CS4186 Computer Vision & Image Processing

Assignment 2

SHYLA KUMAR Rohit (54581876)

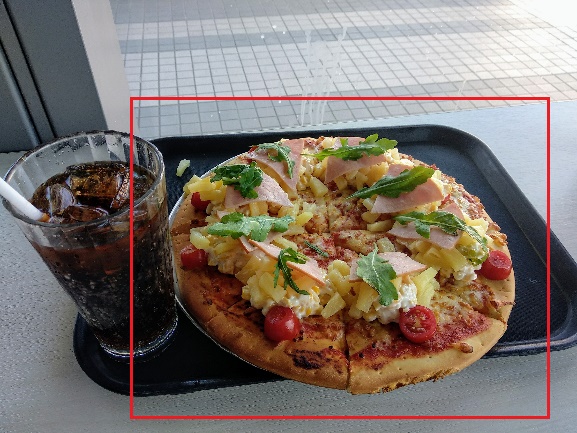
# Part A

### Method

After exploring the grabcut algorithm thoroughly in OpenCV, I used it to extract the contours for the foreground of the three given images. Before using the algorithm, I resized the images to 1200x900, 1200x900 and 900x1200 respectively as the cv2 implementation of grabcut is quite slow and the image resolution does not have a massive impact on the result. The grabcut algorithm is semi-automatic, which means a combination of computational and manual human work is required to obtain the best results. (See script – cs4186assignment2.py)

## Input







## Output





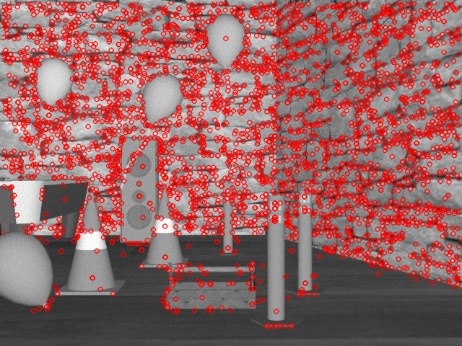
# Part B

## Reconstructing the disparity map

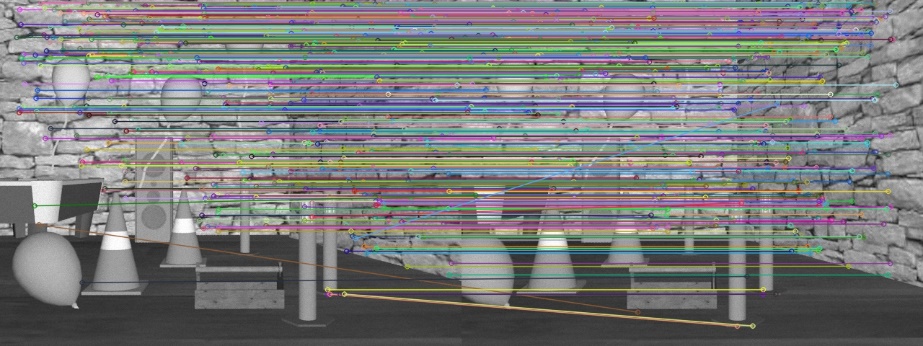
### Method

The following is the method used to generate the disparity maps for each pair of images

Step 1 – Use SIFT (with xfeatures2d.SIFT\_create()) to detect features and draw keypoints and derive descriptors from each pair of images (sift.detectAndCompute() method)

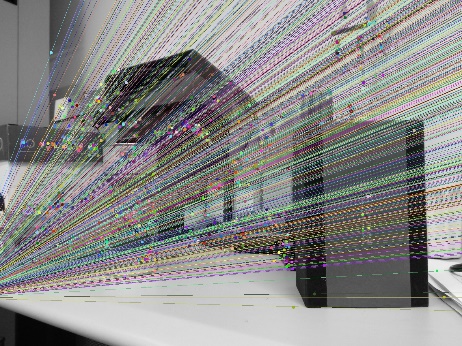
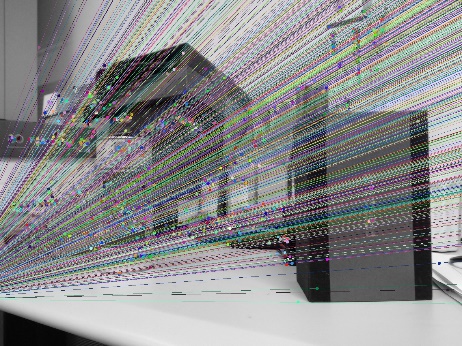




