Bachelor in Computer Science

COMP 425: Computer Vision

Homework 2 Report

Student Name: Alejandro Leonardo García Navarro

Student ID: 40291834

Email: al.garcia636@gmail.com

Instructor: *Yang Wang*

Academic Year: 2023/24

Date: March 18th, 2024

Contents

1 Introduction

This report encapsulates the outcomes and insights gained from the completion of the second homework assignment for the course COMP 425 (Computer Vision). The assignment is structured into two distinct parts: Hough Transform and RANSAC Homography Estimation, each designed to deepen our understanding and practical skills in fundamental computer vision techniques.

In fulfilling the requirements of this homework, a series of algorithms were implemented and applied to various images, as specified in the assignment instructions. The first section of the homework requires the implementation of the Hough transform for detecting straight lines in an image, emphasizing hands-on experience with edge detection and parameter space exploration. This segment introduces the concept of using regions of interest (ROIs) for focused analysis and the challenge of identifying significant patterns within the parameter space defined by the Hough Transform.

The second sections focuses on the RANSAC algorithm for homography estimations and its application in a simple augmented reality task. This involves estimating the transformation between two images and applying this knowledge to accurately superimpose a third image onto a target image.

This report not only presents the results obtained through the implementation of these algorithms but also reflects on the learning process, highlighting both the technical challenges encountered and the solutions devised to overcome them.

To ensure clarity and ease of understanding, all code has been meticulously commented, and the report is organized to systematically present the approach, results, and analysis for each part of the assignment.

2 Part 1 (Hough Transform)

In this very first part of the project, it was required to write a script that carried out the following:

1. Load the ”road.jpg” image, convert it to grayscale, then run Canny edge detector to find edges.
2. Create a binary mask to work with a subset of these edge points within an ROI.
3. Multiply the mask with the edge map to obtain an edge map containing only those edges in the ROI.

4. Implement the Hough transform to find two major lanes.

Let's begin with **Step 1**, already described above. This process was done thanks to the library *skimage*, which helped in loading the input image, converting it to grayscale, and finally running the canny edge detector. Figure 1a corresponds to the result of applying all of these.

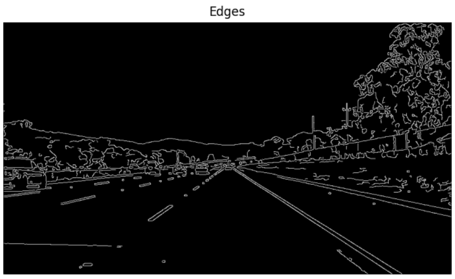
Moving on with **Step 2**, it was accomplished thanks to the given function *create_mask(H, W)*. This function generates a binary mask for a specified height (H) and width (W). It iterates through each pixel position in the mask matrix and sets it to 1 if it falls within a triangular region defined by two linear equations. These equations are used to create a diagonal cut across the image, effectively masking out the upper portion of the image. Additionally, it sets the last few rows of the mask to 0, ensuring that the bottom portion of the image is also masked out. Finally, it returns the generated mask. Essentially, the function creates a mask that isolates a triangular region of interest (ROI) within the image. As a result, Figure 1b is obtained.

Step 3 simply required to multiply the mask with the edge map, giving as a result Figure 2a. Finally there is **Step 4**, which constitutes the longest step of the section. It is done like so, giving as output Figure 2b:

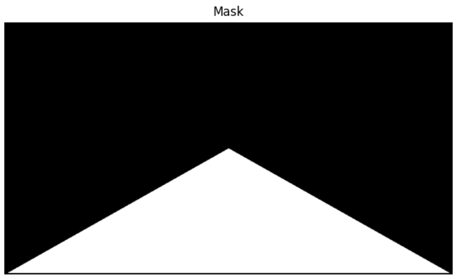
1. **Perform Hough Transform:** Use the Hough transform on an edge-detected image to detect lines, representing potential lanes.
2. **Right Lane Detection:** Identify the peak in the Hough space to detect the right lane. After that, extract the parameters of the detected line and generate coordinates for the right lane line.
3. **Non-Maximum Suppression (NMS):** Then, conduct NMS to suppress neighboring peaks around the detected right lane peak, ensuring accuracy in lane detection.
4. **Left Lane Detection:** Finally, the process is repeated to detect the left lane, using peak detection and parameter extraction similar to the right lane. Coordinates for the left lane line are generated accordingly.

