Visual Indoor Positioning with a Single Camera Using PnP

Edith Deretey, Mirza Tahir Ahmed, Joshua A. Marshall, and Michael Greenspan  
Department of Electrical and Computer Engineering 
Queen’s University, Kingston ON, Canada  
Email: {8ejd1, 81mta, joshua.marshall, michael.greenspan}@queensu.ca

Abstract—This paper introduces an accurate and inexpensive method for localizing a calibrated monocular camera in 3D indoor environments. The objective of this work is to localize in 6 degrees-of-freedom (6 DOF) in the presence of a 3D map that contains 3D point clouds co-registered with intensity information. This is done by solving the Perspective-n-Point (PnP) problem to accurately compute the camera location in 6 DOF. An efficient data structure is used to store a large set of point clouds co-registered with intensity information, image features, and transformations between the frames. This data structure, referred to as the feature database, is implemented such that it retrieves a match for a query image efficiently. Thus the overall process of localization in 6 DOF becomes a real-time process with high efficiency and accuracy. Our technique was tested with two ground truth data sets of indoor environments, an office and a laboratory. The experimental results show the accuracy and the efficiency of our technique, with an average localization error of less than 10 mm from the ground truth in both environments. In addition, localization results on query images obtained using two different cameras in four different environments are presented. This demonstrates that any type of monocular camera may be used during localization, as long as a sufficient number of environmental features can be extracted from the query images.

I. INTRODUCTION

The Global Positioning System (GPS) is a large component of outdoor localization techniques. Along with the use of other sensors such as Inertial Measurement Units (IMUs), outdoor localization systems such as [1] can accurately determine the position of the sensor suite. However, GPS is not available in the case of indoor localization because there is no direct line of sight with satellites. Therefore the use of other sensors is required to obtain information about absolute position.

This paper presents a computationally inexpensive and effective solution for indoor 6 DOF localization using a calibrated monocular camera. The input to the system is a pre-computed 3D map of the environment consisting of 3D point clouds with co-registered intensity information. 3D sensors, such as the Microsoft Kinect or stereo cameras, can be used to capture the environment map offline. Frames are normally registered together to develop a 3D map of the environment using mostly feature based techniques.

The online localization algorithm searches for feature correspondences in the 2D image space between a query image and the map. Once a good number of inlier correspondences are found, the Perspective-n-Point (PnP) problem is used to accurately compute the location of the calibrated monocular camera in 6 DOF. This not only relieves the need of using a 3D sensor for localization, as has previously been proposed, but also improves the performance of the system due to the reliability of feature extraction and matching in the 2D image space. With this technique, an inexpensive calibrated web camera can be used for indoor localization in a 3D map.

A. Related Work

PnP is the problem of determining the position of a calibrated camera in 6 DOF given the 3D location of features and their corresponding 2D image projections. A variety of solutions exist for the PnP problem, such as [2], [3], [4] and [5] to name a few. Some solutions deal with the general case, and other solutions focus on specific configuration of points. Most of the available solutions can be divided into two categories: iterative solutions and direct solutions. Iterative solutions are generally fast and accurate, but need to be initialized accurately in order to obtain a correct solution. On the other hand, direct solutions can only handle a limited number of points and are therefore sensitive to noise and outliers [6]. The PnP problem can result in up to four candidate solutions, depending on the configuration of the input points, only one of which represents the true pose.

The main application of PnP is to find the position of the camera. In addition, PnP can also be used to track and localize objects that are in the camera’s field of view. Some Simultaneous Localization and Mapping (SLAM) systems performed with RGB-D sensors make use of PnP to aid in the localization process such as in [7]. Others use it to relocalize during SLAM when the sensor’s location has been lost [8].

SLAM is the process of building a map of an unknown environment while concurrently determining the position of the sensors, for example a camera. In order to find the position, a model of the environment is required. On the other hand, information about the position must be known in order to create a model of the environment [9]. Stereo vision systems are commonly used to perform visual SLAM [10]. Clipp et al. [11] determine the camera motion by using optical flow, triangulate the features and use PnP to find the camera position. Structure from Motion is also used in monocular SLAM techniques [12].

Visual localization techniques include those using geometric constraints of the features. For example Sun et al. [13] presented a technique for monocular self-localization in an indoor environment using circular landmarks. Their solution requires at least two landmarks, in their case ceiling lights, to be in the field of view of the camera. The geometric shapes
and PnP are combined to solve for position. Xu and Liu [14] proposed a method of monocular localization for planar objects using PnP along with the planar geometry. These localization techniques do not require prior knowledge of the environment. However, they only work in environments where the geometric constraints are met.

