COMP9517

Lab 2, T3, 2020

The lab files should be submitted online.

Instructions for submission will be posted closer to the deadline.

Deadline for submission is Week 4, Tuesday, Oct 6th, 2020, 23:59:59, AEST.

Objectives: This lab presents a revision of important concepts from week 3 and aims to make you familiar with implementing specific algorithms.

Materials: The sample images to be used in all the questions of this lab are available in WebCMS3. You are required to use OpenCV 3+ with Python 3 (see specific version to use for this lab in section 1.1).

Submission: Question 3 is assessable **after the lab** and is **worth 2.5% of the total course marks**. Submit your code and results as a Jupyter notebook in a zip file via WebCMS3 by the deadline.

1 SURF: Speeded-Up Robust Features

A well-known algorithm in computer vision to detect and describe local features in images is the Speeded-Up Robust Feature transform (SURF) [1,2]. Its applications include object recognition, mapping and navigation, image stitching, 3D modelling, object tracking, and others.

A SURF feature of an image is a salient keypoint with an associated descriptor. SURF computation is commonly divided into two steps: 1) detection and 2) description.

At the end of the detection step, for each keypoint the SURF algorithm computes the:
- keypoint spatial coordinates (x, y),
- keypoint scale (in the scale space),
- keypoint dominant orientation.

The description step computes a distinctive 64-dimensional feature vector for each keypoint (128-dimensional vector can be computed for more distinctiveness at the cost of time). SURF is designed to be invariant to scale and rotation (Upright-SURF or U-SURF is a rotation invariant version, which improves speeds). Moreover, the algorithm offers decent robustness to noise, illumination gradients, and affine transformations.

1.1 OpenCV implementation of SURF

In OpenCV the SURF algorithm is available only in the non-free module (OpenCV has both free and non-free modules) [3]. The algorithm has been patented by the creator but can be freely used for academic and research purposes. The non-free modules can be found in the opencv-contrib package. You will first need to install this package, as shown below, and then you can use the SURF module.

Python via Anaconda	**Python via other sources**
1. Create and activate the virtual environment [4] 2. Install a python version >=3.6 and <=3.8 `conda install python=3.6` 3. `conda install -c michael_wild opencv-contrib`	1. Create and activate the virtual environment [5] 2. Install a python version <=3.7 3. `pip install opencv-python==3.4.2.16` 4. `pip install opencv-contrib-python==3.4.2.16`
Make sure that you use the correct python version with opencv and opencv-contrib.	

QUESTION 1: COMPUTING FEATURES

a) Given an image (wagon.jpg) compute its SURF features
b) Display the image with the keypoints on the image.
c) Experiment by providing different Hessian Threshold values to the algorithm and find the relationship between the Hessian Threshold and the number of keypoints detected. (Hint: Start off with a value of 400 and trial increments of 100). Display the image with fewer keypoints.

QUESTION 2: MATCHING TRANSFORMED IMAGE FEATURES

a) Given three points from the input image ([[0,512], [390,520], [390,0]]) and their corresponding locations in the output image ([[50,475], [390,520], [420,50]]) compute the Affine Transformation matrix and transform the image using the matrix.
b) Compute the SURF features of this transformed image and show the keypoints on the transformed image.
c) Match the SURF descriptors of the keypoints of the transformed image with those of the original image using the nearest neighbour ratio method (Hint: Brute-force matching is available in OpenCV for feature matching). Draw the matches using `cv2.drawMatchesKnn(,,,)` function.

QUESTION 3: ASSESSMENT QUESTION (2.5 marks)

a) Given two object images (wheel.png, toy.png) compute the SURF features of each and display the image with its keypoints. Also output the number of keypoints detected for each image.
b) Match the features in each, to those of the original image provided (wagon.jpg). Find the object image that is part of the original image. It will be the object image that has the highest percentage of matching descriptors (if the object image has 'n' keypoints and if 'm' of those have matching descriptors in the original image, percentage of matching descriptors will be m/n*100).
c) Display the matches between the object chosen in step 3.b and the original images.

Note: Refer to https://docs.opencv.org/3.4.3/df/dd2/tutorial_py_surf_intro.html for an example of computing SURF features and showing the keypoints on the image.

2 REFERENCE

[1]. Bay, Herbert, Tinne Tuytelaars, and Luc Van Gool. "Surf: Speeded up robust features." Computer Vision–ECCV 2006. Springer Berlin Heidelberg, 2006. 404-417.

[2]. Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool, "Speeded Up Robust Features", ETH Zurich, Katholieke Universiteit Leuven

[3]. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_surf_intro/py_surf_intro.html

[4]. https://docs.conda.io/projects/conda/en/latest/user-guide/getting-started.html

[5]. https://packaging.python.org/guides/installing-using-pip-and-virtual-environments/