Guangcong Zheng
Student
North Eastern University
13012184221
zgc_troy@163.com

Stereo Project Report

Due Friday, June 14, 2019

Contents

Introduction

Before displaying the result of project, i am going to talk about my preparation and way of thinking when i begin to work on this project.

1. **determine the goal and requirements of the project**

   the goal of the project is to show personal ability in all aspects

   - the ability to search for information
   - the ability of programming
   - the ability to learn new knowledge in a short time
   - the ability to conclude the results and write report papers

   the requirements of the project:

   - the code with its documents
   - the report containing the answer for the problems, the results that of great importance, and the way of thinking

2. **determine the direction and the subject of research by reading the materials provided**

   The project is about **depth estimation using multiple-view geometry** along with some of its basic knowledge such as **monocular camera calibration and undistortion, stereo calibration and rectification**. Obviously, it is a **Computer Vision** project.

3. **evaluate the difficulty of the problems**

   The problem with $\star\star\star$ are around the level that of implementing a classic algorithm from the bottom layer, reproducing a top-level paper, using cuda to refactor a C++ code. Considering the lack of time and my ability, i will not try to solve the problems with three stars until finishing all the two stars problems.

4. **learn more about the background of research and needed basic knowledge**

   Some materials and tutorials have already been provided in the pdf, for example, opencv calibration document [1], the epipolar tutorial in CS543 course of University of Illinois [2], the CSDN blogs [3]. So it is a good way to start with these Computer Vision courses and recommended csdn blogs. According to the moverzp's blog 'Binocular camera calibration parameters' [3], i find its series of blogs such as 'Monocular Camera Calibration Parameters Description[4]. According to opencv calibration document [1], i find the opencv samples in github [5] , programming tutorials about camera calibration [6], epipolar geometry [7], and depth estimation using SGBM [8].

Basic Knowledge

Homogeneous coordinates

- converting to homogeneous image coordinates

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{1}$$

- converting to homogeneous scene coordinates

$$(x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2}$$

- converting from homogeneous image coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w) \tag{3}$$

- converting from homogeneous scene coordinates

$$\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Rightarrow (x/w, y/w, z/w) \tag{4}$$

- $w = 0$ means a infinity point
- A line $ax + by + c = 0$ is represented by the homogeneous vector below (projective line):

$$line = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \tag{5}$$

- Any vector $k \begin{bmatrix} a \\ b \\ c \end{bmatrix}$ represents the same line
- Only the ratio of the homogeneous line coordinate is significant:
- The point x lies on the line l : $x^T l = 0$
- Two points p, q define a line: $l = p \times q$
- Two lines l, m define a point p: $p = l \times m$

Monocular Camera Calibration

Problem 1

What are the intrinsics and extrinsics of a camera? (Learn about camera model in [9]) What is the camera matrix? Write down your answers.

(a) What are intrinsics and extrinsics?

Answer

- **Intricsics**

  The intricsics of a camera can be seen as attributes that are decided by the camera itself and do not depend on the scene viewd. So once estimated, they can be reused as long as the components of a camera are fixed.

  In the linear camera model, the effect of intrinsics is defined as as a matrix called **the intrinsic matrix** K, which is used to denote a projective mapping from **3D camera coordinates** to **Pixel coordinates**.

$$K = \begin{bmatrix} f_x & \gamma & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{6}$$

  The parameters $f_x = f \cdot \alpha_x$ and $f_y = f \cdot \alpha_y$, where f is the **focal length** of the camera, where α_x and α_y are the **scale factor**s relating pixels to distance.

  Nonlinear intrinsic parameters such as **lens distortion** are also important although they cannot be included in the linear camera model described by the intrinsic parameter matrix. We will talk about it later in Problem 5.

- **Extrinsics**

  The extrinsics of a camera corresponds to the relationship between the camera and the outer environment, or specifically, denotes the coordinate system transformations from **3D world coordinates** to **3D camera coordinates**.

  The effect of extrinsics can be represented as a matrix consists of a rotation matrix R and a transformation matrix T. The joint rotation-translation matrix G is called a matrix of extrinsic parameters. It is used to describe the camera motion around a static scene, or vice versa, rigid motion of an object in front of a still camera.

$$G = \begin{bmatrix} R_{3\times3} & T_{3\times1} \\ 0_{1\times3} & 1 \end{bmatrix}_{4\times4} \tag{7}$$

(b) What is camera matrix?

Answer

In general, a **camera projection matrix** P is used to denote a **projective mapping** from **World coordinates** to **Pixel coordinates**.
Let x be a representation of a 3D point in homogeneous coordinates (a 4-dimensional vector), and let y be a representation of the image of this point in the pinhole camera (a 3-dimensional

Problem 1 continued on next page...

vector). Then the following relation holds

$$y \sim Px \tag{8}$$

$$\begin{aligned} P &= K\,[I|0]\,G \\ &= K\,[R|T] \end{aligned} \tag{9}$$

The pinhole camera model can be defined as follow:

$$Z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K\,[R|T] \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{10}$$

Problem 2

(Camera Imaging) Given an ordinary camera with focal length f_x and f_y, optical center (c_x, c_y), the transform between the 3D world coordinates and the 3D camera coordinates is $(R|t)$, can you transform a 3D point $X = (X_1, X_2, X_3)$ onto the image plane? (Consider only pinhole camera model. 3D camera coordinate position: front is $+Z$, left is $+X$, up is +Y. Image plane: right is $+x$, down is $+y$). Write down your answers in matrix form.

(a) transform a 3D point $X = (X_1, X_2, X_3)$ onto the image plane

Answer

1. 3D World coordinates $(X_1, X_2, X_3, 1) \to$ 3D Camera coordinates (X_c, Y_c, Z_c)

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R_{3\times3} & T_{3\times1} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ 1 \end{bmatrix} \tag{11}$$

2. 3D Camera coordinates $(X_c, Y_c, Z_c) \to$ 2D Pixel coordinates (x_p, y_p, w)

$$\begin{bmatrix} x_p \\ y_p \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} \tag{12}$$

3. 3D World coordinates $(X_1, X_2, X_3, 1) \to$ 2D Pixel coordinates (x_p, y_p, w)

$$\begin{bmatrix} x_p \\ y_p \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R_{3\times3} & T_{3\times1} \\ 0_{1\times3} & 1 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ 1 \end{bmatrix} \tag{13}$$

$$w = Z_c \tag{14}$$

4. 2D Pixel homogeneous coordinates $(x_p, y_p, w) \to$ Euclidean coordinates (u, v)

$$w\begin{bmatrix}u\\v\\1\end{bmatrix} = \begin{bmatrix}f_x & 0 & c_x & 0\\0 & f_y & c_y & 0\\0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}R_{3\times3} & T_{3\times1}\\0_{1\times3} & 1\end{bmatrix}\begin{bmatrix}X_1\\X_2\\X_3\\1\end{bmatrix} \tag{15}$$

$$(u,v) = \left(\frac{x_p}{w}, \frac{y_p}{w}\right) \tag{16}$$

Problem 3

Given a 2D image point (u,v), what shape does it correspond to in the 3D camera coordinate? Can you derive its equation?

