SIFT and SURF Performance Evaluation for Mobile Robot-Monocular Visual Odometry

Houssem Eddine Benseddik, Oualid Djekoune, and Mahmoud Belhocine
Computer Integrated Manufacturing and Robotics Division
Center for Development of Advanced Technologies, Algiers, Algeria
Email: {hbenseddik, odjekoune, mbelhocine}@cdta.dz

***Abstract*—Visual odometry is the process of estimating the motion of mobile through the camera attached to it, by matching point features between pairs of consecutive image frames. For mobile robots, a reliable method for comparing images can constitute a key component for localization and motion estimation tasks. In this paper, we study and compare the SIFT and SURF detector/ descriptor in terms of accurate motion determination and runtime efficiency in context the mobile robot-monocular visual odometry. We evaluate the performance of these detectors/ descriptors from the repeatability, recall, precision and cost of computation. To estimate the relative pose of camera from outlier-contaminated feature correspondences, the essential matrix and inlier set is estimated using RANSAC. Experimental results demonstrate that SURF, outperform the SIFT, in both accuracy and speed.**

***Index Terms*—SIFT, SURF, essential matrix, RANSAC, visual odmetry**

I. Introduction

In the last decade, visual odometry has emerged as a novel and promising solution to the problem of robot localization in uncharted environments. The key idea of visual odometry is that of estimating the robot motion by tracking visually distinctive features in subsequent images acquired by an on-board camera [1]. Ego-motion estimation is an important prerequisite in robotics applications. Many higher level tasks like obstacle detection, collision avoidance or autonomous navigation rely on an accurate localization of the robot. All of these applications make use of the relative pose of the current camera with respect to the previous camera frame or a static world reference frame. Usually, this localization task is performed using GPS or wheel speed sensors.

In recent years, camera systems became cheaper and the performance of computing hardware increased dramatically. In addition, video sensors are relatively inexpensive and easy to integrate in mobile platforms. Hence, high resolution images can be processed in real-time on standard hardware. It has been proven, that the information given by a camera system is sufficient to estimate the motion of a moving camera in a static environment, called visual odometry [1]. These properties make visual sensors especially useful for navigation on rough terrain [2], such as in reconnaissance, planetary exploration, safety and rescue applications, as well as in urban environments [3], [4]. Since its early appearance, visual odometry has been based on three main stages: feature detection, feature tracking, and motion estimation. However, several particular implementations have been proposed in literature [5], [6], which mainly differ depending on the type of video sensors used, i.e. monocular, stereo or omnidirectional cameras, on the feature tracking method, and on the transformation adopted for estimating the camera motion.

A variety of feature detection algorithms have been proposed in the literature to compute reliable descriptors for image matching [7]-[11]. SIFT [10] and SURF [11] detectors and descriptors are the most promising due to good performance and have now been used in many applications. For visual odometry as a real-time video system, accuracy of feature localization and computation cost are crucial. Different from matching image applications with large viewpoint changes such as panorama stitching, object recognition and image retrieval, visual odometry is a video sequence matching between the successive frames. When the latter produces a number of false matches that significantly affect localization accuracy. This is mainly due to the fact that many features from an image may have no match in the preceding image.

The essential matrix estimation is one of the stages of Visual odometry: this is where a robot's motion is computed by calculating the trajectory of an attached camera, this matrix encoding the relative orientation and translation direction between the two views, and it is used to estimate the relative position from features matched between two images ('feature correspondences'). Normally some features will be incorrectly matched, so a robust estimation to these outliers must be used.

RANSAC (RANdom SAmple Consensus) [12] is a commonly used approach to achieve accurate estimates also in presence of large fractions of outliers. The use of RANSAC allows for outliers rejection in 2D images corresponding to real traffic scenes, providing a method for carrying out visual odometry onboard a robot. One the

Manuscript received January 4, 2013; revised May 5, 2014.

doi: 10.12720/joig.2.1.70-76

application is for simultaneously finding the fundamental matrix (or essential matrix in the case of calibrated cameras) relating correspondences between two images, and to identify and remove bad correspondences [13].

In this paper, we offer a substantive evaluation of SIFT and SURF to find the most appropriate detector and descriptor to estimate the accurate motion in visual odometry. We have selected the two popular detectors and descriptors which have previously shown a good performance in visual odometry.

