StereoScan: Dense 3D Reconstruction in Real-time

Peirong Ji , pji@stanford.edu

June 7, 2016

1 INTRODUCTION

In this project, I am trying to replicate a published paper, StereoScan: Dense 3D Reconstruction in Real-time, by Andreas Geiger.

As a core subject in computer vision and robotics, 3D perception from stereo video is a practical problem and hence draws a lot of interest from researchers around the world. In the paper, the author proposes a new approach to build 3d maps from stereo sequences in real-time without expensive hardware involved. More specifically, the author combines both efficient stereo matching and multi-view linking scheme for generating consistent 3d point clouds. The experiment shows that the visual odometry part of the algorithm runs at 25 fps and the depth maps 3 to 4 fps, which is sufficient for online 3D reconstruction. This paper has three main significant contribution: real-time scene flow computation with multi-thousand feature matches, robust visual odometry algorithm proposed and solving the associated correspondence problem in 3D reconstruction in a greedy, accurate and efficient way. Even though stereo matching is part of the 3D reconstruction pipeline, it's, however, in the scope of another paper.

In this paper, we assume the stereo system moves smoothly and it mimics what a robot or human can see while it's moving. As for this project, a stereo sequence is downloaded from web[1]. This stereo sequence is calibrated and rectified, which simplifies system pipeline. This data set includes the stereo system parameters. Even though I cannot achieve the same performance as the author's, I am able to generate 3-4 frame per second. All the code in this project are carried out in C language and the only 3rd party library used is the libpng.

2 Related Work and Achievement

2.1 Related Work

Reconstructing 3D scene from stereo sequence is a core topic in computer vision and there are lots of related work. Here is a summary. The most naive one is what we explored in class, 3d voxel. The same idea can be applied to stereo 3D reconstruction, however, the complexity is high if we want to have a good resolution.
SLAM, short of Simultaneous Localisation and Mapping, is a process a mobile device building the 3D map around it and use the information to locate itself. However, for computation reason, most proposed approaches are only to handle very sparse sets of landmarks not dense 3D reconstruction, which is the topic of this paper.
3D reconstruction using classical Structure-from-Motion is also explored by lots of researchers. This technique requires a collection of images of a scene. It's not suitable for mobile stereo system moving continuously.

2.2 Achievement

In this paper, the resolution of 3D reconstruction is retained at the same time with the performance. This is achieved by an accurate egomotion estimation and high speed disparity map generator. The 3D reconstruction pipeline involves feature matching, egomotion estimation, stereo matching(scope of another paper) and 3D reconstruction. As the important parts of this paper, feature matching and egomotion estimation are both fully implemented in C and stero matching is integrated in the pipeline, while the 3D reconstruction is not finished yet.

3 3D RECONSTRUCTION SYSTEM

As shown in 3.1, the 3D reconstruction system is divided into 4 parts. They are feature matching, egomotion estimation, stereo matching and 3D reconstruction. As the first stage, the feature matching stage try to obtain the feature correspondences in four images, which are the left,right images of two consecutive frames. Based on the feature correspondences from the previous stage, the egomotion estimation stage produce the motion matrix of the stereo system, which includes the Rotation matrix R and the Translation vector T. Stereo matching stage is used to obtain dense disparity maps. During the implementation of this paper, I am going to integrate the ELAS library into the system to obtain the disparity map. The last step is the 3D reconstruction step, which creates consistent point-based models from the large amount of incoming data from the previous stages. The author propose a greedy approach which solves the association problem by re-projecting reconstructed 3D points of the previous frame into the image plane of the current frame. Details about each pipeline stage is detailed in the following subsections.

