Autonomous Vehicle Simulation (AVS) Laboratory, University of Colorado

Basilisk Technical Memorandum

Document ID: Basilisk-spacecraftDynamics

MULTI-SPACECRAFT EQUATIONS OF MOTION MODEL

Prepared by	C. Allard

Status: To be Reviewed
Scope/Contents
SpacecraftDynamics is an instantiation of the dynamicObject abstract class. This abstract class is representing systems that have equations of motion that need to be integrated and therefore the main goal of this dynamic object is finding the state derivatives and interfacing with the integrator to integrate the state forward in time. The spacecraftDynamics is specifically geared towards simulating multiple spacecraft at a time that will allow for docking and detaching of spacecraft. If spacecraftPlus is ever removed then this will be the sole dynamicObject that will control the spacecraft dynamics of the system. This current module is IN PROGRESS and burning edge development in its current state.

Rev	Change Description	By	Date
1.0	Initial Draft (Module is still IN PROGRESS)	C. Allard	20180613

Contents

1 Progress Explanation and Current Status **1**

2 Model Description **3**
2.1 Introduction 3

3 Model Functions **3**

4 Model Assumptions and Limitations **3**

5 Test Description and Success Criteria **5**
5.1 SCConnected Test 5

6 SCConnectedAndUnconnected Test **5**

7 Test Results **5**
7.1 SCConnected Test 5
7.2 SCConnectedAndUnconnected Test 8

8 User Guide **15**

---

1 Progress Explanation and Current Status

This module is IN PROGRESS and this section lists the current progress and status of the module as capability is added to the module this document needs to be updated. This document should always be describing the current status of the module.

Below is a list describing the limitations and status of the module

- The first major status that needs to be understood is that the HingedRigidBody stateEffector is the only stateEffector at this time that is ready to be used with the multi-spacecraft architecture. The reason that each stateEffector needs to be updated is because it was found with stateEffectors that have multiple degrees of freedom change the generalized interaction between spacecraftDynamics and the individual stateEffectors. However, this change is relatively small to each stateEffector as long as the developer understands the steps to make the change. The architecture is proven to work with hingedRigidBodies and each stateEffector just needs to follow suit.

  So in other words, if a stateEffector wants the capability of being able to be simulated in the mult-spacecraft environment some modifications need to be implemented to the specific stateEffectors. The hingedRigidBody stateEffector acts as a detailed example on the method to convert a stateEffector to allow for the mult-spacecraft interaction. Below is a further on how to implement this change:

  - The prependSpacecraftNameToStates() method needs to be implemented for the corresponding stateEffector states just like hingedRigidBody does for its states. This allows for the stateEffector to know about the individual spacecraft that it is attached to.

- The linkInStates method needs to prepend the linked states from the spacecraft it is attached to, to get the corresponding state from the spacecraft. However, it should be noted that you can see the hingedRigidBody no longer needs access to the hubs primary state variables. This is because the backSubstitution methods now provide the stateEffectors with the appropriate Hub states in the method calls. Therefore there is not a need for stateEffectors to link to any of the Hubs state variables.
- The last step is to provide the contributions from the stateEffectors about point P and frame $\mathcal{P}$. Again, this cannot be generalized because of multiple degree of freedom effectors so each effector has the job of being provide their contributions about point P and frame $\mathcal{P}$. It should be noted that if the spacecraft is independent then point B will be coincident with point P and dcmPB will be identity. Therefore the math should work for both connected and unconnected spacecraft. Let me explain this in more detail with the hingedRigidBody:

  You can see that in the private variables in hingedRigidBody, that the variables have been changed to be point P and $\mathcal{P}$ relative. Again, if single spacecraft is being implemented then frames $\mathcal{B}$ and $\mathcal{P}$ are coincident. In the stateEffector.h it shows that each stateEffector has access to these two crucial variables: r_BP_P and dcm_BP. r_BP_P is defaulted to zero and dcm_BP is defaulted to identity. With that knowledge, each stateEffector needs to use those variables and defined their contributions about about point P and frame $\mathcal{P}$. about point P and frame $\mathcal{P}$. The easiest way to implement this correctly, is to to change any internal variables (hopefully they are already indicated as private variables) to point P and frame $\mathcal{P}$ relative. Therefore, values being set by python shouldn't be changed, but internal variables or intermediate variables should be changed.
- Once a model has been changed to allow for multi spacecraft, it can be tested by using the test_multi-spacecraft and add the stateEffector to the spacecraft system just like hinged-RigidBodies are. If energy and momentum is conserved the stateEffector most likely was changed correctly. Obviously all other tests should pass.

The following table describes the current status of each stateEffector being implemented in the multi-spacecraft environment. This table should be updated as stateEffectors have this added capability.

Table 2: Test results.

Test	Pass/Fail	
HingedRigidBodies	Implemented	Tested
LinearSpringMassDamper	Not Implemented	Not Tested
SphericalPendulum	Not Implemented	Not Tested
FuelTank	Not Implemented	Not Tested
VSCMGs	Not Implemented	Not Tested
dualHingedRigidBodies	Not Implemented	Not Tested
nHingedRigidBodies	Not Implemented	Not Tested

- DynamicsEffectors need the same point P and frame $\mathcal{P}$ relative math to be implemented. This would be done very similar to the hingedRigidBody effectors and once it has been implemented for the stateEffectors the jump to dynamicEffector is small. But as such a current progress table is shown below:

Table 3: Test results.

