Coursework: Computer Vision

Vinay P. Namboodiri and Da Chen

February 11, 2024

Coursework number	1
Percentage of unit	50%
Submission location	Moodle
Date of Announcement	**12th February 2024**
Group formation deadline	**19 February 2024 at 8 PM**
Submission deadline	**26th April 2024 at 8 PM**
Submission components	PDF report and source code in a zip file
Anonymous marking	No

1 Overview

In this coursework, you have to solve for three tasks. The aim is to obtain an understanding of solving computer vision tasks.

- In the first task, the aim is to use concepts from edge detection and line fitting to be able to measure angles for a pair of lines.
- In the second task, the aim is to use filtering, convolution and template matching to be able to obtain intensity based template matching
- In the third task the aim is to use feature matching using SIFT/SURF features to match the templates using an interest point based object matching routine

Together, the three tasks provide an opportunity to work with a variety of computer vision concepts. You are free to use Python or Matlab to solve your tasks.

2 Submission Details

This coursework is worth 50 points from the overall mark of 100 for this unit. Marks are given beside each task. A report will be your main method of assessment. Students should form groups of up to 3 individuals and submit one report per group. Students can choose their partners until **Feb 19th**. The marks for the group will be awarded as follows: The marks for the whole group that will be graded based on the project will be worth 80% of the coursework marks. 20% of the marks will be obtained by peer-review.

Further information about the moodle tool for peer review marking is available at the following url: `https://teachinghub.bath.ac.uk/tools-and-resources/tel/group-peer-review-tool/`. Each student has to ensure the following, that their group does submit the project report (one submission per group) and that they complete their peer marking for their peers (their peers in the group will do it for them). Thus for a three person group, you would need to give marks for the work for each of the three members and that will determine their final mark. If there are fewer than three members you would still need to ensure that you provide the peer-marking for your group. The peer review marking url has access to how the marking is done.

The submission deadline for your report with code and submission of the peer review marking is:
26th April 2024 20:00 online Moodle submission of your final report.

The main part of your report should not exceed 3000 word limit. After the main part please attach the codes, references if any and any appendices (these are excluded from the word limit). The report should be submitted in PDF format. Further, you should include all codes relevant to your implementations as an appendix or appendices with a clear referencing to the main body of your report. Codes must be well commented (indicating major steps). You should ensure that it is possible for us to run your code on additional test samples. This implies that the code is self-contained without any absolute image paths and any appropriate readme if required should be provided.

First and foremost, your report should demonstrate and evaluate your results. It must include figures and screenshots with appropriate resolutions. Evaluation can be qualitative (how it looks) and, wherever possible, quantitative (tested numerically). You should also concisely explain your approach to solve each task, the choices you make (e.g. hyper-parameters etc), and answer questions in each part. The marks will be decided based on the marking guideline specification that we will follow for this unit.

Figure 1: Measure the angle

3 Detailed Specification of the coursework

3.1 Task1: Measuring the angle (33.33 pts)

In this particular task, given an image with a pair of lines we need to ensure that we can measure the (smaller) angle formed by the two lines. An example image is provided in figure 1. This should be obtained irrespective of the position of the lines in the image.

The main ideas that you can use for solving the problem are edge detection and line fitting. You could use library functions for edge detection. Further, you would be expected to explore line fitting on the edges and subsequently you should measure the angle through the appropriate code. You should demonstrate that you are able to solve the task in a variety of situations. A set of example images is provided for this task.

Analysis: You should provide an analysis with respect to the choice of edge detection function and the range of parameters used for line fitting.

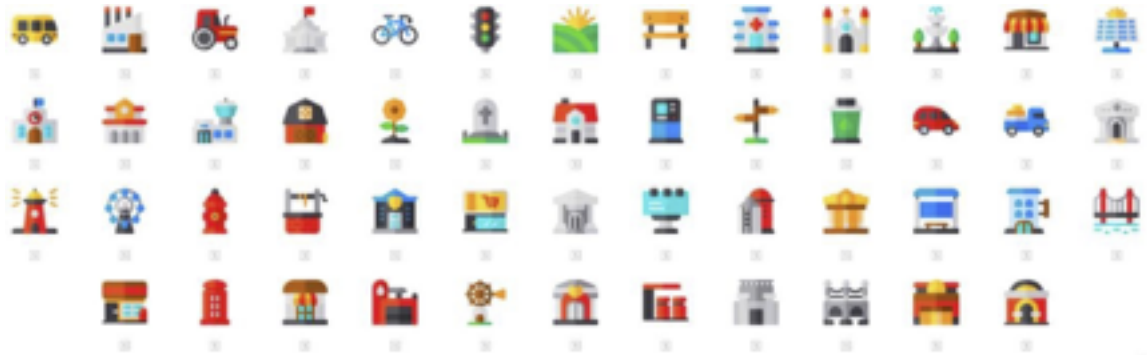

Figure 2: Training images: icons of the village dataset

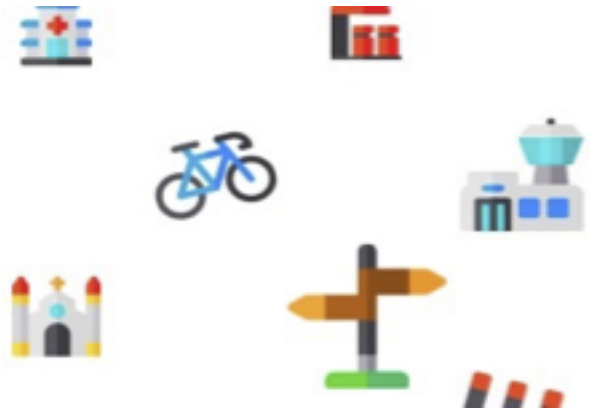

Figure 3: One of the test images to be used for recognition

3.2 Task 2: Intensity-based template matching (33.33 pts)

In this task, you are provided with a set of train images that consist of icons of objects. The set of icons in the training image are provided as shown in figure 2. In this task you will implement algorithms for matching images, similar to the principle of playing Dobble.