(a) what shape does it correspond to in the 3D camera coordinate?

Answer

a optical ray, which is a line in 3D space that passes the optical center and 2D image point (u, v)

(b) Can you derive its equation?

Answer

Given a 2D point $m = (u, v)$, its point in the 3D camera coordinate $M = (X_c, Y_c, Z_c)$ is represented as

$$\begin{aligned} M &= wK^{-1}m \\ \begin{bmatrix}X_c\\Y_c\\Z_c\end{bmatrix} &= wK^{-1}\begin{bmatrix}u\\v\\1\end{bmatrix} \\ &= w\begin{bmatrix}f_x & 0 & c_x & 0\\0 & f_y & c_y & 0\\0 & 0 & 1 & 0\end{bmatrix}^{-1}\begin{bmatrix}u\\v\\1\end{bmatrix}\end{aligned} \tag{17}$$

So we can get the following equation, which can be seen as a parametric line.

$$\begin{cases} X_c &= 0 + \dfrac{u-c_x}{f_x}\cdot w \\ Y_c &= 0 + \dfrac{v-c_y}{f_y}\cdot w \qquad w \in \mathcal{R} \\ Z_c &= 0 + 1\cdot w \end{cases} \tag{18}$$

The line passes through $(0, 0, 0)$, and its $\overrightarrow{n} = \left(\dfrac{u-c_x}{f_x}, \dfrac{v-c_y}{f_y}, 1\right)$, the 3D points on this line is represented as $\left(\dfrac{u-c_x}{f_x}w, \dfrac{v-c_y}{f_y}w, w\right)$, where $w \in \mathcal{R}$.

Then from the knowledge of optical ray in [10], we learn a new way to represent the line. The point M in 3D camera coordinate can be described as a parametric line passing through the camera projection centre $C = (0, 0, 0)$ and a special point (at infinity) that projects onto m:

$$M = \begin{bmatrix}C\\1\end{bmatrix} + w\begin{bmatrix}m\\0\end{bmatrix}, w \in \mathcal{R} \tag{19}$$

Problem 3 continued on next page...

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} + w \begin{bmatrix} u \\ v \\ 1 \\ 0 \end{bmatrix}, w \in \mathcal{R} \tag{20}$$

Now continue to read the camera calibration [1] section in the OpenCV document. (Cease reading when you reach the API descriptions of the function calibrateCamera.) The following problems will be based on the symbols and definitions acquired from this web page. Look out for the coordinate directions.

Problem 4

(Distortion) Describe the distortions of the cameras. Given a 2D image point (u, v), can you find its coordinate before distortion? For simplicity, consider only the distortion model with 4 distortion coefficients, e.g., k_1, k_2, p_1, p_2.

(a) describe the distortions of the cameras.

Answer

- **radial distortion**

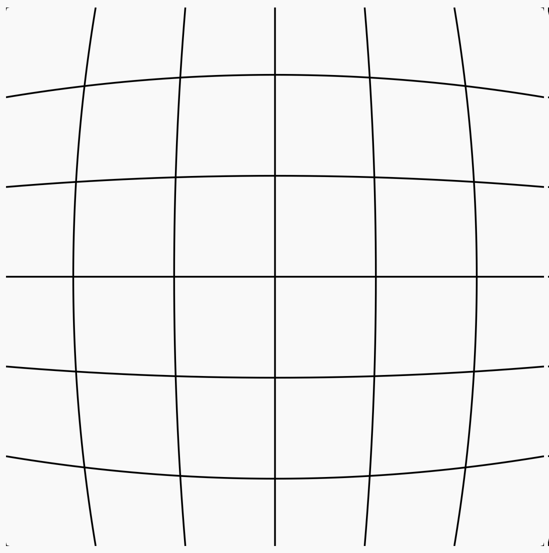

Figure 2: Barrel Distortion

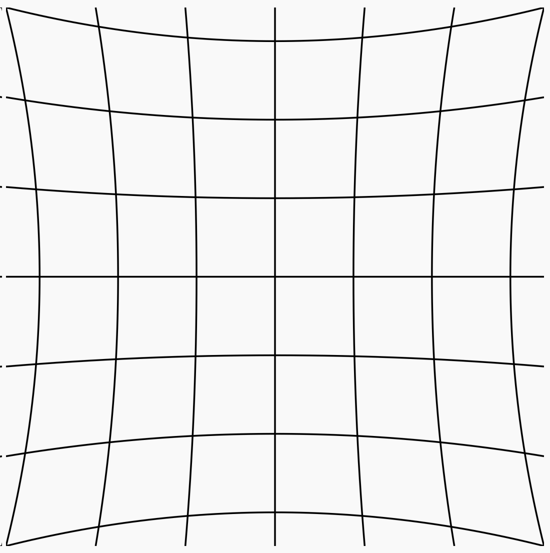

Figure 3: Pincushion Distortion

According to the description of distortion in wiki [11], in barrel distortion, image magnification decreases with distance from the optical axis. The apparent effect is that of an image which has been mapped around a sphere (or barrel). In a zoom lens barrel distortion appears in the middle of the lens's focal length range and is worst at the wide-angle end of the range.

In pincushion distortion, image magnification increases with the distance from the optical axis. The visible effect is that lines that do not go through the centre of the image are bowed inwards, towards the centre of the image, like a pincushion.

$$\begin{cases} x' = x(1 + k_1 r^2 + k_2 r^4) \\ y' = y(1 + k_1 r^2 + k_2 r^4) \end{cases} \tag{21}$$

- **tangential distortion** or **decentering distortion**

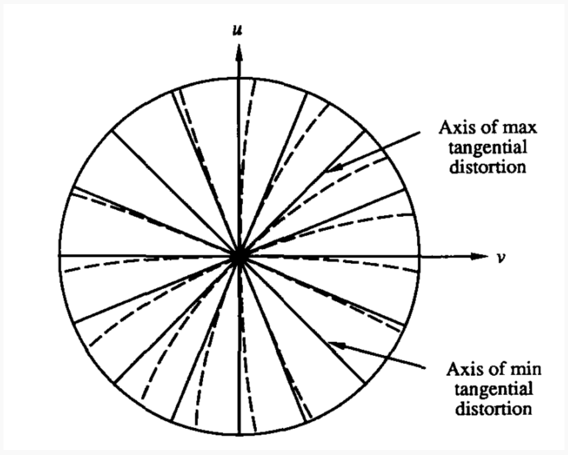
u
v
Axis of max tangential distortion
Axis of min tangential distortion

Figure 3: Tangential Distortion

Tangential distortion is caused by physical elements in a lens not being perfectly aligned[12].

$$\begin{cases} x' = x + 2p_1 y + p_2(r^2 + 2x^2) \\ y' = y + 2p_2 x + p_1(r^2 + 2y^2) \end{cases} \tag{22}$$

(b) find the coordinate of a 2D image point (u,v) before distortion.

