Design Space and Viewpoint Visualizations for Single-Click 3D Navigation

Benjamin Nuernberger* Steffen Gauglitz* Tobias Höllerer* Matthew Turk*
Department of Computer Science, University of California, Santa Barbara

Abstract

Single-click 3D navigation allows users to navigate a virtual space with a single click of the mouse or tap of the finger. As such, it is appealing for its simplicity, a trait especially important for mass consumer applications. In addition, single-click navigation is also suitable for incomplete or sparse models, and is thus commonly used for navigating partial 3D reconstructions based on panoramas or structure from motion. In this paper, we give a description of the design space for single-click 3D navigation, showing that the crucial design choices typically include a viewpoint selection algorithm and a viewpoint visualization. We further present a brief user study on viewpoint visualizations for single-click navigation. Our results indicate that current systems lack adequate visualizations for navigating complex environments and that the viewpoint selection algorithm is perhaps most important, requiring more investigation.

CR Categories: I.3.6 [Computing Methodologies]: Computer Graphics—Methodology and Techniques H.5.2 [Information Systems]: Information Interfaces and Presentation—User Interfaces;

Keywords: 3D navigation, user interfaces, taxonomy, visualizations

1 Introduction

Designing 2D user interfaces for navigating 3D virtual spaces is challenging since the system has to map the low-dimensional input into a space with (at least) six degrees of freedom (DoF). To accomplish this mapping, one approach is to capture a *sequence* of 2-DoF inputs (e.g., [Zeleznik and Forsberg 1999; Hachet et al. 2008]). The downside with this approach, however, is the additional complexity needed to process such inputs and the increased need for training users on how to input such sequences. A different approach is to instead keep the user input as simple as possible and to design the system to take on more responsibility. In this paper, we investigate the simplest possible 2D input, namely, navigation using a single click or tap.

Here, a single click of the mouse or tap of the finger takes the user to a new viewpoint; thus the new viewpoint has to be derived from an input with only 2-DoF. Because of this, single-click navigation is ideally suited for *constrained* navigation, including the case when only a sparse, discrete set of viewpoints is available. In this setting, single-click navigation has been used in popular tools such as Google Street View and Microsoft's PhotoSynth, both of which use partial 3D reconstructions of the world and are designed to explore a physical scene remotely.

Furthermore, single-click navigation is the simplest of cursor-based interfaces, making it appealing even if the sparsity of the data is not a primary concern. Simplicity is especially important in mass consumer applications in which no training time can be assumed; each interface feature needs to be discoverable automatically by "playing with" the interface and to be self-explanatory to the greatest extent possible.

Our main contributions in this paper are (1) a description of the design space for single-click 3D navigation, showing that the crucial design choices typically consist of a viewpoint selection algorithm and a viewpoint visualization; and (2) an exploratory user study of hover-based viewpoint visualizations for single-click navigation, which has led to insights for designing future interfaces for single-click 3D navigation.

2 Related Work

2.1 Single-click navigation in the context of 3D navigation interfaces

Jankowski and Hachet [2013] provide a in-depth survey of 3D interaction techniques including 3D navigation. Generally, 3D navigation can be categorized into three types of tasks: exploration, search, and inspection/maneuvering [Tan et al. 2001; Bowman et al. 2005; Mackinlay et al. 1990]. We will use this categorization below in our description of the single-click design space (cf. Section 3.3 and Fig. 2). Bowman et al. [1997] further describe the mechanics of 3D navigation as having three main categories: direction/target selection, velocity/acceleration selection, and input conditions. In this context, single-click navigation is mainly concerned with the first category, namely, direction/target (i.e., viewpoint) selection.

Single-click 3D navigation can be used as a "point of interest" (POI) navigation technique as described by Mackinlay et al. [1990] and Hachet et al. [2008]. Mackinlay et al. [1990] described a POI technique for specifying a target destination via the mouse cursor and forward/backward logarithmic motion with respect to the object via the keyboard. In comparison, in single-click navigation, a single click selects the POI *and* triggers the movement towards the POI. Thus, in this context, single-click avoids multiple-step interfaces, which, although more powerful (e.g., [Hachet et al. 2008]), are typically less self-explanatory and may require more training.

*{bnuernberger, sgauglitz, holl, mturk}@cs.ucsb.edu

2.2 Recent interfaces using single-click navigation

Two popular publicly available systems using single-click navigation, specifically for constrained navigation in partially reconstructed worlds, are Google Street View (cf. [Vincent 2007]) and Microsoft's PhotoSynth. In the current version, Google Street View uses a single-click interface with two different hover-based visualizations (cf. Section 3.4): a circular visualization along with an 'X' visualization on the street specifies where (if clicked) the next viewpoint origin will be with respect to the street (Fig. 1(a)), while a quadrilateral visualization on building façades specifies a POI which the next viewpoint should show.

PhotoSynth, based on the work of Snavely et al. [2006; 2008], offers several interfaces, including a single-click interface with hover-based "quad" visualizations (see Fig. 1(b) and cf. Section 3.4) indicating other viewpoints onto the scene. A single click onto a quad takes the user to that viewpoint. (The recently added feature of following specific paths, including "panoramas" and "walks" through the scene, does not use single-click navigation and only allows one-dimensional movement along a precalculated path.) The original Photo Tourism work by Snavely et al. [2006] offers an additional single-click interface in which all available camera locations are displayed and the user can click onto each of them to assume its viewpoint.

