2014

Three-dimensional model of a plasma railgun using smoothed particle hydrodynamics

Lloyd M. Jackson

THREE-DIMENSIONAL MODEL OF A PLASMA RAILGUN USING SMOOTHED PARTICLE HYDRODYNAMICS

by

LLOYD M. JACKSON

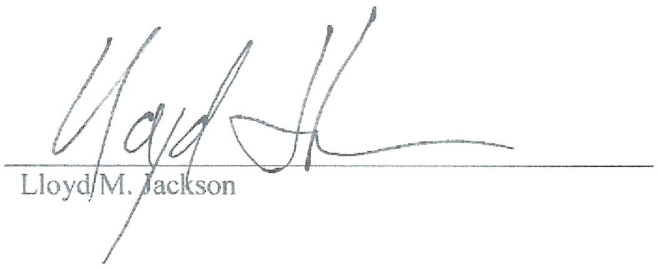
A THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Engineering
in
The Department of Mechanical and Aerospace Engineering
to
The School of Graduate Studies
of
The University of Alabama in Huntsville

HUNTSVILLE, ALABAMA

2014

In presenting this thesis in partial fulfillment of the requirements for a master's degree from The University of Alabama in Huntsville, I agree that the Library of this University shall make it freely available for inspection. I further agree that the permission for extensive copying for scholarly purposes may be granted by my advisor or, in his/her absence, by the Chair of the Department or the Dean of the School of Graduate Studies. It is also understood that due recognition shall be given to me and to The University of Alabama in Huntsville in any scholarly use which may be made of any material in this thesis.



Lloyd M. Jackson

4/3/2014

(date)

THESIS APPROVAL FORM

Submitted by Lloyd M. Jackson in partial fulfillment of the requirements for the degree of Master of Science in Engineering in Mechanical Engineering and accepted on behalf of the Faculty of the School of Graduate Studies by the thesis committee.

We, the undersigned members of the Graduate Faculty of the University of Alabama in Huntsville, certify that we have advised and/or supervised the candidate of the work described in this thesis. We further certify that we have reviewed the thesis manuscript and approve it in partial fulfillment of the requirements for the degree of Master of Science in Engineering in Mechanical Engineering.

Dr. Jason Cassibry	(Date)	Committee Chair
Dr. Robert Frederick	(Date)	
Dr. Kunning Xu	(Date)	
Dr. Keith Hollingsworth	(Date)	Department Chair
Dr. Shankar Mahalingam	(Date)	College Dean
Dr. David Berkowitz	(Date)	Graduate Dean

ABSTRACT

School of Graduate Studies
The University of Alabama in Huntsville

Degree Masters of Science in Engineering College/Dept. Mechanical and Aerospace Engineering

Name of Candidate Lloyd M. Jackson

Title Three-Dimensional Model of a Plasma Railgun Using Smoothed Particle Hydrodynamics Methods

Pulsed plasma accelerators are utilized for in-space propulsion and drivers for inertial fusion concepts. Theoretical models are necessary to assist in diagnostic analysis and for developing scaling laws. SPFMax is a new 3D code which uses smoothed particle hydrodynamics (SPH) to simulate fluid flow, and has been designed specifically for modeling plasmas produced by these pulsed devices. A set of gasdynamic test cases were established and utilized to verify the accuracy of SPFMax for modeling the gas dynamics in a railgun. The free expansion confirmed that the gas expands supersonically without exceeding the predicted maximum value. With the square wave test, SPFMax advected the waves with floating point accuracy. Shocks, expansion waves, and contact surfaces were resolved in both 1D and 3D tests with a relatively low number of particles.

Abstract Approval: Committee Chair

Dr. Jason Cassibry

Department Chair

Dr. Keith Hollingsworth

Graduate Dean

Dr. David Berkowitz

ACKNOWLEDGMENTS

I would like to thank the Missile Defense Agency for funding my assistantship with the Propulsion Research Center. I would like to thank my advisor, Dr. Jason Cassibry, for giving me this opportunity and allowing me to fulfill my dream of working in the aerospace industry. I would also like to thank him for his hard work with getting SPFMax running and his guidance throughout these past few years. Dr. Doug Witherspoon provided me with lots of information and documents during his busy schedule. I would like to thank several colleagues at the Propulsion Research Center, past and present, who helped me overcome some of my weaknesses in MATLAB and not only pass, but excel, in my classes.

Thanks to the many teachers, family members, friends, and other advisors who have encouraged me, prepared me, and carried me up to this point. I could not have come so far without you. And thanks to my loving wife, Erin, for her sustaining support, love, and kindness. She has sacrificed so much for my dreams, and I cannot possibly thank her enough for all she has done for me. Erin, I love you very much. To any others I have not mentioned specifically, thank you. Your influence has made me what I am today.

TABLE OF CONTENTS

ABSTRACT iv

LIST OF FIGURES viii

LIST OF TABLES xi

CHAPTER 1 INTRODUCTON 1

1.1 Overview of Railguns 2

1.2 Modeling of Plasma Railguns 5

1.2.1 2D Modeling Codes 6

1.2.2 3D Modeling Codes 9

1.3 Problem Description 12

CHAPTER 2 NUMERICAL MODELS OF SPFMAX 14

2.1 Coupled Circuit and Lumped Element Model 15

2.2 Governing Equations 17

2.2.1 Fluid Model 17

2.3 Smoothed Particle Hydrodynamics and Electromagnetics Method 19

2.3.1 Smoothed Particle Hydrodynamics 19

2.3.2 SPEM and the Circuit Model 22

2.4 SPFMax Description 24

2.4.1 Geometry 26

2.4.2 SPH Solver 28

2.5 Modeling Procedure 36

CHAPTER 3 RESULTS 38

3.1 1D Lumped Element Circuit Modeling 39

3.2 SPFMax Tests 47

3.2.1 Gas Expansion Test 48

3.2.2 Advection Test 50

3.2.3 Shock Tube Test 52

CHAPTER 4 CONCLUSIONS 61

4.1 Conclusions 61

4.2 Future Work 63

REFERENCES 64

LIST OF FIGURES

Figure 1.1: Three plasma jets produced by MiniRailguns designed by HyperV Technologies Corp. converge on each other in a successful calibration test [1]. 1

Figure 1.2: A simple plasma railgun model illustrates the physics of a plasma armature while firing [8]. 4

Figure 1.3: Models of the same physical geometry show the visual differences between meshed (a) and particle (b) modeling methods [11]. 6

Figure 2.1: Demonstration of material impacts using finite element and meshless numerical methods shows differing qualitative results [37]. 14

Figure 2.2: A 2D representation of a typical SPH particle calculation shows how the kernel function incrementally affects neighboring particles located within the smoothing length [40]. 21

Figure 2.3 The process flow of the SPFMax code up to the SPH solver subroutine indicates the preliminary steps that the code goes through in order to calculate the physical interactions. between particles. 25

Figure 2.4 The flow diagram of the SPFMax subroutine sph_solver.m illustrates the construction and primary processes that take place during a simulation. 29

Figure 3.1: The detailed schematic of the HyperV railgun serves as a basis upon which the geometry of the SPFMax model is based [1]. 38

Figure 3.2: Basic 1D plots will tend to take on an overdamped waveform ($\alpha \gg 1$) or oscillate ($\alpha \ll 1, \beta \ll 1$) depending on the scaling factor [8]. 41

Figure 3.3: The graph of the position of the gas over time shows slow progression that grows exponentially. 42
Figure 3.4: The current plot shows heavy oscillations dampening out over time. 43
Figure 3.5: The velocity plot shows oscillation slowly dampening to zero acceleration. 43
Figure 3.6: The 2D slug model performed using MACH2 shows both a smaller peak current and maximum velocity, likely due to viscous effects [1]. 44
Figure 3.7: During laboratory tests, HyperV measured a peak current at 419 kA. Peak performance requires an approximate current peak of around 600 kA [1]. 45
Figure 3.8: Using a photodiode, HyperV was able to measure a plasma jet velocity of 44 km/s during physical testing [1]. 46
Figure 3.9: The low-resolution geometry of a basic bore model contains a block of gas for the gas expansion test. 49
Figure 3.10: Gas expansion test in the railgun at low, medium, and high resolutions involving 135, 450, and 1,503 particles. For reference, the soundspeed is 20,410 m/s, and the maximum speed predicted by theory is 60,925 m/s. 50
Figure 3.11: A thin-walled cube houses argon gas for the advection test. 51
Figure 3.12: The advection test produces a density square wave that does not change as it propagates. 52
Figure 3.13: Sections of a shock tube show how pressure levels change from the front of the expansion wave to the area in front of the shock wave [45]. 53
Figure 3.14: The geometry of the low resolution shock tube consists of a square bore, capped on both ends, containing two blocks of argon gas at 30,000 K and 3,000 K. 56

Figure 3.15: 1D results of the SPFMax shock tube show improved accuracy against the analytic solution as the resolution increases. 59
Figure 3.16: 3D resolutions of the shock tube simulation show some numerical errors not found in the 1D runs, likely due to a few particles escaping the bore through a small space between the cap and walls. 60

LIST OF TABLES

Table 2.1: Performed fluid model and railgun simulations. 37

Table 3.1: HyperV Initial Conditions [1]..... 39

Table 3.2: 1D shock tube error 57

Table 3.3: 3D shock tube error 58

CHAPTER 1 INTRODUCTON

Figure 1.1: Three plasma jets produced by MiniRailguns designed by HyperV Technologies Corp. converge on each other in a successful calibration test [1].

HyperV Technologies Corp. is a company based out of Chantilly, Virginia, which develops technology that utilizes hypervelocity physics effects. Currently, HyperV's scientists and engineers are primarily working on a project called the Plasma Liner Experiment (PLX) in which several controlled plasma jets form a liner that converges and implodes on a small, magnetized fuel mass to create a fusion reaction [2]. PLX hopes to be able harness this energy for terrestrial power and other endeavors, including pulsed fusion propulsion for space travel.

The plasma jets involved in the preliminary experiments are produced by 30 individual, small, parallel-plate railguns [2]. The railguns are being developed in several stages. First, 1-centimeter square-bore MiniRailguns were built to test the convergence of six plasma jets in order to better understand the dynamics of plasma collision and demonstrate synchronous function of the railguns [1]. After the scaled down experiment was able to show desirable performance of the railguns and produce preliminary plasmoid data, HyperV scientists have started to build and refine full-scale 2.5 centimeter square-bore railguns [3].

The energy involved in firing a jet of plasma, a highly energetic state of matter that can only be controlled using magnetic fields, more than 50 km/s [3] is immense. In fact, the blackbody radiation of plasma is so immense that tiny pieces of the metal electrodes are stripped away [4] each time the railgun is fired. Because of this extreme energy, every aspect of the railgun design must be streamlined for maximum efficiency and durability. Therefore, there is a real need to understand the plasma dynamics involved in this and similar experiments. HyperV's railgun model represents one of the latest and most innovative uses of railgun technology.

1.1 Overview of Railguns

The electromagnetic cannon was invented in the early 20th century by a Norwegian physicist by the name of Kristian Olaf Bernhard Birkeland [5]. As with many great inventions, electromagnetic launch (EML) – a broader term for railguns and their various uses aside from artillery – came about by accident when Birkeland was developing a current breaker which utilized magnetic induction [5]. The solenoid used in

the experiment attracted and accelerated small, nearby, iron objects and propelled them through like projectiles, leading Birkeland to develop a solenoid gun [5].

Since then, EML has seen a lot of development and utilization. From advanced aerodynamics [6] to micro-satellite primary boost concepts [7], EML is proving to be a very promising advanced technology. Since the primary focus of this work is railgun development, discussion will focus on railgun design and function over other types of EML such as coaxial guns and linear pinches, despite potential applications this work may have to those concepts as well. Railguns or rail accelerators are distinct to other forms of EML in that they work using parallel plate electrodes which carry current to and from the conductive armature.

The armature is designed to push out the payload (and can function as the payload itself, such as in the case of pulsed plasma thrusters [8]), and it also simultaneously completes the circuit for the railgun. Many of the differences in design of railguns are found in the armature [4]. The railguns that engineers from HyperV have designed are plasma railguns. However, instead of boosting projectiles, they are simply designed to eject plasma at extremely high speeds (some plasma railguns are designed to fire projectiles, but testing has shown that solid armatures are more effective due to lower electrical resistances found in metals [4]). In a plasma railgun, a voltage is applied so that current can flow through the flat rail electrodes. Ionized gas is injected into the bore. The free electrons of this gas accept the current flowing through the cathode rail and bridge the gap to the anode rail, completing the circuit. As current flows through any conductive material, magnetic fields develop and encompass the conducting object. When magnetic

fields and electric current density interact, a Lorentz force is produced perpendicular to the interaction [9]. Mathematically, Lorentz forces are calculated using equation 1.1 [8].

$$\mathbf{F} = \mathbf{j} \times \mathbf{B} \tag{1.1}$$

F, **j**, and **B** represent the force vector, the current density vector, and the magnetic field vector respectively. As current increases, the Lorentz forces acting on the armature accelerate the plasma particles axially out of the rail bore at very high velocities. This entire process is better visualized in Figure 1.2 below.

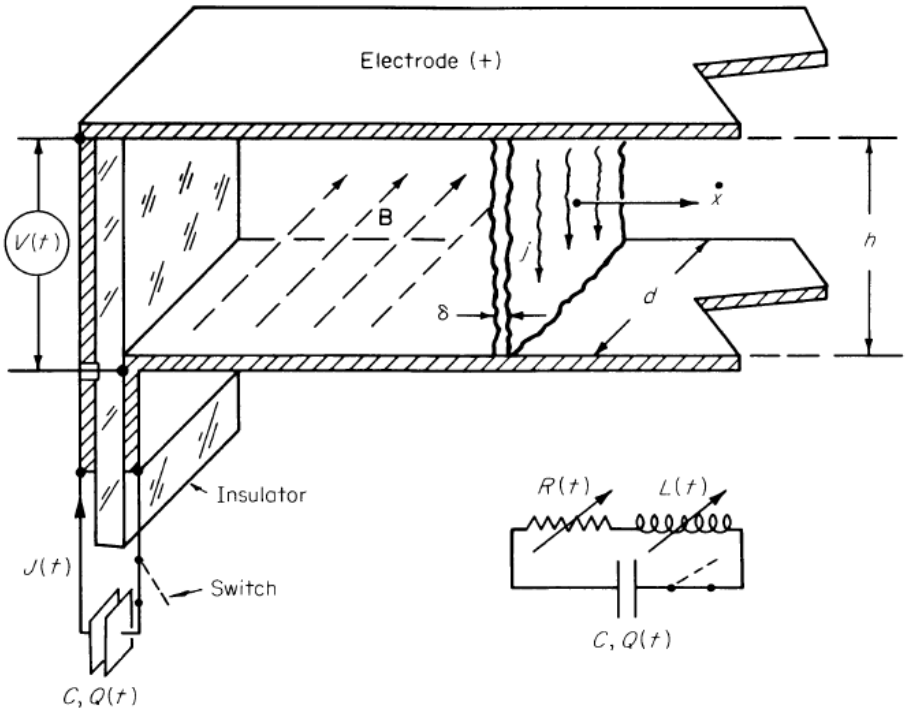


Figure 1.2: A simple plasma railgun model illustrates the physics of a plasma armature while firing [8].

For the most part, railguns are simple machines. However, the energy used to achieve the desired effects of most railgun designs is extremely high, and firing railguns repeatedly can quickly erode and destroy the rail and insulator materials through a

process called gauging [4]. This is especially true of projectile railguns that use metal armatures. Because of these extremely destructive forces on the guns themselves and additional anomalous behavior, such as secondary shocks [10] and plasma sheet canting [9], much effort has been taken to more thoroughly understand plasma interaction. Understanding and predicting the behavior of plasma will lead to both making railguns more efficient in energy usage and more impervious to wear. Computational fluid dynamics models have been the primary methods used to understand railgun dynamics. Section 1.2 discusses common features and design aspects of several modeling techniques.

1.2 Modeling of Plasma Railguns

Computational fluid dynamics problems are generally solved using two basic methods: meshed and unmeshed [11]. A mesh is a group of imaginary nodes connected to one another over an area of a solid, liquid, or gas [11]. The nodes take on material property values at their locations and can be made to travel with a fluid or stay in position [11]. When the nodes stay in position with flow, they are said to be Eulerian [11]. The nodes are said to be Lagrangian if they travel with the fluid. When moving with flow, meshes must be remade after short periods of time due to distortions in the grid [11]. If they stay in the same position, meshes cannot predict material-fluid interaction very easily [11]. Additionally, free boundaries and irregular geometry are difficult to model using meshed techniques [11].

The other basic method of solving computational fluid dynamics problems involves the use of particles over meshes [11]. The particles studied are not actual particles of the fluid; rather they are simply points that represent the domain of the

problem that needs to be solved [11]. Based on the length scale, mesh-free particle methods can be divided into the classes of microscopic, mesoscopic, and macroscopic scales, which means they can be applied towards almost any scale of problem from very finely detailed to rough estimates [11]. Particles do not require the use of meshing, and therefore do not have to be re-meshed after distortion [11]. In that same vein, particles are also much friendlier with complex and irregular geometries because they naturally flow like groups of molecules within a fluid [11]. Particle methods can also be made to handle conservation laws perfectly when they take on the material properties of the fluid being studied [11]. The visual distinctions between particle and mesh modeling can be seen in Figure 1.3.

