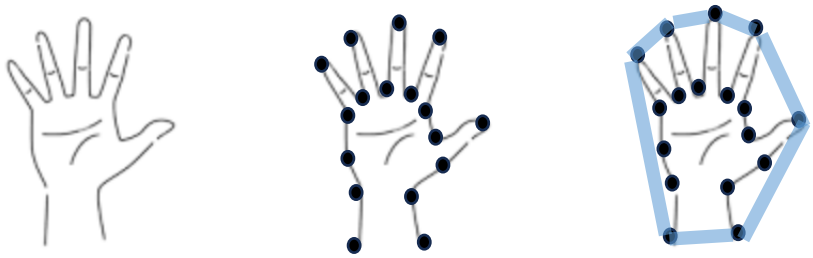
COMP0005 Algorithms
Academic Year 2023-24

Group Coursework: Gesture Recognition via Convex Hull

Released: February 8th, 2024
Submission deadline: March 6th, 2024 @ 4pm GMT

The problem.

Gestures are increasingly being used in a variety of application domains to interact with computing devices – from augmented reality applications to surveillance, from identifying sign language to detecting facial emotions. In these scenarios, images captured by a digital camera need to be processed reliably and in real time to detect and track the object(s) of interest. Consider, for example, the problem of *hand* gesture recognition exemplified below:

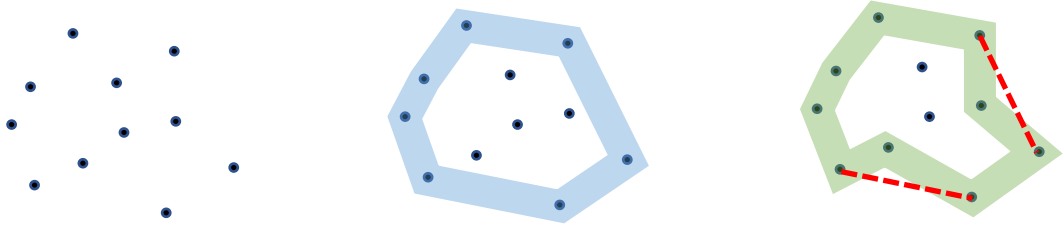


(a) *(b)* *(c)*

Fig. 1 - Hand Gesture Recognition Example

Images of a hand are first captured by a camera (*Fig. 1a*); these images are pre-processed to detect contour points (*Fig. 1b*); finally, the *convex hull* of these points is computed and used for tracking (*Fig. 1c*).

Formally, given a set S of N points (x_i, y_i) on a Cartesian plane (such as the contour points exemplified in *Fig. 1b*), a *convex hull* is defined as the *smallest convex polygon* that contains all points in S. A polygon is *convex* if it contains all the line segments connecting any pair of its points. For example, given the 11 points in *Fig. 2a*:



(a) *(b)* *(c)*

Fig. 2 – Convex Hull Example

the blue-highlighted polygon in *Fig. 2b* is a convex hull, while the green-highlighted polygon in *Fig. 2c* is not, since the red-dashed segments are not contained in it.

Many algorithms exist to compute the convex hull of a set of Cartesian points, with varying computational complexity. Three such algorithms are: the Jarvis march algorithm (also known as the Gift wrapping algorithm) [1], the Graham scan algorithm [2], and the Chan's algorithm [3]. If *n* is the number of points in S and *h* is the number of points forming the convex hull, the (time) computational complexity in the worst-case scenario of the Jarvis march algorithm is O(*nh*), of the Graham scan algorithm is O(*n* log *n*), and of the Chen's algorithm is O(*n* log *h*).

The task.

Your task for this coursework is threefold:

- to study the three convex hull algorithms named above (Wikipedia link provided, as well as reference to the original paper) and implement them in Python;
- to implement an experimental evaluation framework and use it to comprehensively evaluate the above three algorithms, both in their average and in their worst-case scenarios, as you vary both *n* and *h*;
- to thoroughly and critically discuss the circumstances under which some algorithms are better suited than others. Experiment with values of *n* and *h* up to a point where you can observe significant performance differences among the three algorithms.

