CS 330 Artificial Intelligence and Game Development
Fall 2023

Jay Sebastian, jms0147@uah.edu, adapted from the original assignment by Mikel D. Petty, Ph.D.

Programming Assignment 1: Dynamic Movement

Objective

Your primary objective is to implement and test the dynamic movement update and three dynamic movement behaviors, Seek, Flee, and Arrive. Your secondary (optional) objective is to learn how to plot your movement trajectories.

Requirements

Implement the dynamic version of the Newton-Euler-1 movement update algorithm and the dynamic Seek, Flee, and Arrive movement behaviors. These are all described in Chapter 3 of the Millington textbook and Lecture 6. Also implement a Continue movement behavior, which does not alter a character's movement at all, but simply uses the initial values. The Continue behavior is not in the textbook, but it is in the R implementation. For this assignment, the Continue character's initial velocity and rotation will all be 0, so the Continue character should not move at all.

Your program should output each character's trajectory as a text file (.txt). The output file should have one record per character per timestep, including a record for the initial conditions (time = 0). For example, for a scenario with 4 characters that runs for 50 timesteps, there should 204 records in the output file, in the following order. (This list shows proper order, but not proper format for the output file.)

```
character 1, timestep 0
character 2, timestep 0
character 3, timestep 0
character 4, timestep 0
character 1, timestep 1
character 2, timestep 1
character 3, timestep 1
character 4, timestep 1
...
character 1, timestep 50
character 2, timestep 50
character 3, timestep 50
character 4, timestep 50
```

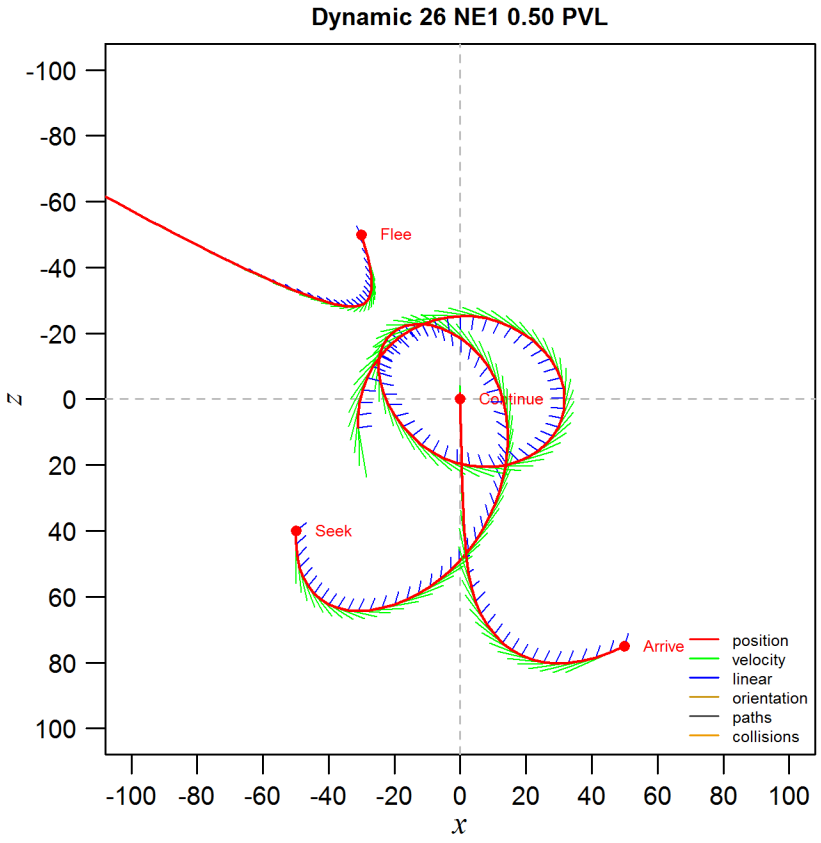
Each record should have the following 11 fields, in the order listed, and separated by commas:

1. simulation time
2. character id (numeric)
3. position *x* (meters)
4. position *z* (meters)
5. velocity *x* (meters per second)
6. velocity *z* (meters per second)
7. linear acceleration *x* (meters per second per second)
8. linear acceleration *z* (meters per second per second)
9. orientation (radians)

10. steering behavior code (1=Continue, 6=Seek, 7=Flee, 8=Arrive)
11. collision status (TRUE if collided, FALSE if not collided; always FALSE for Program 1)

Run your program for 50 simulated seconds with a timestep duration of 0.5 using Newton-Euler-1 integration, i.e., 100 timesteps after initialization. Your scenario should have four characters with the initial conditions in the following table. The movement target of the Flee, Seek, and Arrive characters is the Continue character.