Teach-and-Repeat algorithms are also used for visual localization. These algorithms generally consist of two stages. During the teach stage, the camera is manually guided through a predefined path and collects images of its surroundings. These images are processed and a 3D map of the environment is created using structure from motion [15]. In the localization stage, the sensor captures images of its surroundings and compares it against the 3D model of the environment to localize within the map. Notable teach-and-repeat techniques include [16] for a monocular camera and [17] for a stereo camera.

Our technique falls into the category of teach-and-repeat algorithms and is similar to that of Royer et al. [15], wherein the images used for both the teach and localization component are collected with the same calibrated monocular camera. Structure from motion is used to build the 3D map of the environment. The information from the map is stored in a database containing landmarks with their 3D location, in addition to position information for all the keyframes in the map. Their localization begins by comparing the first image against the database to find the closest match, using that keyframe’s position as an estimate for the starting position. The assumption they use is that the movement of the camera between consecutive frames is small, therefore the approximate pose of the new frame is the same as the calculated pose of the previous frame. They then project the interest points present in the matched keyframe and find the 2D projection of those points on the current frame. The projections are then matched to the image interest points, which should be located within a small region of interest around the projected interest point. The matching step provides them with a list of 2D image points and 3D object points which they pass into PnP. The output of PnP gives them the pose of the camera. This pose is refined to obtain a final solution for the camera position based on the current frame.

In the solution presented by Royer et al., PnP is used to correct the position of the current frame, using the position of the previous frame as a starting point. The system we present relies entirely on PnP to find the position of the camera, with no approximate estimation of position necessary. Therefore, the motion of the camera between consecutive frames does not need to be small, as each frame is only compared against the feature database. The efficiency of the feature database allows us to do this while maintaining quick localization.

The remainder of the paper is organized as follows: Section II describes the acquisition of the map and the creation of the feature database, along with a description of the localization procedure. Section III describes the two experiments that were performed. The first experiment was to test the accuracy of our technique against a ground truth. The second experiment tests the performance of our system in four different and varied environments. Finally, Section IV summarizes the paper and discusses future work.

## II. Method

### A. Mapping

The proposed system assumes a map of the environment has been precomputed offline and is available. There are many registration techniques that stitch together successive point clouds to form a map [18], [7]. RGBDSLAM, an existing open source SLAM solution, is used to generate a map of the environment [7]. RGBDSLAM uses a Microsoft Kinect (version 1) sensor to capture the data. The inter-frame transformations are computed using 3D point correspondences, obtained by computing 2D SIFT [19] or SURF [20] features, and projecting matched 2D features between two frames to 3D points. Normally the Microsoft Kinect captures frames at the rate of 30 frames per second (FPS). Generally, only keyframes are used to generate maps. In the literature, there are several techniques that discuss keyframe extraction for the mapping process [21]. The RGBDSLAM algorithm normally rejects those frames where there is not a significant movement. The whole process is optimized using a pose graph optimization technique [22]. Once the map is generated, all the poses and graph edges are saved.

### B. Feature Database

The feature database described in [23] is used to store the features captured in all the saved frames. The features are stored in such a way such that the best matched frame can be extracted in an efficient manner. As a first step, all the frames in the map are processed to generate the landmark index data structure. Once the landmark index data structure is ready, query frames can be processed and searched to find out the best matched frame. The best matched frame is the database frame containing the most features of the query frame. A short summary of [23] is provided in this section to describe the key elements of the feature database.

1) Landmark Index Generation: A landmark is defined as a set of distinguished features which are regularly observed and re-observed in the mapping environment. Each landmark must be stationary and distinctive in the map. Features from moving objects cannot be considered as landmarks. Landmarks must be repeatable and robust to noise and lighting conditions. A landmark is more stable if it appears in consecutive frames but totally unstable if it appears in all the frames.

