Interactive image-based model building for handheld devices

Pished Bunnun * Dima Damen † Sriram Subramanian ‡ Walterio W. Mayol-Cuevas §

Department of Computer Science
University of Bristol, UK

ABSTRACT

Models of objects in 3D and their creation process are central elements for a variety of applications in Augmented Reality. In this paper we extend our previous work in the area of interactive in-situ model building for handhelds and in particular we study two handheld devices in the context of 3D feature selection. In a controlled experiment we compare two interactive methods to specify 3D vertices and discuss the findings of our results and identify design recommendations. We further extend the system by incorporating an object detection process which starts and/or re-initializes the tracker when lost creating a more fluid interaction process.

1 INTRODUCTION

With the availability of camera enabled mobile devices the possibility of contact-less interactive and in-situ model building becomes closer to reality. This will allow users to capture models of objects they encounter from a mobile device while ensuring that the resulting models contain the desired object's features. While interactive model building from images has existed for some time now, e.g. the seminal work of [4] and recent significant improvements as in [9, 8], the process has normally considered off-line methods and desktop interfaces.

An in-situ modeling process demands a new approach for both techniques and platforms as well as an understanding of what interactive methods are more suitable. An in-situ approach however enables to use the models immediately and opens applications that range from user acquired content for gaming or virtual worlds, to better on-the-move graphical authoring.

One of the central components of any in-situ model building *using a handheld device* is indeed the way to point and select object features to be modeled. Pointing and selection techniques have been studied primarily in the context of Virtual Reality, and a number of approaches exist (see [1] for a review). A conventional approach is to use image-plane techniques that use projections of the 3D objects into a 2D image from where the object or object features are selected.

While there is limited investigation into appropriate interaction mechanisms to aide in-situ 3D model building using cameras there is related work that explores 3D feature selection on volumetric displays [7]. Grossman and Balakrishnan present a suite of interaction techniques that facilitate selection of digital objects on a 3D display. The tasks of selecting objects in a 3D display and in the real-world are however different, since the object cannot be enhanced by any graphical augmentation which can guide the selection process, something that would indeed need a previously acquired model of such object.

*e-mail: pbunnun@cs.bris.ac.uk
†e-mail: damen@cs.bris.ac.uk
‡e-mail: sriram@cs.bris.ac.uk
§e-mail: wmayol@cs.bris.ac.uk

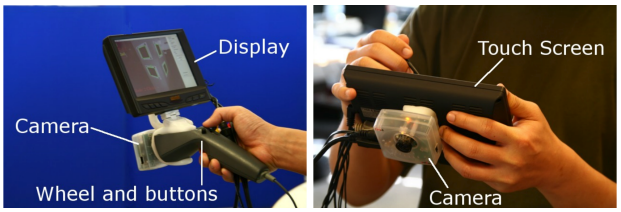


Figure 1: Two devices used in the evaluation. Left: wand-like device with screen, camera and wheel and buttons interface. Right: touch-screen fitted with camera.

To explore devices and interaction techniques that assist in-situ 3D model building, we built two devices (see Figure 1) - a standard touch-screen for two-handed usage and a small display mounted on a wand-like device with a scroll-wheel for one-handed usage; and devised two interaction techniques for 3D point selection – one based on intersection of two rays (two-click) and another based on specifying a ray and moving a point along it (click-and-move).

While some recent systems have explored other outlining methods for model building e.g. silhouette selection as in [10], fine feature modeling may still require interacting with individual 3D vertices or mesh points and these are the ones we concentrate our evaluation in this paper.

In this paper we are also interested in modeling 3D objects that can be defined by their outline edges and less so by their texture appearance. This is in contrast with the objects suitable for the more popular point-based feature descriptors and modeling methods.

In the next section, we explain the system overview and followed by a user study in Section 3. Section 4 describes new model building features. Section 5 briefs an object detection integrated into the system and some results. We end with conclusions in Section 6.

2 SYSTEM OVERVIEW

The main components of an in-situ 3D model building system are the model generation algorithm, the device that is used as the modeling tool and the interaction techniques used to dynamically input model parameters.

2.1 Model Building

As in our previous work that we called OutlinAR [2], we use a method based on the model tracking of Drummond and Cipolla [5] with a few modifications. The method is robust to partial occlusions and fast. This method uses a wireframe model of the object to be tracked, and computes the pose of the camera from this model in all six dimensions (3 translations and 3 rotations). With this method the modeling process starts when the camera is positioned from a known template of known dimensions. The process then bootstraps from this template and adds new line features to the existing wireframe model based on the known camera pose as the camera is moved around. In our system the 3D model of the object itself will be used for tracking while the user is adding a new part of the object. We have, in addition, considered a method that requires no template as described later as well as enhanced the tracker by developing a mechanism for re-positioning the model after the tracker is lost as explained in Sec 5.

