Master Thesis

Simulation and control toolkit for small satellite projects

(Programovy balik pro simulaci a navrh rizeni malych satelitu)

Adam Śmiałek

February 2020

Contents

Abbreviations

ADCS Attitude Determination and Control System
AOCS Attitude and Orbit Control Systems
PDR Preliminary Design Review
ESA European Space Agency
NASA National Aeronautics and Space Administration
SCARS Spacecraft Control Architecture Rapid Simulator
SCARS Spacecraft Control Architecture Rapid Simulator
ESTEC European Space Research and Technology Centre
TEC-ECN Guidance, Navigation, and Control Systems Section
LEO Low Earth orbit
MEMS micro-electromechanical systems
A/VRW Angle/Velocity Random Walk
KSP Kerbal Space Program
kRPC Remote Procedure Call Server for KSP
VRML Virtual Reality Markup Language
WWW World Wide Web
STK Systems Tool Kit
REXUS/BEXUS Rocket and Balloon Experiments for University Students programme
GPS Global Positioning System
GNSS global navigation satellite systems
ECEF Earth-Centered, Earth-Fixed
ECI Earth-Centered Inertial
ECR Earth-Centered Rotational
DLR German Aerospace Center
iMQT ISIS Magnetorquer Board

1 Introduction

The idea to create a simulation and control toolkit for small satellite projects was a byproduct of work done on IRISC project by the author of this thesis. IRISC, or "InfraRed Imaging of astronomical targets with a Stabilized Camera", was a project realized as a part of Rocket and Balloon Experiments for University Students programme (REXUS/BEXUS) programme. The goal of the IRISC experiment was to obtain images in the near infrared (NIR) spectrum from astronomical targets. Possible targets included the Andromeda Galaxy, Pinwheel Galaxy, Iris Nebula, Eagle Nebula and Starfish Cluster. The images were obtained using a highly stabilized telescope with NIR camera mounted on a BEXUS balloon. Author's responsibility covered the design of the control subsystem of the experiment. The stabilization was achieved by a gimbal-like system, to obtain high quality images while being on a moving platform.[?] While the design of the control system was not innovative, nevertheless it was a complicated task for a student with no previous practical experience in that field. This has proven to be especially challenging during early stages of experiment design.

The process of effective space-related project management, from the conception of the idea, through production, to disposal, features high costs and often various unpredictable risks. Due to this, a project life cycle is usually divided into distinct phases, allowing for introduction of conducting product reviews within rigid time-frames. An example of such a workflow, adopted by most major agencies such as ESA[?] and NASA[?], is a division of the project life cycle into phases, as it can be seen on Figure 1.1. While the design of a project is often an iterative process, the phases and reviews that conclude them exist as a checkpoints, after which the design of the project is to be unchanged, on a level of details progressing as phases do. For example, as one can see, Phase B is usually ended by the Preliminary Design Review (PDR). In the case of a spacecraft, for the PDR, a major architecture parameters have to be defined, such as volume and weight ramifications, top-level designs of solutions for major requirements have to be presented - for a practical example: for high-resolution Earth observation mission the type of the actuators which fulfills precision requirements has to be chosen.

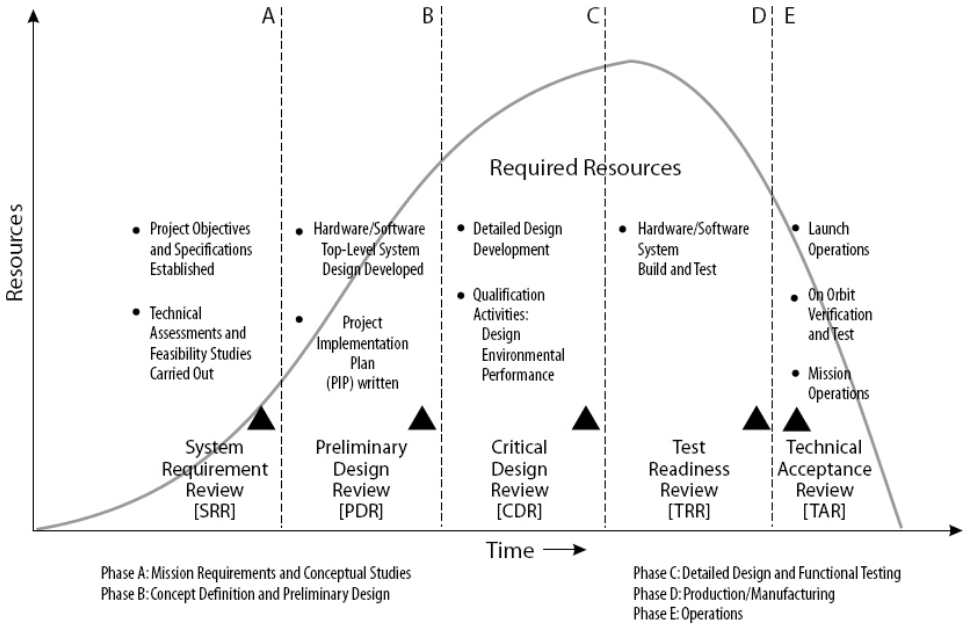

Figure 1.1: Typical space project phases and its life cycle[?]

Taking the experience of how challenging and time-consuming was the process of learning how to produce a reliable simulation of IRISC control system and the knowledge of significance of preliminary design, author decided to produce and publish a simulation, which would be useful for purposes of initial spacecraft design and for beginner engineers to learn how to build a reliable simulation.

1.1 Scope

The thesis covers the process of development of the toolbox for rapid prototyping of satellite's control systems. Chapter 1 describes the aim of this work and discusses the topic of prototyping tools. Chapter 2 goes into detail about the architecture of Spacecraft Control Architecture Rapid Simulator (SCARS), its features and methods of implementation. Also here there are described the ways of connecting SCARS with various visualization tools. Chapter 3 explains the documentation and usage of SCARS, while in Chapter 4 there are examples showing how the toolbox can be used in real life applications. Finally, Chapter 5 discusses the conclusions from the development process and the possibilities for improvements of SCARS.

1.2 Aim

The aim of this thesis work is to build and provide a ready to use open source product - a toolbox for small and low budget satellite projects. The toolbox features allow for a initial design of spacecraft's Attitude Determination and Control System (ADCS), which means that they allow for, i.a. simulation of spacecraft orbit, testing the feasibility of various actuation methods and testing the effectiveness of different control algorithms in given use cases. That software would then allow smaller and inexperienced teams of spacecraft designers to better prepare for design milestones like PDR, when there is not enough time to create a full simulation of their spacecraft ADCS subsystems. Besides the toolbox being a tool for practical use, the thesis also serves as as a review of available solutions, so it can be used by future control engineers as a learning material. The idea is, that some parts of the proposed model could be removed from the model as the students, for the learning purposes, would be tasked with designing a substitution.

refine the list of examples this

For the purposes of later evaluation of how the solutions proposed in this thesis fulfil the goals stated in the preceding paragraph, a following list of objectives was compiled:

- **Conduct a review of existing tools for preliminary spacecraft design**, focusing on mission planning and Attitude and Orbit Control Systems (AOCS) subsystem;
- **Create a spacecraft dynamics and AOCS model**, to be used with minimal set-up;
- **Assemble a library of models**, to be used by other beginner control engineers;
- **Provide a documentation of the toolbox**, explaining not only the purpose and operating principles of individual parts, but the process of using the toolbox to conduct a preliminary design of spacecraft AOCS subsystem;
- **Share the toolbox to be available online**, with principles of open-source software in mind.

Furthermore, the objectives that are set for the design of the toolbox itself are described in Chapter 2.1.

1.3 Digital prototyping tools

1.4 Already existing tools

The idea for a toolbox allowing for spacecraft mission design and AOCS simulation is not a new one. There are various solutions available, ranging from very robust commercial software packages to open-source implementations of individual features for use as a part of MATLAB framework.

