Recovering Ground Depth From Single Surveillance Video For Feature Scale Normalization

Yang Cai

Robotics Institute
School of Computer Science
caiyang@andrew.cmu.edu

1 Introduction

In the problem of surveillance event detection, one important question is how to reasonably represent the events. Seeing the great success of local features(e.g. SIFT [4]) on visual recognition task in image domain, researchers extended the idea to video domain and developed the local spatial-temporal features, such as STIP[3], MOSIFT[6], etc. Even though event detection methods which are based on local spatial-temporal features have shown promising results, the absolute performance is still far from acceptable [1]. One important reason that causes the problem is that the spatial-temporal features still cannot accurately represent the events.

One reason that causes the inaccurate representation can be illustrated by Figure 1, which shows two key video frames that were captured by the same camera at the same position with the same pose. The person is walking at the same speed towards the same direction in both videos. The only difference is the depth of the two events. The red circles are the spatial-temporal interest points extracted from the videos and the circle size indicates the scale of motion of that feature points. It is obvious that the circles in the right video are much larger than the ones in the left. That means, from the feature points' view, the motion in the right video is much stronger. Here comes the problem: even though the underlying actions are similar, the feature representations of them are very different. As the essential reason that causes the problem is the depth difference, so if we know the depth of each feature points, then we can normalize the features into a same scale space and eliminate the representation difference. This is the motivation of the project.

However as surveillance camera can only offer us one view, recovering the depth for the whole image is very challenging. But the question is, for the feature normalization purpose, we are only required to know the depth of part of the image. The Figure 2 explains the reason. Because if we assume the person is a cylinder, then the depth of the feature points in the cylinder, can be approximated by the depth of where the cylinder and ground plane intersect. Namely the red eclipse. Because all the red eclipse is on the ground plane, then, what we need to recover is the depth of the ground plane. Therefore, in this project, instead of recovering the depth of the whole image, we can just estimate the depth of the ground plane, which is enough for the feature normalization and makes the problem more tractable as well.

2 Related Work

Although many works have been done on 3D reconstruction from 2D images, most of them require multiple images(e.g. structure from motion [2]). Because the surveillance camera is single and fixed, they cannot be directly applied to my problem. In [5], the authors proposed an approach to estimating depth from a single monocular image. They used learning approach to train Markov Random Field that models the depth at each individual points and depth relation between points. Different from [5], I try to attack the problem by pure geometrical approach and leverage the cues in a video which contains more information than an image.

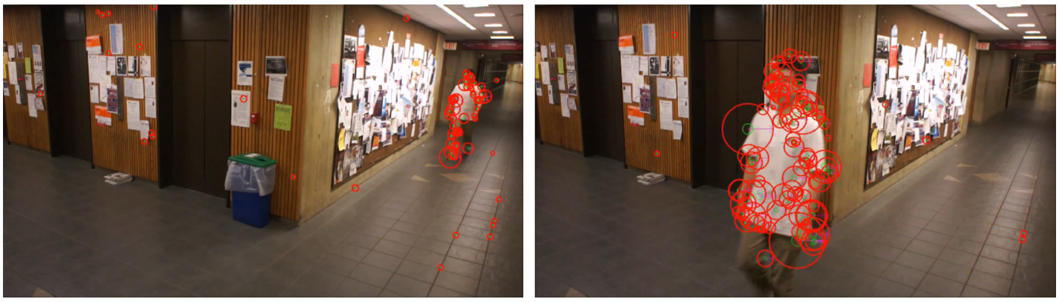

Figure 1: Two key frames of two videos with same event. The red points are extracted MOSIFT [6] features whose scale of motion are indicated by the size of circles.

