Pontifícia Universidade Católica do Rio Grande do Sul

Faculdade de Engenharia – Faculdade de Informática

Modelagem MATLAB para cálculo de atitude e rastreio de satélites artificiais

Trabalho de Conclusão de Curso

Autores
Felipe Groisman Sieben
Plauto de Abreu Neto

Orientador
Prof. Dr. Eduardo Augusto Bezerra

Porto Alegre, dezembro de 2009

Índice

Índice de Figuras

Índice de Tabelas

Lista de Siglas e Abreviações

ACDH	Attitude Control and Data Handling
ADCS	Attitude Determination and Control Subsystem
AEB	Agência Espacial Brasileira
AEL	Aeroeletrônica
AP	Argument of Perigee
BCH	Bose, Chaudhuri and Hocquenghem
BF	Body Frame
CCSDS	Consultative Committee for Space Data Systems
ECEF	Earth Centered Earth-Fixed
ECI	Earth Centered Inertial
ESA	European Space Agency
FACIN	Faculdade de Informática
FENG	Faculdade de Engenharia
FPGA	Field Programmable Gate Array
GNNS	Global navigation satellite system
GSE	Grupo de Sistemas Embarcados
GSO	Geostationary Orbit
HST	Hubble Space Telescope
IAA	Instantaneous Access Area
INPE	Instituto Nacional de Pesquisas Espaciais
INS	Inertial Navigation System
LEO	Low-Earth Orbit
NASA	National Aeronautics and Space Administration
NORAD	North American Aerospace Defense Command
OBDH	On-Board Data Handling
OF	Orbital Frame
PROMELA	Process Meta Language
PUCRS	Pontifícia Universidade Católica do Rio Grande do Sul
RAAN	Right Ascension of Ascending Node
SISCAO	Sistema de Controle de Atitude e Órbita
SPI	Serial Peripheral Interface
SPIN	Simple Promela Interpreter
SSP	Sub Satellite Point
TC	Telecomandos
TLE	Two-Line Elements
TM	Telemetria
UTMC	Unidade de Telemetria e Telecomando

VHDL Very-High-Speed Integrated Circuits Hardware Description Language

1. Introdução

Os satélites artificiais possuem um ou mais subsistemas utilizados na interação com instrumentos de bordo ou com outros subsistemas. Tipicamente, um satélite possui, no mínimo, um subsistema principal que é chamado de *payload* (carga útil do satélite). Um exemplo de *payload* é o espelho primário do Telescópio Espacial Hubble (*Hubble Space Telescope* – HST), um de muitos instrumentos que são utilizados para observar objetos astronômicos [1]. Em outro exemplo, no satélite Intelsat [2], o *payload* é o seu equipamento de comunicação, enquanto que em um satélite militar de defesa o *payload* pode ser um sensor infravermelho. Em todos os casos, para que o satélite possa executar sua tarefa principal, este deve conseguir utilizar componentes do seu *payload* de forma direcionada ao seu alvo, com a precisão requisitada por seu sistema. Essa precisão é normalmente especificada em alguma unidade angular, por exemplo, 1 grau, 1 segundo de arco, 1 miliradiano. O subsistema de controle e determinação de atitude (*Attitude Determination and Control Subsystem* - ADCS) deve ser projetado para cumprir os requisitos de precisão exigidos pelos componentes do *payload* [3].

O custo de colocar um quilograma de massa em uma órbita terrestre baixa (*Low-Earth Orbit* – LEO) é de mais de US$9.000,00 [4], e para colocá-lo em uma órbita geoestacionária (*Geostationary Orbit* – GSO) o custo é ainda maior. Geralmente, um satélite possui uma massa de mais de 500 kg e custa de dezenas a centenas de milhões de dólares para ser projetado, fabricado, testado e preparado para o lançamento. Todo esse recurso é necessário para que o satélite possa cumprir sua missão e, por este mesmo motivo, são utilizados os sistemas de controle de atitude, de propulsão, o veículo de lançamento e assim por diante. Se fosse possível cumprir uma missão sem um ADCS, então a massa associada a esse sistema poderia ser utilizada, por exemplo, para aumentar o tamanho do *payload*, ou poderia-se, então, reduzir o custo do lançamento. Porém, sendo o *payload* e sua missão a razão da existência do satélite, é importante ressaltar a necessidade do ADCS para o cumprimento da missão, ou seja, a justificativa para sua existência em um satélite.

Satélites artificiais, em sua grande maioria, podem ser classificados em duas categorias: telecomunicações e sensoriamento remoto. Os satélites de telecomunicações possuem rádio transceptores, multiplexadores e antenas que lhes provêm a capacidade de comunicação. Historicamente, a maioria dos satélites de comunicação estão em órbita no anel geoestacionário, porém, mais recentemente, alguns têm sido colocados em órbitas LEO. Nesses casos, uma das missões do ADCS é manter o satélite apontado para sua estação terrestre com a maior precisão possível. Quanto mais preciso for o ADCS, mais fácil será a comunicação com a estação terrestre, e menor será a potência requisitada pelo sistema de comunicação.

Com relação à outra categoria, sensoriamento remoto, existem basicamente dois tipos de satélites: de observação da Terra; e de observação do espaço. Os satélites de observação da Terra podem ser do tipo *nadir-pointing* (os que observam o ponto na Terra imediatamente abaixo deles) ou então podem servir para rastrear objetos por toda a superfície terrestre. Para o primeiro caso, um estabilizador passivo por gradiente de gravidade pode ser suficiente, enquanto que para o segundo caso um ADCS ativo seria necessário, utilizando alguma combinação de rodas de momento ou propulsores para modificar o direcionamento do satélite.

Um satélite de observação do espaço, em geral, possui requisitos diferentes como, por exemplo, a impossibilidade de apontar para a Terra ou para o Sol. Este último é o caso dos instrumentos astronômicos do HST que são sensíveis e, se forem apontados em direção ao Sol, possivelmente serão danificados. Satélites dessa espécie necessitam, muitas vezes, da capacidade de realizar grandes movimentos angulares bem como movimentos extremamente precisos.

Outra função importante do ADCS é a de manter os painéis solares do satélite apontados para o Sol. Por exemplo, quando o HST está apontando para um determinado alvo, o satélite ainda possui um eixo de liberdade para rotacionar. Essa rotação pode ser utilizada para orientar o eixo dos painéis solares de forma perpendicular em relação à direção do Sol. Então, os painéis solares podem ser separadamente rotacionados neste eixo para que apontem diretamente para o Sol.

O Trabalho de Conclusão de Curso aqui descrito implementa parte de um sistema de controle de atitude para um satélite de observação da Terra. O algoritmo desenvolvido é capaz de distinguir quando o satélite está em uma posição que permita visualizar o alvo e calcular a atitude (orientação) que o satélite deverá ter em relação aos eixos orbitais para conseguir alcançá-la.

Em decorrência do objetivo principal, um segundo objetivo se fez necessário: o desenvolvimento de um conjunto de ferramentas auxiliares para que a execução do algoritmo possa ser visualizada e validada.

O presente Trabalho de Conclusão de Curso está dividido em sete capítulos, sendo o primeiro composto por esta introdução. O Capítulo 2 contextualiza o Projeto PUC#SAT e seus trabalhos relacionados; o Capítulo 3 apresenta as ferramentas que serão utilizadas; o Capítulo 4 explica alguns conceitos importantes para o entendimento do trabalho; o Capítulo 5 apresenta as ferramentas desenvolvidas; o Capítulo 6 relata os testes e resultados obtidos e, por fim, no Capítulo 7 encontram-se a conclusão e os trabalhos futuros.

2. Projeto PUC#SAT

O PUC#SAT consiste em uma iniciativa de pesquisa e desenvolvimento na área de computação de bordo para aplicações espaciais. Os trabalhos do PUC#SAT são desenvolvidos no Grupo de Sistemas Embarcados da Faculdade de Informática da PUCRS, com financiamento da Agência Espacial Brasileira e do CNPq. No âmbito dessa iniciativa já foram desenvolvidas diversas pesquisas a nível de graduação, mestrado e doutorado. Também foram realizados trabalhos de desenvolvimento de produtos para a área espacial, em parceria com as empresas Aeroeletrônica e Innalogics de Porto Alegre.

O primeiro produto gerado a partir do PUC#SAT consistiu na implementação em hardware da pilha de protocolos do CCSDS [5][6][7] para utilização no Subsistema de Comunicação [8] de um veículo espacial. O desenvolvimento do módulo de comunicação, para uso em satélites do INPE, teve inicio no GSE em 2005 no âmbito do programa Uniespaço da AEB. De forma a auxiliar na conclusão dos trabalhos em andamento no GSE, o INPE contratou o consórcio formado pelas empresas Aeroeletrônica (AEL) e Innalogics. O objetivo principal da atuação dessas empresas no projeto foi fornecer o auxílio necessário ao GSE de forma a transformar os resultados das pesquisas realizadas em um produto que atenda os rigorosos padrões de qualidade do INPE. Outro objetivo foi o apoio financeiro necessário para o pagamento da equipe envolvida, uma vez que o financiamento da AEB não possuía esse intuito. Dessa forma, a Innalogics e a AEL trabalharam juntos ao GSE na produção da UTMC (Unidade de Telecomando e Telemetria) [9] que pertence ao Subsistema de Comunicação do Computador de Bordo do “Sistema de Controle de Atitude e Órbita de um Satélite Estabilizado em Três Eixos (SISCAO)” do INPE.

A seguir são descritos alguns trabalhos desenvolvidos no contexto do PUC#SAT, sendo que os resultados do presente trabalho são de interesse direto da pesquisa em andamento descrita na seção 2.4.

2.1 Códigos Corretores de Erros em Hardware para Sistemas de Telecomando e Telemetria em Aplicações Espaciais

A Dissertação de mestrado de autoria do Gabriel Marchesan Almeida em 2007 [10] expõe uma pesquisa acadêmica no escopo de códigos corretores de erros empregados em sistemas espaciais. O principal objetivo do trabalho contempla o projeto, implementação e validação de circuitos corretores de erros para dados de telemetria e telecomando, seguindo o padrão CCSDS (Consultative Committee for Space Data Systems). Ambos os módulos de telemetria e telecomando são descritos em linguagem VHDL e implementam, respectivamente, os algoritmos de correção de erros Reed-Solomon [11] e BCH (Bose, Chaudhuri and Hocquenghem) [12][13], os quais possuem alta capacidade de correção de erros ocorridos durante o processo de transferência de dados entre o veículo espacial e a base terrestre.

2.2 Sistema de Telecomando CCSDS em FPGAs

Este trabalho de conclusão de Curso foi desenvolvido por Carlos Eduardo Cardoso Reif, Luciano Rigelo Azevedo e Tiago Xavier Bai [14] do curso de Engenharia de computação em 2007.

Em sistemas espaciais, uma estação terrestre envia Telecomandos (TC) a serem consumidos por módulos computacionais presentes nos veículos espaciais, por sua vez, veículos espaciais enviam Telemetrias (TM) para as estações terrestres com as informações requisitadas. O trabalho desenvolvido apresenta o projeto, implementação e validação de funcionalidades básicas do fluxo de Telecomando de veículos espaciais. Essas funcionalidades foram implementadas seguindo recomendações do Consultative Committee for Space Data Systems (CCSDS), adotado pelas maiores agências espaciais mundiais, entre elas a agência americana NASA (National Aeronautics and Space Administration), européia ESA (European Space Agency) e Agência Espacial Russa.

2.3 Verificação da Implementação do Protocolo CCSDS do PUC#SAT usando PROMELA/SPIN

Desenvolvido em julho de 2008 por Alexandre Gomes Madeira como trabalho de conclusão de curso de Engenharia de Computação [15].

Neste trabalho foram utilizadas algumas técnicas de verificação formal na análise da implementação do fluxo de Telecomando. Esse Fluxo de telecomando foi desenvolvido dentro do contexto do projeto PUC#SAT, conforme colocado na seção 2.2.

Foi realizada a modelagem do sistema em linguagem PROMELA [16][17], a partir do código VHDL e das máquinas de estado. Posteriormente, foi utilizada a ferramenta SPIN [16][17][18] para simulação e verificação do modelo. Pode ser encontrada no relatório a descrição do processo de modelagem, da utilização da ferramenta de simulação e análise dos resultados obtidos.

2.4 Concepção do *On-Board Data Handling* com Sensor Inercial para Aplicações Espaciais

Trabalho de conclusão de curso de Engenharia de Computação, desenvolvido por Cristiano Garré Ferreira, Felipe Augusto da Silva e Paulo Ricardo Cechelero Villa no primeiro semestre de 2009 [19].

O trabalho de conclusão de curso desenvolvido abrangeu as áreas: espacial, microeletrônica, sistemas de tempo real e arquitetura de computadores. Possuindo como objetivo central o projeto e a implementação de um modelo básico para gerência e manipulação de dados a bordo de um satélite (On-Board Data Handling - OBDH).

Além de uma implementação básica do OBDH, o trabalho contemplou a integração deste com os demais subsistemas que compõem a Plataforma de Serviços de um satélite.

Um satélite é dividido em três grandes partes [20], uma delas é a Plataforma de Serviço, responsável por realizar a manipulação dos dados provenientes dos sensores e o controle de atitude e órbita do satélite. As outras partes são o payload, que possui equipamentos específicos da missão e o sistema de propulsão, responsável pela navegação.

A integração foi desenvolvida para exemplificar o fluxo completo das mensagens transmitidas entre a Estação Terrestre e o veículo espacial. As mensagens transmitidas da Estação Terrestre são recebidas, verificadas (integridade dos dados) e desempacotadas pelo subsistema de comunicação. Esse repassa as informações para o OBDH, que realiza a interpretação e manipulação das mensagens. Quando o conteúdo da mensagem representa um comando de requisição de dados, o OBDH executa as rotinas de busca nos sensores para capturar as medições aferidas naquele instante. Ao capturar e manipular os dados, o OBDH envia para o Subsistema de Comunicação que se encarrega de transmitir a mensagem do satélite para a Estação Terrestre.

