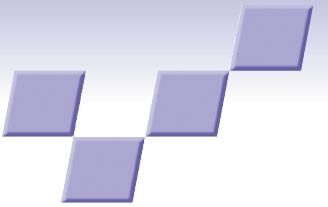


Positioning and Orientation in Indoor Environments Using Camera Phones

Harlan Hile and Gaetano Borriello ■ *University of Washington*

Location technologies provide a way to associate information to a user's location. The authors developed a system that processes a cell-phone camera image and matches detected landmarks from the image to a building. The system calculates camera location and dynamically overlays information directly on the cell phone image.

As people walk through a building, they want help navigating to the right room or accessing directory information related to their current location. Both tasks would benefit from an environment that could directly provide personalized information. Current approaches to this problem use location systems to help the user index into the information. However, the retrieved information needs to be put in context. We've developed a system that uses a camera phone to provide a contextual framework. In our system, the act of pointing the camera forms the query into information about the building. The phone's display then becomes the interface for overlaying information directly onto the image and thus providing the user information in context. The crucial element of this approach is in determining the precise camera pose—that is, its location in 3D and its orientation, in total a full six degrees of freedom.

We're working on two applications of this capability. The first is motivated by an ongoing project to help individuals with cognitive impairments navigate in indoor spaces. This user population often has difficulty navigating complex buildings such as medical centers and shopping malls. In essence, we're trying to provide customized "painted lines on the floor" for users to follow to their destination. Our goal is to overlay directional arrows and navigation instructions onto the image of the user's environment and make it easier to understand directions.[1] The second application targets a more general population that might be interested in information related to a building such as what events are taking place, which resources are reserved and by whom, and when someone was last in his or her office. As people walk down hallways, they can see customized dynamic nameplates that provide this data. Figure 1 shows information-overlay examples for both of these uses.

We've introduced our system approach in earlier work.[2] In this article, we focus on a hallway system prototype and a feasibility study that analyzes our current algorithms' performance in this environment. We begin with a system overview, then describe the components of the prototype—namely, feature detection, feature matching, and image augmentation.

System overview

We base our approach to finding the camera pose on a simple concept: determining "landmarks" in the cell phone image and matching them to previously cached landmarks in the environmental space. By matching enough landmarks, we can precisely compute the camera pose and thereby accurately overlay information onto the display. The different types of landmarks in different kinds of spaces complicate this simple idea.

The landmarks that are available and useful in hallways aren't the same types of landmarks that will work in open areas, such as large rooms. For example, corners, floor-to-wall transitions, and doors are likely to be clearly visible in hallways, but not in cluttered rooms. These features do have a high

 Published by the IEEE Computer Society 0272-1716/08/$25.00 © 2008 IEEE

degree of homogeneity, which means an individual feature isn't uniquely identifiable. We can, however, use the pattern of image features to determine location by comparing them to known feature locations. Our previous work describes our approach for both hallways and open spaces,[2] but in this article we only discuss hallways because the step to match nonunique features requires extra analysis.

Our image-processing system locates these image microlandmarks and compares them to the building's floor plan. We use a building server to hold this floor plan data as well as provide the computation cycles for extracting the microlandmarks and performing the matching. The client and server communicate through a Wi-Fi connection, which many newer cell phones support. We prune the search by using a Wi-Fi location system to coarsely locate the user. We only expect a location estimate that's accurate to within 5 to 10 meters, and several of today's Wi-Fi-based positioning systems can easily achieve this.[3] Our system will support both indoor and outdoor navigation; in outdoor environments, we could use the GPS-based positioning that's available on many newer cell phones.

Figure 2 summarizes how the current system works (see the sidebar on p. 34 for related work). Inputs to the system are a floor plan with relevant features marked on it, a rough location estimate (which includes floor information), and an image from the cell-phone camera. The system sends the image to the server along with Wi-Fi signal-strength information and the type of information requested. In a five-step process, the server

- extracts the relevant features in the image,
- uses the location estimate to select the search area in the floor plan,
- finds the correspondence between the image features and the building's floor plan,
- computes the camera pose from this correspondence, and
- returns an information overlay for display on the phone client.

The method in open areas is similar but uses textural landmarks and matches to previously captured images rather than a floor plan.[2]

Our challenge is to ensure performance on all this computation and communication that's fast enough to support reasonable user interaction speeds. We currently do all the processing on the building's server, although future work includes exploring different task partitions and evaluating their performance.

Figure 1. Information overlays on a camera-phone display image. The image shows both an overlaid navigation aid (the directional arrow) and a "magic lens"-type window for displaying dynamic information about the current surroundings.

