A Warping Framework for Wide-Angle Imaging and Perspective Manipulation

Robert Carroll

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2013-152
http://www.eecs.berkeley.edu/Pubs/TechRpts/2013/EECS-2013-152.html

August 16, 2013

A Warping Framework for Wide-Angle Imaging and Perspective Manipulation

by

Robert Carroll

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Associate Professor Maneesh Agrawala, Chair
Affiliate Professor Aseem Agarwala
Professor Martin Banks
Professor Jitendra Malik

Fall 2013

Chair _____ Date _____

_____ Date _____

_____ Date _____

_____ Date _____

University of California, Berkeley

A Warping Framework for Wide-Angle Imaging and Perspective Manipulation

Abstract

A Warping Framework for Wide-Angle Imaging and Perspective Manipulation

by

Robert Carroll

Doctor of Philosophy in Computer Science

University of California, Berkeley

Associate Professor Maneesh Agrawala, Chair

Nearly all photographs are created with lenses that approximate an ideal pinhole camera—that is, a perspective projection. This projection has proven useful not only for creating realistic depictions, but also for its expressive flexibility. Beginning in the Renaissance, the notion of perspective gave artists a systematic way to represent three-dimensional space on a two-dimensional canvas. However, unlike photographers—who have traditionally been tied to the optical designs of their cameras—other artists have been able modify the perspective projection relatively easily. They can locally adapt the projection in places where it makes objects look distorted, and they can more easily control the projection via two-dimensional image space features.

In this thesis we describe two projects based on image warping that give photographers some of the same control over spatial layout that painters have always had. The first is a technique to minimize distortion in a wide-angle image. Any projection of a 3D scene onto a plane unavoidably results in distortion; current methods either bend straight lines in the scene or locally distort the shapes of scene objects. Here, we present a method that minimizes this distortion by adapting the projection to content in the scene, such as salient scene regions and lines, in order to preserve their shapes. Our optimization technique computes a spatially-varying projection that respects user-specified constraints while minimizing a set of energy terms that measure wide-angle image distortion. We demonstrate the effectiveness of our approach by showing results on a variety of wide-angle photographs, as well as comparisons to standard projections.

Inspired by the way painters sketch vanishing points and lines to guide the construction of perspective images, the second project is a tool that gives users the ability to manipulate perspective in photographs using similar controls. This approach computes a 2D warp guided by constraints based on projective geometry. A user annotates an image by marking a number of image space constraints, including planar regions of the scene, straight lines, and associated vanishing points. The user can then use these constraints as handles to control the warp. Our system optimizes the warp such that straight lines remain straight, planar regions transform according to a homography, and the entire mapping is as shape-preserving

as possible. We demonstrate how this approach can be used to produce a variety of effects, such as changing the perspective composition of a scene, exploring artistic perspectives not realizable with a camera, and matching perspectives of objects from different images so that they appear consistent for compositing.

While the results of these techniques are not true perspective projections of a scene, they resemble perspective images and better satisfy a photographer's artistic vision.

To my parents, Gary and Sally Carroll.

Contents

List of Figures

Acknowledgments

I owe my gratitude to a number of people who have supported and encouraged me through my academic career and life in general.

First and foremost, my advisor Maneesh Agrawala was a driving force whose guidance was invaluable in getting me through graduate school. Aseem Agarwala, my mentor for two internships and co-author on the papers that eventually became this dissertation, was like a second advisor. Thanks you.

Thank you to Steve Seitz, who set me on the path of graphics research as undergraduate with zero experience. Raanan Fattal was an inspiration—I wish our time at Berkeley together had been longer.

Thank you to Marty Banks and Jitendra Malik for providing excellent feedback as part of my committee.

My labmates Kenrick Kin, Robin Held, Floraine Berthouzoz, and Sean Arietta made it a pleasure coming to the office every day. Thanks for all the laughs, coffee, proofreading, and discussions. James O'Brien, I enjoyed our impromptu whiteboard sessions immensely.

I was fortunate to be surrounded with an amazing group of peers during my time at Berkeley, many of whom have become lasting friends. The list is long, but I would especially like to thank Andy Konwinksi, Jacob Abernethy, and Kuang Chen for providing some of the best memories of my time in grad school.

Most of all I am grateful for the never-ending support of my family. My parents, Sally and Gary, have always been encouraging, but never demanding. My sister Megan, to whom I will always look up—thanks for setting a good example.

Chapter 1

Introduction

One function of photography is to relay the visual experience of looking directly at a scene. Cameras capture the same basic information as our own eyes do: rays of light converging on a single viewpoint. While our brains process this signal to create a visual impression of the scene, cameras simply store it as a photograph. We can recapture this signal later by viewing the photograph, but it isn't an exact replica. Almost every aspect of the original signal is altered by going through the photographic proxy, including color, intensity, focus, binocular disparity, and the overall spatial layout of the retinal image. These differences make photography an imperfect proxy, but also enable it to be used as an expressive art form. Artists can manipulate all of these factors to alter our perception of a particular scene.

One of the factors that most influences our perception of a photograph is its projection model: the mapping of rays (or directions) to the image plane. Nearly all photographs are created with lenses that approximate an ideal pinhole camera (i.e., a perspective projection). In this model, points in the scene are projected along straight lines through a single point, the "center of projection" (CoP), and intersected with the image plane. The perspective projection not only creates realistic depictions, it is also quite simple to use in a variety of domains. For example, the simplest camera obscura literally consists of an imaging plane and a pinhole. The perspective projection is also commonly used for computer generated imagery, where the required computations are simple ray-plane intersection. And, although painters can apply the pinhole model directly to create perspective images (perhaps with the aid of optical devices [21]), in practice it is simpler to rely on the two-dimensional signatures of linear perspective; artists can use vanishing points and construction lines to create realistic depictions of a scene without reasoning too extensively about the three-dimensional space [12].

Constructing images in this way also gives artists more flexibility over the perspective composition. They can, for example, more easily control the projection via two-dimensional image space features, such as vanishing points and lines. They can also locally adapt the projection in places where it gives undesirable results. Photographers, on the other hand, have traditionally been tied to the optical designs of their cameras and so do not benefit

from the same adaptability or control mechanisms as other artists. Typical cameras might give control over focal length (in addition to position and orientation), while some advanced designs have a few more degrees of freedom; however, these controls are rather limited compared to those of a painter, and are tied to the construction and placement of the camera rather than to the resulting images.

Artists might wish to adapt the perspective projection for a variety of reasons—most prominently, to compensate for how the photographic proxy inevitably alters our perception, distorting the spatial properties of the scene. In particular, because of the way our visual system interprets flat images, wide-angle photographs often look unnaturally stretched. A canonical example of this is a group photograph taken with a wide-angle lens: faces in the center of the photograph look fine, but those toward the peripheries look stretched. For this reason, photography manuals often recommend using a limited field of view (FoV) for everyday snapshots. In a similar setting a painter could easily draw each person's face as though it were centered, mitigating the apparent distortion.

However, photographers and other artists are not usually interested in simply producing images that approximate the experience of viewing some particular scene. Rather, they might prefer to impart to the viewer an experience of the artist's own design. Manipulating the spatial layout of the image, and perspective projection in particular, has proven to be a powerful technique in this regard.

1.1 A Taxonomy of Linear Perspective

A perspective image is formed by intersecting all rays passing through the CoP with the image plane. As such, the space of perspective projections is parameterized by the positions of the CoP and the image plane relative to the scene. Although this 3D description (known as *primary geometry*) fully determines a perspective image, when drawing in perspective it is often more natural to reason in terms of *secondary geometry*—the 2D foreshortened objects, converging lines, and vanishing points. Any combination of CoP and plane will result in a valid perspective projection, but photographers and artists are frequently interested in special cases where a particular symmetry exists in the secondary geometry.

One important special case is the class of parallel projections (Figure 1.1a). As the CoP is pulled infinitely far from the plane, all rays intersecting the plane become parallel. As this happens, vanishing points in the image are also pushed infinitely far from the image's center. The converging lines in the images, corresponding to parallel lines in 3D, become parallel themselves. These types of projections are impossible to capture with a physical camera, although extreme telephoto lenses can give a similar effect. Parallel projection is frequently employed in engineering diagrams because it doesn't introduce any foreshortening and therefore viewers can directly read the relative sizes of objects.

For a scene dominated by lines oriented along three Cartesian axes, perspective projections are often categorized by the number of (non-infinite) vanishing points in the image plane. A one-point perspective projection corresponds to an image plane that is parallel to

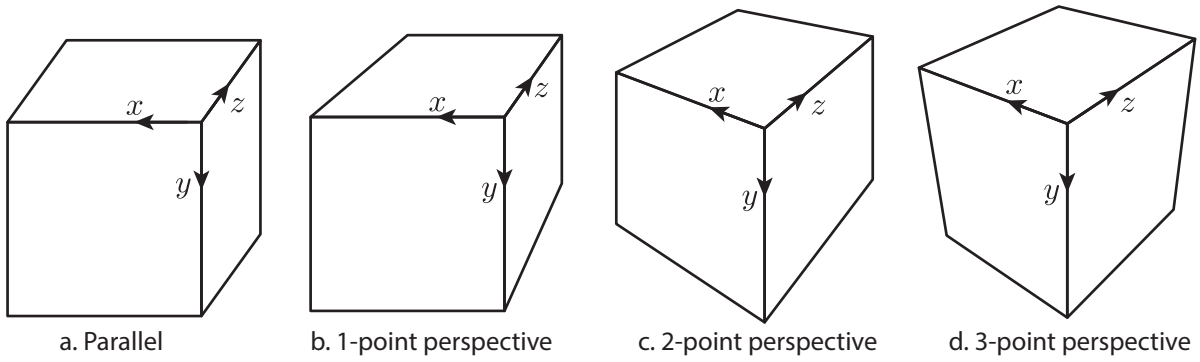


Figure 1.1: Special cases of perspective projection of a cube. a) Parallel projection corresponding to an infinitely distant center of projection. b) 1-point perspective corresponding to an image plane parallel to both the x and y axes. c) 2-point perspective corresponding to an image plane parallel to the y-axis. d) 3-point perspective doesn't correspond to any specific orientation of the image plane or position of the CoP.

two of the axes (x and y in Figure 1.1b), and is the simplest to draw. This projection is useful for depicting facades of buildings or other flat objects while maintaining the orientation of vertical and horizontal lines and without introducing any foreshortening. Similarly, two-point perspective projections are formed when the image plane is parallel to one of the axes (y in Figure 1.1c), and can be used for example to keep all vertical lines in a scene vertical in the image. Three-point is the most general case, where there is no specific relationship between the image plane and the scene (Figure 1.1d). Architectural photographers often prefer one- or two-point perspectives so that the facades of buildings do not converge vertically.

These special cases are extremes in a continuum of possible perspective projections. In addition to their specific use cases, they can also help us grasp the connection between primary and secondary geometry. While a CoP at infinity results in non-converging lines and no foreshortening, a general rule of thumb is that wide fields of view will have more-exaggerated foreshortening and more-quickly converging lines. Likewise, orienting the image plane to be more closely aligned with a given set of parallel lines in the scene will reduce their rate of convergence.

While photographers are able to control perspective composition only via primary geometry, drawing in perspective is typically done directly in terms of secondary geometry. Photographers can typically control the position and orientation of the camera and, within some range, the field of view. Architectural photographers interested in one- and two-point perspective require specialized hardware—for example, tilt-shift lenses and view cameras, to orient the image plane correctly without changing the view direction. And, they are limited to physically accessible locations and views that are unobstructed. For example, taking a narrow-angle photograph of a confined space is not possible, as the photographer cannot move back far enough to capture the entire scene. Perspective composition is relatively sim-

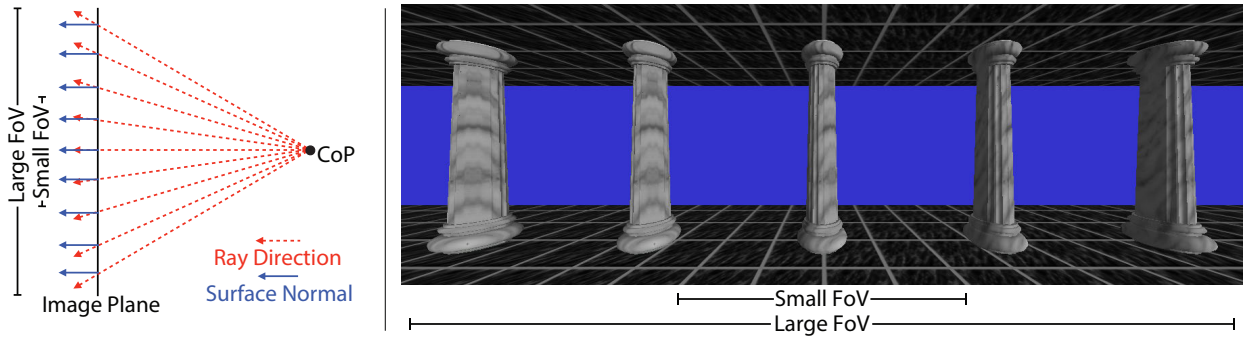


Figure 1.2: Using local-slant compensation, images can be perceived roughly correctly when viewed far from the center of projection [49]. However, this method works well only in areas where light rays hit the picture surface nearly orthogonally—that is, narrow field-of-view perspective images with the optical axis perpendicular to the image plane. Image on the right taken from Agrawala et al. [3].

pler when drawing: an artist can directly place vanishing points wherever he or she wishes, whether or not it corresponds to an accessible viewpoint.

1.2 Perspective and Human Perception

In order to develop techniques for creating customized image projections, it is useful to first understand how the human visual system interprets images, and particularly perspective images. When one views a perspective image from the CoP, the resulting retinal image is roughly equivalent to the one obtained when viewing the original scene directly.[1] As such, perspective is sometimes described as being the only "correct" projection. However, this reasoning neglects the fact that people do not necessarily view perspective images from the CoP. When a viewer looks at a perspective image from somewhere other than the center of projection, his or her retinal image no longer corresponds to the original scene but instead to a distorted one. Under a wide range of common viewing conditions people don't mind this distortion, and they perceive the original scene correctly—a phenomenon commonly referred to as *robustness of perspective* [29]. This robustness is central to our ability to interpret photographs; without it, we would perceive strangely-sheared worlds that look different from each viewpoint.

The mechanism we use for achieving robustness of perspective stems from our simultaneous awareness of the image surface and the scene being depicted [30]. When we observe

[1] Strictly speaking, even when the CoP and viewing location coincide only the overall spatial layout of the retinal image is preserved, and not necessarily other attributes such as luminance or those relating to focus. The camera generally uses a differently sized aperture than the eye, changing the depth of field. When viewing a photograph, the eye focuses on the picture plane rather than the original scene, perhaps influencing our perception. However, this thesis is concerned only with the spatial layout of photographs.

a photograph, our visual system isn't fooled into thinking that the picture frame is a window to another world—it "knows" that the photograph is a flat depiction. People employ a variety of cues to infer the location of the picture surface, including depth estimation through binocular vision and being able to see the image frame. If we remove these cues, for example by viewing the image monocularly through a pinhole to obscure the frame, robustness to different viewpoints breaks down [49]: a perspective image viewed from somewhere other than the CoP is interpreted as a distorted scene. Likewise, if we view an image from its CoP without the presence of these surface-depth cues, the illusion of viewing the scene directly is enhanced [35]. Our method for achieving invariance to viewpoint is not perfect, however; when we view an image away from its CoP, perspective distortion can be apparent, especially when the image depicts a large field of view. Further, even when we do view an image from its center of projection perspective, distortion can still be noticeable [49]. Vishwanath et al. showed that humans achieve invariance to viewpoint using what they call "local-slant compensation," in which the viewer locally estimates the surface orientation at each interest point in the image plane and then corrects the patch under the assumption that the surface is orthogonal to the captured rays. For perspective images, this assumption is true only at the image center, and becomes gradually more erroneous toward the peripheries (Figure 1.2). Therefore, unless viewing monocularly through a pinhole, we can experience perspective distortion when viewing wide-angle images even when the viewing location is near the CoP.

Because eyes and cameras perform seemingly-analogous functions, it is tempting to ask what type of projection is used by our eyes or brain. Such a question isn't particularly well defined, since our internal representation of an image is certainly much different from a flat photograph. We could more specifically ask what type of projection our eyes use to project onto the retina, and in many ways this is a type of perspective projection. The eye captures all rays converging toward a CoP, the pupil.[2] The sizes of objects as projected onto the retina decrease with distance, and the projections of parallel lines converge to a vanishing point. These are the classic trademarks of linear perspective. On the other hand, the exact layout of the retinal image is somewhat meaningless because it is subsequently sampled with rods and cones laid out in some non-rectilinear pattern. The image we see in our mind's eye is constructed by integrating these samples over time, and is much different from a single static image; instead, it is built from many snapshots focused on small portions of the scene, as the eyes continuously (and involuntarily) fixate at different points in the scene. The brain integrates images from both eyes, with distinct viewpoints and over time, as the head subtly changes position [1].

The end result is that when we view the world with our eyes, we experience a wide field of view without any obvious distortion—objects retain their natural shape, and linear structures remain straight. Creating a flat image of such a wide field of view that resembles (under normal viewing conditions) our subjective perception is remarkably challenging.

[2]The pupil is of course not a single point, but a finite aperture.

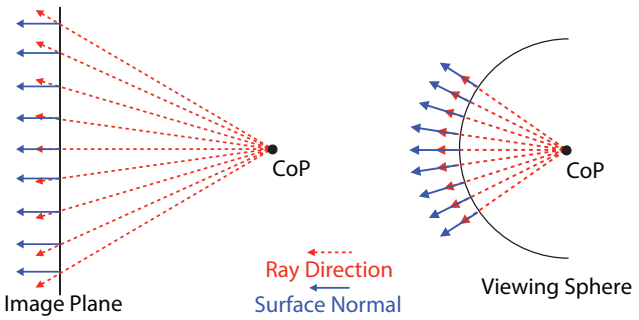


Figure 1.3: Shapes in a wide-angle perspective image look distorted because light rays hit the image plane at an angle. A spherical image has no local shape distortion because light rays always hit the surface orthogonally.