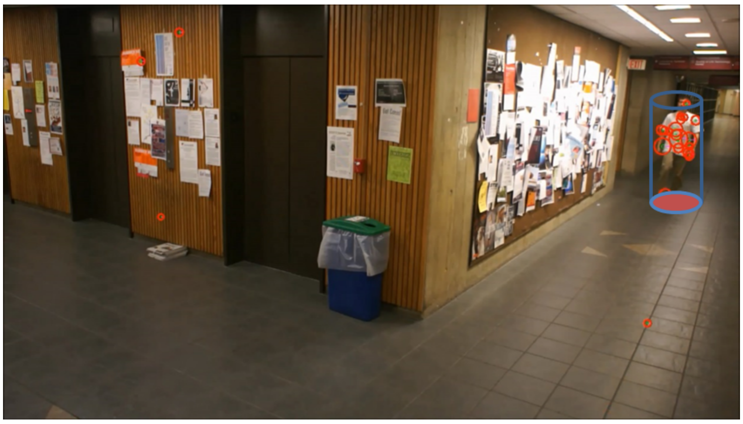

Figure 2: If we consider the person as a cylinder, then the depth of the feature points in the cylinder, can be approximated by the depth of where the cylinder and ground plane intersect.

3 Approach

In this section, I will first give you an overview of the approach to estimating the depth of ground plane used in this project. After that, I will specifically focus on one essential part of my approach which is camera calibration using surveillance video.

3.1 Overview

Instead of directly introducing the approach, I will first talk about the general idea behind the approach, as Figure 3 illustrated. Suppose we can first reconstruct the whole ground plane in the space, namely getting the normal direction of the plane. The only thing left is to find the correspondences between ground pixels in image and the corresponding points on the plane in space. To do that, we just need to get the rays that generate these pixels and find intersections of the rays and the plane. There are mainly fives steps in this approach, as introduced below.

- **Step 1. Camera calibration (K):** At the first step, we need to calibrate the camera. Namely, to get the intrinsic parameter K. I'll specifically introduce two methods used for camera calibration in the next section.
- **Step 2. Ground plane vanishing line detection (l_∞):** In my implementation, I used manual annotation to calculate the vanishing line of the ground plane. To do that, I first annotate two groups of parallel lines which are all parallel to the ground plane in image and then calculate the intersection of each group. Finally, the joint of the two intersections is the vanishing line of the ground plane.
- **Step 3. Ground plane normal direction reconstruction (n):** Once the K and l_∞ are known, we can directly reconstruct the normal direction of the ground plane by using fol-

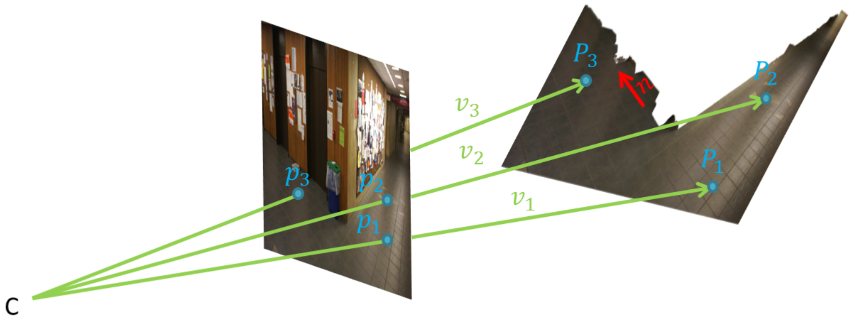

Figure 3: The overview of the proposed approach.

lowing relation:

$$n = K^T l_\infty.$$

- **Step 4. Ground pixel ray direction reconstruction(v):** To reconstruct the ray direction of ground pixels, we need to know two values: the intrinsic parameter K and the position of ground pixels p_i. In this project, to get the ground pixels, I didn't do the automatical ground detection, but used manual annotation to segment the ground plane from the image. Once we know K and p_i, we can directly reconstruct the ray directions of the ground pixels by using following relation:

$$v_i = K^{-1} p_i.$$

- **Step 5. Intersect the ray and reconstructed ground plane in space(P_i):** At the last step, we need to intersect the ray and the reconstructed ground plane in 3D space. As we know each ray goes through the camera center and the direction of the ray, so we can uniquely fix the ray in space and find its intersection with the ground plane in 3D space.