The procedure extracts visual features from images using selected computer vision techniques, for example, SIFT, SURF, or ORB. SURF has been used for all the experiments presented in this paper. The extracted features are processed to create a reduced list of features annotated with the frame numbers they appear in. Given by

\[ \langle f_i, (c_{i1}, c_{i2}, \ldots, c_{iN}) \rangle, i \in [1,M] \]  

where \( M \) is the total number of features, \( N \) is the total number of frames, \( f_i \) is the \( i \)-th feature descriptor and \( c_{ij}, j \in [1,N] \) is a boolean value which is

\[ c_{ij} = \begin{cases} 
0 & \text{if } f_i \text{ does not appear in frame } j \\
1 & \text{if } f_i \text{ appears in frame } j 
\end{cases} \]

This actually maps the similar features to the same landmark which appear in different frames. Such mapping reduces the search space to improve search efficiency.
2) Image Search: Features are extracted for each query frame. FLANN [24], a nearest neighbour search mechanism, is employed to match the query features with the set of database features. Each matched database feature is considered as a vote for the corresponding frame. For each frame the total number of votes is computed. The frame corresponding to the maximum votes is normally the best matched frame. However, it is possible that there are two or more frames with similar number of votes or there is an outlier frame with the same number of votes. Therefore, a moving average of votes of neighbouring frames is computed. The frame with the highest average is considered as the most reliable frame. The search process is extremely efficient in retrieving the query frame from the set of frames stored in the database. Once the frame is retrieved, matched features are used for the PnP process as described in the next section.

C. Localization using a Monocular Camera

A monocular camera is used to obtain images of the environment during localization. The camera is calibrated ahead of time using a checkerboard, so that its intrinsic parameters are known. Each captured frame is processed by the system to find the camera position at the time of image capture. This allows for tracking the movement of the camera through a series of frames. The position of the camera for each frame is presented in the reference frame of the 3D map. In what follows, the images obtained from the monocular camera are referred to as query images, and images obtained from the feature database are referred to as database images.

To begin localization, the current view of the camera serves as a query image to the feature database in order to retrieve the best matched image using landmark indexing [23]. The database also provides information about the retrieved image such as the 2D features and their corresponding 3D points.

PnP takes a list of 3D points and their corresponding 2D image projections as input. Since the query image only provides 2D data, a few steps are required to prepare the data for PnP. First 2D features in the query image are computed using OpenCV’s [25] SURF detector and extractor, which was selected for its speed and accuracy. It is important to ensure that the same feature detector and extractor are used on both the query and the database images. Next, a descriptor matcher is used to find matches between the features in the query image and the matched database image. The current implementation uses the brute force matcher. Since the number of features matched between the query image and the selected database image are relatively small, on the order of tens or hundreds, the brute force matcher is effective. It is possible to use a more efficient descriptor matcher in the future if necessary. These matches provide the location of identical features in the database image. Since the 3D location of the features are available for the database image, the matches are used to find the location of features in the query image. A list of 3D points and their 2D projections in the query image are then created. The list of 2D-3D correspondences, along with the camera calibration parameters are passed into PnP [5] to obtain a 6 DOF solution for the camera pose. The main steps of the localization algorithm are summarized in Figure 1.

The proposed technique uses EPnP, presented by [5], within a RANSAC framework to obtain the position of the camera.

![Flowchart depicting the main components of the localization algorithm.](image)

The main idea behind EPnP is to describe all 3D input reference points as a weighted sum of four virtual control points. After solving for the position of the virtual control points, the rotation and translation of the camera can be found.

The reference points \( p \) can be expressed as

\[
p_i = \sum_{j=1}^{4} \alpha_{ij} c_j
\]

(2)

where \( \alpha_{ij} \) is the weighting for the \( i \)-th reference point in terms of the \( j \)-th virtual control point and \( c_j \) are the coordinates of the virtual control points. The first virtual control point is selected to be the centroid of the input points, and the other three are found using principle component analysis.

The projection of the each reference point onto the image plane can be expressed by

\[
w_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & 0 & u_c \\ 0 & f_v & v_c \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}
\]

(3)

where \( w_i \) is the projective parameter, \( (u_i, v_i) \) is the coordinate of the \( i \)-th image point and \( (x_i, y_i, z_i) \) is the coordinate of the \( j \)-th virtual control point. The last row in (3) is substituted into the first two rows to obtain (4) and (5).

\[
\sum_{j=1}^{4} \alpha_{ij} f_u x_j + \alpha_{ij} (u_c - u_i) z_j = 0
\]

(4)

\[
\sum_{j=1}^{4} \alpha_{ij} f_v y_j + \alpha_{ij} (v_c - v_i) z_j = 0
\]

(5)

The equations can be rearranged to obtain a \( 2n \times 12 \) matrix, solving for the unknown 3D coordinates of the four virtual control points. The solution to the unknowns is found by computing the null space of the matrix containing the known values. In cases where multiple solutions are present, the solution providing the smallest reprojection error is selected as the correct one. The reprojection error is calculated by

\[
\text{res} = \sum_{j} \text{dist}^2 \left( A[R|t] \begin{bmatrix} p_i^w \\ 1 \end{bmatrix}, u_j \right)
\]

(6)

where \( A \) is the camera’s intrinsic matrix, and \( \text{dist}(a,b) \) is the 2D distance between point \( a \) and point \( b \), represented in homogeneous coordinates.

