View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Repositorio Institucional de la Universidad de Alicante

Using a RGB-D camera for 6DoF SLAM

Jose MUÑOZ, Daniel PASTOR, Pablo GIL, Santiago PUENTE, and Miguel CAZORLA [1]
Computer Science Research Institute. University of Alicante. Alicante, Spain

Abstract. This paper presents a method for fast calculation of the egomotion done by a robot using visual features. The method is part of a complete system for automatic map building and Simultaneous Location and Mapping (SLAM). The method uses optical flow in order to determine if the robot has done a movement. If so, some visual features which do not accomplish several criteria are deleted, and then the egomotion is calculated. We use a state-of-the-art algorithm (TORO) in order to rectify the map and solve the SLAM problem. The proposed method provides better efficiency that others current methods.

Keywords. visual features, 3D data, RGB-D data, SLAM.

Introduction

One of the central research themes in mobile robotics is the determination of the movement performed by the robot using its sensors information, which is usually referred as egomotion [1], and also as registration. The method proposed in this work presents some improvements calculating the egomotion which can be used for automatic map building and Simultaneous Location and Mapping (SLAM) [2]. Our main goal is to perform six degrees of freedom (6DoF) visual SLAM in semi-structured environments, i.e., man-made indoor environments. Egomotion can be computed using two main approaches: point-based and feature-based. In the point-based approaches, the most widely used is the Iterative Closest Point (ICP [3]), but it does not provide good results in the presence of outliers and it is time-consuming. Using feature-based (from a RGB-D camera) and using the RANdom SAmple Consensus (RANSAC) method [4] provide a best egomotion calculation.

The system proposed in this paper is similar to the one in [5]. That paper proposes the use of a RGB-D camera (kinect) in order to build a complete SLAM method. The method follows these steps: first, it obtains visual features from two consecutive frames. Using those features and applying a RANSAC-similar method, the egomotion from the two frames is calculated. Then, a global optimization method is applied in order to adjust the map. In this paper, we proposed several improvements which speed up that method.

The remainder of this paper is organized as follows: Section 1 describes the data acquisition system and the Ransac method. Section 2 details the complete method and describes the main improvement of our method. Section 3 shows examples of recon-

[1]Corresponding Author: Dpto. Ciencia de la Computacion e I.A. Universidad de Alicante. P.O.Box 99. 03080 Alicante (Spain); E-mail: miguel.cazorla@ua.es.

struction using our method and, briefly, comments the navigation method used for our experiments. Finally, some conclusions and future work are discussed.

1. Data acquisition

The experiments from this paper were done using the robotic platform shown in Figure 1. The robot platform is a PowerBot from ActiveMedia, which has a good operation autonomy and can carry out several devices over it.

Kinect is used as a main visual sensor. It has provided new ways to interact with environment in the robotics area. It is composed of two sensors: an IR (infrared) projector, IR CMOS camera and a RGB camera. IR sensors provide depth information. The IR projector sends out a fixed pattern of light and dark speckles. Depth is calculated by triangulation against a known pattern from the projector. The pattern is memorized at a known depth and then for each pixel, a correlation between known pattern and current pattern is done, providing the current depth for these pixels. The RGB camera has a resolution of 640 x 480 (307200 pixels) and it can work between 1 and 8 meters, approximately.

Furthermore, in this work, OpenCV (Open source Computer Vision) [6], PCL (Point Cloud Library) [7] and ROS (Robot Operating System) [8] have been used to: detect features, represent 3D information about the environment acquired from Kinect, and control the whole system, from data capture to robot command.

Figure 1. Robot and camera used in our experiments.

We briefly describe here the Ransac method used in this paper. It is an iterative method that estimates the parameters of a mathematical model from a set of observed data which contains outliers. In our case, we look for a 3D transformation (our model) which best explain the data (matches between 3D features). At each iteration of the algorithm, a subset of data elements (matches) is randomly selected. These elements are considered as inliers so that a model (3D transformations) is fitted to those elements. All other data are then tested against the fitted model and if its error is below a threshold then they are included as inliers. In addition, if the estimated model is reasonably good (its error is low enough and it has enough matches), it is considered as a good solution. This process is repeat several times, and the best solution is returned as the system solution.

