Framework for Natural Landmark-based Robot Localization

Andrés Solís Montero, Hicham Sekkati
Jochen Lang and Robert Laganière
lEECS, University of Ottawa
Canada
Email: asolis|sekkati|jlang|laganier@eecs.uottawa.ca

Jeremy James
Cohort Systems Inc.
Nepan, Ontario
Canada, K2J 4P6
Email: jjames@cohortsys.com

***Abstract*—In this paper we present a framework for vision-based robot localization using natural planar landmarks. Specifically, we demonstrate our framework with planar targets using Fern classifiers that have been shown to be robust against illumination changes, perspective distortion, motion blur, and occlusions. We add stratified sampling in the image plane to increase robustness of the localization scheme in cluttered environments and on-line checking for false detection of targets to decrease false positives. We use all matching points to improve pose estimation and an off-line target evaluation strategy to improve *a priori* map building. We report experiments demonstrating the accuracy and speed of localization. Our experiments entail synthetic and real data. Our framework and our improvements are however more general and the Fern classifier could be replaced by other techniques.**

***Keywords*-robot localization; feature matching; Ferns; natural planar landmarks**

I. Introduction

Robot localization is of key importance in mobile robot navigation. It is the process by which the pose of the robot is determined from sensor data. It is well known that odometry and inertial navigation are not sufficient to maintain an accurate estimate of the robot's localization. As a consequence, many exteroceptive sensors including sonar, laser, GPS, and vision systems have been used for more accurate localization. There has been considerable research on developing various localization methods. Many early successful approaches use artificial landmarks, such as infrared light reflectors, ultrasonic beacons and visual patterns. Those methods provide a robust and stable solution for controlled environments, but they are not appropriate for large spaces because of the need of fiducial markers. By contrast, vision-based schemes using natural landmarks are more appropriate for unmodified environments and in general of low cost requiring only a single camera to acquire images. They consist in general of extracting some invariant features from the imaged scene and identifying the features within a map. This map can be built simultaneously while a robot navigates the environment such as in SLAM systems (Simultaneous Localization And Mapping) e.g., as in [1], or it can be learned off-line e.g., as in [2], [3]. In our application, we have access to maps but we annotate the map with natural landmark locations and hence, are concerned with localization of a robot based on natural landmarks.

We are using Ferns as feature descriptors and as a matching scheme. In this paper, we study in depth the performance of Ferns for our robot localization problem using planar targets. The choice of planar targets is based on the use of the Ferns matching scheme that consists, during the training phase, of generating virtually different views of the patches to be matched. While any 3D targets can be used, it requires an accurate 3D model to calculate the perspective projection into the image plane. In indoor environments planar targets abound and they can be easily measured with a ruler for metric localization. We add to the Fern classification a step whereby the quality of a planar target can be evaluated by simulation. This enables rejection of targets that are not expected to work well.

Another major concern in robot localization are changes in the environment that clutter the scene when the robot tries to localize a previously seen landmark. The Fern classifier handles partial occlusions well but it relies on a sufficient number of keypoints to be detected on the imaged target which may not be true in cluttered environments. We develop a simple stratified sampling technique to increase the probability that a feature on a landmark can be detected despite scene clutter. We demonstrate how the accuracy of localization can be increased by applying this sampling technique, and we compare the results with the common solution for pose estimation using the strongest keypoints.

We make the following contributions in this work: an experimental evaluation of the Fern classifier for robot localization, practical techniques to improve the performance of the scheme, and a framework for developing keypoints-based visual robot localization.

II. Related Work

Navigation and localization systems can be roughly divided in those providing the robot with models of the environment, with different degrees of detail, in (*Map-based navigation*), and those that perceive the environment as they navigate through it in (*Mapless navigation*). The method presented in this paper falls within the first category. We

978-0-7695-4683-4/12 $26.00 © 2012 IEEE
DOI 10.1109/CRV.2012.25

IEEE computer society

refer the reader to surveys on robot navigation and localization systems found in [4], [5], [6] for details. Vision-based localization systems have made significant progress among other mobile robot localization techniques, particularly, for indoor applications. In many cases the localization problem is two-dimensional, i.e., a robot position on a plane together with a heading [7], [8], [2], [9], [10]. Other applications require vision-based methods that provide six degree-of-freedom camera pose.

The localization problem is related to the problem of structure-from-motion (SFM) that consist of inferring the environment structure and camera motion under the assumption of *small* baseline motion. SFM-based localization for SLAM has been addressed, e.g., in [11], [12], [13], [14], [15], [16]. In general, SFM-based localization does not consider the kidnapping problem which is that the robot may be picked up and placed at a different position.