The aim of this section is to prove that, while the solutions for spacecraft prototyping are available, there is still a need for open-source, easy-to-sue and modify toolbox. Further below is a compiled list of selected software solutions that fit the most objectives stated in subsection 1.2, with included explanation what they are lacking that discussed toolbox should have.

1.4.1 MATLAB CubeSat Simulation Library

CubeSat Simulation Library is a part of Aerospace Blocks created by MathWorks Aerospace Products Team. Using it one can model motion and dynamics of CubeSats and nano satellites. It provides the most basic features, like the simulation of pre-set attitude scenarios, basic actuators and sensors models and integration with MATLAB's Virtual World visualization tools.

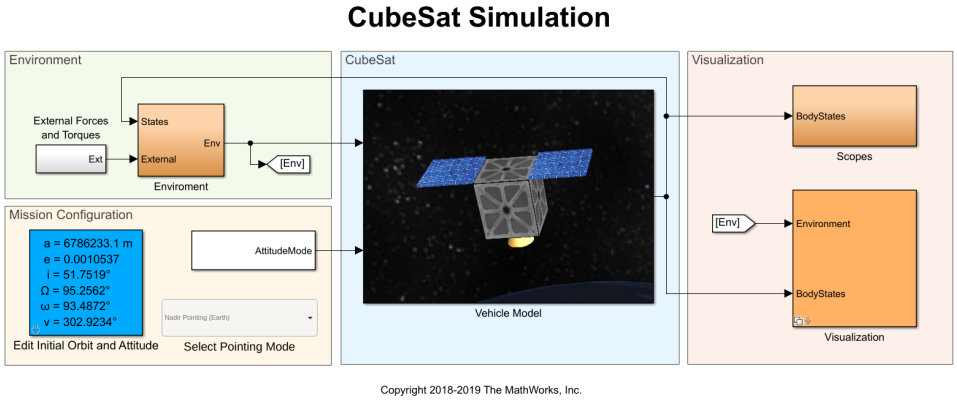

Figure 1.2: Top-level view of the example project of the MATLAB CubeSat Simulation Library

This library, while conceptually most similar to the SCARS, it lacks some functionalities. For example, for actuators, it provides only general

models for perfect and second-order actuators. In SCARS, the actuators are full models, which allows not only for reducing the number of layers of abstraction between the user and the simulation, but also for things like calculation of energy expended by the actuator. Also, this toolbox is sparsely documented - while most functionalities are described within their Simulink block masks, there is no comprehensive guide about how to use them in own models.

does this explanation belong here?

1.4.2 PrincetonSATELLITE Spacecraft Control Toolbox

PrincetonSATELLITE Spacecraft Control Toolbox is a commercial solution for building spacecraft Simulations. I contains over two thousand functions for attitude and orbit dynamics, simulation, estimation, analysis and design. This is the most robust and comprehensive toolbox available, includes online API, well written documentation and additional modules for unique applications like formation flying, fusion propulsion or solar sails. This would be the best choice for most use-cases, yet it is a paid solution and even the cheapest option - CubeSat Edition - may be out of price range for smaller teams.

1.4.3 PROPAT Toolbox

PROPAT is is a small set of functions in Matlab to simulate and propagate orbit and attitude of an Earth's satellite, developed by the single person as an open-source toolbox. Several functions allow to transform between orbit and attitude coordinates and for propagation or rigid body attitude. PROPAT contains only MATLAB scripts, which while useful and can be used as a part of the simulation, do not combine into a model of a whole spacecraft's ADCS subsystem.

1.4.4 GAST Toolbox

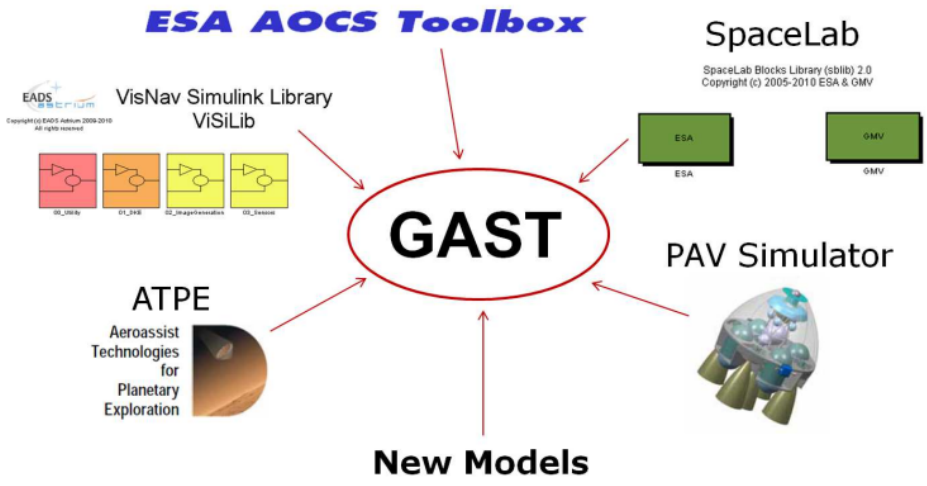

Figure 1.3: Representation of the consolidation of TEC-ECN toolboxes

The GAST toolbox is the result of the consolidation of several toolboxes available in Guidance, Navigation, and Control Systems Section (TEC-ECN) of European Space Research and Technology Centre (ESTEC), such as the old AOCS Toolbox, the SpaceLAB library, the ViSiLib library, the ATPE simulator, and the PAV simulator. In addition to consolidating these toolboxes, new models were developed for the GAST toolbox according to the needs of the section. Figure 1.3 shows a pictorial representation of the consolidation of the toolboxes of TEC-ECN. This software was developed in TEC-ECN in 2008, but since it is a product of European Space Agency (ESA), it is not available for use for wider audience.

1.4.5 User-created modules available on MathWorks MATLAB Central

As MATLAB is the most popular...

MATLAB Central is a network for asking questions about MATLAB software, discussing solutions and sharing MATLAB and Simulink solutions and files. On a subsection called File Exchange there are many files available to use within MATLAB framework, some of them relevant to spacecraft

design. The most notable examples are listed below.

SAT-LAB
SAT-LAB is a MATLAB-based Graphical User Interface (GUI), developed for simulating and visualizing satellite orbits. The primary purpose of SAT-LAB is to provide a software with a user-friendly interface that can be used for both academic and scientific purposes. While a useful tool, it is only suitable for initial mission planning.[?]

Satellite Orbit Modeling
A collection of MATLAB scripts used for modelling of satellite's perturbed motion with special perturbations approach. While it is very robust, as it can be applied to any problem in celestial mechanics, this module is useful for orbit modeling, not AOCS system design.[?]

Smart Nanosatellite Attitude Propagator (SNAP)
The Smart Nanosatellite Attitude Propagator is an attitude propagator for satellites that can be used to analyze the environmental torques affecting a satellite and to design and analyze passive attitude stabilization techniques, such as Passive Magnetic Stabilization, Gravity Gradient Stabilization and Aerodynamic stabilization. This model is the most relevant one for the scope of this thesis, but it lacks possibility to model active attitude stabilization techniques.

Satellite Orbits: Models, Methods and Applications
Rather than spacecraft or orbit model, it is a collection of exercises for book *Satellite Orbits: Models, Methods and Applications*. It is interesting from the educational point and refers at least partially to the problem of control system design - mainly GPS sensor. Yet this is not a toolbox by any means.

Apollo 11 Moon Landing - 50th Anniversary Model
This example shows how Richard Gran and the other engineers who worked on the Apollo Lunar Module digital autopilot design team could have done it using Simulink, Stateflow, Aerospace Blockset and Simulink 3D Animation if they had been available in 1961. Although it is a very notable example of how MATLAB software family can be used to simulate a whole mission, to use it for either own spacecraft or educational purposes would require much more reverse engineering and modifications than creating a new model.