3. Ferramentas Utilizadas

Este capítulo tem como objetivo apresentar as principais ferramentas utilizadas para o desenvolvimento do trabalho, iniciando com a ferramenta principal, MATLAB, e seguindo para outros dois softwares, utilizados principalmente como referência para que fosse possível validar o funcionamento das ferramentas de visualização desenvolvidas.

3.1 MATLAB

3.1.1 Histórico

Criada na Universidade do Novo México em 1970 no Departamento de Ciências da Computação, cujo diretor, Cleve Moler, era também desenvolvedor da linguagem. Utilizada largamente pela comunidade universitária, foi amplamente trabalhada pelos estudiosos de matemática aplicada. Em 1983, o engenheiro Jack Little teve um primeiro contato com o MATLAB numa visita de Moler à Universidade de Standford. Nesse encontro vislumbrou-se uma ferramenta com fins lucrativos, então Little, Moler e Steve Bangert fundaram em 1984 a MathWorks com MATLAB reescrita na linguagem C. LAPACK foi o nome dado às bibliotecas reescritas.

O engenheiro Little, especialista em projetos de controle, tendenciou primeiramente a utilização de MATLAB para a sua área, mas a linguagem acabou rapidamente sendo aplicada em outros campos. Hoje em dia é utilizada em diversas áreas, desde o ensino básico de álgebra linear e análise numérica até a ciência de processamento de sinal e imagem [21].

3.1.2 Visão Geral

O MATLAB é um sistema utilizado para desenvolvimento de algoritmos, visualização de dados, análise de dados e cálculos numéricos. Com sua interatividade e alto desempenho é possível fazer cálculos com matrizes (MATLAB = *MATrix LABoratory*), construções de gráficos, análise numérica, processamento de sinais, entre outras funcionalidades. O ambiente traz recursos que facilitam a sua utilização, podendo assim visualizar os problemas e soluções expressos matematicamente, diferentemente da programação tradicional [22].

Características:

- Linguagem de alto nível para computação técnica;
- Ambiente de Desenvolvimento para o gerenciamento de código, arquivos e dados;
- Ferramentas interativas para exploração iterativa, projeto e solução de problemas;
- Funções matemáticas de álgebra linear, estatística, análise de Fourier, filtragem, otimização e integração numérica;
- Visualização de dados através de gráficos de funções 2-D e 3-D;
- Ferramentas para a construção personalizadas de interfaces gráficas;
- Funções de integração de algoritmos baseados em MATLAB com aplicações externas e linguagens, como C, C + +, Fortran, Java, COM e Microsoft Excel.

3.1.3 Utilização no Projeto

O MATLAB é a ferramenta base de desenvolvimento para esse trabalho de conclusão. Foi escolhido por ser um software robusto que permite a execução de cálculos complexos de maneira rápida, ao mesmo tempo que lida com formas geométricas e gráficos com facilidade.

3.1.4 Versões utilizadas

- MATLAB 7.8 R2009a;
- MATLAB 7.9 R2009b.

3.2 Celestia

3.2.1 Histórico

O Celestia é um programa de astronomia 3D, de código aberto, para Windows, Mac OS X e Linux, criado por Chris Laurel. É baseado no Catálogo Hipparcos, e utiliza o OpenGL. Tanto a NASA quanto a Agência Espacial Européia (European Space Agency, ESA) já usaram o Celestia em seus programas educacionais e também como interface para software de análise de trajetória [23].

3.2.2 Visão Geral

O Celestia é um programa que proporciona ao usuário uma visão completa do Universo conhecido, tornando assim a observação astronômica mais acessível, mostrando objetos em diferentes escalas e em três dimensões. Permite navegar através de um extenso universo modelado segundo a realidade, em qualquer velocidade, direção ou tempo na história. É um simulador gratuito muito eficaz para aqueles que não têm oportunidade de utilizar um telescópio, com gráficos e uma câmera de fácil controle. Com o Celestia é possível explorar o Sistema Solar, galáxias, nebulosas e aglomerados. O programa mostra objetos que variam desde a escala de satélites artificiais enviados ao espaço até galáxias inteiras, em três dimensões usando OpenGL, com perspectivas que não seriam possíveis em um planetário tradicional. Pode-se observar as constelações, ver seus nomes e delimitar seus contornos. O programa possibilita a obtenção de maiores informações direcionando o usuário para uma

página da web com dados sobre o objeto selecionado. Também realiza cálculos de distâncias e permite a simulação de vôos em tempo real com diferentes velocidades.

As fotos mostradas no aplicativo são de alta qualidade, permitindo uma ampliação sem muitas distorções, apesar da coleção de figuras de quase cento e vinte mil estrelas não ser tão grande se comparada ao Stellarium ou ao Google Earth. O usuário pode variar o número de estrelas visíveis na tela e observá-las em diferentes estilos de desenho. Pode também viajar pelo universo por qualquer velocidade desde 0.001 m/s até milhões de anos-luz/s. Pontos de observação podem ter diferentes direções: para frente, para trás ou qualquer outro ângulo. O tempo pode ser simulado em qualquer ponto do futuro ou do passado. Os nomes e as posições dos objetos no espaço podem ser mostrados, bem como os nomes de localidades na Terra.

O Celestia mostra diferentes características dos planetas, tais como atmosfera, brilho, satélites, aurora e pôr-do-sol, nuvens, eclipses, anéis planetários, textura das superfícies, gases, etc. Também informa radianos dos objetos, comprimento do dia sideral, temperatura média, distância até o Sol e luminosidade relativa.

No entanto, há algumas limitações ao modelamento do Celestia:

- O parâmetro *default* para a Terra no Celestia é um esferóide perfeito, o que significa que satélites em órbita terrestre baixa não são modelados com precisão. No entanto, este parâmetro pode ser ajustado;
- Muitas, se não todas as estrelas binárias não são simuladas corretamente;
- Apenas objetos no sistema solar são móveis; estrelas e galáxias são imóveis;
- Originalmente não há nebulosas no Celestia, mas elas podem ser adicionadas.

3.2.3 Utilização no projeto

O Celestia está sendo utilizado como um software de referência para validar uma das ferramentas de visualização desenvolvidas.

3.2.4 Versão utilizada

- Celestia 1.6.0

3.3 Orbitron

3.3.1 Histórico

O Orbitron é um programa gratuito para rastreio de satélites, desenvolvido em 2001 por Sebastian Stoff [24].

3.3.2 Visão Geral

O Orbitron é utilizado como um sistema de rastreamento de satélites para observação celeste e para rádios amadores. É também utilizado por meteorologistas, usuários de comunicação via satélite, astrônomos, ufólogos e até mesmo astrólogos**.** O Orbitron mostra a posição de satélites em qualquer momento, seja em tempo real ou simulado.

Características:

- Modelos de predição NORAD SGP4/SDP4;
- Podem ser carregados 20.000 satélites a partir dos arquivos TLE, através da Internet;
- Todos os satélites podem ser rastreados ao mesmo tempo;
- Rastreamento Solar e Lunar;
- Modo tempo real / Modo simulado;
- Radar;
- Banco de dados das cidades ao redor do mundo;
- Banco de dados das frequências dos satélites;

- Controle de Rotores/Rádios.

3.3.3 Utilização no projeto

O Orbitron está sendo utilizado como um software de referência para validar uma das ferramentas de visualização desenvolvidas.

3.3.4 Versão utilizada

- Orbitron 3.71

4. Fundamentação Teórica

Este capítulo tem como objetivo apresentar os conceitos teóricos que serão necessários para o entendimento do trabalho. Inicia-se com uma explicação sobre alguns eixos referenciais fundamentais e em seguida serão analisadas as órbitas e os parâmetros utilizados para defini-las. Por último, serão conceituados os termos SSP (*Sub Satelite Point*), IAA (*Instant Access Area*) e TLE (*Two Line Elements*) e estabelecida a importância destes conceitos neste trabalho.

4.1 *Reference Frame*

Para analisar o controle de atitude, é necessário entender os *Reference Frames* (Eixos de Coordenadas). *Reference Frames* são conjuntos de três retas ortonormais (ortogonais e de comprimento unitário) [25] que possuem uma orientação conhecida. A atitude de um satélite é fornecida em unidades angulares, medidas a partir destes eixos.

O primeiro dos quatro *Reference Frames* é chamado de ECI (*Earth Centered Inertial*). Esse *Frame* tem como origem o centro da Terra, com seu eixo $\hat{I}$ apontando para o ponto vernal e o plano $\hat{I}\hat{J}$ idêntico ao plano equatorial. O eixo $\hat{K}$ é coincidente com o eixo de rotação da Terra [3]. Esse referencial é estático para todo objeto orbitando a Terra. Para que isso seja verdade, deve-se desconsiderar o movimento de precessão do eixo de rotação da Terra, que é insignificante para intervalos de tempo de dias ou meses.

O segundo *Frame* é chamado de ECEF (*Earth Centered Earth-Fixed*). Também tem sua origem no centro da Terra, porém gira sobre o eixo $\hat{K}$ com velocidade igual à velocidade de rotação da Terra. Sua utilidade reside no fato de que um ponto sobre a superfície da Terra está imóvel em relação a esse *Frame*, que não é estático para um objeto em órbita, uma vez que este gira.

O terceiro *Frame* é de extrema importância: o OF (*Orbital Frame*). Tem como origem o centro de massa do satélite. Seus eixos são chamados de $\hat{O}_1$, $\hat{O}_2$ e $\hat{O}_3$. O Eixo $\hat{O}_3$ é o vetor unitário que aponta do centro de massa do satélite para o centro da Terra; o $\hat{O}_2$ aponta para a direção oposta à normal da órbita. Por sua vez, o eixo $\hat{O}_1$ é o vetor perpendicular a estes dois $\hat{O}_1 = \hat{O}_2 \times \hat{O}_3$, que para o caso de uma órbita circular, coincide com a direção do vetor velocidade do satélite.

O quarto *Frame* de interesse é o BF (*Body Frame*). Seus eixos são chamados de $\hat{B}_1$, $\hat{B}_2$ e $\hat{B}_3$. Assim como o OF, também tem sua origem no centro de massa do satélite. Porém, o BF nem sempre tem seus eixos alinhados com o OF. Essa diferença entre eles caracteriza a atitude do satélite, isto é, a diferença angular entre os eixos do BF e do OF.

4.2 Parâmetros Orbitais

Toda órbita pode ser definida singularmente por um conjunto de parâmetros, que são chamados parâmetros orbitais. Dados um referencial inercial e um *epoch* (um ponto específico no tempo), são necessários seis parâmetros para definir completamente uma órbita, conforme detalhado a seguir.

4.2.1 Inclinação Orbital

A elipse de uma órbita se localiza sobre um plano conhecido como plano orbital. A inclinação deste plano em relação ao plano equatorial (plano perpendicular ao eixo de rotação da Terra) define a inclinação orbital. Satélites GEO têm inclinação orbital zero, pois estão sempre sobre a linha do Equador [26].

Por convenção, inclinação orbital é um número entre zero e 180 graus. Se a inclinação estiver entre zero e 89 graus, o satélite gira na mesma direção da rotação da Terra; caso

estiver entre 91 e 180, o satélite gira na direção oposta à rotação da Terra. A inclinação orbital está ilustrada a seguir na Figura 1.

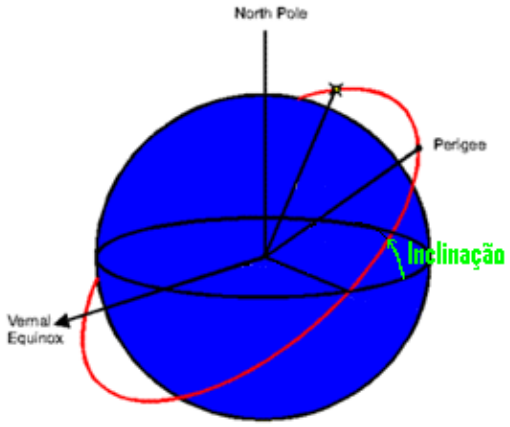


Figura 1 - Inclinação Orbital

4.2.2 *Right Ascension of Ascending Node* – RAAN

RAAN é o ângulo formado entre a reta que sai do centro da Terra em direção ao ponto vernal e o ponto sobre a órbita que cruza o plano equatorial da Terra (nodo ascendente) do sul para o norte. Satélites com RAAN zero passam do Hemisfério Sul para o Hemisfério Norte exatamente sobre o ponto vernal. Por convenção, RAAN é um número entre zero e 360 graus. Na Figura 2 encontra-se ilustrado o ângulo referente ao RAAN.

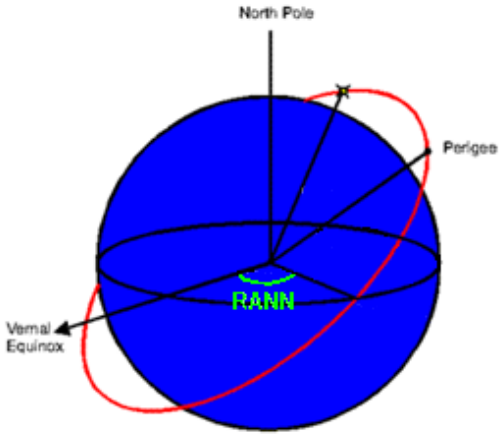


Figura 2 - Right Ascension of Ascending Node

4.2.3 *Argument of Perigee* – AP

Sabendo que o ponto em que o satélite se encontra mais próximo da Terra é chamado de Perigeu (*Perigee*), e o ponto em que está mais afastado é chamado de Apogeu (*Apogee*), é possível entender o conceito de *Argument of Perigee.*

AP é o ângulo entre a linha que liga o centro da Terra ao Perigeu da órbita e a linha que liga o centro da Terra ao ponto em que a órbita passa do hemisfério sul para o norte (Nodo Ascendente). Por convenção, AP é um número entre zero e 360 graus. O AP está destacado na Figura 3.