The following sections are organized as follows: Section II, we briefly discuss the working mechanism of SIFT and SURF followed by discussion of matching algorithm. After, we present the essential matrix, how to estimate the relative pose of two cameras from this matrix and robust motion estimation by RANSAC. In section III, we thoroughly compare matching performance of the two detectors and descriptors and present our evaluation criterions. Finally, we conclude in Section IV.

II. Background

A. Features Description and Matching

One of the most important aspects of visual odometry is features detection and matching. Two well-known approaches to detect salient regions in images have been published: Scale Invariant Feature Transform (SIFT), Lowe [10], and Speeded Up Robust Features (SURF), Bay et al [11]. Both approaches do not only detect interest points or so called features, but also propose a method of creating an invariant descriptor. This descriptor can be used to (more or less) uniquely identify the found interest points and match them even under a variety of disturbing conditions like scale changes, rotation, changes in illumination or viewpoints or image noise. Exactly this invariancy is most important to applications in mobile robotics, where stable and repeatable visual features serve as landmarks for visual odometry and SLAM.

The Scale Invariant Feature Transform (SIFT): Lowe proposed a SIFT detector/descriptor [10], is a local feature extraction method invariant to image translation, scaling, rotation, and partially invariant to illumination changes and affine 3D projection. The extraction of SIFT features relies on the following stages:

- Creation of scale-space: The scale-space is created by repeatedly smoothing the original image with a Gaussian kernel.
- Detection of scale-space extrema (interest point detection): This is done to find peaks in the scale space of image (pixel) positions p = [x, y], and the scales σ. This is done by searching the (x,y, σ) space for extrema, which are filtered using stability criteria (step 4).
- Accurate interest point localization: In the previous step, the interest points were detected in a discrete space. This step determines the location of interest points with sub-pixel and sub-scale accuracy.
- Rejection of weak interest points: All interest points that have low contrast and are lying on an edge are removed.
- Orientation assignment: To obtain rotational invariance, each interest point is assigned an orientation determined from the image gradients of the surrounding patch. The size of the patch is determined by the selected scale.

The SIFT descriptor is a 3D histogram of gradient location and orientation. The magnitudes are weighted by a Gaussian window with sigma equal to one half the width of the descriptor window. These samples are then accumulated into orientation histograms (with eight bins) summarizing the contents over 4x4 sub-regions. The feature vector contains the values of all orientation histograms entries. With a descriptor window size of 16x16 samples leading to16 sub-regions the resulting feature vector has 16x8 = 128 elements. A calculation of descriptor histogram: Given the position, scale and orientation of each interest point, a patch is selected where magnitude and orientation of gradient is used to create a representation which allows, to some extent, affine and illumination changes.

Speeded Up Robust Features (SURF): The SURF detector-descriptor is proposed by Bay *et al.* [11]. Like SIFT, the SURF approach describes a keypoint detector and descriptor. This section gives all details on the following step of SURF algorithm structure:

- Computation of the integral image of the input images.
- Computation of the Box Hessian operator at several scales and sample rates using box filters;
- Selection of maxima responses of the determinant of the Box Hessian matrix in box space
- Refinement of the corresponding interest point location by quadratic interpolation;
- Storage of the interest point with its contrast sign.

The SURF descriptor encodes the distribution of pixel intensities in the neighborhood of the detected feature at the corresponding scale. To extract the SURF-Descriptor, the first step is to construct a square window of size 20σ, (σ is scale) around the interest point oriented along the dominant direction. The window is divided into 4x4 regular sub-regions. Then for each sub-region the values of $\sum d_x$, $\sum d_y$, $\sum|d_x|$, $\sum|d_y|$ are computed, where $\sum d_x$ and $\sum d_y$ refer to the Haar wavelet responses in horizontal and vertical directions in relation to the dominant orientation. This leads to an overall vector of length 4x4x4=64, which corresponds to the scaled and oriented neighborhood of the interest point.

Features Matching: After detecting the features (key-points), we must match them, i.e., determine which features come from corresponding locations in different images. The described descriptors constitute the features used for matching images. Consider two images, I_a for frame a, and I_b for frame b. For both images, local features are extracted (using one of the methods described

above), which results in two sets of features, F_a and F_b. Each feature F = [x, y], H comprises the pixel position [x, y] and a histogram H containing the SIFT or SURF descriptor. The similarity measure $S_{a,b}$ is based on the number of features that match between, F_a and F_b. The feature matching algorithm calculates the Euclidean distance between each feature in image I_a and all the features in image I_b. A potential match is found if the smallest distance is smaller than 60% of the second smallest distance.

