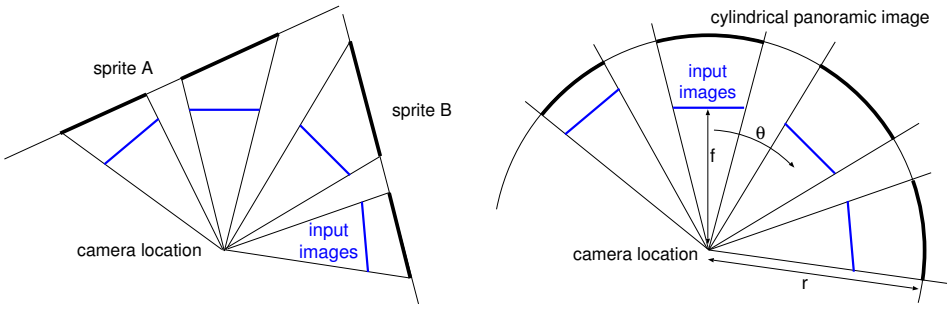
The important thing in science is not so much to obtain new facts as to discover new ways of thinking about them.
(Sir William Bragg)

Chapter 14

Panoramic Video and Floor Plan Reconstruction

Previous chapters often used background-sprite images to create a panoramic overview of the scene environment. The generated background image was modeled as a single, large image on a plane. However, other representations of panoramic images are possible that have certain advantages for other applications. The most frequently-used model is the cylindrical panorama, which allows to capture a 360-degree horizontal view in a single image. This chapter describes the geometry of cylindrical panoramic images and presents various techniques for capturing panoramic images and videos. Since cylindrical images are a special kind of image with geometric distortions, their contents are not always easy to interpret. Therefore, different visualization techniques are explored providing images that are easier to understand. In particular, a new visualization technique is proposed that reconstructs the geometry of the room in which the panoramic image was recorded, and which uses this room reconstruction to show the panoramic image as texture maps on this virtual room. Finally, this concept is generalized to reconstruct a complete floor plan based on multiple panoramic images. Additionally to the aforementioned improved visualization, this enables new applications like the presentation of real estate with virtual tours through the appartment.



(a) Multi-sprite. (b) Cylindrical.

Figure 14.1: *Background image models.*

14.1 Introduction

14.1.1 From background sprites to panoramic images

Chapter 5 described that global-motion parameters can be used to combine several images captured from the same scene into a large scene overview image. A transformation model was used that is compatible with the MPEG-4 sprite coding tools to use these synthesized background images in an object-oriented encoding system. Chapter 6 showed that the projective motion model used in MPEG-4 does not support the synthetization of images covering a large viewing angle. Since the chosen motion model was limited by the possibilities of the MPEG-4 standard, we splitted the background image into several independent images, each covering part of the scene (Fig. 14.1(a)). However, by choosing a different transformation, it is indeed possible to combine images captured during a complete 360-degree pan into a larger panoramic image. One of the most frequently-used models for this is the cylindrical panorama. The principle is to project the surrounding scene onto a virtual cylinder surface around the camera (Fig. 14.1(b)). Unrolling this cylindrical surface gives a rectangular image that comprises the full 360-degree view.

Cylindrical panoramic images are used in a wide variety of applications. One of their advantages is that the extended field of view, compared to the limited view of normal images, fits better to the field of view of the human eye. Hence, panoramic images provide an ideal visualization of landscape-type images. Moreover, 360-degree images show the whole surrounding in only one picture. This is ideal for presentations of hotel rooms or real-estate, because it gives a complete impression of the environment.

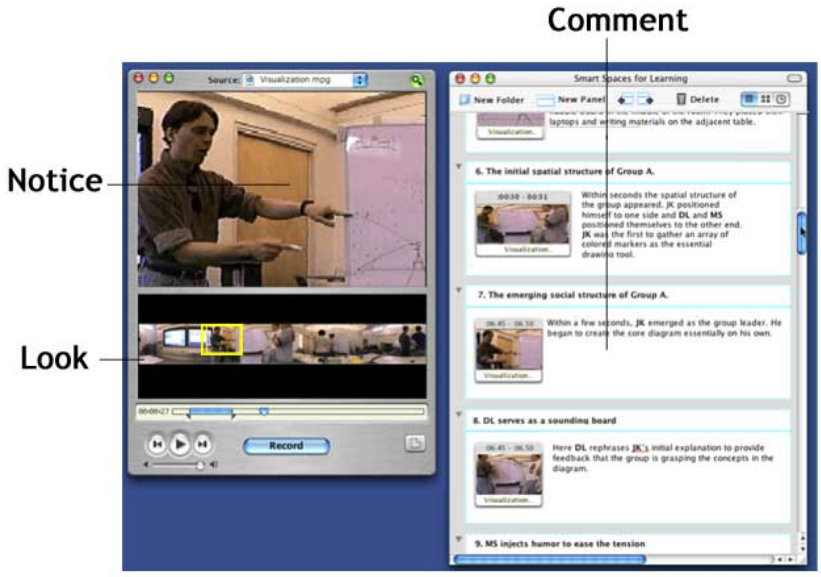


Figure 14.2: *The Diver video annotation software. Magnified views can be extracted from the panoramic video and these clips can be described with textual annotations.*

Finally, since the panoramic image covers the complete environment, it is not necessary to select a suitable view while the image is captured. Instead, events can be recorded in a panoramic video and selections of the most interesting parts of the scene can be made later. This last aspect has been extensively studied in the *Diver* project [142] at the Stanford Center for Innovations in Learning (to which the author had the possibility to contribute). In this project, panoramic videos of classroom education scenes were recorded from the full event. Later, psychologists could analyze the teaching methods and reaction of the students. For this purpose, video clips can be annotated with comments, where a clip is not only a temporal selection out of the video material, but also a spatially restricted view into the full panoramic overview (Fig. 14.2).

14.1.2 Visualization of panoramic images

One important aspect of panoramic imaging is the presentation of the images to the viewer. Directly showing the texture of the unrolled cylinder surface results in a rectangular image, but includes geometric distortions. Moreover, since this image shows all directions around the camera at the same time, the panoramic image itself can be confusing to the viewer.

For this reason, panoramic images are often presented with an interactive *panoramic image browser* (PIB) application which shows a geometrically rectified sub-view of the scene, as if it were captured with a user-controlled virtual camera. The disadvantage of this representation is that it is not possible to offer a fast overview of the scene, and it is not possible to see the complete environment on a static medium like a paper copy. In this chapter, we propose a new visualization technique for panoramic images that is specialized for images captured inside rectangular rooms, which is an important special case that covers many application areas like hotel room advertising or recording of group meetings.

Our visualization is based on an algorithm to reconstruct the 3-D layout of the rectangular rooms from the panoramic image. Once the geometry of the room is known, a 3-D model of the room walls can be synthesized and the wall textures can be added, using the image data from the panoramic image (Fig. 14.8). The proposed representation provides a flexible way to visualize the scene. On one hand, the virtual camera can be placed outside of the room, such that the viewer gets an overview of the full scene appearance and room layout. On the other hand, the virtual camera can also be placed at the position of the original camera. Interactively rotating this virtual camera provides views that equal the output of the PIB technique.

The room reconstruction requires a minimum of user assistance: the user only has to indicate the position of the four room corners in the panoramic image. First, the reconstruction algorithm converts the positions of the corners into the angle between these corners as observed from the camera position. Subsequently, the room shape and the camera position are determined from these angles, and the textured 3-D model is constructed automatically.

14.1.3 Floor plan reconstruction

Following the same reconstruction principle as for rectangular rooms, we can extend the reconstruction algorithm to support arbitrary room shapes or even complete floor plans, comprising several rooms. This floor plan reconstruction enables new applications additional to the aforementioned visualization, like the presentation of real estate, providing virtual tours through an appartment. Generally, for more complicated room shapes or a reconstruction comprising several rooms, a single panoramic image does not provide enough information for a reconstruction. For this reason, the extended algorithm allows the usage of multiple panoramic images for the reconstruction if required.

14.1.4 Chapter outline

This chapter first briefly introduces the geometry of cylindrical panoramic images and describes techniques to compose panoramic images from a collection of small images or to capture panoramic video with specialized cameras. Section 14.3 discusses different visualization techniques and Section 14.4 proposes a new algorithm for the estimation of the wall sizes of rectangular rooms from captured panoramic images. This algorithm is generalized in Section 14.5 to support the reconstruction of a collection of rooms with arbitrary room shapes from multiple panoramic images.

