Bicycle Homework:

Algorithm

The algorithm used for this project were:

Where:

– Velocity

– Power

– Mass

– Density

– Frontal Area

– Change in Time

Where:

– Drag Coefficient

- Height

The first one involved the squared term of the drag force and the second involves the linear and the squared terms of the drag, both of which have been made to where they affect the velocity of the system.

Code

#include <iostream>

#include <iomanip>

#include <cmath>

#include <fstream>

using namespace std;

// Problem 2.2

double BikeVelocity()

{

// Opening the file where the data points will be inputted for Problem 2.2.

ofstream Problem22;

Problem22.open("2.2.txt");

cout << "Problem 2.2" << endl;

Problem22 << "Problem 2.2" << endl << endl;

// These are all the constants that I will be using for the program.

// The dt will be in 1e^(-2) seconds and the entire interval will be 100 seconds.

// Area 2 is .7 of Area 1 because the problem asked what a decrease of the frontal area by 30 percent would do.

long double Power = 445, Power2 = 500, Time = 10000, Mass = 80, dt = .01, TimeCurrent = 0, T = 1, Area = .33, Area2 = (.33 \* .7), C = 1, rho = 1.225,

V\_DragForce = 0, V\_DragForceChangePower = 0, V\_DragForceChangeArea = 0,

V\_DrivingForce = 0, V\_DrivingForceChangePower = 0, V\_DrivingForceChangeArea = 0,

Velocity = 10, VelocityChangePower = 10, VelocityChangeArea = 10;

// This is the header for the display and file.

cout << "Time (sec) Bike Velocity Changing Power Changing Area" << endl;

Problem22 << "Time (sec) Bike Velocity Changing Power Changing Area" << endl;

// Showing the valuse at time zero.

cout << TimeCurrent << " " << Velocity << " " << VelocityChangePower << " " << VelocityChangeArea << endl;

Problem22 << TimeCurrent << " " << Velocity << " " << VelocityChangePower << " " << VelocityChangeArea << endl;

// These are the three cases.

// Loop to do Euler Method

while (T <= Time)

{

// This is case 1 where the Area and the Power are just some numbers that will be as a control.

V\_DragForce = (C \* rho \* Area \* pow(Velocity, 2)) / (2 \* Mass) \* dt;

V\_DrivingForce = (Power / (Mass \* Velocity)) \* (dt);

Velocity = Velocity + V\_DrivingForce - V\_DragForce;

// This is the case where the Power was increased by 55 watts.

V\_DragForceChangePower = (C \* rho \* Area \* pow(VelocityChangePower, 2)) / (2 \* Mass)\*dt;

V\_DrivingForceChangePower = (Power2 / (Mass \* VelocityChangePower)) \* dt;

VelocityChangePower = VelocityChangePower + V\_DrivingForceChangePower - V\_DragForceChangePower;

// This is the case where Area was decreased by 30 percent.

V\_DragForceChangeArea = (C \* rho \* Area2 \* pow(VelocityChangeArea, 2)) / (2 \* Mass) \* dt;

V\_DrivingForceChangeArea = VelocityChangeArea + (Power / (Mass \* VelocityChangeArea)) \* (dt);

VelocityChangeArea = V\_DrivingForceChangeArea - V\_DragForceChangeArea;

// Display the results at each time interval.

Problem22 << T/100 << " " << Velocity << " " << VelocityChangePower << " " << VelocityChangeArea << endl;

// Increase counter by one second so that the loop goes again.

// Displayin the final results.

if (T == 10000)

{

cout << "\nThe final results for Time " << T / 100 << "s are: \nBike Velocity: " << Velocity << "\nChanging Power: " << VelocityChangePower << "\nChanging Area: " << VelocityChangeArea << endl << endl;

}

T++;

}

// Now close the program.

