Monocular Vision for Pose Estimation of Autonomous Vehicles

Saúl Martínez-Díaz
División de Estudios de Posgrado e Investigación
Tecnológico Nacional de México-Instituto Tecnológico de La Paz
La Paz BCS, México
saul.md@lapaz.tecnm.mx

Viviana Flores-Herrera
División de Estudios de Posgrado e Investigación
Tecnológico Nacional de México-Instituto Tecnológico de La Paz
La Paz BCS, México
viviana@7robot.net

Octavio Paz Geraldo-Sánchez
División de Estudios de Posgrado e Investigación
Tecnológico Nacional de México-Instituto Tecnológico de La Paz
La Paz BCS, México
octaviopachi348@gmail.com

Daniel Alejandro Contreras-Suárez
División de Estudios de Posgrado e Investigación
Tecnológico Nacional de México-Instituto Tecnológico de La Paz
La Paz BCS, México

José Luis Gómez-Torres
División de Estudios de Posgrado e Investigación
Tecnológico Nacional de México-Instituto Tecnológico de La Paz
La Paz BCS, México
jgomezt@itlp.edu.mx

Israel Marcos Santillán-Méndez
División de Estudios de Posgrado e Investigación
Tecnológico Nacional de México-Instituto Tecnológico de La Paz
La Paz BCS, México
israel.sm@lapaz.tecnm.mx

***Abstract*— Pose estimation in real world environment is an important topic in autonomous vehicles control. Currently used technology for this purpose has some disadvantages in some cases. For example, GPS systems are susceptible to interference, especially in places surrounded by buildings, under bridges or indoors; on the other hand, RGBD sensors can be used, but they are expensive, and its operational range is limited. In this paper, we introduce a low-cost method to compute depth and estimate pose of a vehicle, from two views of at least three 3D points fixed on real world, captured with a calibrated monocular vision system. Computer simulations showed a good performance of proposed method.**

***Keywords*— Monocular vision, pose estimation, visual odometry, autonomous vehicles**

I. Introduction

Recently, due to increase in processing capacity of computers, it has been possible to process digital images in real time. An important application of image processing techniques (in robotics and other areas) is pose estimation [1,2]. GPS systems can be used for this purpose, but they are susceptible to interference, especially in places surrounded by buildings, under bridges or indoors. Also, they have large error margins, up to several decimeters. On the other hand, RGBD cameras can be used too; however, in addition to their high cost, they use infrared sensors to determine the distance from camera to objects (depth). This hinders and even prevents its application in some places illuminated with natural light. Artificial vision is a low-cost suitable alternative. In binocular systems, usually, it is necessary to perform a stereoscopic calibration to know the rotation and translation of one camera with respect to the other. The first camera position generally serves as coordinates reference. The calibration parameters must be kept fixed to reach good results. To calculate the location of a point in three-dimensional space, each camera must capture an image containing that point; then, it is necessary to identify the 2D coordinates of the point within the two images and triangulate them to obtain their 3D coordinates. However, there are some practical disadvantages in this type of systems based on two (or more) cameras: the difference in the response of each camera to the color and luminance of the input signal makes difficult matching of corresponding points; these systems require more physical space, consume more energy and the computational cost is higher because it needs to process two images on each occasion; besides, it is possibility that the cameras lose calibration due to movements or vibrations. In addition, when distant points with two cameras are observed, the system degenerates and tends to behave like a monocular system. All this puts monocular systems as a good alternative. Monocular vision systems can be designed with low-cost hardware. Moreover, it can be used for indoor and outdoor applications.