Answer

For each pixel (u', v') in a undistorted (corrected) image, we can use the distortion coefficients we obtained from the camera calibration to distort it into a distorted pixel (u, v).

$$\begin{cases} x' \leftarrow \dfrac{u' - c_x}{f_x} \\ y' \leftarrow \dfrac{v' - c_y}{f_y} \\ r^2 \leftarrow x'^2 + y'^2 \\ x \leftarrow x'(1 + k_1 r^2 + k_2 r^4) + 2p_1 y' + p_2(r^2 + 2x'^2) \\ y \leftarrow y'(1 + k_1 r^2 + k_2 r^4) + 2p_2 x' + p_1(r^2 + 2y'^2) \\ u \leftarrow f_x \cdot x + c_x \\ v \leftarrow f_y \cdot y + c_y \end{cases} \tag{23}$$

Now we've got two maps that map a corrected pixel (u', v') to a distorted pixel (u, v):

$$\begin{cases} \text{map}_x(u') = u \\ \text{map}_y(v') = v \end{cases} \tag{24}$$

Problem 4 continued on next page...

We can also get from it the inverse map that can maps a distorted pixel (u, v) to its coordinate before distortion (u', v')

$$\begin{cases} \text{inverse map}_x(u) = u' \\ \text{inverse map}_y(v) = v' \end{cases} \tag{25}$$

Problem 5

(Calibration) Describe what the camera calibration does.

(a) describe what the camera calibration does

Answer

By minimizing the loss between the generated 2D image points using estimated camera matrix obtained from object points and the real 2D image points, camera calibration can find the optimal camera intrinsic and extrinsic parameters from several views of a calibration pattern.

Problem 6

($\star$ Programming) Provided a series of images taken by the camera, can you calibrate the camera with OpenCV functions? (use the images in left.zip. Read more about the APIs in camera calibration 1 section)

(a) progamming the single camera calibartion with OpenCV.

Answer

- **Github: 'Camera' Class**
  - **attributes**
    - **camera_matrix**: camera calibration matrix of the camera
    - **image_root_dir**: the root dir of the images used for calibration
  - **functions**
    - **calibrate** : Given the images, use them to calibrate the camera
    - **undistort** : Given the image after distortion, ouput the image before distortion
- **Github: test_calibrate.py**
- **Results**
  - **calibration error**

    the total sum of squared distances between the observed feature points and the projected object points is 0.4577636
  - **left camera matrix**

$$K_l = \begin{bmatrix} f_x = 536.064 & 0 & c_x = 342.368 \\ 0 & f_y = 536.007 & c_y = 235.531 \\ 0 & 0 & 1 \end{bmatrix} \tag{26}$$

Problem 6 continued on next page...

– **rotation matrix**

$$\begin{bmatrix} -0.26511883 & -0.04659248 & 0.00183174 & -0.00031504 & 0.25213778 \end{bmatrix} \tag{27}$$

Problem 7

(⋆ **Programming)** Undistort the images with the calibration results you computed. What functions of OpenCV do you use?

(a) undistort the images with the calibration results you computed.

Answer

Github: test_undistort.py

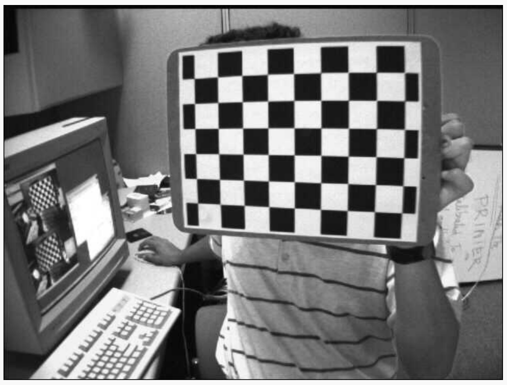

Figure 5: origin image

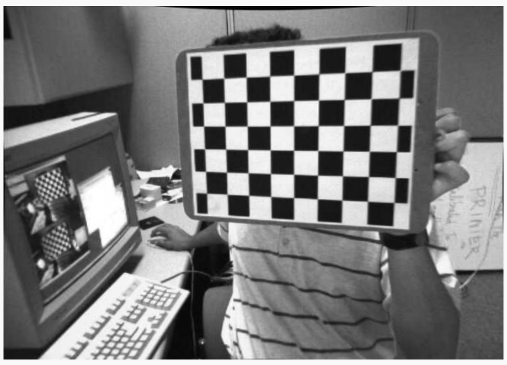

Figure 6: image after undistortion

(b) What functions of OpenCV do you use?

Answer

The following are the two ways of undistorting an image.

1. undistort
2. initUndistortRectifyMap + remap

Estimate the depth of the pixels

Till now you have been skilled with the tricks of a single camera. Can you use a single camera to estimate the depth of the pixels (the distance of their corresponding 3D points to the image plane)? If not, how will you manage to do that? Explain your reasons. (Hint: Recall Problem 3.)

Estimate the depth of the pixels continued on next page. . .

(a) Can you use a single camera to estimate the depth of the pixels?

Answer

We can learn from the Problem 3 that a pixel in the 2D image represents a optical ray or line in 3D space. So it's hard for us to use a single view picture taken by the ordinary camera to estimate the depth of the pixels. However, if we have several pictures that are in multi views, we can estimate the depth of a pixel.

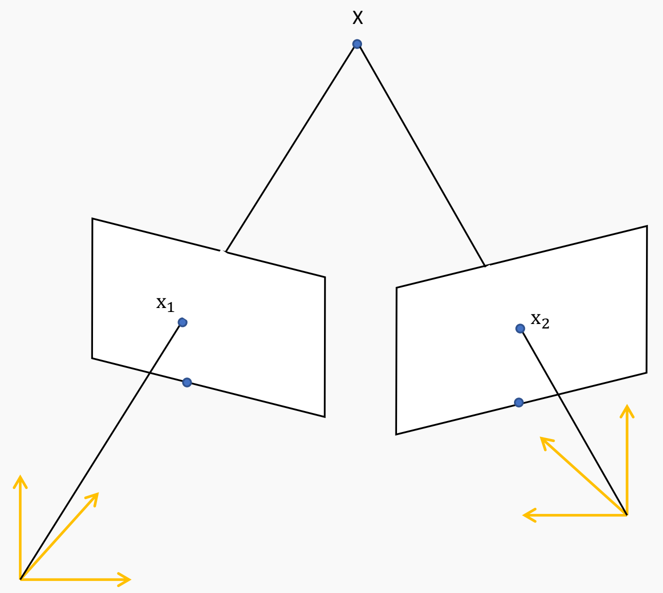

Figure 6: multi view to estimate the depth of a pixel

As shown in Fig. 6, if we can find the two pixels which are the projection of the same 3D point on the two pictures, the intersection of two optical ray is the 3D point.

Binocular Basics

Binocular stereo refers to a stereo system consisting of two cameras. You can locate the 3D coordinates of a point providing its projections in two cameras from different views. This is called the binocular depth estimation. From now on, you will learn how two cameras can be used to estimate the depth of a point. For simplicity, consider the two camera with intrinsics matrix Ml and Mr and temporarily ignore the distortions. The cameras will be distinguished as left and right. Let the 3D world coordinate be aligned with the 3D camera coordinate of the left camera. Denote the transform from the left camera to the right camera as $(R|t)$. $M_l = (f_x, 0, c_x; 0, f_y, c_y; 0, 0, 1)$ and $M_r = (f'_x, 0, c'_x; 0, f'_y, c'_y; 0, 0, 1)$.