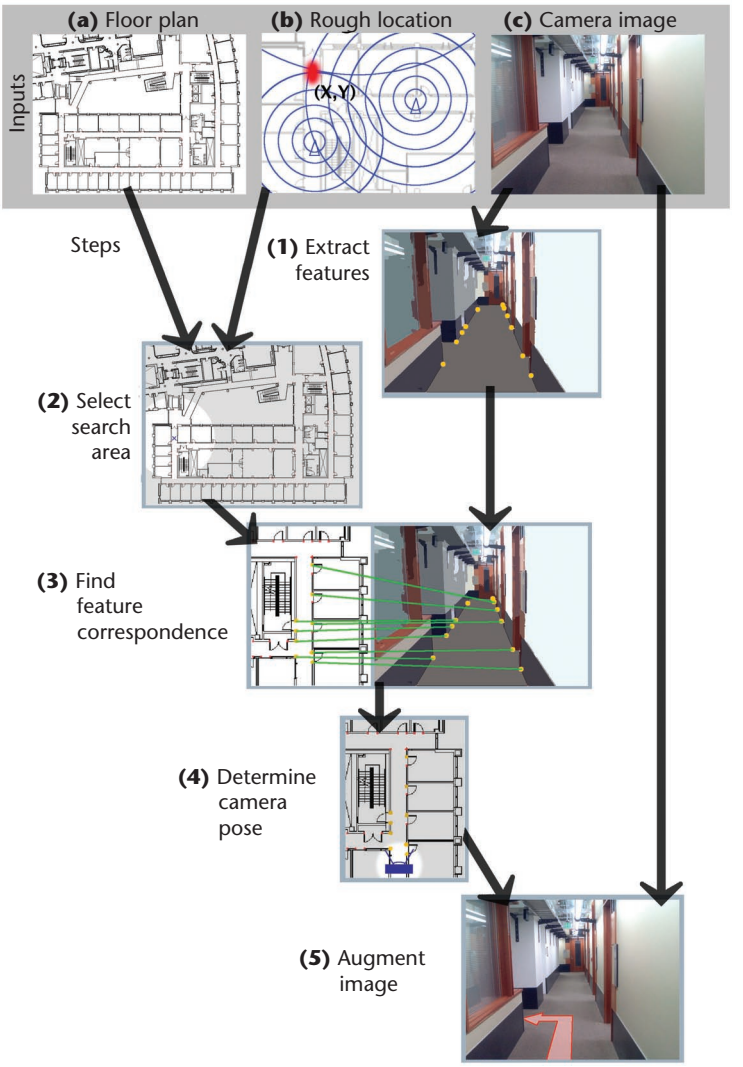


Figure 2. System diagram for calculating camera pose and overlaying information on a camera-phone image. Three input sources generate the augmented image in a five-step process.

Related Work in Image-Based Location Systems

Although we believe our system is novel, various systems can localize a device in its environment through an image. Many of these are designed for robot navigation; they use odometry data and rely on many images to localize accurately. Early work shows that matching building features to refine location is feasible,[1] but it requires a highly accurate estimate of current position and orientation.

More recent work uses robust image features such as those produced by the Scale Invariant Feature Transform (SIFT).[2] The VSLAM system simultaneously builds a map and localizes image features within that map.[3] Other robotic navigation systems use many images along a path and visibility information to calculate likely locations.[4] None of these systems is suitable for our desired scenario, where odometry data and multiple images are not available.

We aren't comparing our system to location systems that don't use image data, such as the signal strength methods from Microsoft, Intel, and others, because these don't provide the accuracy required to do information overlay.

Image analysis systems that aren't intended for localization also provide useful components. The Photo Tourism system can solve for 3D locations of feature points in a scene and calculate camera positions from a large group of images.[5] The system matches new images to the current model, producing an accurate camera pose in relation to the scene. It can transfer annotations between images by using the underlying 3D structure. Photo Tourism relies on distinct SIFT features, which most hallways lack, and it isn't designed for quickly finding a camera position in a large area given a location estimate. For this reason, we might leverage the Photo Tourism system in open areas, but it's not usable for hallways.

Other systems recognize landmarks in outdoor environments and provide annotations.[6] These also rely on SIFT features and don't actually generate a refined location; instead, they merely identify what objects might be in the image. Furthermore, providing a database of geocoded features costs much more than providing a building floor plan. Although these systems support similar interactions, they aren't suitable for use on hallway images.