3.2 Camera calibration

3.2.1 Assumptions

The intrinsic parameter K of a camera is in the form of

$$K = \begin{bmatrix} \alpha_x & s & u_0 \\ 0 & \alpha_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}.$$

If we assume zero skew and square pixel, then we can simplify the form of K to

$$K = \begin{bmatrix} \alpha & 0 & u_0 \\ 0 & \alpha & v_0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Therefore, there are only three unknowns left in K. Under these assumptions, the image of absolute conic(IAC) ω should be in the form of

$$\omega = K^{-T} K^{-1} = \begin{bmatrix} \omega_1 & 0 & \omega_2 \\ 0 & \omega_1 & \omega_3 \\ \omega_2 & \omega_3 & \omega_4 \end{bmatrix}.$$

Even through there are four unknowns in ω, because it is up to scale, it is of degree of freedom 3. That means, as long as we have three or more than three equations, we can solve ω. Once we know ω, we can use Cholesky decomposition to get K. Therefore, in the following sections, I will introduce two methods I used to collect enough equations for solving K.

3.2.2 Calibrate the camera using three orthogonal vanishing points

If the camera happens to be fixed at a ”perfect” position, where we can observe three vanishing points corresponding to three orthogonal directions, as shown in Figure 4, we can easily get three equations.

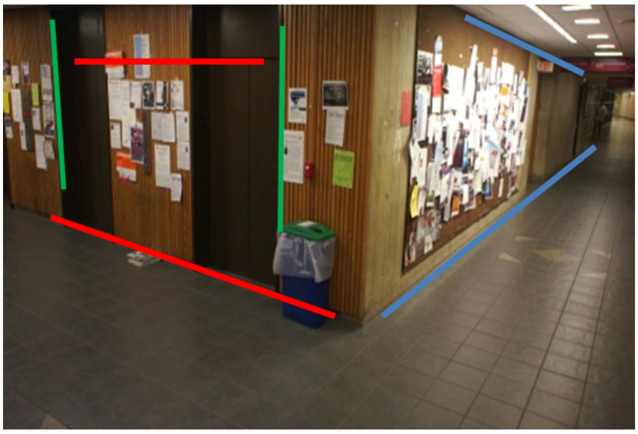

Figure 4: Three groups of parallel lines corresponding to three orthogonal directions.

Because each pair of orthogonal vanishing points (v_i, v_j) provides us one equation,

$$v_i^T \omega v_j = 0$$

and as we have three combinations, we can then get three equations.

However, the problem of this method is the surveillance cameras are not always put at such good position, where we can find simultaneously find three orthogonal vanishing points. In practice, we need to use some other methods to collect enough equations.

3.2.3 Calibrate the camera using moving objects in surveillance video

In surveillance video, even though the camera is fixed, but we can see a lot of moving objects in the view. For example, Figure 5(a) shows three frames of a surveillance video captured in a airport, where we can observe many moving objects, such as people, luggage cars, etc. More importantly, some of the objects have some very useful properties. Let take the vertical bar of the luggage car in Figure 5(b) as an example, we can list three useful properties of it. First, its perpendicular to the ground plane. Secondly, we have multiple observations of them in the video. Thirdly, their physical length wont change in space. Based on these properties, I propose my second method for camera calibration.

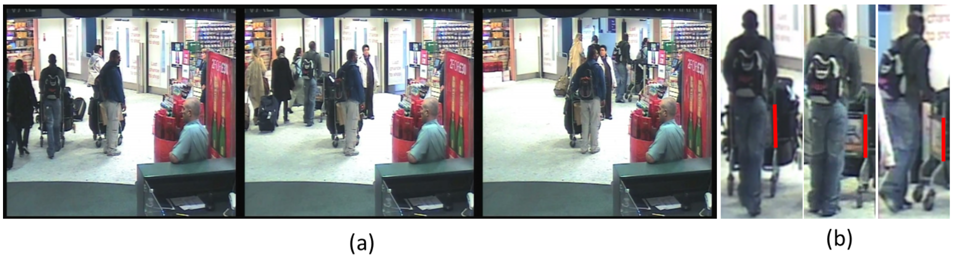

Figure 5: (a) Three frames of a surveillance video captured in a airport. (b) Observations of the same vertical bar of a luggage car in 3 key frames.

