May 2021

Rigid Body Dynamics of Ship Hulls via Hydrostatic Forces Calculated From FFT Ocean Height Fields

Reagan Samuel Burke
Clemson University, rsburke@g.clemson.edu

Rigid Body Dynamics of Ship Hulls via Hydrostatic Forces calculated from FFT Ocean Height Fields

---

A Thesis
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Master of Fine Arts
Digital Production Arts

---

by
Reagan Samuel Burke
May 2021

Accepted by:
Dr. Jerry Tessendorf, Committee Chair
Dr. Eric Patterson
Dr. Victor Zordan

Abstract

An art tool is presented that utilizes a method for simulating the motion of ships in response to hydrostatic forces on the hull from a height-field representation of an ocean surface. Other forces modeled as a PID controller aid to steer the ship and stabilize the motion. The algorithms described can be applied to 3D models of arbitrary shapes composed of polygons "floating" on height fields generated from a myriad of additional spectra. The performance of the method is demonstrated in simple and complex ships, and ocean surfaces of flat, medium, and large waveheights.

Dedication

This paper is dedicated to David, and Tami Burke, who have done nothing but encourage me to pursue an advanced degree in a field I love. Thank you for your unconditional love and support.

Acknowledgments

I would like to thank Dr. Jerry Tessendorf, my thesis advisor, who's love for the Edmund Fitzgerald inspired me to take on such an ambitious project. Without his guidance and patience, this project would not be possible.

I would also like to thank my committee members, Dr. Victor Zordan and Dr. Eric Patterson for their guidance and resources throughout my time in the Clemson Digital Production Arts program.

Finally, I would like to thank my family and friends for their unconditional support and encouragement for my most ambitious endeavors.

Table of Contents

List of Figures

Chapter 1

Introduction

The purpose of this thesis is to describe an art tool for physically-based boat motion. This tool utilizes a method by which 3D models can appear to "float" on the surface of a body of water that is generated by Fast Fourier Transform waves. Buoyancy is the principle that an object will float in a fluid if its total mass is less than the mass of the volume of fluid it displaces. The body of water is a deep ocean surface represented by a height field generated via the Fast Fourier Transform algorithm and statistical analysis of ocean surfaces. Buoyant forces are generated that simulate the forces of fluid displacement based on the properties of faces of a 3D model and the properties of a fluid surface. Polygonal properties that affect buoyancy include surface area, orientation, depth under the surface, and surface normal. Fluid properties that affect the buoyancy are gravity, fluid density and wave height.

In the 2020 FX original series, DEVS, a company called Amaya develops a quantum computer capable of simulating past and future events with 100 percent accuracy. This gives Forest (played by Nick Offerman) and his team the ability to see major historical events unfold in realtime [2]. Similarly, one of the largest motivations for this project is to gain insight into the tragic sinking of the Edmund Fitzgerald. While simulating ship movement using the following demonstrated method is far from perfect– lacking physical phenomena that exist in our real world such as friction, underwater currents, water resistance, and surface tension– records of the night the Edmund Fitzgerald sank provide us with the information needed to observe ship movement in similar ocean conditions to that night [1]. The purpose of this thesis is not to design a visualization of the Edmund Fitzgerald's sinking, but rather to create directable artistic tool by which a recreation of such an

event can easily be made more dynamic.

The importance of the method used in the tool detailed in this thesis lays in its adaptability. Simulated objects do not need to be shaped like ships, but can be arbitrarily shaped 3D models. Mass can be distributed across the surface of the model with artistic control with a third-party tool such as the Maya script demonstrated. Describing the forces acting on the hull of the ship allows for control over the motion of the ship through the introduction of additional non-hydrostatic forces. To demonstrate this, a Proportional-Integral-Derivative (PID) controller is implemented to animate the ship's position and velocity while still allowing the ship to freely bob along the ocean surface. This PID controller is designed to eliminate error in the ship's position and velocity, however it can be used for other purposes such as dampening and animation.

This thesis describes the implementation of Alexadra L. Zheleznyakova's hydrostatic force algorithm [13], and a rigid body motion algorithm described by Dr. Donald House [4]. A Maya script is implemented to control the distribution of mass of an object. Two models with vastly different shapes and sizes are simulated and rendered on a series of 3 wave conditions varying from flat water to storm surges. These results compare motion between simulations with no dampening, with a PID controller dampening velocity, with angular velocity dampening, and with both PID controller velocity dampening and angular velocity dampening. The PID controller is used for animation purposes as well as dampening purposes.

1.1 Related Work

The methods described in this thesis would not be possible without Dr. Jerry Tessendorf's detailed notes on ocean surface height fields [8]. Large parts of this thesis are derived from the methods of calculating hydrostatic force as described by Alexandra L. Zheleznyakova, who states that these methods are relatively fast to compute depending on the size of the model being moved, and can be further optimized to run in realtime [13].

Previous methods of calculating ship buoyancy and rigid body motion have often included calculating body/water interaction using grids. In his 2014 paper, Dr. Timo Kellomäki describes a method of defining both flow blocking and surface floating objects using such a grid-based approach to flowing water [5]. In this paper, Dr. Timo Kellomäki also acknowledges the work of Dr. Jerry Tessendorf and his methods of generating ship wake on interactive FFT ocean wave surfaces [9].

Chapter 2

Research Design and Methods

A ship is described as a rigid body and the motion of this rigid body is found using existing methods such as defined in Dr. Donald House's book on physically-based animation [4]. In this thesis, rigid bodies are modelled as systems of particles with fixed distances from the center of mass that have a moment of inertia. Translation and rotation values are calculated via the sums of forces and torque acting on each of the particles in the system. The centroid positions of the triangular faces of a ship are used as the particles acting in this system. This means that all input geometry must be triangulated before running the simulation. The rotation and translation values calculated from the centroid forces are applied to the vertices in the boat's surface which creates visual movement.

2.1 Rigid Body Movement

The realistic movement of rigid objects can be approximated by treating points on the surface of a model as particles that are capable of interacting with their environment. To determine hydrostatic forces, the centroids of the triangle are used as these particles. The most important factor in rigid body motion is the position and velocity of the center of mass, and a rotation matrix calculated from the angular velocity and moment of inertia. The vertices' locations are updated every frame using the same rotation matrix and center of mass position that are calculated via the forces acting on the centroids.

The center of mass of a rigid body is treated as a single particle in this simulation, with a

velocity, position, and force attribute. Alone, it contains no rotation information. The forces acting on the centroids are summed together to create the total force acting on the total mass at the center of mass. The velocity and position are updated using Newton's law of motion $F = MA$ at discrete time steps.

The center of mass is moved with an explicit solver. This solver is an equation that transforms the continuous derivative of motion into a frame by frame update function using discrete time steps. The partial solvers required to build more advanced solver systems are defined as follows:

$$p_{cm_+}(\Delta t) = p_{cm_-} + v_{cm}\Delta t \tag{2.1}$$

$$v_{cm_+}(\Delta t) = v_{cm_-} + F\Delta t/m \tag{2.2}$$

In equations (2.1) and (2.2), the subscript minus and the subscript plus indicate values from the last time step's calculation and values for the current frame respectively. Mass is represented as m, amount of time between solutions is represented as Δt, force is represented as F, position is represented as p, and velocity is represented as v.

Calculating the rotation matrix is accomplished by first calculating the torque and the moment of inertia for the body. Torque is calculated as the sum of cross-products of vectors $r_a = c_a - p_{cm}$ and the force, F_a, acting on that centroid. r_a is the vector difference between the centroid c_a and p_{cm}. The magnitude of r_a remains constant for throughout the simulation.

$$\tau = \sum_a r_a \times F_a \tag{2.3}$$

The moment of inertia represents the amount of rotational inertia an object has at any given moment of the simulation. This information is useful for finding the angular velocity of an object which is a simpler representational "snapshot" of an objects rotation encoded in a vector. Each element of the 3x3 moment of inertia matrix, $\mathbf{I}_{ij}$, is calculated individually by summing the product of each particle's mass, m_a, and the difference between the product of the product of vector components corresponding to the matrix element being processed from the Kronecker delta, δ_{ij}, and r_a. Using the inverse of this moment of inertia, angular velocity ω is calculated using a constant

delta time.

$$\mathbf{I}_{ij} = \sum_a m_a(\delta_{ij}|r_a|^2 - (r_a)_i(r_a)_j) \tag{2.4}$$

$$\omega_+(\Delta t) = \omega_- + \mathbf{I}^{-1} \cdot \tau \Delta t \tag{2.5}$$

In an equation such as (2.5), the subscript minus and subscript plus indicate the values of ω at the previous and current time step respectively.

The rotation matrix $\mathbf{R}$ is the sum of 3 term. The first term is a product of the identity matrix and the cosine of the magnitude of the angular velocity, which represents an angle. This angle is scaled by Δt. The second term is the outer product of the direction of the angular velocity by itself times 1 minus the cosine of the magnitude of the angular velocity. This magnitude of ω is scaled by Δt. The third term is the sum of the components of the Pauli matrices at index i times the direction of the angular velocity and the sin of the magnitude of the angular velocity. The magnitude of ω is scaled by Δt to provide a matrix representing only the amount of rotation a body has accomplished in the last time step.

$$\mathbf{R}(\omega, \Delta t) = \mathbf{1}\cos|\omega\Delta t| + \hat{\omega} \otimes \hat{\omega}(1 - \cos|\omega\Delta t|) + (\sum_{i=0}^{2} \tau_i \hat{\omega}_i \sin|\omega\Delta t|) \tag{2.6}$$

The Pauli matrices used in equation (2.6) are defined as follows:

$$\tau_0 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \quad \tau_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \quad \tau_2 = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.7}$$

Each rotation matrix describes the amount of rotation the object has completed in 1 frame. To describe the cumulative rotation $\mathbf{R}_{cml}$ of the vector r_a, each matrix needs to be multiplied by the matrix calculated the previous frame. Multiplying the cumulative matrix by r_a and offsetting r_a by p_{cm} provides a description of the final position of the centroid: $c_a = p_{cm} + (\mathbf{R}_{cml} r_{a_{init}})$ where $r_{a_{init}}$ is the initial vector between a vertex and the initial center of mass. Both the centroid and vertex positions need to be updated using this single rotation matrix and center of mass.