Recently, Brivio et al. [2013] introduced a system similar in nature to Photo Tourism called PhotoCloud. PhotoCloud has a complex interface which includes a single-click interface with semi-transparent rectangular visualizations dubbed "framelets" (see Fig. 1(d)) that indicate each available camera's frustum at a certain depth. Users click on a framelet to move to that camera. To reduce visual clutter and viewpoint ambiguity, each framelet's transparency is dependent on its orientation relative to the current viewpoint. One drawback they describe is that appropriate values for the distance at which each framelet is drawn is dependent on the scene type, with values ranging across one order of magnitude. They also include a "focus-and-context" thumbnail bar (see Fig. 1(d)) to support browsing of photos.

In the work by Tatzgern et al. [2014], the presented system selects a viewpoint and a "camera manipulator" based on knowledge of the user's task and the scene semantics. They include icon viewpoint visualizations that indicate the type of viewpoint that will be shown if the user taps onto the screen (see Fig. 1(c)).

The aforementioned works show that using single-click 3D navigation is a popular choice in particular for navigating partially reconstructed scenes. However, to our knowledge, no work exists that provides a taxonomy of this class of interfaces or describes the design choices in a general framework. We attempt to do so in this paper. In Section 3, we describe the design space for single-click 3D navigation in detail. In Section 4, we present an exploratory user study into hover-based visualizations for single-click 3D navigation.

3 Design Space for Single-Click Navigation

We define single-click 3D navigation as one that allows the user to assume/travel to a new viewpoint in a (3D) world based upon a single (2D) click or tap. The main components are: the set of available viewpoints (Section 3.1); choice of clickable regions (Section 3.2); the viewpoint selection algorithm (Section 3.3); and the viewpoint visualization (Section 3.4).

In the following sections, we give an overview of each of these components. Fig. 2 gives a taxonomy of the components of single-click 3D navigation, and Fig. 3 gives a taxonomy for the viewpoint selection algorithm employed by single-click 3D navigation. In each of these figures, we also list relevant examples of interfaces using single-click 3D navigation.

3.1 Set of Available Viewpoints

Single-click 3D navigation can be used with a discrete or continuous set of viewpoints. The former case is commonly found when using panorama or structure from motion scene reconstruction algorithms (e.g., [Vincent 2007; Vergauwen and Van Gool 2006; Klein and Murray 2007; Newcombe and Davison 2010; Agarwal et al. 2011]). Here, single-click navigation is used to select one viewpoint from among a discrete set of viewpoints.

In the latter case of continuous viewpoints, a single-click navigation interface must determine a full viewpoint from a continuous set of viewpoints instead of choosing among a finite set of viewpoints. A recent example of this is [Tatzgern et al. 2014] where the system automatically determines virtual viewpoints from a continuous space of viewpoints. Ideas from perceptual models and canonical views [Secord et al. 2011] may be especially helpful in this scenario in order to select a "good" viewpoint.

The choice of using a discrete or continuous set of viewpoints largely depends on the available data and the task at hand (cf. Section 3.3).

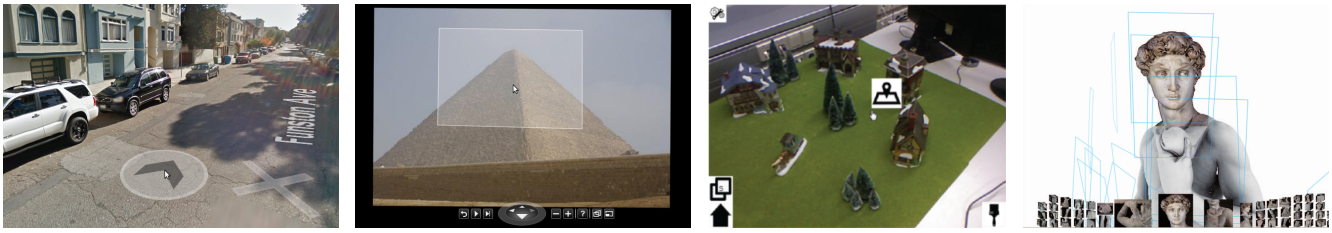
3.2 Clickable Regions

We distinguish between three mutually exclusive alternatives for clickable regions. Clickable regions refers to the fact that the navigation can be triggered by clicks into different areas: when clicking outside the scene, when clicking "clickable anchors," or when clicking directly into the scene.

First, users can click somewhere *outside* the scene, e.g., on a thumbnail list of viewpoints. This may be preferable for example to get to a viewpoint far outside the current field of view, or to select among viewpoints that have very specific textual descriptions or semantic meanings. In this case, the object that is clicked upon (e.g., a thumbnail) uniquely identifies the viewpoint to which to move. Thus, no further inference (i.e., a viewpoint selection algorithm) is needed.

