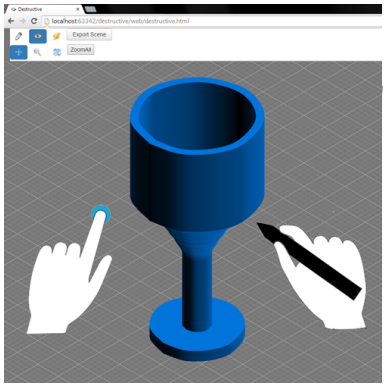
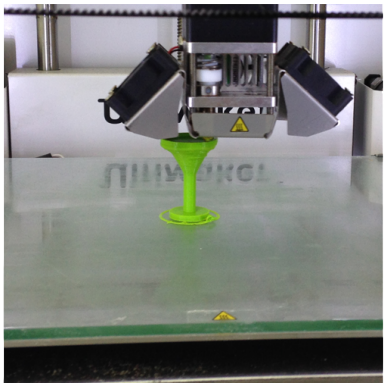
TÉCNICO LISBOA


Modelling on Interactive Surfaces for 3D Fabrication

Miguel Ferreira Nixo

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisor: Prof. Alfredo Manuel dos Santos Ferreira Júnior

Examination Committee

Chairperson: Prof. José Luís Brinquete Borbinha
Supervisor: Prof. Alfredo Manuel dos Santos Ferreira Júnior
Member of the Committee: Prof. Carlos Alberto Pacheco dos Anjos Duarte

May 2015

Acknowledgments

Firstly, I would like to thank Professor Alfredo Ferreira Jr. for the opportunity to carry out this work, as well as for his guidance during its course. I would also like to thank Daniel Mendes for his essential advice, constructive criticism and unconditional availability. I want to thank Nelson Silva and Tiago Cardoso of inEvo, for having me in their business company as part of their own team, and for supporting my work with their experience and time. I thank João Guerreiro for his support during the integration of the EnContRA 3D retrieval system. I'd like to also leave a kind word to all of those who have agreed and had the time to participate in the evaluation sessions, providing me with valuable information and constructive suggestions. Finally, to Instituto Superior Técnico (IST), to the Visualisation and Intelligent Multimodal Interfaces (VIMMI) group and to the people who comprise it, for giving me the means that enabled this work.

I would also like to mention the financial support from the Portuguese Foundation for Science and Technology (FCT, UID/CEC/50021/2013), through the project CluTCh (ADI/QREN/22984).

Resumo

A disponibilidade e utilização de objectos 3D aumentou durante os últimos anos, junto com o fabrico de baixo custo de objectos 3D. Tal cenário torna necessário que existam ferramentas para a criação e modificação destes objectos de forma eficiente e com abordagens naturais, permitindo que estas possam ser usadas por utilizadores experientes ou inexperientes. Com este trabalho, temos como objectivo desenvolver uma abordagem para modelação 3D com foco em impressão, que tira partido de métodos para recuperação de objectos 3D, criando uma experiência interactiva, bem como da utilização de superfícies interactivas. De forma a permitir uma experiência de interacção natural, criámos e discutimos abordagens que são baseadas apenas em toque, caneta, ou ambas. Depois de uma fase de avaliação com utilizadores, os resultados sugerem que abordagens baseadas em interacção com caneta são satisfatórias apesar de não serem tão eficientes e agradáveis quanto as baseadas em toque ou na combinação de toque e caneta, num cenário de modelação para impressão 3D.

Palavras-chave: Multi-toque, Caneta, Modelação 3D, Recuperação de Objectos 3D, Impressão 3D.

Abstract

The availability and use of digital 3D objects increased over the recent years, together with low-cost 3D printing. Such scenario raised the necessity of tools to create and modify these objects efficiently and with natural approaches, so that they can be used by both trained and non-trained users. With this work, we propose to develop a walk-up-and-use 3D modelling approach aimed at 3D fabrication, that takes advantage of methods for 3D object retrieval to provide interactive feedback, and the use of interactive surfaces, as the latter are becoming increasingly more popular in devices such as smartphones, tablets and even tabletops. In order to provide a natural interaction experience, we developed and discussed approaches that are solely based on touch or pen interaction, and others that combine both. After conducting a user evaluation process, results suggest that pen-based approaches are satisfying although not as efficient and entertaining to the average user as the touch-based and combined pen and touch approaches, in modelling scenario for 3D fabrication.

Keywords: Multi-touch Interaction, Pen Interaction, 3D Modelling, 3D Object Retrieval, 3D Fabrication.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The 3D printing technology and community has undergone significant growth during the recent years, empowering the common user to fabricate 3D printed models at home at relatively low costs. Also, with the generalization of interactive surfaces in every-day devices such as smartphones and tablets, Multi-touch enabled interactive tabletops that existed for over forty years are becoming more desired and popular.

The emergence of new technologies and devices naturally leads to the need to develop new interaction techniques. These new approaches will have to be more familiar to the generic user, while offering full control of the application. To do so, natural analogies can be used, such as the bimanual manipulation or the use of a pen for sketching. Although the use of touch and calligraphic interaction has been studied in different works, several questions remained unanswered, namely regarding the advantages and disadvantages over each other and over the combined use of both.

1.1 Motivation

With the proliferation of the 3D printing technology as well as the high availability of 3D resources online in the recent years, it becomes important to devise 3D modelling methods that allow the common user to be able to create and modify models with ease. Also, as interactive surfaces are becoming widely used in smartphones, tablets, laptops, and intelligent tables and boards, new possibilities arise for the development of 3D modelling solutions that are more natural.

Professional tools require advanced knowledge and training in order to be used effectively. Additionally, in recent years we witnessed a significant growth in the field of 3D printing, as the average price of the standard 3D printer has decreased substantially during that period. Taking this into account, it is predictable that 3D printing will gradually be more available to the average user, resulting in the need to develop new 3D modelling methods that allow non-trained users to create and modify 3D models in natural ways. Also, interactive surfaces are currently common in most mobile devices, representing a viable alternative to the standard WIMP (windows, icons, menus and pointer) interaction methodology, allowing fast and yet simple interaction using gestures and multi-touch. For devices that also support

it, pen interaction can also provide advantages when combined with multi-touch as it is naturally more familiar for drawing.

1.2 Problem Description

To allow a natural experience in a 3D modelling context for both trained and non-trained users, it is important to understand the impact of different types of interactions when combined with natural and interactive tools. Furthermore, there's a need to develop a simple yet complete user interface for complex model creation. In order to enhance this natural approach, a 3D object retrieval system can be interactively integrated, in order to assess its utility in a 3D modelling context.

While modelling a complex shape, it is crucial to allow the user to fully manipulate the parts that compose it. Regarding the final output model, it must be considered a production ready model, appropriate for 3D fabrication.

1.3 Objectives

Although much work has already been done to develop professional tools, these require advanced knowledge and training in order to be used effectively, even if the goal is to create an arbitrarily simple shape. Likewise, we aim to develop a walk-up-and-use 3D modelling approach focused on 3D fabrication, that takes advantage of methods for 3D object retrieval and the use of interactive surfaces. Also, developing a user interface as a NUI (natural user interface) allows the average user to be able to use the solution in a more proper and natural way. Additionally, this interface can also be complemented by interactive and non-intrusive tools, able to suggest appropriate outcomes given the current state of the application.

The overall idea of this study is to develop a system that enables any user to create and print an object idealized and modelled by him or herself, with an approach that is preferred over the increasingly banal and exclusively tactile interfaces, or the traditional interaction with a single point of contact, such as mouse or digital pens. Therefore, we take as hypothesis that **the combined use of touch and pen allows 3D modelling to be more appealing and accessible to novice users**, when compared to other approaches that use a single contact point for input.

To evaluate the advantages and disadvantages between different interaction methods, three types need to be devised: Pen-based, Touch-based and Combined Pen and Touch interactions. For the first interaction type, a pen device is used, simulating natural calligraphic interaction, with the aid of a set of simple menus to alternate between the pen functionalities (drawing and manipulation). On the Touch-based approach, users take advantage of multi-touch interaction to design models and use gestures to manipulate them. Finally, the Combined Pen and Touch interaction is comprised of both the previous methods, mainly using the pen interaction for sketching while using the multi-touch functionality for manipulation operations.

As for the interactive experience, the solution is complemented by the EnContRA 3D object retrieval system. Using EnContRA, it is possible to perform searches in order to return similar 3D objects. These results are then presented to the user in the form of an organized non-intrusive list of suggestions, from which it is possible to choose a desired outcome.

This project was carried out in collaboration with inEvo as part of the CluTCh project. Details about the CluTCh project are described in Appendix A.

1.4 Contributions

In order to evaluate our proposed solution and its variations, we developed a 3D modelling tool with support for multi-touch interactive surfaces. Likewise, the main contributions resulting from this work are:

Touch-based 3D object creation techniques Currently used methods for 3D modelling often rely on an interaction based of mouse and keyboard input. Although this conventional type of interaction has been extensively used over the years, with the emergence of new and more natural forms of interaction such as touch, we propose to develop and study sets of techniques that focus on touch-based, pen-based and combined pen and touch interaction for camera manipulation and 3D model creation, in a 3D modelling scenario.

Natural User Interface in a 3D modelling tool Most currently used 3D modelling tools do not incorporate or take advantage of 3D object retrieval systems, even though these and other multimedia retrieval systems have proven to be able to achieve satisfying results in real-time execution. We propose an approach for 3D modelling that takes advantage of the capabilities of a 3D retrieval system in a modelling context, while presenting the possible modelling outcomes in the form of an Expectation List.

CALIforUnity During the development of an early prototype in Unity3D, we had the need to integrate and use the CALI library in order to identify primitive strokes in hand drawn 2D shapes. At the time of the creation of this prototype, there were no known implementations of the CALI library built in C# that could be used in Unity3D. Therefore, we contacted the authors to ensure that we would be working with the latest stable release of CALI. With the permission of the authors, we translated the 1.2.1 C++ version of CALI entirely into the C# version CALIforUnity[1], so it could be used in Unity3D. This package is now publicly available online.

1.5 Document's Structure

In the following section we will present some previous work that is related to our problem. We will start by discussing some popular mouse-based 3D modelling tools and then we will present other 3D

[1] CALIforUnity, `http://web.ist.utl.pt/~ist168170/cali-1.2.1_for_Unity3D%20(v1).zip`

modelling works that are based in sketch interaction. Thereafter, we will present studies in the field of 3D object retrieval as a generic subject and particularly for sketch-based 3D object retrieval, query result visualization and expectation lists.

In section 3 we discuss our preliminary studies and early prototypes, while in section 4 we present our solution in detail. Section 5 describes our evaluation method as well as the results of user testing, in order to compare the different approaches. In the last section, we present our conclusions and notes for future work.

Chapter 2

Related Work

As the use of 3D models in computer graphics applications grew over the years, so did the need to develop tools to create and modify these models. Although these tools are presented to achieve a common generic goal, naturally, some of them are considered to be more appropriate to be used by experienced users over the others, having a high level of control over a volume's properties. On the other hand, other tools focus on presenting simpler interfaces, targeting a broader user group in which untrained casual users can be included.

2.1 3D Modelling Tools

A commonly used tool for modelling is Autodesk's 3ds Max[1]. It features a well built user interface (seen in Figure 2.1) and multiple capabilities for manipulation and animation of 3D models and images. It also includes features for the creation of models from point cloud data sets, which can then be divided as sections, limiting the data set to perform specific tasks and improve performance. Also, 3ds Max provides a chamfer tool, a modifier for 3D objects that allows the automatic and parameterized creation of beveled edges connecting two adjacent faces.

[1] Autodesk 3ds Max, `http://www.autodesk.com/products/3ds-max/`

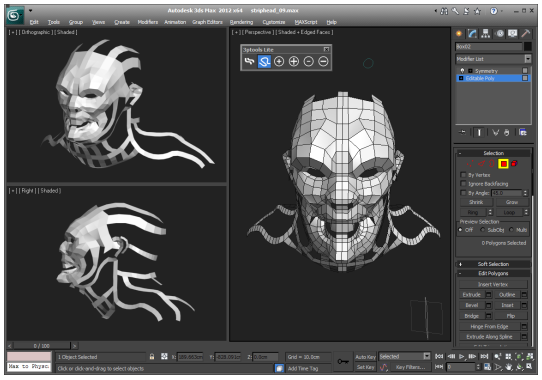


Figure 2.1: The user interface of Autodesk's 3ds Max.

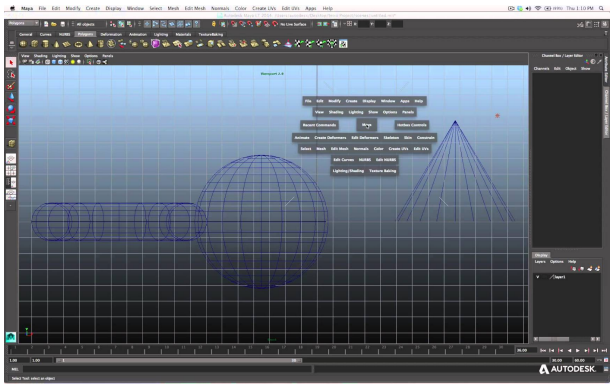


Figure 2.2: The user interface of Autodesk's Maya and the Hotbox.

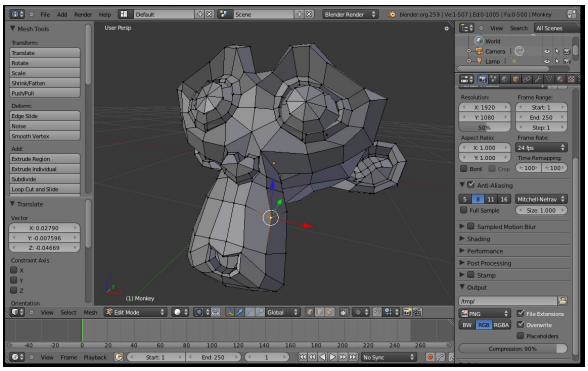


Figure 2.3: The user interface of Blender.

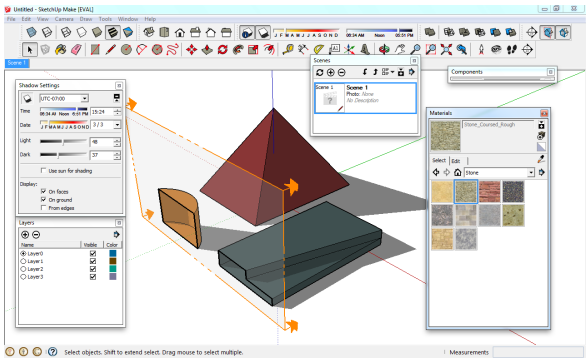


Figure 2.4: The user interface of SketchUp.

Regarding learning 3ds Max, the payed tutorials are made by experienced professionals and present thoroughly detailed information about the tool. Autodesk's 3ds Max also provides an extensive library of default materials, avoiding the need to acquire extra sets of materials externally.

Much like 3ds Max, Maya[2] is also a modelling tool published by Autodesk, although it is considered to be best fit to create visual effects for animated films. Compared to the previous tool, it presents a complex yet consistent user interface (shown in Figure 2.2), with easy to understand shortcuts and more options for visualization and appearance modifiers, causing it to be considered a better tool for artists. An important user interface feature is the Hotbox, a customizable and floating menu that is quickly toggled and contains every action from Maya's interface, divided in subsections referred to as marking menus. Once the Hotbox is toggled, the common menus (File, Edit, etc.) are located in the marking menu above the mouse cursor, while below are the menus of the individual menu sets. The Hotbox also features options for predefined view layouts, display individual interface elements, open windows or editors and selection masks. Both 3ds Max and Maya use architectures that support expansion plugins, to provide extra functionality as it is needed.

As a free and open-source modelling tool alternative to Autodesk's 3ds Max and Maya, there is Blender[3]. Blender provides not only functionality for 3D modelling but also for texturing, rigging, skinning, video editing, tracking and fluid and particle simulation. As of the previous two, Blender also supports expansion through Python scripting, but since the tool is open-source, plugin developers can take advantage of the publicly available information to create more suitable and better optimized plugins. Blender also provides a complex interface (seen in Figure 2.3) and a large set of keyboard and mouse shortcuts for a more agile workflow. Given that the tool is free and has a linear learning curve, it has an extended user group, and information provided by video tutorials and community forums is easily accessible on the internet.

Previously owned by Google, SketchUp[4] is also a 3D modelling software, focused on design and modelling for architecture, civil and mechanical engineering. SketchUp is directly connected with 3D Warehouse, an online repository that allows users to search, store and share free 3D models. Although Sketchup's user interface (depicted in Figure 2.4) might seem limited for being minimal, it provides fast

[2]Autodesk Maya, `http://www.autodesk.com/products/maya/`
[3]Blender, `http://www.blender.org/`
[4]SketchUp, `http://www.sketchup.com/`

and precise modelling, aided by dynamic snapping, input of exact values for distance, angle and radius, while hiding the technical mechanics below. Compared to the previous tools, it is considered to be a less technical tool, since it hides common modelling problems from the user (e.g. topology of faces), providing an easier and natural modelling experience. The models exported from SketchUp are also appropriate for 3D printing since they are production ready models.

In the following section we will present works that focus on the development and improvement of techniques for 3D modelling applications, particularly sketch-based.

2.2 Sketch-based Modelling

A classic example of sketch-based modelling is provided by Zeleznik et al. [2] through the SKETCH application. In SKETCH, the user creates new geometry through sequences of strokes that define the shape to be created and the details of its form. By representing three perpendicular lines, SKETCH acknowledges the input as a box with sides defined by the sizes of the lines. This method allows the user to create basic shapes very quickly, while being able to change them later. Although the application is simple, the original system provided only a small set of basic shapes, allowing the set of available gestures to remain fairly small, and the gestures themselves iconic of the shapes they represent.

As a different solution for processing the user input, Takeo Igarashi et al. [3] use in Teddy an approach that relies solely on the user's stroke rather than trying to understand what previously stored shape was the user referring to. This approach is named blobby inflation and uses the 2D silhouette drawn by the user to create a 3D inflated version of it. The process starts by finding the central spine of the silhouette and displaces the vertices of the spine plane to form a tessellated mesh dome that is mirrored, creating a symmetric and watertight model that is topologically equivalent to a sphere.

Compared to the previously stated method, blobby inflation is not limited by the size of stored dataset, since it generates new geometry procedurally, not needing to map a sketch into an already known shape. On the other hand, the blobby inflation method provides a lower degree of control, and is more

Figure 2.5: Examples of characters created using Teddy.

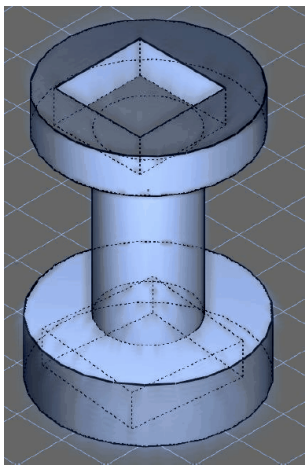


Figure 2.6: A model created in GIDeS.

appropriate for shapes with rounder faces (as seen in Figure 2.5) and with no sharp edges (since those are partially lost due to the inflation).