1.3 The Viewing Sphere and Other Projections

We've seen that images created with perspective projection do not appear distorted under a wide range of viewpoints as long as the field of view is not too wide. For images with a very large field of view (e.g. wide panoramas) it is possible to use other projections to alleviate some of the distortion inherent to perspective, although such alternatives also have their own limitations. The visual information seen from a single viewpoint is naturally represented on a viewing sphere, where the planar projection surface of linear perspective is replaced with a sphere centered on the CoP. This viewing sphere can capture a full 360-degree field of view—the light rays coming from all directions. At every point on a spherical projection surface the normal vector is aligned with the captured ray, and the image looks similar to a narrow field-of-view perspective image centered on that point (Figure 1.3).

To obtain a flat image we must map the viewing sphere onto a plane; however, a sphere is a non-developable surface and therefore any such mapping will introduce distortions [62]. Small portions of the viewing sphere can be well-approximated with a plane, so mapping images with a narrow field of view to a plane can be done without introducing much distortion, as we have already seen in the case of perspective projection. Large fields of view are more challenging. Cartographers have long recognized that mapping a sphere to a plane is difficult, and so have developed hundreds of projections that trade off different types of distortion [44]. Here, we consider the properties of perspective projection and two cartographic projections, Mercator and stereographic, that are commonly applied to very wide-angle photographs (Figure 1.7).

Perspective. Perspective has the unique property of preserving straight lines. Any straight line in the 3D scene will project to straight lines in 2D. Further, when the camera is held parallel to the ground plane, vertical scene lines remain vertical. However, we've also seen that objects located near the periphery of wide-angle perspective images can

appear unnaturally stretched, and in fact as the field of view approaches 180° this stretching becomes infinite on the image plane.

Mercator. The Mercator projection is a cylindrical projection designed to be conformal. Thus, it locally preserves the shape of objects. As in all cylindrical projections, meridians of the viewing sphere are mapped to vertical lines in the image. As such, vertical scene lines will appear vertical in the image when the camera is held parallel to the ground plane. The Mercator projection can handle a complete 360° horizontal field of view, but stretches to infinity as the vertical field of view approaches 180°. The Mercator projection is well suited to panoramic images with large horizontal fields of view, such as images containing large groups of people.

Stereographic. To form a stereographic image, the viewing sphere is projected onto a tangent plane through lines emanating from the pole opposite the point of tangency. Stereographic projections are also conformal, and therefore preserve the shapes of objects locally. In addition, stereographic projection preserves scene lines that pass through the center of the image. Stereographic images appear similar to those produced by fisheye lenses, as well as to wide field-of-view illustrations drawn using curvilinear perspective [14]. Like perspective projection, stereographic projection stretches objects toward the periphery; however, the objects are scaled in a conformal manner.

While none of these projections is perfect, each has properties that are amenable to certain types of scenes. Combining their features, based on the content of a particular image, can produce results superior to those from using any one global projection.

1.4 Perspective in Art

Long before the invention of photography, artists relied on perspective to create highly-realistic depictions. Perspective gave artists a systematic way represent a three-dimensional space on a two-dimensional canvas, and prior to its development artists were restricted to either depicting spatial relationships entirely in 2D or attempting to represent depth in some ad-hoc fashion. During the Renaissance artists first developed and formalized the perspective construction; however, they also quickly realized that perspective should not necessarily be followed rigidly, and soon developed specialized techniques for handling large fields of view [29, 14]. According to Kubovy, "The geometry of central projection was routinely violated to counteract its perceptually unacceptable effects" [29]. Treatises on perspective often suggest deviating from classical perspective when drawing humans in wide-angle scenes and drawing freehand instead [32]. Raphael's *School of Athens* is a classic example of this sort of deviation from the mathematically defined perspective projection (Figure 1.4); in this painting two spherical globes are painted as perfect circles, while the one point perspective projection corresponding to the background architecture would require them to be drawn as ellipses. Our visual system would interpret an ellipse as a distorted sphere—a consequence of

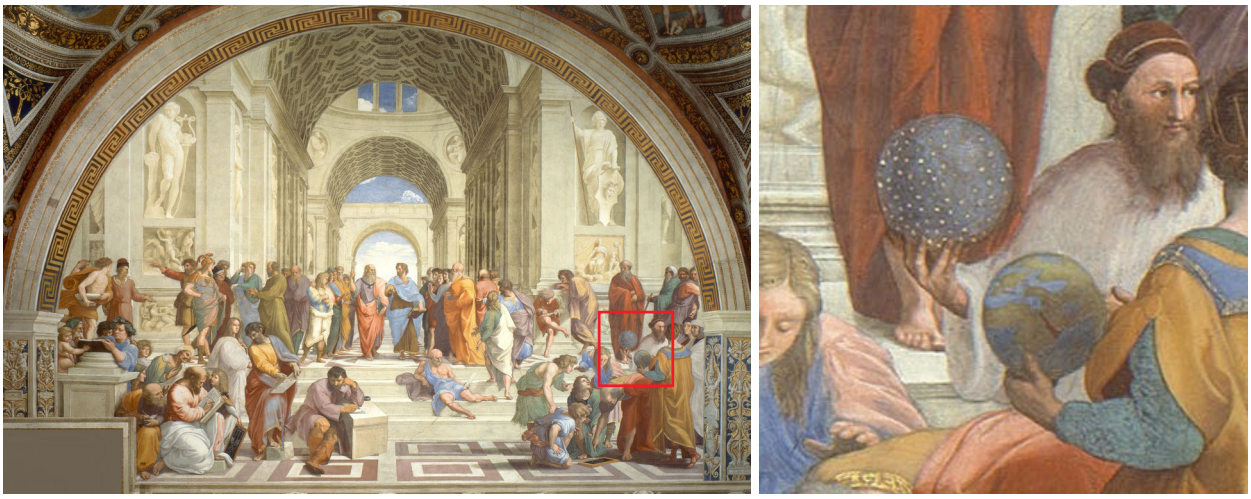


Figure 1.4: Raphael paints the globes as perfect circles in his *School of Athens* (1511).In a geometrically accurate perspective projection a sphere would appear as an ellipse, but that would look incorrect to viewers. Instead, he uses a locally centered projection for the globes and foreground figures.

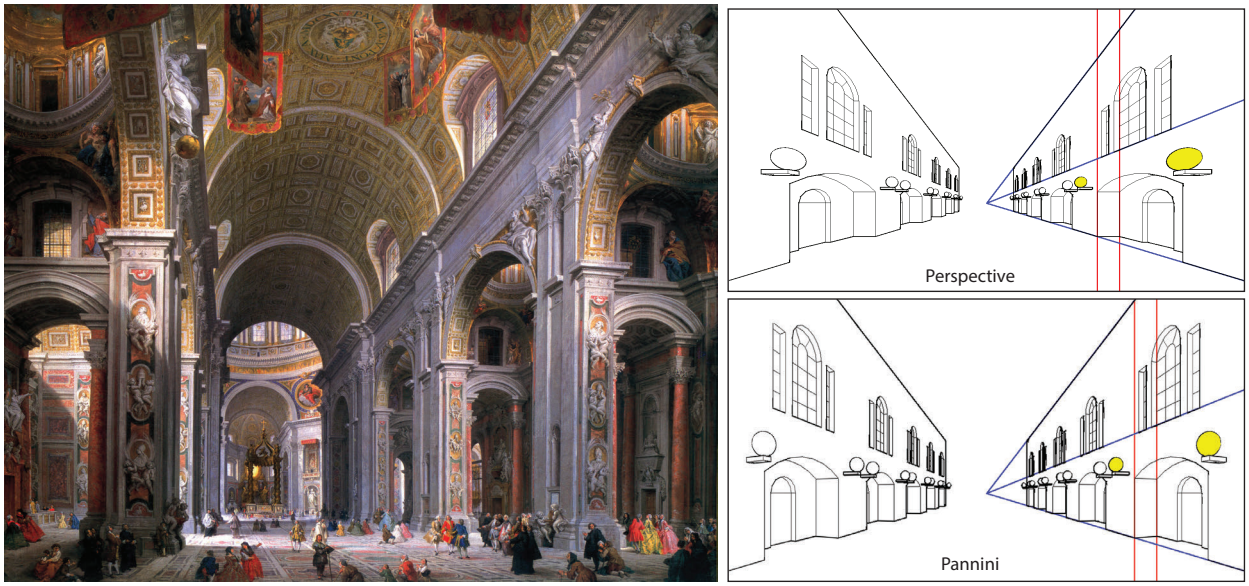


Figure 1.5: Gian Paolo Pannini's *Interior of St. Peter's, Rome* (1754). Pannini and a school of view painters from the Italian Baroque period created wide-angle paintings that appear to use standard perspective projections, but do not exhibit the expected perspective distortion. Sharpless et al. recently reverse engineered their method, which horizontally compresses the regions between radial lines extending from a central vanishing point [41].

Giorgio de Chirico - Th Mystery and Melancholy of a Street (1914)

Richard Estes - 34th Street, Manhattan, Looking East (1979)

Figure 1.6: Two examples of inconsistent vanishing points in paintings. **Left:** Chirico uses different perspectives for the two buildings, dramatically elongating the receding facade of the white building. **Right:** Despite the name, even "Photorealist" painters sometimes mix perspectives. In this case the artist pushed the left and right building closer together, fitting them in the frame without using a wide field of view and thereby introducing perspective distortion. Image taken from slides of Pat Hanrahan, who credits Denis Zorin for the discovery [18].

local-slant compensation. Similarly, Raphael depicts the human figures using locally centered projections where such foreground figures would appear stretched under the perspective projection of the background. While it is common for artists to use local perspectives for individual objects, some also deviate from the perspective rules in a more global way. Sharpless et al. [41] analyze the works of a school of painters called the Vedutisti, who specialized in wide-angle views of architectural scenes (Figure 3.12). These artists took advantage of the particular structure of such scenes to compress the projection horizontally around a central vanishing point, mitigating distortion that would be apparent under a true perspective projection.

Beyond depicting a three-dimensional scene, the end goal of an artist is to impart an experience on the viewer. Kubovy describes a number of ways artists manipulate perspective to direct the viewer's experience and achieve various expressive effects. A common technique is to make vanishing lines converge on a subject of interest, guiding the viewer's attention

there. For example, in Leonardo's *Last Supper* the head of Jesus is placed coincident with the vanishing point of the one-point composition. Kubovy also suggests that artists carefully use foreshortening and choose viewpoints to evoke particular moods. Artists also deliberately introduce perspective inconsistencies as a mode of expression. For example, some artists use a different CoP for different objects and fail to make parallel lines converge to a single vanishing point (Figure 1.6).

In this thesis, we demonstrate a framework to give photographers the ability to manipulate the spatial layout of photographs using controls similar to those employed by other artists to construct perspective images. Our approach is based on a framework of 2D image warps guided by constraints based on projective geometry and human perception. Image warping techniques, which have previously been applied to problems such as texture mapping and image retargeting, transform images by smooth mappings—stretching and bending the image like a rubber sheet. We apply this framework to two projects. In Chapter 3 we present a tool for minimizing perceived distortion in wide-angle images (Figure 1.7). As no global projection can simultaneously avoid bending lines and stretching the shapes of objects, our tool optimizes a projection based on locally defined metrics of these types of distortion. The results are images that keep lines straight like the perspective projection, but depict much larger fields of view like the stereographic and Mercator projections. In Chapter 4 we present a tool that gives the user artistic control over perspective composition (Figure 1.8). With an interface based on secondary geometry, users are able to do things like match a 1-point, 2-point, or orthographic perspective; change the foreshortening of planar scene elements; and create non-photorealistic perspective images.

The images produced by these techniques aren't necessarily accurate perspective projections—that is, they aren't realizable by placing a pinhole camera in the scene. Instead, to casual observers they resemble perspective images, but also better satisfy a photographer's artistic vision.

Figure 1.7: Wide-angle photographs can appear badly distorted under existing projections, including perspective, Mercator, and stereographic projections. Perspective projection preserves linear structures in the scene, but distorts objects. Conversely, Mercator and stereographic projections preserve shapes locally, but bend linear structures. Our projection (Chapter 3)) is designed both to preserve local shape and to maintain straight scene lines that are marked by the user with our interactive tool.

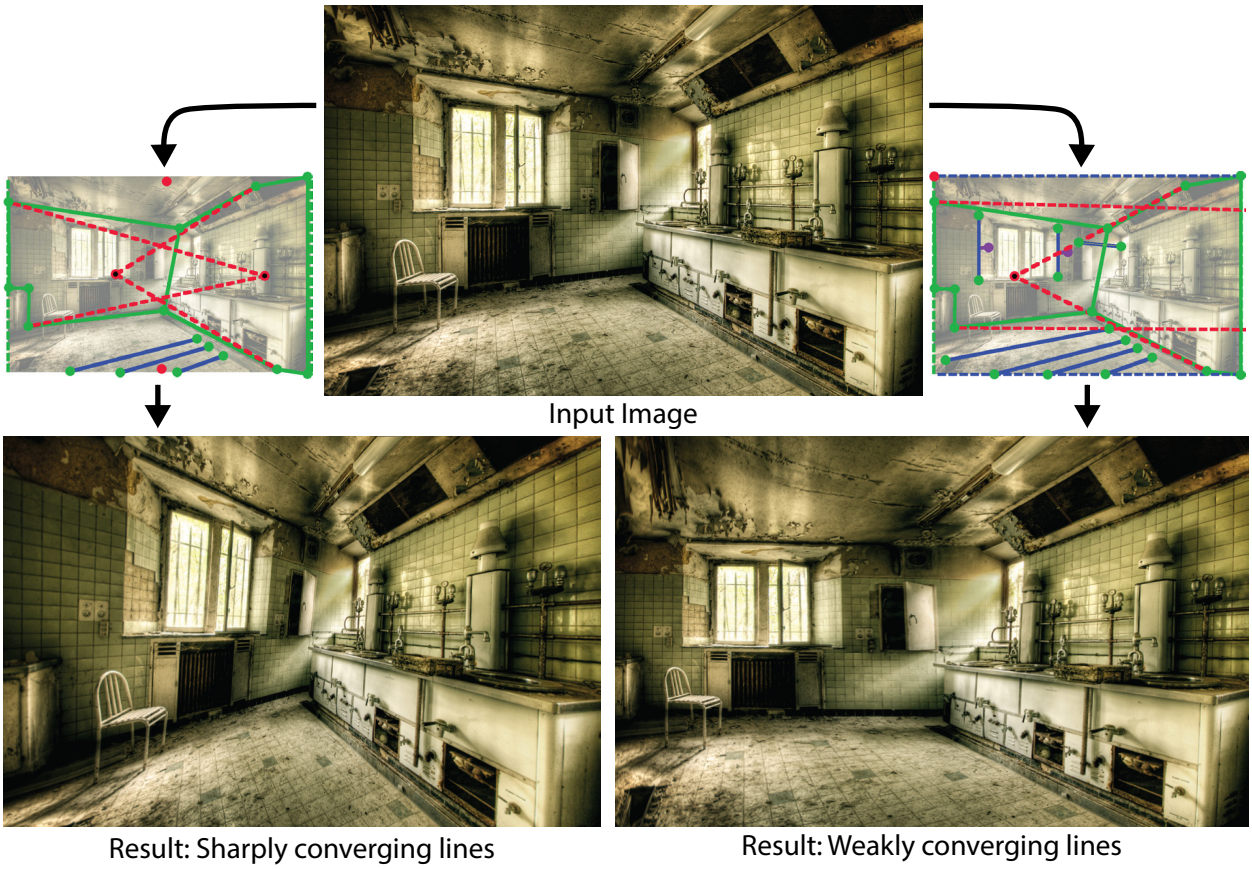


Figure 1.8: In Chapter 4 we present a tool for controlling the perspective in photographs by manipulating vanishing points and lines. We can constrain the lines to converge sharply as in a wide-angle image or more weakly as in a telephoto image. Our perception of the room changes for each result—the former result feels more tightly enclosed, and the later more spacious.

Chapter 2

Related Work

A number of researchers have addressed aspects of wide-angle distortion (Chapter 3) and perspective manipulation (Chapter 4). We give a brief overview of these techniques, as well as previous applications and formulations of image warping.

2.1 Perspective Manipulation

The oldest photographic perspective manipulation tool is the view camera or perspective control lens [47], which allows the photographer to change the orientation of the film plane relative to the scene. The ability to apply a similar effect digitally through a projective transform is a common feature in photography software. Our method can be seen as an enhancement to this approach. We add the ability to specify a projective transform for a specific image region, and the ability to apply different transforms to different regions. Finally, we also propose a different control metaphor through the manipulation of vanishing points and lines (secondary geometry).

One approach to manipulating the perspective in a photograph is to first reconstruct its 3D geometry and camera parameters. Horry et al. [23] introduced a simple interface that takes a central perspective image and fits a scene model consisting of a box for the background and axis-aligned planes for the foreground. Criminisi et al. [11] showed how 3D models could be created from perspective images by propagating the positional information from a few manually-specified 3D seed points with user guidance. This technique can be used to infer the 3D locations of geometric primitives, notably the corners of planar polygons. The interaction with our perspective manipulation tool (Chapter 4) is quite similar, with the user's effort dominated by drawing lines and polygons in the input image. However, our interaction metaphor is based entirely on the 2D secondary geometry, and we do not require the user to specify any 3D coordinates. Also, in many cases we can get away with specifying constraints on only a small portion of the image, whereas with 3D reconstruction only portions of the image with 3D geometry can be re-projected. Automatically reconstructing a 3D scene from a single image is an active topic of research[22, 38]. These techniques are

designed for outdoor scenes and report success rates of 30% and 67%, respectively; our goal in Chapter 4 is to create a more robust, user controllable tool that works on indoor and outdoor scenes. Another issue with single view modeling techniques is that they do not produce a complete representation of the 3D scene. Manipulating perspective with such an incomplete representation can lead to disocclusions or holes that need to be filled.

A more fully explored topic is the artistic manipulation of perspective given known 3D geometry. A number of researchers have addressed the problem of creating projections that contain multiple viewpoints, either interactively [3, 10] or by combining multiple views along a camera path [55, 37, 36]. Several researchers have proposed alternative projection models that go beyond the standard linear projection model, and they have used these models to produce artistic renderings [57, 17]. Unlike these methods our approach does not rely on access to 3D scene geometry.

