1 ELE510 Image Processing with robot vision: LAB, Exercise 2, Image Formation.

Purpose: *To learn about the image formation process, i.e. how images are projected from the scene to the image plane.*

The theory for this exercise can be found in chapter 2 and 3 of the text book [1]. Supplementary information can found in chapter 1, 2 and 3 in the compendium [2]. See also the following documentations for help:

- OpenCV (https://opencv.org/opencv-python-free-courses/)
- numpy (http://numpy.org/doc/stable/)
- matplotlib (https://matplotlib.org/stable/contents.html)

IMPORTANT: Read the text carefully before starting the work. In many cases it is necessary to do some preparations before you start the work on the computer. Read necessary theory and answer the theoretical part first. The theoretical and experimental part should be solved individually. The notebook must be approved by the lecturer or his assistant.

Approval:

> The current notebook should be submitted on CANVAS as a single pdf file.
>
> To export the notebook in a pdf format, goes to File -> Download as -> PDF via LaTeX (.pdf).

Note regarding the notebook: The theoretical questions can be answered directly on the notebook using a *MarkDown* and LaTex commands (if relevant). In alternative, you can attach a scan (or an image) of the answer directly in the cell.

Possible ways to insert an image in the markdown cell:

```
![image name]("image_path")

<img src='image_path' alt='Alt text' title="Title text" />
```

Under you will find parts of the solution that is already programmed.

> You have to fill out code everywhere it is indicated with `...`
> The code section under `########` is answering subproblem a) etc.

1.1 Problem 1

a) What is the meaning of the abbreviation PSF? What does the PSF specify?

1.1.1 Pont Spread Function (PSF)

It is a function that defines how an object would look like when processed or visualized by an imaging system. It is also called impulse response. The PSD is a result of diffraction and interference. The degree of spreading (blurring) of the object is a measure for the quality of an imaging system.

b) Use the imaging model shown in Figure 1. The camera has a lens with focal length $f = 40mm$ and in the image plane a CCD sensor of size $8mm \times 8mm$. The total number of pixels is 4000×4000. How many lines per mm will this camera resolve at a distance of $z_w = 1m$ from the camera center?

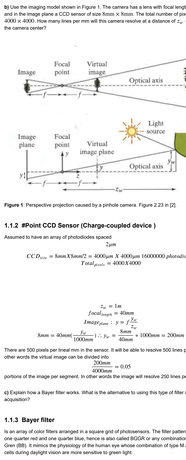


Figure 1: Perspective projection caused by a pinhole camera. Figure 2.23 in [2].

1.1.2 #Point CCD Sensor (Charge-coupled device)

Assumed to have an array of photodiodes spaced

$$2\mu m$$

$$CCD_{size} = 8mmX8mm \Box = 4000\mu m \ X \ 4000\mu m \ 16000000 \ photodiodes$$
$$Total_{pixels} = 4000X4000$$

$$z_w = 1m$$
$$focal_{length} = 40mm$$
$$Image_{size} : y = f\frac{y_c}{z_w}$$
$$8mm = 40mm\frac{y_c}{1000mm} \Box : y_c = \frac{8mm}{40mm} * 1000mm = 200mm$$

There are 500 pixels per lineal mm in the sensor. It will be able to resolve 500 lines per mmm. In other words the virtual image can be divided into

$$\frac{200mm}{4000mm} = 0.05$$

portions of the image per segment. In other words the image will resolve 250 lines per mm.

c) Explain how a Bayer filter works. What is the alternative to using this type of filter in image acquisition?

1.1.3 Bayer filter

Is an array of color filters arranged in a square grid of photosensors. The filter pattern is half green, one quarter red and one quarter blue, hence is also called BGGR or any combination or double Gren (BB). It mimics the physiology of the human eye which has more combination of type M and L cone cells during daylight vision are more sensitive to green light.

Information of the color combination is calculated by debayering algorithms which interpolate the missing information from neighboring pixels and estimate it. This involves image processing that is implemented on the same chip as the image sensor.This process, though can lead to false representation of image quality

An alternative to this architecture is found in the CMY color combination (Cyan, Magenta, Yellow) a set of inverted colors that have and improved light absorption characteristic. Other filter color patterns can be found for different camera manufacturers such as

- Fujifilm "EXR" color filter array
- Fujifilm "X-trans" filter
- Quad Bayer

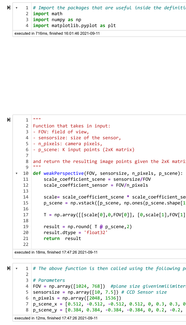
1.2 Problem 2

Assume we have captured an image with a digital camera. The image covers an area in the scene of $1.024m \times 0.768m$ (The camera has been pointed towards a wall such that the distance is approximately constant over the whole image plane, weak perspective). The camera has 2048 pixels horizontally, and 1536 pixels vertically. The active region on the CCD-chip is $10mm \times 7.5mm$. We define the spatial coordinates (x_w, y_w) such that the origin is at the center of the optical axis, x-axis horizontally and y-axis vertically upwards. The image indexes (x, r) is starting in the upper left corner. For simplicity let the optical axis meet the image plane at $(x_0 = 1024, y_0 = 768)$. The solutions to this problem can be found from simple geometric considerations. Make a sketch of the situation and answer the following questions.

a) What is the size of each sensor (one pixel) on the CCD-chip?

b) What is the scaling coefficient between the image plane (CCD-chip) and the scene? What is the scaling coefficient between the scene coordinates and the image indexes?