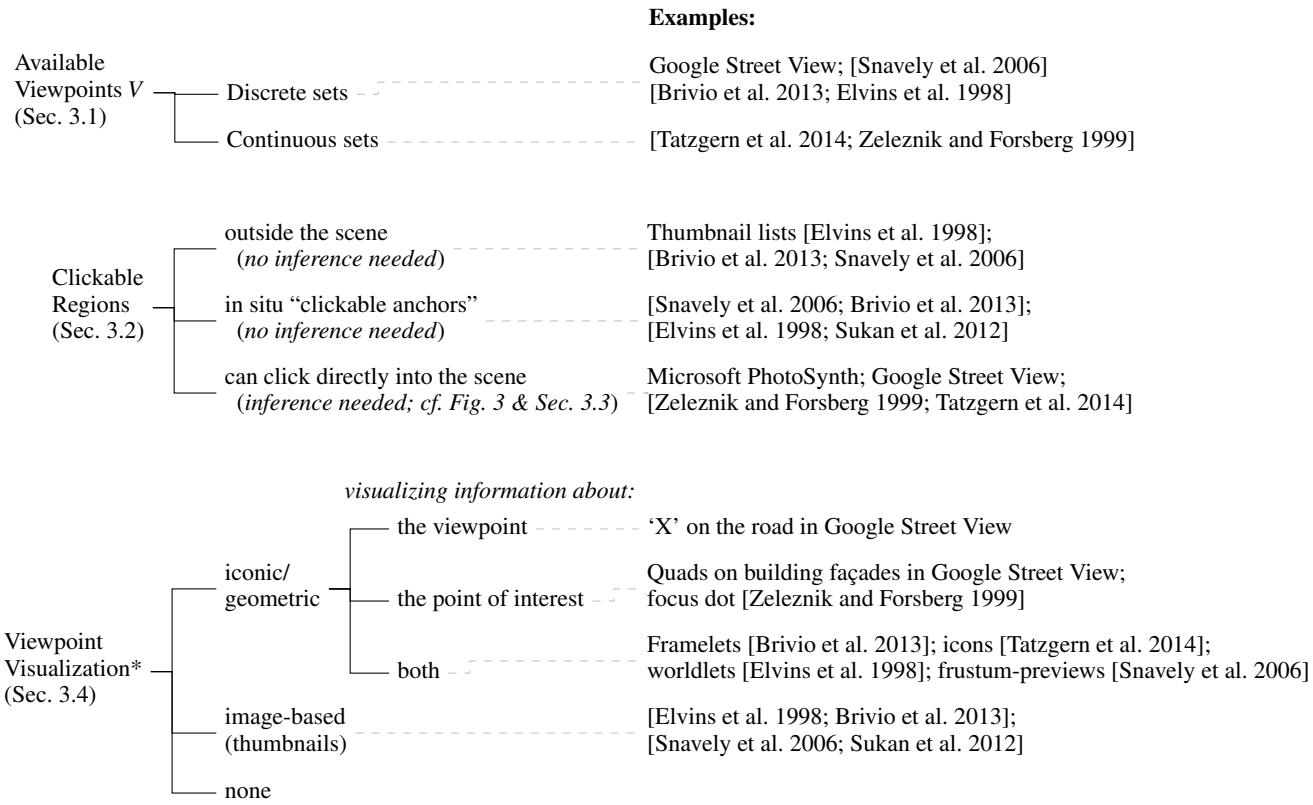
Second, users can click on static viewpoint visualizations that appear *inside* the scene; we may call these "clickable viewpoint proxies" or "clickable anchors" [Elvins et al. 1998]. Here, a clickable anchor represents a viewpoint that the user can move to when it is clicked upon. As with clickable regions outside the scene, the object that is clicked upon uniquely identifies the viewpoint and no further inference is needed. This type may be preferable over clicking outside the scene because the user directly interacts within the space of the scene and the anchor's position and/or orientation can communicate information about the viewpoint (cf. Section 3.4). However, static viewpoint visualizations shown directly in the scene may create visual clutter (cf. [Brivio et al. 2013]) and selecting a specific viewpoint may become difficult when many occupy a small area.

Third, users can click directly into the scene and the viewpoint selection algorithm (Section 3.3) chooses the viewpoint to which to move. Thus, the system must attempt to infer the user's navigation intent from the current cursor location (x,y). With interfaces that support hovering (i.e., moving a cursor without clicking; typically, mouse-based interfaces do, while touchscreens do not), the viewpoint selection algorithm can be run based on the current hover point, and a viewpoint visualization (Section 3.4) can be then used



(a) Google Street View (b) Microsoft PhotoSynth (c) [Tatzgern et al. 2014] (d) [Brivio et al. 2013]

Figure 1: *Viewpoint visualizations in existing works using single-click navigation. The visualizations in (a)–(c) follow the hovering mouse cursor, conveying information about next viewpoint (a,b) and/or the chosen point of interest (c). In (d), the blue "framelets" serve as clickable anchors; in addition, the user can chose a view from the thumbnail list on the bottom.*



