Qualitative Image-Based Localization in a Large Building

Christopher Card, William Hoff
Department of Electrical Engineering and Computer Science
Colorado School of Mines
Golden, USA
ccard@mymail.mines.edu, whoff@mines.edu

***Abstract*—Interest in indoor localization is growing because it is an important component of many applications. Image-based localization, using naturally-occurring features in the environment, is an attractive solution to this problem. A challenge is to be able perform this on a mobile device with limited computing power. Another challenge is that buildings can have locations with a similar appearance, which can confuse an image-based recognition system. Since many applications do not need exact location, we focus on qualitative localization, which is the problem of the problem of determining approximate location by matching a query image to a database of images. We propose a novel approach that uses an efficient hashing scheme to quickly identify candidate locations, then applies a strong geometric constraint to reject matches that have similar appearance. On experiments in a large campus building, we show that this approach can localize a query image with high accuracy and has potential to run in real time on a mobile device.**

***Keywords*-computer vision; indoor localization; image retrieval; vision based navigation.**

I. Introduction

The goal of indoor localization is to determine the location of a mobile device in an indoor environment. Interest in this problem is growing because determining location inside a building is an important component of many applications, such as augmented reality, customer navigation, and behavior and movement tracking.

The problem is very different from outdoor localization because the device no longer has access to a reliable GPS signal. A variety of alternative methods can be used in place of GPS. The most popular approaches require some kind of infrastructure to be present in the building. For example, Tesoriero *et al* [1] places Radio Frequency IDentification (RFID) markers throughout the environment. Another approach is to use Wi-Fi fingerprinting, as done by Hile *et al* [2].

An alternative to RFID and Wi-Fi is to use image-based localization. This is attractive because the vast majority of people already have mobile devices (*e.g.* smart phones) with cameras and the approach is applicable to buildings without RFID or Wi-Fi. A recent survey of optical indoor positioning systems is given in [3].

Image-based localization can use naturally-occurring features in the environment. This has the advantage that no infrastructure is required. One challenge is that doing localization based on naturally-occurring features can be computationally intensive, but we want to be able to run our application on a mobile device with limited computing power. Another challenge is that in a large building, there can be many locations that have a similar appearance, thus potentially confusing an image-based recognition system.

In this paper we describe an approach that can perform localization within a large building using no infrastructure or any special mapping steps. The above challenges are addressed in the following ways: first, an efficient hashing scheme is used to quickly identify candidate locations that match a query image. Next, a strong model based on geometric constraints is employed to identify the correct match, while rejecting matches that have a similar appearance. Finally, a local map in the vicinity of the user is constructed to limit the search for candidate matches. Although the system was not implemented on a mobile device, an initial analysis shows that it has the potential to run in real time on a reasonably capable mobile device.

The remainder of this paper is organized as follows. Section 2 describes previous related work and motivates our approach. Section 3 describes the approach in detail. Section 4 provides an evaluation of the system on a large campus building. Section 5 is the conclusion.

II. Related Work

In this section we focus on methods that perform image-based localization by detecting naturally-occurring features in the environment. The most common way of performing localization using natural features is to match them to a 3D model, or map of the building. The 3D map can be automatically constructed from images taken by standard 2D cameras (such as are present on smart phones) using methods known as “structure from motion” (Sfm). Several groups have performed image-based localization by matching features from a query image to 3D points estimated by Sfm [4],[5]. However, Sfm is computationally expensive and requires accurate camera calibration. In our application we may be using images from smart phone cameras from multiple users with no a priori calibration.

