Autonomous Vehicle Simulation (AVS) Laboratory, University of Colorado

Basilisk Technical Memorandum

Document ID: Basilisk-RadiationPressure

SOLAR RADIATION PRESSURE C++ MODEL

Prepared by	S. Carnahan

Status: Tested
Scope/Contents
The radiation pressure module is responsible for calculating the dyamic effects of radiation pressure on a spacecraft. Effects can be calculated either by use of a simplified "cannonball" method or table look-up. A unit test has been written which checks both calculation methods against expected output values. The code is working fully and providing accurate results.

Rev	Change Description	By	Date
1.0	First draft	S. Carnahan	20170713
1.1	Mostly grammatical edits	S. Carnahan	20171016
1.2	Updated on how to specify the SRP model	H. Schaub	June 12 2018

Contents

---

1 Model Description

The radiation pressure module contains two different models for calculating the effects of solar radiation pressure on spacecraft state. Both methods of calculating solar radiation pressure have simple implementations in Basilisk using basic coefficients and assumptions. The methods of arriving at these coefficients can be complex and making the coefficients time-varying to improve accuracy can greatly increase complexity. The cannonball method used here essentially follows the mathematics described by Vallado.[1]

1.1 Radiation Pressure Model

Radiation is modeled by using the solar flux at one astronomical unit and scaling by distance from the sun relative to 1 AU. The solar flux at one AU is taken as :

$$SF_{\mathrm{AU}} = 1372.5398\left[\frac{W}{m^2}\right] \tag{1}$$

1.1.1 Cannonball Method

The cannonball model assumes the spacecraft is a simple sphere. It is the default SRP model when the `RadiationPressure` module is invoked. The radiation pressure at 1AU, p_{SR}, can be taken as the solar flux divided by the speed of light.

$$p_{SR} = \frac{SF_{\mathrm{AU}}}{c}\left[\frac{N}{m^2}\right] \tag{2}$$

Then, a "scaling factor" can be determined. This "scaling factor" is equivalent to the magnitude of the solar radiation force divided by the distance between the spacecraft and the sun:

$$\frac{|\mathbf{F}_{\text{radiation}}|}{|\mathbf{r}_{\text{sun}}|} = \frac{-c_R p_{SR} A_{\odot} AU^2}{|\mathbf{r}_{\text{sun}}|^3}\left[\frac{N}{m}\right] \tag{3}$$

$\mathbf{r}_{\text{sun}}$ is the vector from the spacecraft to the sun in the spacecraft body frame and c_R is the reflectivity. This factor is then multiplied by the position vector from the spacecraft to the sun to get the force on the spacecraft due to solar radiation pressure.

$$\mathbf{F}_{\text{radiation}} = \frac{|\mathbf{F}_{\text{radiation}}|}{|\mathbf{r}_{\text{sun}}|}\mathbf{r}_{\text{sun}}[N] \tag{4}$$

The user must provide the coefficient of reflection and the equivalent area of the spacecraft to use this method.

1.1.2 Table Look-up Method

For the table look-up method, pre-determined values of torque and force acting on the spacecraft due to radiation pressure are given. It is required that these values be given at 1AU from the sun and with a corresponding direction vector from the spacecraft to the sun in the spacecraft body frame.

The look-up works by finding the direction vector in the given tables which most closely matches the current sun heading vector in the body frame. This is done by taking the maximum of the dot products of each lookup vector entry with the sun heading vector in the body frame. As a visual demonstration, 1 shows that for some current sun heading amongst the body vector entries 1 through 8, the force and torque data corresponding to entry 8 would be chosen due to its proximity to the current sun heading.

