In 2006 IEEE Intl. Conference on Robotics and Automation (ICRA 2006), Orlando, FL, May 2006

Robot Navigation Using 1D Panoramic Images

Amy Briggs* Yunpeng Li† Daniel Scharstein* Matt Wilder*

* Dept. of Computer Science, Middlebury College, Middlebury, VT 05753, *[briggs,schar]@middlebury.edu*
† Dept. of Computer Science, Cornell University, Ithaca, NY 14853, *yuli@cs.cornell.edu*

***Abstract*— This paper presents a new method for navigation and localization of a mobile robot equipped with an omnidirectional camera. We represent the environment using a collection of *one-dimensional panoramic images* formed by averaging the center scanlines of a cylindrical view. Such 1D images can be stored and processed with few resources, allowing a fairly dense sampling of the environment. Image matching proceeds in real time using dynamic programming on scale-invariant features extracted from each circular view. By analyzing the shape of the matching curve, the relative orientation of pairs of views can be recovered and utilized for navigation. When navigating, the robot continually matches its current view against stored reference views taken from known locations, and determines its location and heading from the properties of the matching results. Experiments show that our method is robust to occlusion, repeating patterns, and lighting variations.**

I. Introduction

Vision-based robot navigation and localization is challenging due to the vast amount of visual information available, requiring extensive storage and processing time. To deal with these challenges, we propose the use of features extracted from *one-dimensional panoramic images* taken during navigation. This work builds on prior work for extracting stable features from the scale space of one-dimensional panoramic images [1] and for the global matching of two views using dynamic programming [2]. In this paper we demonstrate the utility of this approach for robot navigation and localization.

Figure 1 shows the robot equipped with an omnidirectional camera, a sample panoramic view, the 1D circular image formed by averaging the center scanlines, and an epipolar-plane image (EPI) [3], i.e., the evolution of the 1D image over time as the robot travels.

A. Motivation

One-dimensional images can be processed quickly with low storage requirements, enabling dense sampling and real-time analysis of views. The reduced dimensionality also aids greatly in image matching, since fewer parameters need to be estimated. However, there are also some factors that make it difficult to extract stable, globally invariant features from 1D omnidirectional images.

First, for global invariance to viewpoints, the imaged scene must lie in the plane traversed by the camera (the epipolar plane). This requires that the robot travels on a planar surface, which limits the applicability to indoor environments. Even then, extracting a single scanline from an omnidirectional view is problematic since it is difficult to precisely maintain the camera's orientation due to vibrations [4].

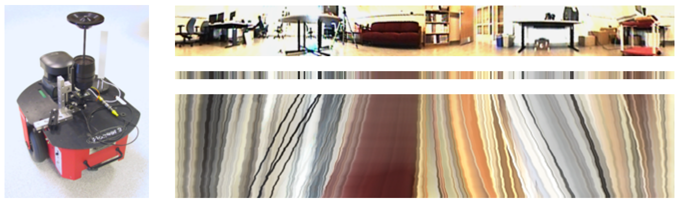


Fig. 1. Our robot with omnidirectional camera, a sample panoramic view, the circular 1D image formed by averaging the center scanlines of the panoramic view, and the epipolar plane image (EPI), a "stack" of one-dimensional images over time as the robot travels.

Instead, we form our 1D images by averaging of the center scanlines of the cylindrical view, typically subtending a vertical viewing angle of about 15 degrees. We thus trade true distance-invariant intensities for robustness. This is not a problem in practice since intensities change smoothly with distance which in turn causes smooth changes in the scale space, from which features are extracted.

A second difficulty of the 1D approach is that one-dimensional images do not carry very much information. Distinct features that can be matched reliably and uniquely over wide ranges of views are rare. A unique descriptor would have to span many pixels, increasing the chance of occlusion. We thus forego global uniqueness of features in favor of a large number of simple features, and use a global matching technique that not only matches individual features, but also considers their spatial relation.

B. Related work

There has been much recent work on invariant features in 2D images, including Lowe's SIFT detector [5], [6], and the invariant interest points by Mikolajczyk and Schmid [7], [8]. Such features have been used for object recognition and image retrieval, as well as robot localization and navigation [9], [10]. A comparison of local image descriptors can be found in [11].

The classic epipolar-plane image (EPI) analysis approach [3] has been applied to panoramic views with explicit image stabilization for 3D reconstruction by Zhu et al. [4].

Ishiguro and Tsuji [12] describe a method for robot localization from memorized omnidirectional views, which are stored using Fourier coefficients; similarly, Pajdla and Hlaváč [13] use the image phase of a panoramic view for robot localization. Cauchois et al. [14] present a method for robot localization by correlating real and synthesized omnidirectional images, but they can only handle small viewpoint changes. Matsumoto et al. [15] present a similar method based on simply comparing cylindrical gray-level images. None of the above methods computes explicit feature correspondences or can tolerate large changes in viewpoints or partial occlusion.