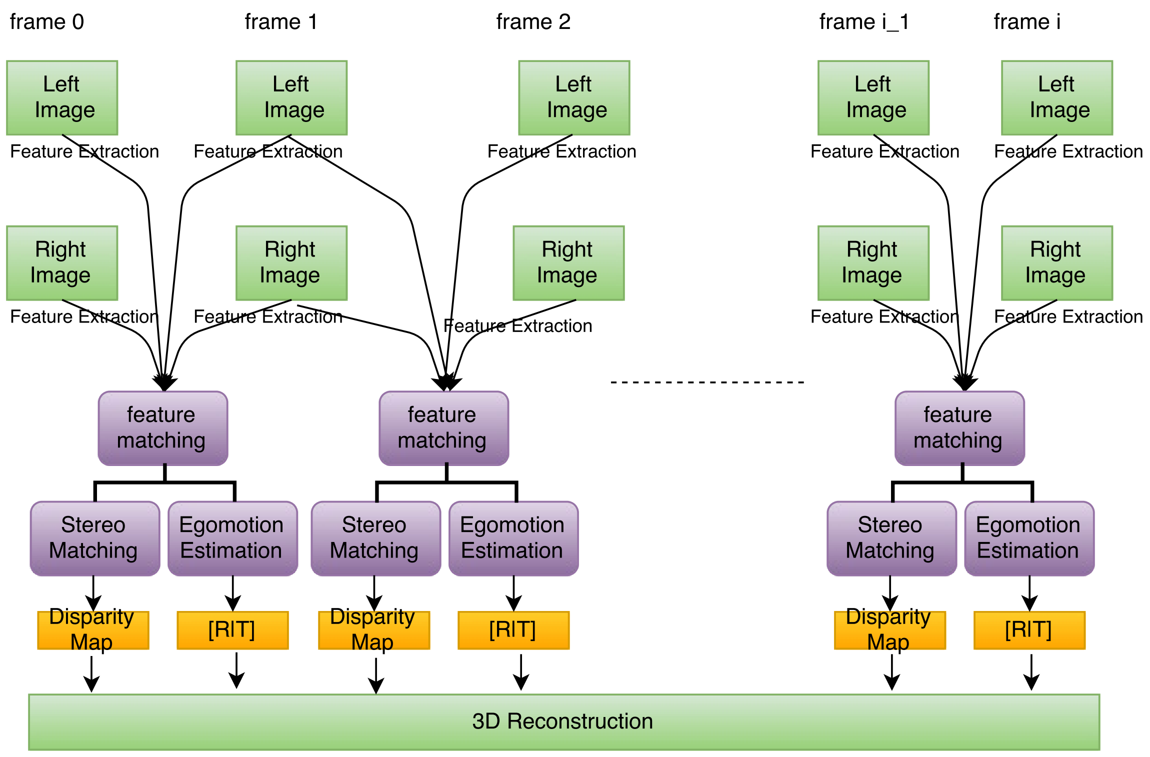
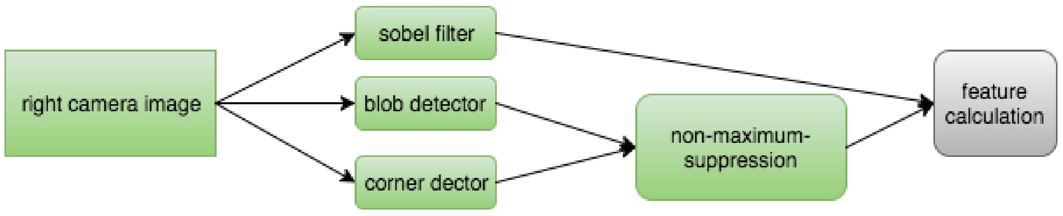


Figure 3.1: 3D Reconstruction System Pipeline

3.1 Feature Extraction and Feature Matching

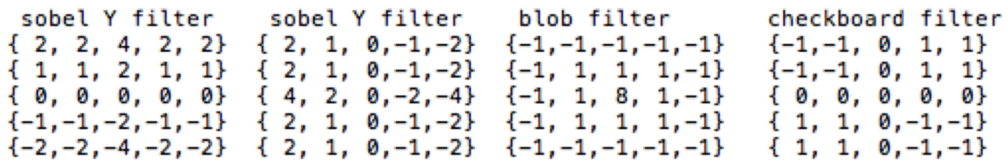
Supposing calibrated stereo setup and rectified stereo images, as the first stage of the pipeline, feature extraction and feature matching take 4 images, involving left and right images of consecutive frames, computes the feature of each image and find matching key points among them.



(a)

Figure 3.2: Feature Extraction

Taking the gray scale stereo images as input, we first calculate the sobel response, blob and corner response of the image. All the filters above are 5x5 and shown in 3.3. Using the output of the blob filter and checkboard filter, we run the non-maximum suppression to reduce the number of potential matching points. Typically, a 0.5M resolution image produces 9K points after the non-maximum suppression stage. Also, the points from the non-maximum suppression are divided into 4 groups, blob max, blob min, corner max and corner min. The feature matching stage only match the point within each group. This can save a lot of computation efforts. As for the feature vector of each potential matching point, they are obtained by concatenating the sobel response at the point, forming a 32x1 feature vector.

