Erschienen in: ACM Transactions on Graphics ; 26 (2007), 3. - 93
https://dx.doi.org/10.1145/1239451.1239544

Capturing and Viewing Gigapixel Images

Johannes Kopf
University of Konstanz

Matt Uyttendaele
Microsoft Research

Oliver Deussen
University of Konstanz

Michael F. Cohen
Microsoft Research

Figure 1: Three views of a 1.5 Gigapixel image.

Abstract

We present a system to capture and view "Gigapixel images": very high resolution, high dynamic range, and wide angle imagery consisting of several billion pixels each. A specialized camera mount, in combination with an automated pipeline for alignment, exposure compensation, and stitching, provide the means to acquire Gigapixel images with a standard camera and lens. More importantly, our novel viewer enables exploration of such images at interactive rates over a network, while dynamically and smoothly interpolating the projection between perspective and curved projections, and simultaneously modifying the tone-mapping to ensure an optimal view of the portion of the scene being viewed.

1 Introduction

One normally thinks of an image as something one can print or display on a screen and then stand back from and view. However, as the resolution (i.e., the number of distinct pixels), the field of view (FOV), and/or the dynamic range of the image exceed some threshold, it is no longer possible to view the image on a screen in its entirety in any single representation. The resolution must be reduced, the field of view must be narrowed or the projection geometry altered, and/or the dynamic range must be mapped to one that can be displayed.

In such cases, experiencing the full content of the image is only possible by coupling the data with an appropriate viewer. In the case of very high resolution images, the ability to zoom and pan allows the user to fully explore the image, while very wide angle imagery is often viewed best with a panoramic browser. A similar argument calls for a specialized viewer for images with a very high dynamic range. We present a novel viewing paradigm for high resolution, wide angle, and/or high dynamic range (HDR) imagery.

We demonstrate a novel viewer designed for smooth real-time viewing of the panorama that dynamically adapts to both the requested FOV and the content of the sub-image being viewed. In particular, we demonstrate a system that smoothly adjusts the mapping of rays to pixels between a perspective projection for narrow fields of view and a cylindrical or spherical projection as the field of view widens (zooms out). It should be noted that this viewer also provides superior methods for viewing standard resolution panoramas (e.g., QuicktimeVR [Chen 1995]). The QuicktimeVR viewer only supports perspective projection which produces distortions and swimming artifacts for wide angle views. Smoothly and dynamically varying the projection between perspective and curved projections avoids such artifacts and allows the full FOV to be displayed.

In addition to modifying the projection, the viewer also dynamically adapts the tone mapping of the sub-image being viewed based on its local histogram. Dynamic modification of the scale and bias brightens dark regions and darkens light regions, while increasing the contrast in low contrast areas of the image. This approach effectively performs haze removal on-the-fly when viewing distant areas of scene, while leaving the haze intact to serve as a depth cue when viewing the overall panorama.

Recently, a number of image sources for very high resolution images have been developed. These images are acquired either with a unique large film back camera such as with Graham Flint's Gigapxl Project (see http://www.gigapxl.org), or by capturing multiple digital images and stitching them together. Satellite imagery of the earth (e.g., NASA's blue marble images, Google Earth, Microsoft's maps.live) forms the basis for other very large virtual images. We demonstrate a new camera mount to acquire very high resolution panoramas with a standard digital SLR camera and long lens.

The specific contributions of our work related to image acquisition and processing include a new hardware design point for acquiring very large HDR panoramas, and a complete processing pipeline from RAW image data for correcting, stitching and exposure compensating hundreds of input photographs with widely varying exposures into a consistent Gigapixel HDR panorama.

2 Capturing BIG Pictures

2.1 Related Work

Capturing panoramic images is almost as old as photography itself. Rotating slit scan cameras were used to construct panoramas one vertical stripe at a time. In the quest for ever higher resolution in the film domain, we have recently seen Graham Flint's wonderful one-of-a-kind large film back camera capable of resolving a billion distinct points or more in a single shot (http://www.gigapxl.org). In the digital domain, Spheron (see http://www.spheron.com) markets a digital panoramic camera able to capture high resolution and high dynamic range imagery. In the past few months, we have also seen a few Gigapixel images appearing on the web constructed from a large number of individual images, for example, NASA's blue marble project (http://earthobservatory.nasa.gov). Xrez (http://www.xrez.com) and others have used standard digital SLRs and automated mounts such as PixOrb (see http://www.peaceriverstudios.com/pixorb) to capture a large number of individual images and stitch them into a panorama [Szeliski 2006]. We have taken a similar approach. However, our stitching method allows hundreds of images taken with varying exposures to be automatically combined into a Gigapixel, HDR, (and optionally tone mapped) panorama. We know of no other automated system with this capability.

2.2 A New Panoramic Camera Mount

Capturing and stitching hundreds of images imposed a number of requirements on the design of the system.

1. A long telephoto lens is required to achieve the desired resolution.
2. Images should be captured as fast as possible to minimize issues of moving elements in the scene and changing lighting conditions.
3. Images should be captured on as regular a grid as possible to simplify the stitching process.
4. The capture should allow for wide variations in dynamic range of the scene.

Figure 2: The Meade LX200 adapted to shoot Gigapixel images.

We began with the PixOrb mount, which resulted in the Seattle skyline in Figure 12. Unfortunately, this system was not designed to carry the heavy payload of a long lens. The long lens and camera on these platforms created vibrations long after each move to a new position and additional vibration was induced by wind and the mirror lockup action in our digital SLR camera. Waiting for vibrations to subside fought against requirement #2. Fortunately, we were able to leverage a platform designed by the astronomy community. We bought a Meade LX200 telescope mount, removed the optical tube assembly, and built a harness that allowed for the precise placement of the nodal point of our camera and lens (see Figure 2). The LX200 met the first 3 requirements. The result is that with a 400mm lens our system can shoot at about 5 seconds per picture.

For our imaging system we used a 16 Megapixel Canon 1DS Mark II or an 8 Megapixel Canon 20D with a 100-400mm zoom lens. The wide dynamic range of most scenes was captured by fixing the aperture of the lens but allowing the autoexposure to set the shutter speed. Since the overall goal is to allow users to interactively view the final result, the images should be as sharp as possible down to the pixel level. Lenses have an optimal aperture at which they are sharpest. An f11 aperture for our lens is close to the optimal and has a reasonable depth of field. The images were stored by the camera in raw format. The user indicates which portion of the panoramic sphere to capture. Everything else is automatic. Given this user input, and a desired 16% overlap between images, a script generates the sequence of pan and pitch angles and drives the motors to point the camera. We pan the scene in vertical scanline order to minimize any illumination changes between adjacent vertical columns. The high resolution images we have captured were constructed from between 250 and 800 individual shots taken over time spans of between 30 and 90 minutes.