Solvers represent operations that update the values of rigid body attributes at discrete time steps. The partial solvers that generate one update for each attribute necessary for rigid body

motion are as follows:

$$S_p(\Delta t)\ p_{cm_-} = p_{cm_+}(\Delta t) \tag{2.8}$$

$$S_v(\Delta t)\ v_{cm_-} = v_{cm_+}(\Delta t) \tag{2.9}$$

$$S_R(\Delta t)\ \mathbf{R}_{cml} = \mathbf{R}(\omega, \Delta t)\ \mathbf{R}_{cml} \tag{2.10}$$

$$S_w(\Delta t)\ \omega_- = \omega_+(\Delta t) \tag{2.11}$$

$$S_c()c_a = p_{cm} + (\mathbf{R}_{cml} r_{a_{init}}) \tag{2.12}$$

The combination of the partial solvers in equations (2.8)-(2.12) make an explicit solver called the leapfrog solver S_{lf}. To update the attributes of the rigid body, S_{lf} updates attributes directly related to position, rotation, and inertia 2 times per sub step with half the value of Δt, and updates velocity and angular velocity only once per time step with the full value of Δt.

The following set of operations defines a combination of partial solvers that make up the rigid body leapfrog solver. The center of mass position is updated with S_p, which updates p_{cm} once with half the value of Δt. The rotation matrix is calculated with S_R, and the centroid positions are moved relative the new value of p_{cm}, and rotated by the new value of $\mathbf{R}$ with S_c. The moment of inertia is calculated with S_I using equation (2.5) at the current moment. At this point, centroids have been moved to a new location, and so forces are calculated before updating the velocity to ensure the most accurate results. Velocity then updates S_v using the full value of Δt. Angular velocity updates with S_ω using the full value of Δt. Then the same process leading up to, but not including, the calculation of forces prior to the velocity update is repeated to complete the update of a single time step.

$$S_{lf}(\Delta t) = S_I\ S_c()\ S_R(\Delta t/2)\ S_p(\Delta t/2)\ S_w(\Delta t)\ S_v(\Delta t)\ S_I()\ S_c()\ S_R(\Delta t/2)\ S_p(\Delta t/2) \tag{2.13}$$

This leapfrog solver can be applied multiple times to form the sixth-order solver S_{six}. Coefficients A and B are defined as $A = 1/(4 - 4^{1/3})$ and $B = 1 - 4A$ [6].

$$S_{six}(\Delta t) = S_{lf}(A\Delta t)\ S_{lf}(A\Delta t)\ S_{lf}(B\Delta t)\ S_{lf}(A\Delta t)\ S_{lf}(A\Delta t) \tag{2.14}$$

2.2 Ocean Surface Construction

Visually appealing oceans can be modeled using a function of position and time. A set of random time-dependant values represented by a spectrum are used as input that gets transformed via the Fast Forier Transform (FFT) algorithm. The application of the FFT algorithm, which is typically used for frequency analysis, transforms the spectrum of frequencies into a grid of amplitudes. Interpolating along this grid creates a reasonable and continuous approximation of the height of an oceans surface.

A variety of frequency spectra may be transformed via FFT to generate ocean height fields. This project implements both the TMA and the Ochi spectrums [3, 11]. Described below is the Philip's spectrum. The Philip's spectrum can be constructed with the numeric constant A, a wave vector k, a limit to the wave height L (described by $L = V^2/g$) and the direction of the wind, $\hat{\omega}$.

$$P_h(k) = A\frac{\exp(-1/kL)^2}{k^4}|\hat{\mathbf{k}} \cdot \hat{\omega}|^2 \tag{2.15}$$

Using this spectrum the heights are found with the following method:

$$\overline{h}_0(\mathbf{k}) = \frac{1}{\sqrt{2}}(\xi_r + \xi_i)\sqrt{P_h(\mathbf{k})} \tag{2.16}$$

Where ξ_r and ξ_i are random numbers with a Gaussian distribution between 0 and 1. Amplitudes along the position k at time t can be found as

$$\overline{h}(\mathbf{k}, t) = \overline{h}_0(\mathbf{k})exp\{i\omega(k)t\} + \overline{h}_0^*(-\mathbf{k})\exp\{-i\omega(k)t\} \tag{2.17}$$

Here, i is the complex unit $i^2 = -1$. Finally, amplitudes along the wave spectrum are transformed via FFT where X and Y are components of the wind direction vector.

$$h(X, Y, t) = \sum_{k_x k_y} \overline{h}(k_x, k_y, t)\exp\{i(k_x X + k_y Y)\} \tag{2.18}$$

2.3 Centroid Submersion and Hydrostatic Force

Hydrostatic force is calculated for all centroids beneath the surface of the water. To determine whether a centroid is under water, the centroid's vertical world position is compared to the

height value stored in the height map at the location on the map that the centroid projects onto. An offset is applied if the ocean's sea level is not positioned on the axis-aligned horizontal plane. If the difference between the the height of a wave and the centroid's position relative to the wave is negative, then the centroid is under water and the forces are applied. Otherwise, only gravity affects the centroid. For centroids that are determined to be submerged, the hydrostatic force applied to that centroid is described as the product of water density ρ, gravity $g = 9.8m/s^2$, centroid depth d_a, triangle surface area A_a, and triangle normal $\hat{n}_a$. Note, this triangle normal is likely a rotated value at any time step beyond $t = 0$

$$F_{hydro} = -\rho g d_a \hat{n}_a A_a \tag{2.19}$$

This force is added to the total force acting on the center of mass, and contributes to the calculation of the torque of the rigid body. Gravitational acceleration, defined as $g = 9.8\ m/s^2$ in a downward direction, is also applied to all of the centroids. These forces are generated just prior to the velocity update in equation (2.14).

2.4 Weight Distribution

Problems arise if each centroid has the same mass. If the triangles in the ship model are not approximately uniformly sized, then large, flat portions of the ship composed of fewer big triangles don't weigh as much as much denser portions of the model made of many small triangles. This can cause naturally buoyant areas of the ship to sink, and naturally heavy areas of the ship to reflect off the surface of the ocean or bounce out of the water. In the most extreme cases, the boat may accelerate upward rapidly or perform a series of flips. A solution to this problem is to uniformly distribute the total mass, m_T, across the centroids as a factor of the rigid body's total surface area A_{tot}.

$$m_a = \frac{m_T}{A_{tot}} A_a \tag{2.20}$$

Using equation (2.20), approximations of real-world ship masses can be distributed across digital rigid body counterparts. Using this approach, boats may begin to roll onto their side as this is not how weight is distributed in many real boats. In reality, ships are much heavier toward the

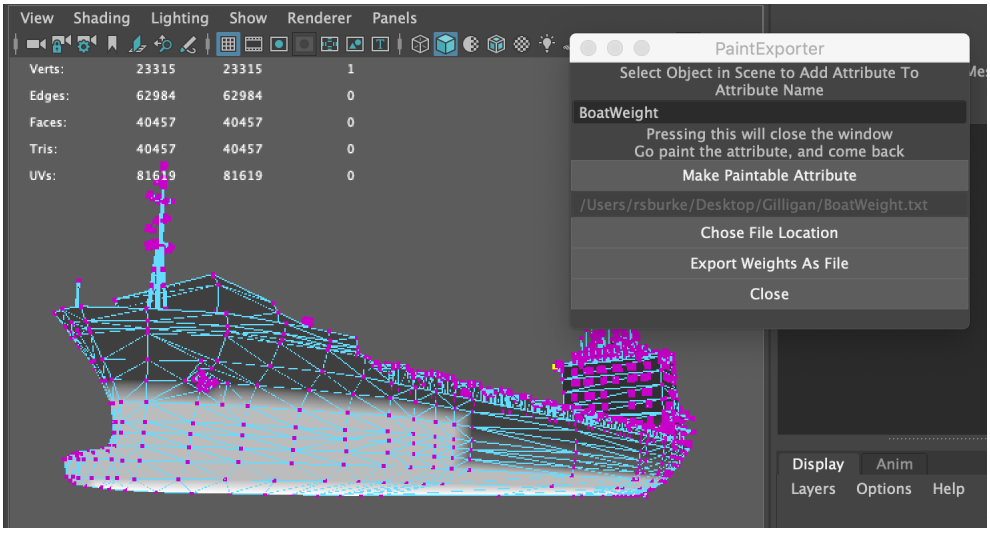

Figure 2.1: A screenshot of the script implemented in Maya can be seen that facilitates the exporting of weighted influences to be used to distribute total mass.

bottom of the hull where engines and cargo may reside, and much lighter towards the top.

Third-party tools, such as a custom script for Maya, may be used to provide a level of artistic control over the location of the center of mass of the ship and the mass of individual centroids. With a painting tool, a floating point value between 0 and 1 is assigned to each vertex of the ship. These values are then exported to a file. A demonstration of the interface of this script, and a visual representation of the floating point weight values can be seen in figure 2.1. Once the weights have been written to a file and read into an array or similar data structure, the mass of the ship is distributed using the weighted values via the methods outlined in equations (2.21) through (2.23).

First, a coefficient, D is calculated from the sum of the products of every triangles' vertex weights and its area. The weights are shown as W_a, W_b, W_c and the area of the triangle is shown as A_c. Next, each weight needs to be scaled by a factor of the total surface area of the model over D. These adjusted weights can be used to redistribute the total mass of the ship to triangles' centroids.

$$D = \sum_{centroids} A_c(W_a + W_b + W_c) \tag{2.21}$$

$$W'_v = W_v \frac{A_{tot}}{D} \qquad (v = a, b, c) \tag{2.22}$$

$$m_a = \frac{m_T}{A_{tot}}(W'_a + W'_b + W'_c)A_a \tag{2.23}$$

Once this has been accomplished, the ship will float in a variety of different ways depending on how the weight is painted onto the boat. It is important to keep the center of mass low yet centered between the bow and the stern. This will prevent the ship from sinking head-first or tail-first into the ocean.

2.5 PID Controller and Dampening

In an environment with no friction forces, rigid body motion can become unstable. To combat this instability with a level of control, a PID controller is used. In cases where a PID controller is not adequate for stabilizing the motion of the rigid body, additional angular velocity dampening factors may be added.

Sometimes, the ship may rock slightly to the side and find equilibrium there. If this is the case, the forces may push against the side of the boat and move it sideways. A PID controller attempts to correct this error by creating forces that counteract the ones pushing the boat sideways. While designing ships with a shape more optimal for floating in water may mitigate this issue, the purpose of this thesis is to describe a method in which arbitrary polygonal shapes may float on the surface of a height field. Therefore, a PID controller proves useful in allowing less optimally shaped objects float without accumulating error in velocity or position.