2.2 Devices

We built two devices (see Figure 1.), the first being a wand-like device that contains the interface of a 3-button wheeled mouse arranged in a handheld case, and a second device being a touch-screen tablet with stylus. Both of these have the same wide-angle lens camera and display. The camera has 80 degrees of horizontal field of view that provides a 320×240 pixel image stream at 30fps. We prefer a monocular system as an example of a non-radiating and currently ubiquitous feature on several kinds of mobiles. A single camera system also highlights the benefit of having an interactive approach with the person providing decisions that could pose significant challenges to fully automatic methods. The display used in both devices is a 7“ VGA LCD touch-screen providing touch input via USB interface. Both devices are connected via cable to an Intel Core2 Duo CPU 2.2 GHz with 2GB of RAM laptop.

2.2.1 Wand-like device (WnD)

This device is aimed for single-handed operation, with the thumb operating the wheel and buttons. The scroll wheel is placed at the centre and one button is activated by pressing the wheel. The other two buttons are located left and right to the wheel. The camera is placed at its tip and the mouse interface inside.

2.2.2 Touch-screen device (TS)

The touch-screen is connected by cable to the VGA card of the computer and it is operated by a stylus. Its nature makes it a two-handed device with one hand used for holding it. Clicking is emulated by tapping the screen, while scrolling emulated by dragging the stylus. Dragging upward gives an upward scrolling and downwards gives a downward scrolling.

2.3 Defining 3D points from 2D views

The most basic operation in generic model building is to specify a point in 3D space to serve as a models vertex, or perhaps to serve as an anchor to where an already defined shape can be attached to. Given the camera pose as computed from the model based tracking, the first step involves defining the first view of an objects vertex. This produces only a 3D ray in space that links the camera with the vertex in space. After defining this ray, there are two techniques that can be used for specifying the 3D vertex in space.

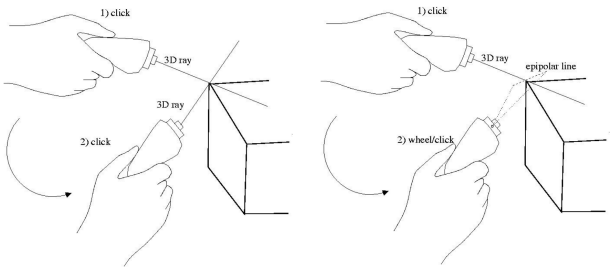


Figure 2: Sketch of the two pointing techniques used to select 3D features. Left: (2Cs) where two rays in 3D intersect to define a vertex. Right: (CnM) where one ray in 3D is used to define the vertex along its length from a second viewpoint.

2.3.1 Two Clicks (2Cs)

The 2Cs method consists of taking the camera to another viewpoint from the one where the first ray was defined, and one more click to define a second 3D ray in space as shown in Figure 2. These rays may not intersect perfectly so the closest point between these rays is used instead.

2.3.2 Click and Move (CnM)

The CnM method defines the vertex by computing the epipolar line based on the first 3D ray for any other viewpoint. The epipolar line is a ray in 3D that projects the first ray into the current view of the camera. Therefore, the positioning of the vertex is constrained to lie along the first ray and what is then needed is to specify where along this ray the vertex is. This is done by using a scroll motion until the vertex is positioned and finally selected by a click. This method is also sketched in Figure 2.

3 User Study

The main experimental goal was to examine the differences in accuracy and speed of the different techniques and devices. The experimental software recorded trial completion time and accuracy in 3D. The completion time was the duration between the first and second clicks for defining 3D points. The accuracy was calculated as the mean squared error between the user-estimated target location and the ground truth that was computed a priori.

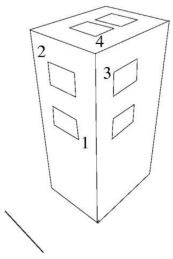


Figure 3: The object used during experiments. Numbers indicate 3D point ID. The line on the ground indicates the boundary the participants were asked not to cross.