Techniques have also been developed that work without the benefit of known 3D geometry. Agarwala et al. [2] described how to create multi-viewpoint panoramas of scenes too long to depict with standard perspective, starting with a series of photographs. Zelnik-Manor and Perona [58] automated the process of compositing a series of photographs from nearby viewpoints into Hockney-like joiners. While these technique combine photos into multi-perspective visualizations, our technique lets users locally modify perspective in a single photograph.

2.2 Wide-Angle Projections

Zorin and Barr [62] proved that no global projection can both keep straight lines straight and preserve the shapes of objects. They propose a one-parameter family of projections that trades off between these two conditions by interpolating between a perspective and stereographic projection. More recently, Zelnik-Manor et al. [59] address the same problem by stitching together multiple linear perspective projections into a "multi-plane" projection; the seams are placed by the user where they are least noticeable, e.g., the edge between two walls. However, any object that passes over such a seam will have a sharp discontinuity. Their method is limited to scenes where such a seam can be well hidden and where the field of view is wide only vertically or horizontally, since their projection can only change in one direction. Our wide-angle projection method (Chapter 3) improves upon these techniques with a locally-varying mapping designed to minimize local shape distortion and line bending where they matter in the image. That is, we do not attempt to keep all possible lines straight, only salient lines that appear in the image.

There is also commercial software for improving wide-angle images, such as DxO Optics Pro (dxo.com) and Fisheye Hemi (imagetrendsinc.com). As far as we can tell, the former allows the user to interpolate between perspective and either a cylindrical or stereographic projection, while the latter produces a cylindrical projection. In contrast, our method produces a spatially-varying mapping.

2.3 Image Warping

Image warping techniques have recently found use in a number of applications such as 2D shape manipulation [24, 39], texture mapping [16], image and video retargeting [54, 52, 26, 51, 34], video stabilization [31], stereoscopic disparity editing [30, 50], 3D parametric reshaping of human bodies [61], and image-based rendering [9]. All of these techniques find a mapping from the image plane to itself that minimizes some measure of distortion (e.g non-uniform stretching) while obeying constraints specific to the particular application (e.g. point correspondences). Researchers have employed a number of representations of the mapping function $f : \mathbb{R}^2 \to \mathbb{R}^2$, including quad and triangle meshes, sparse data interpolation, and moving least squares. We employ the mesh-based approach because meshes discretize the entire image domain and are therefore able to accommodate a wide range of locally defined distortion metrics and constraints. Mesh-based warping is closely related to mesh parameterization, where the goal is to find a mapping from a 2D manifold to the plane [42], and 3D shape deformation, where the constraints are defined in $\mathbb{R}^3$. In Chapter 3 we find a mapping from a meshed sphere to a plane, which is particularly similar to mesh parameterization, although we are still able to formulate our constraints using a continuous model of the sphere rather than an arbitrary mesh.

Researchers have employed a number of metrics of local distortion to regularize mesh based warps. In particular, these distortion metrics can generally be formulated in terms of the derivatives of the mapping function. The Jacobian matrix J of first partial derivatives at each point gives a local linear approximation of the warp, and it is often constrained to belong to some set of allowed transformations at each point in the image. For example, the shape manipulation technique of Igarashi et al. constrains J to be a rotation matrix, leading to a locally as-rigid-as-possible or isometric warp [24]. Another common choice is to constrain J to be a rotation and uniform scale, resulting in an as-similar-as-possible or conformal warp [8, 31, 60, 7, 51, 9]. Some techniques restrict local rotations by constraining J to be a uniformly scaled identity matrix [52, 34], or the identity matrix itself (allowing only translations) [54, 30, 61]. Krähenbühl use a variation of the uniform scaling constraint where the scale factor is enforced to be the same across the entire image [28]. Karni et al. allow different classes of transformations based on local image content, progressing from any affine transformation to similarity transformations to rigid transformations for regions of progressively higher detail [26]. Some warping techniques also constrain the second derivatives of the mapping function to be zero, i.e. the Jacobian should be the same at neighboring points [8, 7, 61]. In this dissertation we use similarity transformations for J and also constrain the second derivatives. We provide motivation for these choices in the following chapters. Each of the cited works rely on a number of other constraints specific to the application, and our categorization is somewhat loose. All of these constraints typically result in an overdetermined system of equations which cannot all be satisfied exactly. Instead they are used to define a least-squares optimization problem, where each constraint is weighted based on its importance relative to other constraints. Some applications, in particular image retargeting, also vary the weighting spatially so that salient image content is better preserved.

Depending on the particular choice of constraints the resulting least squares optimizations may be linear or non-linear. Linear systems can be solved in closed-form using a sparse matrix factorization. While non-linear systems can be solved with generic non-linear least squares methods like Gauss-Newton or Levenberg-Marquardt [33], these black-box algorithms have typically been avoided in favor of strategies customized for the particular constraints. Some non-linear least squares problems can be solved with an alternating least squares (ALS) approach, where the unknowns can be split into two groups and holding either group fixed results in a linear subproblem. Sorkine and Alexa use this strategy, which they call the "local-global" algorithm, for as-rigid-as-possible warps [46]. They alternate between finding best-fit rotations given the current warped mesh (the local step) and solving for the best warp given the rotations (the global step). Karni et al. give a general framework for local-global algorithms [26]. ALS approaches have the advantage that they are often simpler to implement than Gauss-Newton or Levenberg-Marquardt; they do not require derivatives of the energy function and do not require line-searches or trust-region updates. Although it is unclear how the convergence rates of ALS approaches compare to that of the more standard methods, finding the absolute minimum error may be less important than quickly finding a visually pleasing result. We use both an ALS and a Gauss-Newton optimization in this dissertation. The objective function in Chapter 3 uses a single non-linear constraint which is naturally suited to an efficient ALS optimization. Chapter 4 makes use of multiple non-linear constraints, and we found the Gauss-Newton algorithm to have a more straightforward implementation.

Chapter 3

Projections for Wide-Angle Images

We develop a technique for creating wide-angle images that better match the experience of viewing the world with our own eyes. Creating such images is difficult because it requires flattening the viewing sphere, where objects are locally-distortion free, to a plane. Under any standard projection of the viewing sphere onto the image plane objects and structures look unnaturally stretched, sheared or curved (Figure 1.7). The key to our approach is to give photographers the ability to design projections tailored to the content of a specific images.

Our system is comprised of two components: a simple user interface that allows users to control the appearance of the output image by specifying the properties that should be preserved (Section 3.1), and a weighted least-squares optimization procedure for generating the spatially-varying mapping from the viewing sphere to the image plane (Section 3.2).

3.1 User interface

Users can load wide-angle images from any source (e.g. wide-angle perspective lens, fisheye lens, panoramic mosaic, etc.) into our interface as long as the mapping from the source data to the viewing sphere is known. We have pre-built many of these mappings into our system and in most cases the user simply specifies the type of input image (fisheye, cylindrical panorama, etc.) and its field of view. We use the standard projection models for these types of inputs and ignore the issues of lens distortion. Lens distortion has not proven to be a problem for the examples we have tried.

The main task for the user is to identify linear scene structures in the input image that should remain straight in the output. Since only perspective projection can maintain all scene lines as straight, this input allows the algorithm to focus on preserving only the lines that the user considers important. We also considered allowing the user to mark salient regions; however, we never found this input to be necessary, since the algorithm can typically find near-conformal solutions for most sets of line constraints.

Users can directly specify two types of line constraints in our interface. The *general line constraint* is designed to keep linear structures in the scene from bending in the output image.

The *fixed orientation line constraint* modifies the general constraint so that the linear scene structures map to straight lines at a user-specified orientation in the output image. Users specify both of these constraints by identifying linear scene structures in the input image. All straight lines in the scene map to great circles on the viewing sphere. Although the linear structures may appear bent in the input image, all our computations are performed on the viewing sphere. Users click on the two endpoints of the linear structure to specify the constraint and our system computes and draws the corresponding arc of the great circle passing through those points.

This approach also allows users to interactively set the field of view of the input image. Given an input image with unknown field of view, our system initially assumes either a 180° or 360° field of view depending on the user-specified input image mapping (i.e. wide-angle perspective, fisheye, etc.). The user can click the endpoints of a line, and if the field of view is incorrect the computed line will not lie on the linear structure in the input image. The user can then adjust the field of view up or down until the drawn line matches the linear structure. This process allows the user to easily work with an image with unknown parameters. However, the procedure for inferring the FOV will not work if the input image is a perspective projection, since all lines are perfectly straight in any perspective image regardless of focal length. In this case the user must determine the focal length explicitly, possibly using EXIF tags.

To specify the fixed orientation constraint users first create a line and then type 'h' or 'v' to indicate that the line should be oriented either horizontally or vertically in the output image. Although our fixed orientation constraint can work with any user-specified orientation, we have found in practice that horizontal and vertical orientation constraints are most common. Therefore, we chose to simplify the interface rather than allow users to specify arbitrary line orientations. Given these line constraints our algorithm computes a mapping from the viewing sphere to the image plane as we describe in the next sections.

3.2 Mathematical setup

We define an image projection as a mapping from the viewing sphere parametrized by longitude λ and latitude ϕ into a planar domain parametrized by u and v (Figure 3.1). In vector form $\boldsymbol{\lambda} = (\lambda, \phi)$ and $\mathbf{u} = (u, v)$. We can represent the mapping as the two functions $u(\lambda, \phi)$ and $v(\lambda, \phi)$, or in vector form as $\mathbf{u}(\boldsymbol{\lambda})$.

We can describe the local properties of such a mapping in terms of differential north and east vectors, $\mathbf{h}$ and $\mathbf{k}$, of the projection, where

$$\mathbf{h} = \begin{bmatrix} \frac{\partial u}{\partial \phi} \\ \frac{\partial v}{\partial \phi} \end{bmatrix}, \; \mathbf{k} = \begin{bmatrix} \frac{\partial u}{\partial \lambda} \\ \frac{\partial v}{\partial \lambda} \end{bmatrix} \frac{1}{\cos(\phi)}. \tag{3.1}$$

Without the cosine $\mathbf{h}$ and $\mathbf{k}$ correspond to the columns of the Jacobian matrix $J = \frac{\partial \mathbf{u}}{\partial \boldsymbol{\lambda}}$. However, spherical coordinates are non-Euclidean, and the cosine term is necessary to ac-

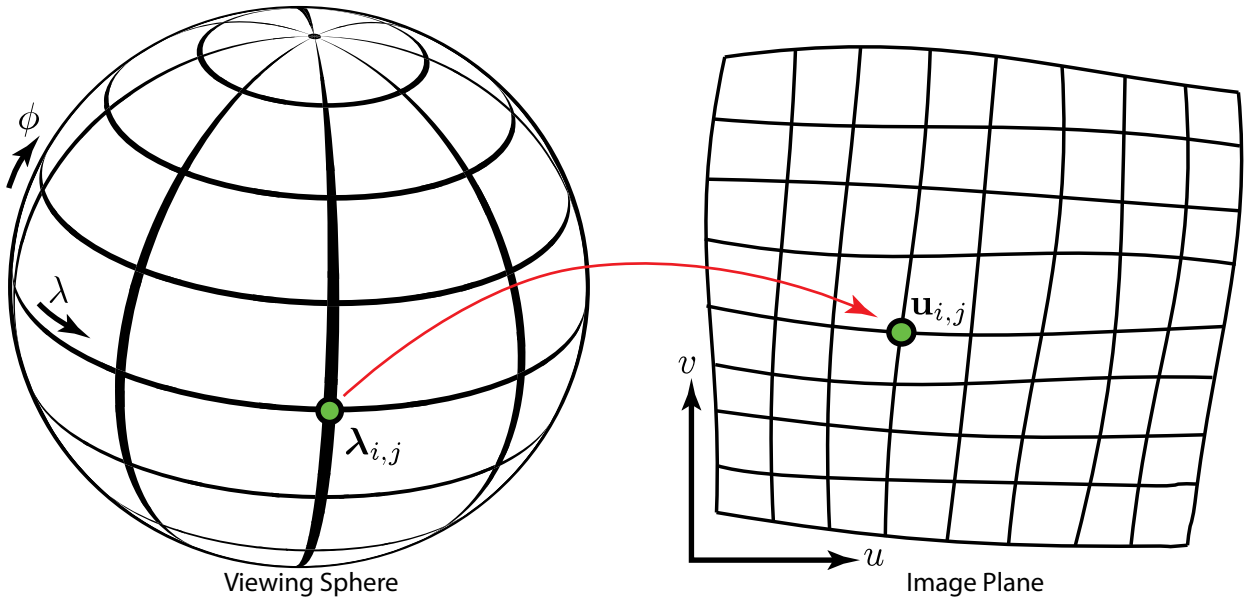


Figure 3.1: We optimize a mapping from the viewing sphere parametrized by (λ, ϕ) to a plane parametrized by (u, v). For each vertex $\boldsymbol{\lambda}_{i,j} = (\lambda_{i,j}, \phi_{i,j})$ on the sphere we calculate its projection $\mathbf{u}_{i,j} = (u_{i,j}, v_{i,j})$ on the image plane.

count for the fact that equal steps in λ travel different distances on the sphere depending on ϕ.

A mapping is conformal if $\mathbf{h}$ is a 90° rotation of $\mathbf{k}$, such that

$$\mathbf{h} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \mathbf{k} \tag{3.2}$$

or, equivalently

$$\frac{\partial u}{\partial \phi} = -\frac{\partial v}{\partial \lambda}\frac{1}{\cos(\phi)}, \quad \frac{\partial v}{\partial \phi} = \frac{\partial u}{\partial \lambda}\frac{1}{\cos(\phi)}. \tag{3.3}$$

These are the Cauchy-Riemann equations for mapping a sphere to a plane [20, 45].

With this setup we can derive and analyze the properties of various projections. For example, in a cylindrical panorama u varies linearly with λ. So, if $u = \lambda$ then $\frac{\partial u}{\partial \lambda} = 1$, which implies that

$$\frac{\partial v}{\partial \phi} = \frac{1}{\cos\phi}, \; v = \ln(\sec\phi + \tan\phi)$$

where the latter equation can be found in a table of integrals. These equations for u and v describe the Mercator projection, which is the only conformal cylindrical projection.

In our case the constraints on the mapping function $\mathbf{u}(\boldsymbol{\lambda})$ and its derivatives vary locally, so we cannot derive a closed-form solution for the projection. Instead, we discretize the mapping by sampling a uniform grid in (λ, ϕ) indexed by integers (i, j) (Figure 3.1). We

define V to be the entire set of vertices (i, j) that fall in the field of view of the input image. The vertices form a quad mesh on the sphere's surface. For each spherical grid vertex $\boldsymbol{\lambda}_{i,j}$, we compute the value of its corresponding $\mathbf{u}_{i,j}$ in the output 2D domain via an optimization.

3.2.1 Conformality

We form conformality constraints on the mesh by discretizing the Cauchy-Riemann equations (3.3), giving

$$u_{i,j+1} - u_{i,j} = -(v_{i+1,j} - v_{i,j})/\cos\phi_{i,j} \quad (3.4)$$

$$v_{i,j+1} - v_{i,j} = (u_{i+1,j} - u_{i,j})/\cos\phi_{i,j} \quad (3.5)$$

When solving differential equations exactly, multiplying both sides of these equations by a constant has no effect on the solution. However, when solving discretized equations in the least squares sense, changing the weight of the constraints does affect the solution. All quads on the viewing sphere are not equal in size, so equally weighting the constraints biases conformality toward regions of the sphere with higher quad density. We therefore weight the constraints by the area of the quad, which at latitude ϕ is proportional to $\cos(\phi)$. We multiply the constraints by $\cos(\phi)$ to define the conformality energy as

$$\begin{aligned} E_c = &\textstyle\sum_{(i,j)\in V} w_{i,j}^2 \left((v_{i+1,j} - v_{i,j}) + \cos\phi_{i,j}(u_{i,j+1} - u_{i,j})\right)^2 \\ &+ \textstyle\sum_{(i,j)\in V} w_{i,j}^2 \left((u_{i+1,j} - u_{i,j}) - \cos\phi_{i,j}(v_{i,j+1} - v_{i,j})\right)^2 \end{aligned} \quad (3.6)$$

where $w_{i,j}$ is a spatially varying weight that we will describe in section 3.2.4.

3.2.2 Straight lines

We define L as the set of all (general and fixed orientation) line constraints marked by the user and L_f as the subset of fixed orientation line constraints. On the viewing sphere straight lines in the scene project to great circles, so we refer to great circles simply as lines.

We constrain all points that lie on a line on the viewing sphere to map to points which are collinear on the image plane. However, since we are working in a discrete domain, few vertices fall directly on the line segment. Instead, if a line intersects a quad we constrain a "virtual vertex" within that quad, placed at the midpoint of the line-quad intersection. We compute the position of a virtual vertex on the sphere, and its bilinear interpolation coefficients (a, b, c, d), as shown in Figure 3.2(left). For each quad intersected by a line l we associate an output virtual vertex $\mathbf{u}_{i,j}^l = a\mathbf{u}_{i,j} + b\mathbf{u}_{i+1,j} + c\mathbf{u}_{i+1,j+1} + d\mathbf{u}_{i,j+1}$ on which we place our line constraints. We denote V_l as the set of vertex indices (i, j) corresponding to quads intersected by the line. We also denote the two line endpoints as $\mathbf{u}_{start}^l$ and $\mathbf{u}_{end}^l$. For the rest of this section we drop the superscript l and assume the $\mathbf{u}$ variables correspond to virtual vertices for a particular line.

