



VALIDATION OF A NOVEL APPROACH TO SOLVING MULTIBODY SYSTEMS USING
HAMILTON'S WEAK PRINCIPLE

THESIS

Ashton D. Hainge, Captain, USAF

AFIT/GAE/ENY/10-M10

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

AFIT/GAE/ENY/10-M10

VALIDATION OF A NOVEL APPROACH TO SOLVING MULTIBODY
SYSTEMS USING HAMILTON'S WEAK PRINCIPLE

THESIS

Presented to the Faculty
Department of Aeronautics and Astronautics
Air Force Institute of Technology
Air University
Air Education and Training Command
In Partial Fulfillment of the Requirements for the
Degree of Master of Science in Aeronautical Engineering

Ashton D. Hainge, BSME
Captain, USAF

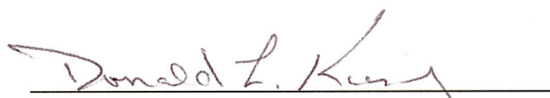
March 2010

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

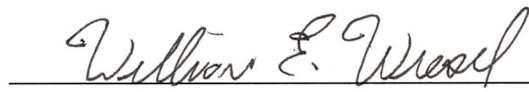
VALIDATION OF A NOVEL APPROACH TO SOLVING MULTIBODY
SYSTEMS USING HAMILTON'S WEAK PRINCIPLE

Ashton D. Hainge, BSME
Captain, USAF

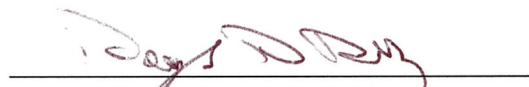
Approved:


D.L. Kunz, PhD, P.E. (Chairman)

12 March 2010
date


W.E. Wiesel, PhD (Member)

March 12, 2010
date


D.D. Decker, PhD (Member)

12 MARCH 2010
date

Abstract

A novel approach for formulating and solving for the dynamic response of multi-body systems has been developed using Hamilton's Law of Varying Action as its unifying principle. In order to assure that the associated computer program is sufficiently robust when applied across a wide range of dynamic systems, the program must be verified and validated. The purpose of the research was to perform the verification and validation of the program. Results from the program were compared with closed-form and numerical solutions of simple systems, such as a simple pendulum and a rotating pendulum. The accuracy of the program for complex systems for which there is no closed-form solution, such as a double pendulum and others, were assessed by calculating energy conservation and constraint violation. The results of this research confirm the validity of this novel approach to multibody system analysis, and pave the way for its application to increasingly complex configurations.

Acknowledgements

First and foremost, I am thankful to God for allowing me this opportunity to study a topic that I enjoy under Dr. Donald Kunz. Also, I could not complete such a task without the support of my wife and family.

Ashton D. Hainge

Table of Contents

	Page
Abstract	iv
Acknowledgements	v
List of Figures	viii
List of Tables	xiii
List of Abbreviations	xiv
 I. Introduction	 1
1.1 Background	1
1.2 Degrees of Freedom	4
1.3 Holonomic and Non-Holonomic	6
1.4 Verification and Validation	8
1.5 Problem Statement	10
1.6 Thesis Outline	11
 II. Literature Review	 12
2.1 Evolution of Hamilton's Principle	12
2.2 Maximum Coordinate Set, Minimum Coordinate Set, and Hamilton's Weak Principle	18
2.2.1 Energy preserving (EP) schemes and energy de- caying (ED) schemes	19
2.3 HWP: A Simple Approach	20
2.3.1 Coding HWP	21
 III. Methodology	 22
3.1 Overview	22
3.2 Program Output	22
3.2.1 Rigid Body Output	22
3.2.2 Joint Output	24
3.3 Energy	24
3.3.1 Potential Energy	24
3.3.2 Kinetic Energy	25
3.3.3 Energy Change	26
3.4 Virtual Work	27
3.4.1 Definition of Virtual Work for a single joint	27
3.4.2 Virtual Work Change	29

	Page
IV. Results	31
4.1 Single Pendulum	31
4.1.1 Single Pendulum Trend Analysis	32
4.1.2 45 Degree Drop Results	36
4.1.3 89 Degree Drop Results	39
4.1.4 135 Degree Drop Results	42
4.2 Double Pendulum	46
4.2.1 Double Pendulum Trend Analysis	46
4.2.2 45 Degree Drop Results	51
4.2.3 89 Degree Drop Results	55
4.2.4 135 Degree Drop Results	60
4.2.5 The Double Pendulum and Chaos	64
4.3 Triple Pendulum	65
4.3.1 Triple Pendulum Results	66
4.4 Top	66
4.4.1 Top Trend Analysis	67
4.4.2 Top - Case 1	70
4.4.3 Top - Case 2	73
4.4.4 Top - Case 3	77
V. Conclusions	82
5.1 Summary	82
5.2 Evaluation of the Software	82
5.2.1 Validation and Verification	82
5.2.2 General Evaluation	83
5.2.3 Problems	83
5.3 Recommendations	84
5.3.1 Future Work	85
5.4 Conclusions	85
Appendix A. Simple Pendulum XML Input	86
Appendix B. Double Pendulum XML Input	88
Appendix C. Triple Pendulum XML Input	91
Appendix D. Spinning Top XML Input	96
Bibliography	98
Vita	99

List of Figures

Figure	Page
1.1. Cam Follower Pair	3
1.2. Kinematic Joints	3
1.3. Planar Motion	4
1.4. Holonomic Surface	6
1.5. Holonomic Device	6
2.1. Diagram of Hamilton's Law	12
2.2. Hamilton's Law and Hamilton's Principle	13
2.3. MSA Hierarchy	17
3.1. Velocity Collection Points	29
4.1. Time Step vs Completion Time for 45 degree drop angle	32
4.2. Time Step vs Completion Time for 89 degree drop angle	32
4.3. Time Step vs Completion Time for 135 degree drop angle	33
4.4. Time Step vs RMS Energy Error for a 45 degree drop angle	33
4.5. Time Step vs RMS Energy Error for an 89 degree drop angle	33
4.6. Time Step vs RMS Energy Error for a 135 degree drop angle	34
4.7. Time Step vs Virtual Work for a 45 degree drop angle	34
4.8. Time Step vs Virtual Work for an 89 degree drop angle	35
4.9. Time Step vs Virtual Work for a 135 degree drop angle	35
4.10. Single Pendulum 45 Degree Drop Path Plot	36
4.11. Single Pendulum, 45 Degree Drop, 0.01 s time interval, 10 second span	37
4.12. Single Pendulum, 45 Degree Drop, 0.01 s time interval, 10 second span	37
4.13. Single Pendulum, 45 Degree Drop, 0.02 s time interval, 100 second span	38

Figure		Page
4.14.	Single Pendulum, 45 Degree Drop, 0.02 s time interval, 100 second span	38
4.15.	Single Pendulum 89 Degree Drop Path Plot	39
4.16.	Single Pendulum, 89 Degree Drop, 0.004 s time interval, 10 second span	40
4.17.	Single Pendulum, 89 Degree Drop, 0.004 s time interval, 10 second span	40
4.18.	Single Pendulum, 89 Degree Drop, 0.01 s time interval, 100 second span	41
4.19.	Single Pendulum, 89 Degree Drop, 0.01 s time interval, 100 second span	42
4.20.	Single Pendulum 135 Degree Drop Path Plot	43
4.21.	Single Pendulum, 135 Degree Drop, 0.004 s time interval, 10 second span	43
4.22.	Single Pendulum, 135 Degree Drop, 0.004 s time interval, 10 second span	44
4.23.	Single Pendulum, 135 Degree Drop, 0.01 s time interval, 100 second span	45
4.24.	Single Pendulum, 135 Degree Drop, 0.01 s time interval, 100 second span	45
4.25.	Time Step vs Completion Time for 45 degree drop angle	47
4.26.	Time Step vs Completion Time for 89 degree drop angle	48
4.27.	Time Step vs Completion Time for 135 degree drop angle	48
4.28.	Time Step vs RMS Energy Error for a 45 degree drop angle	49
4.29.	Time Step vs RMS Energy Error for an 89 degree drop angle	49
4.30.	Time Step vs RMS Energy Error for a 135 degree drop angle	49
4.31.	Time Step vs Virtual Work for a 45 degree drop angle	50
4.32.	Time Step vs Virtual Work for an 89 degree drop angle	50
4.33.	Time Step vs Virtual Work for a 135 degree drop angle	51
4.34.	Double Pendulum 45 Degree Drop Path Plot	51

Figure		Page
4.35.	Double Pendulum, 45 Degree Drop, 0.002 s time interval, 10 second span	52
4.36.	Double Pendulum Joint 1, 45 Degree Drop, 0.002 s time interval, 10 second span	53
4.37.	Double Pendulum Joint 2, 45 Degree Drop, 0.002 s time interval, 10 second span	53
4.38.	Double Pendulum, 45 Degree Drop, 0.003 s time interval, 100 second span	54
4.39.	Double Pendulum, 45 Degree Drop, 0.003 s time interval, 100 second span	54
4.40.	Double Pendulum, 45 Degree Drop, 0.003 s time interval, 100 second span	55
4.41.	Double Pendulum 89 Degree Drop Path Plot	56
4.42.	Double Pendulum, 89 Degree Drop, 0.002 s time interval, 10 second span	56
4.43.	Double Pendulum Joint 1, 89 Degree Drop, 0.002 s time interval, 10 second span	57
4.44.	Double Pendulum Joint 2, 89 Degree Drop, 0.002 s time interval, 10 second span	57
4.45.	Double Pendulum, 89 Degree Drop, 0.02 s time interval, 100 second span	58
4.46.	Double Pendulum Joint 1, 89 Degree Drop, 0.002 s time interval, 100 second span	59
4.47.	Double Pendulum Joint 2, 89 Degree Drop, 0.002 s time interval, 100 second span	59
4.48.	Double Pendulum 135 Degree Drop Path Plot	60
4.49.	Double Pendulum, 135 Degree Drop, 0.002 s time interval, 10 second span	60
4.50.	Double Pendulum Joint 1, 135 Degree Drop, 0.002 s time interval, 10 second span	61

Figure		Page
4.51.	Double Pendulum Joint 2, 135 Degree Drop, 0.002 s time interval, 10 second span	62
4.52.	Double Pendulum, 135 Degree Drop, 0.02 s time interval, 100 second span	62
4.53.	Double Pendulum Joint 1, 135 Degree Drop, 0.002 s time interval, 100 second span	63
4.54.	Double Pendulum Joint 2, 135 Degree Drop, 0.002 s time interval, 100 second span	63
4.55.	Chaos Test 45 deg angle - The two paths are very similar	64
4.56.	Chaos Test 90 deg angle - The two paths are very different despite a small difference in initial conditions	65
4.57.	Previously accomplished top results	68
4.58.	Previously accomplished top results Case 1, Case 2, Case 3 from top to bottom	68
4.59.	Time Step vs Completion Time for the Top	68
4.60.	Time Step vs RMS Energy Error for the top	69
4.61.	Time Step vs Virtual Work for the top	69
4.62.	Top X-Y Path for Case 1, 0.002 s time interval, 10 second span .	70
4.63.	Top Energy Change for Case 1, 0.002 s time interval, 10 second span	71
4.64.	Top Virtual Work for Case 1, 0.002 s time interval, 10 second span	71
4.65.	Top X-Y Path for Case 1, 0.002 s time interval, 100 second span .	72
4.66.	Top Energy Change for Case 1, 0.002 s time interval, 100 second span	72
4.67.	Top Virtual Work for Case 1, 0.002 s time interval, 100 second span	73
4.68.	Top X-Y Path for Case 2, 0.002 s time interval, 10 second span .	74
4.69.	Top Energy Change for Case 2, 0.002 s time interval, 10 second span	74
4.70.	Top Virtual Work for Case 2, 0.002 s time interval, 10 second span	75
4.71.	Top X-Y Path for Case 2, 0.002 s time interval, 100 second span .	76
4.72.	Top Energy Change for Case 2, 0.002 s time interval, 100 second span	76

Figure		Page
4.73.	Top Virtual Work for Case 2, 0.002 s time interval, 100 second span	77
4.74.	Top X-Y Path for Case 3, 0.002 s time interval, 10 second span .	78
4.75.	Top Energy Change for Case 3, 0.002 s time interval, 10 second span	78
4.76.	Top Virtual Work for Case 3, 0.002 s time interval, 10 second span	79
4.77.	Top X-Y Path for Case 3, 0.002 s time interval, 100 second span .	79
4.78.	Top Energy Change for Case 3, 0.002 s time interval, 100 second span	80
4.79.	Top Virtual Work for Case 3, 0.002 s time interval, 100 second span	80

List of Tables

Table		Page
1.1.	Measures of Accuracy	11
2.1.	Programming Methods Similarities and Differences	15
4.1.	Single Pendulum Constants	31
4.2.	Single Pendulum Variables	31
4.3.	Double Pendulum Constants	46
4.4.	Double Pendulum Variables	47
4.5.	Triple Pendulum Constants	66
4.6.	Triple Pendulum Variables	66
4.7.	Top Constants and Initial Conditions	67
4.8.	Top Problem Variables	67

List of Abbreviations

Abbreviation		Page
MSA	multibody systems analysis	1
DoF	degrees of freedom	4
DAEs	differential algebraic equations	6
IEEE	Institute of Electrical and Electronics Engineers	8
HWP	Hamilton's Weak Principle	12
(ODEs)	ordinary differential equations	12
EP	Energy Preserving	19
ED	Energy Decaying	19
RMS	root mean square	30
GUI	graphical user interface	83

VALIDATION OF A NOVEL APPROACH TO SOLVING MULTIBODY SYSTEMS USING HAMILTON’S WEAK PRINCIPLE

I. Introduction

This chapter explores the background and basics of multibody systems analysis (MSA). These basics include the definitions for bodies, joints, and external loads. Degrees of Freedom are outlined to provide the state of the system. Next, an example of a holonomic and non-holonomic system is given to help define them. Finally, the difference between verification and validation is defined and discussed.

1.1 Background

In the 1970s multibody systems analysis MSA developed into the preferred tool for analysis of mechanical systems. Companies looking to accurately model complex systems needed a tool to model these applications. Early applications of MSA were largely used in the automotive industry and aerospace industry; although, MSA has been used in many more practical applications since. Today, computers have revolutionized the complexity of the systems that can be modeled through MSA. A more complex system may contain hundreds of bodies linked representing a larger mechanism. These complexities require efficient computer code languages and flexible programming to accommodate a vast array of problems easily.

Multibody systems can be defined as bodies, joints, and external loads that when combined, make a mechanical system. A body is usually considered to be a rigid or flexible part of a mechanical system. An example of a body is a wing on an aircraft, a flagpole, or even the human forearm. For our purposes, only rigid bodies will be considered.

The second component of a multibody system is the joint. When two or more bodies are joined together, they are connected through a joint. The joint is defined by certain kinematical constraints that restrict the relative motion of the bodies. The

members in a mechanism are connected by kinematic joints. A kinematic joint is formed by direct contact between the surfaces of the members forming that joint. The contact between the surfaces of the members can be point contact, line contact or area contact.

The joints are classified according to the type of contact and relative motion of the members. In the case of the pendulum, the joint or constraint is the connection of the sphere with the string holding it.

There are two types of joints according to the type of contact:

1. Higher pair joint

- Cam pair

- Cam follower

2. Lower pair joint

- Rigid

- Prismatic

- Revolute

- Parallel cylinders

- Cylindrical

- Spherical

- Planar

- Edge slider

- Cylindrical slider

- Point slider

- Spherical slider

- Crossed cylinders

Higher pair joints have contact between the mating surfaces. The higher order pair is defined by motion of a point or line contact as in the case for cam pair and cam-follower as shown in figure 1.1.

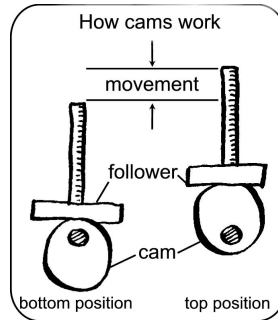


Figure 1.1: Cam Follower Pair

Figure 1.2 shows a variety of lower pair joints. The most constrained joint, of course, is the rigid joint that allows no motion and is completely constrained. Some lower pair joints have area contact between the two mating surfaces of the member's "forming joint", as in the case for slider, revolute and hinge. A spherical joint constrains relative displacements at one point where relative rotation is allowed. By definition, the spherical joint only implies three kinematical constraints of translation in the x, y, and z direction. The revolute joint only allows rotation in one relative direction and implies five kinematical constraints. The prismatic joint only allows displacement along one axis, which constrains relative rotational motion and also implies five kinematical constraints.

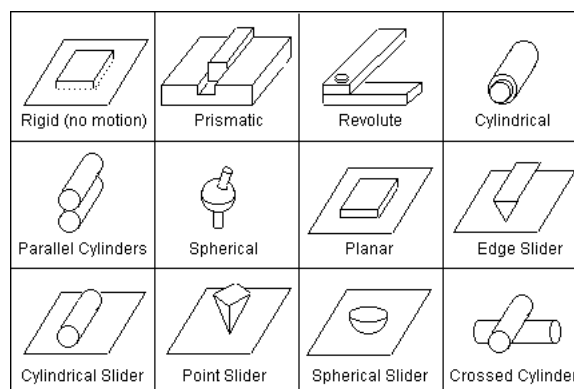


Figure 1.2: Kinematic Joints

1.2 Degrees of Freedom

The next concept behind a multibody system is degrees of freedom (DoF). The degrees of freedom denote the number of independent kinematical possibilities to move. A rigid body has six degrees of freedom in the case of general spatial motion; three are translational degrees of freedom and three rotational degrees of freedom. Real world examples of vehicles that use all DoF in spatial motion include airplanes, spacecraft, and even helicopters. These vehicles are allowed to move in an open environment.

For example, in figure 1.2, the planar motion of a body has only three degrees of freedom; one rotational and two translational degrees of freedom. All three DoF are shown in figure 1.3. The degrees of freedom are: vertical, horizontal, and the

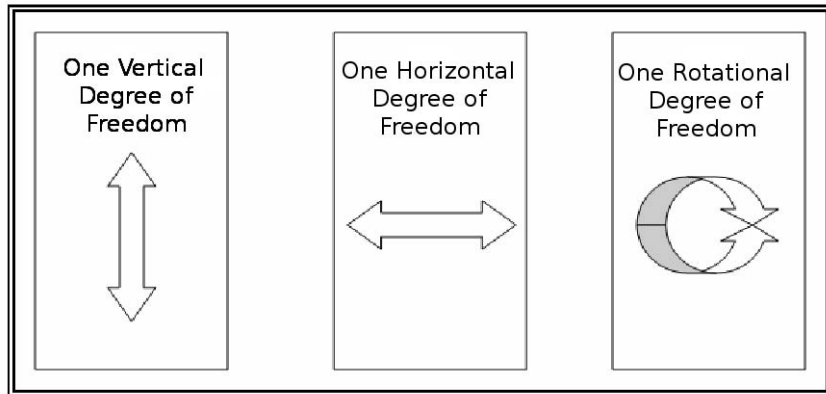


Figure 1.3: Planar Motion

rotational about the horizontal axis.

The number of DoF are defined as the number of independent coordinates required to ascertain the position of the system or its components. DoF in multibody analysis can be used to describe the state of the system and be manipulated to provide an outcome that suits the user. A joint or constraint condition implies a restriction in the kinematical degrees of freedom of one or more bodies. As stated above and in figure 1.3, there are a total of two positional coordinates, one in the vertical and one in the horizontal direction, and one rotational DoF that accounts for a positive and

negative rotation required to specify the position of the body. As the motion of the body is constrained using joints, the DoF will be reduced.

Joints, in mathematical terms, are algebraic equations that define the relative translation or rotation between two bodies. Two simple examples can be pulled from the constraints above. Referring to figure 1.2, the revolute joint can only allow one rotational movement relative to the other body about the joint. The prismatic joint permits only one DoF as shown by the constrained sliding motion. The cylindrical joint releases the rotational DoF allowing the prismatic to slide and rotate.

As a basis, all bodies or members will begin with six DoF. As members are joined together, then constraints will naturally apply and DoF will be reduced. An equation can be written describing the mechanism. Taking n to be the number of members and assuming that each member starts with a DoF of six. Then assume one member is the frame or base thus having zero DoF. The net DoF for a mechanism can be given by:

$$DoF = 6n - m \quad (1.1)$$

Where,

DoF = total coordinates in the mechanism

n = number of rigid bodies

m = number of constraint equations

Using Equation 1.1, let's define a prismatic joint. Assume we have two bodies ($n = 2$) with prismatic motion between the two bodies and one body fixed to ground. The number of constraint equations, m , for the body fixed to ground is six plus five for the prismatic joint, which only allows one direction of translational motion. So, with $n = 2$ and $m = 11$ the $DoF = 6(2) - 11 = 1$. This equation simply identifies the number of DoF for a prismatic joint having one translational DoF with the second body being fixed to ground.

1.3 *Holonomic and Non-Holonomic*

Differential algebraic systems describe a broad class of systems including differential algebraic equations (DAEs) . Many mathematical models, chemical processes, and mechanical systems with holonomic and non-holonomic constraints typically consist of DAEs . Holonomic means the constraints can be written as equations of linear

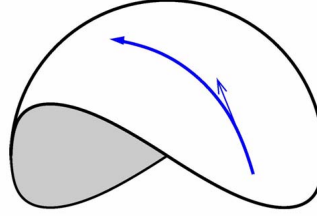


Figure 1.4: Holonomic Surface

motion that excludes translation assuming a no slip constraint. Holonomic systems are easier to characterize with mathematical models. Using Hamiltons Law, we can take advantage of defining a holonomic system using only algebraic equations in contrast to using DAEs . [6]

Holonomic and non-holonomic constraints are used widely in the field of robotics. A robot with no constraints moving along the ground, only translating from side to side, or even with arbitrary planar velocities, is holonomic. Holonomic in Latin refers to the word integer or integrable. A classic example is the unicycle. We define its motion by figure 1.5 and equation 1.2.

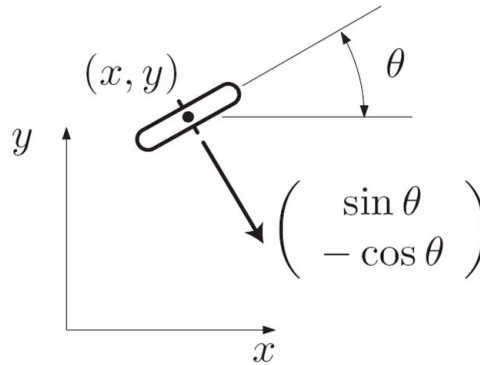


Figure 1.5: Holonomic Device

As a unicycle moves it cannot translate from side to side. In other words, its constraints $f(q, t) = 0$ are independent of $\dot{q}$. The unicycle's angular motion is described by $w_1 = (\sin\theta, -\cos\theta, 0)$. With the no slip condition applied, only translation is holonomic, where w_1 does not have to equal zero and likewise $\dot{q}$ does not have to equal zero, but the dot product of both $w_1 * \dot{q}$ must equal zero. This allows both translation and rotation as long as the dot product is zero. The total motion of the unicycle can then be expressed as:

$$\dot{q} = u_1 g_1 + u_2 g_2 \quad (1.2)$$

where,

$$g_1(q) = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, g_2(q) = \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix} \quad (1.3)$$

The first part, g_1 , of the equation deals with the angular direction of the unicycle and the second part, g_2 , deals with translation of the unicycle. So if translation was zero, the unicycle could just stay in the same place changing it's direction like a spinning coin. Conversely if the angular change was zero, the unicycle would be traveling in a straight line rolling forward or backwards.

Non-holonomic systems are harder to generalize or characterize by an equation. Various approaches have been invented to try and deal with these difficult equations based on linear matrix inequality methods or predictive control. In the case that the velocity constraint condition cannot be integrated in time in order to form a position constraint, it is called non-holonomic. In other words, the dot product $w_1 * \dot{q}$ does not equal zero. A non-holonomic is a system whose state depends on the path taken to achieve it. A characteristic of a holonomic system is that it is path dependant. The original starting point of the object will be in the same point as its original starting condition with initial conditions applied. This is not the case for a non-holonomic system. The original starting point of the object may not be in the same orientation

as its original state, as it returns to the original set of values at the start of the path. A rolling sphere is a good example of a nonholonomic system. Consider a marked point at the top of the ball and a certain path is taken from point a to point b. Then a different path is taken from point b to a. Unless the ball took the exact same route, it would not be in the same position as when it started.

1.4 Verification and Validation

Verification and validation are important steps when dealing with software intensive programs. As the programmer is writing the code, verification and validation helps to identify mistakes, create a better coding process, and ensure the written codes works as it should and for the intended use. In this analysis, verification and validation is used in the methodology to collect pertinent data to help analyze the data, draw conclusions, and verify the results. Most of the attention will be given to the results of the output of the multibody program code.

The Institute of Electrical and Electronics Engineers IEEE definition of validation is, “Confirmation by examination and provisions of objective evidence that meet the particular requirements for a specific intended use are fulfilled.” The IEEE definition of verification is, “Confirmation by examination and provisions of objective evidence that specified requirements have been fulfilled.” The terms above are not only close in spelling, but are subtly different in meaning as well. In terms of progression, validation should come first, then verification. [11]

For example, a program needs to be written to find the area of shapes. Initially, we make sure the program we are writing is valid. For instance, $X + Y$ is correct if $X = 1$ and $Y = 2$ yields a 3. Even though the intended purpose of the program was to figure areas of shapes, we see the basic principles we wanted implement in our program and ensure a robust architecture.

Furthermore, validation uses four distinct types of testing: component, integration, system, and acceptance testing. If your software is divided into different

components, then component testing ensures each segment of software performs correctly. When the components are all validated separately, it is integration testing that combines all the components and tests the overall validity of the combined system. Once the integration testing is complete, it is system testing that tests the software system as a whole for its intended purpose. Finally, it is acceptance testing that tests whether the product meets the initial acceptance criteria and if the customer accepts the product.

Verification is simply ensuring the program written satisfies the initial purpose. For example, calculate the area of a rectangle with length X and width Y . If $X = 1$ and $Y = 2$ the result should be 2. The first program ensuring $X + Y = 3$ is valid but not correct given the requirement; it is not the right program. Again, if our program has no data checking and the body of the program is $X * Y$, then if either X or Y is less than or equal to zero then the program gives no results, results in an error, or gives gibberish results. If in our initial purpose we wanted the program to handle these errors, then the validation would have failed.