3.1 Task and Stimuli

A 3D object as sketched in Figure 3 is used for the experiments. To ensure stable and accurate tracking, six known rectangular templates with high contrast features are placed around an office cabinet. The known coordinates of the templates were given to the tracker. There were 4 corner points labeled with numbers. These 1-4 corners offer a range of positions around the object and were at 70, 119, 119 and 132 centimeters from the ground respectively and in different planes.

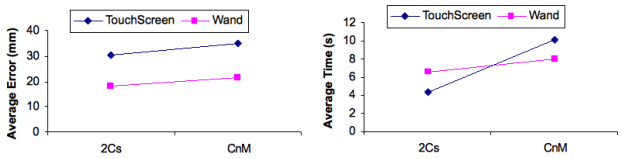
3.2 Procedure and Design

We carried out the experiment with 10 participants (3 female and 7 male) between the ages of 20 and 40. All participants were tested individually. Each experiment lasted approximately 30 minutes. We designed a within-subject experiment to evaluate the techniques and devices and used a $2 \times 2 \times 4$ within participants factorial design. The factors were: Devices WnD and TS; Techniques 2Cs and CnM; Target locations 4 locations as indicated in Figure 3. With a total of 8 trials per condition we collected a total of $10 \times 2 \times 2 \times 4 \times 8 = 1280$ trials.

Participants were briefed on the way to operate the devices and techniques and given several practice trials to get familiar with the task, device and techniques. The order of presentation of techniques and devices was counterbalanced between participants for learning effects using a partial latin square. They were instructed to do the task as fast as they could and told not to move beyond a border 70cm away from the front view of the cabinet, although this was not compulsory. At the end of the experiment users were asked to rank-order the four conditions in terms of overall effort, speed and performance.

3.3 User Study Results and Discussion

The average trial completion time was 7.2s (s.d. 7.26s). There were 48 trials (3% of data) that were outside 3 times s.d. of the mean. These trials were considered outliers and excluded from further analysis. We did a univariate ANOVA on the aggregated data with time as the dependent variable, device and technique as fixed factors. The ANOVA did not show any significant effect of device ($F(1,36) = 0.408, p = 0.5$) but showed a significant effect of technique ($F(1,36) = 13.8, p < 0.01$) on trial completion times. As can be seen from Figure 4 the 2Cs technique was significantly faster than CnM. The average times for both TS and WnD were similar (7.27s and 7.28s respectively). Univariate ANOVA revealed a significant effect of device ($F(1,36) = 17.14, p < 0.01$) but no effect of technique ($F(1,36) = 1.34, p = 0.25$) on error-rate. As can be seen on Figure 4-right, the WnD resulted in significantly less mean errors (19.8mm) vs the TS (32.5mm). The average error for 2Cs was 23.9mm against 28mm for CnM. Figure 5 shows the users ranking of the different techniques and device combination. The TS combined with the CnM was the least preferred technique in terms of Overall Effort (7 out of 10 ranked it 4th), Speed (5 out of 10) and Performance (5 out of 10).



(a) Error (mm) (b) Average trial completion time (s)

Figure 4: Per device and per technique.

Both the WnD based techniques were well liked by the users. Overall users had a marginal preference for WnD with CnM over WnD with 2Cs. When we conceived the wand based technique, one of the problems we faced was that there was no direct way to alter the point of interest on the device, this is that, the point of interest was always the centre of the image and the user had to move this centre (identified by a cross-hair) to the right 3D vertex. This meant that even if the vertex of interest was within camera view the user had to physically move the device to re-position the centre of the display on the vertex.

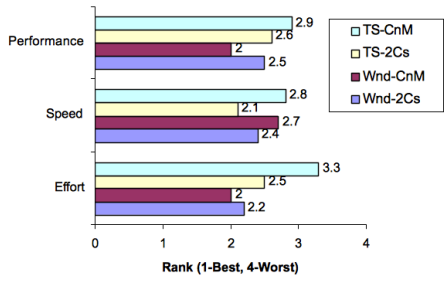


Figure 5: Average user ranking of the different technique and device in terms of Effort, Speed, and Performance.

To overcome this problem we built the TS. In the TS users can easily pick their point of interest by moving the cross-hair to a new position on the visual workspace. However, rather paradoxically this meant that the device did not know what the point of interest of the user was anymore. This meant that the device could not provide cues about the reliability of the selected point until after the user had tapped on the screen. This resulted in an increased number of errors with the TS device. Moreover for the second click in the 2Cs, when using the TS it is not possible to give constant visual feedback based on the point of interest, the user may have to move and click the device screen several times until the second ray is accepted. These conditions made users feel more frustrated.