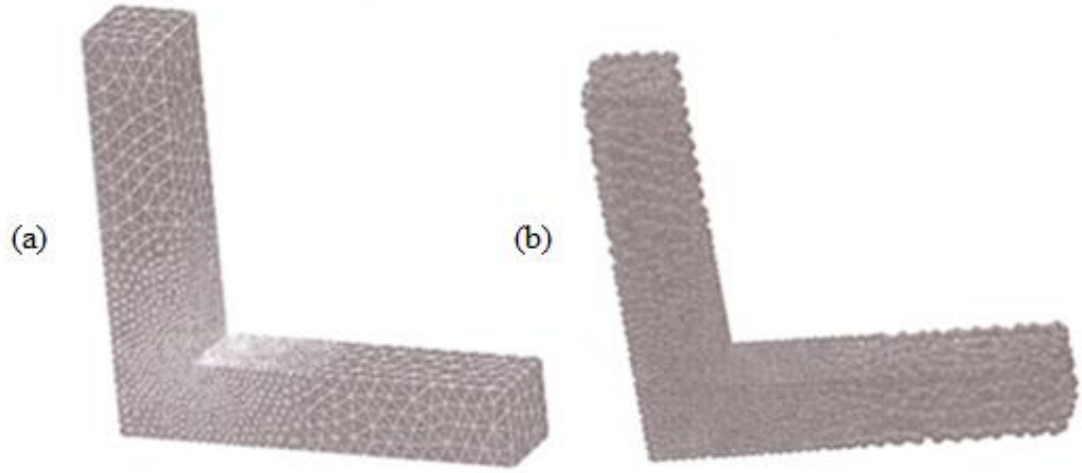


Figure 1.3: Models of the same physical geometry show the visual differences between meshed (a) and particle (b) modeling methods [11].

1.2.1 2D Modeling Codes

Modeling railgun behavior has been a high priority for several decades. Until recently, however, it has been difficult to produce very detailed models of railgun

behaviors without considerable computing power. Many early one-dimensional (1D) and two-dimensional (2D) models analyzed specific physical attributes in order to better explain observed behaviors. For example, in 1989, Rolader et al. focused primarily on the transient temperature properties of plasma armatures in three stages of pulse activation. In the study, several assumptions were made in order to calculate the 1D model [12], and therefore, tended to be drastically oversimplified and error-prone.

With the advent of greater computing power came several codes designed to produce current models for railgun technology, employing a number of numerical strategies to make calculations. Using early current models, forces on railgun payloads could be modeled with reasonable accuracy. LAPLAC, an early 2D code, has been used to calculate Laplacian mathematical equations, using least-squares minimization, in effort to study current distribution for several railgun barrel geometries [13]. LAPLAC is strictly designed to model current while plasma physics is completely neglected. Without the ability to predict plasma dynamics, railgun performance efficiencies could not really be improved upon.

Finite element (FE) methods – methods using nodal localization and interpolation on a fixed grid to calculate equations of state on objects [14] – were heavily employed for structural and thermal testing on railgun circuits [15], bores [16], and, especially, solid armatures [17]. FE models have been able to provide a great deal of knowledge for expected performance of solid structures of a railgun. However, applying the same methods to deforming fluids is a considerably complex problem. Unlike solid structures, fluids, such as plasmas, deform and change over time, requiring continual remeshing.

P-Spice is another grid-based solver which is also primarily designed to solve for current. Based on finite difference numerical methods, P-Spice has been shown to accurately model current flow for complex circuits [18]. However, as in the case of FE modeling, plasma characteristics cannot be addressed using finite difference methods. Additional modeling methods are necessary to understand the complexities of plasma interaction with itself and electrodes.

Even more sophistication and accuracy have been achieved using mathematics based on the physics that specifically describe plasma behavior called magnetohydrodynamics, or MHD (discussed further in Chapter 2 of this work). One advantage to using MHD-centric codes is ease of incorporating additional physics with code development. Boynton and Huerta used an MHD code to simulate rail ablation, incorporating in rail drag into a previously tested model which already accounted for Ohmic heating, radiative heat transfer, ideal gas law, mass conservation, resistivity, and other physics [10]. Smaller time steps were also employed which enabled the simulation to show how secondary arcs formed during the firing of a plasma railgun. Secondary arcs are responsible for removing much of the ablated rail material. For this simulation, Boynton and Huerta assumed a uniform magnetic field, neglecting important interaction between changing magnetic fields and current.

Rhodes et al. used a program called MACH2, another MHD-based modeling code that calculates 2D MHD equations using fixed grids (much like FE methods), to calculate several variables in the firing of a pulsed plasma thruster (PPT) using MHD equations for continuity and momentum [19]. For this case study, Rhodes et al. use experimentally acquired mass flow and energy loss data to model the PPT. The results show accurately

predicted performance of the PPT, including an impulse bit measuring within 99% of the experimental results.

Although the study produced fairly accurate models that agreed well with experiments, certain aspects of the MACH2 code failed to produce a completely accurate model. Recombination rates between electrons and ions within the plume of the PPT were not accurately addressed due to an equilibrium assumption, causing inaccuracies in the electron number density. Other MACH2 models have shown that without additional development in the code, universal application to ablative MHD models fall quantitatively short to experimental results thanks in part to overestimation of electrical conductivity at the breech leading to additional magnetic flux [20] and, ultimately, inaccuracies in current sheet propagation [21].

Additional gaps in understanding plasma mechanics remain. The fixed grid aspects of MACH2 do not allow the code to analyze plasma dynamics from a Lagrangian perspective. As a result, quantitative errors can occur when plasma densities fall below the built-in hydrodynamics and joule heating threshold, creating vacuum blocks prematurely [22]. Additionally, knowledge gaps in the understanding of plasma dynamics continue to be a factor in improving the effectiveness of railgun technology itself. Crawford et al. concluded that three-dimensional (3D) MHD models would be necessary to fill some of the knowledge gaps left over from 1D and 2D codes [23].

1.2.2 3D Modeling Codes

More detailed models have led to greater detail and accuracy predicting railgun performance. Several modeling strategies employ multiple specialized codes in order to observe multi-physics interactions [24], [25], [26], [27]. Kondrashov and Keefer [27]

created a code called IVTAN designed to simulate MHD and electro-dynamic physics in 3D for a plasma armature accelerating a projectile. Capable of solving Navier-Stokes equations and Maxwell's equations simultaneously, the IVTAN code showed decreased current density in the center of the plasma armature as well as plasma flow away from the fired projectile due to the maximum Lorentz forces occurring off axis. Although showing the interaction between circuit and plasma models provided new understanding, implementation of the code centered on grid-based finite difference methods. Particle methods could have provided more perspective for several more variables.

As modeling has progressed, a number of 3D codes have become widely used. Two notable codes are MACH3, the 3D variant of MACH2, and LSP, a versatile 3D particle-in-cell (PIC) code. MACH3 is a grid-based code capable of modeling continuity for all states of matter [28]. MACH3 analyzes fluids from both Lagrangian and Eulerian perspectives, meaning that MACH3 can be used to both produce particle streamlines and spatial characteristics. Using the MACH3 code, complex geometries can be discretized into arbitrary 3D hexahedral elements and assigned boundary conditions [29]. Transient equations of state for ions and electrons are separately solved for via Jacobi iteration while convection terms are expressly solved using a Van Leer algorithm. Changes in the magnetic field are also solved for using specialized numerical methods combined with grid techniques [30]. In addition to resistive diffusion in magnetic fields, the magnetic field solver also is capable of considering the hall effect and thermal sources. MACH3 is also capable of modeling a number of resistive effects, circuit models, and material ablation [31].

The level of detail produced in MACH3 models has allowed it to be employed for several different plasma dynamic case studies, including z-pinch physics [32], Magnetoplasmadynamic (MPD) thruster concepts [31], and laser to target interactions [28]. While MACH codes are quite extensive in capability, there are also a few weaknesses inherent in the programs. Despite being able to produce streamlines of fluid particle movement, MACH3 is not a particle-centric program. MACH3, like MACH2, suffers from the same inherent weaknesses of grid-based numerical modeling methods. MACH3 is also not very accessible to the general engineering public as its development is ongoing [29].

Particle-in-cell (PIC) is a Lagrangian – Eulerian numerical method that analyzes particle interaction from the perspective of a mesh. Meshes serve as spatial references to groups of particles that would affect one another in some way, such as collisions with one another. Grid points are also used to calculate circuit models by approximating charge density over the gridspace where particles are located at a certain time step. Unlike the arbitrary grids used in MACH3 codes, gridspace is discrete in PIC codes and typically allows one particle to occupy any one gridspace at one time [33]. This feature adds a higher level of accuracy than that of MACH3 because statistical switches do not turn on and off physics models. Either particles occupy a grid space or they do not. There is no statistical threshold to consider when applying calculations over a set of gridspace. While PIC simulations can be run in one, two, or three dimensions, 3D PIC simulations have produced results that more closely resemble experimental data than similar simulations using a more simplified 2D code [34].

Large Scale Plasma (LSP) PIC code has been used to study nuclear fusion reactions produced by high energy plasma jets. LSP features two electromagnetic field algorithms designed for fast and slow moving electromagnetic particle simulations, and several algorithms for collisions or pushing between particles that reduce error and noise within the simulation. Multi-physics applications for large simulations include circuit modeling, ionization rates, multi-fluid particle collisions, energy transfer, and several more. A number of companies and research institutions have implemented LSP for plasma-related research and development including HyperV and other PLX research partners.

Research performed by the PLX group using the LSP code has focused on the convergence of high speed plasma jets ejected from plasma railguns. Results of PLX simulations indicate dense, highly pressurized plasma regions at the area of jet convergence and possible shock conditions [35], [36] which could help produce fusion reactions [2]. However, for future simulations analyzing plasma jets ejected out of railguns into an applied magnetic field, a higher grid resolution is needed. Herein lays the primary weakness of PIC codes. Higher resolutions take up valuable computational resources which are already shared between several CPU cores. Ultimately, the computing capabilities of LSP and other PIC codes overpower conventional computing power, leading to lower resolution experiments which, in turn, can produce inconclusive results.

1.3 Problem Description

The modeling of pulsed plasmas has identified several weaknesses in conventional approaches as described above. To summarize, pulsed plasmas commonly

exhibit regions of parameter space in which there is a distinct boundary between vacuum and plasma. The vacuum regions have to be modeled in many cases involving plasma devices coupled to external circuits, and the MHD assumptions are not valid. Further, pulsed systems involve compression and/or expansions often of the order of 10 to 1 or more, which benefit from modeling with adaptive methods in which physics are adequately resolved only where needed. While particle in cell methods address these issues and are considered state of the art, 3D simulations are prohibitively expensive for engineering design in which exploration of the parameter space is desired.

This research documents an alternative solution in the form of a simplified Lagrangian, mesh-less code designed at the Propulsion Research Center of the University of Alabama in Huntsville (UAH) called SPFMax. The focus of this thesis is on testing the gasdynamic portion of the code on a series of test cases relevant to experimental conditions (temperatures, densities, time and spatial scales) to be found inside a rail gun. In the following chapters, SPFMax is introduced and explained, followed by the experimental setup and results of the SPFMax fluid simulations.

CHAPTER 2 NUMERICAL MODELS OF SPFMAX

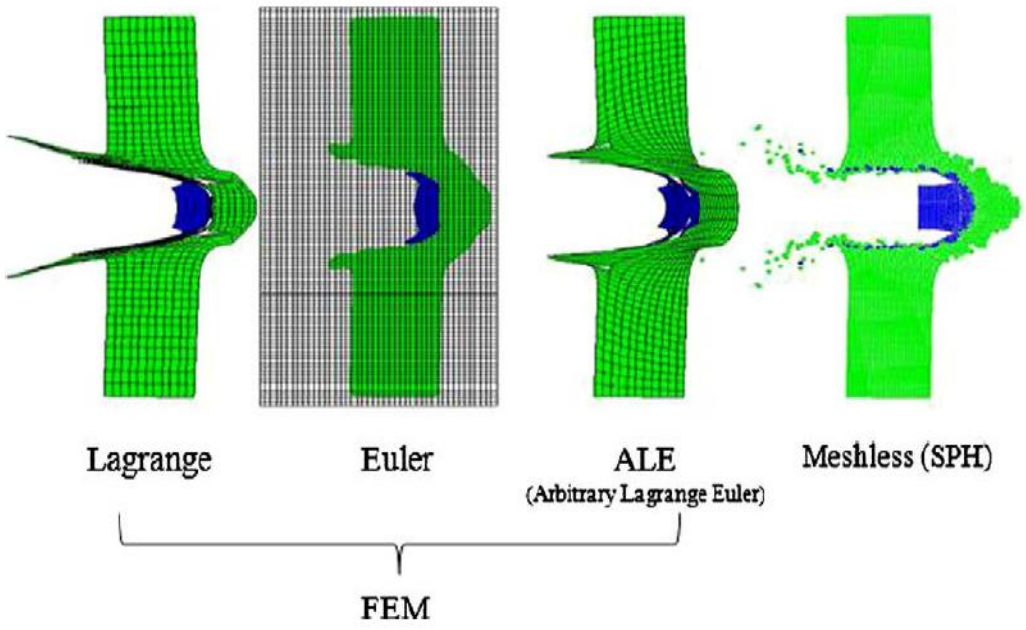


Figure 2.1: Demonstration of material impacts using finite element and meshless numerical methods shows differing qualitative results [37].

Concurrently interacting physical processes are extremely difficult to model in an open continuum with the conventional methods mentioned in Chapter 1 because the most advanced numerical methods rely on a fixed grid-space to solve coupled differential equations and, therefore, require strict boundary conditions, as opposed to an open continuum in which fluid particles can continue to move and interact. Using grids to solve particle-based problems can also lead to statistical errors like the example seen above [37]. However, a grid-less, particle-based simulation method could be used to minimize error and naturally show particle interaction from initialization until the final

designated time step of the simulation is reached. SPFMax (Smoothed Particle Fluid – Maxwell) is a grid-less, Lagrangian fluid simulation program, developed at UAH, that uses Smoothed Particle Hydrodynamics (SPH) numerical algorithms to solve coupled differential equations which link together interacting fluids and, in the future, circuits. The coupled numerical models that make up SPFMax are especially designed to solve 3D MHD equations, making it ideal for modeling the dynamics of a plasma railgun. This chapter serves as an introduction to the design of SPFMax and the railgun case studies performed for this thesis. For completeness, the code and implementation are discussed in entirety. It should be noted however, that only the gasdynamic portion of SPFMax has been utilized. The testing of the electromagnetic and circuit models are reserved for future work as a direct follow on to this thesis.

2.1 Coupled Circuit and Lumped Element Model

As with all numerical solvers, SPFMax has been designed around a set of assumptions to simplify, when possible, the simulated system in order to minimize processing time. A lumped element model for circuit parameters has been chosen over a distributed model for this very reason. The plasma railgun, based on the designs of the 2.5 cm bore railguns developed at HyperV, will be modeled using a 1D (discussed in Chapter 3) and 3D PPT slug model which assumes the plasma is accelerated as a solid mass, with no leakage [8]. The circuit will, therefore, be modeled as a capacitor in series with a constant resistor, a constant inductor, a variable resistor, and a variable inductor. This results in a set of coupled differential equations for voltage, current, velocity and position.

The derivative of voltage with respect to time depends on the current, $I(t)$, and the capacitance, *C*, as shown in equation 2.1 below [8].

$$\dot{V}(t) = -\frac{I(t)}{C} \tag{2.1}$$

The derivative of current with respect to time depends on the voltage, $V(t)$, the current, $I(t)$, the circuit resistance, *R*, the velocity, $u(t)$, the inductance gradient, L', and the total inductance, $L(t)$, as shown in equation 2.2 [8].

$$\dot{I}(t) = [V(t) - I(t) \cdot u(t) \cdot L'] \frac{1}{L(t)} \tag{2.2}$$

The derivative of velocity with respect to time is defined by the inductance gradient, L', the current, $I(t)$, and the mass of the plasma, m, as shown in equation 2.3 [8].

$$\dot{u}(t) = \frac{I(t)^2 \cdot L'}{2m} \tag{2.3}$$

The derivative of position with respect to time is equal to the velocity with respect to time.

$$\dot{z}(t) = u(t) \tag{2.4}$$

The total inductance which is used in the current equation is an implicit function of time and depends on the parasitic inductance, L_0, the position, $z(t)$, and the inductance gradient, L' [8].

$$L(t) = L_0 + L' \cdot z(t) \tag{2.5}$$

The inductance gradient is a function of the geometry, but does not vary with time. The parallel rail geometry consists of the rail spacing, d; the rail width, w; and the permeability of free space, μ_0, as described in equation 2.6, which is approximated to be $4\pi \times 10^{-7}\text{H/m}$ [8].

$$L' = \frac{\mu_0 d}{w} \tag{2.6}$$

Equation 2.6 represents a rough estimation method for calculating L', but other numerical and estimation methods exist for calculating L' which have proven more accurate. An accurate inductance gradient calculation is extremely important since the force on the ejected mass of a railgun is a function of both L' and current, I [8].

$$F = \frac{1}{2} L' I^2 \tag{2.7}$$

One of those methods, discussed later in Chapter 3, uses geometric parameters to estimate the value of L' that is used in the simulations. Implementing the lumped element model requires a simple MATLAB program for the 0D/1D simulations. 3D simulations are calculated using the SPFMax code and coupled with governing fluid equations and Maxwell equations.