Problem 9

(Projection) Given a point with 3D world coordinate $X = (X_1, X_2, X_3)$. Write down its projection in left and right camera planes.

(a) projection in the left camera plane

Answer

$$X_3 \begin{bmatrix} u_l \\ v_l \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ 1 \end{bmatrix} \tag{28}$$

$$\begin{cases} u_l = f_x \cdot \frac{X_1}{X_3} + c_x \\ v_l = f_y \cdot \frac{X_2}{X_3} + c_y \end{cases} \tag{29}$$

(b) projection in the right camera plane

Answer

$$Z_r \begin{bmatrix} u_r \\ v_r \\ 1 \end{bmatrix} = \begin{bmatrix} f'_x & 0 & c'_x \\ 0 & f'_y & c'_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & R_{13} & T_1 \\ R_{21} & R_{22} & R_{23} & T_2 \\ R_{31} & R_{32} & R_{33} & T_3 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ 1 \end{bmatrix} \tag{30}$$

$$\begin{cases} X_r = (X_{r1}, X_{r2}, X_{r3}) \\ X_{r1} = R_{11}X_1 + R_{12}X_2 + R_{13}X_3 + T_1 \\ X_{r2} = R_{21}X_1 + R_{22}X_2 + R_{23}X_3 + T_2 \\ X_{r3} = R_{31}X_1 + R_{32}X_2 + R_{33}X_3 + T_3 \\ u_r = f'_x \cdot \frac{X_{r1}}{X_{r3}} + c'_x \\ v_r = f'_y \cdot \frac{X_{r2}}{X_{r3}} + c'_y \end{cases} \tag{31}$$

Problem 10

($\star\star$ Epipolar Line) Given a pixel in the left image as $x_l = (u, v)$, its corresponding 3D point will also project onto the right image plane. Without knowing its precise 3D coordinate, the probable projection point on the right image will have to lie on a line (This is called the epipolar con- straint). This line is

Problem 10 continued on next page…

called the epipolar line of (u, v). Can you derive the lines equation? (in the right images pixel coordinate) Draw a figure to help you derive.

(a) the figure of the epipolar line

Answer

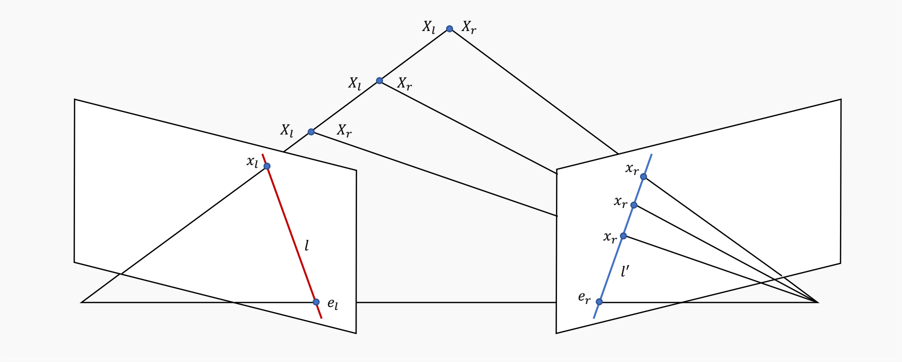

Figure 7: Epipolar Line

(b) the equation of the epipolar line

Answer

$X = X_l = \left(\frac{u - c_x}{f_x} w, \frac{v - c_y}{f_y} w, w, 1 \right), w \in \mathcal{R}$

According to (31),

$$\begin{cases} X_1 = \dfrac{u - c_x}{f_x} w \\ X_2 = \dfrac{v - c_y}{f_y} w \\ X_3 = w \\ u_r = f'_x \cdot \dfrac{R_{11}X_1 + R_{12}X_2 + R_{13}X_3 + T_1}{R_{31}X_1 + R_{32}X_2 + R_{33}X_3 + T_3} + c'_x \\ v_r = f'_y \cdot \dfrac{R_{21}X_1 + R_{22}X_2 + R_{23}X_3 + T_2}{R_{31}X_1 + R_{32}X_2 + R_{33}X_3 + T_3} + c'_y \end{cases} \tag{32}$$

We can set $w = 1$ and $w = 2$ respectively to get two points in the right images' pixel coordinate: (u_{r1}, v_{r1}) and (u_{r2}, v_{r2}). With two points in the right image, we are able to construct a line:

$$\frac{x - u_{r1}}{x - u_{r2}} = \frac{y - v_{r1}}{y - v_{r2}} \tag{33}$$

or we can use the representation of homogeneous line coordinate:

$$\begin{aligned} I &= (u_{r1}, v_{r1}, 1) \times (u_{r2}, v_{r2}, 1) \\ &= \begin{vmatrix} i & j & k \\ u_{r1} & v_{r1} & 1 \\ u_{r2} & v_{r2} & 1 \end{vmatrix} \\ &= (v_{r1} - v_{r2}, u_{r2} - u_{r1}, u_{r1}v_{r2} - u_{r2}v_{r1}) \end{aligned} \tag{34}$$

However, the most efficient way to get the epipolar line I of the left pixel image point $x_l = (u, v)$ is to calculate the Fundamental Matrix F and use the following equation:

$$I = Fx_l \tag{35}$$

Problem 11

(Fundamental Matrix) Generally, given a fixed two camera system. There is a matrix F that for any 3D point, its projections in two images x_l and x_r will satisfy $x_l^T F x_r = 0$. F is called the fundamental matrix. Can you derive the matrix?

(a) derive the fundamental matrix

Answer

The points in the left and right pixel coordinates are represented as x_l and x_r, respectively. Their 3D points in the left and right 3D camera coordinates are represented as X_l and X_r, respectively. They satisfies

$$\begin{cases} X_l = M_l^{-1} x_l \\ X_r = M_r^{-1} x_r \\ X_r = RX_l + T \end{cases} \tag{36}$$

, where R, T is the transformation from the left camera coordinate to the left camera coordinate. Since the rotation R is the orthogonal matrix which satisfies

$$RR^T = E \tag{37}$$

, and means that we can use R^T to take place of the R^{-1} without computing R^{-1}.
Then we can get three vectors : $R^T T$, RX_l, $R^T X_r - R^T T$.

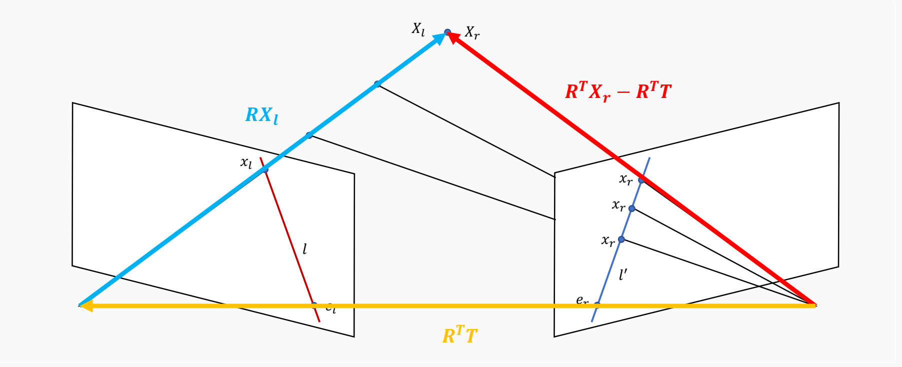

Figure 8: Calculate Fundamental Matrix

Since the $R^T X_r - R^T T$ and $R^T T$ lie in the epipolar plane, then if we take the cross product:

$$\begin{aligned} R^T T \times (R^T X_r - R^T T) &= R^T T \times R^T X_r \\ &= R^T (T \times X_r) \end{aligned} \tag{38}$$

Problem 11 continued on next page. . .