Fonseca et al. [4] present the GIDeS (Figure 2.6) system and study how to improve the usability of a CAD system in the early stages of product design, with a simple and learnable approach. To do so, the system provides paper-like interaction through a calligraphic metaphor, dynamic menus to display the state of the application without interfering with the task and the ability to construct precise drawings from sketches through simple constraint satisfaction. In order to keep the user engaged in the modelling experience, the system suggests shapes according to the current progress, supporting the user and providing a more responsive experience.

While using hierarchical implicit volume models (BlobTrees) as an underlying shape representation, Schmidt et al. [5] propose ShapeShop, a sketch-based solid modelling tool capable of reproducing solid models (with arbitrary topology), with complexity ranging from cartoon-like characters to detailed mechanical parts, using multi-touch input. As in Teddy, 2D contours are inflated into rounded three-dimensional implicit volumes. The underlying volume hierarchy is used as a construction history, allowing individual sketched components to be non-linearly edited and removed. Each BlobTree procedurally defines an implicit volume using a tree of primitive volumes and composition operators, such as CSG and blending. ShapeShop includes sketch-based operations for hole-cutting, oversketched blending and adding of surface detail.

Besides inflation, surfaces can also be created using linear sweeps, in which a drawn profile is interactively adjusted with a slider in order to create a controlled extrusion, or using surfaces of revolution (with spherical or toroidal topology), that revolves a sketch around an axis and thereby creating a model. Regarding the sketches, raw polylines drawn by the user are replaced by smooth 2D variational implicit curves to create discrete samples. For refinement, the user can perform gestures to remove or smooth groups of point samples, or to discard whole top layers of a sketch. In order to resolve viewing issues and depth-determination ambiguities, ShapeShop uses a dynamic cutting plane to edit regions obscured by other parts of the current volume.

Later, Lopes et al. [6] also added pen functionality to the ShapeShop solution, aiming to achieve a more fluid modelling experience, with less interruptions. In this new approach, the authors delegate the sketching operations to the dominant hand using the pen device, while manipulation operations are performed with the non-dominant hand via multi-touch input.

This division of labor in human bimanual interaction was studied in early works such as Guiard's [7], in which guidelines for human-computer interaction techniques are proposed, taking advantage of bimanual coordination. The author describes the dominant hand as the more suitable hand for fine movements and manipulation of tools, while the non-dominant can be used for coarse movements and setting the spatial frame of reference.

Newer works like Mockup Builder [8] take advantage of the distinction between dominant and non-dominant hand operations, to create and manipulate 3D shapes with familiar gestures. As in the pen enabled version of ShapeShop, Mockup Builder allows the drawing of shapes using the dominant hand, while handling non-dominant hand gestures as camera and world operations. Using on-and-above-the-

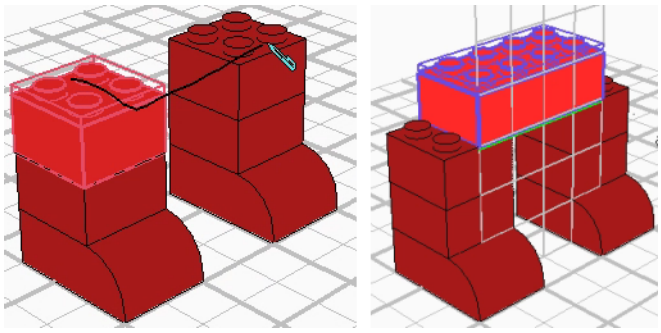


Figure 2.7: Modelling of a LEGO brick and respective outcome in the LSketchIt system.

surface interaction techniques on a tabletop interactive surface, 3D shapes can be created by using push and pull gestures, creating extrusions of previously drawn profiles. Although the authors were able to achieve satisfactory results, the application required a complex hardware apparatus in order to function.

Aiming at a 3D modelling system for LEGO models, Santos et al. [9] developed LSketchIt, a sketch-based modelling tool (as shown in Figure 2.7) that combines calligraphic interaction, a constraint solver and a retrieval mechanism. The user is able to add an object to the scene by drawing it. Then, the 2D sketch is transformed in a 3D sketch using the unprojection of points technique, that converts 2D into 3D points by computing the intersection of a ray from the user point of view with a plane. This 3D sketch is then used as a query, enabling the system to provide a list of results in which the user can select the object to be used.

The system's constraint module is responsible for the creation of connections between parts, propagation of transformations (translations and rotations) and maintenance of the gravity property (when a part is deleted or moved), while the retrieval mechanism relies mainly on the information about dimensions and categories of the parts in the database. The calligraphic interface allows over-sketching of gestures on parts, which enables the refinement of the search. In the following section we will describe other works that focus on 3D object retrieval using sketch-based queries.

Also for LEGO models, Mendes et al. [10] developed an interaction technique for interactive surfaces based on the building block metaphor using multi-touch. With this simplified modelling approach, users are able to elaborate complex LEGO constructions, manipulating the blocks in the 3D space via multi-touch gestures. Authors justified having atomic block transformations instead of combined as it allows a higher degree of control over the operation.

In this section we have presented works that chose take advantage of NUIs to provide a more natural experience to its users, rather than using the WIMP interaction approach. This direction is explained in the study of Jacob et al. [11], that describes reality-based interactions to be more appropriate as they take advantage and are leveraged by user's pre-existing skills and knowledge. Schwesig [12] also describe a user interface as organic if it is capable of triggering the suspension of disbelief, making intangible information feel as part of the tangible physical environment. While combining sensitive devices with responsive graphics, the input device and graphical user interface are experienced as a whole, not as independent elements of the interface.

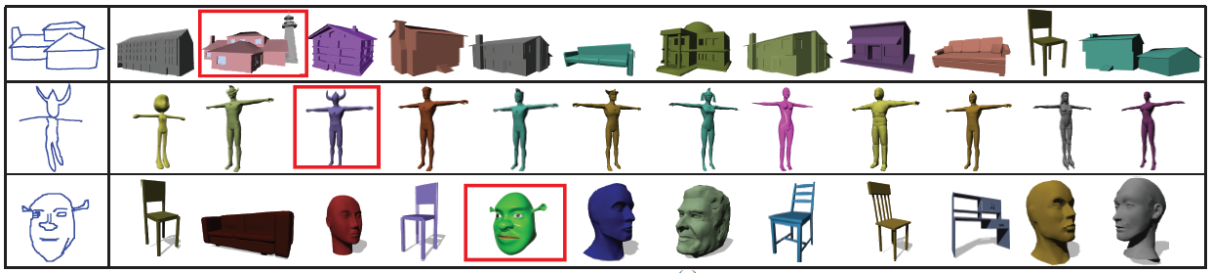


Figure 2.8: Examples of sketch queries and respective results using robust shape matching.

2.3 Sketch-based 3D Object Retrieval

In recent years, Saavedra et al. [13] developed an approach to sketch-based 3D object retrieval called STELA (Structure-based Local Approach). STELA consists of four main steps: get an abstract image, detect keyshapes, compute a local descriptor, and match local descriptors. The authors also define keyshapes as compositions of primitive shapes (straight lines), that allow the representation of an object in a higher abstraction level with the capability of dealing with noise.

The solution also combines the local approach with a global one, to take advantage of the global similarity in order to reduce the number of necessary comparisons between a query sketch and the models of a database. This global approach works as a filtering stage, increasing the efficiency of the local method and improving the retrieval effectiveness, reducing the number of false positives. For each 3D object in the database, fourteen viewpoints are created using suggestive contours, in order to create 2D representations of the object with less noise while keeping only the relevant edges. STELA also takes advantage of the concepts of structural information (what basic components make up a certain object) and locality information, which define relations between a reference component and the others around it.

In the same year, Shao et al. [14] also studied an approach to sketch-based 3D object retrieval using a technique called Robust Shape Matching. Just as in STELA, the input sketch is transformed into vectorized contours to enable the computation of a local similarity measure. Regarding the 3D object database, for each model, seven views (front, side, top and four corners of its bounding cube) are used for the creation of the vectorized contour representations. These vectorized contours only take into account pixels near important existing features, alleviating the inherent large memory consumption and ignoring noisy strokes and pixels near insignificant details. Then, the original contour images are turned into polylines via the Robust Moving Least-Square technique (RMLS), generating thickened contour images with emphasized corner points.

The solution also has the possibility of creating new models from retrieved examples, choosing the most similar 3D model to the query sketch and the deformation engine automatically deforms the model to match its contours to the sketch. This enables the toleration of distortions, but only to a certain degree. Three examples of results in the robust shape matching method are depicted in Figure 2.8.

Even more recently, Xu et al. [15] present Sketch2Scene, a framework that automatically converts freehand sketch drawings inferring multiple scene objects into semantically valid, well arranged scenes

Figure 2.9: Examples of input and respective output of the Sketch2Scene framework.

of 3D models (as shown in Figure 2.9). Sketch2Scene relies on the concept of structural groups, compact summarization of reliable relationships among objects in a large database of well-constructed 3D scenes that enables efficient and effective solutions to co-retrieval and co-placement, which are posed as combinatorial problems. A structural group is represented as a complete graph with each node representing an object category and each edge describing pairwise relationships between two object categories. The solution takes the input and divides the scene into a set of sketches, each of which corresponding to a single scene item.

Rather than having a 3D object database, the Sketch2Scene system uses a repository of relevant and complete 3D scenes, with pre-segmented elements and identified pairwise relationships. These relationships between object categories can be supporting relationships (identifying objects that go under and above other objects), spatial relationships (identifying the position and orientation between two objects), coplanar relationships (identifying objects that have coplanar sides of bounding boxes) or symmetric relationships (representing rotational or axial symmetry between multiple objects).

The sketch-based co-retrieval is done first finding top candidate objects independently for each sketch and then optimizing the combinations of the candidate objects. Silhouettes and canny lines from depth and normal images are used to produce the line drawing contours. In a second step, the top matched objects are reordered by comparing their matched contour images with the query sketch.

In an attempt to develop a 3D object retrieval method purely based on human gestures, Holz and Wilson [16] describe Data Miming, an approach that enables users to spatially describe existing 3D objects to a computer just as they would to another person. By capturing the movements of the user's hands, the Data Miming method is able to create a virtual representation of the user's description by building a discretized volume in a 3D space consisting of voxels, that represent the memory of the system.

In order to cope with the capture of unwanted gesture information (for example, in the moments between the end of a certain feature description and the beginning of another), each voxel has a weight value. This implies that voxels through which the user passes repeatedly or more slowly will have a higher count than voxels the user passes through when moving the arms to the next meaningful location. A simple thresholding across all voxels in the space leaves only the meaningful and relevant parts to be considered. In order to achieve a match, for each candidate object, the user-created model is aligned with the database model for comparison and measurement of similarity.

Taking advantage of immersion in a projection-based virtual reality environment, Nakazato and Huang [17] developed 3D MARS, a content-based image retrieval system. As the user chooses and

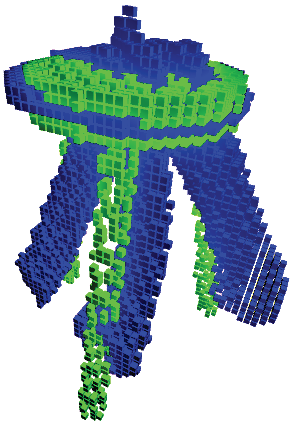


Figure 2.10: Example of query (blue) and respective result (green) in the Data Miming system.

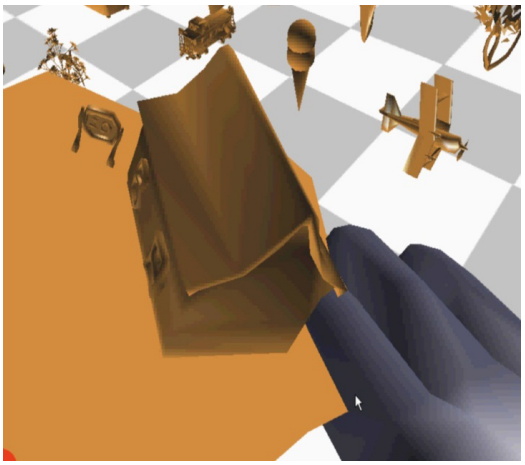


Figure 2.11: Query results in the 3D space of Im-O-Ret.

queries using examples (one or more), new results are shown in a 3D space, with the x-axis, y-axis and z-axis representing colour, texture and structure, respectively. The more similar an image is, the closer to the origin it is located. The result display also supports a ”sphere mode” that represents each image of the query result as a sphere, allowing a simple visualization experience while examining image clusters.

In a similar approach, Pascoal et al. [18] developed the Im-O-Ret (Immersive 3D Object Retrieval) prototype, a system that instead of retrieving the query results as a list of thumbnails, displays three-dimensional representations of retrieved models distributed in a 3D virtual space, organized according to their degree of similarity. The user is then able to navigate through the results and explore them thoroughly using a HMD display and a data glove. Each axis of the 3D space is assigned to a different shape matching algorithm: the x-axis is assigned to lightfield descriptors, the y-axis to a cord and angle histogram descriptor and the z-axis to a spherical harmonics descriptor. The modular architecture of the system enables the possibility of changing shape matching algorithms, in order to achieve more precise results. Figure 2.11 shows an example of a query result in Im-O-Ret.

2.4 3D Object Retrieval

Although multimedia information ranges from a wide variety of types, the need for efficient retrieval methods is a generic concern in all subtypes due to the typical limitations of non-textual information. Since the meta-data usually used to describe this information is created by human beings, there's a high risk of subjectivity as this information is most likely tied to the language, age, sex, culture and other characteristics of its creator. On the other hand, rather than considering meta-information associated with a given multimedia object, content-based retrieval of multimedia information relies on the intrinsic features of those objects, depending on their type.

A generic multimedia retrieval system is comprised of a database of multimedia objects and, for each object, a set of information that distinguishes that object from all the other remaining objects (e.g.

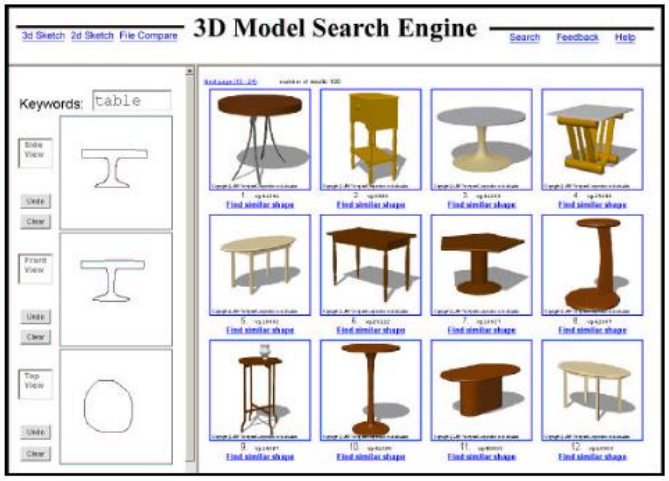


Figure 2.12: Query results in the Princeton Shape Benchmark [1].

semantic, syntactic, topological). As the system receives a query from the user, information must also be extracted from the query. Then, a matching phase occurs in which the system measures which objects previously stored in the database most closely match the queried information. In a final step, the system must display the best matches to the user. Additional details regarding previous works on 3D Object Retrieval are described in Annex B.

To describe and annotate multimedia information, MPEG-7[5] uses a standardization of metadata structures called descriptors and descriptor schemes. Descriptors are syntactic and semantic representations of features of a given multimedia object, that attempt to encapsulate important information about an object in a simpler data structure, allowing direct comparison. On the other hand, descriptor schemes define the structure of relations between descriptors (or even other descriptor schemes).

In particular for 3D objects, MPEG-7 standardizes region-based, contour-based and 3D shape descriptor as the main types of descriptors that have useful application for classifying shapes. Region-based and contour-based descriptors can be applied to representations of a 3D object in a 2D space, much like a view of the model from a certain viewpoint. While region-based shape descriptors extract information from that space that is occupied by the object representation in the frame (identifying regions and positional relations between those regions), contour-based shape descriptors capture the linear relations of the contours of the representation, being robust in terms of partial occlusion. Lastly, 3D shape descriptors focus solely on the feature extraction of 3D models. Additional information about specific MPEG-7 shape descriptors is described in Appendix C.

The standard method for visualization of query results in 3D object retrieval still relies on displaying to the user a list of thumbnails of the most similar shapes in relation to the queried information. This list of thumbnails (as seen in Figure 2.12) is often sorted according to rate of similarity of each object, displaying the objects that are most similar first. The interface used for showing these results very frequently is the same as the interface used to specify the query (i.e. computer monitor) although some systems use two or more viewports to present the user the query results while still enabling interaction

[5]MPEG-7, `http://mpeg.chiariglione.org/standards/mpeg-7`

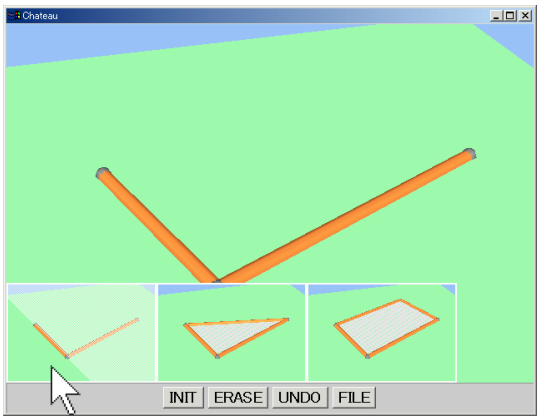


Figure 2.13: Expectation list in the Chateau system.

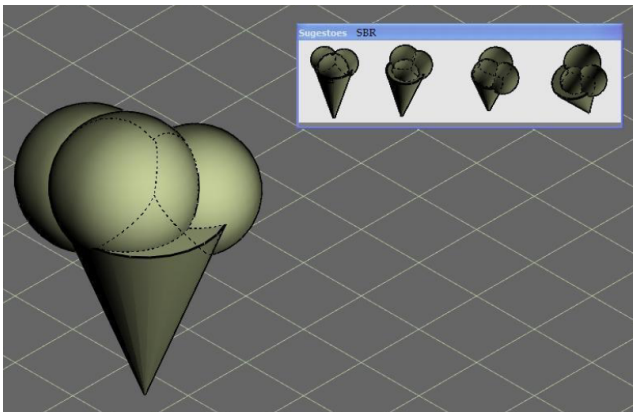


Figure 2.14: Expectation list of the GIDeS system.

with the query specification viewport, allowing query refinement.

2.5 Expectation Lists

As described by Igarashi and Hughes [19], it is difficult to precisely model 3D objects that have exact characteristics like symmetries or repeated substructures, using only gestural interfaces. To cope with this problem, Igarashi and Hughes developed a suggestive interface for 3D drawing as part of Chateau, a 3D modeling system. As a shape is drawn in the system (pictured in Figure 2.13), currently active edges are highlighted (as they are being created or modified), allowing the system to infer possible following operations and present them in a list of thumbnails. If a presented result is desirable, clicking the respective thumbnail completes the drawing as expected. This method of predicting and showing different possible outcomes to the user is often referred to as expectation lists. Chateau, in particular, generates suggestions whenever the user adds, erases, highlights or unhighlights edges, and uses different suggestion engines with mutually exclusive input patterns, preventing the display of an overwhelming number of suggestions.