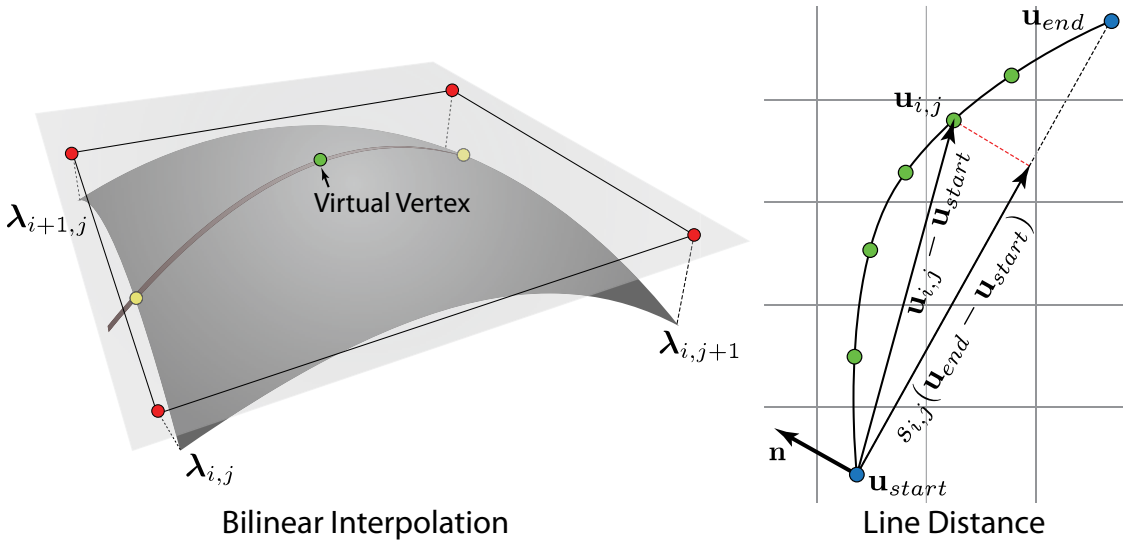


Figure 3.2: **(left)** A line crosses the quad's sides at the yellow points on the sphere. Midway between these on sphere's surface is the midpoint of the quad-line intersection, marked in green. We project the quad vertices onto the plane tangent to the midpoint, and use the inverted bilinear interpolation equations [19] to compute coefficients (a, b, c, d) that express the midpoint as a linear combination of the projected mesh vertices, marked in red. **(right)** We can express the distance of the virtual vertex $\mathbf{u}_{i,j}$ to the line as the dot product of $(\mathbf{u}_{i,j} - \mathbf{u}_{start})$ with the normal vector, or as the difference of $(\mathbf{u}_{i,j} - \mathbf{u}_{start})$ and its component parallel to the line.

The distance of a point $\mathbf{u}$ to a line connecting two endpoints $\mathbf{u}_{start}$ and $\mathbf{u}_{end}$ is $(\mathbf{u} - \mathbf{u}_{start})^T \mathbf{n}(\mathbf{u}_{start}, \mathbf{u}_{end})$, where

$$\mathbf{n}(\mathbf{u}_{start}, \mathbf{u}_{end}) = R_{90} \frac{(\mathbf{u}_{end} - \mathbf{u}_{start})}{\|\mathbf{u}_{end} - \mathbf{u}_{start}\|} \tag{3.7}$$

is the normal vector of the line and R_{90} is a 90 degree rotation matrix. We therefore define the line energy for a constrained line l as

$$E_l = \sum_{(i,j) \in V_l} \left((\mathbf{u}_{i,j} - \mathbf{u}_{start})^T \mathbf{n}(\mathbf{u}_{start}, \mathbf{u}_{end}) \right)^2 . \tag{3.8}$$

We minimize this non-linear energy functions using an alternating least squares method based on two different linearizations of E_l.

We can express the distance of a point $\mathbf{u}$ to a line as the difference between $(\mathbf{u} - \mathbf{u}_{start})$ and its projection onto the line's unit tangent vector as shown in Figure 3.2(right). We can thus express the line energy in an equivalent form as

$$E_l = \sum_{(i,j) \in V_l} \| (\mathbf{u}_{i,j} - \mathbf{u}_{start}) - s(\mathbf{u}_{i,j}, \mathbf{u}_{start}, \mathbf{u}_{end})(\mathbf{u}_{end} - \mathbf{u}_{start}) \|^2 \tag{3.9}$$

where

$$s(\mathbf{u}, \mathbf{u}_{start}, \mathbf{u}_{end}) = \frac{(\mathbf{u} - \mathbf{u}_{start})^T(\mathbf{u}_{end} - \mathbf{u}_{start})}{\|\mathbf{u}_{end} - \mathbf{u}_{start}\|^2} \tag{3.10}$$

is the normalized length of the projection of $(\mathbf{u} - \mathbf{u}_{start})$ onto $(\mathbf{u}_{end} - \mathbf{u}_{start})$, such that $s(\mathbf{u}_{start}, \mathbf{u}_{start}, \mathbf{u}_{end}) = 0$ and $s(\mathbf{u}_{end}, \mathbf{u}_{start}, \mathbf{u}_{end}) = 1$. We now have two ways to simplify the line energy: by fixing the normal vector in equation 3.8 we get

$$E_{lo} = \sum_{(i,j)\in V_l} \left((\mathbf{u}_{i,j} - \mathbf{u}_{start})^T\mathbf{n}\right)^2, \tag{3.11}$$

and by fixing the normalized projections in equation 3.9 we get

$$E_{ld} = \sum_{(i,j)\in V_l} \|(\mathbf{u}_{i,j} - \mathbf{u}_{start}) - s_{i,j}(\mathbf{u}_{end} - \mathbf{u}_{start})\|^2. \tag{3.12}$$

Intuitively, these linearized energy terms allow us to reduce the total line energy in two different ways: E_{lo} allows points to slide freely along the line while fixing the line's orientation, and E_{ld} allows the orientation of the line to change while preventing points from sliding along the line. We use the two line energies alternately in an iterative minimization scheme as follows. First, we initialize each $s_{i,j}$ using the arc length between $\boldsymbol{\lambda}_{i,j}$ and $\boldsymbol{\lambda}_{start}$ on the viewing sphere. We set E_{lo} as the line energy and optimize the mapping to obtain values for each $\mathbf{u}_{i,j}$, which we then insert into equation 3.7 to compute a normal for each line. We use these normals and optimize the mapping using the E_{ld} line energies, giving new values for each $\mathbf{u}_{i,j}$ which we use to estimate the $s_{i,j}$ variables with equation 3.10. We repeat this process until convergence. For fixed orientation lines, there is no need to use E_{ld}, and we use the fixed orientation line energy on every iteration.

Although we cannot prove the convergence of our algorithm, we found it to converge very quickly in practice. After two or three iterations we find the projection is visually comparable to the converged result, and after 10 iterations changes are imperceptible.

3.2.3 Smoothness

The conformality and line constraints alone may lead to visual artifacts in the projection. Since conformality does not consider scale or orientation the projection may change dramatically over the space of the image, especially near line segments (Figure 3.3). Adding a smoothness term limits how quickly scales and orientations are able to change. To form the smoothness constraint we again look at the differential north vector $\mathbf{h}$. If $\mathbf{h}$ changes slowly across the projection, orientation and scale must also change slowly. So, in a least squares sense we would like

$$\frac{\partial \mathbf{h}}{\partial \boldsymbol{\lambda}} = \begin{bmatrix} \frac{\partial^2 u}{\partial\phi\partial\lambda} & \frac{\partial^2 u}{\partial\phi^2} \\ \frac{\partial^2 v}{\partial\phi\partial\lambda} & \frac{\partial^2 v}{\partial\phi^2} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}. \tag{3.13}$$

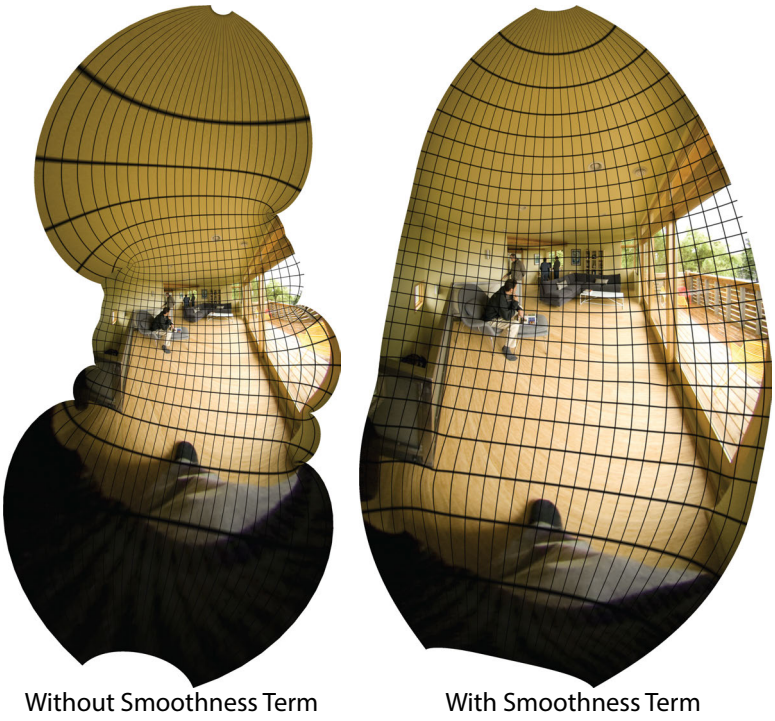


Figure 3.3: This image shows the projection obtained using just conformality and line constraints. The smoothness constraint prevents rapid changes in scale and orientation.

Intuitively, this constraint minimizes the curvature of the mapping. We can discretize this set of equations using finite difference approximations to the second derivatives, again weighted by $\cos\phi_{i,j}$, and take the Frobenius norm of $\frac{\partial \mathbf{h}}{\partial \boldsymbol{\lambda}}$ to give the smoothness energy

$$E_s = \sum_{(i,j)\in V} w_{i,j}^2 \cos^2\phi_{i,j} \left\| \begin{bmatrix} u_{i,j+1} - 2u_{i,j} + u_{i,j-1} \\ v_{i,j+1} - 2v_{i,j} + v_{i,j-1} \\ u_{i+1,j+1} - u_{i+1,j} - u_{i,j+1} + u_{i,j} \\ v_{i+1,j+1} - v_{i+1,j} - v_{i,j+1} + v_{i,j} \end{bmatrix} \right\|^2$$

where $w_{i,j}$ is a spatially varying weight that we describe in section 3.2.4.

Although we could place a similar constraint on $\mathbf{k}$ by setting

$$\frac{\partial \mathbf{k}}{\partial \boldsymbol{\lambda}} = \begin{bmatrix} \frac{\partial^2 u}{\partial \lambda^2} & \frac{\partial^2 u}{\partial \phi \partial \lambda} + \frac{\partial u}{\partial \phi}\frac{\tan\phi}{\cos\phi} \\ \frac{\partial^2 v}{\partial \lambda^2} & \frac{\partial^2 v}{\partial \phi \partial \lambda} + \frac{\partial v}{\partial \phi}\frac{\tan\phi}{\cos\phi} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}. \tag{3.14}$$

we found that this constraint did not improve the results; the comformality constraint ties together smoothness along the two grid axes, so that smoothness of the h vector field will imply smoothness of the k vector field.

3.2.4 Spatially-varying constraint weighting

We associate a weight with each vertex, which allows us to spatially vary the strength of conformality and smoothness constraints and selectively reduce shape distortion in areas where it is likely to be most noticeable. We also increase weight to prevent excessive distortion that can occur near the endpoints of line constraints. We base the weighting function on three quantities: proximity to the endpoint of a line, a local image salience measure, and proximity to a face.

Line endpoint weights. Due to the discontinuous nature of the line constraint, highly distorted and even flipped quads can occur near line endpoints. To counteract such distortion, we increase the weighting function near line endpoints using a Gaussian function. We compute the Gaussian by defining distances between vertices to be the Euclidean distance between indices, and setting the standard deviation equal to the width of the mesh divided by 100. To compute the total line endpoint weight $w^L_{i,j}$ at each vertex we sum over the Gaussians for each endpoint.

Salience weights. We weight our constraints by a local salience term to allow smooth areas of the image to be deformed more than areas with greater variation. While there are more complicated techniques for measuring image salience [25], we found that simply measuring local variance in color was sufficient for our purposes. More specifically we set the local salience weight $w^S_{i,j}$ at each vertex to be the standard deviation in color in a window around the vertex, and then normalize the values between 0 and 1.

Face detection weights. People are particularly adept at noticing distortion of human faces, so we increase the weighting function near faces using the face detection algorithm of Viola and Jones [48], as implemented in OpenCV [6]. The input image may be too distorted for the detector to reliably find faces, so we instead warp it to a Mercator projection, which has two desirable properties for face detection: it is conformal, so faces will not be stretched, and it is cylindrical, so the face will be upright assuming the camera and face were upright in the scene. The face detector returns a center and radius for each face, which we use to compute a Gaussian in the Mercator projection, with standard deviation equal to one third the face radius and unit height. We warp the weight field to the equirectangular grid to define the face weight $w^F_{i,j}$ at each vertex.

Total weight. We combine the line endpoint weights, the salience weights, the face weights, and a baseline weight given to all vertices to define the total vertex weights as

$$w_{i,j} = 2w^L_{i,j} + 2w^S_{i,j} + 4w^F_{i,j} + 1. \tag{3.15}$$

In practice, we keep the weights of this linear combination of terms, along with all other parameters associated with our system, fixed for all our results.

3.2.5 Total energy and optimization

In addition to the spatially varying weights we place on each mesh vertex, we also use global weights for each energy term to set their relative importance. Our total energy function is a

weighted sum of the conformality, smoothness, and line energies:

$$E = w_c^2 E_c + w_s^2 E_s + w_l^2 \left(\sum_{l \in L \setminus L_f} E_l + \sum_{l \in L_f} E_{ld} \right). \tag{3.16}$$

For lines with fixed orientations we use the fixed-orientation energy instead of the general line energy term. To solve this minimization problem, we alternate between optimizing the total energy with fixed line orientations

$$E_o = w_c^2 E_c + w_s^2 E_s + w_l^2 \sum_{l \in L} E_{ld} \tag{3.17}$$

and total energy with normalized projections fixed

$$E_d = w_c^2 E_c + w_s^2 E_s + w_l^2 \left(\sum_{l \in L \setminus L_f} E_{lo} + \sum_{l \in L_f} E_{ld} \right). \tag{3.18}$$

Since we intend the line energies to approximate hard constraints we use a very high weight on the line term. The only other important factor is the relative weighting of the conformality and smoothness terms, which we determine experimentally such that the smoothness weight is minimal, yet strong enough to correct artifacts. We use $w_c = 1$, $w_s = 12$ and $w_l = 1000$ for all results. However, we find that using such a high weight for the first iteration can generate a highly distorted initial mapping, since errors in the initialization of the $s_{i,j}$ are large. The algorithm behaves better using a lower weight $w_l = 10$ for the first iteration.

The quadratic energy function at each iteration of our algorithm results in a sparse linear system

$$A\mathbf{x} = \mathbf{0} \tag{3.19}$$

where A is the constraint matrix and $\mathbf{x}$ is a column vector containing the unknowns $u_{i,j}$ and $v_{i,j}$. This system is homogeneous and has trivial solutions corresponding to mapping all vertices to a single point. We can compute a non-trivial solution using an eigenvector decomposition of A, as described by Forsyth and Ponce [15]. An alternative approach is to add a small regularization term for each vertex preferring some known mapping (we use stereographic), making the right-hand side of equation 3.19 non-zero. We find in practice that solving the regularized system is more efficient and yields visually identical results to the eigenvector solution. We solve the linear system using the PARDISO sparse direct solver [40]. In the last step of the process we render the mesh at the desired image size using bilinear texture mapping [19]. The resulting image usually has irregular boundaries, as shown in Figure 3.4, so we let the user crop the result to a rectangular image.

Figure 3.4: We compare the uncropped and manually cropped results of our system to three global projections. The input image was taken from a 140° horizontal field of view fisheye lens. This field of view is well beyond the typical limits for perspective projection, causing severe stretching. The Mercator projection bends the lines between walls and ceiling, and the stereographic projection bends vertical lines. Our result is similar to the conformal projections, but with straight lines.

3.3 Results

We demonstrate the results of our system on a number examples in figures 1.7, 3.4 and 3.5-3.12. Additional results are included on our project website (http://vis.berkeley.edu/papers/capp/). The input image for Figure 1.7 was taken with a 180° circular fisheye lens, which captures an entire hemisphere of the viewing sphere. As is typical for all our examples, the perspective projection looks undistorted only at the very center of the image, but is extremely stretched toward the peripheries. While objects and faces look undistorted in the conformal projections, bent lines in the bookcase, ceiling and table are immediately apparent. Our result shows neither type of distortion.

In the remaining examples we show the input image with line constraints overlaid, along with the number of line constraints. In many of our examples, our projection tends to look globally very similar to the Mercator and stereographic projections, but with lines straightened. Mercator does well with vertical scene lines, and stereographic does well with scene lines with a common vanishing point near the image center. However, where one of these projections succeeds, the other fails. Our results tend to combine the best of both projections.

In Figure 3.11 we compare our approach to that of Zorin et al. [62], after adjusting their parameter to produce a result that we consider best for the input image. Although their approach does achieve a nice balance of conformality and straight line distortion, both distortions are present in the result. In contrast, our result shows neither type of distortion. Additionally, we were able to use constraints on line orientation to make vertical lines appear vertical in the image. In Figure 3.10 we compare our approach to the "multi-plane" projection of Zelnik-Manor et al. [59]. Where an object (the chair in this example) is distorted by a seam between projections planes, they segment out the object, apply a local perspective projection, composite the object back into the output image, and then fill any remaining holes. Even after this process discontinuity artifacts remain. Our approach finds a smooth projection which keeps all lines straight and does not distort objects. Compared to previous work, our optimization based approach has much more flexibility to adapt to specific scenes. In Figure 3.12 we compare our approach to the Pannini projection of Sharpless et al. [41]. Their projection, designed to mimick the technique used by painters of architectural scenes, does a good job at mainting vertical lines and lines going through the center of the image. Our technique applies more generally and introduces less comformality distortion.

Since our system is interactive, the user can add or change line constraints if they are not satisfied with the results of the first try. We found that while we could often obtain the desired result on the first try, sometimes we noticed additional lines needed to be straightened or fixed in orientation, requiring a few more iterations of adding more constraints. All the results shown in this paper use between 5 and 40 constrained lines, with an average of 20.

Input - 5 lines

Perspective

Mercator

Stereographic

Our Result

Figure 3.5: The input image is a 180° fisheye lens, shown overlaid with the constrained lines. The perspective projection shows severe stretching here, as perspective stretches to infinity for a 180° field of view. The Mercator and stereographic examples both bend lines; for Mercator it is most noticeable on the front edge of the carpet and the top of the back wall, and for stereographic it is most noticeable for the vertical lines toward the image peripheries.

