A Framework for Real Time Vehicle Pose Estimation based on synthetic method of obtaining 2D-to-3D Point Correspondence

Sergey Yun and Moongu Jeon*
*Dept. of Information and Communications, Gwangju Institute of Science and Technology
e-mail : [reigan, mgjeon]@gist.ac.kr

Abstract

In this work we present a robust and fast approach to estimate 3D vehicle pose that can provide results under a specific traffic surveillance conditions. Such limitations are expressed by single fixed CCTV camera that is located relatively high above the ground, its pitch axes is parallel to the reference plane and the camera focus assumed to be known. The benefit of our framework that it does not require prior training, camera calibration and does not heavily rely on 3D model shape as most common technics do. Also it deals with a bad shape condition of the objects as we focused on low resolution surveillance scenes. Pose estimation task is presented as PnP problem to solve it we use well known "POSIT" algorithm [1]. In order to use this algorithm at least 4 non coplanar point's correspondence is required. To find such we propose a set of techniques based on model and scene geometry. Our framework can be applied in real time video sequence. Results for estimated vehicle pose are shown in real image scene.

Keywords: *Vehicle pose estimation; Posit; Point correspondence; PnP; Monocular*

1. Introduction

In computer vision, visual traffic surveillance is an important task, which deals with such applications as estimating traffic scene topology and geometry, controlling traffic activities, detecting abnormal situations. This kind of traffic data is very useful for transportation planning and Intelligent Transportation Systems (ITS). One part of these applications is a problem of pose estimation. It has the aim to find the rotation and translation between an object coordinate system and a camera coordinate system. Given are correspondences between 3D points of the object and their corresponding 2D projections in the image. Additionally the internal parameters focal length and principal point have to be known. There are extensive literatures on this topic, including both non-iterative algorithms and iterative ones. One kind of non-iterative approaches is to develop a large group of linear algebra equations and solve those equations for distance information first, and then camera position and rotation can be achieved by solving an absolute orientation problem [2]-[4]. Most iterative approaches are based on minimizing an error function developed from some nonlinear geometric constraints. The most classical approaches rely on a Gauss-Newton style method and use the reprojection error as the criterion [5], [6]. However, these algorithms are usually not robust under a bad initialization, and suffer from low convergence speed. Also it is possible to estimate the 3D rotation and translation of a 3D object from a single 2D photo if we can register a 3D CAD model over the photograph of a known object by optimizing a suitable distance measure with respect to the pose parameters [7][8]. The distance measure is computed between the object in the photograph and the 3D CAD model projection at a given pose. Perspective projection or orthogonal is possible depending on the pose representation used. This approach is appropriate for applications where a 3D CAD model of a known object (or object category) is available. Another approach is if we know an approximate 3D model of the object and the corresponding points in the 2D image. A common technique for solving this is "POSIT" (Pose from Orthography and Scaling with Iterations), where the 3D pose is estimated directly from the 3D model points and the 2D image points, and corrects the errors iteratively until a good estimate is found from a single image. This iterative algorithm has both coplanar and non-coplanar point's realizations with good convergence speed (at least 4 or 5 iterations are required to find the pose [1]. We adopted it for continuous video sequence of images obtained from CCTV camera to estimate multiple vehicles' poses. The reason why we choose "POSIT" is that it suits our motivation expectations in computational speed, robustness and it does not require initial pose estimate as common techniques do.

Motivation. It is difficult to obtain vehicle pose estimation using intrusive technologies due to roadway geometry (e.g., geometry where there are significant lane changes or where vehicles do not follow a set path in making turns), scenes where it is difficult to extract corners or edges from an image, because of low video quality, and the distance from camera to objects is quite high. To work under such challenging conditions we propose a framework that uses vehicles shape information and intuition on scene geometry. Shape is typically more invariant to color, texture, and brightness changes in the image than other features (e.g., interest points). We assume that:

- Almost all moving objects including vehicles and pedestrians are moving on the ground plane.
- Vehicles always run along the roadway which is supposed to be of a straight or curvature form.
- Vehicle height and pedestrians trunk directions are perpendicular to the ground plane.

These three properties are found in most traffic surveillance scenes. In case of 3D CAD model registration algorithms, we do not deal with these methods due to availability of significant amount of models. Our intuition on approximate 3D shape of model is that any vehicle can be bounded by 3D box with height H, width W and length L.

Shape of the vehicle is represented as edges of that box. Our main contribution in this work is the novel framework for vehicle pose estimation that does neither depend on color or gradient of the object nor on the lightning of the real scene and can deal with far distant objects with low resolution.