Verification encompasses four different approaches; inspection, analysis, testing, and demonstration. Inspection techniques include desk checking, walkthroughs, software reviews, and technical reviews. This method of verification is the most effective for finding anomalies at the component level. Component level testing can be very costly in time and effort, but is a more exhaustive approach to finding failures in the code. The focus of this study deals with the results of the multibody system code and not the code itself. One doesn't look at the details of code when performing system level verification testing. Analysis consists of the mathematical verification of the test item itself. Execution time is included in analysis and will be collected during the test. Estimating systems resources used is another aspect that defines analysis. The third of four verification processes is testing. Inputs into the system are traced through the test item to assure that they generate the expected output values. This method is extremely helpful when you know exactly how each step of your device

should be acting. Output values should match expected values at every output point possible. This method differs from black box testing. [11]

Demonstration or black box testing does not require these intermediate points. Demonstration only requires an input be given and the output to be known. What happens inside the process is not important. Typical techniques include error guessing, boundary-value analysis, and equivalence partitioning.

1.5 Problem Statement

In order to demonstrate the capabilities and accuracy of a computer program that uses direct solutions from Hamiltons Law, simulations for several dynamic systems will be examined. It will be shown that the results of the simulations are accurate representations of the system response resulting from initial conditions and/or external loads. Comparison of these results to similar results from solutions of ordinary differential equations or differential-algebraic equations may not be sufficient to adequately assess the accuracy, since ordinary differential equations and differential algebraic equations are subject to error, as well. Therefore, other measures of accuracy, including energy conservation and constraint violation will be presented as proof of the accuracy of the simulations further defined in section 2.2.1.

The complexity of the system can rest on many factors. Can the complexity of the multibody system be increased from two bodies of 30 equations to ten bodies of 150 equations? What are the implications of increasing the complexity? Can the program handle such an increase? What are the limits in time or program structure? How many different constraints can be introduced and can they be easily added? Assumptions always affect the outcome of the system analysis. Engineers must simplify very complex systems in order to analyze them. A basis, hypothesis, or starting point is also needed to correctly predict the outcome of the system. In multibody system analysis, complex bodies need to be broken down into simple bodies with known constraints in order to be analyzed. Assumptions might be made for gravity, no slip conditions, and energy loss.

Measures of accuracy are a very important step in verification and validation. Table 1.1 offers a look at the different types of measurements that may be used in this analysis. Qualitative and quantitative refer to the nature of how the data is collected and analyzed. In this analysis, qualitative is a visual comparison of direct output of results while quantitative is the numerical collection, analysis, and comparison of data.

Table 1.1: Measures of Accuracy

Comparison to known solutions	Qualitative
Energy Conservation	Quantitative
Constraint Violation	Quantitative
Work done by constraint forces	Quantitative

1.6 Thesis Outline

This chapter has given basic definition to provide a basis for multibody system analysis (MSA) . Chapter II will review the current literature on the subject and provide more detailed information about the evolution of Hamilton’s Principle. Research methodology, including set-up of all runs, measurements used, and information on analysis is covered in Chapter III. Chapter IV details the results of the simulation runs and provides discussion of the findings. Conclusions and recommendations for continued research are laid out in Chapter V.

II. Literature Review

This chapter reviews a brief history of Hamilton's Principle. Next, Hamilton's Law and Principle are compared and contrasted. Then, the methods for measuring error in the system are discussed. Finally, insights on how Hamilton's Weak Principle was coded is given.

2.1 *Evolution of Hamilton's Principle*

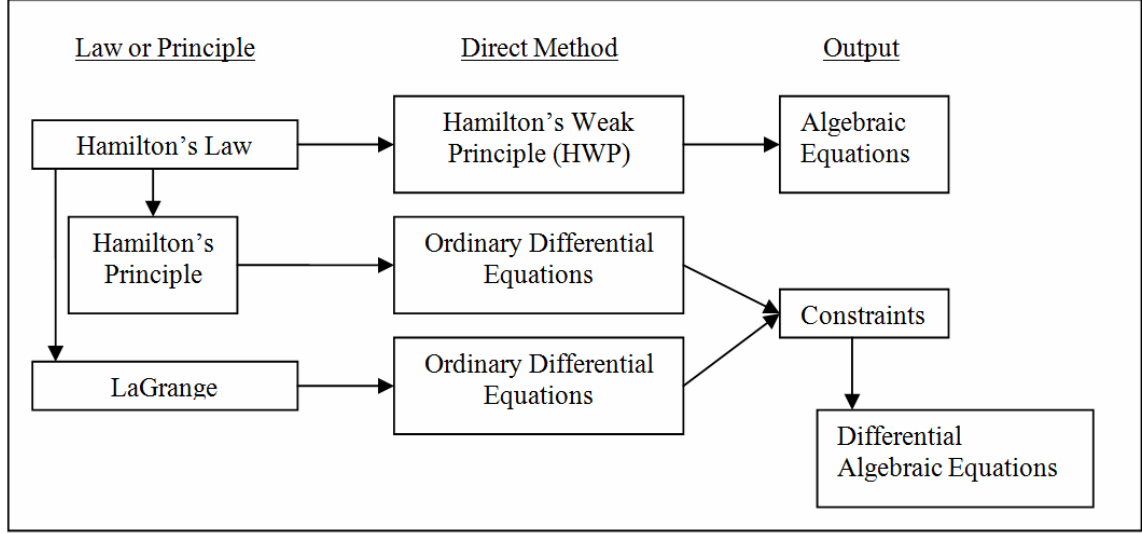


Figure 2.1: Diagram of Hamilton's Law

By 1835 William Rowan Hamilton published two papers that established what is know today as Hamilton's Principle. Hamilton's Weak Principle HWP had been discounted in the early part of the 20th century. Critics lined up to criticize the fitness of HWP , derived from Hamilton's Law, for direct calculations of dynamic systems. [10] The direct attacks and evidence against using HWP for dynamic systems as compared to using ordinary differential equations (ODEs) from Hamilton's Principle or Lagrange should have ended this technique as a possible approach. These criticisms stemmed from incorrect results that were constantly being attained by false assumptions. However, in 1982, Menahem Baruch and Richard Riff published a defensive article on proving 6n Correct formulations of Hamilton's Principle or Law. [12] So, for a one-degree-of-freedom system ($n=1$), there a six (6^1) correct formulations

of Hamilton's Law. The article states that n degrees of freedom produce 6^n correct formulations for Hamilton's Law. HWP provides a powerful alternative to numerical solutions of ordinary differential equations. [8, 10]

The Baruch and Riff article explained the correct way to solve problems using HWP. For Hamilton's Principle or Lagrange techniques, the initial state of the system, final coordinates of the system, and the time period must be given. For HWP, the initial conditions are the only requirement needed in order to find an end state for a given time. [12]

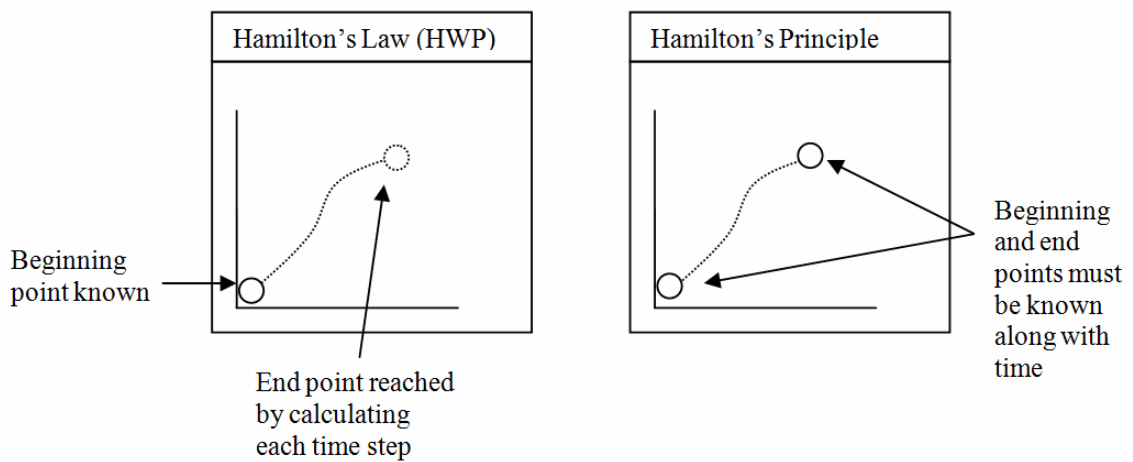


Figure 2.2: Hamilton's Law and Hamilton's Principle

To solve HWP problem correctly, the article specified the need to solve the problem using variance. Hamilton himself called it the law of varying action. HWP was designed to enact a space-time finite-element technique for direct calculations of dynamic systems. Hamilton's Law can be written:

$$\delta \int_{t_o}^{t_f} (T - V) \partial t + \int_{t_f}^{t_f} (\sum_i f_i \partial q_i) \partial t - \sum_i \frac{\partial T}{\partial \dot{q}_i} \Big|_{t_o}^{t_f} = 0 \quad (2.1)$$

Hamilton's Principle is usually found written:

$$\delta \int_{t_o}^{t_f} (T - V) \partial t + \int_{t_f}^{t_f} (\sum_i f_i \partial q_i) \partial t = 0 \quad (2.2)$$

Comparing these two equations we find the last term missing in Hamilton's Principle, equation 2.2. This is because t_f and t_o are given or known in Hamilton's Principle. Since these are known quantities the variation ∂q_i is equal to zero, therefore the term is zero. In Hamilton's Law, t_o , is known so that term's variation is zero but, t_f , at this point is unknown, so the variation is not zero and the term stays in equation 2.1. [12]

As stated above, the solution of the problem relies on its initial conditions (usually at $t = 0$) and the systems final state which is not initially known. These initial conditions are enforced as natural or weak boundary conditions. These values given are discrete values. The only piece that is left is the infinitesimally small steps from start to finish where the variation from this is equal to zero. This is where HWP is efficient and practical. As the system varies from state to state, HWP breaks down the movement into easily solvable algebraic equations. If there are strong boundary conditions they will be transformed into natural boundary conditions. Transforming strong boundary conditions to natural boundary conditions requires adjoining constraint equations. Integrating the constraint equation by parts yields the weak representation of the strong boundary condition. [12]

HWP deals only in straightforward algebraic equations where Hamilton's Principle or Lagrange's approach deals with more complex Differential Algebraic Equations DAE. It is noted that the time step between each stage effects the outcome and is constant over the whole time interval of interest.

The approach is similar to what is known as finite element solutions. Initial guesses must be made for initial conditions and as a small number of elements are solved, more accurate guesses can be made for a larger set of elements. The process gets more efficient for larger sets of elements. The larger the time step, the potentially larger the error. Also, if the problem is non-linear, trivial guesses for initial conditions may not be adequate. [5]

The key advantage of using HWP variation approach over numerical integration is that the solution is stable at all steps for linear problems. So, no matter how large

the time step a finite solution can be found without using error inducing techniques to solve the problem. With this technique there are less unknowns thus requiring less equations per degree of freedom. When dealing with multiple degree of freedom problems, this can effect processing time and the amount of work invested in solving the problem. [5]

Orthogonal properties were used to increase the efficiency of the computation of Hamilton’s Law. Hamilton’s Law using HWP only deals with algebraic equations. In order to make these equations sparse of terms or efficient, an experiment using orthogonal properties was used to cancel terms in the algebraic equations. A spring mass damper system was used to evaluate the results as compared to the standard Runge-Kutta method of solving the problem. The two answers came out nearly identical. [1]

For even the simple concepts of a spring mass damper system, the equations and calculations of multibody systems analysis can get unwieldy and complex. Once these simple designs are built for HWP, they never change. Only the inputs or initial conditions change. Over the years computers have also increased in the amount of computation, capability, and user friendliness. Object oriented approaches have helped simplify, as well as increase the efficiency of programming multibody systems, and quality of outputs. Object-oriented programming roots began in the 1960s. The difference between modular programming and object oriented programming is not in the reusable units of programming logic, but in the fact that it focuses on the sharing of data rather than processes, with programs composed of self-sufficient modules,

Table 2.1: Programming Methods Similarities and Differences

Programming Methods	Modular Approach	Object Oriented Approach
Similarities	Reusable Program Logic	
	Data is pulled from different subroutines	
Differences	Focus on module function	Collection of cooperating modules
	Subroutines are lists of tasks	Each object a “machine”

or objects, each containing all the information needed to manipulate its own data structure. The modular approach focuses on the function of a module, rather than specifically the data. This provides for code reuse, and enables self-sufficient reusable units of programming logic that also enables collaboration through the use of linked modules or subroutines. This more conventional approach, which still persists, tends to consider data and behavior separately. [1] On the other hand, an object-oriented program is a collection of cooperating modules. The modular approach looks at subroutines as a list of tasks to perform. In object oriented programming, each module is capable of receiving messages, processing data, and sending messages to other objects. These objects can be viewed as machines with a distinct role or responsibility capable of receiving, storing, processing, and sending data. The operators on these objects are closely associated with the object. [1]

In reference to multibody systems analysis there are four types of objects: bodies, constraints, loads, and motions. Bodies are defined as a mass with geometry that has a location, inertia, stiffness, and damping. The constraints are what join the bodies together. In order to have initial conditions, there must be external loads initially or continuously applied to the body. When all these objects are combined a motion is produced that fully describes the path. Of course physics and geometry is the common language that ties these objects together. [7]

Other than the point-mass, the rigid-body is the simplest in the body class not only to design and write code to, but also to implement. Other bodies include the uniform Euler-Bernoulli beam and generic bodies. The difference between them is defined by the node, mass distribution, bending stiffness, and torsion stiffness. [7]

One type of joint shown in Figure 1.2 is the spherical joint. The joint itself is not complex, but the calculations are complex due to the three constraints that are associated with it. Many more joints are defined, but the spherical joint is the main joint used in this application.

Considering the loads applied above, there are three types. The first type is the body force. A common body force is gravity and the orientation is fixed. The second load is a point load which is found at any point of application in a defined direction. The third defined load is the distributed load. The distributed load is defined not at just one point, but a load over a defined surface.

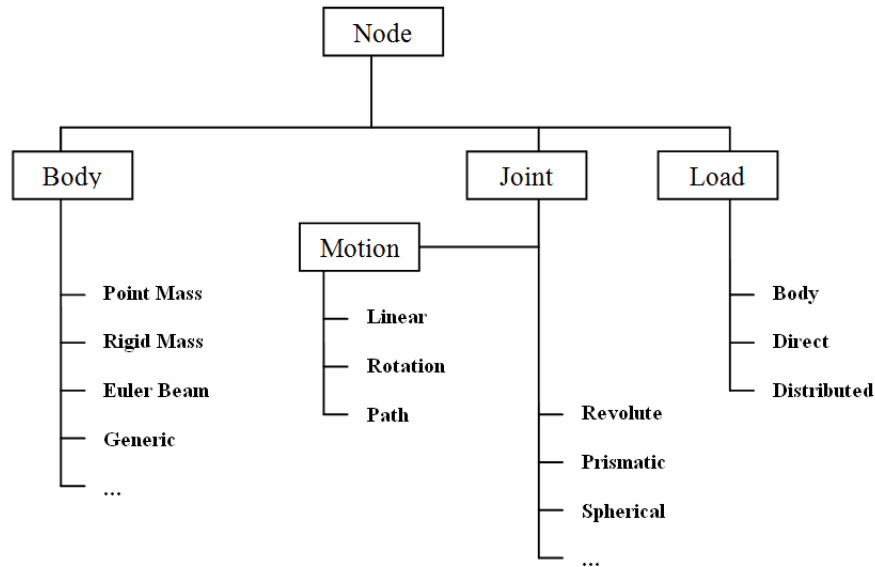


Figure 2.3: MSA Hierarchy

Figure 2.3 shows the hierarchy of how these objects relate to each other for this MSA application. This is not a concrete model for every problem, but provides a model for this application.

Improvement in object oriented architecture needs to be implemented to simplify the program structure, improve maintainability, and simplify program expansion. A unified theoretical basis for MSA that will encompass rigid bodies, elastic bodies, and bodies with an arbitrary number of generalized coordinates needs to be implemented.

2.2 Maximum Coordinate Set, Minimum Coordinate Set, and Hamilton's Weak Principle

The maximum coordinate set approach is found in many commercial MSA codes (ADAMS, DADS, etc). This approach utilizes algebraic constraint equations but still must deal with objects motions which are formulated into DAEs. Lagrange multipliers must be used to combine the constraints and object motion in order to solve the sets of equations. Where maximum coordinate set loses some of its integrity during the final solving computation, minimum coordinate set loses some integrity during the construction of the equations of motion. [8] A first approach to the analysis of constrained multibody systems is to eliminate the redundant degrees of freedom and work with a minimum set. There is still a set of DAEs that must be solved with the constraint equations that are algebraic in nature. These DAEs must be integrated in order to solve. [8]

There are many ways to solve differential equations: the zero eigenvalue method, the coordinate partitioning method (LU factorization), the singular decomposition method, the QR or householder decomposition, and the GrammSchmidt orthogonalization. [8] Unfortunately, these methods are not a practical approach when dealing with finite element formulations because they destroy the banded nature of the governing equations. Another approach involves index reduction techniques. Available results for these systems generally apply to the restricted case of index-3 DAEs . Although, in practice many of these systems often have an index greater than one. Index-3 DAEs are the most difficult form to solve, where index-2 is easier and index-1 are easiest. In this case, index reduction is achieved by enforcing a time derivative of the constraint. However, this approach does not satisfy the constraints and must be manipulated from case to case. Algorithms to solve stiff systems have been developed, however they lack error control and are sensitive to time steps. [8]

Hamilton's Principle cannot be used for direct calculations of dynamic response because it applies only to stationary systems. Instead, engineers use Hamilton's Law

and finite time elements. The principal advantage of using Hamilton's Law in lieu of Hamilton's Principle is that the intermediate step of deriving DAEs of motion are eliminated. Hamilton's Law uses algebraic equations, which are easier to solve than DAEs from Hamilton's Principle. Since all the boundary conditions are made natural or weak and HWP uses nonlinear algebraic equations, there are many ways to easily solve the set for any step in time. [8, 10]

Run times can be an important factor when computing multibody systems. It was found that for more complex systems, the run time for HWP solutions is comparable and sometimes better than ordinary differential equation solutions. Not only were the run times close, but the accuracy of both solutions were close as well. [8, 10]

2.2.1 Energy preserving (EP) schemes and energy decaying (ED) schemes.

When a holonomic constraint is used through the finite element approach on a nonlinear flexible multibody system, the equations of motion form a system of index-3 DAEs. Whereas, when non-holonomic constraints are used a system of index-2 DAEs is formed. These DAEs can be very complex and difficult to manipulate. Recently a different way of approaching this problem using energy schemes has been researched. [3]

EP refers to the total conserved energy still left in a system, where ED refers to the total energy expended by work or loss of the system. EP and ED schemes are used for nonlinear stability and preserve the total mechanical energy of the system and in the absence of externally applied loads, the system is considered to be conservative. [3] EP schemes perform rather poorly when applied to complex simulations of engineering interest, but if broken down into small steps or simple models, this method can be very valuable for a simple idealized case. Using EP and ED schemes, the constraints at the displacement and velocity levels are satisfied and accounts for the work done by the constraint forces. In the absence of externally applied loads, the preservation of the total mechanical energy of the system would be conserved across each time step. [3]

Lagrange multipliers are not used in the energy approach and do not rely on numerical dissipation for constraint stabilization. When the position and velocity constraints on a system are defined at the same time, the energy method could become over-constrained. [3] This is unlike the traditional approaches which called for such information. This is because the energy approach enforces the velocity constraint at the mid-point and the position constraint at the end point in time. Although if there are high frequencies in the system, there needs to be an ED scheme to account for those oscillations due to the equations being physically stiff. [3]

2.3 HWP: A Simple Approach

Algebraic equations associated with direct solution with implicit constraints conserved energy better, and appeared to be better suited for long simulations. The truth can be validated by solving two identical problems in two different ways and comparing the results. This is inconvenient and involves a lot more work. Two different methods will be examined; energy change and constraint violation.

DAEs formulations have three indexes of formulation; index-1, index-2, and index-3. For multiple DAE, the simplest form we can contrive is index-1. Index-1 are usually comprised of simple, smooth functions that integrate and derivate easily with no manipulation. Index-2 requires manipulation in order to solve the system. Index-3 can require user input or information from other sources in order to solve the system. Simplification of the DAEs into simpler more workable forms than an index-2 or index-3 is desired for direct calculation. For ODEs, Lagrange's equations can be used to easily derive ODEs for a particle. Lagrange multipliers were used to incorporate the kinematic equations of motion. An index-2 form can be obtained, but only by differentiating an equation, which causes the numerical system to lose accuracy and drift.

Using Hamilton's Law, six nonlinear algebraic equations are obtained. Adding two constraint equations increases the complexity, but the form is still algebraic. In order to take full advantage of Hamilton's Law, the equation needs to be cast in the

weakest possible form which results in a mixed formulation where all end conditions are natural or weak.

2.3.1 Coding HWP. As discussed in detail above, Hamilton's Weak Principle offers an alternative that results in a large set of non-linear algebraic equations. This provides a basis for an object oriented computer code that incorporates all three-dimensions of a MSA using HWP for rigid bodies with a limited set of joints. The equations obtained from Hamilton's Law were derived specifically for a single dynamic system. For multi-body system analysis, governing equations for the system are built on libraries of equations for various types of bodies and joints. [9,10] The rigid bodies and joints were hard coded and offer no object oriented interaction or changing of components as of yet. Results showed that HWP solutions were similar in accuracy as the ODE solutions. Since all the boundary conditions are natural using HWP, they don't need an initial condition. After evaluating the integrals and simplifying, only algebraic equations remain. This provides a basis for developing a program that can perform simulations of dynamic bodies having arbitrary geometries by using object oriented architecture. Since the initial state of bodies, joints, and loads are given, the solution at any state is highly dependant on the previous or initial state; the smaller the time step, the more accurate the solution. The code includes the component classes; bodies, joints, and external loads. For bodies, there are rigid bodies and ground class. Ground class is the same as rigid body except it neither has states nor residuals. States and residuals refer to the movement and orientation of the body. The only joint that was defined in the code was the spherical joint and the only load was the concentrated load; as compared to distributed load. In the initial testing of the code rigid body equations were tested considering ballistic motion of a point mass, sphere, and a cylinder. The accuracy of each solution was compared to the ideal trajectory with results performing within sufficient accuracy. [9,10]

III. Methodology

This chapter details the methods used during the course of this research. Topics include energy and virtual work and the methods used for calculating them. Also discussed are how energy and virtual work are used for measuring error and accuracy.

3.1 Overview

The methodology for this research may be broken into three stages. The first stage is calculating the energy and virtual work in the system. The second step is comparing the first calculated value of energy to the sum of every other time step value calculated in the system. For virtual work, error is calculated at each joint separately and not added together at each time step like energy.

The program was run using a Dell C6100 Latitude Notebook. CPU type was a Single-Core Intel 1.6 GHz Processor, hard disk drive size was 30GB and RAM size was 512MB. The operating system was UBUNTU version 2.28 using GNU Octave 3.2.2.1.

The multibody software being tested was not evaluated at the code level. The software itself was tested in same way a black box would be tested in a lab. Inputs enter the software and the outputs are collected, analyzed, and studied.

3.2 Program Output

3.2.1 Rigid Body Output. Each experiment involves a certain input that can be seen in sections 4.1, 4.2, 4.3, and 4.4. Of course, as these inputs are put into the software program and run, certain outputs are collected. Each body position in the multibody system are output with respect to the inertial, or ground, reference frame. These outputs are labeled x, y, and z.

A second type of output are the Wiener-Milenkovic parameters. These parameters are useful when converting a coordinate system from one orientation to another. Wiener-Milenkovic motion parameterization is singularity free for displacements and rotations of arbitrary magnitude provided that the rescaling operation is applied to

the rotation parameterization. These parameters are output by the orientation of the body with respect to the inertial reference frame. These outputs are labeled P_1 , P_2 , and P_3 .

Constructing the direction cosine matrix is a straight forward calculation of the Milenkovic parameters that are pulled from the motion data. Equations 3.1, 3.2, 3.3, and 3.4 shows the expressions used in calculating this parameter matrix. [4]

$$[C^{BA}] = \frac{p_o^2 + [p_{B/A}]^T [p_{B/A}] [1] - 2p_o [\tilde{p}_{B/A}] [\tilde{p}_{B/A}]}{(4 - p_o)^2} \quad (3.1)$$

where,

$$p_{B/A} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} \quad (3.2)$$

and,

$$p_o = 2 - \frac{1}{8} [p_{B/A}]^T [p_{B/A}] \quad (3.3)$$

and,

$$\tilde{p}_{B/A} = \begin{bmatrix} 0 & -p_{B/A} & p_{B/A} \\ p_{B/A} & 0 & -p_{B/A} \\ -p_{B/A} & p_{B/A} & 0 \end{bmatrix} \quad (3.4)$$

The next output that can be collected from the output of the system is velocity of each body. Each body's velocity is given in the body reference frame with respect to the inertial reference frame. An example of how the velocity variable is defined is

given by a subscript as shown, $v_{B/I}$, which is defined as the velocity of body B, with respect to I, or the inertial reference frame. Three velocity outputs, v_x , v_y , and v_z , are given for each body in each of the three directions defined in body coordinates.

Angular velocity and linear velocity are defined similarly. The only difference being angular velocity is defined as the rotation velocity of the body. Three angular velocity outputs, w_x , w_y , and w_z , are given for each body in each of the three directions defined in body coordinates.

3.2.2 Joint Output. Not only is rigid body output given, but so is output relating to the joints. Joint output is related to constraint forces within the system. This data is used for virtual work as discussed in section 3.4. These constraint forces are given for each of the three directions the force is acting on the joint; λ_x , λ_y , and λ_z . Each constraint force is found at the center of each joint.

3.3 Energy

All of the systems investigated are conservative, therefore there should be no change in total energy from the initial state. This is one way to measure the accuracy of the solution.

3.3.1 Potential Energy. Potential energy is the capacity for doing work that a body possesses because of its position. For example, a stone resting on the edge of a cliff has potential energy due to its position in the earth's gravitational field. If it falls, the force of gravity will act on it until it strikes the ground; the stone's potential energy is equal to its weight times the distance it can fall. The potential energy for the pendulum, double pendulum, and top were calculated using this principle by multiplying the mass times gravity times the distance above or below the reference height as shown in equation 3.5. [11]

$$PE = -m \cdot (\vec{g} \cdot \vec{r}_{B/I}) = -m \cdot \{g\}^T \{r_{B/I}\} \quad (3.5)$$

Due to the inertial coordinate system being defined as it is, the gravity is only acting in one direction. The potential energy is only calculated in the z-direction on a three dimensional x, y, and z coordinate set.