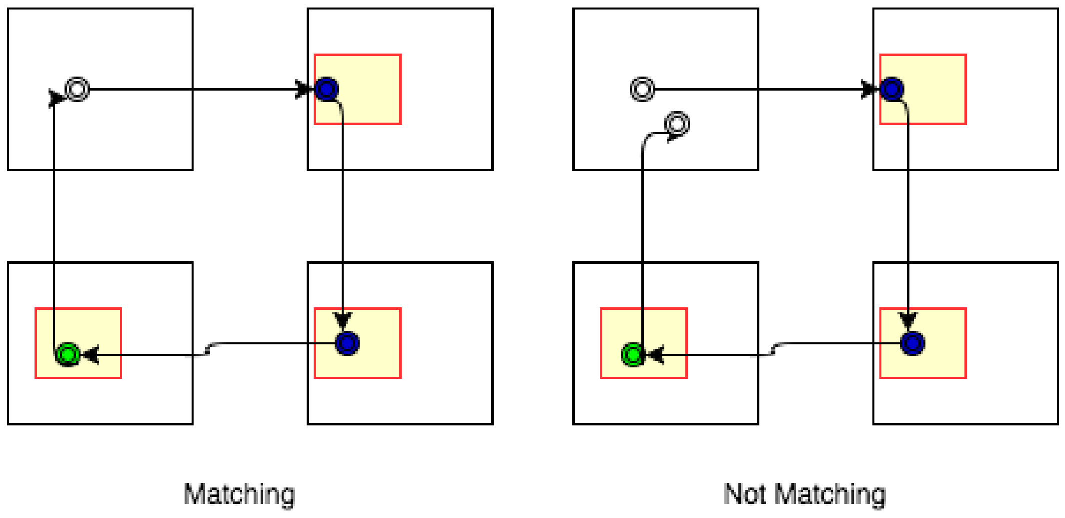


```
 sobel Y filter      sobel Y filter      blob filter        checkboard filter
{ 2, 2, 4, 2, 2}   { 2, 1, 0,-1,-2}   {-1,-1,-1,-1,-1}   {-1,-1, 0, 1, 1}
{ 1, 1, 2, 1, 1}   { 2, 1, 0,-1,-2}   {-1, 1, 1, 1,-1}   {-1,-1, 0, 1, 1}
{ 0, 0, 0, 0, 0}   { 4, 2, 0,-2,-4}   {-1, 1, 8, 1,-1}   { 0, 0, 0, 0, 0}
{-1,-1,-2,-1,-1}   { 2, 1, 0,-1,-2}   {-1, 1, 1, 1,-1}   { 1, 1, 0,-1,-1}
{-2,-2,-4,-2,-2}   { 2, 1, 0,-1,-2}   {-1,-1,-1,-1,-1}   { 1, 1, 0,-1,-1}
```

(a)

Figure 3.3: Feature Extraction

Above we have detailed the feature extraction stage, to find matching key points, we iterate all the feature points in left image of previous frame. For each feature point, we find the best matching point in the right image of the previous frame and use this best matching point to find the next best matching point in the right image of the current frame. So on and so on, we find the matching point in a loop formed by four consecutive images in every two consecutive frames. If this loop closes, then we state that we find a matching group across four images. Here since we take the assumption that the stereo system is moving smoothly, so we can find the matching point in a bounding box, which is 50x50 around the feature point. This procedure is depicted below.



(a)

Figure 3.4: Loop Matching in Four Images

After feature extractor and feature matching, we get a list of matching points of four images. These points are served as input of egomotion estimation and disparity map generator. Typically, we got around 2K key points for each image.

