Safe 3D Navigation

George Fitzmaurice, Justin Matejka, Igor Mordatch, Azam Khan, Gordon Kurtenbach

Autodesk Research
210 King Street East, Toronto, Ontario, Canada
{firstname.lastname}@autodesk.com

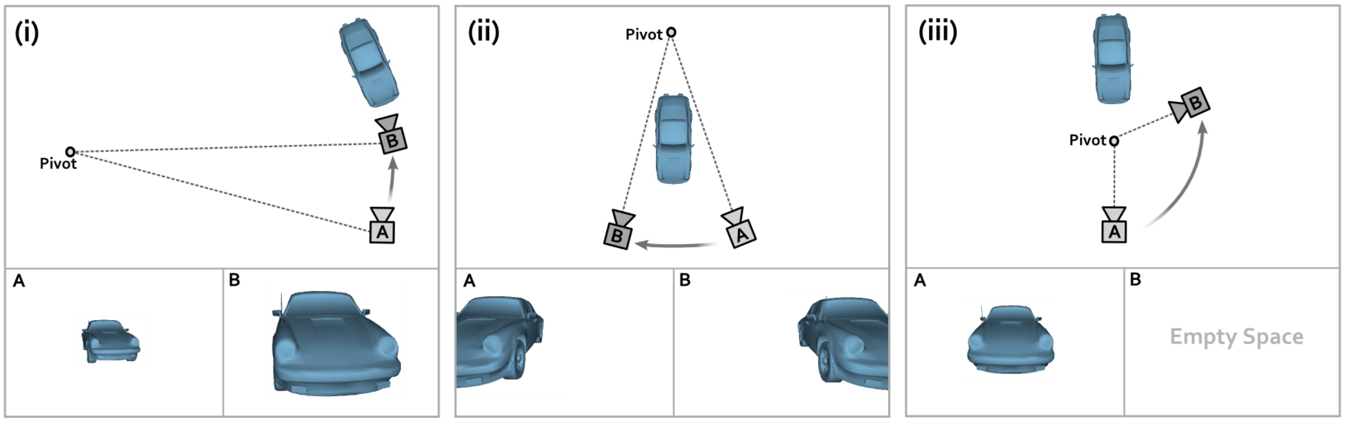

Figure 1: Common problems when using an Orbit tool: above is top view of 3D scene, below is what the user sees on screen (i) The pivot point is off-screen so the Orbit operation has a similar effect as Zoom; (ii) The pivot point is on-screen, but beyond the object, so the Orbit Tool acts like a Pan Tool; (iii) The pivot point is on-screen, but in front of the object, so the Orbit Tool causes the user to look off to empty space.

Abstract

Typical commercial 3D CAD tools provide modal tools such as pan, zoom, orbit, look, etc. to facilitate freeform navigation in a 3D scene. Mastering these navigation tools requires a significant amount of learning and even experienced computer users can find learning confusing and error-prone. To address this we have developed a concept called “Safe 3D Navigation” where we augment these modal tools with properties to reduce the occurance of confusing situations and improve the learning experience. In this paper we describe the major properties needed for safe navigation, the features we implemented to realize these properties, and usability tests on the effectiveness of these features. We conclude that indeed these properties do improve the learning experience for users that are new to 3D. Furthermore, many of the features we implemented for safe navigation are also very popular with experienced 3D users. As a result, these features have been integrated into six commercial 3D CAD applications and we recommend other application developers include these features to improve 3D navigation.

Categories and Subject Descriptors: H.5.2 [User Interfaces]: Graphical User Interfaces (GUI), 3D graphics

Additional Keywords and Phrases: 3D navigation, 3D widgets, Desktop 3D environments, virtual camera.

1 Introduction

For those of us who are familiar with operating 3D applications, we may have long ago forgotten our first experience learning the peculiarities of 3D navigation with a mouse and keyboard. For example the feeling that “I’ve lost the model”—when in fact the model has just been accidentally rotated outside of the viewport (see Figure 1), may rekindle memories that desktop 3D navigation is a learned skilled which can be quite error-prone when the user does not have the proper understanding of the concepts needed and good control of their viewpoint.

With the growing proliferation of 3D CAD tools it is very common for people with some experience with 2D applications to take up 3D applications. We have been studying this class of “2D-to-3D users” for several years and have found that even simple 3D navigation can be very difficult to learn, resulting in many users rejecting 3D tools. Moreover, the users would prefer to remain working with 2D applications even though 3D, if learned, would make their work much easier.

Clearly, new-to-3D users struggle in learning other functions besides navigation in a typical CAD application. However, there is a growing population of people who only really need to learn 3D navigation to perform their job, for example, viewing 3D design plans (rather than creating designs), and therefore we have an opportunity to help these users adopt 3D.

Specifically, we have observed that many users reject 3D tools because their initial interaction with the tool results in an unproductive and unpleasurable experience, even when trying to do the most basic 3D navigation operations like, for example, “just looking at the front and then the back of the gearbox”.

To address this problem we have developed a collection of interaction techniques that support the concept we call “safe 3D navigation”. The goal of safe navigation is that a user’s first learning experience with navigation in a 3D application should be productive and pleasurable. We make 3D navigation “safe” by preventing or circumventing the occurrence of error states and misunderstandings that new-to-3D users typically encounter.

In addition we have found that many of these features that make navigation "safe" also continue to benefit users as their experience grows and they become experts. Moreover, users who are already familiar with 3D, or even experts, can receive the benefits of safe navigation without losing the flexibility and expressiveness of traditional unconstrained navigation.

In this paper we present seven major properties that we have found through user observations that are needed for safe 3D navigation. We describe the interaction techniques we have developed to produce these beneficial properties, and the results of usability testing of these techniques in many different commercial 3D CAD applications.

2 Motivation – Initial User Studies

Initially we conducted several studies where we observed new-to-3D users trying to accomplish simple 3D navigation tasks in a standard commercially available 3D viewer application. A typical user in this study was a "parts order manager", 55 years old, with some experience using 2D applications for reading machine drawings and then ordering parts based on them.