Step 2 – Use FLANN based matcher (substitute for BFMatcher()) to perform knn based matching, keep only the good matches and draw the results



Step 3 – Compute the fundamental matrix using cv2.findFundamentalMat() and compute and draw required epilines using the cv2.computeCorrespondEpilines() function



Step 4 – Rectify the images using cv2.stereoRectifyUncalibrated() function with the keypoints as a parameter

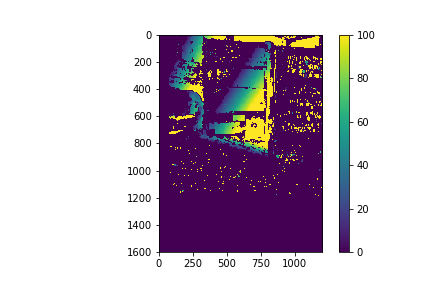
Step 5 – Warp the images using the rectmats obtained in Step 4 using warpPerspective()

Step 6 – Compute the disparity maps for each image pair using the cv2.StereoBM\_create() object

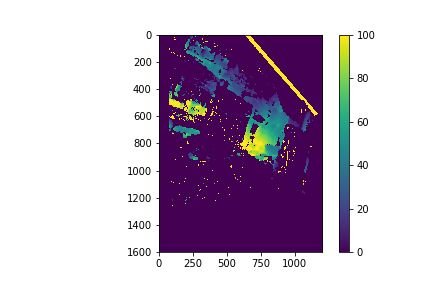
Scripts that implement the above algorithm for each pair of images – coneDisparityMap.py, teaDisparityMap.py, bookDisparityMap.py

### Results

Cones

Books

Tea

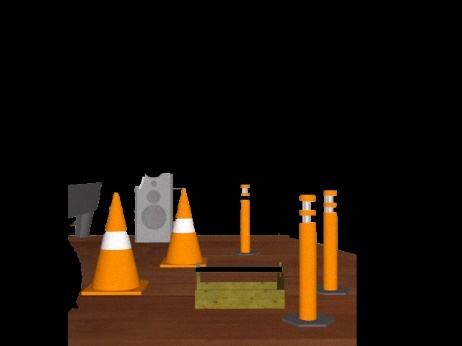
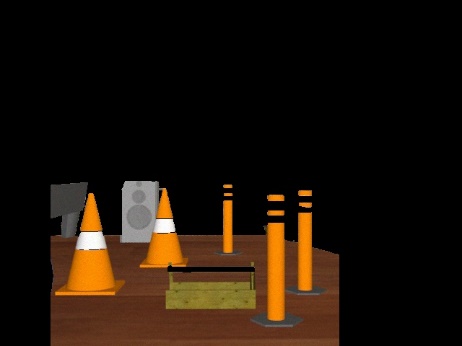


## Estimating Relative Height

### Method

The algorithm I’ve envisioned to complete this task is as follows:

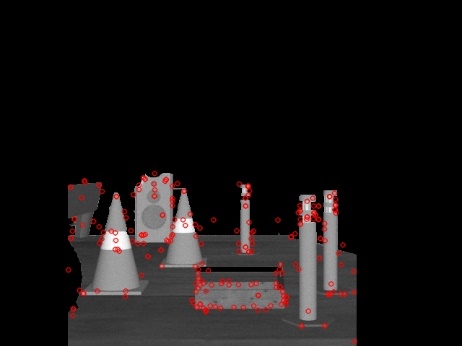
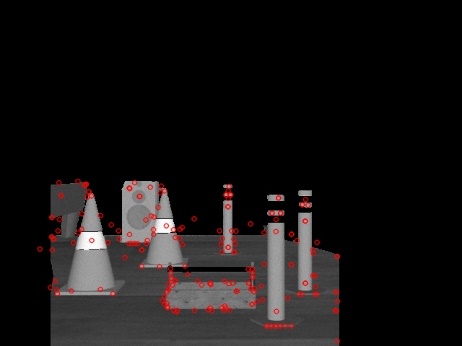
Step 1 - Use the grabcut algorithm to extract the foreground of each pair of images

