Offline Camera Movement Tracking from Video Sequences

Primastuti Dewi[*] · Yeon-seok Choi[*] · Eui-young Cha[*]

[*]Dept. of Computer Engineering, Pusan National University

E-mail : {primastuti, yeonseokchoi, eycha}@pusan.ac.kr

ABSTRACT

In this paper, we propose a method to track the movement of camera from the video sequences. This method is useful for video analysis and can be applied as pre-processing step in some application such as video stabilizer and marker-less augmented reality. First, we extract the features in each frame using corner point detection. The features in current frame are then compared with the features in the adjacent frames to calculate the optical flow which represents the relative movement of the camera. The optical flow is then analyzed to obtain camera movement parameter. The final step is camera movement estimation and correction to increase the accuracy. The method performance is verified by generating a 3D map of camera movement and embedding 3D object to the video. The demonstrated examples in this paper show that this method has a high accuracy and rarely produce any jitter.

KEYWORD

camera tracking, optical flow, video analysis, marker-less AR

Ⅰ. Introduction

The computer vision field has developed rapidly in past view decades. Supported by higher quality of camera, computer hardware, and new technique which are being developed continuously, the cost and time to process image or video is getting lower and results are getting more accurate.

In this paper, we use computer vision technique to track the 6 DoF (six degrees of freedom; forward/backward, up/down, and left/right) movement of camera from a video sequence automatically. Imitating the process in human eye, the movement is detected from relative motion shown by image change in the video sequence time to time which represent by optical flow, as shown in figure 1.

Here, we proposed an offline camera tracking method from a video sequences. This method can be applied simply for video analysis such as tracking the movement of autonomous robot, or can be used as pre-processing step for image stabilizer, match moving, augmented reality, automatic 3D environment mapping and other applications.

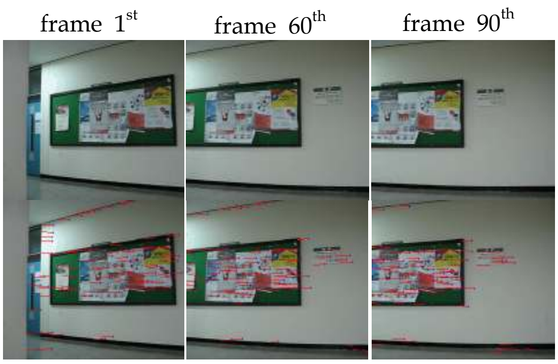

Figure 1. (Top) Example of frames in video. (Bottom) Using the red lines which show optical flow, we can assume that the camera turned to the right side.

The rest of the paper consists of detail explanation about the proposed camera movement tracking method in section II, method verification by using generating 3D viewer of camera movement which relative to camera initial position and embedding 3D object into the video in section III, and the last part, section IV includes the conclusion and the future work.

Ⅱ. Camera Movement Tracking

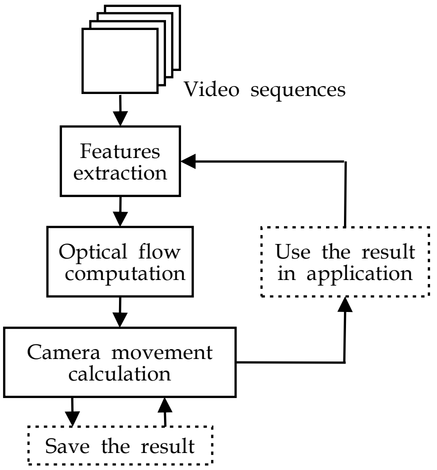

Figure 2. Camera movement tracking method

1. Related works

Depend on the system input and processing time, camera tracking can be divided into online and offline method. Online tracking is track the camera movement in real-time, focus on processing the data as fast as possible. Offline tracking is detect and track the movement of camera using global data from the whole video sequence. This method is more focus on accuracy and reliability, but take longer processing time than online method. In this paper we proposed a method to track the camera movement from a video.

There are several previous methods for camera movement tracking. Generally, the methods can be divided into feature extraction, optical flow computation, and camera relative movement calculation as illustrated in figure 2.

At the first step, there are many information included in an image that can be used as features, but corner points are commonly used because they are easy to detect and handle, relative stable and not inferior to other types of features such as lines or planes[1]. The features are then tracked to compute optical flow which represents the relative movement of camera. Some famous methods to do this process are Kande-Lucas method[2], Horn-Shunck method[3], Kanade-Lucas-Tomasi (KLT) feature tracker and its variants[4], and SIFT[5].

Then the last step is camera movement calculation. Calculate the camera position change (e.g, rotation and translation in each axis) from two frames which is can represented by 3 by 3 homography or fundamental matrix. This is can be done using statistical method such as unscented particle filter[6], extended kalman filter[7], unscented Kalman filter, or other method such as random sampling algorithm[8] and non-linear method.

2. Feature extraction

For the reason that is already mentioned before, we also use corner point as features in each frames. Of course it will very good to track the frames with has full texture because they will contain more corner points. More number of feature will increase the accuracy but it is also increase the processing data and time. The important thing is to get the proper number of feature points which have a high stability in each frame.

One good example is Harris corner detection, which

$$c(x,y)=\sum_W [I(x_i,y_i)-I(x_i+\Delta x, y_i+\Delta y)]^2 \tag{1}$$

$$c(x,y)=\sum_W \left(I(x_i,y_i)-I(x_i,y_i)-[I_x(x_i,y_i)I_y(x_i,y_i)]\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix}\right) \tag{2}$$

$$c(x,y)=\sum_W \left([I_x(x_i,y_i)I_y(x_i,y_i)]\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix}\right) \tag{3}$$

$$c(x,y)=[\Delta x\ \Delta y]\begin{bmatrix}\sum w(I_x(x_i,y_i))^2 & \sum wI_x(x_i,y_i)I_y(x_i,y_i)\\ \sum wI_x(x_i,y_i)I_y(x_i,y_i) & \sum w(I_y(x_i,y_i))^2\end{bmatrix}\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix} \tag{4}$$

$$c(x,y)=[\Delta x\ \Delta y]C(x,y)\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix} \tag{5}$$

$I(x_i, y_i)$ is image pixel intensity at x-th column and y-th row. $c(x,y)$ captures the intensity of the local neighborhood. By analyzing the eigenvalue, it decide whether the pixel is a corner or not.