Se et al. [17], [1] present a trinocular SLAM system using SIFT descriptors as landmarks. The landmarks are matched between the three views to reconstruct the 3D points. The system also uses odometry-based initialization to estimate the ego-motion of the camera. Wuest et al. [18] use the FAST features detector [19] and model the probability distribution of each feature. Each feature is then tracked successfully by means of a finite set of Gaussian mixtures. An extension of this method for a large number of features seems difficult because of the use of a Gaussian kernel. Also, in their visibility modeling only camera translation was considered but orientation is important for robot localization. Closest to our localization is the system presented by Alcantarilla et al. [20] using a non-parametric learning framework to predict for each feature its visibility with respect to the varying camera poses.

III. Keypoints Tracking by Ferns

We will review Fern descriptors here detailing the background for our evaluation of Ferns during localization. Tracking with Ferns descriptors proceeds by recognizing patches of an imaged object from different perspective views using a classification scheme [21]. Ferns rely on an off-line training phase during which multiple views of the image patches are used to train a naive Bayes classifier. Recognition is based on neighborhood intensity comparisons. During training features of the planar patches around a keypoint are synthesized by warping them using random homographies. A recent study has shown the robustness of the Ferns as a feature tracker among other comparable descriptors such as (SURF, SIFT, Randomized Trees) [22]. The Fern descriptor is geared towards tracking in real-time after a training stage. It copes well with dynamic lighting in the scene and motion blur due to camera motion. It has also shown more robustness with respect to scale and perspective distortions compared to other descriptors. Ozuysal et al. [21] have shown the effectiveness of Ferns descriptors in a SLAM system, and they have shown their capability to recover from complete failure. However, they did not address performance during pose estimation which is required by any localization system.

A. Ferns: Basic Model

Keypoints are detected using an extrema of Laplacian operator and binary features $f_{k,1}, f_{k,2}, \cdots, f_{k,N}$ associated with each keypoint p_k. The binary features are computed by comparing image intensities in the neighborhood of a keypoint, that is

$$f_{k,j} = \begin{cases} 1 & \text{if } I(p_l) < I(p_m) \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where $(p_l, p_m) \in \mathcal{N}_k$, with $\mathcal{N}_k$ is the neighborhood of the keypoint p_k. Given the set of features $(f_{k,j}, j = 1, \cdots, N)$, the problem of recognizing a target then becomes identifying the class c_i to which a keypoint p_k belongs by maximizing the following joint conditional probability,

$$\underset{c_i}{\operatorname{argmax}}\, P(f_{k,j}, j = 1, \cdots, N \mid C = c_i) \tag{2}$$

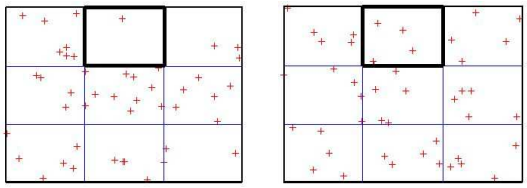
With a high number of features N, the full joint distribution is too large to be fully represented. Instead, the features are grouped into M groups of size S, referred to by the Ferns. By assuming independence between features, the joint probability can be approximated by the product of the individual conditional probabilities, that is

$$P(f_{k,j}, j = 1, \cdots, N \mid C = c_i) \approx \prod_{k=1}^{M} P(F_{k,s} \mid C = c_i) \tag{3}$$

Where $F_{k,s} = \{f_{\sigma(k,1)}, \ldots, f_{\sigma(k,S)}\}$ represents the k^{th} fern with a random permutation function $\sigma(k, s)$ with range $1, \ldots, N$. Independence between features is crucial to successfully classifying keypoints. Lack of independence among sample points can bias estimates and measures of precision. This assumption holds for the case when patches of different keypoints do not overlap. To reduce the chance of such overlaps and to increase the probability that a keypoint lays on an interested area, we propose to use stratified sampling reducing the variance of estimates.

B. Keypoints Selection by Stratified Sampling

Stratified sampling is a technique of sampling that maintains some characteristics of the data set within its sampled subsets. It is achieved by firstly partitioning the data set into a number of mutually exclusive subsets of cases, each of which is representative of some aspect of the real-world process involved. In general, sampling from the population is then achieved by randomly sampling from the various subsets to achieve representative proportions. A tutorial example of random sampling is shown in Figure 1 compared with



(a) (b)

Figure 1. Illustration of stratified sampling in (b) vs. random sampling in (a). Both illustrations show the same number of targets.

stratified sampling using the same total number of samples but in nine strata. Instead of random sampling, the feature detector is applied in the subsets to select the strongest features in the same way it is applied in the whole image. A more detailed performance assessment comparing strongest features with stratified sampling is given in Section IV-A.