While dealing with a more complex set of shapes in a 3D object retrieval scenario, it is likely that these shapes will be less different from each other, leading to a bigger number of scenarios of ambiguity while analysing the user's input. To address this problem, Fonseca et al. [4] and Santos et al. [9] also explored the concept of expectation lists, as a non-intrusive context-based dynamic menu that is triggered when the user's strokes are ambiguously recognized, displaying a menu that can be seen in Figure 2.14. Given that scenario, the user is able to choose from two or more possible interpretations what's the one that better suits his previous stroke stroke.

Schmidt et al. [5] also used expectation lists on ShapeShop to avoid the ambiguity while determining if a sketch should be interpreted as a blobby inflation or as a linear sweep. Instead of small rendered images of what the updated surface would look like for each expectation list icon (heavy on performance), the system uses color-coded representations that also distinguish the creation of new volumes from modifications of the current volume.

2.6 Discussion

Having been presented the currently popular tools for 3D modelling, relevant works in the areas of object retrieval, sketch-based object retrieval in particular, and sketch-based modelling, we will now present our discussion. Below we will describe in more detail our conclusions on the advantages and disadvantages of the presented works.

Regarding popular 3D modelling tools, 3ds Max is still a strong contender as it provides a complete workstation for 3D modelling in a single application. Although its user interface is considered to be well built, it did not suffer from major changes over the years, which doubles as an advantage for experienced users since it requires less effort to model using an earlier or older version of the tool.

Equipped with a complex interface, Maya not only offers in-depth options for 3D object creation and manipulation, as it also makes these options easily available, in the form of the Hotbox or through an extensive sub-menu hierarchy. With a similarly complex interface, Blender presents itself as free and open-source alternative to the previous tools. Blender as a free and growing 3D modelling tool, benefits from a large user community, with a broad set of readily available information online, making it a relatively simple tool to get started on. Both Maya and Blender provide an extended list of shortcut operations for mouse and keyboard but that often leads to redundancy as most of those operations are also nested in the standard menu hierarchy.

In a less technical approach when it comes to 3D modelling, SketchUp provides a simple user interface, well adapted for the modelling and composition of 3D objects for specific areas of architecture and engineering. Although it is a less versatile alternative when compared to the previously presented tools, it allows a fast modelling experience, aided by dynamic tools, without having to deal with model geometry problems (e.g. face topology issues).

Overall, both 3ds Max and Maya are professional tools suited for well-trained users, allowing a controlled in-depth modelling experience. Also, 3ds Max is considered to be particularly suited for modelling 3D assets for videogames, while Maya is generally used for cinematic and film purpose. As for the Blender, it excels in a casual modelling scenario, as it features a broad set of functionalities for 3D modelling while mastering none in particular. Finally, SketchUp presents itself as a capable tool for the sketching and conceptualization of architecture and engineering projects. In general, all these commercial tools' interfaces are based on the WIMP interaction approach (windows, icons, menus and pointer), using keyboard and mouse as the main source of input, being considerably less natural than a standard NUI (natural user interface).

In Table 2.1 and in regard to sketch-based studies, we compare the previously described works. From STELA to Im-O-Ret we present 3D object retrieval works and from SKETCH to Chateau we present 3D modelling works.

The work of Saavedra et al. [13] takes advantage of the structural and locality information, as well as of the global similarity to increase precision, but doesn't consider partial matching. In terms of keyshape detection, only takes in consideration straight lines, which leads to problems when considering shapes composed of rounder primitives.

	3D Object Retrieval	Gesture-based interaction	Calligraphic interaction	Expectation lists	High precision shapes	Visualization of retrieval results in 3D
STELA [13]	✓					
Robust Shape Matching [14]	✓					
Sketch2Scene [15]	✓					✓
Data Miming [16]	✓	✓			✓	✓
3D MARS [17]	✓					
Im-O-Ret [18]	✓					✓
SKETCH [2]		✓				
Teddy [3]		✓	✓			
LSketchIt [9]	✓	✓	✓	✓	✓	
GIDeS [4]	✓		✓	✓	✓	
ShapeShop [5]			✓	✓	✓	
Chateau [19]		✓		✓	✓	

Table 2.1: Feature comparison of the presented works.

In the work of Shao et al. [14], the interactive performance is achieved due to complex techniques such as acceleration schemes, transformation graphs and registration prunning, but the lack of a stop criteria when applying the search algorithm to the transformation graphs might result in a large computation time when considering large shape databases. Also, mismatching situations can be originated from the arguably low number of contour images considered (seven) for each shape and due to the fact that non-artistic users draw only proportions and locations accurately, making the match through global transformations not the best suit for the problem (aggregates small details instead of ignoring them).

These two works are very similar in terms of 3D object retrieval systems features, but STELA [13] goes a step further when comparing with the work of Shao et al. [14], by implementing a global feature approach in the matching process. On the other hand, the work of Shao et al. [14] has the advantage of taking in consideration negative features in the query specification process, enabling the possibility of more complex interrogations.

The Sketch2Scene [15] system, although a sketch-based 3D object retrieval system, is more focused on identifying a semantically valid scene, by taking as input a sketch of objects and considering their spatial and functional relations. Also, this system is able to detect desired touching faces and displays its results in a well organized 3D rendered scene. As a natural approach to 3D object retrieval, Holz and Wilson [16] devise Data Miming as a technique for mimicking features of a solid model in order to perform a query. Using a voxel space representation and voxel weight-based thresholding, a discretized rough volume is built in 3D space and used as an example for querying.

Regarding sketch-based modelling, the works of Zeleznik et al. [2] and Igarashi et al. [3] are similar in terms of simplicity of use when modelling simple and imprecise models, which suits the needs of a doodling application. Teddy [3], although robust and efficient for experimental use, is not suitable for

modelling shapes with sharp edges and does not support the specification of negative features in the drawing process.

In a more complex and precise modelling context, the works of Fonseca et al. [4], Igarashi et al. [19] and Schmidt et al. [5] are successful mostly due to using natural and suggestive user interfaces that take advantage of expectation lists to provide a fluid interaction with the user. Also, the work of Fonseca et al. [4] takes advantage of calligraphic interaction and a global feature approach in a 3D object retrieval context.

On a pure 3D modelling context, Schmidt et al. [5] also use calligraphic interaction in ShapeShop, a versatile tool able to reproduce a wide variety of solids ranging from Teddy blob like characters to detailed exact models. Using a structured hierarchy based on implicit volume models (BlobTrees), allows ShapeShop to be a more flexible tool by being able to keep track of the construction history, enabling the modification of individual components at any moment of the modelling process.

The work of Santos et al. [9] also takes advantage of the use of expectation lists and, although focusing in modelling using pre-existing primitives, also has significant modelling improvements due to the constraint solver, useful tool to determine simple relationships between parts of a shape. The system is able to detect colliding surfaces and maintain gravity while moving parts.

In terms of query results visualization for 3D object retrieval, the standard approach still relies on displaying to the user a list of thumbnails of the most similar shapes in relation to the queried information. With the objective of achieving a more complete visualization experience, Nakazato et al. [17] and Pascoal et al. [18] applies immersion techniques to their works, allowing the visualization of the results in a 3D space in which each dimension can represent a different visualization criteria.

Not only considering the spatial positioning of each result in the 3D space, the Im-O-Ret [18] system also displays each individual shape as a rendered 3D model, allowing thorough inspection through manipulation. Although suitable for the visualization of a large set of results, pure immersion techniques are not optimal for scenarios of 3D object retrieval or modelling, mostly due to the fact that they require specific visualization hardware. Also, regarding the order of the results in the visualization, an organized list (that can consider an average ranking based on multiple criteria) is often sufficient to display the results that most resemble the queried information.

Considering the above, none of the present works is able to combine calligraphic and multi-touch interaction with an interactive and natural user interface (aided by 3D object retrieval and expectation lists), in a walk-up-and-use 3D modelling context. Therefore, we propose to develop an approach that takes advantage of those features, designed so that the generic user, trained or non-trained, can use it to create appropriate models for 3D fabrication.

Following this chapter we present our preliminary exploration in which we test different approaches and technologies to be part of a possible solution, taking in consideration the advantages and disadvantages of the presented related work.

Chapter 3

Preliminary Exploration

As a preliminary study, we analysed different types of interaction and technologies. Regarding interaction, we decided to compare touch-based interaction (as the only source of input) and combined pen and touch interaction, as works such as [6] have proven that the use of a pen in combination with touch interaction may offer several interaction advantages. As to touch handling, it is important to decide what is the best protocol to use, so we considered both native Windows touch and the use of the TUIO protocol. In terms of technology we compared early prototypes implemented in both Unity3D and threeDart. Finally, having discussed the different possibilities for the development of a solution taking in consideration our objectives and available resources, we defined the desired characteristics for a final prototype.

3.1 Touch-based Interaction

As touch-based applications become increasingly more popular, the use of touch as a means of natural interaction has become increasingly widespread. Considering this, at first we pondered to develop a solution that relied solely on touch as form of interaction. Although most mobile devices to this day rely heavily on single touch input for the majority of their interactions, we aimed at developing an approach that could take advantage of multi-touch interaction.

In early prototypes, we explored the possibility of using both single-touch and multi-touch for different tasks. For example, in a 3D modelling context, single-touch is used for free-hand shape creation while multi-touch is used for camera manipulation (the angle and distance between two simultaneous touches control the camera rotation and position). Using a single contact point for shape creation allows a controlled interaction, and closely resembles a free-hand drawing metaphor. Similarly, the multi-touch interaction simulates real-world grabbing and manipulation techniques, appropriate for spatial transformations. This type of interaction can provide a natural experience by having separate functions for each type of touch interaction, possibly allowing the manipulation of the camera as the user draws. Although this possibility seems interesting, in very few scenarios it is viable to manipulate the camera and elaborate a sketch at the same time. Also, given that multi-touch surfaces are not always precise, problems easily arise from both undesired or unread touches, possibly causing a multi-touch gesture to

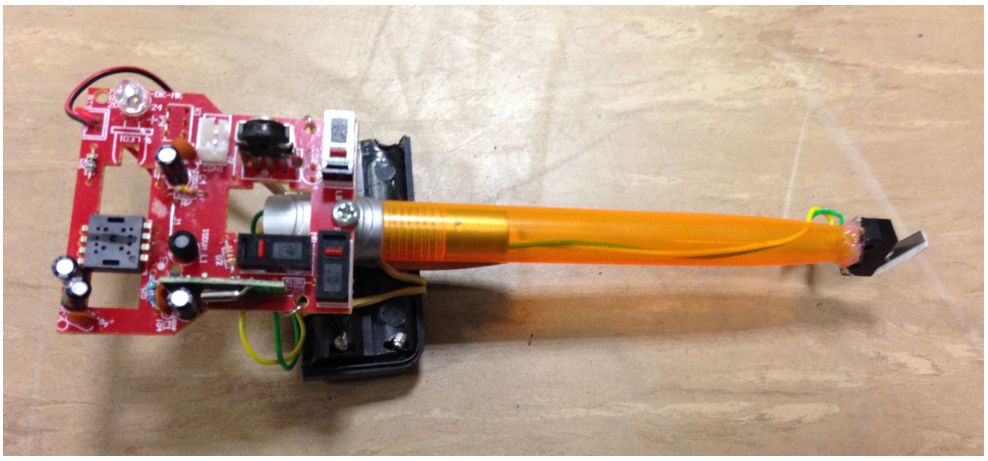


Figure 3.1: Prototype pen.

be misinterpreted as a single-touch interaction.

3.2 Combined Pen and Touch Interaction

Due to the possibility of using a pen in combination with the multi-touch interaction, we considered experimenting the approach in an early prototype. Compared to the single-touch, the pen input allows a more precise interaction since it is unambiguisly detected, being closely similar to mouse input.

In order to prove the concept, in an initial phase we built a prototype pen (pictured in Figure 3.1) based on a wireless mouse peripheral. We mounted the board of the mouse directly on the case of a standard ball-point pen, and wired a middle mouse button (using a momentary switch) directly on the tip of the pen casing. This prototype is powered by a single AA battery and is connected to a computer via a USB Bluetooth adapter, just like the mouse we used would. Although this pen only emits mouse middle click events (button up and button down) it is enough to add a new layer of depth to the interaction. When the user holds the pen against the interactive surface, both a middle mouse click event and a touch event (detected by the interactive surface itself) are fired. Since the mouse event occurs shortly after the touch event, the two can be then be paired, marking the new touch as pen input, so that it can be treated differently from the standard multi-touch input in the above layers of the application.

Initially, the pen device was wired to emit left mouse button clicks instead of middle mouse button clicks. Due to ambiguities caused by touch inputs in Windows machines, that detects tap and tap-and-hold touch gestures in order to emit left and right mouse button clicks, we chose to wire the pen to emit middle mouse button clicks.

3.3 TUIO vs Native Touch Input

Having the need to explore different technologies for input handling, we used both the TUIO protocol and native touch input handling in early prototypes.

Figure 3.2: Usual setup of the used interactive tabletop.

TUIO[1] is an open framework that defines a common protocol and API for tangible multitouch surfaces, that allows the transmission of an abstract description of interactive surfaces, including touch events and tangible object states. This protocol encodes control data from a tracker application and sends it to any client application that is capable of decoding the protocol. The combination of TUIO trackers, protocol and client implementations allows the rapid development of table based tangible multitouch interfaces. TUIO has been mainly designed as an abstraction for large interactive surfaces, but also has been used in many other related application areas.

Although TUIO is consistent over a wide range of platforms, it requires that the platforms have support for the framework, either natively or made publicly available by other developers. On the other hand, the native touch input offered by an operative system grants a higher level of control over the input. As it is preferrable to use native touch input, we could only do so if the technology to be used to implement our prototype can support native touch input from our target range of multi-touch surfaces (from tablets to interactive tabletops). Since our early prototypes were developed using the version 4 of Unity3D, that is missing support for Windows 7 native multi-touch (present on our interactive tabletop 3.2), we chose to use the TUIO framework, supported in Unity3D by the TouchScript[2] package.

3.4 Unity3D vs threeDart

Due to its fast development cycles, we initially chose to use Unity3D[3] to develop our early prototype (as seen in Figure 3.3). With its simple interface and workflow, it allows components to be monitored and manipulated during runtime, while granting portability to a wide variety of devices and operative systems. We also decided to build this prototype from scratch, not using other existing projects as a starter point, as the full control over the prototype functionality was a desired characteristic.

In the first prototype we focused on developing a tool capable of modelling simple volumes only using tube-like primitives in 3D space. Instead of allowing the camera to have completely free movement, we chose to adopt an approach in which the camera always stays directed at the origin of the 3D space.

[1] TUIO, `http://www.tuio.org/`
[2] TouchScript, `http://touchscript.github.io/`
[3] Unity3D, `https://www.unity3d.com/`

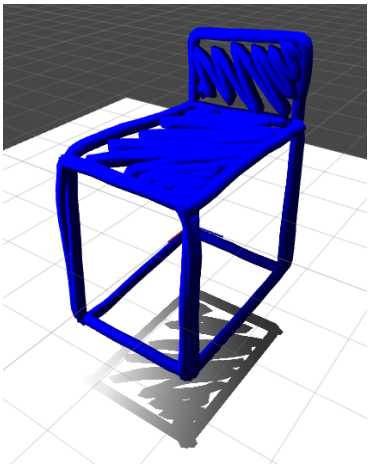


Figure 3.3: Modelling using the Unity3D early prototype.

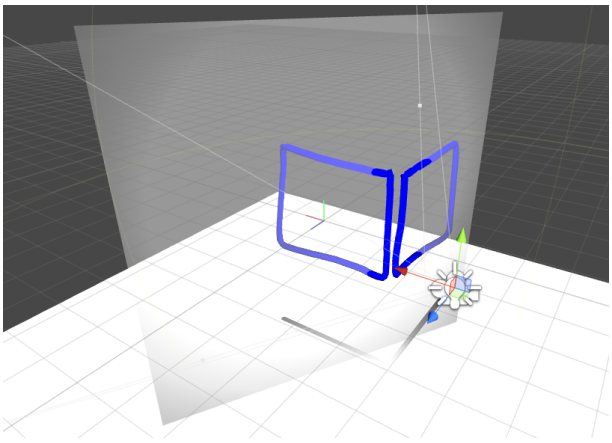


Figure 3.4: Drawing plane of the Unity3D early prototype. The plane can be seen intersecting both the ground plane and the current sketch.

With this constraint, the user is able to move the camera by orbiting around the origin (as in a spherical coordinate system) and also adjust the respective distance, allowing the 3D space to be equally covered and yet maintaining a consistency level that is useful when creating symmetric volumes.

To allow the user to draw, there is a transparent drawing plane (seen in Figure 3.4) grid perpendicular to the camera direction. When the user specifies a 2D sketch, the points of the sketch are projected onto the drawing plane, giving it three-dimensional position in space and creating a tube-like primitive through the newly created three dimensional points. Given the transparency of the drawing plane, it is simple to understand where the plane is cutting previously drawn sketches, allowing the composition of new strokes in precise positions. Shadows cast on the floor of the scene also contribute to the better understanding of the relative position of the existing strokes.

Later on, in order to achieve better consistency regarding the position of the drawing plane, we decided to keep the camera orbiting freely but having the drawing plane rotation snap every 15 degrees (parameterized value), as it is simpler to model perfectly perpendicular and parallel faces while the values of the plane angles are multiples of 15 degrees. Also, to provide a more natural workflow, we introduced the ability to momentarily change the orbit point of the camera once a stroke is finished. This technique allows the next rotation of the camera (and drawing plane) to be executed around the last point of the last stroke, consistently keeping this point on the surface of the drawing plane, as it is likely that the user will want to specify the next stroke based on the last one. Using this method, the specification of multiple continuous strokes in different planes of the 3D space is allowed.

Although the creation of 3D volumes using only free-hand strokes achieved the desired goal for the early prototype, we also decided to use the CALI [20] library, a 2D scribble recognizer, to allow the user to replace a free-hand scribble of primitive shapes with respective exact representations. As stated before in section 1.4, we decided to translate the original C++ non-trainable version of the CALI library into a now publicly available C# version. The early prototype now supports all of the 2D primitives supported by the non-trained CALI library: line, circle, rectangle, triangle, diamond, ellipse and cross.