The idea of matching two panoramic images (or, more generally, circular feature sequences) using dynamic programming originates with the work by Zheng and Tsuji [16], who coined the term *circular dynamic programming*. They match vertical line segments across two panoramic views, and do not model unmatched features explicitly, but allow a line in one image to match multiple lines in the other image. Vertical edges in omnidirectional images are also used by Yagi et al. [17].

In contrast to existing work, our method matches two circular sequences of sparse features while explicitly accounting for unmatched features, thus tolerating occlusion. We also contribute a novel way of estimating the position of each epipole (the respective other viewpoint) from the shape of the matching curve.

C. *Organization of the paper*

The remainder of the paper is organized as follows. Section II reviews the detection and matching of scale-space features. Section III discusses the estimation of viewpoint change and relative orientation from the matching curve. Section IV presents our localization and navigation results in real environments, and we conclude in Section V.

II. Scale-space features

We start with a brief review of the feature detection and matching introduced in [1] and [2].

A. *Feature detection*

The key idea is to compute the scale space $S(x, \sigma)$ of each 1D omnidirectional image $I(x)$, $x \in [0, 2\pi]$, over a range of scales σ, and to detect locally scale-invariant interest points or "keypoints" in this space. The scale space is defined as the convolution of the image with a circular Gaussian kernel $G(x, \sigma)$, using a logarithmic scale for σ, so that neighboring values of σ in the discrete representation of S are a constant factor k apart. We typically use $k = 2^{1/3}$, i.e., 3 samples per octave (doubling of σ). Note that we compute the scale space of the luminance (gray-level) image, while color information is still utilized by storing it in each keypoint's descriptor. The second image of Figure 2 shows an example.

Given a discretized scale space, differences are computed both vertically (between neighboring smoothing scales σ), and horizontally (between neighboring image locations x), resulting in the difference scale spaces D_σ and D_x, respectively. This is equivalent to convolving the original image with difference-of-Gaussian (DoG) operators, which approximate first and second derivatives of the scale space. Interest point selection then proceeds by finding the minima and maxima of D_σ and D_x (see the bottom two images in Figure 2). We obtain subpixel estimates of both location x and scale σ by fitting a quadratic surface to the 3×3 neighborhood of each extremum. This also provides estimates of the local curvature.

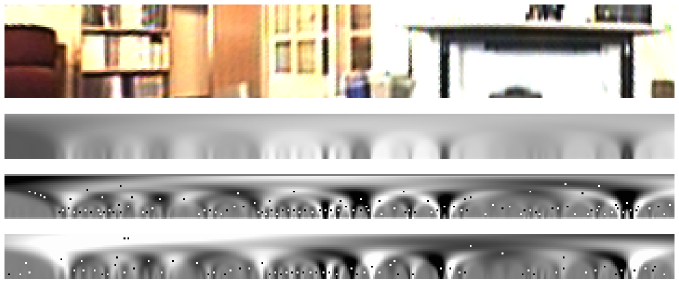


Fig. 2. Feature computation. From top to bottom: part of the circular panorama shown in Figure 1, the gray-level scale space S of the average of all scanlines, differences of rows D_σ, and differences of columns D_x with marked minima and maxima, which are candidate features.

The appeal of finding extrema in scale space is that it provides automatic estimates of both position and scale of features. Intuitively, at each image location, the DoG kernel that best matches the underlying image intensities is selected. Depending on its vertical (σ) position in scale space, an extremum can thus represent an image feature of any size, ranging from small details, such as table legs, to large features, such as a couch or a wall of a room.

B. *Feature matching*

The extrema in both difference scale spaces D_σ and D_x are our candidate features. We exclude clearly unstable features with a small absolute value at the extremum or low curvature around it. Given a typical 1D frame, a circular scanline of 1000 pixels depicting a cluttered indoor scene, we are then left with about 200–400 features. For each feature, we store information about the local shape of the scale space and the original intensities and colors of the corresponding image location in a *feature descriptor*. We define the matching score between two features as the inverse Euclidean distance between their descriptor vectors.

In the absence of narrow occluding objects, the features visible from two different locations will have the same relative ordering. This observation, known as the *ordering constraint*, enables an efficient dynamic programming (DP) algorithm for finding the globally optimal solution to the feature matching problem. This algorithm, described in detail in [2], finds the set of matches that maximizes the total matching score for arbitrary rotations of both views under the ordering constraint while leaving some features unmatched.