, we will get a vector normal to the epipolar plane. This also means that X_r, which lies in the epipolar plane is normal to $R^T(T \times X_r)$, giving us the constraint that ther dot product is zero:

$$\begin{aligned} (R^T(T \times X_r))^T Xl &= 0 \\ (T \times X_r)^T R X_l &= 0 \end{aligned} \tag{39}$$

From linear algebra, we can introduce a different and compact expression for the cross product: we can represent the cross product between any two vectors a and b as a matrix-vector multiplication:

$$a \times b = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} = \left[[a_\times]b\right] \tag{40}$$

Combining this expression with Equation (42), we can convert the cross product term into matrix multiplication, giving

$$\begin{aligned} ([T_\times]X_r)^T R X_l &= 0 \\ X_r^T [T_\times]^T R X_l &= 0 \\ -X_r^T [T_\times] R X_l &= 0 \\ X_r^T [T_\times] R X_l &= 0 \end{aligned} \tag{41}$$

The matrix $E = [T_\times]R$ is known as the Essential Matrix, creating a compact expression for epipolar constraint:

$$X_r^T E X_l = 0 \tag{42}$$

Combining the previous expression with Equation (36),

$$\begin{aligned} X_r^T [T_\times] R X_l &= 0 \\ (M_r^{-1} x_r)^T [T_\times] R (M_l^{-1} x_l) &= 0 \\ x_r^T M_r^{-T} [T_\times] R M_l^{-1} x_l &= 0 \\ x_r^T F x_l &= 0 \end{aligned} \tag{43}$$

The matrix $F = M_r^{-T}[T_\times]RM_l^{-1}$ is known as **Fundamental Matrix**.

Problem 12

(⋆⋆ **Stereo Calibration)** Weve already learnt how to calibrate a single camera. For a binocular camera system, calibration does more than calibrate each of them. The transform $(R|t)$ between the left and the right cameras also needs calibration. To know more about stereo calibration, you may refer to [3]. Now please use OpenCV to calibrate the binocular cameras with images in left.zip and right.zip respectively. Report the results. (Make sure you understand the geometrical meanings of the parameters)

(a) calibrate the binocular cameras with OpenCV

Answer

- **Github: 'StereoCamera' Class**
  - **attributes**
    - **left_camera**: a 'Camera' Class for the left camera
    - **right_camera**: a 'Camera' Class for the right camera
    - **R**: rotation matrix from the left camera to the right camera coordinate

    * **T**: translation matrix from the left camera to the right camera coordinate
    * **E**: essential matrix
    * **F**: fundamental matrix
  - **functions**
    * **calibrate**: Given the images, use them to calibrate the stereo camera
    * **get_rectify_map**: calculate the rectify map
    * **rectify**: Given the image, output the rectified image

- **Github: test_stereoCalibrate.py**
- **Results**
  - **calibration error**

$$0.4469313896904717$$

  - **left camera matrix**

$$\begin{bmatrix} f_x = 536.064 & 0. & c_x = 342.368 \\ 0. & f_y = 536.0071 & c_y = 235.531 \\ 0. & 0. & 1. \end{bmatrix} \tag{44}$$

  - **left distortion coefficients**

$$\begin{bmatrix} -0.2651188 & -0.0465924 & 0.0018317 & -0.0003150 & 0.2521377 \end{bmatrix} \tag{45}$$

  - **right camera matrix**

$$\begin{bmatrix} f'_x = 542.340 & 0. & c'_x = 328.325 \\ 0. & f'_y = 541.601 & c'_y = 246.953 \\ 0. & 0. & 1. \end{bmatrix} \tag{46}$$

  - **right distortion coefficients**

$$\begin{bmatrix} -0.2805925 & 0.1044422 & -0.0005586 & 0.0012990 & -0.0238370 \end{bmatrix} \tag{47}$$

  - **rotation matrix R**

$$R = \begin{bmatrix} 9.99985279e-01 & 4.12821750e-03 & 3.52119099e-03 \\ -4.12719706e-03 & 9.99991439e-01 & -2.97018427e-04 \\ -3.52238700e-03 & 2.82481406e-04 & 9.99993756e-01 \end{bmatrix} \tag{48}$$

  - **translation matrix T**

$$T = \begin{bmatrix} -3.344204 \\ 0.04170044 \\ 0.05281759 \end{bmatrix} \tag{49}$$

  - **essential matrix E**

$$E = \begin{bmatrix} 7.11035120e-05 & -5.28053561e-02 & 4.17158656e-02 \\ 4.10372297e-02 & 1.16271794e-03 & 3.34436910e+00 \\ -2.88976354e-02 & -3.34434752e+00 & 8.46455006e-04 \end{bmatrix} \tag{50}$$

  - **fundamental matrix F**

$$F = \begin{bmatrix} -3.83435556e-09 & 2.84790664e-06 & -1.87538271e-03 \\ -2.21600862e-06 & -6.27934327e-08 & -9.60375054e-02 \\ 1.36441577e-03 & 9.69011989e-02 & 1.00000000e+00 \end{bmatrix} \tag{51}$$

Problem 13

Given left-image pixel p_l, write down its epipolar line. Pick an arbitrary point on the epipolar line as the projection of the 3D point. Can you derive the 3D coordinate of the point?

(a) the epipolar line of left-image pixel p_l

Answer

Given M_l, M_r, R, T, we can get the fundamental matrix $F = M_r^{-T}[T_\times]RM_l^{-1}$.
We kown that if a homogeneous point p is in line I, it satisfies $p^T I = 0$. According to the epipolar constraint

$$\begin{gathered} p_r^T F p_l = 0 \\ p_r^T I = 0 \end{gathered}, \tag{52}$$

we can assume the line equation $I = Fp_l$ is p_l 's epipolar line.

(a) derive the 3D coordinate of the point?

