

DS4Drop

LAB 6
SECTION A

SUBMITTED BY:
SYDNEY EHLINGER

LAB DATE:
10/07/2016 & 10/14/2016

Problem

The purpose of this lab was to create a program that is able to calculate the distance the DualShock 4 fell and the amount of time it took to fall. For part two to the lab we had to take in account of the air resistance. The program must print out when the program has started up, when it's waiting for the DualShock to be dropped, and when it is currently falling. Once it has hit the ground it must calculate the distance it fell and the amount of time that it took and print it out in the console. The objective of this lab is to develop problem solving skills and to develop skills in the use of numerous loops to solve a problem.

Analysis

The problem states that we must create a program that has the ability to calculate the distance and the time that the DualShock 4 fell. Therefore, I know that I have to have two while loops each scanning in the time and acceleration of the DualShock 4. The first loop will be scanning the information and needs to be able to detect when the DualShock is falling. The second loop must be able to scan in the information from the DualShock as well and needs to be able to detect when the DualShock hits the ground. And then finally the program will have to be able to compute the distance and the time that the DualShock fell.

Design

When going about designing my program for part 1 I did it in steps. I knew I would have to have an if statement to check if the program had started up and was waiting to be dropped so I started with that since it was simple. I added print statements to this to output that it was in fact receiving data and that it was waiting. After that was completed I moved onto the first while loop. I started it with a scan statement, and I check if the magnitude is not close to 0 because then we know that it's falling. I then break out of the while loop, and move onto my second one. In this while loop it also starts with a scan statement, and then checks if the magnitude is greater than 1.1, because if it is we know that it had hit the ground. Once it hits the ground its calls my meter function which calculates the distance that the DualShock fell.

Testing

To test my program I dropped my controller 5 different times from a height of around 1 meter. My results were pretty constant, with minimal variation. However, this

variation could have been from human error as I was estimating a meter every time I dropped it. To also test my code I ran the csv file for the second floor and I got 6.202 meters in 1.25 second without air resistance and 4.933 meters with air resistance.

Comments

Not too many issues arose while I was implementing part 2 in my original program. The only thing I would say that was a problem I had was I wasn't getting the time correct as I had to subtract it from the original start time, plus every iteration I had to save the current time and then use it as the initial time for the iteration.

Source Code

Lab6.c

```
/* SE 185 Lab 6 DS4Drop */

#include <stdio.h>
#include <math.h>

#define TRUE 1
#define SWITCH 1

double mag(double x, double y, double z);
int close_to(double tolerance, double point, double value);
double meters(int time);

int main(void) {
    int t, true, startTime;
    double ax, ay, az;
    /* time start and time end to calculate the distance fallen*/

    if(TRUE){
        printf("Ok, I'm now receiving data\n");
        printf("I'm Waiting");
        while(SWITCH){
            scanf("%d, %lf, %lf, %lf, %lf, %lf, %lf, %d, %d, %d, %d", &t, &ax,
&ay, &az);
            /*printf("Echoing output: %d, %lf, %lf, %lf\n", t, ax, ay, az); */
            /*printf("Mag: %lf\n", mag(ax, ay, az));*/
            if(close_to(20, 0, (t % 1000))) {
                printf(" . ");
            }
            if(!close_to(0.3, 0, mag(ax, ay, az))) {
                printf("\n");
                true = 1;
                startTime = t;
                printf("Help me! I'm falling!");
                break;
            }
            fflush(stdout);
        }
        while(true == 1){
            scanf("%d, %lf, %lf, %lf, %lf", &t, &ax, &ay, &az);
            /*printf("%d\n", t);*/
            if(close_to(20, 0, (t % 1000))) {
                printf(" ! ");
            }
            if(mag(ax, ay, az) > 1.1){
                printf("\n");
            }
        }
    }
}
```

```

        printf("Ouch! I fell %.3lf meters in %.3lf seconds", meters(t -
startTime), (t - startTime) / 1000.0);
        break;
    }
    fflush(stdout);
}
fflush(stdout);
}

return 0;

}

double mag (double x, double y, double z) {
    return sqrt(pow(x,2)+pow(y,2)+pow(z,2));
}

int close_to (double tolerance, double point, double value) {
    if((value >= (point - tolerance)) && (value <= (point + tolerance))){
        return 1;
    }else{
        return 0;
    }
}

double meters(int time){
    return 0.5 * 9.8 * pow(time /1000.0,2);
}

