Street Slide: Browsing Street Level Imagery

Johannes Kopf
Microsoft Research

Billy Chen
Microsoft

Richard Szeliski
Microsoft Research

Michael Cohen
Microsoft Research

Figure 1: *A multi-perspective* street slide *panorama with navigational aides and mini-map.*

Abstract

Systems such as Google Street View and Bing Maps Streetside enable users to virtually visit cities by navigating between immersive 360°panoramas, or *bubbles*. The discrete moves from bubble to bubble enabled in these systems do not provide a good visual sense of a larger aggregate such as a whole city block. Multi-perspective "strip" panoramas can provide a visual summary of a city street but lack the full realism of immersive panoramas.

We present Street Slide, which combines the best aspects of the immersive nature of bubbles with the overview provided by multi-perspective strip panoramas. We demonstrate a seamless transition between bubbles and multi-perspective panoramas. We also present a dynamic construction of the panoramas which overcomes many of the limitations of previous systems. As the user slides sideways, the multi-perspective panorama is constructed and rendered dynamically to simulate either a perspective or *hyper-perspective* view. This provides a strong sense of parallax, which adds to the immersion. We call this form of sliding sideways while looking at a street façade a *street slide*. Finally we integrate annotations and a mini-map within the user interface to provide geographic information as well additional affordances for navigation. We demonstrate our Street Slide system on a series of intersecting streets in an urban setting. We report the results of a user study, which shows that visual searching is greatly enhanced with the Street Slide interface over existing systems from Google and Bing.

1 Introduction

The ability to virtually visit remote locations has been one of the central quests in computer graphics, dating back to the seminal work on Movie Maps in the Aspen Project [Lippman 1980]. Immersive experiences based on 360° panoramas [Chen 1995] have long been a mainstay of "VR" photography, especially with the advent of digital cameras and reliable automated stitching software. Today, systems such as Google Street View [Vincent 2007] and Bing Maps Streetside enable users to virtually visit many areas by navigating between immersive 360°panoramas sometimes referred to as *bubbles*[1].

[1]We are not sure if this term is widely used in the graphics community, but appears to be common in discussions of the Google Street View system [Vincent 2007].

Unfortunately, while panning and zooming inside a bubble provides a photorealistic impression from a particular viewpoint, it does not provide a good visual sense of a larger aggregate such as a whole city block or longer street. Navigating such photo collections is thus laborious and similar to hunting for a given location on foot: walk "along" the street, (e.g., jumping from bubble to bubble in Street View) looking around, until you find the location of interest. Since automatically geolocated addresses and/or GPS readings are often off by 50 meters or more especially in urban settings, visually searching for a location is often needed. Within a bubble, severe foreshortening of a streetside from such a distance makes recognition almost impossible.

Multi-perspective "strip" panoramas [Román et al. 2004; Román and Lensch 2006; Agarwala et al. 2006; Rav-Acha et al. 2008], on the other hand, provide a useful visual summary of all the landmarks along a city street. Unfortunately, because all of the source imagery is compressed into a single flat summary, they lack the parallax effects as one moves along a street present in other image-based rendering experiences.

In this paper we combine the best aspects of the immersive nature of bubbles with the overview provided by multi-perspective strip panoramas. We present a dynamic construction of strip panoramas, which overcomes many of the limitations of previous systems. Our strip panoramas are created by aligning and overlaying perspective projections of the bubble images oriented towards the street side. Dynamically altering the alignment and visible portions of each image simulates a pseudo-perspective view of the streetside from a distance. Moving along the street thus gives a strong sense of parallax and enhances the sense of immersion. We call this form of sliding sideways coupled with a dynamic rendering of a street façade a *street slide*.

As the user zooms out to get a broader overview, the multi-perspective panorama dynamically adapts the fields of view of the constituent images to maintain a pseudo-perspective view as opposed to a *pushbroom* projection. The visual effect is one of increasing the apparent viewing distance rather than a simple scaling of the panorama. Since changing the image widths changes the relationship between one image's edge pixels and those of the next image behind it, our system also dynamically adapts the alignment between images to create a more seamless visual experience. As the user zooms out, the wider aspect ratio of the panorama reveals empty space above and below the panorama, which can be used to annotate the imagery with useful information such as business

names and addresses, and other navigation aids, which can be seen in Figure 1.

Zooming back into the image reverses the process and results in a smooth transition between the multi-perspective strip panorama and the traditional immersive 360° bubble. This combination of multi-viewpoint street slide panoramas integrated with traditional cylindrical panoramas provides the means to quickly navigate along streets in an informative way, while enabling the ability to turn around, go down side streets, and to zoom into single panoramas to examine the details of the individual storefronts.

To assess the value of the Street Slide interface, we conducted a user study in which we placed users at some bubble location and asked them to find a nearby landmark. We compared Google Street View with Street Slide and found that users had a much easier time navigating and finding the goal point. We also collected feedback from a more freeform navigation session.

Our paper thus introduces a number of novel contributions to real-world immersive image-based rendering and navigation. The elements of our system include:

- The ability to quickly navigate down a street in an informative view of the street side that maintains both photorealism and parallax by dynamically constructing multi-perspective panoramas.
- The ability to simulate smoothly dollying backwards to provide large scale (distant) overviews, by dynamically aligning adjacent image strips.
- A smooth interpolation between multi-perspective and single perspective panoramas, and
- An intuitive user interface for traversing street side imagery, including annotations for adding information and navigation affordances.

2 Related Work

Our work primarily builds on two intertwined themes in image-based rendering and computational photography: single-perspective panoramic image construction, rendering, and navigation, and multi-perspective image construction.