3. Feature tracking

This step is also known as corresponding problem, where we try to find and match the same feature point in two different frames. In this paper, we adopt the algorithm used in [], a coarse-to-fine method, Lucas-Kanade method using pyramid implementation.

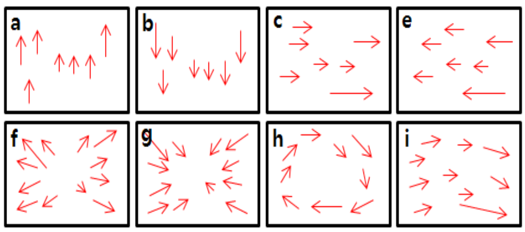

Figure 3. Example of the optical flow.
(a) Move up, (b) move down, (c) turn right, (d) turn left, (f) go forward, (g) go backward, (f) vertical rotation, (i) horizontal rotation.

The sets of feature pairs represent the optical flow which is used to analyze the camera movement. The 6 DoF optical flow and their combination can be seen in figure 3.

The next step is to mathematically model the camera movement. Here, our goal is not track the camera absolute position to the object which usually represent by camera extrinsic parameter as in [8], but get the relative movement from the initial position. Therefore, the camera intrinsic parameter (e.g, focal length, principal point, pixel aspect ratio, and skew) is not essential here.

The basic principles of most of algorithm proposed for this problem is try to estimate the best match which has the minimum fitting error. Unfortunately, in real condition, not all pairs are correct. We consider some of them may be a noise or error in calculation. In order to overcome this problem, we use Random sample consensus (RANSAC) method, an iterative random fitting method which also able to distinguish outlier.

RANSAC algorithm will generate 3x3 homography matrix which include the information of rotation and translation of the camera from one frame to the next frame. The results are then saved and used in the application.

III. Method Verification

The performance of the method proposed in this paper is verified by generating 3D map of camera movement relative to initial position and embedding 3D object in video sequences.

The video is captured using Canon camera digital Power Shot A710 7.1 mega pixels, which is mounted in a robot controlled remotely to move in an indoor environment. The video size is 320 x 240 pixels.

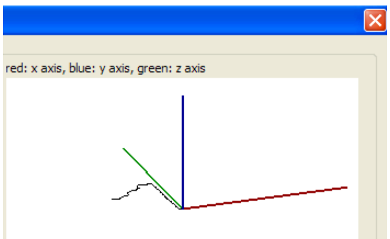

Figure 4. 3D map of video sequence in figure 5. The red line is x axis, blue line is y axis, green line is z axis (depth) and black line is camera movement.

Figure 5. Example of the experiment. (left) The sequences of original video and (right) is the video after 3D object insertion. The figure shows frame 1st, 162nd, 347th, 515th, 560th, 624th, 688th, and 767th frame where we can assume that the camera go straight, turn left and go forward again.

Figure 4 shows the camera movement of the video which is showed in figure 5. Here we assume that there is no information about camera initial position, therefore we set the initial position at the original point heading to the z axis. The 3D map result show that the method can be used to estimate the camera movement from the video where the camera move forward, turn left and go straight again as we can assume from figure 5.

Figure 5 show a 3D object inserted to the video, where the object initial position can be selected the first frame of video and the object position, size and appearance are automatically change depend on the camera movement.

Even the jitter cannot totally avoided practically, the result show that our method can be used to embed 3D object quite naturally.

Ⅳ. Conclusion & Future Work

We proposed a method to track the relative movement of camera from a video sequences by calculating the camera rotation and translation from one frame to the next frame. The homography relation between feature points in adjacent frame is mathematically compute using RANSAC algorithm. The result is then verified by creating camera movement 3D map and inserting 3D object into the video.

This work will be developed continually by adding more feature, thus the system can be used in more general situation, such as allow movement object pass over the video.

References

[1] A. D. Acqua, A. Sarti, S. Tubaro, "3D Motion from structures of points, lines, and planes," Image and Vision Computing, Vol. 26, No. 4, pp. 529-549, 2008.

[2] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," Proceedings of the 1981 DARPA Imaging Understanding Workshop, pp. 121-130, 1981.

[3] B. K. P. Horn and B. G. Schunck, "Determining optical flow," Artificial Inteligence 17, pp. 185-203, 1981.

[4] S. Gibson, J. Cook, T. Howard, et al., "Accurate camera calibration for off-line, video-based augmented reality," Proceedings. International Symposium on Mixed and Augmented Reality, pp. 37-46, 2002.

[5] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," International journal of Computer Vision 60, pp. 91-110, 2004.

[6] Jong-Sung Kim and Ki-Sang Hong, "A recursive camera resectioning technique for off-line video-based augmented reality," Pattern Recognition Letters, Vol. 28, No. 7, pp. 842-853, 2007.

[7] S. Thrun, W. Burgard, and D. Fox, Probabilistic Robotics : Intelligent Robotics and Autonomus Agents, Cambridge, MA, MIT Press, 2005.

[8] A. W. Fitzgibbon and A. Zisserman, "Automatic camera recovery for cloased or open image sequences," ECCV '98 Proceedings of the 5th European Conference on Computer Vision, vol.1, pp. 311-326, 1998.