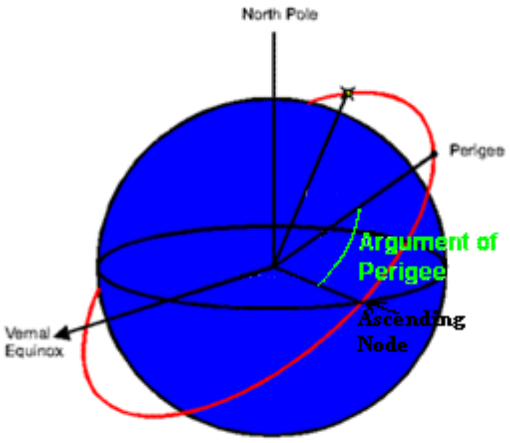


Figura 3 - Argument of Perigee

4.2.4 Semi-Eixo Maior

Dentre as maneiras de se definir a magnitude da órbita (suas distâncias da Terra), optou-se por utilizar a dimensão em quilômetros do Semi-Eixo Maior da elipse que é descrita pela órbita. O Semi-Eixo Maior é a distância entre o centro da elipse e o ponto de Apogeu ou Perigeu, e pode ser visualizado na Figura 4.

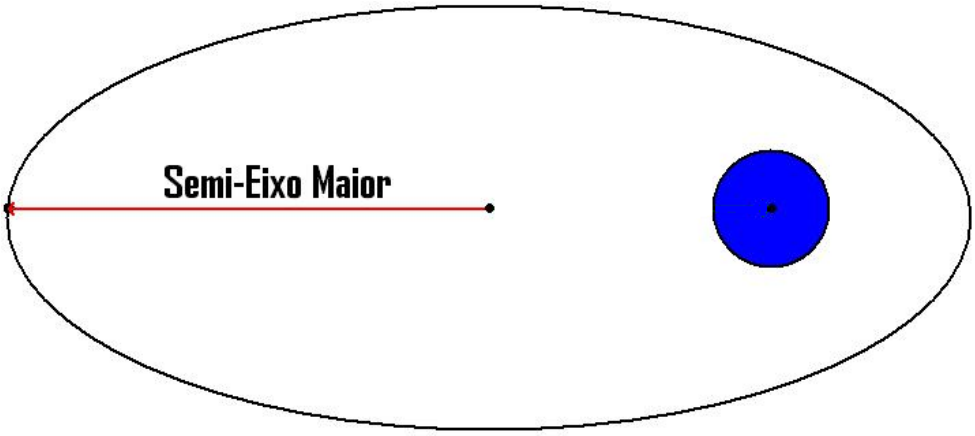


Figura 4 - Semi-Eixo Maior

4.2.5 Excentricidade

No modelo de Kepler os satélites possuem órbitas elípticas. É possível estabelecer o formato dessa elipse definindo apenas o valor de sua excentricidade. A excentricidade de uma elipse é obtida dividindo-se a distância do centro da elipse até um de seus focos pela distância do centro da elipse até o Perigeu (Semi-Eixo Maior).

Uma órbita circular é um caso especial de órbita elíptica: uma elipse com excentricidade zero é um círculo. Quanto mais próxima de um for a excentricidade, mais fina e alongada é a elipse [27]. A Figura 5 ilustra as distâncias utilizadas no cálculo da excentricidade.

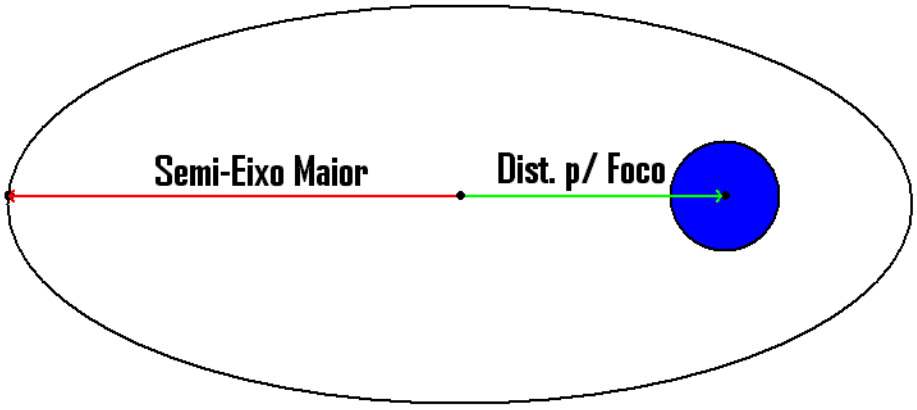


Figura 5 - Excentricidade

4.2.6 Mean Anomaly

Mean Anomaly define a posição do corpo (satélite) na órbita no momento da medição (*Epoch*). É definida como o tempo desde a última passagem pelo Perigeu. A *Mean Anomaly* não tem uma representação geométrica simples; é basicamente tempo medido em radianos.

A partir da *Mean Anomaly* é possível calcular a *Eccentric Anomaly*, e desta, a *True Anomaly*. A *True Anomaly* tem uma representação geométrica simples: é o ângulo entre a linha que liga o centro da Terra ao ponto de Perigeu e a linha que liga o centro da Terra ao centro de massa do satélite [26]. Na Figura 6 visualiza-se a *True Anomaly*.

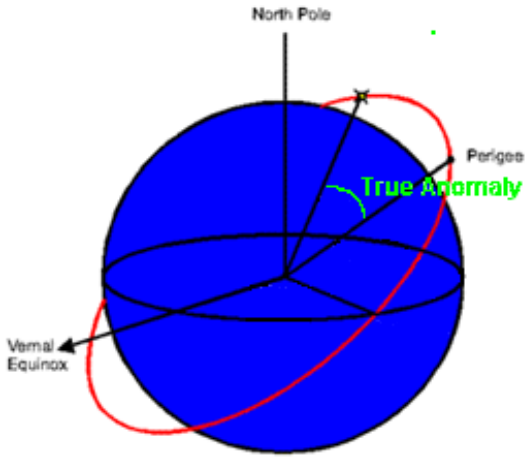


Figura 6 - True Anomaly

4.3 SSP (Sub Satellite Point)

Dada a posição de um satélite em um dado instante, traçando uma reta que liga o centro da Terra ao centro de massa do satélite, invariavelmente a superfície da Terra terá sido cruzada. O ponto em que a reta intersecciona a superfície é denominado SSP [3].

Demarcando infinitos pontos em um intervalo de tempo, o satélite definirá um rastro na superfície da Terra, chamado de *ground track*.

O *ground track* de um satélite é a informação que permite prever sobre quais localidades o satélite passará, possibilitando ou não a observação de um determinado alvo na superfície da Terra.

Embora no espaço os satélites sempre percorram elipses, observando seus *ground tracks* surgirão diferentes figuras, pois o *ground track* é uma combinação de dois movimentos: o do satélite em sua órbita e da rotação da Terra. A Figura 7 mostra o *ground track* de um satélite em órbita geossíncrona com uma inclinação orbital de 60 graus [28].

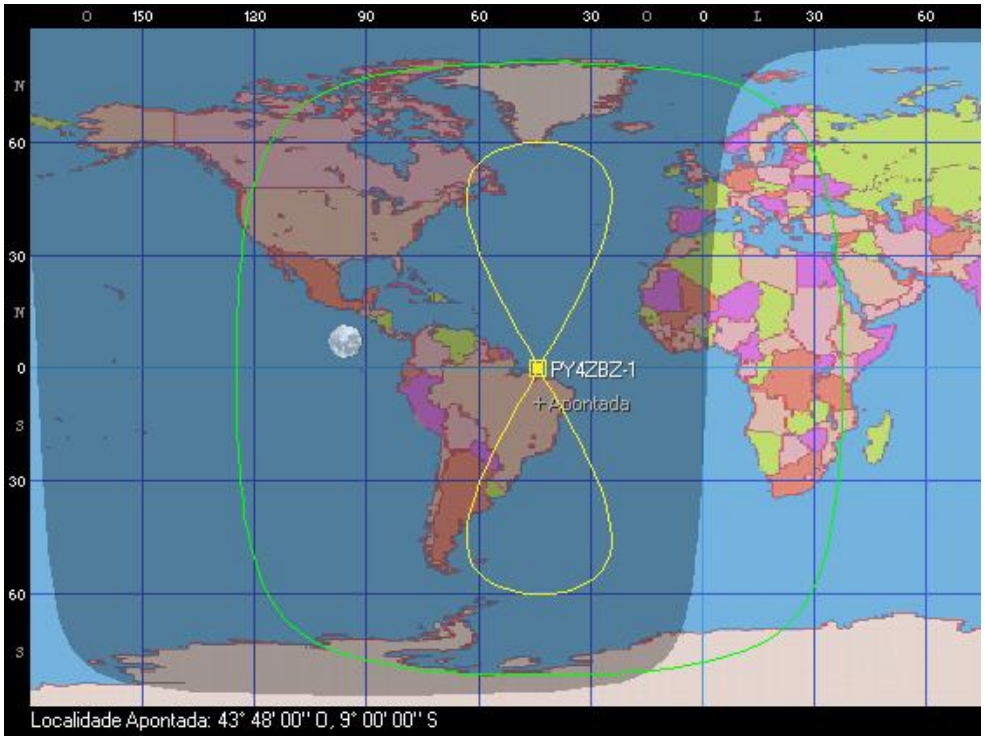


Figura 7 - Ground Track de um satélite GEO, vista no Orbitron.

4.4 IAA (*Instantaneous Access Area*)

Observando a Terra de sua órbita, um satélite enxerga uma região circular centrada em seu SSP. Essa região circular é chamada de IAA [29]. Conforme o SSP se move na superfície da Terra, essa região o acompanha, fazendo uma varredura da superfície.

Esta área circular chamada IAA tem como limites o horizonte observável pelo satélite. É possível concluir intuitivamente que quanto mais próximo da superfície da Terra o satélite se encontrar menor será o raio deste círculo, e menos pontos na superfície poderão ser observados pelo satélite em um mesmo instante.

Essa informação é fundamental para este trabalho, pois é necessária para definir se uma determinada coordenada na superfície da Terra será visível pelo satélite ou não. Se a coordenada estiver dentro dos limites da IAA, pode-se calcular a atitude do satélite para que aponte para este alvo.

4.5 TLE (*Two-Line Element*)

Um TLE é um arquivo padronizado que descreve os elementos orbitais de um satélite terrestre. A partir de um TLE atualizado é possível fazer a predição do posicionamento do satélite em um determinado momento no tempo. O formato utilizado foi criado pela NORAD (*North American Aerospace Defense Command*) em 1972 [30], com o objetivo de transmitir os elementos orbitais (*kleperian elements*) de forma bastante compacta. Por este motivo, são de difícil interpretação a menos que se esteja bastante familiarizado com a sua estrutura.

Por alguma ironia, um *Two-Line Element* possui não duas, mas três linhas. A primeira contém apenas o nome do satélite e possui até 24 caracteres. As outras duas linhas possuem 69 caracteres e contêm os elementos orbitais básicos descritos anteriormente e mais algumas informações não tão relevantes. O formato aceito em um arquivo TLE é bastante restrito: os caracteres válidos são os números de 0-9, as letras maiúsculas A-Z, o espaço, o ponto e os sinais de adição e subtração.

Claro que nem todos os caracteres podem ser usados em qualquer parte do TLE, a organização das informações também é bastante rígida. A Figura 8 mostra quais os tipos de caracteres aceitos em cada coluna do TLE. Colunas com espaço ou ponto só podem conter

esses caracteres. Colunas com um N podem conter números de 0-9 ou em alguns casos, espaço. A Coluna que possui um C pode conter apenas ‘U’ de *Unclassified* ou ‘S’ de *Secret*. Colunas que possuem um A podem conter as letras maiúsculas de A-Z ou espaço. Colunas com ‘+’ podem conter o sinal de adição, de subtração ou espaço, enquanto as colunas com ‘-‘ podem conter apenas os sinais de adição ou subtração.

```
1 NNNNNC NNNNNAAA NNNNN.NNNNNNNN +.NNNNNNNN +NNNNN-N +NNNNN-N N NNNNN
2 NNNNN NNN.NNNN NNN.NNNN NNNNNNN NNN.NNNN NNN.NNNN NN.NNNNNNNNNNNNN
```

Figura 8 - Formatação do TLE

Além destas restrições, alguns campos possuem características que restringem os seus valores válidos. A seguir cada um dos campos será detalhado e estas restrições comentadas.

Começando com a primeira linha:

Coluna 01:

Número que referencia a linha do TLE. Valor obrigatório um.

Colunas 03~07:

Número do cadastro do satélite no catálogo da NORAD, todo satélite artificial tem um número único que o identifica.

Coluna 08:

Classificação do satélite. Letra ‘U’ de *Unclassified* ou ‘S’ de *Secret.* Todos os TLE’s públicos que temos acesso pertencem a satélites do tipo *Unclassified.*

Colunas 10~17:

Referentes ao designador internacional concebido pelo WDC-A-R&S. Também é utilizado para identificar o satélite. As colunas 10~11 contêm os últimos dois dígitos do ano de lançamento do satélite. As colunas 12~14 contêm o número do lançamento no ano, e as colunas 15~17 indicam qual objeto dentre os vários objetos que podem ter sido postos em órbitas com aquele lançamento.