Immersive 360° panoramic image construction and rendering has been a popular topic of research in the graphics community since the introduction of QuickTime VR [Chen 1995] and Plenoptic Modeling [McMillan and Bishop 1995]. A large number of papers have been published relating to the seamless construction of such panoramas. We refer the reader to [Szeliski 2006] for a recent survey.

Navigating from bubble to bubble has been used as a means to explore virtual spaces since the early days of QuickTime VR, and even before that in the Movie Map system [Lippman 1980]. 360° video panoramas captured with a lightweight portable camera are described in [Uyttendaele et al. 2004]. The same camera (the Ladybug by Point Grey Research) was also used in early versions of the Google Street View system [Vincent 2007], which commercialized the idea of immersive videos on a world-wide scale by filming complete cities from moving vehicles.

Panoramic image viewers do not provide a wide-angle "overview" of an immersive environment without generating severe distortions. The work of [Kopf et al. 2007] shows how to mitigate some of these distortions by moving seamlessly between linear perspective projections (for narrow fields of view) and curved projections (for wide fields of view). Unfortunately even wide angle views do not eliminate the severe foreshortening of scene elements that do not face the viewpoint. We thus move from single-perspective to multi-perspective images.

Multi-perspective imaging dates back to the earliest days of aerial photography. Many aerial cameras are "pushbroom" cameras, which only acquire a single strip of pixels at a time and later assemble the strips into large aerial photomosaics [Gupta and Hartley 1997]. A general work outlining the range of possible 2D projections including multi-perspective strip panoramas can be found in [Zomet et al. 2003].

In ground-level photography, the challenge in creating seamless panoramas is the large range of depths that exist in each image. One solution to this problem is to use only a center strip from each image. However, nearby objects such as cars then appear squashed while distant objects such as mountains become stretched.

Román *et al.* [2004] describe a system where the user selects which vertical strips from each image to use in the final panorama. By selecting pixels from the same image at street intersections and limiting transitions to large vertical façades, they maintain the illusion of planar perspective while also reducing visible seams. In a follow-on paper Roman and Lensch [2006] show how this process can be automated. Instead of taking vertical strips, Agarwala *et al.* [2006] use general Markov random field optimization techniques based on graph cuts [Kwatra et al. 2003], with some user input, to cut around objects such as cars and to place inter-image seams in areas of good visible agreement. This paper along with a more recent paper by Rav-Acha *et al.* [2008] provide a good review of related literature. All such constructions are static and independent of viewpoint, and thus break the illusion of immersion as one navigates.

Instead of creating a single seamless panorama, we dynamically align and select portions of each panorama to approximate a perspective or hyper-perspective image from some distance behind the line from which the imagery was captured. The resulting multi-perspective panoramas are related to the idea of stitching the left-most and rightmost columns of an offset rotating camera [Peleg et al. 2001].

Our work is also related to more classic approaches to image-based rendering such as the Lumigraph [Gortler et al. 1996], Light Field [Levoy and Hanrahan 1996], and Unstructured Lumigraph [Buehler et al. 2001], since the images we render can be thought of as adaptive slices (manifolds) [Peleg et al. 2000] through the four-dimensional light field. Our work is also closely related to multi-perspective images [Rademacher and Bishop 1998]. As we will see, instead of explicitly modeling full 3D geometric proxies [Agarwala et al. 2006], we use a simple correlation alignment technique for aligning adjacent vertical strips.

3 Street Slide Overview

Our goal in constructing the Street Slide application is to provide a seamless means to explore the details of individual bubbles while also being able to navigate efficiently and intuitively along the streets. Our solution consists of a hybrid between a traditional pan and zoom interface within a bubble for small fields of view and a multi-perspective panorama for wider views of the street side. As the user's field of view within a single bubble widens to a threshold, we lock the apparent field of view and instead simulate dollying backwards by combining imagery from nearby bubbles to form a multi-perspective panorama (see Figure 2).

When viewing the multi-perspective panorama, the user is able to *slide* along the street to examine the store fronts and other local features to the left and right. Affordances allow turning around to view the other side of the street and turning at intersections. A map also provides context and an easy means to jump to new locations.

As the viewer zooms back in towards the street side, the process is reversed. The view transitions from the multi-perspective panorama back into the local bubble and the viewer once again navigates within this bubble by panning and zooming. After a brief overview of the data acquisition and preprocessing system, we describe each of the above aspects of Street Slide in more detail.

The data used by Street Slide consists of a series of 360°panoramas captured on city streets about every 2 meters. The imagery is captured by a 10 camera panoramic rig mounted on top of a car. The individual images are stitched together into a seamless panorama re-projected onto cube faces with a resolution of 2048 by 2048 pixels per face.

Given a single panorama as in the top of Figure 2, a standard pan and zoom interface [Chen 1995] allows a user to turn the virtual camera, zoom in to details, and zoom outward within limits. The cube faces of the panorama are dynamically remapped to a perspective projection. As the user zooms outward even more, approaching a 90 degree field of view, the direction of view is automatically panned to the closest street side facing direction in preparation for transitioning to the multi-perspective panorama of the street side. Any further outward zooming is converted into a simulated dolly motion away from the scene, as described in Section 5. We describe the construction and navigation of multi-perspective panoramas next, and then discuss the seamless transition between modes.