The controller is a summation of 3 terms. The first term represents a springlike force proportional to the error, $e(t)$, between the position of the ship and the desired position of the ship. The second term is an integral, approximated by a running sum of the errors at every frame, which makes the corrective force stronger every frame the ship is away from it's intended position. The third term is a derivative of the positional error, which represents the velocity the ship should have at any given time. Desired positions and velocities are treated as input variables, and the strength of each component of the force is adjusted with input parameters κ_p, κ_i, and κ_d.

$$F_{PID} = \kappa_p \; e(t) \; + \; \kappa_i \int e(t)dt + \kappa_d \frac{de}{dt} \tag{2.24}$$

A PID controller is also used to generate a dampening force on the ship. When a ship is too high or too low in the water when the simulation begins, the ship will oscillate up and down. These oscillations are caused by gravitational force when centroids are unsubmerged and hydrostatic forces

when centroids are submerged. To reduce the energy causing these oscillations and to ensure the ship reaches a stable resting state, a desired speed of 0 kph, and coefficient values $\kappa_p = 0$, $\kappa_i = 0$, and $\kappa_d > 0$ is used in the PID controller. This causes an effect that strictly reduces the velocity of the rigid body, without attempting to navigate the ship towards a positional target. A downside of dampening motion in this manner is that forces are only being applied to the center of mass' position and velocity. This means the rotation of the ship may still destabilize. To correct the instability in the rotation, angular velocity is also dampened by subtracting a small portion of the last time step's angular velocity from the angular velocity calculated during the current time step.

By using a PID controller to specify target positions and velocities for a ship at varying times during the simulation, an animation system is constructed. Similar to a keyframe animation method, the ship will move toward its intended target at a time specified. It will react to the forces of the water acting on its hull as it moves towards the target position.

2.6 Proxy Geometry

Geometry that is optimal for rendering purposes will differ from what is considered optimal for the sake of simulation. Approximately evenly sized and evenly spaced triangles are preferable for the purposes of simulating interactions between a ship and water, but geometry optimized for rendering– particularly hard surface models such as the ones that represent ships– may have unevenly sized triangles on large patches of flat surface area compared to highly detailed areas. A solution to this problem is to simulate motion using a proxy model that is optimized for simulation, and apply the translation and rotation values resulting from the simulation to a piece of geometry optimized for rendering. Figures 2.2 and 2.3 illustrate the dramatic difference hydrostatic force has on the motion of the ship between the two geometries. These graphs are generated by simulating 500 frames of both render geometry and proxy geometry motion on a flat, waveless ocean surface. The first graph in each figure demonstrates the effect of much less accurate force calculations due to less gradual centroid submersion. The second graph from each of these figures demonstrates the expected sinusoidal motion of the ship's rocking.

Figure 2.4 demonstrates the difference between render geometry and simulation geometry. A model is generated by the DynaMesh operator in ZBrush. This is used to create a less visually detailed model with more surface triangles than the original mesh. The generated triangles are of

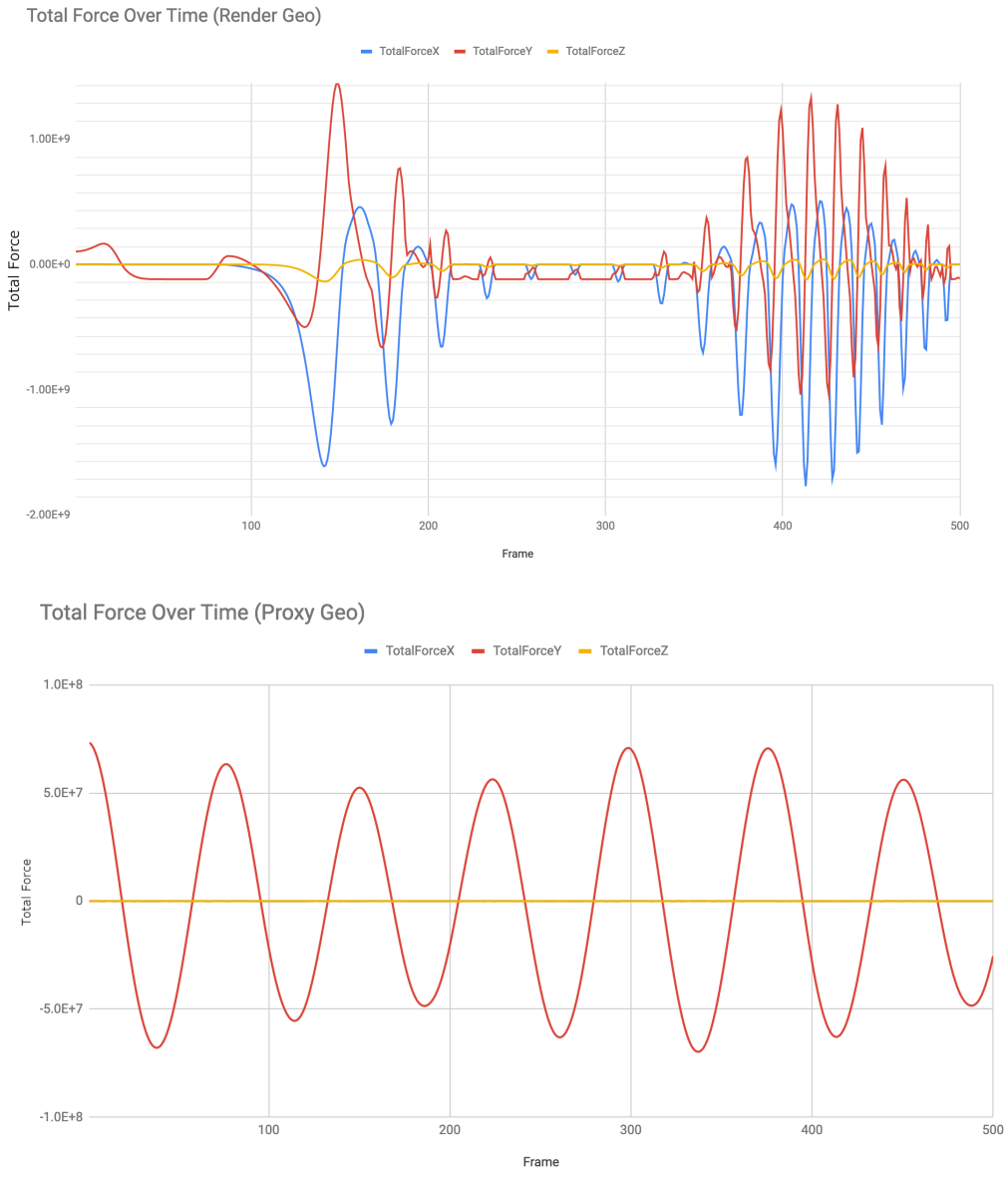

Figure 2.2: A comparison between the total force observed in the render geometry versus the proxy geometry on flat water. Top: Render optimized geometry. Bottom: Simulation optimized geometry.

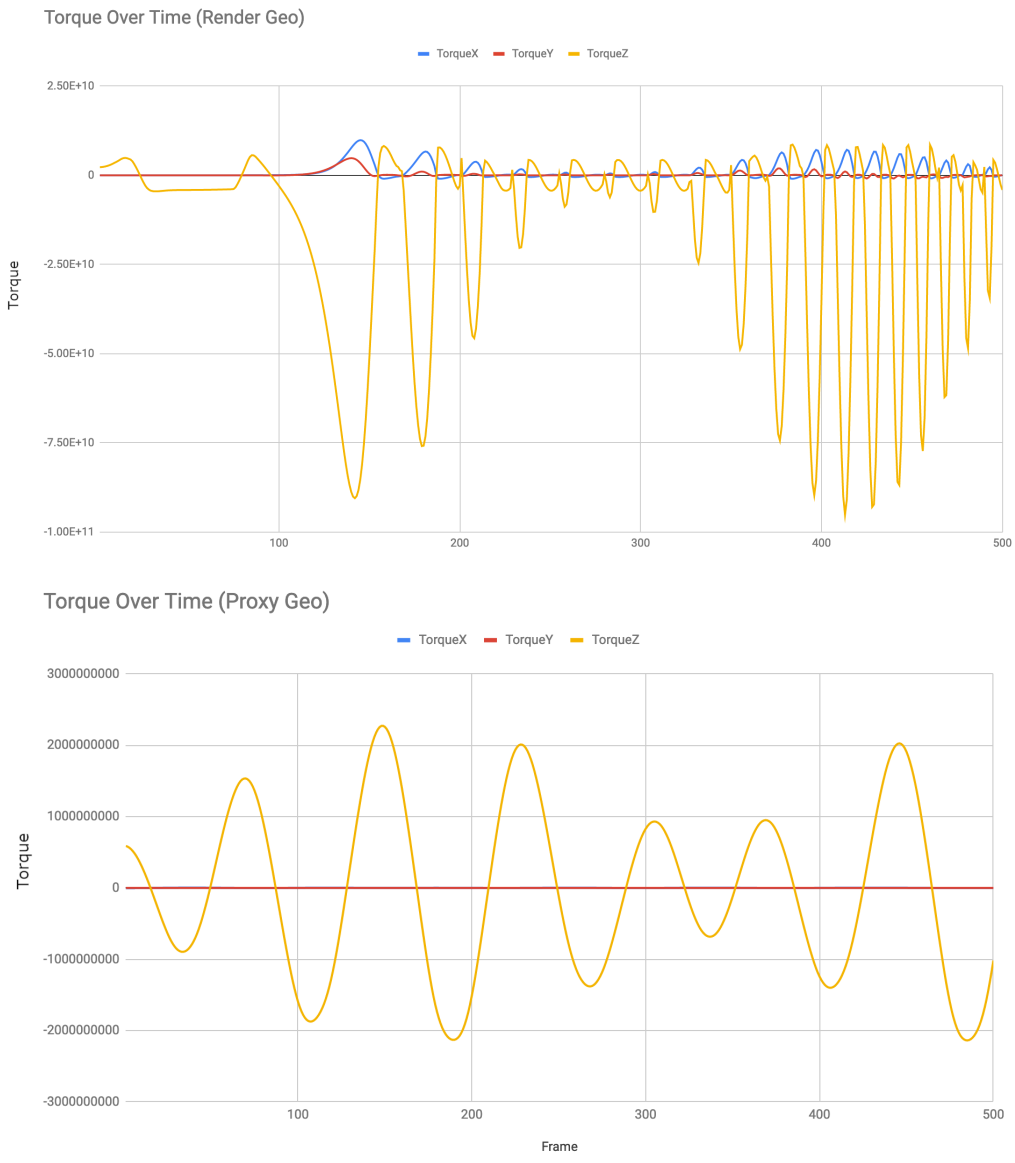

Figure 2.3: A comparison between the torque observed in the render geometry versus the proxy geometry on flat water. Top: Render optimized geometry. Bottom: Simulation optimized geometry.