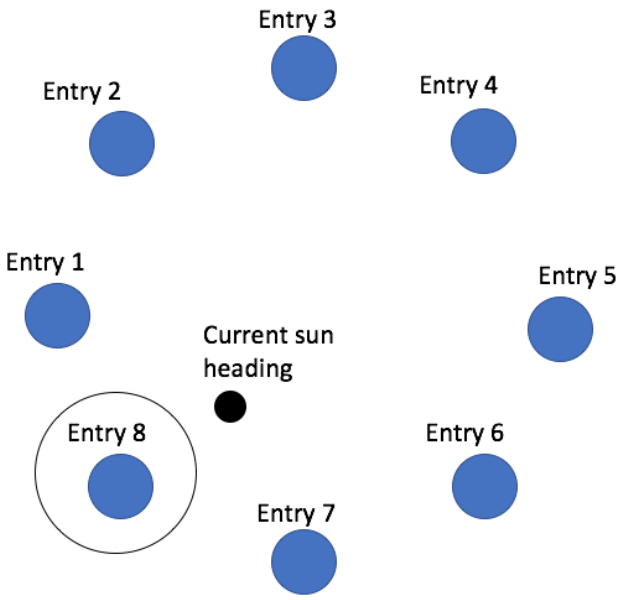

Fig. 1: Visual Description of Table Look-up Method

Then, the corresponding force and torque values are taken from the table and scaled according to

the magnitude of the spacecraft-sun position vector:

$$\mathbf{F}_{\text{radiation,scaled}} = \mathbf{F}_{\text{radiation}}\Big(\frac{AU}{|\mathbf{r}_{\text{sun}}|}\Big)^2 [N] \tag{5}$$

$$\boldsymbol{\tau}_{\text{radiation,scaled}} = \boldsymbol{\tau}_{\text{radiation}}\Big(\frac{AU}{|\mathbf{r}_{\text{sun}}|}\Big)^2 [Nm] \tag{6}$$

Most important to the user of the table look-up method is the required input and format of data. Data must be recorded in XML format. As an example, see ../cube_lookup.xml (in the radiation pressure folder). Additionally, a utility script called parseSRPLookup.py is provided there to read the XML input into numpy arrays. Experienced users are welcome to store their data in their own format and load it into equivalent numpy arrays as they see fit.

An example of using the provided python script to load data is shown in test_radiationPressure.py. Note that this also requires import of the unitTestSupport library.

1.1.3 Solar Eclipses

Solar eclipses are are detected by the basilisk eclipse module. The effects of the eclipse are calculated into a shadow factor, F_{s}, which is applied to the output forces and torques.

$$\mathbf{F}_{\text{out}} = F_{\text{s}}\mathbf{F}_{\text{full\_sun}} \tag{7}$$

$$\boldsymbol{\tau}_{\text{out}} = F_{\text{s}}\boldsymbol{\tau}_{\text{full\_sun}} \tag{8}$$

2 Model Functions

The mathematical description of gravity effects are implemented in gravityEffector.cpp. This code performs the following primary functions

- **Cannonball Method**: The code calculates the force on a spacecraft due to solar radiation pressure.
- **Look-up Method**: The code calculates both the force and torque on a spacecraft due to solar radiation pressure. It uses user-provided tabulated data to do so.
- **Solar Eclipse**: The code takes solar eclipses into account via a "shadow factor". This shadow factor is output from the Basilisk solar eclipse module and can include the effects of multiple planets. It is applied to the force/torque outputs.
- **Interface: Spacecraft States**: The code receives spacecraft state information via the DynParamManager.
- **Interface: Forces and Torques**: The code sends spacecraft force and torque contributions via computeForceTorque() which is called by the spacecraft. If using the cannonball method, the returned torque values are zero.
- **Interface: Sun Ephemeris**: The code receives Sun states (ephemeris information) via the Basilisk messaging system.
- **Interface: Solar Eclipse**: The code receives solar eclipse (shadow factor) information via the Basilisk messaging system.

3 Model Assumptions and Limitations

The two methods of calculation used in this code have their own sets of assumptions and limitations. There are some assumptions which are common to both methods.

- **Cannonball Model**: This default solar radiation pressure model assumes that the radiation pressure will act normal to some equivalent surface area, $A_{\odot}$. While this could be a good assumption, $A_{\odot}$ would have to be time-varying with spacecraft attitude and incorporate spacecraft self-shadowing. In general, the code does not do this. This limits the cannonball method to being most accurate in relatively mundane simulations (no rapid rotations or varying self-shadowing). Additionally, this method does not calculate torques on the spacecraft, so it is limited to cases where high precision is not needed with regards to spacecraft attitude.
- **Look-up Method**: This method utilizes tabulated data. Therefore, there will be error associated with whatever method was used to generate and record the data, but those errors are outside of Basilisk. Furthermore, the algorithm selects the data which *most closely* matches the current position of the spacecraft relative to the sun and does not interpolate between data points. This method also is limited to users who have external models or real data to use to describe their spacecraft.
- **Radiation**: The radiation model is hard-coded to assume that the radiation comes from the Sun. It is not possible to model radiation pressure from other sources with this code. This applies to both the cannonball and look-up methods. The model has no time-varying radiation effects (solar storms, etc.). A more in-depth radiation model would be need if high-accuracy radiation pressure effect calculations are needed.
- **Eclipse**: The shadow factor applies a simple scaling factor to the output forces and torques. This assumes that all portions of the spacecraft are affected equally by the eclipse. This should, in most circumstances, be highly accurate. For exceptionally large $A_{\odot}$ spacecraft which also need highly accurate state calculations, this assumption could fail.
- **Tabulated Data Import** Currently, Basilisk includes a utility script to import data from XML files for use in radiation pressure calculations. While some users could learn to load data in other formats, this is currently a limitation to most users who have data in other forms.