**as static "clickable anchors" or as previews while the cursor is hovering*

Figure 2: *Taxonomy of single-click 3D navigation.*

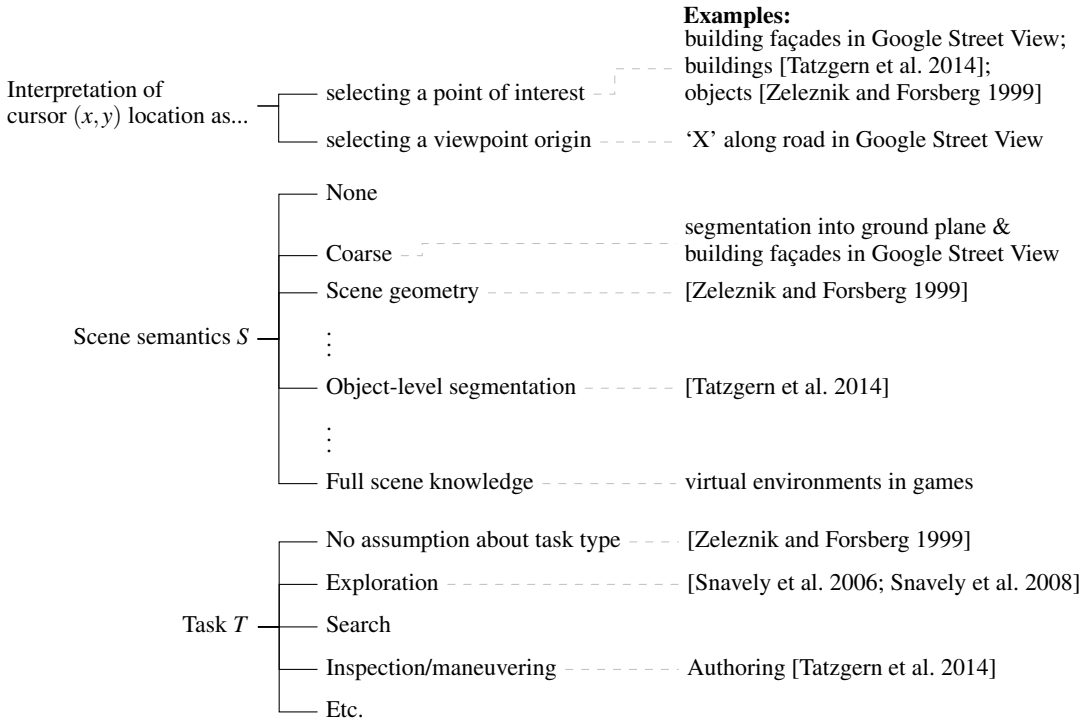


Figure 3: *Taxonomy of the viewpoint selection algorithm for single-click 3D navigation (cf. Section 3.3).*

to communicate this inference back to the user.[1] Thus, as the user moves the cursor without clicking, the visualization supplies visual feedback to the user so that the user knows where to click to initiate navigation. The advantage of this type of clickable region is that the user can click directly (anywhere) into the scene (as compared to clicking only onto clickable anchors), and no multitude of clickable anchors clutters the view or has to be processed mentally. The challenge is to select an appropriate viewpoint based on the cursor location, which is the task of the viewpoint selection algorithm (Section 3.3), and if desired, design an appropriate hover-based visualization.

3.3 Viewpoint Selection Algorithm

The viewpoint selection algorithm infers the user's desired viewpoint from the 2-DoF input of the current cursor location, the scene semantics, and the task scenario. Fig. 3 presents a taxonomy of its components.

A viewpoint selection algorithm $\mathbf{A}$ takes as input the current mouse coordinates (x,y) and determines the next viewpoint $v_i \in V$ that will be chosen if the user clicks the mouse, where V is the set of available viewpoints. Scene semantics (including geometry) S and task knowledge T can also be incorporated. Hence, we have:

$$\mathbf{A}(V,(x,y),S,T) \rightarrow v_i \in V \tag{1}$$

3.3.1 Interpretation of cursor (x,y) location

There are two possible assumptions for interpreting the cursor location (x,y). First, the system may assume that the user clicks on a location because he/she wants to go to this point — here, the end result would be to move the viewpoint origin to the moused-over location or close to it, in the case of discrete sets of viewpoints or when applying certain metaphors such as "walking." This is commonly used in 3D games and applications such as Google Street View when the user clicks on the street. We denote this as "selecting a viewpoint origin" in Fig. 3. Note that for this case, the viewpoint selection algorithm is relatively simple — it has to find the "closest" viewpoint to the selected 3D point that corresponds to the (x,y) cursor location.

Second, the system may assume that the user clicks on a location because he/she wants to see the selected object/area up close — here, the end result would be to move to a particular viewpoint that can see the user-specified point of interest. This is commonly used in adventure games where users can click on items of interest and in applications such as Google Street View when the user clicks on buildings. We denote this as "selecting a point of interest" in Fig. 3. Note that for this case, the viewpoint selection algorithm is to choose a v_i from among all $v \in V$ such that v "best sees" the point of interest; how one interprets "best" is what makes this challenging. Next, we discuss factors by which the viewpoint selection algorithm can be guided to choose v_i.

3.3.2 Scene Semantics S and Task Scenario T

The viewpoint selection algorithm may be guided by the semantic knowledge about the scene (including its geometry) S, and/or the task T for which the system was designed. Tatzgern et al. [2014] mention the former as "Scene Analysis" and the latter as "Task Knowledge." In their work, they segmented the scene into a ground plane and objects, and assumed an authoring task scenario. Based on these assumptions, they gave region top-view viewpoints whenever the user clicked over the ground plane (Fig. 1(c)), object top-view viewpoints whenever the user clicked on the top of an object, and close-up viewpoints whenever the user clicked on the side of an object.

[1] This is commonly used in 2D & 3D adventure games where certain "actionable" items display a visualization when the cursor hovers over it (e.g., informing the user that a door can be clicked upon to open it).

3.4 Viewpoint Visualizations

Viewpoint visualizations can be used as static clickable regions (first and second case in Section 3.2) or as previews while the cursor is hovering (particularly for the third case in Section 3.2). In either case, they are to convey information about the potential future viewpoint. They are typically either iconic/geometric, visualizing information about the viewpoint or the point of interest, or image-based, directly displaying the scene as seen from that viewpoint (e.g., as thumbnails).

3.4.1 Iconic or geometric visualizations

Iconic or geometric visualizations typically appear within the scene, positioned and/or oriented such that their position and/or orientation provides information about the viewpoints that they represent, the point of interest, or both.

Information about the viewpoint that can be visualized includes the viewpoint location (e.g., the 'X' along the road in recent versions of Google Street View as shown in Fig. 1(a)) and the field of view (e.g., the quads in PhotoSynth as shown in Fig. 1(b)). The wireframe frustums in Photo tourism visualizes both.

Further, visualizations can convey information about a point of interest. For example, the quadrilateral on building façades in Google Street View indicates the distance to the detected surface and its orientation.