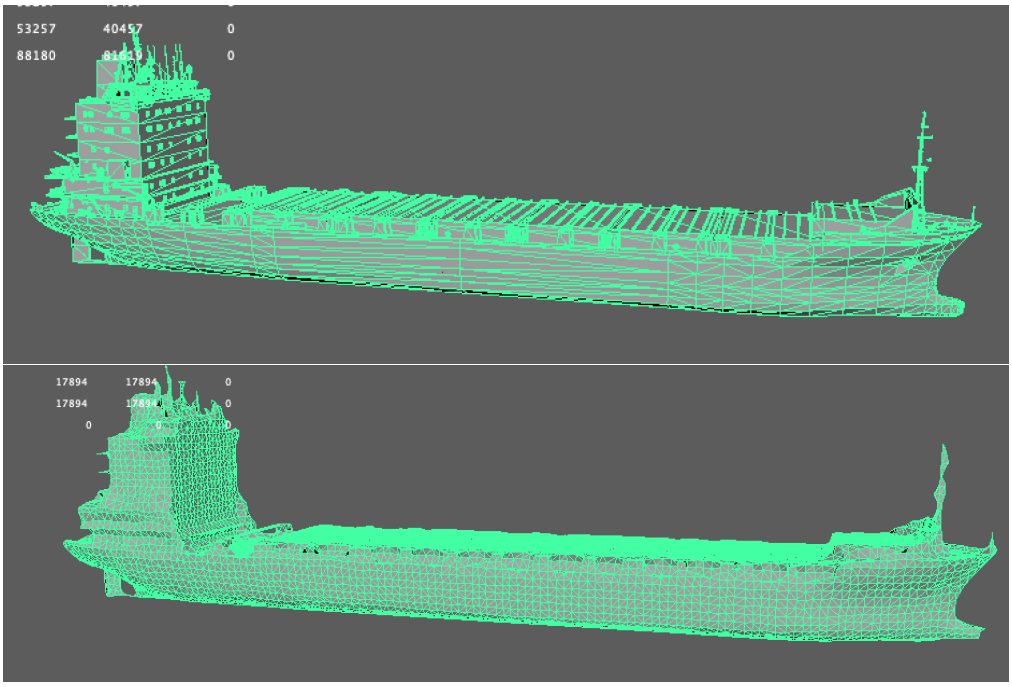

Figure 2.4: A comparison between 2 models. Top is a sparse model optimized for rendering. Bottom is a fine, even model optimized for simulation.

approximately even size and spacing.

Simulating ship motion using proxy geometry defines a rotation matrix and a center of mass position. To extract the rotational vector from the rotation matrix in the simulation, and to apply this rotation vector to the rendered ship, a version of equations 20-23 from Dr. Jerry Tessendorf's paper on motion blurring clipped triangles is used [10]. The center of mass is taken directly from the simulation and used as an offset to the location of the rendered geometry.

Chapter 3

Results

The motion generated in this method of rigid body simulation with the described hydrostatic force visually resembles the motion of a ship on an ocean surface. The forces acting on the ship differ from the forces acting on real ships, because the forces as described are a product of factors such as surface area and depth instead of volume, and certain properties of water such as current and water resistance are not present in height field representations of ocean surfaces. This means that the primary purpose for the simulation methods are of artistic visualisation opposed to precisely accurate recreation of ship motion. As such, the results of the simulation depend on artistic control of input parameters. The user is responsible for choosing a PID controller force and dampening variables as well as painting mass distribution weights onto the ship. A poor selection of such variables will lead to an unstable or unappealing simulation, while a good selection of variables may create a variety of interesting effects on the ship. In this sense, the algorithms discussed are used as a functional creative tool.

3.1 Ship Movement on Varying Surfaces

The algorithms and methods described are used to facilitate motion of a small torus model, and a large ship model driven by optimized proxy geometry. Both models are simulated on 3 ocean environments: a flat water environment where wave height is a constant 0 m, a mild ocean environment generated by an ochi spectrum [7] with average wave height of 0.54 m, and storm waves generated by a TMA spectrum [11, 3] with an average wave height of 2.0 m and a wind speed of

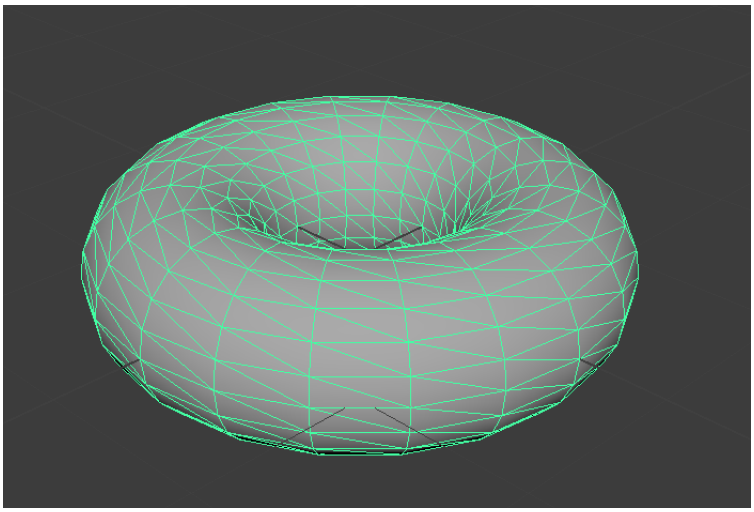

Figure 3.1: The wireframe of the torus model used in the simulations.

30 kph, a fetch of 100 km, and a bottom depth of 200 m. Fluid density is defined as 1020 kg/m^3. Animations of 500 frames are rendered at 20 fps with 100 additional solver substeps calculated by the sixth solver between rendered frames. For each wave condition, the hydrostatic forces are applied to the rigid body under the following conditions: with no additional dampening forces, with a PID controller, with angular velocity dampening, and with both a PID controller and angular velocity dampening. Simulations with an active PID controller use it in a way such that it functions as a velocity dampening component only, having a desired speed of 0 kph, $\kappa_d = 10$, $\kappa_p = 0$, and $\kappa_i = 0$. The position and integral components have no effect. Rigid body simulations containing an angular dampening component have their angular velocity at any given time step reduced by a factor of 0.001 of the previous time step's angular velocity.

3.2 Donuts at Sea

A simple torus shaped model is used as a ship for quickly testing the reliability of the methods and its corresponding input parameters. The torus has a uniform distribution of mass $m = 500\ kg$, as defined in equation 2.20. The torus has a radius of 1 meter and a sectional radius of 0.5 meters. Once triangulated, it consists of 800 faces of approximately uniform size. A wireframe of this model is shown in figure 3.1. Simulating with this model makes debugging problems with the implementation more readily apparent than simulating with a complex or asymmetrical ship model.

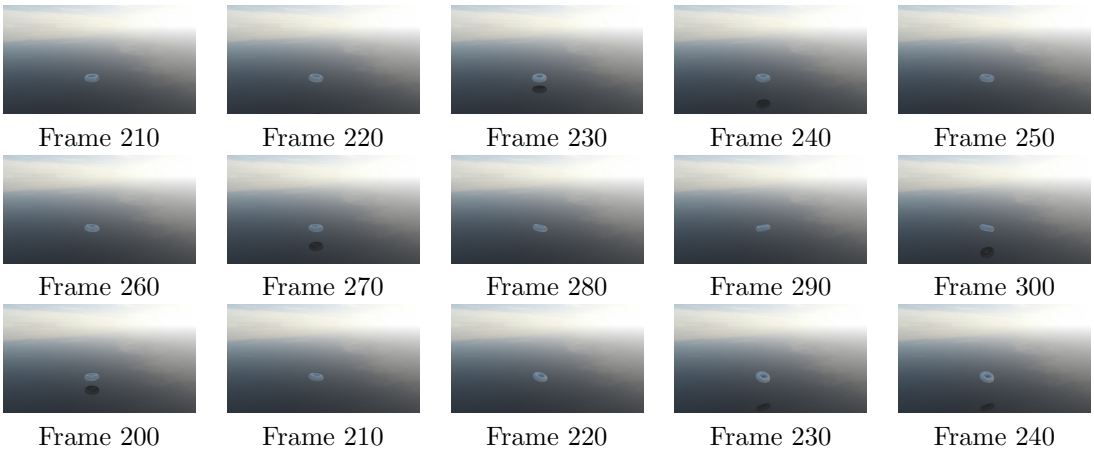

Figure 3.2: An up and down, and side to side rocking motion is observed on a torus when flat waters have no dampening values introduced.

3.2.1 Flat Water

When simulating the movement a torus with no additional dampening forces on a flat ocean surface, if the torus model does not begin at a state of equilibrium between ocean forces and gravity, the rigid body bounces vertically on the surface of the water without ever reaching a stable state of rest on the surface of the water. Small mathematical errors in the force calculation, or minor asymmetries in the surface of the model, accumulate over time, causing the torus to rock side to side with each oscillation. A series of frames demonstrating this motion is shown in figure 3.2.

Using a PID controller to control the amount of velocity a rigid body has is effective at reducing the number of bounces a rigid body will perform on the surface of the water before reaching a resting position. Alexandra Zheleznyakova demonstrates that with sufficiently strong dampening forces, this rigid body position of equilibrium between gravity and hydrostatic forces can be found, and finding it is a useful pre-processing calibration step for deciding optimal ship placement at the beginning of a simulation [13]. In this case, a torus model beginning half submerged in the water with $\kappa_d = 10$ is not enough to find the resting position within 500 frames. Similarly to when no dampening forces are applied, the torus eventually accumulates small amounts of rotational energy and turns towards its side. Given enough time to run, and a sufficiently strong PID, the ship will find a point of equilibrium. This process can be fast, but in the results demonstrated, the values are not arranged in a way that this is the case.