Rather than trying to create a metrically accurate map, a qualitative map can be used. For many tasks, the exact

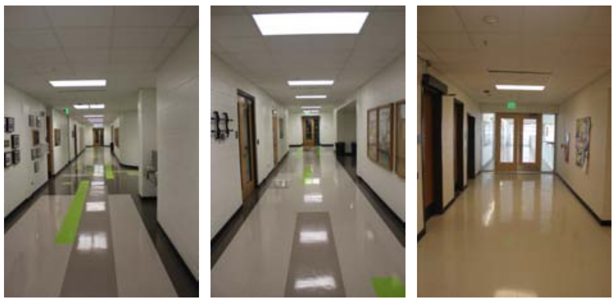
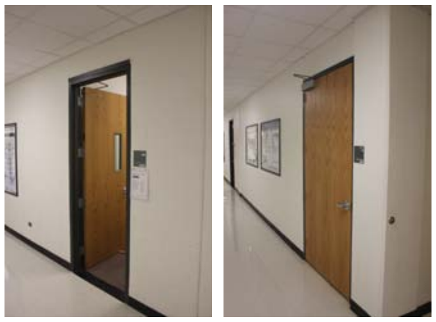


Figure 1. Examples of some of the database images depicting scenes that have similar features but are captured at different locations.



(a) (b)

Figure 2. (a) and (b) show two different doorways that are different but have many similar features.

pose of the camera does not need to be known; approximate locations are sufficient (*e.g.* within 10 or 20 feet). We call this "qualitative localization" (following the terminology of Kosecka [6]). The localization problem then reduces to the problem of matching a query image to an existing image in the database; if such an image can be found then the user is near the location where the database image was taken. The advantage of this "place recognition" approach is that an expensive and difficult map building process is unnecessary and uncalibrated camera images captured by users can be used.

A standard method for place recognition is to use the Bag of Words (BoW) approach. BoW quantizes feature vectors into visual words thus creating a visual vocabulary [7]. To match a query image to an image in a database, the algorithm simply finds the distribution (histogram) of visual words found in the query image and compares this distribution to those found in the database images. Although this could be used for qualitative localization, BoW can fail when the histograms of visual words are too similar.

In a large building, there can be many locations that have a similar appearance. Walls and floors often have little or no texture and doors look very similar. For example, Figure 1 shows a set of images from a large building on the Colorado School of Mines (CSM) campus. There are many features (such as the corners between doors and the floor) which are present in all the images. Thus, the histograms of words are not very distinctive. This would result in incorrect matches to the database.

The *placement* of features in the image is potentially more distinctive than the histogram of features. For example, the images in Figure 2 (a) and (b) are very similar in terms of the types of features that are present. However, the poster to the left of the door is in a different place in each of these two images. This suggests that geometric constraints can be used to uniquely match images. A geometric constraint which is very general and powerful is the fundamental matrix [8]. The locations of all feature points between two images are related via the same fundamental matrix. Therefore, if a fundamental matrix that relates a sufficiently large number of features between two images can be found, then these images were likely taken of the same scene. In our approach, we fit a fundamental matrix to verify candidate image matches. The approach is similar to that of [9], who also uses the fundamental matrix to verify candidate image matches. Using the fundamental matrix as a model for matching is a much stronger constraint than simply comparing histograms of features and should result in much more reliable matches.

III. Localization Algorithm

Our approach is logically divided into three steps which are discussed in the subsections below: (1) feature detection, (2) feature matching, and (3) verification. This section is concluded with a description of the "local map" method.

A. Feature Detection

ORB [10] was chosen as the feature detection algorithm since it provides robustness to image deformation that is close to SIFT and SURF while providing a computational speedup of an order of magnitude [10]. This makes it ideal for localization in real time on a mobile device. In our experiments, ORB descriptors were computed for a query image in 0.05 seconds. The ORB descriptors for the database images were precomputed.

B. Feature Matching

Given a set of ORB descriptors extracted from a query image, these descriptors then need to be matched to the descriptors from the database images. The simplest technique is Brute Force (BF) matching, which exhaustively compares the query descriptor against each database descriptor to find the closest match in feature space. Although the most accurate method, this has the drawback that as the database increases in size the computational time becomes prohibitively expensive.