The matching strategy was to find the descriptor from the initial image that had the smallest Euclidean distance to a given descriptor in one of the secondary images. It guarantees that interest points match substantially better compared to the other feature pairs. In addition, no feature is allowed to be matched against more than one other feature. If a feature has more than one candidate match, the match with the lowest Euclidean distance among the candidate matches is selected. Note that the number of matched features will depend on the order that the features are matched, that is, if each feature in I_b is instead matched with all features in I_a the number of matches may differs. This can be avoided if the matching is done in both ways, where a match is only considered valid if the match occurs twice. The feature matching step results in a set of matched feature pairs $P_{a,b}$, with a total number of $M_{a,b}$.

B. Monocular Visual Odometry

Given that a set of features has been tracked successfully from the previous set of frames, it is now possible to estimate the new location of the camera rig. By five corresponding points, it's possible to recover the relative positions of the points and cameras, up to a scale. This is the minimum number of points needed for estimating the relative camera motion from a calibrated camera and it is called five-point algorithm. Using the five-point algorithm with five correspondences, one can obtain the essential matrices [1]. For each essential matrix four combinations of possible relative rotation R and translation T of the camera can be easily extracted. In order to determine which combination corresponds to the true relative movement, the constraint that the scene points should be in front of the camera for both the two views is imposed.

In this work, we assume that the camera used in the visual odometry is fully calibrated, i.e., intrinsic matrix K is given.

Motion estimation by Essential matrix: The essential matrix, E, is a 3×3 matrix encoding the rotation and translation direction between two views. If the rotation is expressed as a matrix, R, and the translation as a vector, t, then E is defined by:

$$E = [t]_x R \tag{1}$$

where $[t]_x$is the matrix-representation of the vector cross-product, with the property that $[t]_x\, x \equiv t \times x$. As $[t]_x$ has rank 2 in general, E also has rank 2. From two images alone, the length of t cannot be determined, therefore E is only determined up to scale. A matrix can be decomposed into a rotation and translation in this way when it's Singular Value Decomposition (SVD; [10]) has the form:

$$E = U \begin{pmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & s \end{pmatrix} V^T \tag{2}$$

where U, V are orthonormal matrices. Due to the sign and scale ambiguity in E, U, V can always be chosen to be rotation matrices, and s can be chosen to be 1.

If a 3D point X is viewed in two images at locations X and X' (where X, X' are calibrated homogeneous image coordinates), then **E** has the property that:

$$X'^T E X = 0 \tag{3}$$

Expanding this equation gives a single linear constraint in the nine elements of E for every correspondence. From N correspondences, these equations can be stacked to form a 9×N matrix, with the essential matrix lying in the null space of this matrix. To estimate E the 5-point algorithm is used.

The essential matrix has five degrees of freedom and the minimal set is five point matches. A number of practical algorithms have been proposed [14], [15], the most prominent of which (due to its efficiency) is the 5-point algorithm, proposed by Nister [16].

The least-squares fit to (3), is only an essential matrix if it can be decomposed into a rotation and translation as per (1). The essential matrix is given by its SVD (4):

$$E = U \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} V^T \tag{4}$$

E can be decomposed by SVD, to give its corresponding rotation and translation direction, however two rotations and two (opposite) translation directions satisfy (Eq.1), for any given E. The correct R, t pair is identified by reconstructing a 3D point for each possible R, t; the reconstructed point will fall in front of both cameras only for the correct R, t [17].

Robust motion estimation using RANSAC: The RANSAC (RANdom SAmple Consensus [12]) robust estimation framework enables Essential matrix to be estimated from a set of point correspondences contaminated with outliers. RANSAC works by repeatedly choosing small random subsets of five correspondences ('hypothesis sets'), fitting an essential matrix to each hypothesis set, then counting the total number of correspondences where Sampson's error is below a threshold.