We observed that these types of users typically encountered very frustrating problems on their first exposure to 3D:

- Users transfer beliefs and skills from 2D applications to their detriment. For example, some try to solve all navigation tasks with Pan and Zoom and are very ineffective, inefficient, error-prone and most likely unable to reach their destination.
- Users can have strong (but incorrect) ideas about how the tools should work but do not know about the terminology for 3D navigation tools (i.e., orbit, pan, yaw, pitch, azimuth, etc.) and the expected behaviors of these tools.
- Users do not have a sense that a certain collection of tools are needed to successfully navigate through a space or around an object, and typical interfaces do not assist users in understanding that certain tools will be helpful for their current navigation task, while others will confound their current task.
- This misunderstanding in how tools should work leads these users to quickly get into even more confusion. For example users end up being "lost in 3D space" because they are looking at empty space, or see one solid color because they are inside the model and do not know how to get back. Similarly, they end up in states where the model is a tiny object and cannot get it any larger or they end up in situations where the tools seem to behave erratically.
- Another major problem we observed was that when a user gets into navigation trouble they quickly make it worse, typically by trying to use the same tool that lead to the problem to get out of trouble. However, since the user does not know how the underlying tool operates, they are not quite sure what input gestures will reverse their actions. Alternatively, the user gets into further trouble by picking another inappropriate tool that makes their situation worse.
- Users had little appreciation for the suitability of tools to tasks. For example, users would try to use an orbit tool while inside a building which would almost immediately place them in an unknown and unwanted location, such as inside a wall.

Ultimately, these problems confound learning and the feeling of a system "making sense" to the user and being perceived as "learnable". This all leads to what we call "interaction meltdown"—the user gives up altogether on trying to figure out the system. Thus we are highly motivated to address this situation. We concluded from these studies that it seems that a collection of factors contribute to a "meltdown", so what is needed is a collection of techniques that keep a new user out of trouble —safe 3D navigation— and this seems to require some combination of improved learning and mechanisms to prevent the possibility of getting trapped in error situations.

3 Related Work

Much research has been done on improving 3D navigation in different contexts. In general, our work differs in that we are studying the context of 3D commercial desktop applications and how to improve initial learning for new-to-3D users. Bowman et al. [1997, 1999] study navigation but in the context of immersive environments. Tsang et al. [2002] deal with very easy to learn navigation but use a specialized device; a spatially aware display. In the desktop context some research [Igarashi et al. 1998; Zeleznik and Forsberg 1999] differs from our work as they focus on gestural input. Others augment the desktop with continuous bi-manual input [Balakrishnan and Kurtenbach 1999; Zeleznik et al. 1997] whereas our work uses standard mouse and keyboard input. Others report on new interaction metaphors such as flying [Ware and Fleet 1997; Tan et al. 2001], eye-in-hand [Ware and Osborne 1990], using glances [Pierce et al. 1999], and providing a 3D world in miniature map [Stoakley et al. 1995] in attempting to simplify the task of 3D navigation. Our work does not introduce a different metaphor but augments the typical camera controls found in most 3D commercial CAD applications.

Others have looked at methods for constraining or controlling the virtual camera to offer better control. Gleicher and Witkin [1992] propose controlling the virtual camera by directly manipulating the viewing image. Our work differs as we focus on manipulating the camera. Mackinlay et al. [1990] deal with controlling the velocity of approaching an object while navigating. We use a similar approach which is sensitive to the distance to a target, but this is one feature amongst many others that contribute to safe navigation. Steed [1997] looked at ways of easing navigation by maintaining a realistic height above the ground. Our work differs is that we discovered in our context that manually controlling height is less confusing for new-to-3D users. Chapman and Ware [1992] use *predictor based feedback* to improve control while navigating but we do not use this type of feedback.

In addition, other systems attempt to automatically frame views or compose a collection of relevant pathways given the scene content [Bares and Kim 2001; Bares and Lester 1999; Christianson et al. 1996; Drucker and Zeltzer 1995; He et al. 1996; Phillips et al. 1992; Khan et al. 2005; Hanson and Wernet 1997]. Our work differs in that we try to perform minimal analysis of the scene to allow our approach to work on slower computers or extremely large scenes and use standard camera control functions like orbit, pan, etc. Perhaps the most relevant research is the use of guided tours [Burtnyk et al. 2002, 2006; Gaylean 1995]. These approaches ensure a good viewing experience while trading off greater ranges and degrees of freedom of movement. We adopt Burtnyk's ShowMotion system as part of our approach to reduce the need for manual 3D navigation.

3D computer games offer a form of "safe 3D navigation" where users are typically constrained to remain within the region of game play, collision detection is used, navigation maps determine movement paths etc. [Nieuwenhuisen et al. 2007]. This allows the user to play the game without the burden of having to control arbitrary navigation. Our work differs from game navigation in that, first, we are developing safe navigation for the typical navigation tools (pan, zoom, orbit, etc) used in 3D CAD. Game play navigation is designed for the particular game type, for example, simulating running, turning and shooting or driving. In contrast, navigation for CAD needs to be optimized for object creation, inspection, etc. Second, gaming navigation can be much more "authored"—the scenes, types of navigation and routes are authored ahead of time or dynamically whereas in the CAD domain any sort of scene is possible (as users are generally in the act of creating the scene or are free to load an arbitrary scene). Thus, the types of navigation routes needed are largely unknown ahead of time. In summary, while much research has been conducted on providing efficient 3D navigation tools, our research focus is to provide safe 3D navigation for CAD environments.

4 Properties Supporting Safe 3D Navigation

In our investigation to design a safe 3D navigation experience, we have defined seven high-level properties that work collectively to achieve this goal: (1) cluster and cache tools; (2) create task and skill-based tool sets; (3) provide orientation awareness; (4) enhance tool feedback; (5) offer pre-canned navigation; (6) prevent errors; and (7) recover from errors. While many existing navigation tools offer some of these properties, it is important to realize the need to provide all of these properties at a rich level to achieve a rewarding navigation experience.