2. Proposed Approach

In this section we describe an overview of proposed framework. Each section is related to detailed explanation of separate module, including background subtraction, vehicle direction estimation and so on. The flowchart of the whole system is presented on Fig.1. The input frame is noted as a bold arrow that goes to background subtraction module. Earlier we mentioned that shape is the most informative feature of the object, we represent shape as an external contour of the vehicle. To extract it we use specific method for background subtraction that feat our requirements: 1) deal with shadows, 2) robust to color changes, 3) can be applied in real time.

To apply our approach for each vehicle in the scene we use vehicle detector that provides 2D bounding box information: width w, height h and the top-left corner coordinate p of the box. Having such data we can conduct direction estimation for each detected vehicle by using Region of Interest (ROI). To estimate vehicle orientation we use data difference of 2D box estimated in previous frame with current one, under orientation we assume direction angle of the vehicle movement. This information is required for our next module of vehicle classification by its orientation. Once the direction class of the detected vehicle is obtained, we apply method for finding 4 non coplanar points and their correspondence. Finally, having all the mentioned above information, we can estimate the pose of the vehicle.

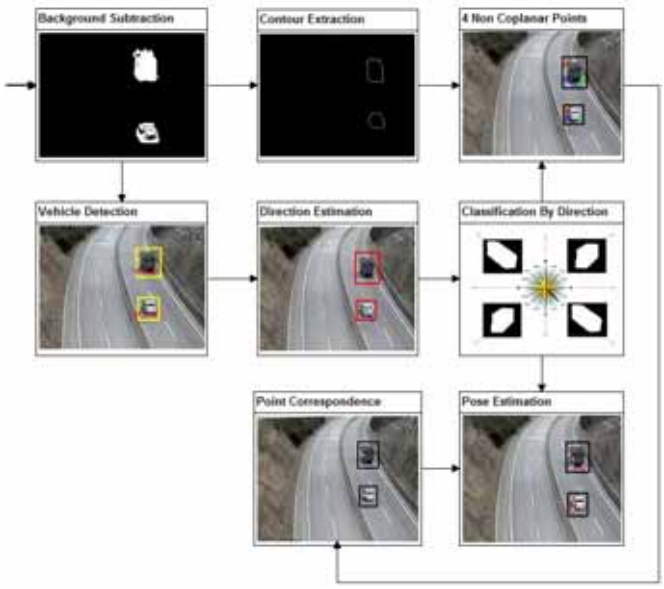


Fig.1. Flowchart of the proposed framework

2.1 Vehicle Detection, Background Subtraction and Contour Extraction

Vehicle detection method based not only on appearance characteristics of vehicles but also utilizes actually observable size-patterns of vehicles in a road [9]. The algorithm works as follows: size information of non-interacting moving-blobs is first collected based on the developed blob-level analysis technique. Then, a new region-wise sequential clustering algorithm is performed to train and maintain the size-pattern model, which is utilized to deform shapes of the sliding windows scene specifically at each image position. All the proposed procedures operate full-automatically in real-time without any assumptions. To extract outer contour of the object, we need background subtraction technique that can preserve shape of the vehicle, deal with its shadow and be fast enough for real time application. For this sake we used method proposed by R. Cucchiara, A. Prati etc. in their work [10]. Performance comparison and the source code of this algorithm are available by the following link[1]. It is obviously not enough to perform background subtraction alone to get accurate foreground mask for contour extraction due to some noise which is presented in binary image. To eliminate that kind of unwanted attributes we use some morphological operations. First, we applied median filter to “eat” some salty noise. Second, to erase noise inside the object we use operation of closing. Once we obtain a clear foreground mask we extract the contour of the vehicle using contour algorithm provided by OpenCV library. Also to eliminate some shape roughness we approximate our contour by polygons. The result of this module you can find on Fig. 2(c).

(a) (b) (c)

Fig.2. Prior evaluation results. (a) Detection module; (b) R. Cucchiara, A. Prati background subtraction; (c) Contour extraction.

2.2 Vehicle Orientation Estimation and Direction Classification

For each detected vehicle in the single video frame we have its ROI. This region provides the information about its top-left corner location p, width w and height h. Using these data we can determine the coordinates of the foreground mask center O as $O_x = p_x + w/2$, $O_y = p_y + h/2$. To estimate vehicle orientation we memorize each vehicles center in the current frame. We denote shift distance along x axes between the current frame center O_2 and the previous O_1 as d_x and along y axes as d_y (Fig. 3(b)). Further we calculate the direction angle α by using (1)-(3).