Augmented reality (AR) systems share a similar goal for information overlay. These systems tend to be object centric. They often tag objects with special markers to facilitate their location in an image.[7] Other systems actively project structured light on the environment to aid in localization,[8] but they would have difficulty in a hallway environment. Existing AR systems provide a variety of information-overlay examples and might be a source for applications of our system, but they don't currently support hallway environments without special tagging or special hardware.

References

1. A. Kosaka and J. Pan, "Purdue Experiments in Model-Based Vision for Hallway Navigation," *Proc. IEEE/RSJ Int'l Conf. Intelligent Robots and Systems* (IROS 95), IEEE Press, 1995, pp. 87–96.
2. D.G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *Int'l J. Computer Vision*, vol. 60, no. 2, 2004, pp. 91–110.
3. N. Karlsson et al., "The VSLAM Algorithm for Robust Localization and Mapping," *Proc. 2005 IEEE Int'l Conf. Robotics and Automation*, IEEE Press, 2005, pp. 24–29.
4. J. Wolf, W. Burgard, and H. Burkhardt, "Robust Vision-Based Localization for Mobile Robots Using an Image Retrieval System Based on Invariant Features," *Proc. IEEE Int'l Conf. Robotics and Automation*, IEEE Press, 2002, pp. 359–365.
5. N. Snavely, S.M. Seitz, and R. Szeliski, "Photo Tourism: Exploring Photo Collections in 3D," *ACM Trans. Graphics*, vol. 25, no. 3, 2006, pp. 835–846.
6. G. Fritz, C. Seifert, and L. Paletta, "A Mobile Vision System for Urban Detection with Informative Local Descriptors," *Proc. 4th IEEE Int'l Conf Computer Vision Systems* (ICVS 06), IEEE CS Press, 2006, pp. 30 (abstract only); doi: http://doi.ieeecomputersociety.org/10.1109/ICVS.2006.5.
7. H. Hile, J. Kim, and G. Borriello, "Microbiology Tray and Pipette Tracking as a Proactive Tangible User Interface," *Pervasive Computing*, LNCS 3001, Springer, 2004, pp. 323–339.
8. M. Kohler et al., "Tracksense: Infrastructure Free Precise Indoor Positioning Using Projected Patterns," *Pervasive Computing*, LNCS 4480, Springer, 2007, pp. 334–350.

Feature detection in hallways

Locating the features in the camera-phone image is the first step to finding how it matches to a floor plan. Although the information in a standard floor plan is limited, it should include locations for doorways and wall corners. These microlandmarks will also be visible in the image. They are located where the vertical lines of doorways and corners meet the edge of the floor.

We implemented a basic feature-detection method based on image segmentation. Instead of looking for the floor's edges directly, we locate the entire floor region and then trace its edges. We chose the *mean-shift* method for segmenting the image.[4] It's difficult to take a picture of a hallway where the floor isn't at the bottom center of the image, so we assume the segment in this position is the floor segment. If necessary, this requirement could be included in the interface that prompts users to take pictures. The system traces the edge of this segment and identifies the corners using a "cornerity" metric.[5] This approach won't locate all the places along the floor

where there's a doorway, but it finds many of them. We can locate more of them by intersecting vertical lines found in the image with the floor boundary, but we don't do that in the examples here.

This tracing method also finds some false corners that don't correspond to anything in the floor plan. False corners at the top of a vertical segment of the floor edge can be discarded as points likely caused by occlusion of the floor.

Although this basic feature detection gives results suitable for demonstrating the system, we're looking at other methods to improve speed and robustness. Image segmentation relies on the floor being a consistent color that's also different from the walls and doors. A feature detector trained for a specific environment could overcome some of these limitations and expand the environments where the approach will work. Figure 3a shows the results of segmentation and corner finding in the hallway prototype implementation. These points are now ready to match to the floor plan.