2.3 Processing the Shots into a BIG Picture

Assembling multiple images into a seamless panorama is possible using several commercial products. However, in order to create a high dynamic range Gigapixel image from a large set of pictures that have varying exposures, several technical challenges must be overcome. We have not seen any other system described that can deal both with the scale and the dynamic range issues. In this section, we briefly describe our geometric and radiometric alignment pipeline, shown in Figure 3.

The first phase of processing is to produce radiance values for each of the input images. We work entirely in the linear domain. This means that, unlike other systems [Debevec and Malik 1997; Mitsunaga and Nayar 1999], there is no need to compute the non-linear transfer function of the camera. Another difference is that these systems describe a fixed camera exposure bracketing a scene. Our current system is a rotating camera capturing the best exposure for each camera position. The result is that for any small field of view we don't capture the full dynamic range as the other systems would. However, the overall composite image will generally cover a very large dynamic range.

The radiometric processing begins by demosaicing the RAW sensor values to produce an (RGB) triplet per pixel. Lens vignetting is then removed by dividing the pixels with a vignette adjustment map. The vignette map for a specific lens and aperture is produced by taking a picture of the very uniform light field generated by an *integrating sphere*. The only manual step in the entire processing pipeline is to select, a neutral (gray value) point in the scene. This defines the color balance for the entire scene. We use Bradford chromatic adaptation [Lam 1985] to compute the transform from the sensor neutral point to an scRGB neutral point. Since all processing is done in the linear domain, we convert to radiance by dividing out the exposure value (in this case simply the shutter speed) of each image.

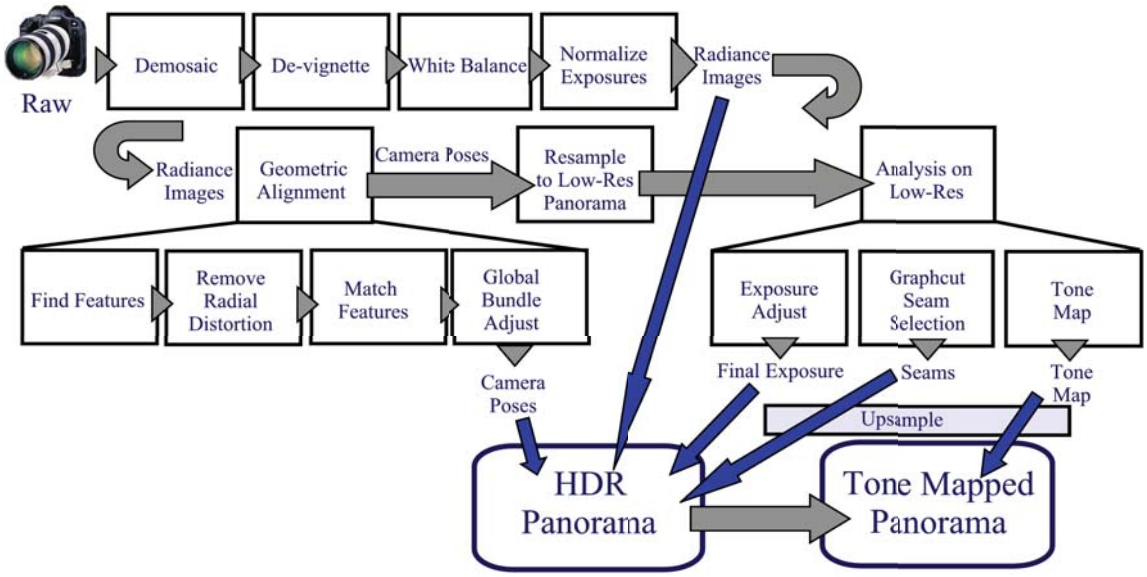


Figure 3: The processing pipeline

The radiance images provide the inputs to the next phase, geometric alignment. Even though the rig moves very precisely its positioning is not accurate enough to use as the true camera pose. Uncertainty in the pose comes from three factors: (1) when the camera is mounted to the rig there is an unknown rotation between the mount and the camera, (2) the uncertainty of the rig's pan/pitch positions is greater than the angular spacing of a pixel, and (3) the focal length is not known to the needed precision. Thus a geometric alignment step is needed in order to compute the pose.

We use a feature based alignment technique [Lowe 2004; Brown et al. 2005]. For each captured image we extract multi-scale oriented patches (MOPS) features [Brown et al. 2005].[1] The first alignment step is to compute the radial distortion. For this, our system automatically finds a small sub-set of the images that have good spatial distribution of features. We then use a Levenberg-Marquardt optimization on the features to simultaneously solve for a common 4th order radial distortion polynomial and an independent homography for each image in the sub-set.

The MOPS features are next mapped through the radial distortion polynomial and a pose is computed. Since our system uses a rotating camera at a fixed position, the pose we solve for is a single common focal length across all images, and a 3D orientation per image. Given the approximate pose from the rig, for each image we search for feature matches only in the 8 images known to overlap it, and within these, only within the known positioning tolerances of the rig. Next, for each pair of adjacent images, a random sample consensus (RANSAC) step eliminates outliers from the putative matches [Fischler and Bolles 1981]. Finally, a bundle adjustment step computes the pose from the valid feature matches. The large number of images and even larger number of features means we needed to carefully craft the bundle adjuster. The formulated problem is sparse, because only adjacent images have overlapping features. Our bundle adjuster employs Levenberg-Marquardt optimization, the inner-loop of which uses a sparse matrix solver where the matrix is stored in "skyline" form [Szeliski and Kang 1994].