3.2 Egomotion Estimation

Given the matching of the previous frame, we can calculate the matching points' 3D location in previous stereo system coordinate via triangulation.

$$d = max(u_{pl} - u_{pr}, 0.001) \tag{3.1}$$

$$z_p = f * STEREO\_BASE/d \tag{3.2}$$

$$y_p = (v_{pl} - v_{pr}) * STEREO\_BASE/d \tag{3.3}$$

$$x_p = (u_{pl} - u_{pr}) * STEREO\_BASE/d \tag{3.4}$$

Equation 3.1 denotes d as the disparity of matching pair in the previous left and right images. (x_p, y_p, z_p) is the matching point's 3D location in the previous stereo system coordinate. Supposing $R(r)$ is the rotation matrix and T is the translate matrix that depicts the stereo system's rotation and movement since the previous frame. f is the camera focus length and $STEREO\_BASE$ is the stereo base. Assuming squared pixels and zero skew, (x_p, y_p, z_p) can be projected into the current image planes using Eq.3.5.

$$\begin{pmatrix} u_c \\ v_c \\ 1 \end{pmatrix} = \begin{pmatrix} f & 0 & c_u \\ 0 & f & c_v \\ 0 & 0 & 1 \end{pmatrix} \left(\begin{pmatrix} r_{00} & r_{01} & r_{02} \\ r_{10} & r_{11} & r_{12} \\ r_{20} & r_{21} & r_{22} \end{pmatrix} \begin{pmatrix} x_p \\ y_p \\ z_p \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} - \begin{pmatrix} s \\ 0 \\ 0 \end{pmatrix} \right) \tag{3.5}$$

When projecting the 3D point in previous coordinate into current image plane, in right image, s is the baseline, in left image, $s = 0$. Let's denote π as a mapper which given a point (x_p,y_p,z_p) and projects it into a 2D plane under the parameter r_{ij} and (t_x, t_y, t_z). Then to estimate the stereo system motion during the past one frame, we try to obtain a pair of r_{ij} and (t_x, t_y, t_z), which minimizes 3.6. u_l and v_l denotes the coordinates in the current left image plane of the 3D points in previous frame.

$$\begin{aligned}\sum_{i=1}^{3} \| \begin{pmatrix} u_i^l \\ v_i^l \end{pmatrix} - \pi^l(\begin{pmatrix} x_p \\ y_P \\ z_p \end{pmatrix}; R, T) \|^2 + \| \begin{pmatrix} u_i^r \\ v_i^r \end{pmatrix} - \pi^r(\begin{pmatrix} x_p \\ y_P \\ z_p \end{pmatrix}; R, T) \|^2 \\ = \sum_{i=1}^{3} (u_i^l - u^l)^2 + (v_i^l - v^l)^2 + (u_i^r - u^r)^2 + (v_i^r - v^r)^2\end{aligned} \tag{3.6}$$

This optimization problem can be solved using Gauss-Newton iteration. The residues is a 12x1 vector, which denotes the distance between the points in the current frame images and the coordinate obtained via re-projection in x coordinate and y coordinate respectively. To do this optimization, we need the Jacobians matrix of the residues. Let's denote the residues vector as $r\left(\beta\right) = (r_0, r_1, r_2, r_3, .., r_{11})$ and parameter vector as $\beta = \left(r_x, r_y, r_z, t_x, t_y, t_z\right)$. Then the Jocobians Matrix is 3.8

$$J = \begin{pmatrix} \frac{\partial r_0}{\partial r_x} & \frac{\partial r_0}{\partial r_y} & \frac{\partial r_0}{\partial r_z} & \frac{\partial r_0}{\partial t_x} & \frac{\partial r_0}{\partial t_y} & \frac{\partial r_0}{\partial t_z} \\ \frac{\partial r_0}{\partial r_x} & \frac{\partial r_0}{\partial r_y} & \frac{\partial r_0}{\partial r_z} & \frac{\partial r_0}{\partial t_x} & \frac{\partial r_0}{\partial t_y} & \frac{\partial r_0}{\partial t_z} \\ \dots\dots \\ \frac{\partial r_1 1}{\partial r_x} & \frac{\partial r_1 1}{\partial r_y} & \frac{\partial r_1 1}{\partial r_z} & \frac{\partial r_1 1}{\partial t_x} & \frac{\partial r_1 1}{\partial t_y} & \frac{\partial r_1 1}{\partial t_z} \end{pmatrix} \tag{3.7}$$

The parameter iteration can be expressed as:

$$\beta^i = \beta^{i-1} - (J^T J)^{-1} J^T r(\beta^{i-1}) \tag{3.8}$$

The pseudo code below details the procedure of egomotion estimation.