2 Spacecraft Control Architecture Rapid Simulator (SCARS) Toolbox

Short description of the chapter.
What SCARS means and what it is.
Describe briefly what is the input and the output of the whole toolbox.

2.1 Objectives

- Provide model of orbital dynamics of Earth orbiting satellite;
- Create models of most common satellite actuators and sensors and parametrize them so that actual hardware can be reproduced in simulation using parameters from datasheets;
- Include sources of environmental forces and torques, modeling most influential sources;
- Implement several of the most basic control methods;
- Create a Simulink Custom Library, with all models masked for quick set up;
- Provide methods of conducting preliminary review of feasibility of used hardware components and control methods;
- Create interfaces allowing the user to connect the toolbox with visualization software;

2.2 Choice of software

To fit with the objectives of accessibility and ease of modification MATLAB family of software was chosen. MATLAB is one of the most popular scripting language and with the addition of Simulink software it can become powerful tool with the ability to set up numerical simulations in short time. MATLAB is taught in most technical universities and there is significant number of both courses available online and materials for self-teaching. For one purpose (described in Section 2.11.3) a Python script acting as a dataflow bridge was used, as it was a simplest method to solve a problem described in that chapter. Several other software solutions were used for visualization purposes, with the reasoning described in subsection 2.11.

Versatility of MATLAB is often attributed to the number of Add-Ons available for it. Spacecraft Control Architecture Rapid Simulator (SCARS) Toolbox uses and requires the following modules:

- Aerospace Toolbox
- ...

- ...

2.3 Architecture

SCARS is divided into two parts: 1) Parts Library and 2) Modular Simulation. The Parts Library contains Simulink subsystems, which can be connected to form simulations of various complexity and for multiple scenarios. The other is a Modular Simulation, which can be set up with either MATLAB command line scripts or graphical user interface.

2.3.1 Parts Library

SCARS Parts Library is a ready to use Simulink Custom Library, that is a collection of blocks available to use in Simulink models. All blocks in library downloaded alongside SCARS are parametrized, masked and described to ease the integration of library parts into user simulation. The library is divided into specific sections:

- Satellite Dynamics
- Reference Frames Transformations
- Environment
- Actuators
- Sensors
- Control Algorithms
- Visualization
- Analysis
- Examples

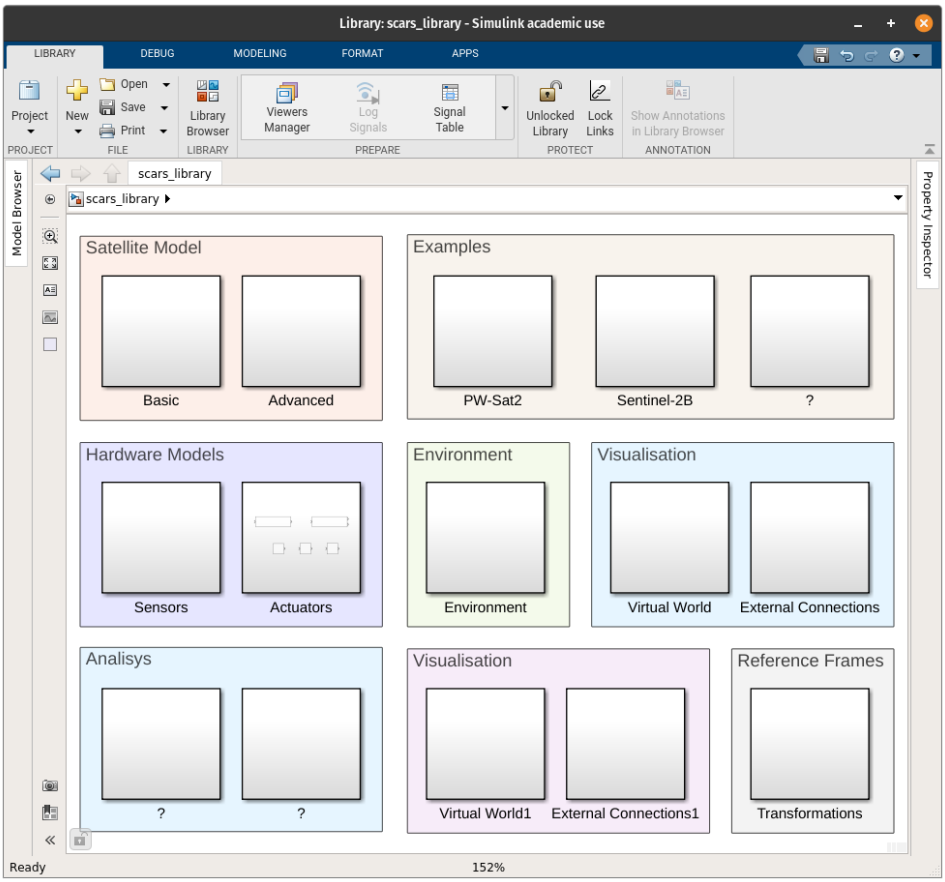

Figure 2.1: SCARS Parts Library screenshot

2.3.2 Modular Simulation

SCARS Modular Simulation is a ready-made simulink model available for setup using prepared scripts and SCARS user interface. The model is a simulation of cube-shaped satellite, which can be set on specified orbit using various initialization methods, such as Keplerian elements in conjunction with Julian date time. (The initialization is further described in Chapter 3). In the same manner, all actuators and sensors available in SCARS library can be chosen. The Modular Simulation makes use of most available blocks, which can be commented out from the model, either by hand or using the user interface, to improve the speed of the simulation.

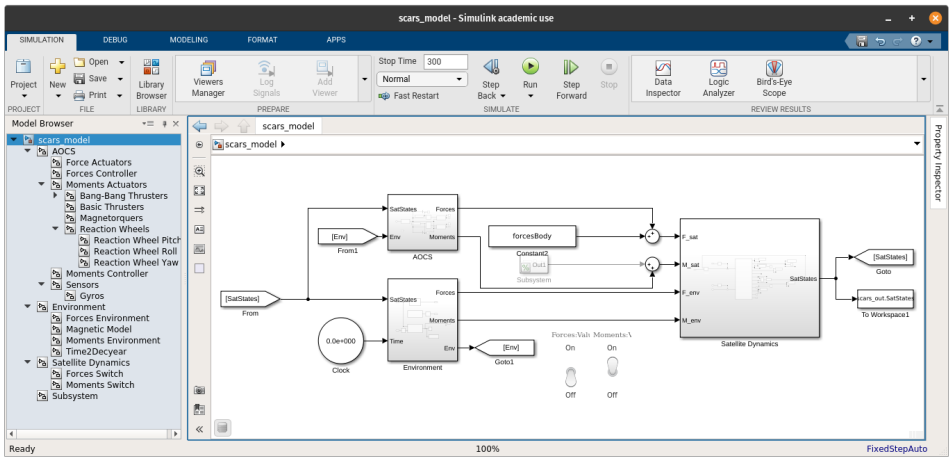

Figure 2.2: SCARS Modular Simulation screenshot

2.4 Orbit dynamics

2.5 Reference Frames

2.5.1 Earth-Centered Inertial (ECI)

The Earth-Centered Inertial (ECI) frame has its origin located in Earth's center of gravity. It has X axis parallel to and directed as vernal equinox direction, its Z axis is constructed from the vector starting in origin and going through Earth's celestial North pole and Y axis is a vector cross product of the other two. Specific ECI frame used in this thesis is J2000 frame, defined with with the Earth's Mean Equator and Equinox at 12:00 Terrestrial Time on 1 January 2000[?]. Position and velocity vectors in this frame will be further noted as $\mathbf{r}$ and $\mathbf{v}$ respectively.

2.5.2 Earth-Centered, Earth-Fixed (ECEF)

The Earth-Centered, Earth-Fixed (ECEF), also known as Earth-Centered Rotational (ECR), is a frame of reference can with origin in Earth's center of mass and it's axes are parallel with international reference pole (Z_{ECEF} axis) and international reference meridian (X_{ECEF} axis), with Y_{ECEF} axis being vector cross product of other two. The ECEF frame includes information about rotation of the Earth, hence a point which would be fixed in the ECI frame would be progressing with time in the ECEF frame. Vectors

in this frame will be further noted as $\mathbf{r_{ECEF}}$ and $\mathbf{v_{ECEF}}$.