Eventually an essential matrix compatible with many correspondences will be found, usually because the hypothesis set contained only inliers. RANSAC effectively finds essential matrices which are approximately correct, and inlier sets consisting mostly of inliers (typically about 90%), but can be very slow to find more accurate solutions [18], because of the large number of iterations needed to find a hypothesis set containing

only inliers, and because 5-point algorithm solvers are sensitive to point localization errors [19]. As a result, inlier sets tend to contain nearby outliers, and to miss some inliers. Raising the inlier/outlier threshold generally increases the numbers of both inliers and outliers, and reducing it reduces the number of both.

The monocular visual odometry scheme operates as follows:

- Extract the features from the images using the SIFT or SURF features detection and descriptor scheme.
- Match interest points over two frames using Euclidian distance.
- Randomly chose a number of samples each composed of 5 matches between the first and the second frame. Using the five-point algorithm generate a number of hypotheses for the essential matrix.
- Search for the best hypotheses using RANSAC and store the correspondent inliers. The error function is the distance between the epipolar line Eq.3 associated with X and X′.
- Extract from the resulting essential matrix E the relative motion (rotation R and translation T) between two frames.
- Repeat from Point 1.

III. Implementation and Evaluations

The visual odometry algorithm relies on accumulating relative motions, estimated from corresponding features in the images acquired while the robot is moving. Thus to achieve a reliable estimates of the camera pose it is very important to have a set of salient features that are well tracked successfully from the previous images.

Out of the many available detectors/descriptors we wanted to test and compare the most frequently used and best performing for visual odometry. In the class of popular detectors/descriptors which have proven to be effective, and tackle issues such as scale, rotation, viewpoint, or illumination variation are SIFT [10] and SURF [11].

We thoroughly compare the performance of the two detectors/descriptors. To ensure our work is compatible with existing analyses, we have chosen to use images boat dataset, for evaluating the performance of the descriptors, facing the challenges of changes zoom and rotation, and tests are based on matches that are found between two images.

With chosen the two types of detectors SIFT and SURF, now it is of interesting to compare them and evaluate them for their robustness under different conditions. For SURF we have used the author's implementation available at [11], and for SIFT feature detectors/descriptors, the extract_features.ln package available at [20].

We will display in Fig. 1 and Fig. 2, an example of image matching with SIFT and SURF. This example based on pictures from the aforementioned 'boat-database'.

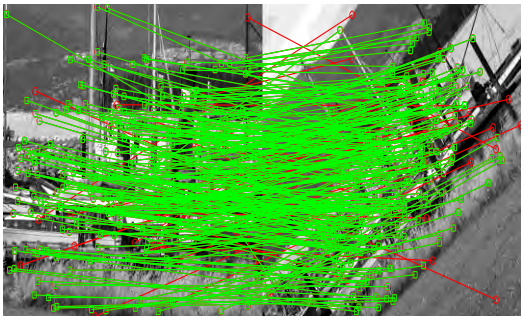

Figure 1. Matching correspondences by SIFT detector/ descriptor. Green lines in the figure correspond to inliers and red lines correspond to outliers.

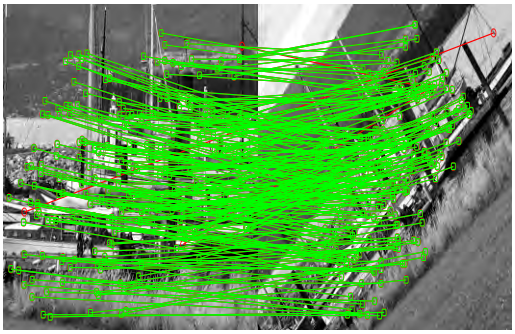

Figure 2. Matching correspondences by SURF detector/ descriptor. Green lines in the figure correspond to inliers and red lines correspond to outliers.

A. Detectors Evaluation

In the first experiment we extracted interest points at each image of the sequences using the methods described in Section II. Next, we computed the numbers of correspondences for each sequence. The result is shown in Fig. 3. The SURF detector gives the maximum number of correspondences than SIFT.

The most important measure that is used for comparing the detectors is the repeatability rate. The number of correspondences found is a raw measure and gives additional information about the results of repeatability comparison. The repeatability measurement tends to give better results if the number of correspondences is higher.