IV. Experimental Evaluation

We present our framework for the experimental evaluation of localization by planar natural landmarks in five subsections summarizing the performance for pose estimation and camera localization. In each subsection, we give the specific experimental framework and report results obtained for the different experiments. The five subsections are: Stratified Sampling versus Most Stable Keypoints selection (§IV-A), Time Performance and Target Detection rate (§IV-B), Pose Estimation Using Corner Points only versus use of all matched features (§IV-C), Pose Estimation Using Real Data (§IV-D) and finally Target Evaluation (§IV-E). All experiments are executed on a MacBook Pro laptop (2.3Ghz Intel Core i5 with 4GB of RAM) with an implementation utilizing OpenCV 2.3.1. The camera resolution is 1280×720 in all experiments.

A. Stratified Sampling vs. Most Stable Keypoints

One of the most important steps in planar object detection is keypoint selection. The selected keypoints are matched against the database of the training set in order to recognize the target in the video frame. Without a sufficient number of good quality keypoints detected on the target, the process of target detection and consequently the camera pose estimation will fail. Commonly, the strongest feature keypoints are selected in an image for processing. Selecting the strongest feature points is motivated by the fact that strong feature points are more likely to be stable under viewpoint changes. However, the strength of feature points in themselves is not always a sufficient criteria for success in target recognition. The feature points also have to be part of the target rather than part of the background scene. We study therefore a stratified sampling strategy where we tessellate the image plane and select strong features in each strata.

We conducted experiments in order to compare the two strategies: a pre-selected number of the strongest feature points and the above stratified sampling strategy. We positioned a target in a cluttered environment and compare the detection results using different numbers of keypoints with the two approaches. The strongest feature points strategy is configured to select a maximum of 300, 700 and 1000 keypoints, respectively. Our stratified sampling approach uses a 7×7 grid where the most relevant points in each cell are selected. For comparison, the total number of keypoints in the image is kept the same for both approaches.

In our experiments, the stratified sampling solution was more robust for target detection , detecting the target (here, a poster with a world map) for all the keypoints configurations in our experiment. This is because the sampling strategy inspects the whole image in search for the most representative points of each cell. When selecting the strongest points, we only were able to detect the target with 1000 keypoints while with a maximum of 700 and 300 no keypoint was detected in the target region. In cluttered scenes, the target is not necessarily the object giving rise to the most keypoints in the image. The surrounding objects may have more representative features and may therefore attract all of the keypoints. In the current test image, we needed to increment the number of keypoints to detect some in the targets area (see Figure 2). However, even increasing the number of keypoints is unlikely to work consistently because in dynamic scenes new clutter can appear in the image and may potentially attract more keypoints than the preset maximum. The stratified sampling solution is more stable in this scenario because adding objects to one region wont affect the keypoint selection in other regions. Figure 3 shows that the stratified strategy is able to detect the target despite the insertion of more clutter that partially occludes the target.

B. Time Performance and Target Detection

While the above example demonstrates the improvements achievable with stratified sampling, we are interested in the impact of the strategy on robot localization. Successful localization depends on the target detection success rate, the occurrence of false positives and the execution time. Execution time in a real-time application such as robot localization will have a direct impact on the utility of the method. Therefore, we evaluate the impact of using the two previous strategies in terms of processing time for each of the steps in detecting a planar target and computing the pose. We include target detection rate as a quality benchmark. For this experiment, we used a recorded video of 1627 frames containing the target seen from a moving camera directed towards the world map target on a random path. The camera has been calibrated with the aid of a printed checkerboard and the calibration procedure available in OpenCV [23]. The timed steps are: remove camera distortion (CD), image preprocessing (PRE), feature extraction (FE), feature matching (FM), and pose estimation (PE). In the experiment we use