```

Lab6-2.c

```

/* SE 185 Lab 6 DS4Drop */

#include <stdio.h>
#include <math.h>

#define TRUE 1
#define SWITCH 1

double mag(double x, double y, double z);
int close_to(double tolerance, double point, double value);
double meters(int time);

int main(void) {
    int t, true, startTime, t0, time;
    double ax, ay, az, vi, x;

    if(TRUE){

```

```

printf("Ok, I'm now receiving data\n");
printf("I'm Waiting.");
while(SWITCH){
    scanf("%d, %lf, %lf, %lf, %lf, %lf, %lf, %d, %d, %d, %d" &t, &ax,
&ay, &az);
    if(!close_to(0.3, 0, mag(ax, ay, az))){
        printf("\n");
        true = 1;
        startTime = t;
        printf("Help me! I'm falling!");
        break;
    }
    fflush(stdout);
}
while(true == 1){
    scanf("%d, %lf, %lf, %lf, %lf", &t, &ax, &ay, &az);
    time = t - startTime;
    vi = vi + (9.8 - (mag(ax, ay, az) * 9.8)) * ((time - t0) / 1000.0);
    x = x + vi * ((time - t0) / 1000.0);
    if(mag(ax, ay, az) > 1.1){
        printf("\n");
        printf("Ouch! I fell %.3lf meters in %.3lf seconds\n",
meters(time), (time) / 1000.0);
        printf("Compensating for air resistance, the fall was %.3lf
meters.\n", meters(time) - x);
        printf("This is %.1lf%% less than computed before." (meters(time)
- (meters(time) - x)) / meters(time) * 10);
        break;
    }
    t0 = time;
    fflush(stdout);
}
fflush(stdout);
}

return 0;

}

double mag (double x, double y, double z) {
    return sqrt(pow(x,2)+pow(y,2)+pow(z,2));
}

int close_to (double tolerance, double point, double value) {
    if((value >= (point - tolerance))&& (value <= (point + tolerance))){
        return 1;
    }else{
        return 0;
    }
}