14.2 Capturing panoramic images and video

The most commonly-used model for panoramic images is the projection onto a cylindrical surface. The cylinder is centered at the camera location and aligned vertically, such that a horizontal camera pan corresponds to a rotation of the cylinder axis. To transform the planar image coordinates (x, y) into cylinder coordinates (θ, h), we use the transformation

$$\tan\theta = x/f \quad \text{and} \quad h = \frac{y \cdot r}{\sqrt{f^2 + x^2}}, \tag{14.1}$$

where f is the focal length (the distance of the image plane to the optical center), and r is the cylinder radius (Fig. 14.3). From these equations, it can be noticed that the cylindrical transformation depends on the focal length f that was used to capture the image. Some digital cameras store the focal length with which an image was recorded in the EXIF metadata. If this is not available, the focal length should be estimated from the image data (see Section 12.2.1). The radius of the cylinder surface can be chosen arbitrarily, since its only effect is a scaling factor for the vertical axis h of the panoramic image. For a practical implementation, we have to consider that images are usually stored with integer pixel positions. Hence, in practice, we set $r = f$ in order to obtain a vertical image resolution in the panoramic image that is approximately the same as in the input image. Geometrically, this means that the input image plane is a tangent plane to the cylinder. It touches the cylinder along $x = 0$, and it follows from Eq. (14.1) that $h = y$ along this line. For the other values of x, it holds that $|h| < |y|$, meaning that there is some loss of resolution in the cylinder projection.

The horizontal axis in the panoramic image represents the rotation angle θ, and we have to define a discretization step-size $\Delta\theta$. Since it is desired to preserve the aspect ratio of the input image pixels for the pixels in the panoramic image, we define the discretization step-size as $\tan\Delta\theta = 1/f$, based on the assumption that the pixel width in the input image is 1.

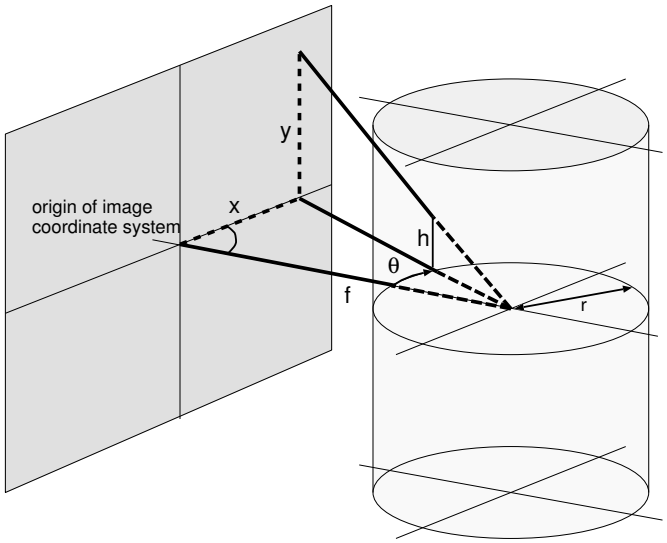


Figure 14.3: *Projection of image coordinates onto cylindrical coordinates.*

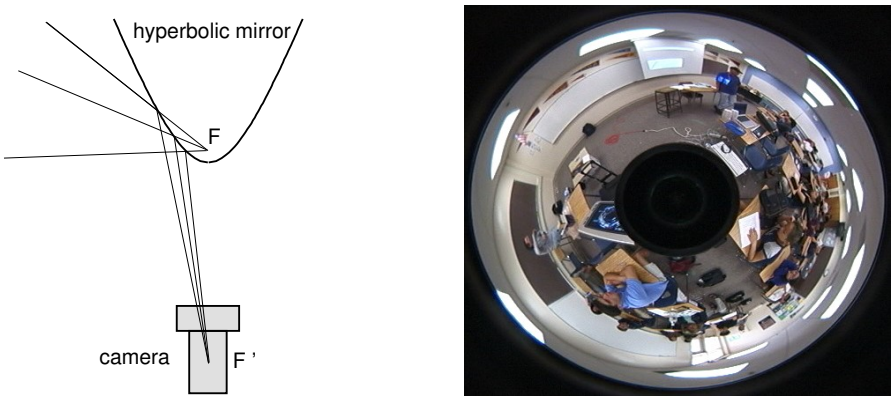
14.2.1 Panoramic image generation

A technique to generate panoramic images is to take a sequence of images while rotating the camera around its vertical axis. It is important to note that the rotation has to be carried out around the optical center, since otherwise the images would not fit together (see Section 2.5.3). Each of these images is first converted to cylindrical coordinates θ, h independently. Because the images were recorded with different camera rotation angles, their position on the cylindrical surface is shifted by some amount θ_i. This shift can be determined easily with a one-dimensional search over θ_i to minimize the image difference

$$E_{ij} = \frac{1}{|\mathcal{A}_{ij}|} \sum_{(\theta,h)\in\mathcal{A}_{ij}} |I_i(\theta - \theta_i, h) - I_j(\theta - \theta_j, h)| \qquad (14.2)$$

in the overlapping image area $\mathcal{A}_{ij}$ of images i and j.

When stitching the individual images together into a single panoramic image, the seams between the images are often visible because of small alignment errors, or because of changes in the illumination conditions between the images. We apply a cross-blending between the two images to obtain a smooth transition. More complex techniques have been proposed for this problem. For example, [34] proposes to determine a path in the overlapping area $\mathcal{A}_{ij}$ from the top to the bottom border that minimizes the



(a) Optics of a hyperbolic mirror.

(b) Example image captured with a hyperbolic mirror.

Figure 14.4: *A 360-degree image recorded using a camera with a parabolic mirror.*

sum of luminance differences along this path. The advantage of this approach is that it also provides a sharp transition if there are moving objects in the scene.

14.2.2 Cameras for recording panoramic videos

For panoramic still images from static environments, we can capture several images sequentially and compose them into one panoramic image. For the recording of panoramic video sequences, the full 360-degree view has to be captured at the same time. This poses the problem of mechanically mounting the cameras such that they cover the complete 360°, but also have an identical optical center. A solution is to place a hyperbolic mirror in front of a camera. A hyperboloid has two focal points with the property that a camera at focal point F' observes a 360-degree image with virtual optical center at the other focal point F (Fig. 14.4). The main disadvantage of this technique is that the image resolution is generally low and unequally distributed in the image. Moreover, the image resolution is generally highest at the floor or ceiling, which are areas that are usually not very important.

In the Diver project, we used a second solution that is based on a setup with several cameras, oriented into different viewing directions to cover the complete 360 degrees. To enable a collision-free mounting of the cameras, mirrors are placed in front of the cameras to redirect the incoming light. With this approach, the cameras can be mounted with sufficient space

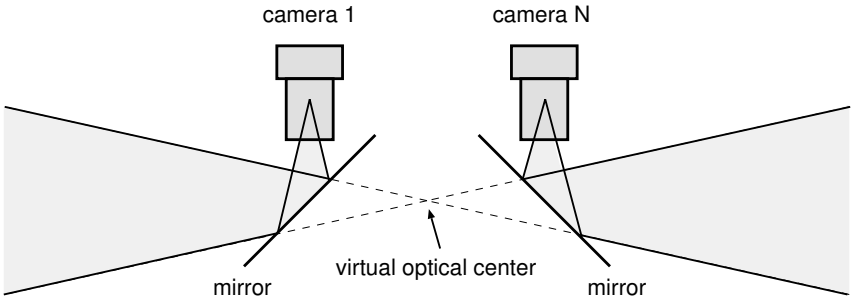


Figure 14.5: *Images that will be combined into a panoramic image must be recorded with a unique optical center. For a static setup with multiple cameras, this is not possible because the cameras would be located at the same place. One solution is to use mirrors to redirect the light direction such that the cameras can be mounted without mechanical problems.*

while the virtual optical center of all cameras is still at a joint position (Fig. 14.5). The advantage of this camera system is a high and uniformly distributed resolution in the panoramic image. For our experiments, we used a panoramic camera composed of five independent cameras.

Since the single cameras show significant lens distortions and are not mounted in perfect geometric alignment, the cylindrical transform cannot be applied directly. Instead, the transformation between input image and cylindrical coordinates was provided by the manufacturer as a regular mesh of calibrated feature-points. Each point in the mesh hereby defines a corresponding position between image coordinates and cylinder coordinates. The transformation for the pixel positions that do not fall exactly on mesh vertices was obtained by bilinear interpolation. Figure 14.6 depicts the mesh for one of the cameras. This example shows that this transformation not only includes the cylinder projection, but also fish-eye lens distortion and a twisted camera mount.