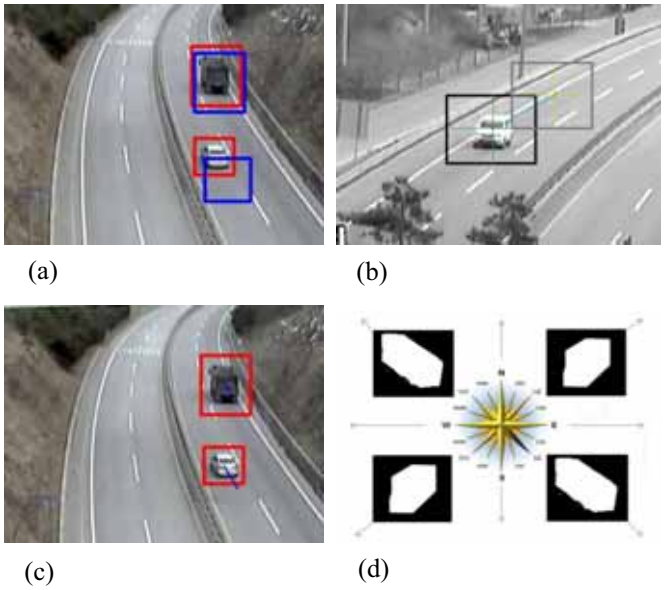
$$d_x = |O_{1y} - O_{2y}|; \quad (1)$$

$$d_y = |O_{1x} - O_{2x}|; \quad (2)$$

$$\alpha = arcSin\,(w/\sqrt{d_x^2 + d_y^2}). \quad (3)$$

Every estimated direction of the vehicle we mark as *top, bottom, left, right, top-left, top-right, bottom-left, bottom-right*. As you can see from Fig. 3(d) the shape of *top-left* foreground mask and *bottom-right* are quite similar, as well as *top* and *bottom*, *left* and *right*, *top-right* and *bottom-left.*

Having such observation we divide each detected vehicle into four 3D-box model classes: *top* and *bottom* as *vertical*, *left* and *right* as *horizontal*, *top-left* and *bottom-right* as *incidental diagonal*, *top-right* and *bottom-left* as *main diagonal*.



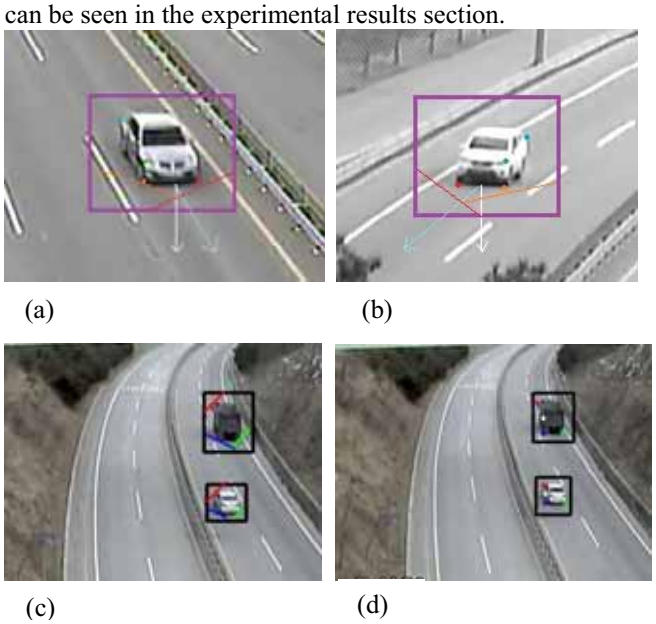
(a) (b) (c) (d)

Fig.3. Direction estimation and classification. (a) Previous and current ROI; (b) Center difference; (c) Direction segment; (d)Direction groups.

1. http://code.google.com/p/bgslibrary.

2.3 Finding 4 Non-Coplanar Points. Pose Estimation

In this section we use all previously obtained features to find 4 non-coplanar points. We propose an iterative method of sliding segments to find each point. The algorithm starts with building the direction line by using estimated angle which goes throw the center of the foreground mask(Fig. 3(c)), next we use direction information to determine which points we are going to search and for what 3D box model. For example if we have *bottom-right* direction information as shown on Fig. 4(a) we create three sliding segments that are moving along the direction line under defined angle until first intersection with the contour set of points. The angle for the first line segment is defined as $\beta = \pi/4 - \alpha/2$, for the second and the third segment as $\mu = \pi/4 + \alpha/2$. First non-coplanar P_1 point is an intersection point of the first line segment with the contour; second point P_2 is an intersection of the second line segment with the contour, similarly we find third one P_3 (Fig. 4(c)). To find fourth non-coplanar point by moving up along y coordinate of the first point P_1 by the distance $t = \frac{\pi \|P_{2x} - P_{3x}\|}{w}$. Same procedure for the rest cases, including the one that is shown on Fig. 4(b), just the position of the segments is different. Now we have got all what we need for the final step - to conduct pose estimation. As we mentioned earlier POSIT algorithm requires at least 4 non-coplanar points founded on the 2D image (Fig. 4(c)) and their correspondence to the 3D model points. In section 2.2 we classified every detected vehicle by its direction information into 4 groups. Each group represents a 3D model for which we have defined correspondent points. The results can be seen in the experimental results section.