1.2.1 Sketch [...]



1.2.2 a) What is the size of each sensor (one pixel) on the CCD-Chip?

$$Scene_{size} = 1024mm \times 768mm$$
$$Camera_{resolution} = 2048 \times 1536 \ pixels$$
$$CCD_{Chip_{size}} = 10mm \times 7.5mm$$
$$Pixel_{size_x} = \frac{10mm}{2048 pixels} = 0.0048mm = 4.8\mu m$$
$$Pixel_{size_y} = \frac{7.5mm}{1536 pixels} = 0.0048mm = 4.8\mu m$$
$$Pixel_{size} = 4.8\mu m \times 4.8\mu m$$

1.2.3 b) What is the scaling coefficient between the image plane (CCD-Chip) and the scene?

$$C_x = \frac{10mm}{1024mm} = 0.97$$

$$C_y = \frac{7.5mm}{768mm} = 0.97$$

The coefficient is 0.97 to 1 or 97 times smaller in the CCD-Chip than in the original scene

1.2.4 What is the scaling coefficient between the scene coordinates and the image pixels?

$$C_x = \frac{1024}{2048} = 0.5$$

$$C_y = \frac{768}{1536} = 0.5$$

The coefficient is 1 to 1/2

1.3 Problem 3

Translation from the scene to a camera sensor can be done using a transformation matrix, T.

$$\begin{bmatrix} s \\ t \end{bmatrix} = T \begin{bmatrix} x_w \\ y_w \end{bmatrix} \tag{1}$$

where

$$T = \begin{bmatrix} a_x & 0 & s_0 \\ 0 & a_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

a_x and a_y are the scaling factors for their corresponding axes.

Write a function in Python that computes the image points using the transformation matrix, using the parameters from Problem 2. Let the input to the function be a set of K scene points, given by a $2 \times K$ matrix, and the output be resulting image points also given by a $2 \times K$ matrix. The parameters defining the image sensor and field of view from the camera center to the wall can also be given as input parameters.

Test the function for the following input points given as a matrix

$$\mathbf{P}_w = \begin{bmatrix} 0.512 & -0.512 & -0.512 & 0.512 & 0 & 0.3 & 0.3 & 0.6 \\ 0.384 & 0.384 & -0.384 & -0.384 & 0.2 & -0.2 & -0.4 & 0 \end{bmatrix} \tag{3}$$

> Comment on the results, especially notice the two last points!

```
In [2]: M ▾  1 # Import the packages that are useful inside the definition of the wea
           2 import math
           3 import numpy as np
           4 import matplotlib.pyplot as plt
```

executed in 710ms, finished 19:51:48 2021-09-11

```
In [47]: M ▾  1 """
           2 Function that takes in input:
           3 - FOV: field of view,
           4 - sensorsize: size of the sensor,
           5 - n_pixels: camera pixels,
           6 - p_scene: K input points (2xK matrix)
           7
           8 and return the resulting image points given the 2xK matrix
           9 """
          10 def weakPerspective(FOV, sensorsize, n_pixels, p_scene):
          11     scale_coefficient_scene = sensorsize/FOV
          12     scale_coefficient_sensor = FOV/n_pixels
          13
          14     scale= scale_coefficient_scene * scale_coefficient_sensor
          15     p_scene = np.vstack((p_scene, np.ones((p_scene.shape[1]))))
          16
          17     T = np.array([[scale[0],0,FOV[0]], [0,scale[1],FOV[1]], [0,0,1]])
          18
          19     result = np.round( T @ p_scene,2)
          20     result.dtype = 'float32'
          21     return result
          22
```

executed in 18ms, finished 17:47:28 2021-09-11

```
In [48]: M ▾  1 # The above function is then called using the following parameters:
           2
           3 # Parameters
           4 FOV = np.array([1024, 768])  #plane size given(millimiters)
           5 sensorsize = np.array([10, 7.5]) # CCD Sensor size
           6 n_pixels = np.array([2048, 1536])
           7 p_scene_x = [0.512, -0.512, -0.512, 0.512, 0, 0.3, 0.3, 0.6]
           8 p_scene_y = [0.384, 0.384, -0.384, -0.384, 0, 0.2, -0.2, -0.4, 0]
```

executed in 12ms, finished 17:47:28 2021-09-11

```
In [49]: M ▾  1 ####
           2 # This cell is locked; it can be only be executed to see the results.
           3 ####
           4 # Input data:
           5 p_scene = np.array([p_scene_x, p_scene_y])
           6
           7 # Call to the weakPerspective() function
           8 pimage = weakPerspective(FOV, sensorsize, n_pixels, p_scene)
           9
          10 # Result:
          11 print(pimage)
```

executed in 704ms, finished 17:47:28 2021-09-11

```
[[0.   4.5  0.   4.5  0.   4.5  0.   4.5  0.   4.5 ]
 [0.   4.5  0.   4.5  0.   4.5 ]
 [0.   4.25 0.   4.25 0.   4.25 0.   4.25 0.   4.25]
 [0.   4.25 0.   4.25 0.   4.25 ]
 [0.   1.875 0.  1.875 0.  1.875 0.  1.875 0.  1.875]
 [0.   1.875 0.  1.875 0.  1.875]]
```

1.3.1 Delivery (dead line) on CANVAS: 12-09-2021 at 23:59

1.4 Contact

1.4.1 Course teacher

Professor Kjersti Engan, room E-431

E-mail: kjersti.engan@uis.no (mailto:kjersti.engan@uis.no)

1.4.2 Teaching assistant

Tomaselli Luca, room E-401

E-mail: luca.tomaselli@uis.no (mailto:luca.tomaselli@uis.no)

1.5 References

[1] S. Birchfield, Image Processing and Analysis. Cengage Learning, 2016.

[2] I. Austvoll, "Machinelrobot vision part I," University of Stavanger, 2018. Compendium, CANVAS.