3.3.2 Kinetic Energy. Kinetic energy is energy a body possesses because it is in motion. The linear kinetic energy of a body with mass m moving at a velocity v is one half the product of the mass of the body and the square of its velocity.

$$KE_{lin} = 1/2 \cdot m \cdot (\vec{v}_{B/I} \cdot \vec{v}_{B/I}) = 1/2 \cdot m \cdot \{v_{B/I}\}^T \{v_{B/I}\} \quad (3.6)$$

The rotational kinetic energy of an object is dependent on its rotational inertia and radial velocity as shown in equation 3.7.

$$KE_{rot} = 1/2 \cdot (\vec{w}_{B/I} \cdot \vec{I}_B \cdot \vec{w}_{B/I}) = 1/2 \cdot \{w_{B/I}\}^T [I_B] \{w_{B/I}\} \quad (3.7)$$

In order to obtain the total kinetic energy of an object, one simply adds the linear and rotational elements as shown in equation 3.8.

$$KE_{total} = KE_{lin} + KE_{rot} \quad (3.8)$$

Energy can neither be created nor destroyed. The conversion of energy from potential to kinetic is seen in the above example of a rock falling from a cliff. The rock's energy of position is changed to energy of motion. In this experiment the movements of a simple pendulum are also in a constant energy fluctuation, but the total energy remains the same. As the suspended body moves upward in its swing, its kinetic energy is continuously being changed into potential energy; the higher it goes the greater becomes the energy that it owes to its position. At the top of the swing the change from kinetic to potential energy is complete, and in the course of the downward motion that follows the potential energy is in turn converted to kinetic energy. In order to obtain the total energy of the system one simply adds the total

kinetic and potential energy as shown in equation 3.9. [11]

$$E_{total} = PE + KE_{lin} + KE_{rot} = PE + KE_{total} \quad (3.9)$$

3.3.2.1 Energy of a Multibody System . When adding energy of two or more objects in a three dimensional system, matrices are introduced into the equations. The velocity in the linear kinetic energy term is multiplied with the other velocity by taking the transpose. When multiplied together with the other velocity matrix it produces a sum of the squares of each individual direction of velocity. The same applies for each direction of rotation speed, w , omega. The total energy is found by adding all the energy terms together for that moment in time.

$$E_{Total}(k) = E_{TotalBody1}(k) + E_{TotalBody2}(k) \quad (3.10)$$

3.3.3 Energy Change. Energy can be calculated at each time step for a given system. Total energy of the system from each object is needed at each singular time step due to interactions and transfers of energy that occur from body to body. The assumption in the model, due to frictionless air and joints, is that all energy is conserved. The total energy for the first time step should be the same from the 50th time step, to the 100th, all the way to the end of the system runtime. The way error was calculated for energy in these systems was to take the first entry in record and compare it to every other time on record. There will always be small variations due to reasons like computer accuracy error in calculation and program algorithm rounding errors. For example, energy at run 1 is 10 and energy at run 2 is 10.000003. The difference between the two is 0.000003 and is also the error for run 2. All runs are then plotted to return an error plot as compared to the first point in the system. For an overall standard value for each system, a RMS value is then calculated.

$$\Delta E(k) = \frac{|E_{Total}(1) - E_{Total}(k)|}{E_{Total}(1)} \quad (3.11)$$

$$E_{RMS} = \sqrt{\frac{\Delta E^2(1) + \Delta E^2(2) + \dots + \Delta E^2(k)}{k}} \quad (3.12)$$

3.4 Virtual Work

3.4.1 Definition of Virtual Work for a single joint. Virtual work on a system is the work resulting from real forces acting through a virtual displacement. In the case of simple joints, like the revolute, there are forces acting on the joint, but the joint itself does not displace. If the principle of virtual work for applied forces is used on a rigid body, the principle can be generalized for a joint: When a rigid body that is in equilibrium is subject to external forces, the total virtual work of all external forces is zero. If the total virtual work of all constraint forces acting on a joint is zero then the joint is in equilibrium.

In this test the real forces are calculated at the joints. The virtual work performed by the constraint forces must be zero because the action of those forces result in no displacement. Therefore, virtual work of the constraint forces is a measure of the enforcement of the constraints.

When the direction cosine matrix is known, the force matrix from each joint can be created. The force, at that particular joint, is applied in all three directions. The velocity of the object working on the joint is also given in three-dimensions, but is referenced in the body frame in reference to the inertia frame. This will take the coordinate transformation above as shown in equation 3.13 and multiplying by the time step, a virtual work value can be calculated for each run. [4] The dot product of the constraint forces and virtual displacements is zero. Since the virtual displacement is parallel to the total velocity of the body, the dot product of the constraint forces and velocities is also zero.

$$\overline{\delta W} = \int_{t_1}^{t_2} (\vec{f} \cdot \partial \vec{q}) dt \quad (3.13)$$

substituting,

$$\partial \vec{q} \| \vec{v}_{B/I} \quad (3.14)$$

the equation now becomes,

$$\overline{\delta W} = \int_{t_1}^{t_2} (\vec{f} \cdot \vec{v}_{B/I}) dt \quad (3.15)$$

In the case of more than one dimension, as in two-dimensional or three-dimensional motion, the force and velocity are expressed in vector form. At this point, consideration needs to be given to the coordinate set of the system. Determining what coordinate system to use is of vital importance to calculating and understanding the correct inputs and outputs of virtual work. When describing a body/joint, the coordinate system can be set somewhere in the body/joint itself or external to the body/joint. A good starting position if inside the body is usually the center of gravity; when summing forces and moments, this makes calculations easier. If the reference point is outside the body, it is usually at a joint or a ground position. When calculating virtual work, the body frame is called in reference to the inertial frame. In other words, the body's motion needs to be converted to the inertial coordinate set that is defined outside of the body. One technique used to do this is through the Wiener-Milenkovic method.

3.4.1.1 Virtual Work for Multiple Joints. Consideration needs to be given to the collection points for the velocity, force, and Wiener-Milenkovic parameters for a multibody system. Velocities are defined at the end of time steps, where forces are defined in the middle of the time steps. Since each new time step starts at each new time interval as shown in figure 3.1, collection must occur sometime during that interval. For example, step 1 occurs between zero and 0.99 seconds and displays a force of one. Step 2 occurs between one and 1.99 seconds and displays a force of two. If force data needed to be captured for step 1, data would need to be pulled at a time between zero and 0.99 seconds, not at one second as done on a single body system.

In order to reconcile both velocity and force, data is recorded at one-half of the time step for all calculations on multibody systems.

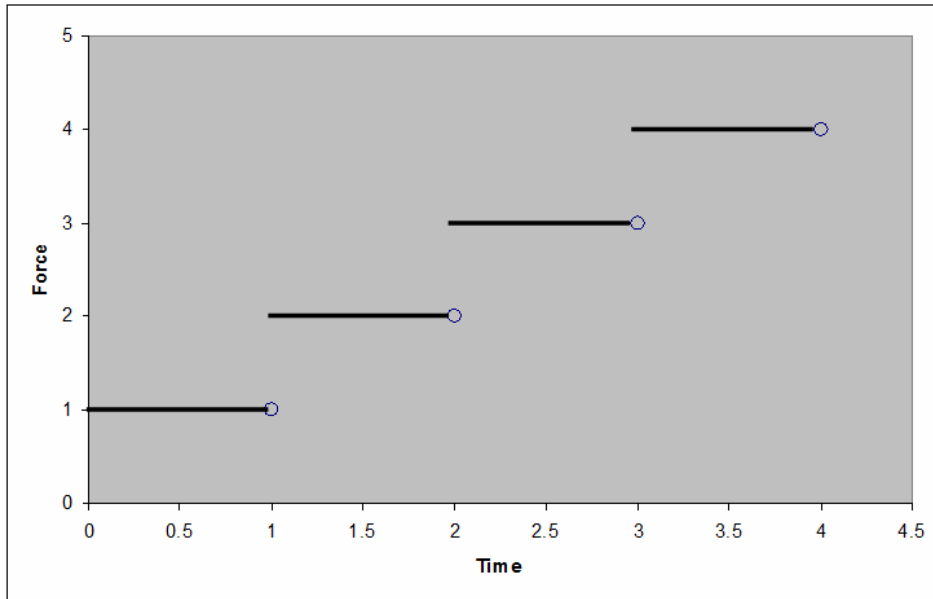


Figure 3.1: Velocity Collection Points

Equations 3.16 through 3.18 describe virtual work of multiple bodies. Velocity is transformed from body frame to the inertial frame as shown in Equation 3.17. The average velocity during the time step is given by Equation 3.18.

$$\overline{\delta W_n} = \{f_n^I\}^T \{V_n^I\} \Delta t \quad (3.16)$$

$$\{V_n^I\} = [C_n^{\frac{I}{B}}] \{V_n^B\} \quad (3.17)$$

$$\{V_n^B\} = \frac{1}{2}(\{V_k^B\} + \{V_{k+1}^B\}) \quad (3.18)$$

3.4.2 Virtual Work Change. Virtual Work can be calculated for each time step in a given joint or joints of a system. Unlike energy, total virtual work of the system is calculated at each joint separately and is not added together. Each joint

will have a separate total root mean square RMS value. The RMS should always be close to zero if the constraints are enforced. Again, there will be small variations due to reasons like computer accuracy error in calculation and program algorithm rounding errors. For an overall standard value for each system, an RMS value is then calculated.

IV. Results

All of the systems investigated are conservative, therefore there should be no change in total energy from the initial state. This is one way to measure the accuracy of the solution. The virtual work performed by the constraint forces must be zero because the action of those forces result in no displacement. Therefore, virtual work of the constraint forces is a measure of the enforcement of the constraints.

4.1 *Single Pendulum*

The first and one of the simplest examples for a multibody system is the single pendulum. This mechanism tests the program engine with simple inputs for simple planar motion. Three-dimensional motion is allowed in order to verify system integrity and monitor error. Similar inputs for different time steps are given for each case. Three different drop angles of 45 degrees, 89 degrees, and 135 degrees were considered. The angles were chosen based on the different potential energies each angle gave according to the initial orientation of the system. Other inputs include the base position, pendulum ball position, mass of the pendulum ball, the inertia of the pendulum ball, the gravity of the system, the initial velocity, the initial angular velocity, and the initial force on the pivot joint. The initial conditions, constants, and variables are shown in the table below.

Table 4.1: Single Pendulum Constants

Gravity (m/s^2)	9.8
Base Position (x,y,z)	(0,0,0)
Pendulum Ball Mass	2 kg
Initial Velocity (m/s)	0
Initial Angular Velocity (m/s)	0
Initial Force on Pivot Joint (N)	0

Table 4.2: Single Pendulum Variables

Pendulum Ball Position (deg)	deg 45, deg 89, deg 135
Sample Rate (s)	0.1, 0.04, 0.01, 0.004, 0.002
Run Time (s)	10, 100

4.1.1 Single Pendulum Trend Analysis. Data was taken in order to find trends and to validate the software program. As data was taken, the time to complete each run, energy error, RMS energy error, virtual work RMS, and virtual work was recorded for each run. The results are shown in the following figures and graphs.

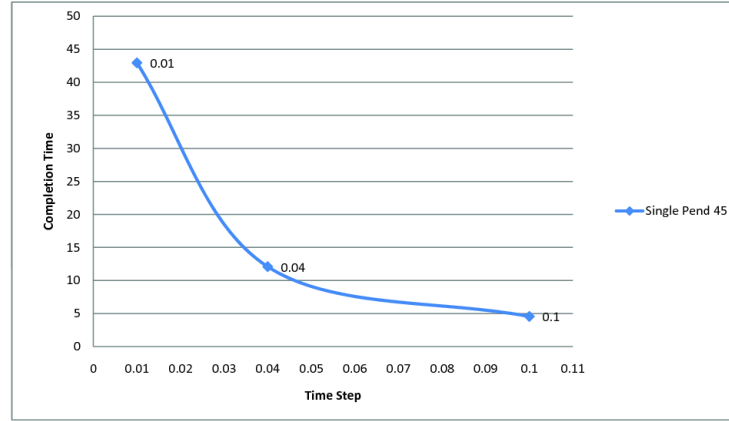


Figure 4.1: Time Step vs Completion Time for 45 degree drop angle

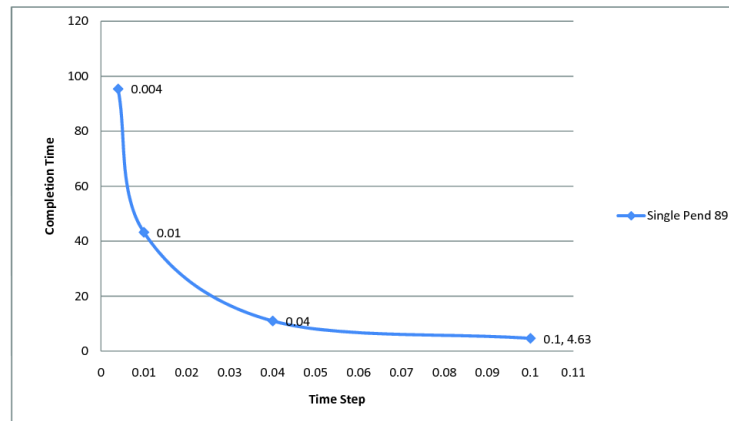


Figure 4.2: Time Step vs Completion Time for 89 degree drop angle

There are trends that can be drawn from the graphs. Figures 4.1, 4.2, and 4.3 show a predicted trend of computation time. As more points were computed, more time was needed for the computer to complete the run. There were no significant gains in efficiency as sample rate increased or decreased.

Unlike completion time, as sample increased, the energy RMS error is decreased. The rate is fairly linear as sample rate is increased with a log-log slope. For a time

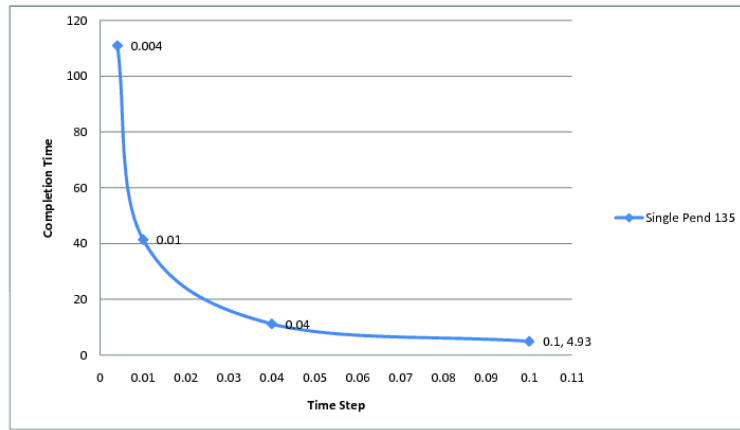


Figure 4.3: Time Step vs Completion Time for 135 degree drop angle

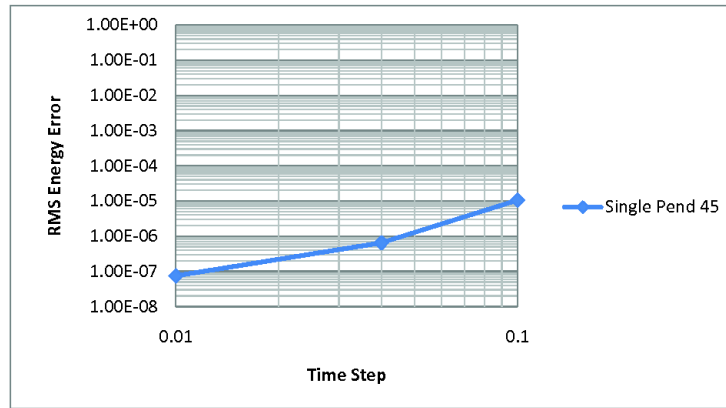


Figure 4.4: Time Step vs RMS Energy Error for a 45 degree drop angle

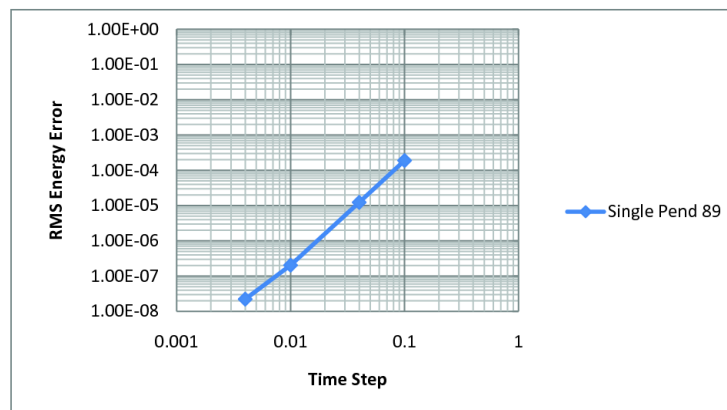


Figure 4.5: Time Step vs RMS Energy Error for an 89 degree drop angle

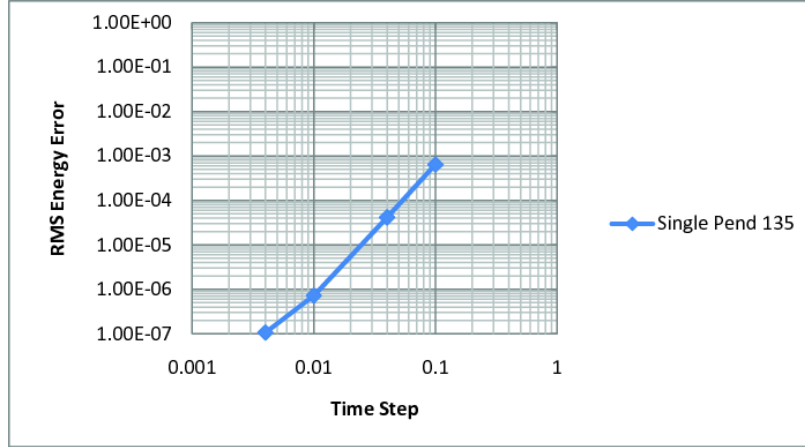


Figure 4.6: Time Step vs RMS Energy Error for a 135 degree drop angle

step difference reduction from 0.01 to 0.001, the error is reduced approximately 100 times.

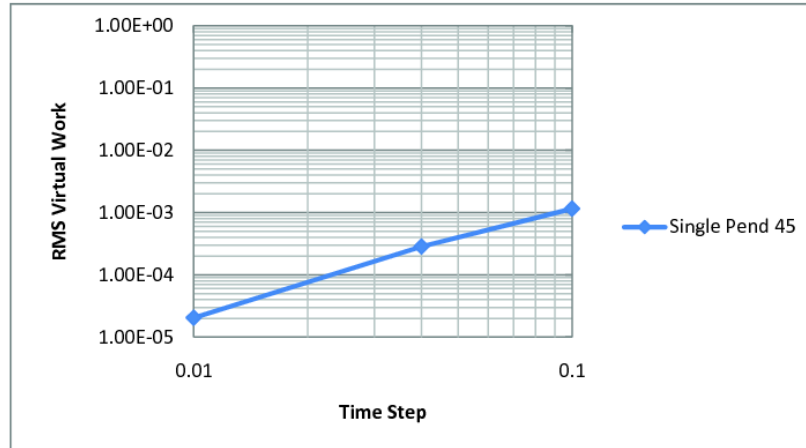


Figure 4.7: Time Step vs Virtual Work for a 45 degree drop angle

As with energy, the trends for virtual work were linear in nature on a log-log plot, with a decrease in error as sample rate increased. If less error is desired, then increasing the sample rate will give you less RMS error at the expense of completion time.

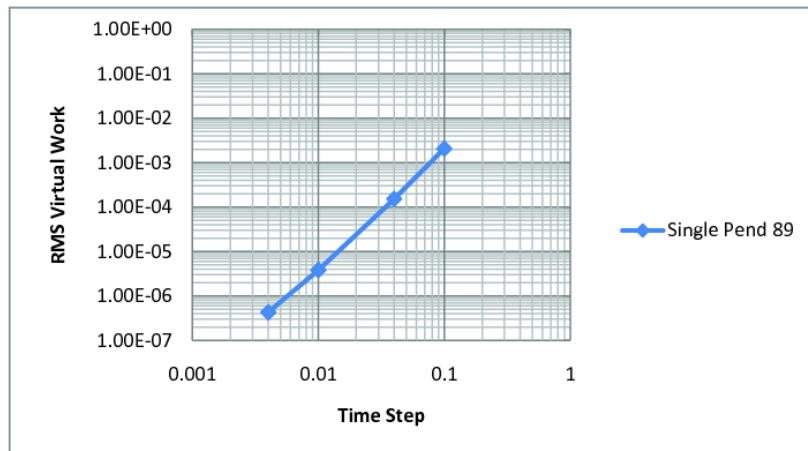


Figure 4.8: Time Step vs Virtual Work for an 89 degree drop angle

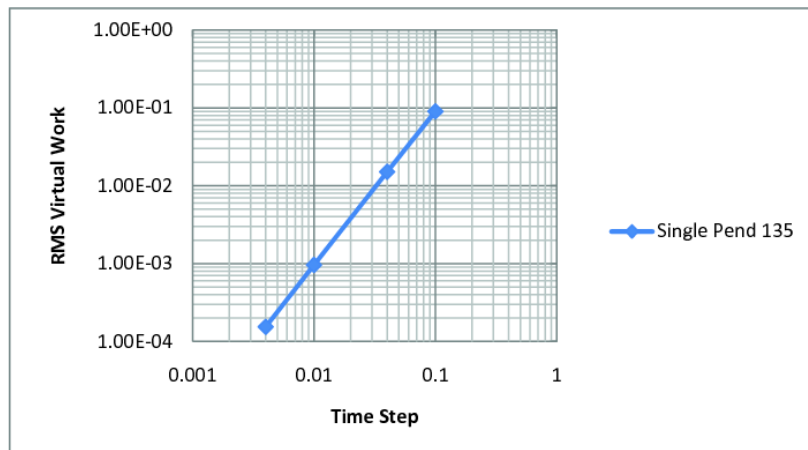


Figure 4.9: Time Step vs Virtual Work for a 135 degree drop angle

4.1.2 45 Degree Drop Results.

4.1.2.1 10 Second Run Time. For a 45 degree drop angle, the results for a nearly perfect environment were very predictable. There is no air drag or friction, just gravity acting on the pendulum ball. With this in mind, the energy difference should be near zero. The reason it should not be zero is because computers have a limited computing capacity. The error of the energy can be attributed to the error of the computer itself. Figure 4.10 shows the path of the pendulum ball.

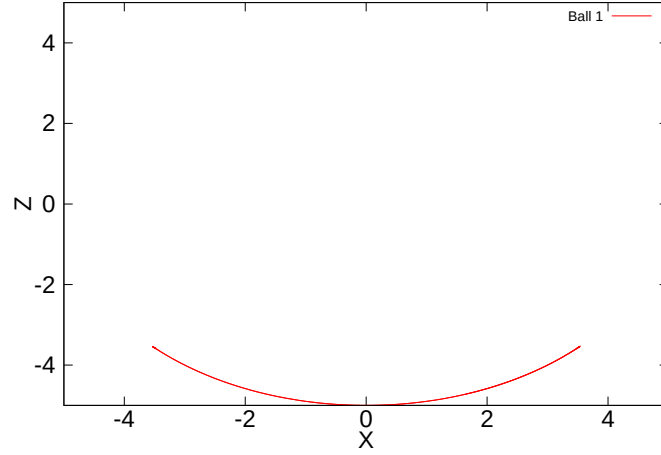


Figure 4.10: Single Pendulum 45 Degree Drop Path Plot

Figure 4.11 displays exactly what was predicted. This is a good starting point to understanding the system being represented. As the pendulum swings with no energy loss, a near zero energy difference is observed.

The second measure of the system is virtual work. When a rigid body that is in equilibrium is subject to virtual displacements, the total virtual work of external forces is zero. Figure 4.12 shows the virtual work at the base of the pendulum.

The pendulum cycle time in this instance is around five seconds. This is where the virtual work begins to repeat itself as well. When a root mean square (RMS) is calculated, the answer turns out to be very close to zero with a value of $2.04332e-05$. This value is well within the acceptable limits for short and long term simulations.

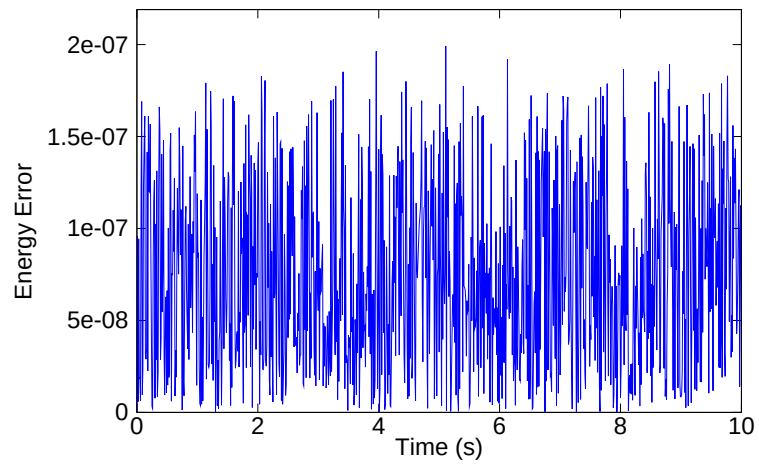


Figure 4.11: Single Pendulum, 45 Degree Drop, 0.01 s time interval, 10 second span

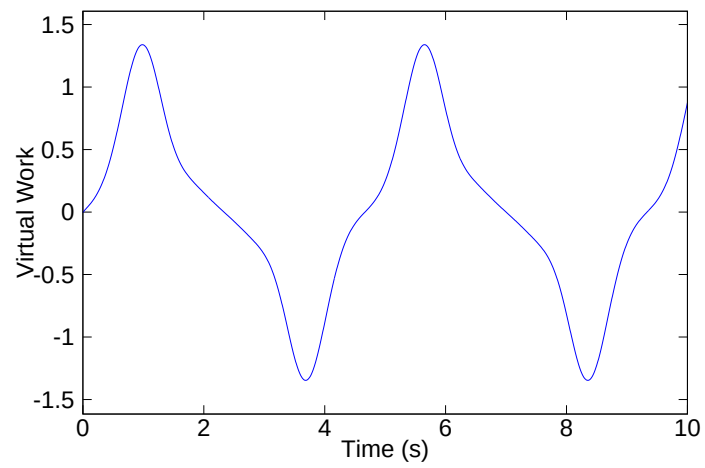


Figure 4.12: Single Pendulum, 45 Degree Drop, 0.01 s time interval, 10 second span

4.1.2.2 100 Second Run Time. A complete run was achieved at a time step of 0.01 seconds. However, this was not the case for smaller time intervals of 0.004 and 0.002. The program produced an error and could not converge using the implemented solver. There could be many reasons for these errors as will be explored later. Results up to the point of termination produced predictable results. For the longer run time of 100 seconds the time interval 0.02 was used in order to complete the run for the longer time interval.