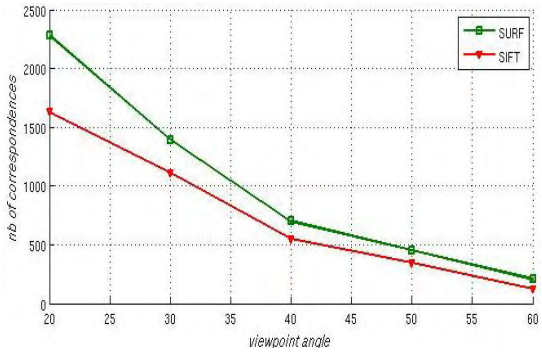

Figure 3. The number of correspondences with viewpoint angle changes

We computed the repeatability rate using (5). The numbers of correspondences in both images is compared, and the smaller of the numbers is used as minimum when calculating the repeatability. In this case only the features that are present in the scene in the both images after the transform are considered.

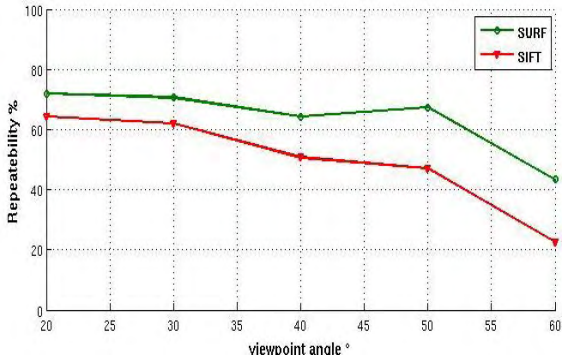

Figure 4. The Repeatability rate of a boat sequence with viewpoint changes.

$$Repeatability\ rate = \frac{the\ numbers\ of\ correspondences}{\min numbersof\ feature(img\ A;img\ B)} \quad (5)$$

In most of the cases, the SURF detector shows the best results. For example, in Fig. 4, it achieves a repeatability rate above 0.65 when image rotated by 50 ° against SIFT which gives 0.45. The results showed that the SURF detector has demonstrated a high stability under changes in scale and viewpoint angle in most of the experiments.

B. Descriptor Evaluation

Various evaluation metrics have been proposed in the literature for analyzing matches. The metric which is widely used for performing such analysis is based on measuring recall and precision. To compute the precision and recall parameters for the matching of descriptors, which are defined as [18]:

$$recall = \frac{the\ numbers\ of\ correct\ matches\ retrieved}{the\ numbers\ of\ total\ correct\ matches} \quad (6)$$

$$precision = \frac{the\ numbers\ of\ correct\ matches\ retrieved}{the\ numbers\ of\ matches\ retrieved} \quad (7)$$

In the expressions above (6) and (7), recall expresses the ability of finding all the correct matches, whereas precision represents the capability to obtain correct matches when the number of matches retrieved varies. The boats scene is used for evaluating scale and viewpoint angle changes; this ranked list of images produces different sets of retrieved matches, and therefore different values of recall and precision. The number of correct matches retrieved is measured by comparing the number of corresponding points obtained with the ground truth and the number of correctly matched points. The ground truth is a homography that projects points to the reference frame.

In a precision versus recall curve, a high precision value with a low recall value means that we have obtained correct matches, but many others have been missed. On the other hand, a high recall value with a low precision value means that we have obtained mostly correct matches but there are also lots of incorrect matches. For this reason, the ideal situation would be to find a descriptor that obtains high values of both parameters simultaneously, thus having values located at the upper-right corner in the precision versus recall curve.

Our comparison of these descriptors has been focused on the assessment of their precision and recall measures under changes of viewpoint angle.

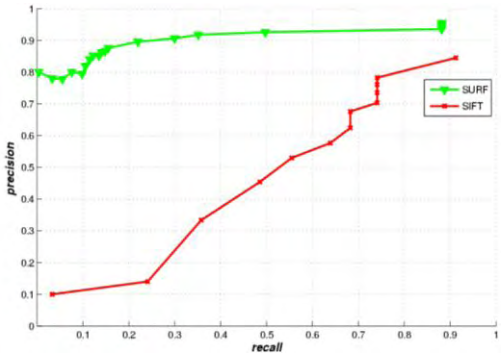

Figure 5. The Recall versus Precision curve with changes in viewpoint angle.

Fig. 5 show the results obtained in viewpoint and scale changing images by boat scene. The figure represents the recall and precision curves for each descriptor. The results are presented in Fig. 5, the figure leads us to the conclusion that SURF is a better descriptor than SIFT.

C. Executing Time Evaluation

As the visual odometry is a real-time application, we also compare the executing time for different detectors and descriptors using the same hardware and software platform. We record the average executing time of each detector/ descriptor in one frame.