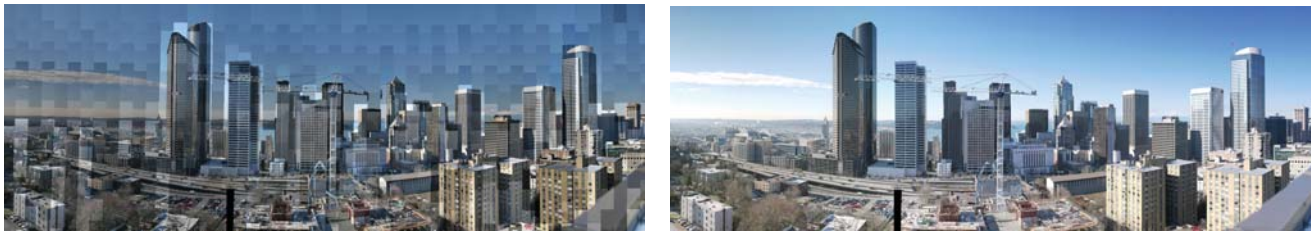
[1]For featureless images such as sky, we simply trust the pose reported by the rig.

Once the pose is computed we assemble a composite image. At first, a low resolution proxy composite is created. The proxy is used for several steps that would not be tractable on the full resolution result. The first of these steps is to refine the exposure values. The simple division by the shutter speed doesn't generate exactly matching radiance values in corresponding images due to slight errors in reported shutter speeds coupled with minor illumination changes across the panorama. Better radiometric alignment is achieved by solving for exposures as described in [Eden et al. 2006].

After the final radiometric and geometric alignment there may still be mismatches between input images due to moving objects or lighting changes. To minimize artifacts, we use the proxy to compute a labelling of which input pixels to use in the final result. This is done by solving a graph-cut problem as described in Agarwala *et al.* [2004]. Note that even the proxy composite represents a very large graph. However the largest graph that we need to create is only the size of one input image. We iteratively solve a series of binary alpha expansion multi-label problems over the footprint of a each image. Thus the label set includes only the current image and the overlapping regions of its eight neighbors.

If a tone-mapped result is desired then the proxy is also used to perform the required analysis. Any tone-mapper could be used here. In this work we used the interactive tone mapper of Lischinski *et al.* [2006].

At this point we have everything necessary to produce the full resolution HDR composite image. We expand the low resolution labelling to the full resolution using a joint bilateral up-sampling method [Kopf et al. 2007]. Then the camera poses, radial distortion parameters and the desired output mapping, for example perspective for a narrow FOV, or cylindrical or spherical for wider fields of view, are used to warp the input radiance images into the output composite. In practice, the final warp is performed in a single step. This avoids multiple resamplings from the original images. If a tone mapped output is desired, then the proxy exposure map is also expanded using the joint bilateral up-sampling method [Kopf et al. 2007]. An example panorama from 750 input images without radiometric alignment is shown in Figure 4(left). Figure 4(right) shows the result after alignment and tone mapping.



(a) (b)

Figure 4: Seattle Skyline (the black bars occurred when the camera battery died. The width of the bar gives a sense of the width (2600 pixels) of one shot.) (a) Captured images with correct geometric alignment but no radiometric alignment. (b) Results after radiometric alignment and tone mapping.

Previous composition techniques have used blending (e.g., Laplacian [Burt and Adelson 1983] or Poisson [Pérez et al. 2003; Agarwala et al. 2004]) to remove the differences between the images in the composite. However blending across widely varying exposures is a poor tone-mapping operator. Here, we have shown that by capturing images in the linear domain and using the radiometric alignment and composition techniques from [Eden et al. 2006] a seamless HDR image can be constructed with no blending at all. This can be further processed by any of the recent tone-mapping techniques to produce a final result.

3 Display of High Resolution Panoramic Imagery

A specialized viewer is required for our imagery for a number of reasons. First, the image contains 3 orders of magnitude more data than can be displayed on a typical monitor. The approximately 1 million pixels on a monitor is only $1/1000^{th}$ the resolution, the high dynamic range (if not pre-tone-mapped) is double the bit depth, and wide angle imagery can only be displayed in full when mapped to curved surfaces such as a cylindrical or spherical projection. Second, such large images represent a 2D space for exploration and thus contain an infinite number of smaller images which the user should be able to discover. Finally, in addition to allowing the selection a subset of the image to view, the viewer should dynamically adapt to the content of the subset.

3.1 Related Work

QuicktimeVR [Chen 1995] (and other similar viewers) deal with wide angle imagery such as 360 degree images by displaying only a small portion of the field of view through a perspective projection. This looks fine for small FOVs but rapidly becomes distorted as the FOV widens and reaches a singularity at 180 degrees. Recent Flash-based viewers such as Zoomify (http://www.zoomify.com/) provide a zooming and panning interface (like that for tiled maps, e.g., Google Maps) for very large images by directly mapping portions of an image in its native projection to the display but do not perform on-the-fly transformations to adapt to the momentary image content.

Real-time tone mapping of high dynamic range imagery has begun to appear in games such as Half-Life 2 that modify the tone map on-the-fly to simulate effects such as the delayed adaptation as one leaves or enters a dark tunnel from the sunlight.

3.2 The BIG Picture Viewer

Our goals for viewing very large images are to:

- download image data only as needed to minimize bandwidth while maintaining consistent image quality,
- display the momentary FOV with a projection that best fits that FOV and smoothly varies the projection as the FOV changes, and
- adapt the tone mapping to the average luminance and contrast of the current image content.

We are aware of no viewer that fulfills these requirements.

The user interface for the viewer is quite simple. There are controls for panning and zooming. Panning is mapped to mouse motion with the left button down, and zooming is mapped to three actions (vertical motion with right mouse down, the scroll wheel, and double clicking the left mouse button). The pan position and zoom level defines the portion of the image to be displayed.

3.2.1 Fetching Image Tiles

Our images are stored as a pyramid of 256^2 pixel tiles. A typical Gigapixel image may contain 9 levels of the pyramid. To maintain image quality we want neighboring pixels on the screen displayed from levels of the pyramid that are not more than 1 level apart. To achieve this and minimize bandwidth, one thread uses the following strategy for bringing tiles into the cache.

Beginning at the top of the pyramid and continuing down to the level of the current view, fetch any tiles overlapping the current view. When all these tiles are cached and the requested view has not changed, work recursively up the pyramid again, fetching immediate neighbors to the tiles in the view. When the cache is full, tiles which are furthest away from the current view are removed.

The first step in the rendering thread for each frame is to render an unwarped/unaltered rectangular image in its native projective mapping to an offscreen texture buffer that contains, at a minimum, all pixels for the current view. All requested pixels are rendered from the highest resolution tile available.

Then, based on the original mapping of the image (perspective, cylindrical, spherical) and the momentary desired projection, the unwarped buffer is remapped and displayed to the screen as described next.