With the addition of the CALI library to the prototype, each time a scribble is submitted for recognition,

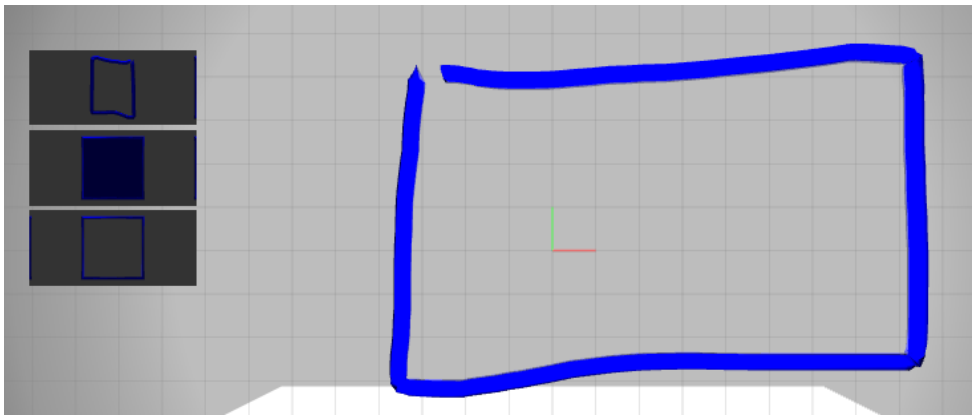


Figure 3.5: Expectation list of the Unity3D early prototype. In this case, it suggests the unmodified stroke, and two exact rectangles (filled and empty)

a list of recognized shapes is created, becoming necessary to question the user about the desired shape. Having understood the advantages of using expectation lists (Section 2.5), we chose to implement a basic interface comprised of small viewports on the left side of the application view, to display the possible outcomes of the recognition to the user. In each outcome the result is 3D rendered and rotated so that the user can fully analyse each option. The expectation list of the Unity3D prototype is depicted in Figure 3.5.

In every shown suggestion of the expectation list, the first result always resembles the raw stroke, as the user might desire to keep it instead of replacing it with a primitive shape. If the user acknowledges one of the options is more suitable, a simple touch on it automatically changes the raw stroke with the desired primitive. On the other hand, ignoring the current expectation list will simply make it disappear once a new stroke is detected.

Given the possibility to integrate this work as part of an international project with the support of inEvo, a portuguese startup company, we had to consider different alternatives in terms of technology. inEvo is a company with extensive experience in 3D modelling applications and is currently using modern technologies focused on developing applications for the web platform. Due to licensing issues, development using Unity3D was not an option. However, one technology currently used by inEvo is Dart[4], a scalable development platform for both server-side and client-side browser-based applications. Since Dart client applications run in a browser environment, portability is assured for most computers and mobile devices. Also, we found that having inputs, such as touch, abstracted for different devices allows a much simpler approach regarding input event handling, since it avoids the need to implement a new touch interface for each device used.

Available for Dart and specially focused on the development of 3D applications for the web, there is the threeDart[5] open-source library, a Dart port of the three.js[6] library, that provides foundations for the development of modern WebGL applications. Using threeDart we were able to develop a second yet simpler prototype, capable of creating similar tube-like shapes for the composition of complex shapes,

[4]Dart, `https://www.dartlang.org/`
[5]threeDart, `https://github.com/threeDart/three.dart`
[6]three.js, `https://github.com/mrdoob/three.js`

although at the time we did not implement other features like expectation lists or 2D shape recognition. With this second early prototype we were able to evaluate the advantages and disadvantages when comparing the development process using the two different technologies.

3.5 Summary

In a first prototype developed using Unity3D we were able to explore a different approach to 3D free-hand modelling, using the projection of two dimensional strokes into the 3D space using an orbiting drawing plane and camera, allowing the user to draw a 3D sketch freely in space. Having the opportunity to integrate a development team at inEvo and develop this work as part of an international project, we considered using Dart and the threeDart library. Although Unity3D as a technology provides fast development cycles for rapid prototyping it still lacks support for the broad range of touch interfaces.

On the other hand, although threeDart and Dart itself are new technologies still in development, they provide a higher level of abstraction regarding input handling (since this abstraction is provided by the web browser itself), while having a higher degree of control of the graphical pipeline when compared to Unity3D, due to threeDart being based in native WebGL. Also, having known that while using threeDart we would have full support from inEvo, we finally decided that we would use it as the main base technology for the final prototype.

Regarding the types of interaction, we consider that it is appropriate to evaluate the advantages and disadvantages between pen-based, touch-based and combined pen and touch approaches. In the following section we will describe how we have developed our solution with threeDart, that takes advantage of the three different and yet consistent interaction techniques.

Chapter 4

3D Modelling on Interactive Surfaces

Having previously described and discussed relevant works towards the presented problem, we have conducted a preliminary exploration phase in order to assess the best methods for the development of an appropriate solution. In order to develop a walk-up-and-use 3D modelling solution that combines natural interaction methodologies with a retrieval system aided by expectation lists, it is necessary to design an architecture that supports the different types of interactions equally, while maintaining a consistent modelling workflow regardless of input type. In this chapter we will describe in detail and justify our proposed solution.

4.1 Architecture

The architecture used in our solution is described in Figure 4.1. In this architecture, we support input interfaces for mouse, keyboard, multi-touch and pen interaction. As described before in section 3.2, the pen interaction is a special case of both touch and mouse input. In the web browser layer, the input data received from both the multi-touch surface and the mouse are abstracted from the application. Once the touch and mouse input information reaches the Dart application, it consists simply of pointer input events that contain information about the state of the pointer (either pointer down, up or move), normalized 2D screen coordinates and an identifier for each current active touch or mouse input.

It is also important to note that the keyboard input module is considered in Figure 4.1 as part of the architecture of the solution for multiple reasons. During the development process of the solution, keyboard input was often useful in order to allow the emulation of a second pointer input when working in a single-pointer environment (e.g. using the mouse). Also, keyboard input was a desired feature in the scope of the CluTCh project.

Additionally, while prototyping the interaction using a combined approach, a major problem with Dart applications was discovered in regards to using both mouse and touch input simultaneously. In Dart browser-based web applications, when a mouse event occurs, all currently active touch events get automatically discarded by the application (and vice-versa). As it is needed to have the pairing of touch and mouse inputs for the pen to work correctly, an alternative solution was adopted. Using an

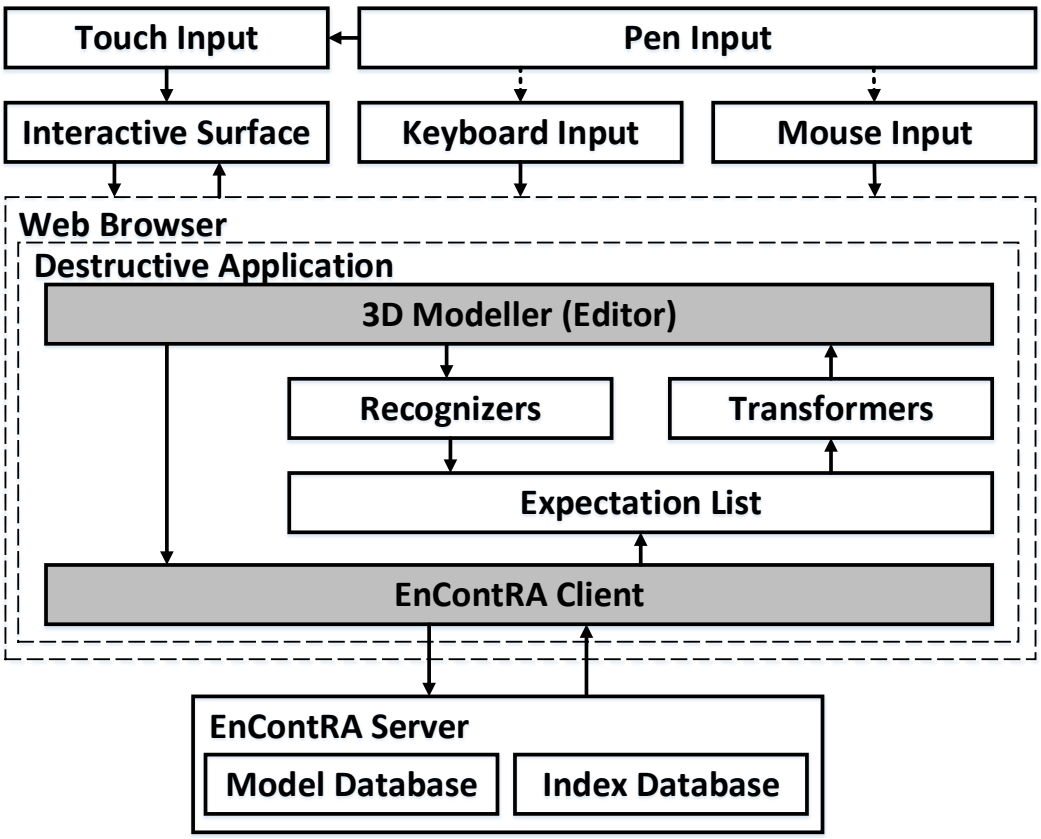


Figure 4.1: Destructive Application architecture. The grayed out modules represent the modules to which we mainly contributed.

additional second machine and a simple *.NET* application, we were able to receive the pen click input on the second computer and send an HTTP request via the LAN network to the main computer running the Dart application, which would then emulate a Windows key-down event to be captured by the Dart application. By doing so, the desired behaviour is achieved as the key-down event is used instead of the middle mouse click event to identify the pen.

Regarding the modelling application, it consists of a 3D modeller module (Editor) for shape creation and visualization, a module for creation and display of expectation lists, a retrieval client and sets of recognizers and transformers. In the application, recognizers are active agents that receive information once a stroke or model is created or modified, producing lists of possible outcomes taking in consideration other objects present in the Editor. The currently implemented recognizers can identify 2D free-hand strokes to produce 3D free-hand outcomes or exact primitive shapes (Axis and CALI recognizers), and stroke proximity to produce combinations with existing objects in the Editor (Intersection recognizer). Once the recognizers evaluate new information, the possible outcomes can be submitted to the expectation list, to allow the user to chose between a set of operations.

Similarly to the recognizers, transformers are also a set of active agents in the application. Transformers are responsible to convert outcome possibilities (created by the recognizers) into actual transformations. Once a user chooses an option from the expectation list, the transformation is assigned to

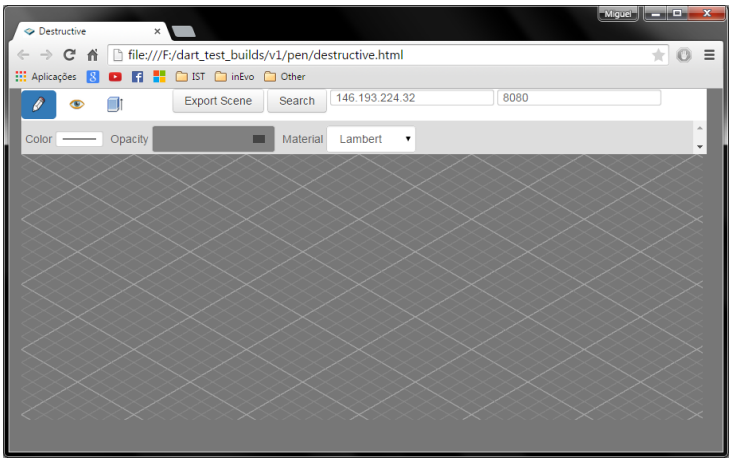


Figure 4.2: Initial state of the application.

the respective transformer and the change is applied. It is important to note that the transformers are also able to undo these operations, if needed. The currently implemented transformers can transform free-hand 2D strokes as 3D strokes or as exact primitive planar shapes (Axis and CALI transformers), transform a path of points as a basic sphere or a blob shape (Sphere and Teddy Blob [3] transformers) and transform two intersecting shapes as a new complex shape (Extrusion and Boolean transformers). Finally, the 3D object retrieval functionality is assured using the EnContRA Client, that is connected to an EnContRA REST Server (implemented in Java) that hosts a set of models and their respective indexes.

Although the focus of this work resides in the implementation of modelling tools and input handling for the 3D modeller module of the Destructive application, additional work has been done in order to better adapt EnContRA (retrieval system) to our solution, thus resulting in the creation of the EnContRA client for Dart.

It is also important to note that although this solution was developed using an interactive tabletop in particular, the application is able to be used in any interactive surface (e.g. tablets, smartphones and other touch-enabled surfaces), and in any computer using mouse and keyboard, as long as the device supports a web browser.

In the following Sections 4.2, 4.3 and 4.4 we describe the overall modelling, camera and object manipulation functionalities, respectively. In Sections 4.5, 4.6 and 4.7 we describe additional interaction-specific details.

4.2 Modelling

Given that the aim of this study is to develop an solution that can take advantage of both multi-touch and pen interaction, a sketch-based approach was deemed appropriate for the 3D modelling component. Similarly to other works presented in section 2.2 such as Teddy [3], Chateau [19], GIDeS [4] and LSketchIT [9], the modelling process relies on the capture of the user's free-hand draw, that is then analysed in order to determine the desired outcome. In our approach, once a user finishes a stroke and the possible outcomes are computed, the user can be asked to choose a possible transformation through a

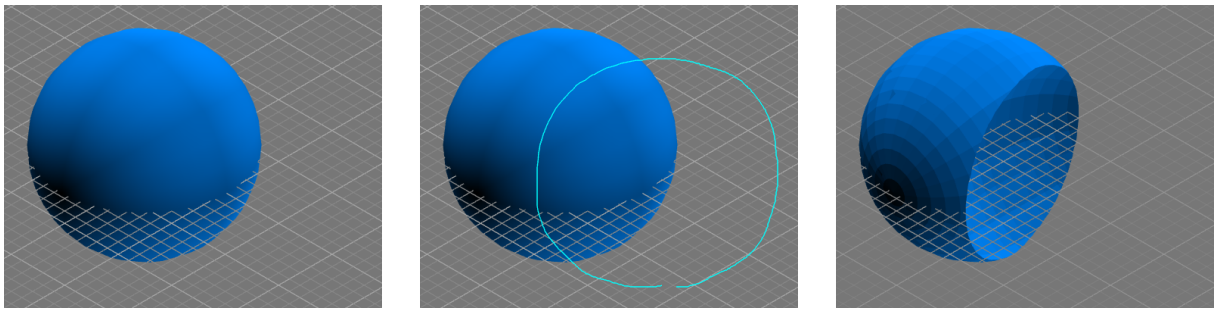


Figure 4.3: Boolean operation process applied to a basic sphere. Drawing a line over the existing sphere applies the transformation, creating a new shape.

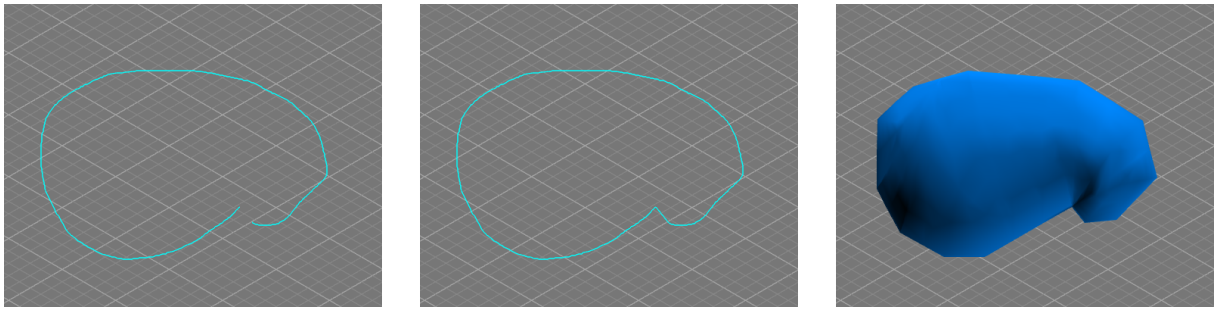


Figure 4.4: Creation of a Teddy Blob. Having drawn a semi-closed profile, the user is able to choose to close the profile. Then, the user is able to transform the closed profile into a similar blob.

non-intrusive interface (expectation list).

4.2.1 Primitive Shapes

Having the application in its initial state, as in Figure 4.2, the user is presented with a gridded surface in the three-dimensional space that resembles the ground level of the modelling scene. In this state, although using an orthogonal view, camera is lightly angled towards the ground plane and slightly rotated to the left, giving the user depth perception. As in any other sketch-based modelling solution, using a draw tool, a user is able to draw a two-dimensional line on the screen surface.

By drawing one or several 2D strokes, the user is able to compose a 2D shape that can later be used for the creation of a 3D model with similar characteristics. To do so, once the user is satisfied with a suggested outcome, the 2D stroke is projected onto the scene's ground plane, locating the stroke in the 3D space to create the new model. The whole purpose of the scene's 2D strokes is to define geometrical boundaries for the creation of 3D objects, therefore serve no purpose by themselves.

4.2.2 Boolean Operations and Blob Shapes

Having transformed 2D strokes into 3D objects, the user is then able to compose complex models by manipulating objects in the 3D space by rotating, translating and scaling. Lines drawn over other existing 3D objects can result in boolean operations, allowing the cut of basic primitives using non-linear profiles, as seen in Figure 4.3. Also, the user is able to create non-primitive solids drawing a 2D profile and choosing the Teddy Blob (as in the work of Igarashi et al. [3]) outcome from the expectation list.

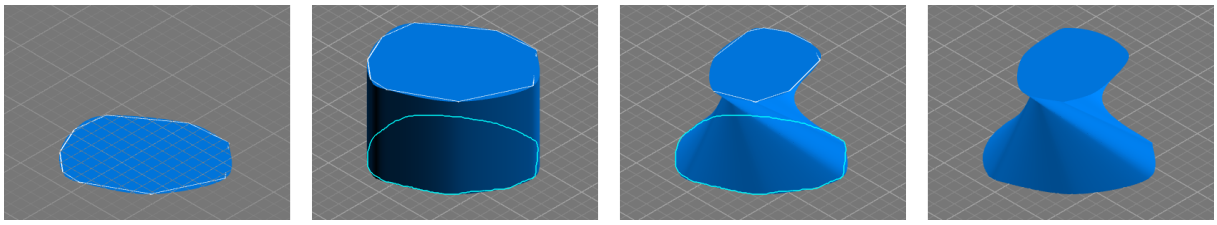


Figure 4.5: An example model created using Swivel Extrusion. Starting the extrusion on a closed profile, the user is then able to extrude the shape and manipulate the profile. Tapping finishes the extrusion

As expected, the result is a blobby inflation that inflates the 2D profile, creating a 3D symmetric and watertight mesh as seen in Figure 4.4.

4.2.3 Swivel Extrusion Tool

As an alternative to the drawing functionality, the Swivel Extrusion method was created. Using Swivel Extrusion, the user is able to create complex extruded shapes with high level of control (an example is shown in Figure 4.5) based on a closed 2D profile that can be represent anything from an exact straight shape to an arbitrarily shaken free-hand stroke. The user can control the extrusion using two distinct areas of the screen. On both left and right side areas of the screen (about 20% of the screen-size on each side) pointer input is used to control the height level (z-axis) of the closed 2D contour, defining the current height of the extrusion. On the remaining center screen area, pointer input controls transformations to the closed 2D profile. These transformations include rotation, scale and displacement (on the x and y-axis) of the profile, allowing the creation of complex models. All three profile transformations can be individually toggled on or off, since controlling all three transformations at a single given time can be excessively complex. Other control details for the Swivel Extrusion method are dependent of the interaction type and are described ahead in this section.

4.3 Camera Manipulation

In order to allow a satisfying and fluid experience, it is crucial to provide natural controls for camera manipulation to the user in a 3D modelling context. To allow the user to freely navigate the scene we provide three different camera manipulation options: panning, zooming and orbiting.