2.2 Governing Equations

2.2.1 Fluid Model

Plasma, like all states of matter, is subject to conservation laws. The combination of fluid equations and equations of electromagnetism makes up the mathematical theory of a branch of physics called magnetohydrodynamics (MHD) – the study of interactions between conductive fluids and magnetic fields. In the SPFMax code, plasma particles interact with and affect one another based on three fundamental MHD equations of conservation: continuity, momentum, and heat balance [38].

$$\frac{\partial n_e}{\partial t} + n_e \nabla \cdot \mathbf{u_e} \tag{2.8}$$

$$\frac{\partial n_i}{\partial t} + n_i \nabla \cdot \mathbf{u_i} \tag{2.9}$$

$$n_e m_e \frac{\partial \mathbf{u_e}}{\partial t} + \nabla p_e + e n_e (\mathbf{E} + \mathbf{u_e} \times \mathbf{B}) = -\nabla \pi_e + \mathbf{R_e} \tag{2.10}$$

$$n_i m_i \frac{\partial \mathbf{u_i}}{\partial t} + \nabla p_i - Z e n_i (\mathbf{E} + \mathbf{u_i} \times \mathbf{B}) = -\nabla \pi_i - \mathbf{R_e} \tag{2.11}$$

$$\frac{3}{2} n_e \frac{\partial k T_e}{\partial t} + p_e \nabla \cdot \mathbf{u_e} = -\pi_e : \nabla \mathbf{u_e} - \nabla \cdot \mathbf{h_e} - (\mathbf{u_e} - \mathbf{u_i}) \cdot \mathbf{R_e} - Q_i \tag{2.12}$$

$$\frac{3}{2} n_i \frac{\partial k T_i}{\partial t} + p_i \nabla \cdot \mathbf{u_i} = -\pi_i : \nabla \mathbf{u_i} - \nabla \cdot \mathbf{h_i} + Q_i \tag{2.13}$$

The subscripts denote electrons (e) and ions (i). The velocity term is represented by **u** (the boldness indicates direction in addition to magnitude). Instead of ρ indicating density, *n* represents a number density because particles are being considered instead of substances. Pressure, temperature, and mass are represented by *p*, *T*, *m* respectively. Boltzmann's constant relating particle energy and temperature is represented by *k.* Magnetic field is represented by **B** in this set of equations (instead of **H**, used later in the circuit equations). Additional energy terms include ion charge number (*Z*) and charge (*e* – non-subscript). The dissipative terms are expressed as resistivity ($\mathbf{R_e}$), viscosity (π), heat conductivity (**h**), and thermal equilibrium (Q_i) [38].

The MHD fluid equations appear very different from the Navier – Stokes fluid equations they are based on. For one, the MHD equations are Lagrangian. In Eulerian space, mass, momentum, and energy equations have a convection term because the property being evaluated is done so from a fixed point along the fluid flow. So in other words, the time rate of change of the fluid is being calculated, as well as the influence of the property around the fixed point. However, a Lagrangian fluid follows individual particles throughout the flow, and there is no need for a convection term. So in the Lagrangian reference frame, the time rate of change of the fluid shows mass

conservation. In MHD physics, electrically conductive fluids also experience Lorentz forces in addition to pressure gradients. Because of this, Lorentz forces are added to the momentum equations. Resistivity also plays a very important role in MHD since magnetic and electric field strength depends on how well electric current is able to move through the plasma, or rather, the electrons of the plasma. Finally, one might notice that the energy balance equations for electrons and ions of the plasma are slightly different from one another. The difference lies in the fact that the kinetic energy of the ionic population is effectively not changing due to the influence of electronic collision, whereas ions tend to affect the kinetic energy of electrons. This is an issue of relative size, not unlike the way that terrestrial gravity can be exploited for boosting the effective speed of artificial satellites [38].

2.3 Smoothed Particle Hydrodynamics and Electromagnetics Method

2.3.1 Smoothed Particle Hydrodynamics

Smoothed particle hydrodynamics (SPH) is a mathematical technique used to solve complex fluid dynamics problems by taking the fluid and replacing it with a set of particles that can move with flow over a period of time. It was first used by Lucy, Monaghan, and Gingold in 1977 to solve three-dimensional astrophysical problems, particularly polytropes, but since liquid and gas flow is so similar, adaptations have been used to accurately solve fluids problems as well. Since its initial use, SPH has been used for several engineering fluids models and has become a staple in the film and video game industries as a method of creating detailed fluids animations [39].

SPH has several advantages over grid-based calculations, including the fact that it is inherently adaptive. SPH accomplishes this because it operates as a Lagrangian fluid

description. Unlike particle-in-cell methods, SPH does not require the use of a grid to create a domain of particles [11]. As a result, when using SPH methods, mesh refinement is not necessary since changes in flow are automatically taken into account. Grid-based codes also lose fluid when performing boundary-less simulations, but SPH codes do not. And because the particles carry with them all the physical dimensional properties, conservation laws are always observed.

The smoothed aspect of SPH means that the material properties of the fluid are approximated over the average effect of the particle domain, or that the material properties are interpolated from the property values of the particles [39]. Focusing on a single particle, the encompassing area of effect on that particle is spanned by length called the smoothing length. Each particle within twice the distance of the smoothing length affects the original particle on a Gaussian scale. The effects that neighboring particles have on the original particle can be quantified via interpolation and can be found using the integral interpolant [39]:

$$A_{\mathrm{I}}(r) = \int A(r')W(r - r', h)\mathrm{d}r' \tag{2.14}$$

where A is the quantity measured (density, temperature, etc.), W is the differentiable kernel function, $\mathrm{d}r'$ is the volume differential, and h is the smoothing length. The integral can then be summed over a group of mass elements [39]:

$$A_{\mathrm{s}}(r) = \sum_b m_b \frac{A_b}{\rho_b} W(r - r_b, h) \tag{2.15}$$

One might notice that this summation is very similar to calculations of probability density. This is no coincidence, as SPH interpolation and summation is based on those principles. The smoothing kernel generally takes on the form [39]:

$$W(r,h) = \frac{\sigma}{h^{v}} e^{-q^2} \tag{2.16}$$

where $q = r/h$ and $\sigma = \left[\frac{1}{\sqrt{\pi}}, \frac{1}{\pi}, 1/(\pi\sqrt{\pi})\right]$ in one, two, or three dimensions. Resolution can be modified with density and can depend on position and time. While performing SPH interpolation, as distance increases away from *h*, *W* will drop to 0 [39]. Think of kernel function as a measure of influence that neighboring particles have on a particular particle. The influence of neighboring particles effectively acts like a Gaussian filter. The Gaussian filter decays rapidly to zero influence as neighboring particles move further away [40]. This principle is possibly better understood through visualization (See Figure 2.2). Additionally, the smoothing kernel must tend to a delta function as h tends to 0 and a normalization condition. In other words, with a smaller *h*, the kernel of the program must also be of a higher order [39]. Higher order spline kernels give better stability because they provide a more detailed solution [39], but they come with a high computational cost, the one major weakness of SPH programs, due to the higher number of contributing neighbor particles.

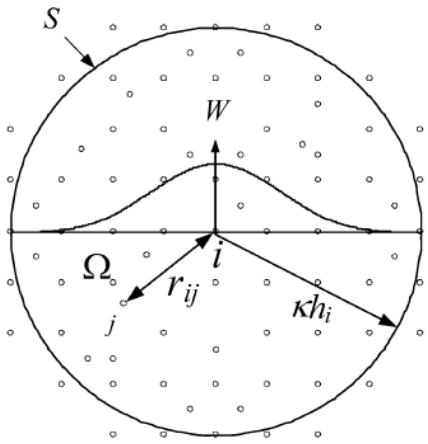


Figure 2.2: A 2D representation of a typical SPH particle calculation shows how the kernel function incrementally affects neighboring particles located within the smoothing length [40].

2.3.2 SPEM and the Circuit Model

Early on in development of SPH, several non-hydrodynamic applications were explored, including modeling Maxwell's curl equations [41]. This eventually led to the development of smoothed particle electromagnetics (SPEM). SPEM algorithms are applied to a circuit model in SPFMax to solve for changes in magnetic and electric fields as well as current propagation throughout electrode solids [41]. Like SPH, SPEM was created to solve complex problems without the need for meshes. In this manner, SPEM is a direct alternative to circuit modeling techniques like finite difference time domain methods. SPEM models solve Maxwell's equations over a domain of arbitrarily spaced particles, just as SPH models are used to solve fluid dynamical equations. Maxwell's electric and magnetic field equations are [42]:

$$\nabla \times \mathbf{H} = \frac{\partial \mathbf{E}}{\partial \mathbf{t}} + \mathbf{J} \tag{2.17}$$

$$\nabla \times \mathbf{E} = -\mu \frac{\partial \mathbf{H}}{\partial \mathbf{t}} \tag{2.18}$$

where **H** represents the magnetic field, **E** – the electric field, **J** – the current density, μ and is the permeability of circuit material. Calculating changes in electric and magnetic fields is a bit different from calculating physical properties such as temperature and density in that changes depend on field strengths of the neighboring particles. In other words, electric field changes depend on the strength of the magnetic field in neighboring particles, whereas magnetic fi/eld changes depend on the strength of the electric field in neighboring particles. For this reason, using SPH to calculate changes in electric field and magnetic field over time requires the implementation of the kernel function (W) so that the effect of particle position (r) on current propagation and chemical reaction can be

illustrated completely. This explanation of the circuit physics becomes clearer in the expanded curl equations in the *x*-, *y*-, and *z*-directions, where new variables ε, σ, and V represent the permittivity, conductivity, and volume of the circuit material respectively [41].

$$\frac{\partial E_x}{\partial t} = \frac{1}{\varepsilon}\sum_{j=1}^{N}\left[H_z\left(r_j^H\right)W_y^E - H_y\left(r_j^H\right)W_z^E\right]V_j + \frac{\sigma E_x(r_i^E)}{\varepsilon} - \frac{J_x}{\varepsilon}$$

$$\frac{\partial E_y}{\partial t} = \frac{1}{\varepsilon}\sum_{j=1}^{N}\left[H_x\left(r_j^H\right)W_z^E - H_z\left(r_j^H\right)W_x^E\right]V_j + \frac{\sigma E_y(r_i^E)}{\varepsilon} - \frac{J_y}{\varepsilon} \qquad (2.19)$$

$$\frac{\partial E_z}{\partial t} = \frac{1}{\varepsilon}\sum_{j=1}^{N}\left[H_y\left(r_j^H\right)W_x^E - H_x\left(r_j^H\right)W_y^E\right]V_j + \frac{\sigma E_z(r_i^E)}{\varepsilon} - \frac{J_z}{\varepsilon}$$

$$\frac{\partial H_x}{\partial t} = \frac{1}{\mu}\sum_{j=1}^{N}\left[E_z\left(r_j^H\right)W_y^E - E_y\left(r_j^H\right)W_z^E\right]V_j$$

$$\frac{\partial H_y}{\partial t} = \frac{1}{\mu}\sum_{j=1}^{N}\left[E_x\left(r_j^H\right)W_z^E - E_z\left(r_j^H\right)W_x^E\right]V_j \qquad (2.20)$$

$$\frac{\partial H_z}{\partial t} = \frac{1}{\mu}\sum_{j=1}^{N}\left[E_y\left(r_j^H\right)W_x^E - E_x\left(r_j^H\right)W_y^E\right]V_j$$

The expanded Maxwell's equations are based on formulations provided by Ala and Francomano with a slight difference in the additional current density term [41], [42]. This term becomes necessary to enable the SPFMax code to simulate interaction between the surface a solid electrode and the charged fluid particles of the fluid model (in this case, the fluid is a plasma). In the propulsion simulation, electrons in the plasma take on current from the circuit model, essentially becoming part of it while simultaneously encompassing the fluid model as well. Certain dynamics of the circuit-fluid model are

very different from the circuit-electrode model. Because the electrode is a solid surface, the permittivity, permeability, and conductivity of the circuit material is assumed constant. However, those same properties must be interpolated in the plasma because the plasma is deforming and accelerating over the course of the simulation. As particle concentration changes, physical properties change.

2.4 SPFMax Description

Two flowcharts are used to depict the major processes and subroutines of SPFMax. The first of which can be found on the next page showing the steps that SPFMax takes in order to set-up the heavy lifting of the SPH solver subroutine. The other flowchart is featured later in this chapter, discussing the SPH solver and its structure and expanding it out to illustrate important SPH features such as the neighbor particle look-up routine and the differentiable kernel function.

For navigation of the flowcharts, the blue box indicates initiation of the SPFMax code. The purple smooth-corner boxes represent subroutines in the code. Major if/then commands and while loops are represented by green diamonds, while dark red ellipses represent major object strings and calculations which are not, themselves, subroutines. Finally, output is represented by an orange parallelogram in the second flowchart.

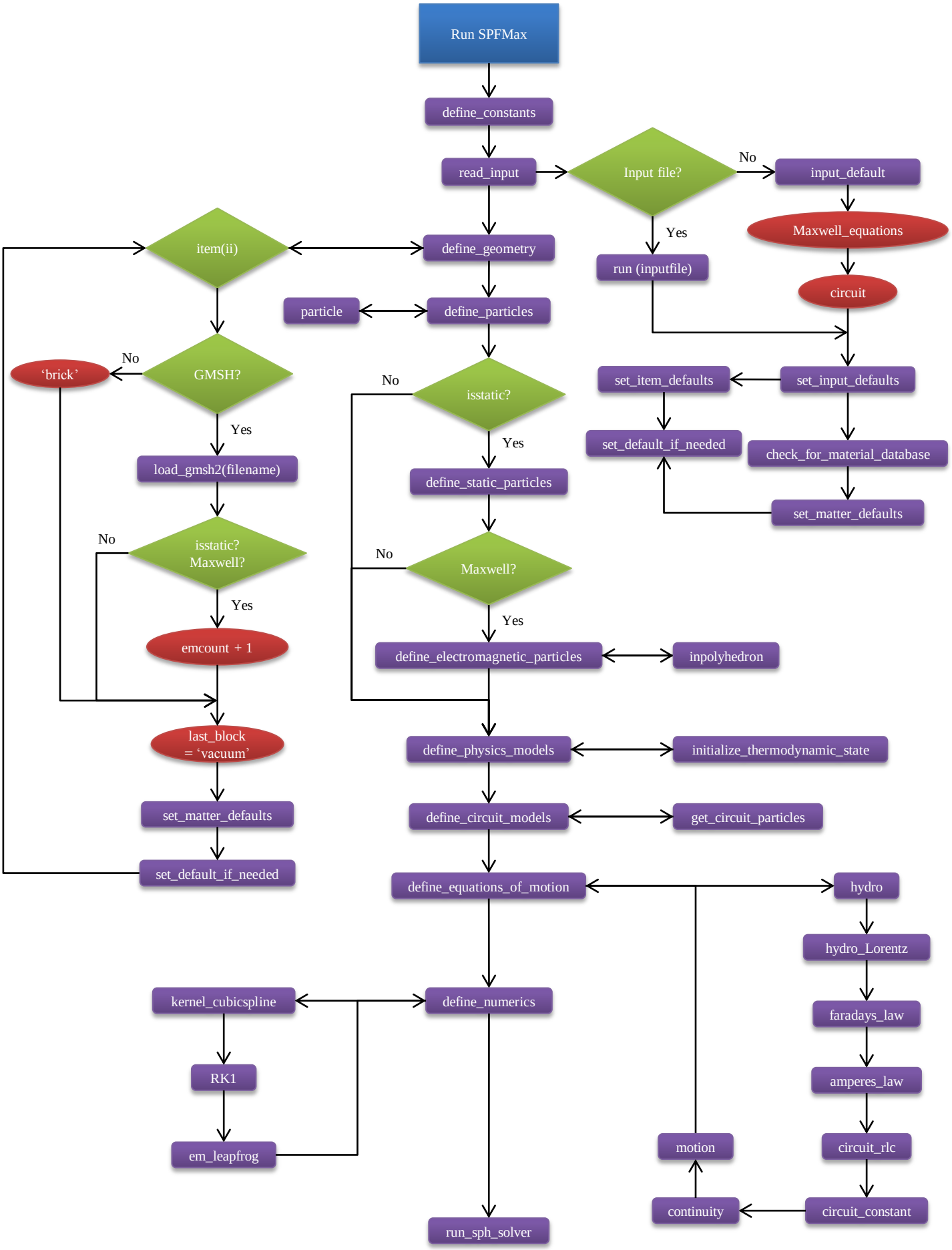


Figure 2.3 The process flow of the SPFMax code up to the SPH solver subroutine indicates the preliminary steps that the code goes through in order to calculate the physical interactions.

2.4.1 Geometry

SPH was first developed to analyze non-uniform plasma interaction and star formation [39]. For these groundbreaking studies, boundary conditions played a very minor role in affecting particle behavior. Since then, geometry has been successfully implemented in many SPH studies, particularly when both boundary conditions and free conditions are necessary to predict behavior. For example, when a dam bursts, water flows through the area of failure into free space. SPH simulations would not just be able to predict the mass flow rate of the water, but it would be able to predict the direction of the flow and how the surrounding area would affect it. In SPH propulsion studies, a geometry serves as a location of intense fluid interaction shortly before those fluids are directionally ejected and become free to move in a simulated vacuum.

2.4.1.1 Introduction to GMSH

SPFMax is designed to run around a simulated geometry that provides the boundary conditions for the fluid reactions. Groups of particles make up the general shape of the geometries and the fluids flowing around them. The particles belonging to solid simulated parts remain in initial positions, while particles of gas are designed to move according to physical laws. Because SPFMax has been developed in MATLAB, it seems only logical to develop geometries using MATLAB as well. This is much easier said than done, however. Simple geometries can be created in MATLAB, as well as complex ones, but creating complex geometries is extremely time consuming and more difficult to design without a visual interface. Therefore, SPFMax employs a finite element meshing program called GMSH to create the more complex geometries needed to simulate thruster performance. Though seemingly contradictory to the nature of SPH

codes, the 3-D tetrahedral meshes that are created using GMSH or DistMesh are used to provide the initial position for particles within the simulation. The meshes themselves are not visualized in the simulation and do not require re-meshing. Therefore, "meshing" the geometry only serves to randomize particle location, and the SPH methods employed in SPFMax remain, in fact, mesh-less.

