Q6(a)



Figure 1 – Position and Velocity of Single Vessel No Drag



Figure 2 – Position and Velocity of Single Vessel Low Drag



Figure 3 – Velocity Position and Height of Sinking Vessel



Figure 4 – Velocity Position and Height of Explicit Euler Evaluation of Three Vessels

A group of graphs showing different types of waves

Description automatically generated with medium confidence

Figure 5 – Velocity Position and Height of 4th Order Runge-Kutta Evaluation of Three Vessels

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Case | System Parameters | | Vessel Parameters | | | | | | Final Parameters | | |
|  |  |  |  |  |  |  |  |  |  |  |
| Single Vessel, No Drag | 1 | 0.0 | 1.0 | 0.1 | 0.0 | -0.1 | 0.0 | 0.0 | -0.0995 | -0.2976 | 0 |
| Single Vessel, Low Drag | 1 | 0.1 | 5.0 | 0.5 | 0.0 | 0.1 | -0.5 | 0.0 | 0.01292 | 0.04506 | 0 |
| Sinking Vessel | 2 | 0.0 | 100.0 | 0.1 | 1000.0 | 0.0 | 0.0 | 0.0 | 16.4294 | 16.8492 | 15.8329 |
| Three Vessels | 10 | 0.01 | 1.0 | 0.1 | 20.0 | -0.1 | 0.0 | 0.0 | 0.1355 | -0.0279 | 0.06360 |
| 10.0 | 0.5 | 10.0 | -0.1 | 0.0 | 0.0 | 0.03972 | -0.1772 | 0.004434 |
| 50.0 | 10.0 | 1.0 | -0.1 | 0.0 | 0.0 | 0.006388 | 0.04621 | 4.9824× 10-6 |

Table 1 – Problem Specifications with Final Positions



Figure 6 – Displacement Relative to Bottom of Vessel (Single Vessel No Drag)



Figure 7 – Displacement Relative to Bottom of Vessel (Single Vessel Low Drag)



Figure 8 – Displacement Relative to Bottom of Vessel (Sinking Vessel)



Figure 9 – Displacement Relative to Bottom of Vessels (Three Vessels)

Q6(b) – By testing multiple vessels, changing for only parameter at a time we can determine that the mass, area and starting water level of the object have an affect on the vessel’s oscillation frequency. We determined that the mass and the starting water level of the object have a significant inverse relation to the frequency, while the area has a direct relation.



Figure 10 – Frequency Under Different Vessel Conditions

Q6(c) – The main advantage of using a compiled language such as C++ instead of an interpreted language such as MATLAB or Python lies in its efficiency. For example, the compilation process to interpret the language is very slow to execute and is a waste of time, especially when the code itself has not changed. In this assignment I found myself making changes to the input file but not the code, in turn I could quickly run the already compiled program. Additionally, complex numerical methods problems make many calculations and require massive amount of memory, something C++ excels at being an older readily used language. Many interpreted languages have memory leak issues where data is automatically overwritten by the interpreter, with C++ the developer has full control over where and what data is saved to the computer’s sometimes limited RAM. On top of that, C++ gives developers control over parallelism allowing the program to run different operations concurrently. While I did not use these features in this program, for a larger more time sensitive project this can be a valuable tool not supported by most interpreted languages.

Q6(d) – For this program I chose to isolate different systems to their own object, this allows for shared system specifications and allows for reading and evaluating multiple files/problems with one run of the program. I chose to use STL vectors and matrices due to their ability to dynamically allocation memory and makes changes to the size of the allocation after initialization. This avoids errors stemming fringe cases where timesteps introduce float point errors, while being less CPU efficient. On the other hand, to introduce memory and CPU efficiency I had multiple operations performed in the same iteration to avoid repetitive steps. I isolated variables to function and loops allowing for the memory to be relocated immediately after use and created and used the option to delete objects after use and wrote data to files to avoid having to rerun the program.