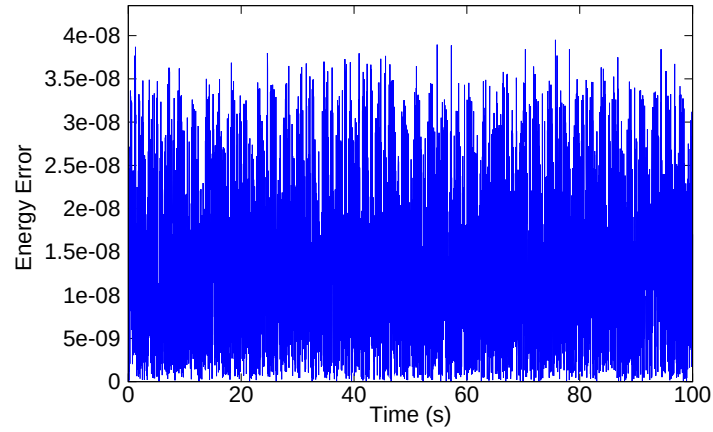


Figure 4.13: Single Pendulum, 45 Degree Drop, 0.02 s time interval, 100 second span

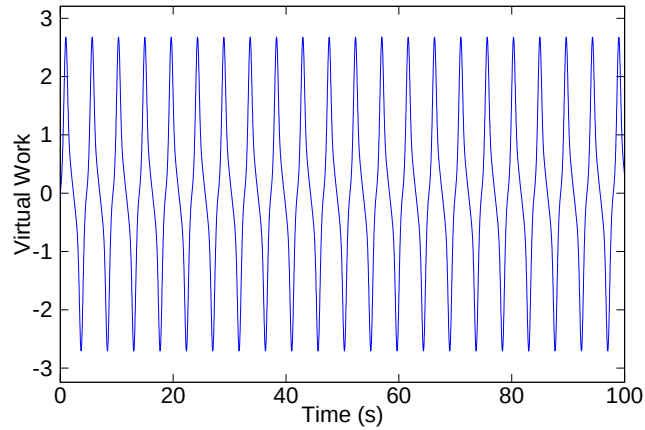


Figure 4.14: Single Pendulum, 45 Degree Drop, 0.02 s time interval, 100 second span

At 0.02 seconds the energy and the virtual work produce similar results as the shorter run time of 10 seconds. The RMS energy error is $1.37740\text{e-}08$ and virtual work

RMS error is 4.08278e-06. Both producing near zero results as expected. In fact, less error was produced for a run with the same time step but longer runtime, indicating RMS error slightly reduces as the time of the run is lengthened.

4.1.3 89 Degree Drop Results.

4.1.3.1 10 Second Run Time. For an 89 degree drop angle, the results for a nearly perfect environment were predictable as well. The same assumptions are considered with this run as with the 45 degree drop angle. With this in mind, the energy difference should be near zero as well. Figure 4.15 shows the path of the pendulum ball.

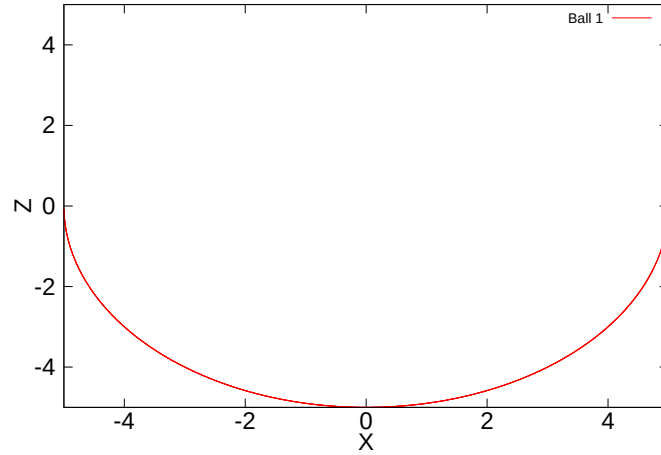


Figure 4.15: Single Pendulum 89 Degree Drop Path Plot

Figure 4.16 displays exactly what was predicted. This is a good starting point to understanding the system being represented. As the pendulum swings with no energy loss, a near zero energy difference is observed. The two and a half second cycle is seen as the energy loss repeats itself. This cycle is caused by the system repeating itself over time.

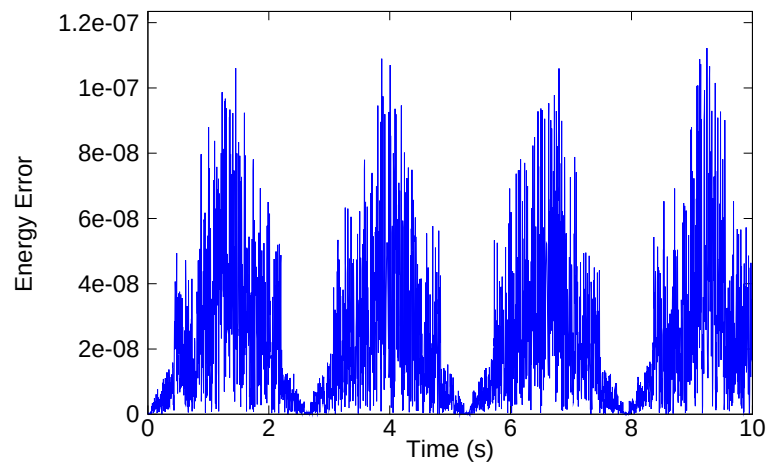


Figure 4.16: Single Pendulum, 89 Degree Drop, 0.004 s time interval, 10 second span

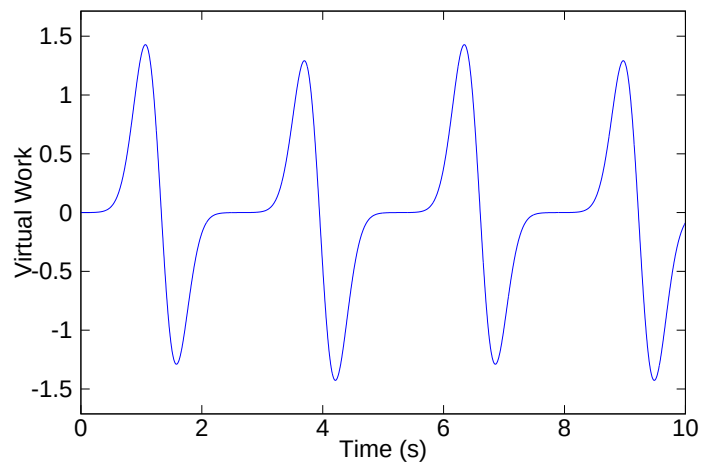


Figure 4.17: Single Pendulum, 89 Degree Drop, 0.004 s time interval, 10 second span

The second measure of the system is virtual work. When a rigid body that is in equilibrium is subject to virtual displacements, the total virtual work of external forces is zero. Figure 4.17 shows the virtual work at the base of the pendulum.

The pendulum cycle time in this instance is around two and a half seconds. This is where the virtual work begins to repeat itself as well. When a root mean square (RMS) is calculated the answer turns out to be very close to zero with a value of $4.2994\text{e-}07$. This value is well within the acceptable limits for short and long term simulations.

4.1.3.2 100 Second Run Time. A complete run was achieved at a time step of 0.004 seconds. However, this was not the case for smaller time intervals of 0.002. The program produced an error and could not converge using the implemented solver. There could be many reasons for these errors as will be explored later.

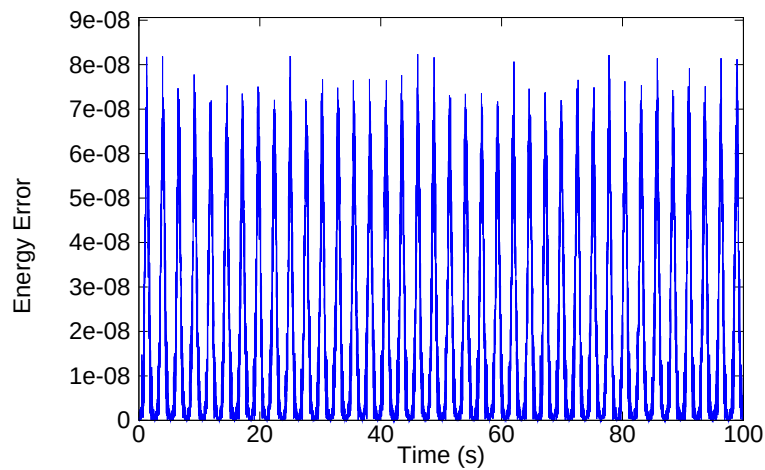


Figure 4.18: Single Pendulum, 89 Degree Drop, 0.01 s time interval, 100 second span

Results up to the point of termination produced predictable results. For the longer run time of 100 seconds the time interval 0.01 seconds was used in order to complete the run for the longer time interval. Every smaller sample rate failed to converge and crashed the program.

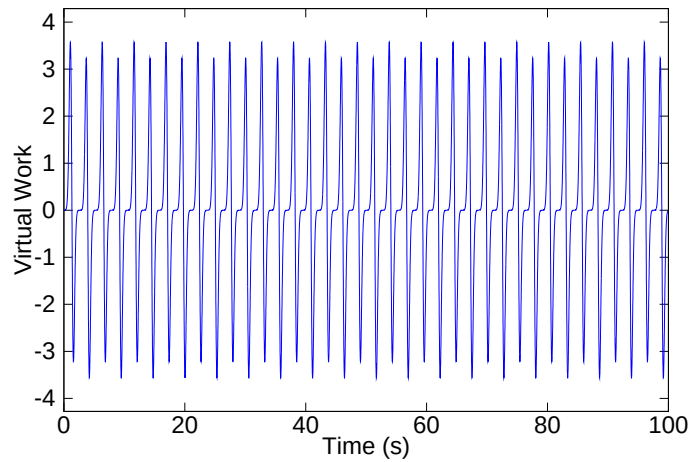


Figure 4.19: Single Pendulum, 89 Degree Drop, 0.01 s time interval, 100 second span

At 0.01 seconds the energy and the virtual work produce similar results as the shorter run time of 10 seconds. The energy RMS error value is $1.9464\text{e-}08$ and virtual work RMS error value is $1.8670\text{e-}06$. Both producing near zero results as expected. In fact, the longer the run was, the less error the program produced.

4.1.4 135 Degree Drop Results.

4.1.4.1 10 Second Run Time. For a 135 degree drop angle, the results for a nearly perfect environment were predictable as well. The same assumptions are considered with this run as with the 45 degree drop angle. With this in mind, the energy difference should be near zero as well. Figure 4.20 shows the path of the pendulum ball.

Figure 4.21 displays exactly what was predicted. Again, as the pendulum swings with no energy loss, a near zero energy difference is observed with a 7 second cycle time. This cycle is caused by the system repeating itself over time.

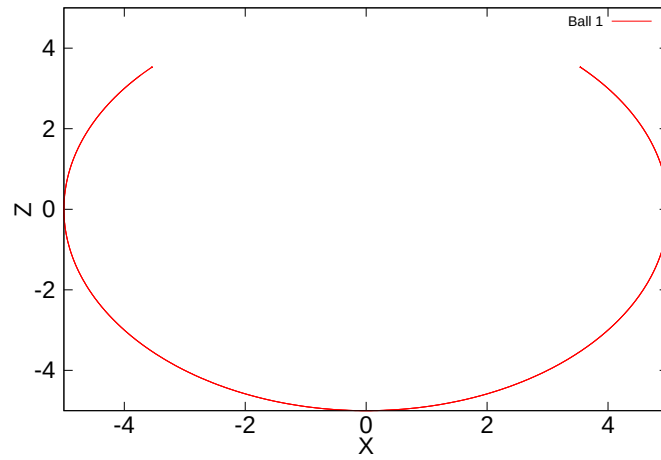


Figure 4.20: Single Pendulum 135 Degree Drop Path Plot

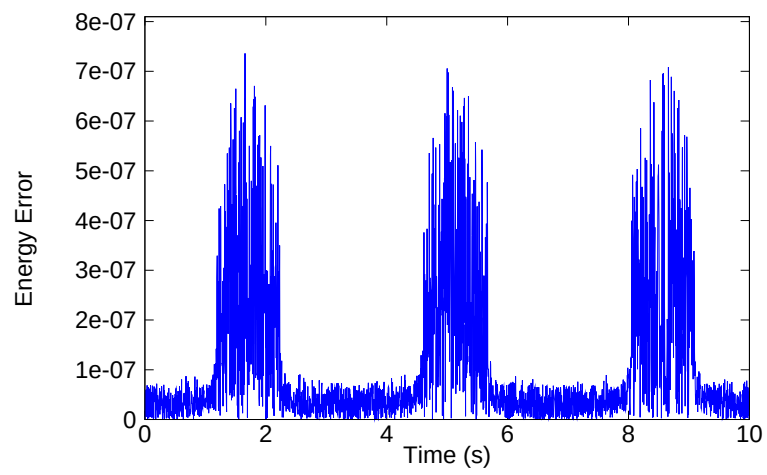


Figure 4.21: Single Pendulum, 135 Degree Drop, 0.004 s time interval, 10 second span

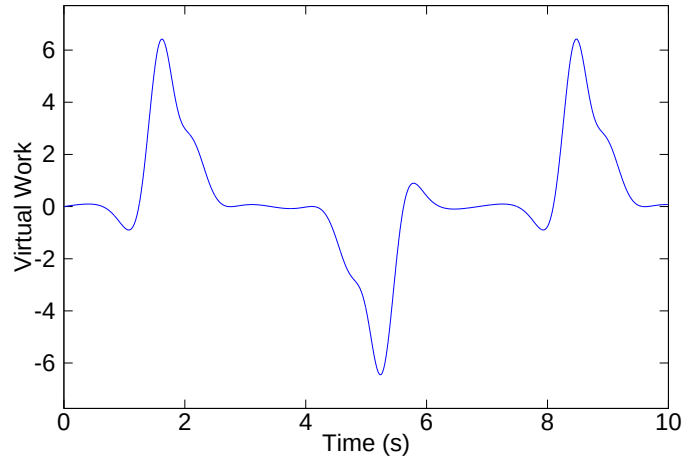


Figure 4.22: Single Pendulum, 135 Degree Drop, 0.004 s time interval, 10 second span

The second measure of the system is virtual work. When a rigid body that is in equilibrium is subject to virtual displacements, the total virtual work of external forces is zero. Figure 4.22 shows the virtual work at the base of the pendulum.

The pendulum cycle time in this instance is around 7 seconds. This is where the virtual work begins to repeat itself as well. When a root mean square (RMS) is calculated the answer turns out to be very close to zero with a value of $1.5400\text{e-}04$. This value is well within the acceptable limits for short and long term simulations.

4.1.4.2 100 Second Run Time. A complete run was achieved at a time step of 0.004 seconds. However, this was not the case for smaller time intervals of 0.002. The program produced an error and could not converge using the implemented solver.

There could be many reasons for these errors as will be explored later. Results up to the point of termination produced predictable results. For the longer run time of 100 seconds the time interval 0.01 seconds was used in order to complete the run for the longer time interval. Every smaller sample rate failed to converge and crashed the program.

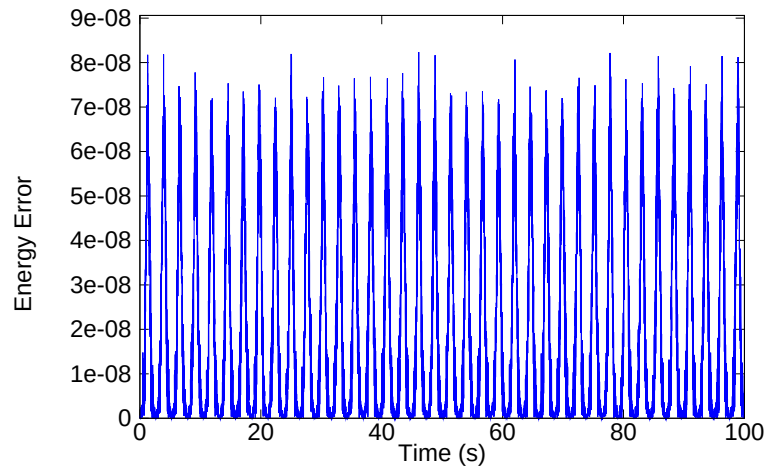


Figure 4.23: Single Pendulum, 135 Degree Drop, 0.01 s time interval, 100 second span

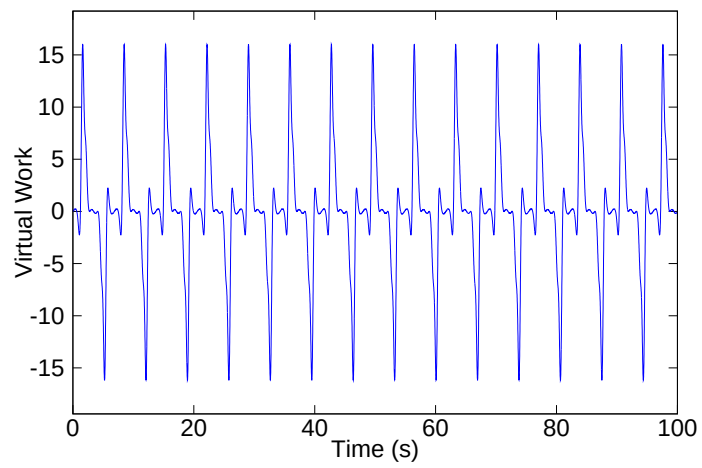


Figure 4.24: Single Pendulum, 135 Degree Drop, 0.01 s time interval, 100 second span

At 0.01 seconds the energy and the virtual work produce similar results as the shorter run time of 10 seconds. The energy RMS error value is 7.0930e-8 and virtual work RMS error value is 9.3934e-06. Both producing near zero results as expected. In fact, the longer the run was, the less error the program produced.

4.2 *Double Pendulum*

The second multibody system tested was the double pendulum. The double pendulum adds complexity by adding a second dynamic body and also a joint. The double pendulum is a very dynamic multibody system. Energy is constantly being transferred to different bodies throughout the system. Similar inputs for different time steps are given for each case. Three different drop angles of 45 degrees, 89 degrees, and 135 degrees were considered. The angles were chosen based on the different potential energies calculated for the system. For simplicity, each pendulum ball was at the same angle for each run. Other inputs include the base position, pendulum ball 1 and 2 position, mass of the pendulum ball 1 and 2, the inertia of the pendulum ball 1 and 2, the gravity of the system, the initial velocity of each body, the initial angular velocity of each body, and the initial force placed on each joint. The initial conditions, constants, and variables are shown in the table below.

Table 4.3: Double Pendulum Constants

Gravity (m/s^2)	9.8
Base Position (x,y,z)	(0,0,0)
Pendulum Ball 1 Mass	2 kg
Pendulum Ball 2 Mass	2 kg
Initial Velocity Ball 1 (m/s)	0
Initial Velocity Ball 2 (m/s)	0
Initial Angular Velocity Ball 1 (m/s)	0
Initial Angular Velocity Ball 2 (m/s)	0
Force on each Joint (N)	0

4.2.1 Double Pendulum Trend Analysis. In order to understand the system, an analysis was done. As data was taken, the time to complete each run, energy error,

Table 4.4: Double Pendulum Variables	
Pendulum Ball 1 and 2 Position (deg)	deg 45, deg 89, deg 135
Sample Rate (s)	0.1, 0.04, 0.01, 0.004, 0.002
Run Time (s)	10, 100

RMS energy error, virtual work RMS, and virtual work was recorded for each run. The results are shown in the following figures and graphs.

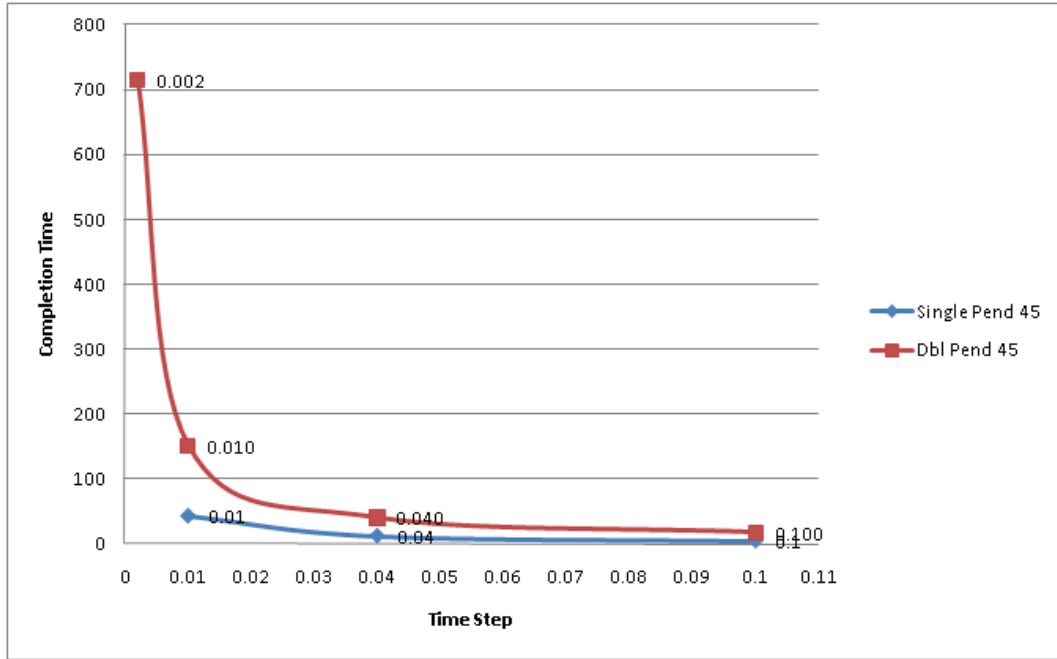


Figure 4.25: Time Step vs Completion Time for 45 degree drop angle

There are trends that can be drawn from the graphs. Does adding a second ball to the pendulum reduce the amount of relative computation compared to having one ball? For instance, if the single pendulum at 45 degrees took 4 seconds to complete, would adding a second ball take more or less than adding 4 more seconds to the computational time? The answer is shown in figures 4.25, 4.26, and 4.27. It takes approximately four times the amount of time to compute the double pendulum compared to the single pendulum. If one more reference point was available, a simple trend could be developed.