2.6 Coordinates Transformations

Used: ECEF, NED
Transformation from Keplerian Elemenets to Kartesian Coordinates

2.6.1 ECI position and velocity vector to Keplerian elements

As mentioned, Earth orbiting satellite's position can be described by it's position vector $\mathbf{r}$ in Cartesian coordinate system, with center corresponding to Earth's geometric center. That vector in connection with spacecraft's velocity vector $\mathbf{v}$ in same coordinate system can be transformed into Keplerian elements by using following equations:

$$a = \frac{\mu}{2\left(\frac{\mu}{r} - \frac{v^2}{2}\right)} \tag{1}$$

$$i = \cos^{-1}(\frac{h_Z}{|\mathbf{h}|} \tag{2}$$

$$e = \sqrt{}(\frac{1 - h^2}{\mu a}) \tag{3}$$

$$\psi = \cos^{-1}\left(\frac{a - |\mathbf{r}|}{ae}\right),\ where\ \sin(\psi) = \frac{\mathbf{r} \cdot \mathbf{v}}{e\sqrt{\mu a}} \tag{4}$$

$$\theta = \sin^{-1}\left[\frac{\sin(\psi\sqrt{1-e^2})}{1 - e\cos(\psi)}\right] \tag{5}$$

$$M = \psi - e\sin(\psi) \tag{6}$$

And Ω and ω can be found from following relations:

$$\sin(\Omega) = \frac{h_X}{\sqrt{h_X^2 + h_Y^2}}\ and\ \cos(\Omega) = \frac{h_Y}{\sqrt{h_X^2 + h_Y^2}} \tag{7}$$

$$\sin(\omega + \theta) = \frac{r_Z}{r\sin(i)}\ and\ \cos(\omega + \theta) = \frac{r_Z\cos(\Omega) + r_Y\sin(\Omega)}{r} \tag{8}$$

Where terms h_X, h_Y and h_Z are components of $\mathbf{h} = \mathbf{r} \times \mathbf{v}$ vector and r_X, r_Y and r_Z are components of position vector.

2.6.2 Keplerian elements to ECI position and velocity vector

To quickly obtain position vector from Keplerian elements one may define a coordinate system with x, y axes on orbit's plane with $z = 0$. Then the following equations describe the coordinates:

$$x = a\cos(\psi) - ae \tag{9}$$

$$y = a\sin(\psi)\sqrt{1 - e^2} \tag{10}$$

Then the position in Earth's inertial Cartesian coordinate system can be found with following system of equations:

$$\mathbf{r} = \begin{bmatrix} r_X \\ r_Y \\ r_Z \end{bmatrix} = [A_Z(\Omega)]^{-1}[A_X(i)]^{-1}[A_Z(\omega)]^{-1} \begin{bmatrix} x \\ y \\ 0 \end{bmatrix} \tag{11}$$

Where $[A_d(\alpha)]$ stands for transformation matrix about axis d by an α angle.

2.6.3 ECI to ECEF

To transform vectors calculated in inertial frame to Earth-Fixed reference frame one has to multiply the ECI vector by following rotation matrix:

$$DCM_{ECEF}^{ECI} = \begin{bmatrix} \cos\theta_{GMST} & \sin\theta_{GMST} & 0 \\ -\sin\theta_{GMST} & \cos\theta_{GMST} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{12}$$

Where θ_{GMST} is Earth's rotation angle; to be calculated with:

$$\frac{1}{240} \cdot mod\left[24110.54841 + 8640185.812866 \cdot Y + 0.093104 \cdot Y^2 - 6.2 * 10^{-6} \cdot Y^3\right] \tag{13}$$

Where

2.7 Environment

Environment block is responsible for outputting few parameters, which are then used by the Satellite subsystem

- Gravity model
- Partial atmosphere?
- Sun's relative position and Earth's shadow
- Magnetic model

2.7.1 Earths's Gravity Model

Main centripetal force acting on a spacecraft on any orbit is gravity - it is defined by the equation derived from the law formalized by Isaac Newton:

$$\ddot{\mathbf{r}} = -\frac{G(m_1 + m_2)\mathbf{r}}{\|\mathbf{r}\|^2} \tag{14}$$

Where r is the position vector, m_1 and m_2 are the masses of two-body system and G is the universal gravitational constant. Simplified with:

$$m_1 = M_{Earth} \gg m_2 = m_{spacecraft} \tag{15}$$

One can derive the corresponding potential function:

$$u = -\frac{GM_{Earth}}{\mathbf{r}} \tag{16}$$

For a spacecraft on Earth's orbit, this model is a very far-stretched approximation, as it leaves out the influence of Earth's non-ideal shape, changes in density gradient in Earth's interior and perturbations caused by gravitational fields of other bodies. While the influence of other celestial objects is omitted in the SCARS toolbox due to it being mostly designed for lower orbits, one can easily account for Earth's non-spherical mass distribution using function constructed with the use of Lagendre polynomials to calculate the correction ϵ to potential function (16):

$$\epsilon(r,\theta,\varphi) = \sum_{n=2}^{\infty} \frac{J_n P_n^0(\sin\theta)}{r^{n+1}} + \sum_{n=2}^{\infty}\sum_{m=1}^{n} \frac{P_n^m(\sin\theta)(C_n^m \cos m\varphi + S_n^m \sin m\varphi)}{r^{n+1}} \tag{17}$$

Where the correction is a function of spacecraft's position in spherical coordinate system - r, θ, φ are in order altitude, latitude and longitude. The coefficients J_n, C_n^m and S_n^m are computed to possibly provide best approximation between observed and calculated orbit. Lagendre polynomials of form

$$\frac{P_n^0(\sin\theta)}{r^{n+1}} \tag{18}$$

are called the zonal terms and Lagendre functions

$$\begin{gathered} \frac{P_n^m(\sin\theta)\cos m\varphi}{r^{n+1}} \\ \frac{P_n^m(\sin\theta)\sin m\varphi}{r^{n+1}} \end{gathered} \tag{19}$$

correspond to tesseral terms. The denominating term is the so-called "J_2 term":

$$\frac{J_2 \ P_2^0(\sin\theta)}{r^3} = J_2 \frac{1}{r^3}\frac{1}{2}(3\sin^2\theta - 1) = J_2 \frac{1}{r^5}\frac{1}{2}(3r^2 sin^2\theta - r^2) \tag{20}$$

While equations (16) and (17) can added together to faithfully model the influence of Earth's gravity field on the spacecraft, it was decided to use a model from Simulink Aerospace Blockset - the Spherical Harmonic Gravity Model, with EGM2008 planetary model, as it is much more detailed and provides better accuracy.

2.7.2 Partial Atmosphere

Earth's atmosphere is composed of complex layers that are bounded basing on their composition and parameters. Man-made objects on Earth's orbit would be located in thermosphere, if their orbit is at least partially under $600km$ altitude above the surface of the Earth, or exosphere if above it. The former consists mostly of molecular hydrogen and nitrogen, while the latter also of hydrogen, helium ans carbon diaoxide. The main effects of the higher layers of atmosphere on the spacecrafts in Low Earth orbit (LEO) are drag, degradation of surface materials and spacecraft glow. For the toolbox, the only relevant effect is the first one, resulting in both aerodynamic force and aerodynamic torque acting on the spacecraft.

Aerodynamic forces are created by spacecraft's movement through the atmosphere. The forces acting on the spacecraft are drag, lift and side slip force, but the only one taken into consideration will be the drag, acting on spacecraft's tangential velocity, since the other are of negligible magnitude. To calculate drag force, one has to use the following equation:

$$F_d = -\frac{1}{2}\rho C_d A v^2 \tag{21}$$

Where C_d is the drag coefficient, ρ is atmospheric mass density, A is body area in a cross-section perpendicular to velocity vector and v is the total velocity of the satellite with respect to the atmosphere.