2.4.1.2 SPFMax Use of GMSH

GMSH functions similarly to other finite element applications and computer aided design programs; however, in addition to creating and meshing 3-dimensional geometries, GMSH creates an array of numerical identifiers that are assigned to each entity of the geometry. Entities include points, lines, arcs, volumes, surfaces, etc. The array can be opened with most operating system note programs, such as Microsoft Notepad, and edited. Once edited and saved, the model can be refreshed and visualized in GMSH with the edited attributes intact. In addition to numerically identifying entities, GMSH also tracks 3-D space coordinates for all boundary points and nodes.

Nodes refer to points that make up element shapes within the meshed geometry. Elements are smaller shapes that make up the body of the geometry in the form of a mesh. In GMSH, 3-D meshes are constructed using tetrahedral elements. As with geometry entities, GMSH creates an array of numerical information that makes up a sort of genetic code of the mesh and volume. Specifically, each node is numbered and tracked with 3-D space coordinates in a node array. In the element array, each element is labeled as an array of node identifiers and additional object or volume identifiers. Subroutines within the SPFMax code read necessary information into MATLAB so that MATLAB

can recreate the geometry exactly to specification and run simulations of fluids moving within it.

2.4.1.3 Using Boundary Conditions

Each boundary condition identifier provides greater flexibility for grouping sets of nodes for use in MATLAB. For example, an object or portion of a geometry can be assigned a volume identifier that MATLAB will always associate with all of the nodes that make up the volume. From MATLAB, the nodes – now referred to as particles – can be assigned physical properties of the volume being modeled. In the case of railgun modeling, gas properties are assigned to volumes representing the gas volume, while the circuit model is assigned to the volumes representing the anode and cathode.

2.4.2 SPH Solver

Much of the code written for SPFMax organizes the information that is eventually calculated by the SPH solver subroutine. Functioning much like the engine of a car, the solver is comprised of several subroutines that work together to correctly interpolate the interaction of simulated particles. These subroutines are designed to primarily initialize and carry out calculations over each time step of the simulation duration and save the data to be presented as an animation or as a frame from a specific time step. The primary processes of the SPH solver subroutine can be found on the flowchart on the next page.

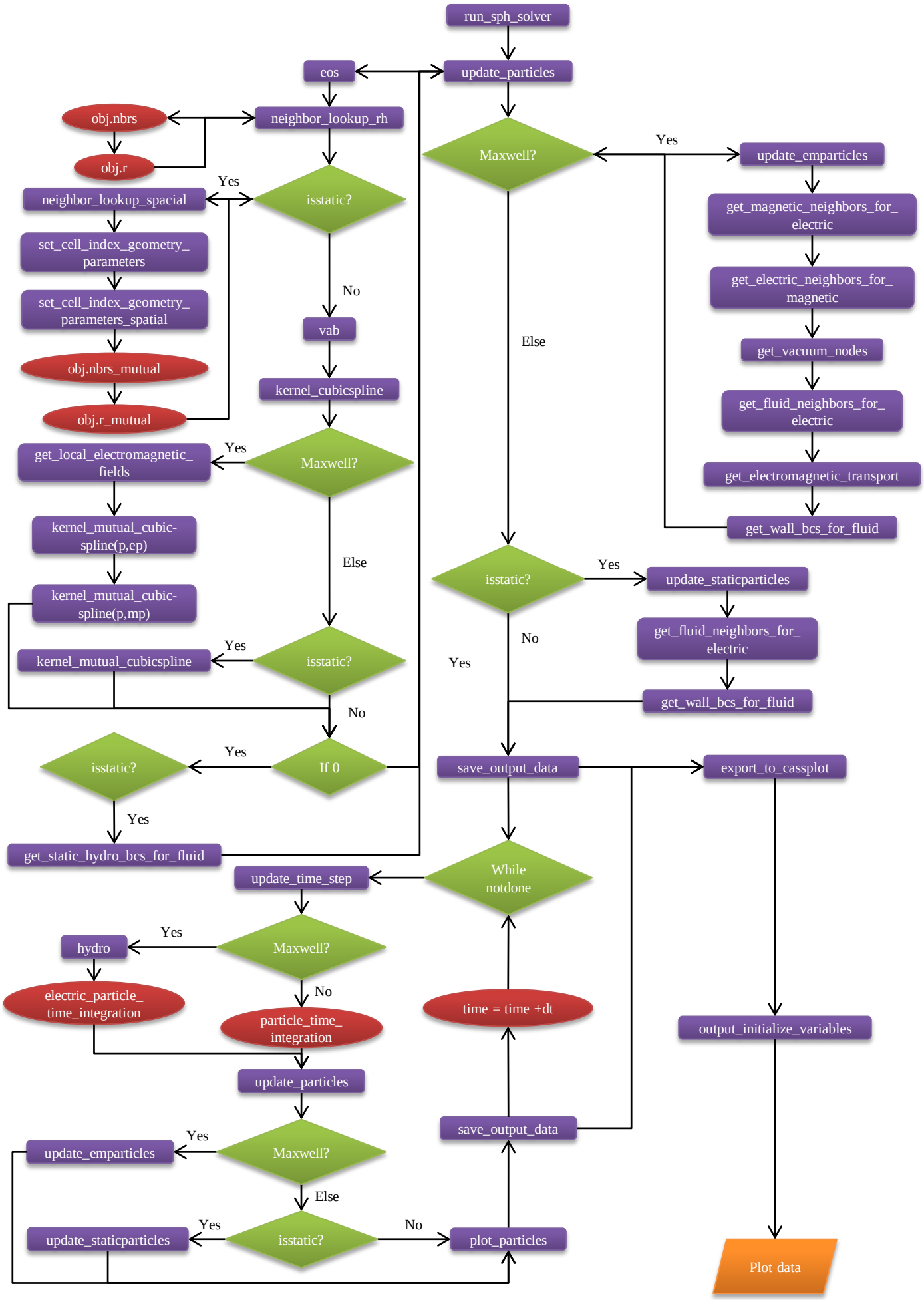


Figure 2.4 The flow diagram of the SPFMax subroutine sph_solver.m illustrates the construction and primary processes that take place during a simulation.

2.4.2.1 Initial Particle Characteristics

The first phase of the solver involves metaphorically firing up the engine by initializing the previously organized structures of information. This process involves having the update_particles.m subroutine take particle information from the subroutines responsible for loading the geometry and circuit model and feed that information into the subroutines responsible for finding the neighbors of particles and the smoothing length over which the kernel function is differentiated.

2.4.2.1.1 Assigning Particle Characteristics

SPFMax has three particle designations or classes: particles (fluid), electric/static particles, and magnetic particles. SPFMax uses certain particle classes for specific calculations throughout the program. The particle subroutine is designed to simplify coding and decrease processing time. How it accomplishes this is by "pointing" to particle associated variables throughout the code using a handle (commonly referred to as a pointer in other programming languages). Each subroutine is a function of calculated variables, but using a single pointer class to point to an entire tree of possible variables eliminates the need for long function statements. The particle subroutine is a variable tree that performs this exact service, allowing most calculations to be functions of a single class instead of multiple imbedded variables.

In addition to simplifying code, pointers make it possible to use previously calculated information to solve complex numerical simulations because pointers use information already stored in memory to run calculations that are functions of variables stored in the pointer class tree. Using a pointer is especially beneficial when handling large data sets like the locations and material properties of hundreds of thousands of

simulated particles. Without the pointer, each particle that changes somehow during the simulation would be copied over and over again with each modification, slowing the processing time significantly. While not completely necessary, the particle subroutine improves upon the organization and processing time of SPFMax.

2.4.2.1.2 Local Magnetic Fields

Magnetic and electric fields in SPFMax are found by looking up the neighbors of the magnetic and electric classes of particles which are pointed to using the particle subroutine. The electric and magnetic fields are then calculated at each time step by finding the overall smoothing length between electric and magnetic particles and interpolating the changes in state via the mutual kernel function. Those interpolations are then summed to get the desired electromagnetic field quantities which are used to calculate effects on fluid particles.

2.4.2.1.3 Neighbor Particles

Finding neighboring particles of a specific particle is essential to solving SPH problems. Neighboring particles refer to particles within the smoothing length of a particle. Each particle in the simulation has a list of neighboring particles found at each time step. There are two subroutines that find neighboring particles: neighbor_lookup_rh.m and neighbor_lookup_spatial.m. Both subroutines perform the same function, but the spatial neighbor subroutine finds neighboring particles of multiple classes in order to solve for certain properties. For example, in order to solve for electric and magnetic fields, particles of both classes are needed to solve for one another. Therefore, neighboring electric and magnetic particles are found to accurately model changes in field strength. Whereas the other subroutine finds particles of a like class to

keep neighbor lists organized and simplified for conditions when only like-class particles interact with one another.

The neighbor look-up subroutines are designed to find the maximum neighboring particles of each particle and present them in a columned array with the total number of particles, taking into account the desired smoothing length value and three dimensional coordinates of the particles. The resulting neighbor particle and coordinate lists are used to calculate changes in state of all the particles using the kernel function. Particles not listed in a specific neighbor list are assumed to have minimal if any effect on the particle for which the neighbor list is created.

2.4.2.1.4 Cubic-spline Kernel Function

As explained earlier, kernel functions are the center of SPH calculations. SPFMax is no different in that regard. Every calculation is centered on a piecewise cubic spline. Monaghan explains that cubic splines make ideal kernel functions because higher order functions, while providing more accurate detail under uniform particle distribution conditions, contain gradients that continually change sign, introducing artificial negative elements which must be cancelled by positive elements [39]. Randomized particle distributions do not have equally spaced particles and are not ideal candidates for higher order kernel functions [39]. The cubic spline, however, does not contain a gradient that changes sign, but it is of a sufficiently high order so that its derivative is continuous. The cubic spline provides a very strong balance of accuracy and simplicity [39].

SPFMax, once again, uses two variations of the kernel function in order to make calculations concerning the two different types of neighbor list. The two subroutines are called kernel_cubicspline.m and kernel_mutual_cubicspline.m. The mutual kernel

function is capable of being differentiated over two different classes of particles. This kernel function is primarily used when considering electric and magnetic particles. The other cubic spline kernel is used solely to calculate the gradient of neighboring particles of a single type, generally fluid particles.

2.4.2.1.5 Continuity Calculations

The velocity of one particle depends on the velocity of its neighbors. Calculating the relative velocity of all the particles requires the neighbor object arrays that were created by the neighbor look-up subroutines. The directional velocity of each particle is replicated into an array with its neighbors. The velocity difference is calculated by subtracting the directional velocities of neighboring particles from the directional velocities of each particle.

2.4.2.1.6 Equations of State

Much of the information being solved for hinges on changes in state of the particles. The state equations are material based, not spatially based. This means that particles are looped into a material identifier and assigned material specific properties. However, once the material properties are considered, other state conditions change over time and space. For example, density changes with different concentrations of mass. Pressure and temperature does as well. Therefore, the equations of state are solved for with kernel interpolation.

2.4.2.2 Updating Electromagnetic Particles

The behavior of the different classes of particles can differ quite a bit. Therefore, it is expected that different strategies to find neighboring particles should be implemented. In addition to specialized neighbor finding and kernel subroutines, there

are two different subroutines dedicated to updating the positions and finding the properties of electromagnetic and fluid particles.

The script dedicated to finding and updating electromagnetic particles is specially designed to calculate the influence of the electromagnetic particles over the other types of particles. Electric propulsion utilizes plasmas to carry electric current and complete the circuit. SPFMax simulates this principle by taking the neighboring electric and magnetic particles and finding neighboring dynamic gas particles that transport the electric current. The first step is setting up imbedded neighbor finding subroutines for magnetic and electric particles. Next the subroutines set aside particles that neighbor fluid particles. Next, a subroutine designed to calculate the electric transport of neighboring fluid particles calculates the fluid kernel (mutual kernel function) for those dynamic fluid particles. Then the mass, density, and conductivity for those particles are calculated over the kernel function. The conductivity over all of the dynamic (those fluid particles that transport electric current) particles is summed.

2.4.2.2.1 Time Integration of Electromagnetic and Fluid Particles

Throughout each electrode geometry, Maxwell's equations are calculated over time to find the voltage and current in each particle. SPFMax employs a pointer for at each time step where the electromagnetic and fluid time derivatives are being calculated called electric_particle.time_integration and particles.time_integration respectively. The pointer signals a first order Runge Kutta calculation to sum and average the physics models and circuit models. This means that all equations of state and physical characteristics are calculated at each time step using the time integration functions.

2.4.2.2.2 Wall and Vacuum Interactions with Fluid Boundary Conditions

For pulsed plasma systems involving a fixed mass, boundaries which interact with the fluid particles include walls and vacuum. Walls are treated as static fluid particles and are handled as follows. If a particle is within the compact support distance (2h) of a wall boundary, the force orthogonal to the wall is turned off. If the particle has an initial velocity towards the wall, it will undergo a specular reflection if the particle comes within 1×h distance of the wall particle. Static particles also can take on transport properties, such as conductivity. As such, the local conductivity which is used in Ohm's law for calculating current will depend on an SPH interpolant evaluated from the local particle conductivities, which may include a mixture of wall and fluid particles. In the absence of either of these particles, the electric and magnetic field particles will be assigned 0 for the local conductivity, which then dictates that the fields propagate as electromagnetic waves. In this manner, the code behaves like the finite difference time domain method which is one of the standards for modeling propagation of electromagnetic fields.

2.4.2.2.3 Time Step

All calculations conducted by SPFMax are ultimately functions of both space and time. As particles interact in three dimensions, they often do so at varying times because they are arbitrarily spaced. Therefore, SPFMax employs a changing Courant-Friedrichs-Lewy (Courant/CFL for short) condition to solve for the appropriate time step. Originally discovered by Richard Courant, et al, the CFL condition was first used to relate time intervals to the physical spaces between points. Without the Courant condition, the time-dependent differential equations calculated in SPFMax would not converge properly

because many of the particle interactions are taking place before and after certain places in time. The Courant condition factors in the minimum amount of space between particles and the time that interactions would be taking place, so that all possible interactions will be considered in the calculations, reducing error significantly.

2.4.2.3 Saving and Plotting Particle Data

Particle data is saved at each time step. Thanks to the pointer object class, MATLAB does not have to recreate previous calculations like it normally would for a conventional while loop. Instead, information from the previous time step is saved and used for calculating the next step, saving considerable processing time. Once saved, an output data counter is updated basically, telling the data plotting subroutine that there is more data that needs to be shown. The subroutine responsible for plotting particle flow is called cassplot.m. This simple formatted tool is designed to be cohesive with common presentation practices, but it also automatically updates the calculated information so that the end result appears like an animation of physical interaction of particles. By default, each particle is color-coated to show number density intensity. For example, dark red shows a compact particle space, while lighter blue shows how particles have become less dense in certain areas. The plotting tool can also be used to show temperature-time and mass-time relations between particles.

2.5 Modeling Procedure

The SPFMax code has been implemented for the purposes of this work to conduct preliminary simulations of high-energy fluid problems. These simulations will pave the way towards full-scale railgun simulations which will couple the completed circuit model with the fluid model. Initially, this work was to include implementation and discussion

simple railgun simulations using SPFMax. However, the scope of the original focus proved too large for a single thesis. Therefore, the rest of this work will primarily discuss the fluid simulations performed using SPFMax, with the exception of a preliminary 1D railgun slug model. Table 2.1 contains a summary of the tests that will be discussed in Chapter 3.

Table 2.1: Performed fluid model and railgun simulations.

Simulation	Wall BCS	Method	Type	Dimension
Slug Model	None	ode45	Railgun	1D
Expansion Model	Square bore	SPFMax	Fluid	3D
Advection Model	Cube	SPFMax	Fluid	3D
Shock Tube Model	Closed bore	SPFMax	Fluid	1D
Shock Tube Model	Closed bore	SPFMax	Fluid	3D

CHAPTER 3 RESULTS

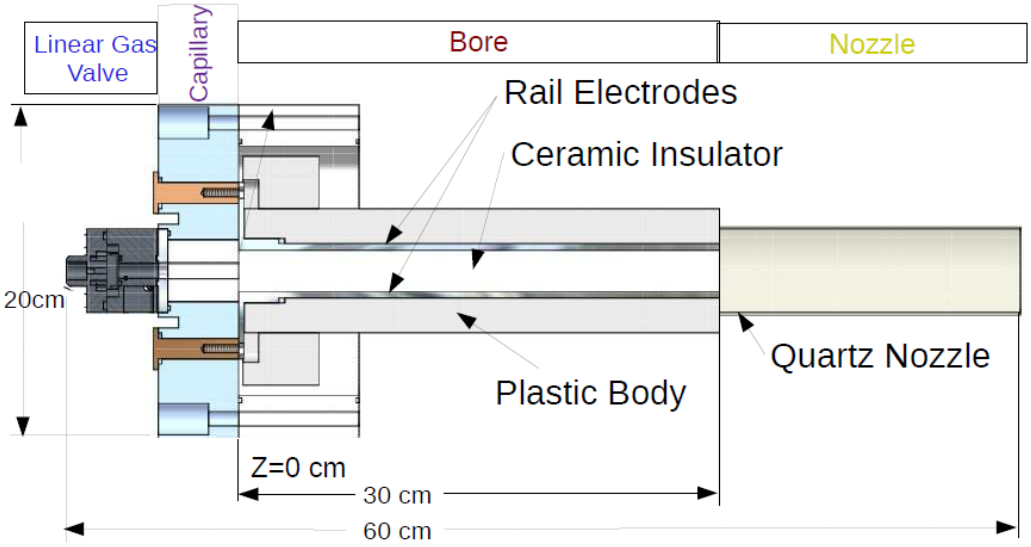


Figure 3.1: The detailed schematic of the HyperV railgun serves as a basis upon which the geometry of the SPFMax model is based [1].

