Name: Divya Patil
Roll No: 08

Experiment No: 04

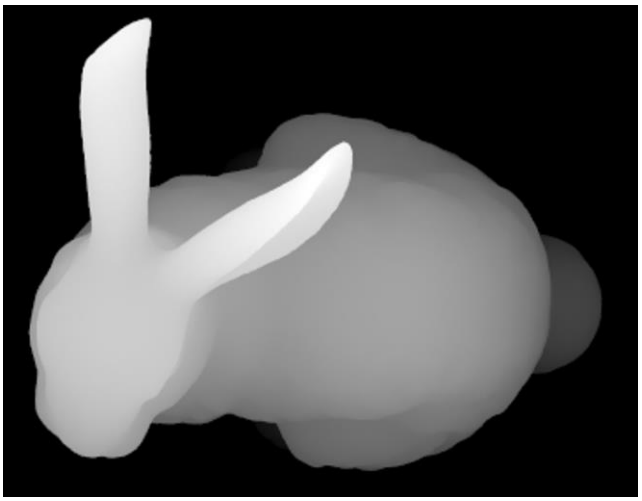
Aim: To study the Depth Estimation

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 maps contain information about the distance of objects from a specific perspective or reference point (like a camera lens). Each pixel is assigned a value to represent the distance of that pixel from the reference point which creates a 3D representation of the scene for its RGB image or virtual scene.



A depth map of the Stanford Bunny

Take a look at the image of the Stanford Bunny above. The white pixels represent the part of the scene that is closest to the camera lens, and the black pixels represent the part of the scene that is furthest. In this case the part of the scene that is closest are the ears of the bunny. The grayscale gradient in between illustrates that the head, neck, and body are a bit further

from the camera, the legs even further, and the tail of the bunny the furthest before the background, or furthest point of the image.

2. Point cloud map

A point cloud is a collection of points of data plotted in 3D space, using a 3D laser scanner. If you're scanning a building, for example, each virtual point would represent a real point on the wall, window, stairway, metalwork, or any surface the laser beam meets.

3. Disparity map

Disparity map refers to the apparent pixel difference or motion between a pair of stereo images. To experience this, 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.

4. A valid depth mask

A valid depth mask in computer vision is an image or data structure that encodes the depth information of a scene, typically represented as grayscale values where each pixel corresponds to a depth value. It is used to distinguish objects at different distances from a camera, with darker pixels indicating objects closer to the camera and lighter pixels representing objects farther away. Depth masks are crucial for tasks like depth perception, 3D reconstruction, and object segmentation in various computer vision applications.

5. Creating a Mask from a disparity map

Creating a mask from a disparity map in computer vision involves thresholding or selecting a specific range of disparity values to isolate objects of interest in a stereo image pair. This mask helps separate foreground objects from the background, aiding in tasks like object segmentation and depth perception. By setting a disparity threshold, you can extract regions with meaningful depth information for further analysis or visualization.

6. Masking a Copy Operation

Masking a copy operation in computer vision refers to the process of selectively copying or transferring pixel values from one image (the source) to another (the destination) based on a binary mask. The binary mask specifies which pixels in the source image should be copied to the corresponding positions in the destination image. This technique is commonly used for

tasks like image composition, where objects or regions of interest are extracted and superimposed onto another image while preserving transparency or alpha channel information.

7. Depth estimation with a normal camera

Depth estimation with a normal camera in computer vision is the process of inferring the depth information of a scene or objects within it using a standard RGB camera. This typically involves analyzing the disparity between corresponding points in stereo images, exploiting motion parallax, or using machine learning techniques to estimate the distance of objects from the camera, enabling applications like 3D reconstruction, object tracking, and augmented reality.

Code:

```
import numpy as np

import cv2

from matplotlib import pyplot as plt

# Read two input images

imgL = cv2.imread('left.jpg')

imgR = cv2.imread('right.jpg')

# Convert the images to grayscale

grayL = cv2.cvtColor(imgL, cv2.COLOR_BGR2GRAY)

grayR = cv2.cvtColor(imgR, cv2.COLOR_BGR2GRAY)

# Initiate a StereoBM object

stereo = cv2.StereoBM_create(numDisparities=16, blockSize=15)

# Compute the disparity map

disparity = stereo.compute(grayL, grayR)

# 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
```

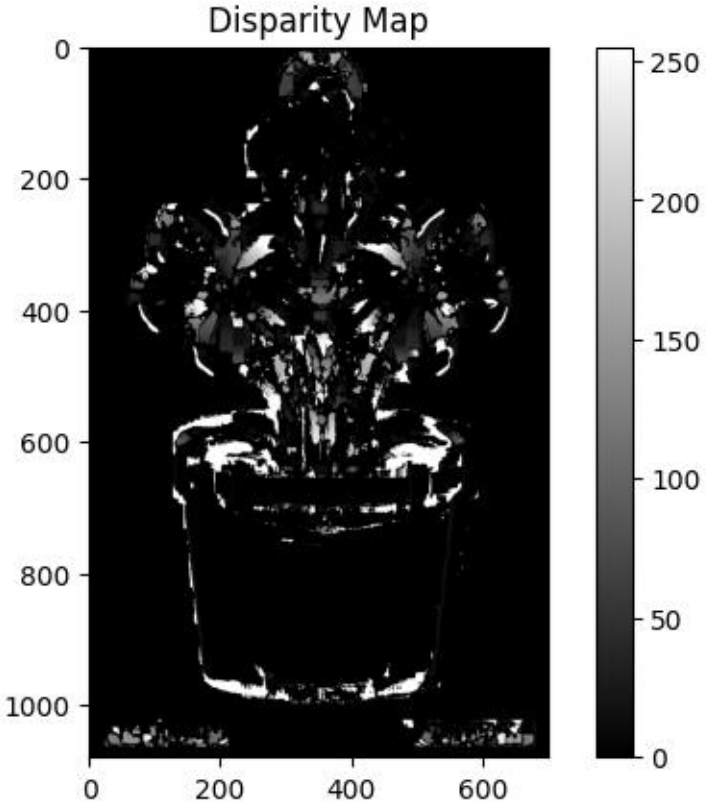
```
plt.imshow(normalized_disparity, cmap='gray')
plt.title('Disparity Map')
plt.colorbar()
plt.show()
# Print the shape of the disparity map
print("Disparity Map Shape:", disparity.shape)
```

Input:

right.jpg

left.jpg

Output:


Disparity Map
0
200
400
600
800
1000
0
200
400
600
250
200
150
100
50
0

Conclusion :

The experiment delved into depth estimation through stereo images using the StereoBM algorithm. This algorithm assessed pixel intensities in both left and right images to determine disparities, which signify the horizontal pixel shifts between the images. A normalized disparity map was generated and displayed using a grayscale colormap, offering valuable insights into the depth relationships within the scene by emphasizing objects with differing disparities. The key stages of the experiment encompassed loading the images, converting them to grayscale, creating a StereoBM object, computing the disparity map, normalizing it, visualizing it, and subsequently analyzing its overall structure.