Figures 3 and 4 show sample matching results under difficult conditions. Figure 3 demonstrates that our method can handle arbitrary rotations (circular shifts) as well as significant lighting changes. It shows two images taken by a robot from the same location but at different orientations. In addition the ceiling lights were turned off in one of the images. The observed matching curve is very close to the expected curve, a straight line at a 45-degree angle (which wraps around since the frames are circular), despite the fact that only few features could be matched. Figure 4 show an EPI of the robot navigating between bookshelves, as well as the curve resulting from matching the first with one of the last frames. The S-

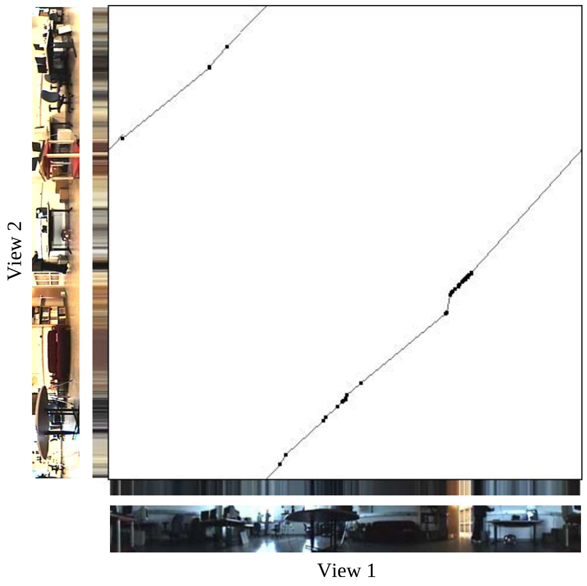


Fig. 3. The matching curve for two omnidirectional images from the same viewpoint under different rotations and in different lighting conditions.

shaped curve, discussed in detail in the next section, indicates a significant change in viewing angle perpendicular to the robot's heading. Despite the large viewpoint change and the locally ambiguous repetitive patterns, our algorithm recovers the correct matches.

The entire matching process is quite fast. Our current implementation on a 3 GHz Pentium 4 takes about 70 ms to match two frames (35 ms for feature extraction and local matching cost computation, and 35 ms for the global matching). Taken together with 25 ms for unwarping the original image, averaging the scanlines, and computing the scale space, the total processing rate is about 10 Hz.

III. Estimation of viewpoint change and rotation

We now describe how to extract the relative orientation and position of two views from the matching curve.

A. *Finding the epipoles*

The characteristic S-shape of the matching curve in Figure 4 results from the fact that features on either side of a translating robot move in opposite directions, while features directly in front and behind the robot remain stationary. The precise location at which there is no visual motion is called the *epipole*. Geometrically, the epipole is the projection (image) of the other viewpoint. Since the images are panoramic, there are two epipole locations, exactly π apart. In terms of visual motion, the front epipole location is the *center of expansion* and the rear location is the *center of contraction*.

Clearly, all possible matching curves have to pass through both epipole locations, while the shape of the rest of the curve depends on the distance to the observed scene points. In general, the shape of the matching curve is constrained to lie in two triangular regions, as shown in Figure 5. The top figure illustrates the case of a robot translating forward in a straight

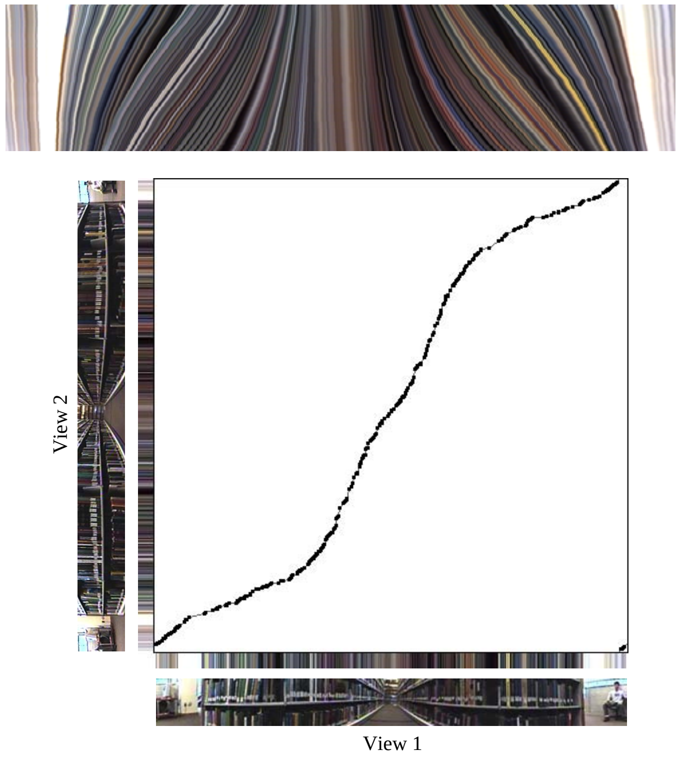
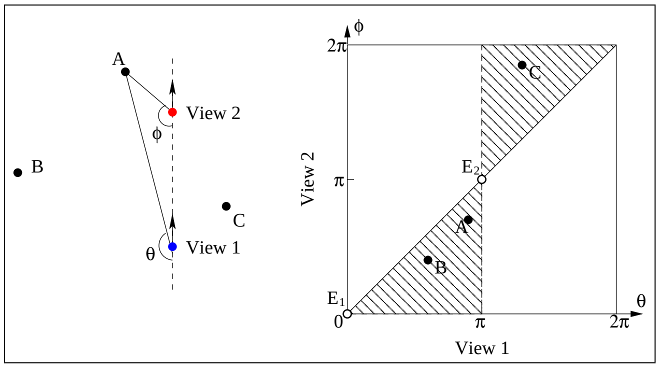