3.2.2 Rendering the Image with Adaptive Projection

Images of approximately 60 degrees or less are well suited to viewing through a perspective projection. This maintains straight lines as straight and for the most part corresponds to what we perceive as we look around. As the field of view of the image on the monitor

greatly exceeds the actual field of view subtended by the monitor, perspective projections begin to exhibit distortions. This typically starts by about an 80 degree FOV and only worsens until a singularity is reached at 180 degrees FOV. To display such wide angle imagery we turn to other *curved* projections such as cylindrical and spherical mappings which are then "unwrapped" onto the flat monitor. While these mappings allow one to see a wide field of view, they incur distortions of their own (albeit less than perspective for very wide angles) such as mapping straight lines to curves (see Figure 5). Figure 6(b)-(c) show mappings from screen position to the angle away from straight ahead in the world for various FOVs for perspective and curved projections. Note that both perspective and cylindrical mappings are quite similar at small FOVs but diverge significantly for larger FOVs. This is not surprising since a small portion of a cylinder mimics a planar patch.

Since we wish to be able to zoom in and out to any FOV on our imagery, while providing an optimal projection, our viewer smoothly adapts the projection from perspective for small FOVs to curved for large FOVs.

Through experimentation we learned that it is desirable to have the user zoom control correlate directly to the image expansion in the center of the screen. Beyond that, all parts of the screen should expand monotonically and as smoothly as possible. We also have the constraint that the projection should be fully curved at fields of view greater than some maximum (colored blue in Figure 7 a-c), and perspective below some minimum (colored red). Based on these criteria we have developed a novel way to interpolate between perspective and cylindrical/spherical projections in this manner.

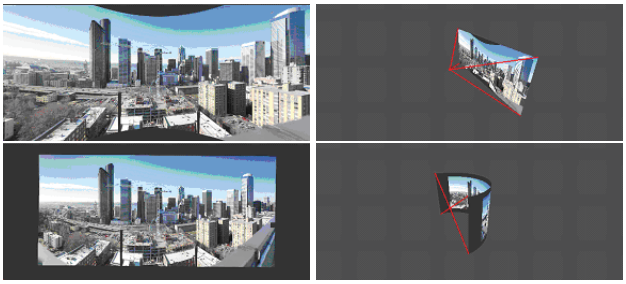


Figure 5: Perspective projection (top row) vs. cylindrical projection (bottom row). The left column shows the projected image, the right column visualizes the projective surface in world space.

To accomplish this, we establish a world coordinate system with the camera sitting at the origin. The directions from the camera into the world are parameterized by two angles (θ, ϕ), where θ is the longitude on the surrounding sphere and ϕ is the latitude. Let $\theta = \phi = 0$ be looking straight ahead along the positive z axis. World x is to the right and world y represents the "up" vector.

A 2D virtual screen or *projective surface* (described shortly) is established in this coordinate system parameterized by x_s and y_s. Rendering a pixel at (x_s, y_s) proceeds in two steps:

1. Compute (θ, ϕ) for each pixel position (x_s, y_s), (screen to world transformation).
2. Compute the corresponding texture coordinates based on how the underlying image is stored, (world to texture transformation).

By doing so, we decouple the viewing projection (how the world is viewed) from the texture projection (how the image is stored). There are two ways to implement this procedure efficiently: using pixel-shaders for newer graphics hardware, or vertex arrays for fixed function graphics hardware.

For the pixel shader implementation we render only a single screen filling quad. The pixel shader gets automatically executed for each pixel, and computes steps 1 and 2. For the vertex array implementation we tessellate the screen into a fine grid of vertices. To render an image, we compute the texture coordinates for each vertex, and then render the quads defined by the vertex array, implicitly interpolating texture coordinates.

3.2.3 The Projective Surface

Smoothly interpolating between perspective and curved projections during panning and zooming is realized by bending, scaling, and rotating the projective surface within the world coordinate system.

For a perspective projection, the projective surface is flat in world space (Figure 5, top row). We will consider the surface situated at a unit distance along the z axis. To produce a cylindrical projection, we bend the surface to form a cylinder (Figure 5, bottom row). Without loss of generality, the fully bent cylinder has unit radius, with the viewpoint in the center, thus the world point $(0,0,1)$ remains stationary at the center of the perspective and cylindrical projective surfaces. (For simplicity, we will continue the discussion considering transformations between perspective and cylindrical, thus we can drop the y coordinate. Spherical projections follow the same logic in both directions.)

Interpolating between cylindrical and perspective projections is accomplished by unbending the projective surface from a cylinder to a plane. This can be viewed as increasing the radius of the cylinder while keeping the viewpoint at unit distance away from the cylinder wall (Figure 6a).

Ignoring the rotation for now, the point $(x_a = 0, z_a = 1)$ is always on the cylinder (see Figure 6a), and is projected to the center of the screen. To compute the cylinder parameters we need one more point on the surface. Let $x_b = x_a + \cos\alpha, z_b = z_a - \sin\alpha$ be that second point, then the cylinder parameters can be computed as:

$$r_c = \frac{1}{2\sin\alpha}, \quad x_c = 0, \quad z_c = 1 - r_c. \tag{1}$$

The parameter $\alpha \in [0..\sin^{-1} 0.5]$ is used to blend between perspective projection ($\alpha = 0$) and cylindrical projection ($\alpha = \sin^{-1} 0.5$).

A screen coordinate x_s can now be projected on the cylinder by

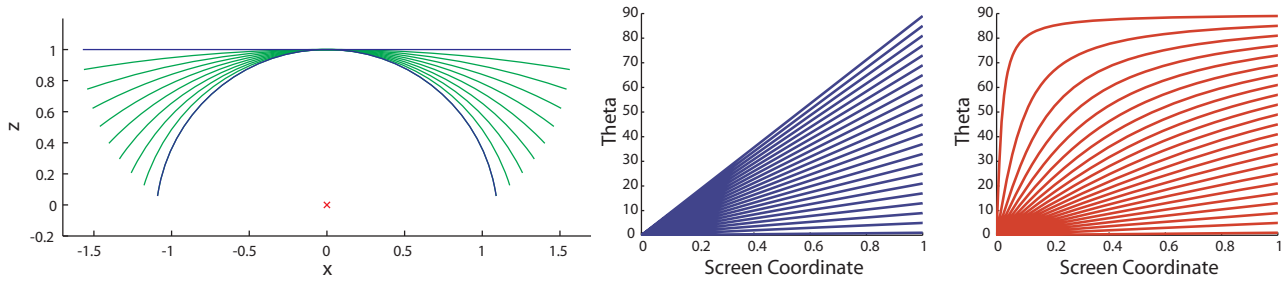
$$\begin{aligned} x_p &= x_c + r_c \sin\tfrac{x_s}{r_c}, \\ z_p &= z_c + r_c \cos\tfrac{x_s}{r_c} + (1 - r_c). \end{aligned} \tag{2}$$

Note that we hit a singularity for $\alpha = 0$. We treat this as a special case in our implementation and compute the perspective projection in this case directly as $x_p = x_s, \; z_p = 1$.