Colunas 19~20:

Contêm os últimos dois dígitos do ano em que foi feita a observação para a aquisição dos dados contidos no TLE (*Epoch Year*). Ainda não houve necessidade de se mudar isso, mesmo com a virada do milênio, pois, como não existe satélite lançado antes dos anos 50, qualquer TLE com o valor de ano abaixo disto foi lançado depois do ano 2000.

Colunas 21~32:

Contêm o dia do ano e a fração do dia em que foi feita a observação para a aquisição dos dados contidos no TLE (*Epoch Day*).

Colunas 34~43:

Derivada de primeira ordem do *Mean Motion*, não tem utilidade em nosso modelo de previsão.

Colunas 45~52:

Derivada de segunda ordem do *Mean Motion*, também não tem utilidade em nosso modelo de previsão.

Colunas 54~61:

Coeficiente de arrasto do satélite, também não tem utilidade em nosso modelo de previsão.

Coluna 63:

Indica qual o modelo orbital utilizado para gerar o TLE, todos os TLE publicados são gerados usando o modelo SGP4/SDP4 e tem esse campo preenchido com zero.

Colunas 65~68:

Indica quantas vezes foram gerados os dados para esse objeto, toda vez que é feita uma atualização este número é atualizado.

Coluna 69:

Check Sum (módulo-10) dos dados da primeira linha do TLE. Todos os números da linha são somados, ignorando-se os espaços, pontos, letras e sinais de adição. Os sinais de subtração valem um. Feito isso, o último digito desta soma é o valor procurado.

A primeira linha do TLE contém poucas informações realmente úteis para esse trabalho. O modelo implementado utiliza apenas o *Epoch* (colunas 19~32) em suas previsões. O restante das informações necessárias são extraídas da segunda linha do TLE, detalhada a seguir.

Coluna 01:

Número que referencia a linha do TLE. Valor obrigatório um.

Colunas 03~07:

Número do cadastro do satélite no catalogo da NORAD, todo satélite artificial tem um número único que o identifica.

Coluna 09~16:

Contêm o valor da inclinação da orbital do satélite. Este valor pode variar de zero a 180 graus.

Colunas 18~25:

Contêm o valor do RAAN da órbita do satélite. Este valor pode variar de zero a 360 graus.

Colunas 27~33:

Contêm o valor da excentricidade da órbita do satélite. Este valor contém um ponto flutuante implícito. Por exemplo, o valor 1234567 neste campo, significa uma excentricidade de 0.1234567.

Colunas 35~42:

Contêm o valor do *Argument f Peregee* da órbita do satélite. Este valor pode variar de zero a 360 graus.

Colunas 44~51:

Contêm o valor da *Mean Anomaly* da órbita do satélite. Este valor pode variar de zero a 360 graus.

Colunas 53~63:

Contêm o valor da *Mean Motion* da órbita do satélite. Este valor está em número de revoluções por dia.

Colunas 64~68:

Contêm o número de revoluções completas até o momento da obtenção dos dados para o TLE.

Coluna 69:

Como na primeira linha, é o *Check Sum* (módulo-10) dos dados da segunda linha do TLE.

Um TLE só é válido quando respeita todas as regras descritas aqui, tanto as relativas à sua estrutura, quanto as referentes aos valores permitidos nos parâmetros orbitais. A seguir duas tabelas que resumem as informações detalhadas nesta seção.

Tabela 1 - Descrição dos campos do TLE, Linha 1.

Primeira Linha	
Coluna	**Descrição**
01	Linha do TLE
03~07	Número do Satélite
08	Classificação
10~11	Designador Internacional (últimos dois dígitos do ano do lançamento)
12~14	Designador Internacional (número do lançamento no ano)
15~17	Designador Internacional (objeto no lançamento)
19~20	*Epoch Year* (últimos dois dígitos do ano)
21~32	*Epoch Day* (dia do ano e fração do dia)
34~43	Primeira Derivada do *Mean Motion*
45~52	Segunda derivada do *Mean Motion*
54~61	Coeficiente de arrasto
63	Modelo Orbital
65~68	Número do TLE
69	*Checksum*

Tabela 2 - Descrição dos campos do TLE, Linha 2.

Segunda Linha	
Coluna	**Descrição**
01	Linha do TLE
03~07	Número do Satélite
09~16	Inclinação
18~25	RAAN
27~33	Excentricidade
35~42	*Argument of Perigee*
44~51	*Mean Anomaly*
53~63	*Mean Motion*
64~68	Número de revoluções até o *Epoch*
69	*Checksum*

5. Ferramentas Desenvolvidas

Existe uma grande dificuldade de visualizar órbitas, rotações e deslocamentos no espaço. Para que fosse possível verificarmos o correto funcionamento do calculo de atitude desenvolvido neste trabalho, foi necessário criar uma série de ferramentas auxiliares de visualização. Essas ferramentas permitiram verificar passo a passo se o entendimento do assunto estava correto. Neste capítulo abordaremos as quatro principais ferramentas (ou funções Matlab) implementadas.

As ferramentas foram todas desenvolvidas utilizando Matlab, portanto, é necessário que este esteja devidamente instalado e atualizado numa versão igual ou superior a que foi utilizada (ver capitulo 2.1). Dentro do Matlab, os arquivos desenvolvidos estão no formato de funções, ou seja, são rotinas chamadas a partir do console, que recebem parâmetros e retornam valores ou imagens.

A seguir estão detalhadas as quatro principais funções, com o formato de suas entradas, os valores aceitos e as saídas esperadas.

5.1 *Extract_Orbit*

Esta é a primeira função que deverá ser utilizada. Ela serve basicamente para extrair de um arquivo TLE válido, os parâmetros orbitais utilizados nas demais funções.

Todas as outras funções dependem do correto funcionamento desta, portanto é muito importante que o arquivo TLE passado a ela respeite todas as regras descritas no capitulo anterior.

A sugestão é utilizar arquivos TLE vindos de fontes confiáveis como do site http://celestrak.com ou http://www.space-track.org. Este segundo é necessário um

cadastro e aprovação por parte do governo americano, portanto, o site Celestrak é o mais aconselhado.

A função *Extract Orbit* deve ser chamada como mostra a Figura 9.

```
>> Extract_Orbit('file_path')
```

Figura 9 - Parâmetros da *Extract Orbit.*

O *file_path* deve ser o caminho para o arquivo texto (.txt) que contém o TLE da órbita com a qual se deseja trabalhar. Essa função irá retornar uma variável que chamamos de KE, esta variável é uma matriz de uma linha e oito colunas, contendo os seguintes parâmetros orbitais: Semi-Eixo Maior (a), excentricidade (e), *right ascension of the ascending node* (RAAN), inclinação (i), *argument of periapsis* (w), *Mean anomaly at Epoch* (ma), *Epoch Year* e *Epoch Day*, nessa ordem.

Note que o Semi-Eixo Maior não é um parâmetro encontrado diretamente no arquivo que descreve a órbita (TLE), porém, este pode ser calculado a partir do valor do *Mean Motion* (mm) encontrado nas colunas 45 a 52 da primeira linha do TLE. Para isso utiliza-se a fórmula da Figura 10.

```
a = (3.99e+5/( ((mm*2*pi)/86400)^2) )^(1/3);
```

Figura 10 - Fórmula para o cálculo do Semi-Eixo Maior da órbita a partir da *Mean Motion* do TLE.

Na equação, “mm” é o valor da *Mean Motion* encontrada no TLE, 86400 é o número de segundos em um dia solar e 3.99e+5 é a constante para que a saída esteja em quilômetros. Optamos por utilizar o Semi-Eixo Maior da elipse como parâmetro de magnitude da órbita, pois este é de mais fácil compreensão e melhor visualização.

A Matriz que a função *Extract_Orbit* retorna deve ser armazenada em uma variável temporária, e esta utilizada como parâmetro para as demais funções.

5.2 *View_Orbit*

Esta segunda função nos permite visualizar como a órbita se posiciona em relação ao ECI. Juntamente com a órbita, uma esfera terrestre também é renderizada para que seja possível visualizar o quanto o satélite se afasta da Terra e como a excentricidade influencia nas distancias do satélite à superfície.

A função *View_Orbit* recebe como parâmetro uma variável do tipo KE, que é uma matriz 1x8 contendo os parâmetros orbitais. Essa variável pode ser gerada manualmente, porem é altamente recomendado que isso seja feito através da função *Extract_Orbit*, pois assim garantimos que os valores contidos na matriz são válidos.

A chamada da *View_Orbit* deve ser feita como mostra a Figura 11.

```
>> View Orbit(KE)
```

Figura 11 - Parâmetros da *View_Orbit.*

O KE deve ser a variável contendo a matriz 8x1 com os parâmetros orbitais como descritos na seção 4.1. Essa função não retorna nenhum valor, ela apenas renderiza a esfera terrestre, a órbita requisitada e uma órbita circular equatorial de mesmo Semi-Eixo Maior que a requisitada. Esta órbita extra serve de referencia para facilitar a visualização. O resultado depende dos parâmetros passados na matriz, porem será algo parecido com o mostrado na Figura 12.

A Figura 12 demonstra o resultado da função *View_Orbit* para uma órbita com Semi-Eixo Maior 33720 km, inclinação 45 graus e excentricidade 0.6, os demais parâmetros estão zerados.

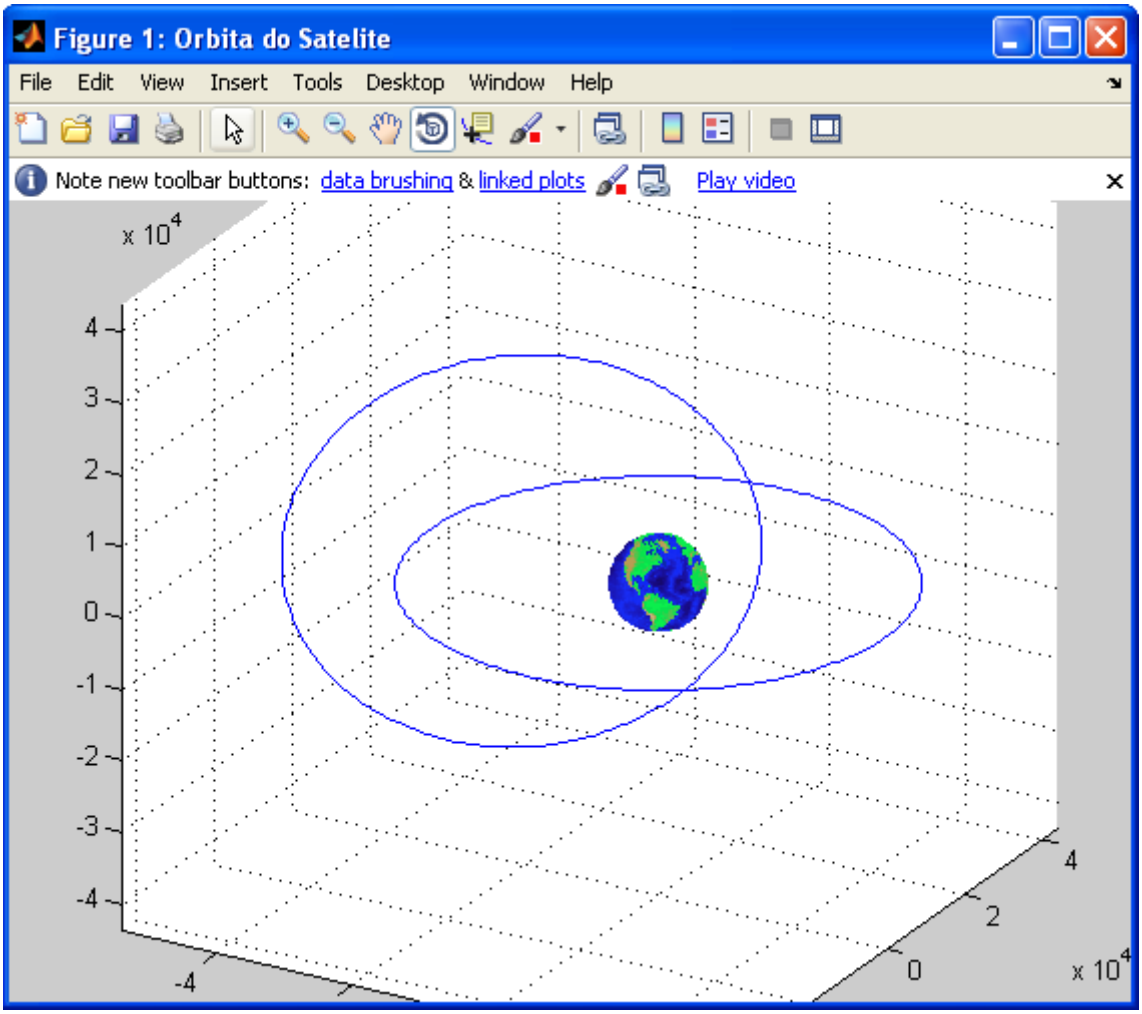


Figura 12 - Resultado da Função *View Orbit.*

Para que fosse possível desenhar a órbita, foi utilizada uma equação que a partir do valor da excentricidade e do Semi-Eixo Maior, nos retorna o raio do vetor que liga o centro da Terra (ou o foco da elipse) a posição do satélite [27]. A Figura 13 apresenta a fórmula.

```
r = (a*(1-(e^2)))./(1+ e*cos(t));
```

Figura 13 - Fórmula para cálculo do raio ao longo da órbita.

Na equação, o "a" é o Semi-Eixo Maior e o "e" é a excentricidade. Estes valores são constantes para uma dada órbita, o valor de "t" é que varia de zero a 2π em intervalos de $\pi/128$. Assim obtemos 256 valores de raio referentes a uma rotação completa. Feito isso, é criada uma série de vetores sobre o plano XY que representam os pontos sobre a elipse da órbita, porém deitada sobre o plano XY.