Simulating the rigid body motion of the torus with angular velocity dampening and no PID

controller yields results similar to the results found when no dampening is used. Though in this case the torus does not gain the required rotational energy necessary to rotate onto its side within 500 frames. The torus appears to bounce on the surface of the water indefinitely with little to no change in rotation at all.

Adding a PID controller to the simulation with angular velocity dampening prevents the torus from rolling onto its side as well as reducing the height of each successive bounce on the surface. Simulating a model's motion on flat water with an angular dampening term and a sufficiently strong PID controller to reduce ship velocity is the most stable method described in this thesis to determine the position of equilibrium for a ship model.

3.2.2 Mild Waves

Simulating rigid body motion with mild waves creates more dynamic rigid body movement than simulations involving no waves at all. Unlike during the simulations on flat water, the rigid body travels horizontally a small amount. This is because the rigid body could interact with the sides of waves, and respond to them, and be pushed along.

As with the simulation on flat water, when no dampening terms are added, the torus eventually turns towards its side and begins to spin similar to a coin spinning on a desk. The center of mass is not moving much, however the energy of the waves has been preserved with no friction, and transformed into rotational energy.

Since the $\kappa_d = 10$ configuration of the PID controller is meant to reduce the velocity of the rigid body's center of mass, then once the center of mass has reached a stable state, the PID controller stops affecting the simulation on a large scale. Once the torus begins to spin like a coin, the center of mass position remained mostly stable. The torus reaches this spinning state faster when using this PID controller. This is possibly because the PID controller limited the speed at which the torus can move away from waves that will contribute to the rigid body's rotational energy. Horizontal movement has been reduced compared to the simulation with no dampening, because the PID controller limits the speed at which the body may travel on this plane, at the cost of more angular motion.

When an angular velocity dampening term is added to the simulation without a PID controller, the results of the simulation began to resemble what is visually expected of a torus shaped floating body. Because the dampening term is small, and the torus geometry naturally is very rota-

Figure 3.3: A visualization of the torus with no PID controller and no angular velocity dampening interacting with mild ocean waves. Frame 91.

tionally responsive to the water due to its low mass, the rigid body is still able to rotate with ease and follow the dynamic motion of the waves. The ship moves horizontally along the surface plane in a controlled manner in response to the waves. Adding a PID controller further stabilizes this, by preventing the torus from moving from its intended position on the ocean surface too quickly. A visualization of the torus on the ocean surface can be seen in figure 3.3

3.2.3 Large Waves

Simulating the torus model's motion with no dampening on large storm waves produced a bouncing motion. This bouncing motion appeared to be the result of fine ocean surface detail "bumping" against the model as it moves down the slope of a large wave. The torus can be seen moving along the surface of the water with more speed than in any other simulations run.

The PID controller slows this movement down, but does not prevent it from occurring. When using the PID controller with no angular velocity dampening on large waves, the ship moves much more gradually along the horizontal plane. A PID controller may be configured to prevent large deviation from a ship's starting position, however this is not the intent of the current configuration. The purpose of the current PID controller settings is to act as a form of velocity dampening

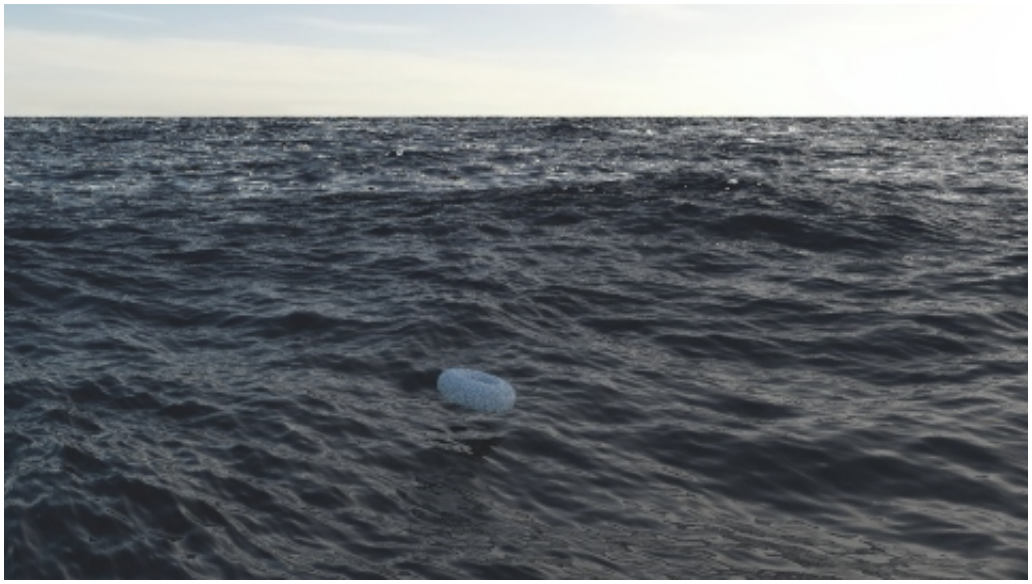

Figure 3.4: A torus in a small free fall. Too much rotational dampening is applied and the resulting motion looks stiff. Frame 266.

only. Observing the movement along the plane provides meaningful insight into the interactions between the ship and the ocean field. As observed in the simulation on mild waves, introducing a PID controller in this fashion causes the model to begin flipping in addition to the spinning motion seen in previous simulations. This flipping motion involved rotation over an axis tangent to the circumference of the torus. The initial bouncing motion observed in the simulation with no dampening is not as noticeable, but the flipping motion makes this less apparent.

Applying angular velocity dampening and no PID controller prevents the torus from flipping, but not from bouncing off the water. This produces an effect that does not look believable, as the torus enters short periods of free-fall without following through with it's rotations. This demonstrates a downside to limiting the angular velocity in the manner demonstrated. As long as the ship is in contact with the water, and the water is providing force that enabled the ship to spin, then dampening is a useful tool to ensure the stability of the simulation. However, once the ship leaves the water, and no force is acting on the angular velocity, the dampening action becomes too strong for the expected rotational resistance air should provide, and the results begins to look stiff. This short period of free fall is seen in figure 3.4. The torus also moves along the plane of the ocean much less than in large waves simulations without an angular velocity dampening term, because the torus spends less time in the air, and more time being directly moved by the waves.

Figure 3.5: A demonstration of the weight distribution used in the simulations. Values range from 0 to 1 with black representing 0 and white representing 1.

3.3 Cargo Ship with Proxy Geometry

To demonstrate the methods described on a ship with a real world counterpart, simulations featuring a model of the Maersk Arun cargo ship are computed. As described in section 2.6, proxy geometry drove these simulations. A mass totalling 12000000 kg is painted onto the hull of the ship as described in section 2.4 to keep the center of mass low and balanced. This is a reason for painting the bottom of the ship with higher values than the top, as seen in figure 3.5, and for adding more weight to the front of the ship than the back. The extra weight in the front helps offset the mass of the cabin at the back of the ship.

3.3.1 Flat Ocean Water

Simulating the ship on a flat water surface with no PID controller or additional dampening terms makes the ship rock back and forth while the center of mass bobs up and down with a spring-like motion. The ship never fully leaves the water, although due to the configuration of the distribution of mass, some parts of the ship sink lower than others. These areas react to the water with more motion. Since this environment is frictionless, the rocking motion will not stop without additional drag forces being applied.

Introducing a PID controller to this simulation causes the center of mass' vertical oscillations to slow down before the rotational energy pumps the boat back upward. Like in the previous scenarios, even if the center of mass becomes steady, rotation needs to be dampened to stabilize the motion. This conservation of energy is not an error but a feature of the system's design. It

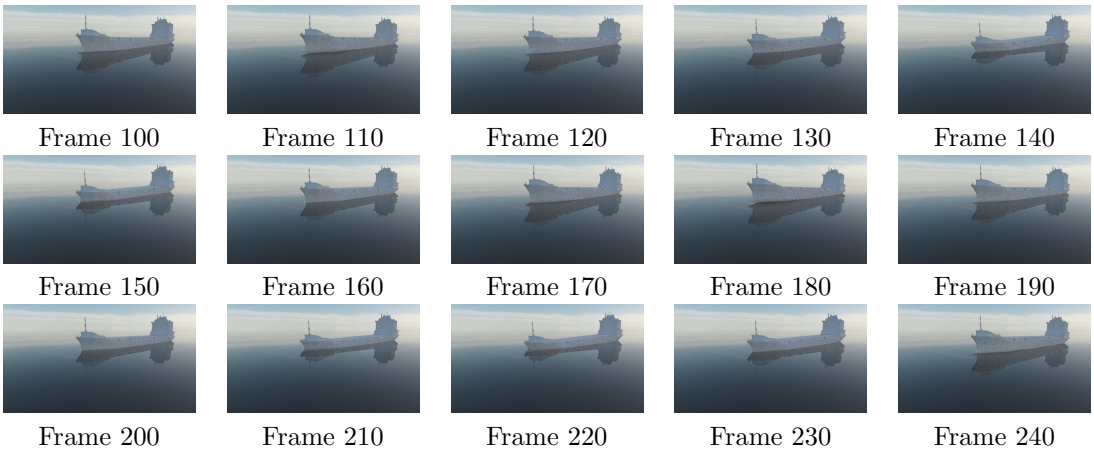

Figure 3.6: A back and forth rocking motion can be observed in the undampened ship with no PID controller on flat water.

is observable and expected that the forces exerted on the center of mass by the PID controller be proportional to the mass of the object being moved. Higher coefficients may be added to the PID controller to quicken the stabilization process.

Dampening the angular velocity without applying the PID controller greatly stabilizes the rocking motion of the ship. In this scenario, the ship still oscillates up and down with a steady motion. This motion will not stop or slow down without a dampening force such as a PID controller. Comparing the motion in figure 3.7 to the motion in 3.6 reveals how this angular velocity dampening term prevents the naturally occurring rocking almost entirely.

Adding the PID controller to a simulation with angular velocity dampening causes no immediately noticeable change when compared to the simulation containing angular velocity dampening and no PID controller. This is because the amount of inertia that a ship weighing 12000000 kg is not easily overcome by small forces. At this scale, a PID controller with $\kappa_d = 10$ is a small force relative to the hydrostatic forces being applied. The PID controller is a useful tool for stabilizing boat motion, and can be turned on or off at any point in the simulation. It is not the goal to completely stabilize the boat in any of these simulations, but rather to demonstrate what effects, if any, it will have on different ship types and different ocean environments. To stabilize a ship in the flat water environment, both the PID controller and the angular velocity dampening components may be turned on for a short period of time, then removed from the simulation.

