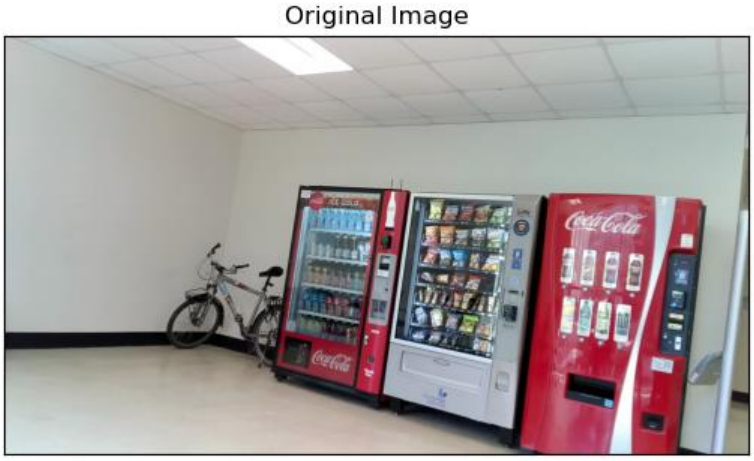
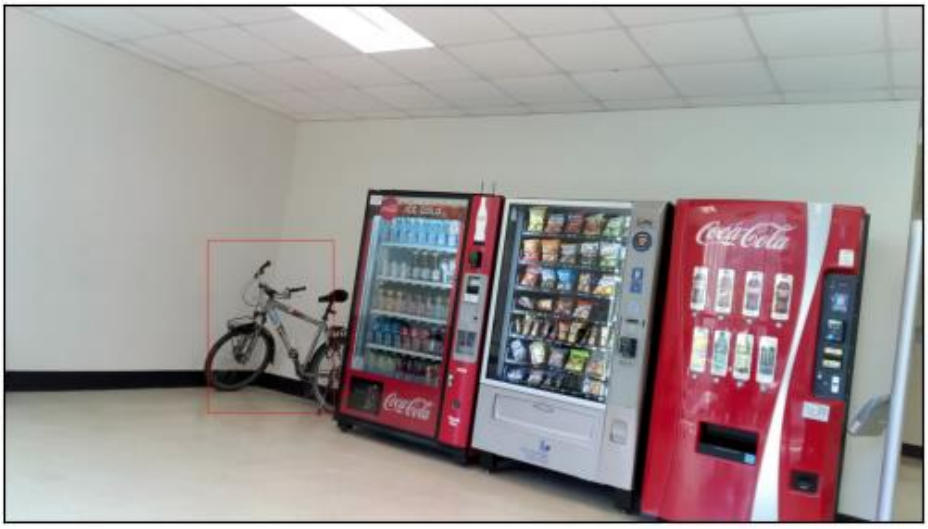
I wrote some code to capture frames using depthai api and the OAK D Lite camera
All source code is in the GitHub link: https://github.com/bdairo/cv-assignment-3

Question 1



For this question, I cropped the bicycle from the original image and used it as a template. I also used Normalized correlation (cv2.TM_CCORR_NORMED) from OpenCV to check for matches between the template and original image. The method from OpenCV has a value normalized to a range of 0 to 1. For my code, I set a threshold of 0.9 to determine if there is a match or not. The results after performing normalized correlation between the template and the comparison dataset was that the bicycle was found in some images and was not found in others. More images are on the notebook.

Detected Point



Question 2

Motion Tracking Equation

(a)

Motion Tracking Equation

Motion tracking is based on optical flow principles. Optical flow is the method to estimate apparent motion of scene points from a sequence of images

* Picking a pixel x, y from two consecutive images of a scene at time t and time $t + \delta t$

We have (x, y) at t and $(x + \delta x, y + \delta y)$ at $t + \delta t$

The displacement of point $(x, y) = (\delta x, \delta y)$

Hence, Optical flow $(u, v) = \left(\frac{\delta x}{\delta t}, \frac{\delta y}{\delta t}\right)$

(u, v) refers to the speed of the point in both directions.

* To estimate the optical flow, we make some assumptions

1) The brightness of an image point remains constant over time

Since we are using consecutive images, Intensity / brightness would remain the same.

