Images View.

In regards to the tCAD application itself, considering the need for a clear interaction space (in order to not overload the user with information and economize screen real estate for possible interactions UI), the following wireframe (see Figure 43) was created. This design focuses on the visualization of the 3D Scene/World, through the representation of axes and grids (from the combination of axes). User interface elements such as buttons and scales are not a constant presence in our design, as this application tries to escape the Wimp (Windows, Icons, Menus and pointers) GUI paradigm [JGH+08]. GUI elements are introduced by adding physical tokens in a series of interaction techniques described in depth in the next chapter (see Chapter 5).

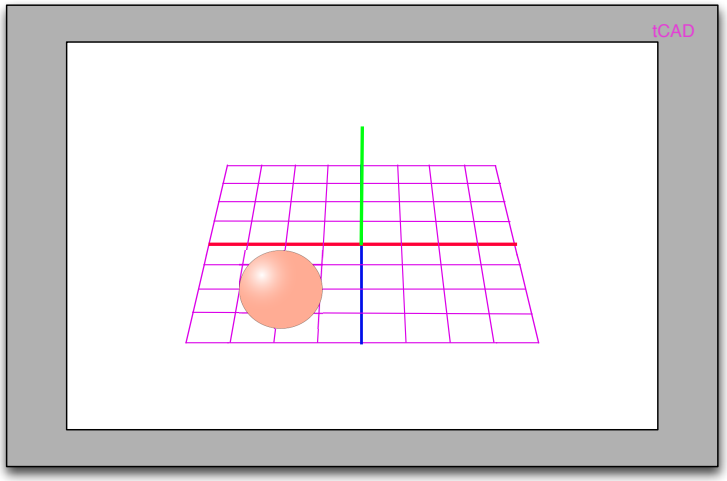


Figure 43 Wireframe #2 for tCAD application. The main view for tCAD application does not present any controllable GUI elements; commands and parameters changes are achieved either through tokens or gestures.

The color scheme for the tCAD was chosen specifically for user experience by facilitating the manipulation of 3D Content. Based on the industry standard of representing the X, Y and Z-axis in red (R), green (G) and blue (B), a plane is colored according to the two axes that form it. Specific manipulation elements in the Manipulation UI are colored according to the axis it affects. Although the metaphor of using tokens as surrogates for virtual content, treating the digital content as if it were physically resting on the surface of the tabletop, supports acting and thinking in real world coordinates, the color scheme significance works as a fail-safe in case the user is confused with the interaction.

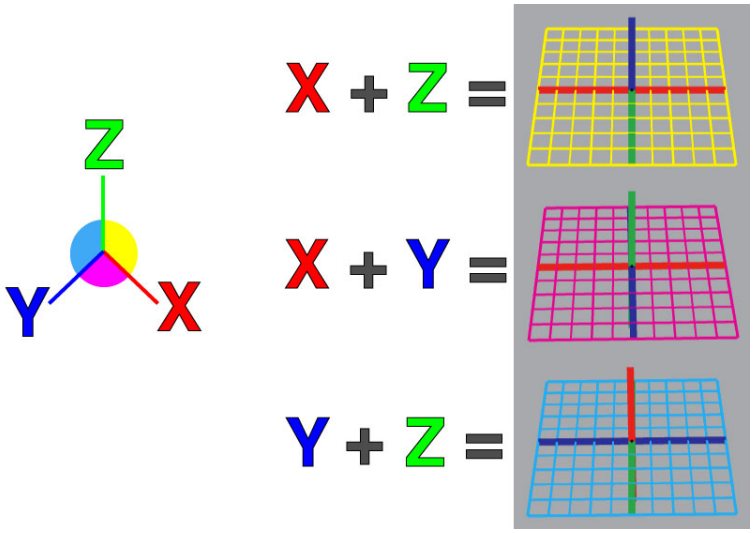


Figure 44 Color significance for axes and planes. Planes (grids) are colored according to the axes that form them; graphical elements on tokens are colored according to the axis they affect.

4.7.2 TOKEN DESIGN

As previously mentioned, physical tokens for tCAD were seven by seven centimeters (and four centimeters high) with five centimeters markers printed in their center. Limited by the shape of tokens, the only differentiation between them is left to their visual identification by presenting iconic or textual information onto to their top to facilitate identification. Accordingly to this concept, Container tokens (see Figure 46, left) were identified by their name function and a white rectangle (making it seem that the token functions as a button), while Shredders (see Figure 46, right) were only identified by name function.

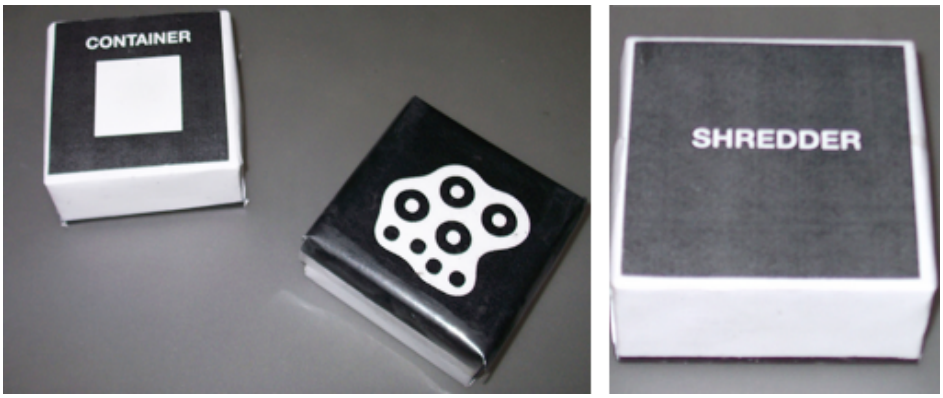


Figure 45 From left to right: Container token (Top and Bottom Views) and Shredder token.

Content Creation tokens (see Figure 45) were identified by iconic depictions relating to their interaction mechanism, which are discussed more in depth in the next chapter (see Chapter 6).

Figure 46 Content Creation Tokens, from left to right: On-table Content Creation token, On-token Content Creation token and In-air Content Creation token.

The plane selection cube token (see Figure 47) is a cube in which three faces present fiducial markers and three faces present iconic depictions representing planes. The position of the iconic representations is meaningful to the user, as he chooses a virtual plane by rotating the cube as a surrogate.

Figure 47 Plane Selection Cube token (top and bottom views).

4.8 LIBRARIES

Regarding implementation, tCAD was developed in C++ using openFrameworks [URL14] 0.07 Libraries. OpenFrameworks is a C++ open source toolkit for "creative coding", namely art installations, large-scale public projections and computer vision projects. The focus on "creative" is accomplished by creating a simplified interface for complex and powerful libraries, such as OpenGL, GLEW, GLUT and Poco.

4.8.1 EXTERNAL DEPENDENCIES

The openFrameworks is structured to allow library add-ons. The following add-ons were used in tCAD:

- ofxDelaunay [URL15] – Addon Wrapper for Paul Bourke's Delaunay Triangulation implementation; used for triangulation during Kinect Scanning.

- ofxKinect [URL16] – Addon for interfacing with Microsoft Kinect.
- ofxOpenCV [URL17] – Addon Wrapper for computer vision library OpenCV [URL6].
- ofxOsc [URL18] – Addon for Open Sound Control (OSC) communication; necessary for TUIO events reception and emission.
- ofxSTL [URL19] - Addon for importing and exporting STL 3D models in both ASCII and binary forms; used to import STL files as basic shapes and exporting 3D Content to 3D printable files.
- ofxTuioWrapper [URL20] – Addon for interfacing with TUIO events.
- ofxUI [URL21] - Addon for creation of user interfaces, allowing rapid UI design and development; used for GUI in plane detection screen.
- ofxXmlSettings [URL22] – Addon for reading and writing of XML files; used to save and load tabletop configurations (planes, calibration, etc.).

4.8.2 OFXCARVECSG

Due to unsatisfying results with Constructive Solid Geometry Libraries (ofxGeometry [URL23], ofxCSG [URL24]), an addon wrapper of Carve CSG library [URL25] was developed. ofxCarveCSG allows Boolean operations (intersection, union, difference and symetric difference) between two ofMesh (openFrameworks native class) meshes. ofxCarveCSG was partially based on the Carve CSG library implementation on Blender 2.62 [URL26].

Following recommended openFrameworks guidelines for the creation of addons, an example project was created and the addon was released on GitHub [URL27], being listed in the directory of extensions and libraries [URL28]. At the moment this dissertation is being written, this addon‘s thread in the official openFrameworks forums [URL29] has already more than 150 views.

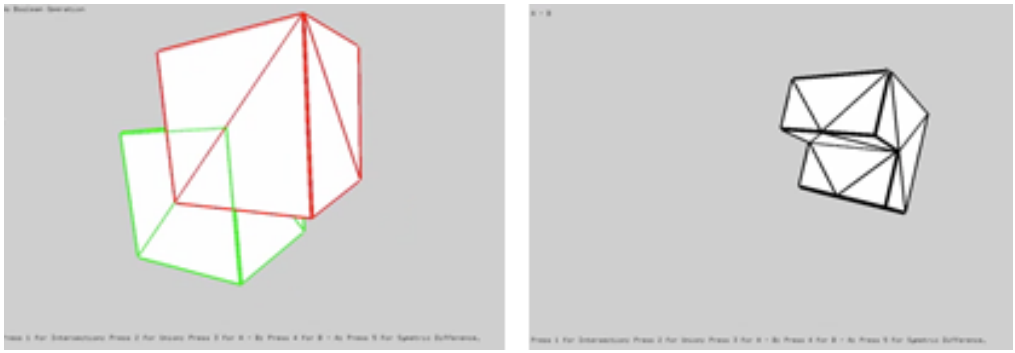


Figure 48 ofxCarveCSG [URL27] tutorial screenshots. From left to right: Two cubes with no Boolean operation performed and result of Boolean operation Difference.

4.9 Development Methodology

The development methodology used for this project was Rational Unified Process (RUP) [K03], a software development methodology focusing on incremental and iterative releases. The goal of this methodology is the delivery of executable and incremental releases of software for each iteration. This approach forces the delivery of content, supporting early problem detection.

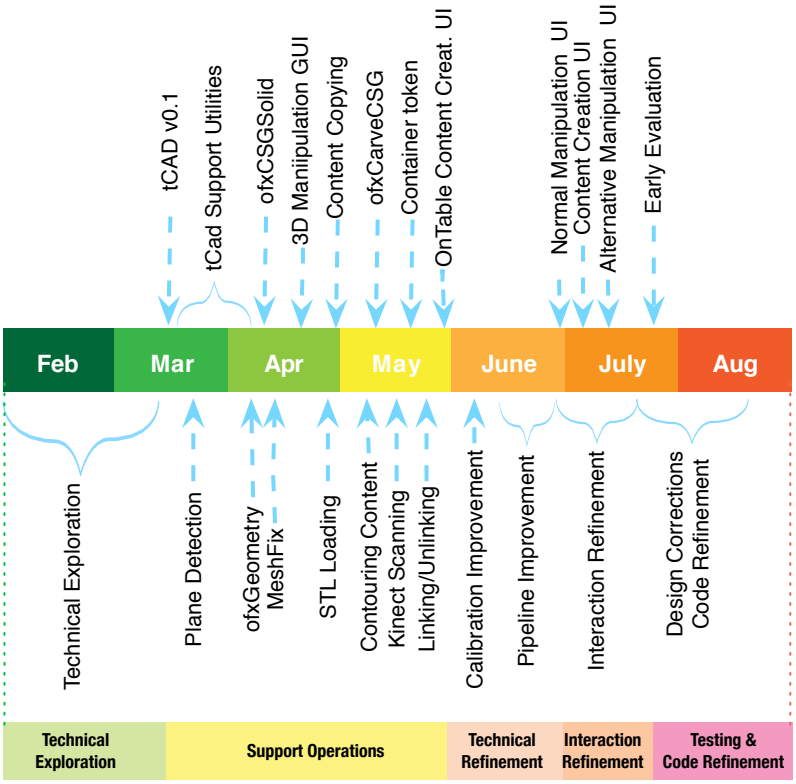


Figure 49 Development Timeline for tCAD application.

The figure above (see Figure 49) shows a timeline of the development process, highlighting key events of the development of the physical setup and of the software. Analyzing this timeline, the development process can be divided in 5 main periods:

- Technical Exploration – Through the month of February until mid-March, project focus was on the exploration of technical details of the physical setup and familiarization with openFrameworks libraries.
- Support Operations – The interval of time between mid-March until the end of May was mainly focused on tCAD design and the construction of support utilities for tCAD use cases. During this time period, support utilities like plane selection, representation of 3D Content, Content Creation and Manipulation (using GUIs instead of physical tokens), were developed. This initial period of confluence of

application design and implementation of support functionalities, allowed us to identify possible technical barriers for use cases or, in the other hand, allowed the expansion of use cases. For example, due to use cases involving Boolean operations, there was a need to test if Boolean operations were supported in the openFrameworks environment. As the native core of openFrameworks libraries does not offer Constructive Solid Geometry features, addons such as ofxGeometry [URL23] and ofxCSG [URL24] were tested. Although the addons allowed for Boolean operations of simple shapes, when dealing with complex shapes (as the ones expected from Kinect Scanning), the application either produced irregular meshes or simply crashed. Additionally, these addons did not allow all of the Boolean Operations desired from the use case. After unsuccessfully trying to correct the crashing behavior by validating meshes with MeshFix [URL30] (through a self made wrapper addon), the future of this use case seemed uncertain. However, the creation of ofxCarveCSG allowed for Boolean operations with complex meshes, reassuring to the use case validity.

- Technical Refinement – The month of June was spent improving tracking of fingers and fiducial markers and calibration in order to make the application more robust and allow for richer interaction techniques.
- Interaction Refinement – After accomplishing stable tracking of fingers and fiducial markers, a portion of July was spent creating and improving the interaction techniques, namely Content Creation UI and Manipulation UI, in order to have the prototype ready for an early evaluation in late July.
- Testing and Code refinement – Finally, the remaining time of development was spent on testing/debugging, design corrections and code refinement. Annex E presents some screenshots of the final tCAD application during runtime.

5 tCAD Interactions

One of the key objectives of tCAD application was the development of interaction techniques that took advantage of the physical setup, namely the merging of tracked objects with direct user input on and above the tabletop surface. The following subsections are structured around different techniques implemented for tCAD; a final subsection regards the early evaluation of the tCAD application, specifically the interaction techniques.

5.1 Content Creation UI

For the purpose of Content Creation, three different interfaces were designed, garnering the power of physical setup into novel interaction techniques on tabletops. Each one of the interfaces was a menu system associated to a physical token and exploring the potential of on-table, on-token or in-air input (see Figure 50). This diversity allowed us to survey the range of possibilities enabled by the Continuous Interaction Space.

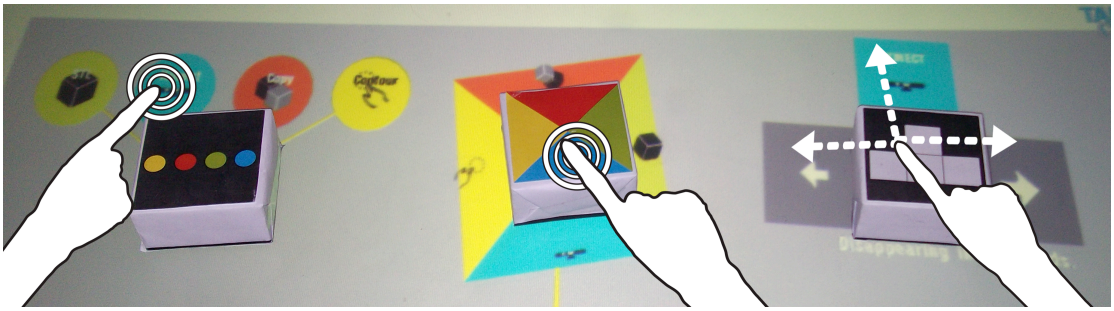


Figure 50 The three tCAD Content Creation User Interfaces, from left to right: On-table Content Creation UI, On-token Content Creation UI and In-air Content Creation UI.

The menu commands enabled by the Content Creation UI were:

- Creation of a basic object (sphere, box, cone, cylinder, tube and pyramid).
- Copying existent objects.
- Capturing an object from a depth map (Kinect Mode).
- Capturing a contour drawn on the table surface (Contour Mode).

Additionally, the Content Creation UI was also responsible for specific options on each of these modes, for example, saving or discarding a contour drawn on the tabletop surface.

For the first Content Creation UI, the on-table interface (see Figure 51), the placement of the token on the surface brought up a display of the commands. By tapping on the top of the token, these commands become invisible in order to limit the amount of on-table screen space they occupy when they are not being used. Additionally, the user can also remove the token from the surface to make the command menu disappear, or they can flip the token in order to make the system not recognize the fiducial marker. An on-screen tap of the commands leads to another group of on-screen commands (the several types of basic objects that can be added to the scene or the existent content that can be copied) or mode entry (the Kinect scanning and the Contour drawing, which both present options to save or discard the content).

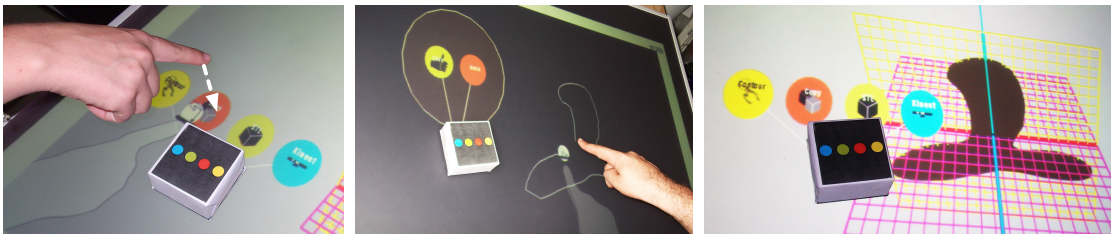


Figure 51 On-table UI, from left to right: on-table options connected to the on-table token; the on-table token is also responsible for options specific to Mode (in this case, Contour Mode); after placing the 3D Content created by Contour Mode, token revert back to their primary behaviour.

The second Content Creation UI, the on-token interface (see Figure 52), segmented the top surface of the token into four equally sized triangular zones, situated along one edge. Tapping on each of the triangles issues a command, which leads to the presentation of options on-screen. This design is intended to support input initiated from each side of the token – the interaction involves rotating the token until the relevant command faces the user and then simply tapping the closest segment. This type of interface is limited to the size of the token and the resolution of the tapping detection. For example, an earlier interface attempt, presented triangles that were too small, leading to difficulties detecting which triangle was tapped and occasionally provoking wrong command issuing. Using bigger triangles (even if they surpassed the area of the token) resolved this issue, as there was a better differentiation between triangles. The usage of the top surface presents another limitation, as input space limits the number of options expressible. For this reason, the several types of objects (sphere, box, etc.) cannot be expressed on the token surface, but rather on the screen.

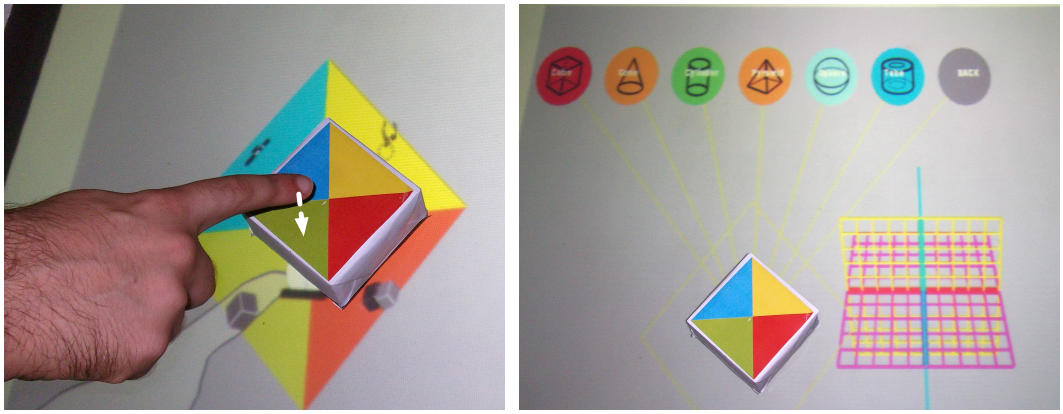


Figure 52 On-token UI, from left to right: options are distributed on the top of the token requiring the user to tap on the token to activate a option; options related to Modes (in this case, STL Mode) are shown on-table.

Finally, the third Content Creation UI, the in-air interface (see Figure 53), when placed on the tabletop surface caused the appearance of a graphical representation in the form of an inverted T shape. This interface includes a menu option presented on the opposite side of the token to the users hand; a hover movement from the token to this item selects it. Alternatively, to the left and right of the user's hand were scrolling zones; hover movement from the token to these items causes the menu option to change according to a carousel metaphor. To limit the amount of on-table screen space they occupy, a timeout mechanism was incorporated that made the interface invisible after six seconds of no use (after six seconds, hovering over the token would make the menu visible again and restart the timeout process).

Figure 53 In-air token, from left to right: hover movements over the token are responsible for selecting an options; hover moments are also responsible for selecting options specific on Modes (in this case, Kinect Mode).

5.2 Linking and Unlinking Content

For the manipulation of 3D Content, tCAD relies on Container tokens, physical objects that act as surrogates for virtual content.

In previous iterations of the design, one of the focuses of concern was how to map content to tokens. A simplistic approach would be to have a one to one mapping, where each token was mapped exclusively to one 3D shape. In this design, when using the Content Creation UI, the content added to the scene would be immediately connected to a free Container token. This approach presented two critical faults: having a one Container to one 3D Content mapping disabled the ability to have group manipulations (as tokens were manipulated separately); the one to one mapping causes an abundance of tokens on the surface (diminishing the amount of free screen real estate).

Faced with these critical faults and based on the MCRit model [UI00], a proposal to use a many to many mapping between Containers tokens and 3D Content seemed better suited for this type of application. This mapping would not only allow for group manipulations (a Container token mapped with several digital representations), but also reduce the number of tokens for the system (as Containers tokens were able to have a mutable combination of digital representations). This mapping also permits cooperation between users, as a digital representation can be mapped to two Containers tokens, which can be manipulated by two different users in distinctive ways.

Therefore, the mapping used for tCAD was: a Container token can be linked to zero or more 3D Content and 3D Content can be linked to zero or more Container tokens. The process of linking and unlinking has to be executed during runtime.

Linking is accomplished by touching the on-table depiction of a virtual object and then, in-air movement to a physical Container token (see Figure 54). A link between the virtual object and the physical token is then established and highlighted on-screen by a line. This procedure is a perfect example of an interaction taking full advantage of Continuous Interaction Space concept [JMG+11]; the interaction starts on the tabletop surface but ends on the object or in the space above the object.