Panning actuates by translating the camera on the x and y-axis, while maintaining a constant distance from the scene floor (z-axis is fixed) and the camera rotation. Zooming is achieved by narrowing or widening the camera's FOV (field-of-view), displaying a larger or smaller portion of the scene, therefore creating the illusion that the camera is moving forward or backward. Finally, camera orbiting is done by using both panning and zooming to simulate spatial orbiting around a given point in 3D space. This orbit manipulation only allows the camera direction to be in a vertical anlge between 15 and 90 degrees in reference to the scene floor. Vertical angles below 15 degrees cause problems when trying to cast rays from camera to the scene floor (camera direction becomes almost parallel with the ground plane) while vertical angles above 90 degrees unnecessarily tilt the camera upside down. Although the camera can

not be manipulated to be positioned under an object or the scene floor, users can fully inspect an object by manipulating it.

Adjusting the camera's FOV is an unusual method for the manipulation of both camera zooming and orbiting. This approach is used due to the special architecture of the Editor module. As a generic component, the Editor module is based on a generic 2D scene that can contain other 3D scenes, and subsequently 3D scene nodes (objects). Using this architecture, the Editor is not necessarily bound to be a 3D element, becoming more flexible component. Since the camera used in the described solution is fixed directly on the 2D scene as a 2D object, it is not positioned in the z-axis, requiring adjustments on the camera's FOV to simulate zooming or orbiting.

Camera manipulation occurs when a specific gesture is applied directly to the scene, and not on any existing selected object.

4.4 Object Manipulation

In order to compose complex models, the presented solution allows the basic manipulation of individual models. In this section, we describe the generic information for object manipulation, while in the interaction-specific sections below we describe specific details in regard to each interaction type. An object can be transformed through translation in the 3D space, simple scaling (proportional in all three axes) and individual axis rotation. For consistency purposes, these object transformations are analogous to the camera transformations: pan, zoom and orbit, respectively.

To maintain an unambiguous environment, the manipulation of a given object can only be done if a gesture is applied to it and the object itself is selected. Once an object is selected a transparent sphere is displayed around the object, meaning that the object is ready to receive transformation commands (seen in Figure 4.6). Naturally, a selected object can also be deselected using the same action (a simple tap gesture). One or more objects can be currently selected at a given moment although transformation gestures only affect the object targeted by them.

As pointer input information is based on 2D screen positions, it can not contain enough information to describe transformation information for all three axes of the 3D space. Therefore, for this solution

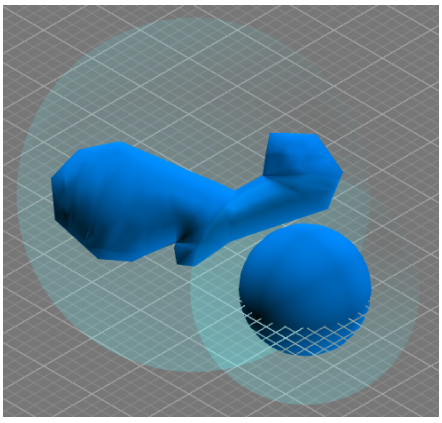


Figure 4.6: Selected Teddy blob (top left) and sphere (bottom right).

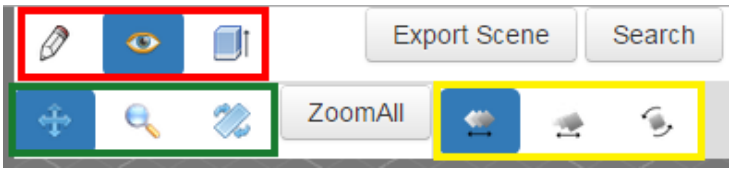


Figure 4.7: Set of menus of the Pen-based approach. On the top (marked as red) are located the draw, manipulation and Swivel Extrusion modes. Inside the manipulation mode, the camera manipulations (pan, zoom and orbit) are marked as green and the object manipulations (translate, scale and rotate) are marked as yellow. Camera pan and object translation are the currently selected manipulations.

we chose to define translation and rotation transformations as camera orientation dependent (similar to the camera manipulation in [10]). By doing so, at any given moment, an object can only be translated or rotated in a maximum of two axes. The axes to be used for these transformations are decided given the current camera orientation. For example, if the camera direction is more closely aligned with the x-axis (x-axis the most perpendicular axis to the view plane), then the other axis to be considered for the translation and rotation transformations are the y and z-axis. This approach not only avoids overwhelming the user with extra menus or complicated input combinations, as it also allows precise positioning and rotation, in an familiar manner.

Having described the overall details regarding our solution, we will now describe particular details related to the different types of interactions.

4.5 Pen-based Interaction

Although the focus of this work is the development of new techniques for a sketch-based 3D modelling environment taking advantage of combined pen and touch interaction, it is important to devise and evaluate other techniques to measure and compare its advantages and disadvantages. To do so, we propose a simple pen-based interaction, as used in other previously presented works such as Chateau [19] and ShapeShop [5]. While using this approach the user utilizes only the pen prototype described in Section 3.2.

4.5.1 Pen-based Modelling

In this pen-based approach for our solution, we focused on defining clearly the different steps of the modelling process, presenting the user with the draw, manipulation and Swivel Extrusion main menus. In the draw section the user is able to specify strokes freely, while accessing the expectation list in order to transform drawn shapes into 3D objects. As described in the generic Section 4.2, these transformations occur by transforming 2D drawn shapes into primitive, complex (boolean operations or Teddy-style blobs) or Swivel Extrusion shapes.

4.5.2 Pen-based Camera and Object Manipulation

While on the manipulation main menu (presented in Figure 4.7), the user is able to select and manipulate existing 3D objects in the scene, as well as manipulate the camera. By default, both camera and object manipulation settings are set to pan and translation, respectively, allowing the user to freely move a selected object on the designated plane (defined by the camera orientation, as described in Section 4.4) by gesturing a stroke starting from the object location to the object's desired final position. On the other hand, performing the same gesture starting over the scene or over an unselected object performs the camera pan manipulation. The selection or deselection of an object is performed by tapping an unselected or selected object respectively.

Also on the manipulation main menu, the user is able to change the default settings for both camera and object manipulation. For camera manipulation, a sub-menu is shown to allow the user to choose between camera panning, zooming and orbiting. Similarly, for the object manipulation the user is able to choose from another sub-menu between object translation, scale and rotation.

4.5.3 Pen-based Swivel Extrusion Tool

As described in Section 4.2, the Swivel Extrusion method can be used once a closed stroke is ready to be used on the scene. Entering the Swivel Extrusion main menu as a closed stroke is on the scene creates a mesh with the profile of the closed stroke, signaling that the application is ready to apply transformation to the swivel shape in progress. On both left and right area sections of the screen the user is able to perform vertical pen strokes to raise or decrease the extrusion level. On the remaining center area of the screen, vertical motion of the pen stroke affects the scale of the extrusion profile while horizontal motion rotates the profile in the z-axis (parallel to the ground plane of the scene). Tapping with the pen during the swivel extrusion terminates the method and finalizes the shape, becoming ready for manipulation.

4.6 Touch-based Interaction

As in the pen-based approach, it is also important to measure how users perform when using an approach solely based on touch interaction. Therefore, a touch-based approach was devised aided by the same main menus used in the pen-based approach (Section 4.5), to isolate the free-hand drawing, the camera and object manipulation, and the Swivel Extrusion functionalities.

4.6.1 Touch-based Modelling

While on the drawing main menu of the multi-touch interaction, the free-hand sketch modelling can be done by using one or more fingers simultaneously. Naturally, drawing using multiple touch points will generate multiple individual strokes. Although these strokes are initially independent from each other, it is possible to recognize patterns from the combination of the multiple ocurring strokes. For instance, if

the user chooses to draw two sides of a square with one finger and the other two sides with a second finger, the recognizers will still be able to acknowledge the both strokes as an intent to create a single square, the same way it would if the user chose to draw the same square with a single continuous stroke. The same principle applies to other recognizable shapes or when closing open sections.

4.6.2 Touch-based Camera and Object Manipulation

Regarding the manipulation main menu, it allows the user to freely navigate the scene and manipulate objects while using specific touch gestures. All object manipulation gestures are described in Figure 4.8. Before an object can be manipulated, it has to be selected by performing a simple touch tap gesture over it. A similar touch tap gesture over an already selected 3D object deselects it.

To apply a translation transformation to a selected object, a single-touch drag gesture (starting from the object) allows it to be moved in space and, when the touch is released, the object is fixed in its current location. For the rotation and scaling of a given selected object, once two touch control points are initialized in a selected object, changes in both distance and angle between the two control points determine transformation in scaling and rotation. The object is enlarged if the distance between the two control points increases and is minimized if the same distance decreases. In terms of rotation, the object is rotated in the closest parallel axis to the camera orientation direction.

As described in Section 4.4, both the translation and the rotation transformations are dependent of the camera orientation. Also, to maintain both scale and rotation proportions in relation to the two touch control points, these are projected onto the closest perpendicular canonical plane, as differences in distances and angles for screen points are not appropriate to be used as source for the rotation and scaling of three-dimensional points. It is important to note that it is allowed to have multiple selected objects at the same time, although only a single object can be manipulated at a time.

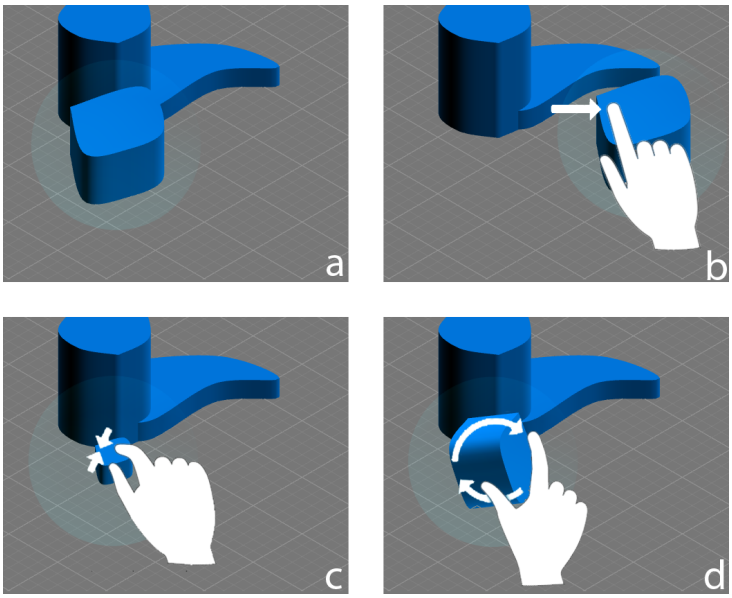


Figure 4.8: Object manipulation touch gestures for both the Touch-based and Combined Pen and Touch approaches. a) initial state of the object; b) translation gesture; c) scaling gesture; d) rotation gesture.

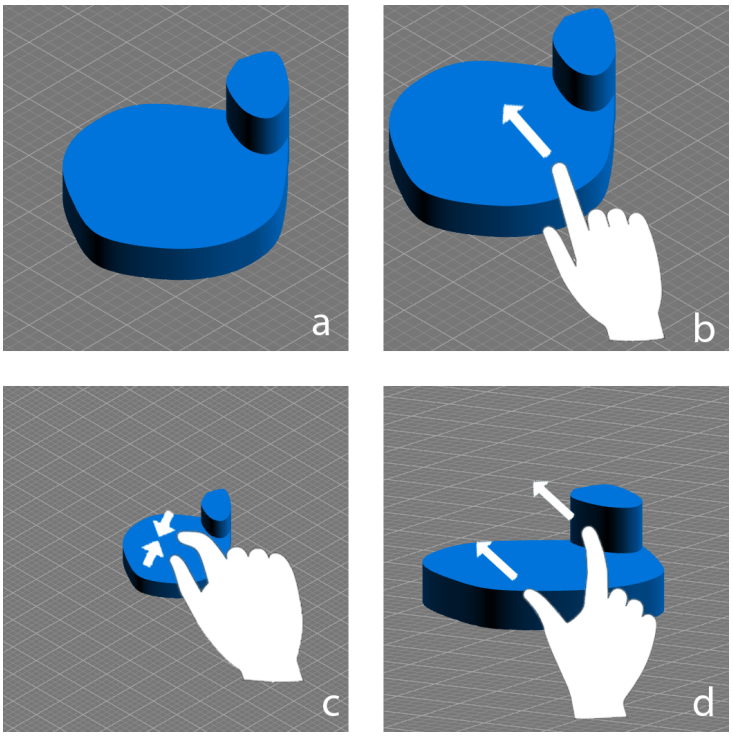


Figure 4.9: Camera manipulation touch gestures for both the Touch-based and Combined Pen and Touch approaches. a) initial state of the camera; b) panning gesture; c) zooming gesture; d) orbiting gesture.

Once the manipulation gestures are applied to the scene or non-selected objects, the camera is manipulated. All camera manipulation gestures are described in Figure 4.9. For camera panning, a single-touch drag gesture pans the camera over the XY plane (ground), maintaining it's distance from the scene ground plane.

Using a touch gesture it is possible to manipulate both camera zooming and orbiting. For both transformations a calculated middle point is used to project a point onto the XY plane, providing a central point for the camera operation manipulations. While zooming, variations in the distance of the dual touch points determines whether the camera is moved forwards or backwards, in relation to the projected center point. On the other hand, for the orbiting, this projected central point acts as the center of the orbit, meaning that by the end of the orbit operation the camera will still be facing it, and at the same distance from it.

In early versions of the touch-based camera manipulation there where attempts to also control the panning via the variations in position of the middle point calculated from the dual touch interaction while zooming or orbiting. Although this approach allowed to manipulate all three camera controls simultaneously using only two touch points, it proved to be too complex to achieve consistent results, and compromising the overall experience.

4.6.3 Touch-based Swivel Extrusion Tool

Having a closed stroke and changing to the Swivel Extrusion main menu initializes the method. On this state, the control of the extrusion level can be done with a single touch along either the left or right side

of the application window. On the remaining central area, the user is able to use dual touch to apply the same modifications to the extrusion profile. Changes in the distance and finger angle of the dual touch interaction alter the scale and rotation (on the z-axis) of the profile, respectively. Tapping an ongoing extrusion finalizes its progress.

4.7 Combined Pen and Touch Interaction

Having described both the pen-based and touch-based approaches, the combined pen and touch interaction aims to take advantage of the most positive features of both methods. In this approach, pen interaction is delegated to the creation of 2D strokes and extrusion level control, while manipulation transformations and expectation list interaction are controlled by multi-touch input. Since pen and touch interaction can be clearly distinguished, more gesture options arise from the combination of the two. Therefore, in the combined approach, no menus are required and all actions are performed directly in the scene view.

4.7.1 Combined Pen and Touch Modelling

As described before, stroke drawing in the combined interaction is done by using the pen device directly in the scene. As in the previous two approaches, expectation results are shown and allow the user to pick a desired outcome with a touch tap gesture.

4.7.2 Combined Pen and Touch Camera and Object Manipulation

Regarding camera and object manipulation, these are directly inherited from the touch-based interaction approach (specifically in the manipulation main menu), resorting to simple tap for selecting/deselecting shapes, single-touch drag and drop for both camera panning and object translation, and multi-touch for camera zooming and orbiting, as well as object scaling and rotation. Likewise, camera and object manipulation gestures are described in Figure 4.9 and 4.8, respectively.

4.7.3 Combined Pen and Touch Swivel Extrusion Tool

Without a menu option for the Swivel Extrusion, the method is initiated once a closed stroke exists in the scene and an upwards pen stroke is performed on either left or right side of the application window. On entering the Swivel Extrusion method, vertical pen strokes adjust the extrusion level (just as in the pen-based approach) while multi-touch interaction controls transformations to the extrusion profile (just as in the touch-based approach). Figure 4.10 illustrates the extrusion process using this approach.

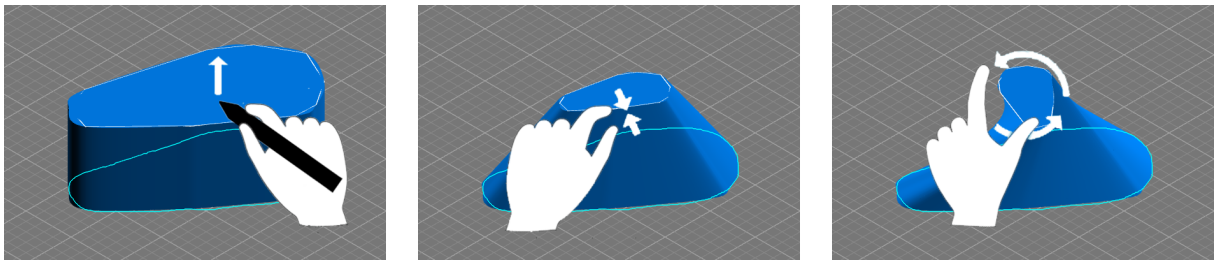


Figure 4.10: Using the Swivel Extrusion tool in the Combined Pen and Touch interaction. Level adjustments are done using the pen device and profile manipulation is achieved via multi-touch gestures.

4.8 Expectation Lists

As proven in works such as Chateau [19], ShapeShop [5] or GIDeS [4], during a user's modelling process it is important keep the user productively engaged at all times by providing the right tools as they are needed. To do so, expectation lists play a key role in the interactive performance of our solution. As described in the architecture of our solution (section 4.1), recognizers are components that actively search for possible transformations given the current state of the application. These new possible modifications are then used to populate the expectation list that will be presented to the user. In the presented solution, the expectation list assumes the form described in Figure 4.11, a simple list of animated thumbnails located at the bottom left of the application window. Each element of the list contains an animated three dimensional preview of the respective outcome. The list is also non-intrusive, meaning that the user can simply ignore suggestions as he/she pleases, without any type of repercussions. It is also important to note that the expectation lists should never overwhelm the user with large sets of outcomes, as most of these will probably not be useful in the current modelling context.

Results of retrieval queries also result in possible outcomes for the expectation list. As the user composes an arbitrarily complex model using other simpler shapes, these can be grouped as a single object to be submitted for retrieval. Once the retrieval server replies with the respective results, the expectation is populated with a small set of the ones that most closely resemble the queried example. In the following section we will describe the retrieval process in detail.

4.9 Retrieval

As it is difficult to produce exact shapes using sketch-based 3D modelling, the integration with a 3D object retrieval system allows a user to specify an object that closely relates to a desired shape and query a database using his/her sketch as an example. With this practice, it is possible to acquire arbitrarily complex shapes by modelling an approximate sketch. Not only is this process faster, but it also simpler. On the downside, it is limited by the set of shapes stored in the retrieval system, as well as by the descriptors used to compute the model indexes.