Finally, some visualizations incorporate information about both the viewpoint and a point of interest simultaneously. An example for this is the "framelets" in PhotoCloud [Brivio et al. 2013] (Fig. 1(d)), where the size of the framelet indicates the field of view and its position is relative to a point of interest. Another example is the visual icons in Tatzgern et al. [2014] (Fig. 1(c)) that appear whenever the cursor hovers over a point of interest or region; here, the icon conveys information about the type of viewpoint that will be shown (region top view, close-up view, or top view). Instead of an icon next to the cursor, the cursor itself can change its appearance to indicate similar categorial information and thus be used as a minimal visualization.

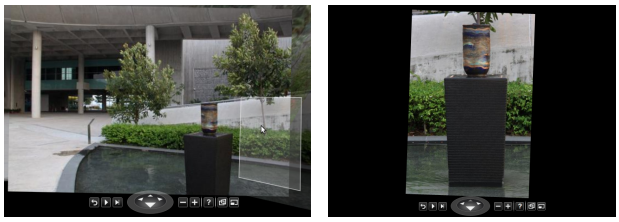
The advantage of iconic or geometric visualizations is their simplicity, ability to be customized, and potential for indicating where and with what orientation in the 3D space the viewpoint is located. The downside is that it is often hard to communicate what exactly the viewpoint will show; this is where image-based visualizations help out.

3.4.2 Image-based Visualizations

Image-based visualizations show an image of the viewpoint in some format, such as a list of thumbnails in a side pane or in situ preview images (i.e., appearing inside the scene). For example, PhotoCloud [Brivio et al. 2013] shows a bar at the bottom of the screen with thumbnails in addition to the in situ framelets (Fig. 1(d)); the systems by Snavely et al. [2006] and Sukan et al. [2012] use in situ preview images.

The advantage of image-based visualizations is that users know exactly what their viewpoint will show if they choose to navigate there. However, image-based visualizations may need more space to be useful and do not directly convey spatial information regarding the position and orientation of the viewpoint.

Iconic/geometric visualizations and image-based visualizations can be combined, by showing a thumbnail next to [Sukan et al. 2012] or embedded into [Snavely et al. 2006] an iconic visualization.



(a) before clicking (b) after clicking

Figure 4: *Challenging situation for the viewpoint visualization in Microsoft PhotoSynth. It is hard to tell from the "quad" in (a) that the next viewpoint will be (b).*

3.4.3 No Visualization

Lastly, some interfaces may lack viewpoint visualizations entirely. In some 3D games, users can click directly into the scene without any viewpoint visualization in order to move their viewpoint accordingly. In this setting, no visual feedback is provided to the user before he/she is taken to a new viewpoint. This option can be used in particular if the mapping between click location and target viewpoint is assumed to be self-evident; this may be the case if the set of viewpoints V is small and/or if semantic scene knowledge S guides the viewpoint selection algorithm (cf. Section 3.3).

An advantage of using no visualization is that no visual information has to be mentally processed or distracts from the scene. The obvious downside is that no visual feedback is provided to the users to indicate where they may end up if they click at the current cursor location.

4 Preliminary Investigation of Viewpoint Visualizations

After observing popular systems such as Google Street View and Microsoft PhotoSynth use single-click navigation with different viewpoint visualizations, we became interested in how users perceive the different visualizations when navigating a virtual scene. In highly constrained settings, such as being confined to streets in Google Street View, existing visualizations arguably work well enough. However, for arbitrarily constrained environments, such as found in Microsoft PhotoSynth, existing visualizations are sometimes challenging to understand (Fig. 4 gives an example). Hence, we designed an exploratory user study in a semi-constrained environment to investigate how different viewpoint visualizations are perceived by users and in how far they influence the task completion time of an elementary and generic primed search/viewpoint manipulation task.

4.1 Study Design

We chose to use a virtual environment with a discrete set of viewpoints (cf. Section 3.1) so that we were able to control the position and orientation of the viewpoints. For clickable regions (cf. Section 3.2), we allowed the user to click directly into the scene. Initially when designing the study, we tried using "clickable anchors" with visualizations similar to that of PhotoCloud's framelets [Brivio et al. 2013] (Fig. 1(d)). However, this was quickly determined to be too cluttered in our environment and we thus focused on letting the user click directly into the scene. The viewpoint visualizations we chose to examine were motivated by the geometric visualizations used by existing systems (cf. Section 3.4 and Fig. 1). Finally, since we focused our study on the visualizations, we chose to use the fol-

Figure 5: *Overhead view of the kitchen scene with viewpoints from environments 2 & 3 (shown as blue) and environment 1 (shown as yellow).*

lowing simple viewpoint selection algorithm (cf. Section 3.3): We first filter out all the viewpoints that cannot see the 3D world point P under the cursor (i.e., outside the field of view or occluded; the latter is implemented by caching the depth maps of each frame and comparing them to the projection of P into each camera). From the remaining viewpoints, we then choose the one whose optical axis is closest to P.

Environment. We used three different environments: All used the same backdrop scene (the kitchen scene depicted in Fig. 5), but the distribution of the available camera viewpoints and the objects of interest were varied. In the first environment, relatively few objects and 68 camera viewpoints were spread out in the kitchen. In the second environment, 93 camera viewpoints were densely clustered around the middle "island" area, with a moderate number of objects on the counter. In the third environment, the same 93 viewpoints were used, but additional objects were placed on the counter, leading to a severely cluttered scene and the need to navigate around occluding objects for most tasks.

Task. Each task consisted of finding a specified piece of information (e.g., the brand name of an object, the time on a clock, nutrition information on a food container, etc.). Note that this task bears similarity with the one used by Brivio et al. [2013]. We intentionally removed the "search" aspect from the task by telling the user *where* the respective object was located, thus, the only task was to navigate towards it.