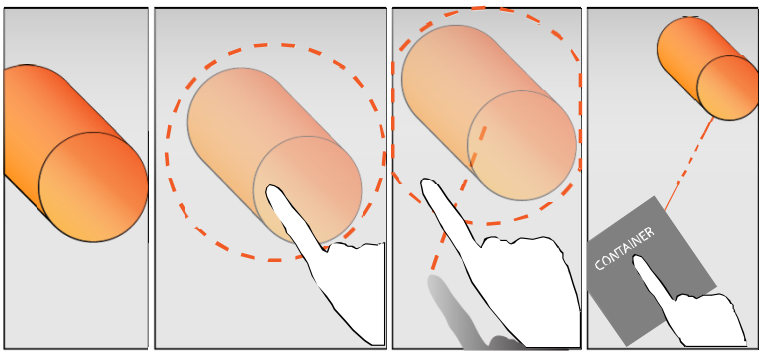


Figure 54 Linking a digital representation to a Container token, by starting the interaction on-screen and ending over the token.

Link cutting was achieved by dragging a finger over the graphical representation of the link, slashing the line and connection between the objects (see Figure 55).

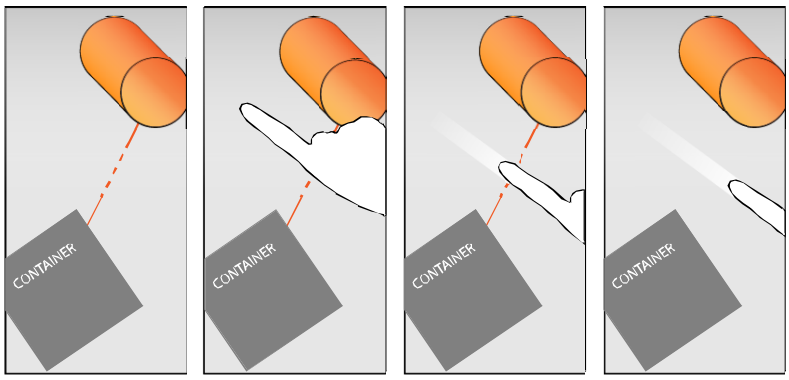


Figure 55 Unlinking a digital representation from a Container token, by slashing the connection between them.

5.3 MANIPULATION UI

For the purpose of manipulation of 3D Content, a simple interface was constructed around the Container tokens. Even tough physical manipulations of the Container token allowed for some of the operation designated in the use cases, the unsupported operations have to be accessed through this Manipulation UI; tapping on the surface of the Container token iterated between these operations.

Token usage was based on the idea of leveraging the Continuous Interaction Space [JMG+11] to maintain a physical parallel between inputs to the system and the changes to the linked virtual object. This was achieved by mapping horizontal movements of the token on the table surface to movements (or rotations) of virtual object along (or around) the current camera plane. Movements orthogonal to this plane (e.g. vertically) were achieved by detecting the presence of fingers in the space above a Container token.

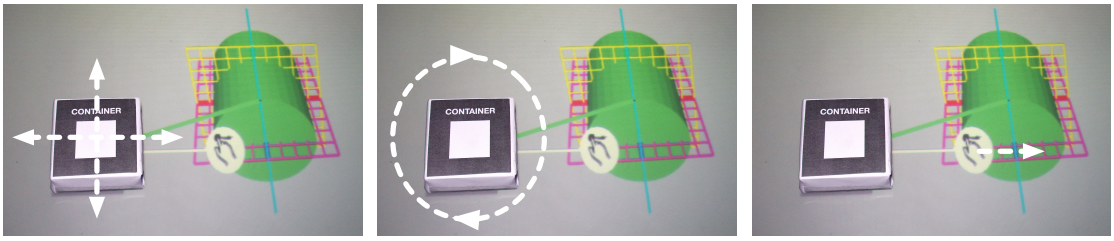


Figure 56 Container token interactions, from left to right: translating, rotating and scaling .

Two alternative mappings for the Manipulation UI were produced. In the first mapping, a fingers relative movement up or down, over the Container UI, was mapped to movements or rotations of the virtual object, for example, pushing it downwards or pulling it up. Initial observations suggested that accurately performing such gestures

was challenging, so an alternative technique based on dwell time over Container. This design resolved the accuracy issues with the initial system.

For each Container, there are four modes accessible by tapping the Container token:

- Translation Mode (see Figure 57) – Translate the linked content in the axis perpendicular to the selected plane.

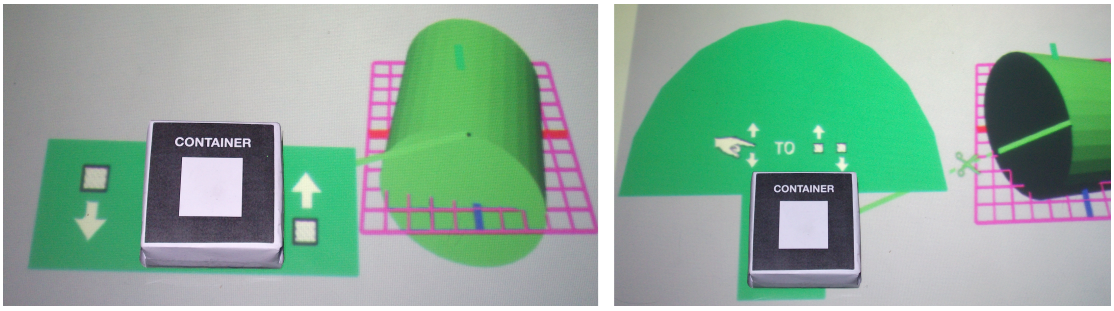


Figure 57 Translation Mode UI, from left to right: in the dwell Container and in the finger movement Container.

- Rotation Mode (see Figure 58) – Rotate the linked content in the two axes that represent the selected plane (physical rotation of the Container token, rotates the object around the axis perpendicular to the selected plane).

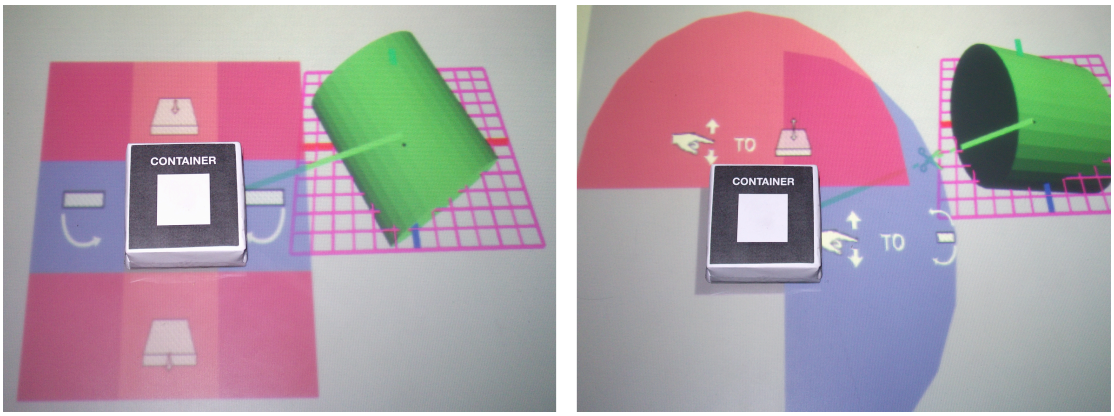


Figure 58 Rotation Mode UI, from left to right: in the dwell Container and in the finger movement Container.

- Scale Mode (see Figure 59) – Scaling the linked content for each individual axis (contrasting to the onscreen scale presented when no mode is selected, for the scaling of three axis at the same time).

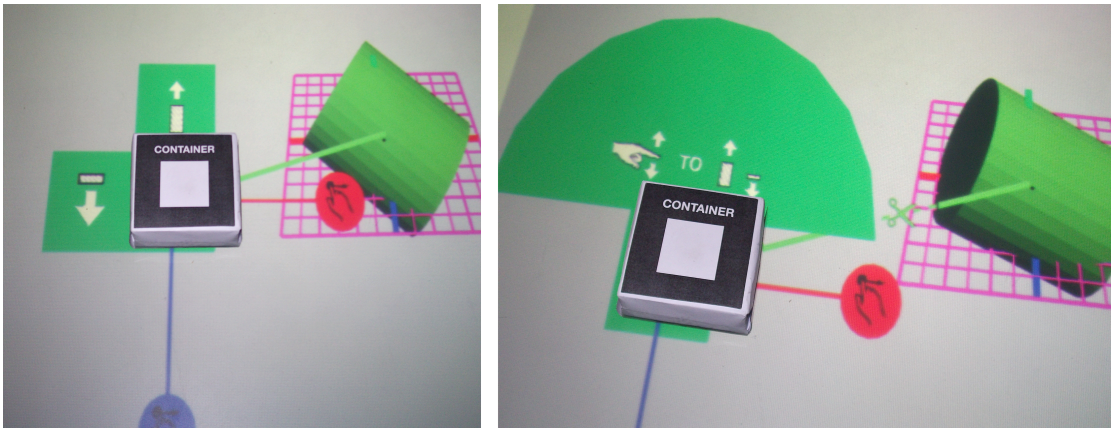


Figure 59 Scale Mode UI, from left to right: in the dwell Container and in the finger movement Container.

- Boolean Mode (see Figure 60) - Boolean Operation when the Container token is linked to two shapes.

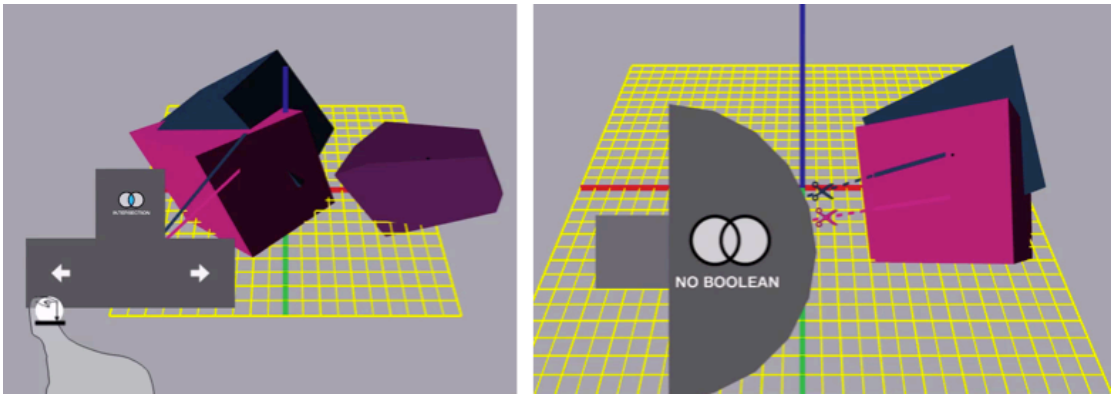


Figure 60 Boolean Mode UI, from left to right: in the dwell Container and in the finger movement Container.

For the first Manipulation UI, the Container that responds to finger movement:

- No Mode chosen (see Figure 56) – A white onscreen draggable scale is shown on screen for the multi-scale scaling of linked content.
- Translation Mode (see Figure 57, right) – An area on opposite side of the token to the user is shown; a finger's relative movement upwards or downwards, is replicated in the linked content as if the user was pulling the content up or pushing it down.
- Rotation Mode (see Figure 58, right) – Two areas are shown: one area on the opposite side of the token to the user and one area on the right side of the user. In each area, the relative movement upwards or downwards, causes a rotation in the corresponding axis, as if the user was pulling or pressing the top or right edge.

- Scale Mode (see Figure 59, right) – Two draggable circles are shown onscreen. A draggable action on these circles corresponds to a single axis scaling along one axis (one of the two axes that define the selected plane). Additionally, an area on opposite side of the token to the user is shown. In this area, a finger's relative movement upwards or downwards, causes the linked contents scaling up or down (on the remainder axis), respectively.
- Boolean Mode (see Figure 60, right) - One area on the right side of the token is shown; a finger's relative movement upwards or downwards, iterates over the various Boolean possibilities, following a carousel metaphor. Choosing a Boolean option is done by leaving the Boolean Mode (through token surface tapping), when the desired option is shown.

For the alternative Manipulation UI, the Container that responds to dwelling fingers:

- No Mode chosen (see Figure 56) – A white onscreen draggable scale is shown on screen for the multi-scale scaling of linked content.
- Translation Mode (see Figure 57, left) – Two rectangular areas are shown. The dwell time on one area causes the upward movement of the linked content, while the dwell time on the other area causes the downward movement of the linked content.
- Rotation Mode (see Figure 58, left) – A grid structure of rectangles are show around the token. The finger's dwell time on an area is translated to a rotation movement as if the user was pressing down on the edge made by that area and the token.
- Scale Mode (see Figure 59, right) – Two draggable circles are shown onscreen. A draggable action on these circles corresponds to a single axis scaling along one axis (one of the two axes that define the selected plane). Additionally, two rectangular areas are shown: one on the opposite side of the token to the user an one on the left of the user. A fingers dwell time on this area in this area causes the linked contents scaling up or down (on the remainder axis).
- Boolean Mode (see Figure 60, left) – Two areas on the left and right side of token is shown; a finger's dwell time on an area iterates over the various Boolean possibilities, following a carousel metaphor. Choosing a Boolean option is done by leaving the Boolean Mode (through token surface tapping), when the desired option is show.

In addition to the Manipulation UI, hovering over the object is also detected and is used to present measurements of each of the linked 3D shapes.

5.4 Depth Layers

The incorporation of depth sensing into tCAD allowed input at different heights above the table to be mapped into different commands, specifying different interaction layers.

Beyond the interactions described previously, finger movements between 30 and 40 centimeters above the surface were used to adjust the camera. This was based on a metaphor of controlling a camera situated above the table. The incorporation of this layer avoided the use of tokens for controlling the camera (and therefore avoided the use of illogical mappings between camera functions and tokens). Additionally, using a layer at that height resulted in slightly uncomfortable gestures for most users, accentuating the difference between the normal interaction space for 3D manipulation and the camera control layer.

The numbers of fingers present in the layer serves to separate between functionalities for camera control. If one finger was present, the camera rotated around the center of the tabletop, mimicking the direction of the users fingers (see Figure 61). The mapping of two fingers is attributed to zooming (e.g. pinch to zoom; see Figure 62), as this is common gesture for this type of feature. The use of three fingers is used to define a point in space to serve as an entry point for new 3D Content (see Figure 63).

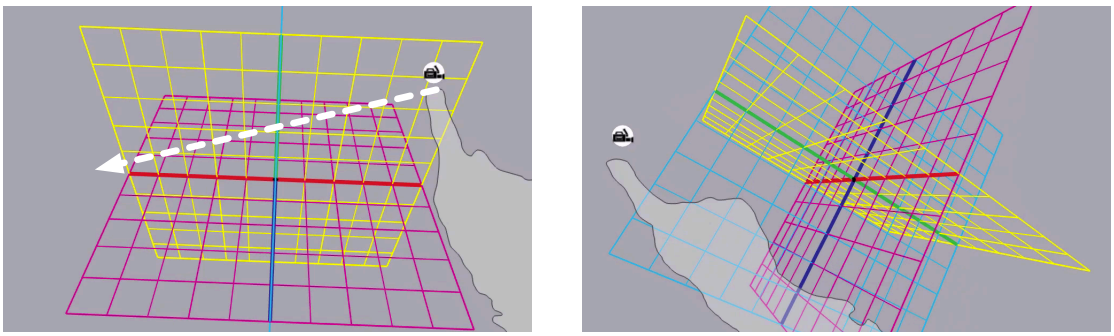


Figure 61 Camera rotation, from left to right: initial state (white arrow shows direction of finger movement) and final state.

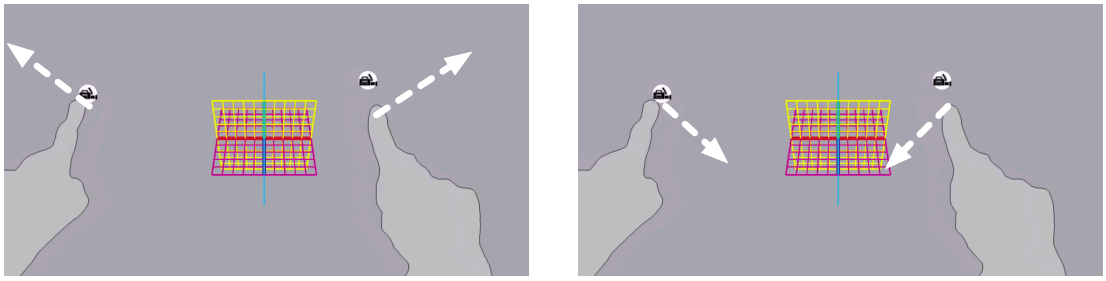


Figure 62 Camera zooming, from left to right: zoom in and zoom out. White arrows represent finger movement.

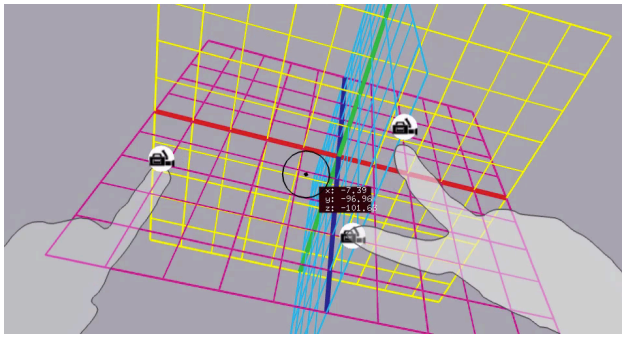


Figure 63 Entry point selecion. Entry point is represented by a blinking black circle.

A common occurrence when dealing with interaction above the tabletop is the uncertainty of movement caused by the perspective correction of the physical setup. For example, hitting a on-screen target in-air is fairly easy when the hand is close to the surface as both the target and the input are present in the same line of sight. As the distance between the hand and the surface increases, this action turns more complex as when the user looks at the target, he is not able to see the input, therefore rendering the relation between the target and the hand more difficult to perceive. The solution often implemented [WIH+08], and which this project follows, is the representation of limbs and finger on-screen as a digital shadow (captured through the Kinect setup).

5.5 Cube for Plane Selection

As previously mentioned, plane selection is accomplished by using a cube in which three faces present fiducial markers and three faces present iconic depictions of planes. Consequently, there is a simple mapping between a side of the object and one of the three planes. Switching between planes involves rotating the cube, until the desired plane is facing the user, at which time, a rotation tween is done and the plane is locked. To unselect the plane, the user can remove the cube from the table or rotate as to allow no fiducial markers to be recognized. The rotation of the cube is perceptible as it similar to the rotation of the planes. Additionally, the iconic representation supports the rotation by indicating the possible directions of rotation.

5.6 Content Destruction

An initial design for content destruction included placing the Container object in a specific area of the tabletop. Although this approach didn't require additional tokens, the use of a constant specific zone on the surface limited the amount off on-screen real state. Additionally, this design might lead to severe consequences if the user accidently placed a Container token in this area (deleting content).

Considering this first design, an alternative was considered in which two steps would be necessary to accomplish the destruction of Content. The first step was constituted of placing two tokens, the Shredders tokens, on the surface in order to make an area between the tokens. This allowed the user to specify the size of the area and to make this area temporary. The second step includes passing a Container token or placing the Container token in the area formed by the Shredder tokens. This approach uses a metaphor of passing paper (the information; the 3D Content) through a Shredder (the area between the tokens). This interaction procedure allows for the user to delete several Containers at the same time, by passing them in a queue or by increasing the size of the area and passing them all of them at the same time. Additionally, instead of dragging the Container token, the user can drag the Shredder tokens in a motion similar to silk printing a t-shirt.

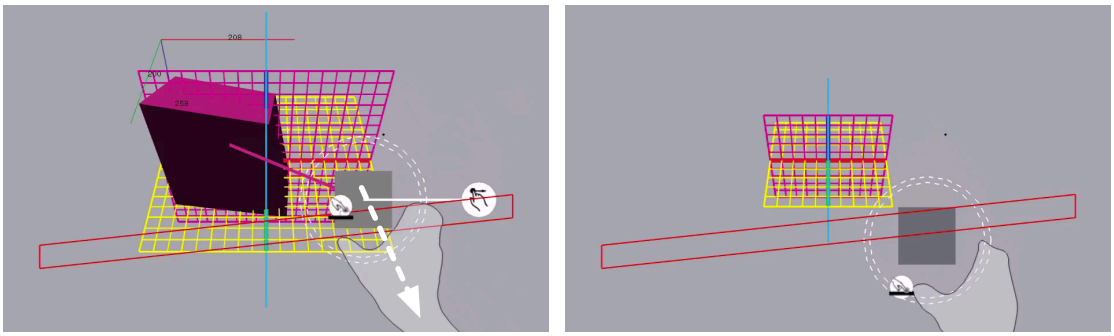


Figure 64 Content Destruction, from left to right: initial (two Shredder tokens create the red rectangular area; white arrow represents direction of movement of the Container token) and final state (when the token enters the red rectangular area, linked content is erased from the 3D Scene).

5.7 Other Interaction Possibilities

Apart from interactions necessary for use cases, an additional interaction possibility was studied using the angle made by the fingertip and the base of the finger. In this interaction, an Orbit token was used with purpose of orbiting all the linked contents of Containers on the surface, around the center of the plane selected. This interaction was initiated with hovering a finger above the center of the token and registering the angle

made by this finger. Over time, changes in this angle were reflected on screen, by the orbiting of 3D Content counter or clockwise. This interaction uses a metaphor of a wrench, where the fix point (bolt) is the token and moveable part (wrench) is the finger.

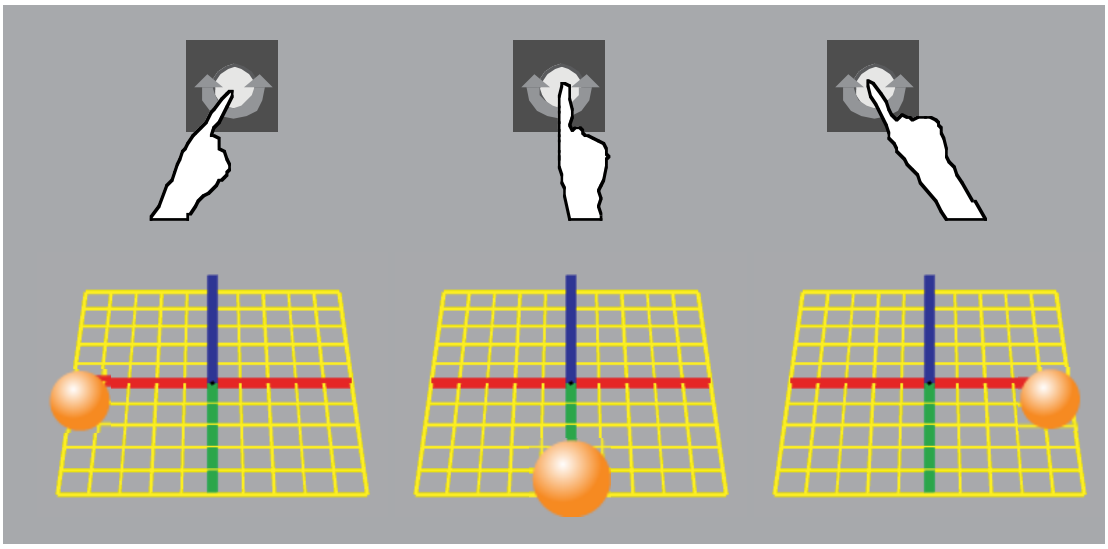


Figure 65 Orbiting Content. Orientation of the finger on the token is replicated in the 3D Scene with the rotation of 3D Content around the center of the plane.

