Project 1: Geometric Methods

Project Due: Fri., Oct 1
Homework Check-in: Sept., 22 *(at Noon)*

Overview. Our first project explores the role of geometry in modeling planning and anticipation in the context of intelligent collision avoidance. Here we will use the classical AI technique of tree search, together with geometric simulation techniques, in order to plan intelligent, efficient motion and paths for virtual agents.

This assignment has two parts. In the check-in assignment, due in a about a week (Sep 22), you will need to implement basic collision checks. For the full project (Due, Oct 1), you will need to develop a basic PRM-based path planning library simulate one or more intelligent agents moving smoothly through a field of obstacles.

You must work alone for part 1 (the homework check-in), but you may work with a partner for part 2 (the simulation project) – just one project turn-in needed per pair.

Part 1: Homework Check-in [100 points]
For the homework check-in associated with Project 1, you will need to implement a collision detection library in Processing, some sample code will be provided to help you get started. We have implemented a simple point-in-circle test to get you started. Your code will be tested automatically, so make sure you support the exact inputs/outputs as specified in the current `collide_()` functions.

Submit just the file *CollisionLib.pde* to the canvas turn in.

Part 2a: Processing Planning Library [40 points]
The second part of the project is split into two parts: a planning library and a planni
For the homework check-in associated with Project 1, you will need to implement a path planning library in Processing, some sample code will be provided to help you get started. We have implemented a simple probabilistic roadmap based on a Depth First Search strategy. You can start with this implementation, or write your own from scratch, but make sure you support the exact inputs/outputs as specified in the current `planPath()` function.

Your path planning will be tested in a variety of scenarios. Your grade will be a function of your solution's runtime (smaller is better), the optimality of the path you return (shorter is better), and the number of collisions along the path (it should be 0 in all cases).

To work well on all three metrics, you likely want to replace BFS with Dijkstra's/UCS or A*. However, there are other improvements to consider as well.

Submit just the file *PRM.pde* to the part 2a canvas turn in.

Part 2b: Navigation Simulation [up to 60 points *(up to 70 for grad students)*]
You will need to write a visual simulation of an agent (or multiple agents) moving through a static environment. Likely you will use a PRM with A*/UCS, but feel free to use another method like visibility graphs or RRTs if it works better for your simulations.

Required components are indicated with a star (*).
Recommended components are indicated with a plus (+), but feel free to do other options instead if you'd like to focus on different aspects of the project.

Single Agent Navigation* (up to 30 points).
Simulate a 2D agent moving through a cluttered 2D environment. The physical extent of the agent should be represented by a bounding geometry (e.g., bounding circle or bounding rectangle). As the agent moves through the environment, its bounding geometry should not overlap with any of the obstacles. To get full credit, the agent should move smoothly through the environment, without unnatural hesitation or large changes in speed.

3D Rendering & Camera (up to 10 points).
Render your navigation example in 3D (only the rendering needs to be 3D, the navigation can be 2D). To get full points the agents and the environment should be visualized with a high-quality 3D rendering, and there should be a natural-to-use camera that allows the user to navigate around the scene from different angles and positions. Try to use texturing and lighting to improve the rendering.

Improved Agent & Scene Rendering (up to 10 points).
Render the agent(s) using a model or image that is not a simple geometric shape, either with a 3D model, or a textured 2D quad. For full credit also render the obstacles in 3D or with images. Importantly, even if you draw complex obstacles, you should still use simplified (invisible) geometry for computing collisions.

Orientation Smoothing (up to 10 points).
Give your agents an orientation based on their direction of motion. To get any points here, you must also render the agent in a way which indicates their orientation (i.e., you can use a circular bounding geometry, but you can't use a pure circular rendering). Points will be awarded for how smoothly your agents turn. Just snapping the orientation directly to the current travel direction will only result in partial credit, you should naturally smooth the orientation as the agent moves.

Planning Rotation (up to 10 points).
Plan the navigation for an agent with a rectangular (or other non-circular) bounding geometry. For these points, the agent must be in an environment where it needs to rotate to reach the goal (e.g., a passage too narrow for the longest dimension).

User Scenario Editing+ (up to 10 points).
Allow the user to edit the scene by placing agents and obstacles during set-up or runtime. For full points the scenario creation process should be natural and mouse-based (or mouse + keyboard).