For the WnD, the device always knows the user's point of interest (it is along the ray passing the centre of the image) and therefore it is possible to provide continuous and dynamic information about that point. This is akin to providing tooltip on a mouse when one hovers over an object of interest. We were able to leverage this, to let the user know before they select a vertex whether the highlighted point can potentially yield good results. Almost all users found this extremely useful and it helped reduce the overall error-rate of the wand-like device. It is important to consider errors relative to the device-object distance and camera resolution. For example although the average errors in this experiment were between 19.8 and 32.5mm, this can also measured as between 5.8 and 9.5 pixels if the camera is at 70cm. Naturally if the modeling tool is closer and is based on visual tracking this has to be balanced with how reliable the tracking can be based on how much field of view the system has. Alternatively, it is expectable that these error figures can be improved if the resolution of the camera is increased. Overall, the 2Cs performs faster than click and move and has less error in both devices. Our results suggest that the wand-like device along with the 2Cs is the best system to use for 3D vertex definition. The wand-like device resulted in significantly fewer errors while the two-click was significantly faster that the click-and-move. It is however important to note that the 2Cs demands stable tracking which can be achievable only by having sufficient features in any view where the device is taken to. This may not be the case and in these situations, the CnM is an option since it allows for the definition of 3D points from narrower angles thanks to the constraint imposed from the first ray. Even though the devices were purpose built for our experiment, we believe that the techniques and hardware can be easily included in emerging camera enabled devices.

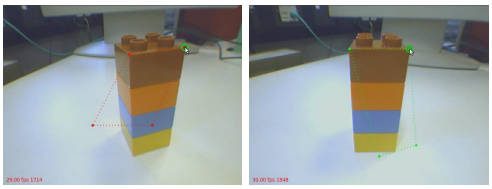
4 OutlinAR-2 on a Touchscreen

The tracker in the original OutlinAR relied exclusively on outline edge features of the target object. The system does not have or aims to build a precise model of the environment we chosen to develop the system in this way to reduce computational time and make it be able to model objects in a situation where a map is impractical to build. However as a result, the tracker will not be stable enough in all cases for users to freely move the device for WnD and 2Cs especially at the beginning of the modeling process.

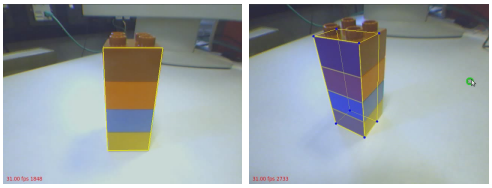
While the evaluation in Sec 3 suggests to use the wand-like device, we are aware that the vast majority of current mobile devices use a touchscreen and therefore expanding our new work to also work with these is important. We have therefore selected TS and 2Cs for the new framework. To start, we have in addition to the original features, added the ability for users to pause an image and select many vertices from one view. With the pause ability, the system is also able to initialize the tracker without any known template with the compromise that the scale of the model will be arbitrary. In the beginning, the user needs to define a 3D plane and builds some parts of the object that belong to the plane. The system will then use this plane to initialize the tracker.

To define a 3D plane, four corners of a polygon are needed to be selected on a single view. The live camera is then paused and each corner is selected and dragged to its corresponding image point. A homography, and then later a rotation and translation between two views will be estimated by using the method explained in [6]. Finally, 3D positions of all four corners are determined by triangulation from which the polygon can be used as a template for building object features. This process is shown in Figure 6.

To define a 3D circle on a plane, five points need to be placed at a circular boundary in the image. The system will use the 2D positions of these points to estimate a flat 3D circle and its plane.



(a) Align corners in 1^{st} view (b) Align corners in 2^{nd} view



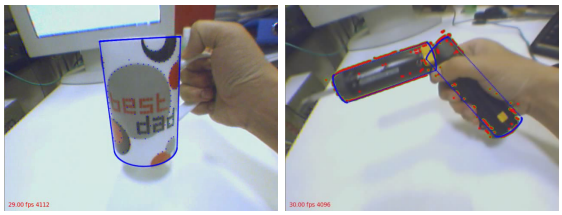
(c) Define 3D plane (d) Polygon is used as template

Figure 6: Defining Rectangular Plane without a Template

Note that this method will need only one view to model the circular plane as shown in Figure 7. Nonetheless, the circular plane remains ambiguous. The user needs to choose the correct plane by defining the plane's normal direction. Now, the relative position between the camera and centre of circle can be estimated but not the orientation around the centre's axis. This problem can be avoided by adding interior edges as show in Figure 8a. This is done by capturing a "snapshot" of edges on the model's surface from a given viewpoint. Inner edges will be later tracked as any other part of the model.