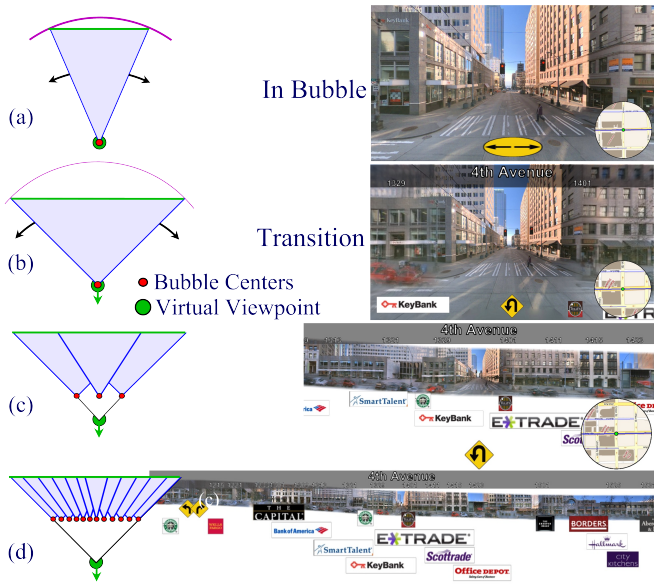

Figure 2: *Starting from within a bubble (a), as the field of view reaches 90°(b), the view transitions to a multi-perspective panorama and the virtual camera begins to dolly backwards (c) revealing slices from adjacent bubbles, (d). Note that the virtual camera remains locked at 90° during the apparent backward motion. Also, the field of view of the central slice reduces from its initial 90° while the edges of outer most slices stay at ±45° to simulate a backward dollying (hyper) perspective camera.*

4 Multi-Perspective Street Level Panoramas

Looking down a street from within a bubble, building facades only a short distance along the street quickly become severely foreshortened and shrink in size. Instead, we would like to virtually move the viewpoint backwards so that a smaller, less distorted, field of view can encompass a wider portion of the street side. Unfortunately, this is impossible to achieve directly for a number of reasons: aside from the need to capture an exponentially larger amount of data, there typically are buildings across the street that preclude directly capturing such a distant view.

Thus, we are left with the task of approximating such a view by combining portions of multiple panoramas captured from differing viewpoints. We take a simpler approach than [Román et al. 2004; Agarwala et al. 2006], first, to avoid the need for user input, and second, to allow the street side panorama to adapt dynamically as the user steps further back and moves left/right along the street.

4.1 Constructing the Multi-perspective Panorama

The simplest idea for constructing the multi-perspective panorama would be to clip out fixed size, *slices* from each bubble oriented towards the street side and lay them side-by-side. (We use the term *slice* to denote a vertical strip of some width and position on the cube face of the bubble facing the street side.) This simple idea presents a number of problems. The width of the slices to just cover what is seen depends on the (unknown) depth of the scene; nearer objects require wider slices (i.e., there is a large disparity), while infinitely distant scenes are fully captured in a single bubble (i.e., there is zero disparity, so all bubbles are identical and any size slice would lead to repeating scene elements).

A second problem is that we also want to simulate the subtle parallax caused by moving backwards, as opposed to just shrinking the panorama. More importantly, the apparent view should exhibit parallax as we move left and right down the street. We achieve these goals by dynamically selecting the width of slices from each bubble, along with their alignment, to construct the multi-perspective panorama at each frame time (see Figure 2). The result is a dynamic multi-perspective panorama displaying parallax as one moves forward and backward and left and right. Note, however, that in any non-planar scene, such a multi-perspective panorama cannot be artifact free due to varying depths within any vertical strip.

The dynamic construction of the multi-perspective panorama begins when the user zooms outward from within a bubble to a field of view greater than 90°. At the transition from the bubble to the street slide (Figure 2b), the 90°field of view slice of the bubble (the full face of the bubble cube) forms the central image of the multi-perspective panorama and fills the view.

Zooming out further by dollying backwards (Figure 2c and d), the central image slice both narrows and scales downward providing space on either side for slices from adjacent bubbles. We no longer have the exact rays needed to fully reconstruct such an image. Taking inspiration from previous image-based rendering work, our goal is for each vertical slice of the multi-perspective panorama[2] to be taken from a single bubble.

At this point, we have some aesthetic freedom in constructing the multi-perspective panorama from the bubbles. We can form an *as-perspective-as-possible* result, or create what we call a *hyper-perspective* panorama. We discuss each of these in order.

4.1.1 An As-Perspective-as-Possible Panorama

To construct an as-perspective-as-possible multi-perspective panorama, for each vertical column of the multi-perspective panorama, we must select the *closest* column available from the bubbles we have. This construction requires selecting where to transition from each bubble to its neighbor and how to align

[2] *Vertical* in this context refers to the up vector in the captured images, which will not match gravity on sloped streets. On such streets, vertical strips get drawn as tilted on the screen to maintain the verticality of buildings–see Section 4.4.

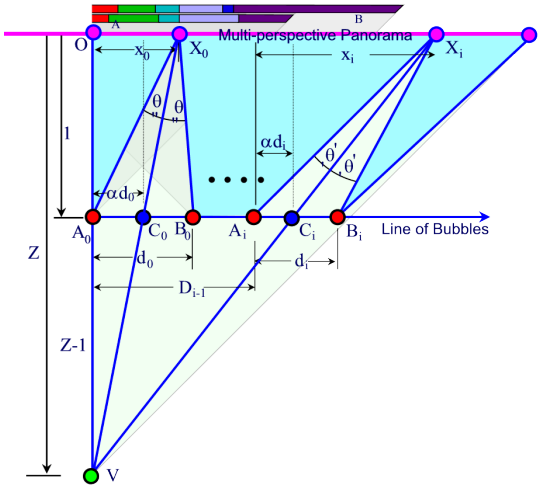

Figure 3: *Computing the slice widths and alignments of bubbles for the right side of the multi-perspective panorama. V is the virtual viewpoint. A and B are neighboring bubbles. X marks the transition point and d is the disparity, while D is the sum of the disparities so far.*