To avoid this problem, we store the database descriptors in a hash table. The same hash is applied to a query descriptor; the database descriptors at that location in the hash table

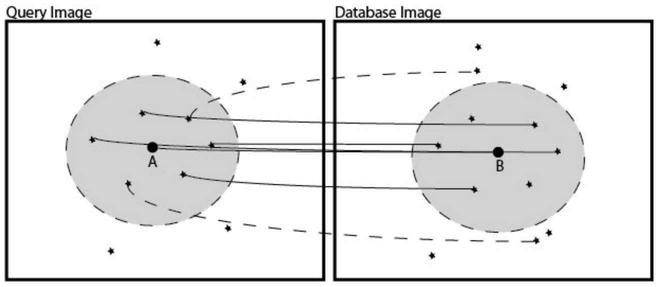


Figure 3. An illustration of spatial consistency voting. This verifies that the kNN of points (A,B) are spatially consistent. Each matched pair of points that is the nearest neighbor to both A and B casts a vote for the match between A and B.

are retrieved. This is a very efficient and fast operation. We use a method called Locality Sensitive Hashing (LSH) [11] which preserves the locality of key points in feature space when generating the hash of the image descriptor. In other words, the difference between hash values is a good approximation of the distance between the points in feature space. This allows finding nearby descriptors in feature space, not just the descriptors at the hash location. This is important because image deformation and noise can cause the descriptors to change. Our algorithm finds the k nearest neighbors for the query point, where $k = 15$.

LSH is extremely fast when matching features against a large database (1,073,903 feature points). In our experiments LSH was able to match against a large database in about 1.47 seconds and it was able to match against a local map (containing 33,000 feature points) in about 0.095 seconds.

The potential matches for each query point, q_i, are then filtered using two steps, as described below:

1) **Ratio test**. The first step of the filter process is to determine if there are multiple feature points from one database image that are nearest neighbors to q_i. If this is the case, the closest feature point from the database image has to be 80% closer to q_i than the second closest feature point; otherwise the match is discarded.
2) **Spatial consistency test**. The next step checks whether each matched pair of points is spatially consistent. The approach of Sivic *et al* [12] is used for this step (see Figure 3). The idea is that neighbors of the query point should have matches that are neighbors of the database point. Here, "neighbors" means that the points are neighbors in image space, not feature space. If the number of spatially consistent neighbors is below a threshold (a threshold of 6 points was used in this work), then the potential match is discarded.

After the two filtering steps are completed, the two database images with the highest number of matches to the query image are selected. These are the candidate matches to the query image. If there is a tie for second place, all images that are tied for second place are kept.

C. Verification

The verification step of the algorithm tests each candidate database image to see if the matching points fit a geometric constraint with the query image. The model used for the geometric relationship is the fundamental matrix [8]. The fundamental matrix models the epipolar geometry between two camera views of the same scene. RANSAC [13] is used to eliminate outliers. A fundamental matrix is found between the query image and every candidate database image. Then the image with the most inliers is found. If the number of inliers exceeds a threshold (described in Section 4), then that database image is determined to be the correct match. If not, then all the candidate images are passed to a secondary processing step.

The secondary processing step rematches all image features in the query image to the candidate set of images, except that it now uses BF matching instead of LSH. The image with the most inliers to a fundamental matrix is found and, if the number exceeds the threshold, it is determined to be the correct image; otherwise, the query image is considered to have no acceptable match to the database.

D. Local Map

If the size of the database can be reduced, this can potentially speed up computation as well as improve the accuracy of matching. To do this, we propose using a "local map" in the vicinity of the user which contains only the database images near the current location of the user. The size of the local map depends on how fast a user can reasonably walk in a given amount of time. As long as the user is within the boundaries of the local map, localization queries can be done by matching to the local map.

Our concept for this is as follows: When a user first runs our system to do localization, the image is sent to a server, which matches the query image to the entire image database. Once the user's approximate location is found, the system sends a local map to the user's mobile device. The user's mobile device then uses the received local map to perform localization. This greatly speeds up processing and improves accuracy which allows the mobile device to perform localization in near real time. When the user approaches the edge of the local map, the server downloads a new local map to the client. Although we did not implement this concept, we did evaluate the potential benefits of using a local map in terms of run time and accuracy, as described in the next section.