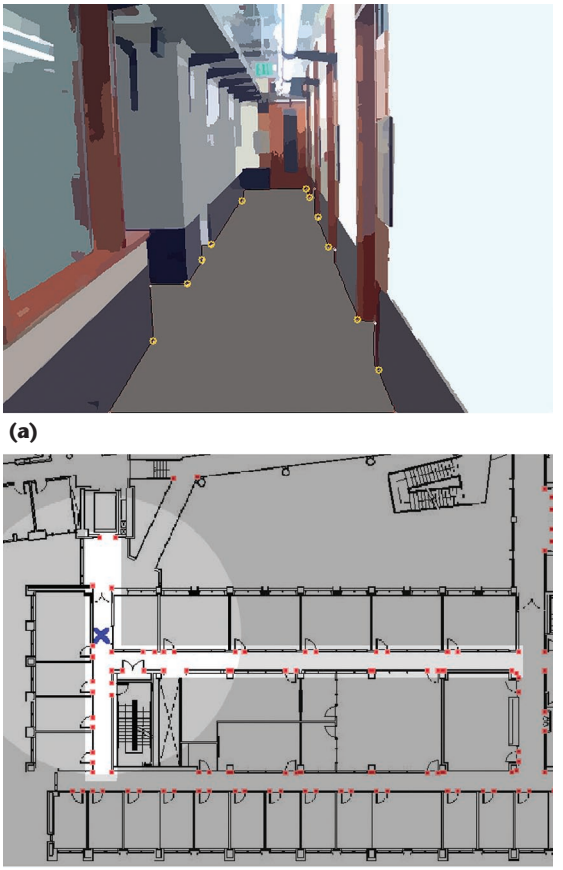


Figure 3. Segmentation and corner-finding results. (a) Our system first segments the hallway image, then traces the floor's edge and locates the corners. Corners marked with orange circles are candidates for matching to floor plan features. (b) The X marks the cell phone's estimated location on the floor plan, and the red dots mark the feature points.

Feature matching

Once the system finds the image features, it must match them to the floor plan features. Our building database is currently an image of the floor plan with the corner and doorway features clearly labeled.

We must first choose a potential set of points in the floor plan to match. This step removes ambiguous cases and reduces the search space. The location system provides a rough location estimate that becomes the center for the region to be tested; the feature-matching system can estimate a search radius on the basis of the location system's accuracy and some idea of the camera's useful visibility range. Figure 3b shows the floor plan, an estimate of the camera's location, and the features to consider for matching to the hallway image. Because the number of points to consider affects the algorithm's speed, instead of including all points within a radius, we approximate a visibility calculation by including only points from hallways that are near the location estimate. We could include other information when calculating the region to consider, such as true visibility calculations, but we don't yet include these.

Once the system defines the two sets of features, it must determine a correspondence. The number of possible ways to match makes this a challenging problem. Four sets of correspondences can define a transformation between the image and map spaces. The example in Figure 3 has 10 image points and 32 map points, resulting in more than 4 billion possible four-point correspondences. To make this problem tractable, we perform this matching using a Random Sample Consensus (Ransac) approach[6] that intelligently selects and prioritizes the hypotheses.

Our method uses minimal structural assumptions to generate the hypotheses for the Ransac algorithm. Lines are fit to the image-space features, which should produce one line corresponding to the hallway's left side and one corresponding to the right side. The system randomly chooses two floor plan lines containing features (for most cases, the region will have only two lines) and a direction relative to the lines and estimated camera position. Now the system can randomly choose two points from each line in a way that orders the points along the lines consistently. The hypotheses for the Ransac algorithm are also prioritized according to the area covered by the four points in the image: the larger the distance between the points, the more likely it will produce a stable camera pose. It uses the area of the four points' bounding box as a measure of the spread and slowly lowers a threshold of the minimum area to consider. It terminates early if the estimate hasn't improved in a "long time" (for example, 5,000 samples) because it's unlikely to improve further as the threshold lowers. Our examples obtain good results testing in fewer than 10,000 hypotheses.

The system estimates the camera pose from these four-point hypotheses using a technique for planar points.[7] The hypotheses are then evaluated