This leads us to the equation

$$I(x + \delta x, y + \delta y, t + \delta t) = I(x, y, t) \quad \text{--- eqn (1)}$$

I at new point — I at old location

11) The displacement $(\delta x, \delta y)$ and time step δt are very small. This allows us to come up with an approximation based on the Taylor's series expansion

Given

$$f(x+\delta x) = f(x) + \frac{\partial f}{\partial x}\delta x + \frac{\partial^2 f}{\partial x^2}\frac{\delta x^2}{2!} + \cdots + \frac{\partial^n f}{\partial x^n}\frac{\delta x^n}{n!}$$

If δx is very small, then higher order terms are almost zero

Hence we can set an approximation for Image brightness

$$I(x+\delta x, y+\delta y, t+\delta t) = I(x,y,t) + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t$$

$$I(x+\delta x, y+\delta y, t+\delta t) = I(x,y,t) + I_x \delta x + I_y \delta y + I_t \delta t$$

↳ eqn (2)

* Subtracting eqn(1) and eqn (2)

$$\therefore I_x \delta x + I_y \delta y + I_t \delta t = 0$$

* divide by δt and take $\delta t \to 0$ as limit to zero

$$I_x \frac{\partial x}{\partial t} + I_y \frac{\partial y}{\partial t} + I_t = 0$$

* Since $(u,v) = \left(\frac{\delta x}{\delta t}, \frac{\delta y}{\delta t}\right)$

Optical flow constraint Equation = $I_x u + I_y v + I_t = 0$

Motion Function Estimate

Motion function Estimate

Picking a point in two consecutive frames

Point 1 with pixel coordinates (492, 240)

Point 2 with pixel coordinates (368, 237)

The displacement

in x direction = 368 - 492 = -124 pixels

y direction = 237 - 240 = -3 pixels

motion vector $\vec{v} = (-124, -3)$

The motion function is essentially a vector field where a point in one frame is mapped to another point in frame 2 via the motion vector.

* we assume that the motion is uniform across the scene

With the information so far

motion function $f(x,y) = (x + \delta x, y + \delta y)$

$f(x,y) = (x - 124, y - 3)$

This function can be used to predict the position of a point in the first frame to its position in the second frame.

Lucas Kanade

(b)

Lucas Kanade Algorithm for motion Tracking

The assumption in the Lucas Kanade Method is that for each pixel, assume motion field, and hence optical flow (u, v) is constant within a small neighborhood W

For all points $(k, l) \in W$

$$I_x(k,l)u + I_y(k,l)v + I_t(k,l) = 0$$

From the question, given:

$$u(x,y) = a_1 x + b_1 y + c_1$$
$$v(x,y) = a_2 x + b_2 y + c_2$$

$u(x,y)$ is assumed to be the horizontal component of motion at point (x,y)

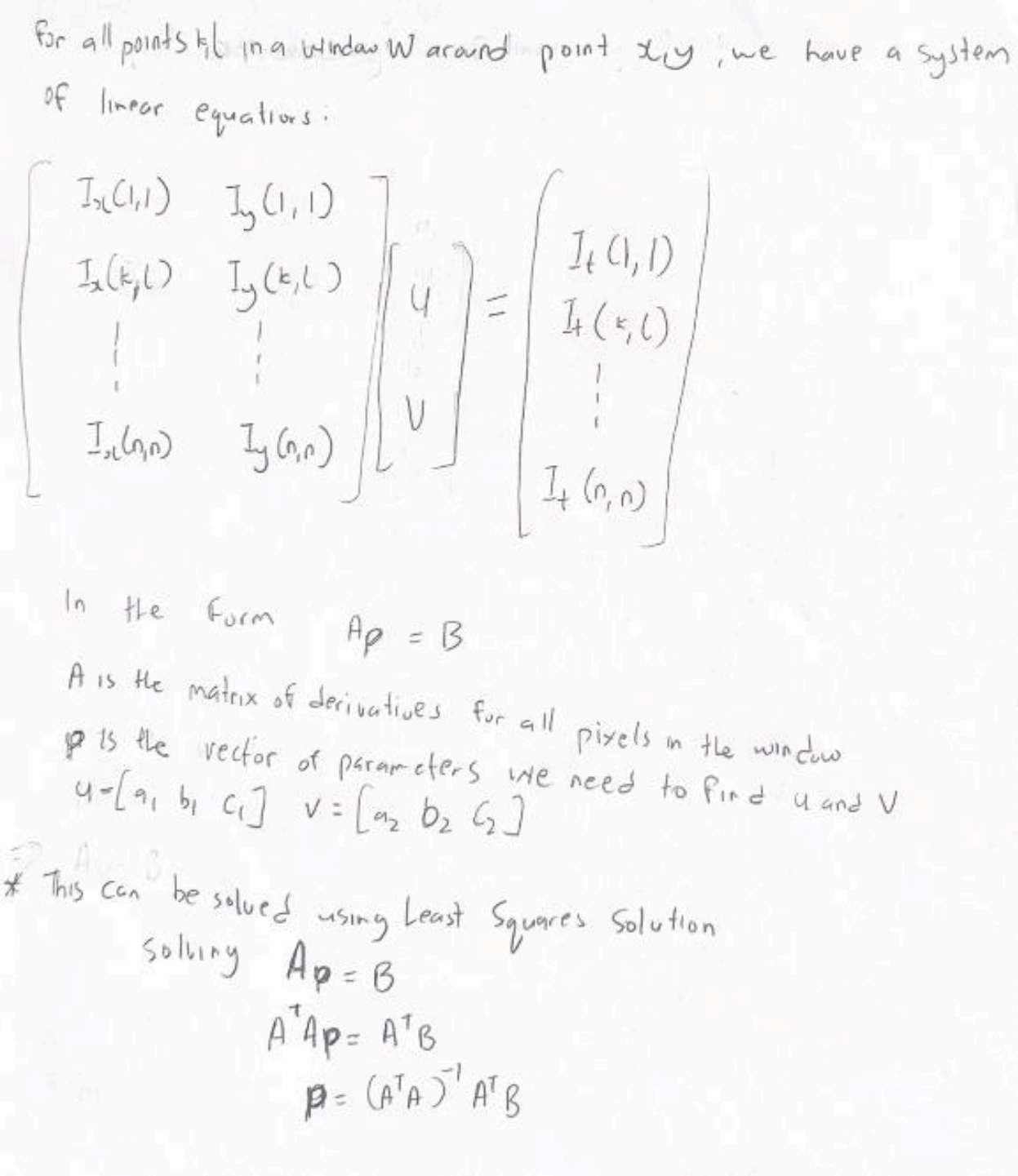
$v(x,y)$ is assumed to be the vertical component of motion at point (x,y)