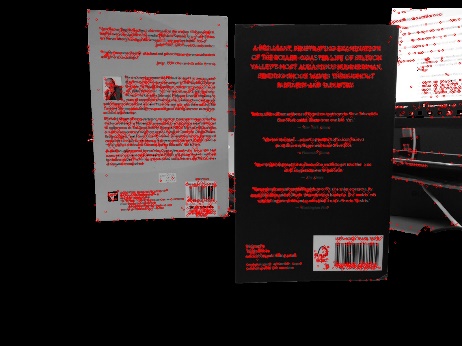




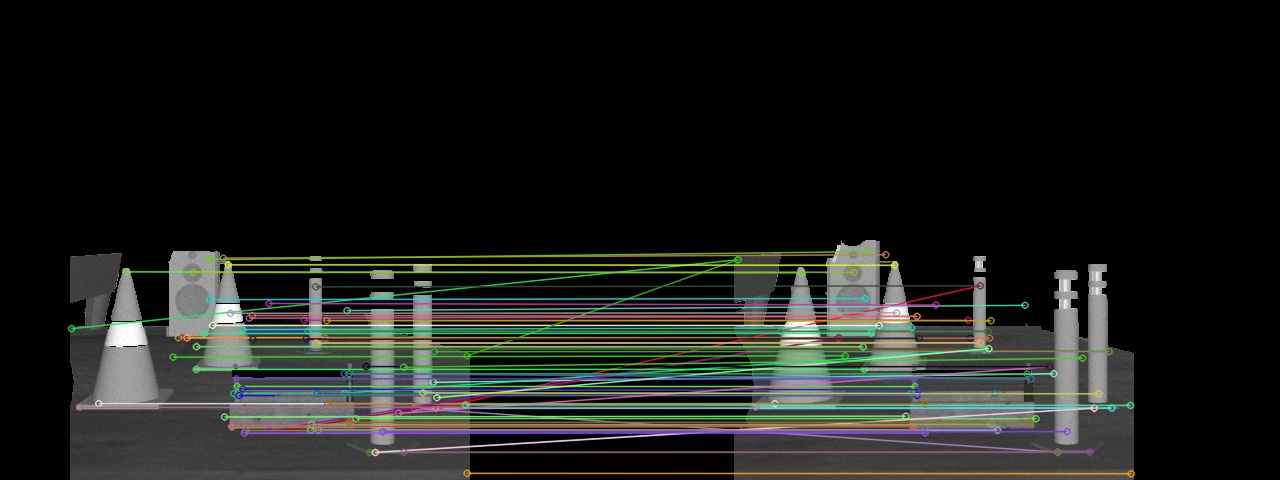


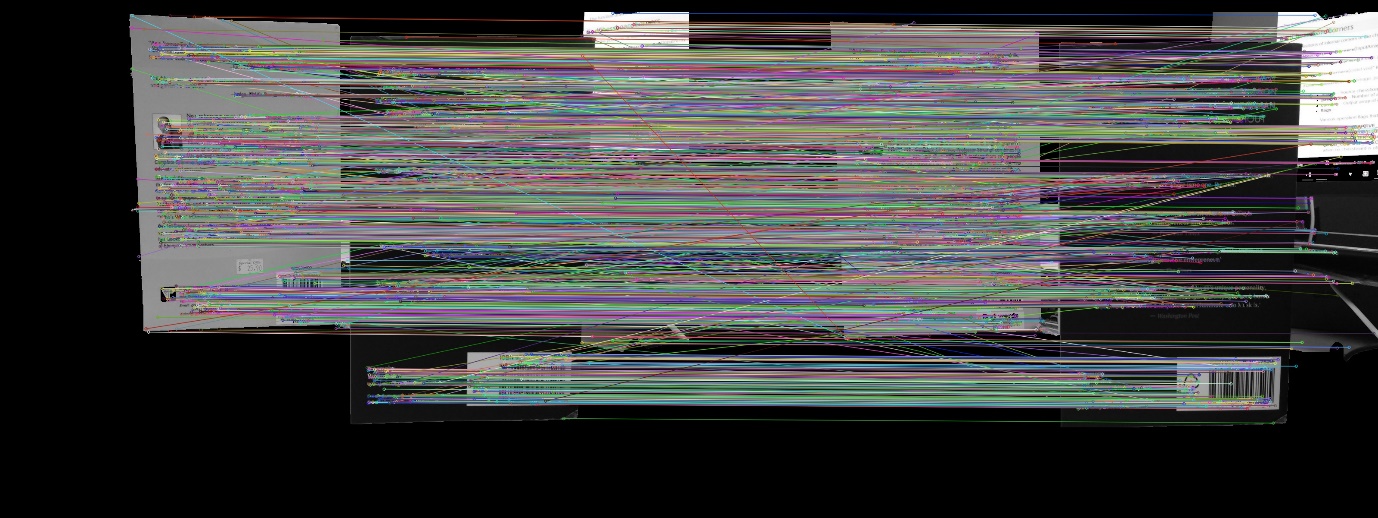
Step 2 - Use SIFT to find the key points and descriptors for each pair of images

