Computer Vision Final Project

Student:

b04507009 何吉瑞 b04901021 陳傳諭 b04901022 黃家翰 b04901117 毛弘仁

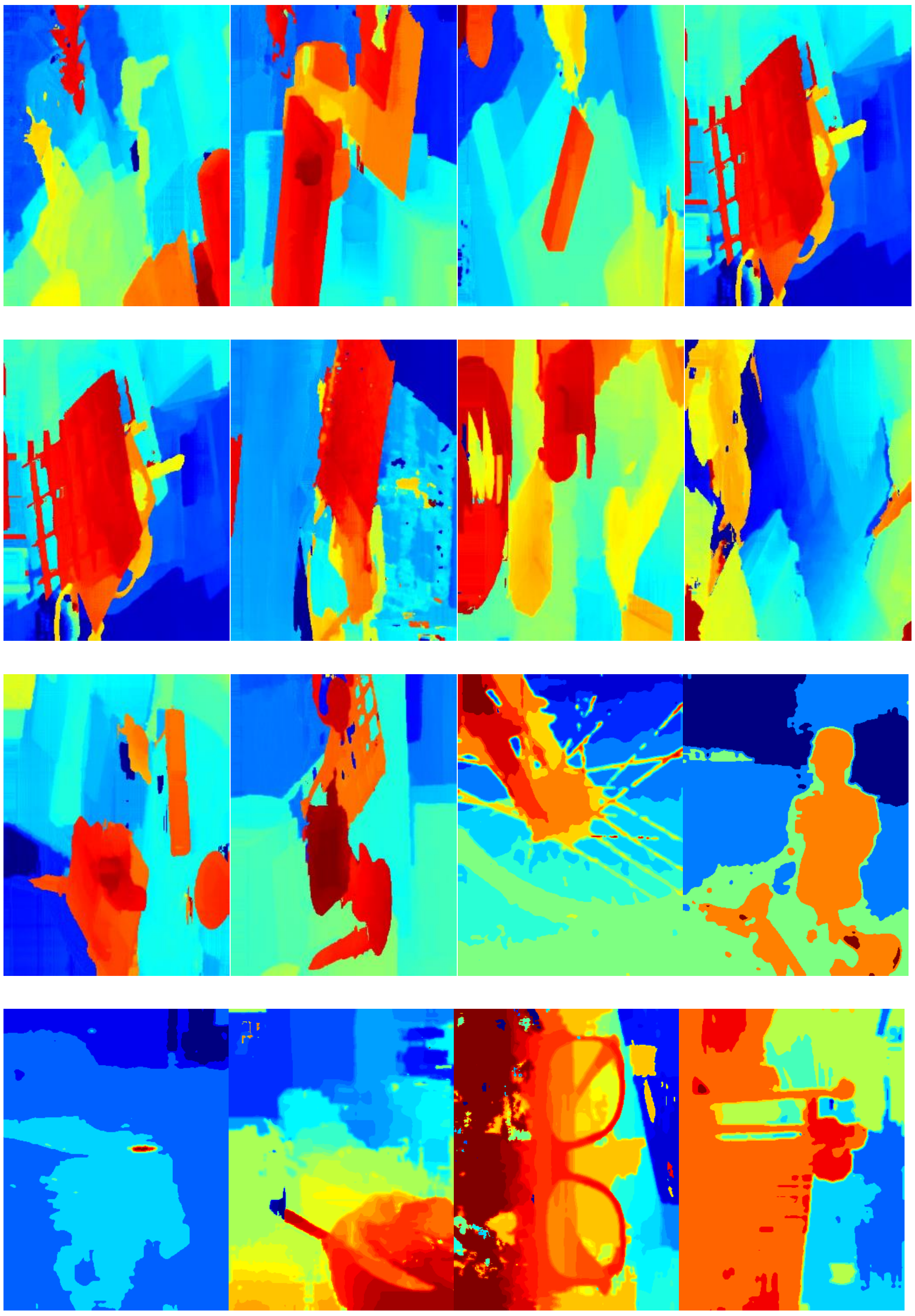
Environment dependency: OpenCV 3.4.12

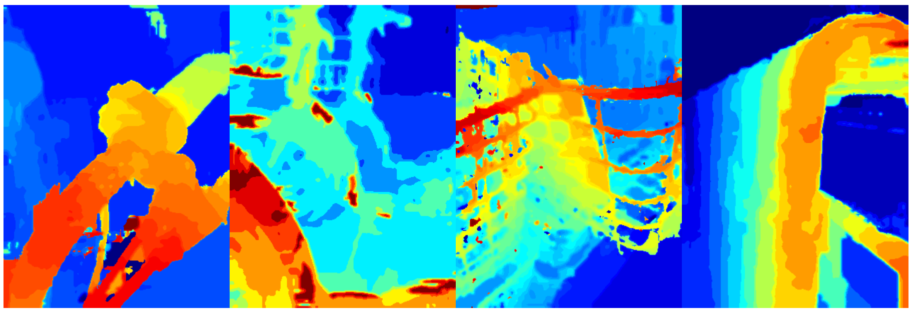
Results:

Synthetic:

Average error on synthetic data: 1.87

Images:





Runtime:

Synthetic (0~9):

242.37, 242.05, 221.80, 277.18, 263.18, 251.52, 222.14, 269.34, 218.27, 261.96 (sec)

Real (0~9):

64.66, 54.91, 70.27, 55.65, 104.91, 97.38, 59.60, 93.24, 62.23, 109.21

Challenges:

The correctness and speed of stereo matching algorithm highly depends on the maximum disparity of the input data. However, this parameter is not given in this project, which means we have to estimate it before computing disparity. A more challenging thing is that the disparity may be negative between real input data. Moreover, there are many repeated patterns in the real input data, which may result in ambiguity. Therefore, we have to improve the methods in previous homework and extend some functions to deal with the challenges in this project.