Participants. In total, 18 users participated in our study. Users were compensated for their time commitment of about one hour with a nominal amount. We tried out various conditions on the first 8 users, including different types of tasks and attempting to use "clickable anchors" as previously mentioned. We also refined study parameters and procedure.

The main part of the study had 10 participants, 5 female and 5 male, 18 to 30 years old (average 20.9). All had 10+ years of experience speaking English and passed a standard colorblindness test. 5 wore corrective lenses; 2 had never previously used 3D software.

Conditions. We compared five different viewpoint visualization conditions. The first four conditions were geometric viewpoint visualizations (cf. Section 3.4 and Fig. 2) that contained varying levels of information about the viewpoint and/or the moused-over point of interest. The fifth condition was no visualization. For easy reference and identification, we colored them in soft pastel colors; however, we explicitly told the participants to use the colors for this purpose only and to disregard them in their ratings:

- **yellow** (Fig. 6(a)): a rectangle which indicates the camera's field of view at the depth of the moused-over object. This visualization is similar to PhotoSynth's "quads."
- **red** (Fig. 6(b)): like yellow, but additionally indicating the camera's origin, showing the full camera frustum up to the object of interest as a wireframe pyramid.
- **blue** (Fig. 6(c)): a rectangle of fixed world size centered dynamically around the mouse cursor and parallel to the image plane. This is similar to the visualization on building façades in Google Street View, except that the latter is oriented to match the object's surface rather than the image plane.
- **green** (Fig. 6(d)): like blue, but additionally shows the camera's origin, thus showing a wedge-shaped part of the camera frustum.
- **no visualization**.

Yellow and red provide more information about the viewpoint, whereas blue and green provide more information about the moused-over point of interest. All four visualizations provide information about the camera's orientation, and all but blue provide some information about the camera's distance. In all four visualizations, we darkened their color when their view's camera direction was opposite to the current camera view; this helps in overcoming ambiguity problems such as found with the Necker cube.

Users were able to rotate (look around) via dragging, but the only way to change the camera origin was to click into the scene in accordance with the visualization. Each change of camera origin was animated by interpolating between the two poses over one second.

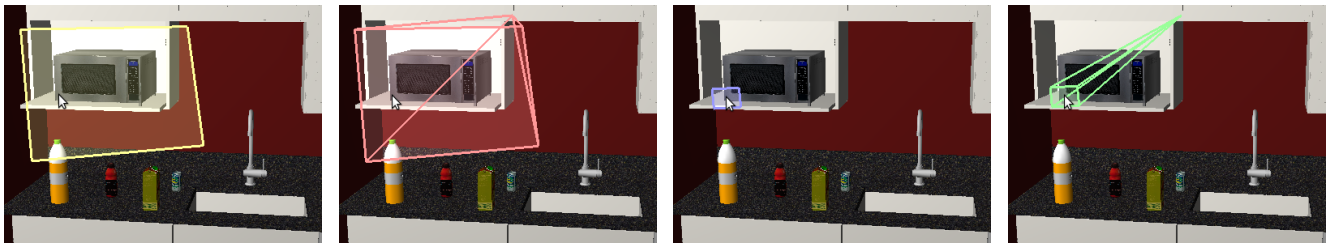
Experimental Design. We used a within-subjects design with a single dependent variable (task completion time). The order of the conditions was fully balanced using a 5×5 Latin square. As we did not seek to compare the environments to each other, we treated the environments as effectively separate experiments and did not randomize their order.

Procedure. After testing for colorblindness and administering a pre-study questionnaire, the experimenter explained the study's scenario and setup. Then, the user was given two training tasks to get used to the first visualization, after which the first block of five tasks was executed. For each task, the experimenter first stated the piece of information to be obtained from the scene. The user pressed a key as soon as he/she was ready, then navigated towards the sought-after object, read the information aloud, and pressed another key to indicate completion of the task. After the five tasks, the setup was switched to the next visualization, and the user was given two training tasks again. In total, each user completed 10 training tasks (2 per visualization) and 75 recorded tasks (5 visualizations $\times$ 3 environments $\times$ 5 tasks each). At the end, the participants were asked to rank and rate the visualizations and enter any comments in a post-study questionnaire.

4.2 Results

For each environment, the task completion times were analyzed using a two-way repeated measures ANOVA (with factors visualization and user and five tasks in each condition). No significant effect of the visualization was found ($F(4,236) = 1.85$, $F(4,236) = 1.54$, $F(4,232) = 0.49$, respectively, with $p > 0.05$ for all).

In the post-study questionnaire, users were asked to rank the five visualizations according to their preference as well as rate them on



(a) yellow: full size frame (b) red: full size frame with origin (c) blue: fixed size frame (d) green: fixed size frame with origin

Figure 6: *Four of the five viewpoint visualization conditions compared in the main study; the fifth condition was no visualization.*

