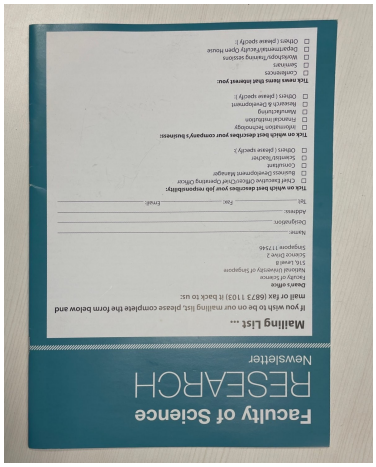
National University of Singapore

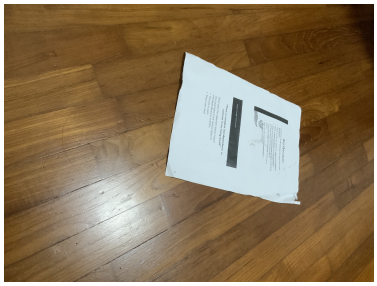
Department of Mathematics

03/2023 DSA5203 Visual Information Processing and Interpretation **Project 2**

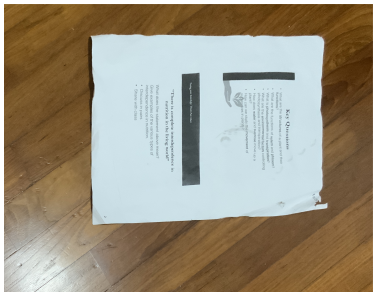
(a) Input image



(b) Rectified

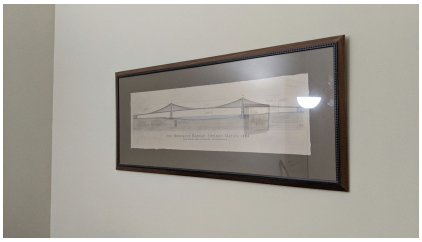


(a) Input image

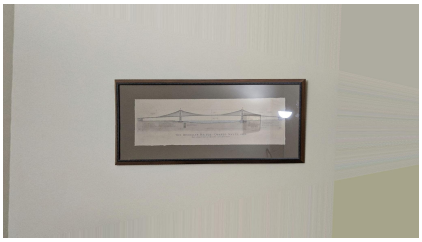


(b) Rectified

Figure 2: Left: Input image; right: Image after rectification



(a) Input image



(b) Rectified

Figure 3: Left: Input image; right: Image after rectification

Goal

Image rectification is the process of rectify an image distorted by affine transform duw to viewpoint, which is often seen in document scanning and digital photography of painting. The goal of this homework is to implement an image rectification algorithm to warp an input image where an object with rectangular boundary is prominently present with a monochorome monochrome background. The resulting rectified images will have the boundary of the object of interest are either along horizontal or along vertical direction of an image.

Individual project

It is an individual project. You are supposed to work alone.

Introduction

The project involves tackling a problem that doesn't have a closed-form solution and can't be solved by simply copying an algorithm from a textbook. Instead, students are encouraged to use a combination of tools taught in the classroom or the related routines from standard industrial packages such as opencv, along with their own ideas, to arrive at a solution.

There are many possible paths which might lead to a working solution.

- For example, one possible approach that only replies on the techniques taught in this module is as follows. Firstly, detecting all lines from the output of the hough transform on the edge map produced by Canny edge detector. Secondly, identifying quadrilateral that reflect the boundaries of the object of the interest. Thirdly, estimating the perspective transform that maps quadrilateral to a rectangular. Lastly, warping the image using the estimated perspective transform using backward warping.

- Another approach can be as follows. Firstly, detecting cornor points from the edge map produced by Canny edge detector. Secondly, identifying the cornor

points that correspond to the cornor of the object of interest. Thirdly, estimating the quadrilateral from identified cornor points. Fourthly, estimating the perspective transform that maps quadrilateral to a rectangular. Lastly, warping the image using the estimated perspective transform using backward warping.

