

cisc3665, fall 2011 / prof sklar  
Lab III.2: Chasing and Evading lab  
REVISED VERSION (with clarifications underlined below)

## Instructions

This is the second lab for unit III, Game Physics, Motion and Perception, and should be submitted for a grade.

The purpose of this lab is to give you a chance to demonstrate what you have learned in this unit about game physics and math, “chasing and evading” strategies and “wayfinding” algorithms.

This assignment is worth **10 points**, or 10% of your term grade.

The assignment is due by on NOVEMBER 7, by 6AM.

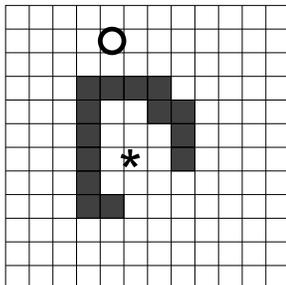
## Preparation

- Read the chapters entitled **Chasing and Evading** (ch 2) and **Basic Pathfinding and Waypoints** (from *AI for Game Developers*, by David M. Bourg and Glenn Seemann, which were handed out in class).
- Download the sample Processing code, **chaser0**, from:  
<http://www.sci.brooklyn.cuny.edu/~sklar/cisc3665/examples/chaser0.zip>

## 1 Written component

You need to submit one file for this part, preferably a PDF document containing your answers to both questions.

- 1.1 Given two agents moving at a constant speeds in a 2-D environment. The first agent,  $A_1$ , is at position (5, 2), moving at a velocity of (1, 1). The second agent,  $A_2$ , is at position (1, 3), moving at a velocity of (2, 1). Compute the **distance** between the two agents and the **angle** between the two agents' directions of motion. Note that the direction of an agent's motion is that of its velocity vector. (1 point)
- 1.2 Using the same information about agents  $A_1$  and  $A_2$ , answer the following question. Assume that the agents move one unit of velocity per second—if  $A_1$  is at (5, 2) at time  $t = 1sec$  and moving at velocity (1, 1), then at time  $t = 2sec$ ,  $A_1$  will be at position (6, 3), etc. Suppose that  $A_2$  is chasing  $A_1$ . Assume that neither agent changes course or speed. Will the paths of  $A_1$  and  $A_2$  ever cross? If yes, where (i.e., at what point)? Will they reach the point at the same time? (1 point)
- 1.3 Given the environment below, containing an agent (circle) and a piece of gold (\*) hidden in a cave:



Explain how you would implement a *pathfinding* algorithm to help the agent locate the piece of gold. Describe any functionalities that you would need the agent to have (e.g., perception capabilities, etc). (1 point)

## 2 Programming Component: Chasing and Evading

You need to submit one program for this part: **chaser1**.

- 2.1 The sample code, **chaser0**, demonstrates a simple chasing/evading game where there are two agents: an avatar, which is controlled by a human, and an opponent, which is controlled by the computer. Try running the sample code. You can control the avatar (pink circle) by pressing on the arrow keys. Each key you press acts like giving the avatar a kick in the direction of the arrow. So if you start by pressing the up-arrow key, then the avatar moves up. If you press the up-arrow key again, the avatar moves up more quickly. You can press **S** to stop the avatar. You can press **R** to reset the game, and **Q** to quit the game.
- 2.2 The chaser code for the opponent is contained in the **chase()** function in the **Agent** class. This implements "line-of-sight" chasing, as described in chapter 2 (of the reading listed above).
- 2.3 Make a copy of the sample code and call it **chaser1**.
- 2.4 Modify the code so that the human player becomes the chaser and the computer player is the evader. To do this, you need to write an evasion function, **Agent.evade()**. This should be called from the **draw()** function, instead of calling **opponent.chase()**. (2 points)

## 3 Programming Component: Pathfinding

You need to submit one program for this part: **chaser2**.

- 3.1 The second part of the assignment is to modify the game so that the game environment contains obstacles, and the computer player has to implement a *pathfinding* algorithm in order to play the game.
- 3.2 Create a new version of the chaser game and call it **chaser2**.
- 3.3 Modify the code so that it contains some obstacles. Generate some black squares in the arena that are the same dimension as the agent (i.e., width and height are equal to **Agent.d**). Place these squares in random locations around the arena. (1 point)
- 3.4 Modify the code so that it will detect if an agent is going to collide with an obstacle. *Hint: look at collision detection in the sample programs from the class on Oct 5.* (1 point)
- 3.5 Modify the code so that it will not allow an agent to move into a position where it collides with an obstacle. The agent should just stop if it is going to bump into an obstacle. (1 point)
- 3.6 Modify the code so that the computer agent uses one of the methods outlined in chapter 6 (of the reading listed above) in order to avoid (go around) obstacles. (2 points)

## Submission

- Zip together the files from all three parts of the assignment.
- Submit your assignment electronically, using the submission page linked on the class web page.