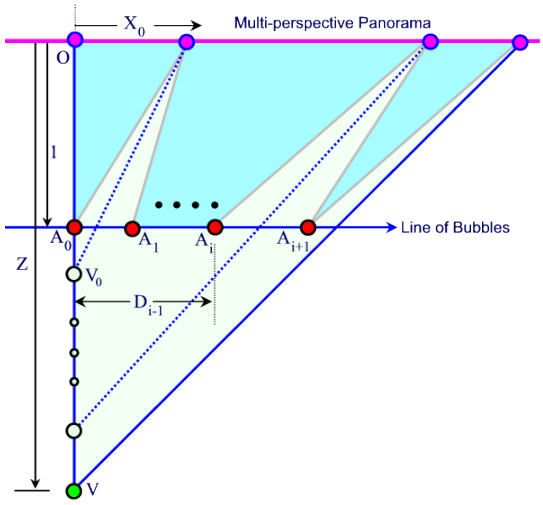

Figure 4: *A hyper-perspective panorama represents a perspective view constructed from a viewpoint varying from V_0 to V.*

the neighboring bubble at that transition point based on image *disparity*. Unfortunately, for each pair of neighboring bubbles, these two decisions are interdependent; i.e., each column may contain different dominant scene depths, and the scene depths affect the alignment. We begin by assuming that, given the transition column, we have a method to determine the alignment as described in Section 4.3.

The transition column is chosen based on the notion of *closest* as was done in Buehler *et al.* [2001], i.e., by minimizing the angular deviations of the column between the reconstructed multi-perspective panorama and the available bubbles. Figure 3 schematically illustrates an overhead view of the construction of the right hand side of the multi-perspective panorama along the pink line through the origin at the top of the figure. Slices of original images captured at A_0, B_0, ... A_i, B_i, which are shown schematically as the colored red, green, ... purple segments. V represents the location of the simulated viewpoint, at some distance Z from the origin. Note that when $Z \leq 1$ the user is "in the bubble", and the central bubble subtends the full view.

Figure 3 shows the selection of the boundary between two pairs of bubbles, A_0 and B_0, and A_i and B_i. X_0 and X_i represent vertical columns in the multi-perspective panorama. They also define the boundaries of the slices from the bubbles used to form the final strip panorama. The alignment distance (horizontal disparity) between bubbles A and B is d. D_i represents the accumulated alignments starting from the center and working outwards toward A_i. The transition point X_0 or X_i should occur when the angle from A to X to B is bisected by the line from V to X.

Given a candidate disparity d, to find X, we leverage the angle bisection theorem, which states that the angle bisector creates equal ratios of edge lengths $|\overline{AX}|/|\overline{BX}| = |\overline{AC}|/|\overline{BC}|$. Unfortunately d depends on X since the alignment depends on the real world distance of the scene along the ray from V to Z.

To solve the above problem, we alternate between choosing a transition column and aligning the adjacent bubble at that point. Thus, referring to figure 3 for the center bubble, the algorithm proceeds as:

1. Initialize $x = 1$, i.e. select the full extent of bubble A.
2. Find d that aligns bubbles A and B at X.
3. Determine α such that $\angle AXB$ is bisected.
4. Given d and α, based on similar triangles VA_0C_0 and VOX_0, reset $x_0 = \alpha d_0 Z/(Z-1)$.
5. Repeat steps 2-4 until x converges.

The alignment in step 2 is discussed in Section 4.3. Step 3, determining α can be done analytically as the solution of a cubic equation. In practice, it is easier to initialize $\alpha = 0.5$, find the corresponding x from step 4 and then reset $\alpha = |\overline{AX}|/(|\overline{BX}| + |\overline{AX}|)$, and repeat for 10 iterations.

The more general case is shown on the right side of Figure 3 for aligning the subsequent bubbles. The only change is that the accumulated offsets are summed into d. Based on similar triangles VA_iC_i and VOX_i, Step 4 above becomes

$$X_i = (D_{i-1} + \alpha_i d_i) Z/(Z-1). \quad (1)$$

The result is similar to rendering the appropriate 2D slice of rays from a 4D Light Field or Lumigraph [Levoy and Hanrahan 1996; Gortler et al. 1996; Buehler et al. 2001].

4.1.2 A Hyper-Perspective Panorama

We now present a second closely related *hyper-perspective*[3] construction of multi-perspective panoramas that accentuates the perspective in the center of the panorama. We will discuss observations relating the hyper-perspective and as-perspective-as-possible constructions after the details.

We begin the hyper-perspective multi-perspective panorama construction by setting the width of the center slice X_0 as a function of Z,

$$X_0 = \max(0.35, 1 - (Z-1)/30). \quad (2)$$

[3]We borrow the term hyper-perspective from art and photography which purposely exaggerates perspective for artistic effect.