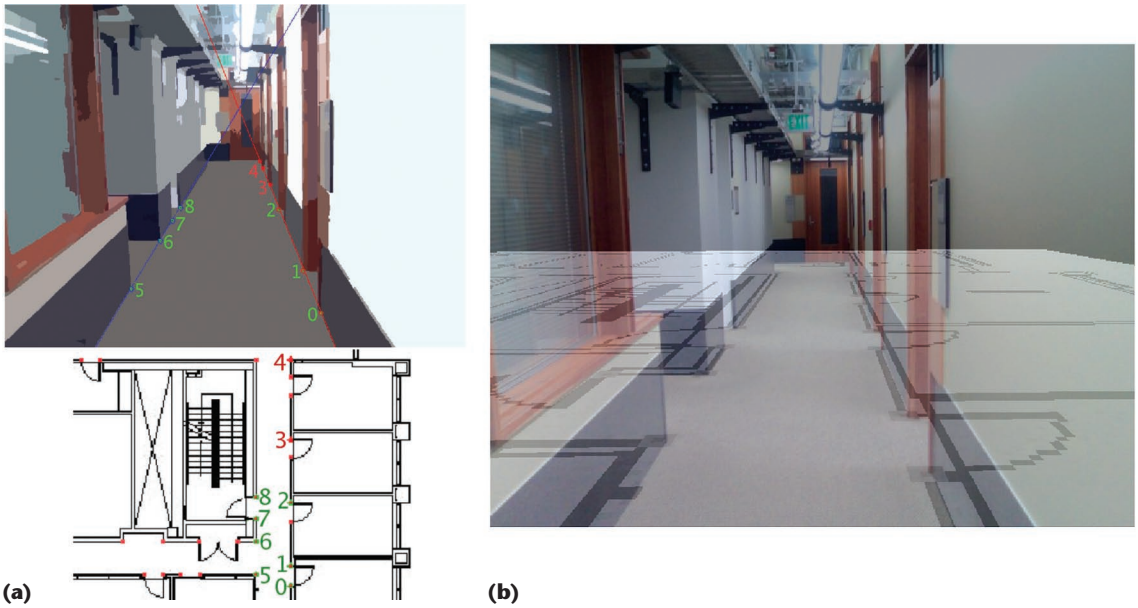


Figure 4. Results of the point-correspondence algorithm in the Computer Science Building. (a) Numbers label the detected points on the image and the corresponding matched points on the floor plan. (b) The floor plan is warped into the image space and overlaid on the original image. This example has 10 image points and 32 map points and completes matching in about 4 seconds.

by using the camera pose to transform the image-space points to the floor plan space and looking at the sum of the squared distance between the two point sets. Weights for points closer to the camera (lower in the image) are higher because they're likely to be detected more accurately in the image. Additionally, unlikely camera parameters (such as a skewed image plane or a large distance from the estimated location) can be penalized to improve the solution ranking. The system uses the highest ranking solution at the search's end.

Environments with a high degree of identical structure can cause problems such as off-by-one errors—for example, mistaking one door for the next. To avoid such problems, the location estimate's accuracy must be greater than the spacing between repetitive structures. Figure 4 shows results from the University of Washington's Computer Science Building. Figure 5 shows results from our Health Sciences Center, an environment that presented more challenges because of the lower contrast, reduced brightness, and reflective floors in the image. Despite these complications, our system detected enough features to produce an accurate match. We nevertheless need to improve feature detection to deal with such challenging environments more robustly.

The Ransac method returns results in 2 to 6 seconds on a 2.8 GHz desktop machine. It's the slowest part of our current pipeline. We believe further improvements are possible that will increase both speed and robustness. However, our system already approaches speeds near our goal to support interactivity.

Augmenting images

While the system finds the correspondence between the image floor plan microlandmarks, it simultaneously solves for both the camera location and its orientation. Applications can leverage the increased location accuracy and camera orientation when determining what information to display. They can then use the mapping between the image and floor plan to overlay information on the camera image. Figure 1, for example, gave navigation directions by drawing an arrow on the floor plan and warping it into the image space. Arrow tips can even disappear behind corners to give an added sense of depth by clipping to the area segmented as the floor, without requiring additional knowledge of the 3D structure.

Because the image also includes knowledge of door locations, applications can mark doors with additional labels (also shown in Figure 1).

Feasibility analysis

Our feasibility study focused on determining latency and accuracy, given a reasonable input image. We believe this analysis demonstrates our system's usefulness and provides motivation and direction for further improvements. The system can currently complete an entire cycle—from taking a picture on the phone to displaying an augmented image—in approximately 10 seconds, using a standard 2.8 GHz computer and processing in Java. About 1 second of this time is spent on image transfer over Wi-Fi. The system processing takes the other 9 seconds. The mean-shift