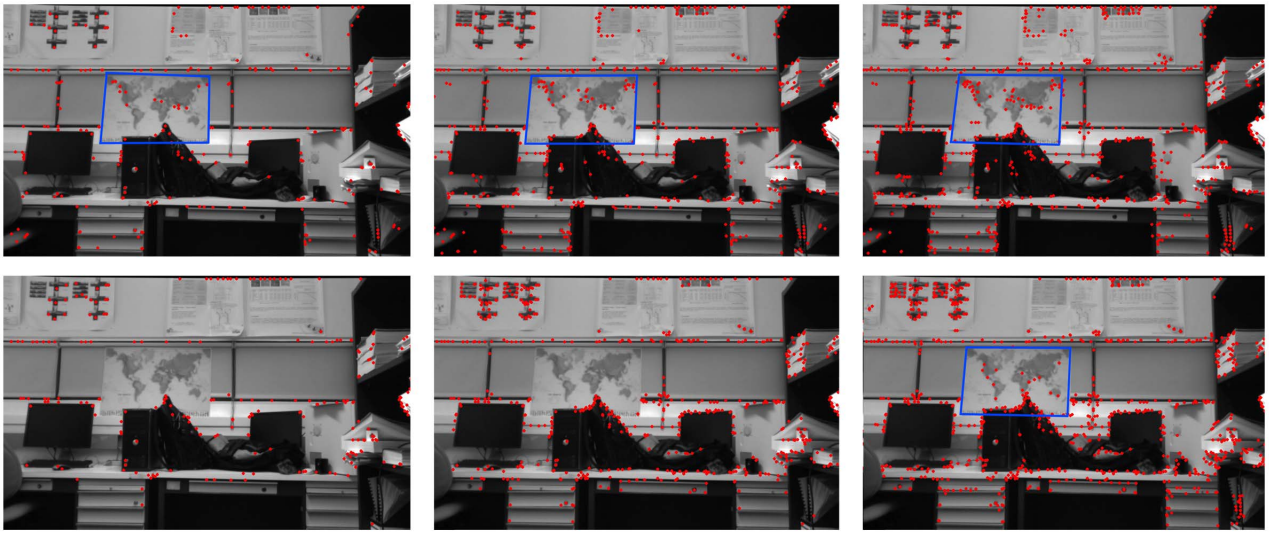


Figure 2. Top row shows the stratified sampling approach with a total of 300, 700 and 1000 keypoints detected. In the bottom row the top 300, 700 and 1000 keypoints in the image were detected.

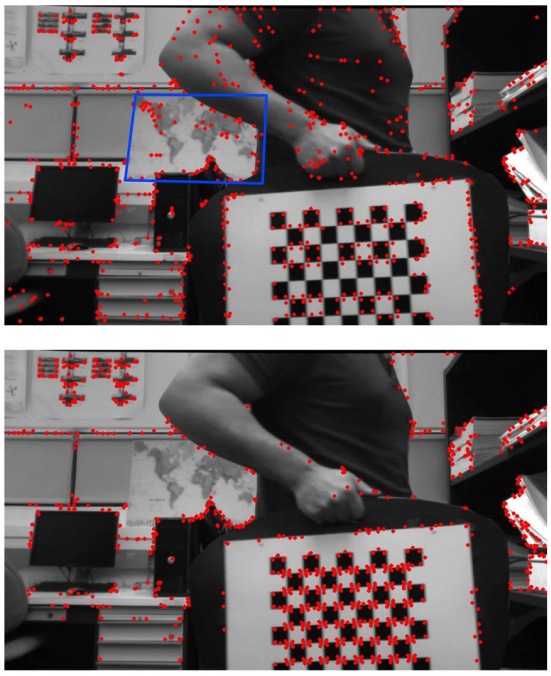


Figure 3. Stratified sampling (top) and top most stable keypoints (bottom) with a total of 1000 keypoints detected in both.

a cap of a total of 1000 keypoints for both strategies.

The FE and FM steps dominate the processing time of the algorithm using about 80% of the total execution time of the five steps (see Figure 4). In general, the total execution time is about the same for both approaches, in particular, stratified sampling does not increase the overall execution time of the localization. The execution time of FE depends on the method of feature extraction, i.e., the Laplacian of Gaussian as in the original Fern approach [21]. The execution time of the FM step is affected by the number of features to match against in the database and the extraction of their descriptors. The time complexity of classifying a keypoint is $O(M)$, where M is the number of classes in the database (i.e. we use 100 classes). On the other hand, the detection rate improves greatly with the stratified sampling approach while the false detection reduces simultaneously (see Figure 5).

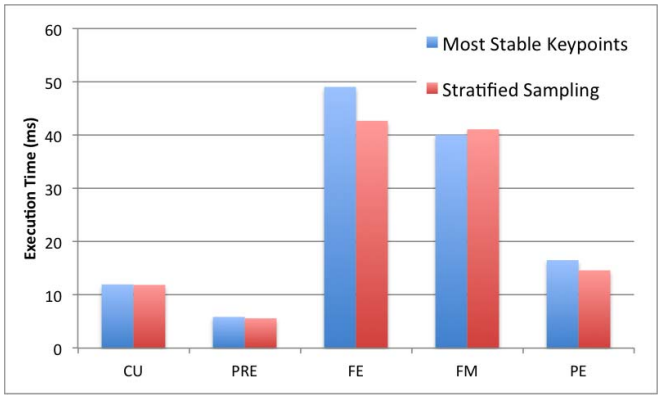


Figure 4. Execution time comparison (in milliseconds) between Most Stable Keypoints and Stratified Sampling. From left to right: Camera Distortion, Pre-processing, Feature Extraction, Feature Matching and Pose Estimation.