Fig. 4. Top: EPI of an image sequence with repetitive patterns taken by a translating robot. Bottom: The global matching curve between the first frame and one of the last frames.

line, i.e., both views being aligned with the direction of travel. We assume in this paper that image location π corresponds to the front of the robot, and image location 0 to the rear. In this case, the epipole locations E_1 and E_2 are $(0, 0)$ and (π, π). The more general case, in which each view's orientation is independent of the direction of translation, is shown in the bottom of Figure 5. Here, the direction of translation is offset by θ^* in the first view and by ϕ^* in the second view. This results in a shifted matching curve—in a circular fashion since the diagram wraps around—such that the epipoles now lie at (θ^*, ϕ^*) and $(\theta^* + \pi, \phi^* + \pi)$.

The question is now, given an arbitrary matching curve, how can we find the epipoles and recover θ^* and ϕ^*? The answer is that for a correct, S-shaped matching curve there is indeed only one way of placing the "feasible" triangular regions such that the curve is fully contained, and thus only one solution for the epipole locations. If the S-shape is not very pronounced, however, estimating the epipole locations becomes unstable. As the amount of translation goes to zero, the matching curve approaches a straight line and the epipole locations become undefined. In fact, this is the situation of Figure 3. It is easy, however, to determine the relative rotation of the two views $\delta^* = \phi^* - \theta^*$ from the amount of shift of the line, even if the position of the epipoles along the line is uncertain.

Thus, the first step of our algorithm is to estimate δ^*, the intercept of the straight line $\phi = \theta + \delta^*$ through the epipole locations. Let the matching curve be the function $\phi = f(\theta)$. We assume that f has been "unwrapped" in the



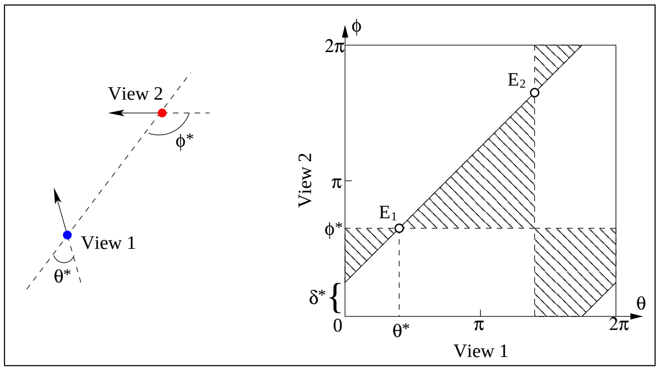


Fig. 5. Top: Two panoramic views whose orientations are aligned with the direction of translation. Each view's orientation (i.e., robot heading) is indicated with an arrow in the picture on the left; this heading corresponds to image location π. For any observed scene point (such as A, B, and C), the point's pair of image locations (θ, ϕ) must lie within the shaded region in the diagram on the right. Bottom: Two views with arbitrary orientations θ^* and ϕ^* with respect to the direction of translation. Their relative rotation $\delta^* = \phi^* - \theta^*$ determines the offset of the diagonal line. In both figures, the epipole locations are marked E_1 and E_2.

ϕ direction so that it is monotone increasing on $[0, 2\pi]$ with range $[f(0), f(0) + 2\pi]$. We define δ^* to be the offset that results in equal amounts of the curve of f above and below the straight line $\phi = \theta + \delta^*$:

$$\delta^* = \arg\min_{\delta} |\#_\theta(f(\theta) > \theta + \delta) - \#_\theta(f(\theta) < \theta + \delta)|, \quad (1)$$

where $\#_\theta$ counts the number of discrete angles (pixel locations) for which its argument is true. The optimal offset δ^* can be found quickly by searching over a small set of discrete angles.

Next, we compute the area A^* between this reference line and the matching curve as an indication of the amount of translation between the two views (see Figure 6 top). Let $g(\theta) = f(\theta) - (\theta + \delta^*)$ be the vertical distance of the matching curve to the reference line. Then

$$A^* = \int_0^{2\pi} |g(\theta)| d\theta. \quad (2)$$

Note that the average absolute vertical distance

$$v^* = \frac{1}{2\pi} A^* \quad (3)$$

measures the *average viewpoint change*, i.e., the average (angular) visual motion between corresponding features. In the absence of odometry information or other knowledge about the absolute size or position of scene features, this is our only way of quantifying the amount of translation between views.

Ideally, at this point, the matching curve would intersect the reference line at two locations, θ^* and $\theta^* + \pi$, yielding the epipoles E_1 and E_2. In practice, however, the matching curve is affected by noise and occlusion, and the intersections may not be exactly π apart. It is also possible that the curve crosses the reference line more than twice. In fact, the (extreme) case of matching two unrelated views would yield an arbitrary matching curve. We thus need not only a robust way of estimating the epipoles, but also a measure for the confidence of the result.