You are expected to implement the convex hull algorithms and experimental framework from scratch. Do not import any library other than `math`, `timeit`, `random`, and `matplotlib`. If in doubt about what you can and cannot use, ask in Moodle "Ask a question" forum. Your experiments must be conducted on your local machine (not on the cloud) using Jupyter Notebook with Python 3.11.

Submission instructions.

This is a group-based coursework and only one submission per group is required. Using the Moodle course website, each group must submit exactly two files:

- A Jupyter notebook `grp-`[your group number]`-0005code.ipynb` containing the group's implementation of the three convex hull algorithms and experimental framework. This notebook must follow the structure of the skeleton code provided. Do not submit the data (i.e., the Cartesian points) you created to experimentally compare the performance of the algorithms; rather, your notebook should contain code to generate it. You may use the Python `random` package for this purpose; limit your `x`-scale and `y`-scale to be integers in the range `[0, 32767]`. Use the Python `timeit` library to measure the execution time of your implementation (do not include the time to generate test data in your measurements). You may use Python `matplotlib` to create charts that visualise your experimental results.

- A PDF document `grp-`[your group number]`-0005report.pdf` of maximum 5 pages (font style: Arial or Times New Roman, font size: 12pt), where you present the results of your experimental evaluation of the three algorithms under consideration. Start the report with a brief overview of the experimental framework you have designed and report the hardware/software characteristics of the platform you have used for experimentations. Then thoroughly and

critically discuss the performance results of each of the three algorithms under study, in their average and worst-case scenarios; also explain how you programmatically "triggered" these scenarios in your framework. Conclude with a comparative assessment of the selected algorithms and discuss the circumstances under which some algorithms are better suited than others. As part of the report, you are also expected to provide a concise assessment of the contribution made by each team member towards the coursework submission. A template with a recommended report structure is provided.

Submission deadline: Wednesday, March 6th 2024, 4pm GMT.

Assessment.

This coursework represents 40% of the final mark for COMP0005 Algorithms. Submissions will be evaluated in terms of the following marking criteria:

- Quality of your code (i.e., is your implementation – of both the convex hull algorithms and the experimental framework – *correct*? Is your code *readable, well documented, reusable*? Is it time and space *efficient*?) [40 points]
- Empirical analysis (e.g., is your experimental framework comprehensive? Have edge-cases been analysed? Have suitable stress-test conditions been experimented with? Are the results of your experimentation correct? Are they thoroughly discussed?) [60 points]

Marks for each criterion will be assigned following the UCL CS Grade Descriptor (criteria 4, 5 and 6).

Note: if a group is experiencing lack of engagement from some members, please report this to the module lead immediately. Students who fail to make a substantial contribution to the group coursework will receive a zero mark.

Academic Integrity.

UCL has a very clear position about academic integrity – what it is, why it is important, and what happens if you breach it. Make sure you have read UCL position on this matter before attempting this coursework.

References.

[1] Jarvis, R. A. "On the identification of the convex hull of a finite set of points in the plane". Information Processing Letters. 2 (1): 18–21. doi:10.1016/0020-0190(73)90020-3.
Wikipedia link: https://en.wikipedia.org/wiki/Gift_wrapping_algorithm

[2] Graham, R.L. "An Efficient Algorithm for Determining the Convex Hull of a Finite Planar Set". Information Processing Letters. 1 (4): 132–133. doi:10.1016/0020-0190(72)90045-2.
Wikipedia link: https://en.wikipedia.org/wiki/Graham_scan

[3] Chan, T. "Optimal output-sensitive convex hull algorithms in two and three dimensions". *Discrete & Computational Geometry*. 16 (4): 361–368. doi:10.1007/BF0271287.
Wikipedia link: https://en.wikipedia.org/wiki/Chan%27s_algorithm