In monocular systems, since just one camera is used, it is necessary to move the same camera and capture images in different positions. Since this movement is unknown, each new relative pose must be estimated. In literature, two approaches have proven successful for monocular pose estimation: Filtering methods and keyframe-based methods [3-6]. Pioneering work of Davison et al [7] recovered trajectories from a monocular camera by detecting natural landmarks using the Shi and Tomasi [8] operator; they made a probabilistic estimation of the state of the moving camera with an Extended Kalman Filter (EKF). In that approach every frame is processed by the filter to jointly estimate the map feature locations and the camera pose. However, Strasdat et al [9] showed that keyframe-based techniques are more accurate than filtering, for the same computational cost. In this paper, we introduce a low-cost technique to calculate depth and estimate pose of a camera mounted on a moving vehicle. The proposed method is based on a group of three points from a reference object fixed in 3D space, using a calibrated monocular system. For this, only the distance between each pair of reference points must be known. This method is less restrictive than other methods proposed in literature. Only the distance between each pair of reference points must be known for depth computation. The points can be in any position and the camera is not required to be perpendicular to the plane formed by the three points. Using this technique in consecutive images containing the reference object, it is possible to calculate the pose (rotation and translation) of the camera each time, requiring only detection of such object. New pose can be estimated from the rigid transformation of the 3D points, fixed in real world. When reference points are not detected, pose is estimated using a keyframe-based approach; the accumulated error of such estimation can be corrected each time that reference object is found again. Computer simulations show a good performance of proposed method. The rest of the paper is organized as follows: section II introduces some basic concepts that support the work, in section III the proposed method is presented, section IV presents experimental results and section V summarizes our main conclusions.

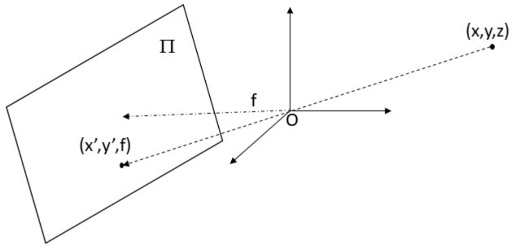

Fig. 1. Pinhole model

II. Basic Concepts

A. Pinhole Model

Due to its simplicity, the pinhole model is widely used to represent the formation of images in a camera. Figure 1 shows pinhole model. As can be seen, each point P with coordinates (x, y, z) in 3D space is projected through the pinhole (which is taken as the origin of the coordinate system) to point P' with coordinates (x', y', f) of the plane Π into the camera. Here, f is the focal length of the camera. From figure 1, we can establish that:

$$\frac{x'}{x} = \frac{y'}{y} = \frac{f}{z} = \lambda \tag{1}$$

Where λ is a scale factor which, when known, allows to calculate the coordinates of 3D point P from the 2D coordinates of the projected point onto the image plane and the focal length. Usually, focal length can be obtained from the camera calibration process.

However, the basic pinhole model does not include any kind of distortions, which usually occur in real world cameras, due to lenses distortions. The main kinds of distortions are tangential and radial, which can be estimated during calibration process. Then, they must be compensated using a suitable algorithm for example by expansion in Taylor´s series, as is described in reference [10].

B. Camera Calibration

Camera calibration gives a model of the camera's geometry and distortions caused by lenses. This information can be used to define the intrinsic and extrinsic parameters of the camera.

Let **P**' be the projected point of a 3D point **P** in the camera plane, as shown in figure 1. By using homogeneous coordinates, we define P = $[x\ y\ z\ 1]^T$ and P' = $[x'\ y'\ 1]^T$ (here superscript T means transpose of vectors). Then, we can express the mapping from **P** to **P'** in terms of matrix multiplication as [11]:

$$\lambda \mathbf{P'} {=} \mathbf{A}[\mathbf{R}\ \mathbf{t}]\mathbf{P} \tag{2}$$

where **P** is a point of the object in 3D, in homogeneous coordinates; $\mathbf{P'}$ is the same point of the object in homogeneous 2D coordinates; [**R t**] is a matrix of extrinsic parameters (rotation and translation); λ is an arbitrary scale factor and **A** is a matrix of intrinsic parameters. The information of the intrinsic parameter matrix is defined by:

$$A = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

Here f_x, f_y provide information (depending on the size of the pixel) of the focal distance in the direction of x and y, respectively; c_x and c_y are the coordinates of the main point of the image. The information of intrinsic and extrinsic parameters can be obtained in the calibration process.

Fig. 2. Calibration pattern.

The authors greatly appreciate the support of CONACyT and Tecnológico Nacional de México/Instituto Tecnológico de La Paz for the realization of this project.