In early phases we worked with an almost finished server implementation (in Java) of the EnContRA retrieval system. This server supported functions for storage of indexes and evaluation of a given example model. Furthermore, we developed the EnContRA client in Dart to be used in our solution. During

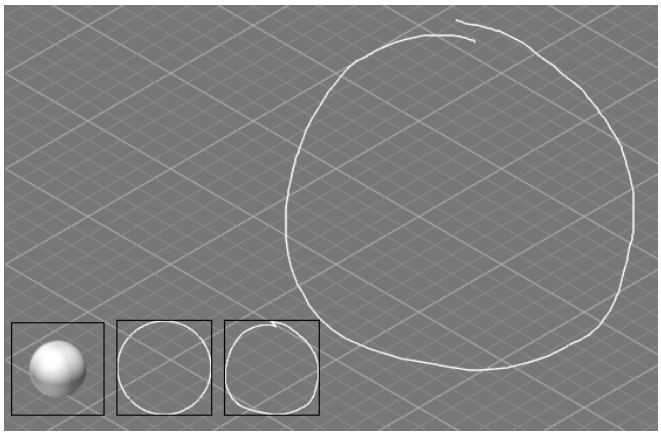


Figure 4.11: Example suggestion list containing suggestions for a circle shaped scribble.

the development process of our approach we also extended the functionality of the server by adding a download function to allow the streaming of a given model from the server to the client.

The EnContRA Server is populated by indexing 3D model files using specific shape descriptors, storing then the index and the model file itself. The model file formats supported by EnContRA are Wavefront (.obj) and Standard Tessellation Language (.stl). Currently, the retrieval system can compute indexes using multiple known shape descriptors, such as D1, D2, D3, D4 and A3. In preliminary tests, due to being the descriptor with best results, we decided to use D1 as the main shape descriptor. For our approach, we only considered indexing models offline, although the current implementation of both the EnContRA client and server support the storage of indexes in runtime.

Once the server has at least one computed index, it is ready to receive requests for similarity evaluation. For every similarity request, the server replies with an ordered list with a result for each stored index in the server. Each result entry contains a similarity value (number between 0 and 1), the identifier and the name of the evaluated model present in the server. As this information is received in the client, the most similar results are chosen to be downloaded from the server so they can be loaded in the expectation scene and possibly used in the modelling process.

Regarding the models used to populate the server, we decided to use sets of publicly available sample Wavefront (.obj) models from the Department of Computer Science of Northwestern University[1] and from the Department of Scientific Computing of Florida State University[2]. These samples include cubes, cones, cylinders, tetrahedrons, octahedron, among others.

Although the retrieval component of the application is considered to be an important feature of the overall solution, due to it not being precise in regard to the measurement of similarity values between 3D models, even using the best performing algorithm (D1), we decided to remove the automatic submission for retrieval from the developed solution. Instead, the user is able to select objects to perform an on-demand query whenever he/she deems as appropriate by clicking a button outside of the scene view, by the modelling, manipulation and Swivel Extrusion Tool options. Even though this approach might not

[1]Sample models from the Department of Computer Science of Northwestern University, `http://www.cs.northwestern.edu/~ago820/cs351/Models/OBJsimpleTestSuite/`

[2]Sample models from the Department of Scientific Computing of Florida State University, `http://people.sc.fsu.edu/~jburkardt/data/obj/obj.html`

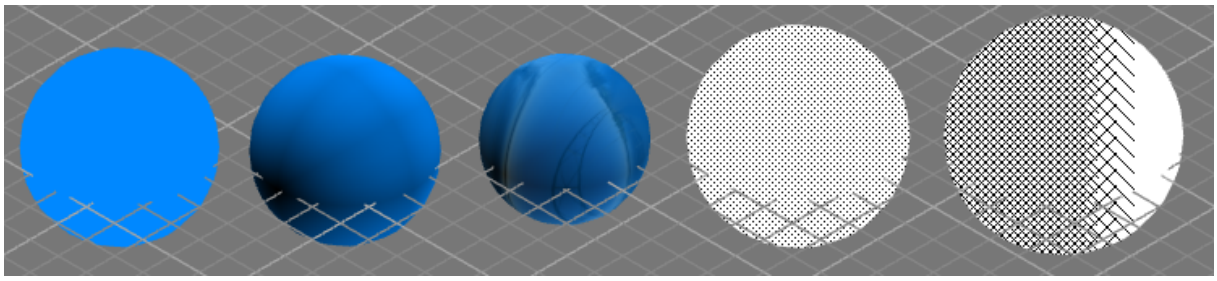


Figure 4.12: Currently available materials. From left to right: Basic, Lambert, Reflective, Dotted and Hatching.

be considered natural, it is preferable instead of having non-similar objects being suggested every time the user selects an object, possibly compromising the overall modelling experience. For future work, as the effectiveness of the shape descriptor algorithms is improved in the EnContRA Retrieval system, the automatic submission for retrieval in the presented solution can be easily reactivated, since the support for this retrieval system is fully operational.

4.10 Complementary Functionalities

During the development of the main components to our solution we also worked on complementary functionalities that although were not considered to be relevant to the natural user interface itself, were useful to the overall functionality and usability of the prototype. Since neither Dart nor threeDart had working implementations of model exporters, we implemented simple versions for both Wavefront (.obj) and Standard Tessellation Language (.stl) file formats. These exporters were essential later to send sets of models in a single file to the EnContRA server, as well as to be able to export the full modelling scene for 3D printing or storage purposes.

A secondary feature present in our solution is the customization of the modelling tools. Users can change the color, transparency of both the stroke and the generated 3D models. Additionally, it is also possible to change the material of the generated 3D models. The currently available materials are shown in Figure 4.12.

4.11 Summary

The main focus of this work is to devise a natural technique that takes advantage of the combination of both touch and pen interaction in a sketch-based 3D modelling context, using interactive surfaces. In this chapter, we presented the main structure and components of our solution, as well as details regarding the specific modelling and manipulation functionalities. Additionally, we described three different interaction approaches: Pen-based, Touch-based and Combined Pen and Touch interaction.

In the Pen-based and Touch-based approaches we utilize standard interaction methods in order to establish reference points to compare with the combined interaction scheme. The Pen-based approach uses a single pen device to perform all actions, requiring a relatively complex menu system to operate.

Regarding the Touch-based interaction, modelling operations are performed using multi-touch, allowing the use of gestures, and therefore needing only a simple menu system consistent of three modes. Finally, the Combined Pen and Touch interaction uses the advantages of both Pen-based and Touch-based interactions to create an approach free of menus, allowing the use of the pen device and multi-touch gestures simultaneously.

In addition to the interaction approaches, we also present the Swivel Extrusion method, a special method for the creation of complex extrusions that allows the extrusion profile to be transformed at any given step of the extrusion process, supported in all three interaction methods.

Aside from the interaction methodology, we also described relevant components of the modelling application such as the expectation lists and the retrieval system. We consider these components to play an important role in a natural modelling tool as they interactively aid the user by presenting relevant suggestions. Although the retrieval system is unable to present useful results at the moment (due to the computation of improper similarity values by the shape descriptor algorithms), the presented solution has full support of the currently available EnContRA system, meaning that if this system is to be improved in the future, the solution is still able to support it seamlessly.

In the next chapter we describe the user evaluation stage, from planning to result analysis, in which we compare the three interaction methods.

Chapter 5

Evaluation

In the previous chapter we presented the developed prototype as well as its different interaction approaches. As the involvement of real users is a crucial step in the development of a user-centered solution, we conducted a series of tests with a group of 20 users in order to evaluate our solution. By doing so, we expected to gather information about the users' expectations and difficulties while interacting with the modelling tool. We had particular interest in understanding whether users could use object and camera manipulation tools both correctly and naturally. Also, the evaluation of and comparison between the main different methods for interaction is a major subject of this study.

Likewise, we evaluated the solution in the form of three different types of interaction: touch-based, pen-based and combined pen and touch interaction.

5.1 Methodology

Each user performed the same task using all three approaches, in alternate order. For each user testing the prototype the expected duration for the evaluation session was between 40 and 60 minutes and was divided as follows:

- Introduction: Users were instructed as to how both the interactive tabletop and the pen device were used.
- Profiling Survey: Collection of the basic profile information for each user (gender, level of education and experience with other modelling tools and multi-touch devices). The profiling survey used is annexed in Appendix I.
- Video presentation of the solution: Before any interaction with the application, each user was presented a short video explaining how to perform the basic functionalities of the prototype (for the respective interaction approach). Each presented video has a duration between 1 and 2 minutes.
- Training session: After the video presentation, users were invited to freely use the application in a training session without any type of constraint. This training session had the maximum duration of 2 minutes.

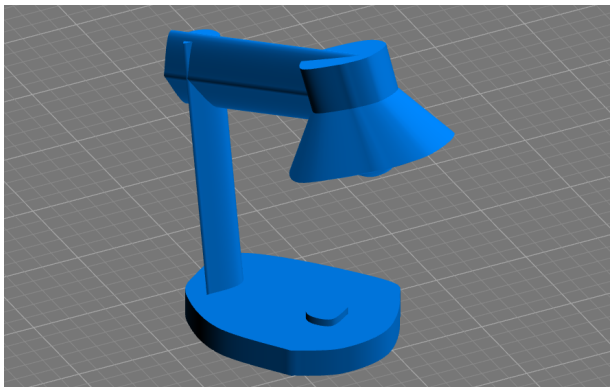


Figure 5.1: Example desk lamp model.

- Task Execution: Given a specific interaction approach, each user was given the task to model a desk lamp. The suggested instructions for the task are described in Appendix E.
- Task Survey: Once a user finished the task using the assigned interaction method, the elapsed time was registered and the user was asked to fill a survey inquiring about the levels of entertainment and difficulty felt during the completion of the task. This survey is described in Appendix G.
- Tools Survey: After finishing the task using the three different interaction approaches and filling the respective task surveys, users were then asked to fill another short survey to measure the level of utility of the different tools present in the application, through all interaction types. This final survey can be found in Appendix H.

It is important to note that the annexed documents relative to the evaluation process are written in portuguese as the tests were performed at Instituto Superior Técnico - Taguspark (Portugal), with portuguese users.

5.2 Modelling task

The proposed task consisted of the modelling of a simple desk lamp (Figure 5.1 represents an example), comprised of five separate solids: base, column, arm, shade and bulb. Each user received an instructions sheet (Annex E) with five advised ordered steps to build the desk lamp, although they were not forced to follow them, and encouraged to try different orders if considered appropriate. Likewise, users were also not required to perform retrieval searches, although were encouraged to try.

In order to correctly model the desk lamp, users had to model simple cylinders for most of the components (column, base and arm), which could be easily done with simple extrusions. As of the bulb, it could be simply modelled using a basic primitive sphere. The most complex element of the lamp is the shade, modelled by an extrusion with at least three steps, requiring the user to also manipulate the profile of the extrusion. Both extrusions and primitive shape modelling require the user to use the expectation list for stroke closing and primitive shape transformation (from a free-hand 2D stroke).

Regarding camera and object manipulation, as it is needed to compose some objects on top of

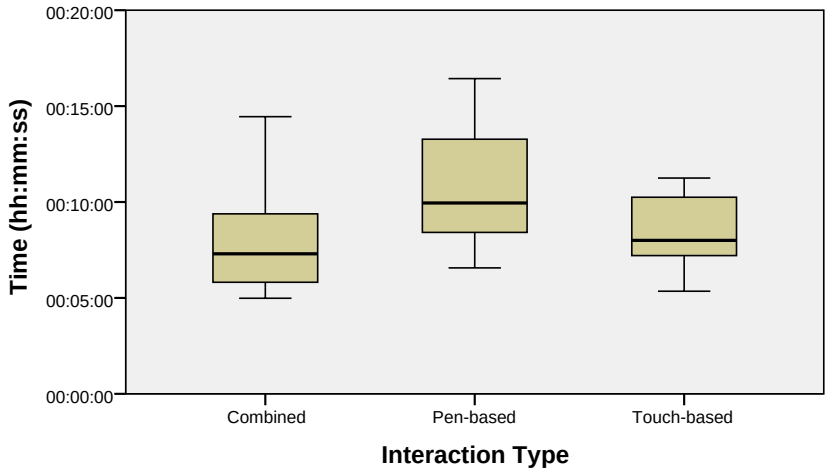


Figure 5.2: Overall view of the time to complete the task using the three interaction types. The graphic presents the median, first and third interquartile ranges (boxes) and 95% confidence interval (whiskers).

others, it is required that the user manipulates the camera in order to assure the correct positioning of the parts. Parts such as the arm and the shade also require the user to rotate the objects. Also, since all parts result of free-hand sketch-based modelling, it is also required to perform scale transformation in different objects in order to maintain the correct proportion of the components.

5.3 Participants and Setup

The evaluation session was attended by 20 users, mostly male, with ages ranging from 21 to 57 years old, although the large majority was below 30 years old. Regarding the education level, most of the users had at least a bachelor's degree. The large majority of the users had vast experience with multi-touch devices while the remainder did not. The results of the profiling survey are described in Annex I.

Regarding the setup for the tests, two computers were used. The developed solution was executed in the main computer (connected to the interactive tabletop), in which the users trained and performed the tests. In a secondary laptop computer, complementary activities were carried out such as the presentation of the instructional videos and filling out the different surveys. Also in this secondary machine, both the EnContRA retrieval server and the pen sender application were hosted. A pen receiver application was used in the primary computer to allow the reception of pen input from the secondary computer (to which the pen device was connected) in the form of keyboard input. Both computers communicated via HTTP requests in a local network.

5.4 Objective Analysis

For objective analysis we compare the three interaction types separately. Having measured the time taken for each user to perform the task using each interaction type, we averaged those values to obtain a mean value for each interaction.

	Pen-based	Touch-based	Combined
Mean	0:10:48	0:08:50	0:07:49
N	20	19	19
Std. Deviation	0:02:54	0:02:31	0:02:43

Figure 5.3: Average task completion times for Pen-based, Touch-based and Combined Pen and Touch interactions.

Applying the Shapiro-Wilk test to the evaluated time samples revealed that only the Pen-based interaction samples are considered normally distributed (significance=.062) while time samples for both the Touch-based and Combined interaction types are not (with significance=.030 and significance=.019, respectively). In addition, the non-parametric Friedman test suggests that there was a statistically significant difference in the time taken by the users to complete the task using the different interaction approaches (χ^2=17.360 p=.001).

Post hoc analysis with Wilcoxon Signed-Rank tests was conducted with a Bonferroni correction applied, resulting in a significance level set at p$<$.017. These tests suggested that statistically significant differences existed only when comparing the Pen-based approach to the Touch-based (Z=-2.495 p=.013) and Combined (Z=-3.461 p=.001) approaches, and not when comparing the Touch-based approach to the Combined approach (Z=-2.200 p=.028).

In Figure 5.3 it is visible that the users performed faster while using the Combined Pen and Touch interaction (mean: 7 minutes and 49 seconds), then using the Touch-based approach (mean: 8 minutes and 50 seconds), and finally slower using the Pen-based interaction (mean: 10 minutes and 48 seconds). Figure 5.2 depicts an overall view of times using the three interaction types.

5.5 Subjective Analysis

In the task-specific surveys, we asked users to classify the perceived difficulty for the task of modelling the desk lamp as well as for various specific camera and object operations, using a 4 point scale (1-Very Difficult, 2-Difficult, 3-Easy, 4-Very Easy). Likewise, we also measured the perceived fun factor (1-Not Fun, 2-Slightly Fun, 3-Fun, 4-Very Fun) and method fluidity (1-Not Fluid, 2-Slightly Fluid, 3-Fluid, 4-Very Fluid). The Wilcoxon Signed Rank test was used once again to determine whether the differences were statistically significant.

Considering the Table 5.1 we can assume that in general, users considered the solution (including all three interactions) to be at least fun and fluid. As for the overall difficulty of the modelling task, participants considered all three interaction types to be relatively similar, averaging a rank between easy and difficult.

In Table 5.2, the mean values for the difficulties of the specific modelling and manipulation operations are presented. Regarding the mean difficulty of drawing 2D shapes, object selection and translation, the differences are negligible between interaction types. Overall, users considered the both the drawing of 2D shapes and the object selection to be trivial operations, while object translation was considered an easy task.

	Fun Factor	Overall Difficulty	Method Fluidity
Pen-based	2.90(1)	2.65(1)	2.65(1)
Touch-based	3.25(1)	2.65(1)	3.00(0)
Combined	3.45(1)	2.75(1)	3.50(1)

Table 5.1: User preference for each interaction technique in regard to fun, overall difficulty and method fluidity (Mean, Inter-quartile range). All ranges are measured between 1 (worse) and 4 (better).

	Draw Difficulty	Panning Difficulty	Orbiting Difficulty	Zooming Difficulty	Selection Difficulty	Translation Difficulty	Rotation Difficulty	Scaling Difficulty
Pen-based	3.75(1)	3.45(1)	3.10(1)	3.40(1)	3.60(1)	3.10(1)	2.70(1)	3.30(1)
Touch-based	3.70(1)	3.15(1)	2.60(1)	2.80(1)	3.50(1)	3.00(2)	2.35(1)	2.80(1)
Combined	3.65(1)	3.15(1)	2.45(1)	2.65(1)	3.60(1)	3.10(1)	2.30(2)	2.55(1)

Table 5.2: User preference for each interaction technique in regard to difficulty of specific tasks (Mean, Inter-quartile range). All ranges are measured between 1 (worse) and 4 (better).

Pen-based	3.80(0)
Touch-based	2.10(1)
Combined	3.70(1)

Table 5.3: User preference for each tool in regard to overall utility (Mean, Inter-quartile range). All ranges are measured between 1 (worse) and 4 (better).

For orbiting and zooming, users strongly agreed that using the Pen-based interaction camera manipulation operations are easier. Orbiting using the pen is considerably simple when compared to orbiting using just touch (Z=-2.887 p=.004) or using the combined (Z=-2.982 p=.003) approach. Similarly, zooming is also considered to be simpler using the pen when compared to both Touch-based (Z=-3.207 p=.001) and Combined (Z=-3.441 p=.001) approaches.

When rotating an object, users considered all approaches to be slightly difficult. On the other hand, scaling was considered to be an easier operation overall. Scaling obtained significantly better results in the Pen-based approach when compared with the Touch-based one (Z=-2.324 p=.020).

As described in the previous chapter, Pen-based interaction divides manipulation operations in different sub-menus. By doing so, only one manipulation operation (at most) is occurring at any given time. As of the other two interaction methods, multiple manipulation operations can be active at a given time, allowing the user to simultaneously rotate and scale an object, or zoom and orbit the scene. Likewise, users naturally considered the manipulation with the pen to be easier, as it is atomic and simpler, although the approach itself requires the user to change the manipulation modes several times for a simple transformation.

Regarding the different tools in the solution (described in Table 5.3), utility was ranked with a 4 point scale (1-Not Useful, 2-Slightly Useful, 3-Useful, 4-Very Useful). Users strongly agreed that both the

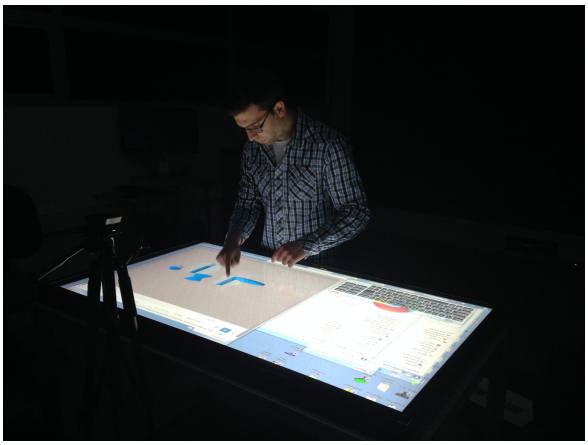


Figure 5.4: User manipulating a shape using multi-touch.