Output: Detect objects in the Test images, recognize to which class (from the 50 Training image classes) they belong, and identify their bounding boxes. One example of a test image is provided in figure 3. For visually demonstrating this, the function should open a box around each detected object and indicate its class label. Demonstrate example images(s) of the outcome detection in your report.

Evaluate your algorithm on all Test images and report the overall False Positive (FPR), True Positive (TPR) and accuracy (ACC) rates, and the

average runtime. Show and explain cases where this scheme finds difficulty to perform correctly. Give insight about the complexity (memory and runtime) of your algorithm in terms of the problem size e.g. how does the complexity scale in terms of the number of image pixels, pyramid scales?

Creating scaled templates: You should create a Gaussian pyramid of appropriate scale levels for each RGB Training image. This task includes implementing appropriate Gaussian blurs and subsampling of the training images through a hierarchy. You may use your own convolution algorithm to apply the Gaussian blur, or you may use the library function. In your report please explain the steps you take to build your Gaussian pyramids and visually demonstrate an example pyramid for a Training image.

After creating scaled templates, you should append this set by creating Gaussian pyramids for an appropriate number of orientations per class. This creates the overall scaled and rotated templates. Justify your choice of parameters (e.g. Gaussian kernel specs, initial/final scales, number of scale levels e.g. how many octaves, number of rotations etc). To justify quantitatively the chosen number of scales, plot or tabulate the overall object detection performance against these parameters and pick parameters that achieve a reasonably good runtime vs. accuracy trade off. Evaluate your algorithm on all Test images to report the overall Intersection over Union (IoU), False Positive (FPR), True Positive (TPR) and Accuracy (ACC) rates, as well as the average runtime. Refer to the following report `http://host.robots.ox.ac.uk/pascal/VOC/voc2012/devkit_doc.pdf` section 4.4 for further information about the evaluation metrics.

Pre-processing: For each (scaled) template set the background to zero, similarly for the Test data. Intensity-based matching (related to output 2): Slide each (scaled and rotated) template over the Test image to measure their similarities across x-y axes. An intensity-based similarity score based matching function. Explain why this function is suitable to measure image similarities? Define appropriate thresholds on this similarity score, a proper non-maxima suppression strategy to avoid false positives within the neighbourhood of the detected objects and return the output in terms of class label and bounding box: [left top right bottom] for each object.

3.3 Feature matching using SIFT or equivalent features (33.33 pts)

Write a function which takes an image from the same dataset for training and testing as in the previous task.

Output: Detect objects in the Test images using SIFT or equivalent fea-

tures (such as SURF), recognize to which class they belong, and identify their scales and orientations. Similar as Task2, for visual demonstration the function should open a box around each detected object and indicate its class label. This box is scaled and rotated according to the object's scale and orientation. Demonstrate example images(s) of the outcome detection in your report. Besides, demonstrate example images(s) that shows the feature-based matches established between the recognised objects and a Test image, before and after the outlier refinement step.

Evaluate your algorithm on all Test images to report the overall Intersection over Union (IoU), False Positive (FPR), True Positive (TPR) and Accuracy (ACC) rates, as well as the average runtime. Refer to the following report `http://host.robots.ox.ac.uk/pascal/VOC/voc2012/devkit_doc.pdf` section 4.4 for further information about the evaluation metrics. Show and explain cases where this scheme finds difficulty to perform correctly. Compare the SIFT/SURF results to that of Task2 algorithm e.g., does it improve the overall speed or accuracy? How much? Why?

Main steps: You first extract keypoints and feature descriptors from your Test and Train images using standard SIFT or SURF feature extraction function from a library. Then you match features between images which will give you the raw noisy matches (correspondences). Now you should decide which geometric transform to use to reject the outliers. Finally, you will define a score on the obtained inlier matches and will use this to detect the objects (icons) scoring high for a given Test image. A basic score is counting the inlier matches, but if you have a better idea report and justify that too.

Similarly, you will have some hyper-parameters to tune. This includes the number of Octaves and the (within-octave) Scalelevels within SIFT to build scale-spaces for keypoint detection, and the MaxRatio parameter within the matchFeatures function to reject weak matches. How are these parameters set for this task? Show quantitatively why.

4 Marking criterion

Your report and your code will be assessed jointly. Detailed marking criteria are listed in the attached marking-template.pdf.

5 Late submissions

The university policy will be followed on late submissions. If a piece of work is submitted after the submission date, the maximum possible mark will be

40% of the full mark. If work is submitted more than five days after the submission date, student will receive zero marks. If you need an extension, please contact your Director of Studies.

6 Plagiarism

Do not plagiarise. Plagiarism is a serious academic offence. For details on what it is and how to avoid it, please visit `http://www.bath.ac.uk/library/help/infoguides/plagiarism.html`