III. PROPOSED METHOD

In this section we describe our proposed method. The main steps are:

1. Capture a pattern image
2. Convert the image to grayscale
3. Correct distortions
4. Search the reference object
5. If object is detected
   a. Select three points with known distance
   b. Calculate depth and location of selected points
   c. Compute pose from the rigid transformation between selected points in current image and previous image with the same points detected
   d. Go to step 1
6. Else
   a. Estimate pose with keyframe-based approach
   b. Go to step 1

A. Object Detection

Detection of reference objects is based on corner detection. Corners are invariant to translation, rotation and illumination, and exist robust algorithms to detect them. The main idea for corner detection is searching for strong derivatives in two orthogonal directions of image *I* at coordinates (*x*, *y*). This can be done by applying a matrix of second-order derivatives (Hessian matrix **Hs**) to image intensities [12]:

$$\mathbf{Hs}(p) = \begin{bmatrix} \frac{\partial^2 I}{\partial x^2} & \frac{\partial^2 I}{\partial x \partial y} \\ \frac{\partial^2 I}{\partial y \partial x} & \frac{\partial^2 I}{\partial x y^2} \end{bmatrix} \tag{4}$$

In practice, the most used approach is by applying the autocorrelation matrix **M** of the second derivative images over a small window W around each point:

$$\mathbf{M}(x,y) = \begin{bmatrix} \sum_{x,y \in W} I_x^2(x,y) & \sum_{x,y \in W} I_x(x,y) I_y(x,y) \\ \sum_{x,y \in W} I_x(x,y) I_y(x,y) & \sum_{x,y \in W} I_y^2(x,y) \end{bmatrix} \tag{5}$$

B. Depth Computation

From equation (1) the following relationships can be obtained:

$$x = \frac{x'}{f} z \qquad y = \frac{y'}{f} z \tag{6}$$

Suppose you have at least three points P_1, P_2, P_3 in 3D space with coordinates (x_1, y_1, z_1), (x_2, y_2, z_2), and (x_3, y_3, z_3), respectively. Suppose also that the distances d_1, d_2 and d_3 between each pair of points are known. Using the relationships in (6), the quadratic Euclidean distance between each pair of points can be calculated by:

$$\begin{aligned} d_1 &= \left(\frac{x'_1}{f} z_1 - \frac{x'_2}{f} z_2\right)^2 + \left(\frac{y'_1}{f} z_1 - \frac{y'_2}{f} z_2\right)^2 + (z_1 - z_2)^2 \\ d_2 &= \left(\frac{x'_1}{f} z_1 - \frac{x'_3}{f} z_3\right)^2 + \left(\frac{y'_1}{f} z_1 - \frac{y'_3}{f} z_3\right)^2 + (z_1 - z_3)^2 \\ d_3 &= \left(\frac{x'_3}{f} z_3 - \frac{x'_2}{f} z_2\right)^2 + \left(\frac{y'_3}{f} z_3 - \frac{y'_2}{f} z_2\right)^2 + (z_3 - z_2)^2 \end{aligned} \tag{7}$$

This system of nonlinear equations can be solved numerically for z_i, which represent the depth at which each point is located with respect to the camera. Substituting back these results in (6) we can also obtain the real-world coordinates (x_i, y_i) of the three points. If two images are taken with the same camera in different poses, applying the proposed method, it is possible to estimate the relative displacement (rotation and translation) between both poses as the rigid transformation of world points.

C. Pose Estimation

When reference objects are not detected in two consecutive images, we use the keyframe-based approach. The basic keyframe-based algorithm is:

- Capture images from video sequence
- Extract interest features (corners) from each image
- Match features between previous and current images and extract 2D points
- Triangulate matched points using calibration parameters
- Compute the current camera position with P3P algorithm [13]
- Apply Random Sample Consensus (RANSAC) [14] to estimate pose robustly
- Optimize estimation with bundle adjustment (BA) [15, 16]

Although there are fast algorithms for BA, since is required to compute error from accumulated views, the computational cost increases with each new image. Therefore, it is necessary to restrict the optimization process to use a fixed number of views, reducing performance of the estimation.