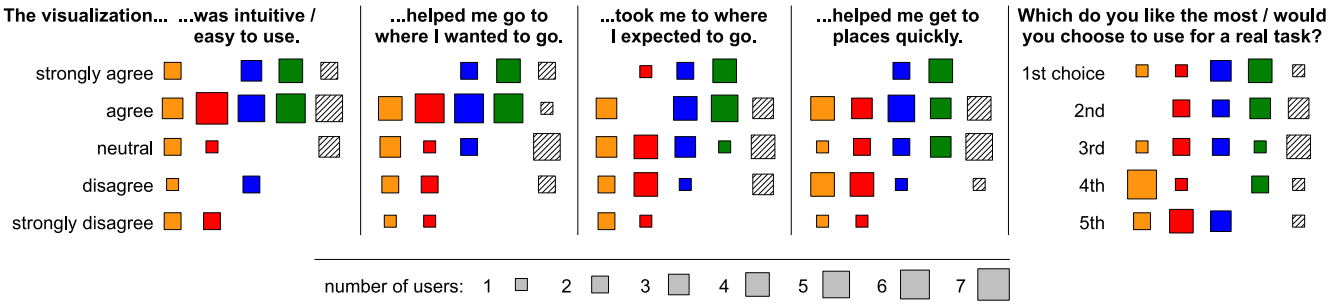


Figure 7: *Aggregated user responses from post-study questionnaire.*

a 5-point Likert scale with respect to four questions related to intuitiveness, helpfulness, predictability, and efficiency; questions and results are aggregated in Fig. 7.

Since these data are ordinal and not normally distributed, we used a non-parametric test for the analysis. At a significance level of 0.05, Friedman's test indicated that there is a significant effect of visualization on questions 2 and 3 ("The visualization helped me go to where I wanted to go," "The visualization took me to where I expected to go") with $\chi^2(4) = 12.85$, $p = 0.012$, and $\chi^2(4) = 14.12$, $p = 0.0069$, respectively.

Pairwise comparisons with Bonferroni's correction, i.e., significance level divided by number of pairwise comparisons (here: 10), used as post-hoc tests could not confirm significance between any particular pair of visualizations except for one case (green was rated significantly better than red ($p = 0.0047$) with respect to "The visualization took me to where I expected to go"). This is arguably not surprising since several factors — the small number of users, the fact that a non-parametric test had to be used, and a rather large number of conditions leading to a large Bonferroni correction factor — negatively impact the statistical power.

4.3 Discussion

We note four remarks on the apparent trend towards a higher rating of the blue and green visualizations, gained through observations and users' comments:

- While yellow and red provide strictly more information than blue and green (cf. Section 4.1), this information (namely, the exact extent of the camera's field of view) was not always needed for the given task. While we believe that the chosen task is very elementary and generic, other tasks (such as exactly matching a specific view) may prompt different ratings.
- In close-up views (such as required especially in our third environment), yellow and red take up a large portion of the screen and are only partially visible, which limits their usefulness.
- While yellow and red change mostly when a new viewpoint is selected, blue and green are always dynamically centered around the mouse cursor. Hence yellow and red may appear "jumpy," while blue and green appear more *fluid*.
- Further, yellow and red make the user more aware of the discrete set of available cameras, which, unless specifically desired, may be considered to decrease *presence* since it emphasizes a property of the model rather than the world in which the user should feel present.

Some observations might help to explain the lack of a significant effect of the visualization on the task performance:

First, in very difficult cases such as very cluttered close-ups, it became apparent that all of the visualizations suffered from certain drawbacks, such as being only partially visible, and thus the visual clutter and cognitive load of trying to identify the "best" view may not have "paid off."

Second, it should be noted that, to guarantee that each task was actually solvable, we had made sure that there was at least one "good" viewpoint for each object of interest. Thus, even though we did not consider our viewpoint selection algorithm to be especially sophisticated, in easy cases it was oftentimes good enough to select this "good" view even if the user just clicked on the object of interest without paying attention to the visualization at all. This helps explain why the "no visualization" option performed comparably to the other four.

Effectively, in severely cluttered scenarios, some users appeared to not pay much attention to the visualizations, and just "clicked around" until they would be able to see the target information, which, given the aforementioned reasons and the fact that switching to a new viewpoint was quick, turned out to be a reasonable strategy for many targets. In those cases, the no visualization option, void of any visual clutter distracting from the environment, would indeed be preferable.

We emphasize that this was a small-scale study with limited statis-

tical power, designed to do a preliminary investigation into viewpoint visualizations for the single-click design space in order to gain some insights, rather than to prove the superiority of a particular visualization. While we tried to design the tasks to be of similar difficulty and balanced across users and conditions, variance in their difficulty increased the noise in our data, which in turn would require more users/trials.

5 Conclusion & Future Work

Single-click 3D navigation is appealing due to its simplicity and suitability for constrained navigation. However, the minimal (2-DoF) input poses a difficult design challenge. In this paper, we gave a description of the design space for single-click 3D navigation and offered an exploratory user study into its viewpoint visualizations. Two main design considerations are the viewpoint selection algorithm and the viewpoint visualization. As shown in Fig. 2 and Fig. 3, many existing systems use single-click 3D navigation in different manifestations.

Creating a design space for single-click 3D navigation brings about a realization that certain design configurations have not been fully explored. For example, relatively few existing works have investigated viewpoint selection algorithms where the user can click directly into the scene. In such a scenario, how does one choose the proper interpretation of the cursor (x,y) location? How does the viewpoint selection algorithm differ based on the scene semantics and task? What about different levels of available viewpoints? Does anything change when, for example, the 3D scene is being reconstructed live [Newcombe and Davison 2010; Klein and Murray 2007]? Two conditions which we observed to be especially difficult to navigate by clicking directly into the scene are zooming in/out and seeing an object from the other side (i.e., orbiting). Brivio et al. [2013] also note similar difficulties.