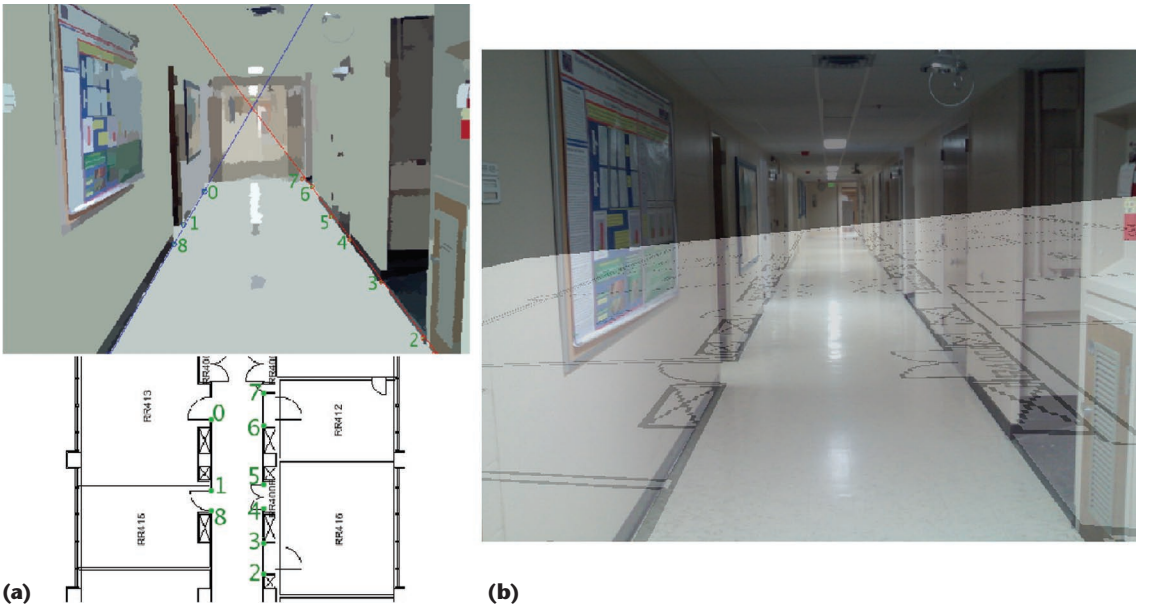


Figure 5. Results of the point-correspondence algorithm in the Health Sciences Center. (a) Numbers label the detected points on the image and the corresponding points on the floor plan. (b) The floor plan is warped into the image space and overlaid on the original image. This example has 9 image points and 19 map points and completes matching in about 2 seconds.

segmentation takes 1.5 seconds, and the edge and corner locations take 1 second. The features correspondence takes 2 to 6 seconds, and the image augmentation before sending it back to the phone for display takes 0.5 second. Our investigation of different performance aspects focuses mainly on the correspondence matching because it requires the majority of the processing.

Speed and accuracy

To evaluate the system's speed and accuracy, we measured the localization accuracy while the Ransac algorithm was running, then averaged these numbers over 200 trials per example image. Figure 6 shows the results from our current implementation. They show variation among examples in different locations but, on average, errors drop to under 30 cm in only a few seconds, with little improvement afterward. The application's stopping condition doesn't use a strict time cutoff, so this result shows that the system, on average, reaches a highly accurate solution quite quickly.

The 30 cm accuracy is far greater than what's available from current systems based on Wi-Fi or GPS. Additionally, we've found that even when the distance error is greater, the calculated alignment between floor plan and image is often visually acceptable for producing information overlays. While these results are promising, we would like to explore other matching systems that might make it easier to include constraints or prior knowledge or might have a better guarantee on convergence.

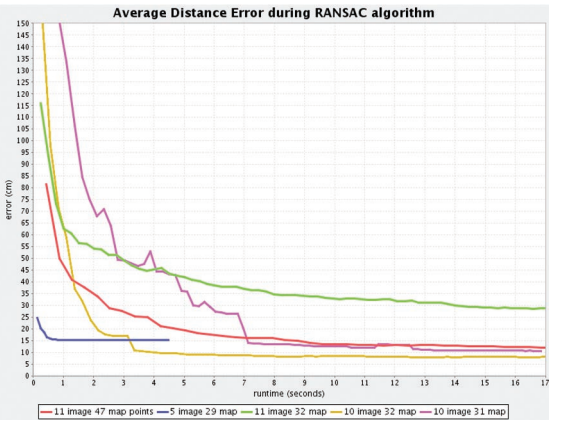
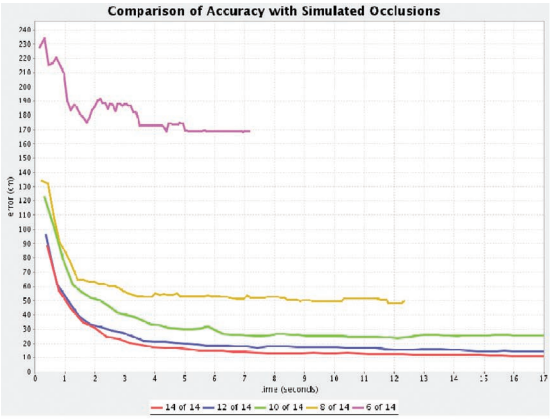


Figure 6. Average localization error as the Ransac algorithm progresses. The examples are from several areas of a building with different numbers of points detected in the image and different numbers of points in the map region of interest.

Feature occlusion

Many environments include clutter or lighting conditions that make it impossible to detect all the microlandmarks in a region. A localization method should degrade gracefully as the amount of data decreases. Figure 7a (next page) shows this to be the case for our method in the five examples that appear in Figure 6. (Other examples produced similar graphs.) Even with almost half of the possible points excluded, it can still get reasonable results. Thus, even if the feature detector can't detect all points because of occlusions or other difficulties, localization can still perform quite well. However, this performance does depend on which points are missing. For example, if only points from one side of the hallway are detected, our system can't produce a result from a set of colinear points. The histogram in Figure 7b shows that the number of trials with no solution increases as more points are occluded.