Input - 16 lines

Perspective

Mercator

Stereographic

Our Result

Figure 3.6

Input - 28 lines

Perspective

Mercator

Stereographic

Our Result

Figure 3.7

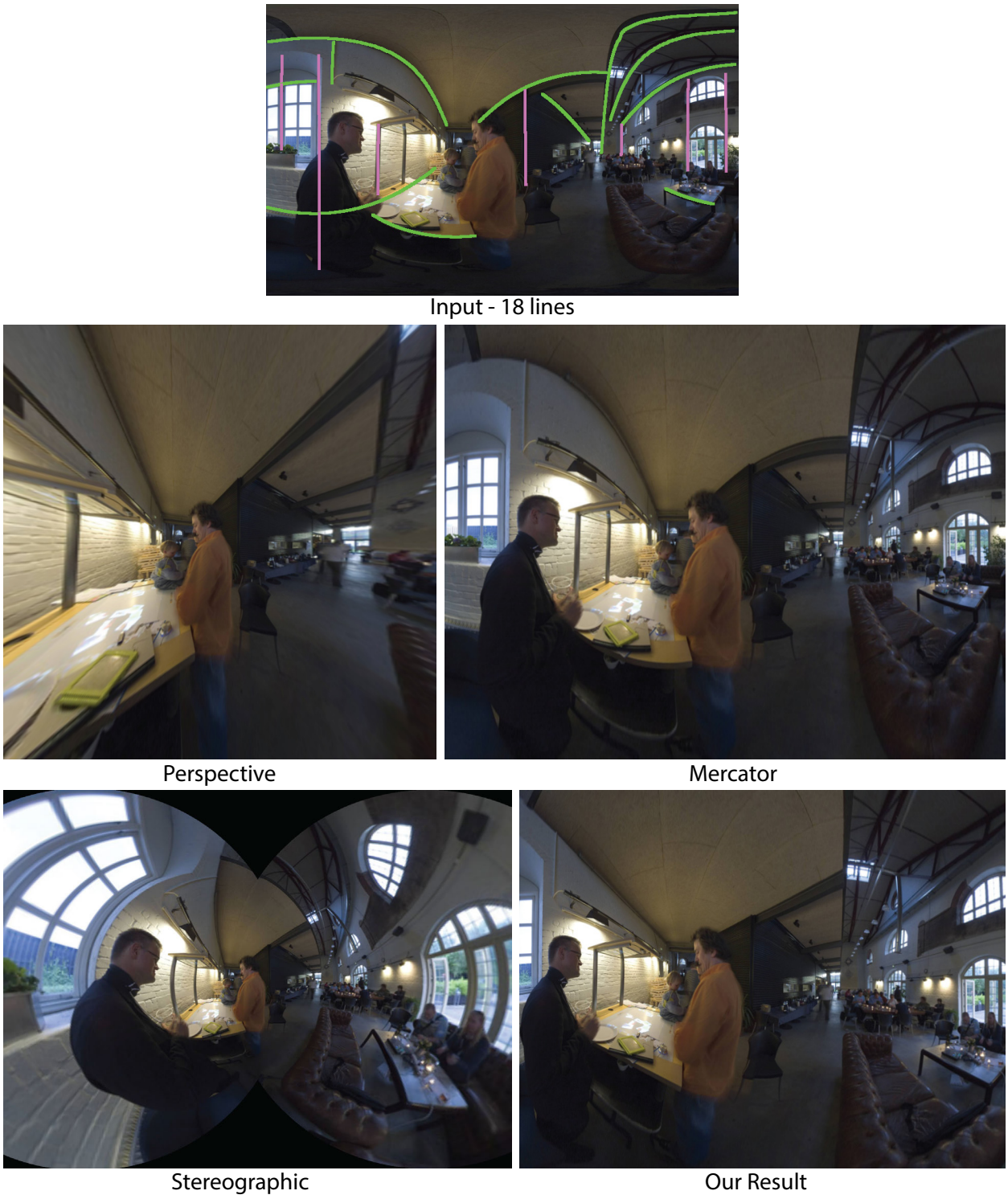


Figure 3.8: The input image is a 290° horizontal by 180° vertical equirectangular panorama. The stereographic projection is not well suited for this example; the orientations of vertical lines and objects in the scene look odd and stretching is severe. The Mercator projection does a relatively good job; however, there is bending of straight lines on the walls. Our result looks similar to Mercator in this example, but the lines are straightened.

Input - 18 lines

Perspective

Mercator

Stereographic

Our Result

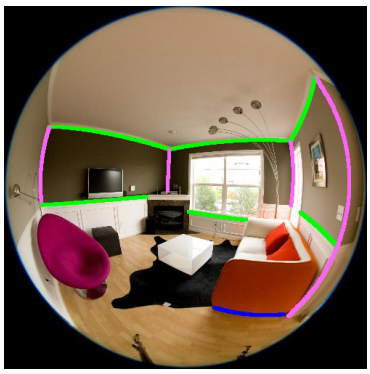
Figure 3.9

Figure 3.10: Distortions caused by discontinuities in "multi-plane" projection of Zelnik-Manor et al. [59] are only partially corrected with object segmentation in their "multi-plane multi-view" projection. Our system handles this case well. The stitching artifacts in the image are part of the source data.

Figure 3.11: The projection of Zorin et al. [62] contains conformality distortion (the people on both sides of the image are stretched horizontally) and bends straight lines (between the ceiling and back wall, and along the door jamb on the left and pipe on the right). Our result shows neither distortion.

3.3.1 Salience

Although we initially believed the salience-based weighting would be key to the success of our method, in the vast majority of cases its effect is subtle. The space of conformal projections is larger than we expected, and for most sets of line constraints our method can find a solution that is nearly conformal everywhere, not just in salient regions. Face detection was slightly more effective at reducing distortion since we are very perceptually sensitive to distortion in faces (Figure 3.13); however, in most cases, it again had little effect. Overall, the local distortion introduced by our constraints is significantly less than for an application like content-aware resizing [4], so salience is less important.



Input - 11 lines

Mercator

Stereographic

Sharpless et al.

Our Result

Figure 3.12: Comparison to Sharpless et al.

Figure 3.13: Our result with and without face detection weighting.

3.3.2 Implementation details

The dimensions of our quad mesh are independent of the image resolution, and are typically much smaller. The number of vertices is dependent on two factors: the resolution of our discretization of the viewing sphere, and the portion of the viewing sphere covered by the input image. We only add vertices to the linear system if they are covered by image data. All our results are computed on some portion of a spherical mesh with $160,000$ vertices. The $180°$ fisheye examples in figures 1.7, 3.5, 3.6 cover half the viewing sphere, requiring meshes of $80,000$ vertices, and had a runtime of approximately 1 minute each on a 2.4Ghz Core 2 Duo CPU. The full frame fisheye images from figures 3.4, 3.7, and 3.9 used approximately $40,000$ vertices, and had a runtime of about 15 seconds. For each result we ran our optimization for 8 iterations, after which all the results converged visually.

3.3.3 Alternative Mesh

The equirectangular used in our initial implementation is simple to work with because it is based on a closed form parameterization of the sphere. This parameterization makes it straightforward to calculate the mapping between spherical mesh vertices and points in the image. It also simplifies derivation of the conformality and smoothness energy terms. However, it has one major disadvantage: singularies at the north and south poles. Near the poles the horizontal spacing of the mesh vertices becomes smaller and the quads get tall and skinny. This singularity results in an excessive number of vertices, slow performance, and uneven treatment of the image depending on what part of the mesh it falls on.

After publishing this work [8], we developed an implementation for Adobe Photoshop (marketed as "Adaptive Wide-Angle") based on an alternative mesh. A family of meshes based on subdividing faces of the platonic solids (e.g the icosahedron) have much more uni-

Figure 3.14: A failure case with long parallel lines between 2 vanishing points.

form mesh spacing. We used the TOAST parameterization, which is based on a subdivided octahedron [56], and has previously been applied to gradient domain processing of spherical imagery [27]. Using this mesh we were able to dramatically reduce the runtime of our optimization, such that the result of adding a line is almost immediate.

3.4 Limitations and future work

Our system does not always compute a satisfactory result. We found three typical scenarios for which the result was often less than we desired (more examples are included on the project website).

(1) When the camera points directly at a long, planar scene, we often found ourselves trying to use the tool to simulate the effect of a multi-viewpoint panorama [2]. However, in such scenes there are multiple parallel lines covering nearly 180° of the viewing sphere, and converging at two visible vanishing points. Straightening such lines unavoidably distorts the region between them (Figure 3.14).

(2) If a scene is covered with large numbers of straight lines in varying directions, it is hard to straighten them all. This challenge is not surprising, given that perspective projection is the only mapping that can preserve all straight lines. This problem often manifests in textures with many directed lines, such as a hardwood or tiled floor. This effect can be observed in figures 3.4 and 3.5.

(3) As noted Section 3.3.3 our original formulation can yield large scale changes near the north and south poles, but we alleviated this issue in our Photoshop implementation with a more uniform mesh.

A final area of concern with our method relates to cropping: the results of our algorithm are not rectangular and need to be manually cropped to produce a final result (Figure 3.4). Often we found users would want to extend a crop to include some portion of the field

of view, but doing so would cause the crop to extend beyond the image data. A natural extension of our method would be to include a cropping box in the set of constraints, so that the box remains filled while salient scene objects do not extend beyond its perimeter. We include uncropped versions of all our results on the project website.

One area for future work is developing a completely automatic system that identifies salient linear structures using line detection algorithms. Our initial experiments in this direction were not successful. For one, the strength of an edge in the image may not correspond to its salience. Second, we often found ourselves adding lines that do not strictly correspond to image lines (e.g., a long horizon frequently occluded by trees).

Chapter 4

Artistic Perspective Manipulation

We present a tool that gives photographers the ability to manipulate perspective in images after they are captured using image space controls similar to those used by artists to construct perspective images (Figure 1.8). We use the same basic building blocks as in Chapter 3, image warping guided user-provided constraints. However, the goal is not to provide minimize percieved distortion, but rather to enable the photographer to design a perspective composition that matches their artistic vision.

With this tool the user annotates an image by marking a number of image space constraints including planar regions of the scene, straight lines, and associated vanishing points. They can then use these primitives as handles to control the warp. Our system optimizes the warp such that straight lines remain straight, planar regions are transformed as they would under a perspective projection (i.e. according to a homography) and the entire mapping is as shape-preserving as possible.

While an image produced by our tool is not an accurate perspective projection of the scene, we are often able to produce perceptually plausible images that look as if the position of the camera moved within the scene. Furthermore, our tool allows users to manipulate images according to constraints that are based on perspective rules, but are not necessarily consistent with a single perspective projection.

4.1 User interface

We provide the following controls for manipulating perspective.

Planar Regions Users can constrain a polygonal image region so that it is transformed as a projection of a rigid 3D plane, i.e., it is a homography.

Line Segments Users can constrain line segments to remain straight. This constraint can also be used in conjunction with the orientation and vanishing point constraints.

Line Orientation Users can constrain a line segment to be oriented vertically or horizontally.

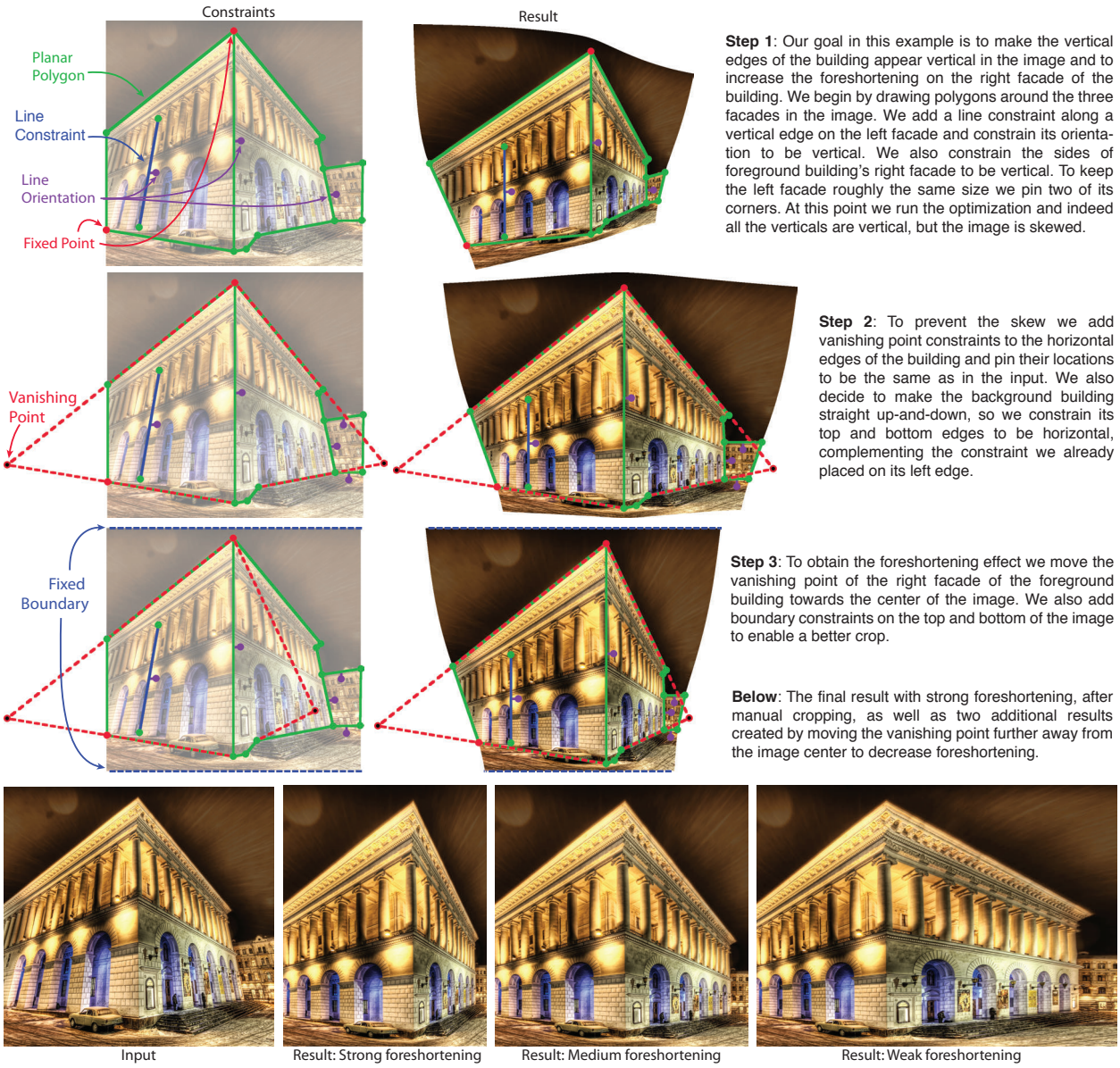


Figure 4.1: A typical interaction with our tool. We show the intermediate steps used to foreshorten the right wall of a building.

Vanishing Points Users can connect multiple line segments to a vanishing point constraint to ensure that they all intersect at the vanishing point. Moving vanishing points around the image plane allows users to control perspective changes.

Fixed Points Users can constrain a point in the input image so that it does not move.

Borders Users can constrain each side of the image boundary to stay fixed or optionally to just remain straight.

We demonstrate how these constraints can be used to control perspective in a typical user interaction with our tool in Figure 4.1. Our goal in this example is to make the vertical edges of the building appear vertical in the image (a two-point perspective) and to increase the foreshortening on the right facade of the building. We begin by drawing polygons around the three facades in the image. We add a line constraint along a vertical edge on the left facade and constrain its orientation to be vertical. We also constrain the sides of foreground buildings right facade to be vertical. To keep the left facade roughly the same size we pin two of its corners. At this point we run the optimization and indeed all the verticals are vertical, but the image is skewed. To prevent the skew we add vanishing point constraints to the horizontal edges of the building and pin their locations to be the same as in the input. We also background building straight up-and-down by constraining its top and bottom edges to be horizontal, complementing the constraint we already placed on its left edge. To obtain the foreshortening effect we move the vanishing point of the right facade of the foreground building towards the center of the image. We also add boundary constraints on the top and bottom of the image to enable a better crop. The final result has strong foreshortening. We also show two additional results created by moving the vanishing point further away from the image center to decrease foreshortening.

4.2 Warping Function

Unlike in Chapter 3 where the input images need to be calibrated so they can be mapped to the viewing sphere, our perspective manipulation tool is designed to work on uncalibrated perspective images. Thus, the image warp is defined as a plane-to-plane mapping, rather than the sphere-to-plane mapping. As shown in Figure 4.2, we define the warp in terms of a mapping from the input image parameterized by $\mathbf{x} = (x, y)$ into a planar domain parametrized by $\mathbf{u} = (u, v)$. We can represent the mapping as the two functions $u(x, y)$ and $v(x, y)$, or in vector form as $\mathbf{u}(\mathbf{x})$. We discretize the mapping by sampling a uniform grid in (x, y) indexed by integers (i, j), forming a quad mesh upon which we define our constraints. For each grid vertex $\mathbf{x}_{i,j}$, we compute the value of its corresponding $\mathbf{u}_{i,j}$ in the output domain via an optimization.

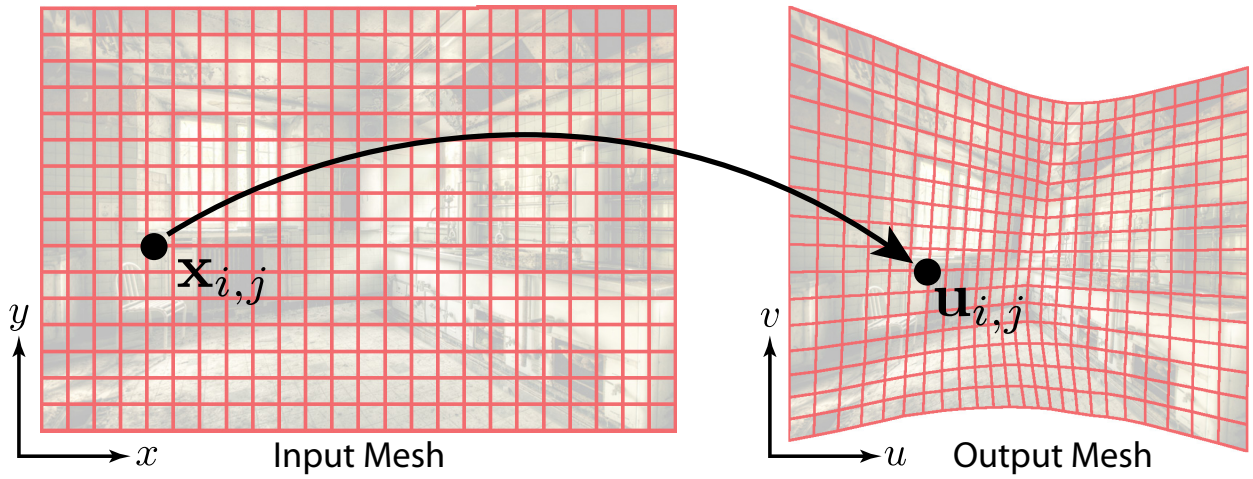


Figure 4.2: The input and warped mesh computed for Figure 1.8, shown at half resolution. The mesh defines a discrete mapping from the xy-plane to the uv-plane.