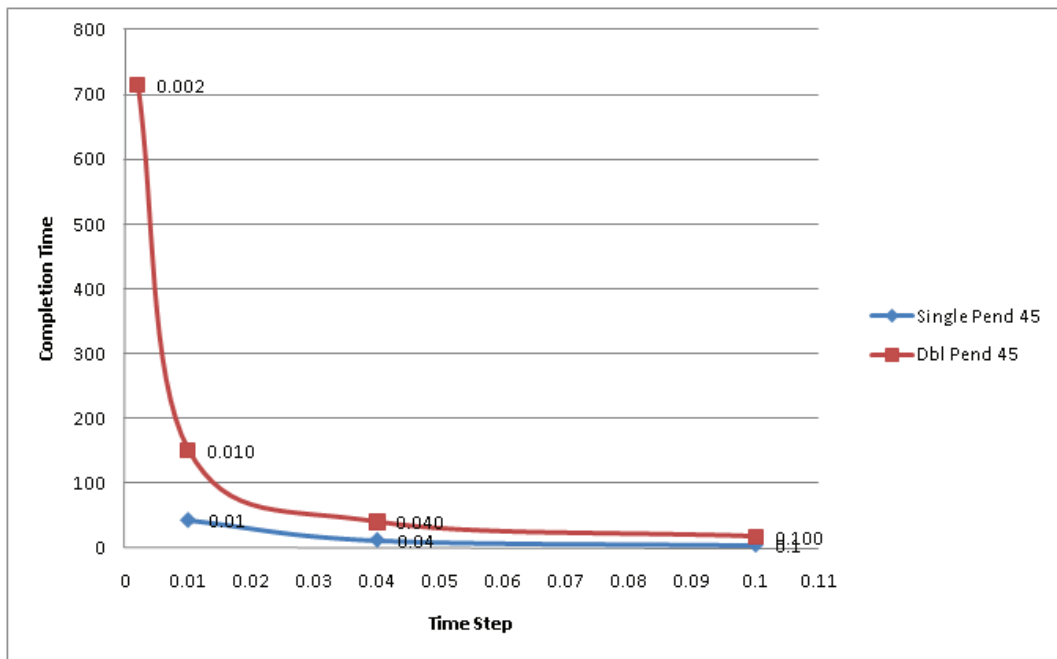


Figure 4.26: Time Step vs Completion Time for 89 degree drop angle

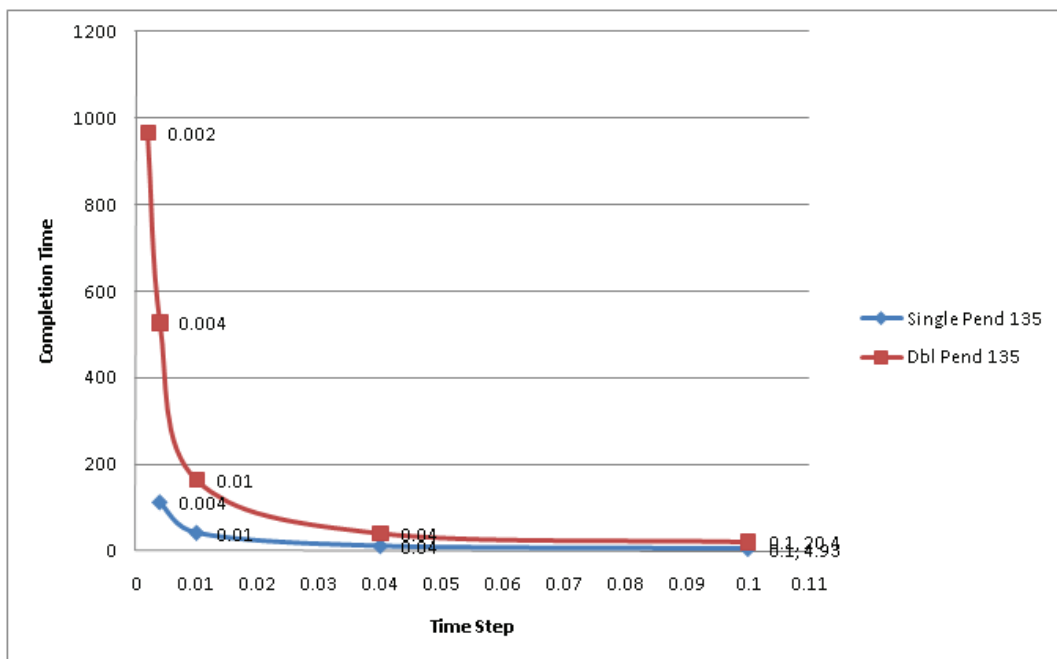


Figure 4.27: Time Step vs Completion Time for 135 degree drop angle

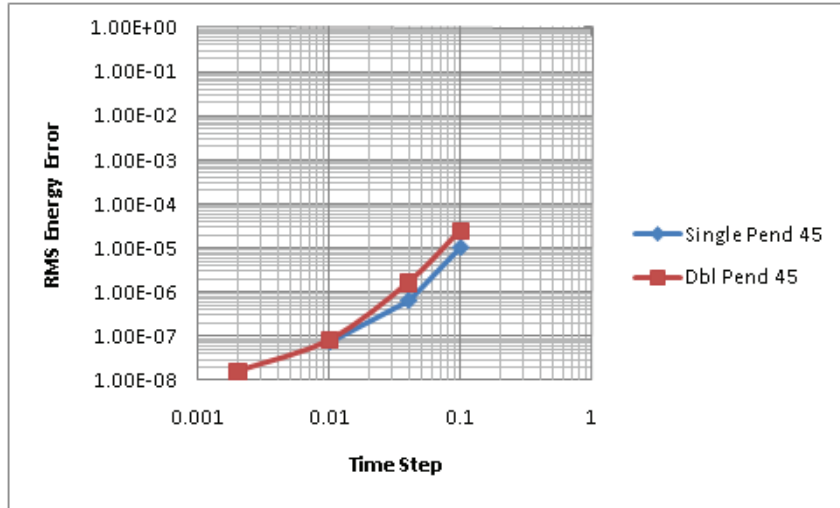


Figure 4.28: Time Step vs RMS Energy Error for a 45 degree drop angle

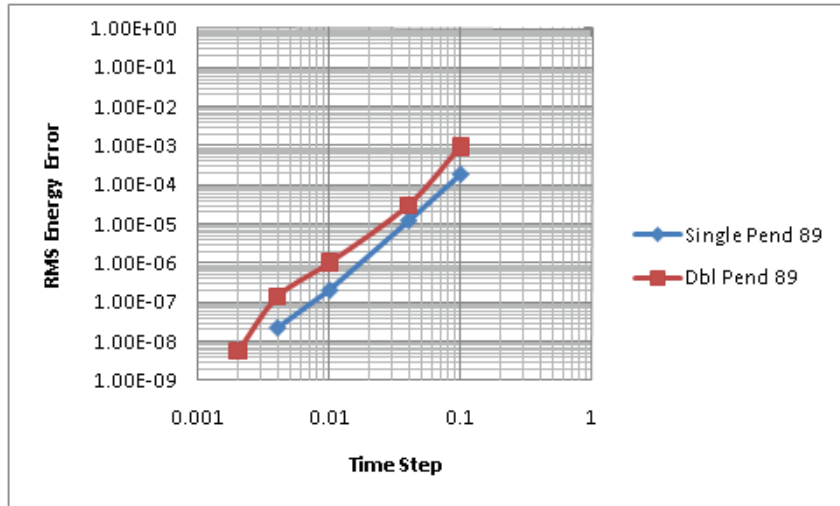


Figure 4.29: Time Step vs RMS Energy Error for an 89 degree drop angle

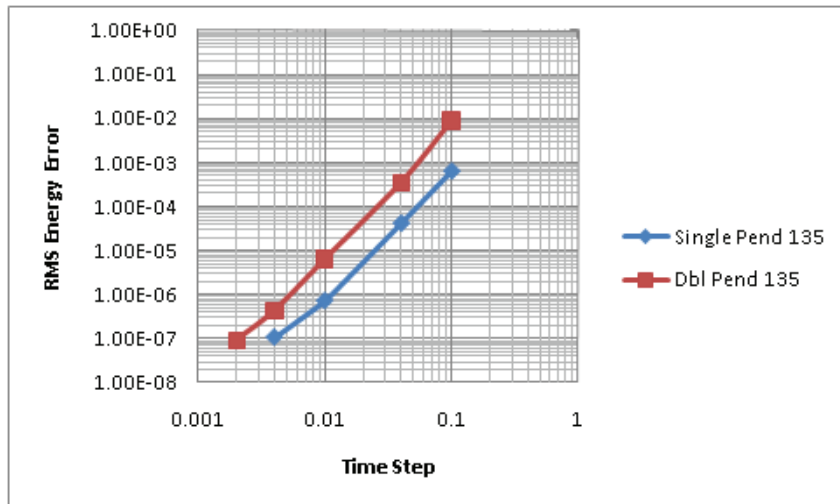


Figure 4.30: Time Step vs RMS Energy Error for a 135 degree drop angle

As sample increased, the energy RMS error is decreased. The rate is fairly linear as sample rate is increased with a log-log slope. For a time step difference reduction from 0.1 to 0.01, the error is reduced approximately 1000 times.

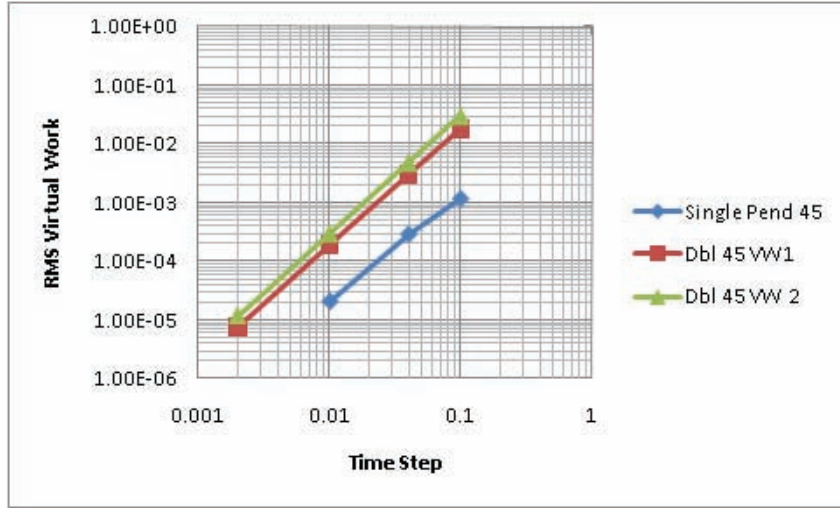


Figure 4.31: Time Step vs Virtual Work for a 45 degree drop angle

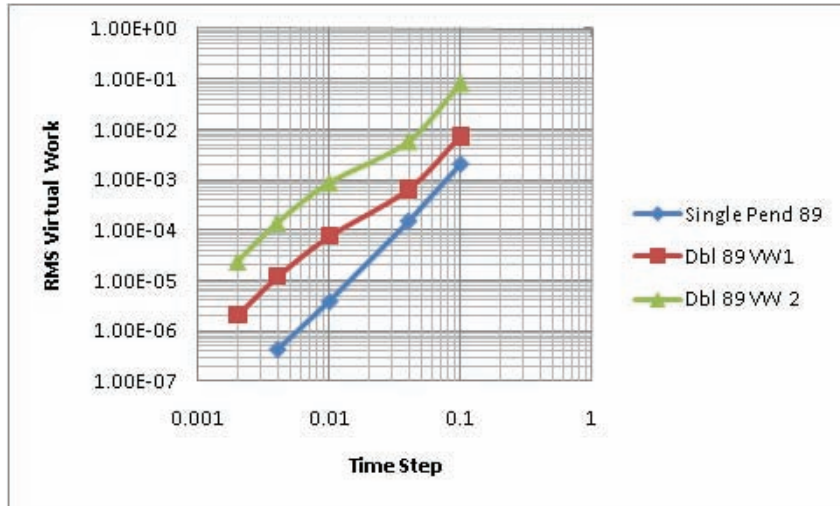


Figure 4.32: Time Step vs Virtual Work for an 89 degree drop angle

For virtual work RMS error, there were some interesting trends. On every plot, the error for the single pendulum was less than the error for any double pendulum virtual work. Also, the virtual work of the base (VW1), was always equal to or less than the error for the first joint (VW2). As with energy, the trends for virtual work were linear in nature on a log-log plot with a decrease in error as sample rate increased.

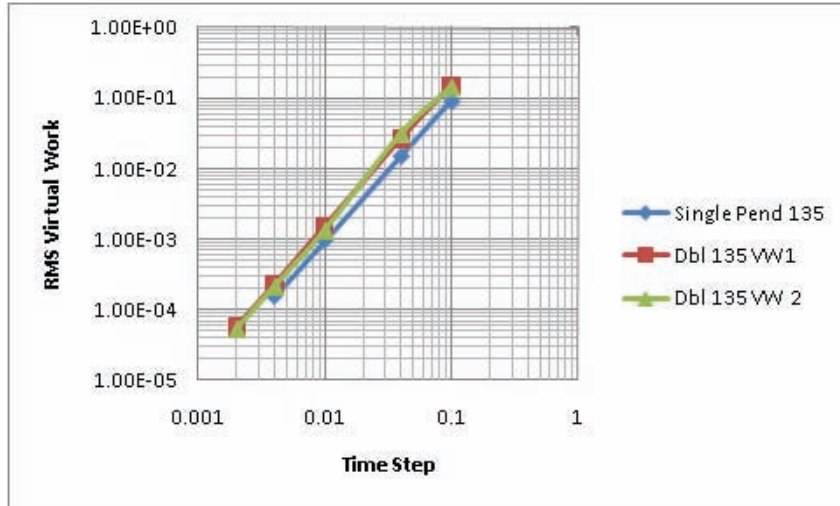


Figure 4.33: Time Step vs Virtual Work for a 135 degree drop angle

4.2.2 45 Degree Drop Results.

4.2.2.1 10 second Run Time. For a 45 degree drop angle, the results for a nearly perfect environment were very predictable. There is no air drag or friction, just gravity acting on the pendulum ball. With this in mind, the energy difference should be near zero. The reason it should not be zero is because computers have a limited computing capacity. The error of the energy relates to the error of the computer itself. Figure 4.34 shows the paths of the pendulum balls.

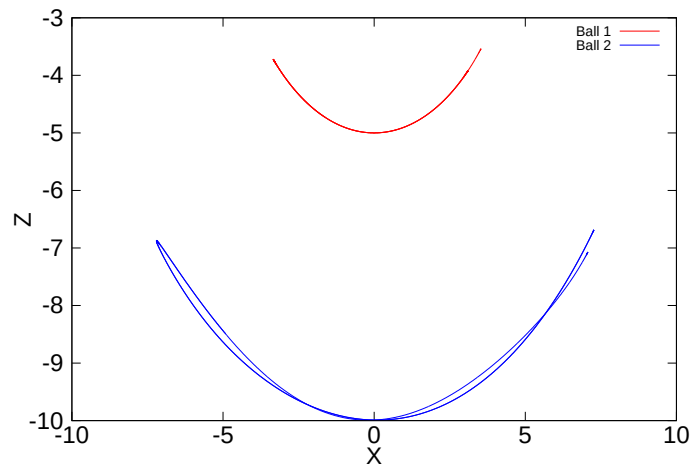


Figure 4.34: Double Pendulum 45 Degree Drop Path Plot

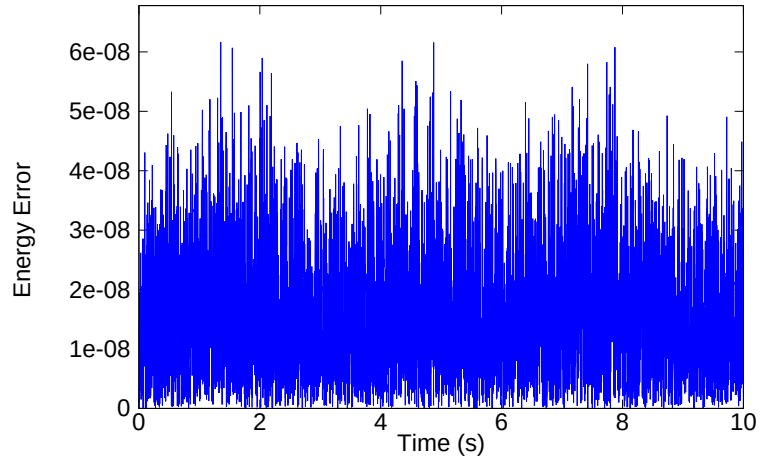


Figure 4.35: Double Pendulum, 45 Degree Drop, 0.002 s time interval, 10 second span

Figure 4.35 displays exactly what was predicted. As the pendulum swings with no energy loss, a near zero energy difference is observed. In this instance compared to the single pendulum, both balls start with and maintain a certain amount of energy as a pair.

The second measure of the system is virtual work. When a rigid body that is in equilibrium is subject to virtual displacements, the total virtual work of external forces is zero. This isn't true for each instance of time as virtual work of each body is added together, but for each individual body. Figure 4.36 and 4.37 figure shows the virtual work at each respective joint of the system.

The apparent pendulum cycle time in this instance is around five seconds. This is where the virtual work begins to repeat itself as well. When a root mean square (RMS) of the virtual work in joint 1 is calculated, the answer turns out to be very close to zero with a value of $1.7163\text{e-}05$ and $3.5257\text{e-}05$ for joint 2. The energy RMS error is $7.5496\text{e-}08$. These values are well within the acceptable limits for short and long term simulations.

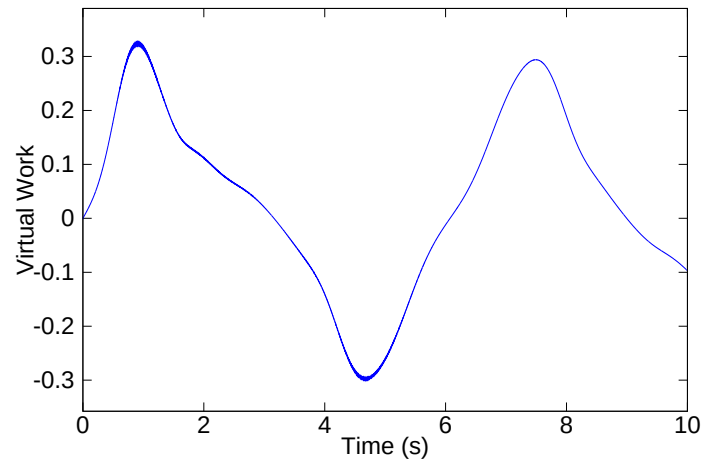


Figure 4.36: Double Pendulum Joint 1, 45 Degree Drop, 0.002 s time interval, 10 second span

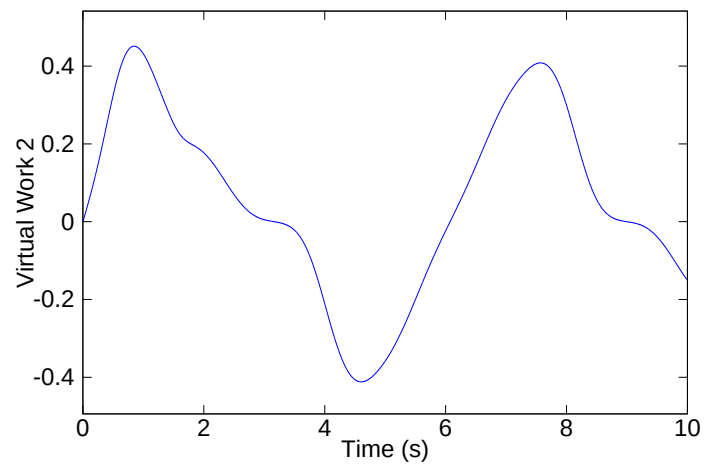


Figure 4.37: Double Pendulum Joint 2, 45 Degree Drop, 0.002 s time interval, 10 second span

4.2.2.2 *100 second Run Time.* A complete 10 second run time was achieved at a time step of 0.002 seconds and at all other sample rates. However, this was not the case for the longer run time of 100 seconds. The program produced an error and could not converge using the implemented solver. There could be many reasons for these errors as will be explored later. The time interval of 0.003 seconds was used in order to complete the run for the longer time interval, which only lasted until 26 seconds.

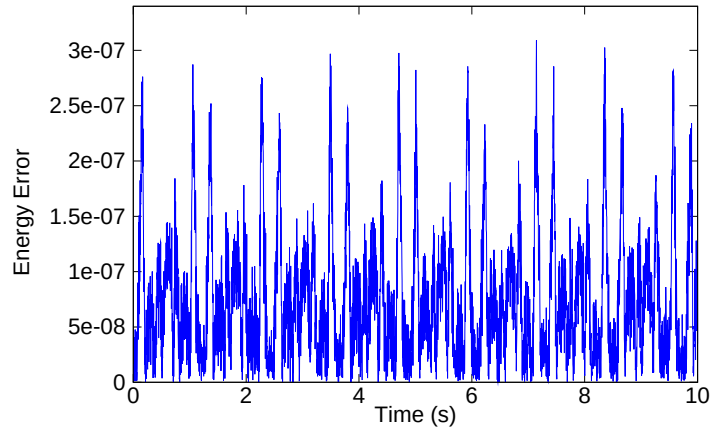


Figure 4.38: Double Pendulum, 45 Degree Drop, 0.003 s time interval, 100 second span

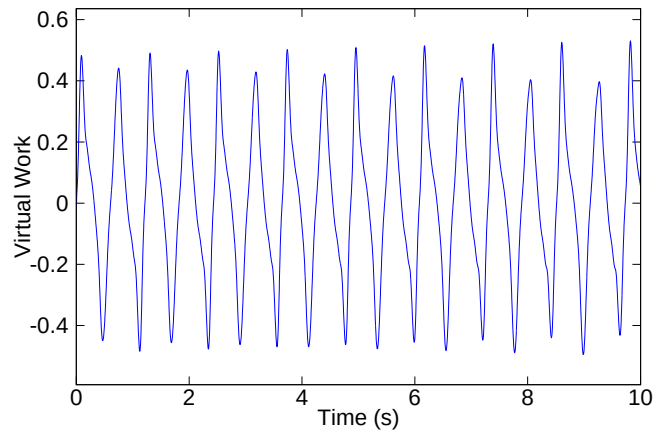


Figure 4.39: Double Pendulum, 45 Degree Drop, 0.003 s time interval, 100 second span

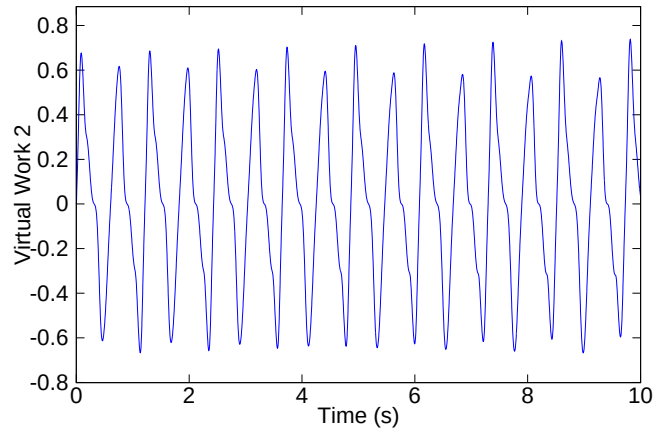


Figure 4.40: Double Pendulum, 45 Degree Drop, 0.003 s time interval, 100 second span

At 0.003 seconds the energy and the virtual work produce similar results as the shorter run time of 10 seconds. The energy RMS error is 4.6614e-08 and virtual work RMS error for joint 1 is 1.6539e-06 and joint 2 is -8.5665e-06. All values produced near zero results as expected. In fact, the longer the run was, the less error the program produced.

4.2.3 89 Degree Drop Results.

4.2.3.1 10 second Run Time. For an 89 degree drop angle, the results for a nearly perfect environment were also very predictable. There is no air drag or friction, just gravity acting on the pendulum ball. With this in mind, the energy difference should be near zero. Figure 4.41 shows the paths of the pendulum balls.

Figure 4.42 displays exactly what was predicted. As the pendulum swings with no energy loss, a near zero energy difference is observed. In this instance compared to the single pendulum, both balls start with and maintain a certain amount of energy as a pair.

The second measure of the system is virtual work. When a rigid body that is in equilibrium is subject to virtual displacements, the total virtual work of external forces is zero. This isn't true for each instance of time, as virtual work of each body

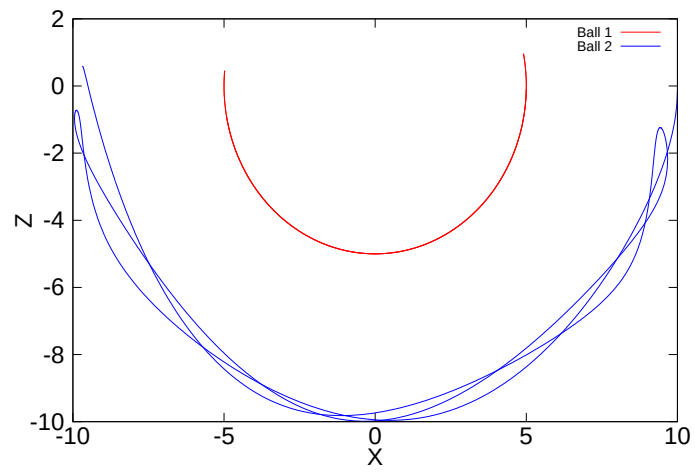


Figure 4.41: Double Pendulum 89 Degree Drop Path Plot

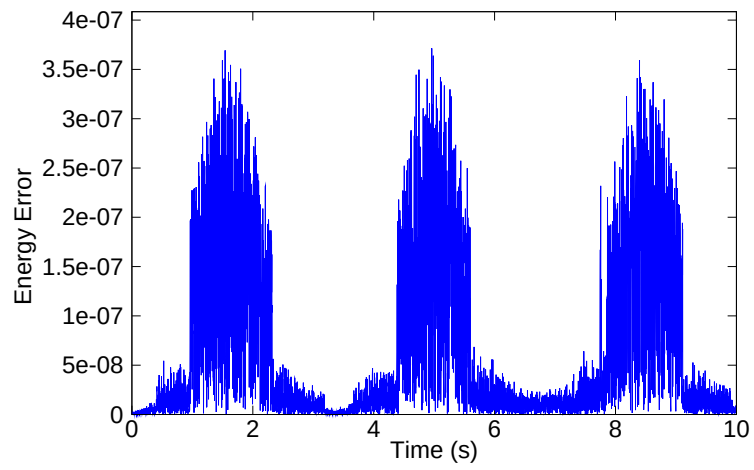


Figure 4.42: Double Pendulum, 89 Degree Drop, 0.002 s time interval, 10 second span

is added together, but for each individual body. Figure 4.43 and 4.44 figure shows the virtual work at each respective joint of the system.

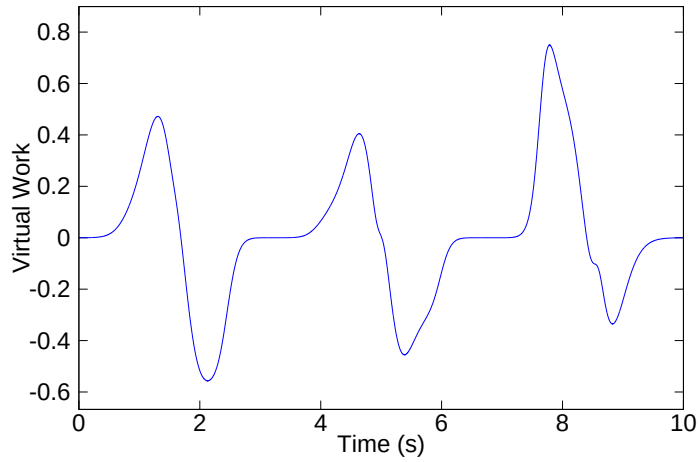


Figure 4.43: Double Pendulum Joint 1, 89 Degree Drop, 0.002 s time interval, 10 second span

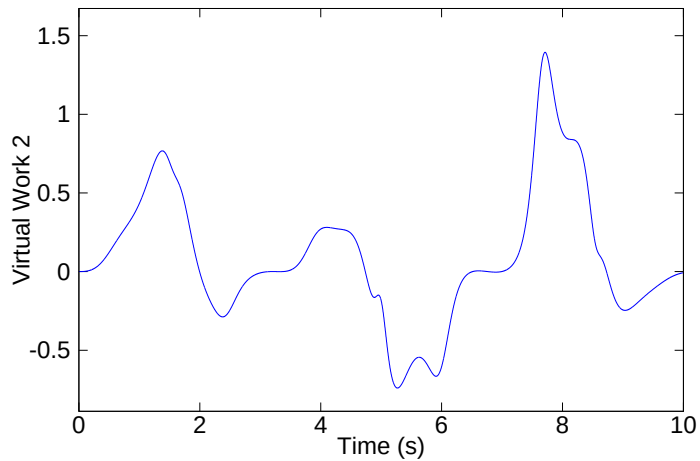


Figure 4.44: Double Pendulum Joint 2, 89 Degree Drop, 0.002 s time interval, 10 second span

Due to the chaotic behavior of the pendulum, there was no cycle to be observed. When a root mean square (RMS) of the virtual work in joint 1 is calculated, the answer turns out to be very close to zero with a value of $8.290e-05$ and $2.346e-05$ for joint 2. These values are well within the acceptable limits for short and long term simulations.

4.2.3.2 100 second Run Time. A complete run was achieved at a time step of 0.002 seconds. However, this was not the case for the longer run time of 100 seconds. The program produced an error and could not converge using the implemented solver. There could be many reasons for these errors as will be explored later. All other results produced predictable results. For the longer run time of 100 seconds, all time intervals were attempted but failed to converge. The time interval of 0.002 seconds was used in order to complete the run for the longer time interval, which only lasted until 25 seconds.

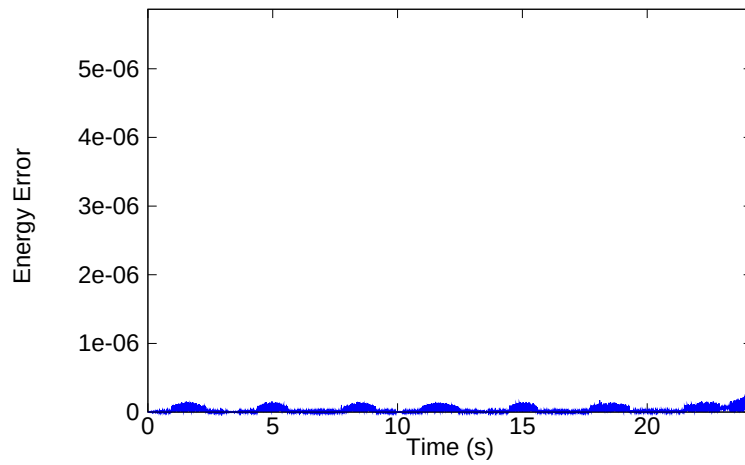


Figure 4.45: Double Pendulum, 89 Degree Drop, 0.02 s time interval, 100 second span

Figure 4.45 shows the energy error increases tremendously before convergence failure of the system occurs. This could be a clue as to why there are convergence errors.