Supplementary interactions possibilities were also researched and prototyped but not included in the final tCAD prototype, as they either were not relevant to the application domains, or they were not stable enough to be used. Among these interactions possibilities was an interaction alternative for the Boolean Mode, where the space above a certain area was divided into a variable number of layers (between 1 and 6). The height of the finger in this area was used to select one of the options. However, considering the limited space of interaction available (around 40 centimeters), using this interaction possibility proved to be hard when there were six layers, as not only the height of each layer was too small, but also, the Kinect noise often triggered the wrong selection. As the system is capable of knowing the height of an object, interaction possibilities like stacking were also considered, but discarded as they did not present a rational mapping to the system.

5.8 EARLY EVALUATION STUDY

With the purpose of being a preliminary validation of the system, namely of the interaction techniques implemented, a short observational study took place in late July.

5.8.1 STUDY PROTOCOL

The short observational study was carried out with 5 people (all university students in Computer Science, with none to limited experience in tabletops or Tangible User Interfaces) in separate 30-minute sessions. Each session started with a brief description of the tCAD application, a description of study structure and a brief demonstrating of the tokens and how they could be used. Observations and comments were recorded via a think-aloud verbal protocol.

After the description of the study, users were asked to complete a set of tasks of increasing complexity that relied on the different interaction techniques:

- Test the Camera Control – A simple interaction technique, as it only requires the users to understand the concept of a layer.
- Add a sphere to the scene using the three different Content Creation tokens – The transition of options on-screen, on-token and in-air increases in interaction complexity, as users are accustomed to only interacting with on-screen options.
- Link and unlink Content to a Container token – After dealing with concept of on-screen, on-token and in-air interaction, this task is meant to bond them (the Continuous Interaction Space).
- Manipulate 3D Content using Container token and Manipulation UI – After dealing with linking 3D Content and in-air interactions, users should be able to comprehend the Manipulation UI (especially the relative movement of finger upwards and downwards or dwell time in areas). This task also serves to introduce the users to the plane selection cube.
- Erase Content – This task permitted users to test the shredder metaphor.

Subsequently, they were given the rest of the thirty minutes session to use the system without restrictions (with the notable exception of the Orbit token and Save token which were not deployed for the session). In this portion of the session, users were asked to choose their favorite Content Creation UI and Manipulation UI for this portion of study, therefore showing their preference for interactions.

Figure 66 Users during the early evaluation study. Please refer to Annex F, for the session's Observation tables.

5.8.2 RESULTS

The results were broadly positive. All users completed the tasks and quickly picked up the interaction techniques, being positively engaged by the systems metaphor and mappings.

The relationship between physical tokens and virtual objects was readily accepted, as the metaphors used were understandable and supported by the color scheme significance.

Due to the large volume of interaction techniques tested, all users experienced some confusion regarding whether particular actions were triggered by pressing on the surface, or pressing on the token or by movements in space. For example, some users in the dwell Manipulation UI tapped on the screen as if the graphical representation of the area was a button. Although this tapping still accomplished the goal at hand (the tapping was still recognized as being dwell), users were not able to distinguish that on-screen options were represented by smaller on-screen circles, while in-air options were represented by big rectangular or half-circle areas. This suggests the range of interactions with objects in an application should be limited, to reduce the memory burden on users, or supported via clear graphical affordances or instructions.

When faced with the ability to choose the preferred tokens, choice seemed to be subjective to the person and not to the interaction style. While some users preferred on-screen options in the first Content Creation UI (as more options were visible at the same time, even tough it took more space on screen), other users preferred the in-air options (as they found the menu style more enjoyable). As for the Manipulation UI, all the users

preferred the dwell Container as they felt it was more responsive that the alternative option.

During the free-play portion of the study, most users focused their attention on other Content Creation Modes besides the ones already used, stating their preference for the Contour Mode as they found it more "creatively freeing". Only two users tested the Boolean Mode, but found it understandable. The content destruction procedure was also comprehensible.

One significant observation relates to the use of cube selection plane. Firstly, some users often forgot to choose a plane before trying to manipulate content. This suggests that the process of manipulating a plane should not incorporate this superfluous step. A possible solution for this would be to use the camera to automatically choose a plane by comparing the current camera position to possible planes. This seems a natural progression for this work (as it would diminish the dependence of single function tokens) and should be addressed in future iterations of the project.

As for the physical setup, users for the most part did not seem to be distracted by the physical setup while interacting with the objects. One user had difficulties pressing options of the table, as she would press with the finger at 90 degrees angle to the surface (where the Kinect camera could not detect the finger as it was covered by the rest of the hand). Arguably, this type of behavior may be caused by the users previous experience with capacitive screen. After realizing this limitation, the user corrected the behavior. This system could allow for this type of gesture if it also took into account touch events from reacTIVision, as in this type of gestures, the hand may occlude the Kinect's IR allowing for the recognition of touch by the touch tracking software.

Some of the users exceeded the field of view or sensing range of the depth camera while performing gestures in free space. This can be attributed to the lack of feedback or information highlighting those boundaries and could be solved by placing the Kinect at a higher place, to achieve a bigger interaction space (although sacrificing resolution). Alternatively, the use of different lenses (Zoom Kinect lenses [URL11]) could modify the interaction boundaries of the space above the tabletop.

In summary, results were positive. Even though the study was short, it was able to validate the core principles of tCAD application, while successfully exploring the interaction space created by the physical setup.

6 CONCLUSION

6.1 SUMMARY

This thesis approached the issue of depth on tabletops by creating a simple setup comprised of a Kinect depth sensor and a tabletop computer. The weaving of depth information and marker tracking by reacTIVision allowed for the creation of a Continuous Interaction Space, where tangible objects not only presented the traditional "move" and "rotate" actions, but also were capable of sensing hovering and touching. The interaction richness of the space above the tabletop was used to explore interaction possibilities for Tangible User Interfaces.

As a proof of concept for the physical setup, a TUI application was developed: tCAD is an application for 3D modeling and manipulation grounded on the use of tangible tokens interweaved with depth sensing, leveraging the Continuous Interaction Space to maintain a physical parallel between inputs to the system and the changes to linked virtual objects.

6.2 CONTRIBUTIONS

Most works exploring depth on tabletops emphasize features and viability of sensing setups and are normally linked to complex bespoke hardware setups. The difficulty of creating and replicating these sensing setups introduces a barrier to their use in research and in commercial avenues.

The first contribution of this thesis lies in dealing with the issue of bespoke hardware setups by updating an existing setup with commercial depth sensing, in order to allow for novel interaction research. This process of updating adds value to past works, by extending their viability and introducing new interaction possibilities.

The union of the Kinect and a tabletop computer was explored in an iterative process to delineate successful (and unsuccessful) strategies for the incorporation of depth on tabletops. This exploration process dealt with plane detection of the tabletop surface, tabletop corner detection, perspective correction, finger tracking, calibration processes and assimilation of depth, finger and fiducial marker information into a consistent model.

Based on the desire to test the physical setup, the dissertation's second contribution arouse from the development and implementation of tCAD, a proof of concept prototype. This prototype fused the domain of tabletop tangibility with the domain of 3D Manipulation, by using real objects as surrogates for virtual objects. This manipulation metaphor is direct and comprehensible as it permits users to think and act in real world dimensions, instead of virtual coordinate space. This dissertation documents the development and implementation process of tCAD through both HCI and TUI methods.

This implementation also allowed the exploration of techniques permitted by the Continuous Interaction Space. As such, several interaction possibilities were designed and implemented. Afterwards, a small early user study was carried out to identify if the interaction techniques applied to tCAD were intuitive and easy to pick up.

6.3 Limitations and Future Work

The main factor limiting this work is current technology as it introduces unstableness to the system. Case in point, Kinect's limitations in terms of noise and resolution prevents an easier configuration process and limits the efficacy of finger tracking process. An improvement in the resolution of the Kinect would lead to better distinction of fingers, while a better solution to Kinect's noise problem, would stabilize depth values and offer more accurate results. An improvement in Kinect range would benefit the physical setup, as it would increase the Continuous Interaction Space.

Due to the finger tracking process adopted, fingers touching the tabletop that are covered by hand are not recognized. As mentioned before, these fingers are detectable by the reacTIVision software as they are not affected by Kinect's IR. Therefore, this limitation should be addressed in future work, by intertwining information from reacTIVision (both markers and fingers, instead of just markers which it does currently) into the application.

Another limitation of this system that should be addressed is the shape of tokens. Currently, tokens that are touchable or "hoverable" are square and have same height, relying on iconic representations to offer meaning. As seen in many TUIs (e.g. [P95]), shape of token is often utilized do not only distinguish the tokens between themselves, but also to affect how a token is used. Therefore, limiting token expressiveness (shape, material, size, etc.) in an application, limits interaction possibilities and does not take advantage of tokens affordances.

A future implementation of tCAD should review tokens and eliminate unnecessary ones, since these clutter the tabletop and overload the users cognitive model. Namely, the cube token is an unnecessary token, as scene plane could be estimated from the current position of the camera. The removal of this token would not only simplify current interactions like the manipulation of 3D Content, but would increase the manipulation power of tokens as more planes would be available.

One aspect considered, but not implemented, was the detection of objects while in air. In this type of interaction, tokens on the surface would be tracked when the user lifted them on the table, allowing use of the token's height as input. Although this interaction would be in line with the interaction metaphor for tCAD, previous attempts at this interaction [KNF12] seemed to be too unstable for use.

Lastly, the space above the tabletop presents a rich interaction space with almost limitless possibilities. Understandably, not all of these possibilities could be implemented in this project. One promising direction that could be followed would be the study of proxemics on tabletops. For instance, tokens could be implemented to have different actions considering which hand touched them (left hand to go forwards in the manipulation menu and right hand to go backwards, as an example). Alternatively, tokens could be assigned to a user when added to the tabletop, refusing manipulation by other users.

Future work on this area should be focused on:

- Tackling current applications limitations, namely, limitations that focuses on the accuracy of finger and marker tracking. For instance, one possible improvement in depth sensing technology is the use of custom lenses (Zoom [URL11]) to better support short range sensing. This type of custom lenses not only increases the range for short distances, but also increases the peripheral vision, which allows for better proxemics interactions (by not only tracking fingers and hands but also users).
- Continuing work on interaction techniques by developing new techniques in other application domains.
- Conducting a rigorous user study to demonstrate (or disprove) the value of depth sensing on tabletops.

6.4 Final Remarks

The work presented in this thesis has a strong experimental component by describing technical details on how depth sensing can be entrenched in tabletop computers. This dissertation pretends to connect how the extension of tabletops can advance Tangible User Interfaces, namely the fiducial tracking paradigm, by refreshing the interaction possibilities afforded by objects.

This dissertation also presents a strong conceptual component with the development of the TUI tCAD, a simple 3D Modeling application, following both HCI and TUI-specific methods.

REFERENCES

[A79] Aish, R. 3D input for CAAD systems. *Computer Aided Design* 11, 2 (1979), 66–70.

[AD07] Abdelmohsen, S. M. and Do, E. Y.-L. TangiCAD: Tangible Interface for Manipulating Architectural 3D Models. In *Proc. CAADRIA '07*, CAADRIA (2007), 29-36.

[ADL10] Aguerreche L., Duval, T. and Lécuyer, A. Reconfigurable tangible devices for 3D virtual object manipulation by single or multiple users. In *Proc. VRST '10*, ACM Press (2010), 227-230.

[AGW+11] Annett, M., Grossman, T., Wigdor, D. and Fitzmaurice, G. Medusa: A Proximity-Aware Multi-touch Tabletop. In *Proc. UIST '11*, ACM Press (2011), 337-346.

[AN84] Aish, R. and Noakes, P. Architecture without numbers. *Computer Aided Design* 16, 6 (1984), 321–328.

[ASA10] Aliakseyeu, D., Subramanian, S. and Alexander J. Supporting Atomic User Actions on the Table. In Müller-Tomfelde, C. *Tabletops - Horizontal Interactive Displays*. Springer Publishing Company, London, 2010, 223-247.

[ATF12] Au, O. K.-C., Tai, C.-L. and Fu, H. Multitouch Gestures for Constrained Transformation of 3D Objects. *Computer Graphics Forum* 31, 2.3 (2012), 651–660.

[B90] Buxton, W. A three-state model of graphical input. In *Proc. INTERACT '90*, North-Holland Publishing Co. (1990), 449-456.

[BBE+02] Bellotti, V., Back, M., Edwards, W., Grinter, R., Henderson, A. and Lopes, C. Making sense of sensing systems: Five questions for designers and researchers. In *Proc. CHI '02*, ACM Press (2002), 415–422.

[BBR10] Baudisch, P., Becker, T. and Rudeck, F. Lumino : Tangible Blocks for Tabletop Computers Based on Glass Fiber Bundles. In *Proc. CHI '10*, ACM Press (2010), 1165–1174.

[BD97] Brave, S. and Dahley, A. inTouch: A Medium for Haptic Interpersonal Communication. In *Proc. CHI EA '97*, ACM Press (1997), 363–364.

[BDL+11] Blackshaw, M., DeVincenzi, A., Lakatos, D., Leithinger, D. and Ishii, H. Recompose. In *Proc. CHI EA '11*, ACM Press (2011), 1237-1242.

[BHG+08] Block, F., Haller, M., Gellersen, H., Gutwin, C. and Billinghurst, M. VoodooSketch: Extending interactive surfaces with adaptable interface palettes. In *Proc. TEI '08*, ACM Press (2008), 55–58.

[BHO09] Bartindale, T., Hook, J. and Olivier, P. Media Crate: Tangible Live Media Production Interface. In *Proc. TEI '09*, ACM Press (2009), 255–262.

[BIF04] Benko, H., Ishak, E. W. and Feiner, S. Collaborative mixed reality visualization of an archaeological excavation. In *Proc. ISMAR '04*, IEEE (2004), 132–140.

[BM92] Besl, P. J. and McKay, N. D. A Method for Registration of 3-D Shapes. *IEEE Trans. Pattern Anal. Mach. Intell.* 14, 2 (1992), 239-256.

[BMG10] Ballendat, T., Marquardt, N. and Greenberg, S. Proxemic Interaction: Designing for a Proximity and Orientation-Aware Environment. In *Proc. ITS '10*, ACM Press (2010), 121-130.

[BSK+05] Benford, S., Schnädelbach H., Koleva B., Anastasi, R., Greenhalgh, C., Rodden, T., Green, J., Ghali, A., Pridmore, T., Gaver, B., Boucher, A., Walker, B., Pennington, S., Schmidt, A., Gellersen, H. and Steed, A. Expected, sensed and desired: A framework for designing sensing-based interaction. *ACM Transactions on Computer-Human Interaction (TOCHI)* 12, 1 (2005), 3–30.

[BSP+93] Bier, E. A., Stone, M. C., Pier, K., Buxton, W. and DeRose, T. D. Toolglass and magic lenses: the see-through interface. In *Proc. SIGGRAPH '93*, ACM Press (1993), 73-80.

[BWB06] Benko, H., Wilson, A. and Baudisch, P. Precise selection techniques for multi-touch screens. In *Proc. CHI '06*, ACM Press (2006), 1263–1272.

[CDJ+02] Camarata, K., Do, E. Y.-L, Johnson, B. R. and Gross, M. D. Navigational blocks: Navigating information space with tangible media. In *Proc. IUI '02*, ACM Press (2002), 31–38.

[CLS06] Chung, H., Lee, C.J, and Selker, T., Lover's cups: Drinking interfaces as new communication channels. In *Proc. CHI '06*, ACM Press (2006), 375-380.

[CMN+10] Correia, N., Mota, T., Nóbrega, R., Silva, L. and Almeida, A. A multi-touch tabletop for robust multimedia interaction in museums. In *Proc. ITS '10*, ACM Press (2010), 117-120.

[CMR+12] Chan L., Müller S., Roudaut A. and Baudisch, P. CapStones and ZebraWidgets. In *Proc. CHI '12*, ACM Press (2012), 2189-2192.

[CPS+97] Czernuszenko, M., Pape, D., Sandin, D., DeFanti, T., Dawe, G. L. and Brown, M. D. The ImmersaDesk and infinity wall projection-based virtual reality displays. *SIGGRAPH Computer Graphics* 31, 2 (1997), 46–49.

[CRK01] Chang, A., Resner, B., Koerner, B., Wang, X. and Ishii, H. LumiTouch: An emotional communication device. In *Proc. CHI EA '01*, ACM Press (2001), 313–314.

[CRR08] Couture, N., Rivière, G. and Reuter, P. GeoTUI: A tangible user interface for geoscience. In *Proc. TEI '08*, ACM Press (2008), 89-96.

[CWP99] Cohen, J., Withgott, M., and Piernot, P. Logjam: A tangible multi-person interface for video logging. In *Proc. CHI '99*, ACM Press (1999), 128–135.

[D01] Dourish, P. *Where the Action Is: The Foundations of Embodied Interaction*. MIT Press (2001).

[DL01] Dietz, P. and Leigh, D. DiamondTouch: A multi-user touch technology. In *Proc. UIST '01,* ACM Press (2001), 219–226.

[EB09] Edge, D. and Blackwell, A. Peripheral tangible interaction by analytic design. In *Proc. TEI '09*, ACM Press (2009), 69–76.

[EO10] Esteves, A. and Oakley, I. Mementos: A Tangible Interface Supporting Travel. In *Proc. NordiCHI '10*, ACM Press (2010), 643-646.

[EO11] Esteves, A. and Oakley I. Eco Planner: A Tabletop System for Scheduling Sustainable Routines. In *Proc. Work-in-Progress Worskshop TEI '11*, ACM Press (2011), 139-144.

[F95] Frazer, J. *An Evolutionary Architecture* (Themes). Architectural Association Publications London (1995).

[F96] Fitzmaurice, G. W. *Graspable User Interfaces*. Dissertation, Computer Science, University of Toronto, Canada (1996).

[FB97] Fitzmaurice, G. W. and Buxton, W. An empirical evaluation of graspable user interfaces: Towards specialized, space-multiplexed input. In *Proc. CHI '97*, ACM Press (1997), 43–50.

[FF80] Frazer, J. and Frazer, P. Intelligent physical three-dimensional modeling systems, in *Proc. Computer Graphics '80*, Online Publications (1980), 359–370.

[FF82] Frazer, J. and Frazer, P. Three-dimensional data input devices. *Computer Graphics in the Building Process*, Washington (1982), 409-416.

[FI12] Follmer, S. and Ishii, H. KidCAD: digitally remixing toys through tangible tools. In *Proc. CHI '12*. ACM Press (2012), 2401-2410.

[FIB95] Fitzmaurice, G. W., Ishii, H. and Buxton, W. Bricks: Laying the foundations for graspable user interfaces. In *Proc. CHI '95*, ACM Press (1995), 442–449.

[FMS93] Feiner, S., Macintyre, B. and Seligmann, D. Knowledge-based augmented reality. *Communications of the ACM* 36, 7 (1993), 53–62.

[GB05] Grossman, T. and Balakrishnan, R. The bubble cursor: Enhancing target acquisition by dynamic resizing of the cursor's activation area. In *Proc. CHI '05*, ACM Press (2005), 281–290.

[GB06] Grossman, T. and Balakrishnan, R. The design and evaluation of selection techniques for 3D volumetric displays. In *Proc. UIST '06*, ACM Press (2006), 3–12.

[GK99] Greenberg, S. and Kuzuoka, H. Using digital but physical surrogates to mediate awareness, communication and privacy in media spaces. *Personal Technologies* 3, 4 (1999), 182-198.

[GMB+11] Greenberg, S., Marquardt, N., Ballendat, T., Diaz-Marino, R. and Wang, M. Proxemic interactions: the new ubicomp? *Interactions* 18, 1 (2011), 42-50.

[GO07] Godlewski, J. and Obarowska, M. Organic light emitting devices. *Opto-Electronics Review* 15, 4 (2007), 179–183.

[GSH+07] Girouard, A., Solovey, E. T, Hirshfield, L. M., Ecott S., Shaer, O. and Jacob, R. J. K. Smart blocks: A tangible mathematical manipulative. In *Proc. TEI '07*, ACM Press (2007), 183–186.

[GW10] Grossman, T. and Wigdor, D. On, Above, and Beyond: Taking Tabletops to the Third Dimension. In Müller-Tomfelde, C. *Tabletops - Horizontal Interactive Displays*. Springer Publishing Company, London, 2010, 277-301.

[GWB04] Grossman, T., Wigdor, D. and Balakrishnan, R. Multi-finger gestural interaction with 3d volumetric displays. In *Proc. UIST '04*, ACM Press (2004), 61–70.

[H04] Huang, C. Not just intuitive: Examining the Basic Manipulation of Tangible User Interfaces. In *Proc. CHI EA '04*, ACM Press (2004), 1387-1390.

[H05] Han, J. Y. Low-cost multi-touch sensing through frustrated total internal reflection. In *Proc. UIST '05*, ACM Press (2005), 115–118.

[H12] Hornecker, E. Beyond affordance: tangibles' hybrid nature. In *Proc. TEI '12*, ACM Press (2012), 175-182.

[H66] Hall, E.T. *The Hidden Dimension*. Doubleday, 1966.

[HB06] Hornecker, E. and Buur, J. Getting a grip on tangible interaction: a framework on physical space and social interaction. In *Proc. CHI '06,* ACM Press (2006), 437-446.

[HE04] Hoven, E. and Eggen, B. Tangible computing in everyday life: Extending current frameworks for tangible user interfaces with personal objects. In *Proc. EUSAI '04*, Springer-Verlag (2004), 230–242.

[HIW+09] Hilliges, O., Izadi, S., Wilson, A. D., Hodges, S., Garcia-Mendoza, A. and Butz, A. Interactions in the air. In *Proc. UIST '09*, ACM Press (2009), 139-148.

[HKK08] Hofer, R., Kunz, A. and Kaplan, P. MightyTrace: Multiuser tracking technology on LC-displays. In *Proc. CHI '08*, ACM Press (2008), 215–218.

[HLL08] Hinske, S., Langheinrich, M. and Lampe, M. Towards guidelines for designing augmented toy environments. In *Proc. DIS '08*, ACM Press (2008), 78–87.

[HMT03] Hiraoka, S, Miyamoto, I. and Tomimatsu, K. Behind touch, a text input method for mobile phones by the back and tactile sense interface. In *Proc. IPSJ Interaction '03*, Information Processing Society of Japan (2003), 131–138.