Another issue of the camera setup was that the single cameras only provided interlaced video. Because this would introduce severe distortions at moving objects during the irregular resampling in the dewarping process, it is important to deinterlace the input image prior to synthesizing the panoramic image. We implemented a fast *ad-hoc* deinterlacing algorithm which carries out deinterlacing selectively for the motion areas only.[1]

[1]The algorithm considers small blocks in the image independently. For each block, it computes the sum of luminance differences between adjacent same-parity field lines and the sum between adjacent differing-parity lines. If motion is visible in the block, the differences between lines of differing parity is larger, since the object moved during the

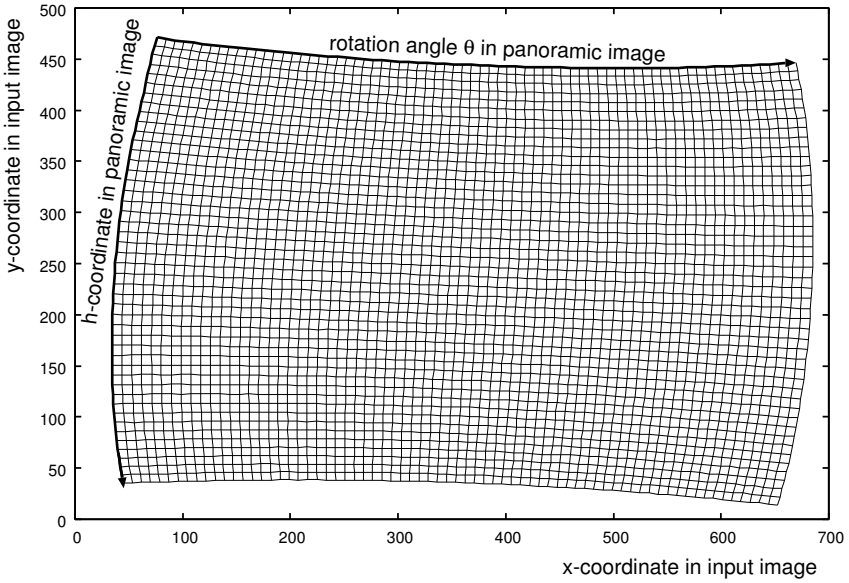


Figure 14.6: *Calibration data for one of the five cameras as it was provided by the manufacturer. This calibration information is the direct transformation from the camera image to the cylindrical panoramic image. Hence, it includes the correction of lens distortions, tilted camera mounting, and the transform to cylinder coordinates.*

Because of the high data rate generated by the five cameras at full NTSC resolution (720×486), we first recorded the video stream of each camera independently. Afterwards, each video stream was deinterlaced, and all five streams were combined into a panoramic image sequence. The resulting panoramic video has a resolution of 3552×480 pixels. Figure 14.7(a) shows an example picture.

14.3 Visualization of panoramic videos

A panoramic image or video is a complete 360-degree view of the environment around the camera. Hence, it is not an ordinary flat image and a variety of visualizations for this special images have been proposed. We briefly introduce the most important in the following, ending with our new

time-difference of the two fields. Hence, if this difference is significantly larger than the difference between lines of the same parity, the block is deinterlaced by duplicating the content of on field.

proposal of a visualization employing a *3-D room reconstruction*.

- **Unwrapped cylinder.** The most common display technique for cylindrical panoramas is to unwrap the cylindrical surface to a flat image (Fig. 14.7(a)). At first glance, this looks like an image with very wide field of view. However, there are two properties that distinguish this image in cylindrical coordinates from a normal, planar image. First, the image shows a complete 360-degree surrounding, such that the viewer looks in all directions around him at the same time. This is an unusual experience, since the normal human view is limited to about 180-200 degrees (160 degrees with one eye) [100]. Second, straight lines are not preserved by the cylindrical projection. Hence, geometrical concepts like parallel lines and vanishing points, which are important for an intuitive understanding of the scene, cannot be applied easily. As a consequence, this mapping is difficult to understand and interpret for humans.

- **Panoramic image browser (PIB).** A visualization technique that preserves straight lines is the generation of virtual views from the position of the capturing camera. Based on the cylindrical panoramic image, the viewer application uses the inverse of Eq. (14.1) to generate rectified, flat views with a limited field of view from the camera position. Since these views cannot cover the complete 360 degrees, the user can interactively turn the displayed view in the left and the right direction[2]. The advantage of this technique is that the generated views look identical to real-world views. Especially, the synthesized images preserve the straight lines from the real world. However, the disadvantage of this technique is that a static visualization (e.g., a printout on paper), of the complete environment is impossible.

- **3-D cylinder projection.** A visualization that combines the interactivity of the previous method with the possibility to have a quick scene overview is to display a 3-D view of the cylinder surface with the panoramic image as texture (Fig. 14.7(b)). This approach can be used in the following two ways. First, the virtual camera can be placed at the center of the cylinder, such that the generated views look similar to the previous PIB approach. The main user interaction at this position is to turn the camera to look into different directions. Second, moving the virtual camera outside of the cylinder gives a static scene-overview by showing the complete cylinder at

[2]This presentation has become popular with Apple's Quicktime VR standard.

(a) Unwrapped cylinder.

(b) 3-D cylinder model.

Figure 14.7: *The unrolled image from the virtual cylinder (a), and a 3-D view onto the virtual cylinder (b).*

once. The combination of these two possibilities makes the visualization very flexible. It is important to note that the global view onto the cylinder gives some indication of the spatial arrangements in the scene, but the intuitive perception of this overview is often misleading. For example, consider that the panoramic image is recorded in a square room. The intuitive assumption is that every wall should cover 90 degrees in the panoramic image. However, this is not true, since the actual angle depends on the camera position. To see this, assume that the camera is placed close to a wall. Clearly, this wall will cover almost 180 degrees in the panoramic image. In fact, the symmetric uniformity of the cylindrical visualization and the absence of distinguished geometric features is often misleading and actually complicating an intuitive scene understanding.

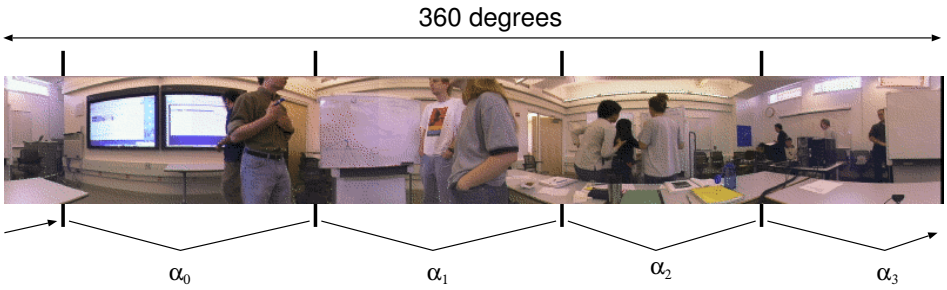
- **3-D room projection.** Although the previously discussed 3-D cylinder projection already combines the advantages of interactive rectified

views and a scene overview, the overview image is often misleading. The reason is that the cylinder surface is a virtual object that is not related to the original scene objects. In man-made environments, especially indoor locations, the space is usually defined by flat walls, which are perpendicular to each other. These walls are important for our orientation, but during the projection onto the 3-D cylindrical surface, these hints for the human perception are lost; the cylinder looks the same from every direction.

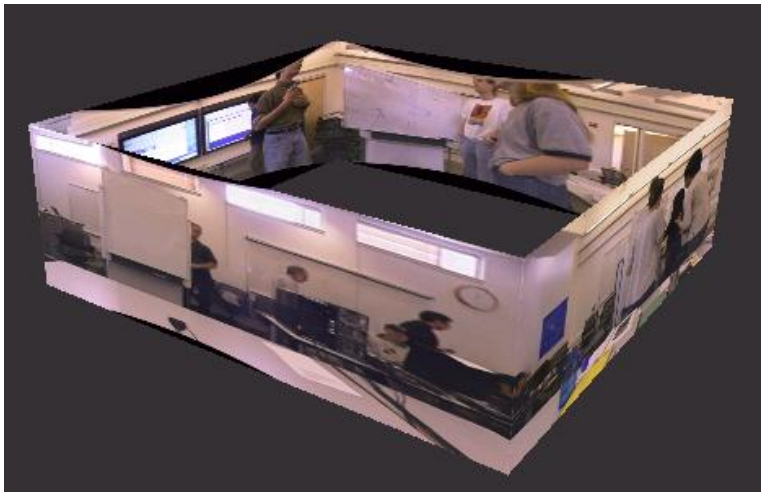
To provide hints about the scene geometry to the viewer, it is important to present the scene overview in a way that mimics the original geometry. In particular, we propose a new visualization technique for the special case of indoor scenes where the room walls provide the main hints for orientation. The difference to the previous approach is that instead of projecting the surrounding onto a cylinder surface, we reconstruct the real room shape and use the panoramic image as wall textures (Fig. 14.8).

From the original camera position, the visualization appears equal to the PIB technique. However, for a distant camera, the scene geometry indicates the layout of the walls and the camera position during recording. Note that the projection onto flat walls also preserves straight lines, which makes the wall textures look realistic. However, the visualization should not be misunderstood as a complete 3-D reconstruction. Changes of depth that are not modeled in the reconstruction can lead to perspective distortions if the camera was recording the area in an acute angle. On the other hand, our visualization technique is much easier to implement and use than a full 3-D reconstruction.

In the following, we describe the 3-D room reconstruction technique for rectangular rooms in more detail. The generalization to arbitrary room shapes or floor plans follows in Section 14.1.3.



(a) Unwrapped cylinder with angle measurements.



(b) View from above.

(c) Texture 1. (d) Texture 2. (e) Texture 3. (f) Texture 4.