4.1 Cluster & Cache Tools

In order for the user to **rapidly access** and experiment with a collection of navigation tools, they should be quickly accessible and always near the user's visual focus. To achieve this, we built our 3D navigation widget (see Figure 2) using a tracking menu [Fitzmaurice et al. 2003]. With a tracking menu design, the tools travel with the cursor. They use a click-through paradigm where the arrow cursor moves within a larger mobile and semi-transparent menu of graphical buttons. Unlike traditional menus, when the cursor crosses the exterior edge of the menu, the menu is moved to keep it under the cursor. The cursor can also be moved within the menu to select items and selecting an item with a button press also "clicks-through" to provide pointing/selection to the data beneath the menu.

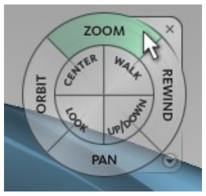

Figure 2: Full Navigation Wheel using a tracking menu.

Menu items placed on the exterior region of the tracking menu have the additional benefit of being easy to hit since a large gesture in that direction will guarantee that menu selection (e.g., a large cursor gesture to the left will select the Orbit command in the Full navigation widget shown in Figure 2).

Next, the user needs to know **which tools** are appropriate and which ones are more important than others. In our solution, we introduce a circular navigation widget with the most common tools (e.g. Pan, Zoom, Orbit) on the exterior and the less used tools on the interior regions (e.g. Center, Up/Down). See Figure 2.

The user must be made **aware of what tools** are available for navigating the scene. Many times this is achieved by clustering tools in a toolbar. However, we have found that this is not sufficient as new-to-3D users typically do not know where to start. As such, we have introduced a "First Contact" experience. It consists of our navigation widget which is grayed-out and sitting in the bottom left of the 3D canvas. Users are naturally drawn to this visual and once the cursor enters the widget, a large graphical tooltip dialog appears to inform them how to use and select a navigation widget for their particular task (see Figure 3). If the user clicks on the tooltip, the navigation widget is activated; otherwise if the cursor rolls off of the tooltip, it is dismissed. This design offers a light-weight way of inquiry and optional activation of the navigation widget.

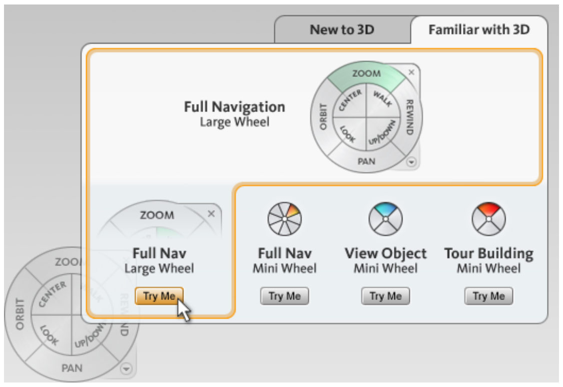

Figure 3: First Contact Dialog.

4.2 Task & Skill Based Configurations

Early in the design process we quickly realized that it is valuable to provide a different collection of navigation tools based on the skill of the user and their intended task. First, it is important to distinguish if a user is familiar with working in 3D. Novices and experts have different needs and expectations. With novices, we found that they need to move around the space with few surprises and do not know how to problem solve to get out of trouble. Experts desire more tools and fewer constraints imposed on the tools. Our First Contact dialog has two tabs at the top for users to self-select whether they are "New to 3D" or "Familiar with 3D". This will guide them to choose the proper navigation widget.

Next, if we know what type of navigation task the user wishes to conduct, we can customize both which tools are available (and as importantly, which are not available) and the behavior of the individual tools. This is particularly useful for novices who can easily get into trouble. In general we distinguish between two high level navigation tasks: "viewing an object" and "touring a building" (see Figure 4) and offer a total of 6 different wheels.

For novices, the **View Object** navigation wheel offers only four tools: Center, Zoom, Orbit and Rewind. With the Center tool, users can interactively place the pivot point by picking a point on the model and this point is moved to the center of the 3D canvas. The Zoom tool performs a "center of canvas" zoom at this pivot point. Orbit operations occur about the pivot point as well. This combination ensures that the 3D model remains visible at all times. Note that we do not offer a Pan operation as this can cause the 3D model to move off of the screen by excessive panning often leading to confusion about the location and behavior of the pivot point and subsequent operations that use the pivot point. In addition, point zoom (i.e. zooming at the current cursor location) is not offered as this also can cause the 3D model to easily fall outside the viewing region by conducting a zoom in open space or by a subsequent orbit operation. Again, safety is achieved by the combination of tools and keeping the pivot point on the model surface. Notice also that the layout of this navigation wheel has the Orbit tool occupying ¾ of the outer region and so, is larger than the other tools. This prominence signals to the user that Orbit is more important and more frequently used than the other tools.

Also for novices, the **Tour Building** navigation wheel offers only four tools: Forward, Look, Up/Down and Rewind. With the Forward tool, users click on the 3D model and move towards that point. If they click and release, they are brought 50% of the way to the point they selected on the 3D surface using a ½ second animation. Alternatively, if they click and drag, a 3D slider is presented (see Figure 8) where they can move from their current position to the surface. They are constrained to not move more than 95% towards the front surface and are also able to move backwards as well (we shoot a ray behind them for collision detection to prevent them from going backward through a wall). The Forward tool in combination with the Look tool (which mimics head movement while a person is stationary) allows users to effectively and safely move around the space. Note that, for example, the Orbit tool is not offered as this often causes unexpected and disastrous results for novices when used inside a 3D model. As a convenience we introduced the Up/Down tool which acts like an elevator and provides a graphical slider to allow the user to adjust their height in the scene (see Figure 7). This tool is often more intuitive for novices than using a constrained Pan operation. In terms of layout, the Forward tool occupies ¾ of the outer region and is larger than the other tools to indicate its importance and ease of access.