We propose to include an on-line check for detection of false positives after matching keypoints. Using prior-knowledge of the fact that we are looking for a planar object

imposes constraints on the image of the keypoints belonging to the target. After removal of camera distortion, all the inliers must be imaged inside the area of the estimated target and this image must be planar. We check if the convex hull of the features for the estimated target are inside the corners of the estimated target pose projected into the image. A false detection is flagged when the obtained homography from the inliers computed using RANSAC [24] does not project to a planar target. The average processing time of the planarity check is about 0.015 milliseconds for each video frame. It has therefore negligible impact on the total execution time for pose estimation.

Figure 5. Detection rate (false detection,true detection) for stratified sampling and strongest keypoints.

C. Corner Points vs. Matched Points

In the next experiment we compare the impact of computing the pose, i.e., solving the PnP (perspective n-point) problem using two slightly different approaches.

For this experiment, we render images by projecting the target into the image plane from different angles and depths. We use the camera's intrinsic parameters obtained in the calibration of the real camera through the camera module of our implementation. We use the synthetic images to test for planar target detection accuracy and the subsequent pose estimation using the method by Schweighofer and Pinz [25]. In the synthetic images we know the exact projection of the target given the orientation and position of the camera. We generate 1000 images at three different distances from the target with orientation angles between 30 and 150 degrees and compute the projection , translation and rotation error. When generating the image using the camera information, we include random background and foreground noise, and random Gaussian Blur to simulate the real capture process (see Figure 6). The background is filled with uniform noise with parameters from 0 to 255 grayscale values. The foreground noise added is additive white Gaussian noise with zero mean and variance of 10 grayscale values. The Gaussian Blur uses uniformly distributed random kernel size (ksize), odd values from 3 to 9 and standard deviation $\sigma = 0.3 * (ksize/2 - 1) + 0.8$.

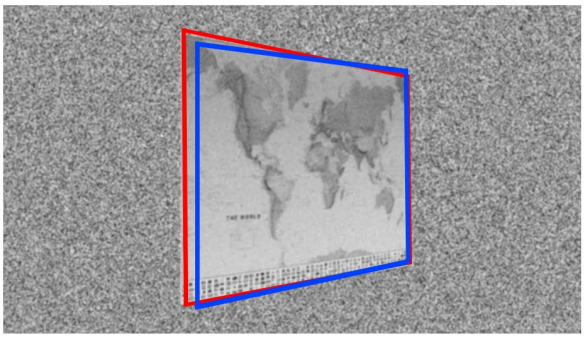


Figure 6. Example of generated target view by rotation around the image y-axis. Random noise and blur are added to simulate the real capture process. Red boundary is the ideal target projection, blue boundary an estimate.

The first approach, named here *Corner Points* (CP) uses only four points (i.e. corners of the target) to estimate the pose. The corners of the targets are acquired by computing the homography from the training image and the matching points in the video frame. The homography is obtained using RANSAC [24]. Transforming the four corners of the targets training image we can get the position of the target in the video frame. The pose estimation is finally computed by the 3D correspondence of the four corners of the target and the translated corners positions (see Figure 7).

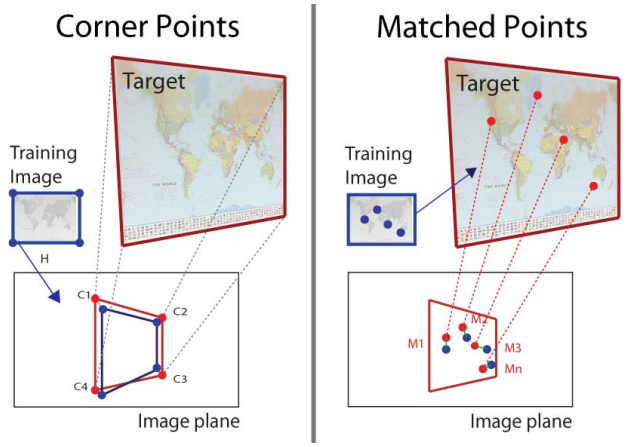


Figure 7. Outline of the Corner Points and Matched Point approaches. CP uses the matched points to obtain the homography and then translate the training image to the video frame. PnP is solved only with the four corners (i.e. $C_1 ... C_4$) of the target. MP translates the matching points in the training image to its 3D coordinates and solve PnP using their correspondence (i.e. $M_1 ... M_n$).

For the second approach named here Matched Points (MP), we skip the homography computation and the projection of the corners. Instead we use all the matching points of the target between its training image and the analyzed video

Table I
COMPARISON BETWEEN MP AND CP AT DIFFERENT TARGET'S DEPTHS

Target Depth	200 cm		300 cm	
	MP	CP	MP	CP
True Detection (%)	92	90	86.1	82.9
False Detection (%)	3.8	5.8	3	6.2
Reproj. error (pixel)	7.2	11.4	7.46	16.3
Translation error (cm)	1.31	1.93	2.32	4.2
Rotation error (angle)	1.32	1.44	2.11	3.82
Location error (cm)	5.19	6.02	13.736	21.138

frame to compute the extrinsic camera parameters. The 3D position of the matched points are obtained from the four 3D coordinates of the targets corners and the fact that it is a planar object and we know the image coordinates of the match in the training image (see Figure 7 right).

