Experiment No. 4
To study the Depth Estimation
Name of Student :- 09_Amruta Poojary
Date of Performance: 9/7/2023
Date of Submission: 16/7/2023

Aim: To study the Depth Estimation

Objective: To Capturing Frames form a depth camera creating a mask from a disparity map Masking a copy operation Depth estimation with normal camera

Theory:

1.Depth map

Depth map is a grayscale image in which each pixel value is the estimated distance from the camera to a surface. Specifically, each pixel value is a 16-bit unsigned integer representing a depth measurement in millimeters. OpenNI 2, allows us to request depth map channel from a depth camera through cv2.CAP_OPENNI_DEPTH_MAP

2.Point cloud map

Point cloud map is a color image in which each color corresponds to an x, y, or z spatial dimension. Specifically, the channel yields a BGR image, where B is x (blue is right), G is y (green is up), and R is z (red is deep), from the camera's perspective. OpenNI 2, allows us to request Point cloud map channel from a depth camera through cv2.CAP_OPENNI_POINT_CLOUD_MAP

3.disparity map

Disparity maps are grayscale images in which each pixel value is the stereo disparity of a surface. To conceptualize stereo disparity, let's suppose we overlay two images of a scene, shot from different viewpoints. The result would be similar to seeing double. For points on any pair of twin objects in the scene, we can measure the distance in pixels. This measurement is the stereo disparity. Nearby objects exhibit greater stereo disparity than far-off objects. Thus, nearby objects appear brighter in a disparity map.
cv2.CAP_OPENNI_DISPARITY_MAP is a disparity map with 8-bit unsigned integer values and cv2.CAP_OPENNI_DISPARITY_MAP_32F is a disparity map with 32-bit floating point values

4. A valid depth mask

A valid depth mask shows whether the depth information at a given pixel is believed to be valid (shown by a non-zero value) or invalid (shown by a value of zero). For example, if the depth camera depends on an infrared illuminator (an infrared flash), depth information is invalid in regions that are occluded (shadowed) from this light.

5. Creating a Mask from a disparity map

Let's assume that a user's face, or some other object of interest, occupies most of the depth camera's field of view. However, the image also contains some other content that is not of interest. By analyzing the disparity map, we can tell that some pixels within the rectangle are outliers—too near or too far to really be a part of the face or another object of interest. We can make a mask to exclude these outliers. However, we should only apply this test where the data is valid, as indicated by the valid depth mask.

6. Masking a Copy Operation

In it we copy only those pixels in the source rectangle where the mask's value is not zero and other pixels retain their old values from the destination image.

7. Depth estimation with a normal camera

We can use one or more normal cameras and we can estimate relative distances to objects based on triangulation from different camera perspectives. If we use two cameras simultaneously, this approach is called stereo vision. If we use one camera, but we move it over time to obtain different perspectives, this approach is called structure from motion.

Code:

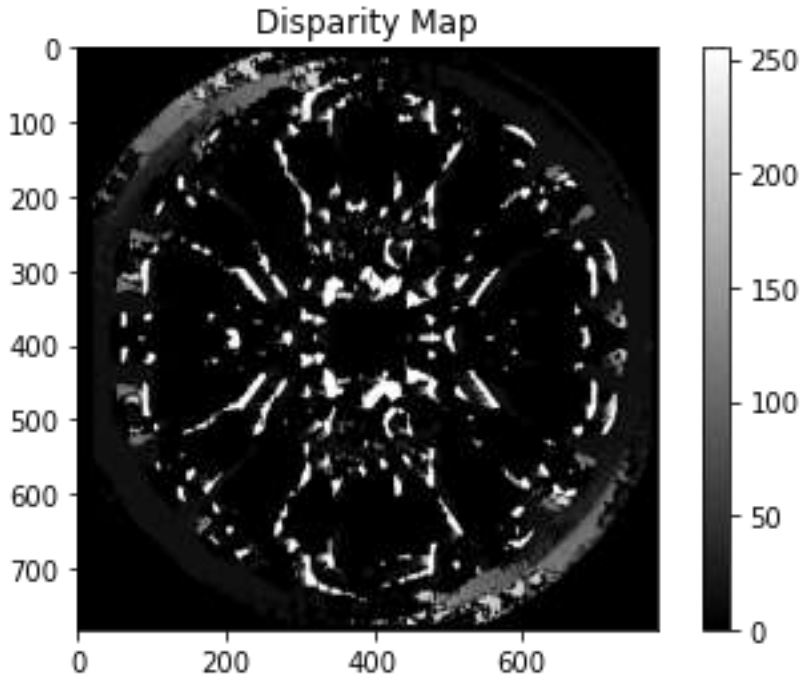
```
import numpy as np
import cv2
from matplotlib import pyplot as plt

# Read two input images as grayscale images
imgL = cv2.imread('JUNO1.png', 0)
imgR = cv2.imread('JUNO2.png', 0)

# Initiate a StereoBM object
stereo = cv2.StereoBM_create(numDisparities=16, blockSize=15)

# Compute the disparity map
disparity = stereo.compute(imgL, imgR)

# Normalize the disparity map for visualization
normalized_disparity = cv2.normalize(disparity, None, alpha=0, beta=255,
norm_type=cv2.NORM_MINMAX, dtype=cv2.CV_8U)

# Display the disparity map using matplotlib
plt.imshow(normalized_disparity, 'gray')
plt.title('Disparity Map')
plt.colorbar()
plt.show()

# Print the shape of the disparity map
print("Disparity Map Shape:", disparity.shape)
```

Input:

JUNO1.png (LEFT)

JUNO2.png (RIGHT)

Output:-



Conclusion:

Using OpenCV we can perform major tasks in machine vision such as using data from a depth camera to identify foreground and background regions, such that we can limit an effect to only the foreground or only the background as well as use a depth camera to capture depth maps, point cloud maps, disparity maps, images based on visible light, and images based on infrared light and convert a disparity map into a mask that differentiates between foreground and background regions which are essential tasks involved in developing machine vision applications.