Given a candidate location $\hat{\theta}$ for the first epipole, recall that we expect the first half of the matching curve to be below the reference line, and the second half above it (the shaded areas in Figure 5). That is, we expect $g(\theta - \hat{\theta})$ to be negative on $[0, \pi]$, and positive on $[\pi, 2\pi]$. Using the sign function

$$h(\theta) = \begin{cases} -1, & 0 \leq \theta < \pi \\ 1, & \pi \leq \theta < 2\pi \end{cases} \quad (4)$$

we define the *signed* area between the matching curve and the reference line for a candidate offset $\hat{\theta}$ as

$$S(\hat{\theta}) = \int_0^{2\pi} h(\theta) g(\theta - \hat{\theta}) d\theta \quad (5)$$

We find the epipole location by maximizing this area:

$$\theta^* = \arg\max_{\hat{\theta}} S(\hat{\theta}), \quad (6)$$

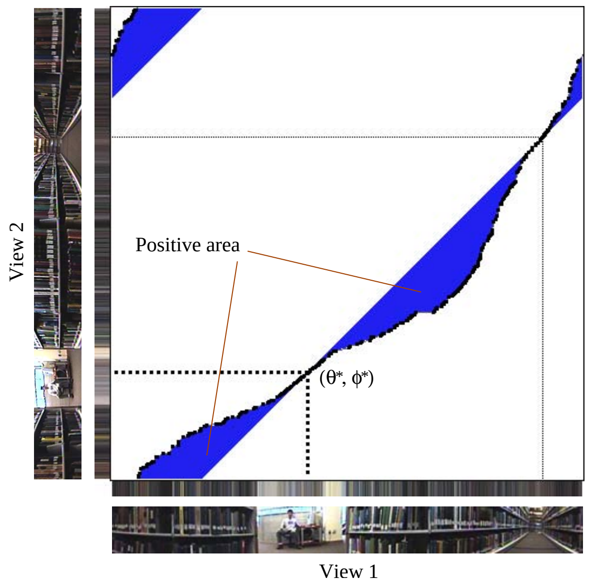
and denote the maximal signed area $S^* = S(\theta^*)$. Again, the optimum can be found quickly by searching over the discrete values of $\hat{\theta}$.

B. Match confidence

Figure 6 shows two matching curves. The top curve results from matching two frames under a significant viewpoint change, but without occlusion. In such a case, the matching curve fully obeys the S-shape rule, ensuring that an optimal θ^* value can been found such that the curve only enters the positive areas. In this case the signed area S^* is equal to the absolute area A^*. In the presence of occlusion, or when attempting to match unrelated scenes, the curve shape becomes more irregular. Even for the optimal value of θ^*, there are regions in which the area is negatively weighted. This is illustrated in the bottom of Figure 6, where we attempt to match two frames from entirely different viewpoints. Thus, negatively weighted areas indicate a "matching error", or lack of confidence in the current match (and consequently, in the estimates of the epipole locations). Since the area that is weighted negatively is proportional to the difference between absolute and signed areas, we define the *error measure* as

$$e^* = \frac{1}{2\pi}(A^* - S^*). \quad (7)$$

This measure indicates the reliability of matching: the greater e^*, the more likely the matching is erroneous. In practice, e^* is typically less than 1 degree for correct matches.



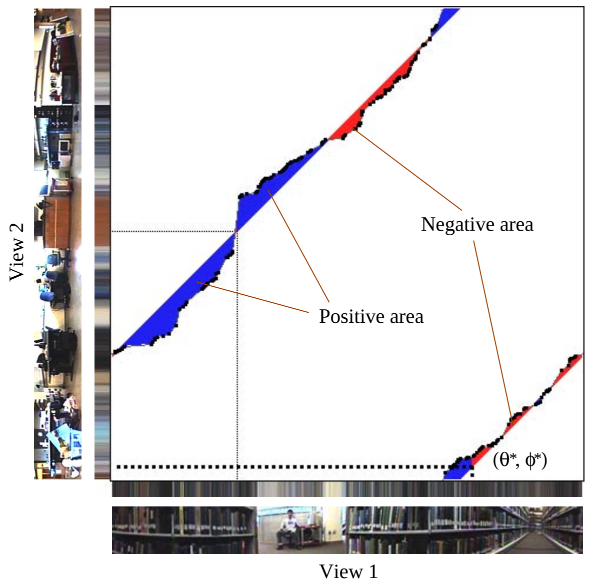
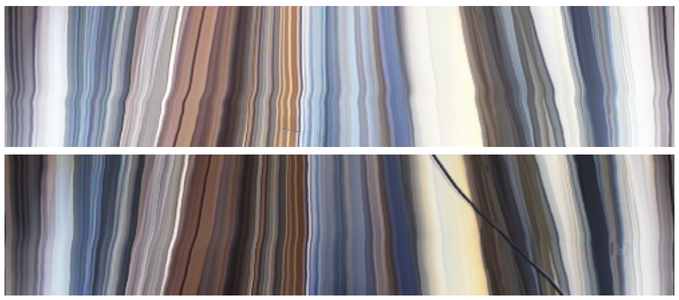
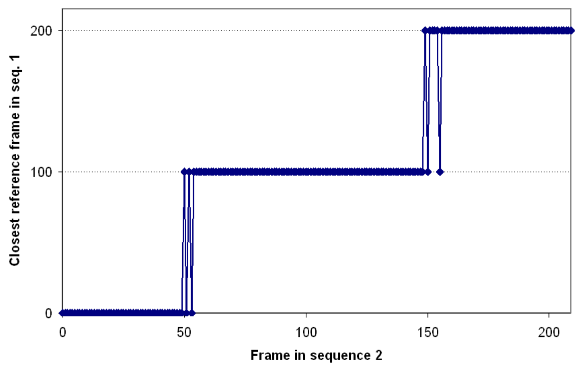