Figure 5.5: User drawing with the pen device using the combined pen and touch approach.

Swivel Extrusion tool and the expectation list are very useful in a 3D modelling context. Specifically for the Swivel Extrusion tool, it allowed the modelling of most of the parts required for the desk lamp task with minimal interaction and yet accurate precision. As for the expectation list, not only was it considered crucial to transform 2D strokes into primitive 3D shapes, as it also provided useful suggestions for the closing of 2D paths and presentation of retrieval results.

Regarding the Retrieval tool, users considered it to be slightly useful, given the fact that the returned results were most of the times incorrect due to the miscalculation of similarity values in the current implementation of the EnContRA server. However, most of the users agreed that having a functional 3D object retrieval system as part of a 3D modelling application could be substantially useful.

5.6 Observations

As reported by the users, one aspect that led to the majority of the errors while modelling was the capturing of unwanted input. Regardless of the type of interaction, users naturally induced unwanted input on the interactive surface, either by tapping unwillingly or by touching the interface with more fingers than desired (specially when using the pen device). Once users understood this, they rapidly adapted their hand posture and carried out the tests carefully in an attempt to not repeat the mistake. Figures 5.4 and 5.5 depict users performing the task during the evaluation process.

As the prototype was run on a web browser, before any interaction users were advised to adjust the window size of the browser as they felt was more comfortable. Users that initially chose to maximize the browser window also ended up changing the view back to a smaller window view, as the interaction using the menus on the top left of the interface became more difficult, mostly due to the the size of the tabletop.

Another relevant detail related to the overall modelling experience noted by the users is that in the Swivel Extrusion Tool, tapping to end the extrusion process can be a problem as unwanted taps can occur. Some users suggested the introduction of a confirmation menu to ensure that the user is about to finish the extrusion, while others suggested the use of a more complex gesture instead of a simple tap.

In regards to object selection, some users revealed some difficulty while picking an object, as this object has to occupy a considerable amount of screen space to be pickable. To counter this, a user suggested the creation of a box selection tool, capable of selecting surely one or multiple objects inside a user defined area at the same time.

While using a Combined Pen and Touch approach on the interactive tabletop, some users described the object and camera manipulation as a difficult task when performed using multi-touch gestures with a single hand. Therefore, these users felt more comfortable putting the pen device aside while making complex object and camera manipulations and using both hands (as in the Touch-based interaction, although some hadn't yet performed the task using this interaction type), and then picking the pen up again when needed. Regarding the Touch-based and Pen-based interactions, the menus were considered to be less intuitive, as they often required the users to learn what button was associated to the desired transformations. Also, the method is purely based on gestures and interaction type, with the lack of visual references some users struggled initially, having trouble reminding how to perform a certain manipulation or action.

In general, users who chose to design all the pieces first and then assemble them later were able to perform the task faster. Performing similar tasks sequentially required fewer changes between menus and between two-hand and dominant hand interaction.

Finally, users that were not familiar with touch-based applications suffered from major difficulties while trying to perform the task using the Touch-based and Combined interactions. In this special situation was the 57 year-old user who only had little experience with computers in general, and was only able to finish the task using the Pen-based approach (the one that most closely relates to the mouse interaction), having given up while performing the task using the other two interactions, mainly because of difficulties while trying to use the multi-touch to manipulate the camera and the objects.

5.7 Summary

In this chapter we have presented the evaluation of our solution. Each of the 20 participants has performed once with each of the three interaction approaches. Through surveys we have also profiled the users and registered their beliefs in regard to different aspects of the execution of the tasks.

In general terms, participants were able to perform the task faster using the Combined Pen and Touch approach, and slower using the Pen-based approach. In a similar relation, users also considered the approaches to be increasingly more fun and fluid from Pen-based, to Touch-based, and finally to Combined Pen and Touch approach. On the other hand, the overall difficulty differences between approaches were considered minimum and averaging a medium difficulty level between hard and easy.

In particular for the Pen-based approach, users strongly agreed that most manipulation operations were considered to be easier when compared with the other interactions, as when using this approach the transformations are atomic, and therefore more controlled. However, having to change multiple times between manipulation modes using the menus is time consuming and less fluid when compared to the other approaches.

As for utility, participants considered the both the Swivel Extrusion tool and the expectation lists to be considerably useful.

Chapter 6

Conclusions

Interfaces based on the WIMP paradigm (windows, icons, menus and pointer) have a long history in the field of user-centered design, being used for most applications' user interfaces, including 3D modelling tools. However, as tactile surfaces are increasingly more common in every-day devices, it becomes necessary to devise new and more natural interaction techniques that can take advantage of those surfaces. Additionally, 3D fabrication has become a popular subject in the recent years, propelled by the high availability of 3D resources online. Therefore, it is also necessary to develop new modelling techniques that empower the average user to be able to create 3D physical models in a simple and natural manner.

Over the recent years, several works have been presented in order to solve common interaction problems when modelling using interactive surfaces. Although most presented satisfying results, they could not be generalized to all modelling scenarios, particularly for 3D fabrication.

In order to achieve an interactive modelling experience, some works proposed the use of non-intrusive suggestion lists. By doing so, given a current state of the modelling application, it is possible to suggest appropriate transformations by displaying different outcomes in a organized and explicit list, allowing the user to quickly modify the model. These suggestion lists are particularly helpful when used to suggest other existing objects that are considered to be similar to the result of the modelling in progress. With the development of new and more efficient methods for 3D object retrieval, it is possible to integrate this type of system in a 3D modelling solution, maintaining interactive performance.

In this study, we proposed to identify the advantages and disadvantages of different natural interaction types in a 3D modelling context, that explore the use of interactive surfaces, multi-touch and/or calligraphic interaction. Regardless of the interaction type, it was intended to develop a solution that can be used by both experienced and novice users, requiring only a brief learning process.

To do so, three main interaction types were devised: Pen-based, Touch-based and Combined Pen and Touch interaction. For each interaction type, different methods are used in order to create and transform shapes, as well as manipulate both the objects and the view. These interactions were also evaluated with a set of users, in order to assess the differences between them, measuring the performance and the usability of the different sub-operations of the solution. Additionally, users evaluated generic tool

components such as the Swivel Extrusion tool, the expectation list and the 3D object retrieval system.

Analysing the results of the evaluation with users revealed that although users were not as familiar with the Combined and Touch-based approaches, they were able to perform the task faster using the Combined, then the Touch-based and finally with the Pen-based approach. This relation is also consistent with the menu complexity level between approaches, as the Pen-based uses two levels of menus, while the Touch-based uses only one level, and finally the Combined approach uses no menus. These results are coherent with others present in related works, regarding the levels of naturalness and menu complexity between interaction types.

Considering all the above, we were able to develop a solution that supports three different interaction types, enhanced by non-intrusive interactive tools, such as the expectation lists, 3D object retrieval functionality and the Swivel Extrusion. The evaluation phase revealed that users, regardless of having experience with other modelling tools, are able to autonomously create 3D models (that are production ready and can be later exported, stored or printed) with satisfying results in regard to the perceived difficulty level and time taken to complete the task.

6.1 Future Work

After measuring the results of the user testing, we consider that some solution details are worthy of improvement. Some of them are:

3D Object Retrieval: Although it was not considered to be the main focus of the developed application, the 3D object retrieval component was an important part of the solution. To improve it, it is necessary to inspect and correct the current implementation of the EnContRA Server, specifically the computation of shape descriptors and evaluation of similarity values.

Multi-touch Gestures: As some users stated, using a simple tap gesture to end an extrusion can lead to it being ended unwillingly early (due to unwanted interaction). Instead, alternatives must be found to substitute these overly simple gestures.

Save States: In the current solution we have implemented a very basic save state hierarchy as a complementary module for mostly debugging purposes, that allows both ”redo” and ”undo” operations. In a 3D modelling application, these operations can be crucial to maintain a simple and encouraging experience. In the future, it is appropriate to review this component and make it available to all interaction types via a menu or by multi-touch gestures.

New tools for modelling: After obtaining good results in the user evaluation phase, it became clear that users feel engaged while using familiar and simple tools to create complex shapes, such as the Swivel Extrusion tool. As a follow-up for the Swivel Extrusion, we considered the creation of a similar simple tool to model a surface revolution, taking advantage of the multi-touch interaction.

Bibliography

[1] The princeton shape benchmark. In *Proceedings of the Shape Modeling International 2004*, SMI '04, pages 167–178, Washington, DC, USA, 2004. IEEE Computer Society.

[2] Robert C. Zeleznik, Kenneth P. Herndon, and John F. Hughes. Sketch: An interface for sketching 3d scenes. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 163–170, New York, NY, USA, 1996. ACM.

[3] Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. Teddy: A sketching interface for 3d freeform design. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, pages 409–416, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[4] Manuel J. Fonseca, Alfredo Ferreira, and Joaquim A. Jorge. Towards 3d modeling using sketches and retrieval. In *Proceedings of the First Eurographics Conference on Sketch-Based Interfaces and Modeling*, SBM'04, pages 127–136, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association.

[5] Ryan Schmidt, Brian Wyvill, Mário Sousa, and Joaquim Jorge. ShapeShop: Sketch-Based Solid Modelling with BlobTrees. *EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling*, 2005.

[6] Pedro Lopes, Daniel Mendes, Bruno Araújo, and Joaquim A. Jorge. Combining bimanual manipulation and pen-based input for 3d modelling. In *Proceedings of the Eighth Eurographics Symposium on Sketch-Based Interfaces and Modeling*, SBIM '11, pages 15–22, New York, NY, USA, 2011. ACM.

[7] Yves Guiard. Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model. In *Journal of Motor Behavior 19*, pages 486–517, 1987.

[8] Bruno R. De Araùjo, Géry Casiez, and Joaquim A. Jorge. Mockup builder: Direct 3d modeling on and above the surface in a continuous interaction space. In *Proceedings of Graphics Interface 2012*, GI '12, pages 173–180, Toronto, Ont., Canada, Canada, 2012. Canadian Information Processing Society.

[9] Tiago Santos, Alfredo Ferreira, Filipe Dias, and Manuel J. Fonseca. Using sketches and retrieval to create lego models. In *Proceedings of the Fifth Eurographics Conference on Sketch-Based*

Interfaces and Modeling, SBM'08, pages 89–96, Aire-la-Ville, Switzerland, Switzerland, 2008. Eurographics Association.

[10] Daniel Mendes, Pedro Lopes, and Alfredo Ferreira. Hands-on interactive tabletop lego application. In *Proceedings of the 8th International Conference on Advances in Computer Entertainment Technology*, ACE '11, pages 19:1–19:8, New York, NY, USA, 2011. ACM.

[11] R.J.K. Jacob, A. Girourard, L.M. Hirshfield, M.S. Horn, O. Shaer, E.T. Solovey, and J. Zigelbaum. Reality-based interaction: unifying the new generation of interaction styles. In *CHI'07 Extended abstracts on Human factors in computing systems*, pages 2465–2470. ACM, 2007.

[12] Carsten Schwesig. What makes an interface feel organic? *Commun. ACM*, 51(6):67–69, June 2008.

[13] J. M. Saavedra, B. Bustos, T. Schreck, S. Yoon, and M. Scherer. Sketch-based 3d model retrieval using keyshapes for global and local representation. In *Proceedings of the 5th Eurographics Conference on 3D Object Retrieval*, EG 3DOR'12, pages 47–50, Aire-la-Ville, Switzerland, Switzerland, 2012. Eurographics Association.

[14] Tianjia Shao, Weiwei Xu, KangKang Yin, Jingdong Wang, Kun Zhou, and Baining Guo. Discriminative sketch-based 3d model retrieval via robust shape matching. *Comput. Graph. Forum*, 30(7):2011–2020, 2011.

[15] Kun Xu, Kang Chen, Hongbo Fu, Wei-Lun Sun, and Shi-Min Hu. Sketch2scene: Sketch-based co-retrieval and co-placement of 3d models. *ACM Trans. Graph.*, 32(4):123:1–123:15, July 2013.

[16] Christian Holz and Andrew Wilson. Data miming: Inferring spatial object descriptions from human gesture. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 811–820, New York, NY, USA, 2011. ACM.

[17] M. Nakazato and T.S. Huang. 3d mars: immersive virtual reality for content-based image retrieval. In *Multimedia and Expo, 2001. ICME 2001. IEEE International Conference on*, pages 44–47, Aug 2001.

[18] Pedro Pascoal, Alfredo Ferreira, and Joaquim Jorge. Towards an immersive interface for 3d object retrieval. In *Proceedings of the 5th Eurographics Conference on 3D Object Retrieval*, EG 3DOR'12, pages 51–54, Aire-la-Ville, Switzerland, Switzerland, 2012. Eurographics Association.

[19] Takeo Igarashi and John F. Hughes. A suggestive interface for 3d drawing. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology*, UIST '01, pages 173–181, New York, NY, USA, 2001. ACM.

[20] Manuel J. Fonseca, César Pimentel, and Joaquim A. Jorge. Cali: An online scribble recognizer for calligraphic interfaces. In *Sketch Understanding, Papers from the 2002 AAAI Spring Symposium*, pages 51–58, 2002.

[21] Johan W. Tangelder and Remco C. Veltkamp. A survey of content based 3d shape retrieval methods. *Multimedia Tools Appl.*, 39(3):441–471, September 2008.

[22] Benjamin Bustos, Daniel A. Keim, Dietmar Saupe, Tobias Schreck, and Dejan V. Vranić. Feature-based similarity search in 3d object databases. *ACM Comput. Surv.*, 37(4):345–387, December 2005.

[23] Cha Zhang and Tsuhan Chen. Efficient feature extraction for 2d/3d objects in mesh representation. In *Image Processing, 2001. Proceedings. 2001 International Conference on*, volume 3, pages 935–938 vol.3, 2001.

[24] Ryutarou Ohbuchi, Tomo Otagiri, Masatoshi Ibato, and Tsuyoshi Takei. Shape-similarity search of three-dimensional models using parameterized statistics. In *Proceedings of the 10th Pacific Conference on Computer Graphics and Applications*, PG '02, pages 265–, Washington, DC, USA, 2002. IEEE Computer Society.

[25] Mihael Ankerst, Gabi Kastenmüller, Hans-Peter Kriegel, and Thomas Seidl. 3d shape histograms for similarity search and classification in spatial databases. In *Proceedings of the 6th International Symposium on Advances in Spatial Databases*, SSD '99, pages 207–226, London, UK, UK, 1999. Springer-Verlag.

[26] Michael Kazhdan, Thomas Funkhouser, and Szymon Rusinkiewicz. Rotation invariant spherical harmonic representation of 3d shape descriptors. In *Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, SGP '03, pages 156–164, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[27] H. Sundar, D. Silver, N. Gagvani, and S. Dickinson. Skeleton based shape matching and retrieval. In *Proceedings of the Shape Modeling International 2003*, SMI '03, pages 130–, Washington, DC, USA, 2003. IEEE Computer Society.

Appendix A

CluTCh Project

The CluTCh project (Collaborative Three-D Construction) aims to enable the average user (trained or non-trained) to become a creator of new and creative 3D objects. CluTCh allows the collaborative construction of 3D models in a distributed environment, as well as its distribution and sharing in a social network context.

The availability of a 3D modelling platform will promote the democratization of the creation and three-dimensional manufacturing process, enhancing the use of rapid prototyping machines and processes, while transporting these 3D creations from the digital world to produce physical copies, encouraging collaborative customization of everyday goods.

The main result of the project will allow access to a modelling tool, either from mobile devices or through the standard browser, for viewing and customization of models available in community catalogs, as it already occurs with other types of media such as videos (YouTube) and photos (Flickr), in the context of a social network.

Within a modelling scenario, an important aspect is the retrieval of the 3D objects already cataloged. This search can be triggered when a model is being displayed or created, presenting a list of similar models. In the modelling scenario the retrieval of 3D objects can be especially productive for models composed of multiple parts, as the user can iteratively retrieve the different parts as he/she is composing the complex model.

Appendix B

3D Object Retrieval

A diagram of a generic multimedia information retrieval system is described in Figure B.1. The main problems in multimedia information retrieval revolve around finding the right methods for semantic information extraction and matching, as multimedia objects tend to be rich in different types of information, making the decision of whether a feature is relevant or not a non-trivial task.

In order to allow two multimedia objects to be considered more or less similar, similarity measures must be applied. In this case, similarity measures are metrics that represent the similarity between two descriptor objects. These metrics can be adjusted in order to obtain different results, depending on desired features. It is important to note that maintaining a descriptor database in a multimedia information retrieval system can be particularly useful (since the descriptor extraction process can be slow for complex objects), by indexing the important features of each multimedia object in the form of an easily measurable object. Regarding the multimedia object collection, for optimal results, all objects should have their descriptors calculated offline, leaving only the interrogation's descriptor extraction to be computed online.

The generic structure of a 3D object retrieval system not only inherits the properties of a multimedia information retrieval system (Figure B.1), it is also usually composed of a collection of 3D models, a database that includes a set of meta-information for each model, describing features relevant for shape identification, and an interface that allows the user to specify his interrogation. The interrogation types typically range between text-based query, sketch-based query or query by example. So that a match can be made between the user's interrogation and one or more models of the database, query descriptors must also be extracted from the interrogation.

Tangelder and Veltkamp [21] have a very thorough survey regarding 3D object retrieval, explaining different methods as well as discussing the advantages and limitations of different approaches. In this survey, shape matching approaches are classified as either feature based, graph based or geometry based methods.

Feature based shape matching relies on the geometric and topological properties of 3D shapes. These properties can be taken into account as local or global features, which can be useful if the interrogation of a user represents the model as a whole, a simple feature of the desired object or a generic

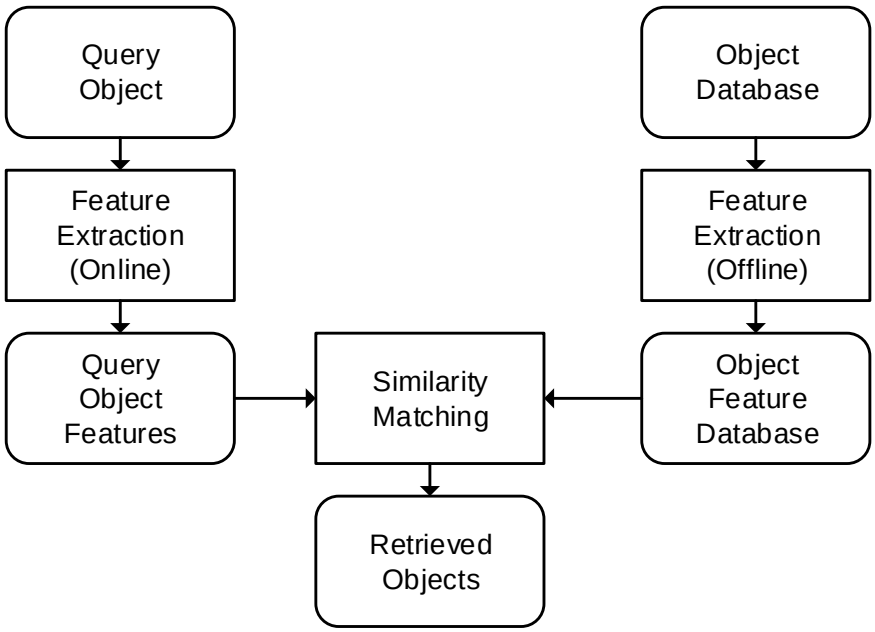


Figure B.1: Generic scheme of a Multimedia Information Retrieval system.

object with a specific feature in a specific location. On the other hand, graph based shape matching considers the positioning of the different components of a shape and how those are linked. Lastly, geometry based shape matching compares views of models in different angles, resembling the more natural practice of how a human being would compare two shapes.