[HNK09] Hofer, R., Naeff, D. and Kunz, A. FLATIR: FTIR multi-touch detection on a discrete distributed sensor array. In *Proc. TEI '09*, ACM Press (2009), 317–322.

[HPG+94] Hinckley, K., Pausch, R., Goble, J. and Kassel, N. Passive real-world interface props for neurosurgical visualization. In *Proc. CHI '94*, ACM Press (1994), 452–458.

[HRL99] Holmquist, L. E., Redström, J. and Ljungstrand, P. Token-based access to digital information. In *Proc. HUC '99*, Springer-Verlag (1999), 234–245.

[HSJ08] Horn, M. S., Solovey, E. T., and Jacob, R. J. K. Tangible programming for informal science learning: Making TUIs work for Museums. In *Proc. IDC '08*, ACM Press (2008), 194–201.

[HVT05] Holman, D., Vertegaal, R. and Troje, N. PaperWindows: Interaction techniques for digital paper. In *Proc. CHI '05,* ACM Press (2005), 591–599.

[I08] Ishii, H. The tangible user interface and its evolution. *Communications of the ACM* 51, 6 (2008), 32-36.

[IU97] Ishii, H. and Ullmer, B. Tangible bits: Towards seamless interfaces between people, bits and atoms. In *Proc. CHI '97*, ACM Press (1997), 234–241.

[J02] Rekimoto, J. SmartSkin: An infrastructure for freehand manipulation on interactive surfaces. In *Proc. CHI '02*, ACM Press (2002), 113–120.

[J08] Jordà, S. On stage: The reactable and other musical tangibles go real. *International Journal of Arts and Technology (IJART)* 1, 3 (2008), 268–287.

[JGA+07] Jordà, S., Geiger, G., Alonso, M. and Kaltenbrunner, M. The reacTable: Exploring the synergy between live music performance and tabletop tangible interfaces. In *Proc. TEI '07,* ACM Press (2007), 139–146.

[JGH+08] Jacob, R. J. K., Girouard, A., Horn, M., Hirshfield, L. M., Shaer, O., Solovey, E. T. and Zigelbaum, J. Reality-based interaction. In *Proc. CHI '08*, ACM Press (2008), 201-210.

[JMG+11] Jota, R., Marquardt, N., Greenberg, S. and Jorge, J. The Continuous Interaction Space: Interaction Techniques Unifying Touch and Gesture On and Above a Digital Surface. In *Proc. INTERACT '11*, Springer-Verlag (2011), 461-476.

[K03] Kruchten, P. *The Rational Unified Process: An Introduction* (3 ed.). Addison-Wesley Longman Publishing Co., Boston, 2003.

[K09] Kirsh, D. Problem solving and situated cognition in Robbins, P. and Aydede, M. *The Cambridge Handbook of Situated Cognition*, Cambridge University Press (2009), 264-306.

[KB07] Kaltenbrunner, M. and Bencina, R. reacTIVision: A computer-vision framework for table-based tangible interaction. In *Proc. TEI '07*, ACM Press (2007), 69–74.

[KBI+00] Kato H., Billinghurst, M., Imamoto, K. and Tachibana K. Virtual object manipulation on a table-top AR environment. In *Proc. ISAR '00*, IEEE (2000), 111–119.

[KCS+03] Kruger, R., Carpendale, S., Scott, S. and Greenberg, S. How People Use Orientation on Tables: Comprehension, Coordination and Communication. In *Proc. GROUP '03*, ACM Press (2003), 396-378.

[KCS+05] Kruger, R., Carpendale, S., Scott, S. and Tang, A. Fluid integration of rotation and translation. In *Proc. CHI '05*, ACM Press (2005), 601–610.

[KCZ11] Kazi, R. H., Chua, K. C., Zhao, S., Davis, R. and Low, K.-L. SandCanvas: A Multi-touch Art Medium Inspired by Sand Animation. In *Proc. CHI '11*, ACM Press (2011), 1283-1292.

[KF10] Kunz, A. and Fjeld, M. From Table–System to Tabletop: Integrating Technology into Interactive Surfaces. In Müller-Tomfelde, C. *Tabletops - Horizontal Interactive Displays*. Springer Publishing Company, London, 2010, 51-70.

[KHT06] Klemmer, S. R., Hartmann, B. and Takayama, L. How bodies matter: Five Themes for Interaction Design. In *Proc. DIS '06*, ACM Press (2006), 140-149.

[KNF+01] Klemmer, S. R., Newman, M. W., Farrell, R., Bilezikjian, M. and Landay, J. A. The designers outpost: a tangible interface for collaborative web site design. In *Proc. UIST '01*, ACM Press (2001), 1-10.

[KNF12] Klompmaker, F., Nebe, K., and Fast, A. dSensingNI: a framework for advanced tangible interaction using a depth camera. In *Proc. TEI'12*, ACM Press (2012), 217-224.

[KST+09] Kirk, D., Sellen, A., Taylor, S., Villar, N. and Izadi., S. Putting the physical into the digital: issues in designing hybrid interactive surfaces. In *Proc. BCS-HCI '09*. British Computer Society (2009), 35-44.

[KWD05] Kabisch, E., Williams, A. and Dourish, P. Symbolic objects in a networked gestural sound interface. In *Proc. CHI EA '05*, ACM Press (2005), 1513–1516.

[LBB+07] Loenen, E., Bergman, T., Buil, V., Gelder, K., Groten, M., Hollemans, G., Hoonhout, J., Lashina, T. and Wijdeven, S. EnterTaible: A solution for social gaming experiences. In *Tangible Play: Research and Design for Tangible and Tabletop Games, Workshop*, IUI Conference (2007), 16–19.

[LHY+08] Leitner, J., Haller, M., Yun, K., Woo, W., Sugimoto, M., and Inami, M. IncreTable, a mixed reality tabletop game experience. In *Proc. ACE '08*, ACM Press (2008), 9–16.

[LLD+11] Leithinger, D., Lakatos, D., DeVincenzi, A., Blackshaw, M. and Ishii, H. Direct and gestural interaction with relief. In *Proc. UIST '11*, ACM Press (2011), 541-548.

[LPS+06] Liu, J., Pinelle, D., Sallam, S., Subramanian, S. and Gutwin, C. TNT: Improved rotation and translation on digital tables. In *Proc. GI '06*, Canadian Information Processing Society (2006), 25–32.

[MDB+11] Marquardt, N., Diaz-Marino, R., Boring, S. and Greenberg, S. The Proximity Toolkit: Prototyping Proxemic Interactions in Ubiquitous Computing Ecologies. In *Proc. UIST '11*, ACM Press (2011), 315-326.

[MF10] Müller-Tomfelde, C. and Fjeld, M. Introduction: A Short History of Tabletop Research, Technologies, and Products. In Müller-Tomfelde, C. *Tabletops - Horizontal Interactive Displays*. Springer Publishing Company, London, 2010, 1-25.

[MF12] Muller-Tomfelde, C. and Fjeld, M. Tabletops: Interactive Horizontal Displays for Ubiquitous Computing, *Computer* 45, 2 (2012), 78-81.

[MF99] Mackay, W. E. and Fayard, A.-L. Designing interactive paper: Lessons from three augmented reality projects. In *Proc. IWAR '98*, A. K. Peters, Ltd. (1999), 81-90.

[MH04] Moscovich, T. and Hughes, J. F. Navigating documents with the virtual scroll ring. In *Proc. UIST '04*, ACM Press (2004), 57–60.

[MHM+08] Marshall, P., Hornecker, E., Morris, R., Dalton, N. S. and Rogers, Y. When the fingers do the talking: A study of group participation with varying constraints to a tabletop interface. In *Proc. TABLETOP '08*, IEEE (2008), 33–40.

[MO98] MacKenzie, I. S. and Oniszczak, A. A comparison of three selection techniques for touchpads. In *Proc. CHI '98*, ACM Press (1998), 336–343.

[MPW08] Maquil, v., Psik, T. and Wagner, I. The ColorTable: A design story. In *Proc. TEI '08*, ACM Press (2008), 97–104.

[MR97] Matsushita, N. and Rekimoto, J. HoloWall: Designing a finger, hand, body, and object sensitive wall. In *Proc. UIST '97*, ACM Press (1997), 209–210.

[MRG07] Mugellini, E., Rubegni, E., Gerardi, S. and Khaled O. A. Using personal objects as tangible interfaces for memory recollection and sharing. In *Proc. TEI '07*, ACM Press (2007), 231–238.

[NNG03] Newton-Dunn, H., Nakano, H. and Gibson, J. Block Jam: A tangible interface for interactive music. In *Proc. NIME '03*, 2003, 170–177.

[NPG+10] Nacenta, M. A., Pinelle, D., Gutwin, C. and Mandryk, R. Individual and Group Support in Tabletop Interaction Techniques. In Müller-Tomfelde, C. *Tabletops - Horizontal Interactive Displays*. Springer Publishing Company, London, 2010, 303-334.

[NPS+01] Nacenta, M. A., Pinelle, D., Stuckel, D. and Gutwin, C. The effects of interaction technique on coordination in tabletop groupware. In *Proc. of GI '07*, ACM Press (2007), 191–198.

[OF04] O'Malley, C. and Fraser, D. S., Literature review in learning with tangible technologies. NESTA futurelab Report 12, Bristol (2004).

[OW08] Olwal, A. and Wilson, A. D. SurfaceFusion: unobtrusive tracking of everyday objects in tangible user interfaces. In *Proc. GI '08*, ACM Press (2008), 235-242.

[P07] Petersen, M. G. Squeeze: Designing for playful experiences among co-located people in homes. In *Proc. CHI EA '07*, ACM Press (2007), 2609–2614.

[P76] Perlman, R. Using Computer Technology to Provide a Creative Learning Environment for Preschool Children. *MIT Logo Memo* 24, 1976.

[P95] Poynor R. The hand that rocks the cradle. *ID Magazine*, May/June (1995), 60–65.

[PBN+09] Pinelle, D., Barjawi, M., Nacenta, M. and Mandryk, R. An evaluation of coordination techniques for protecting objects and territories in tabletop groupware. In *Proc. CHI '09*, ACM Press (2009), 2129–2138.

[PH09] Pedersen, E. W. and Hornbæk, K. mixiTUI: A tangible sequencer for electronic live performances. In *Proc. TEI '09,* ACM Press (2009), 223–230.

[PHH11] Pyryeskin, D., Hancock, M. and Hoey, J. Extending Interactions into Hoverspace Using Reflected Light. In *Proc. ITS '11*, ACM Press (2011), 262-263.

[PI07] Patten, J. and Ishii, H. Mechanical constraints as computational constraints in tabletop tangible interfaces. In *Proc. CHI '07*, ACM Press (2007), 809-818.

[PIH+01] Patten, J., Ishii, H., Hines, J. and Pangaro, G. SenseTable: A wireless object tracking platform for tangible user interfaces. In *Proc. CHI '01*, ACM Press (2001), 253–260.

[PNG+08] Pinelle, D., Nacenta, M., Gutwin, C. and Stach, T. The effects of co-present embodiments on awareness and collaboration in tabletop groupware. In *Proc. GI '08*, Canadian Information Processing Society (2008), 1–8.

[PNO07] Poupyrev, I., Nashida, T. and Okabe, M. Actuation and tangible user interfaces: the Vaucanson duck, robots, and shape displays. In *Proc. TEI '07*, ACM Press (2007), 205–212.

[PRI02] Piper, B., Ratti, C. and Ishii, H. Illuminating clay: A 3-D tangible interface for landscape analysis. In *Proc. CHI '02*, ACM Press (2002), 355–362.

[PWS88] Potter, R. L., Weldon, L. J. and Shneiderman, B. Improving the accuracy of touch screens: An experimental evaluation of three strategies. In *Proc. CHI '88*, ACM Press (1988), 27–32.

[R97] Rekimoto, J. Pick-and-drop: A direct manipulation technique for multiple computer environments. In *Proc. UIST '97*, ACM Press (1997), 31–39.

[RC99] Ryokai, K. and Cassell, J. StoryMat: A play space for collaborative storytelling. In *Proc. CHI '99*, ACM Press (1999), 272–273.

[RFK+98] Rauterberg, M., Fjeld, M., Krueger, H., Bichsel, M., Leonhardt, U. and Meier, M. BUILD-IT: A planning tool for construction and design. In *Proc. CHI EA '98*, ACM Press (1998), 177-178.

[RI02] Patten, J., Recht, B. and Ishii, H. Audiopad: A tag-based interface for musical performance. In *Proc. NIME '02*, National University of Singapore (2002), 1-6.

[RIS+03] Rekimoto, J., Ishizawa, T., Schwesig, C. and Oba, H. PreSense: Interaction Techniques for Finger Sensing Input Devices. In *Proc. UIST '03*, ACM Press (2003), 203-212.

[RM00] Ren, X. and Moriya, S. Improving selection performance on pen-based systems: A study of pen-based interaction for selection tasks. *ACM Transactions on Computer-Human Interaction (TOCHI)* 7, 3 (2000), 384–416.

[RM06] Rogers, Y. and Muller, H. A framework for designing sensor-based interactions to promote exploration and reflection in play. *International Journal of Human Computer Studies* 64, 1 (2006), 1–14.

[RMB+98] Resnick, M., Martin, F., Berg, R., Borovoy, R., Colella, V., Kramer, K. and Silverman, B. Digital Manipulatives: New Toys to Think With. In *Proc. CHI '98*, ACM Press (1998), 281–287.

[RMI04] Ryokai, K., Marti, S. and Ishii, H. I/O brush: Drawing with everyday objects as ink. In *Proc. CHI '04*, ACM Press (2004), 303–310.

[RPI04] Rafflee, H. S., Parkes, A. J. and Ishii, H. Topobo: A constructive assembly system with kinetic memory. In *Proc. CHI '04*, ACM Press (2004), 647–654.

[RUI01] Rekimoto, J., Ullmer, B. and Oba, H. DataTiles: A modular platform for mixed physical and graphical interactions. In *Proc. CHI '01*, ACM Press (2001), 269–276.

[RWL+01] Raskar, R., Welch, G., Low, K.-L. and Bandyopadhyay, D. Shader Lamps: Animating real objects with image based illumination. In *Proc. EGWR '01,* Eurographics Association (2001), 89-102.

[S00] Seitz, J. The bodily basis of thought. *New Ideas in Psychology* 18, 1 (2000), 23-40.

[S08] Saffer, D. *Designing gestural interfaces: Touchscreens and interactive devices*. O'Reilly Media, Inc., North Sebastopol, 2008.

[SAL06] Subramanian, S., Aliakseyeu, D., and Lucero, A. Multi-layer interaction for digital tables. In *Proc. UIST '06*, ACM Press (2006), 269-272.

[SB08] Sheridan, J. G. and Bryan-Kinns N. Designing for performative tangible interaction. *International Journal of Arts and Technology (IJART)* 1, 3 (2008), 288–308.

[SBB97] Schäfer, K., Brauer, V. and Bruns, W. A new approach to human-computer interaction—synchronous modelling in real and virtual spaces. In *Proc. DIS '97*, ACM Press (1997), 335-344.

[SC10] Scott, S. D. and Carpendale, S. Theory of Tabletop Territoriality. In Müller-Tomfelde, C. *Tabletops - Horizontal Interactive Displays*. Springer Publishing Company, London, 2010, 1-25.

[SG09] Saul, G. and Gross, M. D. Co-designed paper devices. In *Programming Reality: From Transitive Materials to Organic User Interfaces, a CHI '09 Workshop*, 2009.

[SGH12] Song, P., Goh, W. B, Hutama, W., Fu, C.-W. and Liu, X. A handle bar metaphor for virtual object manipulation with mid-air interaction. In *Proc. CHI '12*, ACM Press (2012), 1297-1306.

[SH10] Shaer, O. and Hornecker, E. Tangible User Interfaces: Past, Present, and Future Direction. *Foundations and Trends of Human-Computer Interaction* 3, 1–2 (2010), 1-137.

[SHS+99] Singer, A., Hindus, D., Stifelman, L. and White, S. Tangible progress: Less is more in somewire audio spaces. In *Proc. CHI '99*, ACM Press (1999), 104–111.

[SJ08] Schiettecatte, B. and Vanderdonckt, J. AudioCubes: A distributed cube tangible interface based on interaction range for sound design. In *Proc. TEI '08*, ACM Press (2008), 3–10.

[SJ09] Shaer, O and Jacob, R. J. K. A specification paradigm for the design and implementation of tangible user interfaces. *ACM Transactions on Computer-Human Interaction (TOCHI)* 16, 4 (2009), Article 20.

[SK93] Suzuki, H. and Kato, H. AlgoBlock: A tangible programming language, a tool for collaborative learning. In *Proc. Eurologo '93*, (1993), 297–303.

[SK95] Suzuki, H. and Kato, H. Interaction-level support for collaborative learning: AlgoBlock — An open programming language. In *Proc. of CSCL '95*, (1995), 349-355.

[SLC+04] Shaer, O., Leland, N., Calvillo-Gamez, E. H. and Jacob R. The TAC paradigm: Specifying tangible user interfaces. *Personal and Ubiquitous Computing* 8, 5 (2004), 359–369.

[STR+04] Stringer, M., Toye, E. F., Rode, J. and Blackwell, A. Teaching rhetorical skills with a tangible user interface. In *Proc. IDC '04*, ACM Press (2004), 11–18.

[SWK+04] Sharlin E., Watson B., Kitamura Y., Kishino F. and Itoh, Y. On tangible user interfaces, humans and spatiality. *Personal and Ubiquitous Computing* 8, 5 (2004), 338–346.

[T91] Tang, J. C. Findings from observational studies of collaborative work. *International Journal of Man-Machine Studies* 34, 2 (1991), 143–160.

[TT06] Toney, A. and Thomas, B. H. Considering reach in tangible and table top design. In *Proc. TABLETOP '06*, IEEE (2006), 57-58.

[TT06B] Toney, A. and Thomas, B. H. Applying reach in direct manipulation user interfaces. In *Proc. OZCHI '06*, ACM Press (2006), 393-396.

[UI00] Ullmer, B. and Ishii, H. Emerging frameworks for tangible user interfaces. *IBM Systems Journal* 39, 3–4 (2000), 915–931.

[UI97] Ullmer, B. and Ishii, H. The metaDESK: models and prototypes for tangible user interfaces. In *Proc. UIST '97*, ACM Press (1997), 223–232.

[UI99] Underkoffler, J. and Ishii, H. Urp: a luminous-tangible workbench for urban planning and design. In *Proc. CHI '99*, ACM Press (1999), 386–393.

[UIJ05] Ullmer, B., Ishii, H. and Jacob, R. Token+constraint systems for tangible interaction with digital information. *ACM Transactions on Computer-Human Interaction (TOCHI)* 12, 1 (2005), 81–118.

[URL1] Gartner's 2010 Hype Cycle Special Report Evaluates Maturity of 1,800 Technologies http://www.gartner.com/it/page.jsp?id=1447613

[URL2] Kinect for Xbox 360 - Xbox.com. http://www.xbox.com/kinect

[URL3] LEGO.com MINDSTORMS. http://mindstorms.lego.com/

[URL4] PicoCricket - Invention kit that integrates art and technology. http://www.picocricket.com/

[URL5] Wii.com. http://wii.com/

[URL6] SourceForge.net: Open Computer Vision Library - Project Web Hosting - Open Source Software. http://opencvlibrary.sourceforge.net/

[URL7] DSI - Diffused Surface Illumination http://iad.projects.zhdk.ch/multitouch/?p=90

[URL8] Unibrain Fire-i firewire digital OEM board camera http://www.unibrain.com/products/visionimg/fire_i_bc.htm

[URL9] Video Tutorial - PS3 Eye Camera: Removing IR Blocking Filter, Installing Visible Blocking Filter (BandPass), and IR Light Tests - NUI Group Community Forums http://nuigroup.com/forums/viewthread/4189/

[URL10] Community Core Vision http://ccv.nuigroup.com/

[URL11] Nyko - Products – Zoom http://nyko.com/products/product-detail/?name=Zoom

[URL12] PCL - Point Cloud Library http://pointclouds.org/

[URL13] paulobala/tCAD https://github.com/paulobala/tCAD

[URL14] openFrameworks http://www.openframeworks.cc/

[URL15] obviousjim/ofxDelaunay https://github.com/obviousjim/ofxDelaunay

[URL16] ofTheo/ofKinect https://github.com/ofTheo/ofxKinect

[URL17] openframeworks/ofxOpenCv https://github.com/openframeworks/ofxOpenCv

[URL18] openframeworks/ofxOsc https://github.com/openframeworks/ofxOsc

[URL19] obviousjim/ofxSTL https://github.com/obviousjim/ofxSTL

[URL20] arturoc/ofxTuioWrapper https://github.com/arturoc/ofxTuioWrapper

[URL21] rezaali/ofxUI https://github.com/rezaali/ofxUI

[URL22] openframeworks/ofxXmlSettings https://github.com/openframeworks/ofxXmlSettings

[URL23] roxlu/ofxGeometry https://github.com/roxlu/ofxGeometry

[URL24] ofxCSG - Constructive Solid Geometry Addon - openFrameworks forum http://forum.openframeworks.cc/index.php?topic=5324.0

[URL25] Carve CSG - a fast and robust constructive solid geometry library http://carve-csg.com/

[URL26] blender.org - Home http://www.blender.org/

[URL27] paulobala/ofxCarveCSG https://github.com/paulobala/ofxCarveCSG

[URL28] ofxAddons http://ofxaddons.com/

[URL29] ofxCarveCSG - openFrameworks forum http://forum.openframeworks.cc/index.php/topic,10450.0.html

[URL30] meshfix - Repairing and modification of triangle meshes - Google Project Hosting http://code.google.com/p/meshfix/

[VB04] Vogel, D. and Balakrishnan, R. Interactive public Ambient displays: transitioning from implicit to explicit, public to personal, interaction with multiple users. In *Proc. UIST '04*, ACM Press (2004), 137-146.

[VB07] Vogel, D. and Baudisch, P. Shift: A technique for operating pen-based interfaces using touch. In *Proc. CHI '07*, ACM Press (2007), 657–666.

[W07] Wilson, A. D. Depth-Sensing Video Cameras for 3D Tangible Tabletop Interaction. In *Proc. TABLETOP '07*, IEEE (2007), 201-204.

[W10] Wilson, A. D. Using a depth camera as a touch sensor. In *Proc. ITS '10*, ACM Press (2010), 69-72.

[W91] Wellner, P. The DigitalDesk calculator: Tactile manipulation on a desktop display. In *Proc. UIST '91*, ACM Press (1991), 27–33.

[W93] Weiser, M. Some computer science issues in ubiquitous computing. *Communications of the ACM* 36, 7 (1993), 74–84.

[WB03] Wu, M. and Balakrishnan, R. Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. In *Proc. IUIST '03*, ACM Press (2003), 193–202.