Figure 6 is an illustration of the camera calibration method using moving objects. Suppose the three red bars are three observations of the same objects at different positions. Then we can get two useful values based on these information. First of all, because the length the red bars are same in space, then

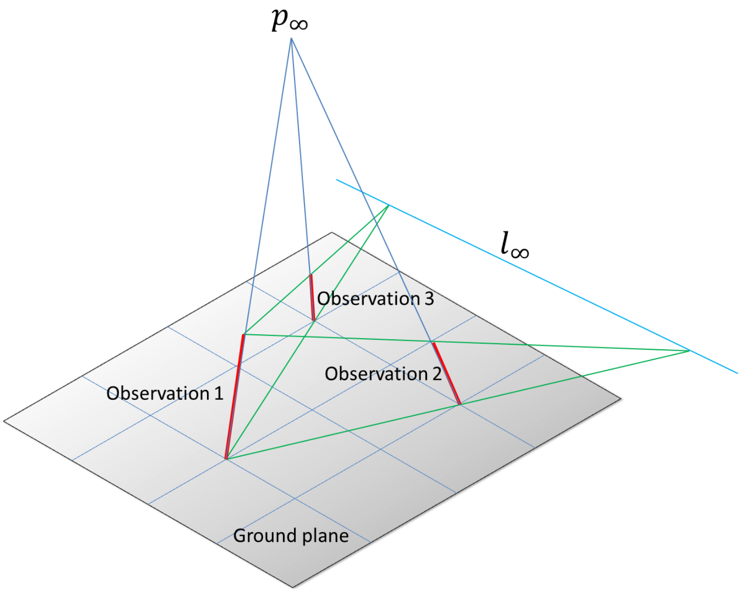

Figure 6: The illustration of the camera calibration using moving objects.

the each group of green lines are parallel in space and all of them are parallel to the ground plane. Using this, we can get the vanishing line l_∞ of the ground plane. Secondly, as all these red bars are perpendicular to the ground plane, then we can get the vanishing point p_∞ that is perpendicular to the ground plane. Once we know these two values, we can get two equations using this relation

$$l_\infty = \omega p_\infty.$$

4 Results

4.1 Recovered depth visualization

To qualitatively evaluate the proposed method, I visualize the recovered ground depth in this section. Figure 7 shows four examples where depth is encoded by using different color. The red means parts far from us while blue indicates parts near to us. Because the camera position is different and the depth range in the view is different, we can see different depth changing patterns across the image. For example, the two image in the middle, which contains a very deep corridor, the depth first changes very slowly at the beginning, while changes sharply at the end. For the last one, whose depth range is small, we can see the depth changes smoother than previous ones.

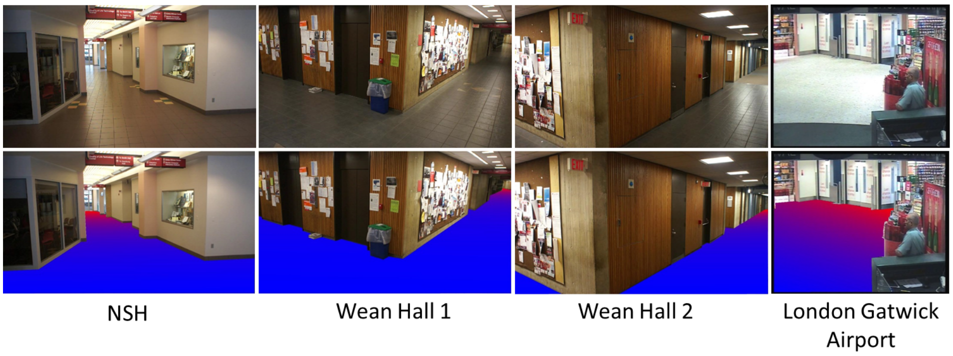

Figure 7: The visualization of the recovered ground plane depth.

4.2 Demo:InsertMe

In this section, I use a demo called ”InsertMe” to better quantitatively evaluate the recovered depth. This demo tries to insert a person template in to the picture and set the size reasonably. As a person’s size in image is inversely proportional to the person’s depth, we can compare the inserted person size and the actual size captured by the camera to see if the recovered depth is accurate. Figure 8 shows the insertion results on 4 key frames. The left column are key frames with the person of actual size while the right column shows the inserted template at the same place. Even though due to the pose difference, the inserted template cannot perfectly fit the actual person, the size of the inserted template is still reasonable. This indicates that the recovered depth is also accurate.