As seen from our preliminary user study, we also noticed that providing good viewpoint visualizations for single-click 3D navigation can be nontrivial (especially for clicking directly into the scene). On the one hand, for simple environments with a small set of available viewpoints or a good viewpoint selection algorithm, no visualization may even be necessary for continued use (although it may be useful for *discoverability* of the single-click feature). On the other hand, geometrically complex and arbitrarily constrained scenes, especially in close-ups of cluttered environments, pose a significant challenge to the usefulness of current visualizations. In our study, we found that users seem to prefer simple and fluid visualizations even if they provide less information. Thus, we suggest that a smart viewpoint selection algorithm, selecting predictable and possibly perceptually preferred [Secord et al. 2011] viewpoints, is perhaps more important than the visualization and deserves more in-depth investigation for single-click 3D navigation interfaces.

It would further be interesting to investigate in how far visualizations help the user to determine that there is *no* suitable viewpoint available (in analogy to the result by Pausch et al. [1997], who found that their more immersive setup did not help the user in finding an existing object, but did help when the object was not present).

Overall, we believe our taxonomy for single-click 3D navigation will prove useful for 3D user interface designers, specifically for navigating 3D scenes and understanding the design components that make up single-click 3D navigation. We feel that simple user interfaces, such as those employing single-click 3D navigation, are a crucial ingredient to bring 3D navigation of virtual scenes to the masses.

References

AGARWAL, S., FURUKAWA, Y., SNAVELY, N., SIMON, I., CURLESS, B., SEITZ, S. M., AND SZELISKI, R. 2011. Building rome in a day. *Commun. ACM 54*, 10 (Oct.), 105–112.

BOWMAN, D. A., KOLLER, D., AND HODGES, L. F. 1997. Travel in immersive virtual environments: An evaluation of viewpoint motion control techniques. In *Proceedings of the 1997 Virtual Reality Annual International Symposium (VRAIS '97)*.

BOWMAN, D. A., KRUIJFF, E., LAVIOLA, J., AND POUPYREV, I. 2005. *3D User Interfaces: Theory and Practice*. Addison-Wesley, Boston.

BRIVIO, P., BENEDETTI, L., TARINI, M., PONCHIO, F., CIGNONI, P., AND SCOPIGNO, R. 2013. PhotoCloud: Interactive remote exploration of joint 2D and 3D datasets. *IEEE Comput. Graph. Appl. 33*, 2, 86–96, c3.

ELVINS, T. T., NADEAU, D. R., SCHUL, R., AND KIRSH, D. 1998. Worldlets: 3d thumbnails for 3d browsing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, CHI '98, 163–170.

HACHET, M., DECLEC, F., KNODEL, S., AND GUITTON, P. 2008. Navidget for easy 3d camera positioning from 2d inputs. In *3D User Interfaces, 2008. 3DUI 2008. IEEE Symposium on*, 83–89.

JANKOWSKI, J., AND HACHET, M. 2013. A survey of interaction techniques for interactive 3D environments. In *Proc. Eurographics*.

KLEIN, G., AND MURRAY, D. 2007. Parallel tracking and mapping for small AR workspaces. In *Proc. 6th IEEE and ACM Intl. Symposium on Mixed and Augmented Reality (ISMAR)*.

MACKINLAY, J. D., CARD, S. K., AND ROBERTSON, G. G. 1990. Rapid controlled movement through a virtual 3D workspace. In *Proc. 17th annual conference on Computer graphics and interactive techniques (SIGGRAPH)*, ACM, New York, NY, USA, 171–176.

NEWCOMBE, R. A., AND DAVISON, A. J. 2010. Live dense reconstruction with a single moving camera. In *Proc. Computer Vision and Pattern Recognition (CVPR)*.

PAUSCH, R., PROFFITT, D., AND WILLIAMS, G. 1997. Quantifying immersion in virtual reality. In *Proc. 24th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 13–18.

SECORD, A., LU, J., FINKELSTEIN, A., SINGH, M., AND NEALEN, A. 2011. Perceptual models of viewpoint preference. *ACM Trans. Graph. 30*, 109:1–109:12.

SNAVELY, N., SEITZ, S. M., AND SZELISKI, R. 2006. Photo tourism: exploring photo collections in 3D. *ACM Trans. Graph. 25*, 3, 835–846.

SNAVELY, N., GARG, R., SEITZ, S. M., AND SZELISKI, R. 2008. Finding paths through the world's photos. In *ACM SIGGRAPH*, ACM, New York, NY, USA, SIGGRAPH '08, 15:1–15:11.

SUKAN, M., FEINER, S., TVERSKY, B., AND ENERGIN, S. 2012. Quick viewpoint switching for manipulating virtual objects in hand-held augmented reality using stored snapshots. In *Proc. 11th IEEE Int'l Symp. on Mixed and Augmented Reality (ISMAR)*.

TAN, D. S., ROBERTSON, G. G., AND CZERWINSKI, M. 2001. Exploring 3D navigation: combining speed-coupled flying with orbiting. In *Proc. ACM SIGCHI Conference on Human Factors in Computing Systems (CHI)*, ACM, New York, NY, USA, 418–425.

TATZGERN, M., GRASSET, R., KALKOFEN, D., AND SCHMALSTIEG, D. 2014. Transitional augmented reality navigation for live captured scenes. In *Virtual Reality (VR), 2014 iEEE*, IEEE, 21–26.

VERGAUWEN, M., AND VAN GOOL, L. 2006. Web-based 3d reconstruction service. *Mach. Vision Appl. 17*, 6 (Oct.), 411–426.

VINCENT, L. 2007. Taking online maps down to street level. *Computer 40*, 12 (Dec.), 118–120.

ZELEZNIK, R., AND FORSBERG, A. 1999. Unicam—2d gestural camera controls for 3d environments. In *Proceedings of the 1999 Symposium on Interactive 3D Graphics*, ACM, New York, NY, USA, I3D '99, 169–173.