Test	Pass/Fail	
Thrusters	Not Implemented	Not Tested
ExtForceTorque	Not Implemented	Not Tested
ExtPulsedForceTorque	Not Implemented	Not Tested
dragEffector	Not Implemented	Not Tested
radiationPressure	Not Implemented	Not Tested

- On further evaluation, there is a bug believed to be in spacecraftDynamics with the multi-connected hubs. The attached hubEffectors needs to provide their contribution to the primary spacecraft about point P and $\mathcal{P}$. Just like the other stateEffectors the hubEffector needs to be changed. This should be a small change, but nees to be done! The tests in this document do not show this bug because the contributions are not being added in spacecraftDynamics. So to be clear there is a bug in both spacecraftDynamics (because the attached hubs are not being looped over) and in the hubEffector (not providing contributions about point P and $\mathcal{P}$) This is BUG that needs to be fixed.

- The docking and detachment of spacecraft needs to be implemented in the spacecraftDynamics class. Currently the docking and detaching could be done using python (by stopping the sim at a specific time step and dock/detach from python) but this is not the intended use of this multi-spacecraft architecture. This NEEDS to be implemented in the spacecraftDynamics class that would allow for a message to be written out to the system which would allow for detachment and docking of the spacecraft. The kinematics of the docking locations would need to be taken into account in this docking/detaching method. (DockingData is the struct that holds the docking information of each dock) However, once this method is added, the rest of architecture would allow for the docking and detaching to happen seamlessly. The docking and detaching method should be called at the end of the SpacecraftDynamics::integrateState method.

- The interaction between sensors and environment models has yet to be tested for the multi-spacecraft architecture.

2 Model Description

2.1 Introduction

A detailed descrtiption of this architecture can be seen in: http://hanspeterschaub.info/Papers/grads/CodyAllard.pdf. Figure 1 shows a schematic of what this module is intended to to do: simulate multiple spacecraft at a time that can dock and detach throughout the simulation.

3 Model Functions

The model functions need to be updated once the module has been finalized and fully tested

4 Model Assumptions and Limitations

Please see the first section describing current progress of this module.

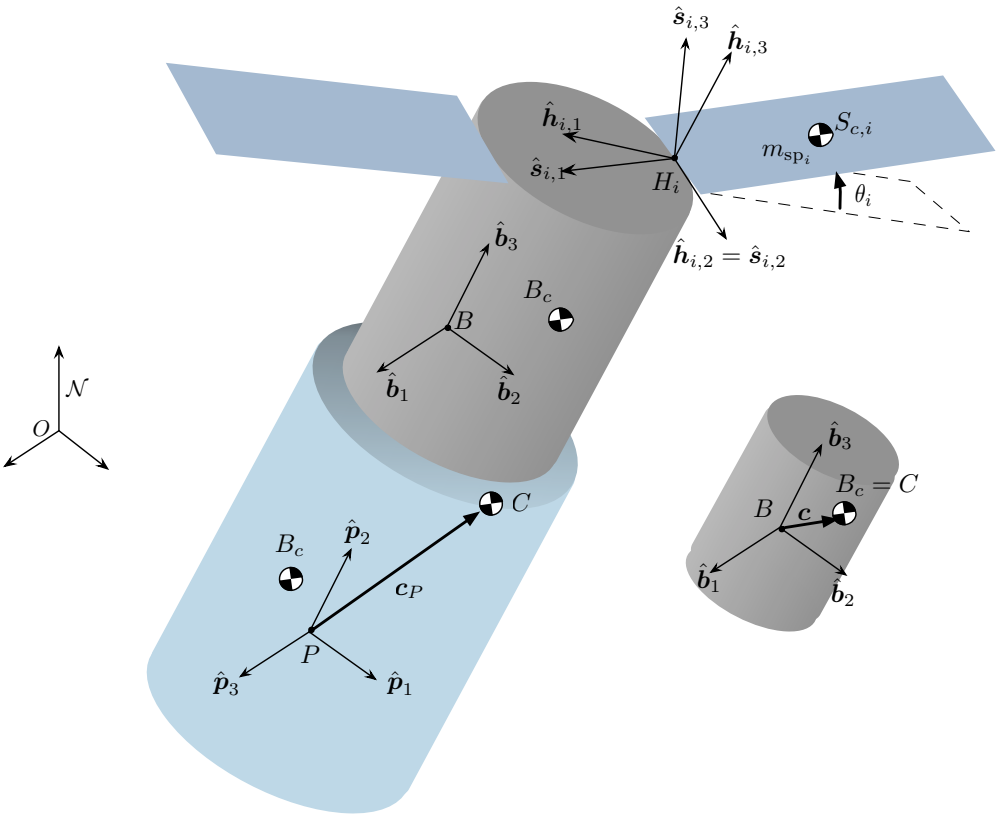

Fig. 1: Complex multi-spacecraft that can be simulated using spacecraftDynamics

5 Test Description and Success Criteria

This test is located in `simulation/dynamics/spacecraftDynamics/_UnitTest/test_multiSpacecraft.py`.

5.1 SCConnected Test

This test ensures that a spacecraft system with hingedRigidBodies attached to it conserves energy and momentum.

6 SCConnectedAndUnconnected Test

This test ensures that a spacecraft system with hingedRigidBodies attached to it and other unconnected spacecraft can be simulated at the same time and conserve energy and momentum.