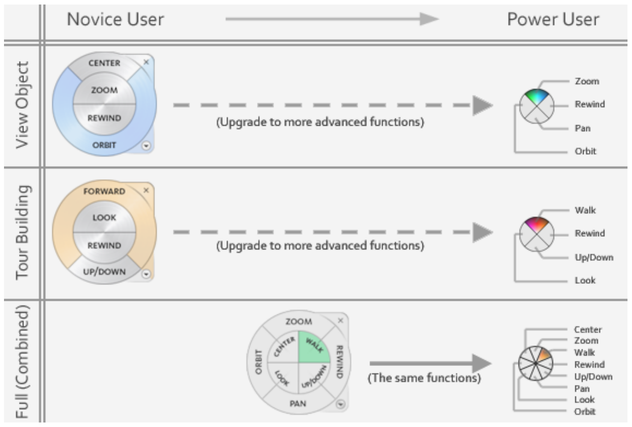

Figure 4: Six navigation wheels whose tool collection, tool placement and size are based on a user's task and skill level.

The **Full** (Combined) navigation wheel (see Figure 4) is designed for more advanced users who are already familiar with working and navigating in 3D. In this wheel we offer tools for both "object inspection" and "touring a building". Specifically, the collection of tools consists of: Orbit, Zoom (point zoom), Pan, Center, Look, Up/Down, Walk (instead of Forward) and Rewind. The more common tools are on the exterior region (Pan, Zoom and Orbit) which are larger and more easily accessed. The Walk tool offers the ability to simultaneously adjust the user's movement speed and viewing direction. With this tool, we found it important to warp the cursor to the center of the screen and create a dead zone region where no movement occurs. This properly orients the user to better predict the walk behavior.

For more advanced users, we offer miniaturized versions of the large navigation wheels for faster tool access and to have a smaller visual footprint. The "mini wheels" (see right-hand column of Figure 4) are roughly the size of the cursor and replace the arrow cursor [Fitzmaurice et al. 2008]. Note also that we upgrade some of the individual tools in the mini wheels. For example, instead of offering the Forward tool, we use a Walk tool in the "Mini Tour Building" wheel.

4.3 Orientation Awareness

Another common problem we have observed with new-to-3D users is loss of orientation. This problem manifests in some different ways. One surprising but common problem is what we call "3D amnesia"; users seem to forget where they are in 3D. For example, in observing some skilled Photoshop (2D) users, we found that some misinterpreted 3D scenes. For example, when first shown a ¾ perspective view of a box and then switched to a side view of the box (by clicking on a "side view" icon) without any animated rotation between the two views, users misinterpreted the results. Typical comments would be: "my box disappeared and there's only a square now". Clearly for some users, they were confounded by their familiarity with 2D applications and had a difficult time understanding and realizing that there were hidden elements in the depth dimension. Once they were told, or realized that the object was three dimensional, users requested different views of the object not by specific view naming conventions like "front view" or "bird's eye view", etc. but by spatial relationships relative to their current view. For the box example, participants stated that they wanted "to look at the side around the edge" or "go around to the back side."

Another way disorientation manifests itself is in the commonly observed problem where the user navigates to a view without understanding how they got there (and how to get back) and hence lose their orientation. When the part of the 3D scene visible on the screen is very sparse (e.g. an all black color because the user is looking at the underside of a landscape) the view offers no clues as to what they are looking at or where they are looking from and so, users become disoriented.

To improve orientation awareness, we developed a 3D widget called the ViewCube (Figure 5). The ViewCube is a cube-shaped widget placed in the corner of the screen that serves as a proxy object for the 3D scene being viewed (see Figure 9). No matter where the user navigates or what view they switch to, the ViewCube, shows the orientation of the scene's sides (top, bottom, left, right, back, front) relative to the current viewpoint.

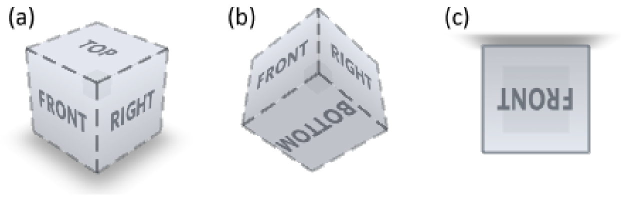

Figure 5: ViewCube showing orientation for: (a) Top-Front-Right view, (b) view from Bottom looking up, and (c) a view when the scene or object is upside down.

Note that the ViewCube provides information beyond that available from the typical ground plane grid where a user may be able to infer the height from which they are looking at the ground plan but not from which side of the grid plane they are looking. The typical 'xyz' triad often shown in 3D scenes can be used to infer the current orientation but the user is still left to infer how these mathematical abstractions relate to their scene and orientation. We believe the ViewCube proxy representation of the scene preserves the user's sense of orientation much better than the grid and triad. Later we describe how the ViewCube serves as a view switching controller. Also see [Khan 2007] for a complete description and evaluation of the ViewCube.

4.4 Enhanced Feedback

We found that some particular types of visual feedback greatly enhance "safety". The first, and perhaps most critical feedback, is to explicitly display the navigation pivot point in the scene.

4.4.1 Pivot Point. Typically in 3D applications many of the navigation operations are based on or affect the pivot point but it is usually not displayed. However, we have observed that this lack of display leaves many users unaware of the existence and effect of the pivot point placement. This in turn can result in some of the most extreme confusion.

For example, the pivot point for the orbit function is generally at the center of the scene or at the center of the currently selected object. When this is the case, turning an object using the orbit command results in the object staying in view and the user having the sensation that they are "orbiting" around the object.

However, there are many common operations or situations that result in the orbit pivot point being located away from the center of the scene or off of the center of the selected object and this can result in great confusion. For example, (see Figure 1i) suppose the pivot point is located at the extreme left side of the screen in the scene. A user clicks and drags to perform an orbit operation which results in an orbit around a very large radius. While this mathematically makes sense, to a new user, unaware of the displaced pivot point (or even the existence of a pivot point) the orbit movement appears to make the user zoom in/out from model. We have observed that users become confused and report things like "oh, I must have picked zoom by accident", "I thought I knew how orbit worked, but now…". The net result is that these types of perceived error states confound learning and create a bad experience for the user.