For each point (x,y), Optical flow constraint Equation is

$$I_x(x,y)u + I_y(x,y)v + I_t(x,y) = 0$$

$$= I_x(a_1 x + b_1 y + c_1) + I_y(a_2 x + b_2 y + c_2) + I_t = 0$$

$$\Rightarrow I_x(a_1 x + b_1 y + c_1) + I_y(a_2 x + b_2 y + c_2) = -I_t$$



Question 3

This problem required estimating the distance between the camera and the images using stereo vision theory. To solve this, I took the following steps:

- **Created a large white canvas with a dark circle in the center** I wrote some code to generate a white canvas with a dark circle in the center.
- **Captured two canvas images** using the left and right mono cameras and depth ai library.
- **Got the coordinates of a point in both images** by using OpenCV algorithms.
- **Used the formula.**

  D = (focal length x Translation)/ Disparity to calculate depth

  Translation = 7.5cm
  This value was obtained from the OAK D Lite camera documentation as the distance between the two cameras.

  Focal length = 442.590392 using the camera matrix from previous assignment

  Disparity as the difference between the x coordinates of the point in both images

Results

I was able to estimate the distance as 26.77cm. The actual distance in my set up was 27.5 cm with a difference of less than 1cm. This indicates the effectiveness of the stereo vision system using the OAK-D Lite camera.

Question 4, 5, 6, and 7 are contained in their separate folders in the github repo

Question 8

- **Object detection using markers**: I decided to use ArUco markers for object detection. This was done in two steps:
    - **Marker Generation:** I used a predefined ArUco dictionary (DICT_6X6_250) to generate a marker with Id 23.
    - **Marker detection:** I wrote the Marker Detection code that uses OAK D Lite camera's sensor, Frames captured from the camera are analyzed using the same ArUco dictionary to detect the presence and ID of the marker in the frame.

      Results
      The marker was successfully detected the ArUco marker with the id of 23 also shown in the detection.
      A demo is in this link: https://youtu.be/W0xZA8dilmQ?si=_cOWjmbj7Tj0NrCJ

- **Object detection without markers:** I implemented this in the following steps:
    - **Feature Detection:** I utilized SIFT algorithm from OpenCV to detect and compute unique features in the frames. I tuned the parameters of the SIFT
    - **Feature Matching:** This was done using FLANN-based matcher for efficient descriptor matching between consecutive frames.
    - **Clustering of features:** Applied DBSCAN for clustering feature points based on the assumption that feature points of distinct object are in close proximity.
    - **Object Tracking:** Bounding boxes were dynamically drawn around identified objects to track their movements across frames.

      Results
      The tracker successfully identified and followed objects across frames in real-time, using frames captured by the OAKD camera, showcasing robust performance in handling object detection.

      A demo is in this link: https://youtu.be/mZln7qDN_YA?si=e0R4pSV88VAlfhQh