7 Test Results

7.1 SCConnected Test

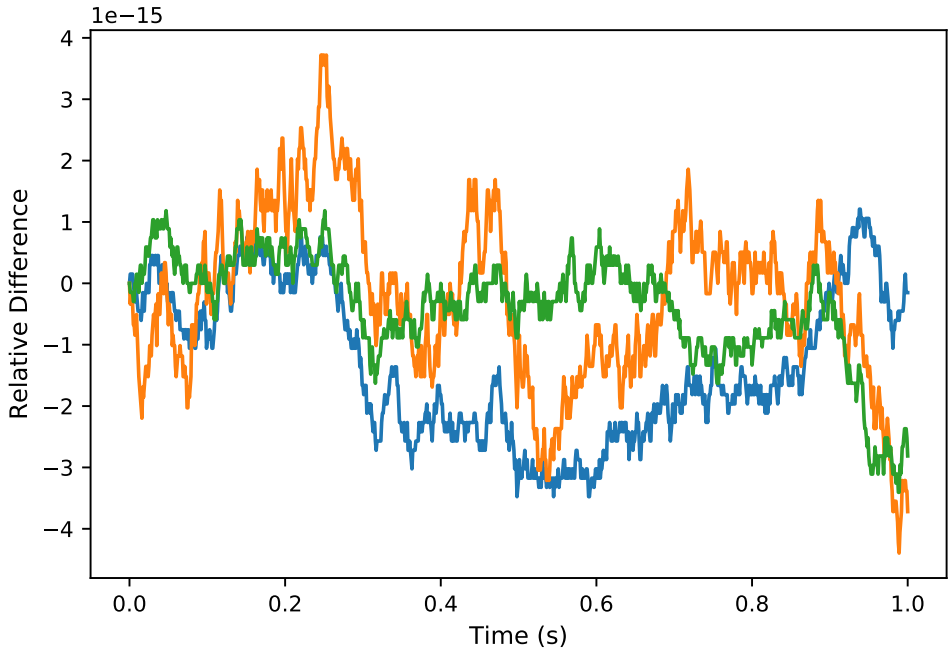

Fig. 2: Change in Orbital Angular Momentum System with Gravity

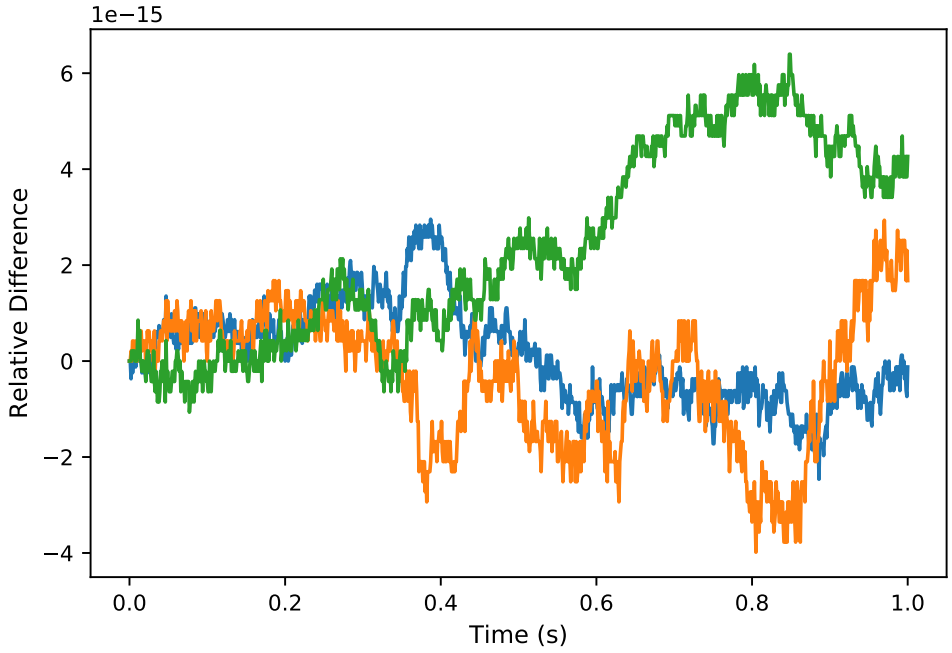

Fig. 3: Change In Rotational Angular Momentum System with Gravity

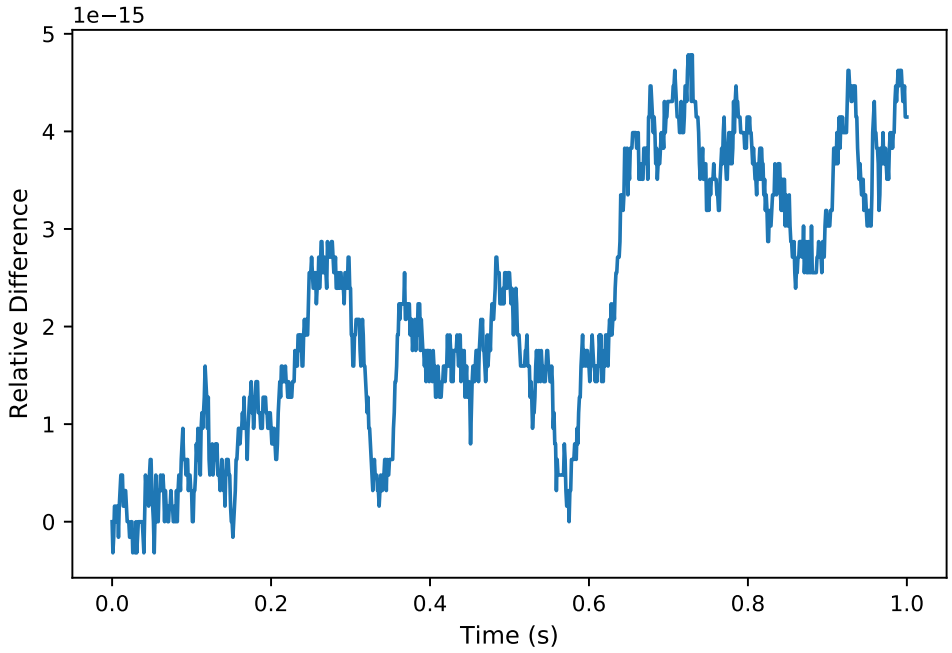

Fig. 4: Change In Rotational Energy System with Gravity