Figure 14.8: *A sample reconstruction of a rectangular room.*

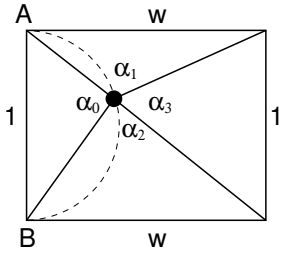


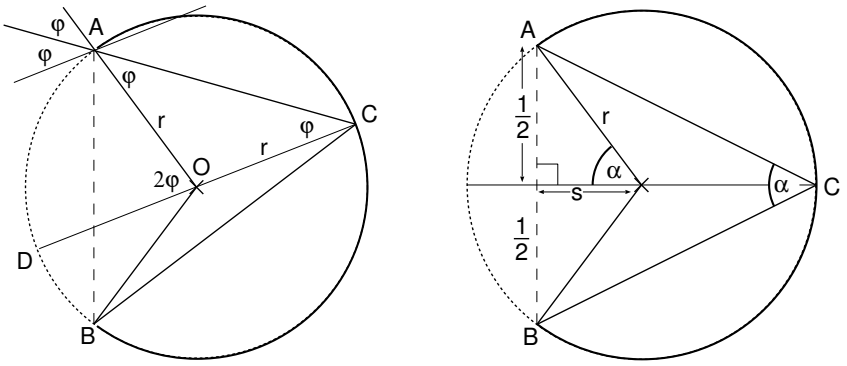
Figure 14.9: *The room geometry should be reconstructed from the measured angles* $\alpha_0, \ldots, \alpha_3$.

14.4 Reconstruction of rectangular rooms

In this section, we consider the problem of determining the wall sizes of a rectangular room from a cylindrical panoramic image captured in this room. Once we know the sizes of the walls and the position of the camera, we can project the panoramic image content onto these virtual walls and create the geometrical model for our visualization. While the wall sizes and camera position could be measured by hand, it is more convenient to obtain these values directly from the recorded image. The idea of our approach is to derive this information from the angles between the room corners, which the user has to mark in the image (Fig. 14.8(a)).

Since the panoramic image is given in cylindrical coordinates, the horizontal distance between two corners in the panoramic image corresponds to the angle between these corners, measured from the camera position (Fig. 14.9). Knowing these four angles (of which only three are independent, since they sum up to 2π), we can determine the ratio of the room dimensions and the camera position. It is not possible to recover the absolute room size, but this is also not required for the visualization, and we can simply set the size of one wall to unity.

The reconstruction is carried out in two steps. First the algorithm makes a preselection of positions that could potentially be the true camera positions. We derive that the valid camera position must be located on a circular connecting two room corners. With this information, we restrict the possible camera positions to a one-dimensional search space. Second, the algorithm carries out a binary search to determine the specific camera position on the circular arc. The search in the second step is guided by the pre-knowledge that the reconstructed room should be rectangular.



(a) Double angle theorem.

(b) Determining the position and radius of the circular arc.

Figure 14.10: *Determining the circular arc of valid camera positions.*

14.4.1 The circular arc of possible camera locations

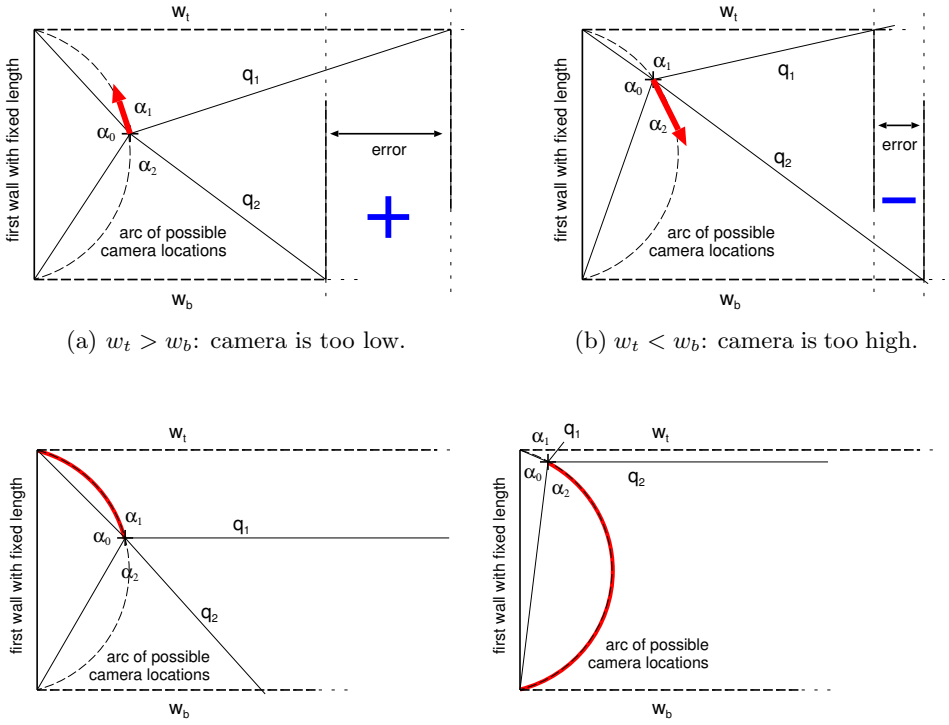
Prior to developing the actual algorithm, let us first examine possible positions of the camera in the reconstructed room. For this, we make use of the following theorem, which we briefly prove here for the convenience of the reader.

Theorem: (Euclid, Elements, Book III, Proposition 20.) Given three points A, B, C on a circular arc ACB with center O. It holds that $\angle AOB = 2\angle ACB$.

Proof: Consider Figure 14.10(a). Since the triangle AOC is an isosceles triangle, $\angle ACO = \angle OAC = \phi$. But then, $\angle AOD = 2\phi$ (consider the reappearing angles at A). The same construction holds for the triangle BOC. Hence, the total angle $\angle AOB = 2\angle ACB$. □

Note that in the preceding lemma, the location of C on the circular arc ACB does not influence $\angle AOB$. Hence, it also holds that the angle at C is independent of its position on the arc. This lets us conclude that for fixed points A,B, all positions of C that have a fixed angle $\angle ACB = \alpha$ lie on a circular arc. To find the center position and radius of this circular arc, let us consider the special case where C is on the perpendicular bisection of AB. Assume that the points A and B have unit distance (Fig. 14.10(b)). Then, we get the radius from $\sin\alpha = 1/(2r)$ and the distance of the center from $\tan\alpha = 1/(2s)$.

Now, we are going to apply this theorem to our estimation problem. Let us normalize the room size such that the left (and right) wall has unit length



(a) $w_t > w_b$: camera is too low.

(b) $w_t < w_b$: camera is too high.

(c) q_1 is horizontal: lower bound for camera position.

(d) q_2 is horizontal: upper bound for camera position.

Figure 14.11: *The camera position is located on the circular arc, but its position is unknown. (a) and (b): A binary search is applied to find the position for which the error $w_t - w_b$ is zero. (c) and (d): the positions for which the rays q_1 or q_2 are horizontal define the initial interval for the binary search.*

and the top (and bottom) wall has length w (Fig 14.9). We denote the four angles under which the four room walls are observed with $\alpha_0, \ldots, \alpha_3$. First, we concentrate on the left wall AB of unit length, which is observed with an angle α_0. Because of the previously derived theorem, we know that the camera position C must lie on the circular arc ACB, and we can compute the position and size of this arc from the angle α_0.

14.4.2 Searching for the camera position

Once we know the circular arc on which the camera is located, the remaining step is to find its position on the arc. To verify a potential camera position, we compute the wall sizes that would result for this position and accept the camera position if the resulting room is rectangular.

We begin the construction with the left wall, which has unit length. Since the assumption is that the room is rectangular, we know that the top and bottom wall must be perpendicular to this left wall. The width of the top and bottom walls are unknown, but their widths should be equal because the wall on the right side is parallel to the left wall.

Let us choose an arbitrary camera position on the arc and consider this position. Then, the corner-to-corner angles α_1 and α_2 define the direction of two rays q_1, q_2 emanating from the camera position in the direction of the room corners (Fig. 14.11). These rays intersect the top and the bottom walls in a distance w_t and w_b from the left wall, respectively. Because we know that the top and bottom wall should have equal length, w_t should equal w_b. However, if we have chosen the wrong camera position on the circular arc, this will not be true.

Notice that if we move the camera upwards along the arc, the top intersection point moves to the left (w_t decreases), while the bottom intersection point moves to the right (w_b increases). To find the camera position for which $w_t = w_b$, we can exploit this behaviour by applying a binary search for the correct camera position. If $w_t > w_b$, the camera position is further to the top, while for $w_t < w_b$, the camera position is lower.

For some camera position, the ray direction of q_1 or q_2 becomes horizontal. For these positions (and the more extreme positions), there is no intersection of the rays with the top or bottom wall. These critical camera positions can be used to determine an initial interval of camera positions for the binary search. Starting the search with this interval not only reduces the number of iterations needed for the binary search, but it also removes the requirement to handle the special case in which the rays q_1, q_2 do not intersect the top or bottom walls.