Answer

According to the Equation (36), given $p_l = (u_l, v_l, 1)$, $p_r = (u_r, v_r, 1)$, M_l, M_r, we can get two line in 3D coordinate, then the intersection of two line is their 3D point.
Solving the following equations, we can determine the value of w_l and w_r, and then use them to get X_l and X_r.

$$\begin{cases} X_l = \begin{bmatrix} X_{l1} \\ X_{l2} \\ X_{l3} \\ 1 \end{bmatrix} = w_l \cdot M_l^{-1} \begin{bmatrix} u_l \\ v_l \\ 1 \end{bmatrix} & , w_l \in \mathcal{R} \\ X_r = \begin{bmatrix} X_{r1} \\ X_{r2} \\ X_{r3} \\ 1 \end{bmatrix} = w_r \cdot M_r^{-1} \begin{bmatrix} u_r \\ v_r \\ 1 \end{bmatrix} & , w_r \in \mathcal{R} \\ \begin{bmatrix} X_{l1} \\ X_{l2} \\ X_{l3} \\ 1 \end{bmatrix} = \begin{bmatrix} R_{3\times3} & T_{3\times1} \\ 0_{1\times3} & 1 \end{bmatrix}^{-1} \begin{bmatrix} X_{r1} \\ X_{r2} \\ X_{r3} \\ 1 \end{bmatrix} \end{cases} \tag{53}$$

$$\begin{cases} X_l = \left(\frac{u_l - c_x}{f_x} w_l, \frac{v_l - c_y}{f_y} w_l, w_l \right), -\infty < w_l < +\infty \\ X_r = \left(\frac{u_r - c_x'}{f_x'} w_r, \frac{v_r - c_y'}{f_y'} w_r, w_r \right), -\infty < w_r < +\infty \\ X_r = RX_l + T \end{cases} \tag{54}$$

Problem 14

(⋆⋆ **Rectification**) The pose of the two cameras are usually arbitrary. It is usually necessary to make the two cameras parallel so that the epipolar line will become horizontal. This is achieved by adding a

Problem 14 continued on next page. . .

Rotation transformation to one of the cameras. Now use OpenCV to rectify the left and the right images with the calibration results obtained from Problem 13. Display the images and check with your eyes to see if they are really rectified. (A pair of the images is enough. Remember to undistort the images)

(a) use OpenCV to rectify the left and the right images

Answer

The epipolar lines are drawn both in origin images and rectified images.

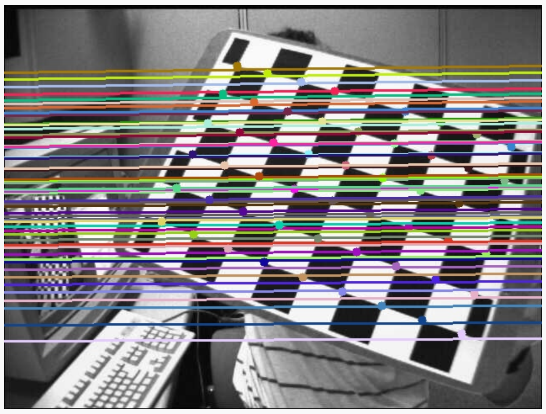
Figure 10: origin left image

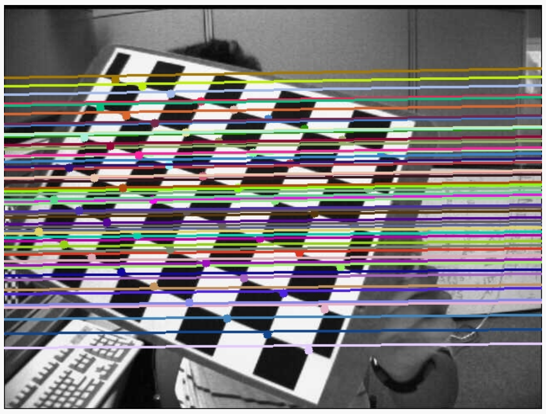
Figure 11: origin right image

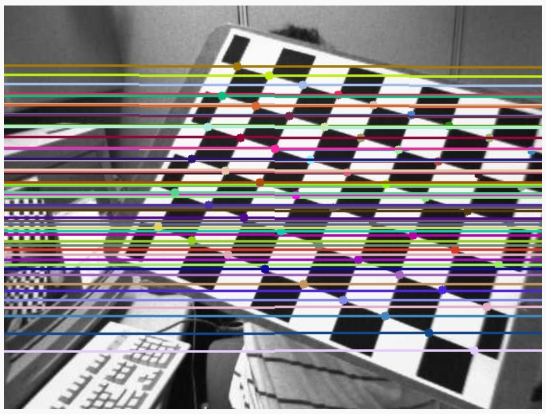
Figure 12: rectified left image

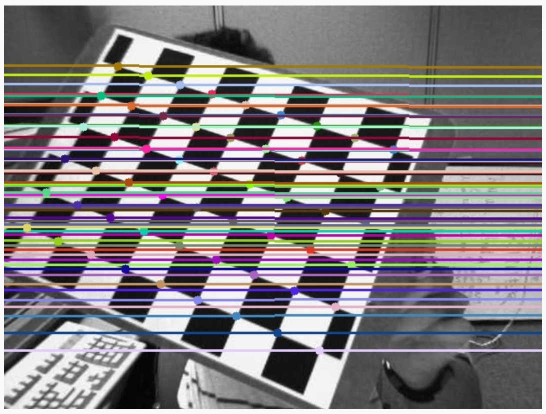
Figure 13: rectified right image

Problem 15

Once the images are rectified, the epipolar lines will become parallel to the image axes. For such a binocular camera system, the transformation between them will be simplified to $(I|t)$. The rotation matrix will become unit matrix since their axes are parallel. The translation matrix t will become $(-b, 0, 0)$ with the coordinates defined in the OpenCV document [1]. Can you derive the baseline b from the results in Problem 14?

Problem 15 continued on next page...

(a) derive the baseline b from the results in Problem 14

Answer

In OepncV, the API function StereoRectify will output P_1,P_2,Q, where P_1, P_2 are the new projection matrixs for the left and right camera, and Q is a disparity-to-depth matrix.

$$P1 = \begin{bmatrix} f = 538.804 & 0. & c_x = 350.439 & 0. \\ 0. & f = 538.804 & c_y = 242.883 & 0. \\ 0. & 0. & 1. & 0. \end{bmatrix} \tag{55}$$

$$P2 = \begin{bmatrix} f = 538.804 & 0. & c'_x = 350.439 & -b * f = -1802.235 \\ 0. & f = 538.804 & c_y = 242.883 & 0. \\ 0. & 0. & 1. & 0. \end{bmatrix} \tag{56}$$

In P_2, the third parameter in the first row represents $b * f$, so the base line or the relative distance between rectified left and right camera coordinate $b = 3.344$.

Problem 16

(Depth-Disparity) For a calibrated binocular camera system, the epipolar lines can be made parallel with rectification. For a pixel on the left image with coordinate (x_l, y), its matching point (the projection of the 3D point on the right image) must have coordinate of the form $(x_l - d, y)$. $d = x_l x_r$ is called the pixels disparity. Can you derive the 3D points coordinate given baseline b and the camera matrices? If the camera has identical vertical and horizontal focal lengths, can the depth z (Z coordinate value) of a pixel be written as $z = bf/d$? Write down your derivation.