TABLE I. AVERAGE RUN-TIME OF THE TWO FEATURE DETECTORS/ DESCRIPTORS IN ONE FRAME.

	SIFT	SURF
TIME (MS)	2.212	1.019

It can be seen from Table I, that the computational cost of SURF is an average of 2.17 times lower than SIFT. Consequently, in order to obtain a comparable level of accuracy, a robot utilizing the SIFT algorithm would be required to travel much slower than a robot utilizing SURF respectively.

The performance evaluation of two popular detectors/descriptors is presented in this section, to find the best among them that would allow us to perform Visual Odometry. The experimental results suggest the SURF detector/descriptor may be a proper solution for monocular visual odometry, when considering the robustness, accuracy and executing time in all.

D. Testing Visual Odometry

The experimental results have been focused on the accuracy of the proposed algorithm. The wheel odometry is also compared to the visual odometry using SIFT or SURF features. We consider wheel odometry as the ground truth of the robot motion. Fig. 6 shows the trajectories estimated by the proposed algorithm (red and blue line for SIFT and SURF features, respectively) for this trial. The wheel odometry (green line) is also drawn in the figure. We have writing a matlab program to compute visual odometry of the first 1001 images of "Flr3-2" dataset, to produce 1000 camera rotations or differential heading changes (dθ). With the rotation information, assuming a robot linear velocity of 0.022 m/image (dx = 0.022), derive and plot the robot trajectory by integrating visual odometry over time. On the same figure, plot the robot trajectory from the wheel odometry in the data set as a reference.

As it is drawn in the Fig. 6, the visual odometry obtains a reliable estimate of the robot displacement, more similar to the trajectory given by the wheel odometry, and improving the internal odometry at the end of the experiment.

It can be clearly seen that SURF is more efficient than SIFT and produce the best trajectory estimation. There are small differences between the real trajectory of robot and estimated trajectory obtained using SURF-visual odometry. So, although the quality and total number of the detected features and their descriptors are influenced to the trajectory estimation by visual odometry.

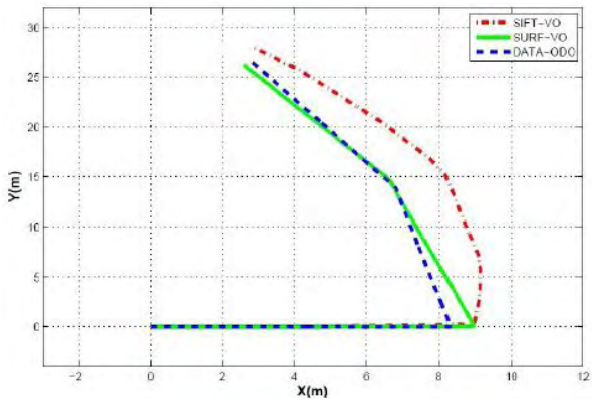

Figure 6. Trajectories estimated by visual (SIFT, SURF) and wheel odometry(red, green and blue lines, respectively).

IV. Conclusion

In this paper we describe a framework for presentation and comparison of visual odometry with a monocular camera. We focus on evaluating two features detectors/ descriptors which are commonly used in lots of computer vision algorithms are Scale-Invariant Feature Transform (SIFT) and Speeded-Up Robust Feature (SURF), to find the most appropriate solution in monocular visual odometry.

Experimental results proved that SURF detector/descriptor outperformed SIFT in all the situations analyzed in this paper. It increased the accuracy percentage which means more reliability in image feature detection and description, yet SURF has a considerably lower computation time. It should therefore be clear that SURF is better suited for the task of visual odometry, but this is only when one considers the matching technique used in this paper. In fact, we believe that SURF might be useful for doing 'coarse' motion estimation, there are cases when SURF does not return a sufficiently high number of correspondences in order to allow precise pose estimation. In these cases, SIFT, which in general returns a higher number of correspondences, might be a better choice.

In our future work, we will test using complex images scenario with more hard conditions like motion blur. We suggest taking advantage of the benefits provided by omnidirectional images, which is advantageous as it captures in a single image the whole surrounding structure, and then to evaluate which detector/ descriptor may be the most suitable solution to the visual odometry using an omnidirectional camera.