Having the projected point we finally compute the angle:

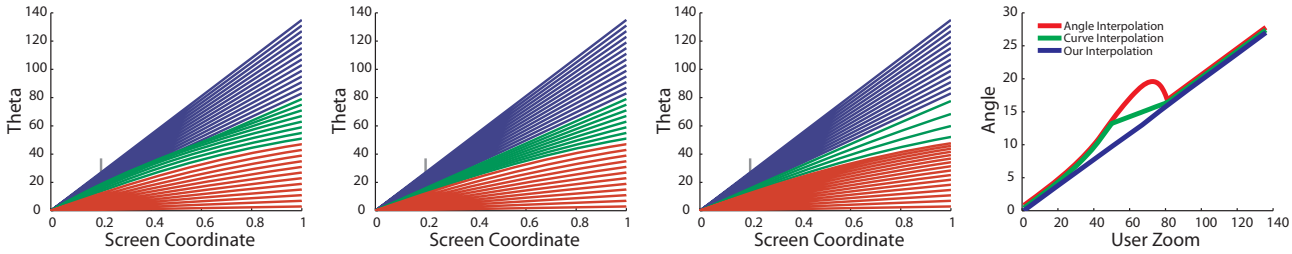
$$\theta = atan2(x_p, z_p). \tag{3}$$

We'll now extend the discussion to the vertical dimension and spherical projections. An advantage of our projection method is the ability to blend *independently* in the horizontal and vertical directions. The vertical angle ϕ is computed similarly to θ. Here each vertical strip on the screen has its own cylinder with centers on a straight line through point (x_p, z_p). A blending parameter



(a) (b) (c)

Figure 6: Interpolating between perspective and cylindrical projections. (a) Varying the parameter α moves between cylindrical and perspective projection by unbending the projective surface: Increasing the radius of the cylinder and moving it back along the z-axis to maintain a unit distance from the viewer we transform from a cylinder to a plane. (b)-(c) Projections: the diagrams show how screen position is mapped to angles for the center horizontal scanline for different fields of view (the angle at which the curve hits the edge of the screen). Only the right half of the screen is shown: (b) pure cylindrical and (c) pure perspective projections.



(a) (b) (c) (d)

Figure 7: Interpolation between perspective and cylindrical projection from the diagrams in Figure 6(b) and (c). Blue/red denote pure cylindrical/perspective, and green mixed projections. (a) Angle interpolation: Interpolating corresponding curves from perspective and cylindrical mappings for the desired FOV causes an undesired effect where the screen-center alternates between enlarging and shrinking during zooming. (b) Curve interpolation: Interpolating between the perspective curve at the start of the transition and the cylindrical curve for the end of the transition is better, however, now all interpolated (green) curves run through a very small corridor near the center of the screen. This causes the center of the screen to appear to stop moving during interpolation. (c) Our interpolation method. By construction, the center of the screen always moves at constant velocity during a zooming operation providing the best visual result. (d) A comparison of the three interpolation methods at a distance one fifth of the way from the center to the edge of the screen. Note that our method produces a linear mapping between the users specified zoom and the angle displayed at that point on the screen.

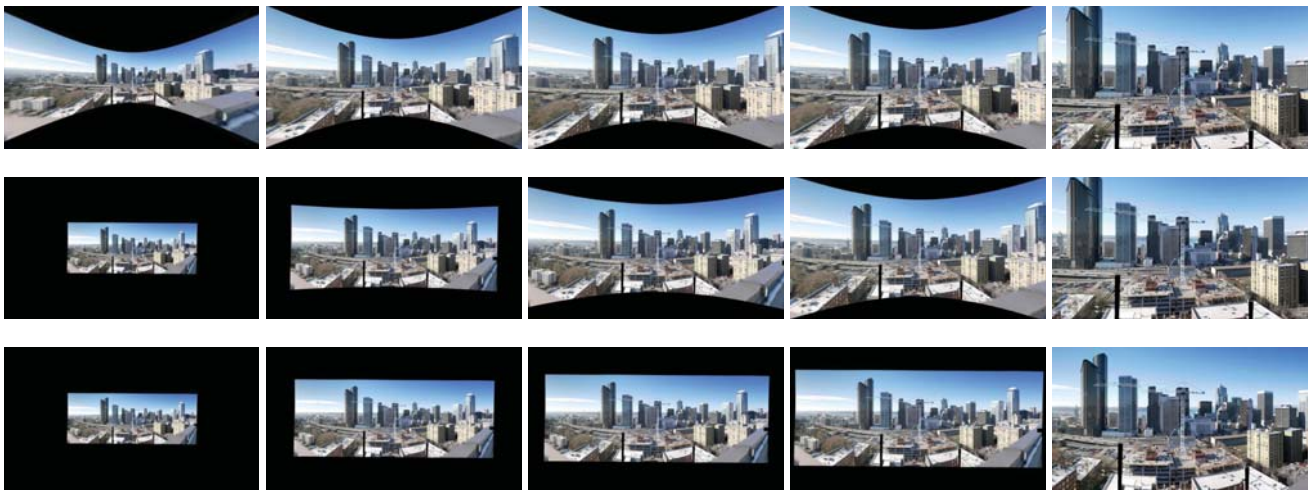


Figure 8: Zooming with perspective projection (top row), cylindrical projection (bottom row), and our adaptive projection (center row). The perspective projection exhibits strong distortion when zoomed out (left column), whereas it produces a natural image when zoomed in (right column). The cylindrical projection produces an undistorted image when zoomed out, but has a less natural appearance when zoomed in. Our adaptive projection combines the best of both worlds. It converges to cylindrical projection for zoomed-out views (left), and to perspective projection for zoom-ins (right).