14.4.3 Creating a virtual room visualization

When the sizes of the room walls and the camera position are known, we can create a virtual 3-D model of the room and generate textures for the room walls. To create the texture maps, we scan the 3-D wall plane with the desired resolution of the texture maps and we compute the respective pixel position in the panoramic image by using the inverse of Eq. (14.1).

This obtained 3-D room model is rendered using an OpenGL-based

viewer application. The scene is built with the estimated camera position as the origin of the 3-D coordinate system. The user can control the rotation of the scene around the x and y axes, as well as the distance d of the camera to the origin. The viewing transform is set up as

$$\mathbf{p}' = \mathbf{K} \left[\mathbf{R_x R_y}\, \mathbf{p} + \begin{pmatrix} 0 \\ 0 \\ d \end{pmatrix} \right], \tag{14.3}$$

where $\mathbf{p}$ denotes the 3-D point position, $\mathbf{R_x}, \mathbf{R_y}$ are the rotation matrices and $\mathbf{K}$ is the perspective projection matrix. This particular sequence of transformations allows for a very intuitive navigation. When the distance of the camera to the origin is decreased, the program avoids d to become negative. This makes it very easy to place the camera at the position of the real camera (move forward until the virtual camera reaches the original camera position). From that position, the user views the scene just as if he would be at the camera position in the real world. Panning with the virtual camera at this special position gives exactly the output as displayed by popular viewers for panoramic images. The second useful viewing position is looking down on the complete room from above the scene, since this gives a quick overview of the general scene layout. An example visualization created with the described algorithm is depicted in Figure 14.8.

14.5 Reconstruction of floor plans

The reconstruction algorithm described in the previous section was limited to rectangular rooms. In this section, we extend this algorithm to enable it to reconstruct the geometry of arbitrary rooms. We keep the principle that corners are manually marked in panoramic images, and that the algorithm derives the camera position and wall sizes from the angles between room corners. The layout of the room walls is also specified by the user. For more complex rooms, it is often impossible to see all walls in only one image because of occlusions. In these cases, the algorithm uses several panoramic images captured from different positions.

14.5.1 Previous work

Several algorithms for 3-D reconstruction have been proposed. They can be coarsely divided into algorithms without pre-knowledge about the scene and algorithms making use of a scene model. Algorithms of the first class are usually very complex to implement [149] and they are probably not robust enough in cases of low-textured surfaces. Algorithms of the second class

employ a complete geometric model of the object or scene and they only adapt the sizes based on the observed images. An algorithm of this second class is described in [37]. Another algorithm [168] considers specifically the reconstruction of room shapes from panoramic images. Compared to our algorithm, it supports more general geometries than a collection of walls, but compared to our proposal, it is more complex to implement and to use.

14.5.2 Reconstruction algorithm concept

Our floor plan reconstruction is based on the same user interaction as in the simpler rectangular room case. The user also marks the position of the room corners in the input image. However, while we previously only considered reconstruction from a single panoramic image, we now allow for an arbitrary number of panoramic images. This is necessary since many room corners can be occluded from some camera positions. The more panoramic images are used, the more information we have available for the reconstruction. The less pre-knowledge about the room geometries is available (non-perpendicular walls, unconnected free-standing walls), the more images are required.

The algorithm starts with an initial room configuration that defines the room layout (position of walls and constraints about perpendicular walls), but that does not yet include the correct wall sizes. For an example, see Fig. 14.13. This figure shows a user-supplied geometric room model, where the outline of the room is specified, but the correct wall sizes are still unknown. The basic principle of the algorithm is to compute the *corner-to-corner* angles from the current model and compare them with the measured angles. A gradient descent search is used to adapt the wall sizes such that the differences between angles in the model and the measured angles are as small as possible.

In the following, we describe the algorithm in four steps.

- **Section 14.5.3**. First, the parameterization of the model is constructed. Parameters are chosen such that hard constraints like perpendicular or parallel walls are enforced by the parameterization itself.

- **Section 14.5.4**. Second, we present the parameter-estimation algorithm. This step adapts the parameters such that the corner-to-corner angles in the model fit to the angles measured in the input images.

- **Section 14.5.5**. The convergence robustness depends on the definition how measured angles and angles in the model are compared. We compare an *inner-angle* definition with an *outer-angle* definition

by examining the error function for local minima or plateaus, which decrease the robustness of the optimization.

- **Section 14.5.6**. Because the optimization is based on a gradient descent approach, a good initialization is required. The evaluation of the error function will show that local minima and plateau region can be avoided if the initialization satisfies some ordering conditions. In this last step, we explain how the initialization is obtained.

14.5.3 Modeling the floor plan geometry

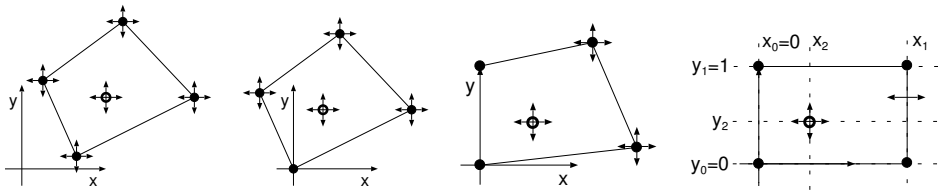
The floor plan reconstruction algorithm uses two types of information for the estimation:

- the angles between room corners measured from their position in the panoramic images, and
- the predefined geometrical layout of the room. This geometrical model includes the relative position of the walls, but not their size. The model also considers pre-knowledge about right angles between walls.

Let us first consider the number of degrees of freedom when estimating the floor plan geometry. A floor plan is parameterized by the 2-D positions of the room corners and the camera positions. The camera positions are required to carry out the texture mapping.

We start with a simple example of a rectangular room and one camera, which is similar to the special case that we considered in the previous section. This configuration gives 4×2 parameters for the room corners plus two parameters for the camera position (Fig. 14.12). However, the absolute placement of the room in our coordinate system is arbitrary and we can fix one corner to a predefined position, like $(0, 0)$. Moreover, we can fix the overall rotation angle of the floor plan, and as absolute size cannot be determined, we can also fix the length of one wall to, e.g., unity. The easiest way to do this is to fix the position of a second corner to, e.g., $(0, 1)$. In total, this reduces the number of degrees of freedoms by four.

The reduction from ten parameters to only six was obtained by eliminating superfluous degrees of freedom in the parameterization. On the other hand, we can add more pre-knowledge about the room geometry. For example, we can assume that the room shape is rectangular. This pre-knowledge can be expressed with three constraints, each forcing one wall to be perpendicular to another wall. These three constraints further reduce the number of free parameters from six to three, thereby making a reconstruction possible.



(a) Free position (10 parameters).

(b) Fixed position (8 parameters).

(c) Fixed rotation, size (6 parameters).

(d) Orthogonal geometry (3 parameters).

Figure 14.12: *By adding geometry constraints to remove unnecessary over-parameterization, we can reduce the number of parameters from 10 in the general case (a) to only 3 for a rectangular room (d). The free parameters are indicated with double arrows.*

For our general floor plan reconstruction, we enforce the constraints for perpendicular walls implicitly through the parameterization. We normalize the rotation of the complete floor plan such that (most) walls will be aligned along the horizontal and vertical coordinate axes. Each wall that is aligned to the coordinate axes can be parameterized with only three parameters. For example, a vertical wall is parameterized by the two corner positions, but both positions share the same x coordinate. For the right wall in Fig. 14.12(d), we would get the corner positions (x_1, y_0) and (x_1, y_1).

Furthermore, we also add the normalization of the floor plan position and size as hard constraints in the parameterization. For this, we select one vertical wall and define one corner postition to $(x_0, y_0) = (0, 0)$ and the other corner position to $(x_0, y_1) = (0, 1)$. Note that using this parameterization for the rectangular-room case, only x_1 for the right wall position and x_2, y_2 for the camera location remain, so that we can compute these three free parameters from the three angle measurements.