III. Experimental Results

The results of two sets of experiments are presented in this section. In the ground truth experiments, the accuracy of the localization was compared against the position of the
frames obtained by RGBDSLAM. This was done by using the images in the database as the query images. In the localization experiments two different cameras were used to collect query images in various environments to test the performance of the system.

A. Ground Truth Testing

Two data sets were used for the ground truth tests: Office and Lab. Both data sets were collected using the Microsoft Kinect (version 1) sensor. The Kinect obtains intensity images and dense depth maps at a rate of 30 FPS.

Fig. 2: 3D map of Office dataset

The Office data set contains 50 frames that were collected in the Electrical and Computer Engineering department office at Queen’s University, with an area of 139 m². During the first 36 frames of the data set, the Kinect was slowly rotated 360° around its vertical axis at the center of the office. For the remaining 14 frames, the sensor was moved along the length of the office. Most of the office doors were closed and the blinds were drawn in order to keep the lighting conditions consistent and to obtain good quality images with many features.

Fig. 3: 3D map made from Lab dataset

The Lab data set consists of 33 frames collected in a laboratory environment. This is a much smaller environment than the Office data set, with an area of 37 m². The average distance of the objects was 1.5 m.

The accuracy of the proposed system was tested against the Robot Operating System’s RGBDSLAM [9], which was used as the reference position. One of the outputs of RGBDSLAM is a series of transformation matrices representing the Kinect’s location during each frame’s capture. The frames are in the local reference frame of the first frame. Therefore, the rotation matrix of the first frame is the identity matrix. However, all the other frames have transformation matrices that represent the movement of the Kinect relative to this reference frame.

In this experiment, a subset of the 2D images in the database was used as the monocular camera query images. The output of the experiment was a series of transformation matrices describing the position of the sensor. This was compared against RGBDSLAM’s transformation matrix by comparing the translation component and the rotation component respectively. The position of the camera is equivalent to the translation vector. The translation error in the camera position was found using the Euclidean distance formula

\[ d(p, q) = \sqrt{(q_x - p_x)^2 + (q_y - p_y)^2 + (q_z - p_z)^2} \]  

(7)

where \( p \) is the position of the camera provided by RGBDSLAM and \( q \) is the position of the camera recovered through the proposed system. The rotation is compared by computing the difference in the Euler angles. The translation error between the system and the reference position for the Office data set ranges between 0.9 mm to 35 mm. The translation error was plotted as a function of the number of inliers in the image and can be seen in Figure 4. Inliers are features that have a reprojection error that is smaller than the selected threshold. The more inliers there are, the higher the accuracy of the solution tends to be. Considering that the size of the office is in the order of metres, a position error in the order of millimetres is very small.

In cases when the inliers happen to be at a similar depth with respect to the camera, the position error slightly increases. This is because it is a degenerate case for the PnP to determine information about depth when all the reference points are coplanar. A similar pattern was found for the Lab data set,
Fig. 5: Plot of the translation error of query frames in the Lab dataset as a function of the number of inliers.

as can be seen in Figure 5. For this experiment, the number of minimum inliers was set to a threshold of 100, with a translation error ranging from 0.3 mm to 17 mm.

Fig. 6: Visualized camera positions in Office data set. The rectangular region in the first image is magnified in the second image. Similarly, the rectangular region in the second image is magnified in the third image.

Figure 6 shows three screen captures from the visualization of the camera position within the Office data set. Two different coloured spheres are used to represent the position of the camera. The green sphere represents the ground truth camera position obtained by RGBDSLAM and the red sphere represents the camera position obtained by our localization technique. The centre of the sphere is the camera position. Since the spheres occupy the same area in the image, they appear to be within one another. Due to this, in cases where the obtained position is less than 1 mm away from the reference position, the spheres will exhibit a single colour. For some of the positions, the spheres were offset a little, which shows the translation error discussed above. However since most of this error is in the order of millimetres, it is difficult to see a large offset.