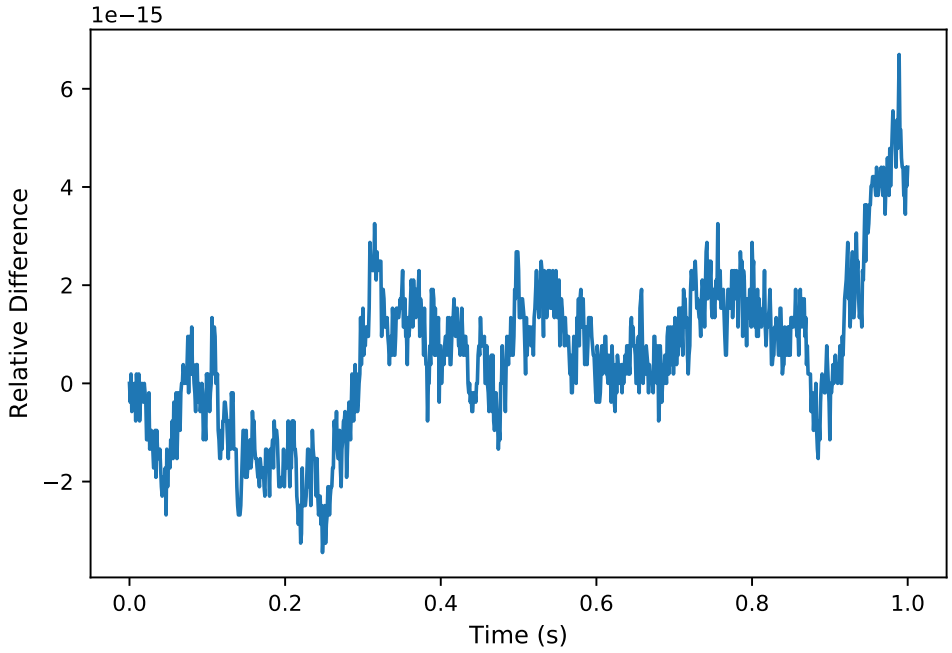

Fig. 5: Change in Orbital Energy System with Gravity

7.2 SCConnectedAndUnconnected Test

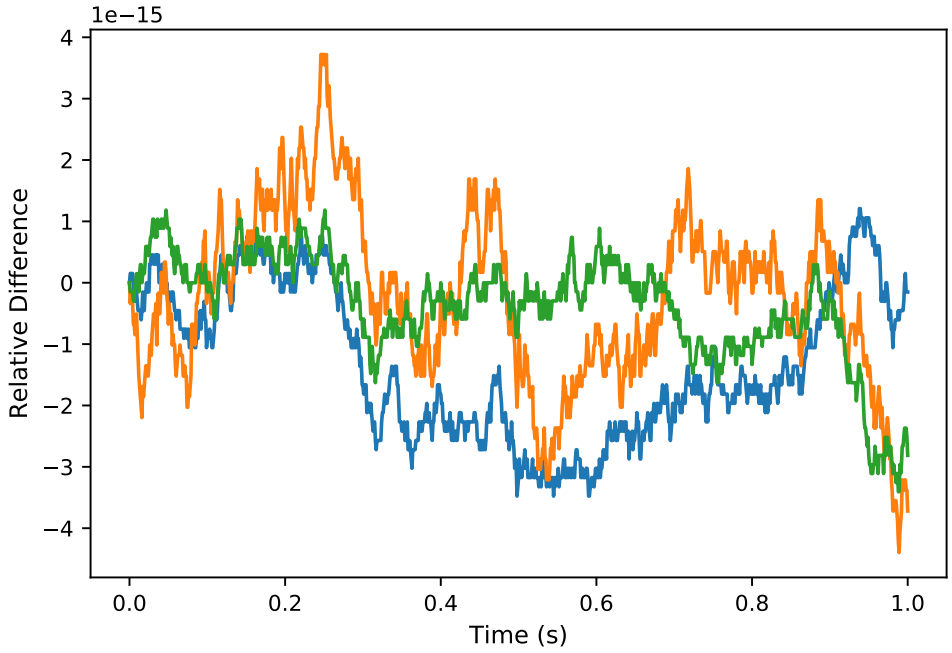

Fig. 6: Change in Orbital Angular Momentum System with Gravity

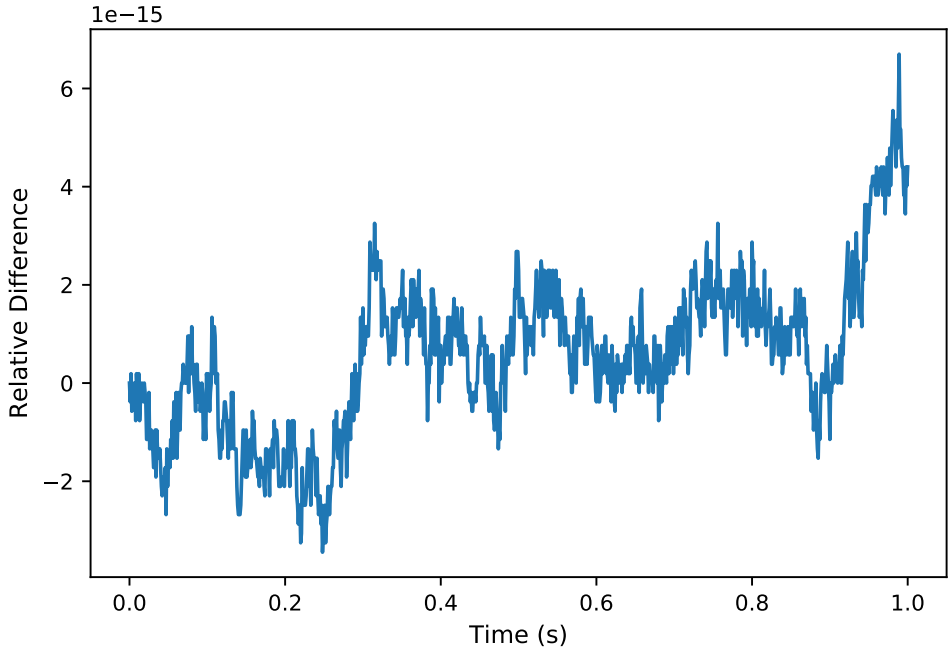

Fig. 7: Change in Orbital Energy System with Gravity

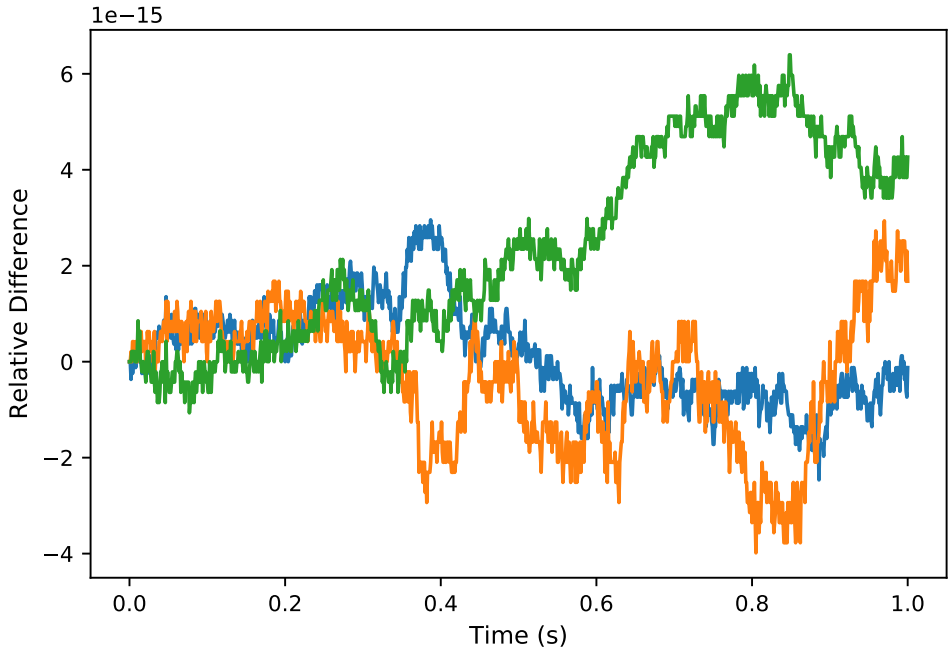