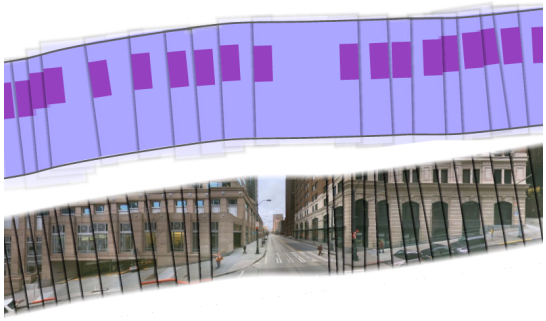

Figure 5: *Starting from the center slice, each neighboring slice is rotated to place the world vertical direction pointing up, then aligned by translating the neighboring slice until the best match is found between the dark rectangular region and the aligned region behind it. A smoothed line through the slice centers is used to trim the top and bottom. Edges between slices are also feathered with more aggressive feathering at the bottom of the slices.*

The value of 0.35 was selected to create a minimum field of view of the central slice of 40°.

To determine the alignments between adjacent bubbles, i.e., the A_i in Figure 4, we first compute the alignment d_0 for the first pair of images (at X_0) and set $D_0 = d_0$. As we align the images, we want the artificially large field of view of the central image to smoothly transition toward a true perspective rendering at the outer edges. To accomplish this, we take the distance along the "Line of Bubbles" in Figure 4, which is $Z - 1$ (because the virtual camera's field of view is 90°), and stretch it to cover the remaining pixels along the top row, which is $Z - X_0$. The resulting general formula for the X_i is therefore given by

$$X_i = X_0 + \frac{D_{i-1}(Z - X_0)}{(Z - 1)}. \tag{3}$$

This construction is equivalent to beginning with a virtual camera at some distance $Z_0 > 1$ and sliding back toward the true virtual camera at Z as shown in Figure 4.

4.1.3 Discussion of the Alternative Constructions

We have described two alternatives to the construction of multi-perspective panoramas. The first observation, worth repeating, is that there is no *correct* multi-perspective panorama in the sense that no fixed camera can ever observe one at a single point in time. That said, although the as-perspective-as-possible panorama represents a more theoretically correct view one would get from a distance, we have found that the accentuated perspective in the center of the multi-perspective panorama of the hyper-perspective construction has a number advantages over the as-perspective-as-possible model. First, our mental model of looking down a crossing street is better captured by a wider view from near the intersection rather than from a long distance. Second, as discussed in section 4.3, the alignment is only approximate when there are multiple depths in a local region, which is especially the case at intersections. Thus, the hyper-perspective construction tends to hide alignment artifacts better, especially in the visual center of the panorama. Finally, the accentuated central perspective also exaggerates the parallax as one slides along the panorama, as discussed next. Given these observations, we have chosen the hyper-perspective model for the study outlined in Section 7.

4.2 Sliding

In addition to moving the view backwards, one can also *slide* along the multi-perspective panorama. The above construction shows the view from a position directly behind one bubble. As the user slides the view sideways, we flip the next bubble to be central one as we cross the centerline between them, i.e., at half of the initial alignment d. This creates a flipbook animation as the central image changes. It is this change of central bubble coupled with the choice of slices that creates parallax as one translates the view sideways.

4.3 Image Alignment

When we first transition from a bubble to a street slide strip panorama, we first assemble the set of 90° field of view faces from each bubble's cubemap oriented towards the street side. We align each neighboring pair of images as best we can to minimize the seam between them (see Figure 5). It should be stressed that unless there is a constant depth to the scene, no alignment can achieve a perfectly seamless result. In general, the bottom half of each image depicts the roadway, sidewalk and transitory objects like cars and pedestrians. The upper half generally contains building fronts, while the uppermost part of the image depicts the distant parts of a scene (e.g., the sky). We thus choose to optimize our alignment for the portion of the image between the horizontal midline and three quarters of the way up.

Depending on the momentary zoom and slide position along the street, the vertical column we wish to align with the next bubble changes dynamically. The alignment (or disparity) value, d, thus needs to be determined on-the-fly. Unfortunately, these alignments are too costly to perform at render time. Instead, we precompute the disparity between each pair of neighboring bubbles at 11 different vertical columns, from the centerline out to the right extent at 10% increments for the right neighbor, and similarly for the left neighbor.

We also compute the vertical disparity to account for slight rolling of the capture vehicle. Finally, since streets are not all level, we use the camera pose estimation recorded at capture time to avoid searching over rotation, i.e., in-plane rotation of neighboring images is done based on the known difference in gravity vectors before performing the 22 alignments as shown in Figure 5.

Each alignment is performed offline using a simple SSD (sum of squared differences) minimization. For our 512×512 images, we align a 64 by 128 pixel rectangle lying just above the midline and just inside the column to be aligned (see the dark rectangles in Figure 5). We perform an exhaustive search over 192 horizontal pixel offsets by 31 vertical offsets using Fast Fourier Transforms to efficiently perform this computation [Szeliski 2006].

A lack of texture or aliasing due to fine repeated structures can produce erroneous results. We thus perform a median filtering across the 5 neighboring bubbles and across 7 alignments at the differing vertical columns to reduce such errors. Finally, the 11 alignments with the right neighbor and similarly for the left neighbor are linearly interpolated to produce an alignment for any given vertical column. With our current data set, we have found this approach to produce good alignments for almost all pairs.

4.4 Rendering the Multi-Perspective Panorama

Given the selection and alignment of the slices for a given view, we render the slices outwards-to-inwards, drawing one over another. Alignment errors due to non-planar scenes are mitigated by rendering the slices with alpha blended feathering of the slice boundaries. The region in which the alpha channel varies linearly from zero to one varies from 1 to 5 times the width of the visible slice. The

blending region is set to 1x the visible width for the top half of the slice. Since most alignment errors are found in the bottom half of the slices, we use a curve to smoothly increase the feathering region there from 1x to 5x of its size in the top half. Also, the vertical disparities especially on sloped roads creates a stair-step-like top and bottom edge. We thus trim the vertical extents with a smoothed curve parallel to one fit to the slice centers. We trim off the top 3% and bottom 15% of the slice height to create the final multi-perspective panorama (see Figure 5).

