# COT 4521: Intro. to Computational Geometry (Fall 2020)

## Final Project

### Objectives

For this assignment is the you will propose, implement, experiment or demonstrate, and report results for one or more algorithms. The goal is that you provide some new information to the rest of the class.

### Ground Rules

This assignment can be done in pairs. Pairs are expected to select a topic approximately twice as extensive as individuals. However, the code and reporting must be yours.

### Assignment Instructions

The point of the project is to propose and execute some kind of experiment (i.e., comparison between algorithms) or demonstration of an algorithm.

3 deadlines will appear in Canvas. (1) Project proposal; (2) Mid-Project Report; (3) Final project submission. You will need to submit/upload the respective text, code, images, etc. for each.

- **Project Proposal** – The project proposal will be 2 paragraphs describing your planned project. The first paragraph will describe the topic. Be explicit—for example, if you are implementing 2 algorithms, tell me which 2 and why you picked them. The second paragraph should describe what experimentation or demonstration you plan to perform. Again, be explicit—if you plan performance comparison, describe what types of comparisons you will do. If you plan to demonstrate something, describe exactly how (like what types of input data you will use). Keep in mind, if your project is "easy" we expect a higher-quality / more complete result. If your project is more challenging, we will be more flexible with I only want 1 copy per team.
- **Mid-Project Report** – By the time the mid-project report is due, you should be done or close to done with your code. For this report, you will need to provide a copy of your code with instructions for compiling/running and a 2 paragraph status report. The first paragraph will provide detail about the status of the development and experimentation/demonstration. The second paragraph will describe any changes in the scope of your project.
- **Project Presentation** – Towards the end of the project, you will be allotted ∼5 minutes in class to present your work. Your presentation should be approximately 4-5 slides (pptx or pdf format). In the slides, you will introduce the project, describe any experimentation, and discuss results and conclusions.
- **Final Project Submission** – Your final submission will include your final code and a short (1-2 page) report outlining the problem you were addressing, results of your work, and what you failed to accomplish. *Note: admitting you were unable to deliver on your original proposal will not lead to a lower grade (in fact it might result in a higher grade). The work delivered will be judged somewhat independently of what was proposed.

## Potential Topics

The choice of topic is yours. Below is a suggested but not exclusive topic list. Important note: We can't have topics that overlap too much, and topics are first come, first serve. If this is discovered, students with overlapping topics will be asked to revise their planned projects.

- Implement two (non-trivial) algorithms for a specific topic (triangulation; convex hulls; point searches; etc.).
- Implement one (non-trivial) algorithm in 2 or more programming languages (Python, Java, and C for example). The programming languages should have significant enough distinctions (interpreted vs GC/JIT compiled vs native compiled) that will potentially result in performance differences.
- Implement incremental Voronoi construction algorithm.
- Calculate geodesic distance using a relative neighborhood graph and compare to Euclidean distance.
- Multiple ($> 2$) convex polygon intersections, unions, differences, etc.
- Write a Processing sketch that animates a (non-trivial) algorithm.
- Find the intersection of halfplanes.
- Implement the art gallery problem solution discussed in class. Enable checking if user selected guard locations cover the polygon.
- Compute and Voronoi diagram and Delaunay triangulation of a point set.
- Implement and compare 2 clustering algorithms*.
- Implement and compare 2 spatial subdivision strategies*.
- Implement and compare 2 nearest neighbor strategies*.
- Experiment with generate and visualizing Reeb Graphs using ReCon (https://github.com/harishd10/recon)
- Perform experimental data analysis with Persistent Homology using ripser (https://github.com/Ripser/ripser)
- Other ideas are welcome (see `http://www.cse.usf.edu/~sarkar/cGeom/Computational_Geometry/Project.html` for more examples)