PROBLEM

Project Description:

In this project, we are going to implement the concept of Stereo Vision. We will be given 3 different datasets, each of them containing 2 images of the same scenario but taken from two different camera angles. By comparing the information about a scene from 2 vantage points, we can obtain the 3D information by examining the relative positions of objects.

Dataset Preparation:

Please use this link to download the 3 Datasets that you will try to implement stereo vision on, individually. Each folder contains a calib.txt file

A brief explanation of the terms used in the txt files:

cam0,1: camera matrices for the rectified views, in the form [f 0 cx; 0 f cy; 0 0 1], where f: focal length in pixels; cx, cy: principal point

doffs: x-difference of principal points, doffs = cx1 - cx0 (here always == 0)

baseline: camera baseline in mm

width, height: image size

ndisp: a conservative bound on the number of disparity levels; the stereo algorithm MAY utilize this bound and search from d = 0 .. ndisp-1.

vmin, vmax: a tight bound on minimum and maximum disparities, used for color visualization; the stereo algorithm MAY NOT utilize this information

Pipeline for creating a Stereo Vision System

1. **Calibration**
    - First, we need to compare the two images in each dataset and select a set of matching features. You can use any inbuilt function for feature matching.
    - Estimate the Fundamental matrix using the features obtained in the previous step. Refer to section 3.2.2 in this link to get an overall understanding of Fundamental matrix estimation. You can use the inbuilt SVD function to solve for the fundamental matrix. You have to use the RANSAC method to estimate the fundamental matrix.
    - Estimate Essential matrix E from the Fundamental matrix F by accounting for the calibration parameters. You should implement the functions to estimate the Essential matrix and also to recover the rotation/translational matrices.
    - Decompose E into a translation T and rotation R.

PROBLEM

2. **Rectification:**
    - Apply perspective transformation to make sure that the epipolar lines are horizontal for both images.
    - You can use inbuilt functions for this purpose.
    - Print the H1 and H2, homography matrices for both left and right images that will rectify the images.
    - Plot the epipolar lines on both images along with features points

3. **Correspondence:**
    - For each epipolar line, apply the concept of the matching window (discussed in class such as SSD or Cross Correlation).
    - Calculate Disparity.
    - Rescale the disparity to be from 0-255 and save the resulting image.
    - You need to save the disparity as a grayscale and color image using heat map conversion.

4. **Compute Depth Image:**
    - Using the disparity information obtained above, compute the depth information for each image pixel. The resulting depth image has the same dimensions as the disparity image but it has depth information instead.
    - You need to save the depth image as a gray scale and color using heat map conversion.