A more complex example is depicted in Fig. 14.13. The room shape has eleven walls, but it is parameterized with only six free parameters $x_1, \ldots, x_3$, $y_2, \ldots, y_4$. Additionally, the two camera positions add four parameters x_4, y_5, x_5, y_6. From the image of the left camera, we can obtain nine angle measurements, since two of the walls are at least partly occluded. The right camera can contribute seven angle measurements. In total, we have 16 measurements for 10 parameters and the reconstruction is possible. Note that a reconstruction would also be possible with only the left camera. In this case, we would only have nine measurements, but also only eight parameters, since the position of the right camera is not included. On the

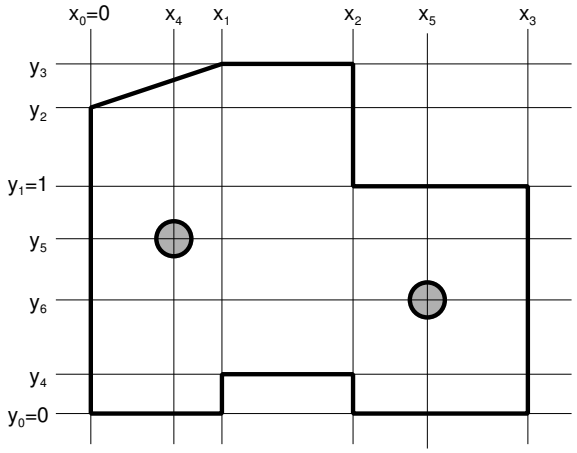


Figure 14.13: *Room corners are specified by coordinates x_i, y_i. Horizontal and vertical walls will reuse the same x_i or y_i coordinate for both corners. This implicitly encodes the pre-knowledge that these walls have to be horizontally or vertically aligned. Camera positions are assigned their own pair of x_i, y_i coordinates.*

other hand, a reconstruction from only the right camera is impossible, since we would have eight parameters to estimate from only seven measurements. Note that a sufficient number of measurements does not generally assure that the reconstruction is possible. This is the case when there are more measurements available than required for some walls and at the same time, too few measurements for other parts. However, in practice, this is rarely the case.

14.5.4 Estimating the floor plan parameters

The central task in the floor plan reconstruction is to estimate the model parameters based on the angle measurements that were taken from the panoramic images. The model parameters consist of the coordinates x_i, y_i of the wall corners and the camera positions. According to the geometric constraints, some of these coordinates can appear in the specification of several positions. All coordinate values that appear in the model are collected in a large parameter vector

$$\mathbf{v} = (x_0 = 0, x_1, x_2, x_3, \ldots,\ y_0 = 0, y_1 = 1, y_2, y_3, \ldots), \tag{14.4}$$

in which three entries are fixed (namely $x_0 = y_0 = 0$, and $y_1 = 1$) to remove the superfluous degrees of freedom. To find the corresponding coordinates

for a position $\mathbf{p}_i$, we use two index sets m_i and n_i into the parameter vector $\mathbf{v}$ to define $\mathbf{p}_i = (x_{m_i}, y_{n_i})^\top$,

From the captured panoramic images, we obtain a set of angle measurements. Each measurement gives an angle $\alpha_{i,j,k}$ between corners $\mathbf{p}_i$ and $\mathbf{p}_j$, seen from camera position $\mathbf{p}_k$. We denote the set of available measurements as $\mathcal{M} = \{(i,j,k)\}$. Furthermore, we can compute angles $\beta_{i,j,k}$ that correspond to the measured angles from the geometric model as

$$\beta_{i,j,k} = \arccos \frac{\mathbf{d}_{ik}^\top \mathbf{d}_{jk}}{||\mathbf{d}_{ik}|| \cdot ||\mathbf{d}_{jk}||}, \tag{14.5}$$

where $\mathbf{d}_i = \mathbf{p}_i - \mathbf{p}_k$ and $\mathbf{d}_j = \mathbf{p}_j - \mathbf{p}_k$ are the vectors from the camera position k to the corners i and j. This equation defines $\beta_{i,j,k}$ as the inner angle between these vectors. Actually, we will shortly replace this definition with a slightly modified one that gives a better convergence behaviour. For an error-free ideal case, all measured angles $\alpha_{i,j,k}$ should equal the angles $\beta_{i,j,k}$, computed from the adapted model. Because of noisy measurements, these angles will not be exactly equal, and we define the total error of the floor plan model as

$$E = \sum_{(i,j,k)\in\mathcal{M}} |\beta_{i,j,k} - \alpha_{i,j,k}|, \tag{14.6}$$

which we minimize with a Quasi-Newton optimization. The convergence of this optimization depends on two factors: the smoothness of the cost function E, and the initialization. We discuss these two topics in the successive two sections.

14.5.5 Improving the convergence behaviour

When considering again the definition of $\beta_{i,j,k}$ from Eq. (14.5), we notice that the definition gives the non-oriented inner angle $\beta_{i,j,k} \in [0;\pi]$ between two vectors. During experimenting with this measure in the optimization process, we occasionally observed that the optimization did not converge. To see why the above angle definition can cause problems in the optimization process, consider the very simple case that there is only one camera which observes a single wall. Imagine that the camera is moved on a line perpendicular to the wall from one side of the wall through the wall to the other side. While approaching the wall, the angle $\beta_{i,j,k}$ increases to π and after crossing the wall, it decreases again. For this case, the cost function E is symmetric to the wall. In the optimization process, this has the disadvantage that there is a minimum of E on each side of the wall (Fig. 14.14).

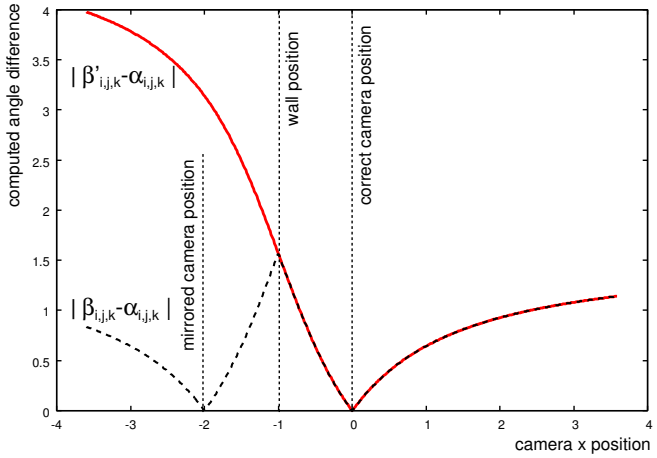


Figure 14.14: *With the non-oriented angle $\beta_{i,j,k}$, it cannot be distinguished on which side of a wall the camera is located. The oriented angle $\beta'_{i,j,k}$ has a single minimum at the correct side.*

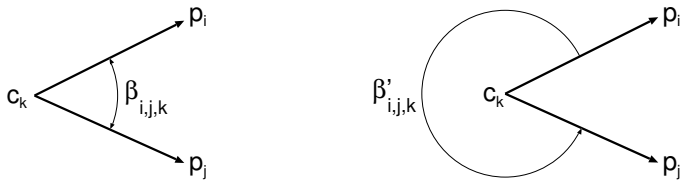
To prevent this effect, we changed the definition of the angle $\beta_{i,j,k}$ to an oriented angle. We define the *oriented angle* $\beta'_{i,j,k}$ as the angle from corner $\mathbf{p}_i$ to corner $\mathbf{p}_j$, measured in counter-clock-wise direction (see Fig. 14.15). The orientation of the two corners is detected by computing the signed area spanned by the two vectors $\mathbf{d}_{ik}$ and $\mathbf{d}_{jk}$ from the camera to the wall corners. The signed area is obtained easily from the determinant of the matrix composed of these two vectors. Thus, we can compute the oriented angles as

$$\beta'_{i,j,k} = \begin{cases} \beta_{i,j,k} & \text{if } \det[\mathbf{d}_{ik} \,|\, \mathbf{d}_{jk}] \leq 0, \\ 2\pi - \beta_{i,j,k} & \text{if } \det[\mathbf{d}_{ik} \,|\, \mathbf{d}_{jk}] > 0. \end{cases} \tag{14.7}$$

In comparison with the previous inner angle definition, the new oriented angle can distinguish between the camera being on the correct side of a wall and being on the backside of the wall. Using this angle definition, we obtain a clear minimum at the correct side of the wall, and the error E increases monotonically with increasing distance from the optimal position (Fig. 14.14).

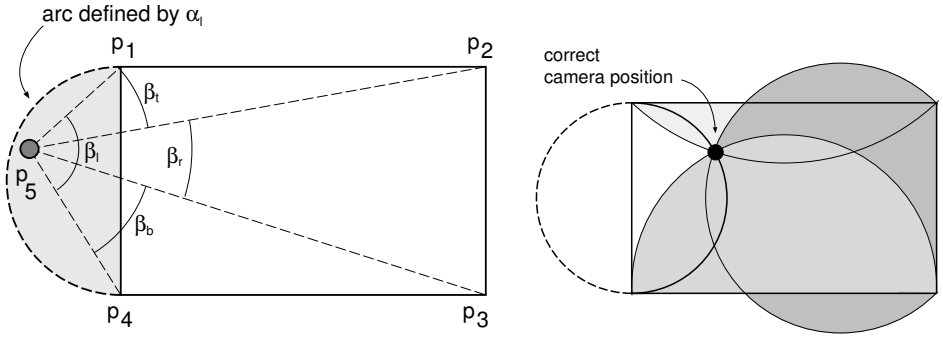
Dependency of the model error on the camera position

Let us now examine a more complex case with a variable camera position in a rectangular room having fixed walls at $x = \pm 1$ and $y = \pm 1$. The error surface of E for the two angle definitions is depicted in Fig. 14.17. In the



(a) $\beta_{i,j,k}$ (b) $\beta'_{i,j,k}$

Figure 14.15: *Definition of angle differences. While $\beta_{i,j,k}$ is the inner angle between the two vectors (a), $\beta'_{i,j,k}$ is defined as the angle from $\mathbf{p}_i$ to $\mathbf{p}_j$ in counterclockwise direction (b).*



(a) (b)

Figure 14.16: *Illustration of angles in a plateau area. (a) Moving the camera within the grey plateau area does not change the total cost E. (b) In each of the grey areas, $\alpha_i < \beta_i$. The symmetric areas, mirrored at the walls are not shown.*

case of the inner angle definition, we notice that there are *plateau* regions around the room walls with constant E. These areas impose difficulties, since the gradient based optimization can get stuck in this area.