4.2.1 Planar Regions

Under a perspective projection planar regions in the scene are transformed according to a homography. Therefore, we constrain the set of mesh vertices enclosed by a user marked polygon to be mapped according to some (unknown) homography. In homogeneous coordinates this constraint can be expressed as

$$\begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{4.1}$$

and $(u, v) = (U, V)/W$. We use this constraint to define an energy term for a planar polygon as

$$E_h = \sum_{i,j} \left| \begin{bmatrix} u_{i,j} \\ v_{i,j} \end{bmatrix} - \begin{bmatrix} \frac{h_1 x_{i,j} + h_2 y_{i,j} + h_3}{h_7 x_{i,j} + h_8 y_{i,j} + 1} \\ \frac{h_4 x_{i,j} + h_5 y_{i,j} + h_6}{h_7 x_{i,j} + h_8 y_{i,j} + 1} \end{bmatrix} \right|^2 = \sum_{i,j} |\mathbf{u}_{i,j} - \mathbf{H}(\mathbf{x}_{i,j})|^2 \tag{4.2}$$

In general, both the homography variables $h_{1\ldots8}$ and the output coordinates $(u_{i,j}, v_{i,j})$ are unknowns that we optimize.

4.2.2 Homography Compatibility

If two planar polygons share a common edge it is necessary to constrain their homographies to be compatible along that edge to prevent discontinuities in the mesh. That is, if $\mathbf{x}_1$ and $\mathbf{x}_2$ are points that define a shared edge between two polygons in the input image domain with associated homographies H_a and H_b, we must ensure $\mathbf{H}_a(\alpha\mathbf{x}_1 + (1-\alpha)\mathbf{x}_2)) = \mathbf{H}_b(\alpha\mathbf{x}_1 +$

$(1-\alpha)\mathbf{x}_2))$ for all $\alpha \in [0,1]$. It is sufficient to constrain the homographies for a few values of α evenly sampled from 0 to 1, giving the energy term

$$E_{hc} = \sum_{i=0}^{n} \left|\mathbf{H}_a\left(\alpha_i\mathbf{x}_1 + (1-\alpha_i)\,\mathbf{x}_2\right) - \mathbf{H}_b\left(\alpha_i\mathbf{x}_1 + (1-\alpha_i)\,\mathbf{x}_2\right)\right|^2, \tag{4.3}$$

where $\alpha_i = \frac{i}{n}$. We found $n = 10$ to be sufficient. The homography parameters are the only unknowns in this energy function.

4.2.3 Straight Lines

The line constraint is similar the one used in Chaper 3, and in fact uses the same energy function, but we implement it slightly differently. Instead of using the alternating linearizations we add an auxilliary variable θ for the orientation of the line, which we jointly optimize with the mesh. This formulation simplifies the vanishing point constraint in Section 4.2.4 and is better suited to our optimization strategy described in Section 4.2.8. We constrain a user-specified line segment to remain straight by discretizing it into a set of points $\mathbf{x} \in V_l$, one for each quad the line segment crosses. The mapping $\mathbf{u}(\mathbf{x})$ of all $\mathbf{x} \in V_l$ should be collinear. If we parameterize lines by the mapping of one of these points $\mathbf{u}(\mathbf{x}_0)$ and an orientation θ, then the distance of a point $\mathbf{u}$ to the line is $(\mathbf{u} - \mathbf{u}(\mathbf{x}_0))^T\mathbf{n}(\theta)$, where $\mathbf{n}(\theta)^T = [\sin(\theta)\ \cos(\theta)]$ and $\mathbf{x}_0$ is one of the endpoints of the input line segment. We can constrain points to be collinear in the output image with the energy term

$$E_l = \sum_{\mathbf{x}\in V_l}((\mathbf{u}(\mathbf{x}) - \mathbf{u}(\mathbf{x}_0))^T\mathbf{n}(\theta))^2. \tag{4.4}$$

where θ may be known or unknown in our optimization depending on whether the user fixed the orientation of the line. Since a line segment is unlikely to pass through any of the mesh vertices exactly, each $\mathbf{u}(\mathbf{x})$ is a bilinear combination of the form $\mathbf{u}(\mathbf{x}) = a\mathbf{u}_{i,j} + b\mathbf{u}_{i+1,j} + c\mathbf{u}_{i+1,j+1} + d\mathbf{u}_{i,j+1}$.

4.2.4 Vanishing Points

This constraint restricts a set of lines to intersect a common vanishing point. Given a user specified vanishing point $\mathbf{u}_{van}$ and a set of lines L_{van} that pass through it, we define the vanishing point energy term as

$$E_v = \sum_{l\in L_{van}} ([\sin(\theta_l)\ \cos(\theta_l)](\mathbf{u}_{van} - \mathbf{u}(\mathbf{x}_{0,l})))^2. \tag{4.5}$$

The coordinates of the vanishing point may be knowns if the user fixes them, or they can be unknowns, which could be useful if the user only wants the set of lines be consistent with each other. However, in all our results we fixed the location of the vanishing points.

4.2.5 Point Constraints

This energy term restricts the location of a vertex to some fixed target location, and is given by

$$E_p = |\mathbf{u}(\mathbf{x}) - \mathbf{u}_{target}|^2. \tag{4.6}$$

where $\mathbf{u}(\mathbf{x})$ is bilinearly interpolated from neighboring vertices.

4.2.6 Borders

In some cases it is useful to allow the user to fix all or part of the image boundary in order to produce rectangular edges. Without the constraint the resulting image will usually have an irregularly shaped boundary that may be difficult to crop into a rectangle. For a particular side of the boundary we constrain all vertices to map to the same u coordinate (left and right sides) or v coordinate (top and bottom), but we allow vertices to slide along the boundary. The energy function is given by

$$\begin{array}{c} E_b = \sum_{j=1}^{h}(u_{1,j} - u_{left})^2 I_{left} + (u_{w,j} - u_{right})^2 I_{right} \\ + \sum_{i=1}^{w}(v_{i,1} - v_{top})^2 I_{top} + (v_{i,h} - v_{bottom})^2 I_{bottom} \end{array} \tag{4.7}$$

where I_{side} is a binary indicator variable specifying whether a side is fixed. The variables $(u_{left}, u_{right}, v_{top}, v_{bottom})$ may be knowns or unknowns in the optimization depending on whether the user chooses to fix the locations of the borders or only constrain the boundaries to remain vertical/horizontal. The former approach is useful for ensuring that the aspect ratio of the image is fixed, while the latter approach can be useful for maintaining a rectangular image with arbitrary aspect ratio. However, in many cases constraining the entire border is too limiting, causing distortions in the result, and we often leave some or all of the borders free.

4.2.7 Shape Preservation

The constraints described so far generally only involve a subset of the mesh vertices, and may have multiple valid solutions. We prefer mappings that change smoothly and are conformal, and we use constraints similar to those in sections 3.2.1 and 3.2.3. Because we aren't computing a mapping from the viewing sphere, the comformality constraint doesn't measure perspective distortion in the same way as in Chapter 3; it measures distortion relative to the original perspective image. The formulation of these constraints is also simplified in plane-to-plane case. In the plane-to-plane case a mapping is conformal if satisfies the Cauchy-Riemann equations $\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y}$ and $\frac{\partial u}{\partial y} = -\frac{\partial v}{\partial x}$. We can turn these equations into an energy function based on finite differences

$$\begin{array}{c} E_c = \sum_{(i,j)\in V} \left((v_{i+1,j} - v_{i,j}) + (u_{i,j+1} - u_{i,j})\right)^2 \\ + \sum_{(i,j)\in V} \left((u_{i+1,j} - u_{i,j}) - (v_{i,j+1} - v_{i,j})\right)^2. \end{array} \tag{4.8}$$

Figure 4.3: Insufficiently strong weighting of the homography constraint can lead to distortions. This example contains a vertical line constraint inside a homography constraint (see Figure 4.1). A low weight on the homography constraint produces a wavy distortion on the building facade.

We measure the smoothness of the mapping using the Hessian of $\mathbf{u}(\mathbf{x})$, an order 3 tensor containing all the second partial derivatives, which is the rate of change of the Jacobian matrix: $H = \frac{\partial J}{\partial \mathbf{u}}$. If the mapping is perfectly smooth, the Jacobian will be constant and $H \equiv 0$. We again compute second derivatives using finite difference approximations and use the Frobenius norm to give the smoothness energy as

$$E_s = \sum_{(i,j)\in V} \left\| \begin{bmatrix} u_{i,j+1} - 2u_{i,j} + u_{i,j-1} \\ v_{i,j+1} - 2v_{i,j} + v_{i,j-1} \\ u_{i+1,j} - 2u_{i,j} + u_{i-1,j} \\ v_{i+1,j} - 2v_{i,j} + v_{i-1,j} \\ u_{i+1,j+1} - u_{i+1,j} - u_{i,j+1} + u_{i,j} \\ v_{i+1,j+1} - v_{i+1,j} - v_{i,j+1} + v_{i,j} \end{bmatrix} \right\|^2, \tag{4.9}$$

where the partial derivatives have been rearranged here into a column vector.

4.2.8 Optimization

The total energy for the optimization is a weighted sum of the energy terms defined in the previous section.

$$\begin{aligned} E = w_h^2 \sum_{planes} E_h + w_{hc}^2 \sum_{pairs} E_{hc} + w_l^2 \sum_{lines} E_l + w_v^2 \sum_{vanpts} E_v \\ + w_p^2 \sum_{fixedpts} E_p + w_b^2 E_b + w_c^2 E_c + w_s^2 E_s. \end{aligned} \tag{4.10}$$

We weight the homography, homography compatibility, straight line, vanishing point, fixed point, and border energy terms very strongly compared to the shape preservation energy terms, because they are meant to mimic hard constraints. We determined the weights by experimenting with different values and visually inspecting the results. Although we found a large range of relative weights to work well, for all the results in this paper we use weights of $w_h = w_{hc} = 200$ and $w_l = w_v = w_p = w_b = 100$ for the energy terms that are meant to approximate hard constraints, and $w_s = 12$ and $w_c = 1$ for the weak constraints. The homography and homography compatibility constraints are weighted especially strongly compared to the other strongly weighted constraints. With lower weights on these two constraints we noticed visible deviations from true homographies (Figure 4.3), and tearing between neighboring planar regions.

In some cases the constraints will directly oppose each other. For example, constraining a region with a homography constraint can make that region less conformal overall, so it is important that the homography constraint be weighted much more strongly than conformality. The combination of these two constraints will result in the homography that is most conformal, while still satisfying other constraints.

None of our constraints are necessarily satisfied absolutely, because even the strongly weighted constraints are defined by least-squares energy terms. Global interactions between the constraints can lead to cases where, for example, vertically constrained lines are not quite vertical, or lines do not intersect an associated vanishing point exactly. Usually in cases where such deviations are noticeable the user has specified a set of constraints that is impossible or difficult to satisfy exactly.

The total energy we minimize is a least-squares function and is nonlinear because of the line and homography constraints. We optimize the warp using the Gauss-Newton method with a backtracking line-search procedure to ensure the residual error is reduced at each iteration [33]. At each iteration we update the mapping parameters as $\mathbf{x}_{t+1} = \mathbf{x}_t + \alpha\Delta$, where α scales the Gauss-Newton step Δ. We initialize α to 1, and if the error increases in any iteration we divide α by two and recompute $\mathbf{x}_{t+1}$. The optimization time is dominated by solving the linear system at each iteration; we use the PARDISO sparse direct solver [40]. For a single linear system a significant portion of the solver's time is spent analyzing the sparsity pattern of the matrix; however, the sparsity pattern is constant for all iterations and we perform this analysis only once as a preprocess. Finally, we render the mesh using bilinear texture-mapping.

4.3 Results

We have tested our tool on a variety of images and with a number of different goals; our results are shown in Figures 1.8,4.1,4.4-4.9. We group these results into several different categories.

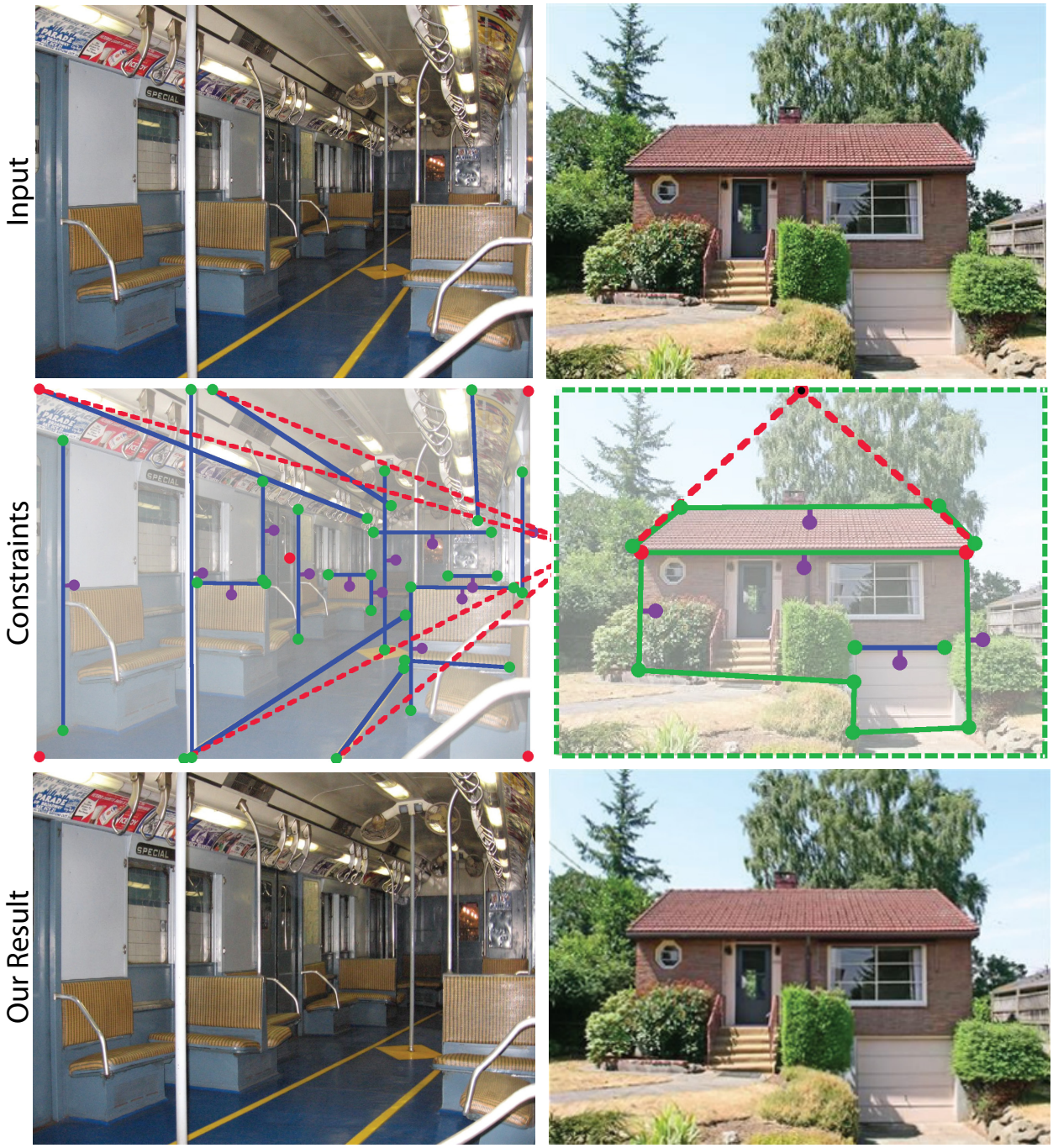


Figure 4.4: **Left**: This train interior was shot with a wide angle lens. We generate a more telephoto appearance from a viewpoint that is unreachable without exiting the train. We move the vanishing point for the receding parallel lines further from the center of the image. We also add vertical and horizontal constraints to preserve straight lines within the scene. Our result appears to flatten the space, as if it were shot with a longer focal-length lens. **Right**: The input image is a frontal perspective of a house, but the viewpoint is slightly off-center and the vanishing point of the roof is not aligned with the center of the house. We center the vanishing point and fix the misalignment.

4.3.1 Viewpoint Shift

In some examples we simulate perspectives that are physically realizable, but were not captured in the input image, either by mistake or because the viewpoints were not reachable. In Figure 1.8 we show two manipulations of an image to make it look more like a wide-angle photo shot from up-close, or more like a telephoto image taken from far away. In Figure 4.1 we show various edits which correspond to several different perspectives of a building. Figure 4.4-left simulates a viewpoint of a subway car that is further away than the confined space would allow. This effect is often called a *dolly-zoom* or *Vertigo* effect when applied to video. Figure 4.4-right shows a shift to a centered viewpoint on a house which could not be achieved with a simple transformation like a single homography. Figure 4.8-bottom shows an image of buildings on a street corner that could have been created by a view camera from the appropriate viewpoint (but not from the viewpoint captured).

4.3.2 Non-photorealistic Perspectives

In other examples, we show perspectives that are not physically realizable by normal cameras, similar to those sometimes used by artists. In Figure 4.5 we take two walls in a room that should share a single vanishing point and separate their converging lines into two vanishing points. Figure 4.6-top shows an oblique projection, which is commonly used in architectural renderings, but impossible to capture with a standard camera. In Figure 4.6-bottom and Figure 4.8-top we show orthographic projections of cityscapes. The results are not truly orthographic in that occlusions do not change and the sizes of objects still diminish with distance, but the lack of converging lines gives a qualitatively more orthographic effect, as if the camera were zooming in from a distance. These results also exhibit a mix of perspectives, as the larger buildings appear orthographic, but some of the smaller buildings do not.