IV. RESULTS

A. Depth Computation Test

To test the depth calculation of proposed method, the chessboard pattern shown in figure 2 was used as reference object. Each square is 27x27 millimeters. We used a low-cost Microsoft USB camera, mounted on a small car and configured to capture images in RGB format with a low resolution of 640x480x3 pixels. The illumination source provided non-homogeneous light from a set of AC lamps.

In this work, the calibration of the cameras was carried out with the technique proposed by Zhang [17]. With this calibration technique, it is only required that the camera observes a flat pattern (figure 2) taken from different orientations. The pattern or camera can be moved freely, and it is not necessary to know the movement made. This calibration allows to obtain simultaneously the intrinsic and extrinsic parameters of the camera. For comparison of results, Kinect 2 device was used. Kinect is a RGBD system, which makes available 3D coordinates of real-world points. It contains a visible light camera with a high resolution of 1920x1080x3 and an infrared system to provide depth information.

Now, car was moved at different distances from the pattern and the calculation algorithm was applied. At first, with the 3D coordinates of points detected with the proposed depth computation algorithm we calculated the rigid transformation (rotation and translation) with respect to the first view. Then, when reference pattern is not detected correctly, estimation was done with the keyframe-based approach until reference pattern is detected again.

Figure 3 shows a comparison of distances obtained with the proposed method versus these obtained with the Kinect 2 device. We marked manually coordinates of the camera into each image acquired with Kinect color device; then, information of depth device was incorporated, and 3D coordinates were obtained by using the functions of MATLAB toolbox for Kinect. There is a correlation of 98.4163% between Kinect distances and these measured by our method, with 9% of average percent error. Note that, due to manual procedure, human mistakes and other errors could be introduced. However, results are promising.

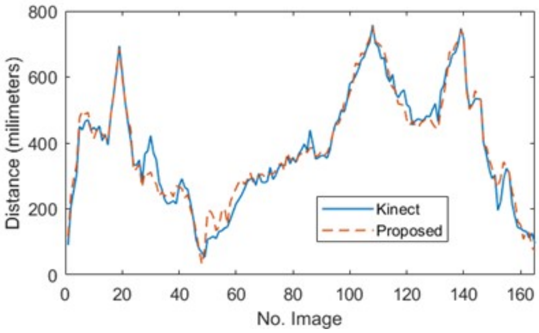

Fig. 3. Comparison of distances measured with Kinect and the proposed method.

B. *Pose Estimation Test*

To test keyframe-based pose estimation of the system, we use 600 images from sequence 00 of KIITI dataset. This dataset contains images acquired with PointGray Flea2 grayscale cameras. Each grayscale image is 376x1241 pixels of resolution. The dataset includes ground-truth references obtained with a high-precision GPS/IMU inertial navigation system. Figure 4 is an example of used images, which were acquired into a residential environment.

Only for the initial scale adjustment, we used the first two rows from ground truth file. From each image we detected and matched corner features; then, P3P algorithm with RANSAC estimation was performed. To reduce computation time, BA is performed with 12 frames each time. Figure 5 shows results of comparison between ground-truth and estimated data, for the rest of dataset images. There is a correlation of 0.9996 between both graphs with 5.88% of average percent error in the estimation process.

Fig. 4. Example of image from KITTI dataset.

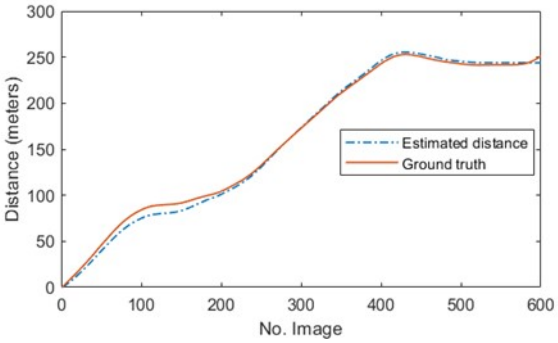

Fig. 5. Comparison of ground truth distances from KITTI dataset and estimation with the proposed method.

Due to BA is performed with a few frames, when the reference object is not detected for a long time (more than 600 frames), keyframe-based estimation error increases considerably. This error can be reduced by recognizing several reference objects placed along the scene. which implies using a robust object detection technique, such as a convolutional neural network.