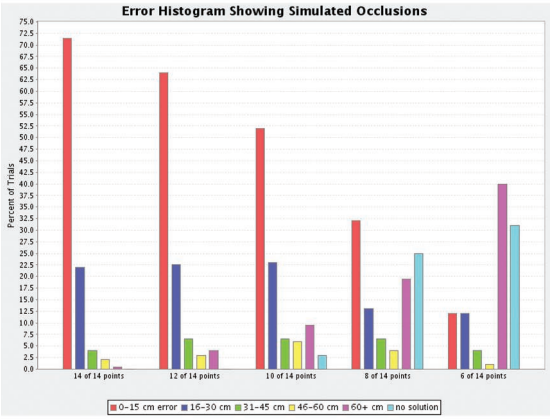


Figure 7. Analysis of error and speed as points are randomly dropped from 200 trials of an example to simulate occlusion. (a) Error histogram showing the distribution of errors for each example at 5 seconds of runtime. (b) Error histogram showing simulated occlusions.

Although this experiment doesn't reproduce true situations, exploring how the number of available points influences performance is useful. Our example images confirm these results where the feature detector doesn't locate all possible microlandmarks but the distribution is sufficient to allow localization. It might be possible to achieve reasonable results with fewer points detected if we also use line constraints or constrain the camera pose by other means—for example, using other sensors or image analysis.

In addition to missing features, our feature-detection algorithm might also detect false features. For this reason, our Ransac algorithm doesn't require all the detected features to match to floor plan features. Although we haven't produced quantitative measures, various examples have shown that the algorithm handles these distracter points well as long as approximately 80 percent of the points detected are true features. A more intelligent feature detector could also help this situation by classifying detected features' quality.

System extensions

Many opportunities exist for improving and expanding our image-based localization system. As expected, the simple methods proposed here don't work for all situations. Figure 8 illustrates three of many cases that cause problems for our current feature-extraction method. The biggest problem area is in segmenting the floor. If the floor contains a large-scale pattern, or if the walls and floor are a similar color, our simple method will fail. We're therefore investigating feature detectors that are tailored to the building, or potentially even different portions of the building. In addition to providing better feature detection in more environments, these specialized detectors would also provide more information, such as feature type (wall corner, left side of door, and so on) that would reduce the number of possible matches and speed up the algorithm. Our current feature detection also takes a significant portion of the runtime, and a more targeted system might run faster.

To cover a wider range of environments, we can use a mixture of floor-plan-based and image-based features. Although our motivating example of navigation instructions isn't as useful within a room, other applications could use this feature. Floor plans have few features that are visible in an average room, but objects in the rooms provide features that are more likely distinguishable with SIFT and Photo Tourism (described in the sidebar on page 34). Our prior work explains how to adapt our system to these open types of environments.[2]

It's important to consider response time because it strongly affects user experience. Although our current system is nearly fast enough, there are further improvements to consider. If all the processing is done on a server and not on the phone, the application must consider the time to send the image across a network. We assume a Wi-Fi enabled phone for the location system and use 640 × 480-pixel images, but other scenarios might take significantly more network time. It might also be possible to perform some or all of the processing on the phone as the processing power of phones continues to increase. This would reduce the amount of data going across the network and the time needed for transfer.

Video-based "magic lens" or AR applications would also require optimization. Even if the initial camera pose takes some time to compute, if we assume relatively small motion between frames, we should be able to compute a camera pose update in a fraction of the time required for the general case. Because this approach uses data in the image itself to calculate where to display information on the image, it should avoid the disconcerting lag or misalignment in systems that use separate sensors to determine where the camera is aimed.

Such a system would make developing and using AR applications for indoor environments much