Describe aerodynamic toruqe

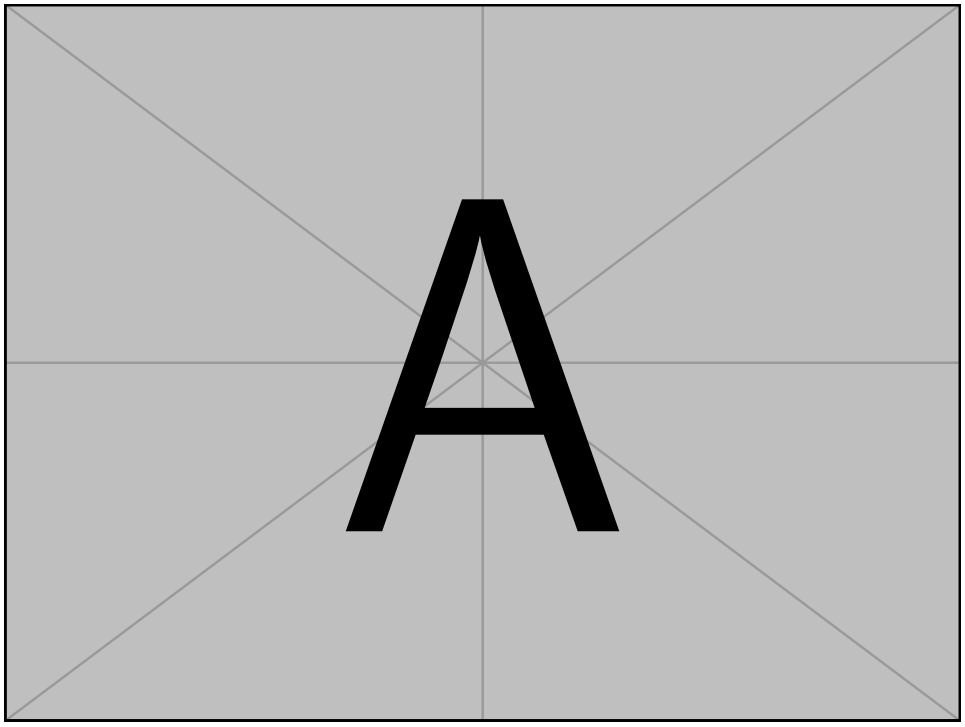

Figure 2.3: Model of Earth's atmosphere layers

The reference atmospheric model used in SCARS is NRLMSISE-00, which takes date and position of the object in geographic coordinate system as inputs and outputs temperature and density of the atmosphere components. As it was built for satellites, it allows for altitudes up to $1000km$. In the toolbox, orbits above that are considered to have negligible impact of the atmosphere and therefore atmospheric forces are set to zero above this threshold.

2.7.3 Sun and Earth Relative Position

...

2.7.4 Earth's Magnetic Model

...

2.8 Actuators

2.8.1 Thrusters

Parameters:

- Thrust range;
- Nominal thrust: (find a way to model change?)
- Specific impulse (also ranges?)
- Max propellant
- Total impulse
- Power (at nominal thrust?)
- Mass
- Dimensions?
- Hot standby Power
- Time delay to control
- In both directions?

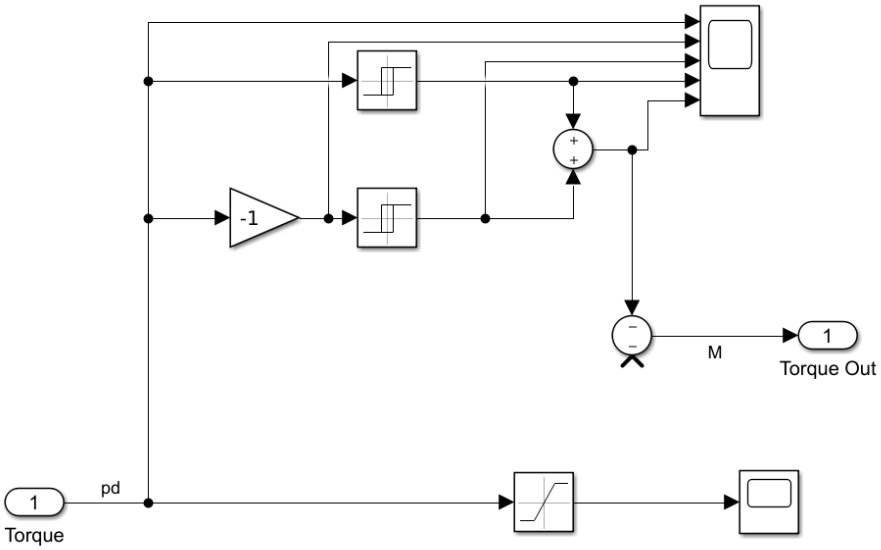

Figure 2.4: Simulink Thrusters model

2.8.2 Reaction Wheels

Ideal to real: Bearing Noise, Transport Delay, Saturation, Quantization.

Fast attitude control can be also achieved by the use of reaction wheels - mechanisms consisting of rotating flywheel and proportional electromagnetic torquer, such as DC motor. This allows for very precise attitude maneuvers, with the possibility to eliminate most disturbance torques. Reaction wheels operate around at a non-zero reference speed and change in their angular velocity imposes corresponding torque on the spacecraft. The disadvantage of this solution is that reaction wheels have fixed operating range and to achieve higher angular velocities for the spacecraft, the wheels have to be desaturated using another actuators. In cubesats, for example, most commonly this would be solved by the addition of magnetorquers.

In fast attitude control the motion about each spacecraft body axis can be considered to be decoupled from motion about two other axes. The equations of motion that describe the influence of reaction wheels angular velocity $\dot{q_w}$ on total angular momentum H are as follows:

$$I_y\dot{q} = Ni + Q_f + Qdy \tag{22}$$
$$\dot{\Theta} = q \tag{23}$$
$$J\dot{q_w} = -Ni - Q_f \tag{24}$$
$$Ri = e - N(q - q_w) \tag{25}$$
$$Q_f = -c(q - q_w) \tag{26}$$
$$H = I_y q + Jq_w \tag{27}$$

Where e, i, R are respectively steering voltage, current in DC motor and armature resistance. N is torque per unit current and c is viscous friction coefficient. Said equations were modelled in the toolbox with a following diagram:

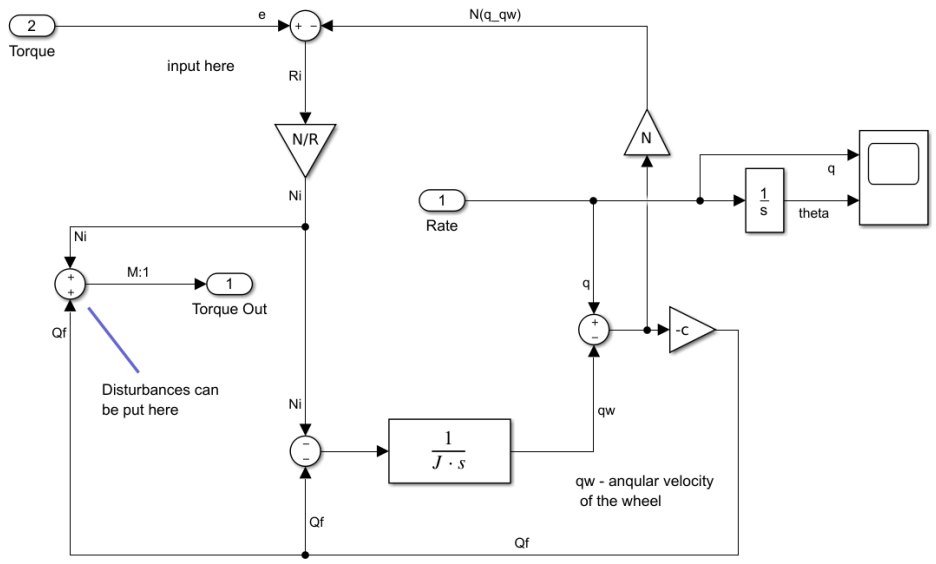

Figure 2.5: Simulink Reaction Wheels model

The problem with modeling off-the-shelf reaction wheels is that datasheets rarely provide the value of viscous friction coefficient c in the DC motor. Therefore to model this kind of actuator, SCARS requires the user to only input following data, with c being optional:

2.8.3 Magnetorquers

A magnetorquer is an attitude actuator which uses Earth's geomagnetic field to generate controlling torque. The active part in the magnetorquer is the solenoid, which generates the magnetic dipole moment proportional to the current conducted by the coil. This interaction is described with the following equataion:

$$\tau_B = M \times B \tag{28}$$

Where τ_B is mechanical torque acting on the spacecraft, M the generated magnetic moment inside of it and B is the magnetic field density. In matrix

form it is:

$$\begin{bmatrix} \tau_{Bx} \\ \tau_{By} \\ \tau_{Bz} \end{bmatrix} = \begin{bmatrix} 0 & B_z & -B_y \\ -B_z & 0 & B_x \\ B_y & -B_x & 0 \end{bmatrix} \begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} \tag{29}$$

The drawback of using magnetorquers for attitude control is that they are unfit for fast maneuvers. Moreover, since Earth's magnetic field density is inversely proportional to cube of distance from Earth's center, then without high grade sensors or on-board models, they don't allow for precise maneuvering on higher orbits.

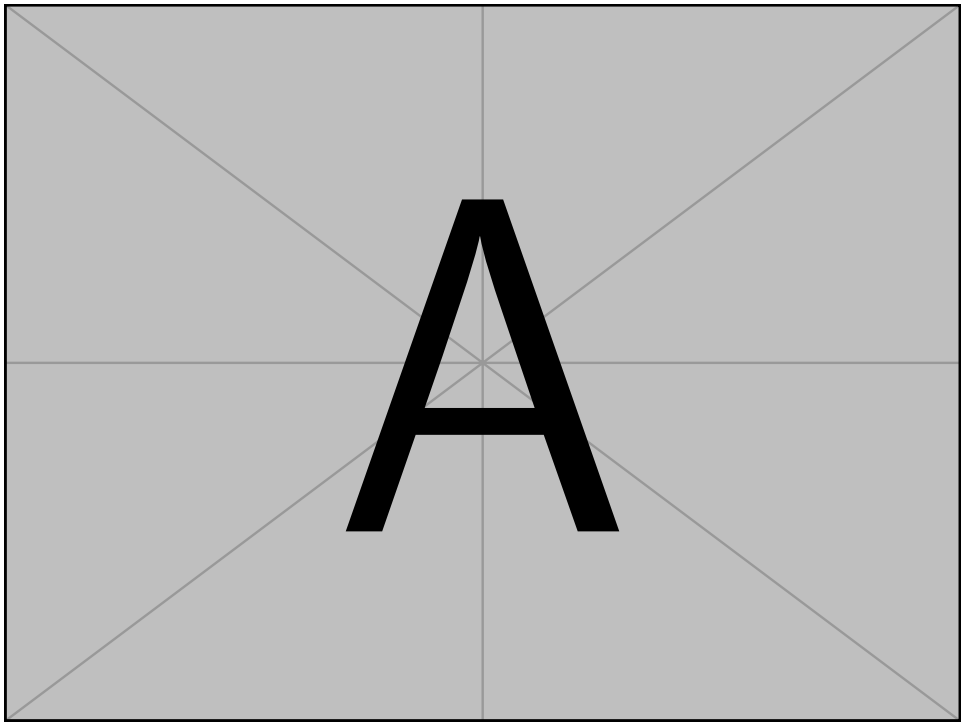

Figure 2.6: Magnetorquers model

2.8.4 Drag Sail

Drag sails use the occurrence of partial atmosphere (described in 2.7.2) to lower satellite's tangential velocity and therefore to quicken the deorbitation of the spacecraft. The premise is to increase area-to-mass-ratio by deploying a large and lightweight structure near the planned end-of-life of

the spacecraft. Due to this operating principle, drag sails are only relevant for low and medium mass spacecrafts and are applicable only on LEO. To calculate the perturbing acceleration following equation is used:

$$F = -\frac{1}{2}\rho C_d A v^2 sin\alpha \tag{30}$$

Where *alpha* is the angle between the sail's plane and satellite's velocity vector. For now the moment of sail's deployment is not simulated in SCARS toolbox.

2.9 Sensors

2.9.1 GPS Receivers

The Global Positioning System (GPS) is a global navigation satellite systems (GNSS) owned adn operated by United States government. It allows for determining position, velocity and time using data taken from at least four GPS satellites.

Previously using GPS receivers in LEO was burdened with technical challenges, as off-the-shelf components were mostly designed for terrestial operations, not encompassing for example for large variations in the received signal Doppler frequencies. Recently smaller GPS receivers became available, even for CubeSat use, such as *Venus838FLPx GPS Receiver*[?], allowing for real time orbit determination using GPS navigation in smaller satellite projects.[?] When choosing a GPS receiver one must take several parameters into consideration:

- Update rate
- Horizontal position accuracy
- Vertical position accuracy
- Velocity accuracy
- Decay factor?
- Failure rate?

All listed parameters are set up in SCARS *GPS Receiver* part, but rather than designing a model from scratch a MATLAB's Navigation Toolbox function, `gpsSensor` was nested inside a masked Simulink block.

2.9.2 Accelerometers

2.9.3 Magnetometers

2.9.4 Gyroscopes

Gyroscopes, which fall under category of inertial sensors, measure angular rate around fixed axis. In smaller spacecraft, which is in great deal of SCARS toolbox use-cases, the conventional spinning mass gyroscopes are rarely used, due to limitations in mass and size. Recent developments allow for use of much smaller and cheaper micro-electromechanical systems (MEMS) gyroscopes, which are vibrating angular rate sensors. They were chosen to model for the toolbox, as of popularity in projects with highly restricted budget.[?] Inside of vibrating gyroscope the Coriolis effect is a cause the vibrating core to produce a force acting on its support. The measurement of the force is used to determine the rate of rotation of the body around gyroscope axis. MEMS gyros are similar to integrated circuits, which use the miniaturized version of mechanisms based on principles of operation of either vibrating wheels, tuning forks, resonant solids or similar common designs.[?] While, besides previously mentioned qualities, the upsides of MEMS gyroscopes are availability of both analog and digital outputs, low power consumption and commercial availability. On the other hand, MEMS gyros have shorter lifetime and lower performance when compare to pricier alternatives.

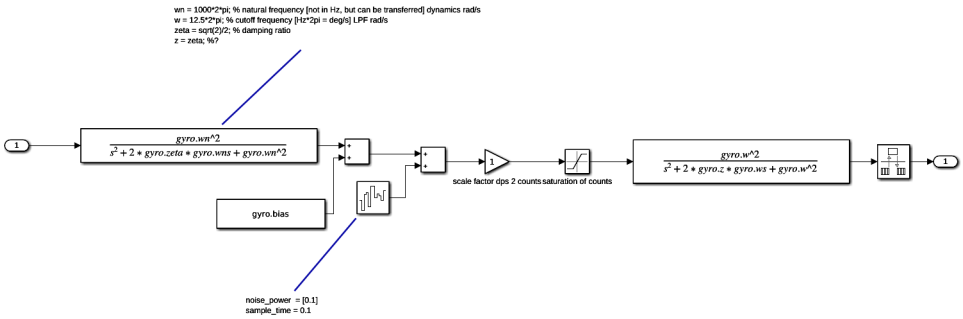

Figure 2.7: Gyroscope model

In the toolbox the following sources of gyroscope errors are modeled:

- Bias offset:
- Scale factor:
- Angle/Velocity Random Walk (A/VRW): A high frequency noise term

that has a correlation time much shorter than the sample time. Can be defined as white noise component on the sensor output. Specified either in $\frac{deg}{\sqrt{h}}$ or, as a power spectral density, in $\frac{(deg/h)^2}{Hz}$.