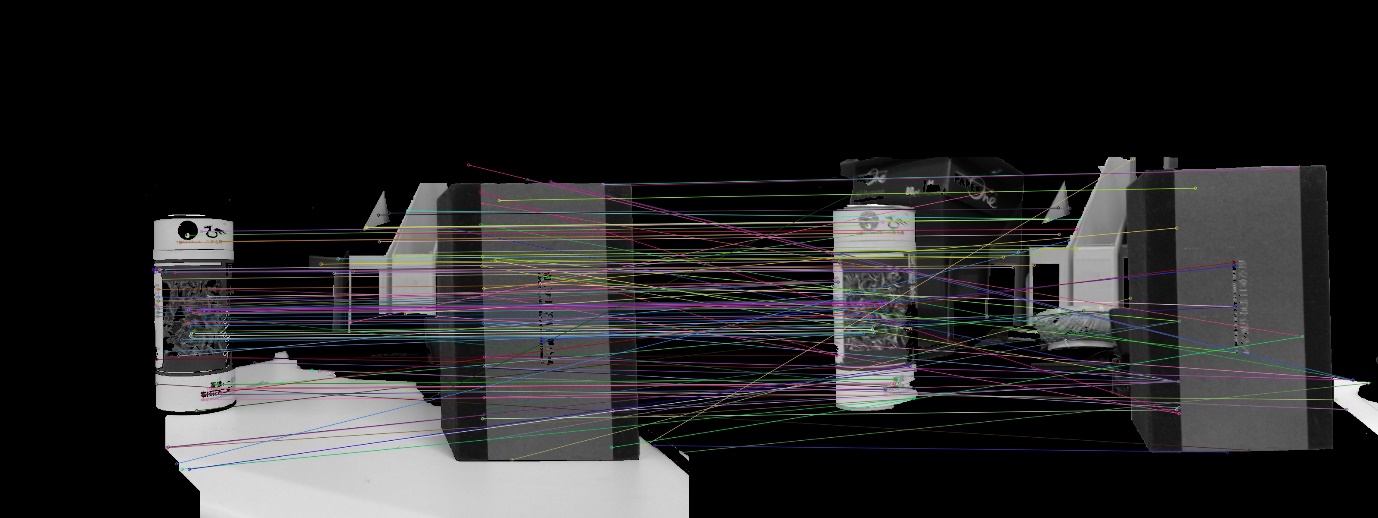




Step 3 – Use a FLANN based matcher to perform knnMatch() to match keypoints of the images







Step 4 – Only keep the best matching points and convert them to integers

Step 5 – Iterate over every point in both sets and compare their y coordinate (since we are looking for relative height) by diving the keypoint in the left image by its match in the right image

Step 6 – Store and return the average ratio obtained as the height ratio between the two images

Scripts that implement the above algorithm for each pair of images – coneHeightEstimate.py, teaHeightEstimate.py, bookHeightEstimate.py

### Results

1. Cones - Ratio of left to right = 1.0194120704428493
2. Books - Ratio of left to right = 0.9587935015301942
3. Tea - Ratio of left to right = 0.9977403036746977