Approach:

We discuss the applied approach in two parts. The first part is about the enhancement of methods used in homework 4, and the second part is about the extension work to deal with the specific challenges in this project.

Enhancement:

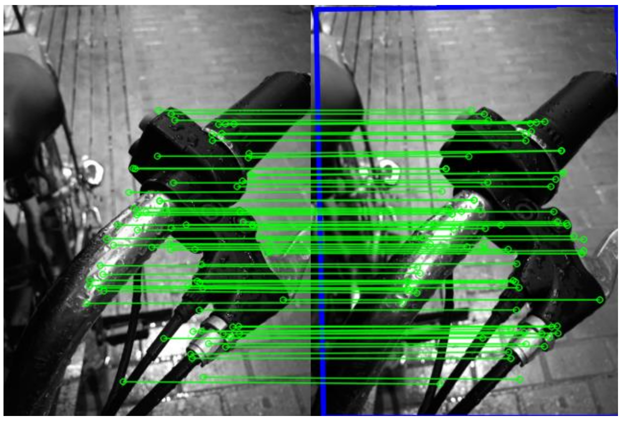
1. Matching Cost:
   We use normalized absolute difference cost and census cost to estimate the matching cost between two patches. Note that we use a large window (27) to avoid the ambiguity resulted by repeated patterns.
2. Cost Aggregation:
   For each pixel, we extend vertical and horizontal arms. The pixels involved in the arms have similar pixel value to the center pixel. Then, we integrate the cost in the range of the arms for each pixel. This process helps us to consider more information when the neighboring pixels have similar value.
3. Cost Optimization and Refinement:
   We use winner-take-all method in cost optimization. In the refinement part, we

first generate two disparity maps by above steps. One is from left to right, called left disparity map, and the other is from right to left, called right disparity map. Then, we test if these two maps correspond to each other by warping the one to the other. For every pixel in the left disparity map, if its value is close to the warped target at the right disparity map, we say this pixel is valid. Otherwise, this pixel is invalid. After this step, we can find invalid pixels. For each invalid pixel, we find its neighboring valid pixels and determine the value of invalid pixel by voting. We hold the vote five times to ensure the result is reliable enough. An example of outlier map is shown as follow. The invalid points are mainly in discontinuous points.

Extension:

1. Estimate the range of disparity:

   We match the two input images by SURF and calculate the displacement between the matched feature points. After excluding the extreme values. The range of the remaining values is our estimated range of disparity. An example of the matching result using SURF is shown as follow:



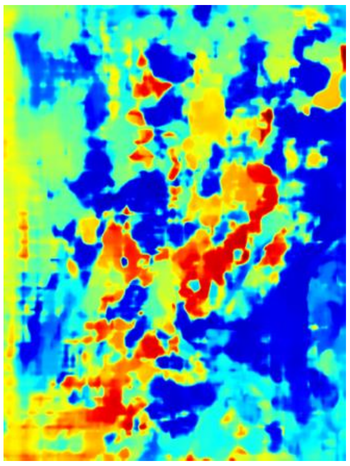
2. Histogram Equalization:

   In test data set, the intensity or contrast of some image pairs are different, which makes it difficult to compute the intensity difference cost correctly. The intensity difference of two images also affects the arms generation process. To adjust the intensity and contrast, we apply histogram equalization to all images. This pre-process not only adjusts the intensity of all images to the same level, but also enhances the contrast of all images, which makes the object border clearer. The costs have more discrimination in high contrast and border-clear images.

3. Image Segmentation
   In the disparity map, the disparity distribution is not smooth at the object border. And there are many outliers lies around the object border. To improve the result, we give an assumption that if the image can be will divide into several segments, then the disparity of each pixel in a segment is close. Based on the assumption, a image segmentation algorithm is applied to our algorithm. After the cost optimization process, the non-outlier pixels vote for the outliers in each segment. The disparity of outliers in a segment is determine by the major disparity of non-outliers in that segment.

Experiments with deep models:

1. Stereo matching: we tried using Pyramid Stereo Matching Network (PSMNet), but discovered that the pre-trained model yielded poor results on the Synthetic dataset (shown right is the output generated from TL0). We then fine-tuned the pre-trained model with the 10 Synthetic images, but performance was still worse than that achieved with our “traditional” method, so we decided to focus efforts on improving the “traditional” method instead. We did not re-train PSMNet from scratch using the 10 Synthetic images for risk of overfitting. In retrospect, if we had searched for models more thoroughly, we might have discovered PWC-Net like many other groups, and used it to achieve good results. However, using “traditional” methods has less risk of overfitting and hence could be considered more robust.



2. Segmentation: we have already introduced how image segmentation is an

important part of our processing pipeline. We were wondering if a pre-trained image segmentation model would perform better than “traditional” approaches, and ran the 10 Real images through a “ResNet50dilated + PPM_deepsup” model pre-trained on a scene parsing dataset (MIT ADE20K). From the output images (3 of which are shown below), we inferred that the model is good at segmenting objects commonly found in MIT ADE20K (e.g. cars and people), but performs badly on others (e.g. close-ups of baskets and glasses). For disparity estimation, it is actually alright if a segment does not enclose an entire semantic object, but we wish for a segment to preserve edges. Considering these criteria, the watershed algorithm was actually a better fit for our use case.