At 0.002 seconds the energy and the virtual work produce similar results as the shorter run time of 10 seconds. The energy RMS error is $4.6614\text{e-}08$ and virtual work RMS error for joint 1 is $1.6539\text{e-}06$ and joint 2 is $-8.5665\text{e-}06$. All values produced near zero results as expected. In fact, the longer the run was, the less error the program produced.

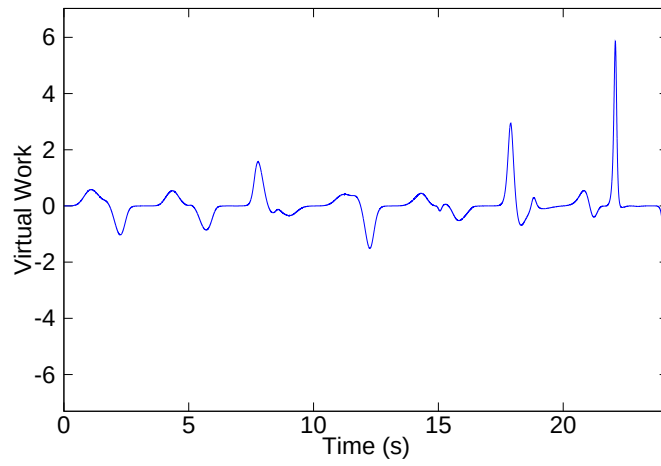


Figure 4.46: Double Pendulum Joint 1, 89 Degree Drop, 0.002 s time interval, 100 second span

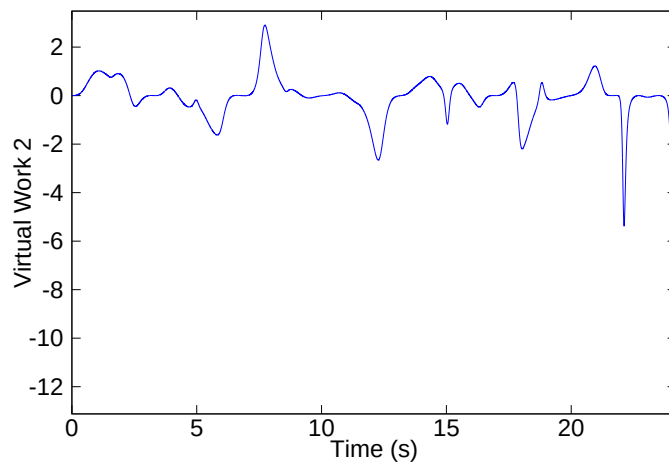


Figure 4.47: Double Pendulum Joint 2, 89 Degree Drop, 0.002 s time interval, 100 second span

4.2.4 135 Degree Drop Results.

4.2.4.1 10 second Run Time. For a 135 degree drop angle, the results for a nearly perfect environment were also very predictable. There is no air drag or friction, just gravity acting on the pendulum ball. With this in mind, the energy difference should be near zero. Figure 4.48 shows the path of the pendulum ball.

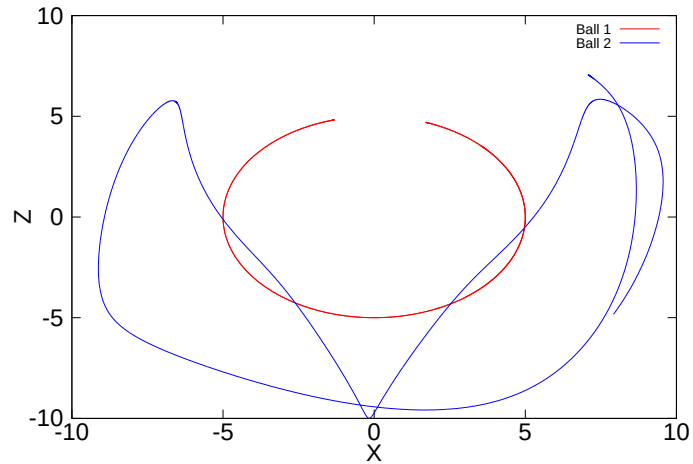


Figure 4.48: Double Pendulum 135 Degree Drop Path Plot

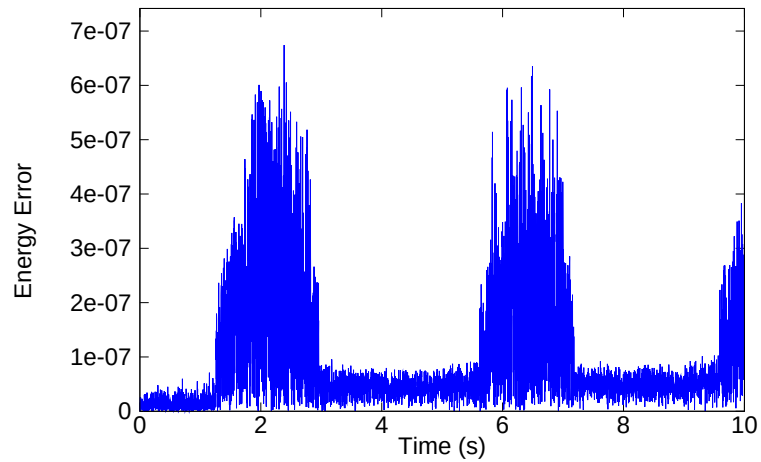


Figure 4.49: Double Pendulum, 135 Degree Drop, 0.002 s time interval, 10 second span

Figure 4.49 displays exactly what was predicted. As the pendulum swings with no energy loss, a near zero energy difference is observed. In this instance compared to the single pendulum, both balls start with and maintain a certain amount of energy as a pair.

The second measure of the system is virtual work. When a rigid body that is in equilibrium is subject to virtual displacements, the total virtual work of external forces is zero. This isn't true for each instance of time, as virtual work of each body is added together, but for each individual body. Figure 4.50 and 4.51 figure shows the virtual work at each respective joint of the system.

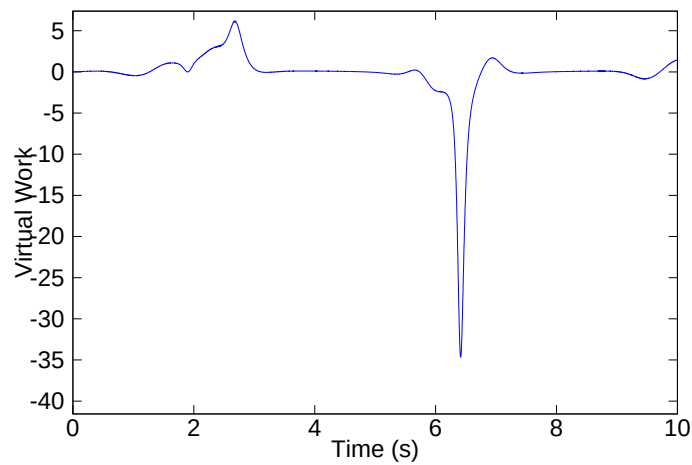


Figure 4.50: Double Pendulum Joint 1, 135 Degree Drop, 0.002 s time interval, 10 second span

Due to the chaotic behavior of the pendulum, there was no cycle to be observed. When a root mean square (RMS) of the virtual work in joint 1 is calculated, the answer turns out to be very close to zero with a value of $8.290e-05$ and $2.346e-05$ for joint 2. These values are well within the acceptable limits for short and long term simulations.

4.2.4.2 100 second Run Time. A complete run was achieved at a time step of 0.002 seconds. However, this was not the case for the longer run time of 100 seconds. The program produced an error and could not converge using the implemented solver. There could be many reasons for these errors as will be explored later. All other results produced predictable results. For the longer run time of 100

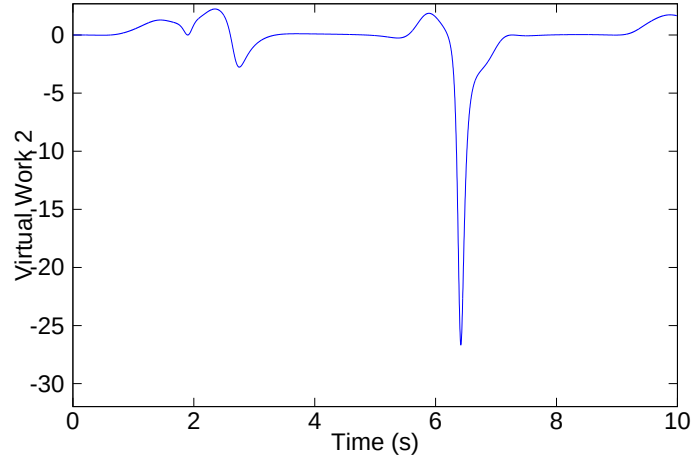


Figure 4.51: Double Pendulum Joint 2, 135 Degree Drop, 0.002 s time interval, 10 second span

seconds, all time intervals were attempted but failed to converge. The time interval of 0.002 seconds was used in order to complete the run for the longer time interval, which only lasted until 19 seconds.

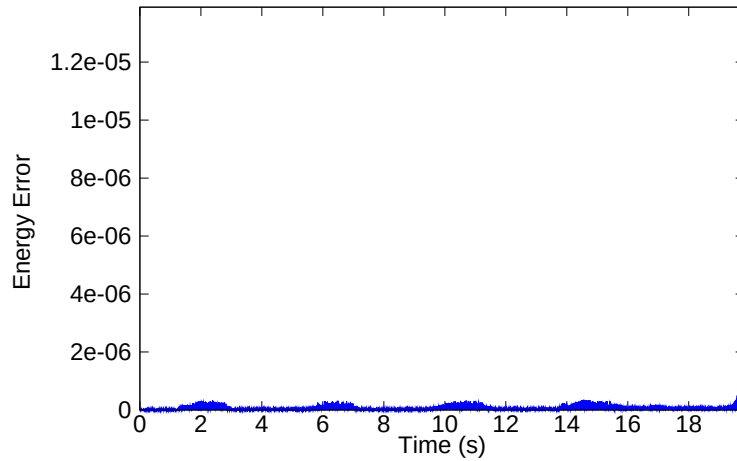


Figure 4.52: Double Pendulum, 135 Degree Drop, 0.02 s time interval, 100 second span

Interestingly, Figure 4.52 shows a similar occurrence as Figure 4.45. The energy error increases tremendously before convergence failure of the system occurs. At 0.002 seconds the energy and the virtual work produce similar results as the shorter run time of 10 seconds. The energy RMS error is $9.9443\text{e-}08$ and virtual work RMS error for

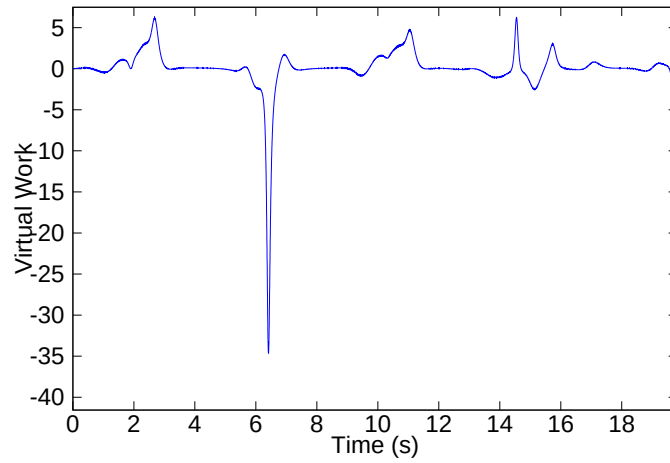


Figure 4.53: Double Pendulum Joint 1, 135 Degree Drop, 0.002 s time interval, 100 second span

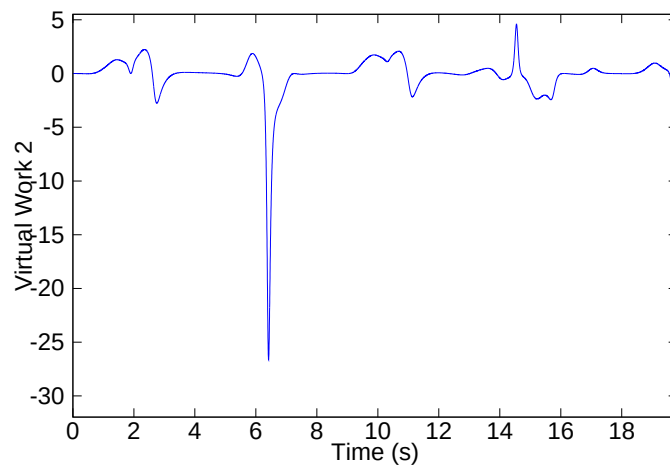


Figure 4.54: Double Pendulum Joint 2, 135 Degree Drop, 0.002 s time interval, 100 second span

joint 1 is $-5.0424\text{e-}07$ and joint 2 is $-1.7151\text{e-}05$. All values produced near zero results as expected. In fact, the longer the run was, the less error the program produced.

4.2.5 The Double Pendulum and Chaos. Chaos is a special condition of nature. The behavior of systems that follow deterministic laws but appear random and unpredictable. Chaotic systems are very sensitive to initial conditions; small changes in those conditions can lead to vastly different outcomes. In aeronautics, the chaotic behavior of the flow of air in conditions of turbulence represent chaos.

In order for the written code to be valid, it must demonstrate chaos, as in nature. Two different drop angles for the double pendulum were observed and recorded, 45 degrees and 90 degrees. The definition of chaos tells us that slight changes in the initial conditions should change the outcome of the system vastly. It is known that the double pendulum displays chaos at or around 90 degrees for slight changes and at 45 degrees it does not. Table 4.3 shows the initial conditions for the run, while the initial drop angle for both initial conditions of 90 and 45 will be altered by 0.01 degree. Figure 4.55 shows the 45 degree X-Y path and the slightly changed 44.99 degree X-Y path. Figure 4.56 shows the 90 degree X-Y path and the slightly changed 89.99 degree X-Y path.

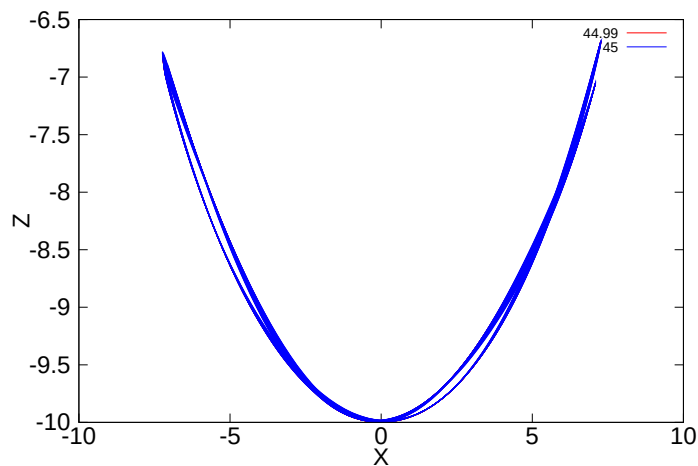


Figure 4.55: Chaos Test 45 deg angle - The two paths are very similar

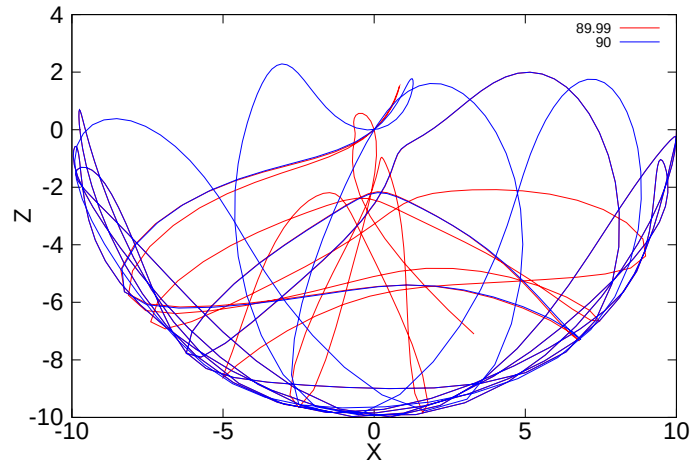


Figure 4.56: Chaos Test 90 deg angle - The two paths are very different despite a small difference in initial conditions

The results were as expected. The 45 degree and the 44.99 degree paths are nearly identical as predicted. Although, one can see for the setup at 90 degrees and 89.99, the results are quite different. This is because the system at this point is in chaos; a slight change in the initial conditions led to a drastic difference in behavior.

4.3 *Triple Pendulum*

The third multibody system tested was the triple pendulum. By adding another body and joint to the system, the complexity increases. As complexity increases, the program engine becomes more stressed and limitations are found. The triple pendulum, just as the double pendulum, is also a very dynamic multibody system. The potential energy should be transferred to different bodies throughout the run and end with the same amount of total energy. Similar inputs for different time steps are given for each case. Three different drop angles of 45 degrees, 89 degrees, and 135 degrees were considered. The angles were chosen based on the different potential energies each angle gave according to the initial orientation of the system. For simplicity, each pendulum ball was at the same angle for each run. Other inputs include the base position, each pendulum ball position, mass of each pendulum ball,

the inertia of each pendulum ball, the gravity of the system, the initial velocity of each body, the initial angular velocity of each body, and the initial force placed on each joint. The initial conditions, constants, and variables are shown in the table below.

Table 4.5: Triple Pendulum Constants

Gravity (m/s^2)	9.8
Base Position (x,y,z)	(0,0,0)
Pendulum Ball 1 Mass	2 kg
Pendulum Ball 2 Mass	2 kg
Pendulum Ball 3 Mass	2 kg
Initial Velocity Ball 1 (m/s)	0
Initial Velocity Ball 2 (m/s)	0
Initial Velocity Ball 3 (m/s)	0
Initial Angular Velocity Ball 1 (m/s)	0
Initial Angular Velocity Ball 2 (m/s)	0
Initial Angular Velocity Ball 3 (m/s)	0
Initial Force on each Joint (N)	0

Table 4.6: Triple Pendulum Variables

Each Pendulum Ball Position (deg)	deg 45, deg 89, deg 135
Sample Rate (s)	0.1, 0.04, 0.01, 0.004, 0.002
Run Time (s)	10, 100

4.3.1 Triple Pendulum Results. No results could be compiled due to convergence errors. All three drop angles were tested at every time step shown in Table 4.8 with no convergence at the very first time step. No data was recorded. The convergence errors could stem from different variables. Adding an extra body may have overwhelmed the software's limits of data storage. The program's solver may not be robust enough or not using the right algorithms to handle such a case. A deeper investigation needs to be pursued in determining the reason for this error.

4.4 Top

The last body to test is the top. Even though three-dimensional motion was allowed with the other cases, it was not meant to enter into it. The top translates in

all directions and also rotates in all directions. Just as with the single pendulum, the top's energy should remain constant. Three different cases for the top were considered. Similar inputs for different time steps are given for each case. Inputs for the top include the base position, top position, mass of top, the gravity of the system, the initial velocity, the initial angular velocity, and the initial force placed on the base or top itself. The initial conditions, constants, and variables are shown in the table below.

Table 4.7: Top Constants and Initial Conditions

Gravity (m/s^2)	3.0
Top Mass	1 kg
Moment of Inertia (x,y,z)	(0.40, 0.40, 0.75)
Orientation to yz plane (deg)	10 deg
Base Position (x,y,z)	(0,0,0)
Initial Velocity Top (x,y,z) (m/s)	(0,0,0)
Initial Force on Base (x,y,z) (N)	(0,0,0)
Initial Force on Top (x,y,z) (N)	(0,0,0)

Table 4.8: Top Problem Variables

Initial Angular Velocity Top (x,y,z) (rad/s)	(0, 0.9888, 7.5167)
	(0, 0.20905, 6.2964)
	(0, 0, 6.3794)
Sample Rate (s)	0.1, 0.04, 0.01, 0.004, 0.002
Run Time (s)	10, 100

Qualitatively the results of this section will be compared to the 1991 analysis: “Primal and mixed forms of Hamilton’s principle for constrained rigid body systems: numerical studies.” [2] The two figures, 4.58 and 4.57 below are taken directly from the analysis that was previously accomplished. All input parameters shown in Table 4.7 and Table 4.8 were also taken directly from this previous study.

4.4.1 Top Trend Analysis. In order to understand the system, an analysis was done. As data was taken, the time to complete each run, energy error, RMS energy error, virtual work RMS, and virtual work was recorded for each run. The results are shown in the following figures and graphs.

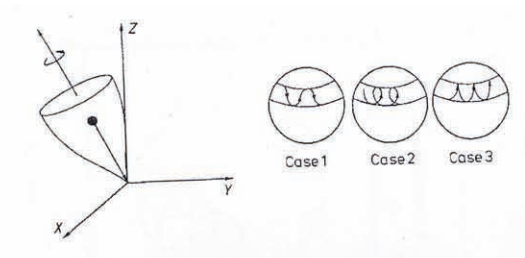


Figure 4.57: Previously accomplished top results

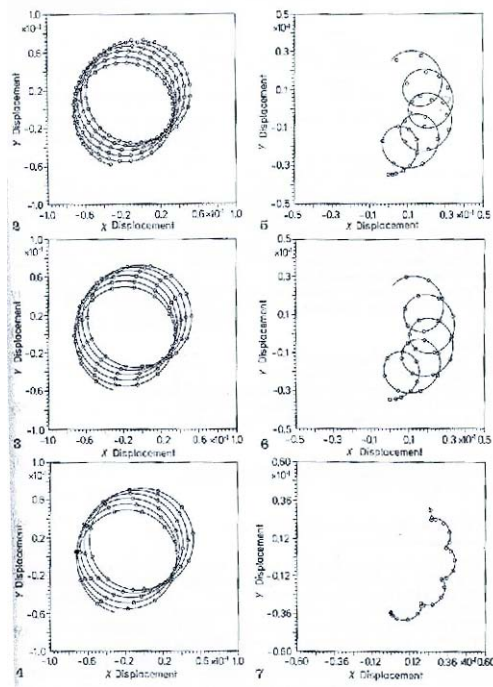


Figure 4.58: Previously accomplished top results Case 1, Case 2, Case 3 from top to bottom

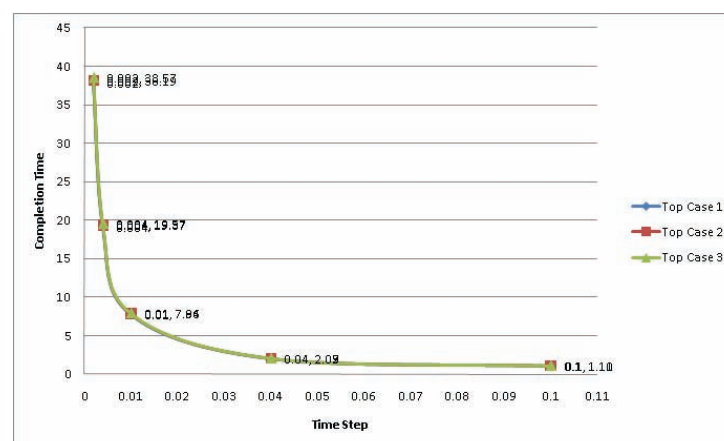


Figure 4.59: Time Step vs Completion Time for the Top

The initial conditions for each case are given in Table 4.7 along with the variables in Table 4.8. There are a couple of trends that one can conclude from the graphs. The results for the top are very consistent. Figure 4.59 shows that for each run, the times to complete each run were consistent. This is not surprising due to the nature of the variable, moment of inertia, that was changed for each case. Increasing the sample rate, increases computational time on a log-log scale. No computing advantages can be drawn from increasing sample rate.

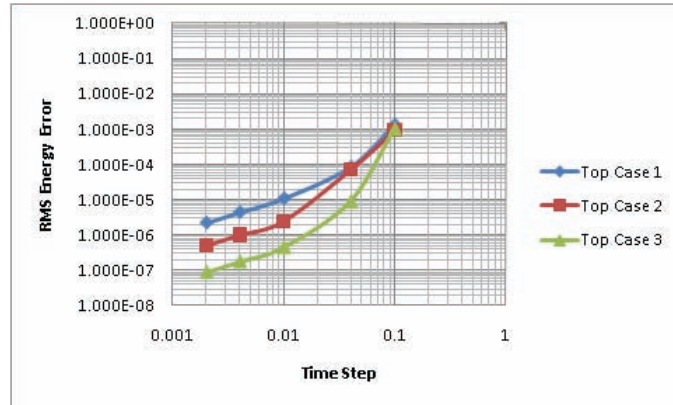


Figure 4.60: Time Step vs RMS Energy Error for the top

Unlike completion time, as sample increased, the energy RMS error is decreased. The rate is fairly linear as sample rate is increased. For a time step difference from 0.01 and 0.002 of five percent, the error is reduced 10 times as seen in Figure 4.60.

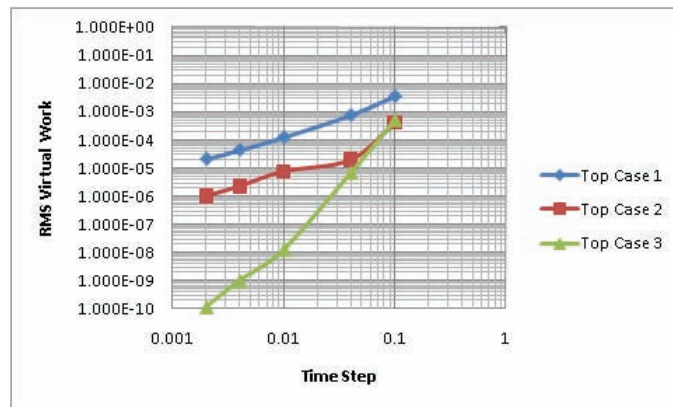


Figure 4.61: Time Step vs Virtual Work for the top

The RMS error associated with virtual work showed some interesting trends. The initial prediction was that all the top cases would yield similar results within a

few percent. On every plot, the RMS error for case 1 had significantly more RMS error than Case 2. Case 2 also had more RMS error than case 3. As with energy, the trends for virtual work were linear in nature on a log-log plot with a decrease in error as sample rate increased.