Fig. 6. Top: A good matching curve, which has only positive-signed area (blue/dark). Bottom: A bad matching between two unrelated views with significant amount of negative-signed area (red/light).





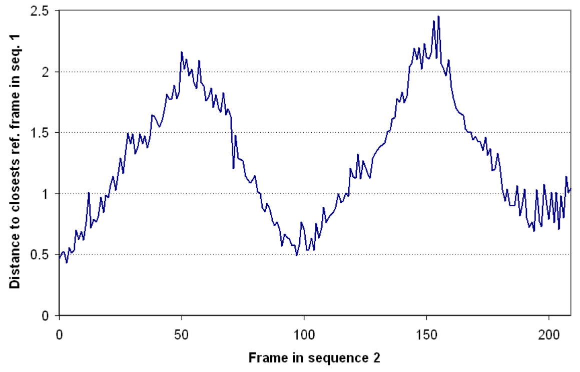


Fig. 7. Top: Two EPIs of sequences taken along similar paths with different lighting and occlusion. Every 100-th frame of the first sequence was stored as a reference frame. Middle: The index of the closest reference frame for each frame of the second sequence. Bottom: The viewpoint change v^* to the closest reference frame, i.e., the average absolute change in viewing angle in degrees.

C. *Evaluating the viewpoint change*

We can experimentally evaluate the accuracy of our measure v^* for viewpoint change as follows: Given an image sequence taken by the robot, we choose a small set of reference frames, for example every 100-th or 200-th frame. We then select for each frame of a different sequence the closest reference frame based on the computed viewpoint change v^* between the two frames. Figure 7 shows the results for two sequences taken along similar paths, but under different lighting conditions and occlusions. It can be seen that the viewpoint change between frames and reference frames correlates with the actual distance, and that the selected closest reference frame only briefly oscillates for views halfway between reference frames. We have performed such experiments for other pairs of sequences and for sparser sets of reference frames, with similar results. Overall we have found that the measure v^* is robust in the presence of lighting changes and some occlusion.

IV. Localization and Navigation

In this section we demonstrate the feasibility of reliable navigation from 1D omni-directional views in indoor environments.

A. *World model*

We represent the environment using a collection of 1D panoramas associated with locations in an absolute coordinate system. For each such reference view, we store the 1D image, the 1D set of features, the position and orientation in the world coordinate system, and a timestamp of when the view was recorded.