IV. Experiments

This section describes the database used and details the methodology applied to test the algorithm. We implemented our algorithm using C++ and the open source software OpenCV. The algorithm was tested on a laptop running Windows 7 with a 2.6GHz processor and 4GB of RAM.

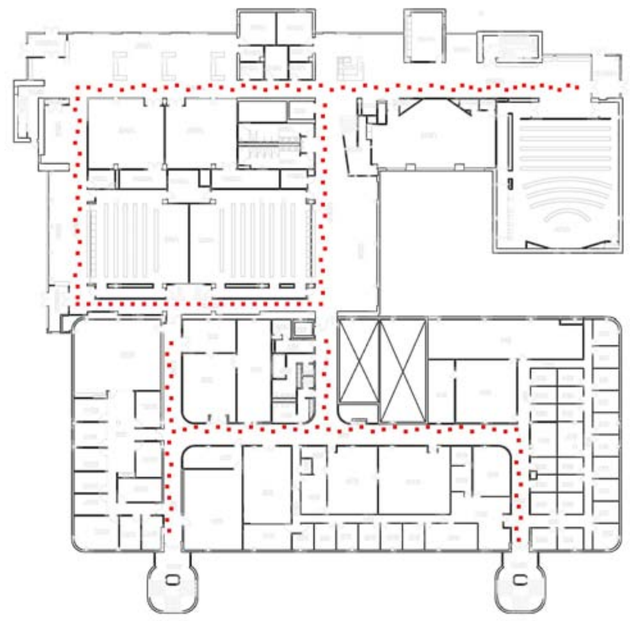


Figure 4. Red dots indicate where images were captured on the 2^{nd} floor of Brown Hall.

A. Database

The database was captured using a Cannon Rebel t2i Single Lens Reflex (SLR) camera with 8 megapixels per image. The database was captured in Brown Hall at CSM which is a large (100,000 square foot) building containing offices, classrooms and laboratories. The database consists of the 1^{st}, 2^{nd}, and 3^{rd} floors of Brown Hall, as these floors contain a representative sample of indoor environments which contain sparse texture and similar structural features. The images were taken at intervals of approximately 5 feet (see Figure 4). At each position multiple images were taken, facing both directions in the hall and additional directions to capture the appearance of nearby characteristic features (*e.g.* doors, side halls). The location where each image was taken was physically measured and recorded. The operation of the system is not dependent upon knowing these locations. These measurements were taken solely to test the system's accuracy.

The database is comprised of 1,382 images with a total of 1,073,903 feature points. Images in the database overlap, meaning that nearby images typically view a portion of the same scene (see Figure 1 for example images).

B. Tests

The following subsections describe the tests used to evaluate the algorithm using the collected database. A match is deemed to be correct if the location of the database image is less than 21 feet from the query image. In our tests, the correct match to a query image was in the database about 92% of the time.

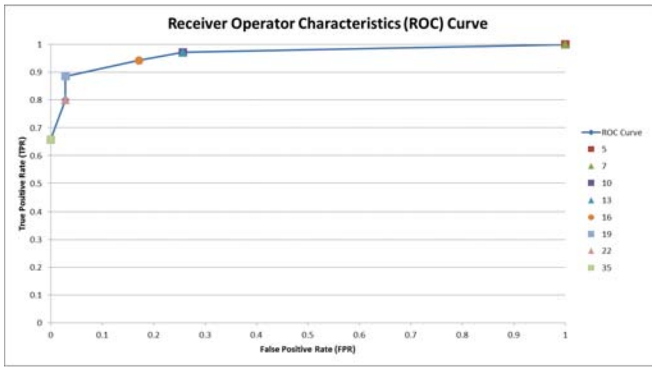


Figure 5. The ROC curve for a test set of 70 images matched against the remainder of the database. The same test set was used for all 8 runs.

1) Parameter Evaluation: One of the most important parameters in the algorithm is the threshold for the number of inliers to a fundamental matrix, which determines if a query image is successfully matched.