References

[1] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in *Proc. IEEE Computer Society Conference Computer Vision and Pattern Recognition*, vol. 1, 2004, pp. 652–659.

[2] M. Maimone, Y. Cheng, and L. Matthies, "Two years of visual odometry on the Mars Exploration Rovers," *Journal of Field Robotics*, vol. 24, no. 3, pp. 169-186, 2007.

[3] J. Courbon, Y. Mezouar, L. Eck, and P. Martinet, "A generic framework for topological navigation of urban vehicle," in *Proc. of ICRA09 Workshop on Safe Navigation in Open and Dynamic Environments Application to Autonomous Vehicles*, Kobe, Japan, May 12th, 2009.

[4] A. I. Comport, E. Malis, and P. Rives, "Accurate quadri-focal tracking for robust 3D visual odometry," in *Proc. IEEE Int. Conf. on Robotics and Automation*, Rome, Italy, April 2007.

[5] D. Scaramuzza and R. Siegwart, "Appearance-guided monocular omnidirectional visual odometry for outdoor ground vehicles," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1015-1026, October 2008.

[6] A. Milella, G. Reina, and R. Siegwart, "Computer vision methods for improved mobile robot state estimation in challenging terrains," *Journal of Multimedia*, vol. 1, no. 7, pp. 49-61, November/December 2006.

[7] T. Linderberg, "Feature detection with automatic scale selection," *International Journal of Computer Vision*, vol. 30, pp. 79-116, 1998.

[8] K. Mikolajczyk and C. Schmid, "An affine invariant interest point detector," in *Proc. European Conference on Computer Vision*, 2002, pp. 128-142.

[9] T. Tuytelaars and L. Van Gool, "Wide baseline stereo based on local, affinely invariant regions," in *Proc. British Machine Vision Conference*, 2000, pp. 412-425.

[10] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, 2004.

[11] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," in *Proc. the Ninth European Conference on Computer Vision*, Graz, Austria, May 2006.

[12] M. Fischler and R. 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.

[13] R. Raguram, J. M. Frahm, and M. Pollefeys, "A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus," *Lecture Notes in Computer Science, Springer*, vol. 5303, pp. 500-513, Berlin / Heidelberg, 2008.

[14] J. Philip, "A non-iterative algorithm for determining all essential matrices corresponding to five point pairs," *Photogrammetric Record*, vol. 15, no. 88, pp. 589-599, 1996.
[15] B. Triggs, "Routines for relative pose of two calibrated cameras from 5 points," *Technical report INRIA*, July 2000.
[16] D. Nister, "An efficient solution to the five point relative pose problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 756-770, June 2004.
[17] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge, UK: Cambridge University Press, 2003.
[18] A. J. Lacey, N. Pinitkarn, and N. A. Thacker, "An evaluation of the performance of RANSAC algorithms for stereo camera calibration," in *Proc. British Machine Vision Conference*, 2000.
[19] T. Botterill, S. Mills, and R. Green, "Fast RANSAC hypothesis generation for essential matrix estimation," in *Proc. Digital Image Computing: Techniques and Applications*, 2011.
[20] Affine Covariant Region Detectors. [Online]. Available: http://www.robots.ox.ac.uk/~vgg/research/affine/detectors.html

Houssem Eddine Benseddik Born in Tlemcen-Algeria, in 1986. After completing his engineering degree in Electronic Technology (2008) and

Magister degree in Advanced Techniques of Signal Processing (2010) from Polytechnic Military School, he is preparing a PhD thesis in Robotics the from the Department Electrical Engineering, University of Tlemcen. He joins the Vision Team of the Robotics Division of the CDTA (Centre de Développement des Technologies Avancées) in Algiers at (2012), as an Researcher in the Computer Vision and Image Analysis.

A.Oualid Djekoune was born in Algiers, Algeria, in October 1969. He received the electronic engineering degree from the electronic institute of USTHB, Algiers, in 1993, the Magister degree in signals and systems in real time from USTHB, in 1997. Today, he works like researcher in the Robotics Laboratory of the Development Center of the Advanced Technologies.

Mahmoud Belhocine gained his Doctorat in robotics in 2006 from USTHB, Algiers (Algeria). He is now head of the Vision Team of the Robotics Division of the CDTA (Centre de Développement des Technologies Avancées) in Algiers. His fields of interest are computer vision, pattern recognition, augmented reality and visual control.