Agora com a elipse representada em vetores, podemos utilizar outra função, criada para fazer a rotação de vetores sobre um eixo e, assim, rotacionar os vetores que representam a órbita, primeiro do valor do RAAN sobre o eixo Z, seguido por uma rotação do valor da inclinação sobre o eixo X (lembrando que o eixo X também rotacionou do valor do RAAN juntamente com a órbita na primeira rotação) e mais uma rotação do valor do *argument of periapsis* novamente sobre o eixo Z (tendo o eixo X feito as duas rotações posteriores juntamente com a órbita). Por fim esses vetores são renderizados no espaço 3D.

5.3 *View_SSP*

Esta terceira função possui dois objetivos principais. O primeiro é o de mostrar como o satélite se move em sua órbita, mostrando não somente o caminho por ele percorrido, mas também a variação da velocidade conforme a distância do mesmo ao centro da Terra muda. O segundo objetivo é o de mostrar por sobre que localidades na superfície da Terra o satélite passou durante o período de um dia solar, logo após a aquisição dos dados do TLE.

Assim como a *View_Orbit*, a função *View_SSP* recebe como parâmetro uma variável do tipo KE contendo os dados referentes à órbita que se deseja observar. Em um primeiro momento, o algoritmo executa uma animação, onde podemos observar a variação da posição e da velocidade do satélite.

A animação contém 720 *frames* (ou passos), onde a cada passo se recalcula a *Mean Anomaly* do satélite para aquele momento segundo a fórmula da Figura 14 [3].

```
MeanAnomaly = 2*pi/T*119.6722*ani + ma;
```

Figura 14 - Fórmula da *Mean Anomaly*.

Na equação, “ma” é a *Mean Anomaly at Epoch* contida na matriz KE passada por parâmetro, “ani” é o passo corrente da animação, “119.6722” é quantos segundos deveriam ter passado entre cada um dos frames da animação e “T” é o período da órbita, calculado pela fórmula da Figura 15 [27].

```
T = (2*pi)* sqrt((a^3)/3.99e+5);
```

Figura 15 - Fórmula do Período.

Segundo a terceira Lei de Kepler [32], podemos calcular o período da órbita apenas sabendo o valor do seu Semi-Eixo Maior e, como temos esse valor previamente calculado (Figura 10) na nossa matriz KE, o utilizamos na fórmula da Figura 15 como a variável "a" e obtemos o período da órbita.

Tendo o valor da *Mean* Anomaly do satélite para aquele instante no tempo, podemos então calcular a *Eccentric Anomaly* e, a partir desta, a *True Anomaly*, que nos fornece a real posição do satélite na órbita. Devido à natureza do cálculo da *Eccentric Anomaly*, é necessário utilizar um método numérico para sua solução [22]. A Figura 16 mostra como esse procedimento é feito no Matlab.

```
f = @(x)x-e*sin(x) - MeanAnomaly;
EA = fzero(f,5);
trueAnomaly = 2*atan2(sqrt(1+e)*sin(EA/2), sqrt(1-e)*cos(EA/2));
```

Figura 16 - Cálculo da *Eccentric Anomaly* e *True Anomaly*.

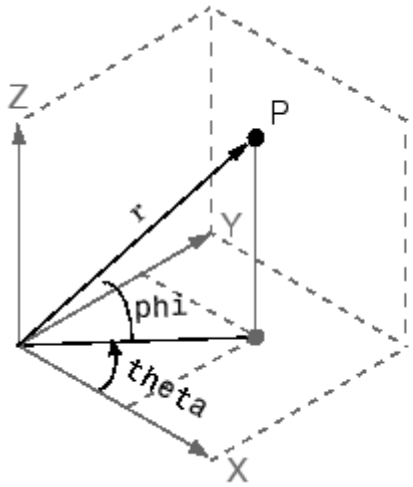
Na Figura 16 temos "f" recebendo uma função e, em seguida, utilizamos o método do matlab fzeros(), que resolve numericamente a função "f" e retorna o valor encontrado em "EA" (*Eccentric Anomaly*). Este valor é então usado na equação da *True Anomaly*, juntamente com a excentricidade "e" contida na matriz KE.

De posse destes dados, podemos fazer variar a real posição do satélite ao longo de intervalos regulares de tempo; no caso, a cada dois minutos. Assim teremos uma animação realista que inclui a variação da velocidade do satélite ao longo do tempo.

Ao encerrar-se a animação, a segunda etapa da função entra em vigor, onde é criado um mapa bidimensional da superfície da Terra, e sobre ele é traçado o rastro do SSP do satélite. A segunda etapa faz uso dos vetores calculados na primeira etapa.

Os 720 vetores de posição do satélite calculados para a animação são então convertidos de coordenadas cartesianas para coordenadas esféricas e, com isso, obteremos o ângulo

que esses vetores fazem com os eixos do ECI. A Figura 17 mostra como é feita essa conversão.




```
theta = atan2(y,x)
  phi = atan2(z, sqrt(x.^2 + y.^2))
    r = sqrt(x.^2+y.^2+z.^2)
```

Figura 17 - Conversão de coordenadas cartesianas para esféricas.

Uma vez que a Terra também esta centrada nos eixos do ECI, podemos inferir que o ângulo “theta” pode ser diretamente relacionado a uma longitude e o ângulo “phi” a uma latitude. Porem, a Terra não esta estática em relação ao ECI, como ela possui um movimento de rotação sobre o eixo Z, ainda é necessário definir o quanto a Terra girou em relação ao ECI na data em que foi obtido o TLE.

Para isso foi desenvolvida a função *JDate_Angle*, que recebe como parâmetros o *Epoch Year* e o *Epoch Day of the Year* contidos na matriz KE e retorna o ângulo que o meridiano de Greenwich (longitude zero) faz com o vetor que aponta para o ponto vernal (Eixo X do nosso referencial).

Pode ser observado na Figura 18 que diferentemente da função *View_Orbit*, a função *View_SSP* se preocupa com a orientação da Terra na data, e não somente a da órbita. Para facilitar a visualização do SSP, uma linha ligando o satélite (ponto vermelho na órbita) ao centro da Terra (que cruza a superfície terrestre no SSP) pode ser vista varrendo a sua superfície conforme o satélite orbita.

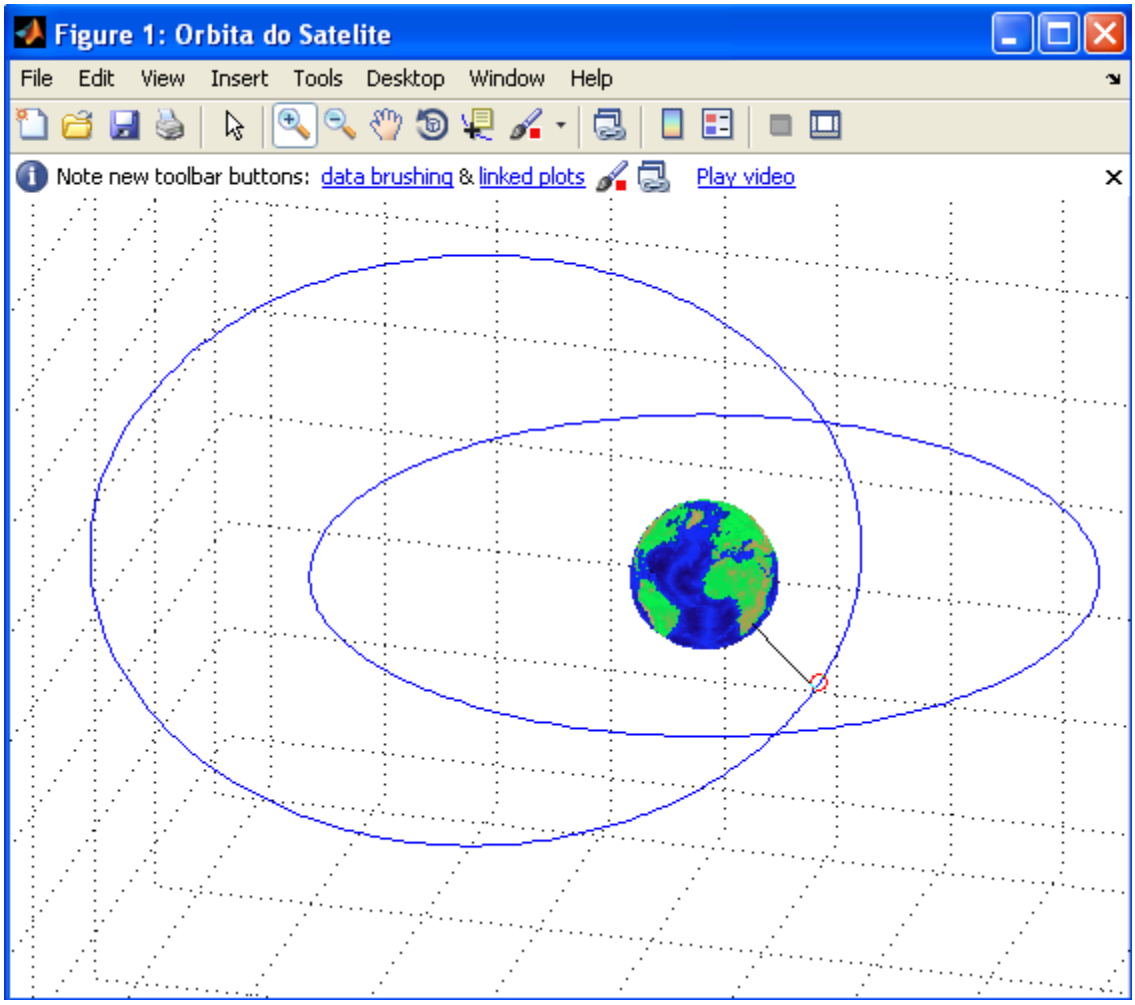


Figura 18 - Imagem mostrando a órbita, a posição do satélite e o SSP.

Agora que sabemos a rotação em graus da posição do satélite em relação eixo X, bem como a rotação da superfície da Terra em relação a esse eixo, podemos obter a longitude e a latitude do satélite em relação à superfície terrestre subtraindo um do outro. De posse desses 720 pares de longitude e latitude, os inserimos sobre um mapa 2D da superfície terrestre. A Figura 19 mostra o *SSP maping* para uma órbita com 45 graus de inclinação, 33720 km de Semi-Eixo Maior, excentricidade 0.6, *Epoch Year* 2009 e *Epoch Day* 314 (10 de novembro).

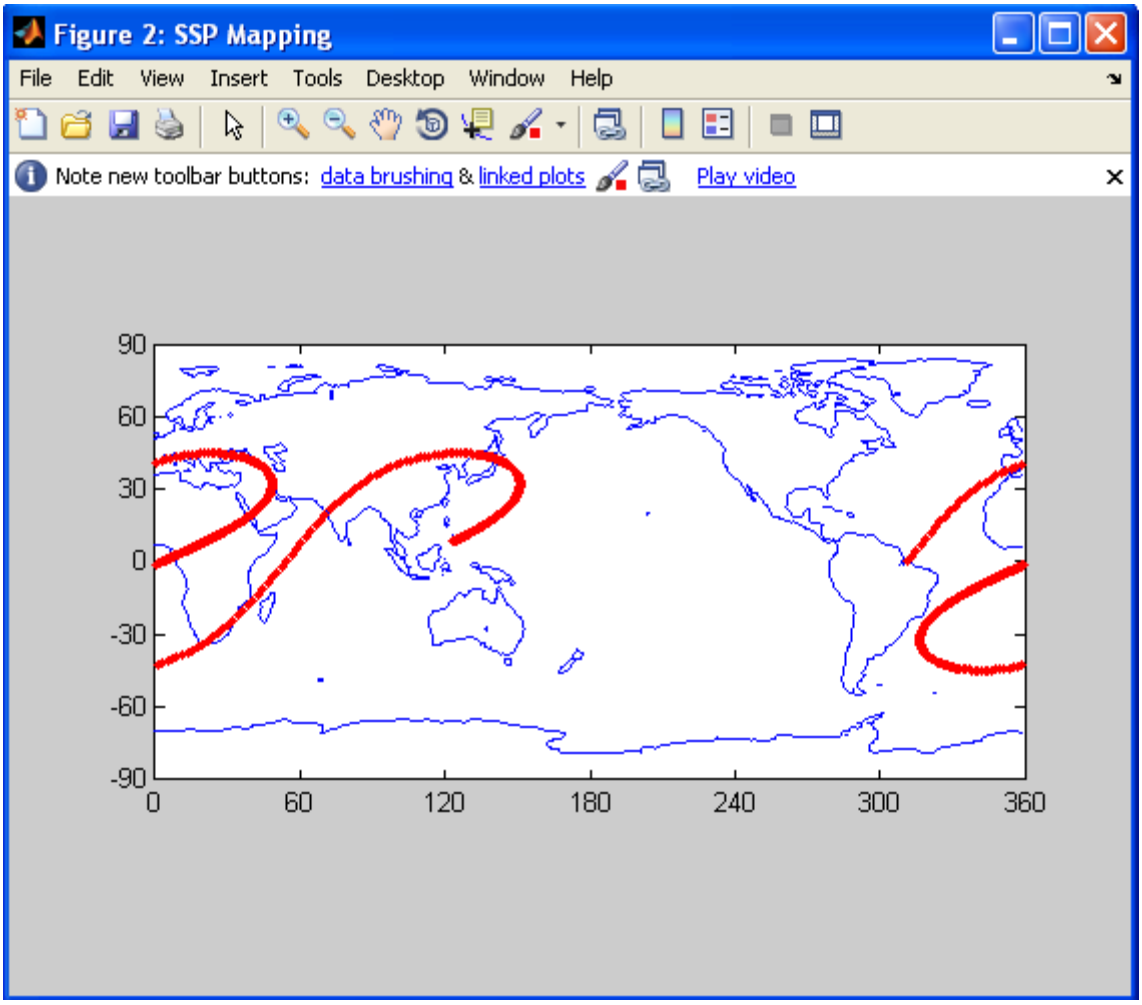


Figura 19 - *SSP Mapping*.