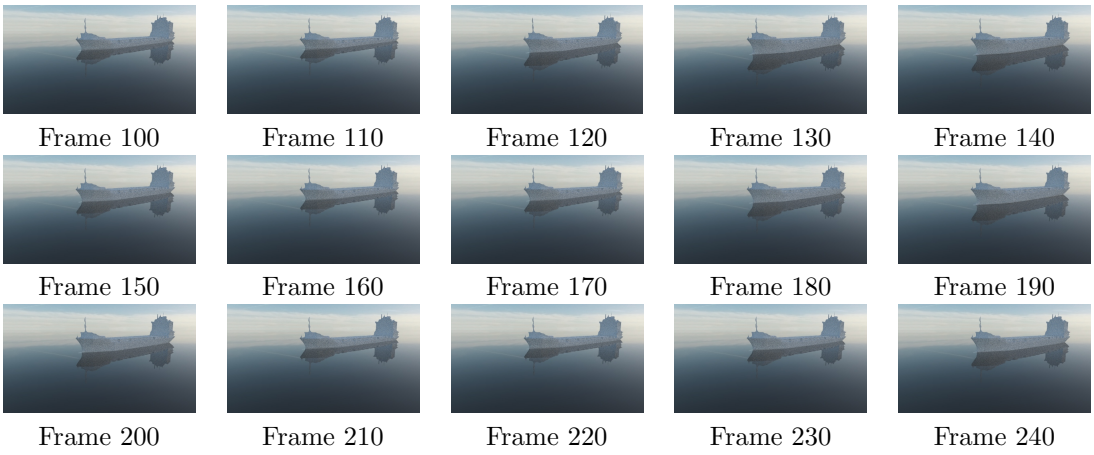

Figure 3.7: An up and down oscillation with no rotation is observed on a ship with no PID controller and added angular velocity dampening factor of 0.001 on flat water.

3.3.2 Mild Ocean Waves

Simulating rigid body motion with no PID controller and no rotational dampening on mild ocean waves yields similar results as seen in similar simulation scenarios on flat ocean waters. This is because the ship has enough mass and inertia to be mostly unaffected by the height of these small waves. When comparing the first and the last frame of the simulation, however, it becomes obvious that the forces of the waves are acting on this ship and causing it to move in a small yet noticeable way. Most notably, the ship rocks side to side a small amount in addition to the back and forth rocking motion that is observed on flat ocean water. Two images 120 frames apart from each other were included in figure 3.8 to demonstrate the stability of this ship geometry on mild ocean waves.

Including the PID controller with its input values as described in section 3.1 to this simulation again has little effect on the results of the simulation. The forces simply aren't strong enough to make a noticeable different in movement within 500 frames.

Removing the PID controller and adding angular velocity dampening results in the ship bobbing up and down on the water with little rotational energy. While it seems as though there is too much rotational dampening, as the ship does not react much to the waves, it is easier to determine this when the waves are much larger as small waves have little effect on a ship this size.

Observing movement from a simulation including both the PID controller and angular velocity dampening yields results similar to simulations in which angular velocity dampening and no PID controller are calculated. Again, this is to be expected because the force of the PID controller

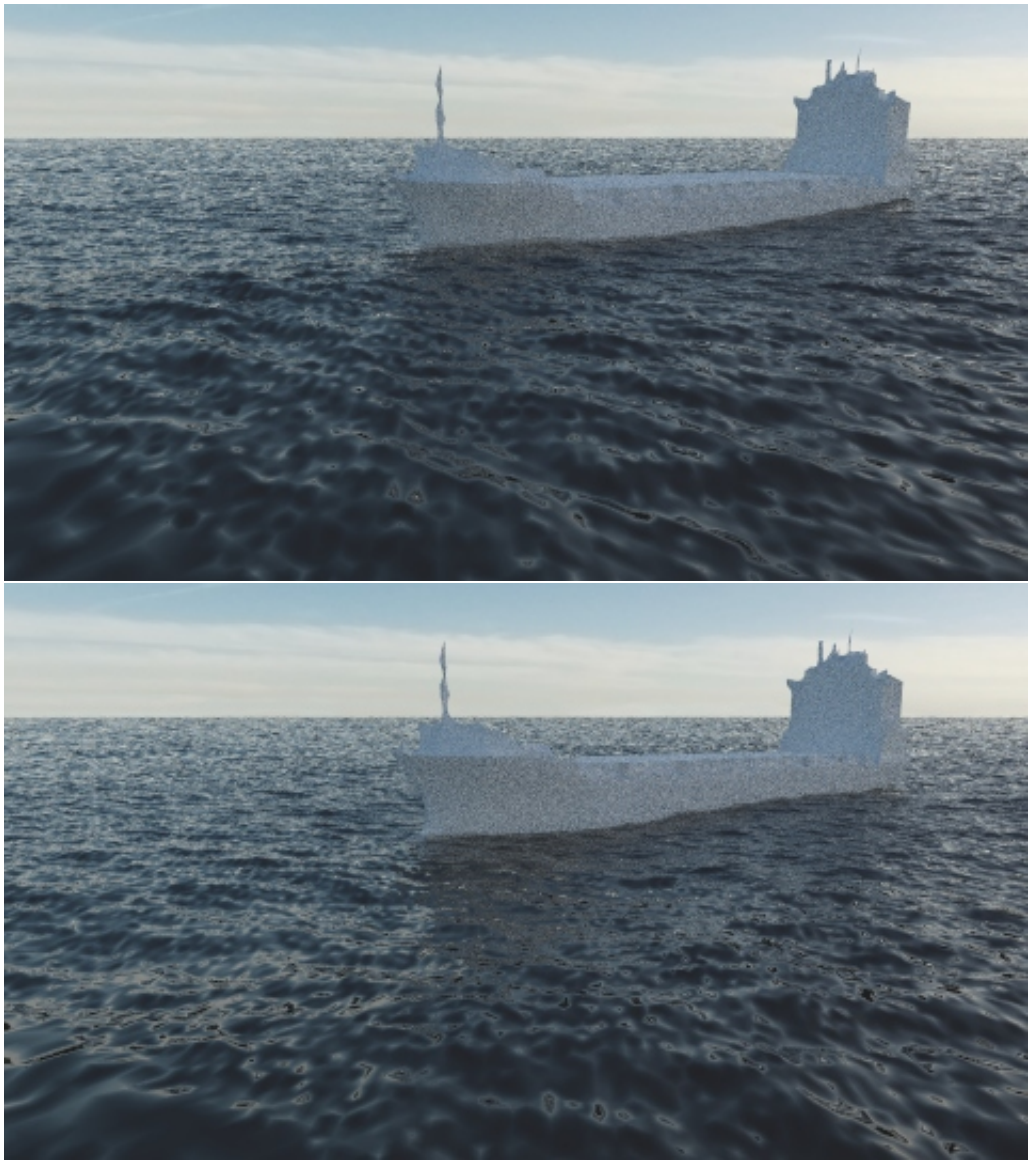

Figure 3.8: The ship on a mild ocean surface with no PID controller and no angular velocity dampening. Top: Frame 50. Bottom: Frame 170.

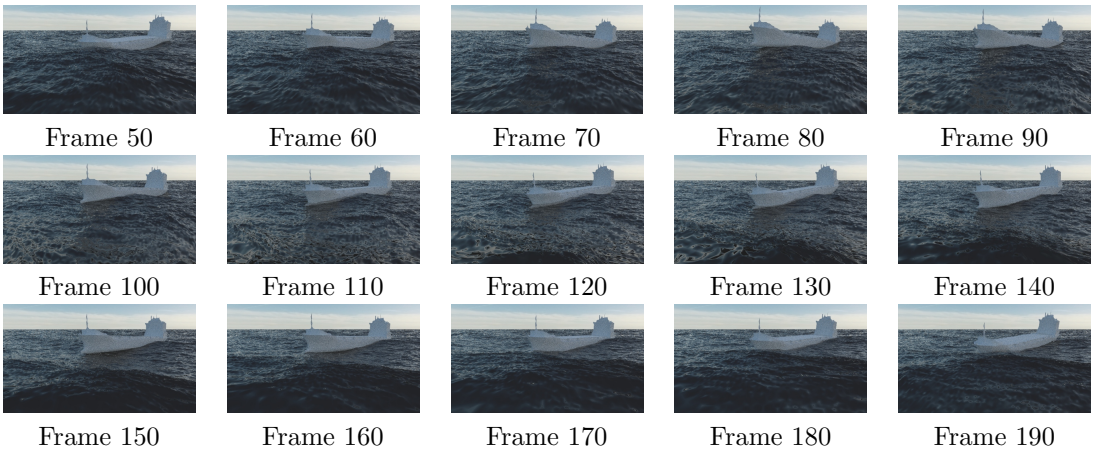

Figure 3.9: A ship with no angular velocity dampening or PID controller floating on large ocean waves.

is simply not strong enough to make a clear change at this scale, but rotational dampening terms have a much larger effect.

3.3.3 Large Ocean Waves

Simulating motion on large ocean waves with no additional dampening terms or PID controller yields one of the most visually appealing simulations explored. The ship rides up and down the waves, bobbing in response to them. At moments the ship appears to list to one side, but later corrects itself and returned bobbing. This dynamic rocking and bobbing motion is seen in figure 3.9. This is a demonstration of how lowering the center of mass can ensure that the ship stays upright. The ship slowly moved horizontally in response to the waves.

Adding the PID controller to this simulation causes the ship's center of mass to bob up and down less, but translate horizontally with a more gradual motion. The boat also rocks with more strength than when the simulation is run without this PID controller. Similar to results found when simulating the torus shape with a PID controller, this is most likely because reduced bobbing causes increased submersion and water surface contact, which lead to greater rotational forces.

Applying angular velocity dampening causes the ship to react less actively with the water. The ship doesn't bob as strongly, and the ship barely rocks with the waves. This is the effect of over-dampening that subsection 3.3.2 discusses is more apparent with large waves.

Rigid body simulations on large waves with a PID controller and angular velocity dampening

Frame 1 Frame 50 Frame 200 Frame 300 Frame 350

Figure 3.10: The PID controller causes the ship to accelerate slowly before rapidly causing the ship to exit the view of the camera.

reduces the rotational energy of the ship even further. The ship gently rides over the waves, but resists almost all rocking motions.