Character number	1	2	3	4
Character id	2601	2602	2603	2604
Steering behavior	Continue	Flee	Seek	Arrive
Initial position	0, 0	-30, -50	-50, 40	50, 75
Initial velocity	0, 0	2, 7	0, 8	-9, 4
Initial orientation	0	$\pi / 4$	$3\pi / 2$	π
Max velocity	0	8	8	10
Max acceleration	0	1.5	2	2
Target	0	1	1	1
Arrival radius	0	0	0	4
Slowing radius	0	0	0	32
Time to target	0	0	0	1



Replicate, as closely as possible, the character movement trajectories shown in the preceding image. You are not required to implement a trajectory plotting program yourself; I will provide three different plotting programs (see Resources below). In fact, you are not required to plot your trajectories at all (but see recommendation 2 in Assignment-specific Recommendations below).

Deliverables

1. Source code for the dynamic movement program
2. Trajectory data file, with trajectory data for all four characters
3. (Optional) Trajectory plot image, with all four characters

Grading

The assignment is worth a maximum of ten (10) points. It is 10% of your final semester grade, so each point is 1% of your semester grade. The grading rubric is as follows:

Points	Criterion
0	Assignment not submitted or deliverables missing
1	Assignment deliverables all submitted
0-1	Software engineering quality of the program reasonably good
0-2	Movement update present and correct
0-3	Movement behaviors (Seek, Flee, Arrive, Continue) present and correct
0-1	Trajectory data file in proper format and readable by trajectory plotting program
0-2	Trajectories reasonably similar to example trajectories
0-10	Total

The criteria are independent of each other (except, of course, that you cannot get any of the other points if you don't turn the assignment deliverables in). Note that if your trajectory data file is not readable by the trajectory plotting program, the highest score possible may be 7.

Resources

All of the following have been posted to the course Canvas page in the Files > Programming Assignment 1 – Dynamic Movement:

1. Example R code for Dynamic Movement; this consists of 5 code modules, numbered 0-4. Module 3 contains the main timestep loop, the movement update function, and movement behaviors. Module 4 is the trajectory plotting program.
2. Two additional trajectory plotting programs, one written in Python and one written in Matlab. You are welcome to use these trajectory plotting programs if you do not want to learn how to use the R trajectory plotting program. Disclaimer: These programs were written by students, not me. I can try to help to a certain extent, but you need to put in the work. You may have to modify them to make them work properly, especially if your trajectory file is formatted slightly differently than what the plotting program expects.
3. A partial example of the trajectory file for this assignment's scenario generated by the R implementation of the assignment movement behaviors, with times 0-4 for all four characters.
4. An example of a complete trajectory file for a different scenario, which you can use while learning to plot trajectories. (CS330, Dynamic trajectory data.txt) Your solution should NOT have these values.

5. An image of the output of the above trajectory file. (outputPlot.png) Your solution should NOT look like this scenario.
6. An image of the complete trajectories for this assignment, i.e., the same image as above. (Your solution SHOULD look like this one)

Due date and time

See Assignments page in the course Canvas website.

Assignment-specific recommendations
(See the general instructions for general recommendations.)

1. Implement and test the movement behaviors one at a time. Start with Continue. Once you get Seek working, Flee is a trivial change. Then the code for Seek can be enhanced for Arrive.
2. Test your program by plotting your trajectory data file with one of the trajectory plotting programs that have been posted to Canvas (see Resources above). I may input your deliverable trajectory data file into my plotting program when grading your assignment. If your trajectories do not resemble mine, or if your data file is not properly formatted and thus unreadable by my program, you may not know it unless you attempt to generate the plots.
3. If you decide to team, one team member should immediately start by figuring out how to plot the output trajectory data file, using my example trajectory data file. If you are working alone, you should do that first. Seeing a plot of your trajectories will make testing and debugging the movement behaviors much easier.

End of Programming Assignment 1