Como o satélite do exemplo da Figura 19 possui uma órbita relativamente alta e com uma excentricidade mediana, ao colocarmos os 720 pontos no mapa eles ficam muito próximos uns dos outros, parecendo uma simples linha. Porém em órbitas mais baixas ou com elevada excentricidade, onde o satélite percorre grandes distâncias em intervalos de tempo menores, podemos observar que a densidade do traçado varia com a velocidade do satélite.

5.4 *View_IAA*

Esta quarta função existe com a finalidade principal de mostrar graficamente quais os pontos na superfície da Terra estão visíveis para o satélite em um determinado momento no tempo. Para isso é utilizado um mapa 2D da superfície, onde é desenhada uma série de pontos que demarcam os limites visuais do satélite.

Como já foi explicado anteriormente (seção 4.4), em qualquer ponto da sua órbita o satélite enxerga uma região circular da superfície da Terra, centrada em seu SSP. Porém, ao observarmos a área delimitada por esse círculo em um mapa 2D, veremos que nunca será desenhado exatamente um círculo. Isso ocorre devido as deformidades impostas pela conversão de 3D para 2D. Para entendermos isso, basta lembrar que, em um mapa como o da Figura 19, todos os pontos com latitude 90 são na realidade o mesmo ponto, independentemente de sua longitude.

A função *View_IAA*, diferentemente das funções anteriores, não trabalha com um intervalo de tempo e sim com um momento específico no tempo. Devido a isso, além da matriz contendo os parâmetros orbitais, deve ser informado em que data se deseja visualizar a IAA do satélite. A Figura 20 mostra como devem ser passados os parâmetros.

```
>> View_IAA(KE,ano,mes,dia,hora,min,seg)
```

Figura 20 - Parâmetros da *View_IAA*.

Na Figura 20, “KE” contém a matriz com os parâmetros orbitais, as variáveis “ano”, “mes”, “dia”, “hora”, “min” e “seg” contêm os valores de data e hora em que se deseja visualizar a IAA. Além do mapa com as marcações, no console serão informadas pela função a longitude e a latitude do SSP do satélite na data requerida.

Para descobrir o raio do círculo visto pelo satélite foi utilizada trigonometria. A distância do satélite ao centro da Terra é conhecida, assim como o raio da Terra. Utilizando essas duas informações, montamos o triângulo retângulo mostrado na Figura 21.

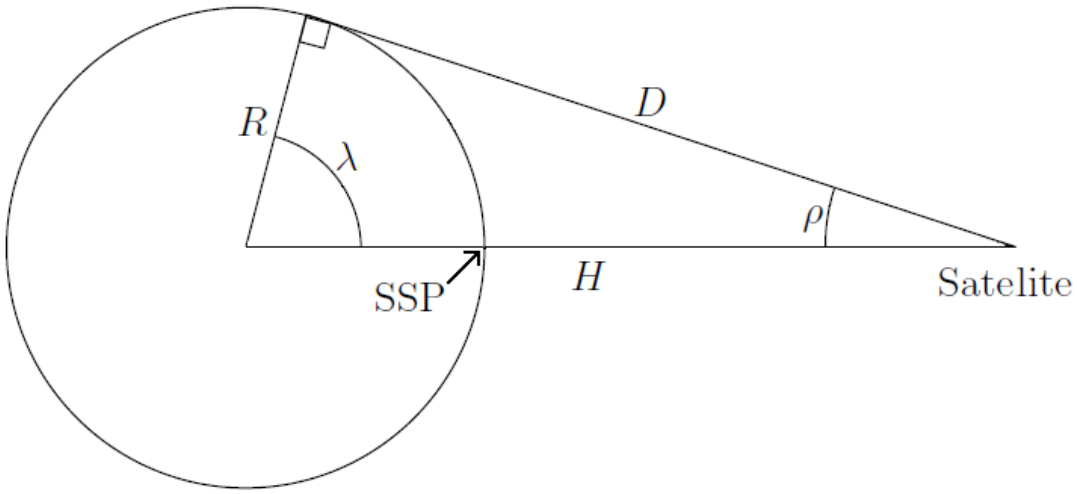


Figura 21 - Geometria do cálculo do raio da IAA.

A Figura 21 nos mostra um corte equatorial na esfera terrestre, onde R é o raio da Terra, H é a distância do satélite ao centro da Terra e λ é o máximo ângulo de visão do satélite visto do centro da Terra. Podemos descobrir o valor de λ utilizando a fórmula da Figura 22.

```
anguloRad = pi/2 - asin(raioTerra/rSat);
```

Figura 22 - Fórmula do ângulo máximo de visão do satélite.

Na fórmula da Figura 22, "anguloRad" é o λ da Figura 21, "asin(X)" é a função Matlab que retorna o arco-seno de X, "raioTerra" é o raio da Terra e "rSat" é a distância do satélite ao centro da Terra.

Observando a Figura 23 e imaginando o círculo como uma esfera, foi definido a partir do valor do ângulo λ que os pares de longitude e latitude sob a linha vermelha delimitam a área visível do satélite. São então escolhidos 256 pares sob essa linha e estes são desenhados sobre o mapa 2D.

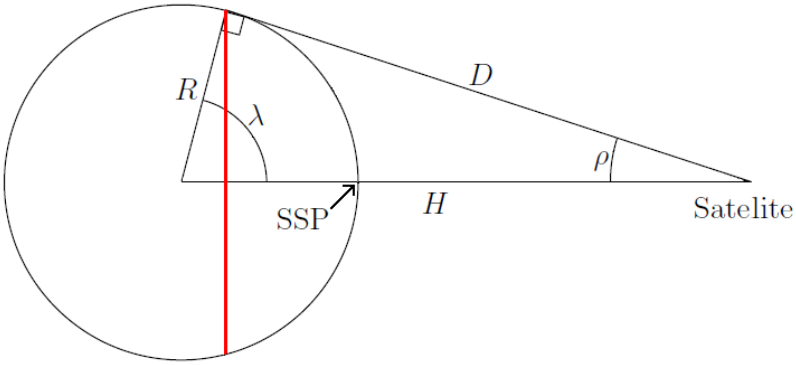


Figura 23 - Círculo que delimita a IAA.

A saída gráfica da função, além da IAA, também mostra a atual posição do SSP do satélite na data e o caminho percorrido por este ao longo das próximas 24 horas. A Figura 24 mostra a IAA delimitada em vermelho, a posição do SSP do satélite com uma estrela verde e, em amarelo, o caminho percorrido pelo SSP nas próximas 24 horas.

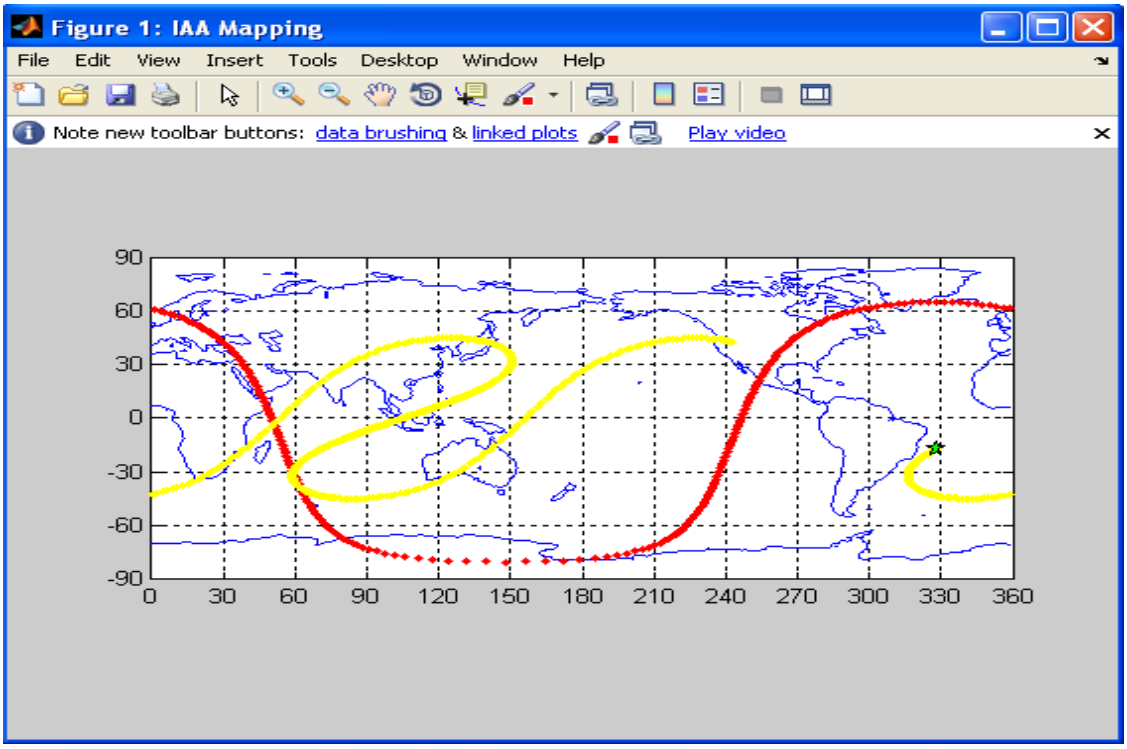


Figura 24 - IAA *Mapping*.

Na Figura 24, a área delimitada pela série de pontos vermelhos e que contém o SSP do satélite (estrela verde) é a região observável pelo satélite no dia e hora passados por parâmetro na chamada da função.

5.5 *Attitude_Estimation*

Esta é a quinta e última função principal criada nesse Trabalho de Conclusão de Curso. Seu objetivo é dizer o quanto que os atuadores do satélite devem girá-lo em cada um dos seus três eixos de liberdade, para que um dos eixos (eixo X) aponte para o alvo na superfície da Terra.

Os parâmetros que devem ser passados para essa função são os mesmo da função *View_IAA*, porém com a adição de mais dois, a longitude e latitude do alvo que se deseja apontar o satélite. A Figura 25 mostra como devem ser passados os dados.

```
>> Attitude_Estimation(KE,ano,mes,dia,hora,min,sec,Long,Lat)
```

Figura 25 - Chamada da função *Attitude_Estimation.*

Como visto no capítulo anterior, o satélite tem uma região observável limitada por sua posição e altitude, portanto o primeiro passo é descobrir qual é a região coberta pela IAA. Isso é feito pelo mesmo processo da função *View_IAA*. De posse dessa informação, verificamos se a longitude e a latitude informadas nos parâmetros da função estão contidas dentro dessa região.

Podemos fazê-lo calculando a diferença angular entre o SSP do satélite e a localização do alvo. Se essa diferença for menor do que a diferença angular entre o SSP do satélite e a máxima longitude observável, então o ponto está dentro da região visível pelo satélite.

Caso a localização na superfície da Terra esteja fora da região observável pelo satélite naquele momento, não faz sentido calcularmos a atitude necessária para apontar para aquele ponto, então, neste caso, a função é abortada e apenas a primeira saída gráfica é mostrada.

A primeira saída gráfica da função é muito semelhante à saída da função *View_IAA*, o que as diferencia é que, além da posição do SSP, existe uma segunda marcação em estrela, dessa vez em cor vermelha, mostrando a localização do alvo para o qual se deseja apontar o satélite. Caso o alvo esteja fora do IAA, essa será a única saída da função. A Figura 26 mostra a primeira saída gráfica da função *Attitude_Estimation* com um alvo visível para o satélite.

A Figura 26 demonstra o resultado da função *Attitude_Estimation* para uma órbita com 45 graus de inclinação, 33720 km de Semi-Eixo Maior, excentricidade 0.6, *Epoch Year* 2009, *Epoch Day* 314 (10 de novembro), duas horas após o *Epoch* e tendo como alvo o par longitude 300, latitude 30.

Caso o alvo esteja dentro da IAA, a função *Attitude_Estimation* fornece quatro saídas no console do Matlab, bem como mais uma saída gráfica. As três primeiras saídas no console informam em quantos graus o satélite deve girar para apontar o seu eixo X para o alvo. Primeiramente mostra quantos graus no eixo X, seguido de quantos graus no eixo Z e, por fim, mostra o mesmo para o eixo Y. A ordem em que devem ser executadas as rotações é de extrema importância, uma vez que alterada essa ordem a posição final pode não ser a desejada. A Figura 27 mostra a segunda saída gráfica para a mesma entrada da Figura 26.

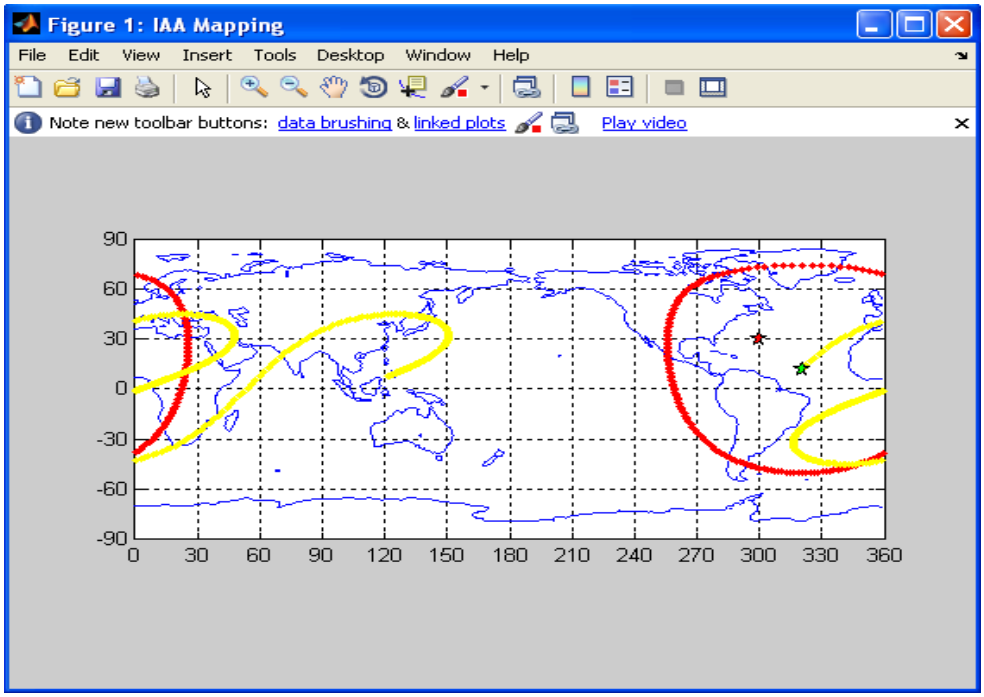


Figura 26 - Primeira saída gráfica da função *Attitude_Estimation*.