In the design process of their MiniRailgun technology, HyperV performed simulations using the MACH2 code discussed in Chapter 1. As discussed in that chapter, MACH2 has been used to accurately predict plasma physics for several systems. True Lagrangian codes like SPFMax, utilizing SPH numerical methods, are more natural representations of plasma behavior, however, and may be able to provide additional insight into improving emerging technology. For this reason, UAH has ventured to develop the SPFMax code. In this chapter, several tests are performed as precursors toward the ultimate goal of a working railgun model. First, a simple 1D coupled differential equation model establishes the inductance gradient that will be employed in

the completed circuit model. Then the SPFMax simulations are presented and compared with analytic solutions for maximum expansion speed, advection, and a shock tube scenario.

3.1 1D Lumped Element Circuit Modeling

Circuit modeling is performed in order to provide guidance for time scales and velocities encountered in a railgun. Further, with comparisons among this model, a higher fidelity calculation on a validated code such as MACH2, and experimental data, such a 1D model can serve as a test case in the future once the limitations are clearly understood. The differential equations that govern the function of the RLC circuit, defined in Chapter 2, can be integrated numerically given a set of initial conditions for position, velocity, current and charging voltage. A 1D MATLAB program was created to integrate the RLC circuit equations using a built-in ordinary differential equation function and establish the modeling method employed in the 3D model. HyperV has performed similar tests using the 2D MACH2 code. Implementing the same initial conditions as the HyperV model (Table 3.1), the 1D and 3D simulations are designed to also reproduce the results of the HyperV simulations and physical experiments as closely as possible.

Table 3.1: HyperV Initial Conditions [1]

Bore size	2.54 cm square
Bore length	30 cm
Voltage	45 kV
Capacitance	36 μF
Inductance	150 nH
Inductance Gradient (L')	~0.5 μH/m
Gas	Argon (Ar)
Mass of Gas Slug	8,000 μg

Leading up to the creation of the 1D simulation code, a method for modeling the RLC circuit needed to be chosen that would closely replicate the HyperV experimental results. In *Physics of Electric Propulsion*, Robert Jahn describes a number of different methods to model plasma dynamics in a parallel-rail accelerator. The slug model, a model which assumes the initial mass of gas acts like a projectile, is used in both this research and the HyperV simulations. The slug model has a major advantage over other models: it is fairly simple and easy to implement in all dimensions while producing fairly accurate results. In an effort to provide greater accuracy while simplifying numerical processing, HyperV has chosen to use an estimation method based on a series solution of weighted analytic components for calculating the inductance gradient [43].

$$L' = \frac{10^{-6}}{0.5986\frac{h}{s} + 0.9683\frac{h}{s+2w} + \frac{4.3157}{\ln\left(\frac{4(s+w)}{w}\right)} - 0.7831} \tag{3.1}$$

As with the basic L' equation noted in Chapter 2, Equation 3.1 is a function of the geometry of the conductive rails where *h*, *w*, and *s* represent the thickness, width, and separating distance of the conductive rails, respectively. Using the HyperV railgun measurements, the inductance gradient is approximated to be $0.5\ \mu\mathrm{H/m}$. The inductance gradient plays a major role in determining the shape of the basic time-dependent plots of current, distance, and velocity, Figure 3.2.

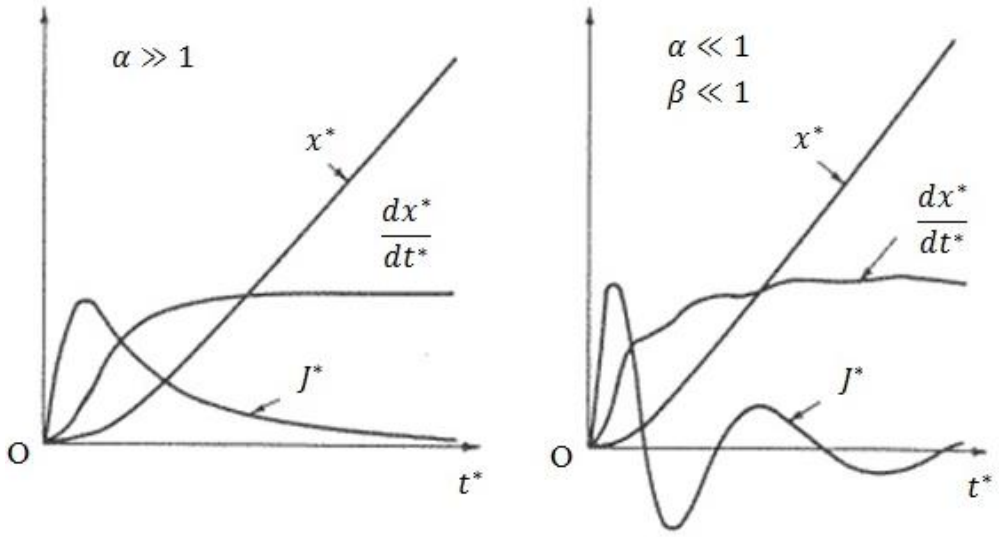


Figure 3.2: Basic 1D plots will tend to take on an overdamped waveform ($\alpha \gg 1$) or oscillate ($\alpha \ll 1, \beta \ll 1$) depending on the scaling factor [8].

Oscillating waveforms depict more natural behavior of time-dependent factors, indicating that the inductance gradient factor is not too high. Certain scaling factors, represented by α and β also predict the behavior of plots and are functions of the inductance gradient. When α and β are very large, graphs of current and velocity will be overdamped and, therefore, inaccurate. However, when α and β are significantly less than 1, plots are oscillatory. The scaling factors are calculated using the following relations [8]:

$$\alpha = \frac{1}{2}\left(\frac{{}^1\!/_2\, CV_0^2}{{}^1\!/_2\, mx'^2}\right)\left(\frac{L'}{R}\right)^2 \beta^2 \tag{3.2}$$

$$= \frac{1}{8\pi^2}\left(\frac{{}^1\!/_2\, CV_0^2}{{}^1\!/_2\, m\dot{x}^2}\right)\left(\frac{2\pi\sqrt{L_0 C}}{L_0/L'}\right)^2 \tag{3.3}$$

Where the circuit variables C, V_0, L_0, and R represent the capacitance, initial voltage, parricidic inductance, and resistance, respectively. Mass and time-dependent distance are represented by m and $\dot{x}$.

The 1D simulations are carried out by a simple coupled differential equation code not associated with SPFMax. MATLAB has a built in differential equation function based on the Runge-Kutta method called ode45 which was used to couple the differential equations discussed in Chapter 2. The majority of the initial conditions of the 1D simulations are the same initial conditions of the HyperV MACH2 simulations (See Table 3-1). The simulation takes place over a timespan of 30 μs with a variable time step that built in to ode45. Tolerances are set in order to find the end of the rails, at which point no additional acceleration should be noted, and oscillations should dampen out.

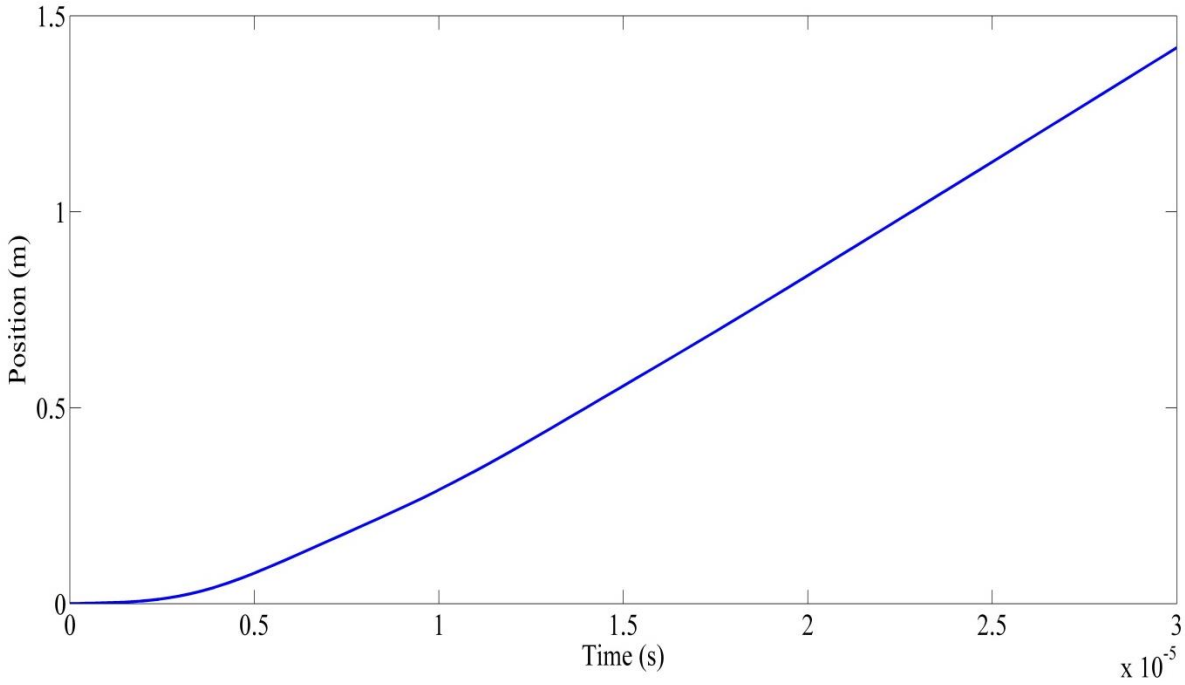


Figure 3.3: The graph of the position of the gas over time shows slow progression that grows exponentially.

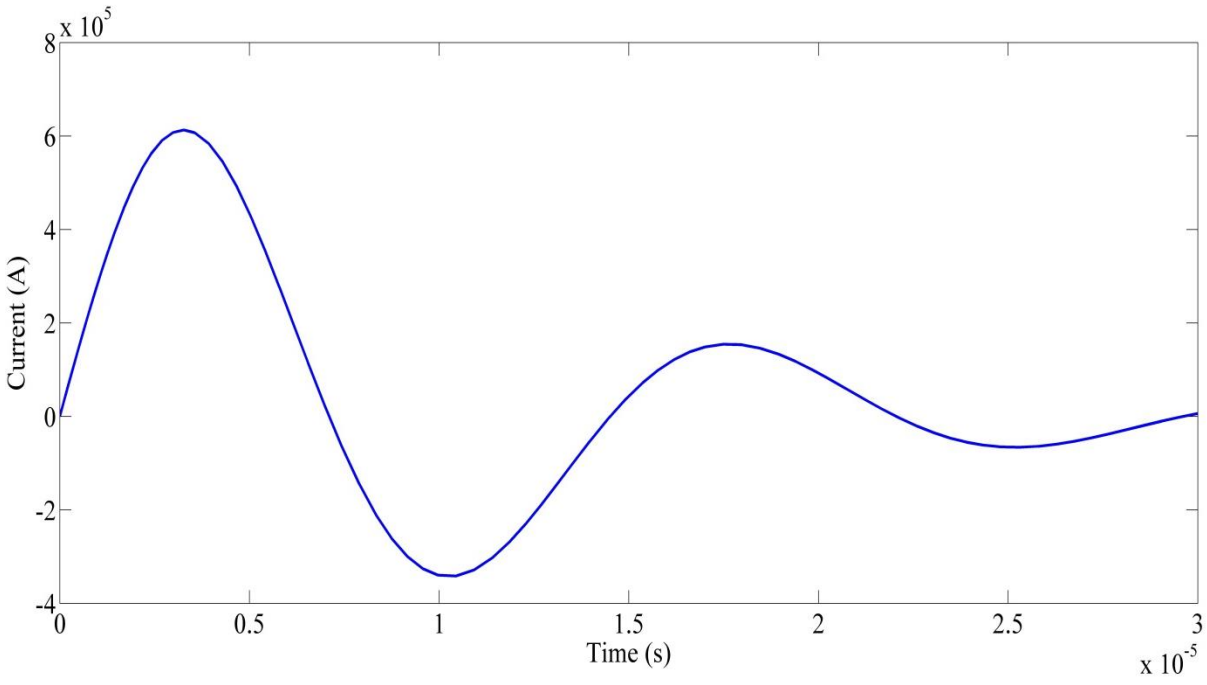


Figure 3.4: The current plot shows heavy oscillations dampening out over time.

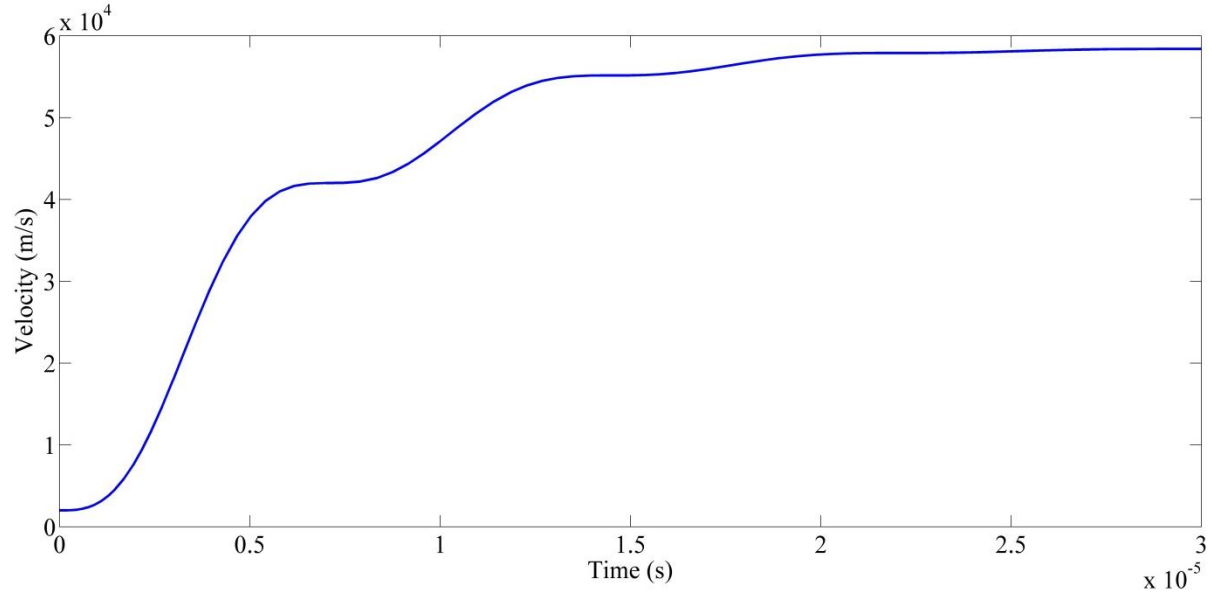


Figure 3.5: The velocity plot shows oscillation slowly dampening to zero acceleration.

Comparing the 1D results with Figure 3.2, the velocity, current and distance plots are in agreement. This means that the scaling factors, α and β, are significantly smaller than 1, and that the inductance gradient is appropriately quantified for the railgun geometry. However, 1D simulations are not indicative of accurate 3D physics, and the 1D simulation only represents an ideal result, assuming constant gas density and mass. The 1D simulation also does not account for viscous drag that the rails have on the gas or ambient mass ahead of the plasma armature that would exist in an experiment. Both of these effects could be included, but would have to be fit to very careful experimental data. MACH2 does account for viscous effects, and more importantly, 2D spatial gradients in the plasma. Consequently, the HyperV simulations predict plasma acceleration more accurately as discussed below.

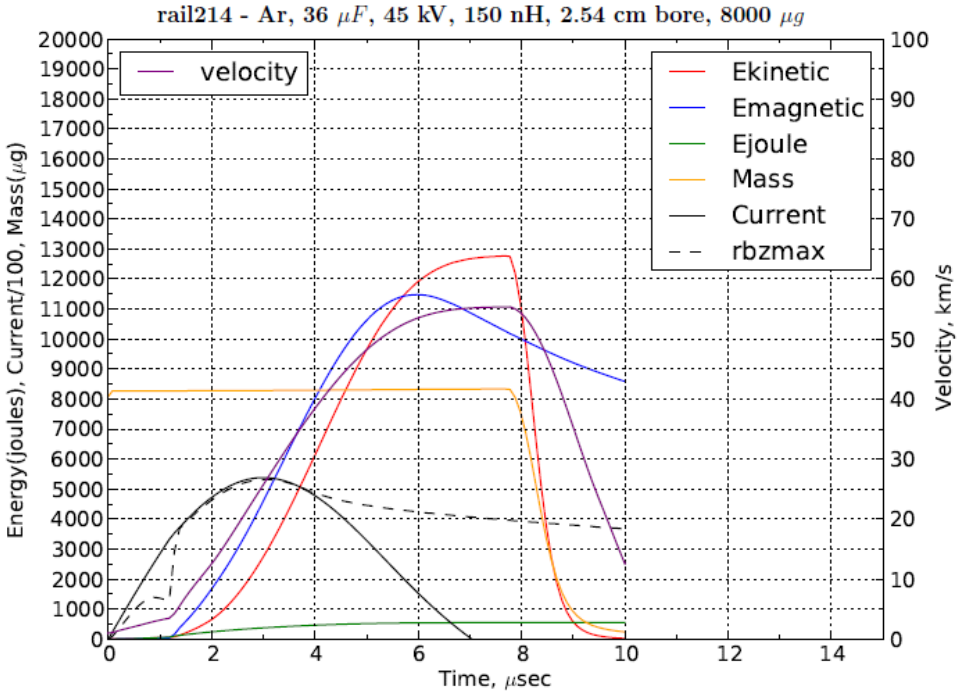


Figure 3.6: The 2D slug model performed using MACH2 shows both a smaller peak current and maximum velocity, likely due to viscous effects [1].