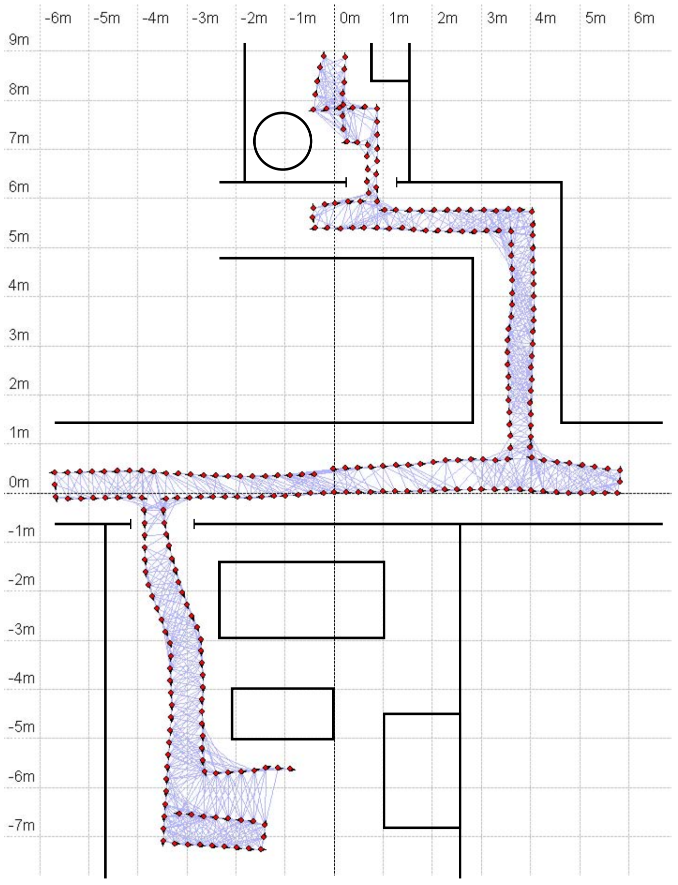


Fig. 8. A sample world model. The small (red) circles represent the locations and orientations of the reference views. The thin (blue) visibility lines indicate which pairs of reference views can be matched and are used for path planning. The black lines illustrate the environment features such as walls, table, and doorways, but are not part of the robot's model.

When building a world model, we manually guide the robot through the environment and take views that are spaced by about 25cm to achieve a fairly dense sampling of the surroundings. We derive the global coordinates from the initial odometry values, followed by a global correction to achieve loop closing and alignment with known locations, necessary to compensate for odometry errors.

Figure 8 shows the layout of one of our world models, acquired from a single run. In practice we store reference views from multiple runs performed at different times of day to get a sampling of different lighting conditions. Note that the architectural features (walls, doorways, tables) are only sketched for illustration, but are not part of the world model.

In our current system, each reference location requires approximately 60kB of memory (uncompressed). About 80% is used to store the features and their descriptors, the rest for the panoramic view. Because of the relatively low cost per location, we are able to store many reference images, and can also afford to store multiple lighting conditions. The robot could, however, still navigate using sparser models. Also, the global accuracy of the locations is not critical since the robot localizes itself with respect to nearby images.

B. Localization

The key component of our system is the accurate localization of the robot from a small set of nearby reference views. During navigation, the robot's approximate location is known, and we can match the closest reference frames first, until a large enough set of reference views has been found whose matching error is below a threshold. If the robot is placed at an unknown location, however, we must match all reference views until enough good matches have been found.

While the viewpoint change v^* of the matched reference views can be used for a first location estimate, the most precise location and orientation estimate of the robot can be obtained from the epipole angles. Formally, localization involves estimating the robot's position (x_R, y_R) and orientation θ_R from a set of good matches M (i.e., matches for which e^* is below a certain threshold). For each reference view $i \in M$, we denote its known absolute location and orientation (x_i, y_i, ϕ_i), and its epipolar angles derived from the matching curve with the robot's current view as θ_i^* and ϕ_i^*.

We first estimate the robot's heading θ_R. For each $i \in M$ we get a separate estimate

$$\theta_{R_i} = \phi_i - \theta_i^* + \phi_i^*. \tag{8}$$

We combine these estimates using a weighted average, where the weights reflect the match quality.

The next step is to find the robot's position. Given an estimate for its location (x_R, y_R) and the known position (x_i, y_i) of reference view i, the line connecting the two has orientation

$$\gamma_i = \arctan \frac{y_i - y_R}{x_i - x_R}. \tag{9}$$

From the match of these two views we also obtain a measured value for this angle:

$$\gamma_i^* = \phi_i - \phi_i^*. \tag{10}$$

Our goal is thus to find the robot position that minimizes the error function

$$O(x, y) = \sum_{i \in M} \|\gamma_i^* - \gamma_i\|^2. \tag{11}$$

See Figure 9 for illustration. We solve this non-linear minimization problem using the Levenberg-Marquardt algorithm.

We have also implemented a different localization algorithm that computes a weighted average of the intersection points of the lines associated with all pairs of γ^* values. The two algorithms yield comparable results in practice.

C. Navigation

Given a goal location, the robot first plans a path from its current location to the goal, then travels it while orienting itself using reference views. For simplicity, our current path planner uses the graph of "visibility" edges between reference views, i.e., edges between reference views that can be matched. This graph is constructed once and stored. Notice that the localization scheme described above permits navigation in areas that are not part of this graph.

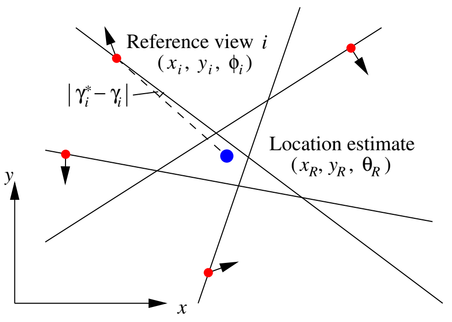


Fig. 9. Localization based on direction constraints. Solid lines represent possible robot locations based on the epipole angles derived from a match with each reference view i. The optimal robot location minimizes the difference between measured angles γ_i^* and measured angles γ_i.