Listing 1: Egomotion Estimation

```
1. Calculate the matching points' 3D location (x,y,z)
   in previous stereo system coordinate
2. for i=1:50    // RANSAC x50
     random pick 3 matching points from feature matching stage
     initialize $\beta^0 = (0,0,0,0,0,0)$
     for j=1:20
         calculate Jacobian Matrix J
         $\beta^i = \beta^{i-1} - (J^T J)^{-1} J^T r(\beta^{i-1})$
         // if converge
         if $(J^T J)^{-1} J^T r(\beta^{i-1}) < \delta$ :
             break
     end
     calculate inliers using $\beta$ and save the best $beta$
   end
```

The accuracy of the egomotion matrix is critical as we are going to use it to re-project all the 3D cloud point into a common coordinate system. As you will see in the experiment, we achieved pretty good accuracy.

3.3 Stereo Matching[1]

Stereo matching stage takes the output of the feature matching stage and builds a disparity map for each frame. In this paper, the disparity map is built with a free library called ELAS[2]. ELAS is a novel approach to binocular stereo for fast matching of high resolution imagery. It builds a prior on the disparities by forming a triangulation on a set of supporting points which can be robustly matched, reducing matching ambiguities of the remaining points. This allows for efficient exploitation of the disparity search space, yielding accurate and dense reconstructions without global optimization.

3.4 3D Reconstruction

The simplest method to point-based 3d reconstructions map all valid pixels to 3d and projects them into a common coordinate system using the egomotion matrix. However, if we keep storing the incoming points every frames, then the storage requirement will grow rapidly. In this paper, the author proposes a greedy approach which reprojecting reconstructed 3d points of the previous frame into the image plane of the current frame. In case a point falls onto a valid disparity then we fuse both 3D points by taking the mean of them. This not only improves the accuracy of reconstruction but also reduces the memory requirement quite a lot.

4 EXPERIEMENT

In this section, results of previous section are shown. Fig.4.1 shows the result of the feature matching. The dot point is a matched points in the left image of previous frame and the line pointing to the location of the matched points in the left image of the current frame. Around 2K points are matched among four consecutive images. As for egomotion estimation, the

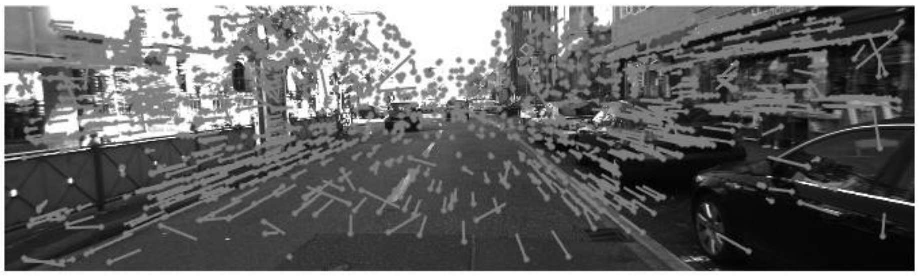


Figure 4.1: Feature Matching Among Four Consecutive Images

video can be accessed in on web. Please follow the link https://www.youtube.com/watch?v=xvqmTMpK87E . As you can see in the video, the accumulated rotation and translation matrix follow the stereo system accurately. The disparity map can be viewed on web, please following the link $https://www.youtube.com/watch?v=LJF0hh\_rgvc$. Unfortunately, I don't have a project partner nor enough time to merge all the stage together and output the 3D reconstruction map.
The performance of each pipeline stage are given below.

Stage	Time
Feature Matching	90ms
Egomotion Estimation	35ms
Disparity Map	350ms

5 CONCLUSION

In this project, I learn a lot about feature extraction, feature matching and egomotion estimation and lots of 3D reconstruction concept. All the code are implemented in C language, including image convolution, linear equation elimination, Gauss-Newton iteration and RANSAC and etc. Even though the performance here is acceptable, it can be boost using SIMD instruction.
Finally, my code is located at https://www.dropbox.com/s/qhmedzb4n3qqjrt/finalprojectcode.zip?dl=0

6 REFERENCE

[1] Andreas Geiger, Julius Ziegler and Christoph Stiller, "StereoScan: Dense 3D Reconstruction in Real-time", Intelligent Vehicles Symposium (IV),2011
[2] Andreas Geiger, Martin Roser and Raquel Urtasun, "Efficient Large-Scale Stereo Matching", Asian Conference on Computer Vision (ACCV),2010