To understand why these plateau regions exist, let us concentrate on one of these regions as depicted in Figure 14.16(a). The depicted area corresponds to the area outside of the room, for which $\alpha_l < \beta_l$, which means that the camera in the model is closer to the wall than in reality. For simplicity of notation, we use the notation $\alpha_l = \alpha_{1,4,5}, \beta_l = \beta_{1,4,5}$ as abbreviation for the angles corresponding to the left wall. We use similar abbreviations for the top (t), bottom (b), and right (r) walls. For the considered plateau area at the left wall, $\beta_r < \alpha_r, \beta_t < \alpha_t, \beta_b < \alpha_b$ as

illustrated in Figure 14.16(b). If we compute the total angle error for all four walls as

$$\begin{aligned} E =& |\beta_l - \alpha_l| + |\beta_t - \alpha_t| + |\beta_r - \alpha_r| + |\beta_b - \alpha_b| \\ =& |\beta_l - \alpha_l| + |\beta_t - \alpha_t| + |\beta_r - \alpha_r| + |\beta_l - \beta_t - \beta_r - \alpha_b|, \end{aligned} \tag{14.8}$$

we can resolve the absolute-value operators to derive

$$\begin{aligned} E =& (\beta_l - \alpha_l) - (\beta_t - \alpha_t) - (\beta_r - \alpha_r) - (\beta_l - \beta_t - \beta_r - \alpha_b) \\ =& -\alpha_l + \alpha_t + \alpha_r + \alpha_b, \end{aligned} \tag{14.9}$$

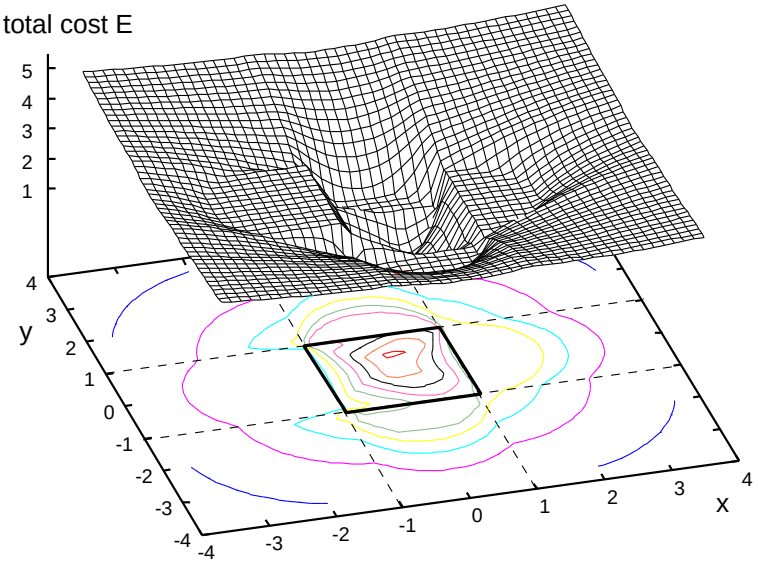
which is a constant. This explains the plateaus in the error function at each wall.

If we consider the same room geometry, but with the oriented angle, we obtain the error surface as depicted in Figure 14.17(b). This error function shows neither plateau areas nor local minima for varying camera positions. Instead, the error surface shows discontinuities whenever the camera crosses a wall plane, because in this moment, the oriented angle jumps between 0 and 2π. However, these steps in the error function impose no problem for the gradient descent search, since the step is always downwards in the direction to the minimum. Consequently, while the optimization can get stuck on the plateau areas using the inner angle definition, convergence is ensured with the oriented angle.

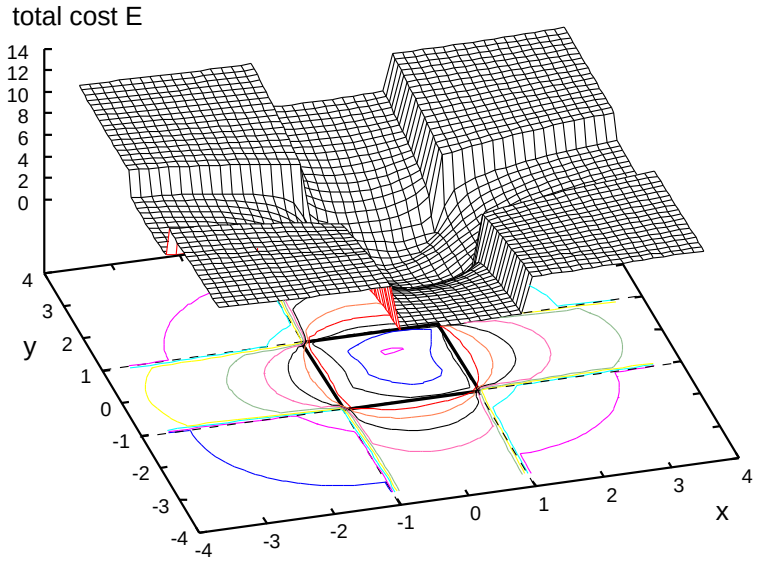
Dependency of the model error on the wall positions

A similar behaviour of the model error can be observed when keeping the camera position constant and varying the wall positions. We examine again the case of a rectangular room, with walls at $x, y = \pm 1$ and the real camera at $(0, 0)$. However, now we consider the position of the right wall as unknown and variable. Figure 14.18 depicts the resulting model error E that is obtained for each angle definition. In Figure 14.18(a), the camera is set to the correct position at $(0, 0)$, while it is set to $(-0.5, 0)$ in Figure 14.18(b). We can observe that the error function for the non-directed angles β show larger plateau regions and even local minima. Similar to the previous example, the directed angle definition β' leads to steps in the error function, but no local minima. Note that the left plateau area for oriented angles starts when the right wall position is moved so far to the left, that it crosses the left wall such that it is actually left of the left wall. For non-oriented angles, the plateau region already starts when the wall crosses the camera position.

We can conclude that the oriented angle β' shows clear advantages over the inner angle β. The oriented angle does not result in plateau regions

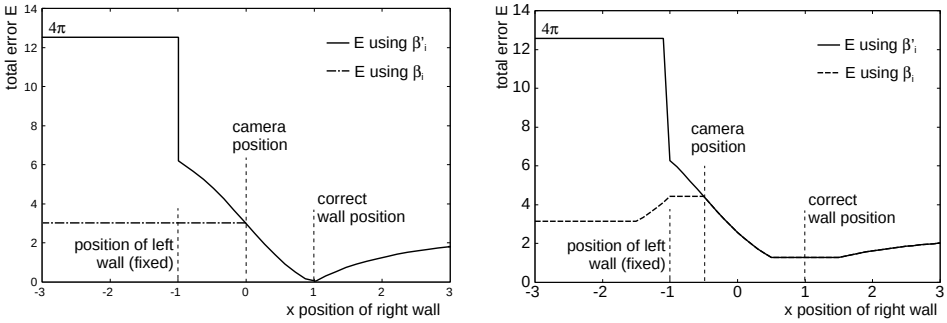


(a) Model error E computed with $\beta_{i,j,k}$.



(b) Model error E computed with $\beta'_{i,j,k}$.

Figure 14.17: *Model error E when moving the camera position while keeping the wall coordinates constant.*



(a) Error function with camera at correct position ($x = 0$).

(b) Error function with camera at incorrect position ($x = -0.5$).

Figure 14.18: *(a) Total model error for a rectangular room, depending on the position of the right wall. (b) If the order of the left and right wall is interchanged, it can lead to a constant error E.*

for varying camera positions and it provides a clear error minimum. Both angle definitions lead to plateaus when the order of room walls is swapped, but for the oriented angles, these regions are smaller.

We conducted the same experiments with defining the error as the sum of squared angle differences. However, this definition leads to an error surface with many local minima, so that we did not pursue this further.

14.5.6 Initialization of the floor plan layout

In the last section, we observed that the error surface of E is smooth and has a unique minimum as long as the cameras are placed within the rooms, and as long as the order of the room walls are not interchanged. Hence, the optimization should be started with a configuration in which these conditions are satisfied to ensure convergence.