2. SLAM from visual features

The method used for SLAM using visual features is shown in Figure 2. The method can be described as follow:

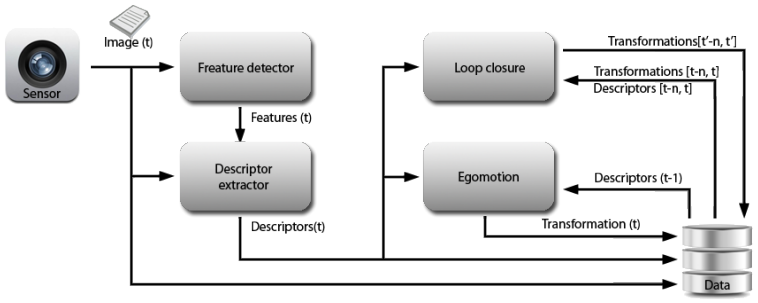

Figure 2. Description of the data flow into the SLAM application.

1. Detecting features for each two frames (f_{k-1} and f_k). In our previous work [9] we made a study of the different visual features which provide best results for the SLAM problem. As a result of that study, the combination of ShiTomasi detector [10] together with SURF descriptor [11] provided the best results. Those combination of detector and descriptor are the one used for this work. In this step, also a filter in order to reduce the number of processed frames have been added. This is done due to the required time to process two consecutive frames could be high with respect to acquisition rate. Then, some useful frames could not be processed. We would like to avoid processing frames where the robot has not changed its pose or the movement is below a given threshold.

   - Firstly, when a new image is captured, the ShiThomasi detector is computed. Then, the optical flow is calculated from those two images, using the iterative disperse version of the Lucas-Kanade pyramidal optical flow [12].
   - If the result from the optical flow indicates that the transformation is below a given threshold, the system does not process the current frame.
   - If the optical flow indicates a significative change, the descriptors from the points detected by ShiThomasi are calculated. This results is passed to the next step.

2. From the features of the previous and current frame, we estimate the egomotion done by the robot. We do not use the result from the optical flow due to that result is noisy and not as accurate as the one proposed here. Features are matched using their descriptors information. But first, we propose to use several filters in order to eliminate bad features. These filters are:

   - Intersection: Features outside the common point of view are rejected.
   - Unicity: One feature in one image must only match to one feature in the other image.
   - Bijective: If a feature in one image matches to a feature in the other feature, the former feature must match the same feature in the first image.

- Directionality: If a direction of a match are outside the normal distribution of directions, this match is deleted. This is done iteratively, calculating the distribution of matching directions and deleting the ones that are outside the distribution.
- Dispersion: If two matches are close enough, one of them is deleted. This is done due to avoid coplanarity in the transformation calculus.

3. At this time, only a few matches have survived (see Figure 3). The 3D information for each feature is assigned. Then, the Ransac method is applied, providing a 3D transformation between these two frames. Each processed frame is a node in a graph, where the transformation between two consecutive frames are the edges.

Figure 3. Two consecutive frames and the points obtained after the filters have been applied.

4. In order to solve the SLAM, we apply the Tree-based netwORk Optimizer (TORO) method [13]. This method uses a tree structure to define and efficiently update local regions at each iteration by applying a variant of stochastic gradient descent. The nodes of the structure are the robot poses and the edges between nodes are the transformations obtained in the last step. TORO, once applied, provides the corrections of the errors accumulated in the path. For loop detection (Figure 2), the search is performed in the neighborhood of the current position. The neighborhood is represented by the nodes close to the current position inside the graph. These neighboring nodes are checked against the current frame, using the method described in the previous step. If a loop closure is detected, the TORO method is applied in order to correct the transformations between nodes and thus obtaining the rectified map.

The execution mean time for the different steps are (in a Intel Core2 Duo T7500 2.2Ghz of CPU with 2GiB of memory RAM):

- ShiTomasi detector: 45 milliseconds.
- Optical flow: 150 milliseconds.
- Surf descriptor: 57 milliseconds.
- Ransac: 30 ms
- TORO: 100 milliseconds.

The detector and the optical flow are executed each time a new image is captured, but the rest of the process is executed by demand, i.e. only when the optical flow filter passes the information to the next step. The TORO method can be executed in a new thread,

even a different machine. The use of the optical flow provides not only a reduction in execution time (descriptor plus ransac plus TORO are not executed), but also the number of nodes in the graph is reduced, reducing at the same time the TORO execution. Without the use of the optical flow step, the TORO step increases its execution time, due to the greater number of nodes in the graph.