[WB10] Wilson, A. D. and Benko, H. Combining multiple depth cameras and projectors for interactions on, above and between surfaces. In *Proc. UIST '10*, ACM Press (2010), 273-282.

[WG01] Weinberg, G. and Gan, S. The squeezables: Toward an expressive and interdependent multi-player musical instrument. *Computer Music Journal* 25, 2 (2001), 37–45.

[WHB10] Weiss, M., Hollan, J. and Borchers, J. Augmenting Interactive Tabletops with Translucent Tangible Controls. In Müller-Tomfelde, C. *Tabletops - Horizontal Interactive Displays*. Springer Publishing Company, London, 2010, 149-170.

[WIH+08] Wilson, A. D., Izadi, S., Hilliges, O., Garcia-Mendoza, A. and Kirk, D. Bringing physics to the surface. In *Proc. UIST'08*, ACM Press (2008), 67–76.

[WMG93] Wellner, P., Mackay, W. and Gold, R., Computer-augmented environments. Back to the real world. *Communications of the ACM* 36, 7 (1993), 24–26.

[WPP11] Wang, R. Y., Paris, S., and Popovi, J. 6D Hands: Markerless Hand Tracking for Computer Aided Design. In *Proc. UIST '11*, ACM Press (2011), 549-558.

[WWH08] Werner, J., Wettach, R. and Hornecker, E. United-pulse: Feeling your partners pulse. In *Proc. MobileHCI '08*, ACM Press (2008), 535–553.

[WWJ+09] Weiss, M., Wagner, J., Jansen, Y., Jennings, R., Khoshabeh, R., Hollan, J. D. and Borchers, J. SLAP Widgets: Bridging the gap between virtual and physical controls on tabletops. In *Proc. CHI '09*, ACM Press (2009), 481–490.

[ZGL+11] Ziola, R., Grampurohit, S., Landes, N., Fogarty, J., and Harrison, B. Examining interaction with general-purpose object recognition in LEGO OASIS. In *Proc. VL/HCC '11*, IEEE (2011), 65-68.

[ZHS+07] Zigelbaum, J., Horn, M., Shaer, O., and Jacob, R. The tangible video editor: Collaborative video editing with active tokens. In *Proc. TEI '07*, ACM Press (2007), 43–46.

ANNEX A - TCAD BRAINSTORMING

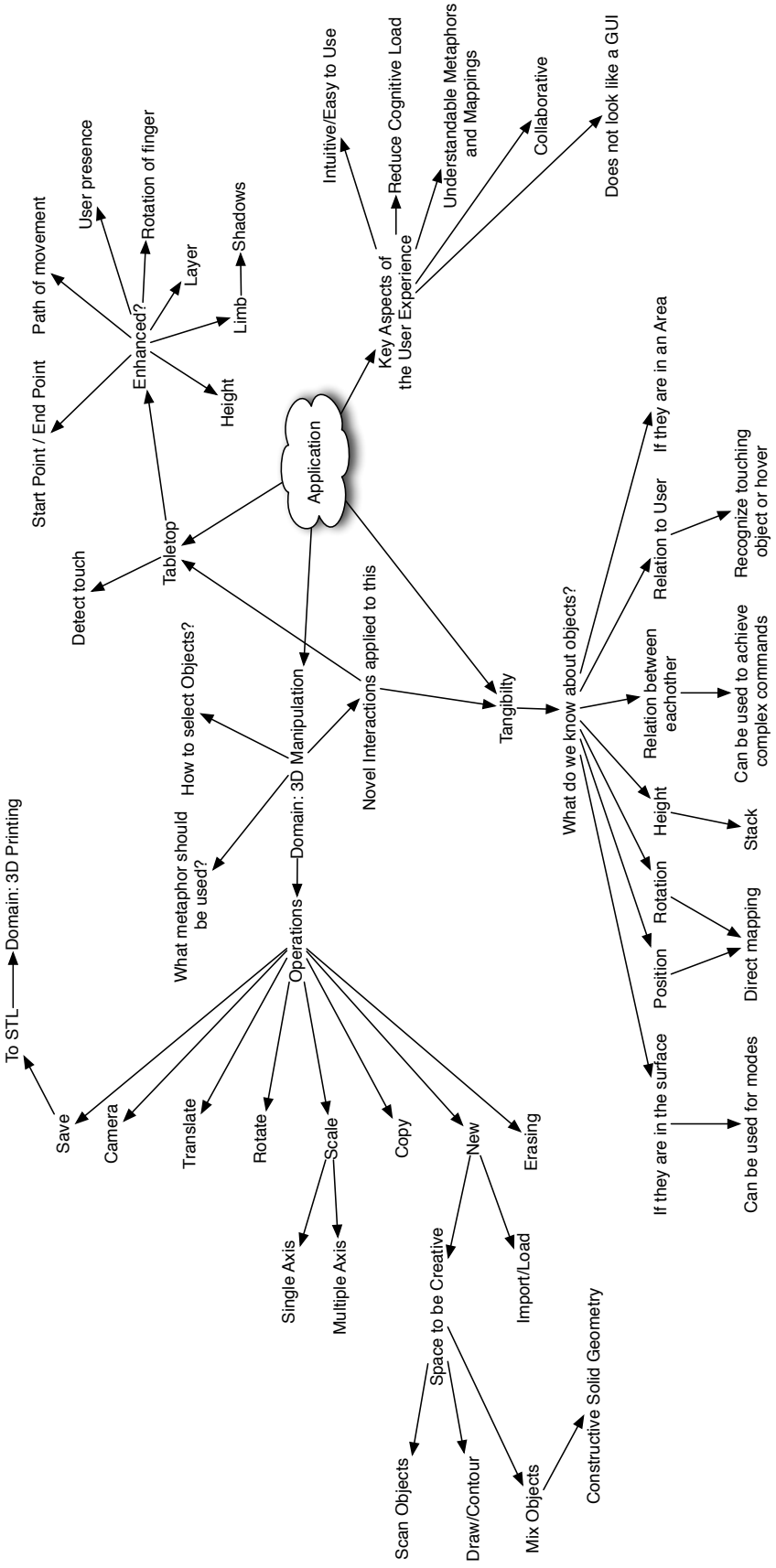


Figure 67 Brainstorming session for tCAD.

ANNEX B - TCAD USE CASE NARRATIVES AND ACTIVITY DIAGRAMS

Name	Link 3D Content
Assumptions	
Pre-conditions	
Use Case Initiation	This use case is initiated by demand (when the user starts a new touch event on the table surface).
Dialog	The system verifies with the camera, if there are any 3D shapes represented in the position of the user touch. If no 3D shapes are found, stop. Selected 3D shapes change appearance and dashed lines between the shapes and the finger are drawn.
Use Case Termination	The user can hover over the Container token (Normal Termination). The user can remove the finger from the scene (Cancel).
Post-conditions	Normal Termination: Selected 3D shapes change appearance; Selected 3D shapes are added to the Container token; Solid lines between the shapes and the Container token are drawn. Cancel: Selected 3D shapes change appearance.

Table 4 Use Case Narrative for use case "Link 3D Content".

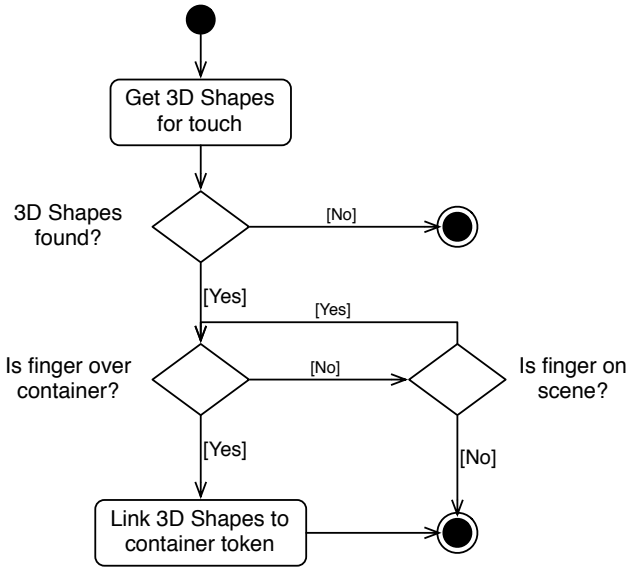


Figure 68 Activity Diagram for use case "Link 3D Content".

Name	Unlink 3D Content
Assumptions	
Pre-conditions	
Use Case Initiation	This use case is initiated by demand (when the user starts a new touch event on the table surface).
Dialog	The system verifies if line A (made between the initial point of the touch and the current point of touch) and line B (between any Container token and corresponding linked 3D Shapes) intersects in the "cutable"/dashed portion of line B. If intersection is found, the linked 3D Shape from line B is removed from Container token from line B.
Use Case Termination	The user removes the finger from the scene or starts another use case.
Post-conditions	No post-conditions for normal termination.

Table 5 Use Case Narrative for use case "Unlink 3D Content".

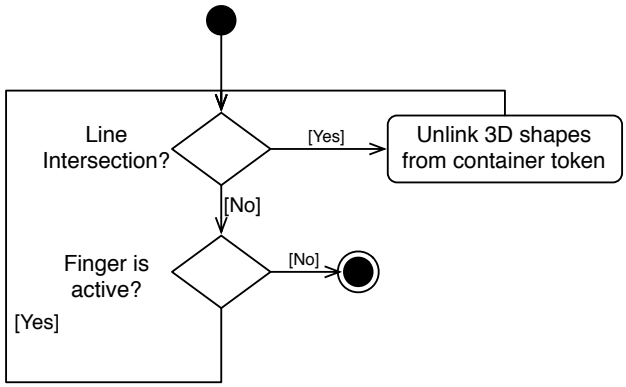


Figure 69 Activity Diagram for use case "Unlink 3D Content".

Name	Rotate Camera
Assumptions	
Pre-conditions	Camera is not locked (use case "Plane Selection").
Use Case Initiation	This use case is initiated by demand (when the user has one finger present in the camera control layer).
Dialog	The system performs an arc ball camera rotation according to the movement of the finger present in the camera control layer.
Use Case Termination	The user removes the finger from the camera control layer. The user starts another use case involving camera control.
Post-conditions	

Table 6 Use Case Narrative for use case "Rotate Camera".

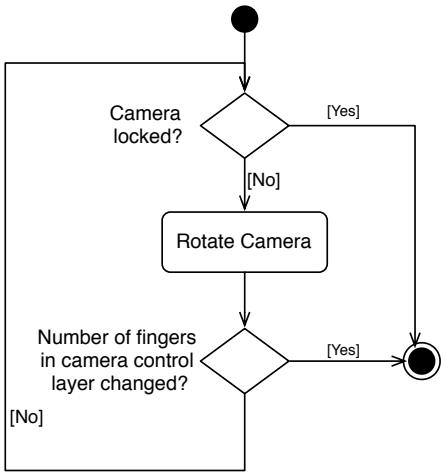


Figure 70 Activity Diagram for use case "Rotate Camera".

Name	Zoom Camera
Assumptions	
Pre-conditions	
Use Case Initiation	This use case is initiated by demand (when the user has two fingers present in the camera control layer).
Dialog	The system analyses if the distance between the two fingers is increasing or decreasing. If the distance is increasing, a zoom out operation is done. If the distance is decreasing, a zoom in operation is done.
Use Case Termination	The user removes both fingers from the camera control layer. The user starts another use case involving camera control.
Post-conditions	

Table 7 Use Case Narrative for use case "Zoom Camera".

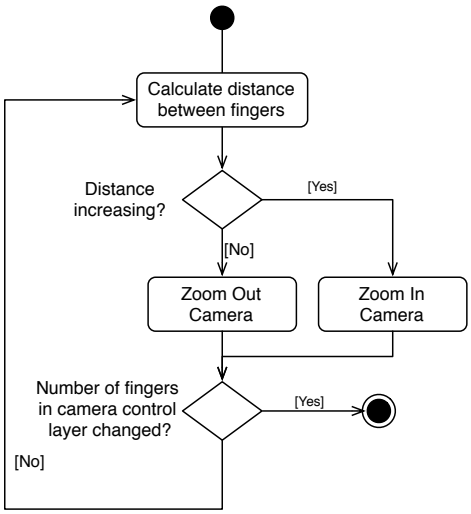


Figure 71 Activity Diagram for use case "Zoom Camera".

Name	Plane Selection
Assumptions	
Pre-conditions	
Use Case Initiation	This use case is initiated by demand (when the user introduces a plane according to the plane selection cube token).
Dialog	The system accomplishes a rotation tween to correct the camera position to the expected position of the desired plane. The system locks the camera.
Use Case Termination	The user removes the plane selection cube token.
Post-conditions	The system unlocks the camera.

Table 8 Use Case Narrative for use case "Plane Selection".

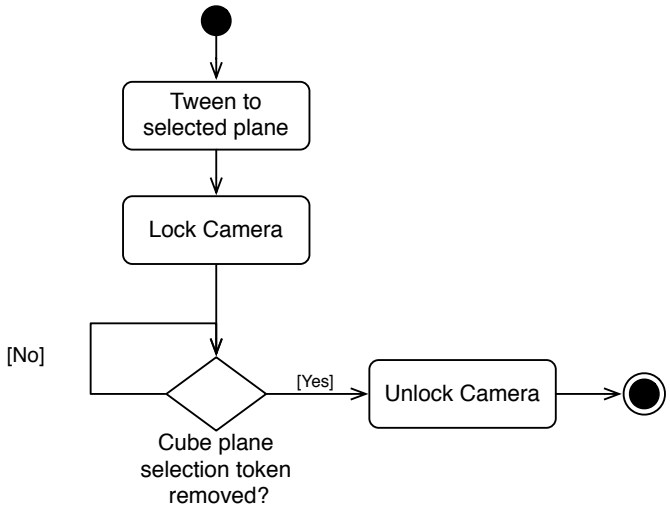


Figure 72 Activity Diagram for use case "Plane Selection".

Name	Choose Entry point for 3D Content
Assumptions	
Pre-conditions	
Use Case Initiation	This use case is initiated by demand (when the user has three fingers present in the camera control layer).
Dialog	The system chooses an entry point for 3D Content according to the projected point resulted from the intersection of the three fingers.
Use Case Termination	The user removes all fingers from the camera control layer. The user starts another use case involving camera control.
Post-conditions	

Table 9 Use Case Narrative for use case "Choose Entry point for 3D Content".

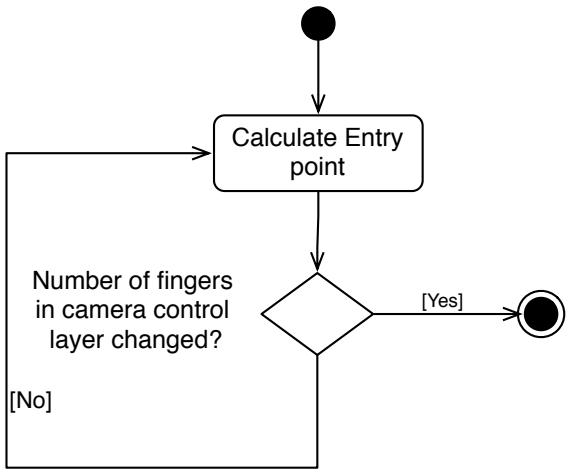


Figure 73 Activity Diagram for use case "Choose Entry point for 3D Content".

Name	Reset Camera
Assumptions	
Pre-conditions	
Use Case Initiation	This use case is initiated by demand (when the user has 10 fingers (2 hands) present in the camera control layer).
Dialog	The system resets the cameras position, distance and rotation to the applications standards.
Use Case Termination	The user removes at least one finger from the camera control layer. The user starts another use case involving camera control.
Post-conditions	

Table 10 Use Case Narrative for use case "Reset Camera".

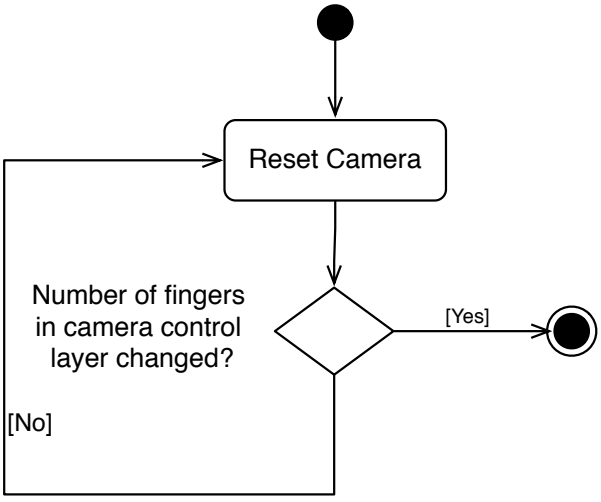


Figure 74 Activity Diagram for use case "Reset Camera".

Name	Create 3D Content from Kinect
Assumptions	
Pre-conditions	
Use Case Initiation	This use case starts on demand (by selection on the Content Creation UI)
Dialog	The system shows a 2.5D textured depth map in the application scene.
Use Case Termination	The user chooses to save the 3D Content (Normal termination). The user chooses to discard the 3D Content (Cancel)
Post-conditions	Normal termination: the 2.5D textured depth map is added to the scene as 3D Content. Cancel: No 3D Content is added to the scene.

Table 11 Use Case Narrative for use case "Create 3D Content from Kinect".

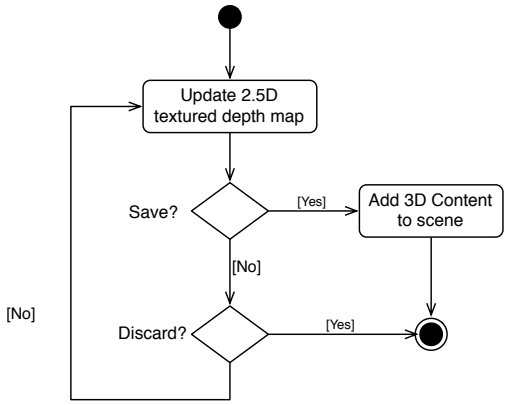


Figure 75 Activity Diagram for use case "Create 3D Content from Kinect".

Name	Create 3D Content from freehand Contour
Assumptions	
Pre-conditions	
Use Case Initiation	This use case starts on demand (by selection on the Content Creation UI)
Dialog	The user draws a contour by adding touches on the surface or continuously dragging on the surface.
Use Case Termination	The user chooses to save the Contour content (Normal termination). The user chooses to discard the Contour content (Cancel)
Post-conditions	Normal termination: the contour is tessellated and converted into a 3D Shape; this shape is then added to the scene. Cancel: No 3D Content is added to the scene.

Table 12 Use Case Narrative for use case "Create 3D Content from freehand Contour".

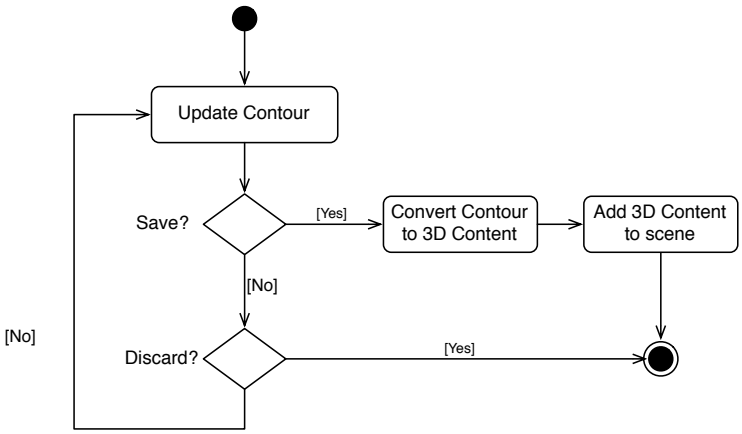


Figure 76 Activity Diagram for use case "Create 3D Content from freehand Contour".

Name	Create 3D Content from 3D Shapes
Assumptions	
Pre-conditions	
Use Case Initiation	This use case starts on demand (by selection on the Content Creation UI)
Dialog	The system asks the user to select one of the existing shapes Cube, Sphere, Pyramid, Tube, Cone, Cylinder).
Use Case Termination	The user chooses one of the Shape Options (Normal termination). The user does not chose one of the Shape Options (Cancel)
Post-conditions	Normal termination: the chosen 3D shape is added to the scene as 3D Content. Cancel: No 3D Content is added to the scene.

Table 13 Use Case Narrative for use case "Create 3D Content from 3D Shapes".

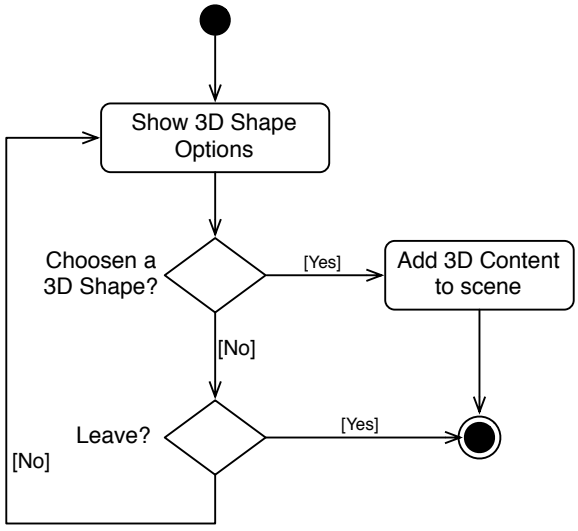


Figure 77 Activity Diagram for use case "Create 3D Content from 3D Shapes".

Name	Copy 3D Content
Assumptions	
Pre-conditions	
Use Case Initiation	This use case starts on demand (by selection on the Content Creation UI).
Dialog	The system asks the user to select one of the Container token with linked 3D shapes.
Use Case Termination	The user chooses one of the Container token's Copy Options (Normal termination). The user does not chose one of Container token's Copy Options (Cancel).
Post-conditions	Normal termination: the 3D shapes linked to the selected Container token are copied to the scene as 3D Content. Cancel: No 3D Content is copied to the scene.

Table 14 Use Case Narrative for use case "Copy 3D Content".

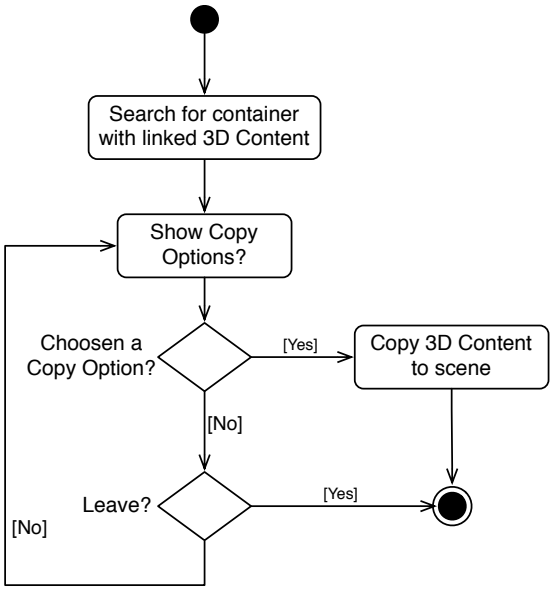


Figure 78 Activity Diagram for use case "Copy 3D Content".