3 Part 2 (RANSAC, Homography)

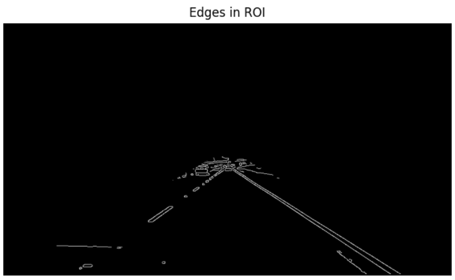
This last segment of the project focused estimating the homography between "cv_cover" and "cv_desk". Then using the estimated homography to warp "hp_cover" appropriately and composite with "cv_desk", so that the Harry Potter bookcover completely covers the Computer Vision bookcover in "cv_desk". This involved completing the code for some functions in "homography.py":



(a) Edges



(b) Mask



(a) Edges in ROI

(b) Output

- **matchPics(I1, I2):** This function runs SIFT on two images and finds candidate matching points between two images.
- **computeH_ransac(matches, locs1, locs2):** This function implements the RANSAC algorithm for estimating the homography.
- **compositeH(H, template, img):** This function uses the estimated homography H to warp a template image ("template") with a target image ("img"), then return the composite image.

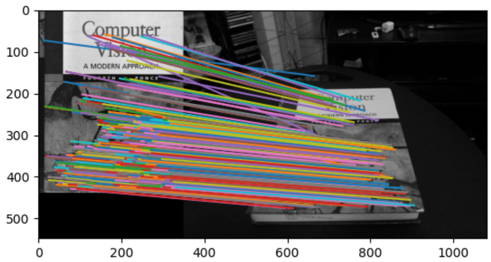
After completing these functions, the following process was followed:

1. Find candidate matching pairs between two images (Figure 4a).
2. Use RANSAC to estimate homography (see matrix 3) and find inlier matches (Figure 4b).
3. Visualize the bounding box in target image (Figure 5a).

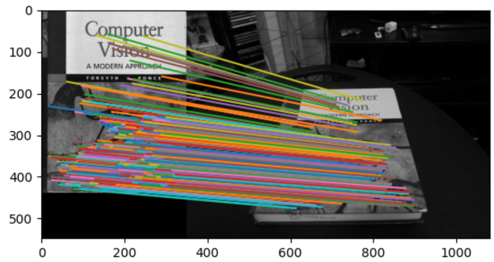
4. Create the final composite image (Figure 5b).

$$\begin{bmatrix} 7.45875366e-01 & -3.40483774e-01 & 2.37345496e+02 \\ -2.63541933e-03 & 2.32032698e-01 & 1.91469217e+02 \\ -8.44643551e-06 & -9.08272224e-04 & 1.00000000e+00 \end{bmatrix}$$

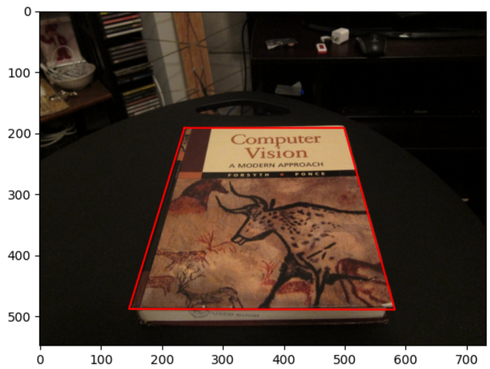
Figure 3: Estimated Homography



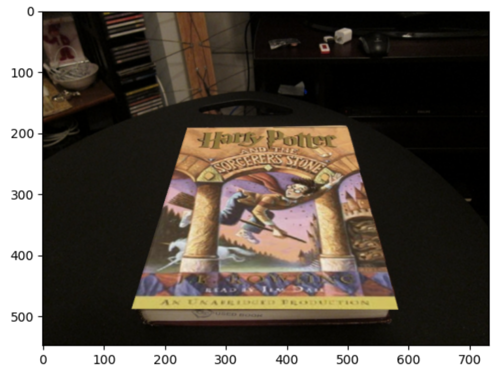
(a) raw matching result



(b) matching result after RANSAC



(a) visualization of bounding box



(b) final result

4 Conclusions

The completion of this second homework offered a deep dive into essential computer vision techniques, spanning key concepts such as Hough transform, RANSAC and homography.

Implementing these algorithms provided practical insights into their operational intricacies and their vital roles in analyzing and interpreting visual data. The tasks underscored the importance of finding candidate matching pairs in relating two images.

Challenges encountered along the way were key in refining problem-solving skills and deepening the understanding of theoretical concepts through hands-on application. The visual results not only demonstrated the effectiveness of each method but also highlighted the importance of careful technique selection based on the desired outcome.