The Street Slide system is implemented in C++ using OpenGL. Images are rendered off screen and then composited with the annotations. The data is loaded over the network in a separate thread. We achieve a frame rate between 40 to 60 frames per second on a modern PC.

5 Transitions between Bubbles and Street Slide

An important aspect of our Street Slide system is the seamless transition between the 360° bubbles and the multi-perspective panorama. Starting from within a bubble, as we zoom out and approach a field of view of 90°, the panning is automatically adjusted to bring the orientation to the nearest side of the street. At 90°, the view of the bubble exactly matches the central image of a corresponding multi-perspective panorama with $Z = 1$. Zooming out beyond 90°triggers a cross fade from the wider projection of the bubble to the multi-perspective panorama, as portions of the slices from the neighboring bubbles appear. This provides the visual transition between the bubbles and the street slide.

The affordances of the navigation also undergo a state change. Right-button-pressed vertical mouse moves, as well as scroll wheel movement, are always mapped to zooming in or out in both the bubbles and when viewing the multi-perspective panorama. Within the bubbles, horizontal motion with the left mouse button pressed corresponds to panning (i.e., turning the virtual camera). After the transition to the street slide, the same mouse motion is mapped to translation along the street. Small motions are visually equivalent at the transition point, since small rotations are indistinguishable from small translations.

We also leverage the bubbles to transition between street slides. For example, if we are looking at the north side of the street and want to turn around to look at the south side, we automatically generate the following sequence. The view zooms in until the transition to the bubble occurs, the view rotates 180°, and finally, the view zooms back out enough to transition back to the street slide of the other side of the street. At intersection bubbles, a right or left turn is created the same way, except that the rotation in the bubble is set to the angle of the intersection (90° for orthogonal streets). Visual affordances in the user interface trigger the turns (see Figure 1).

6 Annotations

As the view of the multi-perspective panorama zooms out, the scale of the imagery decreases, and more slices from neighboring bubbles appear to fill the horizontal aspect of the window. This change in aspect ratio of the street slide leaves unused screen real estate above and below the multi-perspective panorama. We leverage this space to add annotations for information and navigation aids.

Diamond shaped street signs placed below the bottom of the street slide, as seen in Figure 1, provide the affordances for transitioning between multi-perspective panoramas. Street names lie directly above the strip with building numbers that when clicked slide the view smoothly to land in front of that address. Finally, icons containing store names and/or logos below the strip, can be selected

Figure 6: *A typical goal image.*

to smoothly slide the strip and zoom the view to place the viewer inside the closest bubble with a view directed at that storefront.

A mini-map in the lower right corner of the screen is centered on the current location of the bubble the view is in (in bubble mode) or the central image of the strip. The mini-map zoom level is set relative to the view's zoom level. The map is also oriented to keep the viewing direction upward. If the user flips to the other side of the street in street slide mode, the map spins 180°.

Mousing over the map doubles the map's scope and size. With the mouse in the map, the user can drag the center point left and right along the current street. This induces a corresponding translation of the street slide. Finally, the user can double click on any street location to be transported to that location. Moving the mouse out of the mini-map returns it to its original size.

7 User Study

To assess the value of the Street Slide (SSL) interface, we conducted a user study to compare it to current street level navigation tools such as Google Street View (GSV), as well as to gain general insights into the usability of our interface.

In particular, we tested the efficacy of the two interfaces to find a visual landmark when already positioned near the goal location. This problem is motivated by two common real world scenarios. When searching for a particular address, errors in the geocoding of an address will place a user nearby but often not directly in front of the sought after address. A similar situation occurs in a mobile setting where GPS coordinates, particularly in urban canyons, can be significantly off.

7.1 The Study

We had 20 subjects, all heavy computer users, ranging in age from 25 to 55. All but one were male. We began the session with two questions using a 5 point Likert scale to assess the subjects familiarity with (a) Seattle's downtown since this is the imagery we used, and (b) with Google Street View.

A 5 minute introduction to both Google Street View and Street Slide included demonstrating the ability to jump to discrete locations based on geometric features in Google Street View. As noted above the hyper-perspective construction was used for the Street Slide. We disabled the annotations in SSL, and disabled the mini-map in both interfaces to provide a more direct comparison of the in-image navigation between interfaces. The subjects were then asked to use the interfaces until they stated they felt comfortable with them.

Each subject was then tested 8 times; four times with each interface, either 4 GSV's followed by 4 SSL's, or vice-versa. The interface started in a given location and oriented to face a street side. For each test, the subject was given one of 8 images of a building front

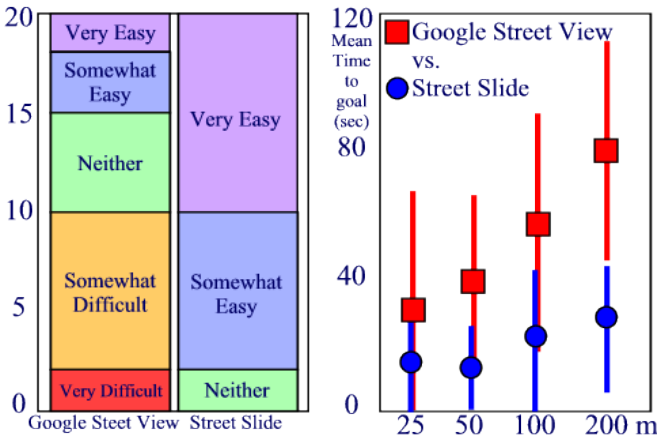

Figure 7: *Ease (left) and timings (right) for finding the goal images. The mean times for the different distances clearly show that Street Slide is much faster for the given task than Google Street View. Times also suggest a linear increase with distance in both interfaces.*

to find as a goal (e.g., Figure 6). Subjects were told that the goal was on the same street and on the same side of the street they were facing within 2 blocks (about 200 meters) in either direction. We placed goals at approximately 25, 50, 100, and 200 meters from the starting positions in randomized order within each interface's 4 tests. For each subject, the 8 tests covered all 8 goal images. The goal images were captured independently of either data set to avoid bias due to the lighting and other differences between interfaces.