$\beta \in [0..\sin^{-1}0.5]$ controls the radius of the vertical cylinders, similarly to α. This combination of two cylinders makes our projective surface essentially a torus, where α controls the outer radius and β the inner radius. Perspective and cylindrical projections are the special cases where the inner or outer radius hits infinity. A spherical projection occurs when the inner and outer radius are one.

We begin by computing a point that has unit distance in the normal direction from the surface: $x_d = x_p - \sin\frac{x_s}{r_c}$, $z_d = z_p - \cos\frac{x_s}{r_c}$. Then, the full projection of the screen coordinate x_s, y_s is computed by:

$$\begin{aligned} x'_p &= x_d + (x_p - x_d)\left(\cos\tfrac{y_s}{r_v} + (1 - r_v)\right), \\ y'_p &= r_v \sin\tfrac{y_s}{r_v}, \\ z'_p &= z_d + (z_p - z_d)\left(\cos\tfrac{y_s}{r_v} + (1 - r_v)\right), \end{aligned} \tag{4}$$

where $r_v = \frac{1}{2\sin\beta}$. Again, $\beta = 0$ needs to be treated as a special case, in which we compute the projection by $x'_p = x_p$, $y'_p = y_s$, $z'_p = z_p$.

Finally, the ϕ angle is given by:

$$\phi = atan2(y'_p, \sqrt{x'^2_p + z'^2_p}) \tag{5}$$

Zooming is done in this system by simply scaling the screen coordinates $(x'_s, y'_s) = (x_s v_{zoom}, y_s v_{zoom})$. Rotation is realized by rotating the projected points (x'_p, y'_p, z'_p) around the origin. The bending parameters α and β are set automatically in our viewer by applying a sigmoid function of the zoom:

$$\alpha = \frac{\sin^{-1}0.5}{1 + b_{scale}\exp(v_{zoom} - b_{mid})} \tag{6}$$

In this Equation b_{scale} and b_{mid} control the shape of the sigmoid function. Figure 9 illustrates the effect. We use $b_{scale} = 6$ and $b_{mid} = 1.74$ as default values.

An important point to note is the linear relationship between the user affordance to change the zoom value, v_{zoom}, and the scaling of the screen. This results in the uniform and consistent appearance during zooming unlike more naive approaches described in Figure 7(a)-(b).

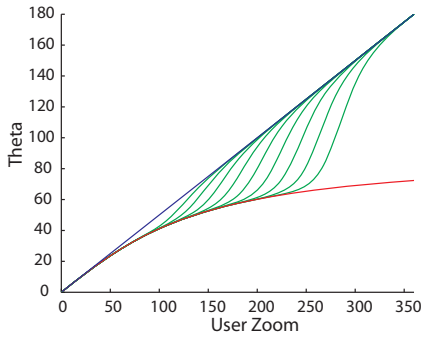


Figure 9: Relationship of user zooming and the field of view. The red line shows a pure perspective projection, while the blue line is for a pure cylindrical projection. The green lines illustrate our adaptive projection for different parameters of b_{mid}.

3.3 Dynamic Tone Mapping

Our panoramas are initially stored in high dynamic range in HD Photo format [HDP 2006]. Tone mapping is required to map the HDR image to the limited dynamic range of the monitor. A number of different operators have been tested, Fattal *et al.* [2002], Durand and Dorsey [2002] as well as Lischinski *et al.*[2006] create convincing results; however, none of the methods is able to produce results in real-time, especially not for Gigapixel images.

We first tried to use Reinhard's fast operator [2002] that can be applied in real-time. While Reinhard tried to find automatic settings in [Reinhard 2002] the results are not optimal, as the images still exhibit a grayish and washed out look.

Our viewer operates on either of two types of data. It can process the HDR images directly, or can operate on already quantized and compressed JPEG tiles for faster download times and less requirements on the client machine. In both cases an initial global tone mapping is performed. We use the tone mapper of Lischinski *et al.* [2006] to first create a high quality tone map in a pre-process. It is not feasible to create a carefully crafted tone map for every possible view of the image on-the-fly. Instead, we combine a single (possibly manually-crafted) global tone mapping with a fast interactive local histogram-based tone mapping.

The output of the global tone mapper is an image where the luminance is mapped to the interval $[0,1]$, where zero means totally black and one means pure white. This allows us to compute the *key* of the current view as the average luminance. The key indicates whether a scene is subjectively light, normal, or dark [Reinhard et al. 2002]. In addition to the key we also measure the *range* of luminance values for the current view.

We measure both the key and range based on the input luminance histogram of the pixels in the current view. The key, k_{in}, is determined as the average of the 99^{th} and 1^{st} percentiles, and the range, s_{in}, is the difference between the 99^{th} and 1^{st} luminance percentiles.

At each frame, we *stretch and bias* the histogram: we move the key of the histogram towards middle-gray to brighten dark regions or darken light regions. At the same time, we also stretch the histogram to enhance contrast.

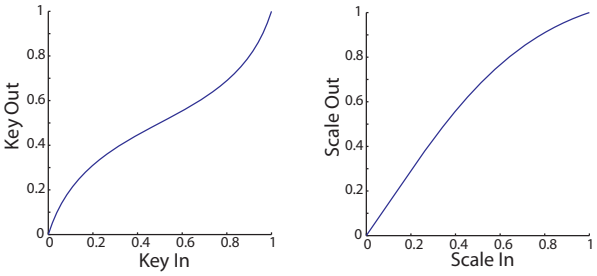


Figure 10: Histogram stretch and bias curves. Left: key bias; the curve always moves the key towards middle gray level. Right: scale stretch; the curve always enhances the scale slightly.

Figure 10 show the functions we use to stretch and bias the histogram. The key function is based on a tangent curve, and the scale function is a sigmoid. More specifically, the key curve is defined as follows:

$$k_{out} = \frac{1}{1 + \exp(-b \cdot (k_{in} + a))} \tag{7}$$

where $a = -0.1$ and $b = 10p_k$ to control the curve with a single parameter $p_k \in [0,1]$. If $p_k = 0$ then the key is unchanged, if $p_k = 1$

Figure 11: Different Tone Mappings. The manual tone mapping produces a great result for overviews of the full image, and respects the artistic tonal decisions of the photographer. However, when zooming into dark or bright spot of the photo one might want to enhance some details there. With Reinhard's global operator the resulting images often look a bit washed out and also the colors seem to shift a bit. In all three case, our operator produces a good result at interactive rates.

then the output key is always middle gray. In between, dark and light keys are smoothly brought towards middle gray based on their distance from middle gray.

