**CSE-483: Mobile Robotics** **Monsoon 2019**

# Stereo Geometry

Disparity, Rectification and Point Cloud Generation

Prepared by: Pranay Gupta & Kanav Gupta

## Contents

## 1 Introduction

In these notes we discuss, disparity, rectification and point cloud generation for stereo images. We will start with a brief intro to stereo vision and will further provide mathematical explanation for the above mentioned topics.

## 2 Stereo Vision

Stereo Vision is extraction of 3D information from the digital images by comparing about a scene from two vantage points. Stereo Vision can be used for three major purposes:

- **Correspondence**: Given a point in one image, find the same point in some other image of same scene
- **Camera Geometry**: Given corresponding points in two images find the camera matrix, position and pose of camera
- **Scene Geometry**: Find coordinates of 3D point from its projection in two or multiple images

In stereo vision images can be obtained using multiple cameras or one moving camera. Ideal Stereo setup has two cameras which are identical and aligned. This setup provides complete information about the 3D point whose image is formed in both the cameras. Term **binocular vision** is used for such setup.

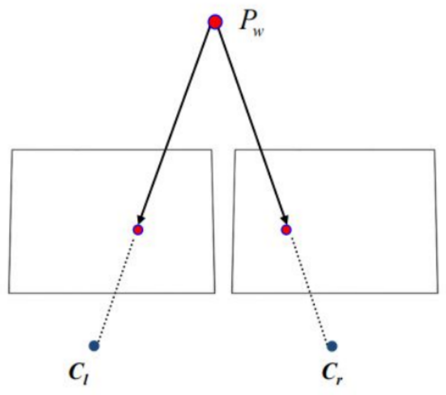


Figure 1: Stereo Camera Pair

## 3 Disparity

In this section we will discuss what is disparity and how to calculate it.

### 3.1 Definition

**Binocular disparity** refers to the difference in image location of an object seen by the left and right eyes, resulting from the eyes horizontal separation. In computer vision, binocular disparity refers to the difference in coordinates of similar features within two stereo images.

The disparity of features between two stereo images are usually computed as a shift to the left of an image feature when viewed in the right image. For example, a single point that appears at the x coordinate t in the left image may be present at the x coordinate t-3 in the right image. In this case, the disparity at that location in the right image would be 3 pixels.

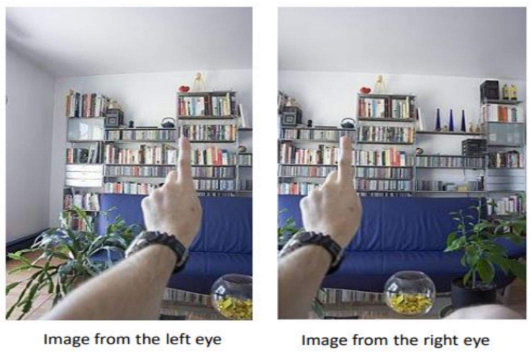


Figure 3: Images scene from left eye and right eye respectively

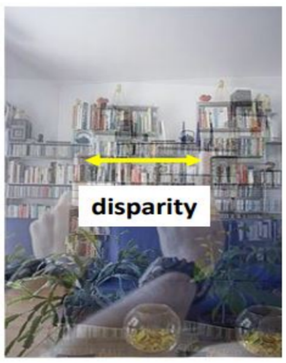
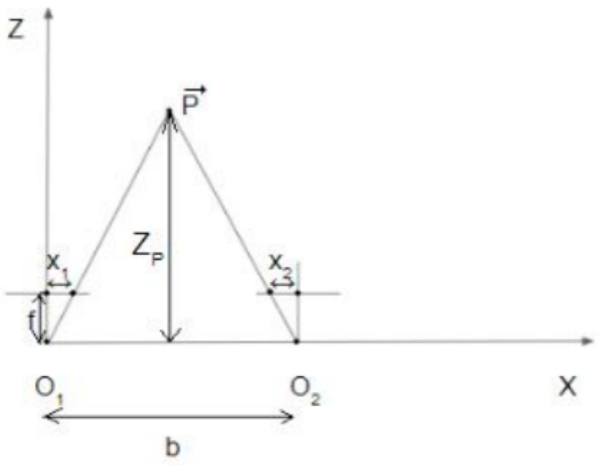


Figure 4: Disparity

Stereo images may not always be correctly aligned to allow for quick disparity calculation, in those cases we first need to align the images through a process called *image rectification*, so that the disparity only lies in the horizontal direction.

## 3.2 Disparity Calculation

### 3.2.1 Ideal Case



In the ideal case we assume that the optic axis of the two stereo cameras are parallel.