The HyperV simulations show a peak current (black) around 550 kA which is a bit smaller than the peak current of the 1D simulations, peaking at just over 600 kA. The HyperV velocity curve (in purple) shows much more significant differences from the 1D simulation. The velocity of the gas peaks at 55 km/s. While this is only 4% below the 1D maximum velocity of about 57 km/s, the HyperV velocity curve quickly drops after peaking instead of remaining constant like the 1D simulation, due to the presence of drag terms in MACH2 which decelerate the plasma.

When further comparing the 1D simulation to laboratory experiments conducted by HyperV, even greater differences between velocity peaks and current peaks can be seen.

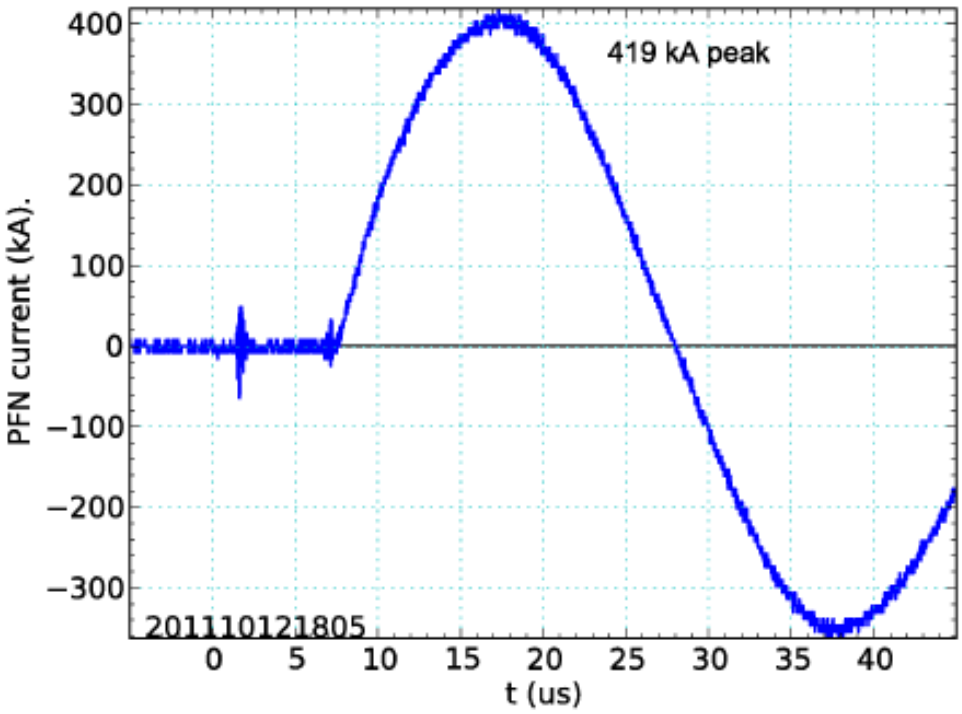


Figure 3.7: During laboratory tests, HyperV measured a peak current at 419 kA. Peak performance requires an approximate current peak of around 600 kA [1].

Current measured in the pulse forming network (PFN) peaks at 419 kA. This further shows that the 1D simulation represents the ideal outcome for the stated initial

conditions because a peak current of around 600 kA is ideal for maximum performance. The velocity curve, in experimental conditions, is measured using a photodiode. The actual velocity is extrapolated using spectroscopy to measure the intensity of light (see Figure 3-8). Compared with the ideal 1D simulation, the experimental plasma jet velocity, measured to be about 44 km/, is ~20% below the model's predicted value.

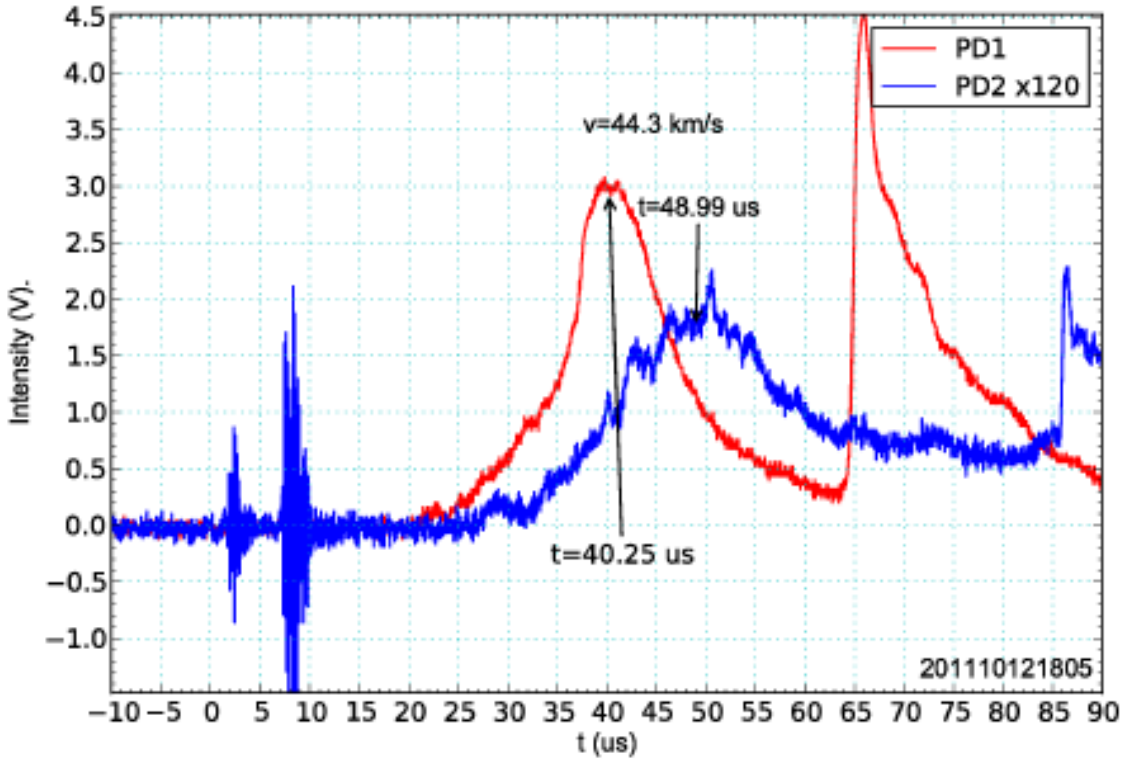


Figure 3.8: Using a photodiode, HyperV was able to measure a plasma jet velocity of 44 km/s during physical testing [1].

Two points from this section should be noted in the context of the thesis. First, test cases for railguns should include plasmas reaching ~40 to 50 km/s. Second, drag terms and other 2D/3D effects appear to reduce the final velocity compared to what is predicted by more idealized 1D RLC lumped element circuit models. Thus, for such a test case to be interpreted properly, the 3D simulation should be expected to produce a

slower velocity by about 10 to 20%, and a set of conditions should be considered in which these departures can be minimized. Under those circumstances, the test case should be considered to be qualitatively accurate and quantitative to within ~20%. Departures in behavior could be quantified and included as drag terms in the equation of motion for the 1D model, in which better agreement between the two models could be achieved. Further confidence in utilizing such a test could include a particle resolution self-convergence study to assure that departures are not attributed to numerical artifacts. The test cases that follow do not reach these velocities, because acceleration is only caused by pressure gradient in the purely gasdynamic problems. Such velocities would require temperatures of ~100 eV, which is unrealistically high. As a compromise, we chose to study cases in which temperatures reach 3 eV, which is typical of the experimental conditions.

3.2 SPFMax Tests

SPFMax tests have been conducted in such a way to ensure that proper physical behavior is being modeled. A gas expansion model is conducted, followed by an advection model and a shock tube model. Collectively, these tests check the code's ability to capture expansion of a gas into vacuum, advect a wave in the flow, and capture shocks and rarefaction waves. Further, these tests are conducted with conditions comparable to the plasma in a railgun, in a relevant geometry subject to potential interference from walls. All of these effects are important to include so that such effects can be mitigated prior to running cases involving circuits and field propagation, to facilitate debugging issues purely associated with field and circuit solvers.

3.2.1 Gas Expansion Test

One of the most basic physics simulations for preliminary study is the expansion of a block of gas inside a geometric structure and out of an opening of the structure. This test serves two purposes. First, the gas expansion test ensures that the geometric structures behave the way they are supposed to and prevent fluid particles from leaking through solid structures. Secondly, the gas block expands and exits openings within the geometry according to thermodynamic laws. For the first SPFMax simulation, a simplified square bore geometry houses a block of gas, initially at rest, that expands and interacts with wall boundaries. The bottom of the bore is capped to prevent the escape of gas except through the top of the structure. The square bore closely resembles a simplified railgun structure that will be used in future coupled-model testing.

As gas is accelerated to its maximum speed, the density and pressure tend to 0, according to the Second Law of Thermodynamics. Since sound speed cannot be negative, the maximum speed that a gas can travel is a function of a gas' sound speed [44]:

$$v = \frac{2c}{\gamma - 1} \tag{3.4}$$

In this function, c represents the local speed of sound of the expanding gas, and γ represents the ratio of specific heats. The SPFMax expansion tests use argon as the working gas.

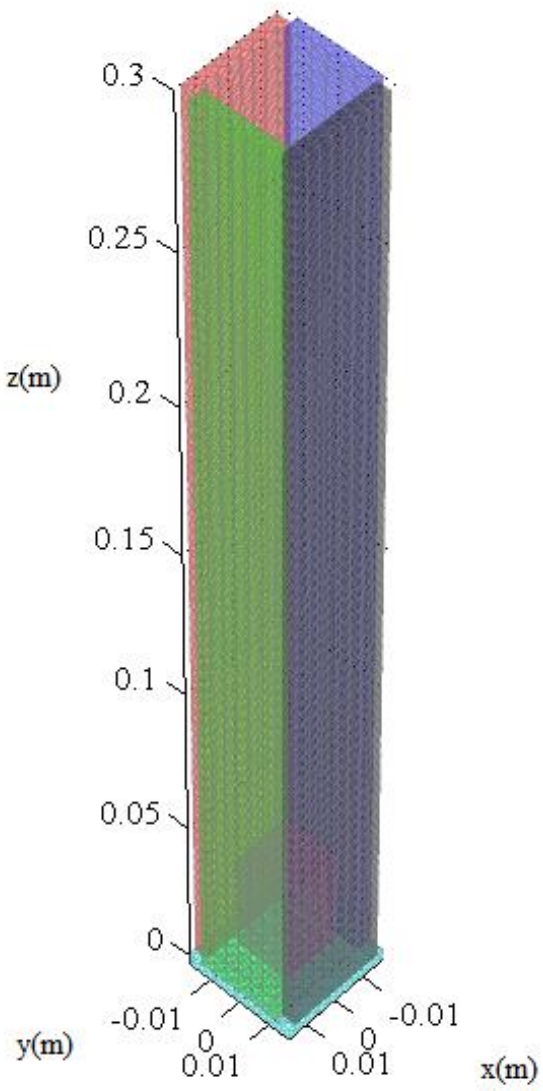


Figure 3.9: The low-resolution geometry of a basic bore model contains a block of gas for the gas expansion test.

In addition to the natural speed limit of the gas, density will also never increase in argon's expanded state. Argon will naturally diffuse into the open vacuum, causing more compact atoms to spread out freely. Therefore, a suitable secondary test for the SPFMax expansion model is to monitor the density of the expanding gas and compare it with the density of the rest of the argon block that has not expanded as much.

The expansion test was conducted for three different resolutions. The low, medium, and high resolution cases include 135, 450 and 1,503 particles respectively, which was comparable to spatial scales of 4, 1.2, and 0.36 mm. With an initial temperature of 30,000 K, the maximum velocity is predicted to be 60,925 m/s. In all cases this value is not exceeded. The expansion processes is within 20% for all resolution tests, with the peak expansion speed increasing monotonically with resolution, Figure 3.10.

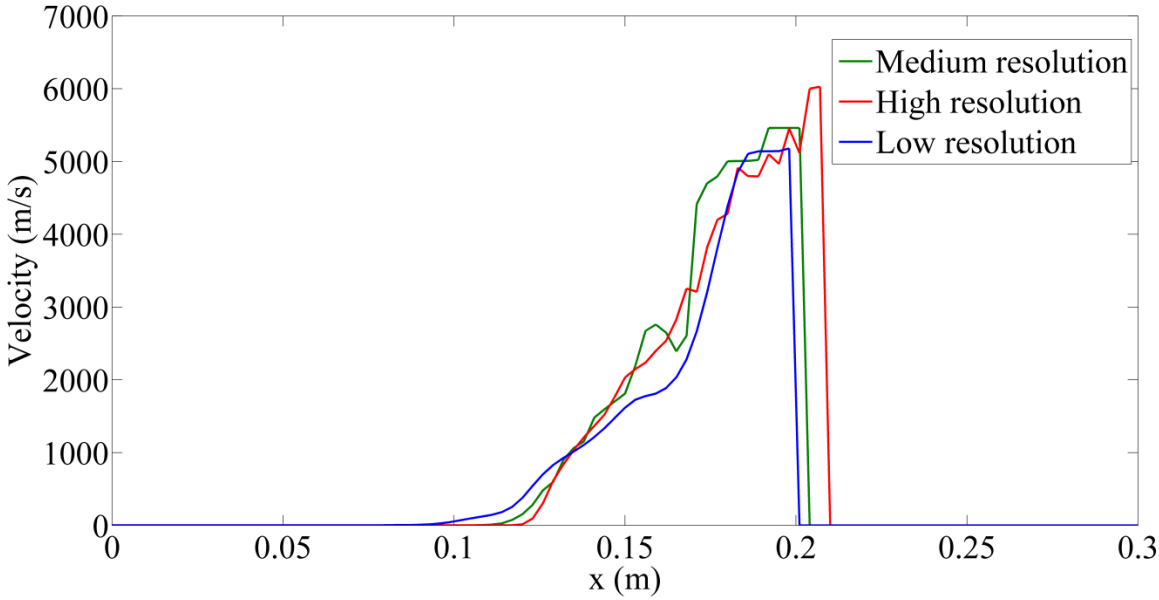


Figure 3.10: Gas expansion test in the railgun at low, medium, and high resolutions involving 135, 450, and 1,503 particles. For reference, the soundspeed is 20,410 m/s, and the maximum speed predicted by theory is 60,925 m/s.

3.2.2 Advection Test

The advection test will isolate diffusive and dispersion errors in wave propagation. To accomplish this, a geometry file of a gas completely bounded by thin walls is created (see Figure 3.11). The box is designed to move at uniform velocity, with the gas, in the x-direction until a specified time is reached. During this time the kernel

function of SPFMax solves and sums the MHD mass conservation equations discussed in Chapter 2 (see Equations 2.7 and 2.8).

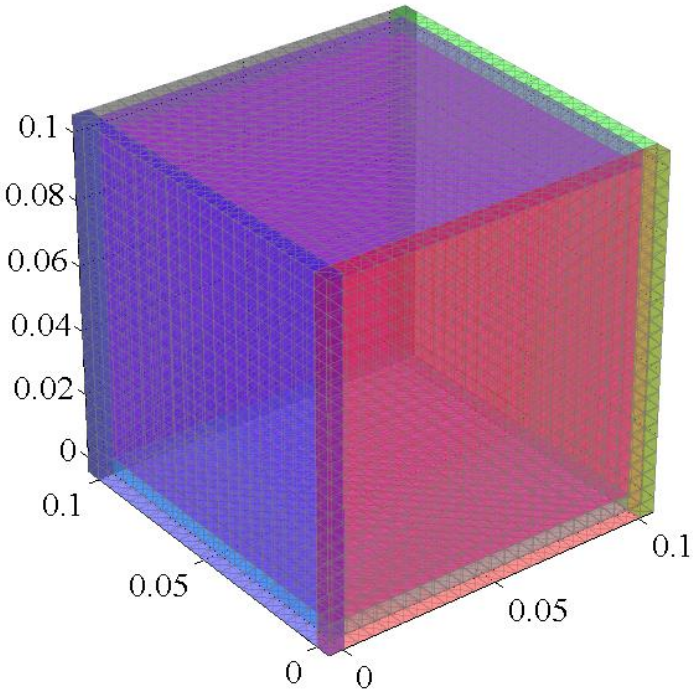


Figure 3.11: A thin-walled cube houses argon gas for the advection test.

For the advection test, temperature is initialized to be reciprocal to the density (i.e. the temperature dips when the density spikes) such that the pressure is constant. In a typical fluid solver, the sharp discontinuity in density and temperature will cause local oscillations or smearing of the wave to the extent that the sharp boundaries are not maintained as the wave propagates. Smearing and oscillations are generally attributed to numerical dispersion and diffusion errors. However SPFMax is not subject to those same errors, so the density square wave will always remain sharp down to floating point accuracy. The initial density starts at 1 kg/m^3 and then peaks at 2 kg/m^3. Temperature

(not shown in the figure) starts at 300 K and dips to 150 K in the same location where density spikes. The velocity remains constant at 1000 m/s throughout the simulation. The run stops at 10^{-4} s, after the wave has propagated 1 unit length of the wave.