To combat this, we have made the pivot point a visibly labeled *green-ball* in the scene with visual hints about its depth in the scene relative to the model (i.e. inside, outside or on the surface of a 3D model). Furthermore, the green-ball is wrapped in three orthogonal "belt lines" to make any rotation of the ball perceivable (see Figure 6).

When we introduced this feature, it had an immediate positive effect in all of our user tests. Users immediately understood the role of the pivot point ("seems to revolve around that ball") and confusions between commands as described above were eliminated. We introduced the visible pivot point at the same time as allowing the user to move the pivot point with the center command (described previously). Thus we saw that once users could see and move the pivot point, they could successfully experiment with it, and place it at different spots to see the effect. This resulted in very positive experiences where users quickly learned the basic role of the pivot point and developed strategies for using it effectively.

The green-ball also has a special behavior during navigation commands. For the Zoom and Forward commands, as the user moves forward or zooms in, the pivot point grows/shrinks in size accordingly to give a strong visual clue as to the type of movement that is occurring. It also becomes transparent as it becomes larger to reduce the occlusion of the view. Once the command is completed the green-ball disappears and is resized to its original size on the screen for the next operation.

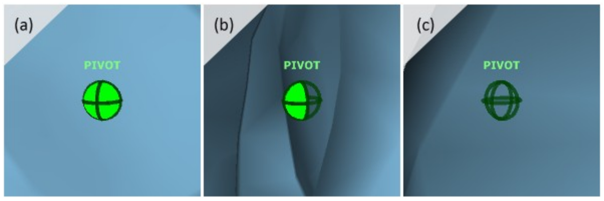

Figure 6: Visual appearance of the green-ball pivot point: (a) in front of a surface –fully visible, (b) on a surface –partially occluded, and (c) behind a surface –fully occluded.

4.4.2 Sliders. Another type of feedback that we believed would reduce confusion and user errors was to provide feedback on dragging directions. Typical 3D applications implement GUI tools such as orbit and pan where dragging in the 3D content window operates the tool. These particular tools use both dimensions of mouse movement and a mapping between how one moves the mouse (up/down, left/right) which is generally known in the case of pan, or quickly learned for orbit.

However, for tools such as the Up/Down tool, we present a graphical slider on mouse-down which simplifies and instructs the user how to operate the tool (see Figure 7). For example, the vertical slider indicates up/down mouse input, the labels denote the range available, the slider signals the current altitude, and the ghosted slider thumb shows the initial altitude for additional orientation.

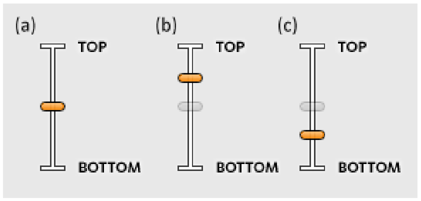

Figure 7: Up/Down tool slider: (a) initially activated, (b) as the cursor is dragged up or (c) down. A ghost slider shows initial position.

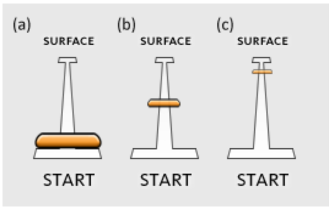

Figure 8: Forward tool "perspective" slider: (a) initial state, (b) 50% and (c) 90% to target surface.

We discovered that more ambiguous tools such as the Forward tool require additional assistance in relating the direction of user input to the 3D scene motion. Thus we introduced a perspective slider (see Figure 8). When the user clicks on the 3D model with the Forward tool active, the green-ball is placed on the target surface. Then the top of the perspective slider is placed on top of the green-ball. Releasing the mouse causes an animation to move towards the green-ball by 50% of this distance. During this navigation animation, the perspective slider remains visible and the slider moves in unison (see Figure 8b). Alternatively, a user can drag the mouse and move forward or backwards by any percentage. Drawing the slider with a perspective foreshortening, and having the thumb size change appropriately, signals depth information to the user and better relates their input actions to the 3D scene motion.

In our subsequent user testing, the benefits of this explicit feedback were immediate and we observed users now had little trouble learning to use these tools.

4.4.3 Labels vs. Icons. Through our studies we have found that new-to-3D users can get distracted when trying to read graphical icons that attempt to identify navigation tools. In fact, we noticed that some users were distracted by attractive toolbar icons and would get into trouble by picking them. For example, in our commercial application testing, one attractive tool icon happened to be "create and manipulate a 3D cutting plane". When selected in the normal exploration of available tools, this tool would really confuse the user and severely contaminate their mental model of how the navigation tools worked.

Our philosophy has been to use text labels to describe the tools instead of graphical icons. Thus, our navigation wheels all use text labels instead of icons to label the individual tool wedges. We believe that their functionality can be more readily associated and consumed by users learning a new system. In contrast, the graphical icons start as, essentially, a foreign language or notation that the user must learn.

In addition, we have found that users rarely monitor the shape and appearance of the cursor. Perhaps this is due to the fact that often times a cursor is not updated, changes to a wait cursor, or the user is focusing on the application data. Also, it is another graphical symbol that must be translated and understood in the mind of the user. Thus, we provide textual cursor labels to reinforce the selection of the current tool and we have found this to be very effective in our user studies.

Lastly, we have found that enhancing feedback can be as simple as providing helper text messages during tool usage to guide the user through the learning stages. For example, if the user clicks down in open space while in the Center tool (which is invalid), we present the helper text "Click or drag on model to set pivot" until the user drags the cursor over a valid part of the 3D model.

4.5 Pre-Canned Navigations

We have found that for novices, since freeform 3D navigation can be difficult and may require the use of several different tools, at times the best way to improve the experience is to reduce the need for freeform navigation. Thus, we offer facilitators such as the ViewCube which allows users to quickly get to standard views. It can be dragged on, or the faces, edges or corners can be clicked on, to easily orient the scene to the corresponding view. Once a component of the ViewCube is selected, the system smoothly animates to the new view to allow the user to stay oriented. When on a face view, additional triangle elements appear to offer pre-canned navigation to adjacent faces. Lastly, a home icon is always available to smoothly transition the user to a safe preset view.