```

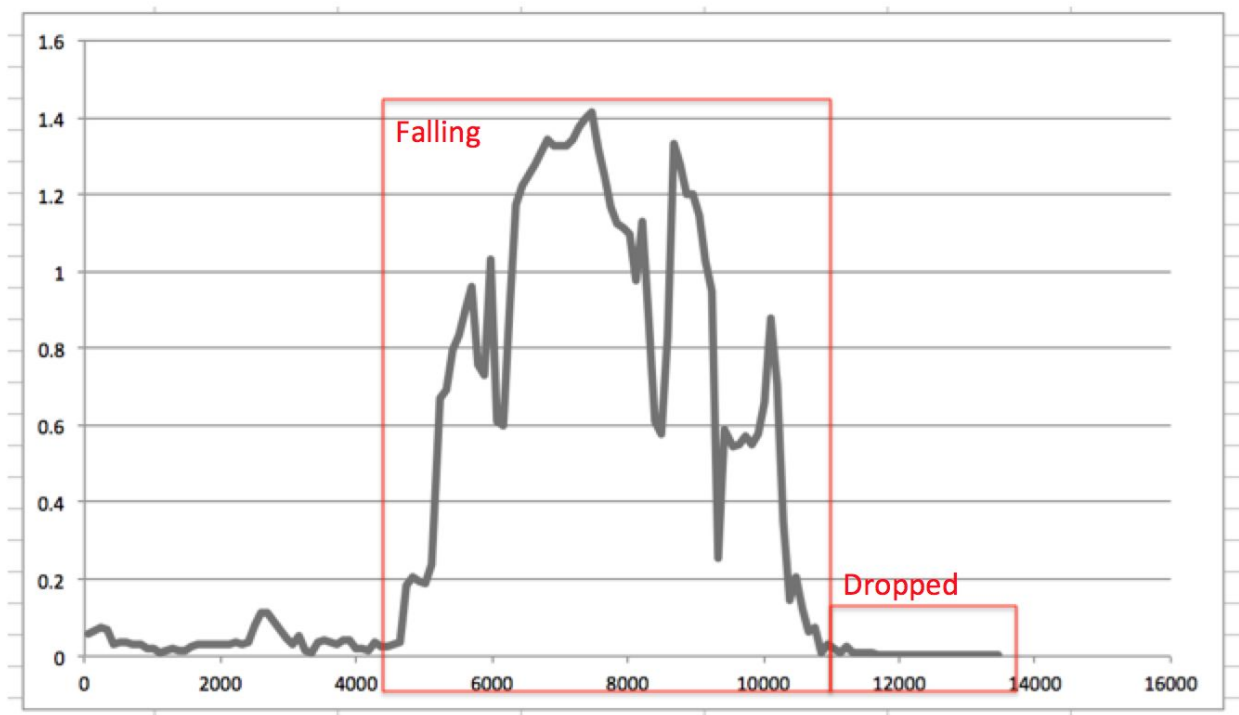
```

    }
}

double meters(int time){
    return 0.5 * 9.8 * pow(time / 1000.0, 2);
}

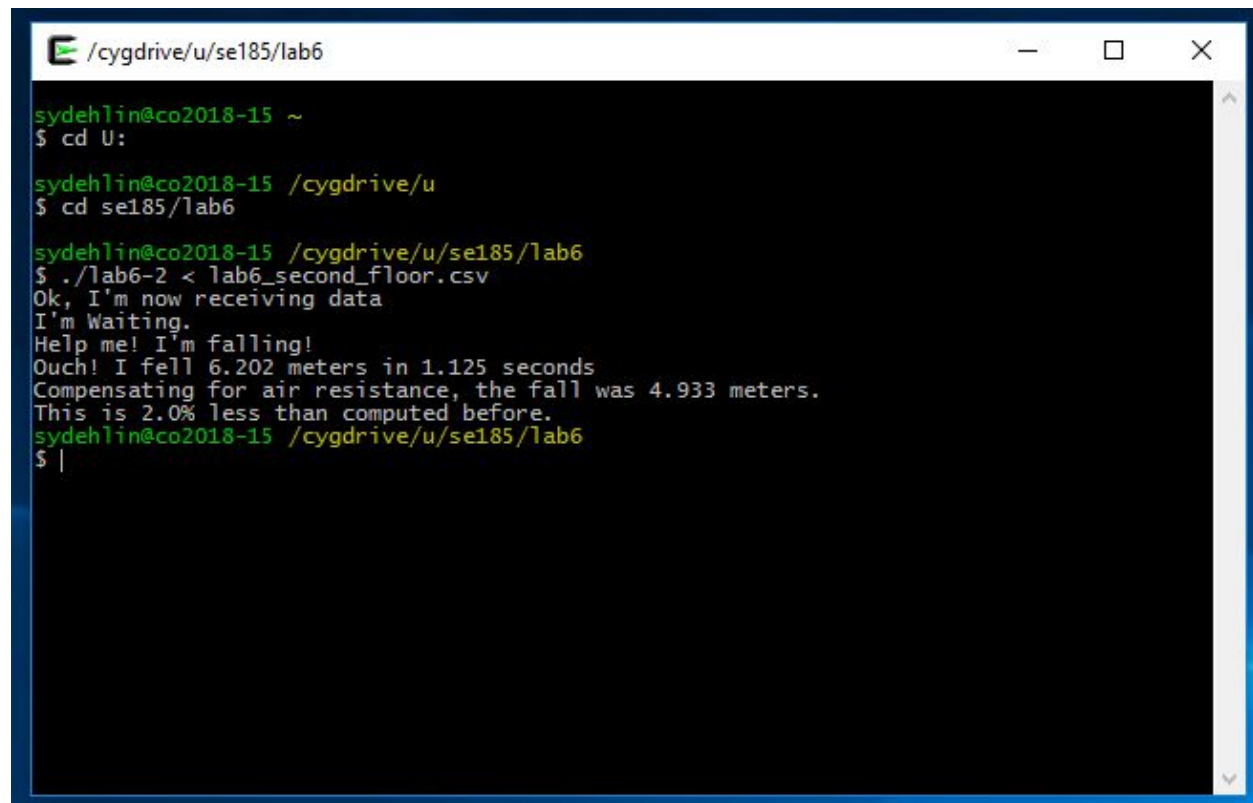
```

Graph



The tolerances that I chose was not close to the value of 0 with a tolerance level of 0.3 for the magnitude, because when I outputted my data to an excel file I noticed when it was falling the magnitude was greater than 0.3 so I set it to that. Also when the DualShock hit the ground the magnitude jumped to greater than 1.1, so I checked that the magnitude was greater than that.

Output



```
/cygdrive/u/se185/lab6
sydehlin@co2018-15 ~
$ cd U:
sydehlin@co2018-15 /cygdrive/u
$ cd se185/lab6
sydehlin@co2018-15 /cygdrive/u/se185/lab6
$ ./lab6-2 < lab6_second_floor.csv
Ok, I'm now receiving data
I'm Waiting.
Help me! I'm falling!
Ouch! I fell 6.202 meters in 1.125 seconds
Compensating for air resistance, the fall was 4.933 meters.
This is 2.0% less than computed before.
sydehlin@co2018-15 /cygdrive/u/se185/lab6
$ |
```