3. Results and navigation

In this section, we show some results obtained by our method. In Figures 4 and 5 the reconstructed environments are shown, before and after the TORO SLAM method was applied. In both images, the top image shows the reconstructions using only the egomotion calculated from two consecutive frames. At the bottom one, a rectification of the complete trajectory is done, once the loop detection has been detected.

3.1. Navigation

We are also interested in using the reconstructed map for navigation tasks, even we would like to use partial maps in order to guide the robot through the environment, guiding the robot to unexplored areas. However, the huge quantity of points (more than 300.000 3D points for each frame) makes the map unmanageable. For that reason, we discretize the environment into a more simple one, using voxels. A voxel is a 3D cube which contains all the points inside its coordinates. Thus, instead of managing millions of points, we manage hundred of voxels. These voxels are used for the system in the navigation task. Figure 6 shows an example of voxel reconstruction. The navigation method used is simple: we calculate a method similar to the Virtual Field Histogram (VFH) [14], looking for unexplored areas.

Sometimes, the system is not able to find the egomotion. This problem occurs when a time delay obtaining the data is present or when there is not enough features in the image. To solve this, every robot command is stored in a stack. When the system detects any error (lack of features, bad egomotion), it executes the stored robot commands in a reverse order to reach the last stable pose. Then, it continues with the process.

4. Conclusions

In this paper, a complete and fast method for 6DoF SLAM based on visual features have been decribed. This method is able to build a 3D map and navigate inside it. The proposed method speeds up previous methods and incorporates several filters in order to reduce the inherent error when calculating the egomotion. Results of the SLAM show the validity of our method.

Acknowledgments

These authors want to express their gratitude to Spanish Ministry of Science and Technology (MYCIT) and the Research and Innovation Vice-president Office of the University of Alicante for their financial support through the projects DPI2009-07144 and GRE10-16, respectively.

Figure 4. Top: Reconstruction made only with egomotion. Bottom: Results after applying the TORO algorithm.