4.3.3 Compositing

Another usage scenario is matching perspective for compositing, which is a frequent problem when combining images [13]. In the three examples in Figure 4.7, the perspectives of the composited objects do not match the target scenes. We reconcile the disparate perspectives by aligning their vanishing points using our warping tool.

Finally, we compare our results to both standard techniques and ground-truth. For most of our examples the constraints are not satisfiable by a standard transform such as a homography, and attempting to find a homography that best matches our results demonstrates the advantage of our approach (Figure 4.8-top). One case where the constraints can be met by a single homography is Figure 4.8-bottom, but our result is still more faithful to the shapes in the input. We can compute the best-fit single-homography version of any result by adding a single polygon around the entire image. In cases where a single homography will suffice, our tool still provides an attractive alternative to the standard interface of dragging four corners of an image or manipulating a virtual camera. Instead, the user directly specifies the desired

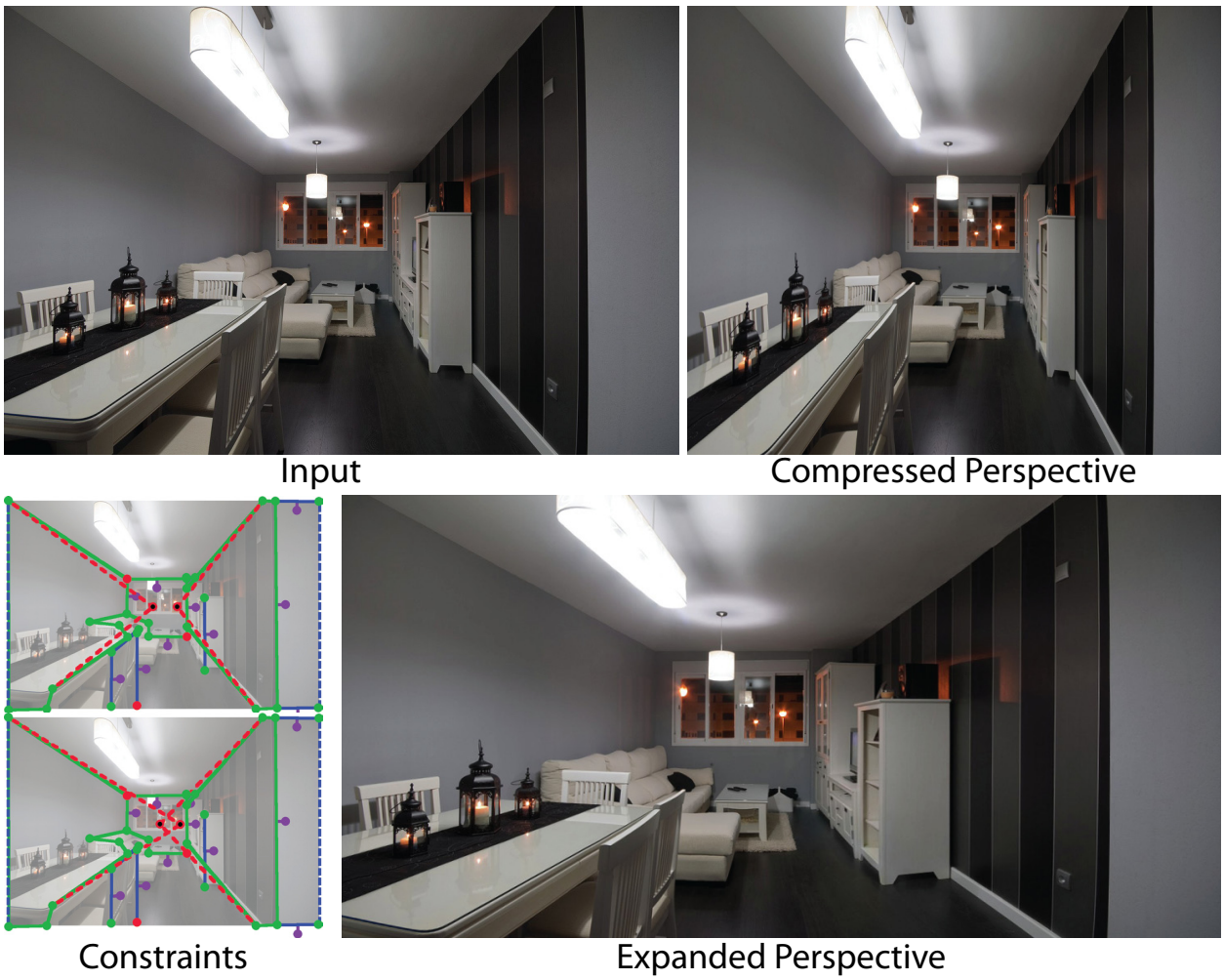


Figure 4.5: Two perspective manipulations of a room. The left and right walls are parallel and therefore share a vanishing point at the center of the input image. We create two non-photorealistic perspectives by separating the vanishing points of the left and right walls. Pulling each vanishing point in towards its corresponding wall creates the perception of a thinner, longer room, while crossing the vanishing points makes the room appear wider.

Figure 4.6: **Top**: We transform the building so that it appears as it would under an oblique projection. Parallel lines in the scene remain parallel in our result and the front of the building is parallel to the image plane. **Bottom**: We simulate an orthographic view of downtown Los Angeles by constraining the faces of buildings with homographies and constraining all horizontal lines to be horizontal in the image.

image-space properties. In Figure 4.9, we use computer graphics renderings to compare to ground-truth. Our results are visually plausible and look quite similar to the ground-truth renderings, but there are differences in occlusions and object shapes.

4.3.4 Performance

Our tool does not currently respond in real-time to user-specified constraints, but it does respond quickly enough to allow the user to quickly explore many iterations of constraints. For all our results we use a mesh 50 quads wide with height dependent on aspect ratio, i.e., 50x50 for a square image. On average we used 2155 vertices and the optimization averaged 12 iterations and 3.37 seconds (run on a 2.4GHz Intel Core 2 Duo). Most of the user's time is spent specifying constraints, mostly polygon edges and line segments. On average we specified a total of 22 line segments and polygon edges per image.

4.4 Discussion and Limitations

Our results demonstrate various ways our system can be used to manipulate perspective by interacting with the lines, planes, and vanishing points in an image; however, its applicability

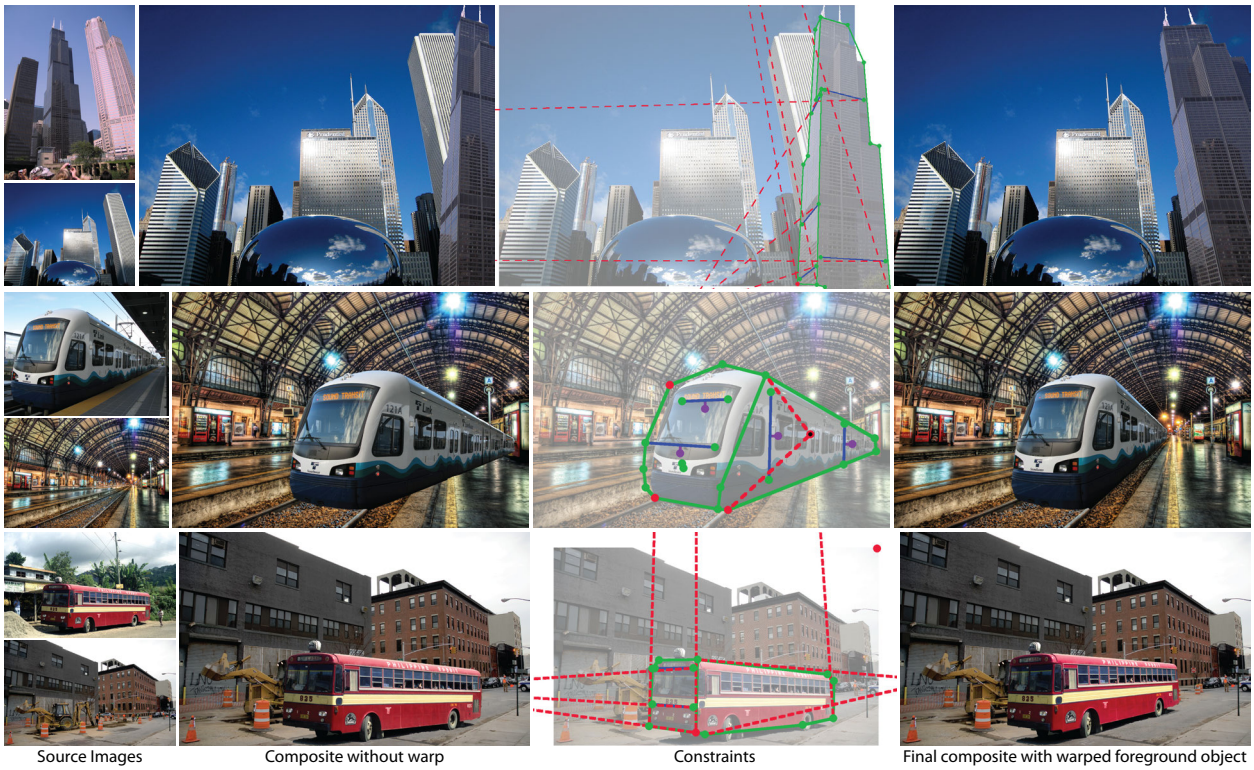


Figure 4.7: Three compositing examples where we match the perspective of an object to a target scene. **Top**: The black building from the source image looks inconsistent when pasted into the target image, until we warp it so all the vanishing points match. **Middle**: The vanishing point of the lines along the length of the train are shifted to match the vanishing point of the train tracks, and the horizontal lines along the front face of the train are constrained to be level. **Bottom**: All three vanishing points of the mutually orthogonal bus faces are aligned with the three vanishing points of the buildings.

is limited. From the outset we designed our tool around an image warping paradigm which ignores three dimensional effects like parallax and thus it can not be used to reproduce these effects, as demonstrated in Figure 4.10. This example also demonstrates another limitation of our warping technique; it will not generally produce a rectangular image. Most results need to be manually cropped, unless border constraints are used on all four sides. Furthermore, as with all warping techniques, rendering may introduce sampling artifacts, especially in areas undergoing a scale increase. For example, decreasing the foreshortening of a plane will require upsampling of the distant end.

Our tool is designed primarily around manipulating planes and straight lines and is thus well suited to man-made scenes, a property that is also true for the perspective constructions used by painters. Natural scenes do demonstrate perspective effects like foreshortening and diminution, but they do not typically contain strong parallel lines converging to vanishing

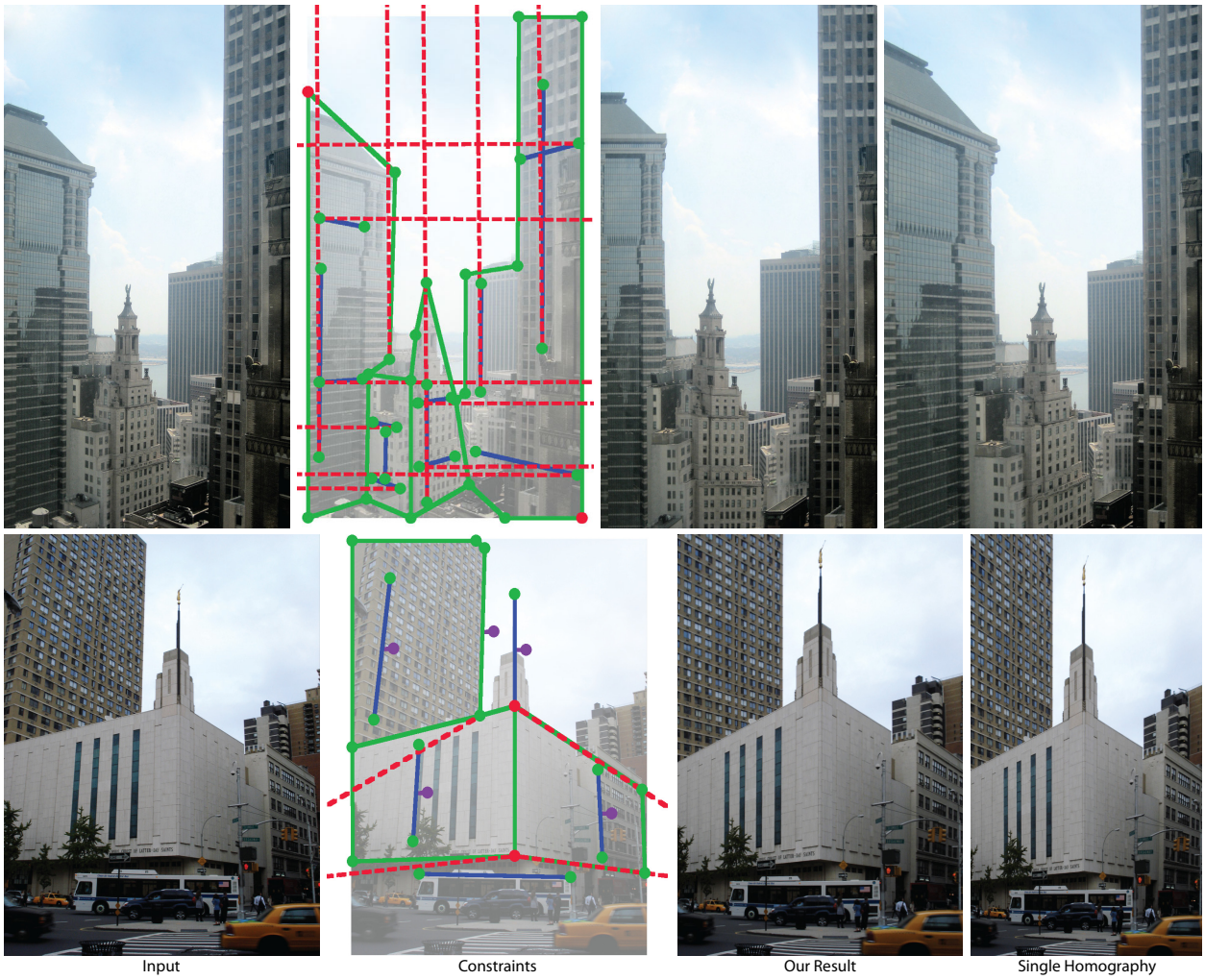


Figure 4.8: **Top**: We simulate an orthographic projection of a cityscape. We represent the complex geometry roughly with four polygons and move the vanishing points off to infinity. We also create a homography that best matches the constraints, which we compute by drawing a polygon around the entire image. Unlike our result the homography does not achieve the desired orthographic look. **Bottom**: We warp an image so that both buildings are upright and the church is shown from a straight-on viewpoint, where both left and right faces recede at the same angle. To create our result we constrain two vanishing points and several vertical lines. A single homography can satisfy these constraints; however, this homography horizontally compresses the two buildings compared to the input and our result.

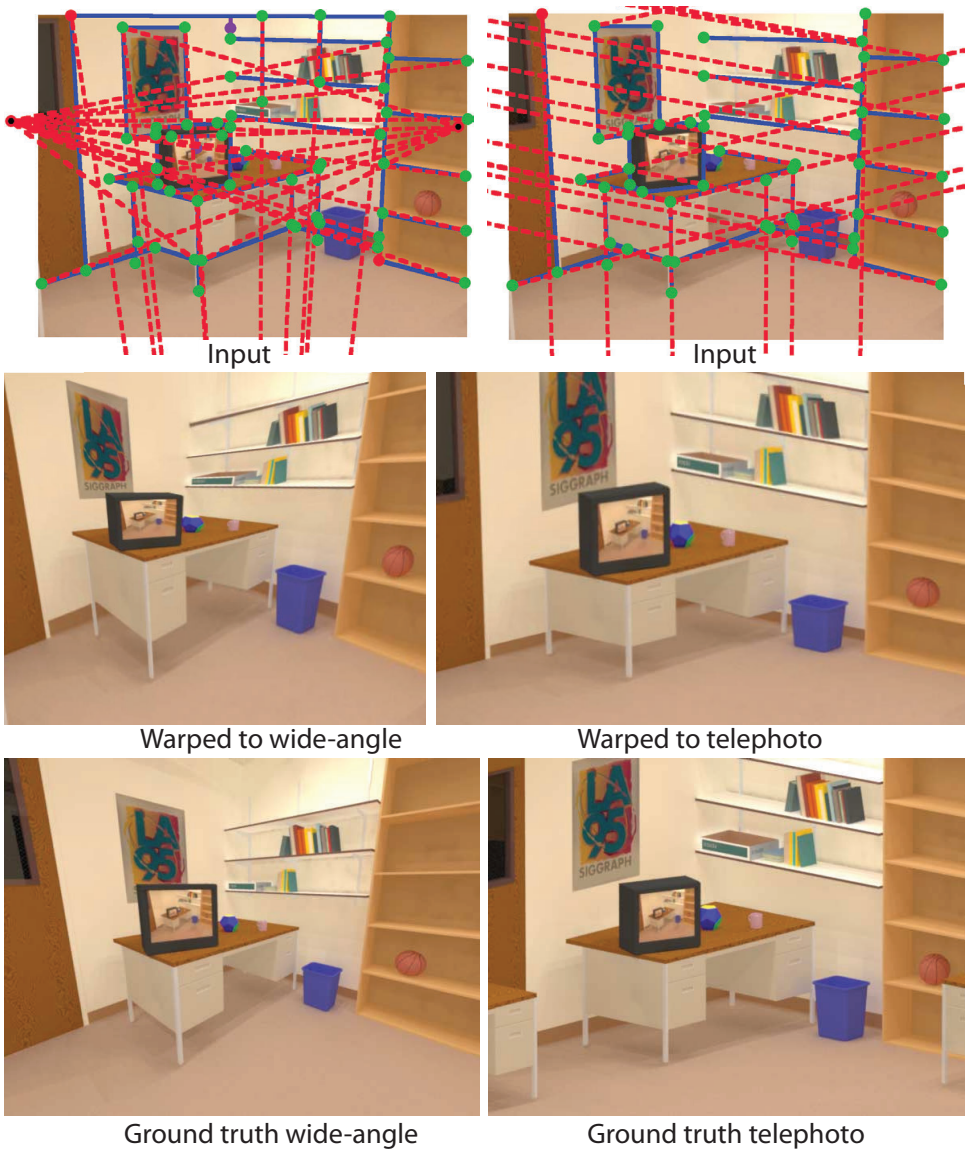


Figure 4.9: In this example taken from the work of Zorin and Barr [62], we compare our approach to ground-truth for a rendered computer graphics scene. We begin with an image rendered with a standard focal length and create warps that resemble wide-angle and telephoto renderings. Our warps look plausible, but compared to ground-truth differences can be seen in visibility and the shape of individual objects.