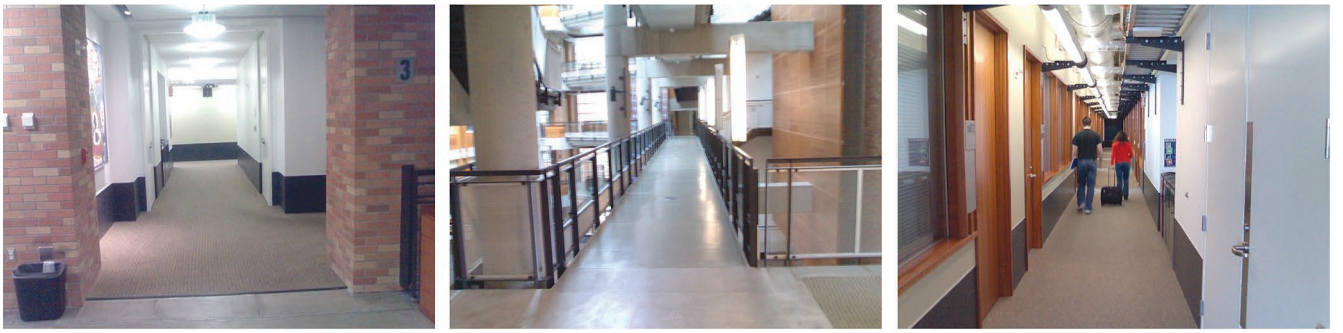


Figure 8. Examples of images that pose problems for simple segmentation. Changes in floor material, reflective floors, and unusual structures such as catwalks break simple assumptions, but they should all be handled properly. Additionally, people or clutter in the hallways will cause problems detecting features, as will environments with low contrast difference between floors and walls.

easier. Environments wouldn't need to be instrumented with additional features. At a minimum, the system could deploy with just a floor plan, although additional information might improve robustness. Users would only require a camera phone to take advantage of the applications. We believe the navigation application is a compelling example that will work well with this system, so it will be our first application. However, the system could support many other application types. For example, an application could supply timely building information, such as nearby conference room availability, or it could link with a calendaring system. More detailed building plans might allow the display of electrical or plumbing lines in x-ray vision style to aid service workers. Extending the system design to outdoor environments will enable applications that can leverage the GPS infrastructure, such as tour guide systems or an easy index into information about the current surroundings.

Our study results demonstrate the possibility of overlaying information on camera phones for indoor environments. Our approach enables localization from images in existing environments with standard hardware. As demonstrated, we can obtain highly accurate results in a short time in a system that degrades gracefully when less data is available. The current system's limitations in handling the full range of environments means we must improve the feature detection and matching while still maintaining low latency. However, our prototype achieves the low latency required for our target application, and we are planning to construct a simple, low-cost navigation assistant to provide a low-cognitive-load interface on a user's standard camera phone.

Acknowledgments

The National Institute on Disability and Rehabilitation Research supports this work under the University of Washington's Access project. Thanks to Noah Snavely, Steve Seitz, Linda Shapiro, and Alan Liu for their assistance on this project.

References

1. A.L. Liu et al., "Indoor Wayfinding: Developing a Functional Interface for Individuals with Cognitive Impairments," *Proc. 8th Int'l ACM SIGACCESS Conf. Computers and Accessibility* (Assets 06), ACM Press, 2006, pp. 95–102.
2. H. Hile and G. Borriello, "Information Overlay for Camera Phones in Indoor Environments," *Proc. 3rd Int'l Symp. Location and Context Awareness* (LoCA 07), Springer, 2007, pp. 68–84.
3. B. Ferris, D. Haehnel, and D. Fox, "Gaussian Processes for Signal Strength-Based Location Estimation," *Proc. Robotics: Science and Systems*, G.S. Sukhatme et al., ed., MIT Press, 2006, pp. 303–311.
4. D. Comaniciu and P. Meer, "Mean Shift Analysis and Applications," *Proc. Int'l Conf. Computer Vision* (ICCV 99), vol. 2, IEEE CS Press, 1999, pp. 1197–1203.
5. D. Guru, R. Dinesh, and P. Nagabhushan, "Boundary Based Corner Detection and Localization Using New 'Cornerity' Index: A Robust Approach," *Proc. 1st Canadian Conf. Computer and Robot Vision* (CRV 04), IEEE CS Press, 2004, pp. 417–423.
6. M.A. Fischler and R.C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Comm. ACM*, vol. 24, no. 6, 1981, pp. 381–395.
7. Y. Hung, P.S. Yeh, and D. Harwood, "Passive Ranging to Known Planar Point Sets," *Proc. IEEE Int'l Conf. Robotics and Automation*, IEEE Press, 1985, pp. 80–85.

Harlan Hile *is a PhD candidate at the University of Washington. His research interests include camera-based interfaces and tools for visual navigation. Hile received his MS in computer science from the University of Washington. Contact him at harlan@cs.washington.edu.*

Gaetano Borriello *is a professor of computer science and engineering at the University of Washington. His research interests include ubiquitous computing, location-based systems, sensor-based inferencing, and use of mobile devices for data collection in public-health applications in the developing world. Borriello received his PhD in computer science from the University of California at Berkeley. Contact him at gaetano@cs.washington.edu.*