Taking Origin at O1, and $\vec{P} = (X_p, Y_p, Z_p)$ is the point to be considered. Using same ratio properties of similar triangles,

$$\frac{f}{Z_P} = \frac{x1}{X_P}$$

Also,

$$\frac{f}{Z_P} = \frac{-x2}{b - X_P}$$

On solving above equations, we get,

$$Depth Z_P = \frac{bf}{x1 - x2}$$

$$Disparity = x1 - x2$$

### 3.2.2 General Case

In the general case, the optic axis of the two cameras might not be parallel.

Taking Origin at O1, and $\vec{P} = (X_p, Y_p, Z_p)$. We assume that the rays are formed by back projecting $x_1$ and $x_2$ intersect at the point $\vec{P}$.

Camera 1, Projection Matrix $P_1 = K_1(I|0)$
Camera 2, Projection Matrix $P_2 = K_2(R|T)$

$\lambda_1 x_{1h} = P_1 \vec{P_h}$
$\lambda_2 x_{2h} = P_2 \vec{P_h}$

Since the cross product of parallel lines is 0

$\therefore x_{1h} P_1 \vec{P_h} = 0$ and $x_{2h} P_2 \vec{P_h} = 0$

$$\implies \left[x_{1h}\right]_x P_1 \vec{P_h} = 0 \text{ and } \left[x_{2h}\right]_x P_2 \vec{P_h} = 0 \qquad (1)$$

where, $\left[x_{1h}\right]_x = \begin{bmatrix} 0 & -1 & y_1 \\ 1 & 0 & -x_1 \\ -y_1 & x_1 & 0 \end{bmatrix}$

In equation 1, both the equations contribute 2 equations each. So we have 4 equations and 3 unknowns for each pixel pair. Then we can solve for $\vec{P_h}$ using SVD, and taking the vector corrresponding to the smallest eigen value.

Now, disparity for each point can be calculated using, $d_i = \frac{bf}{Z_{Pi}}$

If the rays dont intersect, non-linear approach like LM algorithm can be used to minimize reprojection error. loss $= \min((x_{11}(\vec{P}))^2 + (x_{22}(\vec{P}))^2)$

Result of this linear approach can be used as initialization for the non-linear approach

## 4 Rectification

In this section we will learn about image rectification in stereo images and method to do the same.

## 4.1 Definition

Image Rectification is the process of re-sampling pairs of stereo images taken from widely differing viewpoints in order to produce a pair of “matched epipolar projections”. These are projections in which the epipolar lines run parallel with the $x$-axis and match up between views, and consequently disparities between the images are in the $x$-direction only, i.e. there is no $y$ disparity.

In other terms, given a pair of stereo images, intrinsic parameters of each camera and extrinsic parameters of the system, image rectification is computation of image transformation that makes epipolar lines collinear and parallel to $x$-axis

Image Rectification is widely used in stereo vision, some uses are mentioned below:

- This speeds up matching because searching horizontal epipolar lines is easier than general epipolar lines
- Rectification is a good pre-processing step if you want to do correspondence faster
- Correlation based correspondence algorithms always assume a simple stereo configuration

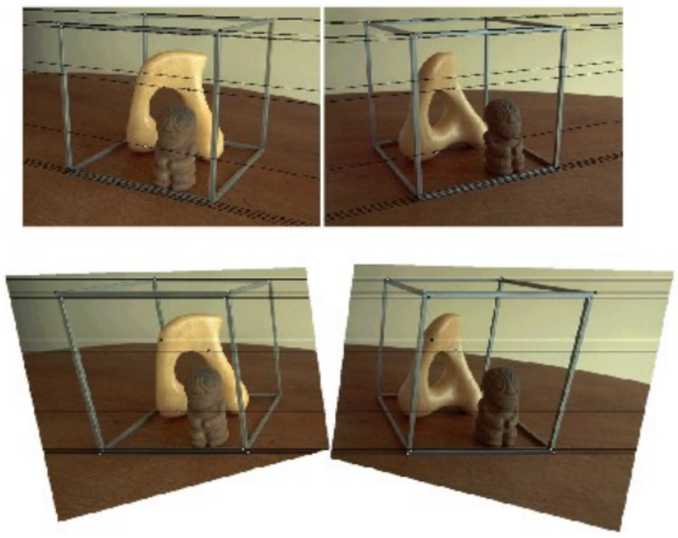


Figure 2: Image pairs - Before and After Rectification

### 4.2 Rectification Algorithm

Algorithm used for image rectification assumes that we have information about Rotation matrix and camera center for both the cameras is known and has four major steps:

1. rotate the left camera so that the epipole goes to infinity along the horizontal axis.
2. Apply the same rotation to the right camera to recover the original geometry.
3. Rotate the right camera by R
4. Adjust the scale in both camera reference frame