4.4.2 Top - Case 1.

4.4.2.1 10 Second Span. The top used the same programming engine, seen in appendices, as the pendulum. The difference being, the inputs assigned to the xml input form bodies that are characteristic of a top. The moment of inertia is such that the center of mass is situated above the base at a 10 degree angle, like a top tilted at a ten degree angle. An initial velocity is fed to the top and the physics of the engine take control. Unlike the pendulum, all runs were completed with no faults or errors. The initial XY path is shown in Figure 4.62.

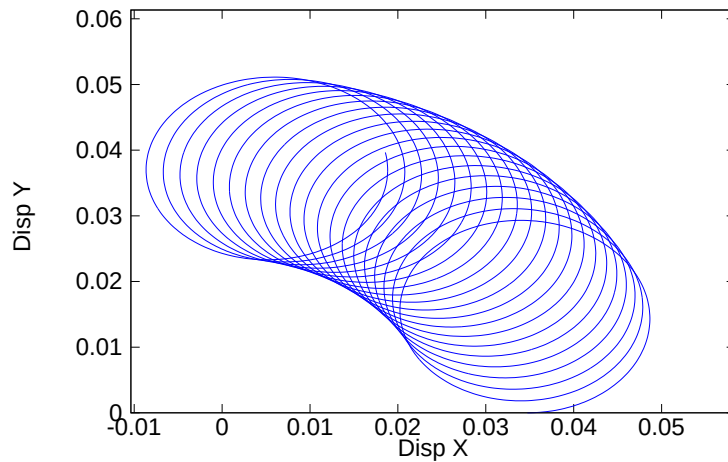


Figure 4.62: Top X-Y Path for Case 1, 0.002 s time interval, 10 second span

A consistent path, of the point of the top, is shown in the figure, which continues in a repeating cycle.

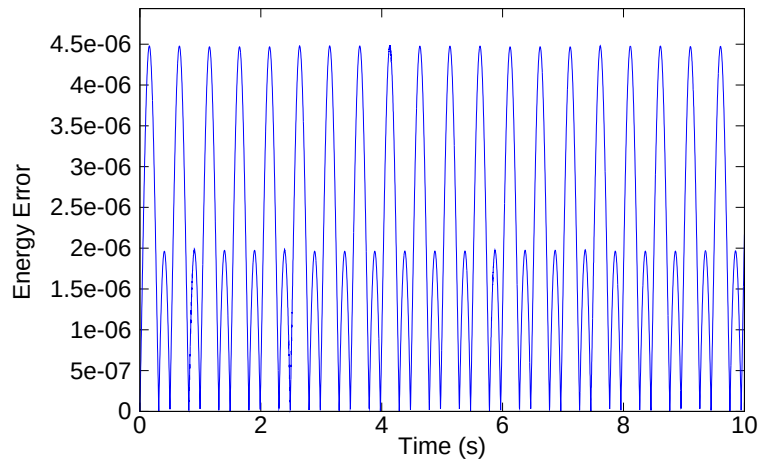


Figure 4.63: Top Energy Change for Case 1, 0.002 s time interval, 10 second span

As with the pendulum, there is no energy consumed in the near perfect environment of the top as shown in Figure 4.63. The RMS error value for energy was $2.201\text{e-}06$.

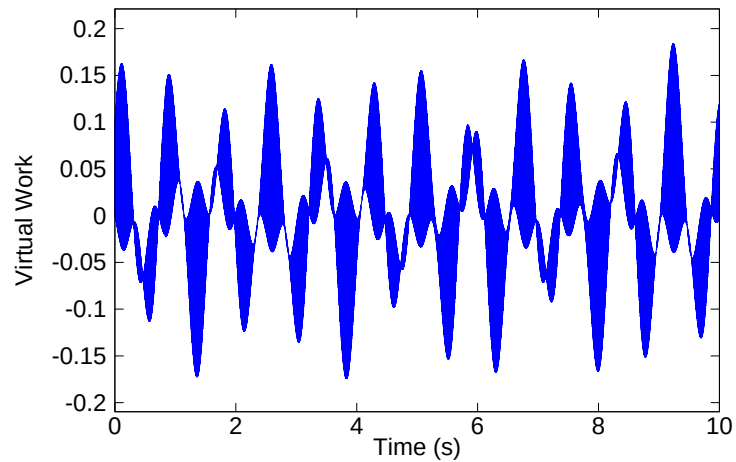


Figure 4.64: Top Virtual Work for Case 1, 0.002 s time interval, 10 second span

As with the pendulum, the virtual work should also be close to zero. Figure 4.64 shows the virtual work of the top for case 1 over 10 seconds. The RMS error value for virtual work was $-2.071\text{e-}05$, which is indeed close to zero.

4.4.2.2 *100 Second Span.* 100 second runs were completed using the same input data as the 10 second span. The complete path is shown in Figure 4.65.

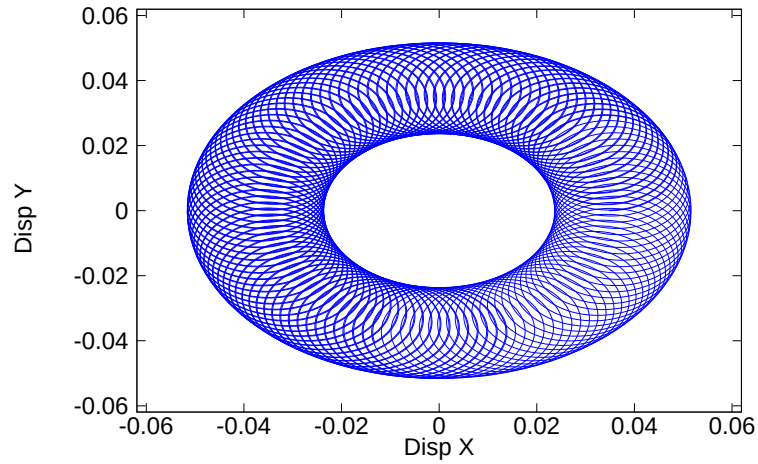


Figure 4.65: Top X-Y Path for Case 1, 0.002 s time interval, 100 second span

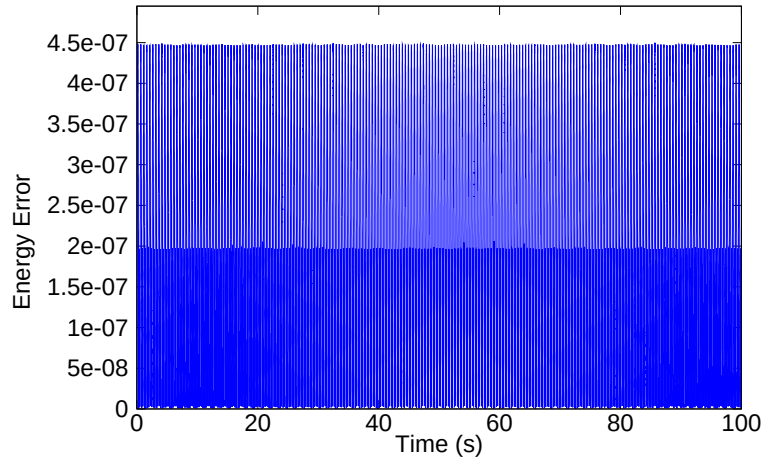


Figure 4.66: Top Energy Change for Case 1, 0.002 s time interval, 100 second span

A complete cycle of the top is achieved and continues as shown in Figure 4.66. As with the pendulum, there is no energy consumed in the near perfect environment of the top as shown in Figure 4.63. The RMS energy error was $2.207\text{e-}07$. As with the pendulum, the virtual work should also be close to zero. Figure 4.64 shows the

virtual work of the top for case 1 over 10 seconds. The RMS error value for virtual work was $-2.073\text{e-}06$

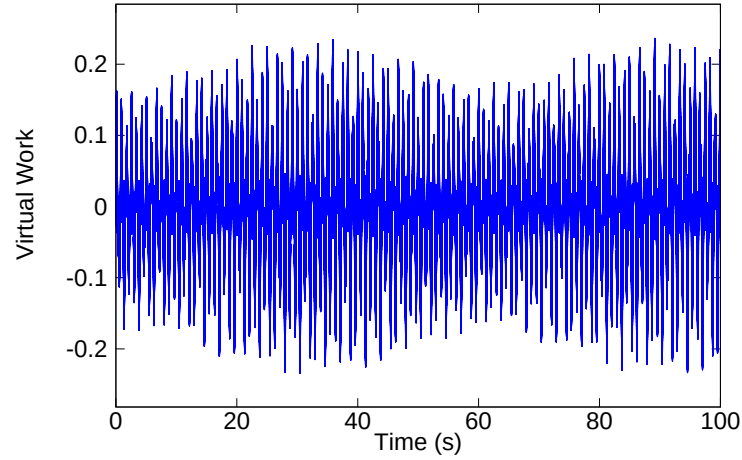


Figure 4.67: Top Virtual Work for Case 1, 0.002 s time interval, 100 second span

4.4.3 Top - Case 2. The only thing that changes in case 2 is that the angular velocity, according to Table 4.8. The results are as follows.

4.4.3.1 10 Second Span. The top used the same basic engine as the pendulum. The characteristics of the top are formed through the inputs assigned to the bodies. The moment of inertia is such that the center of mass is situated above the base at a 10 degree angle, like a top tilted at a ten degree angle. An initial velocity is fed to the top and the physics of the engine take control. Unlike the pendulum, all runs were completed with no faults or errors. The initial XY path is shown in Figure 4.68.

A consistent path is shown in the figure, which continues in a repeating cycle.

As with the pendulum, there is no energy consumed in the near perfect environment of the top as shown in Figure 4.69. The RMS error value for energy was $5.052\text{e-}07$.

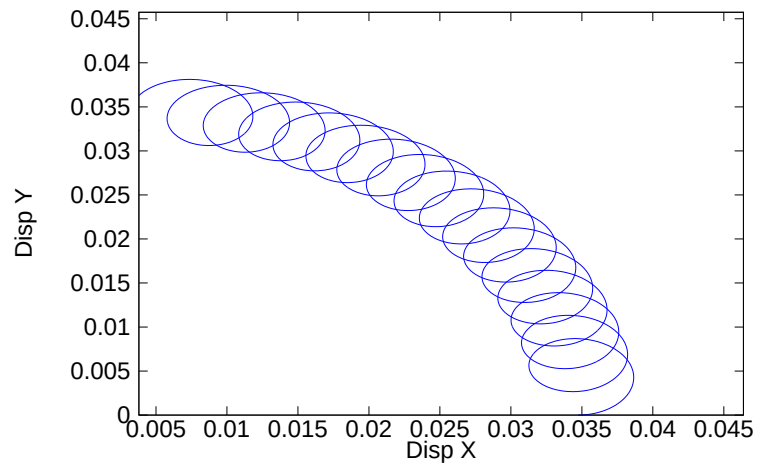


Figure 4.68: Top X-Y Path for Case 2, 0.002 s time interval, 10 second span

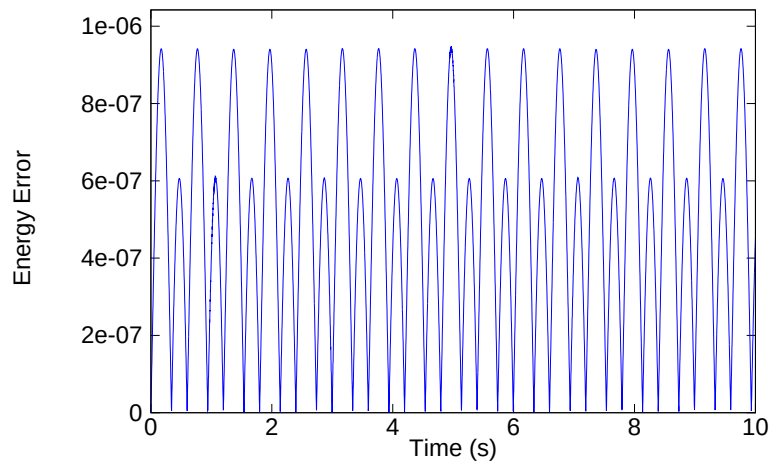


Figure 4.69: Top Energy Change for Case 2, 0.002 s time interval, 10 second span

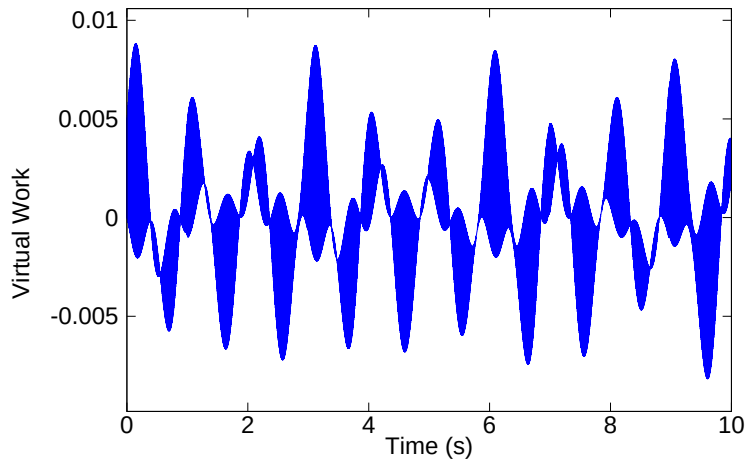


Figure 4.70: Top Virtual Work for Case 2, 0.002 s time interval, 10 second span

As with the pendulum, the virtual work should also be close to zero. The fluctuations in the system are caused by varying constraint forces acting on the joint. Figure 4.70 shows the virtual work of the top for case 1 over 10 seconds. The RMS error value for virtual work was $-1.009\text{e-}06$, which is indeed close to zero.

4.4.3.2 100 Second Span. 100 second runs were completed using the same input data as the 10 second span. The complete path is shown in Figure 4.71. As with the pendulum, there is no energy consumed in the near perfect environment of the top as shown in Figure 4.69. The RMS energy error was $5.044\text{e-}08$. The virtual work should also be close to zero. Figure 4.70 shows the virtual work of the top for case 1 over 10 seconds. The RMS error value for virtual work was $-1.011\text{e-}07$.

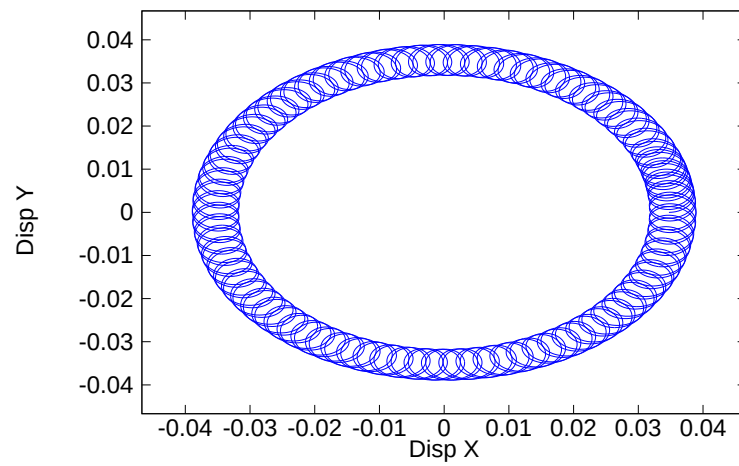


Figure 4.71: Top X-Y Path for Case 2, 0.002 s time interval, 100 second span

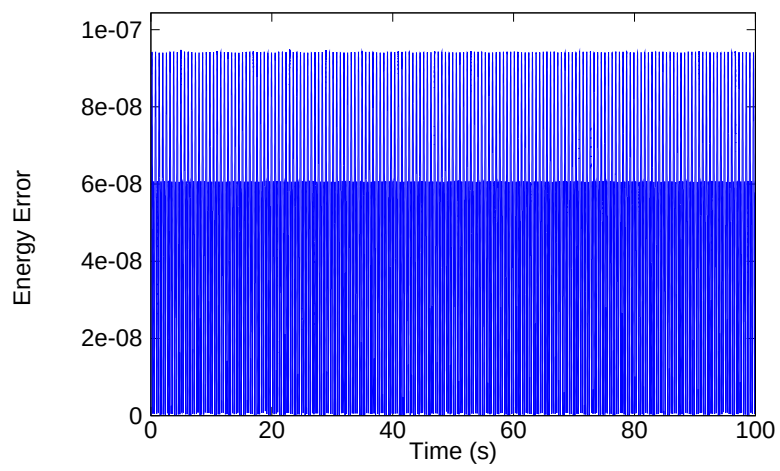


Figure 4.72: Top Energy Change for Case 2, 0.002 s time interval, 100 second span

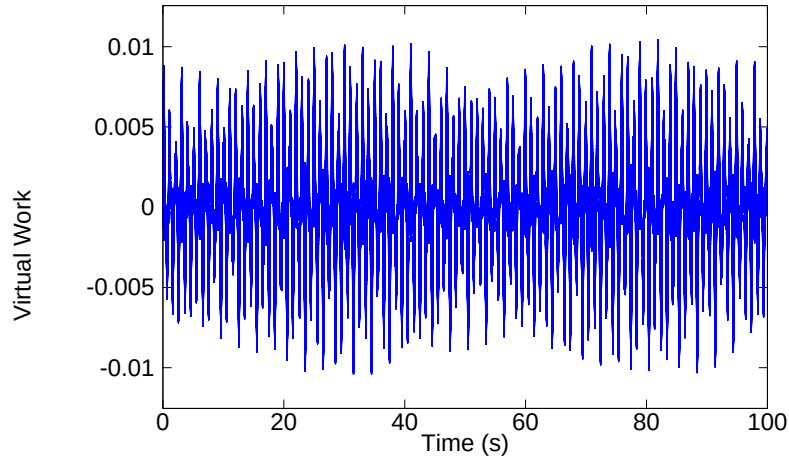


Figure 4.73: Top Virtual Work for Case 2, 0.002 s time interval, 100 second span

4.4.4 *Top - Case 3.* Again, angular velocity is the one variable that changes in case 3 according to Table 4.8. The results are as follows.

4.4.4.1 *10 Second Span.* The top used the same program as the pendulum. The characteristics of the top are formed through the inputs assigned to the bodies. The moment of inertia is such that the center of mass is situated above the base at a 10 degree angle, like a top tilted at a ten degree angle. An initial velocity is fed to the top and the physics of the engine take control. Unlike the pendulum, all runs were completed with no faults or errors. The initial XY path is shown in Figure 4.74.

A consistent path is shown in the figure, which continues in a repeating cycle.

As with the pendulum, there is no energy consumed in the near perfect environment of the top as shown in Figure 4.75. The RMS error value for energy was 8.815e-08.

As with the pendulum, the virtual work should also be close to zero. Figure 4.76 shows the virtual work of the top for case 1 over 10 seconds. The RMS error value for virtual work was -1.175e-10, which is indeed close to zero.

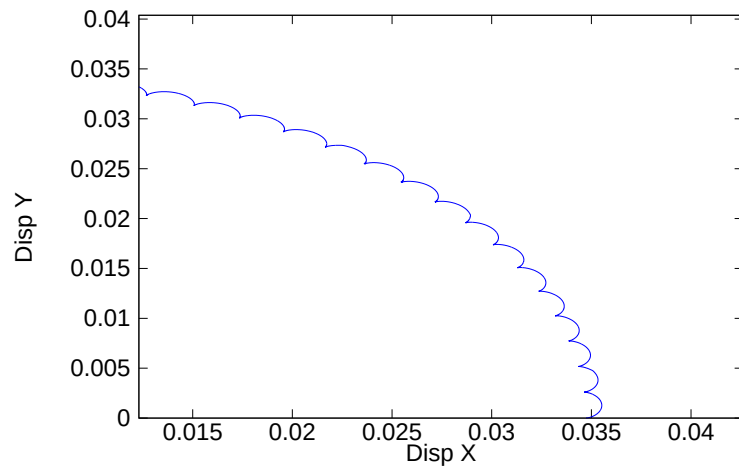


Figure 4.74: Top X-Y Path for Case 3, 0.002 s time interval, 10 second span

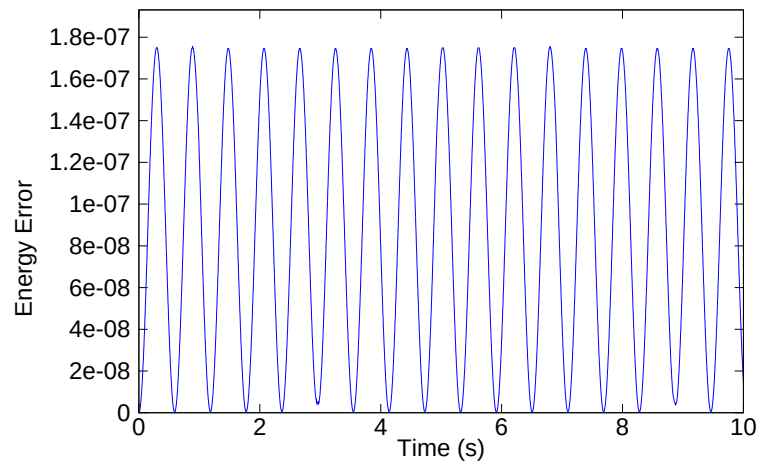


Figure 4.75: Top Energy Change for Case 3, 0.002 s time interval, 10 second span

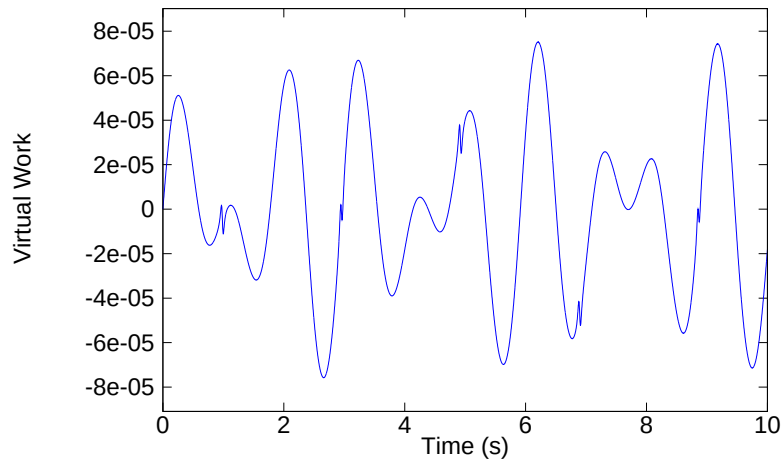


Figure 4.76: Top Virtual Work for Case 3, 0.002 s time interval, 10 second span

4.4.4.2 100 Second Span. 100 second runs were completed using the same input data as the 10 second span. The complete path is shown in Figure 4.77.

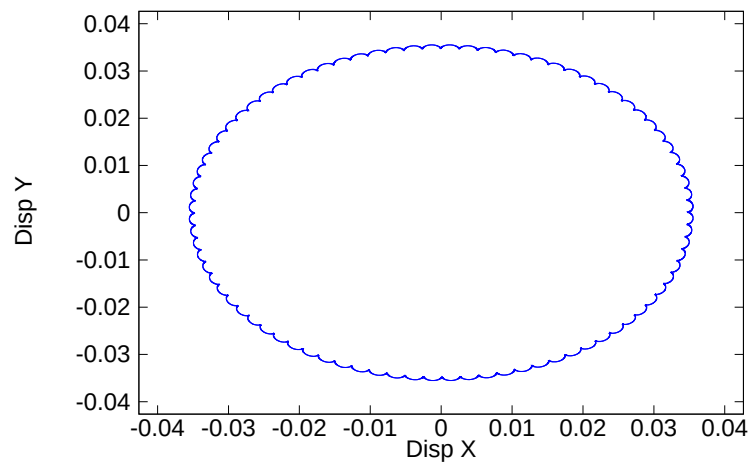


Figure 4.77: Top X-Y Path for Case 3, 0.002 s time interval, 100 second span

A complete cycle of the top is achieved and continues as shown.

As with the pendulum, there is no energy consumed in the near perfect environment of the top as shown in Figure 4.75. The RMS energy error was 8.770e-08.

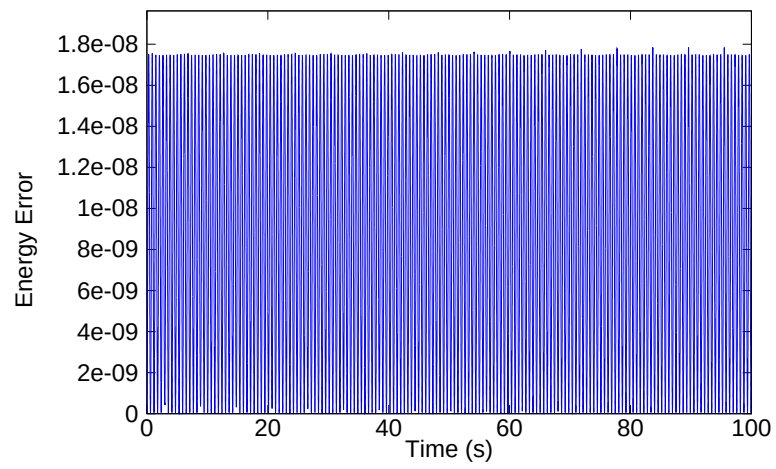


Figure 4.78: Top Energy Change for Case 3, 0.002 s time interval, 100 second span

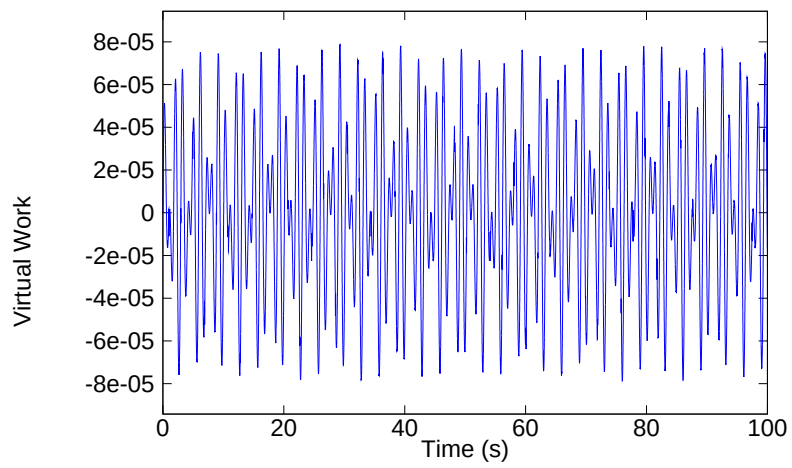


Figure 4.79: Top Virtual Work for Case 3, 0.002 s time interval, 100 second span

As with the pendulum, the virtual work should also be close to zero. Figure 4.76 shows the virtual work of the top for case 1 over 10 seconds. The RMS error value for virtual work was $-8.075\text{e-}12$

V. Conclusions

5.1 *Summary*

The goal of this research was to verify and validate the multibody software written using Hamilton's Law to compute motion of objects. Use of this software can offer an alternative, algebraic only, non-differential equation solution to modern day multibody systems. The software written has the potential to be very accurate and computationally simpler than any other method today. As the bugs are worked out of the software and user friendliness is built in, this method has a lot of potential to be the method of choice for complex multibody systems. Validation of this research includes test runs using multiple sample rates and drop angles of a pendulum, double pendulum, and spinning top. Energy preserving and decaying schemes were used to validate the results. The RMS error calculated in each run was satisfactory for both short and long term simulation of most multibody systems.