Fig. 8: Change In Rotational Angular Momentum System with Gravity

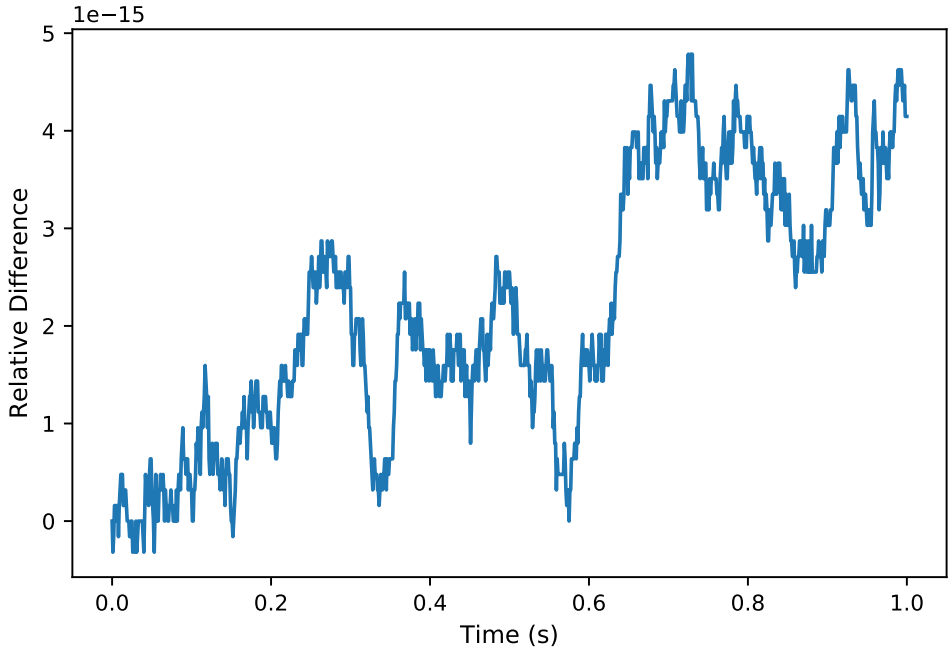

Fig. 9: Change In Rotational Energy System with Gravity

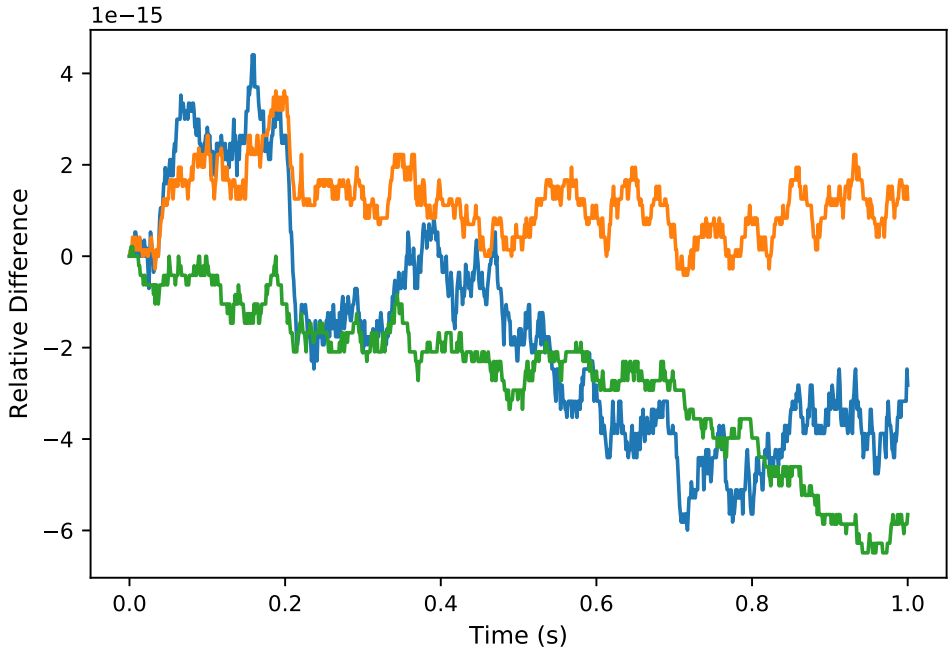

Fig. 10: Change in Orbital Angular Momentum SC 1 with Gravity

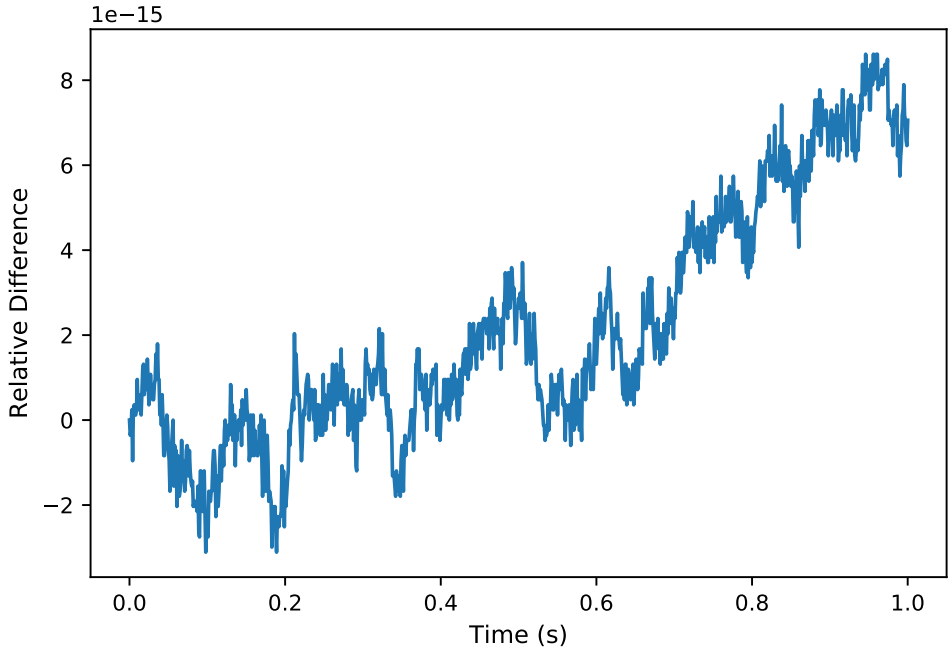

Fig. 11: Change in Orbital Energy SC 1 with Gravity