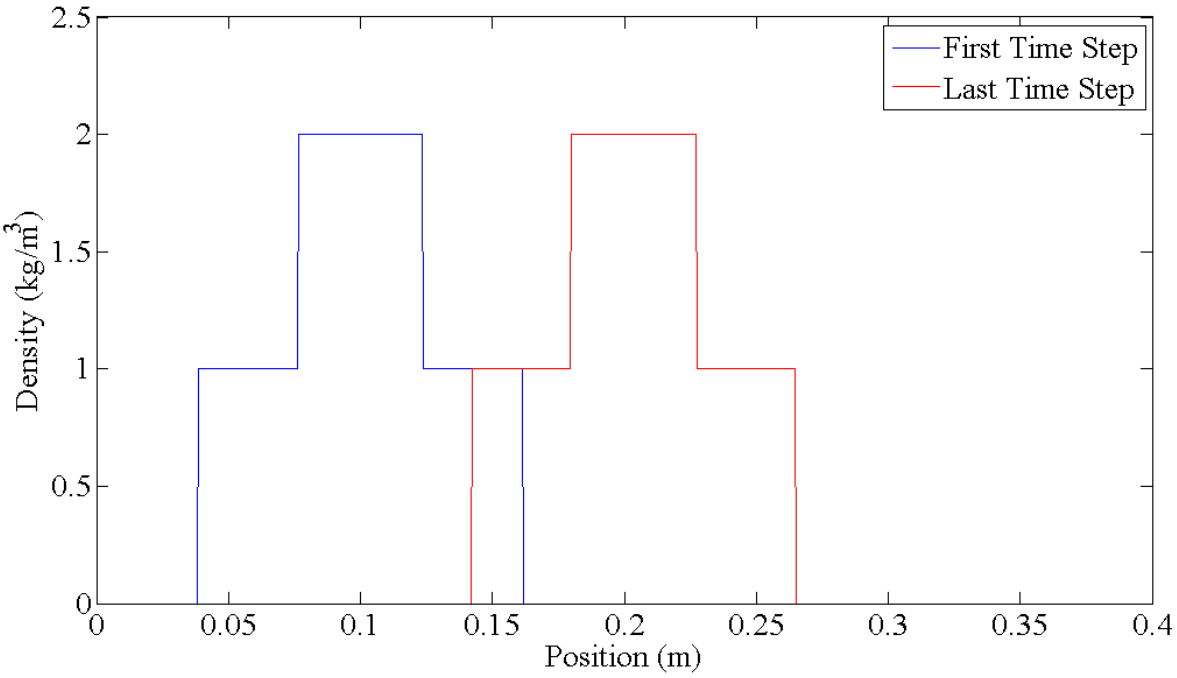


Figure 3.12: The advection test produces a density square wave that does not change as it propagates.

3.2.3 Shock Tube Test

The shock tube tests the ability of a code to capture shocks, shock propagation speed, rarefaction waves, and contact discontinuities. This test consists of an initial condition marked by two regions distinguished by high and low pressure joined at a common boundary. In experiments, this is accomplished by a diaphragm which is burst to initiate the flow. As the separating diaphragm between these sections of gas is burst, a shock wave forms and propagates through the low pressure area at some velocity, W, away from the high pressure area. At the same time an expansion wave forms in the high

pressure area and moves in the opposite direction of the shock wave. The gas dynamics observed in a shock tube experiment are very similar to the behavior of a plasma railgun and, therefore, make a suitable preliminary test to perform to ensure that SPFMax properly models those typical gas dynamics.

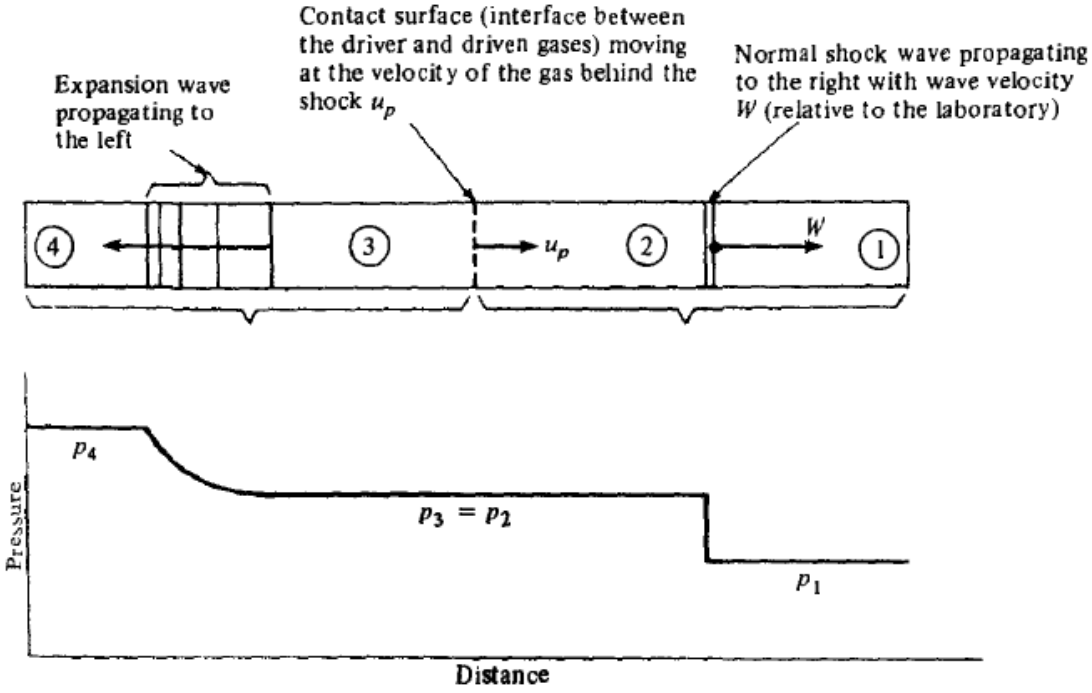


Figure 3.13: Sections of a shock tube show how pressure levels change from the front of the expansion wave to the area in front of the shock wave [45].

High and low pressure sections of a shock tube are further broken down into sections separating the expansion wave in the high pressure area and the shock wave in the low pressure area (see Figure 3.14). Behind the shock wave, gas moves at some velocity, u_p, up to the start of the expansion wave. Pressure between the expansion wave and the shock wave is also equal. However density and temperature vary between the four areas of the shock tube. Physical relationships between the four sections can be expressed mathematically starting with relating the pressure ratio between sections 1 and

4 (in front of the shock wave and in front of the expansion wave respectively) to the pressure ratio between pressures in section 1 and 2 (low-pressure area behind the shock wave) using the following expression [45]:

$$\frac{p_4}{p_1} = \frac{p_2}{p_1}\left\{1 - \frac{(\gamma_4 - 1)(a_1/a_4)(p_2/p_1 - 1)}{\sqrt{2\gamma_1[2\gamma_1 + (\gamma_1 + 1)(p_2/p_1 - 1)]}}\right\}^{-\frac{2\gamma_4}{(\gamma_4 - 1)}} \tag{3.5}$$

Where a represents sound speed, and γ represents the specific heat ratio. The temperature ratio between sections 1 and 2 can then be found using the equation [45]:

$$\frac{T_2}{T_1} = \frac{p_2}{p_1}\left(\frac{\frac{\gamma + 1}{\gamma - 1} + \frac{p_2}{p_1}}{1 + \frac{\gamma + 1}{\gamma - 1}\frac{p_2}{p_1}}\right) \tag{3.6}$$

Similarly, the density ratio can be found using [45]:

$$\frac{\rho_2}{\rho_1} = \frac{1 + \frac{\gamma + 1}{\gamma - 1}\frac{p_2}{p_1}}{\frac{\gamma + 1}{\gamma - 1} + \frac{p_2}{p_1}} \tag{3.7}$$

The velocity in section 2 (and consequently the velocity in section 3) is found using the expression [45]:

$$u_p = u_2 = u_3 = \frac{a_1}{\gamma_1}\left(\frac{p_2}{p_1} - 1\right)\left(\frac{\frac{2\gamma_1}{\gamma_1 + 1}}{\frac{p_2}{p_1} + \frac{\gamma_1 - 1}{\gamma_1 + 1}}\right)^{1/2} \tag{3.8}$$

And because p_2 and p_3 are equal, finding the pressure ratio for sections 3 and 4 is fairly straightforward [45]:

$$\frac{p_3}{p_4} = \left(\frac{p_3}{p_1}\right)\left(\frac{p_1}{p_4}\right) \tag{3.9}$$

Gas in front of and behind the expansion wave is isentropic, and other properties of those sections can be found using isentropic relationships [45]:

$$\frac{p_3}{p_4} = \left(\frac{\rho_3}{\rho_4}\right)^{\gamma} = \left(\frac{T_3}{T_4}\right)^{\gamma/(\gamma-1)} \tag{3.10}$$

The shock wave velocity can be found by, first, finding the Mach number of the shock wave using properties of sections 1 and 2 [45]:

$$M_s = \sqrt{\frac{\gamma+1}{2\gamma}\left(\frac{p_2}{p_1}-1\right)+1} \tag{3.11}$$

Then using the sound speed in section 1, the velocity of the shock wave is simply [45]:

$$W = a_1 M_s \tag{3.12}$$

The expansion wave properties are slightly more complex because local properties vary in time and space within the longer wave, whereas the shock wave produces instantaneous changes to properties in sections in front and behind the wave. Gas velocity grows linearly away from the front of the expansion wave and can be quantitatively represented as a function of time and position using the following equation [45]:

$$u = \frac{2}{\gamma+1}\left(a_4 + \frac{x}{t}\right) \tag{3.13}$$

Temperature decreases linearly across the expansion wave from section 4 to section 3 of the shock tube [45].

$$\frac{T}{T_4} = \left[1 - \frac{\gamma-1}{2}\left(\frac{u}{a_4}\right)\right]^2 \tag{3.14}$$

However, pressure and density decrease exponentially across the expansion wave [45]:

$$\frac{p}{p_4} = \left[1 - \frac{\gamma-1}{2}\left(\frac{u}{a_4}\right)\right]^{2\gamma/(\gamma-1)} \tag{3.15}$$

$$\frac{\rho}{\rho_4} = \left[1 - \frac{\gamma - 1}{2}\left(\frac{u}{a_4}\right)\right]^{2\gamma/(\gamma-1)} \tag{3.16}$$

Using MATLAB, a 1D code was developed to produce and exact ideal solution for a 30 cm shock tube scenario. 1D and 3D from a similar simulation performed using SPFMax. For both problems, argon gas, in two different thermodynamic states, is separated symmetrically at $t = 0$. The section represented by x_4, starting at $x = 0$, contains high-pressure, high-temperature gas at $T_4 = 30{,}000$ K. At the other end, x_1, argon is contained at $T_1 = 3{,}000$ K.

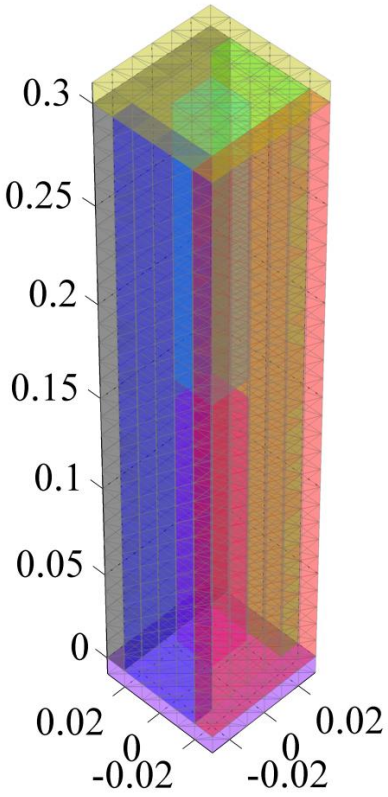


Figure 3.14: The geometry of the low resolution shock tube consists of a square bore, capped on both ends, containing two blocks of argon gas at 30,000 K and 3,000 K.

The shock tube simulation is designed to produce an interaction between high-energy and low-energy gas block. Once the simulation is initiated, high-temperature, high-pressure gas expands and produces a shock wave in the section of the low-energy gas. The shock wave propagates through the low-pressure, low-temperature end. After a certain amount of time, the four stages of gas become distinctly visible and continue to develop until a time of 30 microseconds is reached (the final time step assigned in the shock tube input files).

As with the first expansion tests, three resolutions were tested for both 1D and 3D models. Each run is compared with the 1D analytic, or ideal, solution of the shock tube problem. The results of the 1D and 3D tests are featured in the next two full-page figures. Examining the results of 1D and 3D simulations, low-resolution tests produce large numerical errors in the areas of the expansion and shock waves. Error, in the form of plot deviations from the analytic solution, is much smaller with the medium- and high-resolution tests. Error was calculated using the following expression:

$$\text{Error} = \sum_{i=1}^{n} \left(\frac{|v - v'|}{v} \right) \frac{1}{N} \tag{3.17}$$

In the Equation 4.1, v and v' represent the variable points in the exact solution and model solution respectively, while N represents the total length of the axis along which the model is observed.

Table 3.2: 1D shock tube error

	Number of Particles	Temperature	Density	Pressure
Low Resolution	270	10.27%	12.97%	21.86%
Medium Resolution	900	3.53%	6.82%	6.48%
High Resolution	3,006	2.40%	5.38%	4.82%

Table 3.3: 3D shock tube error

	Number of Particles	Temperature	Density	Pressure
Low Resolution	270	10.81%	13.83%	23.17%
Medium Resolution	900	3.99%	7.10%	6.57%
High Resolution	3,006	2.59%	5.04%	4.29%

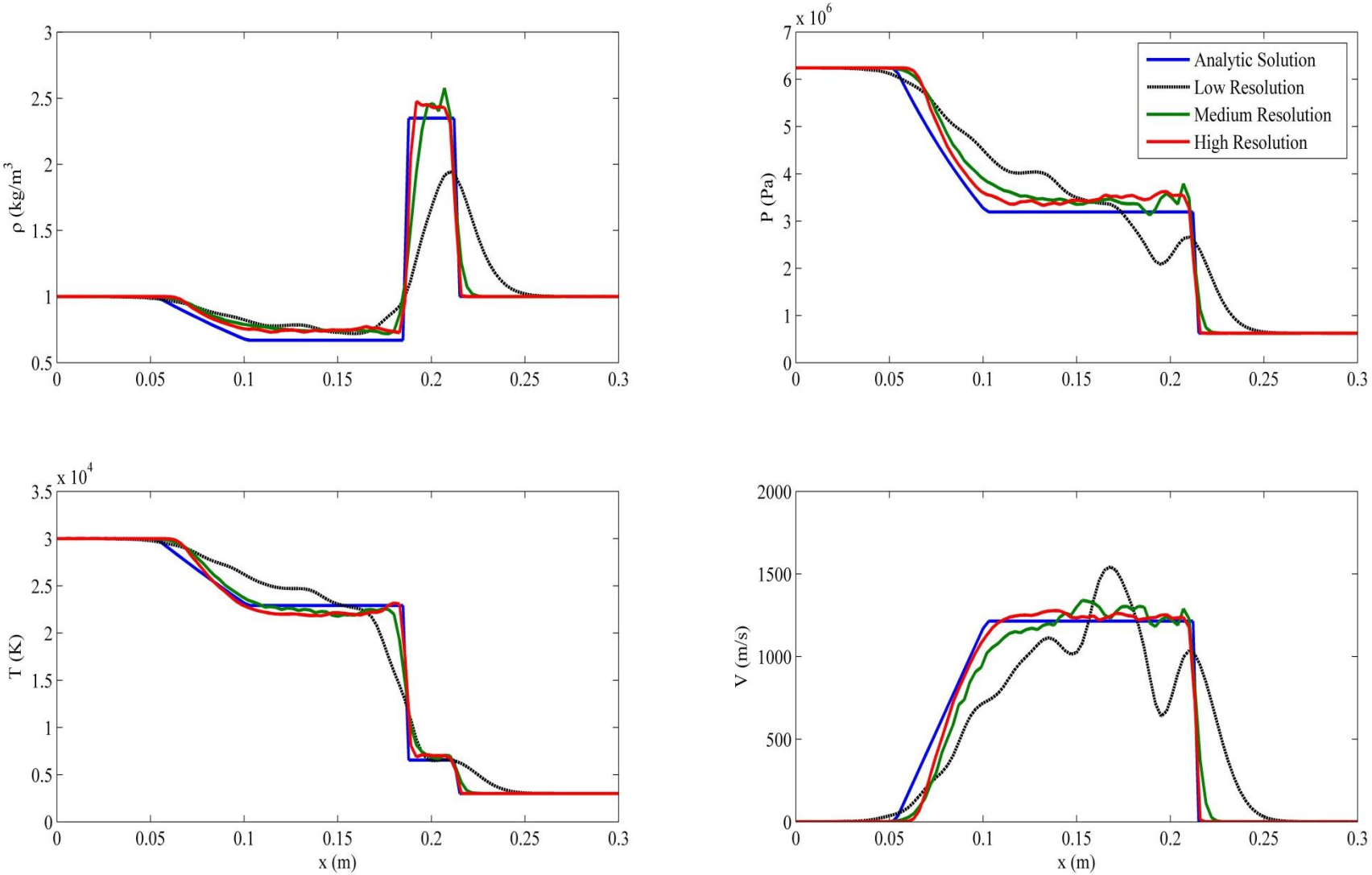


Figure 3.15: 1D results of the SPFMax shock tube show improved accuracy against the analytic solution as the resolution increases.