Another strength associated with the program, in general, is the portability of the engine. Referring back to Chapter III, the system used was a five year old laptop with a free version of Linux. Any modern system today and tomorrow should be compatible with and powerful enough to run these multibody systems at a fairly fast rate.

5.2 *Evaluation of the Software*

5.2.1 Validation and Verification. Through testing, the software has been found to be valid. The basic principles of mathematics are satisfied along with the basics principles of physics. When initial conditions are entered into the program, reasonable responses are received. When unreasonable conditions are input, the program fails to converge accordingly. Run times were collected and analyzed in order to characterize the efficiency and validity of the software program. These times were analyzed and returned predictable results.

In order for verification to be satisfied, the program must satisfy the user's intent of the program. In order to model multibody correctly, the user required chaos theory

to be displayed just as it is in nature. When small changes in the environment were made, it produced large changes in the result of the system. Correctly, the double pendulum system displayed chaos around a 90 degree drop and not around a 45 degree drop. Although the system failed in robustness due to an insufficient solver embedded in the software program. As a visual indicator, dynamic motion of the system was observed through a Visual Basic script in Excel using the data that was output from the program.

5.2.2 General Evaluation. The software was very easy to use after installing the correct environment, Linux and Octave. Of course, the pendulum is a very simple system and is a good starting point to validate the software. The results of the pendulum are well known and other models exist to compare results. Further, using Excel, a dynamic plot was made to view the motion of the pendulum ball as it translated in the environment. This program was written to be the first line of verification to check if the output data and linkages between joints was correct and motion was fluid as one would expect from a pendulum. This was done for the double pendulum and spinning top as well. For the output of each case, the Excel program produced a dynamic motion that was indicative of each individual body. The benefit of viewing this motion comes in handy when simple mistakes can find their way in to coding your multibody system. More than once, a mistake of inputting the wrong body or joint was found by physically viewing the dynamic response of the output.

This software is fairly robust for modeling multibody systems, and produced valid results using a stand-alone laptop computer. Each solution required between three seconds and ten minutes of CPU time, followed by another three seconds to ten minutes to run the post processing application to report energy, virtual work, and draw the path of the object. The overall time to run all simulations could be shortened with a robust graphical user interface, GUI interface.

5.2.3 Problems. Throughout this whole process, very few errors or mistakes were encountered. The system output predicted results and errors were close to

zero. Although, throughout the process and when the triple pendulum was tested, a convergence error was encountered. The convergence error seemed to show its head with small sample times and chaotic movement of the system. Also on runs longer than ten seconds, the program would tend to have a point at where it would consistently not converge. During a few instances, the RMS energy value was observed to rise very rapidly as the system approached non-convergence as seen in figure 4.52. When the triple pendulum was tested, not one run would converge. Different starting points and initial conditions were used to try and jump start the program to no avail. The top had no problems no matter how long it ran and always converged no matter how small the sample rate.

Other than the problems stated above, few problems were encountered once the simulation environment was setup. The setup of the environment takes skill and troubleshooting that most people do not have or cannot afford timewise. Periodic updates to the Linux/Ubuntu system were neglected out of fear of losing compatibility.

5.3 Recommendations

It is recommended that the source of the convergence error be found. Speculation has been made to the problem being a solver that is very delicate. Otherwise, the program is very robust and needs the chance to show its power through more complicated multibody systems. A more detailed look at the solver should be accomplished. Although, at the conclusion of testing, the triple pendulum was run in a different environment with success. This goes back to the premise of the sensitivity of the installation environment of the program itself. Future work in this area should be considered.

Data can also quickly pile up without a good file management plan. Considerable time was given to naming conventions of the bodies, constants, and variables. Illogical naming conventions forced many re-runs due to improper naming.

5.3.1 Future Work. In order to avoid the headaches with the environment, a self sufficient compliance package for the software should be written to avoid compatibility issues. This package would include all necessary files to run the software package and documentation on how to overcome basic problems. Version testing should also be accomplished and documented in order to simplify the process.

Of course, for the average user, a GUI for inputting different environments, bodies, constraints, and joints should be made. Consideration should be given to the purpose and use of this program to specific users.

5.4 Conclusions

The multibody software program that simulates dynamic motion was found to be valid. The validity is based on quantitative and qualitative results. The quantitative results include energy decaying and preserving results, X-Y path plots, and the presentation of chaos. Qualitative results include comparing top diagrams from a previous study with the results in this study.

Besides the solver, the mathematics engine behind the software was quick, accurate, and robust. Each run produced results that were predictable and repeatable. Once the bug with the solver gets tweaked, the software is likely to be ready to take the next step in testing. Once more testing is accomplished, the outer body of the program, the GUI, can be added and user functionality can greatly increase.

Using HWP to solve multibody problems is a unique and functional way to solve a problem. HWP provides a solid algebraic alternative to what can sometimes be a more complicated process using derivatives and integrals.

Appendix A. Simple Pendulum XML Input

Listing A.1: XML input file for a simple pendulum

```
1  <?xml version="1.0" encoding="UTF-8"?>
   <data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="/home/hainge/msaHL/...
        msaHLData.xsd">
   <unitSystem>SI</unitSystem>
   <system>
6    <gravity>
      <g3 units="m/sec^2">9.80665</g3>
    </gravity>
      <bodies>
11     <ground>
        <name>Base</name>
      </ground>
      <rigidBody>
        <name>PendulumBob</name>
        <mass units="kg">2.0</mass>
16     <rGB>
        <x units="m">0.0</x>
      </rGB>
      <moi>
21     <Ixx units="kg-m^2">0.008</Ixx>
        <Iyy units="kg-m^2">0.008</Iyy>
        <Izz units="kg-m^2">0.008</Izz>
      </moi>
      <initialConditions>
26     <position basis="PendulumBob">
        <x units="m">0.0</x>
        <y units="m">0.0</y>
        <z units="m">5.0</z>
      </position>
      <orientation>
31     <eulerRotation>
        <phi units="deg">45</phi>
        <u1>0.0</u1>
        <u2>1.0</u2>
        <u3>0.0</u3>
36     </eulerRotation>
      </orientation>
      <velocity basis="PendulumBob">
        <v1 units="m/sec">0.0</v1>
        <v2 units="m/sec">0.0</v2>
41     <v3 units="m/sec">0.0</v3>
      </velocity>
      <angularVelocity basis="PendulumBob">
        <omega1 units="rad/sec">0.0</omega1>
        <omega2 units="rad/sec">0.0</omega2>
46     <omega3 units="rad/sec">0.0</omega3>
      </angularVelocity>
    </initialConditions>
  </rigidBody>
```

```

51     </bodies>
    <joints>
      <spherical>
        <name>Pivot</name>
        <parent>
          <name>Base</name>
56          <position basis="Base">
            <x units="m">0.0</x>
            <y units="m">0.0</y>
            <z units="m">0.0</z>
          </position>
61        </parent>
        <child>
          <name>PendulumBob</name>
          <position basis="PendulumBob">
            <x units="m">0.0</x>
66            <y units="m">0.0</y>
            <z units="m">-5.0</z>
          </position>
        </child>
        <initialConditions>
71          <reaction basis="Base">
            <xForce units="N">0.0</xForce>
            <yForce units="N">0.0</yForce>
            <zForce units="N">0.0</zForce>
          </reaction>
76        </initialConditions>
      </spherical>
    </joints>
  </system>
  <simulation>
81    <title>SphericalPendulum</title>
    <deltaT units="sec">0.002</deltaT>
    <tFinal>10.0</tFinal>
  </simulation>
</data>

```

Appendix B. Double Pendulum XML Input

Listing B.1: XML input file for a double pendulum

```
<?xml version="1.0" encoding="UTF-8"?>
<data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="/home/hainge/msaHL/...
      msaHLData.xsd">
  <unitSystem>SI</unitSystem>
5  <system>
    <gravity>
      <g3 units="m/sec^2">9.80665</g3>
    </gravity>
    <bodies>
10    <ground>
      <name>Base</name>
    </ground>
    <rigidBody>
      <name>PendulumBob</name>
15    <mass units="kg">2.0</mass>
      <rGB>
        <x units="m">0.0</x>
      </rGB>
      <moi>
20    <Ixx units="kg-m^2">0.008</Ixx>
      <Iyy units="kg-m^2">0.008</Iyy>
      <Izz units="kg-m^2">0.008</Izz>
    </moi>
    <initialConditions>
25    <position basis="PendulumBob">
      <x units="m">0.0</x>
      <y units="m">0.0</y>
      <z units="m">5.0</z>
    </position>
30    <orientation>
      <eulerRotation>
        <phi units="deg">135.0</phi>
        <u1>0.0</u1>
        <u2>1.0</u2>
35    <u3>0.0</u3>
      </eulerRotation>
    </orientation>
    <velocity basis="PendulumBob">
40    <v1 units="m/sec">0.0</v1>
      <v2 units="m/sec">0.0</v2>
      <v3 units="m/sec">0.0</v3>
    </velocity>
    <angularVelocity basis="PendulumBob">
45    <omega1 units="rad/sec">0.0</omega1>
      <omega2 units="rad/sec">0.0</omega2>
      <omega3 units="rad/sec">0.0</omega3>
    </angularVelocity>
    </initialConditions>
  </rigidBody>
```

```

50    <rigidBody>
      <name>2ndBall</name>
      <mass units="kg">2.0</mass>
      <rGB>
55        <x units="m">0.0</x>
      </rGB>
      <moi>
        <Ixx units="kg-m^2">0.008</Ixx>
        <Iyy units="kg-m^2">0.008</Iyy>
        <Izz units="kg-m^2">0.008</Izz>
60      </moi>
      <initialConditions>
        <position basis="2ndBall">
          <x units="m">0.0</x>
          <y units="m">0.0</y>
65          <z units="m">10.0</z>
        </position>
        <orientation>
          <eulerRotation>
70            <phi units="deg">135.0</phi>
            <u1>0.0</u1>
            <u2>1.0</u2>
            <u3>0.0</u3>
          </eulerRotation>
        </orientation>
75      <velocity basis="2ndBall">
        <v1 units="m/sec">0.0</v1>
        <v2 units="m/sec">0.0</v2>
        <v3 units="m/sec">0.0</v3>
      </velocity>
80      <angularVelocity basis="2ndBall">
        <omega1 units="rad/sec">0.0</omega1>
        <omega2 units="rad/sec">0.0</omega2>
        <omega3 units="rad/sec">0.0</omega3>
      </angularVelocity>
85    </initialConditions>
  </rigidBody>
</bodies>
<joints>

90    <spherical>
      <name>Pivot1</name>
      <parent>
        <name>Base</name>
        <position basis="Base">
95          <x units="m">0.0</x>
          <y units="m">0.0</y>
          <z units="m">0.0</z>
        </position>
      </parent>
100    <child>
      <name>PendulumBob</name>

```

```

    <position basis="PendulumBob">
      <x units="m">0.0</x>
      <y units="m">0.0</y>
105     <z units="m">-5.0</z>
    </position>
  </child>
  <initialConditions>
    <reaction basis="Base">
110     <xForce units="N">0.0</xForce>
      <yForce units="N">0.0</yForce>
      <zForce units="N">0.0</zForce>
    </reaction>
  </initialConditions>
115 </spherical>
<spherical>
  <name>Pivot2</name>
  <parent>
    <name>PendulumBob</name>
120    <position basis="PendulumBob">
      <x units="m">0.0</x>
      <y units="m">0.0</y>
      <z units="m">0.0</z>
    </position>
125    </parent>
  <child>
    <name>2ndBall</name>
    <position basis="2ndBall">
130     <x units="m">0.0</x>
      <y units="m">0.0</y>
      <z units="m">-5.0</z>
    </position>
  </child>
  <initialConditions>
135    <reaction basis="PendulumBob">
      <xForce units="N">0.0</xForce>
      <yForce units="N">0.0</yForce>
      <zForce units="N">0.0</zForce>
    </reaction>
140    </initialConditions>
  </spherical>
</joints>
</system>
<simulation>
145 <title>2Pend</title>
  <deltaT units="sec">0.002</deltaT>
  <tFinal>100.0</tFinal>
</simulation>
</data>

```

Appendix C. Triple Pendulum XML Input

Listing C.1: XML input file for a triple pendulum

```
1  <?xml version="1.0" encoding="UTF-8"?>
   <data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:noNamespaceSchemaLocation="/home/hainge/msaHL/...
       msaHLData.xsd">
     <unitSystem>SI</unitSystem>
     <system>
6      <gravity>
        <g3 units="m/sec^2">9.80665</g3>
      </gravity>
        <bodies>
          <ground>
11         <name>Base</name>
          </ground>
          <rigidBody>
            <name>PendulumBob</name>
            <mass units="kg">2.0</mass>
16         <rGB>
           <x units="m">0.0</x>
         </rGB>
          <moi>
            <Ixx units="kg-m^2">0.008</Ixx>
21         <Iyy units="kg-m^2">0.008</Iyy>
            <Izz units="kg-m^2">0.008</Izz>
          </moi>
          <initialConditions>
            <position basis="PendulumBob">
26         <x units="m">0.0</x>
           <y units="m">0.0</y>
           <z units="m">5.0</z>
         </position>
            <orientation>
31         <eulerRotation>
           <phi units="deg">05.0</phi>
           <u1>0.0</u1>
           <u2>1.0</u2>
           <u3>0.0</u3>
36         </eulerRotation>
        </orientation>
            <velocity basis="PendulumBob">
              <v1 units="m/sec">0.0</v1>
              <v2 units="m/sec">0.0</v2>
41         <v3 units="m/sec">0.0</v3>
            </velocity>
            <angularVelocity basis="PendulumBob">
              <omega1 units="rad/sec">0.0</omega1>
              <omega2 units="rad/sec">0.0</omega2>
46         <omega3 units="rad/sec">0.0</omega3>
            </angularVelocity>
          </initialConditions>
        </rigidBody>
```

```

51 <rigidBody>
    <name>2ndBall</name>
    <mass units="kg">2.0</mass>
    <rGB>
        <x units="m">0.0</x>
    </rGB>
56 <moi>
    <Ixx units="kg-m^2">0.008</Ixx>
    <Iyy units="kg-m^2">0.008</Iyy>
    <Izz units="kg-m^2">0.008</Izz>
</moi>
61 <initialConditions>
    <position basis="2ndBall">
        <x units="m">0.0</x>
        <y units="m">0.0</y>
        <z units="m">10.0</z>
66 </position>
    <orientation>
        <eulerRotation>
            <phi units="deg">05.0</phi>
            <u1>0.0</u1>
71 <u2>1.0</u2>
            <u3>0.0</u3>
        </eulerRotation>
    </orientation>
    <velocity basis="2ndBall">
76 <v1 units="m/sec">0.0</v1>
        <v2 units="m/sec">0.0</v2>
        <v3 units="m/sec">0.0</v3>
    </velocity>
    <angularVelocity basis="2ndBall">
81 <omega1 units="rad/sec">0.0</omega1>
        <omega2 units="rad/sec">0.0</omega2>
        <omega3 units="rad/sec">0.0</omega3>
    </angularVelocity>
    </initialConditions>
86 </rigidBody>

<rigidBody>
    <name>3rdBall</name>
    <mass units="kg">2.0</mass>
91 <rGB>
    <x units="m">0.0</x>
    </rGB>
    <moi>
        <Ixx units="kg-m^2">0.008</Ixx>
96 <Iyy units="kg-m^2">0.008</Iyy>
        <Izz units="kg-m^2">0.008</Izz>
    </moi>
    <initialConditions>
        <position basis="3rdBall">
101 <x units="m">0.0</x>

```

```

        <y units="m">0.0</y>
        <z units="m">15.0</z>
    </position>
    <orientation>
106     <eulerRotation>
        <phi units="deg">05.0</phi>
        <u1>0.0</u1>
        <u2>1.0</u2>
        <u3>0.0</u3>
111     </eulerRotation>
    </orientation>
    <velocity basis="3rdBall">
        <v1 units="m/sec">0.0</v1>
        <v2 units="m/sec">0.0</v2>
116     <v3 units="m/sec">0.0</v3>
    </velocity>
    <angularVelocity basis="3rdBall">
        <omega1 units="rad/sec">0.0</omega1>
        <omega2 units="rad/sec">0.0</omega2>
121     <omega3 units="rad/sec">0.0</omega3>
    </angularVelocity>
    </initialConditions>
</rigidBody>
</bodies>
126 <joints>

    <spherical>
        <name>Pivot</name>
        <parent>
131     <name>Base</name>
        <position basis="Base">
            <x units="m">0.0</x>
            <y units="m">0.0</y>
            <z units="m">0.0</z>
136     </position>
        </parent>
        <child>
            <name>PendulumBob</name>
            <position basis="PendulumBob">
141     <x units="m">0.0</x>
            <y units="m">0.0</y>
            <z units="m">-5.0</z>
            </position>
        </child>
146     <initialConditions>
        <reaction basis="Base">
            <xForce units="N">0.0</xForce>
            <yForce units="N">0.0</yForce>
            <zForce units="N">0.0</zForce>
151     </reaction>
        </initialConditions>
    </spherical>

```

```

156 <spherical>
    <name>Pivot2</name>
    <parent>
        <name>PendulumBob</name>
        <position basis="PendulumBob">
            <x units="m">0.0</x>
            <y units="m">0.0</y>
161         <z units="m">0.0</z>
        </position>
    </parent>
    <child>
        <name>2ndBall</name>
166         <position basis="2ndBall">
            <x units="m">0.0</x>
            <y units="m">0.0</y>
            <z units="m">-5.0</z>
        </position>
171     </child>
    <initialConditions>
        <reaction basis="PendulumBob">
            <xForce units="N">0.0</xForce>
            <yForce units="N">0.0</yForce>
176         <zForce units="N">0.0</zForce>
        </reaction>
    </initialConditions>
</spherical>
<spherical>
181     <name>Pivot3</name>
    <parent>
        <name>2ndBall</name>
        <position basis="2ndBall">
            <x units="m">0.0</x>
            <y units="m">0.0</y>
186         <z units="m">0.0</z>
        </position>
    </parent>
    <child>
191         <name>3rdBall</name>
        <position basis="3rdBall">
            <x units="m">0.0</x>
            <y units="m">0.0</y>
            <z units="m">-5.0</z>
196         </position>
    </child>
    <initialConditions>
        <reaction basis="2ndBall">
            <xForce units="N">0.0</xForce>
            <yForce units="N">0.0</yForce>
201         <zForce units="N">0.0</zForce>
        </reaction>
    </initialConditions>
</spherical>

```

```
206     </joints>
      </system>
      <simulation>
        <title>3Pend</title>
        <deltaT units="sec">0.1</deltaT>
211     <tFinal>10.0</tFinal>
      </simulation>
    </data>
```

Appendix D. Spinning Top XML Input

Listing D.1: XML input file for a spinning top

```
<?xml version="1.0" encoding="UTF-8"?>
2 <data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="/home/hainge/msaHL/...
    msaHLData.xsd">
  <unitSystem>SI</unitSystem>
  <system>
    <gravity>
7      <g3 units="m/sec^2">-3.0</g3>
    </gravity>
    <bodies>
      <ground>
        <name>Base</name>
12      </ground>
      <rigidBody>
        <name>PendulumBob</name>
        <mass units="kg">1.0</mass>
        <rGB>
17      <x units="m">0.0</x>
        </rGB>
        <moi>
          <Ixx units="kg-m^2">0.4</Ixx>
          <Iyy units="kg-m^2">0.4</Iyy>
22      <Izz units="kg-m^2">0.75</Izz>
        </moi>
        <initialConditions>
          <position basis="PendulumBob">
            <x units="m">0.0</x>
27      <y units="m">0.0</y>
            <z units="m">0.2</z>
          </position>
          <orientation>
            <eulerRotation>
32      <phi units="deg">10.0</phi>
            <u1>0.0</u1>
            <u2>1.0</u2>
            <u3>0.0</u3>
            </eulerRotation>
37      </orientation>
          <velocity basis="PendulumBob">
            <v1 units="m/sec">0.0</v1>
            <v2 units="m/sec">0.0</v2>
            <v3 units="m/sec">0.0</v3>
42      </velocity>
          <angularVelocity basis="PendulumBob">
            <omega1 units="rad/sec">0.0</omega1>
            <omega2 units="rad/sec">0.9888</omega2>
            <omega3 units="rad/sec">7.5167</omega3>
47      </angularVelocity>
          </initialConditions>
        </rigidBody>
```

```

    </bodies>
    <joints>
52      <spherical>
        <name>Pivot</name>
        <parent>
          <name>Base</name>
          <position basis="Base">
57            <x units="m">0.0</x>
            <y units="m">0.0</y>
            <z units="m">0.0</z>
          </position>
        </parent>
62      <child>
        <name>PendulumBob</name>
        <position basis="PendulumBob">
          <x units="m">0.0</x>
          <y units="m">0.0</y>
67          <z units="m">-0.2</z>
        </position>
      </child>
      <initialConditions>
        <reaction basis="Base">
72          <xForce units="N">0.0</xForce>
          <yForce units="N">0.0</yForce>
          <zForce units="N">0.0</zForce>
        </reaction>
      </initialConditions>
77    </spherical>
  </joints>
</system>
<simulation>
  <title>SphericalPendulum</title>
82  <deltaT units="sec">0.1</deltaT>
  <tFinal>10.0</tFinal>
</simulation>
</data>

```

Bibliography

1. Agrawal, Venkata R. Sonti O. P. “Design Sensitivity Analysis of Dynamic Systems Using Hamilton’s Law of Varying Action”. *International Journal of Mechanical Sciences*, 37, No 6:601–613, 1994.
2. Atluri, M. Borri F. Mello S. N. “Primal and mixed forms of Hamilton’s principle for constrained rigid body systems: numerical studies”. *Computational Mechanics*, 7:205–220, 1991.
3. Bauchau, Olivier A. “A self-stabilized algorithm for enforcing constraints in multi-body systems”. *International Journal of Solids and Structures*, 40:3253–3271, 2003.
4. Bauchau, Olivier A. Jou-Young Choi. *The Vector Parameterization of Motion*. Technical report, Georgia Institute of Technology, School of Aerospace Engineering, 2003.
5. Bless, Dewey H. Hodges Robert R. “A Weak Hamiltonian Finite Element Method for Optimal Control Problems”. *Journal of Guidance, Control, and Dynamics*, 14, No 1:148–156, 1991.
6. Flannery, M. R. *The enigma of nonholonomic constraints*. Technical report, School of Physics, Georgia Institute of Technology, 2004.
7. Kunz, Donald L. “An object oriented approach to multibody systems analysis”. *Computers and Structures*, 69:209–217, 1998.
8. Kunz, Donald L. “Multibody System Analysis Based on Hamilton’s Weak Principle”. *AIAA Journal*, 39, No12:2382–2388, 2001.
9. Kunz, Donald L. “Implementation of a Generalized Multibody Approach for Analysis of Dynamic Systems”. volume AIAA 2005-2348. 2005.
10. Kunz, Donald L. “A numerical Investigation of Constrained Direct Solutions Using Hamilton’s Law”. volume AIAA 2009-2576. AIAA, 2009.
11. Press, Columbia University (editor). *Conversion and Conservation of Energy*. The Columbia Electronic Encyclopedia, 6th Ed., 2007. URL <http://www.infoplease.com/ce6/sci/A0857980.html>.
12. Riff, Menahem Baruch Richard. “Hamilton’s Principle, Hamilton’s Law - 6ⁿ Correct Formulations”. *AIAA Journal*, 20, No 5:687–691, 1982.

Vita

Capt Ashton Hainge graduated from Purdue University with a BS in Mechanical Engineering. For his first assignment, Captain Hainge was sent to the Electronic Systems Center at Hanscomb AFB, MA where he served as a command and control modeling and simulation engineer. His next assignment was at the Air Force Operational Test and Evaluation Center (AFOTEC) where Capt Hainge served as a flight test engineer for the MQ-9 Reaper unmanned aerial system. While at Edwards Capt Hainge also worked with the B-2 Radar Modernization Program in outfitting the B-2 with an updated radar system. Captain Hainge's next assignment is located at Arnold Engineering and Development Center near Tullahoma, TN where he will work in super sonic wind tunnel testing.

Permanent address: 2950 Hobson Way
Air Force Institute of Technology
Wright-Patterson AFB, OH 45433

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 074-0188	
The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of the collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.					
1. REPORT DATE (DD-MM-YYYY) 25-03-2010		2. REPORT TYPE Master's Thesis		3. DATES COVERED (From – To) August 2008 – March 2010	
TITLE AND SUBTITLE Validation of a Novel Approach to Solving Multibody Systems Using Hamilton's Weak Principle				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S) Hainge, Ashton D., Captain, USAF				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/ENY) 2950 Hobson Way, Building 640 WPAFB OH 45433-8865				8. PERFORMING ORGANIZATION REPORT NUMBER AFIT/GAE/ENY/10-M10	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) POC: Dr. David Stargel, Phone: (703) 696-6961 Email: david.stargel@afosr.af.mil Air Force Office of Scientific Research 875 N. Randolph, Suite 325, Rm 3112 Arlington, VA 22205				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT A novel approach for formulating and solving for the dynamic response of multibody systems has been developed using Hamilton's Law of Varying Action as its unifying principle. In order to assure that the associated computer program is sufficiently robust when applied across a wide range of dynamic systems, the program must be verified and validated. The purpose of the research was to perform the verification and validation of the program. Results from the program were compared with closed-form and numerical solutions of simple systems, such as a simple pendulum and a rotating pendulum. The accuracy of the program for complex systems for which there is no closed-form solution, such as a double pendulum and others, were assessed by calculating energy conservation and constraint violation. The results of this research confirm the validity of this novel approach to multibody system analysis, and pave the way for its application to increasingly complex configurations.					
15. SUBJECT TERMS Hamilton's Law, multibody, analysis, pendulum, chaos					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT UU	18. NUMBER OF PAGES 115	19a. NAME OF RESPONSIBLE PERSON Donald L. Kunz, Ph.D., P.E.
a. REPORT U	b. ABSTRACT U	c. THIS PAGE U			19b. TELEPHONE NUMBER (Include area code) (937) 255-3636, ext 4548 (Donald.Kunz@afit.edu)