The projection error for CP is computed as the average distance of each corner between our approximation and the real projection (see Equation 4). For the MP the projection error is computed as the average distance of each matched point real projection and its detected position by the Fern classifier (see Equation 5).

$$CP_{error} = \frac{\sum_{i=1}^{4} |C_i - \hat{C}_i|)}{4} \tag{4}$$

$$MP_{error} = \frac{\sum_{i=1}^{n} |M_i - \hat{M}_i|)}{n} \tag{5}$$

The rotation error is the angle in degree between the normal of the projected and estimated target. Translation and camera location error are defined as the norm between the real and the estimated value.

MP outperformed CP in all the classifications, increasing the detection rate and reducing false positives, re-projection, translation, rotation and camera location errors (see Table I). MP is a more stable approach than only using the four target's corners directly from the homography estimation.

D. Pose Estimation Using Real Data

For this experiment we placed four targets and positioned the camera pointing towards their center at different ranges and view angles. Accurate ground truth for the pose of the camera is not available to us, therefore we evaluate the accuracy in terms of quantities that can be precisely measured. We place multiple targets at precisely known relative position and orientation from each other. Now, if we estimate the pose of the camera between two or more targets, we can calculate the difference between those poses. In a common coordinate system, the difference of the poses will be exactly the spatial transformation between the targets (see Figure 8). Suppose (R_a, T_a) and (R_b, T_b) are the pose of the

Figure 8. Detection output example of our implementation using the four target setup.

camera with respect to a pair of targets (a, b), respectively. Then the relative pose between the two targets satisfies

$$\hat{T} = T_a - RT_b \text{ and } R = R_a R_b^{-1}. \tag{6}$$

The errors of the relative translation and rotation are defined as the Euclidean distance between the real translation T and its estimation $\hat{T}$ and the angle between the two estimated normal vectors n_a and n_b of the planer targets using the dot product (See Equation 7).

$$e_T = |T - \hat{T}| \text{ and } e_R = arccos(n_a \cdot n_b). \tag{7}$$

The pose estimation is evaluated between four targets having different textures. Figure 9 shows those targets and their relative translations. The rotation between each pair of targets is the identity matrix. We placed the camera around the center of the four targets in nine equally spaced directions from $-60°$ to $60°$ degrees, and at different distances with respect to the center of the targets from 0.90 m to 2.10 m. We obtained an average translation error of 2.74 cm over the whole dataset, and a rotation error of $2.6251°$. These values are in line with our experiments with synthetic images in Section IV-C. Next, we will discuss our strategies to evaluate the feasibility of a target for robot localization based on a *a priori* test.

E. Target Evaluation

Pose estimation is highly dependent on the target detection process. The number of correct matches between the extracted keypoints and the training images of the target improve the accuracy. We use natural targets to solve the localization problem; therefore, the selection of a target with stable repeatable keypoints from different views and depths will aid the performance of the localization. We therefore like to evaluate the feasibility of a target before adding it