Problem22.close();

return 0;

};

// Problem 2.3

double BikeVelocityWithDrag()

{

// Opening the files for the programs.

ofstream Problem23;

Problem23.open("2.3.txt");

Problem23 << "Problem 2.3" << endl << endl;

ofstream Problem232;

Problem232.open("2.3-2.txt");

Problem232 << "Problem 2.3" << endl << endl;

// Header for the file.

cout << "Problem 2.3" << endl;

// Declaring the Constants. dt for Water had to be smaller in order to give good results.

long double Power = 445, Time = 10000, Mass = 80, dt = .01, dt2 = .0001, TimeCurrent = 0, T = 1,

DragCoefficientAir = 2e-5, DragCoefficientWater = 1e-3,

V\_DragForceAir = 0, V\_DragForceWater = 0,

V\_DrivingForceAir = 0, V\_DrivingForceWater = 0,

V\_LinearDragForceAir = 0, V\_LinearDragForceWater = 0,

VelocityAir = 10, VelocityWater = 10,

Height = 1, Area = .33, C = 1, rhoAir = 1.225, rhoWater = 9999.97;

// Creating the labels.

Problem23 << "Time (s) Velocity With Air Drag" << endl;

Problem232 << "Time (s) Velocity with Water Drag" << endl;

cout << "Time (s) Velocity with Air Drag Velocity With Water Drag" << endl;

// Displaying the initial values.

Problem23 << TimeCurrent << " " << VelocityAir << endl;

Problem232 << TimeCurrent << " " << VelocityWater << endl;

cout << TimeCurrent << " " << VelocityAir << " " << VelocityWater << endl;

while (T <= Time)

{

// This includes the linear drag term for air

V\_LinearDragForceAir = ((DragCoefficientAir \* Area) \* (VelocityAir / Height)) / (Mass) \* dt;

V\_DragForceAir = (C \* rhoAir \* Area \* pow(VelocityAir, 2)) / (2 \* Mass) \* dt;

V\_DrivingForceAir = (Power / (Mass \* VelocityAir)) \* (dt);

VelocityAir = VelocityAir + V\_DrivingForceAir - V\_LinearDragForceAir - V\_DragForceAir;

// This is what would happen if the bike is in water

V\_LinearDragForceWater = ((DragCoefficientWater \* Area) \* (VelocityWater / Height)) / (Mass)\* dt2;

V\_DragForceWater = (C \* rhoWater \* Area \* pow(VelocityWater, 2)) / (2 \* Mass) \* dt2;

V\_DrivingForceWater = ((Power) / (Mass \* VelocityWater)) \* (dt2);

VelocityWater = VelocityWater + V\_DrivingForceWater - V\_LinearDragForceWater - V\_DragForceWater;

// Displaying the results in terms of

Problem23 << T/100 << " " << VelocityAir << endl;

Problem232 << T/1000 << " " << VelocityWater << endl;

if (T == 10000)

{

// Displaying the final results.

cout << "\nThe final result for Time " << T / 100 << "s is: Velocity in Air - " << VelocityAir << endl;

cout << "\nThe final result for Time " << T / 1000 << "s is: Velocity in Water - " << VelocityWater << endl;

}

T++;

}

// Closing the files.

Problem23.close();

Problem232.close();

return 0;

};

int main()

{

// Making something clear.

cout << "All velocity terms are in meters per second." <<endl << endl;

// Calling the subroutines

BikeVelocity();

// Adding some space.

cout << endl;

BikeVelocityWithDrag();

return 0;

}

How to Run the Code

This code is written in C++ so in order to run it, the g++ compiler should be used. This compiler is already in Omega so all needed to be done is to run it.

Results and Analysis

For the first graph, this was just some semi-random variables that I plugged in to have a control for the other two cases. It can be seen that after a while, the bike reaches a terminal velocity.

From this graph, a change of 55 watts of power led to an increase in the terminal velocity, but the overall shape of the graph stayed the same.

When the area is changed by 30 percent but the power was the same from the first graph, the change in terminal velocity was much greater than that of both the other graphs.

If in the case of the first graph, the linear drag term is not ignored, then the effect on the overall velocity is negligible since the terminal velocities are basically identical.

Since water density is much greater than air, the velocity achievable in water is going to be much less that that achievable in just water.

Conclusion

Any changes in any of the constants values will result in a change in the terminal velocity. If any of the drag terms are big, then the terminal velocity is going to be less than that of whatever starting velocity.