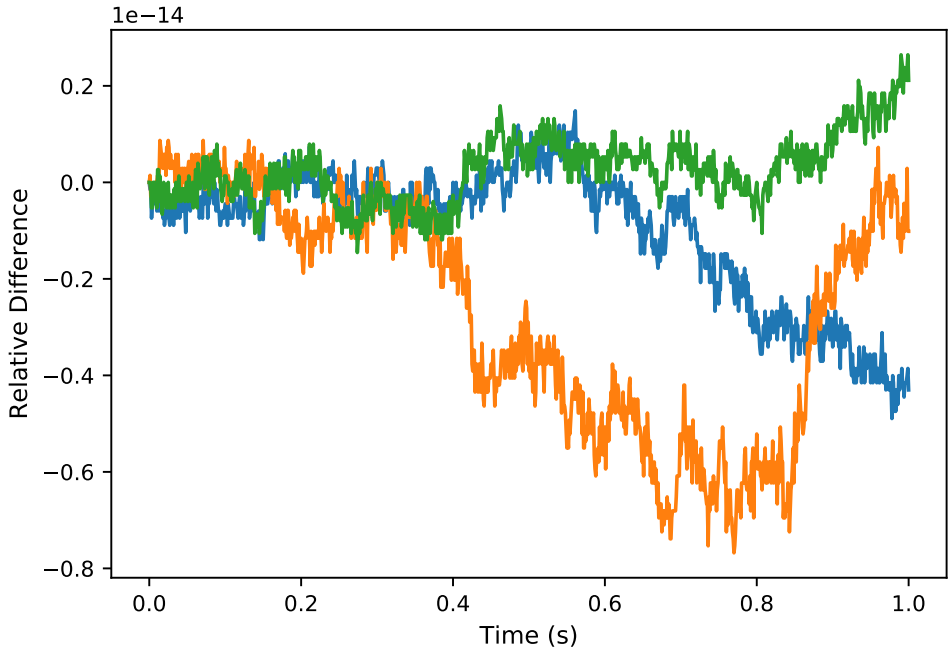

Fig. 12: Change In Rotational Angular Momentum SC 1 with Gravity

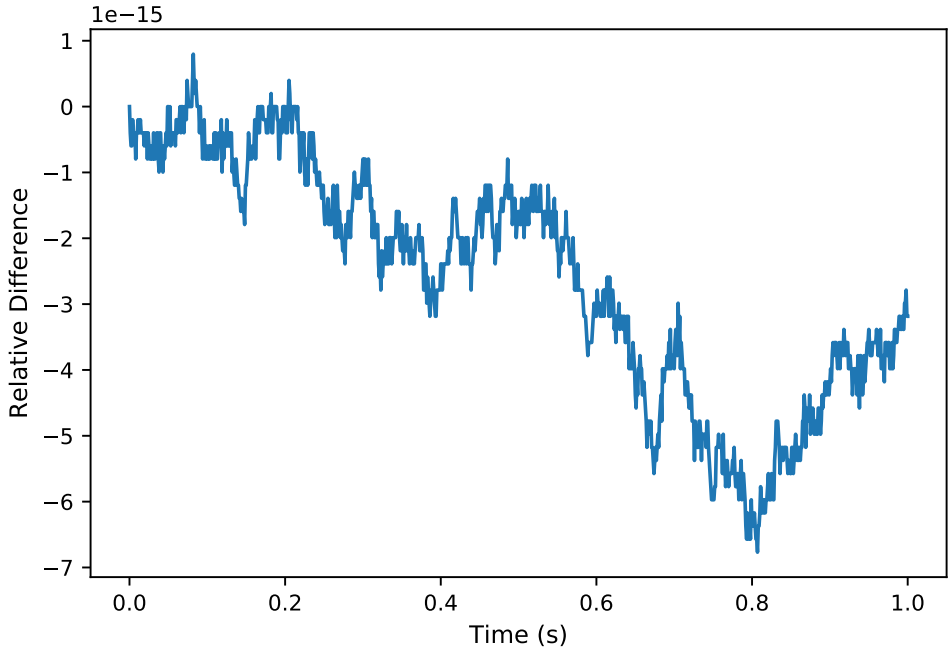

Fig. 13: Change In Rotational Energy SC 1 with Gravity

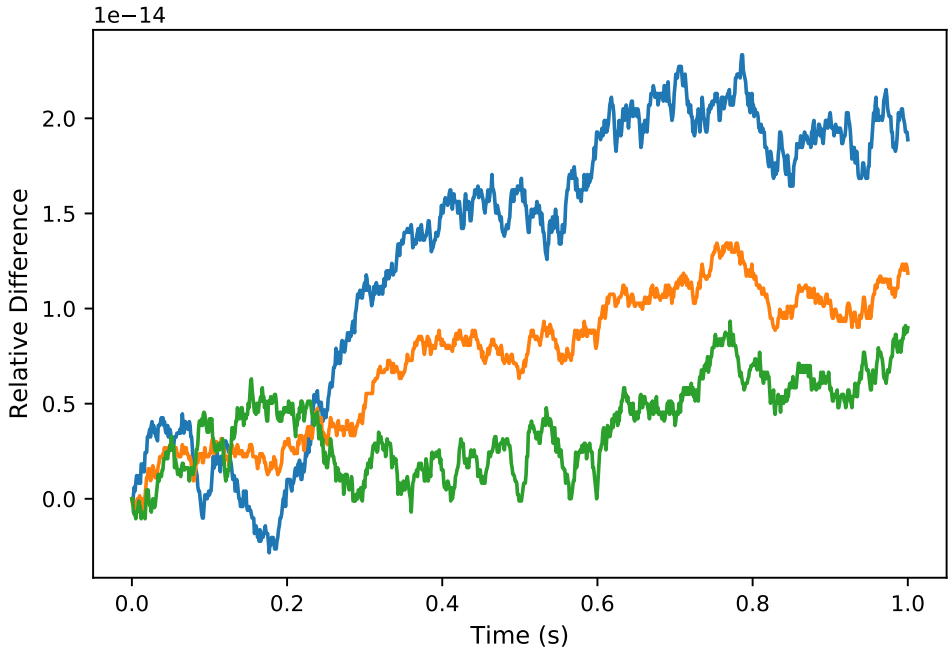

Fig. 14: Change in Orbital Angular Momentum SC 2 with Gravity

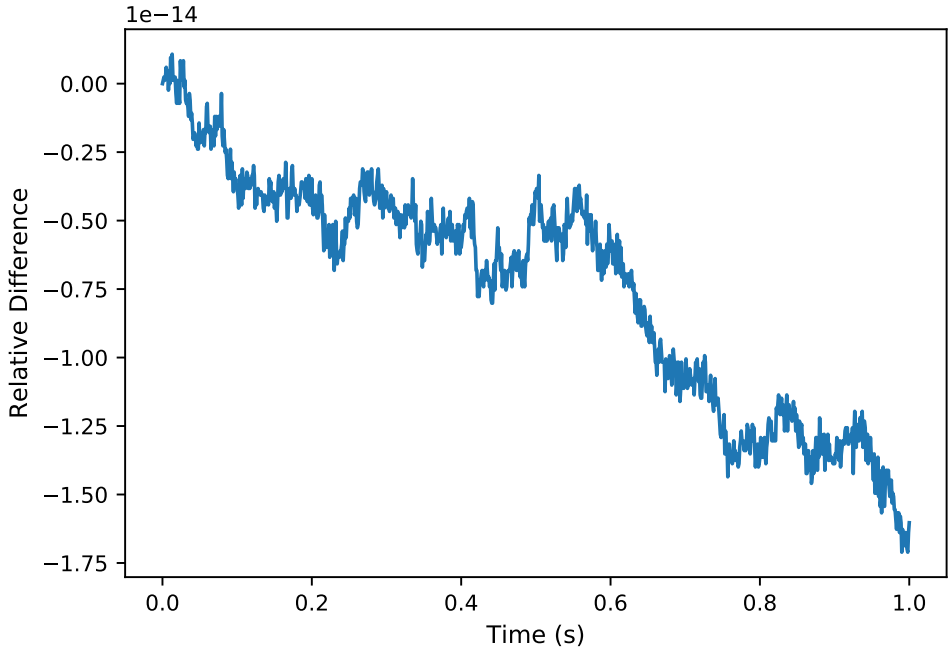