The stretch curve is defined as:

$$s_{out} = 0.5 + c\ \tan(d \cdot (2s_{in} - 1)) \tag{8}$$

where again, c and d are chosen so that the curve can be controlled with a single parameter $p_s \in [0,1]$: $d = 0.5\pi \cdot \log(20p_s + 1)/log(21)$ and $c = \frac{1}{2\tan(d)}$.

Figure 11 shows comparison of three tone mappings, (a manually constructed global tone map, Reinhard's operator [Reinhard et al. 2002], and our dynamic tone mapping), applied to a wide angle view and three narrow fields of view. While the manual global tone map, by definition, results in a good wide angle image, local details are too bright or dark, or lack contrast due to the lack of local adaptation. Reinhard's operator tends towards create too dark and/or poor contrast. Our dynamic mapping, however, creates a good local solution at interactive rates.

3.3.1 Hysteresis

In fact, we do not stretch and bias the histogram to its optimal value immediately at each frame for two reasons. Small changes in the momentary pan and zoom can create large changes in the 1^{st} and 99^{th} percentiles in the luminance histogram. We want to avoid oscillations in tone adjustment. Also, just as in the way the human visual system slowly adapts to sudden luminance changes, we wish the tone adjustment to "adapt" to viewing changes. Thus, we add a *hysteresis* term that blends the currently computed key and stretch values with those used in the previous frame. Thus,

$$k_t^* = H * k_{out} + (1-H)k_{t-1}^* \tag{9}$$

$$s_t^* = H * s_{out} + (1-H)s_{t-1}^* \tag{10}$$

where H is adjustable but typically set to 0.1. And, finally the luminance of each pixel is

$$Y_{out} = s^*(Y_{in} - k_{in}) + k^* \tag{11}$$

4 Results

We have found exploring very large images to be great fun and a source of constant amazement at the details one discovers. The fluid nature of the interaction provides a natural feel to the exploration. The best way to evaluate the results is to try it (the viewer and a number of example Gigapixel images is available on the accompanying web site http://research.microsoft.com/ivm/HDView.htm).

Figures 1 and 12 show overviews and details of some images we captured. The images vary from 1.2 to 4.5 Gigapixels. The images took between 30 and 90 minutes to capture. Approximately 3-6 hours of total processing time was required to produce the final image pyramid. Total storage requirements for JPEG tiles is approximately 100 Megabytes per Gigapixel, and about twice that for HDR pyramids compressed with HD Photo.

Viewing the images over our internal network as well as the internet with a fast connection was very fluid with almost no visible delay in tile loading.

Watching others use our viewer has been gratifying. They have discovered many details we have missed even after our own hours of exploration.

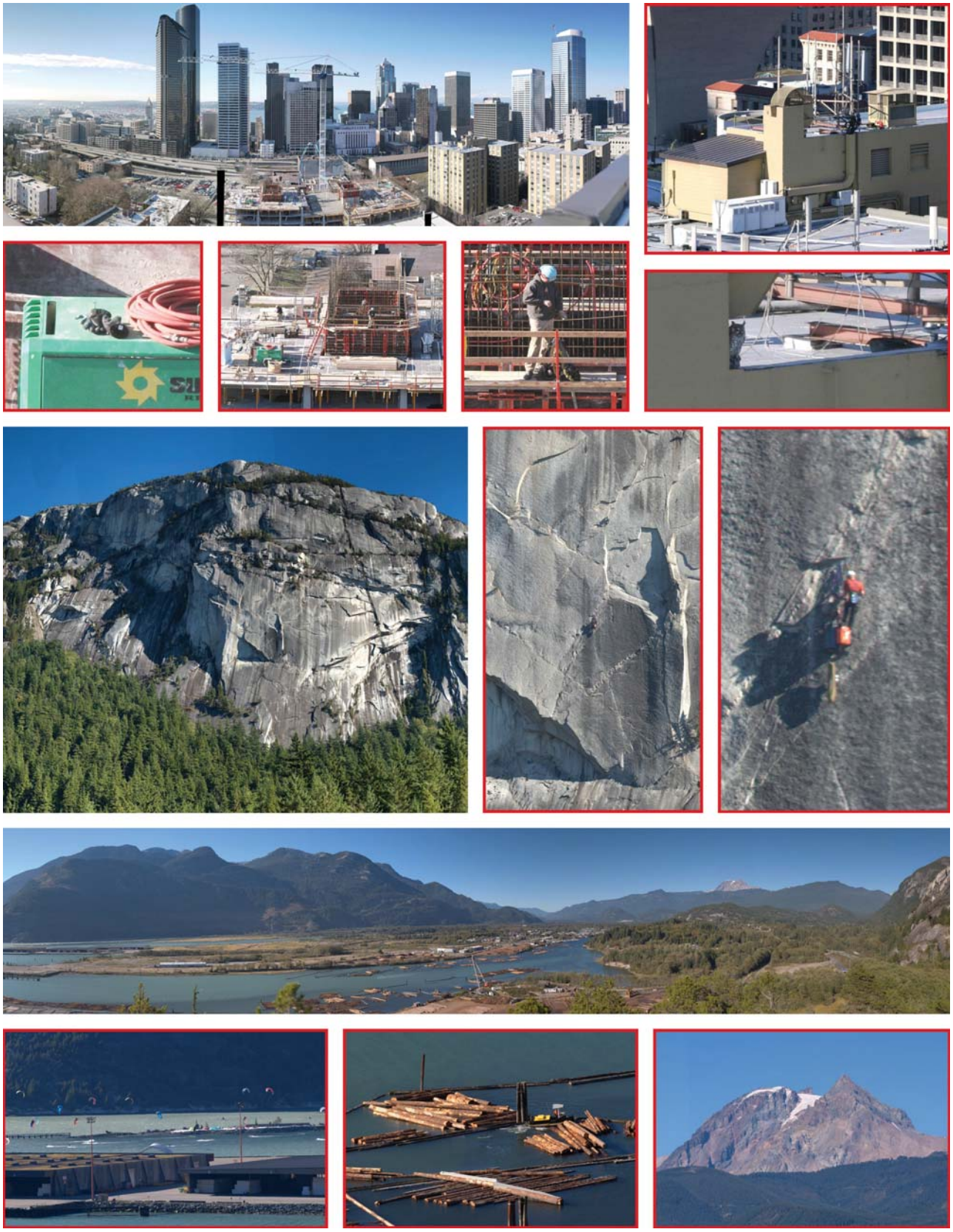


Figure 12: Exploring Gigapixel Images

5 Discussion and Conclusion

We have demonstrated a system for the creation, processing, and interactive display of images with very high resolution, high dynamic range and wide angle fields of view. A specialized capture device allows us to acquire images with nearly unlimited resolution. Efficient methods for the geometric alignment, radiometric alignment and tone-mapping automatically produce smooth, convincing results. The results can be displayed at interactive rates with our viewer that smoothly adapts both the projection and tone mapping.