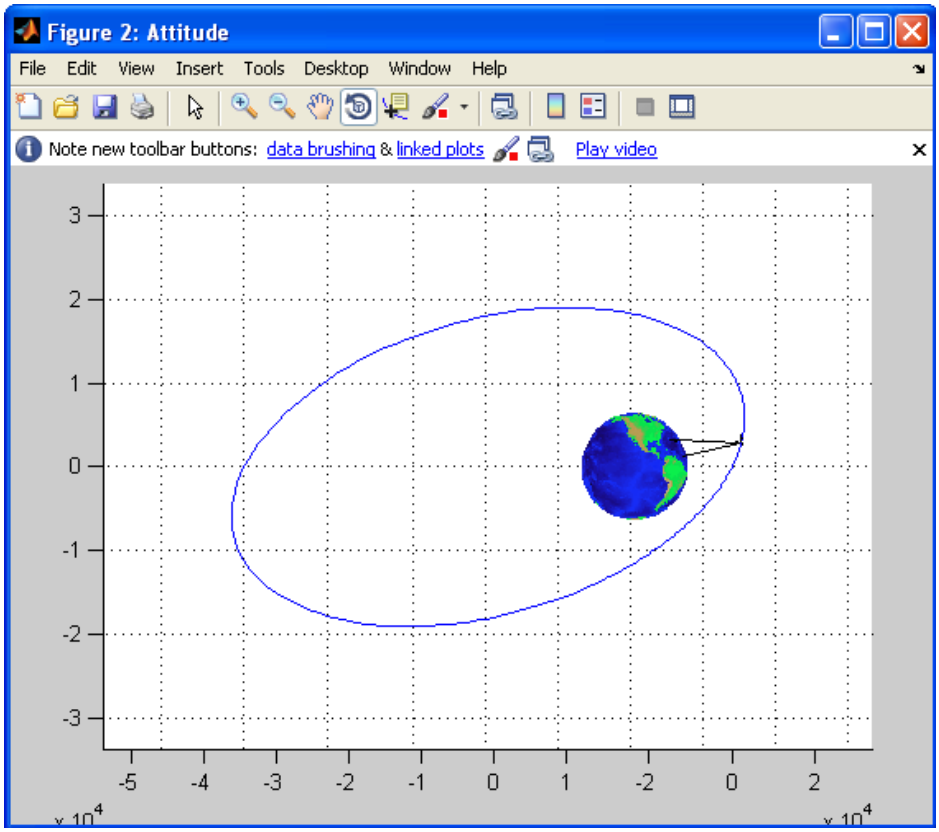


Figura 27 - Segunda saída gráfica da função *Attitude_Estimation*.

A segunda saída gráfica serve para mostrar o resultado do direcionamento dos eixos do satélite após as rotações calculadas pela função. Observando a Figura 27 vemos a órbita, a Terra e duas linhas ligando o satélite à superfície terrestre. Uma dessas linhas é o vetor que aponta diretamente para o SSP do satélite, enquanto que a outra é o vetor que aponta para o alvo requerido.

Em um primeiro momento é apenas isso o que se pode observar, porém, ao aproximarmos a imagem na posição atual do satélite, é possível visualizar três outros vetores. Estes são os vetores que representam o *Body Frame* do satélite e, segundo foi explicado anteriormente, o eixo X do satélite deve apontar diretamente para o alvo. Observando a Figura 28, podemos ver o vetor do eixo X do *Body Frame* alinhado com o vetor que aponta para o alvo.

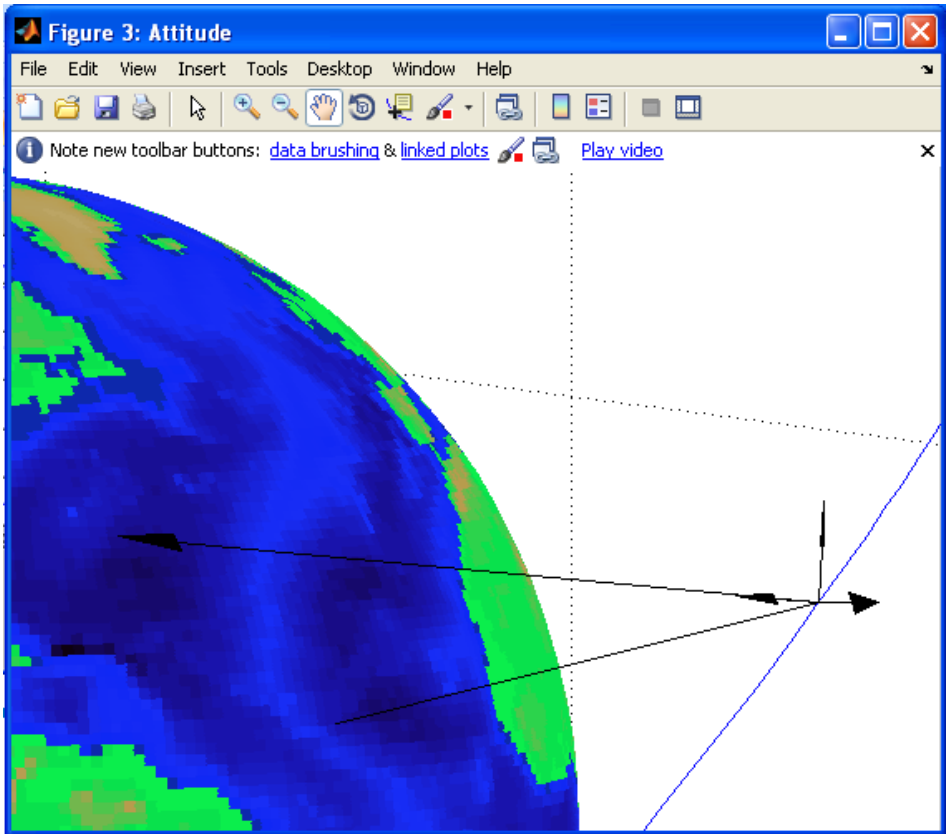


Figura 28 - Vista aproximada da segunda saída gráfica da função *Attitude_Estimation*.

A quarta saída no console Matlab, que serve para confirmar o sucesso da operação, é o resultado da diferença angular entre o vetor que aponta para o alvo e o vetor que representa o eixo X do *Body Frame* em graus.

Vale salientar que a orientação dos eixos do *Body Frame* é feita apenas utilizando as rotações calculadas pelo algoritmo. Essas rotações são feitas considerando os eixos do satélite previamente alinhados com o Orbital Frame.

6. Resultados Obtidos

Este capítulo descreve os resultados obtidos pelas funções desenvolvidas nesse trabalho. Como parâmetros para comparação, foram utilizadas as saídas de dois programas, o Celestia e o Orbitron, sendo estes referências em suas áreas de utilização. O capítulo está dividido em três seções: A primeira seção mostra o funcionamento da função *View_Orbit*, comparando sua saída a órbitas observadas no Celestia. Na segunda seção é comparada a saída da função *View_IAA* com o programa Orbitron, lembrando que a função *View_IAA* também mostra a saída da função *View_SSP*, portanto, são validadas ambas as funções. A última seção explica como foi feita a validação da função *Attitude_Estimation*, uma vez que para essa não foi encontrado nenhum programa que possa ser utilizado como referência.

6.1 Validação utilizando Celestia

Nessa seção é verificado o correto funcionamento das saídas que expõem as órbitas em 3D. Como todas as saídas 3D utilizam o mesmo procedimento de plotagem, será mostrada apenas a saída da função *View_Orbit*. Como entradas para a função foram criados os TLE's vistos na Figura 29, a estrutura de um TLE foi detalhada na seção 4.5.

```
Orbita1
1 99999U 06001A   09314.00000000 -.00000000  00000-0  00000+0 0  0015
2 99999 00.0000 000.0000 0000000 000.0000 000.0000   1.40273000  0018

Orbita2
1 99999U 06001A   09314.00000000 -.00000000  00000-0  00000+0 0  0015
2 99999 00.0000 000.0000 7000000 000.0000 000.0000   1.40273000  0018

Orbita3
1 99999U 06001A   09314.00000000 -.00000000  00000-0  00000+0 0  0015
2 99999 45.0000 090.0000 7000000 000.0000 000.0000   1.40273000  0018

Orbita4
1 99999U 06001A   09314.00000000 -.00000000  00000-0  00000+0 0  0015
2 99999 45.0000 090.0000 7000000 090.0000 000.0000   1.40273000  0018
```

Figura 29 - TLE's usados como entrada para os testes.

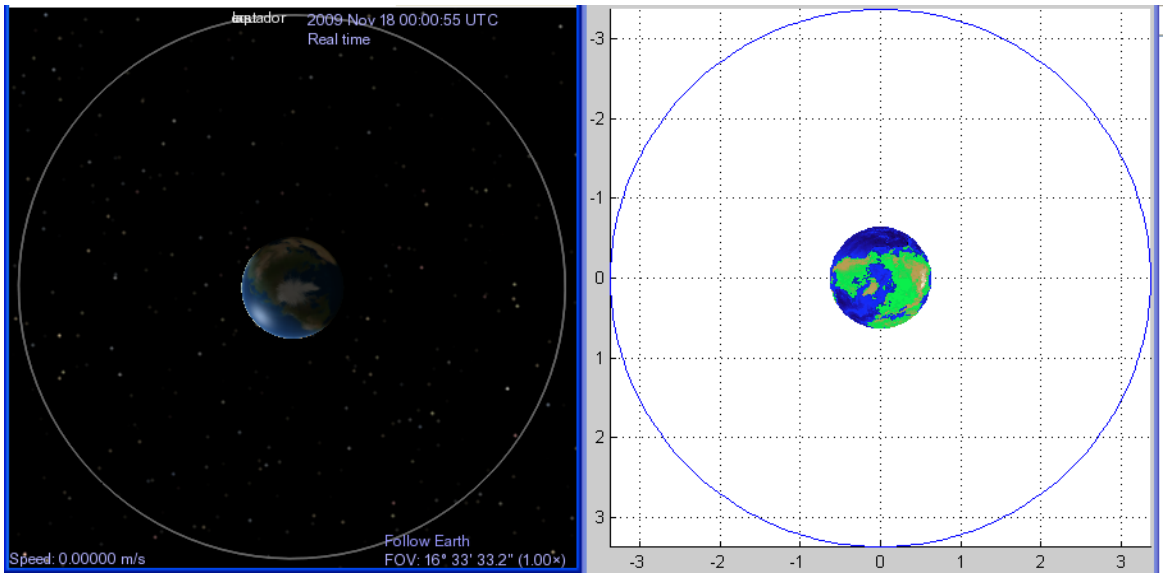
Esses TLE's variam os seus parâmetros gradativamente. O primeiro possui todos os valores angulares em zero e *Mean Motion* 1.40273. O segundo agrega uma Excentricidade de 0.7 à órbita, em seguida, se altera a RAAN para 90 graus e a inclinação para 45 graus. Por fim, o *angle of perigee* é alterado para 90 graus.

O Celestia utiliza arquivos em um formato diferente ao do TLE, porém, os valores necessários para definir o posicionamento das órbitas estão definidos de maneira bem intuitiva, na Figura 30 temos um exemplo do formato do arquivo de entrada para o Celestia, onde: SemiMajorAxis é o Semi-Eixo Maior, e no caso, 33720 km equivale a um *Mean Motion* de 1.40273; *Eccentricity* é a excentricidade; *Inclination* é a inclinação; AscendingNode é a RAAN e ArgOfPericenter é o *Argument Of Perigee*. Os valores dessas variáveis foram alterados para ficarem idênticas aos valores nos respectivos TLE's testados.

```
"last" "Sol/Earth"
{
        Class "spacecraft"
        Mesh "last.cmod"
        Radius 0.007
        Beginning          2448007
        # Ending 2010?

        EllipticalOrbit {
                Period           1
                SemiMajorAxis    33720
                Eccentricity     0.6
                Inclination       45.0
                AscendingNode     00.0000
                ArgOfPericenter   00.0000
                MeanAnomaly      000.0000
                Epoch            2452028.18381755
        }

        Albedo        0.10
}
```

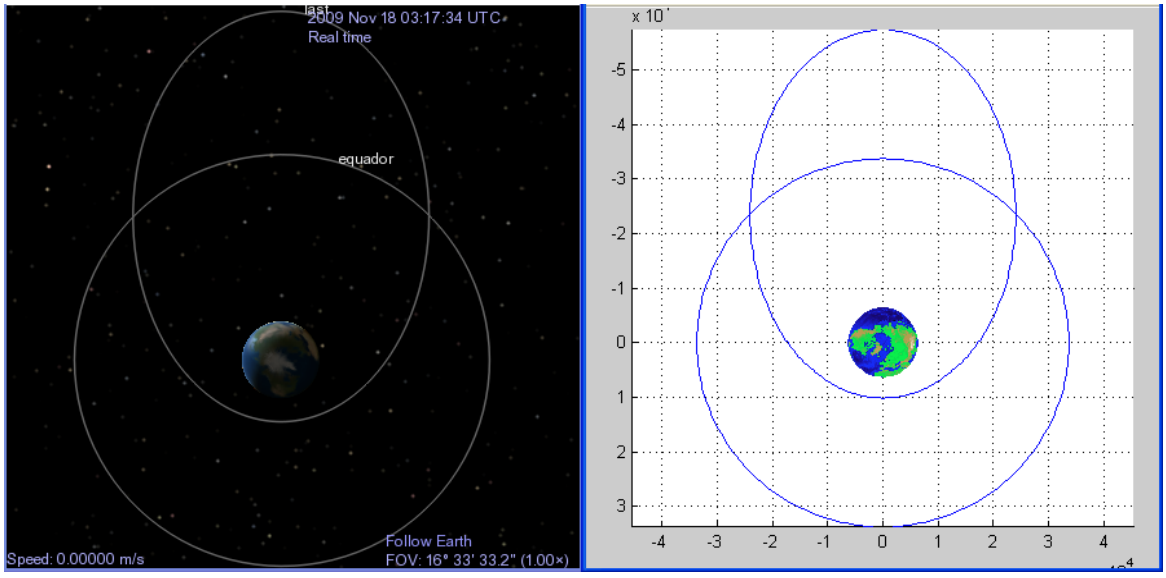
Figura 30 - Formato do arquivo de entrada do Celestia.