- Quantization Error: An error which is caused by the digital quantization of output signal, obtained when sampling analog input.
- ...

- Cite this:[?]

2.10 Control Methods

2.10.1 PID

2.10.2 LQR

2.10.3 Quaternion Feedback Control

2.10.4 Analisys Tools

2.11 Visualization Tools

2.11.1 MATLAB Virtual Reallity Toolbox

Virtual Reality Toolbox is an extension for MATLAB which allows creating and interacting with 3D virtual reality models of dynamic systems. In its core it uses Virtual Reality Markup Language (VRML), a language created in the early days of World Wide Web (WWW) to display 3D objects and animations. This toolbox provides a way for implementing the VRML models inside MATLAB script or Simulink simulation, and allows for control of driving display or animation with MATLAB variables and Simulink signals. Moreover, the toolbox is integrated with VRML viewer and VRML editor, allowing for building and displaying models directly from MATLAB environment.

Virtual Reality Toolbox was used in SCARS as most core method of visualization. The VRML model is set up with 3 objects: Satellite, Earth and Sun, as they can be considered most useful when observing the effects of the simulation. Satellite model also includes objects representing antennas' range or optical instrument's field of view, if set up in simulation. This feature can be useful for analysis of imaging capabilities.

The transformations required to process the data generated by SCARS'

Vehicle Dynamics block into VRML parameters are as follows:

$$r_{ecef}^{VRML} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} r_{ecef}^{Simulink} \tag{31}$$

Finish describing transformations

2.11.2 Systems Tool Kit

Systems Tool Kit (STK), formerly named Satellite Tool Kit, is a platform for analyzing and visualizing variety of ground, sea and space platforms missions. STK is a commercial software solution used by most major organizations and companies such as National Aeronautics and Space Administration (NASA), ESA, German Aerospace Center (DLR), Boeing, ICEYE. Features most relevant to the topic of this thesis are the graphical engine allowing for displaying the position and attitude of the satellite, and the set of analytical tools, such as ground station connection time calculator, allowing for fine-tuning of mission details.

To visualize SCARS simulation results with STK, one must generate timestamped ephemeris and attitude files. SCARS can generate such files for the user, with predetermined format according to STK documentation [?]. Both files contain the preamble specifying parameters such as scenario epoch time, central body, coordinate system, distance unit and format of the file. As SCARS relies mostly on ECEF reference frame, it is also chosen for ephemeris and attitude files. After the preamble, the file contains the lines for each data point. In case of ephemeris file (`.e` file) the have a format of:

```
<TimeInSeconds> <X> <Y> <Z> <xDot> <yDot> <zDot> <xDotDot> <yDotDot> <zDo
```

Where the unit of time is seconds and relative to defined scenario epoch and following parameters are ECEF vectors in m, m/s and m/s^2 respectively. For the attitude file (`.a` file), the format is:

```
<TimeInSeconds> <q1> <q2> <q3> <q4>
```

Where time is formatted in same manner as in ephemeris file and the following parameters build a quaternion vector, with fourth element being scalar component.

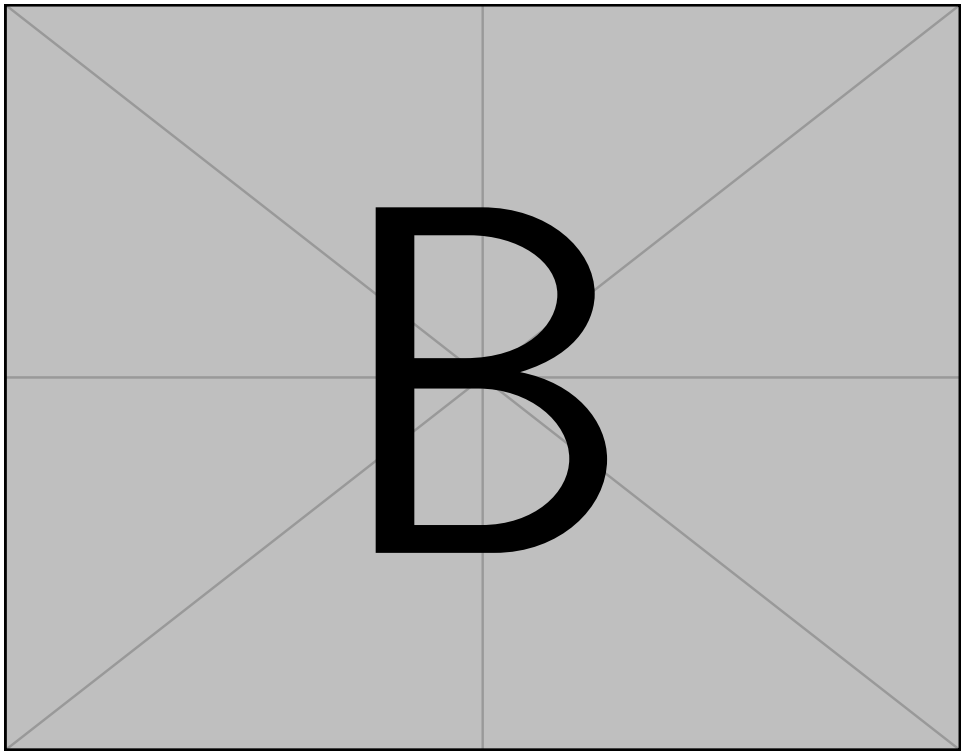

Figure 2.8: Example of STK satellite visualization

2.11.3 Kerbal Space Program

Finally, Kerbal Space Program (KSP), a space flight simulation video game, can be used as a nonconventional method to visualize the results of SCARS Toolbox simulation. In KSP the player directs a developing space program originated on fictional Earth-like planet Kerbin. The game provides the tools for the players to design and fly rockets, probes, satellites, spaceplanes, rovers, and other spacecraft from a library of components.[?] The aim of this visualization method was to build a sample satellite in KSP, simulate it in SCARS and execute a Hohmann Transfer within a game, using simulation outputs as game inputs.

The connection between MATLAB and KSP is possible because of fan-made Remote Procedure Call Server for KSP (kRPC) mod. It creates a API server running alongside the game, with which calls can be made using already written clients in most popular languages, like C++, Python, Lua,

Java, etc. Integrating it with MATLAB has proved to be a difficult task, as MATLAB doesn't provide simple means for threading, which means that inputs for the game have to be precalculated to work in real time. Moreover, there is no kRPC library written directly for MATLAB, therefore a simple Python bridge was written to parse the data taken from the game, compare them with pre-generated SCARS simulation scenario outputs and send them to KSP as in-game AOCS subsystem inputs.

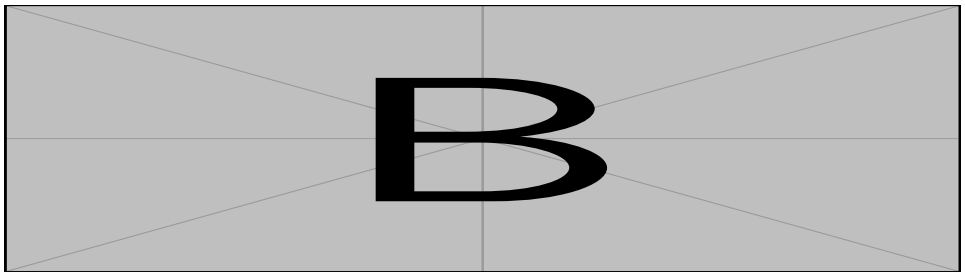

Figure 2.9: Example of kRPC Python client code

3 Documentation of SCARS

3.1 MATLAB Masks and help

3.2 Website?

4 Examples of usage

4.1 Setting up spacecraft ADCS architecture in SCARS

4.1.1 Basic made up spacecraft

4.1.2 PW-Sat2

Magnetorquers
Two modes of control: Detumbling Control Mode; Sun Pointing Mode

Detumbling
Detumbling maneuver is performed after deployment of the spacecraft from a carrier rocket. As the satellites are separated from the deployment mechanism, they are burdened by non-zero initial angular rates. To counteract that and stabilize a satellite PW-Sat2 is equipped with a set of two perpendicular magnetorquer rods and one air core, in total one coil acting along each of satellite's body axis. The most important parameters used by SCARS model of PW-Sat2 are listed in Table 4.1.2.[?] Exact values are taken directly from the datasheet of ISIS Magnetorquer Board (iMQT).[?]