To carry out this method, we construct three mutually orthogonal unit vectors $e_1, e_2, e_3$. Since the problem is under constrained, we can make an arbitrary choice. The first vector $e_1$, is given by the epipole, since the image center is in the origin, $e_1$ coincides with the direction of translation, in mathematical terms

$$e_1 = \frac{T}{||T||}$$

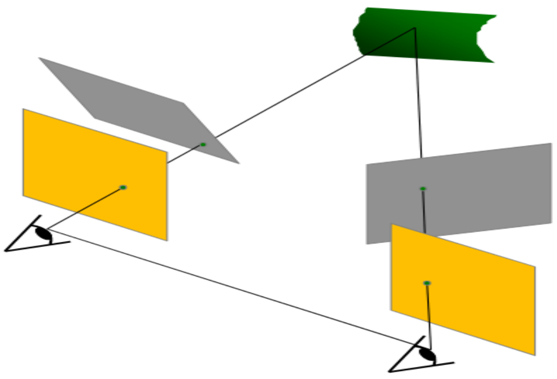


Figure 3: Stereo Image Rectification

where $T$ represents the vector of the baseline Now $e_2$ is has a constraint that it must be orthogonal to vector $e_1$. To solve this purpose we compute and normalize cross product of $e_1$ with the old $z$-axis of left camera, to obtain

$$e_2 = r_3 \times e_1^T$$

Now third uint vector $e_3$ can be easily obtained by computing cross product of $e_1$ and $e_2$, So

$$e_3 = e_1 \times e_2$$

Now it is easy to check that othogonal matrix defined as

$$R_{rect} = \begin{bmatrix} e_1^T \\ e_2^T \\ e_3^T \end{bmatrix}$$

rotates the left camera about the projection center in such a way that epipolar lines become parallel to horizontal axis. This is implementation of first step of the rectification algorithm. Since remaining steps are pretty straightforward, customary algorithm is given below

Note: Input of algorithm is formed by intrinsic and extrinsic parameters of stereo system and set of points in each camera to be rectified.

1. Build $R_{rect}$ matrix using above described method
2. Set $R_l = R_{rect}$ and $R_r = RR_{rect}$
3. For each left camera point, $p_l = [x, y, f]^T$ compute

   $$R_l p_l = [x', y', z']$$

   and coordinates of corresponding rectified point, $p'_l$

   $$p'_l = \frac{f}{z'}[x', y', z']$$

4. Repeat above steps for right camera using $R_r$ and $p_r$

The output is the pair of transformations to be applied to two cameras in order to rectify the two input point sets,as well as the rectified sets of points.

## 5 3D Point Cloud

### 5.1 Disparity Map

Disparity Map refers to the apparent pixel difference or motion between a pair of stereo images.

**Exercise:** Try closing one of your eyes and then rapidly close it while opening the other. Objects that are close to you will appear to jump a significant distance while objects further away will move very little. That motion is the disparity.

Procedure for calculating disparity map:

1. For each pixel on left image calculate corresponding point on right image
2. Compute disparity for each pair using method described in section 3
3. Plot the difference as intensity on an image to visualize the disparity map.
4. This map can be used to generate 3D point cloud as discussed in next section

## 5.2 3D Point Cloud generation

As, we know from section 3, the depth $Z_p$ is given by,

$$\frac{bf}{x1 - x2}$$

and the disparity is given by $x1 - x2$. Thus, the depth at various scene points may be recovered by knowing the disparities of the corresponding image points.

Also, we can compute:

$$x = \frac{b(x1 + x2)}{2(x1 - x2)}$$

$$y = \frac{b(y1 + y2)}{2(x1 - x2)}$$

So, having these x and y we can compute the point cloud.

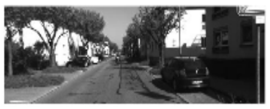


Figure 12: Left camera image

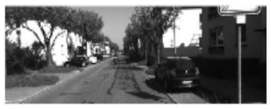


Figure 13: Right camera image

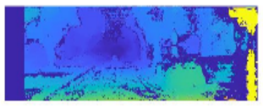


Figure 14: Disparity Map

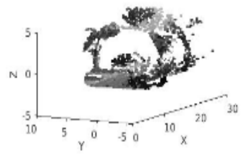


Figure 15: Point Cloud

## References

[1] MAndrew Zisserman Richard Hartley. Multiple View Geometry in computer vision. Cambridge University Press, New York, 2009.

[2] Wikipedia - Computer Stereo Vision

[3] Wikipedia - Binocular Disparity

[4] Trucco and Verri, Introductory Techniques for 3D Computer Vision

[5] nternational Journal ”Information Content and Processing”, Volume 3, Number 2, 2016. Point cloud registeration and generation from stereo images, Aram Gevorgyan Vladimir

[6] Lecture Slides