The subjects were asked to navigate to the goal as quickly as possible. Each test was timed. Subjects were asked to verbally indicate when they saw the goal. The test moderator confirmed this as well. Tests that ran over two minutes were deemed to have timed out and the timing was coded as 120 seconds.

After the 8 tests, the subjects were asked to assess the ease of finding the goals. Three questions asked about the general difficulty independent of interface, and the difficulty with each interface. We used a 5 point Likert scale from very difficult to very easy. We also asked for general comments about the differences in the interfaces.

Finally, after the formal test, we presented the more complete Street Slide interface including street names and the mini-map and allowed the subject to freely roam the streets. We encouraged them to make any general comments about what they found confusing or good about the interface for use in any re-design.

7.2 Study Results

Familiarity: Only four of the subjects expressed a high degree of familiarity with downtown Seattle. Six subjects were already familiar with Google Street View. None had used Street Slide before.

Ease of Finding the Goal: There were significant differences in the subjective assessment of the difficulty of finding the goal between the two interfaces. On the 5 point scale, only 5 subjects described finding the goal either easy of very easy with GSV. In contrast, 18 of the 20 subjects found it easy or very easy with SSL. See Figure 7.

Timings: The timings confirm the subjective assessments. Figure 7 shows the mean times and standard deviations as they relate to the distance to the goal from the starting point, and separated by interface. The tests timed out after two minutes 9 out of 80 times with the GSV interface and never with SSL. Interestingly, both interfaces showed a roughly linear relationship between the distance and mean time to goal. At a distance of 100 meters (about one block between the start and the goal), users of SSL had a mean time about 2.5 times faster than with GSV (mean of 20 seconds vs. 51 seconds), and 3 times faster at 200 meters (mean of 24 seconds vs. 75 seconds).

An analysis of variances (ANOVA) tested for significant differences between timing means as they relate to interfaces, distances and interface interacted with distance. The results were for interface, ($F(1, 59) = 57.88$, $p < 0.001$), for distance ($F(3, 57) = 8.65$, $p < 0.001$), and for the interaction ($F(3, 57) = 2.71$, $p = 0.047$). Thus all three were significant with the least significant, the interaction term, still about a 95% confidence level.

We also ran a linear regression for time to the goal, controlling for the familiarity with Seattle and with Google Street View, as well as the two interfaces, distance to the goal, and an interaction between interface and distance, in order to assess the trend in the growths of time with distance and between interfaces. The regression also took into account possible correlation of the timing tests by subject. Statistically significant effects were found for all variables: interface, distance, and the familiarity terms. All t-tests had $p < 0.01$ except familiarity with GSV which was still significant with $p = 0.044$. The results suggest that users of SSL are 17 seconds faster holding distance and familiarity constant. The tests with GSV suggest a trend towards increasing time with distance of 0.24 extra seconds for each additional meter (approximately 24 extra seconds per city block) while the SSL trend was just one third of this, at 0.08 seconds with every added meter of distance, again holding everything else constant. The coefficient for familiarity with Seattle was -17 seconds, while familiarity with GSV was -10 seconds.

General Comments: We received valuable feedback in the form of general comments. In comparing GSV to SSL, a number of subjects found an initial confusion in SSL but then found it easier after the first minute. In contrast, many found GSV easy at first, but then became frustrated with the navigation consisting of discrete jumps down the street. At the same time, many were frustrated that they could not travel "down the street" when in bubble mode in SSL.

In the last part of the session, we received many more comments on Street Slide. Some users were at first confused by the left/right turn signs for transferring the view to a crossing street. This is due to the conflict between facing sideways and thus looking down the crossing street with the need to ""turn" to get "onto" that same street. The mini-map was liked by many but they missed the lack of a compass heading as the map turned. Some subjects found the zoom-in, spin, zoom out aspect of the turns disorienting. Interestingly, there was not a single comment related to the alignment and/or rendering.

Study Discussion: The results make it clear that, at least for the given task, Street Slide was a more efficient interface. The general comments also indicated a preference from most for SSL as a general browsing interface for street side imagery. We are already working on changing the interface based on the additional comments. The fact that we only had a single type of task may also mean that SSL is only better at this particular task. We hope to conduct more tests for other tasks.

8 Discussion and Future Work

We have presented Street Slide, which (as can be seen in the accompanying video) provides an interactive means to efficiently explore street level imagery. Smoothly transitioning between panning and zooming within bubbles and translating across a multi-perspective panorama of the street sides provides a seamless way to gain both an overview for navigating larger distances as well as the ability to zoom into details at any point along the road. We presented a novel dynamic construction of multi-perspective panoramas, which provides a sense of physical motion as one moves along a street and

Figure 8: *We have begun porting Street Slide to the iPhone.*

zooms out to get an overview. Annotations and mini-maps also aid in the navigation.