3.4 PID Animation

The Mearsk Arun model is animated linearly over a distance of 400 m using a PID controller with $\kappa_p = 100$, $\kappa_i = 200$, and $\kappa_d = 150$. Every frame, the desired position of the ship inside the PID controller will be updated linearly such that at $frame = 0$ the ship's intended location is $p_{cm_x} = -200$, $p_{cm_y} = 3.0$, $p_{cm_z} = 0$ and at $frame = 500$ the ship's intended location is $p_{cm_x} = 200$, $p_{cm_y} = 3.0$, $p_{cm_z} = 0$. A 500 frame animation is rendered at 20 fps. The motion generated from this process allows the ship to venture towards a target destination, and react to the water as it moves. This method alone does not handle ship turning very well because the PID controller only exerts force on the center of mass, and not the centroids. Force exerted on individual centroids is necessary to contribute to a rigid body's torque and rotation. The ship appears to naturally follow the intended path, which demonstrates the versatility of combining a PID controller with this rigid body simulation.

The animation resulting from applying force via a PID controller successfully demonstrates a method for moving a ship linearly while it reacts to the water. Under the current PID configuration, the acceptable error is too high to accelerate the ship quickly from a non-moving state. This causes the target position to migrate away from the ship while following the intended linear path while forces build up. As the target position moves further from the ship, and the error the PID controller is determined to correct gets larger, the ship picks up more force than is necessary to accelerate this ship to the target. This is demonstrated in mechanical PID controllers as overshoot. This causes the ship to move faster over a shorter period of time, which makes the motion look unnatural. This initial delay in acceleration followed by rapid acceleration is seen from overhead in figure 3.10. A

solution to this problem is to increase κ_d, therefore controlling the velocity of the ship more. Another solution is to increase κ_p and decrease κ_i. This will cause the PID controller to apply more initial force to the ship, and less of a delayed response as the integral force builds up. Another solution to this problem is to start the ship with the velocity used as PID controller input. This will ensure that the ship doesn't need to overcome it's inertia and lag behind the target position.

3.5 Finding Equilibrium

As mentioned in section 3.3.1, when no dampening is added to the system, energy is conserved forever. If the ship is dropped from a height that is too high, gravity will accelerate the ship and the inertia will carry the ship low into the water. The force of the water then pushes the ship back out and into the air, creating a continuous bobbing motion as gravity then pulls it back down. Similarly, if the boat's initial position is too low in the water, then a portion of the ship will be pushed out of the water, and gravity will pull the ship back into the water, and this process will repeat indefinitely. Between the highest point in this bobbing motion and the lowest point, there exists a position of equilibrium where the ship does not bob significantly, but instead rests gently on the surface of the water. This point of equilibrium can be found through 2 methods.

The first method to quickly approximate the height at which the ship must begin the simulation to find stasis is to average the location of the ship at every frame for a period of motion defined by the peak of the bobbing motion and the trough. If the simulation starts and the ship immediately begins moving upward, the position of the ship is averaged at every frame, until the ship begins to move back downward. If the ship immediately begins to move downward, the position of the ship is averaged at every frame until the ship begins to move back upward. The average of these points may not indicate precisely the position at which the ship is stationary on the water, but this methods finds a close approximation that greatly reduces the bobbing motion.

The second method to quickly approximate the height at which the ship is stationary on the water is to apply a PID controller with $\kappa_p = 0$ and $\kappa_i = 0$. The desired velocity is 0 kph in every direction and κ_d is very high relative to the ship's mass. κ_d needs to be high enough to overcome the amount of inertia that the ship has as it moves in and out of the water. With these input parameters the ship begins the simulation bobbing, but each successive bounce is lower to the water and closer to the point of equalibirum. After a few bounces, the ship finds an approximate point of rest on the

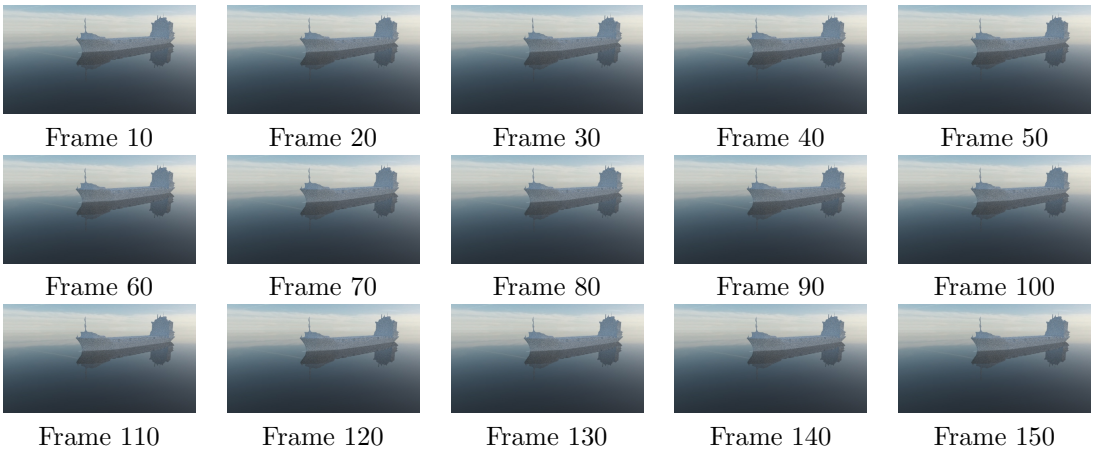

Figure 3.11: A ship that is in a state of relative rest on the surface of flat water.

water's surface.

The results from assigning the initial position of the center of mass to the point of equilibrium through averaging a period of center of mass positions can be seen in figure 3.11. The ship represented by this figure has no additional dampening terms applied to it. Comparing these results to the ones found in figure 3.6 described in section 3.3.1 makes the effect of this resting point quickly apparent. The motion resulting from the simulation where the ship's initial position is the point of equilibrium contains much less bobbing and rocking than the simulation where the ship is placed too high above the water as it is in all the simulations observed in section 3.3.1.

Figure 3.12 demonstrates the results of applying the same averaging technique to a ship with no additional dampening that is on mild ocean waves. Unlike in the flat water example, additional waves enact force that then ship reacts to. The ship rolls and pitches, and slides across the surface of the water. These motions are not seen in the simulation where the ship is resting on flat water. These motions are also less noticeable when the ship is bobbing up and down, as it is in the mild ocean water results seen in section 3.3.2.

3.6 The Effects of Weight Painting

One of the factors that makes this tool so art directable is the ability to paint weights onto the vertices of the ship. Changing the mass of the ship has a dramatic effect on the resulting simulated motion. While the painting process does not require precision, as many ships will float

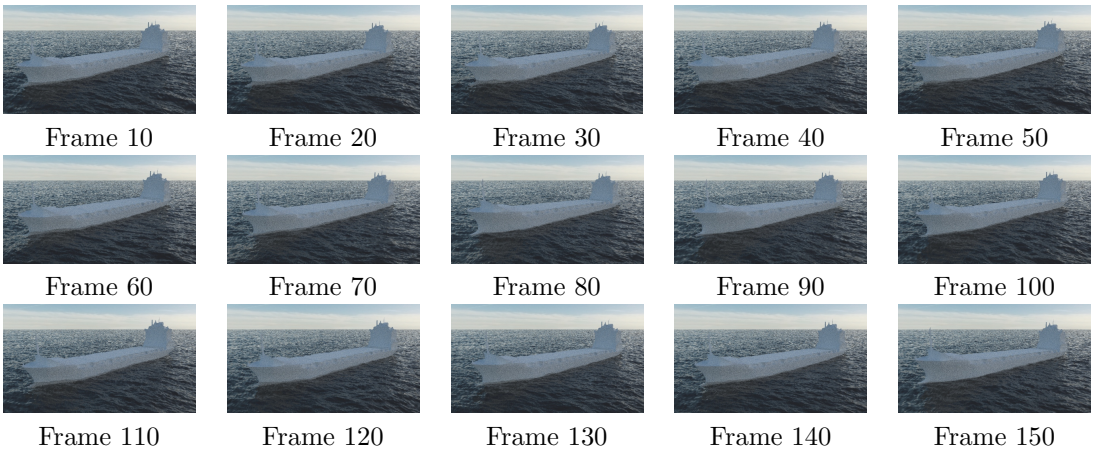

Figure 3.12: A ship that is in a state of relative rest on the surface of mild waves.

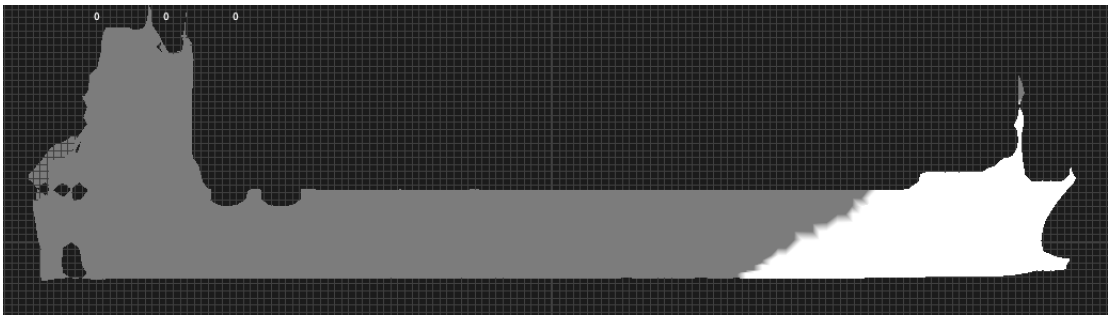

Figure 3.13: The distribution of mass used to illustrate the effectiveness of painted mass distribution.

well with minor imbalances in weight, the painting tool may be used to distribute weight in a fashion that may greatly change the way a ship floats in the water.

Below is an example of a ship that has been painted in a way such that most of the mass is located in the front of the ship as shown in figure 3.13. Simulating with this distribution of weights results in the ship nose sinking into the water, while the back of the ship lifts up out of the water. After some time, the back of the ship will drop back into the water, before once again bobbing back out.