Name	Rotate 3D Content
Assumptions	
Pre-conditions	The Container token has been linked to 3D Content (use case "Link 3D Content"). A plane has been selected (use case "Plane Selection").
Use Case Initiation	The use case is initiated by demand (when the user rotates the Container token or by choosing in the Manipulation UI).
Dialog	The system rotates the linked 3D Shapes according to the plane selected and direction of rotation.
Use Case Termination	The user removes the Container token from the surface. The user removes the plane selection cube token from the surface.
Post-conditions	

Table 15 Use Case Narrative for use case "Rotate 3D Content".

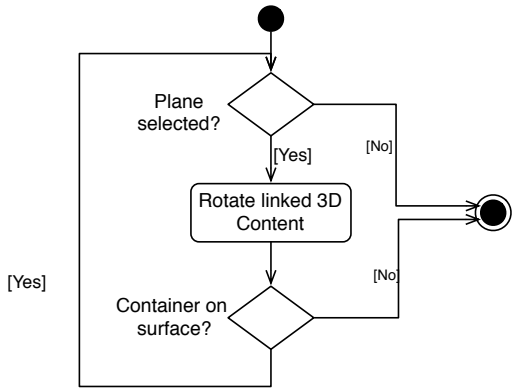


Figure 79 Activity Diagram for use case "Rotate 3D Content".

Name	Translate 3D Content
Assumptions	
Pre-conditions	The Container token has been linked to 3D Content (use case "Link 3D Content"). A plane has been selected (use case "Plane Selection").
Use Case Initiation	The use case is initiated by demand (when the user moves the Container token or by choosing in the Manipulation UI).
Dialog	The system translates the linked 3D Shapes according to the plane selected and speed of movement.
Use Case Termination	The user removes the Container token from the surface. The user removes the plane selection cube token from the surface.
Post-conditions	

Table 16 Use Case Narrative for use case "Translate 3D Content".

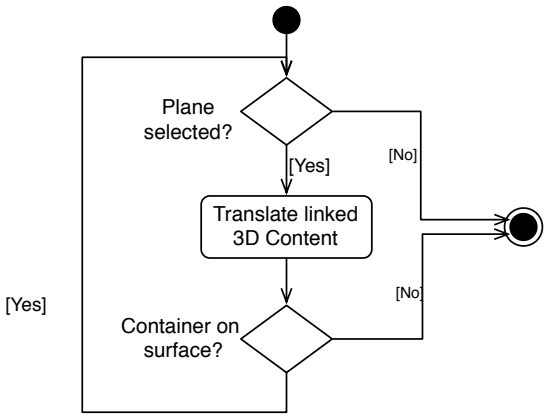


Figure 80 Activity Diagram for use case "Translate 3D Content".

Name	Multi Scale 3D Content
Assumptions	
Pre-conditions	The Container token has been linked to 3D Content (use case "Link 3D Content").
Use Case Initiation	The use case is initiated by demand (user drags an onscreen scale).
Dialog	The system scales the linked 3D Shapes in X, Y and Z-axes, according to the distance of the scale in relation to the Container token position.
Use Case Termination	The user removes the finger selecting the scaling scale. The user removes the Container token from the surface.
Post-conditions	

Table 17 Use Case Narrative for use case "Multi Scale 3D Content".

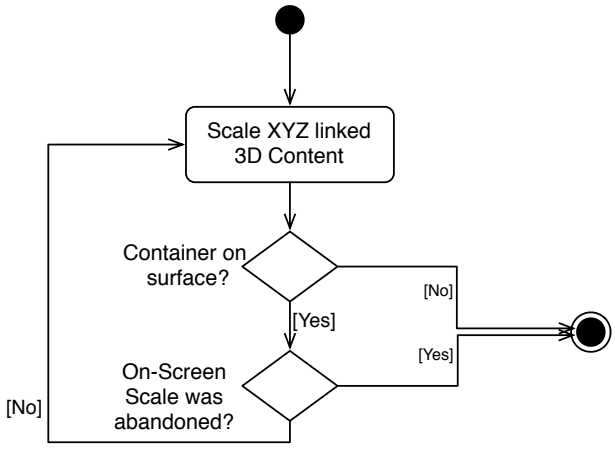


Figure 81 Activity Diagram for use case "Multi Scale 3D Content".

Name	Single Scale 3D Content
Assumptions	
Pre-conditions	The Container token has been linked to 3D Content (use case “Link 3D Content”). A plane has been selected (use case “Plane Selection”).
Use Case Initiation	The Use Case is initiated by demand (by selection on the Manipulation UI).
Dialog	The system translates the linked 3D Shapes according to the plane selected and the Manipulation UI.
Use Case Termination	The user removes the Container token from the surface. The user removes the plane selection cube token from the surface.
Post-conditions	

Table 18 Use Case Narrative for use case “Single Scale 3D Content”.

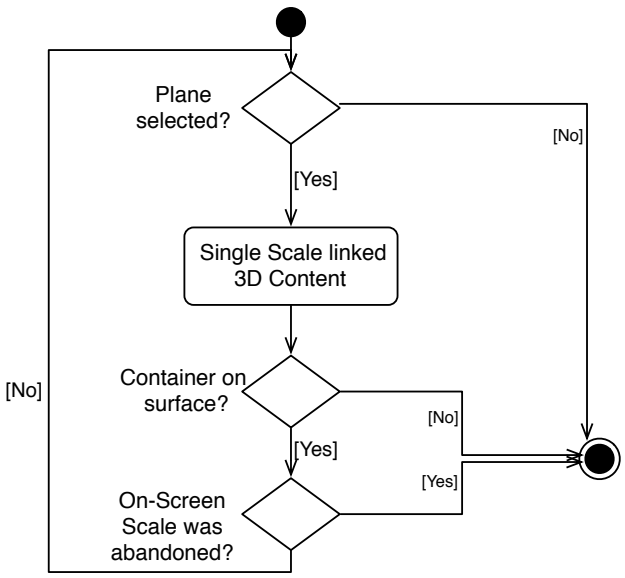


Figure 82 Activity Diagram for use case “Single Scale 3D Content”.

Name	Boolean Operation on 3D Content
Assumptions	
Pre-conditions	The Container token has been linked to 3D Content (use case "Link 3D Content"). A plane has been selected (use case "Plane Selection").
Use Case Initiation	The use case is initiated by demand (by selection on the Manipulation UI, if there are two 3D Shapes linked to Container token).
Dialog	The system asks the user to select one of the existing Boolean operations (Intersection, Union, Difference AB, Difference BA and Symmetric Difference)
Use Case Termination	The user chooses one of the Boolean Operations (Normal termination). The user does not chose one of the Boolean Operations (Cancel) The user removes the Container token from the surface. The user removes the plane selection cube token from the surface.
Post-conditions	Normal termination: the result of the chosen Boolean Operation is added to the scene as 3D Content. Cancel: No 3D Content is added to the scene.

Table 19 Use Case Narrative for use case "Boolean Operation on 3D Content".

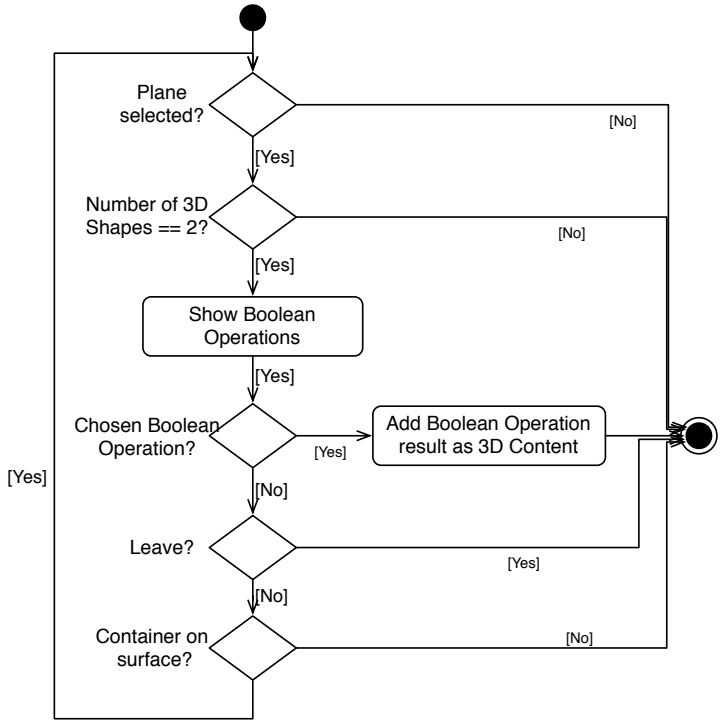


Figure 83 Activity Diagram for use case "Boolean Operation on 3D Content".

Name	Erase 3D Content
Assumptions	
Pre-conditions	
Use Case Initiation	The use case is initiated by demand (by placement of the Container token in the area formed by two Shredder tokens).
Dialog	The system removes 3D shapes linked with the Container from the scene (and from other linked Containers).
Use Case Termination	The user removes the empty Container token. The user removes a Shredder Container token.
Post-conditions	

Table 20 Use Case Narrative for use case "Erase 3D Content".

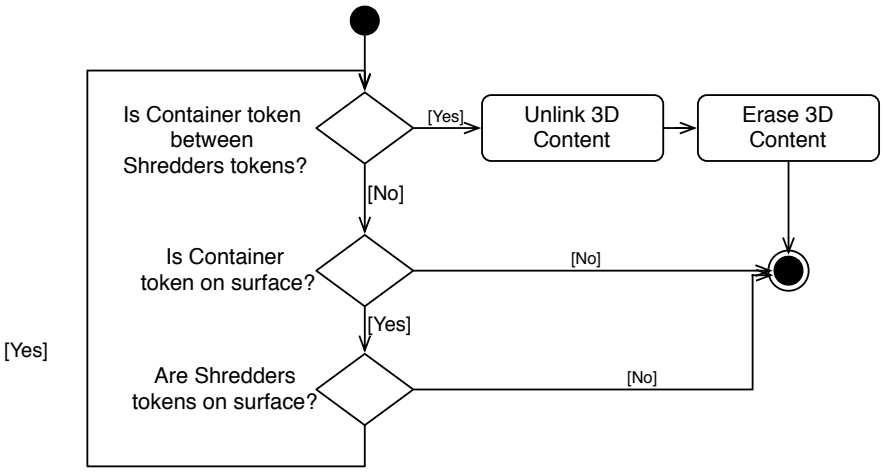


Figure 84 Activity Diagram for use case "Erase 3D Content".

Name	Save 3D Content
Assumptions	
Pre-conditions	
Use Case Initiation	The use case is initiated by demand (by placement of the Save token).
Dialog	The user presses the Save token. System saves 3D shapes on the scene into an STL file (whose name reflects the system time).
Use Case Termination	The user removes the Save token.
Post-conditions	

Table 21 Use Case Narrative for use case "Save 3D Content".

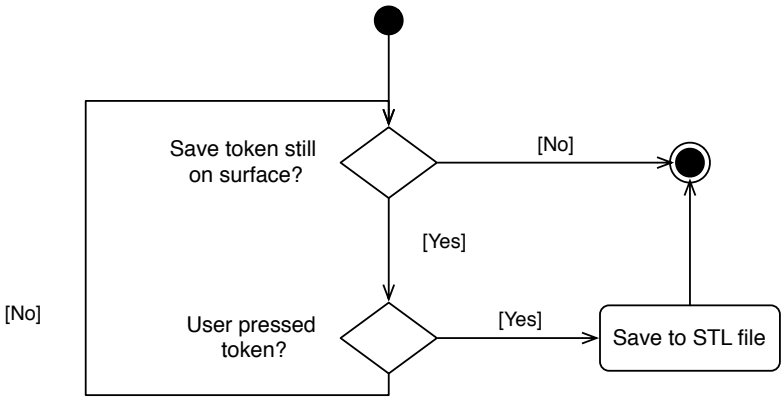


Figure 85 Activity Diagram for use case "Save 3D Content".

ANNEX C - TCAD TUIML

Token	Representation
Container token	
Shredder token	
On-table Content Creation token	
On-token Content Creation token	
In-air Content Creation token	
Cube token	
Save token	

Table 22 Tokens for TAC paradigm for tCAD TUI.

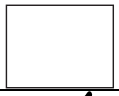
Constraint	Representation	Conceptual Relations	Action
Tabletop Surface		Identity, presence, containment, position (x, y)	Add, remove, move
Finger		Identity, presence, position (x, y, z)	Hover, touch

Table 23 Constraints for TAC Paradigm for tCAD TUI.

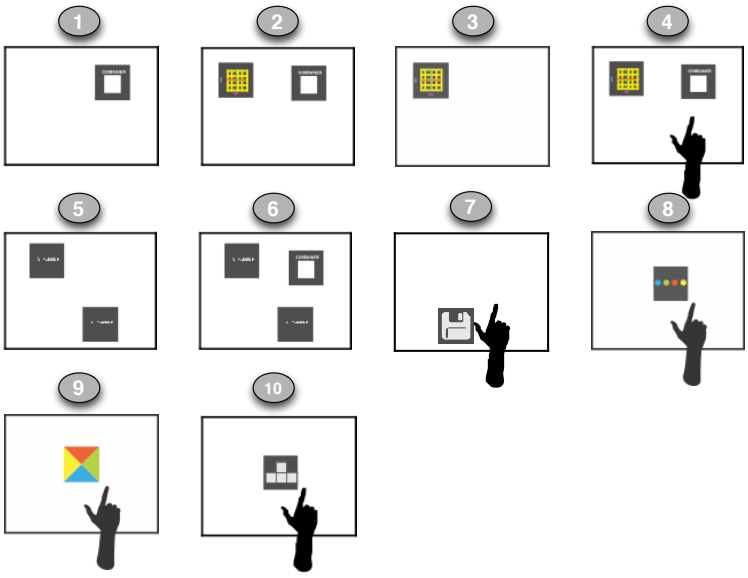
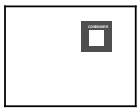
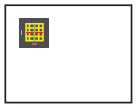
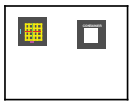
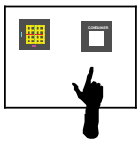
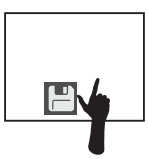


Figure 86 TACs for tCAD TUI.

TAC	Representation			Association	Manipulation	
	Variable	Token	Constraint	TAC Graphics	Action	Response
1	3D Content	Container token	Surface		Add	Displays link between Container token and 3D Content
					Move	Update link between Container token and 3D Content
					Remove	Remove link between Container token and 3D Content
2	Plane	Cube token	Surface		Add	Tween the camera to the plane specified by the cube; Locks camera
					Remove	Unlocks Camera
3	3D Content	Container token	Surface, Cube token		Move	Translates the 3D Content along the 2 axis defined by the plane; Update links
					Rotate	Rotates the 3D Content along the axis not defined by the plane
4	3D Content	Container token	Surface, Cube token, Finger		Hover	Show measures of 3D Content; Depending on Manipulation Mode, triggers manipulations.
					Touch	Iterates over Manipulations Mode.
5	Content Destruction	Shredder Tokens	Surface		Add	Displays area between Shredder tokens
					Move	Update area between Shredder tokens
					Remove	Removes areas between Shredder tokens
6	3D Content	Container token	Surface, Shredder tokens		Between	Deletes 3D Content
7	All 3D Content	Save token	Surface, Finger		Add	
					Remove	
					Touch	Saves 3D Content into an STL file

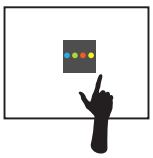
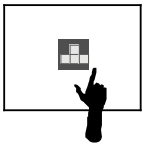
8	Content Creation Options	On-table Content Creation token	Surface, Finger		Add	Displays Content Creation Options
					Move	Updates Content Creation Options on the display
					Remove	Removes Content Creation Options from the display
					Touch	Makes the Content Creation Options visible/invisible
9	Content Creation Options	On-token Content Creation token	Surface, Finger		Add	Displays Content Creation Options
					Move	Updates Content Creation Options on the display
					Rotate	Updates Content Creation Options on the display
					Remove	Removes Content Creation Options from the display
					Touch	Choses one of the Content Creation Options.
10	Content Creation Options	In-air Content Creation token	Surface, Finger		Add	Displays Content Creation Options
					Move	Updates Content Creation Options on the display
					Rotate	Updates Content Creation Options on the display
					Remove	Removes Content Creation Options from the display
					Hover	Choses one of the Content Creation Options when a specific movement is made

Figure 87 TAC palette for tCAD TUI.

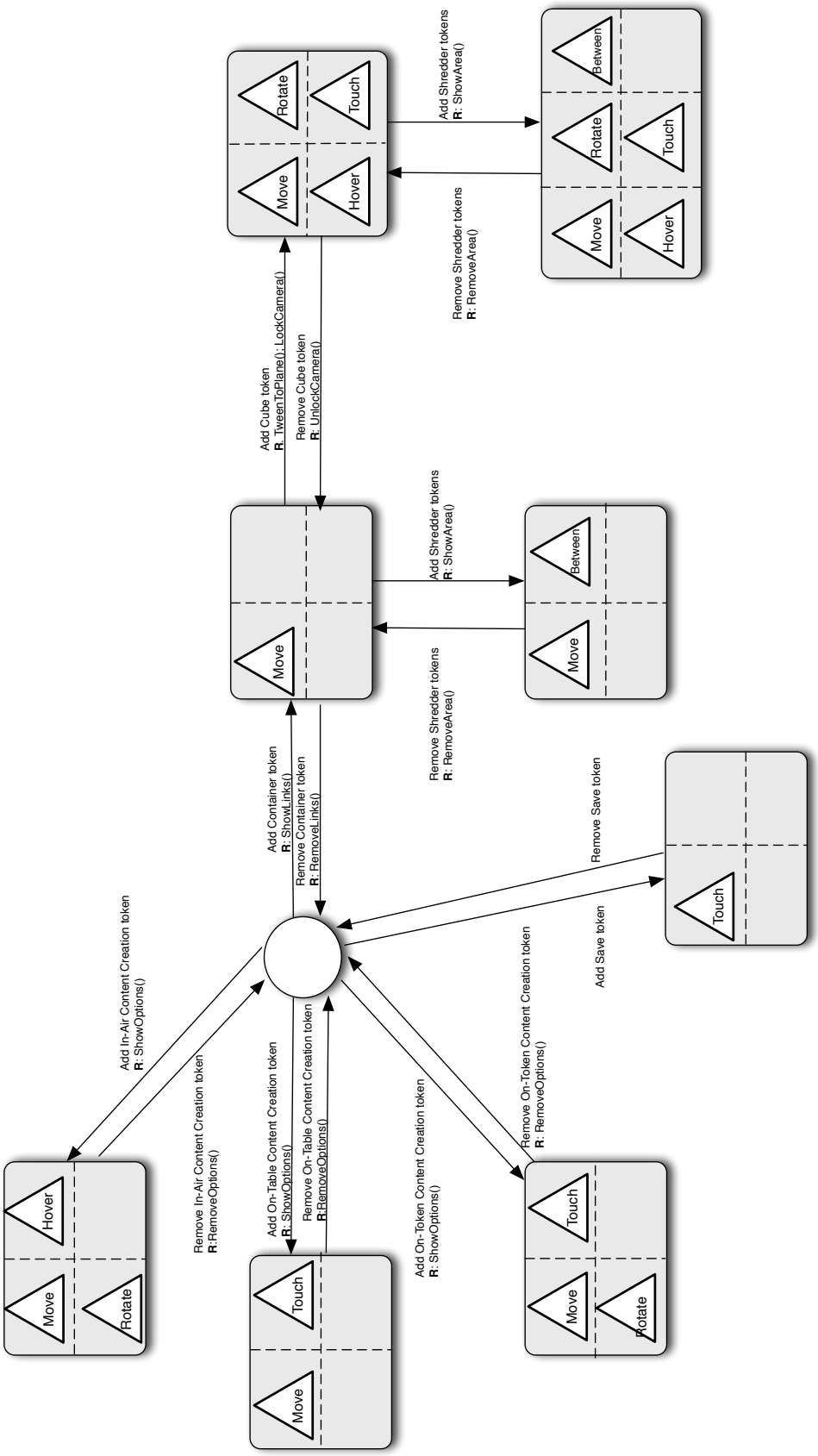


Figure 88 Dialogue Diagram for tCAD TUI.

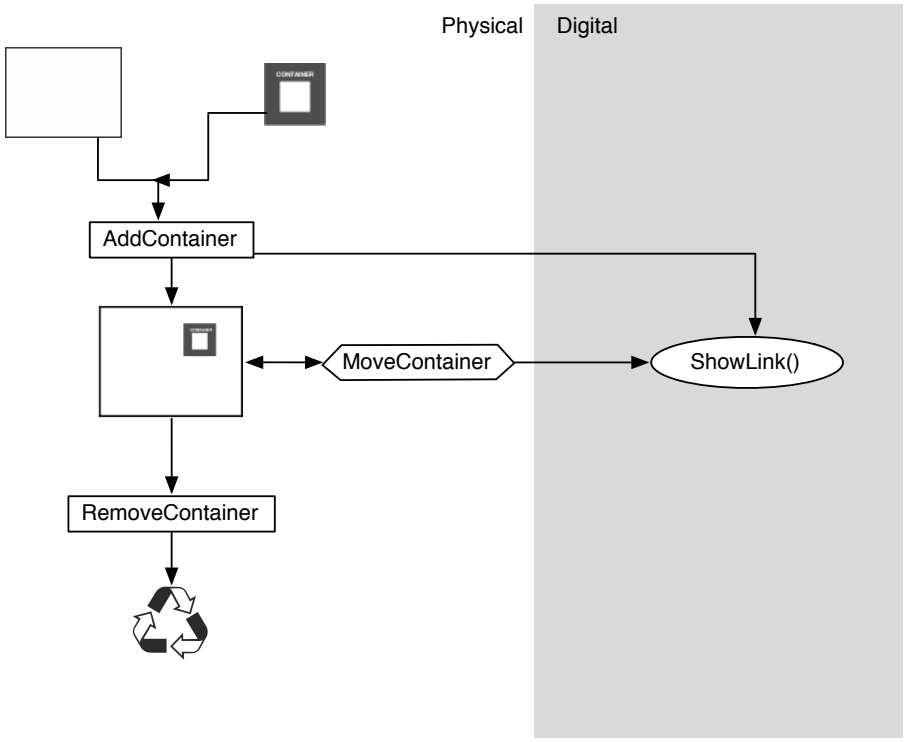


Figure 89 Interaction Diagram for TAC 1 on tCAD TUI.