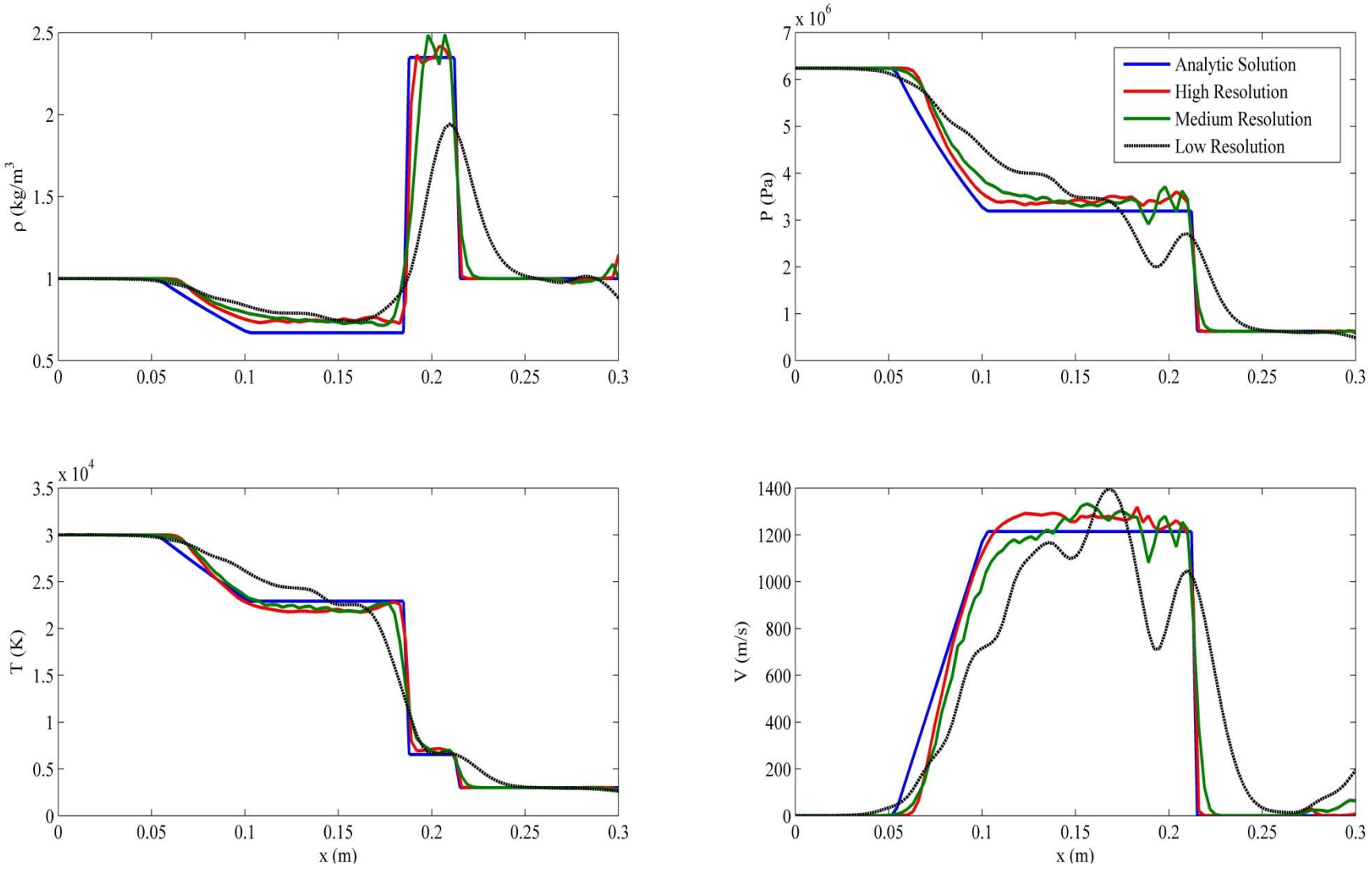


Figure 3.16: 3D resolutions of the shock tube simulation show some numerical errors not found in the 1D runs, likely due to a few particles escaping the bore through a small space between the cap and walls.

CHAPTER 4 CONCLUSIONS

In this work, a series of high-energy fluid particle tests have been conducted to form the basis of future plasma railgun modeling, using a smoothed-particle hydrodynamics approach. For the past several years of development, SPFMax has undergone several iterations and numerous testing scenarios that have gradually improved its effectiveness and accuracy, eventually leading to a difinitive fluid model and the path forward to a working circuit model.

4.1 Conclusions

Plasma experiments produced by pulsed power involve external circuits coupled to electrodes in which the plasma completes the circuit, and are utilized in plasma propulsion and some inertial fusion concepts. The plasmas are characterized by vacuum/plasma interfaces and interactions with electrode walls. Traditional magnetohydrodynamic methods utilized to model these plasmas have limited applicability, especially at the vacuum/plasma interface. That has motivated the development of a Lagrangian fluid approach to developing a new 3D code to specifically model these plasma devices (SPFMax), in which smooth particle hydrodynamics solving the fluid equations is coupled with the finite difference time domain solving Maxwell's equations method.

As part of the development of SPFMax, in this thesis a set of test cases relevant to railguns were devised to verify the accuracy of the 3D gasdynamics modeled by

SPFMax. A 1D lumped element circuit model was first tested in order to provide some guidance on the choice of initial conditions for these tests, and comparisons between this model and a 2D MHD model helped identify how such a 1D test could be utilized in future verification of the coupled circuit/electromagnetic field/fluid solvers in SPFMax.

The first set of gasdynamic tests demonstrated gas expansion in a long, open, square bore. A block of gas, initially at rest, is allowed to expand into open vacuum. The test resulted in high-temperature argon gas expanding supersonically, beneath the maximum possible rate which can be achieved by free expansion in a vacuum. This test confirmed that the gas qualitatively expands correctly and that the walls were not interfering with the flow. The latter observation is not trivial in the general purpose way in which walls were implemented, via static fluid particles conforming to any general shape representing a solid object. Walls were implemented in this way such that future tests of SPFMax with the electromagnetic field solver could include conductors, dielectrics, and other materials in addition to plasma.

The advection test was conducted to test the code's ability to advect square waves without dispersion or diffusion errors. In this test, a square block of gas surrounded by walls on all sides moves at constant velocity along the x-axis. Compared with the exact solution, the 3D SPFMax solution was good down to floating point accuracy.

Finally, shock tube simulations were performed as a precursor to future railgun tests. Initially, the shock tube model was restricted to 1D particle movement in order to replicate the 1D analytic solution as closely as possible. Three resolutions of particles were tested, comprised of a square bore capped at both ends and two thermodynamically different blocks of argon gas. Using a smaller smoothing length and increasing the

number of particles proved to decrease the calculated error, which was calculated for temperature, pressure, and density. The same tests were performed using 3D particle interaction, and the 3D SPFMax results actually showed marginal improvements in accuracy in both density and pressure (high-resolution) compared with its own 1D simulation. The greatest improvement in accuracy was produced after the lowest resolution was doubled to a medium level resolution. High resolution produced the lowest level of error as expected.

4.2 Future Work

The foundation for a completely integrated, multi-physics railgun model has been laid in this work. Future development of the SPFMax code will include the testing and debugging of the circuit model and electromagnetic field solver, which will then be paired with the fluid model to produce a completed railgun simulation. Further improvements to the model include the implementation of tabular equation of state and transport physics. Additional applications of the code to other plasma devices should be explored.

REFERENCES

[1] F. D. Witherspoon, S. Brockington, A. Case, S. J. Messer, L. Wu, R. Elton, S. C. Hsu, J. T. Cassibry, and M. Gilmore, "Development of MiniRailguns for the Plasma Liner Experiment (PLX)," presented at the 53rd Annual APS-DPP Meeting, Salt Lake City, UT, 17-Nov-2011.

[2] A. G. Lynn, E. Merritt, M. Gilmore, S. C. Hsu, F. D. Witherspoon, and J. T. Cassibry, "Diagnostics for the Plasma Liner Experiment," *Rev. Sci. Instrum.*, vol. 81, no. 10, p. 10E115, Oct. 2010.

[3] S. Brockington, A. Case, S. J. Messer, and F. D. Witherspoon, "The HyperV 8000 ug, 50 km/s Plasma Railgun for PLX," presented at the APS DPP Meeting, Providence, Rhode Island, Nov-2012.

[4] I. R. McNab, M. T. Crawford, S. S. Satapathy, F. Stefani, and T. J. Watt, "IAT Armature Development," *IEEE Trans. Plasma Sci.*, vol. 39, no. 1, pp. 442–451, 2011.

[5] A. Egeland, "Birkeland's electromagnetic gun: a historical review," *IEEE Trans. Plasma Sci.*, vol. 17, no. 2, pp. 73–82, 1989.

[6] J. Sirohi, B. Pafford, and L. L. Raja, "Experimental Characterization of the RailPAc Plasma Flow Actuator," 2013.

[7] "Development of Electromagnetic Railgun Technology for the Launch of Payloads into Suborbital Altitudes," in *55th International Astronautical Congress of the International Astronautical Federation, the International Academy of Astronautics, and the International Institute of Space Law*, 0 vols., American Institute of Aeronautics and Astronautics, 2004.

[8] R. G. Jahn, *Physics of Electric Propulsion*. Dover Publications, Incorporated, 2006.

[9] T. E. Markusic, "Current Sheet Canting in Pulsed Electromagnetic Accelerators," Dissertation, Princeton University, Princeton, New Jersey, 2002.

[10] G. C. Boynton and M. A. Huerta, "Secondary arcs in 2-D MHD numerical simulations of EML plasma armatures," *IEEE Trans. Magn.*, vol. 31, no. 1, pp. 564–569, 1995.

[11] G.-R. Liu and M. B. Liu, *Smoothed Particle Hydrodynamics: A Meshfree Particle Method*. World Scientific, 2003.

[12] G. E. Rolader, J. H. Batteh, J. D. Powell, and P. V. Desai, "Transient modelling of railgun plasma armatures," *IEEE Trans. Magn.*, vol. 25, no. 1, pp. 489–494, 1989.

[13] J. Leuer, "Electromagnetic modeling of complex railgun geometries," *IEEE Trans. Magn.*, vol. 22, no. 6, pp. 1584–1590, 1986.

[14] G. A. Shvetsov and S. V. Stankevich, “Comparison Between 2-D and 3-D Electromagnetic Modeling of Railgun,” *IEEE Trans. Magn.*, vol. 45, no. 1, pp. 453–457, 2009.

[15] Y. He, Y. Guan, G. Gao, Y. Li, X. Qiu, B. Wei, and S. Song, “Efficiency Analysis of an Electromagnetic Railgun With a Full Circuit Model,” *IEEE Trans. Plasma Sci.*, vol. 38, no. 12, pp. 3425–3428, 2010.

[16] L. Tumonis, M. Schneider, R. Kacianauskas, and V. Vadluga, “The Structural Mechanics of Rail Guns With Discrete Supports Showing the Influence of DES,” *IEEE Trans. Plasma Sci.*, vol. 39, no. 1, pp. 144–148, 2011.

[17] L. Rip, S. Satapathy, and K.-T. Hsieh, “Effect of geometry change on the current density distribution in C-shaped armatures,” *IEEE Trans. Magn.*, vol. 39, no. 1, pp. 72–75, 2003.

[18] J. Wey, E. Spahn, and M. Lichtenberger, “Railgun modeling with the P-Spice code,” *IEEE Trans. Magn.*, vol. 33, no. 1, pp. 619–624, 1997.

[19] Robert Rhodes, Dennis Keefer, and H. Thomas, “A numerical study of a pulsed plasma thruster,” in *37th Joint Propulsion Conference and Exhibit*, 0 vols., American Institute of Aeronautics and Astronautics, 2001.

[20] J. T. Cassibry, “Effects of Equation of State and Transport on Pulsed Plasma Accelerator Modeling,” *J. Propuls. Power*, vol. 23, no. 2, pp. 507–510, Mar. 2007.

[21] J. T. Cassibry, F. Thio, T. E. Markusic, and S. T. Wu, “Numerical Modeling of a Pulsed Electromagnetic Plasma Thruster Experiment,” *J. Propuls. Power*, vol. 22, no. 3, pp. 628–636, May 2006.

[22] J. T. Cassibry, Y. C. F. Thio, and S. T. Wu, “Two-dimensional axisymmetric magnetohydrodynamic analysis of blow-by in a coaxial plasma accelerator,” *Phys. Plasmas*, vol. 13, no. 5, p. 053101, May 2006.

[23] R. Crawford, D. Keefer, and R. Tipton, “Velocity limiting magnetohydrodynamic effects in railgun plasma armatures,” *IEEE Trans. Magn.*, vol. 29, no. 1, pp. 781–786, 1993.

[24] D. Motes, J. Keena, K. Womack, F. Stefani, and M. Crawford, “Thermal Analysis of High-Energy Railgun Tests,” *IEEE Trans. Plasma Sci.*, vol. 40, no. 1, pp. 124–130, 2012.

[25] N. Esposito, M. Raugi, and A. Tellini, “3-D numerical simulation of plasma armature railguns,” *IEEE Trans. Magn.*, vol. 33, no. 1, pp. 225–230, 1997.

[26] R. Critchley, C. Leyden, and A. P. J. Argyle, “The use of coupled EM-hydro finite element techniques for the design of railguns,” *IEEE Trans. Magn.*, vol. 31, no. 1, pp. 576–581, 1995.

[27] D. Kondrashov and D. Keefer, “3-D plasma armature railgun simulations,” *IEEE Trans. Magn.*, vol. 31, no. 1, pp. 634–639, 1995.

[28] J. MacGillivray, J. Peterkin, R.E., D. A. Burke, M. H. Frese, S. Frese, J. Neri, and J. W. Schumer, “MACH3: A CHSSI code for computational magnetohydrodynamics of general materials in generalized coordinates,” in *User Group Conference, 2003. Proceedings*, 2003, pp. 228–235.

[29] S. D. Frese and M. H. Frese, “Recent improvements to MACH2 and MACH3 for fast Z-pinch modeling,” in *2002 14th International Conference on High-Power Particle Beams (BEAMS)*, 2002, vol. 2, pp. 380–383.

[30] D. Lileikis and R. Peterkin, r, “Effects of azimuthal propellant injection asymmetry on MPD thruster performance using the Mach3 code,” 1995.

[31] P. G. Mikellides and M. R. LaPointe, “Magnetohydrodynamic MACH Code Used to Simulate Magnetoplasmadynamic Thrusters,” *Res. Technol. 2001*, pp. 55–57, 2002.

[32] J. Peterkin, R.E., N. F. Roderick, S. Colella, and D. E. Lileikis, “Three-dimensional hydromagnetic simulation of a high-velocity gas-puff Z-pinch,” *IEEE Trans. Plasma Sci.*, vol. 27, no. 1, pp. 118–119, 1999.

[33] J. M. Dawson, “Particle simulation of plasmas,” *Rev. Mod. Phys.*, vol. 55, no. 2, pp. 403–447, Apr. 1983.

[34] Z. Qian, P. Wang, and Z. Du, “Three Dimensional Numerical Study of Plume Characteristics of a Pulsed Plasma Thruster,” 2008.

[35] A. Case, S. Messer, S. Brockington, L. Wu, F. D. Witherspoon, and R. Elton, “Merging of high speed argon plasma jets,” *Phys. Plasmas*, vol. 20, no. 1, p. 012704, Jan. 2013.

[36] C. Thoma, D. R. Welch, and S. C. Hsu, “Particle-in-cell simulations of collisionless shock formation via head-on merging of two laboratory supersonic plasma jets,” *Phys. Plasmas*, vol. 20, no. 8, p. 082128, Aug. 2013.

[37] G. Wang, A. Al-Ostaz, A. H.-D. Cheng, and P. R. Mantena, “A macroscopic-level hybrid lattice particle modeling of mode-I crack propagation in inelastic materials with varying ductility,” *Int. J. Solids Struct.*, vol. 46, no. 22–23, pp. 4054–4063, Nov. 2009.

[38] Hans Goedbloed and Stefan Poedts, *Principles of Magnetohydrodynamics*. Cambridge University Press, 2004.

[39] J. J. Monaghan, “Smoothed Particle Hydrodynamics,” *Rep. Prog. Phys.*, vol. 68, pp. 1703–1759, 2005.

[40] M. B. Liu and G. R. Liu, “Smoothed Particle Hydrodynamics (SPH): an Overview and Recent Developments,” *Arch. Comput. Methods Eng.*, vol. 17, no. 1, pp. 25–76, Mar. 2010.

[41] G. Ala and E. Francomano, “An improved smoothed particle electromagnetics method in 3D time domain simulations,” *Int. J. Numer. Model. Electron. Netw. Devices Fields*.

[42] A. Elsherbeni and V. Demir, *The Finite Difference Time Domain Method for Electromagnetics: With MATLAB Simulations*, Har/Cdr. SciTech Publishing, 2009.

[43] A. Keshtkar, S. Bayati, and A. Keshtkar, “Derivation of a Formula for Inductance Gradient Using Intelligent Estimation Method,” *IEEE Trans. Magn.*, vol. 45, no. 1, pp. 305–308, 2009.

[44] E. M. Lifshitz and L. D. Landau, *Fluid Mechanics, Second Edition: Course of Theoretical Physics Volume 6*, 2nd ed. Moscow: Institute of Physical Problems, USSR Academy of Sciences, 1987.

[45] J. Anderson, *Modern Compressible Flow: With Historical Perspective*. McGraw-Hill Education, 2003.