To avoid incorrect matches, it is desirable to use a higher threshold for the required number of inliers. This reduces the probability of a false match. However, this also reduces the probability of a true match. Conversely, lowering this threshold makes it more likely that a query image will be successfully matched to the correct database image. However, if a query image actually has no correct match in the database, lowering the threshold also increases the probability that a false match will occur.

To evaluate the effect of changing (*i.e.* tuning) this parameter on the probability of getting a false match, the following study was done. We randomly chose 35 images and removed them from the database and ensured that each of the 35 images had a correct match in the database. Thirty-five other images were captured (using the same camera) from parts of the building that were not in the database. These images have no correct match in the database.

A Receiver Operating Characteristics (ROC) curve was generated. ROC curves are based on a 2x2 confusion matrix, which records the count of the four possible outcomes of running the localization algorithm at each setting of the algorithm parameter. The four possible outcomes are:

- True Positive (TP). The correct match to the query image was in the database and the system found the correct match.
- True Negative (TN). The correct match to the query image was not in the database and the system correctly decided that there was no match.
- False Positive (FP). The correct match to the query image was not in the database, but the system matched it to an image that was not correct.
- False Negative (FN). The correct match to the query image was in the database, but the system was unable to find a match.

Table I
THE RESULTS FOR THE SUBSAMPLE TESTS. EACH PART OF THE TABLE CONTAINS THE SUM FROM 20 SUBSAMPLE TESTS.

Outcome	Number
Query has a match in database and algorithm found a correct match	538
Query has a match in database and algorithm found an incorrect match	14
Query has a match in database and algorithm declared “no match”	27
Query has no match in database and algorithm found an incorrect match	4
Query has no match in database and algorithm declared “no match”	17
Total number of queries	600

The True Positive Rate (TPR) is defined as the ratio of true positives (TP) to the total number of positives (TP+FN). The False Positive Rate (FPR) is defined as the ratio of false positives to the number of total negatives (FP+TN) [14].

The ROC curve is formed by plotting TPR against FPR. The resulting ROC curve is shown in Figure 5. As can be seen, the TPR is fairly high for most parameter settings. For example, using a threshold of 16, the TPR is about 94%, meaning that if the correct match is in the database the system will find it 94% of the time. The FPR for this case is about 17%, meaning that in those cases when the correct match is not in the database, the system finds an incorrect match instead of outputting a “no match” decision. Although this FPR seems high, the number of cases where there is no correct match in the database is small, so this outcome is relatively rare.

2) Subsample Test: To assess the overall accuracy of the algorithm over multiple runs, a subsample test was performed. Twenty test sets were created, where each test set consisted of 30 randomly chosen images from the full database, with no restriction on proximity. For each test set, the 30 images were removed from the database and then were used to query the database. In this experiment a threshold of 16 inliers was used as the decision threshold.

Overall, the algorithm performed well. Combining the results from all 20 subsample tests, the algorithm achieved an accuracy of 92.5% (see Table I). Here accuracy is defined as the fraction of all outcomes that were correct. Specifically, it is the number of outcomes in rows 1 and 5 in the table divided by the total number of trials. These results show that the algorithm can localize a query image with a high degree of confidence. Two examples of TPs are shown in Figure 6. Two examples of the query image having a match in the database but the algorithm found an incorrect match (FP) are shown in Figure 7. The FPs were caused by a set of highly clustered points. An FN is shown in Figure 8 where an insufficient number of inliers were found.