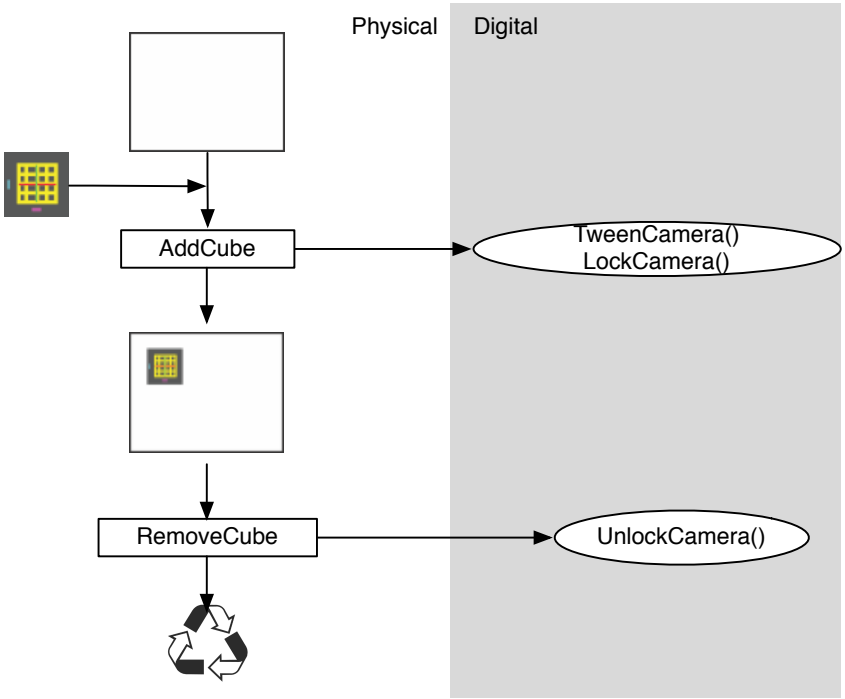


Figure 90 Interaction Diagram for TAC 2 on tCAD TUI.

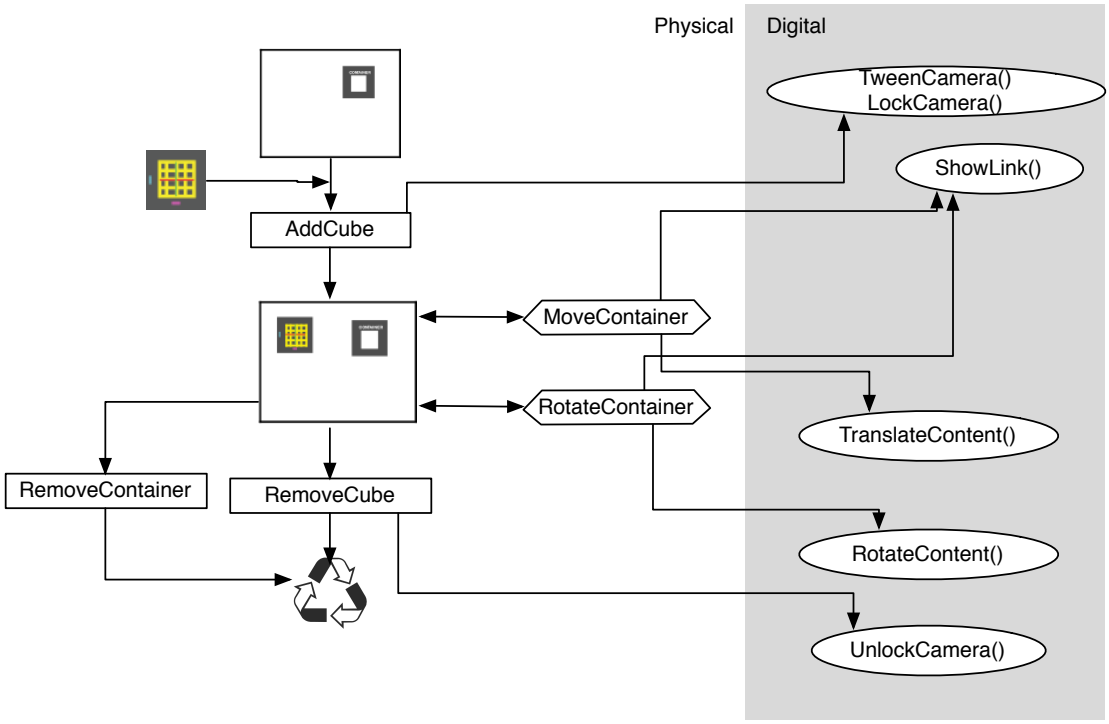


Figure 91 Interaction Diagram for TAC 3 on tCAD TUI.

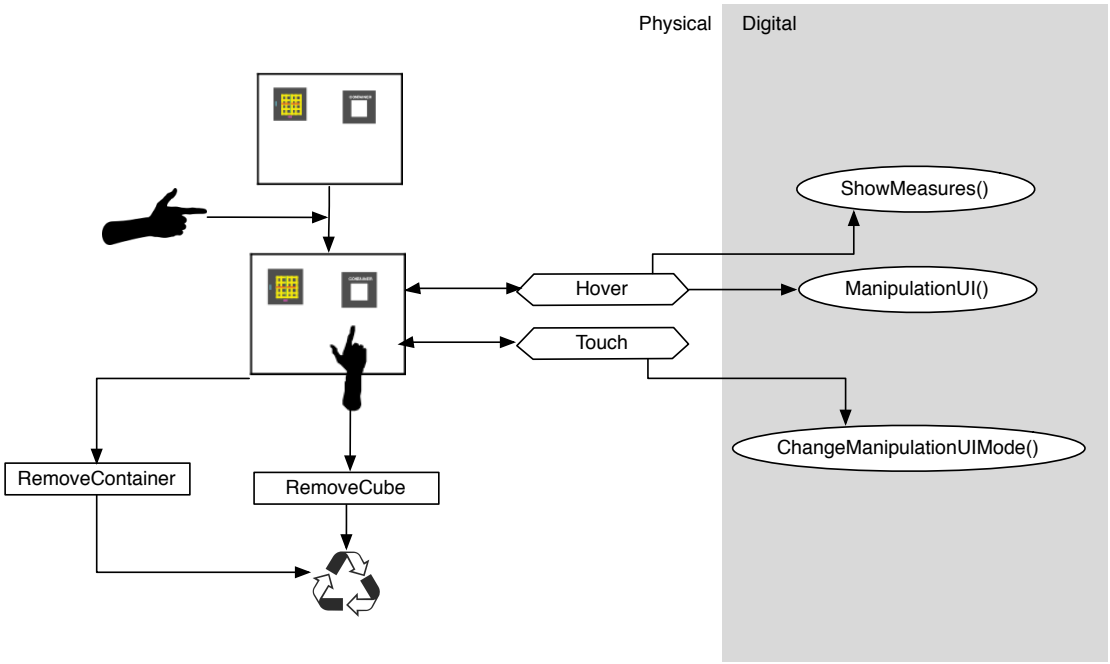


Figure 92 Interaction Diagram for TAC 4 on tCAD TUI.

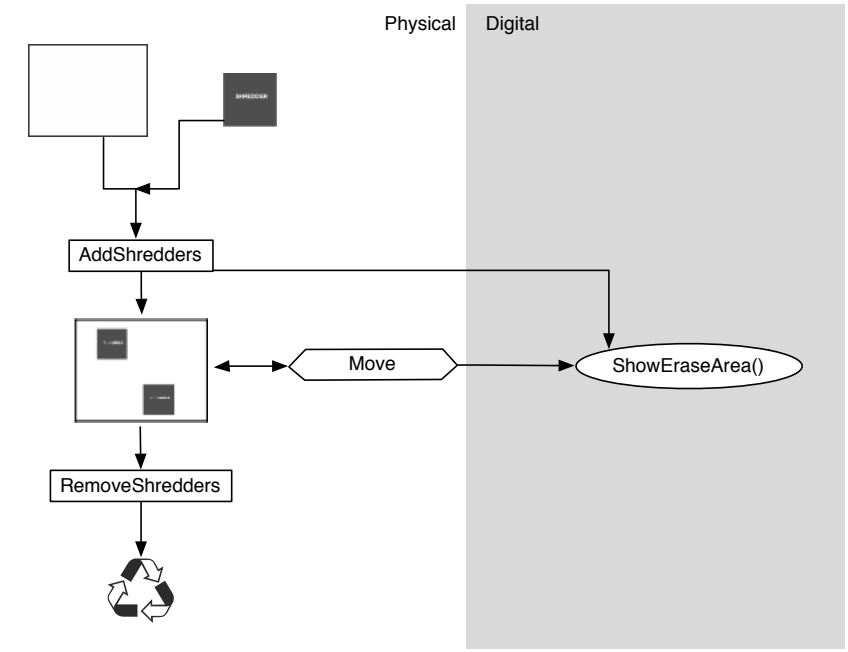


Figure 93 Interaction Diagram for TAC 5 on tCAD TUI.

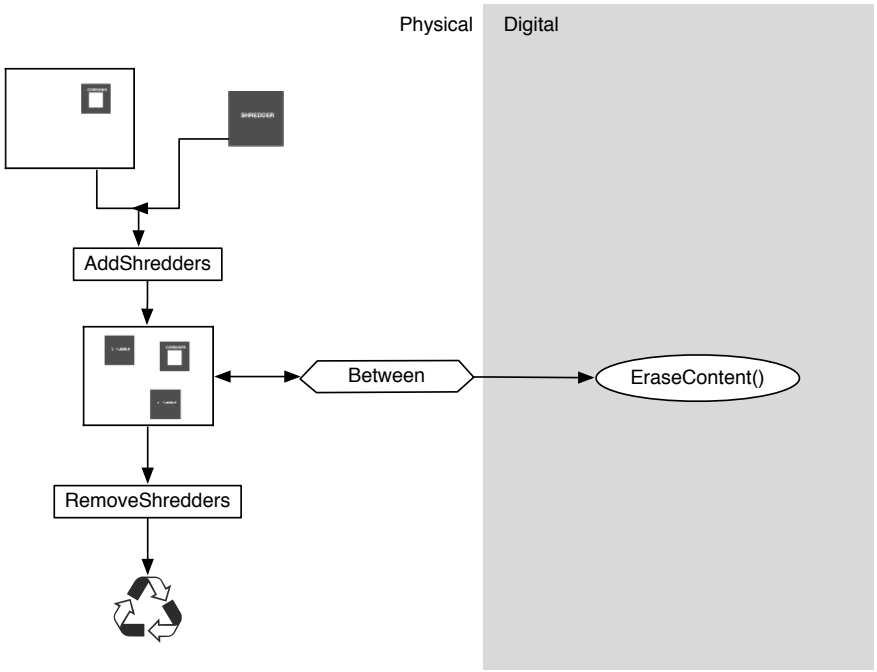


Figure 94 Interaction Diagram for TAC 6 on tCAD TUI.

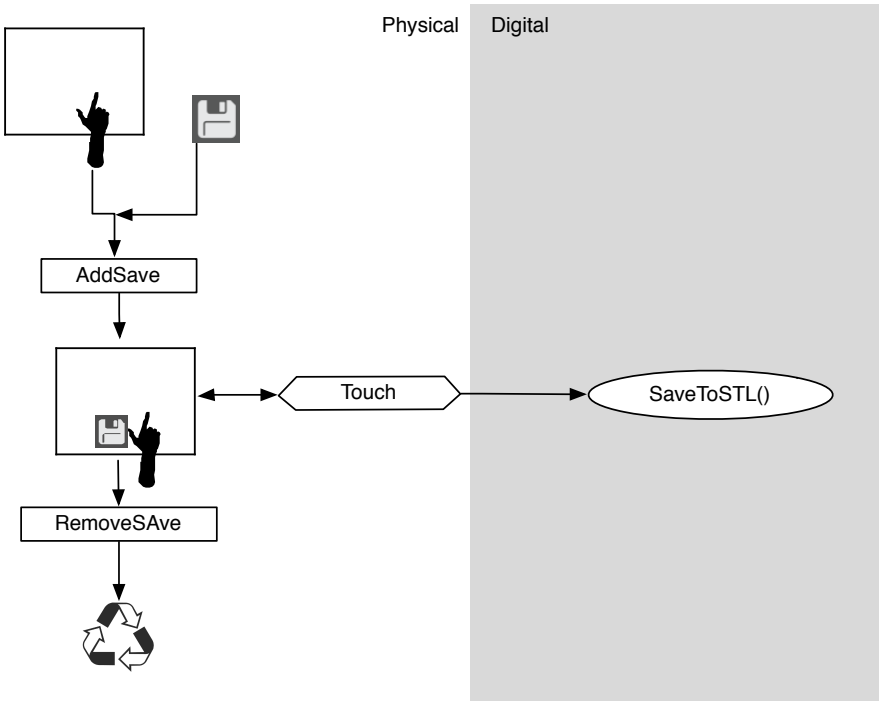


Figure 95 Interaction Diagram for TAC 7 on tCAD TUI.

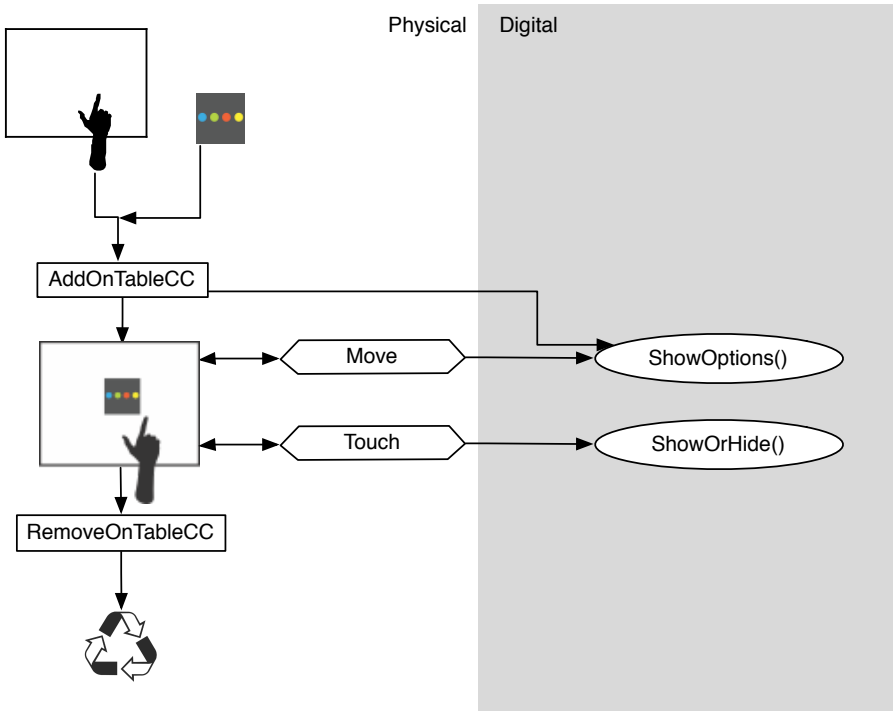


Figure 96 Interaction Diagram for TAC 8 on tCAD TUI.

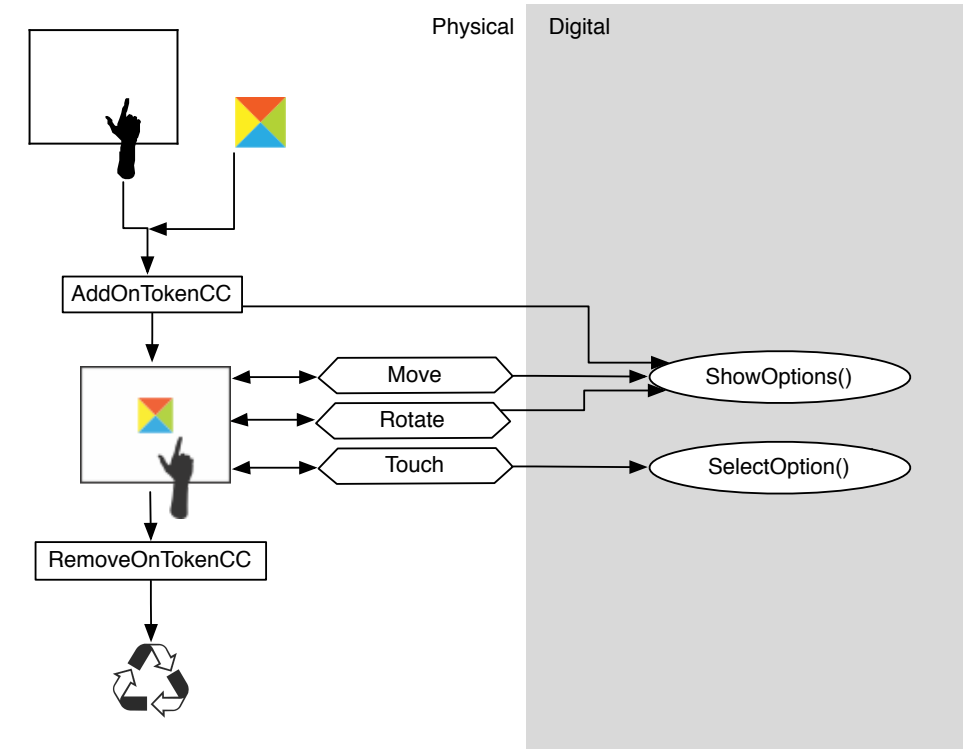


Figure 97 Interaction Diagram for TAC 9 on tCAD TUI.

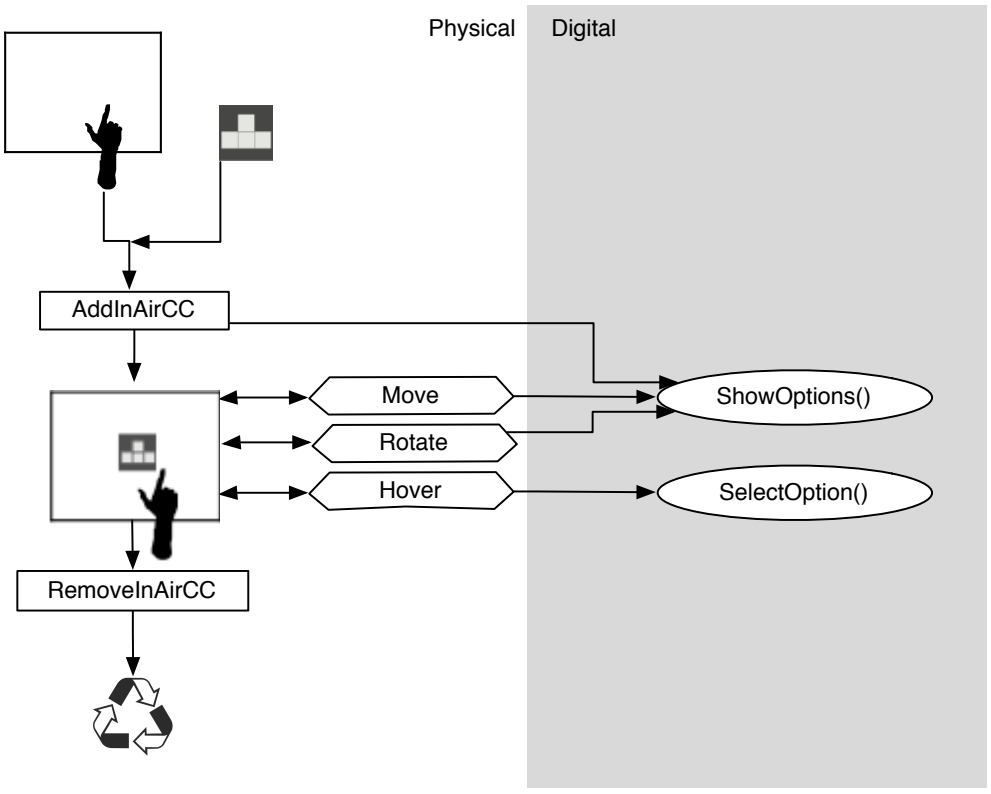


Figure 98 Interaction Diagram for TAC 10 on tCAD TUI.

ANNEX D – TCAD CLASS DIAGRAM

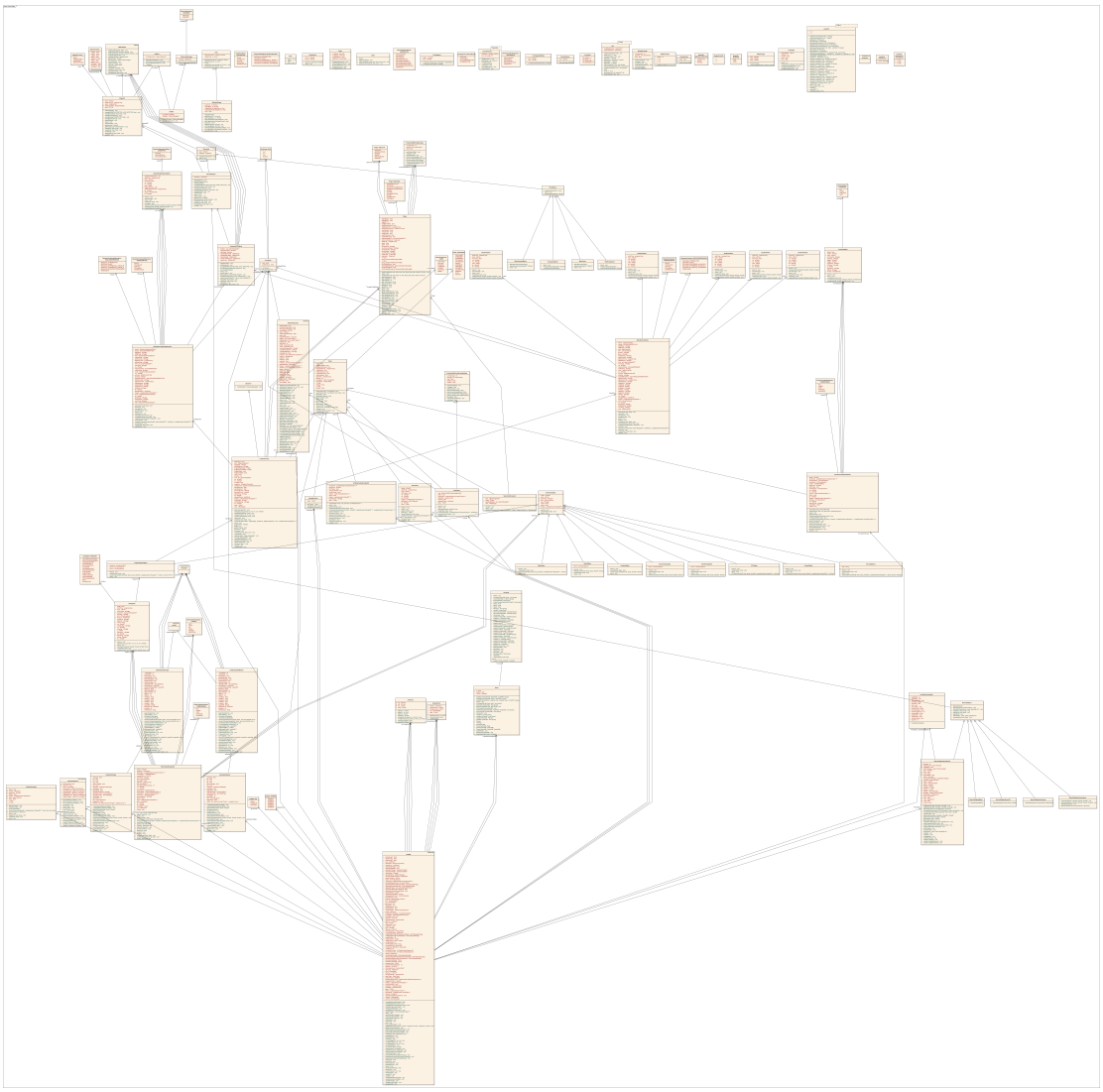


Figure 99 Class Diagram for tCAD Application. For a bigger resolution of this image, please refer to [URL13], where this image (classDiagram.png) and the source code are being hosted.