The localization system was implemented in C++ using the OpenCV and Point Cloud Library and tested on an Intel Core i7 machine with 8 GB of RAM. The efficiency of the system was assessed by measuring the total run time, and the time for searching through the feature database. The Lab dataset contained 12842 features and had a total runtime of 31.26 sec for localizing 33 query frames (i.e., 0.95 sec/frame). The Office dataset contained 6000 features in the database, with a total runtime of 28.52 sec for 50 frames (i.e., 0.57 sec/frame). The system is based on image features, therefore even though the Lab dataset had less query frames, the program had a longer runtime because there were twice as many features to search through in the database per frame. The average localization time for both data sets is 0.72 sec per frame which is similar to consumer-level GPS.

B. Localization Results

Four datasets have been used to test localization in different and varied environments. These datasets are called Office712, JDUC, WLH, and Mine. The 3D maps for all these datasets were collected using RGBDSLAM with the Kinect. For the first three datasets, the query images used for localization were collected with a Logitech webcam with 640 × 480 resolution. The query images for the Mine dataset were obtained using two GigE Point Grey Blackfly cameras with a resolution of 1280 × 1024. Therefore the query images used for localization were all collected with a different camera than the database images. This section demonstrates that the system is not constrained to one particular camera, and that different cameras with different resolutions can be used at the time of localization. It also demonstrates how the system performs in a number of different environments.

The Office712 dataset, containing 10918 features in the feature database, was collected in a graduate student office at Queen’s University. The size of the area where the data was collected is about 8 m by 10 m. Six images form the feature database have been included in Figure 7 to provide a better idea of the environment. The dataset contains a part of the office surrounded by three walls, which contains six large cubicles in a 3 × 2 arrangement and a set of eight smaller cubicles in a 4 × 2 arrangement. The divider between the cubicles is visible in the 3D map, along with some parts of the tables and chairs. One side of the dataset is surrounded by a wall containing very few features, the opposite side contains bookshelves, making it a feature-rich environment, and the third wall contains large windows. The blinds were drawn during data collection to minimize the amount of light entering the room, because the Kinect sensor does not work well in areas containing direct sunlight.

Figure 7 shows three sets of query images for Office712 were collected in adjacent cubicles. In each cubicle the camera was rotated 360° to collect images of all the surroundings. All the images were passed into the system, but only the positions of images containing ten or more inliers were used for localization. While only four inliers per frame are required for our system to obtain a position, increasing the minimum number of inliers to ten ensures that most of the erroneous positions are eliminated. Unless otherwise mentioned, a minimum of ten inliers was used for all other datasets.
Fig. 7: Sample of images from the Office712 database.

Fig. 8: Office712 query frame locations collected in cubicle 1.

Figure 8 shows the output of the system’s localization in Cubicle 1. A total of 152 query frames were localized, with each sphere representing the position of one of the query frames. The green spheres represent acceptable positions, as they are within the cubicle where the data was collected. The red spheres present outliers, as these locations are not near where the images were captured.

For a clearer representation of the position, the location of all the query frames were averaged and the average position is shown in Figure 9 for each cubicle. Cubicle 1 and Cubicle 2 were closer to the location from where the database images were collected therefore their position is more accurate than Cubicle 3. This can also be seen by looking at the standard deviation of the x, y, and z position components presented in Table I. The standard deviation for the position in all three directions is smaller for Cubicle 1 indicating that the camera positions calculated by our system is much closer to each other. As we move to Cubicle 2 and Cubicle 3, the standard deviation values increase and the estimates for the position therefore contain more error. Throughout the experiments, it was noticed that more frames are matched and better matches are present if the query image is collected near where the database images were collected. However, the system still works in larger environments, as will be presented later. Additionally, Cubicle 3 was right next to a wall which contained very few features, and close to the windows where similar features were present. This is why the average position for Cubicle 3 is located further from the center of the cubicle, as compared to the other two.