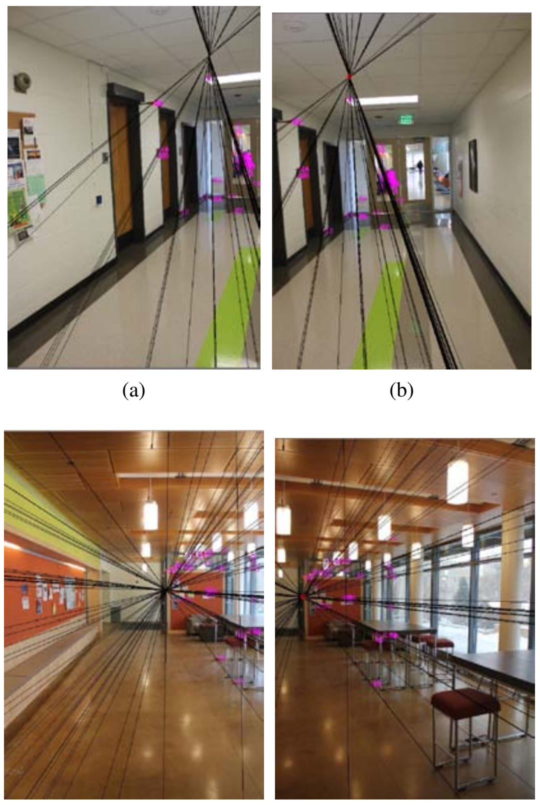

(a) (b)
(c) (d)

Figure 6. These are two examples of TP matches; (a) and (c) are the retrieved database images to their respective query image (b) and (d). The black lines are the epipolar lines found using the fundamental matrix and the pink numbers are points that are inliers to the fundamental matrix.

The average time to match a single query image to the full database (minus the 30 images for each test set) was 6.22 seconds. While not especially fast, this only needs to be done once, when the user first performs the localization step. After that, the localization steps are performed with a local map that has a much smaller database of images. These steps are much faster, as is described in the next subsection.

This testing procedure was also used to compare the accuracy of using BoW as the indexing method instead of LSH. Following the method of Sivic *et al* [12], a vocabulary of 10,000 words was created from the ORB descriptors extracted from the database images. A query image is then mapped into its constituent visual words, and the distribution of visual words is converted to a “term frequency-inverse document frequency (tf-idf)” vector. The 10 most similar database images are returned as candidates, and the geometric constraint verification step is applied to each of these.

Using BoW as the indexing method resulted in an accuracy of 83.7% which is lower than using LSH, which had 92.5% accuracy. One possible reason for the lower performance of the BoW method is that it computes the distribution of visual words from the entire image. If two

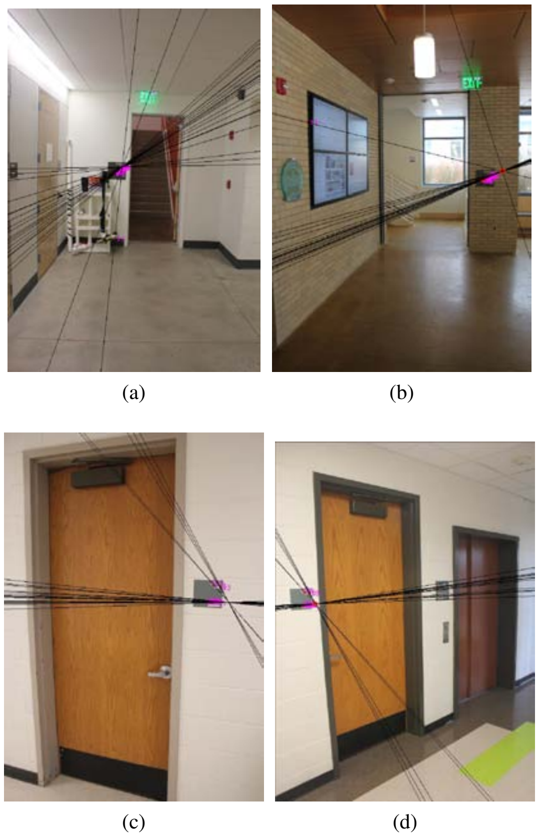


Figure 7. These are two examples of FP matches; (a) and (c) are the retrieved database images to their respective query image (b) and (d). The black lines are the epipolar lines found using the fundamental matrix and the pink numbers are points that are inliers to the fundamental matrix.

images have only a small overlapping area, the distributions from the two images can be significantly different. For example, Figure 9 shows a query image and the database image it should match. However, BoW failed to match these two images because they only overlap by 20-30% whereas LSH correctly matched these two images.