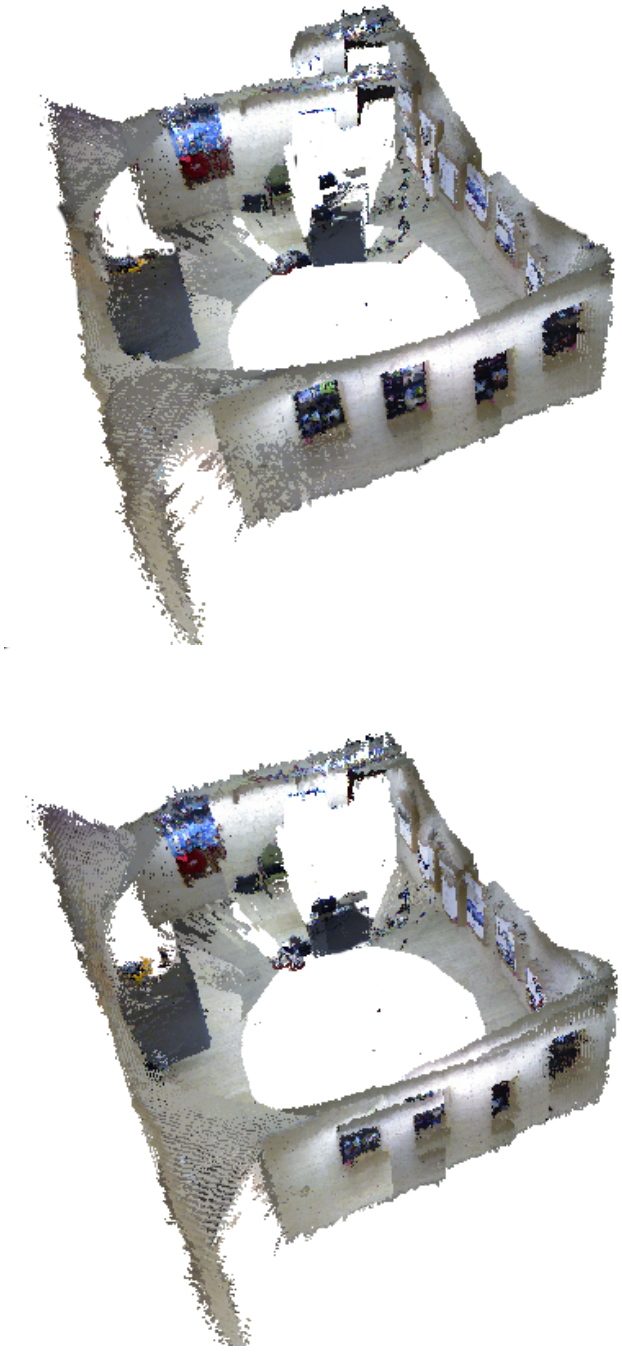

Figure 5. Top: Reconstruction made only with egomotion. Bottom: Results after applying the TORO algorithm.

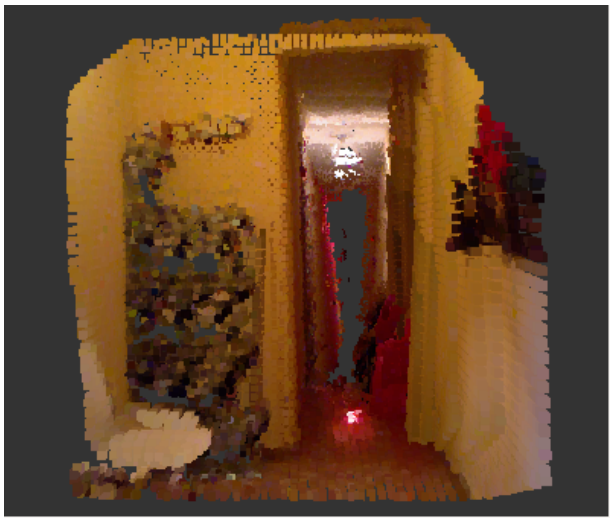

Figure 6. Voxel view of a partial reconstruction.

References

[1] O. Koch, S. Teller, Wide-area egomotion estimation from known 3d structure, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition CVPR '07, pp. 1-8.

[2] M. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, M. Csorba, A solution to the simultaneous localization and map building (slam) problem, Robotics and Automation, IEEE Transactions on 17 (2001) 229241.

[3] P. Besl, N. McKay, A method for registration of 3-d shapes, IEEE Trans. on Pattern Analysis and Machine Intelligence 14 (1992) 239256.

[4] M. A. Fischler, R. C. Bolles, Random sample consensus: a par- adigm for model fitting with applications to image analysis and automated cartography, Commun. ACM 24 (1981) 381395.

[5] Peter Henry, Michael Krainin,Evan Herbst, Xiaofeng Ren and Dieter Fox. RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments. The International Journal of Robotics Research April 2012 vol. 31 no. 5 647-663

[6] OpenCV web site: http://opencv.willowgarage.com/wiki/

[7] PCL web site: http://pointclouds.org/

[8] ROS web site: http://ros.org/

[9] J.L. Muñoz, D. Pastor, P. Gil, S.T. Puente, M. Cazorla. A study of 2D features for 3D visual SLAM. 43rd International Symposium on Robotics (ISR). Taipei (Taiwan) 2012.

[10] J. Shi and C. Tomasi, Good Features to Track, in Proc IEEE Conf. on Computer Vision on Pattern Recognition (CVPR), Seattle, 1994, pp. 593-600.

[11] H. Bay, A. Ess, T. Tuytelaars, L.V. Gool, SURF: Speeded up robust features, Computer Vision and Image Understanding, 2008, vol. 110, no. 3, pp. 346-359

[12] B. D. Lucas and T. Kanade (1981), An iterative image registration technique with an application to stereo vision. Proceedings of Imaging Understanding Workshop, pages 121–130

[13] G. Grisetti, C. Stachmiss, W. Burgard, Non-linear Contraint Network Optimization for Efficient Map Learning, IEEE Trans. on Intelligent Transportation Systems, 2009, vol. 10, no. 3, pp. 428-439.

[14] Borenstein, J. and Koren, Y., 1991, "The Vector Field Histogram – Fast Obstacle-Avoidance for Mobile Robots." IEEE Journal of Robotics and Automation, Vol. 7, No. 3., June 1991, pp. 278-288.