EXDROP

LAB 6

SECTION B

SUBMITTED BY:

CODY BROOKS

SUBMISSION DATE:

2015-10-23

Problem

The problem of this lab was to understand how to measure and the applications of free fall motion. A program must be written that collects accelerometer data from an Esplora. The accelerometer data must be used to compute the distance the Esplora has fallen. The objective is to use sensor data to compute a practical result.

Analysis

The problem states that the distance fallen should be calculated with two different methods. The first method uses the equation $x_i = x_0 + v_0 t + \frac{1}{2} g (t_i - t_0)^2$. Because the initial velocity and the starting position of the board are both zero, the equation simplifies to $x_i = \frac{1}{2} g (t_i - t_0)^2$. The Esplora's magnitude of acceleration when it is sitting still is very close to 1. When the Esplora falls, the magnitude of its acceleration will be close to zero. The second method sums incremental measurements of distance calculated using the equations $v_i = v_{i-1} + g \cdot (1 - \textit{magnitude of acceleration}) \cdot (t_i - t_{i-1})$ and $x_i = x_{i-1} + v_i (t_i - t_{i-1})$. When the Esplora stops falling, its magnitude of acceleration will return to zero.

Design

The problem can be solved using two nested while loops using both methods of calculation. The outer while loop checks to see when the Esplora starts to fall by constantly calculating its magnitude of acceleration using the **`mag()`** function created for previous labs. Once the magnitude is close to zero (using the **`closeTo()`** function), which indicates that the board has started falling, the program will enter the second loop. The second loop also monitors the magnitude of acceleration of the board, but checks to see when the magnitude is close to 1, which indicates that the board has stopped falling. Once the magnitude of acceleration returns to close to 1, the program will exit both loops and finish any necessary calculations.

Using the first calculation method, when the program first enters the second loop, the time will be recorded as t_0. Likewise, when the board stops falling (after the second loop exits), the time is recorded as t_i. The distance fallen is then calculated using $x_i = \frac{1}{2}g(t_i - t_0)^2$ with $g = 9.8m/s^2$.

Using the second method, once the board starts falling, the distance fallen is computed each iteration of the inner while loop and added to the cumulative sum of the distance fallen. Each iteration, the current time is scanned in and compared to the time of the last iteration. The change in time and the current magnitude of acceleration are then plugged into the equation $v_i = v_{i-1} + g \cdot (1 - \textit{magnitude of acceleration}) \cdot (t_i - t_{i-1})$ to find the board's current velocity. The board's current velocity is then compared to the velocity during the last iteration and plugged into the equation $x_i = x_{i-1} + v_i(t_i - t_{i-1})$. At the end of the second loop (when the board is done falling), x_i is the total distance fallen.

Testing

The first test would be to hold the Esplora still and run the program. The program should not register any falling motion or compute a distance. After it is confirmed that it handels stillness correctly, drop the board about 0.5-1 meter into someone's hands. The program should register the fall and print a distance that is very close to the distance it was dropped. Do this several times within the lab with varying heights no more than 1.5 meters, verifying each drop.

To test with longer drops, use a padded wireless device (like a Wiimote inside a foam football) with an accelerometer and an available wrapper program to test a few longer drops and verify larger distances. Since the program shows the difference in calculated distance between the two methods, one can see how increasing the distance fallen also increases the error.

Comments

One thing that I learned from this lab was that you should always know what kind of result to expect when testing. During our first short drop tests, we got results, but they were not correct because we did not know how far the board actually dropped. Once we found out that we were not getting accurate results, we were able to debug the problem. It was a good reminder of how methodical debugging should be to yield the desired results.

Questions and Experiments

Part 1

1. The results between drops were accurate within about 10% of each other. One thing that could cause variation is how the board is being dropped and caught. If there is a lot of movement/cushion when someone catches the board, the program may think that the board is still falling. Depending on the orientation of the board (i.e. flat downward, on its side, etc.) Air resistance might play a small factor in how fast the board falls.
2. According to my code, the distance from the third floor to the bottom floor is 8.763 meters.
3. [GRAPH?] We used a tolerance of 0.25 because that is the tolerance we found worked best and yielded the most accurate results when performing Lab 5. It prevented any "jittery" readings and still was a low enough tolerance to always register a fall or change in magnitude of acceleration.

Part 2

1. In the lab, there was only a few centimeters of error, whereas with the three story drop, there was more than a half a meter of error.
2. One of the issues I had while implementing part 2 was the misunderstanding of the given equations. I thought that the equations were to calculate one rectangle of the Riemann sum, whereas the equation already accounted for summing all of the rectangles.