For each navigation task, the robot's estimated position and the goal location are added to the visibility graph, and the shortest path is computed using Dijkstra's algorithm. This path is approximated with a small set of straight-line segments, which form the robot's target trajectory. The robot then navigates along this trajectory while localizing itself in real time using nearby reference frames. In each localization cycle, the robot tries to match up to 12 reference frames with the goal of achieving 5 good matches, from which the location is estimated. We have found that this achieves a good balance between accuracy and speed. Each cycle takes typically less than 1 second (on the robot's 500MHz processor), allowing the robot to travel at an average speed of about 20 cm/s.

D. Discussion

Our experiments demonstrate that the robot can successfully navigate various indoor environments in the presence of moderate scene and illumination changes, some occlusion, and repeating patterns. Large amounts of occlusion or uneven surfaces can hinder the matching. Strong lighting changes (e.g., day vs. night), however, can be handled by storing multiple reference views, whose timestamps allow the robot to select which views to match first. Featureless areas such as long corridors can be ambiguous globally, but usually contain enough information for local navigation when the robot has an estimate of its position.

V. Conclusion

We have presented a new method for navigation and localization of a mobile robot using one-dimensional panoramic images. Our method employs scale-space features and uses circular dynamic programming for image matching. The reduced dimensionality allows dense sampling of reference views and real-time analysis of views. We have presented a new method for computing the relative orientation and position of two views from the omnidirectional matching curve. Accurate robot localization is achieved by combining the angle and distance estimates from a small set of nearby reference views. Experimental results demonstrate that mobile robot navigation from 1D panoramas is feasible in indoor environments in the presence of lighting changes, repeating patterns, and some occlusion.

In future work we will explore ways to reduce the number of stored reference views by analyzing the redundancy of the data. The goal is to achieve a sparser sampling of both viewpoints and lighting conditions without affecting the navigation reliability.

Acknowledgments

The authors are grateful to Carrick Detweiler and Peter Mullen for their contributions to our feature matching system. Support for this work was provided in part by the National Science Foundation under grant IIS-0118892 and by Middlebury College.

References

[1] A. Briggs, C. Detweiler, P. Mullen, and D. Scharstein, "Scale-space features in 1D omnidirectional images," in *Omnivis 2004, the Fifth Workshop on Omnidirectional Vision*, May 2004, pp. 115–126.
[2] A. Briggs, Y. Li, and D. Scharstein, "Matching features across 1D panoramas," in *Omnivis 2005, the Sixth Workshop on Omnidirectional Vision*, Oct. 2005.
[3] R. C. Bolles and H. H. Baker, "Epipolar-plane image analysis: A technique for analyzing motion sequences," AI Center, SRI International, Tech. Rep. 377, Feb. 1986.
[4] Z. Zhu, G. Xu, and X. Lin, "Panoramic EPI generation and analysis of video from a moving platform with vibration," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Fort Collins*, June 1999, pp. 531–537.
[5] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the International Conference on Computer Vision, Corfu, Greece*, Sept. 1999, pp. 1150–1157.
[6] ——, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.
[7] K. Mikolajczyk and C. Schmid, "Indexing based on scale invariant interest points," in *Proceedings of the International Conference on Computer Vision, Vancouver, Canada*, July 2001, pp. 525–531.
[8] ——, "An affine invariant interest point detector," in *Proceedings of the European Conference on Computer Vision, Copenhagen*, vol. 4, May 2002, pp. 700–714.
[9] S. Se, D. Lowe, and J. Little, "Global localization using distinctive visual features," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems EPFL, Lausanne, Switzerland*, Oct. 2002, pp. 226–231.
[10] ——, "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks," *International Journal of Robotics Research*, pp. 735–758, Aug. 2002.
[11] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Madison*, vol. II, June 2003, pp. 257–263.
[12] H. Ishiguro and S. Tsuji, "Image-based memory of environment," in *Proceedings of the IEEE International Conference on Intelligent Robotics and Systems*, 1996, pp. 634–639.
[13] T. Pajdla and V. Hlavac, "Zero phase representation of panoramic images for image based localization," in *Proceedings of the Eighth International Conference on Computer Analysis of Images and Patterns, Ljubljana, Slovenia, Springer LNCS 1689*, Sept. 1999, pp. 550–557.
[14] C. Cauchois, E. Brassart, L. Delahoche, and A. Clerentin, "3D localization with conical vision," in *Omnivis 2003, the Fourth Workshop on Omnidirectional Vision*, June 2003.
[15] Y. Matsumoto, K. Ikeda, M. Inaba, and H. Inoue, "Visual navigation using omnidirectional view sequence," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'99)*, 1999, pp. 317–322.
[16] J. Y. Zheng and S. Tsuji, "Panoramic representation for route recognition by a mobile robot," *International Journal of Computer Vision*, vol. 9, no. 1, pp. 55–76, 1992.
[17] Y. Yagi, Y. Nishizawa, and M. Yachida, "Map-based navigation for a mobile robot with omnidirectional image sensor COPIS," *IEEE Transactions on Robotics and Automation*, vol. 11, no. 5, pp. 634–648, 1995.