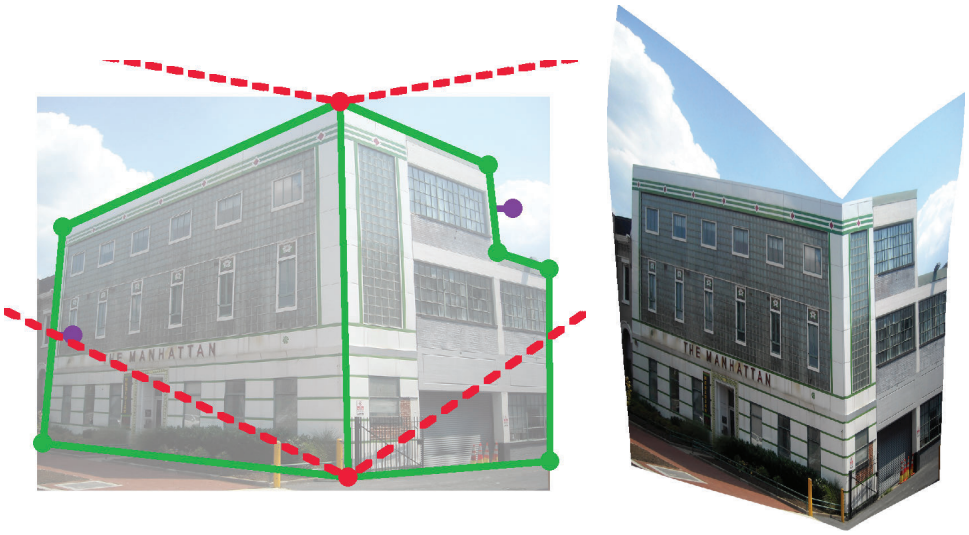


Figure 4.10: We simulate a view looking down on the building from above. Moving the vanishing points upward gives the desired perspective for the facades. However, the roof is not visible in the input image and the building looks hollow in the result.

points. While it is acceptable for more organic textures to be present in the images we manipulate, and such textures can even be useful for hiding distortion, they should not be the focus of the manipulation.

The interface of our tool does have a learning curve, and a user needs to understand the basic principles of perspective construction, such as vanishing points and lines, in order to use it. However, we expect that users interested in manipulating photographic perspective are likely to be familiar with these concepts. Also, it is quite possible for a user to input a set of constraints that are so far from the original image that they cannot be satisfied without extreme distortion. As future work, we would like to explore constraining the interface to only allow feasible constraint configurations. Finally, the shape preservation terms of the energy function are rotation invariant, but are not scale invariant. Mappings that globally reduce image size can produce lower energies for these terms. If the user does not enter at least two fixed point constraints or several boundary constraints, the output will tend to rotate and shrink, possibly to a single point. As future work we would like to explore additional default constraints that prevent such rotations or shrinkage.

Chapter 5

Conclusion and Future Work

In this dissertation we investigated improvements to practical application of the perspective projection, which is used in the vast majority of all photographs. Taking inspiration from current understanding of human perception of photos, as well as the long history of artistic use of the perspective projection, we developed interactive tools that let photographers modify the projection toward two goals: reducing distortion in wide-angle photographs and artistic manipulation of perspective composition. Our general framework for building these tools is based on image warping guided by energy functions measuring local distortion and constraints provided through an interactive drawing interface.

In Chapter 3 we developed a technique for creating wide-angle images that resemble standard perspective images, but contain much less perceived distortion. Producing a wide-angle image free of distortion is challenging because there is no global projection which is free of all types of perceptual distortion. The perspective projection keeps all lines in the scene straight, but stretches objects. Conversely, conformal projections don't stretch objects, but they bend straight lines. Our primary contribution in this work is an optimization approach for computing a spatially varying mapping from the viewing sphere to a planar image that preserves user-specified lines while minimizing a set of energy terms that measure wide-angle image distortion.

In Chapter 4 we built on the framework developed for the wide-angle tool to develop a technique for flexibly controlling the perspective composition. While photographers control composition via primary geometry (i.e. physically moving the camera), other artists make direct use of secondary geometry by placing vanishing points and construction lines. Our perspective manipulation tool addresses this distinction with a set of constraints based on projective geometry. Users can annotate an image by marking planar regions of the scene, straight lines, and associated vanishing points and then use these constraints as handles to control a warp.

5.1 Future Work

5.1.1 Video

The tools developed in this dissertation were designed for creating visually pleasing static images. The extension of our techniques to video introduces additional challenges in terms of both the warp optimization and the user interface. The warps produced by our tools have irregular bending features and sharp kinks that are hidden by the content of the static image. If one of these warps were applied, for example, to a video from a static camera with moving foreground objects, these features of the warp could be visually objectionable. Video from a moving camera amplifies this issue and also makes specifying constraints more challenging. Wei et al. extended our wide-angle projection technique to video by adding temporal coherence energy terms and keyframed constraint propagation, but only demonstrate results on short video with a static camera or simple panning motion [53].

5.1.2 Beyond Warping

We based the techniques presented in this dissertation on image warps, which are continuous mappings. Relaxing this continuity requirement could provide an additional degree of freedom for the user. For example, instead of trying to compress a textured region of an image into a smaller space, a better solution might be to simply remove some of the texture. Doing this automatically while maintaining the appearance of a seamless image is a challenging problem. Bidirectional similarity is one potential alternative to warping that has the capable of supporting these kind of edits [43]. Originally applied to the image retargeting problem, bidirectional similarity ensures that each patch in the original image is present in the output image (completeness) and vice-versa (coherence). Barnes et al. describe an accelerated optimization technique for bidirectional similarity that allows it to work interactively and showed that user specified constraints could be imposed [5]. Reformulating the applications described in this dissertation in terms of something like bidirectional similarity might alleviate some of the limitations imposed by the warping framework.

5.1.3 Perceptual Evaluation

While our wide-angle projection tool was designed to reduce perceptual distortion, we have not rigorously evaluated the true effect these projections have on a viewer's perception of a scene. Although they contain wide fields of view, the images produced with our technique more closely resemble narrow field of view perspective images, as traditionally only narrow FoV images have all straight lines and little shape distortion. This could potentially lead a viewer to interpret the images as if they had a narrow FoV, distorting his or her perception of the spatial arrangement of objects in the scene. For example, objects on opposite sides of one of our wide-angle projections might appear to both be in front of the viewer, when in fact they are on opposite sides of the viewer. Global projections like Mercator and stereographic

may suffer from this issue as well, but their bending of lines may also provide viewers with a cue that a non-standard projection is being used. Our projections lack this cue. Conducting user-studies designed to evaluate how these different projection affect our perception, as well as isolate the effect of various cues, could illuminate new directions to further improve wide-angle projections.

Bibliography

[1] Aseem Agarwala. "Authoring effective depictions of reality by combining multiple samples of the plenoptic function". PhD thesis. University of Washington, 2006.

[2] Aseem Agarwala et al. "Photographing long scenes with multi-viewpoint panoramas". In: *ACM Trans. on Graph.* 25.3 (July 2006), pp. 853–861.

[3] Maneesh Agrawala, Denis Zorin, and Tamara Munzner. "Artistic Multiprojection Rendering". In: *Rendering Techniques 2000: 11th Eurographics Workshop on Rendering*. June 2000, pp. 125–136.

[4] Shai Avidan and Ariel Shamir. "Seam Carving for Content-Aware Image Resizing". In: *ACM Trans. on Graph.* 26.3 (July 2007), 10:1–10:9.

[5] Connelly Barnes et al. "PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing". In: *ACM Transactions on Graphics (Proc. SIGGRAPH)* 28.3 (Aug. 2009).

[6] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. Cambridge, MA: O'Reilly, 2008.

[7] Robert Carroll, Aseem Agarwala, and Maneesh Agrawala. "Image warps for artistic perspective manipulation". In: *ACM Trans. on Graph.* 29.4 (Aug. 2010). ISSN: 0730-0301.

[8] Robert Carroll, Maneesh Agrawala, and Aseem Agarwala. "Optimizing content-preserving projections for wide-angle images". In: *ACM Trans. on Graph.* 28.3 (Aug. 2009), pp. 1–9. ISSN: 0730-0301.

[9] Gaurav Chaurasia, Olga Sorkine, and George Drettakis. "Silhouette-Aware Warping for Image-Based Rendering". In: *Computer Graphics Forum (Proceedings of the Eurographics Symposium on Rendering)* 30.4 (2011).

[10] Patrick Coleman and Karan Singh. "Ryan: rendering your animation nonlinearly projected". In: *NPAR 2004*. June 2004, pp. 129–138.

[11] A. Criminisi, I. Reid, and A. Zisserman. "Single View Metrology". In: *Int. J. Comput. Vision* 40.2 (2000), pp. 123–148. ISSN: 0920-5691.

[12] Joseph D'Amelio. *Perspective Drawing Handbook*. Dover Publications, 2004.

[13] Katrin Eismann. *Photoshop Masking & Compositing*. New Riders Press, 2004.

[14] Albert Flocon and André Barre. *Curvilinear Perspective: From Visual Space to the Constructed Image*. UC Press, 1988.

[15] David A. Forsyth and Jean Ponce. *Computer Vision: A Modern Approach*. Prentice Hall, Aug. 2002. ISBN: 0130851981.

[16] Ran Gal, Olga Sorkine, and Daniel Cohen-Or. "Feature-Aware Texturing". In: *Rendering Techniques 2006: 17th Eurographics Workshop on Rendering*. June 2006, pp. 297–304.

[17] Peter M. Hall et al. "RTcams: A New Perspective on Nonphotorealistic Rendering from Photographs". In: *IEEE Transactions on Visualization and Computer Graphics* 13 (2007), pp. 966–979. ISSN: 1077-2626.

[18] Pat Hanrahan. *Realistic or Abstract Imagery*. 2005. URL: `http://www.graphics.stanford.edu/~hanrahan/talks/realistic-abstract/walk013.html`.

[19] Paul S. Heckbert. *Fundamentals of Texture Mapping and Image Warping*. Tech. rep. UCB/CSD-89-516. EECS Department, University of California, Berkeley, June 1989.

[20] D. Hilbert and S. Cohn-Vossen. *Geometry and the Imagination*. New York: Chelsea, 1952.

[21] David Hockney. *Secret Knowledge: Rediscovering the Lost Techniques of the Old Masters*. New York: Viking Studio, 2006, p. 328.

[22] Derek Hoiem, Alexei A. Efros, and Martial Hebert. "Automatic photo pop-up". In: *ACM Trans. on Graph.* 24.3 (Aug. 2005), pp. 577–584.

[23] Youichi Horry, Ken-Ichi Anjyo, and Kiyoshi Arai. "Tour into the picture: using a spidery mesh interface to make animation from a single image". In: *Proc. SIGGRAPH*. 1997, pp. 225–232.

[24] Takeo Igarashi, Tomer Moscovich, and John F. Hughes. "As-rigid-as-possible shape manipulation". In: *ACM Trans. on Graph.* 24.3 (Aug. 2005), pp. 1134–1141.

[25] Laurent Itti, Christof Koch, and Ernst Niebur. "A Model of Saliency-Based Visual Attention for Rapid Scene Analysis". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 20.11 (1998), pp. 1254–1259. ISSN: 0162-8828.

[26] Z. Karni, D. Freedman, and C. Gotsman. "Energy-based image deformation". In: *Proceedings of the Symposium on Geometry Processing*. SGP '09. Berlin, Germany: Eurographics Association, 2009, pp. 1257–1268.

[27] Michael Kazhdan, Dinoj Surendran, and Hugues Hoppe. "Distributed gradient-domain processing of planar and spherical images". In: *ACM Trans. Graph.* 29.2 (Apr. 2010), 14:1–14:11. ISSN: 0730-0301.

[28] Philipp Krähenbühl et al. "A system for retargeting of streaming video". In: *ACM Trans. Graph.* 28.5 (Dec. 2009), 126:1–126:10. ISSN: 0730-0301.

[29] Michael Kubovy. *The psychology of perspective and renaissance art*. Cambridge University Press, 1986.

[30] Manuel Lang et al. "Nonlinear Disparity Mapping for Stereoscopic 3D". In: *ACM Trans. Graph.* 29.3 (2010), p. 10.

[31] Feng Liu et al. "Content-preserving warps for 3D video stabilization". In: *ACM Trans. Graph.* 28.3 (July 2009), 44:1–44:9.

[32] William Longfellow. *Applied Perspective for Architects and Painters*. The Riverside Press, 1901.

[33] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. 2nd. New York: Springer Verlag, 2006.

[34] Daniele Panozzo, Ofir Weber, and Olga Sorkine. "Robust Image Retargeting via Axis-Aligned Deformation". In: *Computer Graphics Forum (proceedings of EUROGRAPHICS)* 31.2 (2012), pp. 229–236.

[35] M. H. Pirenne. *Optics, Painting & Photography*. Cambridge University Press, 1970.

[36] Voicu Popescu, Paul Rosen, and Nicoletta Adamo-Villani. "The graph camera". In: *ACM Trans. on Graph.* 28.5 (Dec. 2009), pp. 1–8.

[37] Paul Rademacher and Gary Bishop. "Multiple-Center-of-Projection Images". In: *Proc. SIGGRAPH*. July 1998, pp. 199–206.

[38] Ashutosh Saxena, Min Sun, and Andrew Y. Ng. "Make3D: Learning 3D Scene Structure from a Single Still Image". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 31.5 (2009), pp. 824–840.

[39] Scott Schaefer, Travis McPhail, and Joe Warren. "Image deformation using moving least squares". In: *ACM Trans. on Graph.* 25.3 (July 2006), pp. 533–540.

[40] O. Schenk and K. Grtner. "Solving Unsymmetric Sparse Systems of Linear Equations with PARDISO". In: *Journal of Future Generation Computer Systems* 20.3 (2004), pp. 475–487.

[41] Thomas K Sharpless, Bruno Postle, and Daniel M German. "Pannini: A New Projection for Rendering Wide Angle Perspective Images". In: *International Symposium on Computational Aesthetics*. London, 2010.

[42] Alla Sheffer, Emil Praun, and Kenneth Rose. "Mesh parameterization methods and their applications". In: *Found. Trends. Comput. Graph. Vis.* 2.2 (2006), pp. 105–171.

[43] Denis Simakov et al. "Summarizing visual data using bidirectional similarity." In: *CVPR*. IEEE Computer Society, Feb. 6, 2009.

[44] John P. Snyder. *Flattening the Earth, two thousand years of map projections*. University of Chicago Press, 1993.

[45] John P. Snyder. *Map Projections – A Working Manual: U.S. Geological Survey Prof. Paper 1395*. U.S. Gov. Printing Office, 1987.

[46] Olga Sorkine and Marc Alexa. "As-Rigid-As-Possible Surface Modeling". In: *Proceedings of EUROGRAPHICS/ACM SIGGRAPH Symposium on Geometry Processing*. 2007, pp. 109–116.

[47] Leslie Stroebel. *View Camera Technique*. 7th. Focal Press, 1999.

[48] Paul Viola and Michael J. Jones. "Robust Real-Time Face Detection". In: *International Journal of Computer Vision* 57.2 (2004), pp. 137–154.

[49] D. Vishwanath, A.R. Girshick, and M.S. Banks. "Why pictures look right when viewed from the wrong place". In: *Naure Neuroscience* 8.10 (2005), p. 1401.

[50] O. Wang et al. "StereoBrush: interactive 2D to 3D conversion using discontinuous warps". In: *Proceedings of the Eighth Eurographics Symposium on Sketch-Based Interfaces and Modeling*. SBIM '11. Vancouver, British Columbia, Canada: ACM, 2011, pp. 47–54.

[51] Yu-Shuen Wang et al. "Motion-based Video Retargeting with Optimized Crop-and-Warp". In: *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)* 29.4 (2010), article no. 90.

[52] Yu-Shuen Wang et al. "Optimized Scale-and-Stretch for Image Resizing". In: *ACM Trans. on Graph.* 27.5 (Dec. 2008), 118:1–118:8.

[53] Jin Wei et al. "Fisheye Video Correction". In: *IEEE Transactions on Visualization and Computer Graphics* 18.10 (2012), pp. 1771–1783. ISSN: 1077-2626.

[54] Lior Wolf, Moshe Guttmann, and Daniel Cohen-Or. "Non-homogeneous Content-driven Video-retargeting". In: *IEEE International Conference on Computer Vision*. 2007.

[55] Daniel N. Wood et al. "Multiperspective Panoramas for Cel Animation". In: *Proc. SIGGRAPH*. Aug. 1997, pp. 243–250.

[56] WWT. *WorldWide Telescope*. 2008. URL: `http://www.worldwidetelescope.org/`.

[57] Jingyi Yu and Leonard McMillan. "General Linear Cameras". In: *European Conference on Computer Vision*. May 2004.

[58] Lihi Zelnik-Manor and Pietro Perona. "Automating joiners". In: *NPAR '07: Proceedings of the 5th international symposium on Non-photorealistic animation and rendering*. 2007, pp. 121–131.

[59] Lihi Zelnik-Manor, Gabriele Peters, and Pietro Perona. "Squaring the Circles in Panoramas". In: *IEEE International Conference on Computer Vision (ICCV)*. Vol. 2. 2005, pp. 1292–1299.

[60] Guo-Xin Zhang et al. "A Shape-Preserving Approach to Image Resizing". In: *Computer Graphics Forum* 28.7 (2009), pp. 1897–1906.

[61] Shizhe Zhou et al. "Parametric Reshaping of Human Bodies in Images". In: *ACM Transactions on Computer Graphics: Special Issue of ACM SIGGRAPH 2010* 29.4 (2010).

[62] Denis Zorin and Alan H. Barr. “Correction of Geometric Perceptual Distortion in Pictures”. In: *Proc. SIGGRAPH*. Aug. 1995, pp. 257–264.