The multi-perspective panorama constructions described in Section 4 are not without artifacts. In fact, they cannot be artifact free. Our current constructions have an operating range that covers most urban cores quite well; i.e., streets in which many of the facades form an approximate planar surface. We are actively working to extend this range to any street by applying vision techniques to first reconstruct the geometry and then finding appropriate optimization methods to avoid most artifacts.

Recent surveys have found that people are using mapping applications more on mobile devices than on fixed platforms. It makes sense that mapping applications can have their largest value when used in situ. To that end we have begun to implement Street Slide on a mobile platform. Figure 8 shows a snapshot of the application on an Apple iPhone.

We currently have processed about 2400 panoramas covering about 4 kilometers on 6 streets with 8 intersections. The annotations such as street names and numbers and store names have been entered manually so far. We are currently integrating our viewer into a larger database containing millions of bubbles of street level imagery, which will inevitably raise many new challenges. As the system scales up, the annotations will be drawn directly from a geographic database. Many other user supplied annotations can be imagined such as allowing users to add comments tied to specific locations while browsing in Street Slide. We hope that this work points the way to new modalities for remotely exploring the full richness of visual imagery from around the world.

References

AGARWALA, A., AGRAWALA, M., COHEN, M., SALESIN, D., AND SZELISKI, R. 2006. Photographing long scenes with multi-viewpoint panoramas. *ACM Transactions on Graphics 25*, 3 (August), 853–861.

BUEHLER, C., BOSSE, M., MCMILLAN, L., GORTLER, S., AND COHEN, M. 2001. Unstructured lumigraph rendering. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 425–432.

CHEN, S. E. 1995. QuickTime VR – an image-based approach to virtual environment navigation. *Computer Graphics (SIGGRAPH'95)* (August), 29–38.

GORTLER, S. J., GRZESZCZUK, R., SZELISKI, R., AND COHEN, M. F. 1996. The Lumigraph. In *Computer Graphics Proceedings, Annual Conference Series*, ACM SIGGRAPH, Proc. SIGGRAPH'96 (New Orleans), 43–54.

GUPTA, R., AND HARTLEY, R. 1997. Linear pushbroom cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence 19*, 9 (September), 963–975.

IGARASHI, T., AND HINCKLEY, K. 2000. Speed-dependent automatic zooming for browsing large documents. In *UIST '00: Proceedings of the 13th annual ACM symposium on User interface software and technology*, ACM, New York, NY, USA, 139–148.

KOPF, J., UYTTENDAELE, M., DEUSSEN, O., AND COHEN, M. F. 2007. Capturing and viewing gigapixel images. *ACM Transactions on Graphics 26*, 3 (August).

KWATRA, V., SCHÖDL, A., ESSA, I., TURK, G., AND BOBICK, A. 2003. Graphcut textures: Image and video synthesis using graph cuts. *ACM Transactions on Graphics 22*, 3 (July), 277–286.

LEVOY, M., AND HANRAHAN, P. 1996. Light field rendering. In *Computer Graphics Proceedings, Annual Conference Series*, ACM SIGGRAPH, Proc. SIGGRAPH'96 (New Orleans), 31–42.

LIPPMAN, A. 1980. Movie maps: An application of the optical videodisc to computer graphics. *Computer Graphics (SIGGRAPH'80) 14*, 3 (July), 32–43.

MCMILLAN, L., AND BISHOP, G. 1995. Plenoptic modeling: An image-based rendering system. *Computer Graphics (SIGGRAPH'95)* (August), 39–46.

PELEG, S., ROUSSO, B., RAV-ACHA, A., AND ZOMET, A. 2000. Mosaicing on adaptive manifolds. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22*, 10 (October), 1144–1154.

PELEG, R., BEN-EZRA, M., AND PRITCH, Y. 2001. Omnistereo: Panoramic stereo imaging. *IEEE Transactions on Pattern Analysis and Machine Intelligence 23*, 3 (March), 279–290.

RADEMACHER, P., AND BISHOP, G. 1998. Multiple-center-of-projection images. In *Computer Graphics Proceedings, Annual Conference Series*, ACM SIGGRAPH, Proc. SIGGRAPH'98 (Orlando), 199–206.

RAV-ACHA, A., ENGEL, G., AND PELEG, S. 2008. Minimal aspect distortion (MAD) mosaicing of long scenes. *International Journal of Computer Vision 78*, 2-3 (July), 187–206.

ROMÁN, A., AND LENSCH, H. P. A. 2006. Automatic multiperspective images. In *Eurographics Symposium on Rendering*, 83–92.

ROMÁN, A., GARG, G., AND LEVOY, M. 2004. Interactive design of multi-perspective images for visualizing urban landscapes. In *IEEE Visualization 2004*, 537–544.

SNAVELY, N., SEITZ, S. M., AND SZELISKI, R. 2006. Photo tourism: Exploring photo collections in 3D. *ACM Transactions on Graphics 25*, 3 (August), 835–846.

SZELISKI, R. 2006. Image alignment and stitching: A tutorial. *Foundations and Trends in Computer Graphics and Computer Vision 2*, 1 (December).

UYTTENDAELE, M., CRIMINISI, A., KANG, S. B., WINDER, S., HARTLEY, R., AND SZELISKI, R. 2004. Image-based interactive exploration of real-world environments. *IEEE Computer Graphics and Applications 24*, 3 (May/June), 52–63.

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

ZOMET, A., FELDMAN, D., PELEG, S., AND WEINSHALL, D. 2003. Mosaicing new views: The crossed-slits projection. *IEEE Transaction on PAMI* (June), 741–754.