Secondly, we offer ShowMotion [Burtnyk et al. 2006], a system to create a collection of important views (i.e. "shots") and easily attach transition effects (such as fade-to-black) and camera motions (such as pan, zoom, crane-up, etc.). These ShowMotion shots are represented as thumbnail icons and collected in a strip along the bottom of the 3D canvas (Figure 9). They can be named and grouped into sets. This allows users to quickly create and organize a collection of important views. Most importantly, ShowMotion provides users a visual catalog of important views that they can click on and, consequently, move to and see the 3D data presented not as a static image but using motion. Thus, their freeform 3D navigation is replaced with pre-canned high quality camera motion.

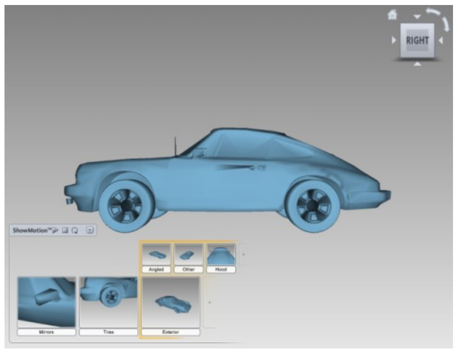

Figure 9: 3D canvas with ShowMotion system shown at the bottom and the ViewCube in the top right corner.

4.6 Constraints (Error Prevention)

A number of constraints have been added to our safe 3D navigation system to prevent errors. As mentioned earlier, providing only a subset of navigation tools which are selected based on user experience and the navigation task can be quite effective. For example, not offering the Orbit tool when new-to-3D users navigate through a building prevents the scene from "flying away".

More conventionally, the navigation tools themselves offer constraints. For example, in the Forward Tool, we prevent the user from easily moving through walls. The Up/Down slider has limits imposed on how high or low the user can move based on the bounding box of the 3D model. Zooming in the View Object wheel has a maximum and minimum object size so that the object is always visible. Lastly, the Pan tool shoots a ray into the scene and adjusts the algorithm to keep the cursor on the initial click down point on the 3D geometry to imitate 2D panning where the cursor and the image being panned are locked together.

The design of the green-ball and its placement algorithms serve as additional error prevention mechanisms. For example, preventing the pivot point from moving off screen helps users understand and predict the interaction behavior. When a user places the pivot point on the 3D model, the system attempts to keep it "sticky" and moves the pivot point along with the 3D model. If that part of the model moves off screen, the pivot point is centered in the canvas but keeping the same depth level. These designs, in particular, making the pivot point visible, allow the user to see the relationship between mouse movement and tool action.

Another technique we have adopted to improve a user's first experience and to improve learning is "cursor wrapping". We observed that many users became confused about how a particular navigation tool works when dragging the cursor and hitting the edge of the screen. The tool would, in their mind, stop working and we would get comments like "this [tool] seems to have quit working…" or "I thought I knew how this worked but now I'm not so sure…" We believe the root cause of the problem is that users are paying attention to the movement of the scene (rightly so) and not paying attention to the location of the cursor once they start dragging. Since the mouse is still free to move even though the cursor is against the display edge, confusion sets in.

Subsequent user testing showed that a very effective fix for this problem was "cursor wrapping" where we essentially remove the typical screen edge constraint. When the cursor hits the edge of the 3D canvas, we warp the cursor over to the opposite side of the 3D canvas and continue movement (see Figure 10). We were surprised how effectively cursor wrapping eliminated this type of confusion and also how users did not even notice or find the warping of the cursor confusing or out of the ordinary –lending more evidence that users are not attending to the cursor once they start moving in the scene.

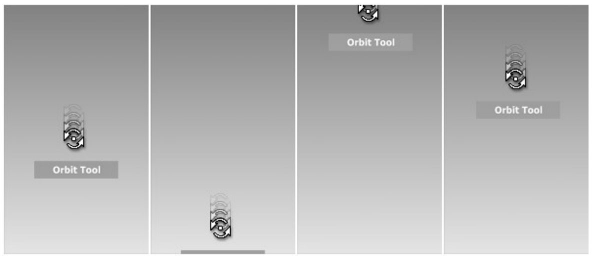

Figure 10: Cursor wrapping from bottom to top when moving the cursor downwards (captured in time from left to right).

4.7 Error Recovery

We observed in our user studies that once new-to-3D users get into trouble while navigating, they try to get out of trouble by navigating. For example, a user accidentally gets to an "empty view" using orbit. They then try to find their model and get back to a view of something by trying to zoom out or by panning. In many cases, if their belief about where they were was wrong, this quite often made the situation even worse.

One might expect that previous experience with other software applications such as word processors or web browsers would lead users to request "recovery" features found in those applications such as "undo" or "home". However, we noted that users did not look for or ask for these types of functions when they were disoriented. This was especially poignant when we added the ViewCube with its "home" function but users who got into very difficult 3D navigation error states still ignored the ViewCube and continued to struggle with navigation tools or gave up on the task altogether. Again, once a user gets in trouble, they attempt to use the same tool (which is immediately at hand) to get out of trouble.

With this in mind we introduced the Rewind tool into the wheels to try to get undo and home type of functionality immediately "at hand". Figure 4 shows the Rewind tool button in every wheel. The Rewind tool works as follows: when a user presses on the Rewind tool in the wheel, a thumbnail strip of the history of a user's navigation (segmented according to drag events) appears at the location of the cursor. The user can then drag to the desired thumbnail. As they drag over each thumbnail the scene does a smooth animated transition to the view associated with the thumbnail. In effect, users are able to "scrub" back and forth in their navigation history. Also, just a single click on the Rewind tool moves the user back a single "step" to the previous view.

The good news in our user testing was that most users found and understood the Rewind tool very quickly when placed in the wheels. The bad news was that despite being made aware of Rewind, most new-to-3D users still used a strategy of trying to manually reverse their navigation when they got into trouble. However, as they grew more comfortable they began to use Rewind. Expert users adopted a "use rewind" strategy almost immediately after learning about the tool.