Parameter	**Value**
Nominal magnetic dipole	$0.2Am^2$
Maximum actuation envelope error	$3\mu T$
Power consumption during actuation	$1.2W$
Mass	196g

Deorbitation with drag sail
One of the main objectives of PW-Sat2 mission was to deploy and test the effectiveness of it's drag sail in deorbitation maneuver. The sail was 2x2m square made from aluminized polyester boPET film.[?]

In this example, SCARS toolbox was tested against data points derived from NORAD measurements. The simulation was run in two cases: only with drag sail set up and with drag sail and magnetorquers on to keep sail's plane perpendicular to spacecraft's orbit tangent vector.

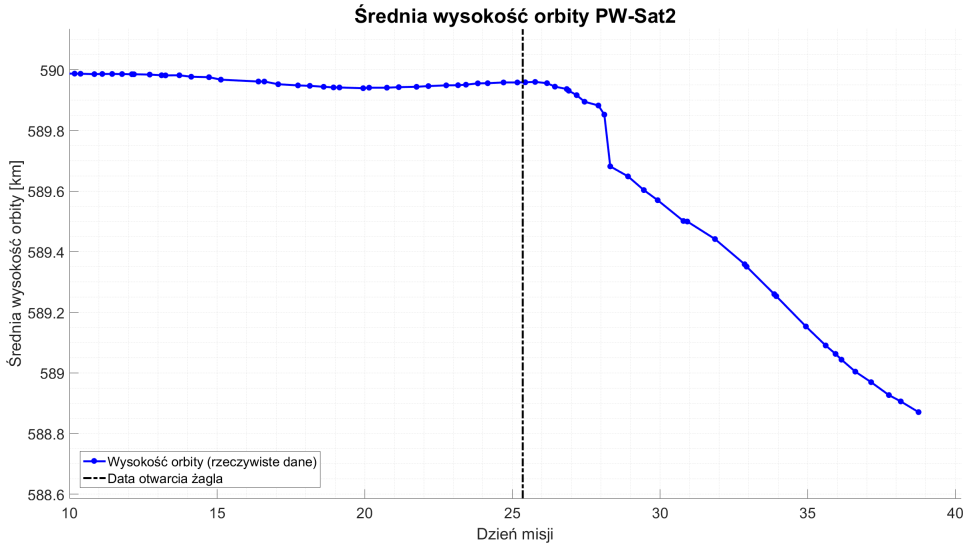

Figure 4.1: Average altitude of PW-Sat2 satellite as taken from NORAD measurements. On X axis there is mission time in days, on Y axis there is average altitude in km.

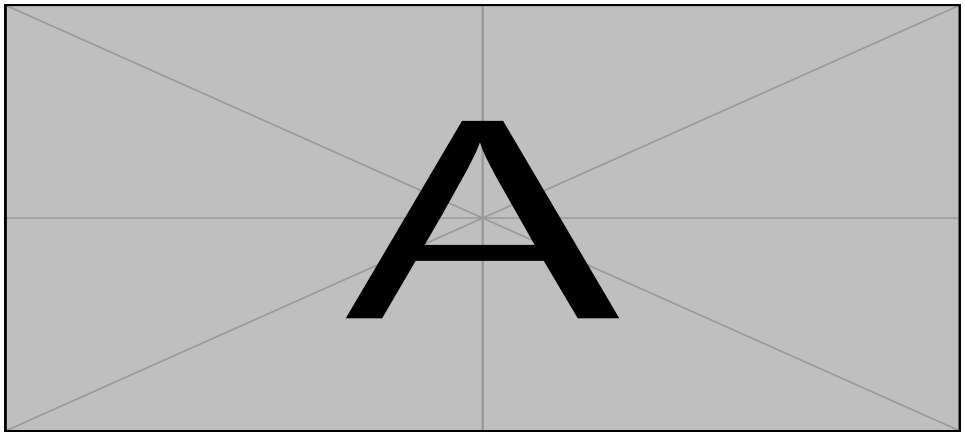

Figure 4.2: Results from SCARS simulation, with only drag sail included

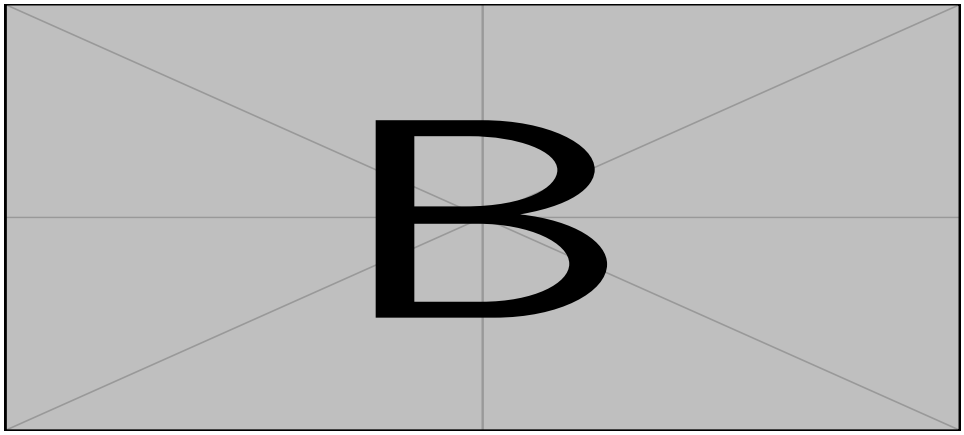

Figure 4.3: Results from SCARS simulation, with magnetorquer-driven attitude correction

4.1.3 Sentinel-2

Sentinel-2 is a European polar satellite mission carried out by ESA as a part of Copernicus Programme. It consists of constellation of twin polar orbit satellites, Sentinel-2A and Sentinel-2B and it's aim is to deliver Earth observation data to broad public, providing wide range of services such as natural emergency management, agricultural monitoring or water classification.[?]

As per document describing Sentinel-2 ADCS subsystem, the satellites operate on a sun-synchronous oribt, with $786km$ mean altitude and $10:30$ local time of descending node. They maintain Earth-oriented attitude in all operational modes. The required pointing performance is moderate, but the main design driver is the need for precise geo-location of the images.[?]

The actuators on board of Sentinel-2 and simulated with SCARS toolbox for demonstration purposes are decribed in Table 4.1.3.

No.	Unit	Type	Supplier	Name
3	MAG	3-axis fluxgate magnetometer	ZARM Technik	FGM-A-75
2	GPRS	2 band GPS receiver	RUAG	-
3	STR	Active pixel sensor star tracker	Jena Optronik	Astro APS
4	IMU	High performance fibre optical gyro	Astrium	ASTRIX 200
3	MTQ	$140Am^2$ magnetic torquer	ZARM Technik	MT140-2
4	RW	$18Nms$ reaction wheel	Honeywell	HR12
8	THR	$1N$ monopropellant thruster	EADS ST	CHTIN-6

4.2 Examples of tests possible with SCARS

4.2.1 Control system robusntess analisys

4.2.2 Long term simulation

4.2.3 Contingency scenarios

5 Conclusions

References