V. Conclusions

In this paper, a method to calculate the depth and location of points in three-dimensional space with a monocular vision system was proposed. This depth and location information serves to estimate pose of a vehicle with a low-cost monocular vision system mounted on it. Reference points can be obtained from any known object encountered in real world. For the calculation, it is only necessary to know the distance between each pair of points. If the reference object is not detected, depth is estimated using a keyframe-based algorithm. Error due to drift can be corrected when reference object is detected again. This pose information can serve for applications of autonomous robot navigation, industrial control systems or augmented reality, by using a single camera.

The experimental results show a good performance of the depth computation algorithm in images taken with a low-cost camera, under uncontrolled conditions. The maximum error obtained was less than 0.25%, compared with a calibrated standard ruler. Besides, for initial depth calculation, this method is less restrictive than state of the art methods in literature.

On the other hand, although keyframe-based estimation algorithms are widely used, they must be improved for speed and accuracy. A drawback of keyframe-based estimation is that error increases considerably when reference object is not detected in several frames (in our experiments, more than 600 frames) and loop closure is not used. To avoid error accumulation, it could be necessary to search for different reference objects in each scene and, when one of them is detected, compute depth and adjust scale of estimation to correct accumulated error. In that case, a classifier system is required to detect such objects. This system must be trained to recognize the most common items in each environment and provide information about size of salient features in each object.

Acknowledgment

The authors want to thanks to Consejo Nacional de Ciencia y Tecnología (CONACyT) and Tecnológico Nacional de México/Instituto Tecnológico de La Paz for the facilities and support given to this project.

References

[1] K. Wang, Y. Liu and L. Li, "Vision-based tracking control of underactuated water surface robots without direct position measurement," IEEE Transactions on Control Systems Technology, vol. 23(6), pp. 2391-2399, 2015.

[2] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras," IEEE Transactions on Robotics, vol. 33(5), pp. 1255-1262, 2017.

[3] E. Mouragnon, , M. Lhuillier, M. Dhome, F. Dekeyser and P. Sayd, "Real time localization and 3d reconstruction," in 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 363–370, 2006.

[4] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR), Nara, Japan, pp. 225–234 November 2007.

[5] R. Mur-Artal, J. M. M. Montiel and J. D. Tardos, "ORB-SLAM: A versatile and accurate mo-nocular SLAM system," IEEE Trans. Robot, vol. 31(5), pp. 1147–1163, 2015.

[6] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," Int. J. Robot. Res., vol. 34(3), pp. 314–334, 2015.

[7] A. J. Davison, I. D. Reid, N. D. Molton and O. Stasse, "MonoSLAM: Real-time single camera SLAM," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29(6), pp. 1052–1067, 2007.

[8] J. Shi and C. Tomasi, "Good features to track," in Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 593-600, June 1994.

[9] H. Strasdat, J. M. M. Montiel and A. J. Davison, "Visual SLAM: Why filter?," Image and Vision Computing, vol. 30(2), pp. 65–77, 2012.

[10] G. Bradski and A. Kaehler, Learning OpenCV, O'Reilly, pp. 370–375, Sebastopol, California, USA, 2008.

[11] R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge University, 2004.

[12] C. Harris and M. Stephens, "A Combined Corner and Edge Detector," in Proceedings of the 4th Alvey Vision Conference, pp. 147–151, September 1988.

[13] Sh. Gao, X.-R. Hou, J. Tang and H.-F.Cheng, "Complete solution classification for the perspective-three-point problem," IEEE PAMI, vol. 25(8), pp. 930-943, 2003.

[14] P. H. S. Torr and A. Zisserman, "MLESAC: A New Robust Estimator with Application to Esti-mating Image Geometry," Computer Vision and Image Understanding, vol. 18(1), pp. 138–156, 2000.

[15] M. I. A. Lourakis and A. A.Argyros, "SBA: a software package for generic sparse bundle ad-justment," ACM Trans. Math. Soft, vol. 36(1), 2009.

[16] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," Quarterly of Applied Mathematics, pp. 164–168, 1944.

[17] Z. Zhang, "A flexible new technique for camera calibration," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22(11), pp. 1330-1334, 2000.