(a) derive the 3D points coordinate given baseline b and the camera matrices

Answer

After rectifying the left and right camera, the binocular camera system will become ideal. And we will get the new projection matrix P_l,P_r and the new camera intrinsic matrix K_l,K_r.

$$\begin{cases} P_l = K_l = \begin{bmatrix} f_x & 0 & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \\ P_r = K_r[I|t] = \begin{bmatrix} f'_x & 0 & c'_x & -bf'_x \\ 0 & f'_y & c'_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{cases} \tag{57}$$

Consider recovering the position of (X, Y, Z) from its projections $m_l = (x_l, y_l)$ and $m_r = (x_r, y_r)$,

$$\begin{cases} w_l \begin{bmatrix} x_l \\ y_l \\ 1 \end{bmatrix} = P_l \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \\ w_r \begin{bmatrix} x_r \\ y_r \\ 1 \end{bmatrix} = P_r \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \\ x_r = x_l - d \\ y_l = y_r \end{cases} \tag{58}$$

After solving the equations, we can get the 3D world coordinate (X, Y, Z):

$$\begin{cases} X = \dfrac{1}{f'_x(x_l - c_x) - f_x(x_l - d - c'_x)} \cdot (x_l - c_x) f'_x \cdot b \\ Y = \dfrac{1}{f'_x(x_l - c_x) - f_x(x_l - d - c'_x)} \cdot \dfrac{(y - c_y) f_x f'_x}{f_y} \cdot b \\ Z = \dfrac{1}{f'_x(x_l - c_x) - f_x(x_l - d - c'_x)} \cdot f_x f'_x b \end{cases} \tag{59}$$

According to [13], the new projection matrix will have both the same orientation but different positions. Positions (optical centres) are the same as the old cameras, while orientation changes because we rotate both cameras around the optical centres in such a way that focal planes becomes coplanar and contain the baseline. In order to simplify the algorithm, the rectified PPMs will have also **the same intrinsic parameters**. The new projection matrix will differ only in their optical centres.
Thus, we can derive $f_x = f'_x$, $f_y = f'_y$, $c_x = c'_x$, $c_y = c'_y$ and the 3D world coordinate can be simplified as

$$\begin{cases} X = \dfrac{x_l - c_x}{d} \cdot b \\ Y = \dfrac{y_l - c_y}{d} \cdot \dfrac{f_x}{f_y} \cdot b \\ Z = \dfrac{f_x}{d} \cdot b \end{cases} \tag{60}$$

If the camera does not have the indentical vertical and horizontal focal lengths, which meas $f_x = f_y = f$, then the 3D world coordinate is reprensented as

$$\begin{cases} X = \dfrac{x_l - c_x}{d} \cdot b \\ Y = \dfrac{y_l - c_y}{d} \cdot b \\ Z = \dfrac{f}{d} \cdot b \end{cases} \tag{61}$$

Problem 16 continued on next page. . .

(a) the derivation of depth when the camera has indentical vertical and horizontal focal lengths

Answer

If the camera has identical vertical focal length f_x and horizontal focal length f_y, the depth Z (Z coordinate value) of a pixel can not be written as $Z = \frac{bf}{d}$. Instead, it should be

$$Z = \frac{bf_x}{d}. \tag{62}$$

Stereo Matching

From the last problem, we can find a way to estimate the depth of pixels with a binocular camera system. The calibration can give us the baseline b and all the camera intrinsics and extrinsics. We can undistort the images, then rectify them to have a simple relation between the disparity and depth as derived in Problem 16. What remains is to compute the disparity of each pixel. This is a one-dimensional search along the horizontal epipolar line for each pixel. The task is called stereo matching overview.

Stereo matching is a classical task in computer vision and has a long history. A lot of approaches have been developed during the past decades. A classical algorithm named SGM (semi-global matching) [2] is the most widely used approach and also stands as the baseline for many radical new methods. OpenCV has an implementation of a variation of this algorithm.

Problem 17

(⋆ **SGBM)** Can you use OpenCV to compute the disparity maps for the images you used for stereo calibration? Visualize several disparity results and check with your eyes to see if the results are reasonable.

(a) use SGBM to compute the disparity maps with OpenCV

Answer

Github: test_SGBM.py

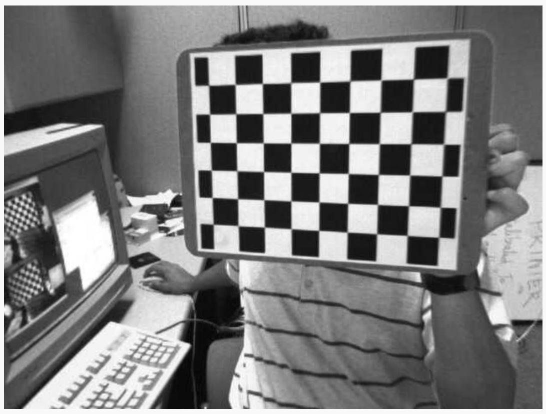

Figure 14: left01.jpg

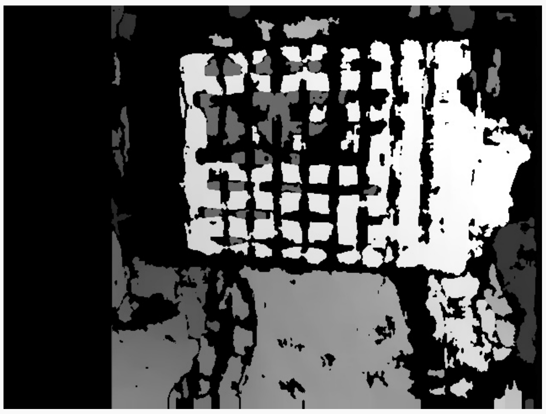

Figure 15: the disparity map of left01.jpg

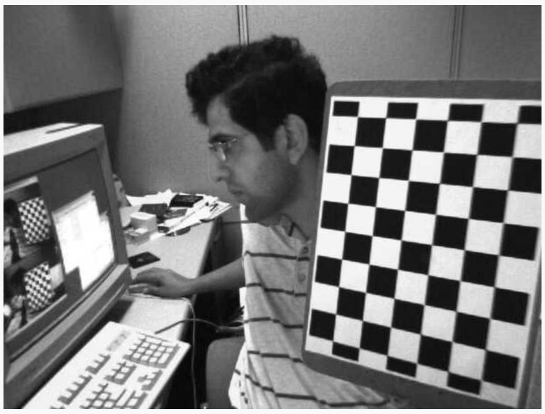
Figure 16: left06.jpg

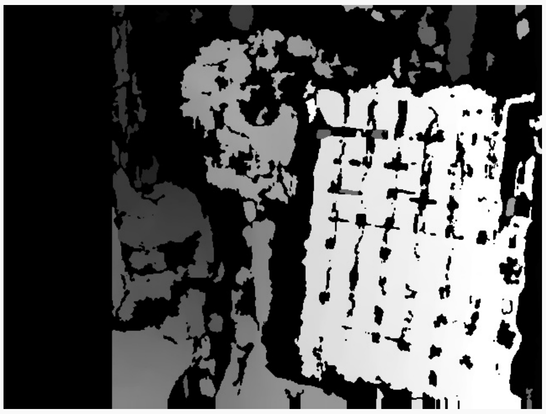
Figure 17: the disparity map of left06.jpg