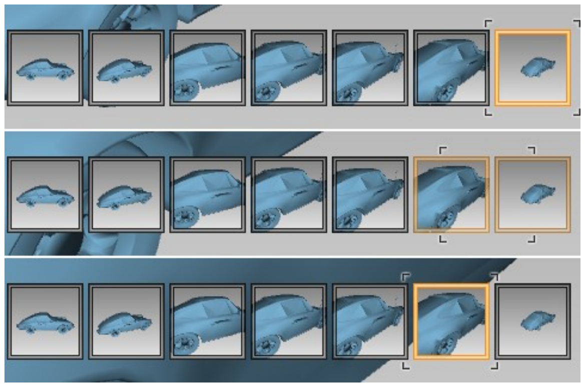

Figure 11: Using the Rewind Tool. Dragging the cursor left, reverses the user's navigation in a smooth continuous fashion. The thumbnail strip provides visual history guidance.

While applications often have a navigation queue to undo/redo camera events, we found that they are not very robust. For example, they are often represented as two small icons somewhere on the bezel of the view which offer only discrete navigation event traversal. Often the transaction cost of accessing the tool and the user not knowing how they segmented their navigation make this facility rarely used. In contrast, we promote the Rewind tool as a first class tool, have it readily available on the navigation wheel, allow continuous and discrete history traversal and use thumbnails as visual history guidance.

Finally, we note that new-to-3D users often do not immediately use Rewind to get out of trouble and that this indicates that Rewind cannot be used as the cure-all for navigation problems. Thus our emphasis has been on tools that prevent errors and help users to learn how the tools work as quickly as possible.

5 Usability

The source of our observations and conclusions made above are from the research and development of these techniques, which took place over a two year period, in the context of a product development process. The goal of our work was to add these navigation controls into the next product release of six commercial 3D CAD applications. This would make it easier for new-to-3D users to learn 3D navigation and for experienced 3D users to have better navigations tools. Additionally, these tools would be the same across applications so an experienced user could easily transfer their 3D navigation skills from one application to another.

We conducted observational usability studies on approximately 55 individuals trying out various design variations of our tools as we refined the design. The type of user varied from new-to-3D users to very experienced (professional) 3D users. Typically the new-to-3D users had some computer experience with office automation software. The experienced 3D users were a range of CAD professionals from mechanical engineers to architects.

We asked users to "think aloud" and typically spent 45 minutes asking them to attempt particular navigation tasks or simply to explore and navigate around different types of models. Typically a group of four interaction designers watched the session via video connection and analyzed the behaviors and results.

The previous sections have described the major specific results we observed. In general, over the course of testing and modifying for new-to-3D users, we were able to move from very error prone and unsuccessful navigation sessions to sessions where the users were successfully completing the tasks without requiring tester intervention to recover from errors. Experienced 3D users, typically in our early user tests, reported they would still prefer the navigation tools of their favorite or working 3D application (e.g. 3D Max, Inventor, etc.). By the end of testing, experienced users typically reported that they "definitely would learn the tools" as they seemed "really promising". Moreover, some particular tools like Rewind and Walk were highlighted as "got to have!" A small number of expert users (1 in 10 in our last series of tests) quickly understood and learned the tools but struggled to overcome their habituated behaviors of traditional tool bar tools —they involuntarily tried to pick a tool off of the wheel and then move away from the wheel to apply it. This indicates that our tools may be easy to learn but still, for some very skilled users, some habituated motor actions will have to be unlearned.

Our techniques were also reviewed and tried by at least 12 product designers for the target products. The designers are typically very experienced and savvy 3D users. In general feedback was very positive, and, like our tested expert 3D users, some functions like Rewind and Up/Down were very welcome. We were also surprised that some of these designers noticed that we were constraining the pivot point in some way. For example, one designer reported to us: "you are doing something very clever with pivot point so I don't get into trouble". Finally, this group of product designers was extremely enthusiastic about our power-user "mini-wheels" as they had already become experienced wheel users and welcomed the miniaturization.

6 Conclusions

Our goal was to have users who are "new to 3D" experience a productive and pleasurable first impression when navigating in 3D. The initial user studies showed that new-to-3D users can get into navigation trouble very quickly (within 30 seconds) and once in trouble, have a hard time getting back to a known, comfortable viewing state. By applying the seven properties outlined above, we have shown that a significant advance can be made in offering a "safe 3D navigation" experience.

Clearly there are some limitations with our approach. For example, users could bypass the First Contact dialog, pick the wrong wheel for the wrong navigation task, get confused over the tracking menu paradigm, and be annoyed with the tool subsets. Moreover, our design has small learning discontinuities when a new-to-3D user transitions to an intermediate 3D user by changing some of the tools and tool behavior. In practice, we have found this disruption to be very minimal.

Future research would focus on improving the navigation tools to detect and react to the 3D geometry. While we have some of this "smart navigation" working in our lab, it utilizes the GPU which is often different or not available on the wide range of computers on which commercial applications must run.

While some of our seven properties may sound obvious, it is how intensely and richly each one is supported that constitutes a better and safer navigation system. For 3D navigation in CAD and similar application domains to be more widely adopted, as an industry we must evolve the tools to be more approachable and less brittle to ease 2D users into the joys of 3D.

Acknowledgments

We would like to thank the product teams at Autodesk involved in this project, especially Thomas White, Michael Glueck, Ruslana Goncharenko, Ryan Schmidt, Lynn Miller, Desiree Sy, John Schrag, Tekno Tandean, Sean Zhang, Jose Madeira Garcia, Elena Fadeeva, Charles Haughey, Paul Hanau, Tom Vollaro, Darin Hughes, Milan Sreckovic, Pam Polizzi, Mark Petit, and Larry Phillps.

References

BALAKRISHNAN, R. AND KURTENBACH, G. 1999. Exploring bimanual camera control and object manipulation in 3D graphics interfaces. In *ACM CHI*. pp. 56-63.