Realtime User Interaction (up to 10 points).
These points come from user interaction directly with the simulation itself (not from controlling the camera). To get full points, the user should have a clear, smooth, natural, and continuous way to interact with the agent/agents as they move through the environment. Discrete interactions such as toggling some behavior on/off will only receive a couple of points. Look for continuous interaction such as allowing the user to move an obstacle with the mouse during the simulation.

Multiple Agents Planning+ (up to 10 points).
Support multiple agents moving simultaneously in the scene. All agents need to move towards their own independent goals, each with their own path planning (though the agents can share an underlying roadmap). *[Agents do not need to avoid each other, unless you are going for the additional crowd simulation points below.]*

Part 3 - Challenge: Crowd Simulation [20 point (grad*), 10 points (undergrad)]
The challenge simulation is <u>required</u> if you are a graduate student, and optional extra credit for undergraduates. If you complete the challenge, you have a 72-hour extension on the project.

Description (up to 20 points).
Have multiple agents moving simultaneously to their goals in a shared environment, while avoiding collisions with each other. Feel free to use whatever method you like for collision avoidance (boids, social forces, TTC force, etc). <u>For full points the agents should display smooth, anticipatory, collision-free motion in *at least two interesting scenarios*</u> -- so try to choose a technique where you can get good results.

Art Contest
If you generate a pretty image (even by accident), save it to submit to the class art contest. A pool of honorable mentions will be given 2 points, and the grand winner gets 5 points. All winners will be chosen *completely subjectively*.

Part 2 Project Report & Video* (10 points).
Your submission must be in the form of webpage with:

- Images of your agent(s) navigating their environments
- A brief description of the features of your implementation and timestamp of where they occur in your video(s).
- An explicit list of which features you attempted
- Code you wrote
- List of the tools/library you used
- Brief write-up explaining difficulties you encountered
- One or more videos showcasing features of your simulation
- Submission for the art contest (optional)

These 10 points for the submission itself will be based on the clarity of expression of the report, and to the degree which it quickly communicates what you tried, what worked well, and what didn't.

<u>Additionally, each feature you expect to get credit for must be documented in your submission videos in a way which clearly shows the resulting behavior</u>. If you do not show a feature in your submission video(s) you will not receive credit for it.

Part 2 Grading Criteria
Simulations must animate well and look convincing to get full credit. Partially implemented features will receive partial credit. Points past those needed for full credit will count as extra credit, though at a discounted rate (see Scoring below). If you do other things you think are cool and worth credit let us know beforehand and be sure to document it in the report.

Project Scoring
Undergrads may submit up to 120 points of work:
40 for part 2a
60 for part 2b
10 for the challenge
10 for the report
… if you submit more than the limit, we will grade a random subset.

Graduate students may submit up to 140 points of work:
40 for part 2a
70 for part 2b
20 for the challenge
10 for the report
… if you submit more than the limit, we will grade a random subset.

Partial credit will be given. Scores computed as follows (points above 100 possible):
-*Undergraduate*: Grade is $\sqrt{(\text{totalPoints} * 100)}$ [e.g., 100 points will be full credit]
-*Grad students*: Grade is $\sqrt{(\text{totalPoints} * 84)}$ [e.g., 120 points will be full credit]

*Extra credit will be given only projects with an A- or higher on required features.

Use of other code and tools
Anything you are getting credit for must be code you wrote for this course. You must write the code for the simulation yourself! I know there are many great path planning or crowd simulation libraries out there -- I've written one or two myself :) --- and learning how to work with them is very useful but it will not count towards this assignment. Likewise, finding fully working obstacle navigation code from the internet may be useful for future personal projects, but <u>to receive a grade for this assignment you must turn in your own simulation code you wrote yourself</u>. External libraries may be used for aspects that are not related to simulation (e.g., rendering, camera motion, video capture) just be sure to document that you used these.

Partners & Groups
You are strongly encouraged to work in pairs for the project. Each pair should turn in only one assignment. Both people will be given the same grade. <u>You cannot repeat the same partner from a previous project.</u>

If you need help creating a webpage, many online resources exist. UMN's Google Site: https://sites.google.com/a/umn.edu is a great place to start, especially if you have never made a webpage before.