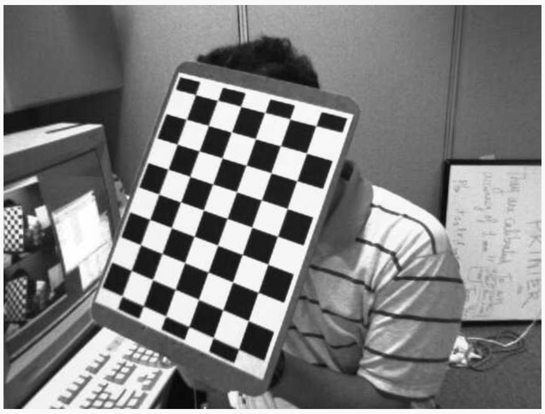
Figure 18: left07.jpg

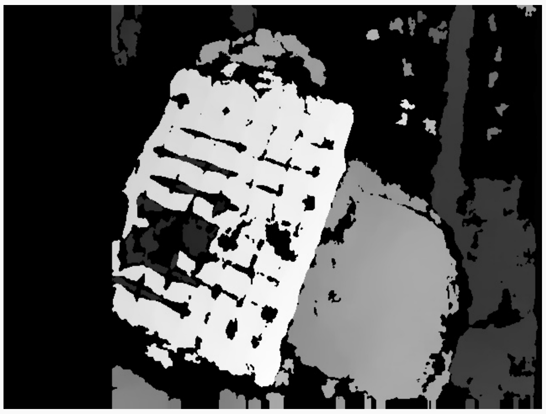
Figure 19: the disparity map of left07.jpg

The KITTI vision benchmark suite 5 holds a benchmark for stereo matching. You can see that most recently developed approaches are deep learning based. The deep learning based approaches are usually supervised.

Problem 18

(⋆ **Unsupervised)** Do you think that supervised deep learning approaches can be easily used for stereo matching? Why or why not? Recently unsupervised deep learning based approaches have been proposed. Can you find out these approaches from the evaluation page?
The following is my policy for searching information:

- Search the key word "survey/introduction for stereo matching" to learn about the research background of stereo matching. Deep learning based approaches may be one of the research stages.
- Knowing that traditional methods $\rightarrow$ supervised deep learning based approaches $\rightarrow$ semi-supervised deep learning based approaches $\rightarrow$ unsupervised deep learning based approaches, unsupervised methods must have solved some problems that is difficult in supervised methods, so reading papers about unsupervised methods will make me informed of the question that whether supervised deep learning based approaches can be easily used for stereo matching. Thus, the next key word I

Problem 18 continued on next page...

will search for is ”unsupervised stereo matching”. Insteard of reading the details of approach, I will focus on reading the **abstract, introduction, related work, conclusion and future perspective**.

(a) can supervised deep learning approaches be easily used for stereo matching?

Answer

No, supervised deep learning approaches cannot be easily used for stereo matching for the following reasons [14]:

1. The lack of data in numbers or the insufficient-data problem. Due to the difficulty of labeling ground truth depth, usable data for system training is rather limited, making it difficult to apply the system to real applications
2. Most datasets are of specific scenes. For example, the KITTI dataset is for autonomous driving and all image pairs are taken with street views. Networks trained on them can hardly perform similarly well in general scenes.
3. Sizes of datasets with real photos are small. Large-scale depth data are mostly synthesized from 3D models. Images in these datasets are different from real world images in appearance and structure.
4. Theremore, in order to generalize well to an unseen dataset, it needed fine tuning, also requiring ground truth data on samples from that dataset. Ground truth motion estimation are not easily available though[15].

(a) Can you find out the unsupervised approaches from the evaluation page?

Answer

In KITTI 2015, two unsupervised deep learning based approaches are found.

1. MADnet [16]
2. OASM-Net [17]

Problem 19

Find out the most efficient deep learning based approach and compare it with SGM. In real time scenarios like autonomous driving, which approach will you prefer? Explain your reason.

(a) compare the most efficient deep learning based approach with SGM, which approach do you prefer in autonomous driving

Answer

In [18], the author compared MC-CNN and GC-net with the SGM. It experimentally proved that the conventional methods and the deep learning based methods performed similarly, and the latter had greater potential to be explored. Although now deep learning based methods perform

Problem 19 continued on next page...

almost the same level as conventional methods, it should be addressed that deep learning based methods only leverage stereo information up-to-now and is extremely faster than conventional methods if pre-trained.
I prefer to use the deep learning based approaches in real time scenarios. As if pre-trained, the deep learning based approach can be faster than SGM, which is crucial in real time scenarios. And we can learn from the KITTI ranking that if given enough data to train, the deep learning based approaches will have a much better performance than SGM does.

References

[1] opencv dev team, "Camera calibration and 3d reconstruction," https://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html, 2019.

[2] D. Hoiem, "Computer vision cs543 of university of illinois," https://courses.engr.illinois.edu/cs543/sp2011/, 2011.

[3] moverzp, "Machine vision study notes (6) - binocular camera calibration parameters," http://blog.csdn.net/xuelabizp/article/details/50417914, 2015.

[4] ——, "Machine vision study notes (4)) - monocular camera calibration parameters description," https://blog.csdn.net/xuelabizp/article/details/50314633, 2015.

[5] opencv dev team, "opencv samples," https://github.com/opencv/opencv/tree/master/samples, 2019.

[6] ——, "Camera calibration," https://docs.opencv.org/3.3.0/dc/dbb/tutorial_py_calibration.html, 2017.

[7] ——, "Epipolar geometry," https://docs.opencv.org/3.3.0/da/de9/tutorial_py_epipolar_geometry.html, 2017.

[8] ——, "Depth map from stereo images," https://docs.opencv.org/3.3.0/dd/d53/tutorial_py_depthmap.html, 2017.

[9] W. contributors, "Camera resectioning — Wikipedia, the free encyclopedia," https://en.wikipedia.org/w/index.php?title=Camera_resectioning&oldid=878694066, 2019.

[10] A. Fusiello, "Elements of computer vision: Multiple view geometry," http://www.diegm.uniud.it/fusiello/teaching/mvg/corsoPadova.pdf, 2005.

[11] W. contributors, "Distortion (optics) — Wikipedia, the free encyclopedia," https://en.wikipedia.org/w/index.php?title=Distortion_(optics)&oldid=898595830, 2019.

[12] J. Weng, P. Cohen, and M. Herniou, "Camera calibration with distortion models and accuracy evaluation," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 10, pp. 965–980, 1992.

[13] A. Fusiello, E. Trucco, and A. Verri, "Rectification with unconstrained stereo geometry." in *BMVC*, 1997, pp. 400–409.

[14] C. Zhou, H. Zhang, X. Shen, and J. Jia, "Unsupervised learning of stereo matching," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1567–1575.

[15] A. Ahmadi and I. Patras, "Unsupervised convolutional neural networks for motion estimation," in *2016 IEEE international conference on image processing (ICIP)*. IEEE, 2016, pp. 1629–1633.

[16] A. Tonioni, F. Tosi, M. Poggi, S. Mattoccia, and L. Di Stefano, "Real-time self-adaptive deep stereo," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[17] Y. Wang, Y. Yang, Z. Yang, L. Zhao, P. Wang, and W. Xu, "Occlusion aware unsupervised learning of optical flow," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4884–4893.

[18] J. Liu, S. Ji, C. Zhang, and Z. Qin, "Evaluation of deep learning based stereo matching methods: From ground to aerial images." *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, vol. 42, no. 2, 2018.