(a) (b) (c) (d)

Fig.4. Points evaluation. (a) & (b) Sliding segments example; (c) First intersection; (d) Estimated points.

3. Experimental Results

Experimental results on three video scenes of the different vehicle types and colors are shown in Fig. 5(a, b, c). The video frames are taken from CCTV camera and include realistic conditions like shadows and surface reflections. Line segments represent the projected 3D model box which bounds the vehicle. For the last image we printed the rotation matrixes and translation vectors of detected vehicles as well as coordinates of four estimated non-coplanar points and projected points after multiplying 3D box model by transformation matrix. Also it is clear that obtained results are not ideal and contain some pose errors, if we check source image points and estimated points the values are sometimes not close to be identical.

(a)

(b)

(c)

(d)

1) Source image points | Estimated image points
7, -11 | 7, -11
14, -4 | 13.238, -3.12022
-14, -12 | -11.7948, -12.1379
7, 0 | 7.00178, 1.46775

Estimated rotation
0.175766 | -0.00118364 | -0.984431
0.202936 | 0.977255 | -0.0615664
0.962113 | -0.188955 | 0.172009

Estimated translation
5.48834 | -8.62453 | 784.048

2) Source image points | Estimated image points
28, -39 | 28, -39
41, -22 | 41.2957, -22.2342
-28, -35 | -29.04, -35.0657
28, 0 | 27.9973, -0.704897

Estimated rotation
0.155014 | -0.00373565 | -0.987905
0.12278 | 0.990468 | 0.0624314
0.978255 | -0.130972 | 0.153995

Estimated translation
7.24127 | -10.0861 | 258.617

3) Source image points | Estimated image points
20, -17 | 20, -17
25, -19 | 24.0335, -19.6872
-21, -17 | -14.7164, -17.0107
20, 2 | 19.9959, 6.20231

Estimated rotation
0.0817397 | 0.000582924 | -0.996653
-0.058114 | 0.998309 | -0.00169557
0.994966 | 0.0580581 | 0.0816586

Estimated translation
8.60215 | -7.31182 | 430.107

Fig.5. Final results. (a),(b)&(c) Different video frames; (d) Rotation matrixes and translation vectors of the three detected vehicles in (c) scene.

4. Conclusion

In this work we showed that proposed framework can deal with the posed problem. The best use of it is in the situations when vehicles are located at a far distance from the camera and the camera video quality is quite low. Also to estimate pose we don't need a 3D CAD model and initial pose guess. In future we plan to refine the estimation results of our approach.

Acknowledgement

This work was supported by the Center for Integrated Smart Sensors funded by the Ministry of Science, ICT & Future Planning as Global Frontier Project (CISS-2011-0031868)

References

[1] D.F. DeMenthon "Model-based object pose in 25 lines of code", Proc. DARPA, Computer vision ECCV, 1995.

[2] Z. Zhong, J. Yi, D. Zhao, Y. Hong. "Effective pose estimation from point pairs," Image and Vision Computing, 2005.

[3] M. L. Liu, K. H. Wong. "Pose estimation using four corresponding points", Pattern Recognition Letters, 1999.

[4] L. Quan, Z. Lan. "Linear N-point camera pose determination", IEEE Transactions on Pattern Analysis and Machine Intelligence, 1999.

[5] D.G. Lowe. "Three-dimensional object recognition from single two-dimensional image", Artificial Intelligence.

[6] H. Araujo, R. Carceroni, and C. Brown. "A fully projective formulation for Lowe's tracking algorithm", Technical Report 641, 1996.

[7] Srimal Jayawardena and Marcus Hutter and Nathan Brewer. "A Novel Illumination-Invariant Loss for Monocular 3D Pose Estimation", International Conference on Digital Image Computing: Techniques and Applications, 2011.

[8] M. Hodlmoser, M. Kampel. "Classification and pose estimation of vehicles in videos by 3D modelling within Discrete-Continuous optimization", 3DIMPVT, 2012.

[9] S. Noh, D. Shim and M. Jeon. "A New Vehicle Detection Method for Intelligent Transport Systems Based on Scene-Specific Sliding Windows", ICINCO, 2013.

[10] R. Cucchiaram and C. Grana and M. Piccardi and A. Prati. "Detecting Moving Objects, Ghost and Shadows in Video Streams". PAMI 25, 2003.