We obtain the initial placement of the walls by examining the user-specified floor plan model. Assuming that the walls are oriented along north-south or west-east direction, we can determine the direction going from one wall corner $\mathbf{p}_i$ to the other corner $\mathbf{p}_j$. Diagonal walls are not considered here. Based on this information, we build a west-east ordering of the points such that point $\mathbf{p}_i <_x \mathbf{p}_j$ if corner i is to the west of j. A similar ordering $<_y$ can be defined for the north-south direction. Subsequently, these orderings can be used to assign increasing coordinates. Note that the orderings do not necessarily impose a unique valid ascending enumeration of the coordinates. For example, in Fig. 14.19, the coordinates x_2 and x_3

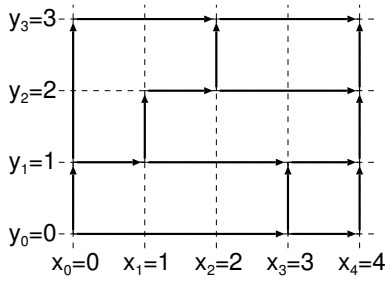


Figure 14.19: *An example initialization of a floor plan, based on the predefined ordering. Note that swapping x_2 and x_3 also gives a valid ordering.*

could also be swapped and still fulfill $<_x$. Any of these admissible orderings provides a good initialization of the wall positions. Finally, we initialize the position of the cameras at the center of all wall corners that are seen in each camera's image.

14.5.7 Obtaining wall textures from the panoramic images

After the optimization process has converged, the position of the walls and the cameras are known, and we can generate texture maps for the walls by projecting the panoramic image content onto the wall planes. Since we may have several views of the same walls, recorded by different cameras, we have to decide from which camera image we extract the image data. The following points have to be considered.

- The wall should be visible. Cameras that are located on the back side of the wall, or which are occluded by other walls cannot be used.
- The larger the distance of the camera to the wall, the lower the texture resolution that is obtained.
- If the camera is too close to the wall, parts of it are viewed in an acute angle. As a result, changes of depth that are not reflected in the floor plan model can lead to perspective distortion artifacts. This applies, e.g., to furniture that is not projected orthogonally onto the wall.
- A wall may not be visible completely in any single camera view. In this case, the texture information has to be collected from several camera images.

The walls are processed independently, where we first determine which cameras are located at the front side of a wall. This information is obtained easily using the oriented angle from Eq. (14.7). If $\beta'_{i,j,k} > \pi$, then the camera $\mathbf{p}_k$ is located at the backside of wall $\mathbf{p}_i, \mathbf{p}_j$. Cameras that are at the backside are excluded from the further processing.

To decide from which camera the wall texture should be taken, we evaluate the expected image quality by determining the deviation of the camera position from an ideal camera position. For room corners $\mathbf{p}_1, \mathbf{p}_2$, we define the ideal camera position as $\mathbf{p}_c = \frac{1}{2}(\mathbf{p}_1 + \mathbf{p}_2) + \frac{1}{2}\mathbf{R}_\perp(\mathbf{p}_2 - \mathbf{p}_1)$, where $\mathbf{R}_\perp$ is a rotation by $\pi/2$. This places the ideal camera position on the perpendicular bisection of the wall at a distance that is half of the wall width. All cameras that are not at the backside are ordered according to the distance of this ideal position. In this ordering, the camera which is closest to the ideal position, comes first. For every column of texture pixels, the ray between the first camera and a pixel in the column is checked for intersection with other walls. If there is an intersection, the second camera is checked for free sight to the pixel, and so on. Note that only one pixel in the column has to be checked since all walls are upright planes.

14.6 Experimental Results

Experiments have been carried out for both reconstruction algorithms presented in this chapter. For the rectangular-room reconstruction, the input images were captured with the panoramic video camera described in Section 14.2.2. These are well calibrated and generate undistorted panoramic images. An example reconstruction result is shown in Fig. 14.8.

Example results for the floor plan reconstruction are shown in Figure 14.21 and Figure 14.22. The input images for the floor plan reconstruction were captured with a digital still camera and combined into a panoramic image later. The focal length of the camera had to be estimated, since the EXIF data did not contain this information. During the stitching process, small inaccuracies in the image alignment were observed, which lead to inaccurate angle measurements. The computation time for the reconstruction was clearly below one second in all of our examples. The time for generating the texture maps depends on the required resolution and the number of walls, and was about one second for our most complex model. We evaluated the accuracy of the reconstruction result by comparing the normalized size of the walls in the reconstruction with their real sizes. The average deviation was about 4%, which is probably mainly due to the inaccurate alignment of the input images. Moreover, for simplicity, we assumed that the walls itself have zero depth, which is obviously wrong

in reality and which also leads to small deviations in room size. Note that these inaccuracies are not obviously visible in the reconstruction, because the wall textures are stretched by this factor. Corners in the texture image always map exactly to corners in the geometric model.

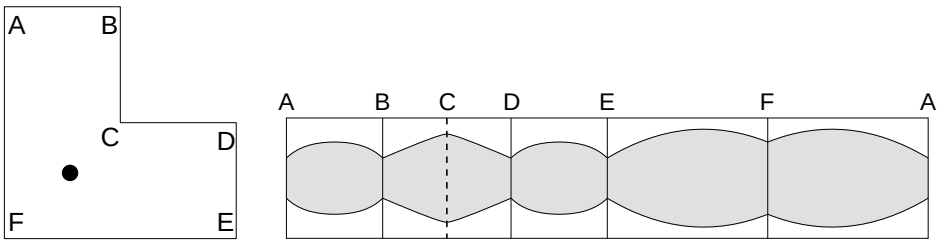
14.7 Conclusions

In this chapter, we have described techniques to capture panoramic images and videos and we have discussed ways for optimal presentation of these panoramic images to the user. We have proposed a visualization specialized for panoramic images recorded in a rectangular room, which reconstructs the room geometry from the panoramic image and presents the panoramic image as the projection onto the room walls. The reconstruction algorithm requires only minor user support and is guaranteed to find the optimum solution. Furthermore, we generalized the concept to the reconstruction of floor plans, comprising an arbitrary number of arbitrarily shaped rooms (preferably but not necessarily with perpendicular walls).

Our conclusion is that the proposed visualization can provide a better understanding of the scene to the user than a flattened panoramic image or a projection onto a cylinder, where the information about the room geometry is lost. Applications of our proposal, especially for the floor plan reconstruction, are also the advertisement of appartments or hotel rooms, for which virtual tours could be made available online. Another application could be the reconstruction of scenes in surveillance systems, in which the objects are extracted from the video and inserted into the 3-D model at their corresponding real-world position. It should be noted that both reconstruction algorithms can be used directly with panoramic video instead of single images, providing video textures on the walls of the 3-D model. Therefore, the geometry model only has to be computed once if the camera positions are kept fixed.

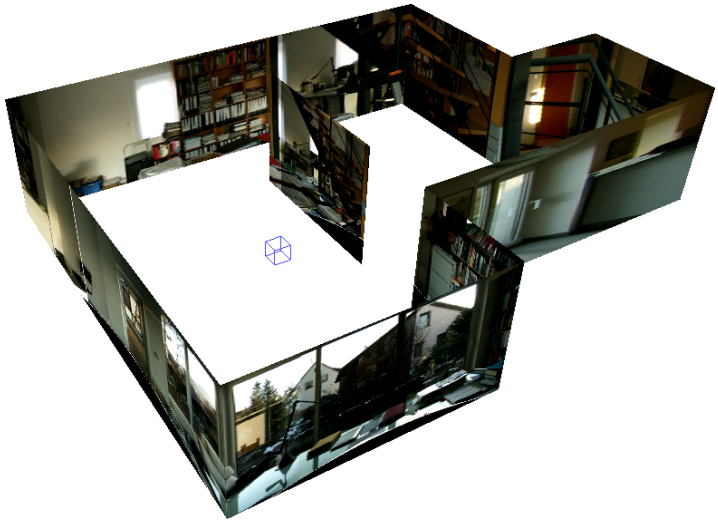
Future research

In future research, the reconstruction could be extended to a completely automatic process. Note that in a cylindrical panoramic image of a room, the vertical lines of the room corners remain straight, while the horizontal lines at ceiling and the ground become bent (see Fig. 14.8(a)). Tracing along the bent horizontal lines, it is easy to find the room corners, because these corners are always located at sudden changes of the line direction. Depending on the angle in which these lines meet in the corner, it is even possible to distinguish between *concave* and *convex* corners, corresponding



(a) Room shape. (b) Corresponding cylindrical panoramic image.

Figure 14.20: *The example room (a) is recorded with a camera located at the black spot. This results in the panoramic image (b). The convex corner C is indicated with a dashed line.*



(a)

Figure 14.21: *Example reconstruction of a single, non-rectangular room from only one panoramic image.*

to an inwards (90 degrees) or outwards (-90 degrees) corner (Fig. 14.20). Furthermore, corners at occluding walls show as discontinuities between the bent horizontal lines. If several panoramic images from the same room are available, corresponding corners could be identified by comparing the wall texture.

Figure 14.22: *Example reconstruction for a complete appartment.*