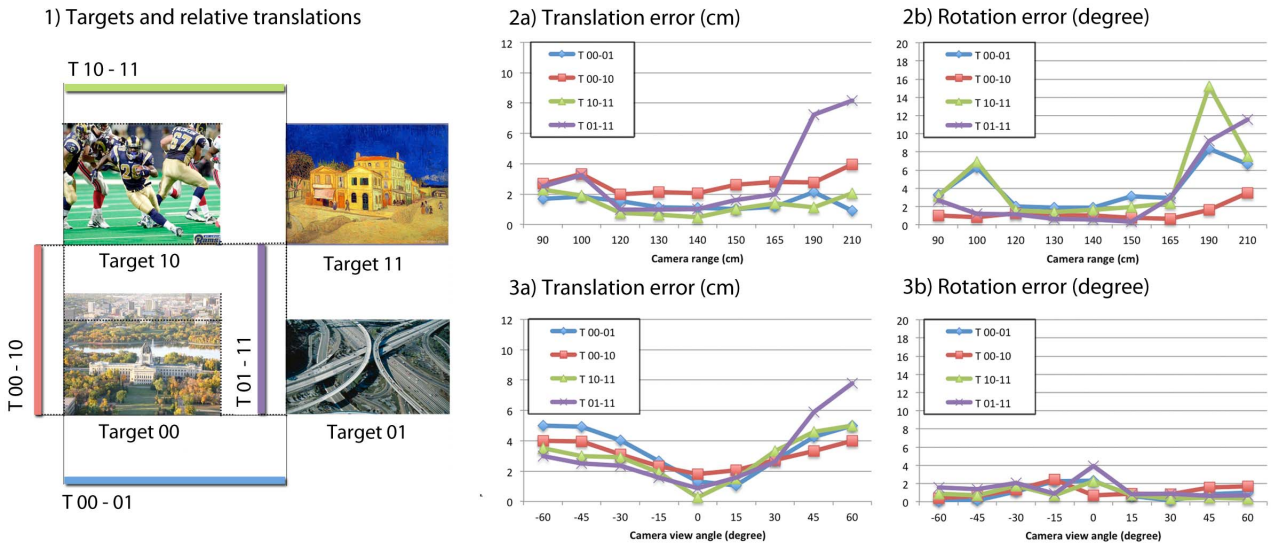


Figure 9. Image 1) shows the four targets $Target - \{00, 01, 10, 11\}$ and their positions. The translation $T00 - 01$ is the relative translation from $Target$ 00 to $Target$ 01. The same scheme applies for the other three relative translations shown in (1). The target dimensions are 27x20 cm, except for $Target - \{01\}$ which is 27x18 cm. Images 2a) and 2b) are the translation and rotation errors computed while moving the camera from 0.90 m to 2.10 m ranges. Images 3a) and 3b) are the translation and rotation errors with camera view angles from $-60°$ to $60°$ at 150cm range.

to the map. To evaluate a target we use the results of the target's performance measured using synthetic data.

The target evaluation process consists of two steps. First, we train the system using one fronto-parallel image of the target. The feature database is created using the Fern classifier of the most stable feature points of the target image. Once the learning process has finished, we use the information of the camera to project the target from different cameras views by setting the distance from the camera to the target and rotating around the targets center. The background of the generated image is filled with random noise and blur. Then, for each generated frame we evaluate the target according to detection rate, re-projection error, pose estimation and camera location, just as in our synthetic data simulation in Section IV-C. These values serve as a prediction of the *quality* of the target, i.e., an indication of the accuracy to expect when the particular target would be employed for robot localization.

As an illustrative example, we take the four targets from the previous section and run the synthetic target evaluation on them. The results are highly related to the pose computed from real data (see Section IV-D). The target evaluation strategy confirms the instability of $Target$ 01 and $Target$ 11 compared to the other two (See Figure 10). The pose estimation inaccuracy of these two targets is therefore the likely cause of the translation and rotation errors in the real data experiment (i.e. translations T00-01, T01-11, and T10-11 are noisy if the camera distance is beyond 1.65 m). This gives us confidence to use the above evaluation process as a target selection strategy before adding targets to the system's database.

V. Conclusion

In this paper we presented a framework and experimental evaluation of the Fern classifier for robot localization using natural landmarks along with practical improvements to increase the performance of the scheme. These improvements are stratified sampling, on-line checking of false target detection, use of all matching points to improve pose estimation and an off-line target evaluation strategy. Our stratified sampling technique is able to double the true rate of detection while reducing false target detection without affecting execution time. On-line checking of the detection of a planar target helped reducing false positives. The use of all matching points vs. simply the corners of a rectangular target improved the accuracy of localization by up to 50%. Finally, our off-line target evaluation predicts the quality of the target and hence, the precision to expect when the particular target would be employed as part of a map. The experimental results illustrate that natural targets and Fern classifiers with our additions are a feasible solution for visual robot localization. All our extensions of the Ferns localization are not specific to Ferns and could be integrated with other robot localization schemes. In future work we would like to evaluate different feature matching for robot localization in our framework.

Acknowledgement

We gratefully acknowledge the financial support from the Natural Sciences and Engineering Research Council of Canada (NSERC) and from Cohort Systems Inc.