3) Local Map Test: Once the first query image is localized by the server, the mobile device receives a local map of images surrounding its current location. The number of images in the local map is chosen so that users will likely remain within this local map for only a short time. Thus, all queries performed in that time frame will most likely correctly match to an image in the local map. The motivation for using a local map is that it will reduce the time to perform a query as well as improve the accuracy of the algorithm.

In our test, 65 images were used to form the local map because that number of images approximates the distance a user who is unfamiliar with a building would travel in about 20 seconds. For query images, 19 test images from the 3^{rd} floor of Brown Hall were captured independently from the database in the same area as the images in the local map.

The results from localization using the local map showed

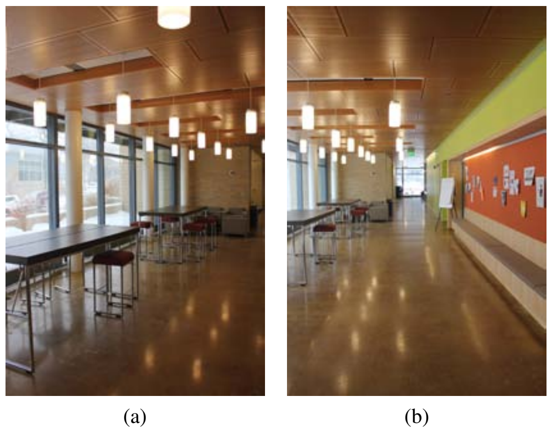


Figure 8. (a) was the query image used in the test and (b) was the database image. These two images should have resulted in a TP but an FN occurred.

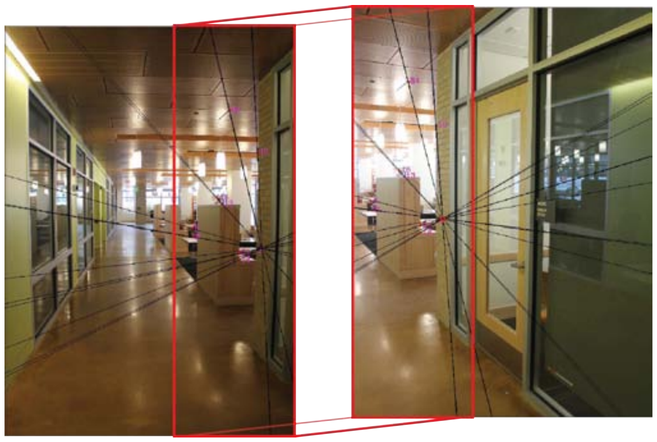


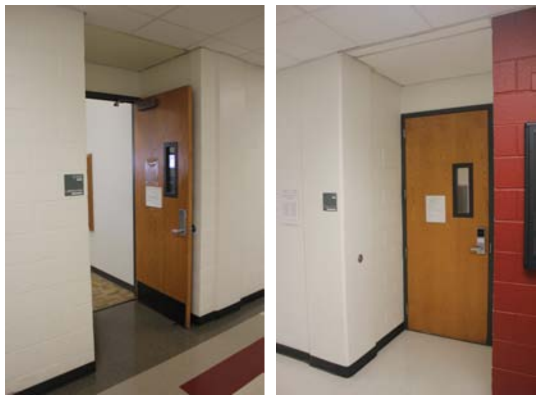
Figure 9. The right is the query image and the left is the database image that it should match. The red rectangle indicates the overlap of the images. LSH correctly matched these two images (the pink points show the matched features), but BoW failed to match these two images.

an accuracy of 94.74% (see Table II). The fact that the accuracy of the test is not closer to 100% is because of minor changes to the environment between the time that the database images and the query images were captured (see Figure 10). However, changes like this are to be expected as the indoor environment is not static. If the environment changes the database needs to be updated. This is a problem for this approach as well as the other approaches researched ([2], [15], [16]).

The system took an average of 1.902 seconds to localize a query image. These results show that the algorithm has the potential to run on a mobile device in near real time.