Comparing this motion to the ship floating on mild ocean waves, at a point of rest on the water as described in section 3.5 highlights the dramatic effect of the weight on the ship. The ship that is painted with optimal upright flotation in mind shows no such nose-dive motion, and the ship maintains a steady upright position on the surface of the waves. However, when the ship is painted with a heavy nose, the ship shows an obvious nose-down bias. A comparison between a ship that

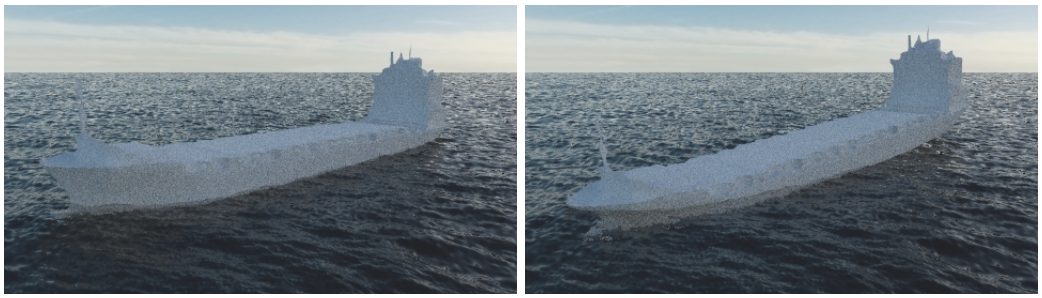

Figure 3.14: An illustration of how dramatic the effects of painting mass onto the ship can be. Top: Mass painted as shown in figure 3.5. Bottom: Mass painted as shown in figure 3.13. Both images taken from frame 57.

has been painted with flotation in mind, and a ship that has been poorly painted as seen in figure 3.13 is shown in figure 3.14.

3.7 Art Directability

The methods outlined in this thesis are demonstrated through the project results in this chapter as a tool for creating physically-based animations. The collection of input parameters available to the user offer art directability. Fluid density is adjusted to control the overall strength of the hydrostatic forces. The PID controller and angular velocity dampening offer a means by which the overall motion of the ship can be controlled beyond the forces of the waves. Painting mass onto the ship has been shown in section 3.6 to have a large effect on the flotation of the ship. The large effect that each of these parameters has on the ship motion, and the ease in which they can be adjusted by the user demonstrates the accessibility and flexibility of this art tool's capabilities.

To further demonstrate the breadth of the tool's art directability, the simulation of the ship on large waves with no dampening outlined in section 3.3.3 is improved. Originally, the fluid density was $1020kg/m^3$. Lowering the fluid density to $900kg/m^3$ reduces the amount of force that the waves enact on the ship. Applying a PID controller with a desired velocity of 0 kph in every direction, $\kappa_p = 0$, $\kappa_i = 0$, and $\kappa_d = 1000$ reduces the speed and energy in which the ship bounces out of the water. Applying angular velocity dampening of a factor of 0.0001 reduces the amount of rotation the ship engages in, but not to a degree in which the ship stops noticeably rotating altogether. These factors create ship motion on large waves that are much more realistic than the motion of the ship when no additional dampening parameters are used. The motion in figure 3.15 demonstrates how

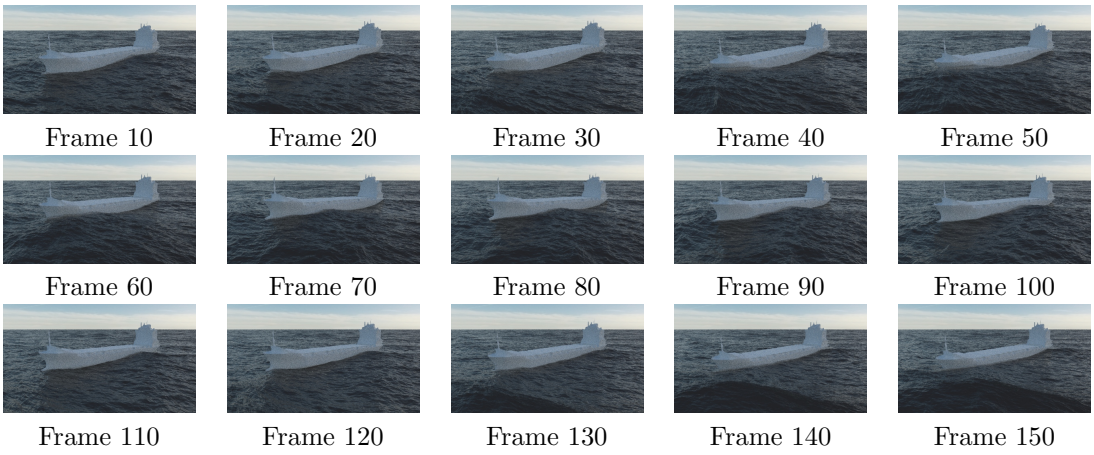

Figure 3.15: Realistic ship motion demonstrated through the tool's available parameters.

realistic the motion generated from this tool can be with the right set of art directed parameters.

Chapter 4

Conclusions and Discussion

The tool described for creating rigid body dynamics on ship hulls using hydrostatic forces is a successful at designing interaction scenarios between oceans and boats. The algorithm can be optimized to be very fast when implemented with appropriate data organization structures. Alexandra L. Zheleznyakova states in his paper that the methods described can run in realtime [13], even though for the implementation of this thesis a non-realtime environment is chosen.

4.1 Strengths

This method for creating ship motion on an ocean surface proves to be stable, and customizable with a useful level of art directability. Users can model ships of arbitrary design and distribute mass onto the vertices in a manner that favors the ship design chosen. A PID controller allows the ability to move the ship while conserving dynamic ocean movement. Additional dampening terms may be uniquely adjusted to achieve the preferred outcome. Finally, ocean surfaces themselves are widely customizable, with a variety of real world conditions being possible to replicate across many available wave spectra. Due to this, the tool is not only a useful art tool, it is but potentially useful for the purposes of ship/ocean interaction prediction.

Often, motion generated with just a few parameters produces aesthetically pleasing results. Drag is a tool the system can employ to enhance the motion of the ship, but is not necessary in most cases. Ship motion generated with no dampening parameters or outside forces added often looks good without the need for further refinement.

The algorithm is fast and reliable, is agnostic to the shape of the ship being simulated, and requires minimal pre-processing. It can be adapted into a variety of environments including games engines and animation software such as Maya and Houdini. Overall, the algorithm is quite robust.

4.2 Drawbacks

The stability of the described method of rigid body ship motion is dependant upon triangle size of the input geometry. Ideally, proxy geometry would be unnecessary in a buoyancy calculation. A possible solution to this includes dynamically subdividing submerged triangles to determine the forces acting on the model with higher accuracy. This would allow the program to produce better results with less optimal input geometry.

It can be difficult controlling the dampening parameters. As seen in chapter 3, angular velocity dampening coefficients as small as 0.001 can have dramatic effects on the movements of the ship, often times making results look stiff. Other methods for dampening the rotational energy of the system should be considered. Conversely, PID controllers often times may not be strong enough to influence a ship without very large input coefficients. This makes estimating productive values for these coefficients difficult.

4.3 Directions for Further Research

The methods of rigid body motion described in this thesis can be combined with existing methods of altering the height field, such as the generation of a wake. These adjustments to the height field may affect the ship movement, which may further alter the height field, creating a feedback loop. The ship motions and wake patterns generated by this feedback loop are worth researching further.

The methods outlined in this thesis describe buoyancy as a force generated proportional to the submerged surface area of the ship and its depth. In the real world, buoyancy is proportional to the mass of the fluid that an object displaces. If a real object is too heavy to float on a fluid, it will continue to sink until it reaches the bottom of the container. In the presented methods, hydrostatic force is proportional to the submerged depth of an object. A sinking rigid body may artificially build up force once fully submerged and stop sinking at a depth where it reaches an equilibrium. A

suggestion to converge on a closer approximation to real buoyancy forces is to replace the polygonal model with a levelset. By sampling grid points in a levelset instead of centroids, and determining whether each point is above or below the surface of the water, an estimation of the fluid mass proportional to the submerged volume of an object may be obtained. This submerged volume may then be used as a factor of the hydrostatic force equation (2.19) in the place of surface area, deriving more accurate results.

The method of directly dampening a rigid body's angular velocity is an effective means of quickly stabilizing the motion of the body, but it does not mirror realistic drag force phenomena. In real-world ship-water interactions of partially submerged bodies, a ship meets a drag force from both the air and the water, however these forces differ from each other in strength. As part of further research, drag forces generated by the water may be applied to specifically centroids that are submerged under the height field. Similarly, if the general direction of the waves can be derived through a process of finding the height field's rate of change [12], an approximation of underwater current forces may be applied.

The PID controller may be adapted to accept vector κ values. These modified κ values can control the amount of positional error correction and velocity error correction per axis independently. This would allow the ship to continue bobbing up and down in the water completely unhindered by the target position on the vertical axis, and still allow for a method of target-based animation on the horizontal plane. A PID controller may be attached to the rotation vector to help the ship remain aligned with the direction it's traveling in.

Bibliography

[1] Marine accident report ss edmund fitzgerald sinking in lake superior. Technical report.

[2] Alex Garland. Devs, 2020.

[3] Christopher J. Horvath. Empirical directional wave spectra for computer graphics. In *Proceedings of the 2015 Symposium on Digital Production*, DigiPro '15, page 29–39, New York, NY, USA, 2015. Association for Computing Machinery.

[4] D. House and J.C. Keyser. *Foundations of Physically Based Modeling and Animation*. CRC Press, Taylor & Francis Group, 2017.

[5] Timo Kellomäki. Rigid body interaction for large-scale real-time water simulation. *Int. J. Comput. Games Technol.*, 2014, January 2015.

[6] Robert I McLachlan and G Reinout W Quispel. Geometric integrators for ODEs. *Journal of Physics A: Mathematical and General*, 39(19):5251–5285, apr 2006.

[7] Roozbeh Panahi, Mehdi Shafieefar, and Ali Ghasemi. A new method for calibration of unidirectional double-peak spectra. *Journal of Marine Science and Technology*, (167-178):18, 2016.

[8] Jerry Tessendorf. Simulating ocean water. 1999.

[9] Jerry Tessendorf. Interactive water surfaces. *Game Programming Gems*, 4(265-274):8, 2004.

[10] Jerry Tessendorf. Motion blur algorithm for clipped triangle rendering. 2004.

[11] Jerry Tessendorf. *Gilligan: A Prototype Framework for Simulating and Rendering Maritime Environments*, 2017.

[12] Charles G Torre. Foundations of wave phenomena. 2014.

[13] Alexandra L. Zheleznyakova. Physically-based method for real-time modelling of ship motion in irregular waves. *Ocean Engineering*, 195:106686, 2020.