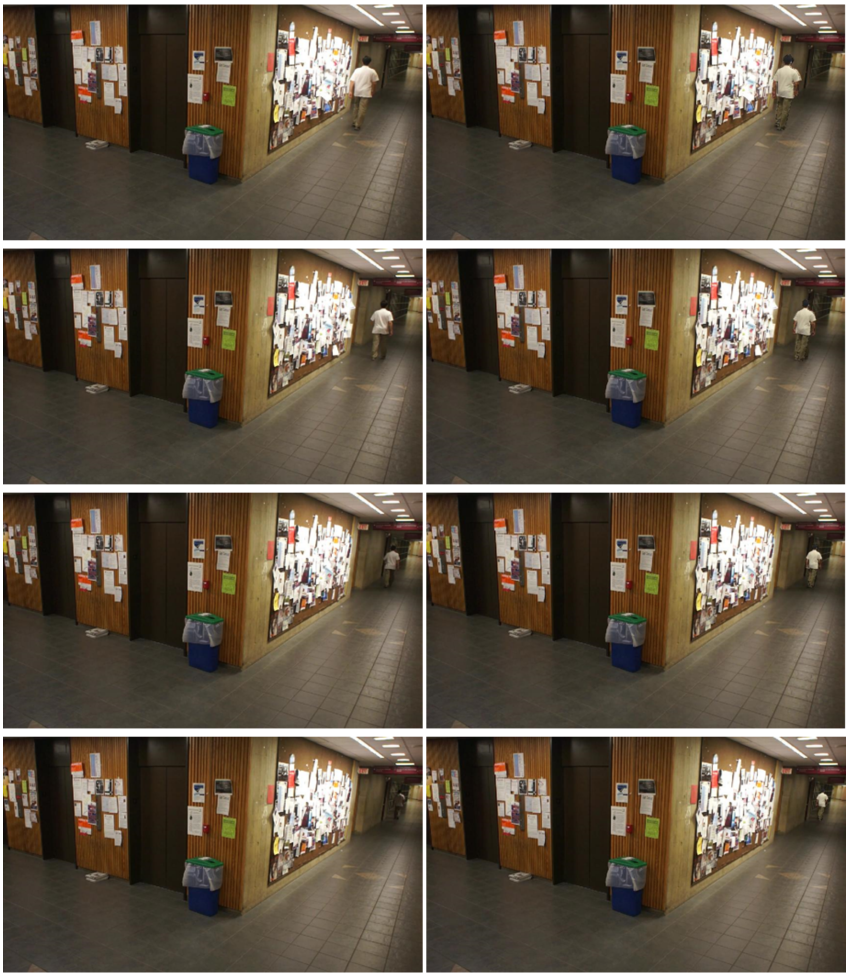

Figure 8: The left column are key frames with the person of actual size while the right column shows the inserted template.

5 Conclusion and Future Works

In this project, I implemented an approach to recovering the depth of ground plane. The general idea of the approach is straightforward and easy to implement. Specifically, observing some important characteristics of the surveillance videos, I studied one camera calibration method which utilized moving objects in a video. To evaluate the recovered depth, I first visualized the depth in image to get a qualitatively understanding and then designed a demo to better quantitatively evaluate the results. The results were reasonable in both experiments.

The future works involve in two aspects. First of all, current implementation involves a lot of manual annotations, which is not very efficient. However, some steps like ground plane pixel detection can be done automatically. I may need to do some survey on existing works in literature and replace the manual modules with automatical ones. Secondly, there are some numerical issues in current implementation. For example, some little error in ground plane vanishing line estimation may make the recovered depth very inaccurate. To solve the problem, I may leverage the large number of moving object observations and parameter estimation techniques like RANSAC to have more robust depth recovery.

References

[1] J. Aggarwal and M. Ryoo. Human activity analysis: A review. *ACM Computing Surveys(CSUR)*, 2011.

[2] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[3] I. Laptev. On space-time interest points. *International Journal of Computer Vision (IJCV)*, 2005.

[4] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 2004.

[5] A. Saxena, S. H. Chung, and A. Y. Ng. 3-d depth reconstruction from a single still image. *Int. J. Comput. Vision*, 2008.

[6] M. yu Chen and A. Hauptmann. Mosift: Reocgnizing human actions in surveillance videos. In *CMU-CS-09-161*, 2009.