![Figures 7 to 9](image1.png)

TABLE I: Standard deviation of query image positions

<table>
<thead>
<tr>
<th>Cubicle Number</th>
<th>Total Frames</th>
<th>Frames Localized</th>
<th>Standard Deviation (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>200</td>
<td>152</td>
<td>0.478825 0.345871 0.17133</td>
</tr>
<tr>
<td>2</td>
<td>150</td>
<td>99</td>
<td>0.619230 0.376961 0.468211</td>
</tr>
<tr>
<td>3</td>
<td>155</td>
<td>110</td>
<td>1.38270 0.794103 0.613596</td>
</tr>
</tbody>
</table>

The JDUC dataset, containing 6987 features in the database, was collected on the lower floor of the JDUC building at Queen’s University, and is approximately 30 m² in size. The area is a large open space that has couches in the middle and is surrounded by offices and a store. Most of the area is covered by a ceiling, which can be seen in the map in Figure 10. One of the corners is made up of stone, which does not contain many features. The query images used for localization were collected in three of the corners of the area. At each corner the camera was slowly moved 360° to get an image of the surroundings.

Figure 10 shows the average localization result obtained in the three corners of the JDUC dataset. The position of 103 query frames were found for corner 1, 120 frames were found for corner 2 and 47 frames were found for corner 3. Corner 3 was located furthest from where the database images were captured, therefore fewer query frames had enough feature matches to meet the required minimum inlier threshold.

The WLH dataset, containing 10574 features in the database, was collected in the lobby of Walter Light Hall at Queen’s University. The lobby was about 12 m² in size and was connected to two hallways, an auditorium, two computer labs, a classroom, and stairs leading to the entrance of the building. One entire wall of the lobby was made up of glass doors leading to the outside of the building, therefore this area did not contain many features. The map building was done from the center of the lobby.

Two sets of query images were collected for the WLH dataset. The first set was collected near the center of the lobby, close to where the images used for building the map were obtained. The second set was collected in one of the corners of the lobby, close to one of the auditorium doors. In
both locations the query images were obtained by moving the camera around 360°. The average position computed by the system for both locations is shown in Figure 11. Please note that Figure 11b is shown upside down in order to make the sphere visible, as it would otherwise be covered by the ceiling in our 3D map. Since position 1 was close to the location where the database images were collected, and also closer to feature-rich areas, a total of 125 query images were localized. On the other hand, position 2 was much further from where the database images were captured, therefore only 39 of the query frames contained enough inliers to be localized.

The Mine dataset, made up of 13584 features, was obtained in the NORCAT Underground Training Centre near Sudbury, Ontario. The area of the mine where the data was collected was dark and 1600 Lumens LED Worklights were used to illuminate the wall, one light per camera. The 3D map was created with two Kinects facing the wall at 45° and looking up at 45°. RGBDSLAM was run on the data obtained from both Kinects to make two separate portions of the wall and then the two maps were combined to look like a tunnel. Two Point Grey Blackfly cameras, facing the same direction as each Kinect, were used to collect the query images.

The proposed system was used to localize two cameras in the Mine dataset. The trajectory of the two cameras were averaged to obtain the vehicle trajectory, shown in Figure 12. A minimum of six inliers were required in this dataset in order to obtain a camera position. Since the sensors were being moved along the mine, the movement of the camera is represented using lines instead of spheres. Both trajectories remain within the wall of the mine, which shows that the localization process is reliable, as the cameras trajectory follows the path we expect. The total trajectory was only about 5 m long and the tunnel was about 4 m wide, with the wall about 1-3 m away from the camera.
IV. Conclusion

This paper presents a technique that can be used to localize within a previously mapped indoor environment using a single camera. At runtime the proposed technique requires only an inexpensive, calibrated monocular camera and the 2D images obtained from the camera are matched against a feature database to obtain the camera position. Our technique’s accuracy was tested against the positions obtained using RGBDSLAM, which we used as ground truth. We found the results of our technique to be very close to the ground truth, i.e. a few millimetres within an area of about 100 square metres. The system was also tested in four different environments with query images collected using two different cameras. The average position calculated by the system closely matches the location where the data was collected.

Future work includes testing the system to further quantify its performance and explore its limitations. These tests include trying different feature detectors, running the technique in more varied environments, using larger datasets, testing with different cameras and lenses for obtaining image inputs, and improving the accuracy of the technique. When working with large datasets, improvements to the database can be made to speed up the system. Lastly, the system will be modified to run on a mobile device.

Acknowledgment

The authors would like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC), Barrick Gold, and Peck Tech Consulting Ltd. for supporting this research, and the Northern Centre for Advanced Technology (NORCAT) for helping us collect the mine data at their site. We would also like to thank Cody Stewart for help with the data collection software, Mustafa Mohamad for feedback on the paper, and the anonymous reviewers for their helpful comments.

References