ANNEX E - TCAD INTERFACE SCREENSHOTS

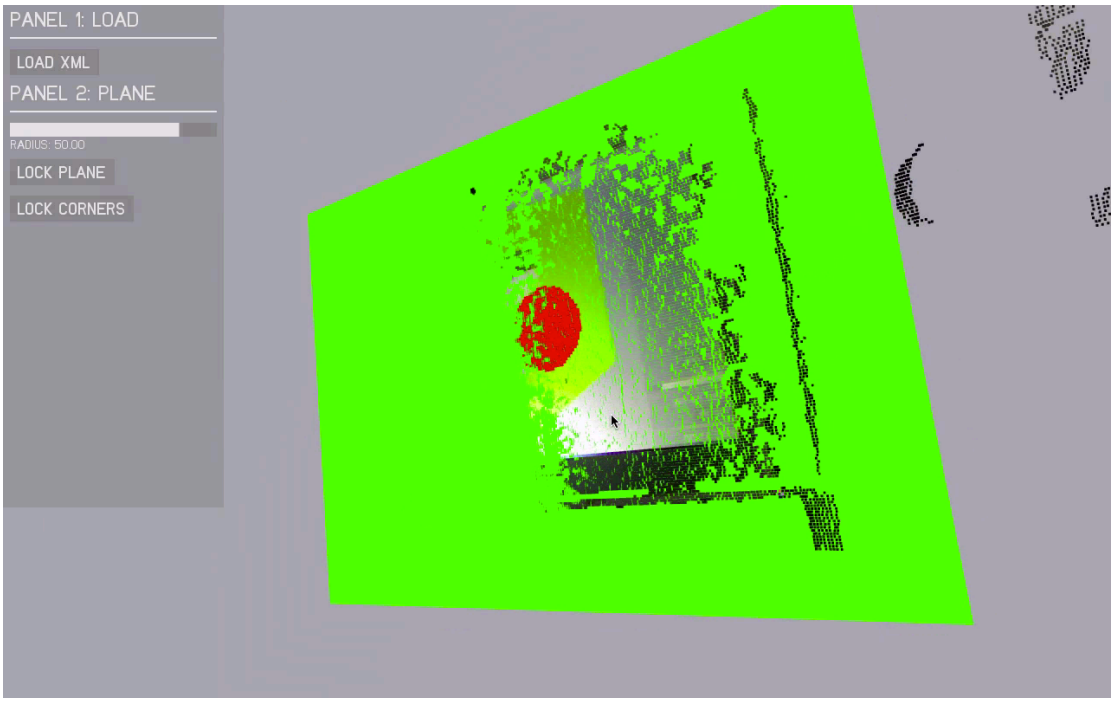


Figure 100 Screenshot of tCAD application corresponding do wireframe #1 (see Figure 41) with Point Cloud View. Red circle corresponds to pixels being used in the best-fit algorithm to get the green plane.

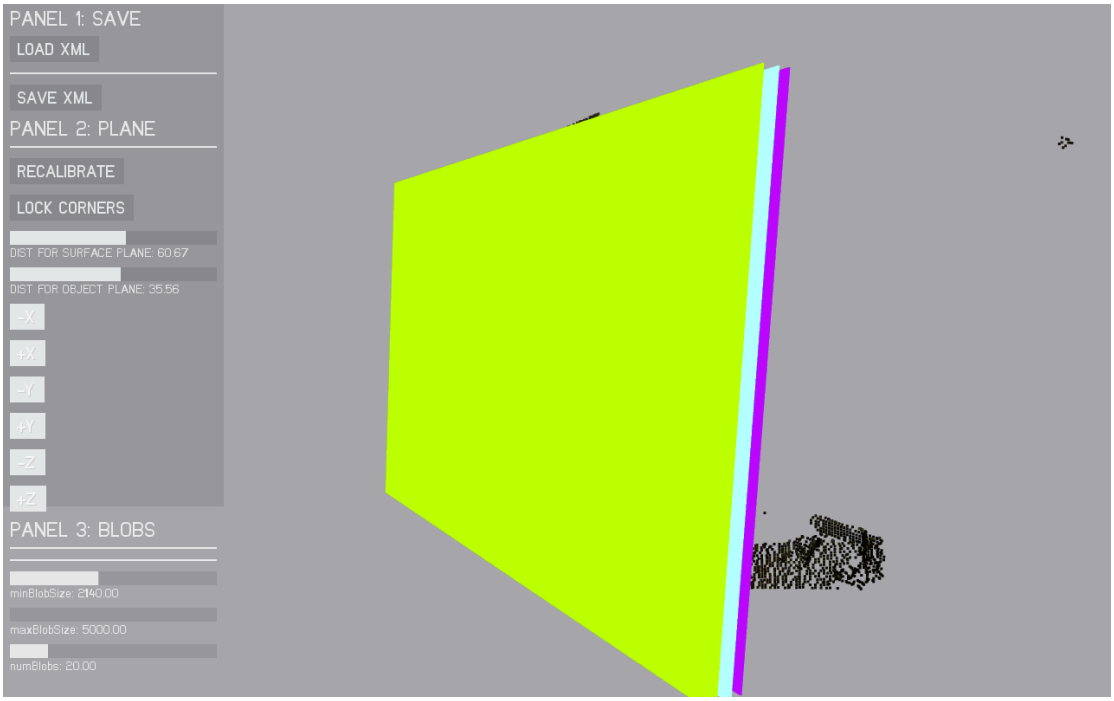


Figure 101 Screenshot of tCAD application corresponding do wireframe #1 (see Figure 41) with Point Cloud View. Green planes corresponds to plane defined by the three colums; blue plane is plane above objects; magenta plane is plane above surface.

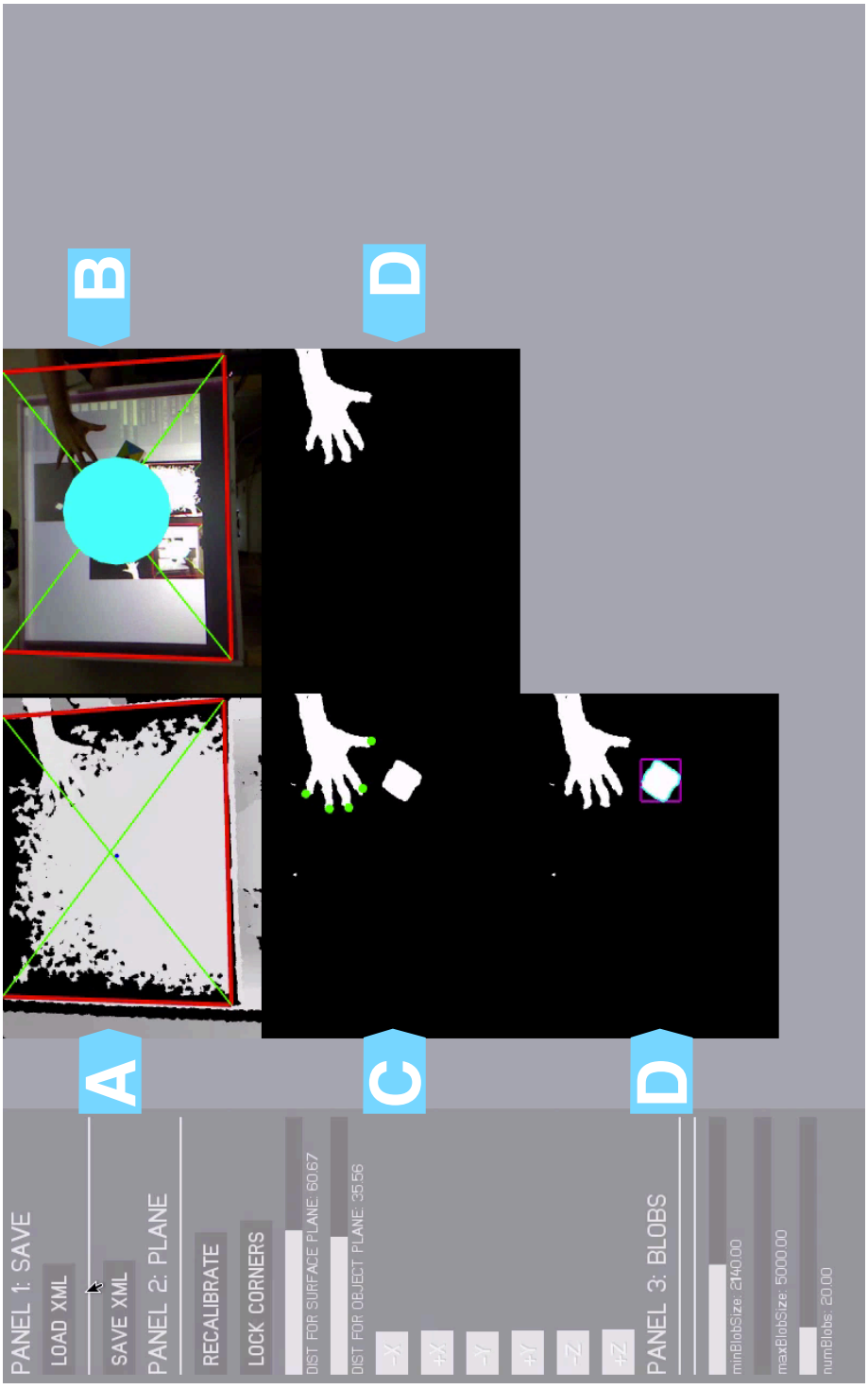


Figure 102 Screenshot of tCAD application corresponding do wireframe #1 (see Figure 41) with Depth Images View (see Figure 42). Image A correspond to a Kinect depth image, with red lines indicating the tabletop edges; Image B corresponds to Kinect real image, with a blue circle marking pixels used for plane detection; Image C correspond to a masked and perspective corrected depth image of the plane above the surface, with green dots indicating finger extremities; Image D corresponds to a masked and perspective corrected depth image of the plane above the objects; Image E corresponds to a masked and perspective corrected depth image of the plane above the surface, including found blobs in green.

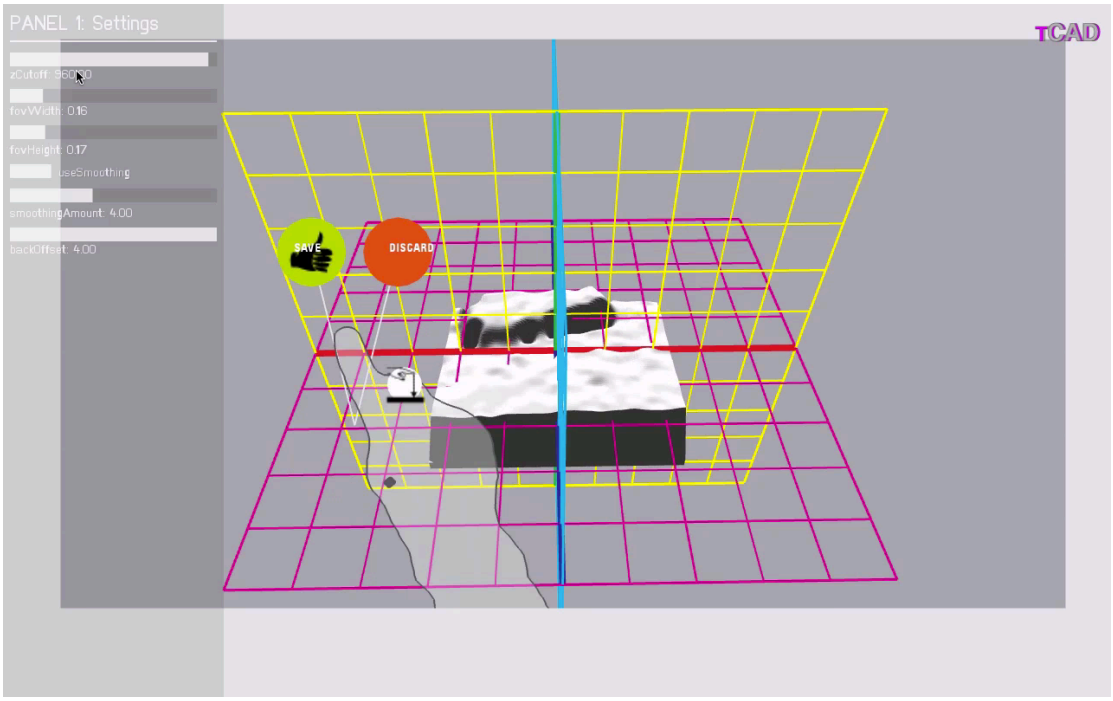


Figure 103 Screenshot of tCAD application corresponding do wireframe #2. User used a Content Creation token to enter Kinect Mode, scanning a object on tabletop into a 3D textured depth map.

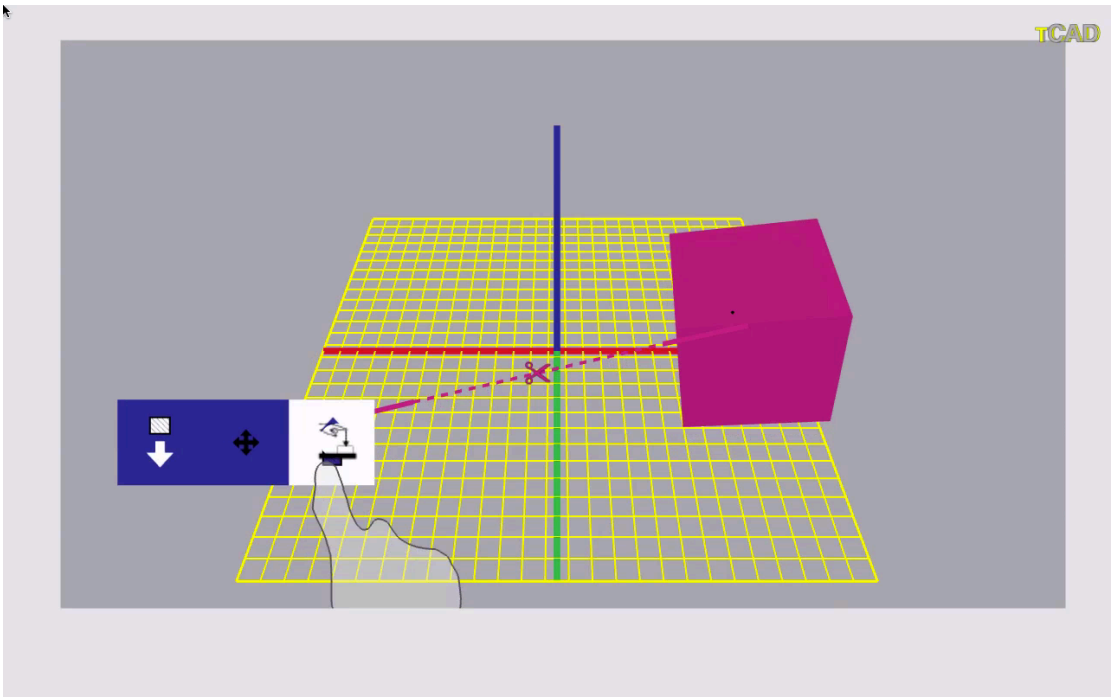


Figure 104 Screenshot of tCAD application corresponding do wireframe #2. User is manipulating 3D Content by hovering over the Manipulation UI around a Container token.

ANNEX F - TCAD EVALUATION OBSERVATIONS

Time	Description
01:00	User started by testing the camera control. The user stated that he preferred to keep his hand near the table limits because he had more control when he needed to stop the interaction.
03:00	User tried out the On-table Content Creation token, quickly achieving the end of the task.
04:00	User experimented the On-token Content Creation token and similarly to the previous token, promptly finished the task.
05:00	User tested the In-air Content Creation token. Given all the tokens, the user preferred the first one because all the options were shown at once and the interaction was quick, especially compared to the third token, which involves a longer interaction.
07:30	User linked and unlinked the sphere and the Container successfully, without complications.
09:00	User rotated and moved an object by manipulating the Container object; also, scaled the sphere using the draggable scale.
11:00	User used the Manipulation UI with dwell detection to manipulate translation, scale and rotation.
13:00	User used the Manipulation UI with finger movement detection but preferred the first one, as he had issues understanding the action required for the second UI.
15:00	When trying to erase a sphere, the user placed the Shredder tokens on the table and tried to pass the virtual content in between the Shredder tokens instead of passing the Container token between the Shredder tokens. The user corrected this behavior and completed the task.
18:00	The remainder of the session was spent of free play where the user explored the Manipulation UI, Boolean Operations and the Content Creation Modes (Kinect Scanning, Contour and Copy).

Table 24 Observation table for user #1.

Time	Description
00:09	User starts to control camera, first by zooming with two fingers (separate hands) and then rotating the camera with one finger.
00:32	Users states that she is having difficulty knowing where the interaction space ends.
01:04	After mastering the camera control, user tested the first token for Content Creation (On-table). User moves the token to a position that is more confortable (center of the tabletop). After pressing the wrong option on screen, the user goes back and quickly corrects the behavior.
02:00	User tests the second token for Content Creation (On-token). Starts by rotating the token, until the desired option is directly in front.
02:40	User has difficulties hitting targets, as she presses with her finger directly perpendicular to the table (and the tracking process fails to detect the finger). This is probably due to her experience with capacitive screens were the angle of finger is not relevant to the interaction.
04:30	User tests the third Content Creation token (In-air). Easily understands the carrousel metaphor. Complains about the difficulty of knowing which height to use.
05:20	User prefers the first (On-table) token because it presents more options simultaneously. User states the In-air token interaction takes too long.
06:00	User placed Container token on table and linked the sphere to the Container. After moving the Container, user placed the Container in different location as to expose the cutable part of the link and proceed to cut the link.
07:00	After redoing the link, the user picks up the cube token, chooses a plane by inspecting the cube sides and then places it on the table. User manipulates the token by moving it around and rotating it. User avoids picking up the object because she thinks it will break the connection with the object.
09:30	User starts to test the Manipulation UI with the dwell recognition. User thinks that the hit areas should be pressed instead of being hovered upon. User states that the color scheme used in the Manipulation UI makes it easy to know which axis is going to be influenced.
12:00	User starts to test the Manipulation UI with the finger movement recognition. Easily understands the metaphor and accomplishes the manipulations.
15:00	User liked both Manipulation UIs but prefers the dwell menus because they are more responsive and allow for easier manipulations.

17:00	User starts the free play period by copying the sphere using the Content Creation token.
19:00	User deletes the shape by passing the Container token between the Shredder tokens.
22:00	User tries out the Kinect Mode and the Contour Mode, electing the latter as her favorite.
25:00	The remainder of this session is spent using the Contour Mode.

Table 25 Observation table for user #2.

Time	Description
01:00	User tests Camera Control, specifically the entry point selection. User favored using only one hand for the camera control.
02:30	User starts testing the On-table Content Creation token. The users initial instinct is to slash the option in-air. After observing that the action did not work, the user pressed on-screen but did this action so quickly that it was not detected. The stabilization of Kinect data hinders the speed of tracking.
04:00	User starts testing the On-token Content Creation token. Does not rotate the token, choosing to touch the token sideways.
06:00	User tests the In-air Content Creation token. User has difficulties understanding the carrousel metaphor.
09:00	User links and unlinks sphere to Container token successfully.
11:00	After adding the cube, user translates and rotates the sphere by physically manipulating the object.
14:00	User tests the Manipulation UI with dwell recognition. The user also believes that the interaction involves touching the surface, so starts a tapping movement.
19:00	User tests the Manipulation UI with finger movement recognition.
21:00	User deletes contents using the Shredder tokens without problems.
22:00	User preferred the On-table Content Creation token after mastering it. As for the Content Manipulation UI, user has no preference.
24:30	The rest of session was spent on freeplay, namely, on Kinect scanning and Contour Modes.

Table 26 Observation table for user #3.

Time	Description
01:00	User tested camera control successfully.
03:00	User starts testing on On-table Content Creation token. The user had no difficulty executing the task.
04:00	User tests the On-token Content Creation token. The user does not rotate the token but instead rotates the hand.
05:00	User test the In-air Content Creation token. The user had no difficulty executing the task. This token was the users favorite because the carrousel metaphor was familiar.
08:00	User successfully linked and unlinked a sphere with the Container
09:00	User moved and rotated the sphere using the Container token.
10:00	User tested the Manipulation UI with dwell detection without issues
13:00	User tested the Manipulation UI with finger movement detection without issues. Although the user liked both alternatives, the dwell Container was the favorite.
17:00	User effectively destroys the Content on the scene by placing the Container token between the Shredders (not by dragging it).
20:00	Freeplay activities included Kinect Scanning, Contour Mode and copying existent content.

Table 27 Observation table for user #4.

Time	Description
00:30	User starts by controlling camera. Has no difficulties interacting with the depth layer.
02:00	User tests the first Content Creation token (On-table). Moves the token closer to him and then chooses the onscreen options.
3:30	User tests the second Content Creation token (On-token). Rotates the token and presses to select option. Expects that when you select a sphere, to choose size immediately.
05:00	User tests the third Content Creation token (In-air). Initially, the user has difficulties understanding the carrousel metaphor, namely in finding the desired options.
09:00	User links the virtual Content with the Container, cutting the link and restating the link without difficulty.
10:00	User places a cube token and proceeds to move and rotate the Container token.
13:00	User suggests that lifting one side of the cube should be detected but realizes that would not lead to recognizing the fiducial markers.
14:00	User tests the Manipulation UI with dwell time. User states that the rotation UI is what he wanted to do with previous suggestion.
18:00	User tests the Manipulation UI with finger movement. User doesn't like this Manipulation UI because it is harder to control.
20:30	User destroys the content using Shredder tokens.
22:00	User experiments with the other Content Creation Modes (Copy, Contour and Kinect). User states that he is acostumed to interfaces with menu and dialogs in lists and that it is difficult for him to use interface elements that may be scattered on screen.
24:00	The user prefers the In-air Content Creation token; although he struggled to master it, the user prefers this one because it's the most stable.
26:00	The rest of the session was spent on freeplay; the user experimented with the Boolean operations and the Contour Mode; the user also experimented with having many objects linked to the same Container.

Table 28 Observation table for user #5.