4 Test Description and Success Criteria

This test is located at `simulation/dynamics/RadiationPressure/_UnitTest/test_radiationPressure.py`
In order to get good coverage of all the aspects of the module, the test is broken up into three sub-tests.
In each sub-test, a spacecraft is placed in the solar system and acted upon by the Sun.

4.1 "Cannonball" Method

This test utilizes the "cannonball" method to calculate the effects of radiation pressure on spacecraft dynamics. The cannonball method approximates the spacecraft as a sphere. External forces in the inertial and body frame, as well as external torques in the body frame, are checked against known values.

4.2 Table Look-up Method

This test uses a stored table of known effects of radiation pressure. It looks up values and compares them to the expected result to validate radiation pressure table look-up capabilities.

4.3 Table Look-up with Eclipse

This test is the same as the Table Look-up Method except that the spacecraft experiences a partial eclipse during the test. The experiences forces and torques are expected to be exactly half of what they are without the eclipse.

4.4 Table Look-up Compared to Cannonball

This test compares the output of the table look-up method to the outputs of the cannonball method. This time, the look-up tables have been generated assuming a spherical spacecraft so that no torques are calculated. The tables also indicate that the spacecraft has a radius which works with the solar radiation pressure to generate a 1 N away from the sun. Furthermore, the cannonball method is given a very specific area input to generate the appropriate force. This test validates that the look-up method is generating reasonable results.

5 Test Parameters

This section summarizes the error tolerances for each test. Error tolerances are determined based on whether the test results comparison should be exact or approximate due to integration or other reasons. Error tolerances for each test are summarized in table 2.

Table 2: Error tolerance for each test.

Test	Tolerated Error
"Cannonball"	1.0e-12
Look-up	1.0e-12
Look-up With Eclipse	1.0e-12
Look-up (torque)	1.0e-14
Look-up With Eclipse (torque)	1.0e-14
"Cannonball" Look-up	1.0e-12
"Cannonball" Look-up (torque)	1.0e-14

Note that the lookup model tests utilize more stringent tolerances for torques. This is because the torque values are too small to use the same tolerances as the force calculations.

6 Test Results

All checks within test_radiationPressure.py passed as expected. Table 3 shows the test results.

Table 3: Test results.

Test	Pass/Fail	Notes
"Cannonball"	PASSED	
Look-up	PASSED	
Look-up (torque)	PASSED	
Look-up With Eclipse	PASSED	
Look-up With Eclipse (torque)	PASSED	
"Cannonball" Look-up	PASSED	
"Cannonball" Look-up (torque)	PASSED	

7 User Guide

This section contains information directed specifically to users. It contains clear descriptions of what inputs are needed and what effect they have. It should also help the user be able to use the model for the first time. Some possible examples are included below:

7.1 Variable Definition and Code Description

The SRP model type can be specified using the following python commands:

- `setUseCannonballModel()` — (default) uses the spherical spacecraft cannonball model
- `setUseFacetedCPUModel()` — uses a faceted spacecraft model where the SRP forces and torques are evaluated on the CPU

The variables in Table 4 are available for user input. Variables used by the module but not available to the user are not mentioned here. Variables with default settings do not necessarily need to be changed by the user, but may be.

Table 4: Definition and Explanation of Variables Used.