Fig. 15: Change in Orbital Energy SC 2 with Gravity

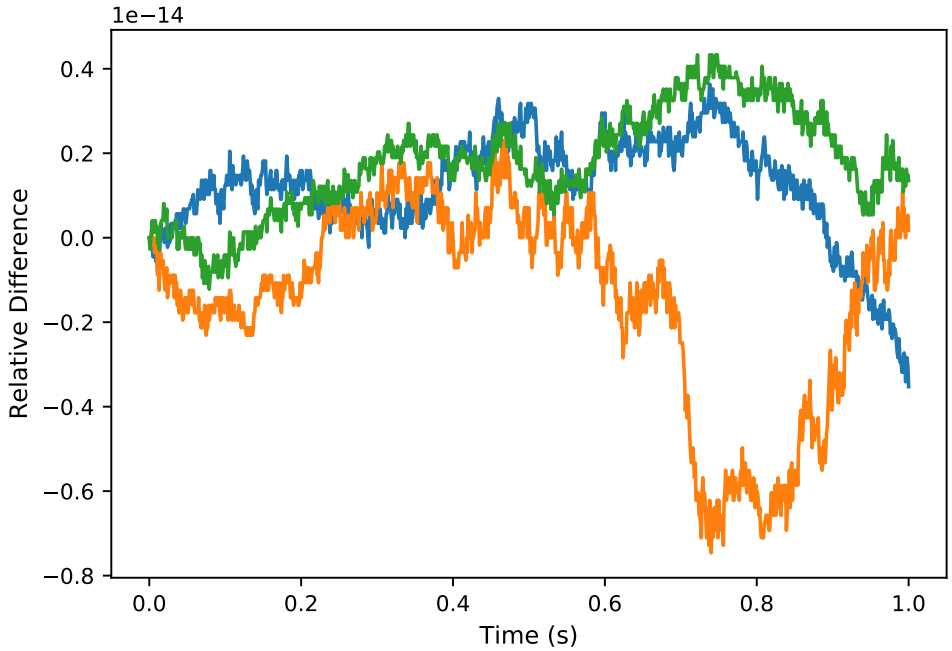

Fig. 16: Change In Rotational Angular Momentum SC 2 with Gravity

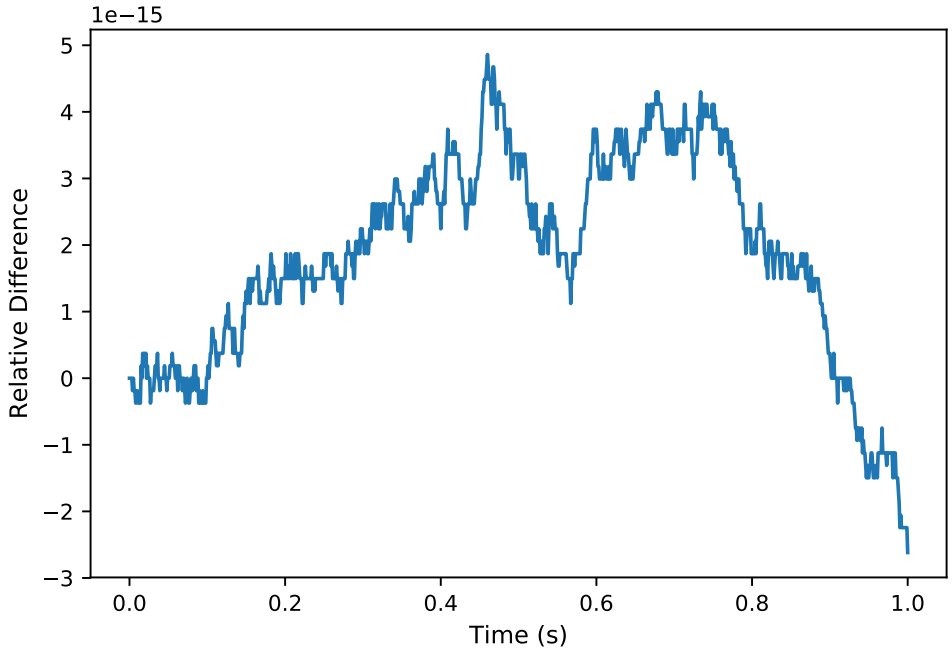

Fig. 17: Change In Rotational Energy SC 2 with Gravity

8 User Guide

This user guide will be updated further but please see the test: test_multiSpacecraft.

REFERENCES

[1] C. Allard, Hanspeter Schaub, and Scott Piggott. General hinged solar panel dynamics approximating first-order spacecraft flexing. In *AAS Guidance and Control Conference*, Breckenridge, CO, Feb. 5–10 2016. Paper No. AAS-16-156.

[2] Hanspeter Schaub and John L. Junkins. *Analytical Mechanics of Space Systems*. AIAA Education Series, Reston, VA, 3rd edition, 2014.