That said, capturing gigapixel images by scanning the scene with a conventional camera does cause difficult issues. In particular, time is our main enemy in the capture process. Changing illumination makes radiometric alignment difficult. Large moving objects such as the crane in Figure 4 cannot be aligned. Other capture systems can help solve the time issue but at other costs. Large film format cameras are one of a kind and thus not accessible to most photographers, scanning cameras such as Panoscan and Spheron HD are faster but introduce smearing artifacts for moving objects. Camera arrays [Wilburn et al. 2005] may improve speed but at considerable cost. Clearly, there is room for further research and exploration in this space.

We have many items on our list for further work. Our next step is to capture multiple images per direction varying the exposure and focus. Although to date we capture a high dynamic range across the panorama, within each shot we rely on the auto exposure mode of the camera. Similarly, long lenses provide a very shallow depth of field, thus for scenes with both near and far objects we need to focus bracket as well. We will then rely on the work outlined in the Photomontage paper [Agarwala et al. 2004] to combine individual bracketed shots while stitching the full panorama.

Some other stitching issues that we would like to address are the slight radiometric mismatches visible in smooth regions such as sky and visible changes in the noise levels due to changing camera exposures. The first issue could be addressed by introducing a Poisson blending step. The second problem may be ameliorated by exposure bracketing to reduce noise, and by adding a term in the graph-cut selection step that favors less noisy pixels.

We are eager to develop applications using the Gigapixel images and viewer. Games (e.g., finding hidden objects, solving mysteries), embedding information (e.g., names of objects, stores, etc.), inserting sounds and/or video, and creating portals to other gigapixel images are among the ideas. Integration with online mapping applications can provide a new modality for exploring locations. We are also planning to develop a multi-resolution editor for such images to allow touch-up and insertion and deletion within the images.

We have, hopefully, also provided an answer to those who question why we would want more resolution than found in today's cameras. We hope these images and the associated viewer will inspire many others to acquire and display very high resolution images. We are certain they and others will have as much fun viewing them as we already have.

References

AGARWALA, A., DONTCHEVA, M., AGRAWALA, M., DRUCKER, S., COLBURN, A., CURLESS, B., SALESIN, D., AND COHEN, M. 2004. Interactive digital photomontage. *ACM Transactions on Graphics 23*, 3 (Proc. SIGGRAPH 2004), 294–302.

BROWN, M., SZELISKI, R., AND WINDER, S. 2005. Multi-image matching using multi-scale oriented patches. *Proceedings of CVPR 2005*, 510–517.

BURT, P. J., AND ADELSON, E. H. 1983. The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications COM-31*, 4, 532–540.

CHEN, S. E. 1995. QuickTime VR – an image-based approach to virtual environment navigation. *Proceedings of SIGGRAPH '95*, 29–38.

DEBEVEC, P. E., AND MALIK, J. 1997. Recovering high dynamic range radiance maps from photographs. *Proceedings of SIGGRAPH 97* (August).

DURAND, F., AND DORSEY, J. 2002. Fast bilateral filtering for the display of high-dynamic-range images. *ACM Transactions on Graphics 21*, 3 (Proc. SIGGRAPH 2002), 257–266.

EDEN, A., UYTTENDAELE, M., AND SZELISKI, R. 2006. Seamless image stitching of scenes with large motions and exposure differences. *Proceedings of CVPR 2006) 3*, 2498–2505.

FATTAL, R., LISCHINSKI, D., AND WERMAN, M. 2002. Gradient domain high dynamic range compression. *ACM Transactions on Graphics 21*, 3 (Proc. SIGGRAPH 2002), 249–256.

FISCHLER, M. A., AND BOLLES, R. C. 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM 24*, 6, 381–395.

HDP, 2006. Hd photo. Microsoft Corporation, http://www.microsoft.com/whdc/xps/wmphoto.mspx.

KOPF, J., COHEN, M., LISCHINSKI, D., AND UYTTENDAELE, M. 2007. Joint bilateral upsampling. *ACM Transactions on Graphics 26*, 3 (Proc. of SIGGRAPH 2007).

LAM, K. 1985. *Metamerism and Colour Constancy*. PhD thesis, University of Bradford.

LISCHINSKI, D., FARBMAN, Z., UYTTENDAELE, M., AND SZELISKI, R. 2006. Interactive local adjustment of tonal values. *ACM Transactions on Graphics 25*, 3 (Proc. SIGGRAPH 2006), 646–653.

LOWE, D. 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision 60*, 2, 91–100.

MITSUNAGA, T., AND NAYAR, S. 1999. Radiometric self calibration. *Proceedings of CVPR '99 2*, 374–380.

PÉREZ, P., GANGNET, M., AND BLAKE, A. 2003. Poisson image editing. *ACM Transactions on Graphics 22*, 3 (Proc. SIGGRAPH 2003), 313–318.

REINHARD, E., STARK, M., SHIRLEY, P., AND FERWERDA, J. 2002. Photographic tone reproduction for digital images. *ACM Transactions on Graphics 21*, 3 (Proc. SIGGRAPH 2002), 267–276.

REINHARD, E. 2002. Parameter estimation for photographic tone reproduction. *Journal of Graphics Tools 7*, 1, 45–52.

SZELISKI, R., AND KANG, S. B. 1994. Recovering 3D shape and motion from image streams using nonlinear least squares. *Journal of Visual Communication and Image Representation 5*, 1 (March), 10–28.

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

WILBURN, B., JOSHI, N., VAISH, V., TALVALA, E.-V., ANTUNEZ, E., BARTH, A., ADAMS, A., HOROWITZ, M., AND LEVOY, M. 2005. High performance imaging using large camera arrays. *ACM Transactions on Graphics 24*, 3 (Proc. SIGGRAPH 2005), 795–802.