Development of MATLAB-Based 3D Multibody Kinematics and Dynamics Simulation Engine

Michael F. Vignos

University of Wisconsin-Madison

# Table of Contents

# Introduction

# Problem Statement

The goal of this work was to develop a 3D multibody kinematics and dynamics simulation engine within MATLAB. The theory used to develop this simulation engine was taken from the following sources:

1. Haug, Edward J. *Computer Aided Kinematics and Dynamics of Mechanical Systems*. Boston: Allyn and Bacon, 1989. Print.
2. Negrut, Dan. *Mechanical Engineering 751: Advanced Computational Multibody Dynamics.* University of Wisconsin-Madison, Madison, WI. Fall 2016. Lecture.

# Simulation Code Framework

As previously mentioned, all code for this simulation engine was developed using MATLAB R2014b. It is likely that this code will function properly for future versions of MATLAB as the functions used are not version specific. However, this has not been verified.

## Object-Oriented Programming

The framework of this code was written using object-oriented programming. This approach allows for definition of MATLAB classes, which allows for the grouping of commonly used functions and which create a hierarchical structure in the definition of a multi

## Example Model Definition

# System Constraints

## Kinematic Constraints

### Basic Constraints

### Intermediate Constraints

### Joints

## Driving Constraints

# Externally Applied Forces and Torques

# Analysis of Mechanisms

## Kinematics Analysis

## Dynamics Analysis

### Methods of Computing Iteration Matrix

## Inverse Dynamics Analysis

## Assembly Analysis

## Prescribing Initial Velocities

# Validation Efforts

## Validation of Joints

## Comparison to Benchmark Problems

# 

Things to do still:

~~Implement simple pendulum~~

~~Implement N-bar mechanism~~

~~Implement method to prescribe initial velocities~~

~~Implement example to validate a cylindrical joint (could maybe be a simple pendulum with a cylindrical joint instead of a revolute joint??)~~

Implement method to prescribe angular velocity at a joint, rather than using a DP1 constraint. A DP1 constraint is plagued with too many singularity issue.

~~Implement method to remove redundant constraints??~~ Not sure how robust my method is.

Implement ability to read model parameters from a file???

Flyball governor mechanism?

Bricard’s mechanism?

Andrew’s mechanism?

Things done:

~~Implemented ability to prescribe all basic constraints, all intermediate constraints, and all joints discussed in class.~~

~~Implemented ability to perform kinematics, inverse dynamics, and dynamics analyses.~~

~~Validated all joints~~

~~Validated ability to prescribe constant torque  
Implemented kinematics, inverse dynamics, and dynamics analyses~~

~~Implemented different methods of computing iteration matrix~~

~~Implemented various test cases validated by book and by group that created the benchmark problems~~

~~Implemented method to prescribe initial velocities for dynamics analysis.~~