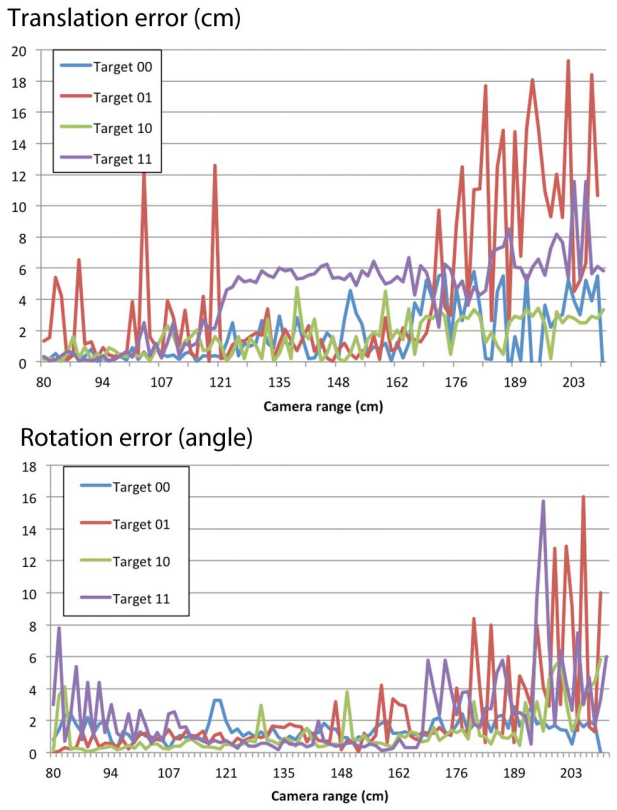


Figure 10. Target evaluation of the four targets. Translation and rotation error show the instability of $Target$ 01 and $Target$ 11 compared with the other two. In the experiment, we simulate the movement of the camera in front of the target from 0.8 m to 2.1 m. The average translation error for the complete set is 2.85 cm and 1.45° of average rotation error.

REFERENCES

[1] S. Se, D. G. Lowe, and J. J. Little, "Vision-based global localization and mapping for mobile robots," *IEEE Trans. on Robotics*, vol. 21, no. 3, pp. 364–375, 2005.

[2] R. Sim and G. Dudek, "Learning visual landmarks for pose estimation," in *ICRA*, 1999, pp. 1972–1978.

[3] B. Charmette, E. Royer, and F. Chausse, "Matching planar features for robot localization," in *I. Symp. on Visual Computing*, vol. 5875. Springer, 2009, pp. 201–210.

[4] J. Borenstein, H. R. Everett, L. Feng, and D. Wehe, "Mobile root positioning: Sensors and techniques," *J. of Robotic Systems*, vol. 14, no. 4, pp. 231–249, 1997.

[5] G. N. DeSouza and A. C. Kak, "Vision for mobile robot navigation: A survey," *IEEE PAMI*, vol. 24, no. 2, pp. 237–267, 2002.

[6] F. Bonin-Font, A. Ortiz, and G. Oliver, "Visual navigation for mobile robots: A survey," *J. of Intelligent and Robotic Systems*, vol. 53, no. 3, pp. 263–296, 2008.

[7] E. Krotkov, "Mobile robot localization using A single image," in *ICRA*, 1989, pp. 978–983.

[8] S. Atiya and G. D. Hager, "Real-time vision-based robot localization," *IEEE TRA*, vol. 9, pp. 785–800, 1993.

[9] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," in *ICRA*, 1999, pp. 1322–1328.

[10] S. Thrun, "Probabilistic algorithms in robotics," *AI Magazine*, vol. 21, no. 4, pp. 93–109, 2000.

[11] A. J. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *ICCV*, 2003, pp. 1403–1410.

[12] M. L. Pupilli and A. D. Calway, "Real-time camera tracking using a particle filter," in *BMVC*, 2005, pp. xx–yy.

[13] T. W. Drummond and E. D. Eade, "Edge landmarks in monocular SLAM," in *BMVC*, 2006, p. I:7.

[14] D. Chekhlov, M. L. Pupilli, W. W. M. Cuevas, and A. D. Calway, "Real-time and robust monocular SLAM using predictive multi-resolution descriptors," in *Advances in Visual Computing*, 2006, pp. II: 276–285.

[15] B. Williams, G. Klein, and I. D. Reid, "Real-time SLAM relocalisation," in *ICCV*, 2007, pp. 1–8.

[16] G. Klein and D. W. Murray, "Improving the agility of keyframe-based SLAM," in *ECCV*, 2008, pp. II: 802–815.

[17] S. Se, D. G. Lowe, and J. J. Little, "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks," *IJRR*, vol. 21, no. 8, pp. 735–760, 2002.

[18] H. Wuest, A. Pagani, and D. Stricker, "Feature management for efficient camera tracking," in *ACCV*, 2007, pp. I: 769–778.

[19] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *ECCV*, vol. 1, 2006, pp. 430–443.

[20] P. F. Alcantarilla, S. M. Oh, G. L. Mariottini, L. M. Bergasa, and F. Dellaert, "Learning visibility of landmarks for vision-based localization," in *ICRA*. IEEE, 2010, pp. 4881–4888.

[21] M. Özuysal, M. Calonder, V. Lepetit, and P. Fua, "Fast keypoint recognition using random ferns," *IEEE PAMI*, vol. 32, no. 3, pp. 448–461, 2010.

[22] S. Gauglitz, T. Höllerer, and M. Turk, "Evaluation of interest point detectors and feature descriptors for visual tracking," *IJCV*, vol. 94, no. 3, pp. 335–360, 2011.

[23] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

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

[25] G. Schweighofer and A. Pinz, "Robust pose estimation from a planar target," *IEEE PAMI*, vol. 28, no. 12, pp. 2024–2030, 2006.