Regarding feature based shape matching, similarity measures must be used to allow two different shapes to be compared. Bustos et al. [22] describe feature vectors (FVs) as vectors of numerical values that represent the characteristics of an object based on the nature of the extraction method. These feature vectors are nothing but numeric descriptors. The authors define five different categories of descriptors: statistics (reflect basic properties like the number of vertices or faces), extension-based (calculated from features sampled along certain spatial directions from an object's center), volume-based (obtained by discretizing object surface into voxel grids or relying on the models to already have volumetric representation), surface geometry (focus on characteristics derived from model surfaces) and image-based (reducing to an image similarity problem by rendering and comparing 2D projections).

An effective descriptor should be invariant to changes in the translation, rotation, reflection and scale of 3D models in their reference coordinate system (invariance property), as well as robust with respect to small changes in the level of detail, geometry and topology of the models (robustness property).

Voxels can also be used to compute the volume of 3D models. Zhang and Chen [23] describe voxels as individual points in space, that are either '1' or '0', and indicate whether the point is inside or outside the object, respectively. After the model is converted into a discrete 3D grid of voxels, the sum of voxels inside the model can be calculated and treated as an approximation of the volume of that model. Despite being a reliable method, transforming a 3D mesh model into its volumetric representation is

a time-consuming task and requires a large storage space. Alternatively, Zhang and Chen use an approach to calculate 3D volumes that is extended from the calculation of 2D polygon areas. Before the actual calculation, a preprocessing phase called triangulation occurs, that ensures that all polygons in the model are triangles.

Triangulation allows the normal of the faces to be computed trivially, applying the right-hand rule by the order of the vertices. If the common edge of two triangles has different directions, then the normal of the two triangles are consistent. Then for each triangle, each vertex is connected with the origin and a tetrahedron is formed. Each elementary tetrahedron has a volume and a sign that is determined by checking if the origin is at the same side as the normal with respect to the triangle. The sum of the volume of all the elementary tetrahedron results in the volume of the model.

To ensure that directions and distances can be compared between different models, a canonical reference coordinate frame has to be found, and as described by Bustos et al. [22] (and also used by Zhang and Chen [23]), the PCA method (pose estimation by principal component analysis) is the predominant method to do so.

With a global-feature based shape matching approach, Ohbuchi et al. [24] also developed a method for shape similarity systems that can handle ill-defined shape definitions, often referred as ”polygon soups”. These shape definitions give a visual impression of 3D shapes by using collections of polygonal meshes, independent polygons, line fragments and points. In this particular method, the shapes are analysed subdividing the model into slabs along the axis and then, for each slab it is computed an average from the distance relation between the surface points and the center of the slab.

Ankerst et al. [25] explored a 3D shape similarity model with shape histograms as intuitive and discrete representations of complex spatial objects, and with an adaptable similarity distance function to cope with shape histograms that may take small shifts and rotations into account. The histograms are based on a partitioning of the space in which the objects reside, with complete and disjoint decompositions into cells which correspond to the bins of the histograms. The authors describe three decomposing models: a shell model (decomposes the model into concentric rings of various sizes), a sector model (divides the model into concentric ”slices”) and a spider web model (combines the divisions of the first and second model). Funkhouser et al. [26] also described the Spherical Harmonic Descriptor as a rotation invariant representation of the Gaussian Euclidean Distance Transform, obtained by computing the restriction of the function to concentric spheres and storing the norm of each harmonic frequency.

In the field of graph based shape matching and retrieval, Sundar et al. [27] developed a method that encodes the geometric and topological information in the form of a skeletal graph and uses graph matching techniques to match the skeletons and compare them. The authors describe the structure of a skeleton as a capable shape descriptor as it captures the notion of parts and components. Also, being intuitive, it can be easily edited by the user to help refine a particular search query. The steps for the skeleton matching process include obtaining a volume, computing a set of skeletal nodes, connecting the nodes into a graph, and then indexing into a database and/or verification with one or more objects. The graph matching algorithm outputs the number of nodes matched, size of the clusters of nodes matched, and information about which nodes were matched to which other nodes, allowing to measure

the quality of the match.

In the matching phase, the computed descriptor and the indexing structure are used to determine what are the closest results in the multi-dimensional space. The end result is a list of the objects present in the database that most closely resemble the queried information. The next step of the 3D object retrieval process is the presentation of the results to the user, who may wish to refine the query with specific details or even use a result (or a combination of results) as query for a posterior search.

Appendix C

MPEG-7 Descriptors

The Perceptual 3D Shape Descriptor (P3DS) is a MPEG-7 descriptor that uses information of the relations between the different parts of a given 3D model to classify it. To compute the P3DS descriptor, a model is subdivided into parts as primitive shapes (ellipsoidal blobs) and an attributed relational graph is constructed using the information about the relative positioning and size of the parts, resulting in the shape descriptor. Another MPEG-7 3D shape descriptor is the Shape Spectrum Descriptor (SSD), which extracts information from the local attributes of a 3D surface and forming an histogram descriptor based on the principal curvatures of a shape. In general, 3D shape descriptors take an important role in 3D object retrieval, as described in the following section.

Appendix D

User Evaluation Guide

Guião para Testes com Utilizadores

Antes de qualquer teste o setup deverá ser o seguinte:

- Computador auxiliar:
  - Iniciar o emissor da aplicação de transmissão de cliques da caneta (com o IP desta máquina);
  - Iniciar a build final do EnContRA server;
  - Ligar o transmissor BlueTooth da caneta e ligar a caneta;
- Computador da mesa multi-toque:
  - Iniciar o receptor da aplicação de transmissão de cliques da caneta (emparelhado com o emissor no computador auxiliar por IP);
  - Builds finais da aplicação Destructive para as 3 abordagens (caneta, multi-toque e combinado de toque e caneta), prontas a correr no browser Chromium.
  - Aplicação de screen capture pronta a gravar os testes.

Para cada utilizador, o processo de avaliação deverá ser o seguinte:

1. Apresentação.
2. Realização do questionário de perfil pessoal;
3. Breve apresentação da solução e das diferentes abordagens através de um curto vídeo explicativo;
4. Sessão de treino (com período máximo de 3 minutos);
5. Tarefa de modelação do candeeiro de secretária (com período máximo de 5 minutos) utilizando uma das 3 abordagens estudadas (a ordem deverá ser alternada para os diferentes utilizadores). Preenchimento do questionário respectivo à abordagem praticada. Medição do tempo dispendido na tarefa e armazenamento do resultado final em forma de modelo 3D. Repetir para as restantes abordagens.
6. Questionário sobre aspectos gerais da aplicação.
7. Agradecimentos.

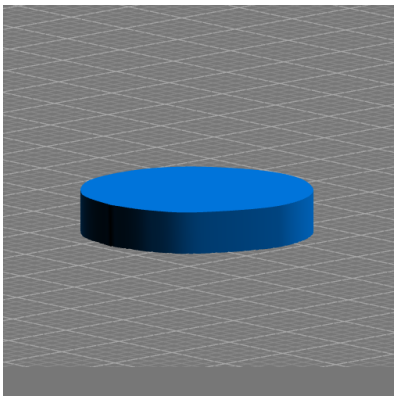
Todos os utilizadores deverão realizar os testes de pé e junto à superfície multi-toque interactiva. Todos testes têm início no estado inicial da aplicação Destructive, em que não existem outros objectos na cena. Com o devido consentimento dos utilizadores, deverá ser documentado o processo de testes sempre que possível, seja por forma de notas, fotografia ou vídeo.

Appendix E

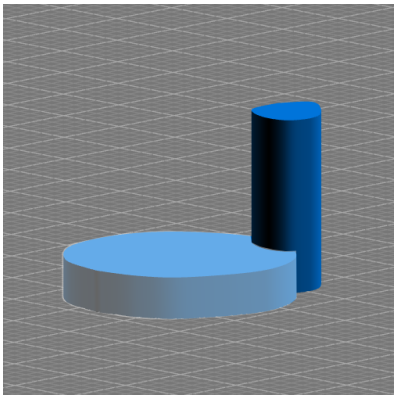
User Evaluation Task

Tarefa: Modelação do Candeeiro de Secretária

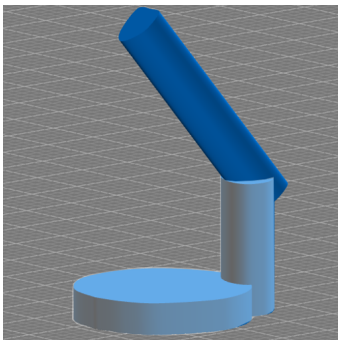
Utilizando a técnica indicada, modele um candeeiro de secretária simples no Destructive. Em baixo descrevem os passos aconselhados para a construção do modelo.



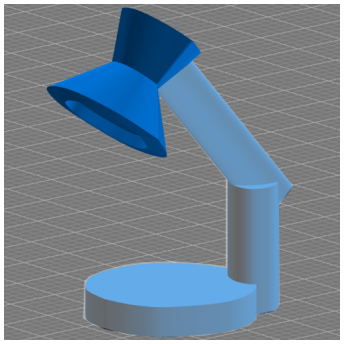
Passo 1: Base



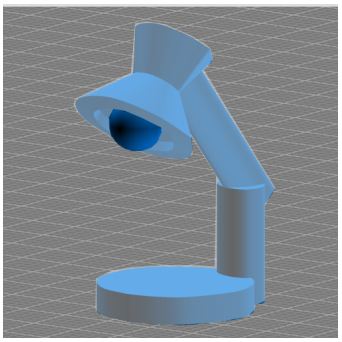
Passo 2: Coluna



Passo 3: Braço



Passo 4: Abajur



Passo 5: Lâmpada

Muito obrigado pela sua colaboração!

Appendix F

User Evaluation Form - Profile

1.1 Género *

- A) Masculino
- B) Feminino

1.2 Idade *

1.3 Nível de escolaridade *

1.4 Classifique a sua experiência com: *

	A) Nenhuma	B) Pouca	C) Alguma	D) Muita
1.4.1 Aplicações de Modelação 3D (SketchUp, Blender, etc.)				
1.4.2 Dispositivos Multi-toque (smartphones, tablets, etc.)				

1.5 Se indicou que tem experiência com dispositivos multi-toque, indique com que frequência os utiliza:

- A) Menos de uma vez por semana
- B) Uma vez por semana
- C) Uma vez por dia
- D) Múltiplas vezes por dia

1.6 Se indicou que tem experiência com dispositivos multi-toque, indique quais:

Appendix G

User Evaluation Form - Task using a specific Interaction

Abordagem utilizada *

- Multi-toque
- Caneta
- Combinação de caneta com multi-toque

2.1 Quão divertido foi modelar o objecto? *

- A) Nada divertido
- B) Pouco divertido
- C) Divertido
- D) Muito divertido

2.2 Classifique o grau de dificuldade para: *

	A) Muito difícil	B) Difícil	C) Fácil	D) Muito Fácil
2.2.1 Modelar o objecto				
2.2.2 Desenhar um traço				
2.2.3 Mover a câmera				
2.2.4 Rodar a câmera				
2.2.5 Fazer zoom com a câmera				
2.2.6 Seleccionar um objecto				
2.2.7 Mover um objecto				
2.2.8 Rodar um objecto				
2.2.9 Ampliar/Reduzir um objecto				

2.3 Classifique a fluidez do método: *

- A) Inflexível
- B) Pouco fluído
- C) Fluído
- D) Muito fluído

Appendix H

User Evaluation Form - Tools

3.1 Classifique a utilidade das ferramentas utilizadas: *

	A) Inútil	B) Pouco útil	C) Útil	D) Muito útil
3.1.1 Ferramenta de Extrusão	○	○	○	○
3.1.2 Ferramenta de Pesquisa	○	○	○	○
3.1.3 Lista de Expectativas	○	○	○	○

Appendix I

User Evaluation Results - Profiles

UserID	1.1	1.2	1.3	1.4.1	1.4.2	1.5	1.6
1	A	24	Licenciado	B	D	D	1.6.B
2	A	25	Estuda Licenciatura	C	D	D	1.6.B
3	A	24	Licenciado	A	D	D	1.6.A
4	A	25	Estuda Licenciatura	C	D	D	1.6.A
5	A	24	Licenciado	B	D	D	1.6.B
6	A	24	Licenciado	C	D	D	1.6.B
7	A	22	Estuda Licenciatura	B	D	D	1.6.B
8	A	26	Licenciado	C	D	D	1.6.A
9	B	24	Estuda Licenciatura	A	D	D	1.6.B
10	A	23	Licenciado	B	D	D	1.6.B
11	A	25	Estuda Licenciatura	A	D	D	1.6.A
12	A	24	Licenciado	B	D	D	1.6.A
13	A	23	Licenciado	B	D	D	1.6.A
14	A	22	Licenciado	C	D	D	1.6.A
15	A	24	Licenciado	A	D	D	1.6.A
16	B	21	Curso Profissional	A	D	D	1.6.A
17	B	24	Licenciada	A	D	D	1.6.A
18	B	50	12º ano	A	C	D	1.6.A
19	B	53	11º ano	A	B	D	1.6.A
20	A	58	9º ano	A	A	B	1.6.A

Legenda:
1.6.A Smartphone
1.6.B Smartphone e Tablet

Appendix J

User Evaluation Results - Pen-based Approach

UserID	2.1	2.2.1	2.2.2	2.2.3	2.2.4	2.2.5	2.2.6	2.2.7	2.2.8	2.2.9	2.3	Time
1	C	A	C	C	B	C	C	C	B	C	C	15:10
2	C	B	C	C	C	C	C	C	A	C	B	8:19
3	D	C	D	D	B	D	D	C	D	D	C	7:55
4	B	C	C	C	C	C	B	C	C	C	B	14:06
5	C	B	D	D	B	D	C	B	A	C	D	8:19
6	C	C	D	D	D	D	D	C	C	C	C	8:23
7	B	C	D	C	C	C	D	C	B	C	B	6:34
8	B	D	D	D	D	D	D	D	D	D	C	9:57
9	B	C	D	D	C	C	D	C	C	D	C	9:27
10	C	C	D	D	C	D	D	D	C	D	B	8:30
11	C	B	D	C	C	C	D	D	D	D	C	12:05
12	C	C	D	D	D	D	D	C	C	C	B	8:49
13	C	B	C	C	C	C	D	B	C	C	B	10:17
14	B	C	D	D	D	D	D	D	C	D	B	9:57
15	C	C	D	D	D	D	D	D	D	D	B	8:27
16	B	B	D	C	C	C	D	C	B	C	B	12:27
17	D	C	D	D	D	D	D	D	C	D	C	11:33
18	D	C	D	C	C	C	D	D	C	D	C	14:37
19	C	B	C	C	C	C	B	A	A	A	C	14:56
20	D	C	D	B	B	B	C	B	B	B	D	16:26

Appendix K

User Evaluation Results - Touch-based Approach

UserID	2.1	2.2.1	2.2.2	2.2.3	2.2.4	2.2.5	2.2.6	2.2.7	2.2.8	2.2.9	2.3	Time
1	C	C	C	C	C	D	C	C	C	C	B	8:00
2	D	B	D	C	C	C	B	C	C	D	C	7:15
3	C	B	D	C	B	C	C	B	B	C	C	9:50
4	C	C	D	C	B	B	D	D	B	C	C	11:15
5	C	B	D	C	B	C	D	C	A	B	C	5:50
6	C	C	D	D	C	C	D	D	C	C	C	7:59
7	C	C	C	C	B	C	C	C	B	B	C	5:21
8	D	D	D	D	C	C	D	D	C	D	C	10:36
9	C	C	C	D	C	C	D	C	B	C	C	8:10
10	D	C	D	C	C	C	D	C	C	C	C	10:20
11	B	C	D	C	C	C	C	C	C	C	C	7:10
12	D	C	D	D	D	D	D	C	C	C	C	7:30
13	C	C	D	C	C	C	D	C	C	C	D	6:34
14	D	C	D	C	C	C	D	C	C	C	C	9:30
15	D	D	D	D	C	C	D	D	C	D	D	7:10
16	C	B	D	C	B	B	C	B	B	B	C	10:10
17	C	C	D	C	C	C	D	D	C	D	D	8:00
18	C	B	C	B	B	B	C	B	A	A	B	10:57
19	C	A	C	C	B	B	C	B	A	B	B	16:23
20	C	A	C	B	A	A	C	B	A	A	C	-

Appendix L

User Evaluation Results - Combined Pen and Touch Approach

UserID	2.1	2.2.1	2.2.2	2.2.3	2.2.4	2.2.5	2.2.6	2.2.7	2.2.8	2.2.9	2.3	Time
1	C	C	D	C	C	D	D	D	C	C	D	6:20
2	D	B	D	C	B	C	C	B	B	C	B	6:38
3	D	C	D	B	B	B	C	C	D	B	B	5:23
4	C	B	C	C	B	B	C	C	B	B	B	9:10
5	C	C	D	B	B	C	D	D	A	B	D	7:28
6	C	C	D	D	C	C	D	C	C	C	D	6:15
7	D	C	C	C	B	B	D	C	A	B	D	5:10
8	D	C	D	D	C	C	D	D	C	C	D	10:36
9	C	B	C	C	B	B	C	C	A	B	D	4:59
10	D	D	D	D	D	D	D	C	C	C	D	5:38
11	D	C	D	D	C	C	D	C	C	C	D	7:23
12	D	C	D	C	B	C	D	C	C	C	D	7:18
13	C	C	D	C	C	C	D	C	B	C	D	6:00
14	D	C	D	D	C	C	D	D	C	D	D	11:50
15	D	D	D	D	C	C	D	D	C	C	D	5:02
16	C	C	C	C	B	B	C	C	B	B	D	9:36
17	D	C	D	D	C	C	D	D	C	D	D	7:28
18	C	B	C	B	B	B	C	B	A	A	C	12:03
19	B	B	C	C	B	B	C	B	B	B	B	14:27
20	C	A	C	B	A	A	C	B	A	A	C	-

Appendix M

User Evaluation Results - Tools

UserID	3.1.1	3.1.2	3.1.3
1	D	B	C
2	D	B	D
3	D	B	D
4	D	B	D
5	D	C	D
6	D	B	C
7	D	C	D
8	D	C	D
9	C	A	C
10	D	B	D
11	D	C	D
12	D	B	D
13	C	B	D
14	D	B	D
15	D	B	D
16	D	A	D
17	D	B	C
18	C	B	C
19	C	A	C
20	D	C	D