(a) 5 points on a circle (b) Estimate 3D Circular Plane

Figure 7: Defining a Circular Plane without a Template



(a) Inner edges (blue dots) (b) Tracking a screw driver

Figure 8: More Modeling Examples

5 Edge-based Object Detection

It is inevitable that either during the modeling process, the tracking stage or at the point of using the created models for the first time a method is needed to re-position the model according to the input image. In some instances this could be done by the user, however if an automatic method exists this will reduce burden and increase fluidity of use. We have developed a method for the detection of modeled objects based around collections of edges.

Visible edges of the model are used to generate groups of edge clusters which will be used later to detect the object. The approach is similar to the work in [3]. In our case we work with multiple views of a given object generated as a by-product of the modeling process and thus views, visible 3D vertices and camera poses are kept together.

When the detector is active, it will try to match groups of edge clusters between the input image and the stored data. A threshold decides on the matching fitness and the result of the detection is used to estimate a homography between the template and test image. A rotation and translation are then extracted to estimate camera pose. However, the translation is not an absolute value. The 3D visible vertices will be used to correct for scale. After getting the translation and rotation from the detector, the tracker will try to track the model, if it is able to track the model, the system will switch from detecting to tracking as shown in Figure 9. Blue dots show the control points from the tracker, while red dots those generated from the object detector. The performance in Figure 9 goes from 1-3fps during detection in a cluttered scene to 40fps during tracking for a single 3D object using 640x480 images.

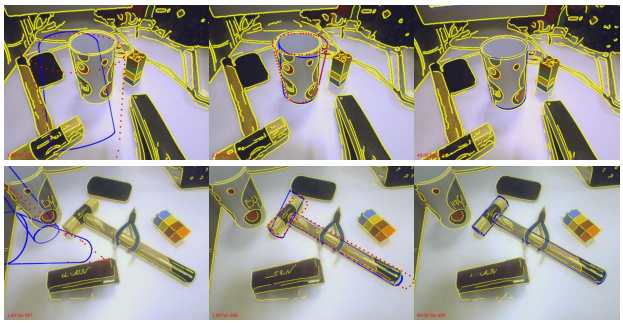


Figure 9: The object detector initializes the tracker again in a cluttered environment for a mug and a hammer.

6 Conclusions

In this paper we investigated interaction issues surrounding 3D model building using mobile devices through an experimental evaluation. We presented interaction techniques and devices aimed at supporting users for in-situ 3D model building. We choose to use the touch-screen device and two-clicks technique in our system. We also have shown how an edge object detection method is integrated with the tracker to offer greater operational flexibility.

References

[1] D. Bowman, E. Kruijff, J. L. Viola, and I. poupyrev. *3D user interfaces theory and practice*. Addison Wesley, 2005.

[2] P. Bunnun and W. Mayol-Cuevas. Outlinar: an assisted interactive model building system with reduced computational effort. In *Proc. IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'08)*, 2008.

[3] O. Danielsson, S. Carlsson, and J. Sullivan. Automatic learning and extraction of multi-local features. In *12th International Conference on Computer Vision*, 2009.

[4] P. E. Debevec, C. J. Taylor, and J. Malik. Modeling and rendering architecture from photographs. In *SIGGRAPH96*, August 1996.

[5] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *IEEE Trans. on PAMI*, 24(7):932–946, July 2002.

[6] O. Faugeras and F. Lustman. Motion and structure from motion in a piecewise planar environment. Technical report, INRIA, 1988.

[7] T. Grossman and R. Balakrishnan. The design and evaluation of selection techniques for 3d volumetric displays. In *Proc. the 19th Anual ACM Symposium on User interface Software and Technology*, 2006.

[8] S. Sinha, D. Steedly, R. Szeliski, M. Agrawala, and M. Pollefeys. Interactive 3d architectural modeling from unordered photo collections. In *Proceedings of SIGGRAPH Asia*, 2008.

[9] A. van den Hengel, A. Dick, T. Thormahlen, B. Ward, and P. H. S. Torr. Videotrace: Rapid interactive scene modelling from video. In *Proceedings of SIGGRAPH*, 2007.

[10] A. van den Hengel, R. Hill, B. Ward, and A. Dick. In situ image-based modeling. In *Proc. IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'09)*, 2009.