- Or you can used the existing routines from cv2 to run some contour detection to identity the boundaries of possible objects. Then, identify the correct quadrilateral that corresponding to the boundaries of the object of interest. Then, estimating the perspective transform that maps quadrilateral to a rectangular. At last, warping the image using the estimated perspective transform using backward warping.

Remark on the expected result

(a) It is a good practice to align the center of the rectified image with the center of the rectangular after warping to avoid cutoff of the object of interest.
(b) The rectified images can be different in terms of heading-direction. As long as the object of interest is shown with its boundaries lying on horizontal and vertical direction, it is considered as a valid result.
(c) There is no standard definition of how the image size after warping should be. A common expectation in practice is that size of warped images is in the same range of original image, e.g. the size of warped image should be more than $\frac{1}{2}$ and less than 2 times of original image size.
(d) You should keep the background as much as possible, do not only give one warped image which only contains the object of of the interest without any background. Such result will lead to some point deduction even though the rectification is correct.
(e) It is OK that there are some black (or white) regions in your rectified image, as those are the regions not shown in the original input image.

Figure 4: Image rectification is not foreground object extraction. do not crop all background.

Some hints on identifying the quadrilateral of the object from many lines

It often happens that there are multiple lines detected in hough transform, or some other routines. The challenge is how to determine the lines out of many that belongs to the boundary of the main object. Some heuristics are listed below which might be helpful to accurately detect object boundaries.

- Excluding the lines whose edge points are not connected well (there existing big empty gaps among edge points)
- Using Harris corner point detection to identify lines whose intersection is indeed a strong corner point in the image, which imposed that the boundary of the object should intersects inside the image
- Choose lines such that the resulting parallelogram has sufficiently large area.

Routine for rectification

The follow main routine are expected.

imrect(im1)

Warping the input im1 distorted by an affine transform

Parameters: **im1: numpy.ndarray**
an array representing the image for rectification

Returns: **out: numpy.ndarray**
Rectified image

Academic Integrity

You are expected to implement the project on your own. You may discuss the ideas with your classmates, but the sharing of code is strictly prohibited. You are not allowed to use any code from online sources or from other classmates' projects, which will be checked via some standard code plagiarism checking system. Any suspicious code identified by the system will lead to further investigation. Anything that breaks these principles will commit a violation of theHonor Code, and will be reported to the university for the corresponding disciplinary action.

Forbidden function: Most functions in opencv is are allowed. However, you may NOT directly call the routine from existing library or from online source for image rectification. The following sub-routines used in the suggested approach are allowed, which includes Canny, hough transform, perspective transform estimation, warpperspectivetransform, findcontours, arclength. If you are unsure about whether a function in existing package is allowed, please check with me in advance.

Data for testing and experiments

Two images are included and their rectified versions are included for you to test the code. Note that your results might be different from the rectified images given, in terms of resolution, size and principle orientation.

Grading policy

The grade will depend on the performance of your methods on images attached in the project and performance of your method on the images used for testing you code (blind to you), and the quality of your report. Note that the performance is measured by both rectification accuracy and image quality. Your results do not need to be exactly the same as the results attached on 2 test images. Your result can have different heading-direction, different size and different rectified area. As long as the object is rectified correctly with sufficient image details. It is viewed as correct solution.

Submissions

You are required to submit the following:

1. *Python file "main.py" contains the function "imrect" and other supplementary function. A reference implementation of main.py is attached in the zip file. Please modify it accordingly. To facilitate the grading process, do not modify the IO format of "main.py".*

2. *Two rectified images, in PNG image format, from the given test images*

3. *A report on the discussion of your method and any implementation tricks and modifications you believe it contributes to the performance.*

Please package all your files in a single zip file named by the assignment ID and your student ID (e.g. hw2-u839393a.zip). Then upload the zip file to Canvas. The deadline for submission is **Sunday. April-02-2023**. *Any submission after the deadline will lead to score deduction.*

If you have any question, please contact Dr. Ji Hui by email: matjh@nus.edu.sg for help.