V. Conclusion

In this paper we have presented a novel approach to indoor localization that does not require any additional infrastructure or any special mapping techniques. Using only naturally-occurring features in the environment it was



(a) (b)

Figure 10. The query image (a) incorrectly matched the database image (b). This FP is the result of a change in the environment that caused (a) to have a similar appearance to (b).

Table II
THE RESULTS FOR THE LOCAL MAP TESTS.

Outcome	Number
Query has a match in database and algorithm found a correct match	16
Query has a match in database and algorithm found an incorrect match	1
Query has a match in database and algorithm declared "no match"	0
Query has no match in database and algorithm found an incorrect match	0
Query has no match in database and algorithm declared "no match"	2
Total number of queries	19

demonstrated that our approach can qualitatively localize an image in a large building with a high degree of confidence. The results also show that the use of a local map around the mobile device's known location improves the accuracy of localization. Although the approach was not implemented on a mobile device our analysis shows that it has the potential to run in real time on such a device. Future research could include implementing this approach on a mobile device as well as exploring other ways that the accuracy of the algorithm can improved.

REFERENCES

[1] R. Tesoriero, R. Tebar, J. A. Gallud, M. D. Lozano, and V. M. R. Penichet, "Improving location awareness in indoor spaces using RFID technology," *Expert Systems with Applications*, vol. 37, no. 1, pp. 894–898, 2010.

[2] H. Hile and G. Borriello, "Positioning and Orientation in Indoor Environments Using Camera Phones." *IEEE Computer Graphics and Applications*, vol. 28, no. 4, pp. 32–39, 2008.

[3] R. Mautz and S. Tilch, "Survey of optical indoor positioning systems," in *Indoor Positioning and Indoor Navigation (IPIN), 2011 Intl Conf on*. IEEE, 2011, pp. 1–7.

[4] H. Lim, S. N. Sinha, M. F. Cohen, and M. Uyttendaele, "Real-time image-based 6-dof localization in large-scale environments," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conf on*. IEEE, 2012, pp. 1043–1050.

[5] T. Sattler, B. Leibe, and L. Kobbelt, "Fast image-based localization using direct 2d-to-3d matching," in *Computer Vision (ICCV), 2011 IEEE Intl Conf on*. IEEE, 2011, pp. 667–674.

[6] J. Kosecka, L. Zhou, P. Barber, and Z. Duric, "Qualitative image based localization in indoors environments," in *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conf on*, vol. 2. IEEE, 2003, pp. II–3.

[7] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer, 2010, section 14.4.1 pg 697-701.

[8] Q.-T. Luong and O. D. Faugeras, "The fundamental matrix: Theory, algorithms, and stability analysis," *Intl journal of computer vision*, vol. 17, no. 1, pp. 43–75, 1996.

[9] D. Sinha, M. T. Ahmed, and M. Greenspan, "Image retrieval using landmark indexing for indoor navigation," in *Computer and Robot Vision (CRV), 2014 Canadian Conf on*. IEEE, 2014, pp. 63–70.

[10] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: an efficient alternative to SIFT or SURF," in *Computer Vision (ICCV), 2011 IEEE Intl Conf on*. IEEE, 2011, pp. 2564–2571.

[11] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proceedings of the twentieth annual symposium on Computational geometry*. ACM, 2004, pp. 253–262.

[12] J. Sivic and A. Zisserman, "Efficient visual search of videos cast as text retrieval," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 4, pp. 591–606, 2009.

[13] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[14] J. Egan, *Signal Detection Theory and ROC-analysis*, ser. Academic Press series in cognition and perception. Academic Press, 1975.

[15] M. Werner, M. Kessel, and C. Marouane, "Indoor positioning using smartphone camera," in *Indoor Positioning and Indoor Navigation (IPIN), 2011 Intl Conf on*. IEEE, 2011, pp. 1–6.

[16] H. Kawaji, K. Hatada, T. Yamasaki, and K. Aizawa, "Image-based indoor positioning system: fast image matching using omnidirectional panoramic images," in *Proceedings of the 1st ACM international workshop on Multimodal pervasive video analysis*. ACM, 2010, pp. 1–4.