Variable	LaTeX Equivalent	Variable Type	Notes
stateInMsgName	N/A	string	Default setting: "inertial_state_output". This is the message from which the radiation pressure module receives spacecraft inertial data.
sunEphmInMsgName	N/A	string	Default setting: "sun_planet_data". This is the message through which radiation pressure gets information about the sun and planets.
area	$A_{\odot}$	double	[m2] Default setting: 0.0f. Required input for cannonball method to get any real output. This is the area to use when approximating the surface area of the spacecraft.
coefficientReflection	c_R	double	Default setting: 1.2. This is a factor applied to the radiation pressure based on spacecraft surface properties.
sunEclipseInMsgName	N/A	string	No default. Must be set. If an eclipse model is being used, this should reference eclipse output.
sunEclipseMsgData	N/A	string	No default. If creating a "fake" eclipse message, set to radiation_pressure.EclipseSimMsg()
sunEclipseMsgData.shadowFactor	F_{s}	double	Default setting: 1.0. i.e. there is no shadow by default.

It is important to note that in order for the sunEclipse information in the table above to be utilized, one must call: unitTestSupport.setMessage(unitTestSim.TotalSim, testProcessName, sunEclipseInMsgName, sunEclipseMsgData). In other cases, the eclipse message will be set by the eclipse module and the radiation pressure module will only need to be told where to look for it.

7.2 Code Diagram

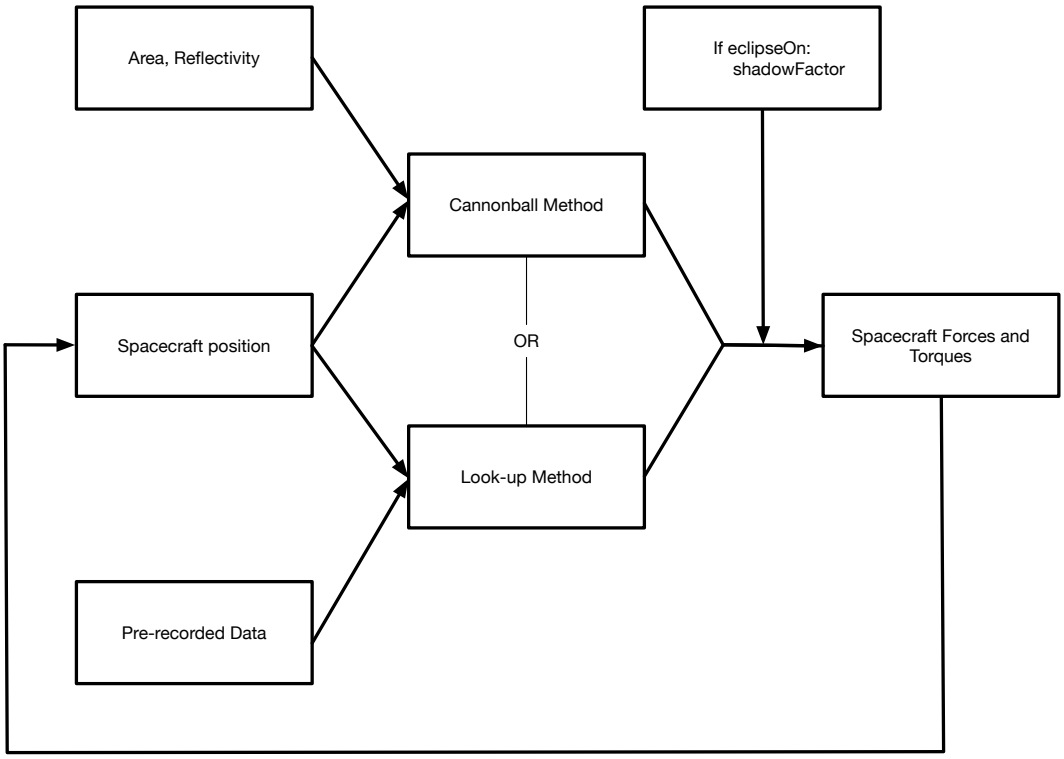

Fig. 2: A pseudo-code diagram showing the flow of inputs and outputs in the radiation pressure module.

The diagram in Fig. 2 demonstrates the basic iterative logic of the gravity effector module. There is extensive additional code that deals with things from the messaging system to transforming the spacecraft position from one frame to another.

After the inputs are given, radiation_pressure.cpp calculates the effects of radiation pressure via one of the two methods. At the end, the eclipse factor scales the output forces and torques.

REFERENCES

[1] David Vallado. *Fundamentals of Astrodynamics and Applications*. Microcosm Press, 2 edition, 2001.