BARES, W. AND KIM, B. 2001. Generating Virtual Camera Compositions, In *Proceedings of ACM IUI'01*, pp. 9-12.

BARES, W. AND LESTER, J.C. 1999. Intelligent Multi-Shot Visualization Interfaces for Dynamic 3D Worlds. *Proceedings of ACM IUI '99*, pp. 119-126.

BOWMAN, D., JOHNSON, D. AND HODGES, L. 1999. Testbed environment of virtual environment interaction. In *Proceedings of ACM VRST*. pp. 26-33.

BOWMAN, D., KOLLER, D. AND HODGES, F.H. 1997. Travel in immersive virtual environments. *IEEE VRAIS'97 Virtual Reality Annual International Symposium*. pp. 45-52.

BURTNYK, N., KHAN, A., FITZMAURICE, G., AND KURTENBACH, G. 2006. ShowMotion: Camera Motion based 3D Design Review, In *Proceedings of ACM I3D 2006*, pp. 167-174.

BURTNYK, N., KHAN, A., FITZMAURICE, G., BALAKRISHNAN, R. AND KURTENBACH, G. 2002. StyleCam: Interactive Stylized 3D Navigation using integrated Spatial and Temporal Controls, In *Proceedings of ACM UIST 2002*, pp. 101-110.

CHAPMAN, D. AND WARE, C. 1992. Manipulating the future: predictor based feedback for velocity control in virtual environment navigation. *ACM Symposium on Interactive 3D Graphics*. 63-66.

CHRISTIANSON, D.B., ANDERSON, S.E., HE, L.-W., WELD, D.S., COHEN, M.F., AND SALESIN, D.H. 1996. Declarative camera control for automatic cinematography. *Proceedings of AAAI '96* (Portland, OR), pp. 148-155.

DRUCKER, S. AND ZELTZER, D. 1995. CamDroid: A System for Implementing Intelligent Camera Control. In *ACM I3D*, pp. 139-144.

FITZMAURICE, G., KHAN, A., PIEKE, R., BUXTON, B. AND KURTENBACH, G. 2003. Tracking menus. *ACM UIST 2003*. pp.71-79.

FITZMAURICE, G., MATEJKA, J., KHAN, A., GLUECK, M., KURTENBACH, G. 2007. PieCursor: Merging Pointing and Command Selection for Rapid In-place Tool Switching. To appear in *ACM CHI 2008*.

GALYEAN, T. 1995. Guided navigation of virtual environments. *ACM Symposium on Interactive 3D Graphics*. pp.103-104.

GLEICHER, M. AND WITKIN, A. 1992. Through-the-lens camera control. *ACM SIGGRAPH 92*. pp. 331-340.

HANSON, A. AND WERNET, E. 1997. Constrained 3D navigation with 2D controllers. *IEEE Visulization*. pp. 175-182.

HE, L., COHEN, M. AND SALESIN, D. 1996. The virtual cinematographer: a paradigm for automatic real-time camera control and directing. *ACM SIGGRAPH 96*. pp. 217-224.

IGARASHI, T., KADOBAYASHI, R., MASE, K. AND TANAKA, H. 1998. Path drawing for 3D walkthrough. *ACM UIST*. pp. 173-174.

KHAN, A. KOMALO, B., STAM, J., FITZMAURICE, G. AND KURTENBACH, G. 2005. HoverCam: interactive 3D navigation for proximal object inspection, In *Proceedings of ACM I3D*, pp. 73-80.

KHAN, A., MORDATCH, I., FITZMAURICE, G., MATEJKA, J., KURTENBACH, G. 2007. ViewCube: A 3D Orientation Indicator and Controller. In *Proceedings of ACM I3D 2007*.

MACKINLAY, J., CARD, S. AND ROBERTSON, G. 1990. Rapid controlled movement through a virtual 3D workspace. *ACM SIGGRAPH 90*. pp. 171-176.

NIEUWENHUISEN, D., KAMPHUIS, A. AND OVERMARS, M. H., 2007 High quality navigation in computer games, *Science of Computer Programming*, Vol. 67, 1, Elsevier North-Holland, Inc, pp. 91-104.

PHILLIPS, C.B., BADLER, N.I., AND GRANIERI, J. 1992. Automatic Viewing Control for 3D Direct Manipulation. In *ACM I3D*, pp. 71-74.

PIERCE, J. S., CONWAY, M., VAN DANTZICH, M., AND ROBERTSON, G. 1999. Toolspaces and glances: storing, accessing, and retrieving objects in 3D desktop applications. In *ACM I3D*, pp. 163-168.

STEED, A. 1997. Efficient navigation around complex virtual environments. *ACM VRST*. pp. 173-180.

STOAKLEY, R., CONWAY, M. AND PAUSCH, R. 1995. Virtual reality on a WIM: Interactive worlds in miniature. *ACM CHI*. pp. 265-272.

TAN, D., ROBERTSON, G. AND CZERWINSKI, M. 2001. Exploring 3D navigation: combining speed-coupled flying with orbiting. *ACM CHI*. pp. 418-425.

TSANG, M., FITZMAURICE, G.W., KURTENBACH, G., KHAN, A. AND BUXTON, B. 2002. Boom Chameleon: Simultaneous capture of 3D viewpoint, voice and gesture annotations on a spatially-aware display. *ACM UIST*. pp. 111-120.

WARE, C. AND FLEET, D. 1997. Context sensitive flying interface. *ACM Symposium on Interactive 3D Graphics*. pp. 127-130.

WARE, C. AND OSBORNE, S. 1990. Exploration and virtual camera control in virtual three dimensional environments. *ACM I3D*. pp. 175-183.

ZELEZNIK, R. AND FORSBERG, A. 1999. UniCam - 2D Gestural Camera Controls for 3D Environments. *ACM I3D*. 169-173.

ZELEZNIK, R., FORSBERG, A. AND STRAUSS, P. 1997. Two pointer input for 3D interaction. *ACM I3D*. PP. 115-120.