As figuras de 31 a 34 mostram saídas da função *View_Orbit* para cada um dos TLE's da Figura 29. A parte direita das imagens contém a saída da função desenvolvida enquanto a parte esquerda possui a saída de referência fornecida pelo Celestia. Para fins de visualização, em ambas as imagens de cada Figura podem ser vistas também órbitas equatoriais circulares de raio 33720 km.



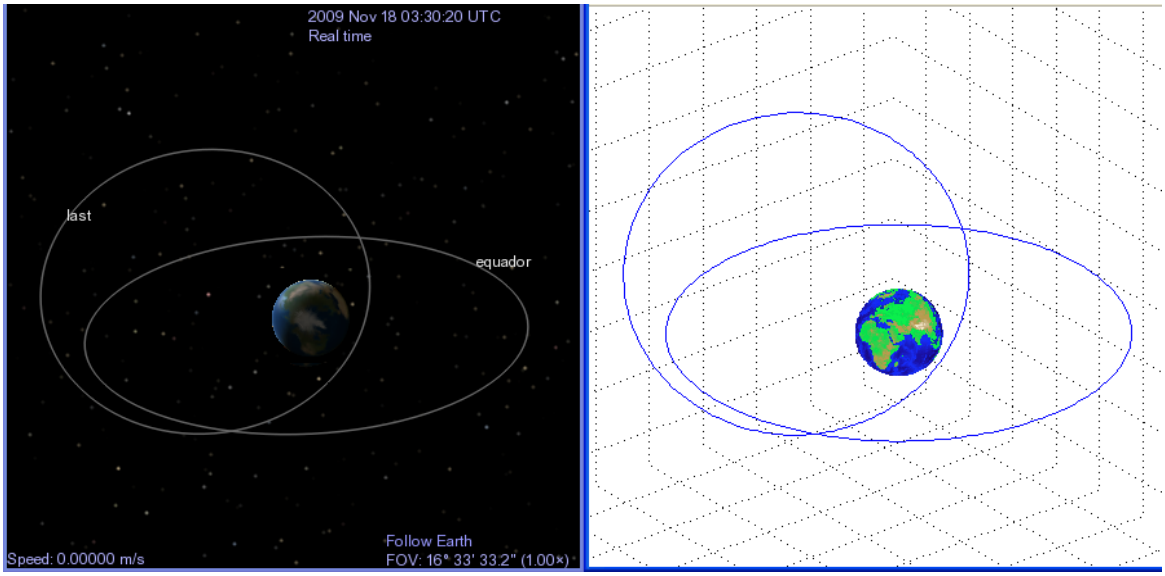
(a) **(b)**

Figura 31 - Saídas para a primeira órbita. (a) Celestia; (b) *View_Orbit*.



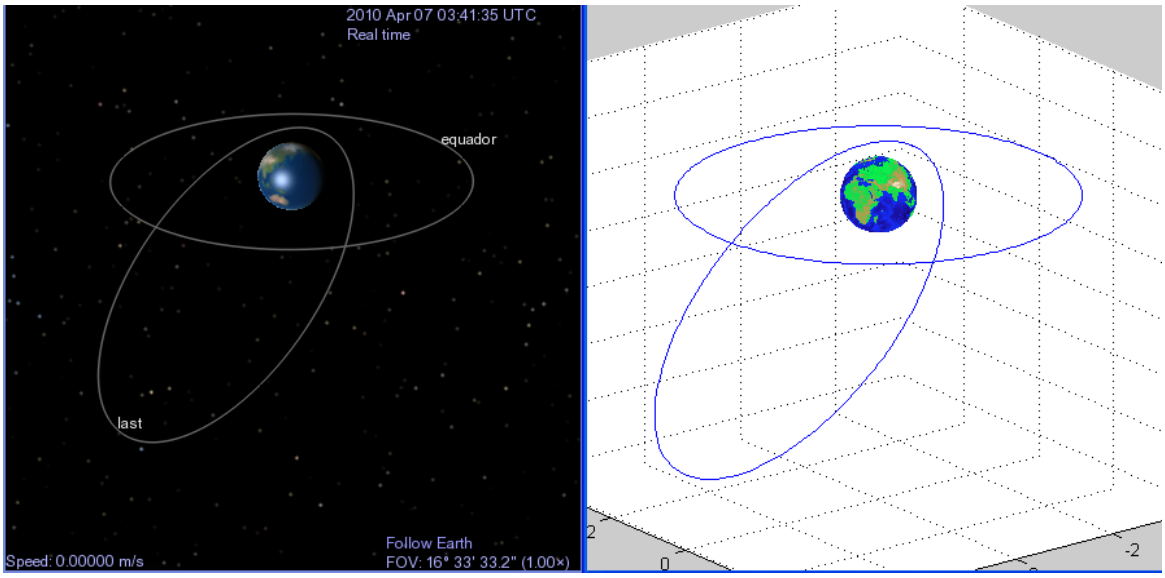
(a) **(b)**

Figura 32 - Saídas para a segunda órbita. (a) Celestia; (b) *View_Orbit*.



(a) **(b)**

Figura 33 - Saídas para a terceira órbita. (a) Celestia; (b) *View_Orbit*.



(a) **(b)**

Figura 34 - Saídas para a quarta órbita. (a) Celestia; (b) *View_Orbit*.

Vale lembrar que nesta etapa a orientação da Terra em relação a órbita não é relevante, pois a função *View_Orbit* independe de uma data, ao contrário da a orientação da Terra.

Outro fato importante é que tanto o Celestia quanto a função View_Orbit permitem livre rotação no espaço ao redor da imagem, portanto houve a necessidade de manualmente posicionar o ângulo de visão, o que pode ter gerado pequenas diferenças nas imagens.

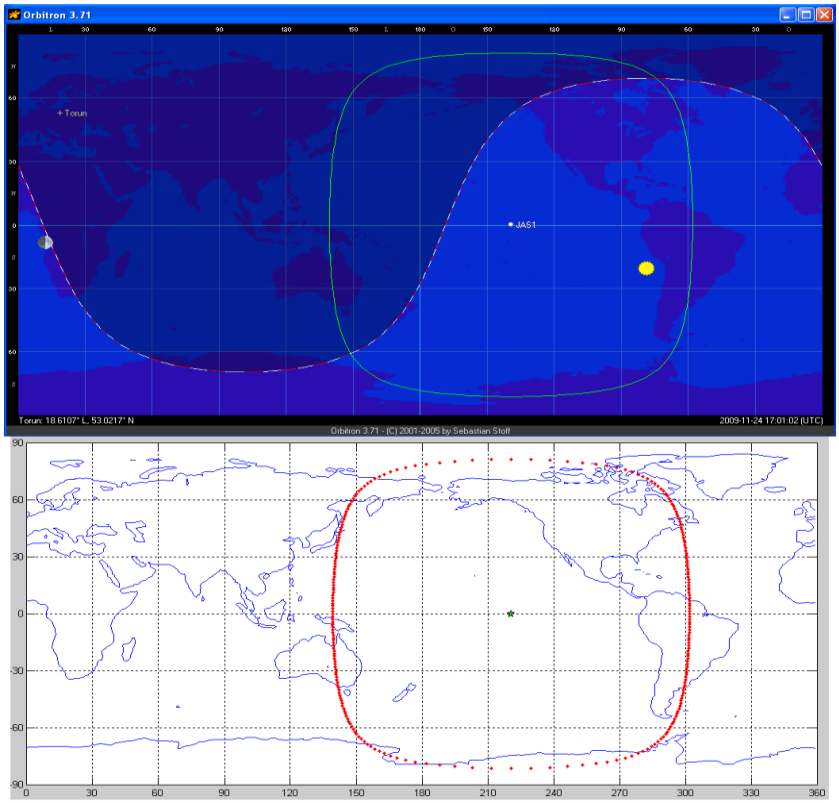
6.2 Validação utilizando o Orbitron

Nessa seção pode-se verificar o correto funcionamento das funções que calculam o SSP e a IAA. Para isso, foi utilizado o programa de rastreio de satélites Orbitron, que assim como as funções desenvolvidas, utiliza como entrada arquivos TLE válidos. Os TLE's utilizados nestes testes são mostrados na Figura 35. Além dos TLE's, a função *View_IAA* necessita de uma data para calcular os dados, durante os primeiros TLE's a data é mantida constante, no caso, 24 de Novembro de 2009 à 0h0min0seg. Por fim, alteramos a data no último TLE para 27 de Novembro de 2009 às 22h16min11seg.

```
Test1
1 99999U 06001A   09314.00000000 -.00000000  00000-0  00000+0 0  0015
2 99999  00.0000 000.0000 0000000 060.0000 210.0000  1.00273000  0018

Test2
1 99999U 06001A   09314.00000000 -.00000000  00000-0  00000+0 0  0015
2 99999  45.0000 000.0000 7000000 060.0000 210.0000  1.00273000  0018

Test3
1 99999U 06001A   09314.00000000 -.00000000  00000-0  00000+0 0  0015
2 99999  45.0000 090.0000 4000000 000.0000 210.0000  1.00273000  0018

Test4
1 99999U 06001A   09314.00000000 -.00000000  00000-0  00000+0 0  0015
2 99999  45.0000 090.0000 4000000 000.0000 210.0000  1.70273000  0018
```

Figura 35 - TLE's para teste com o Orbitron.

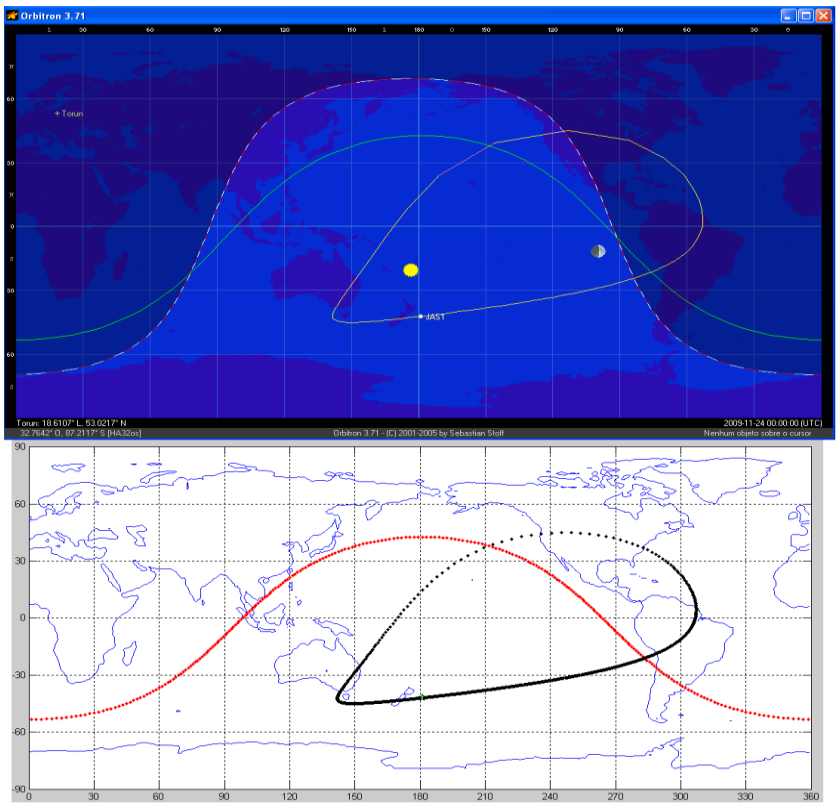
As figuras de 36 a 40 mostram as saídas do programa Orbitron na sua metade superior e as saídas da função *View_IAA* na metade inferior. As pequenas diferenças que podem ser observadas são devido às deformações decorrentes das diferentes resoluções entre os programas. Nas imagens do Orbitron, as linhas verdes demarcam a IAA e as linhas amarelas o rastro do SSP, já nas imagens da função *View_IAA*, as linhas vermelhas demarcam a IAA e as linhas pretas o rastro do SSP.



(a)

(b)

Figura 36 - Órbita geoestacionária Test1. (a) Orbitron; (b) *View_IAA*.



(a)

(b)

Figura 37 - Órbita geossíncrona Test2. (a) Orbitron; (b) *View_IAA*.

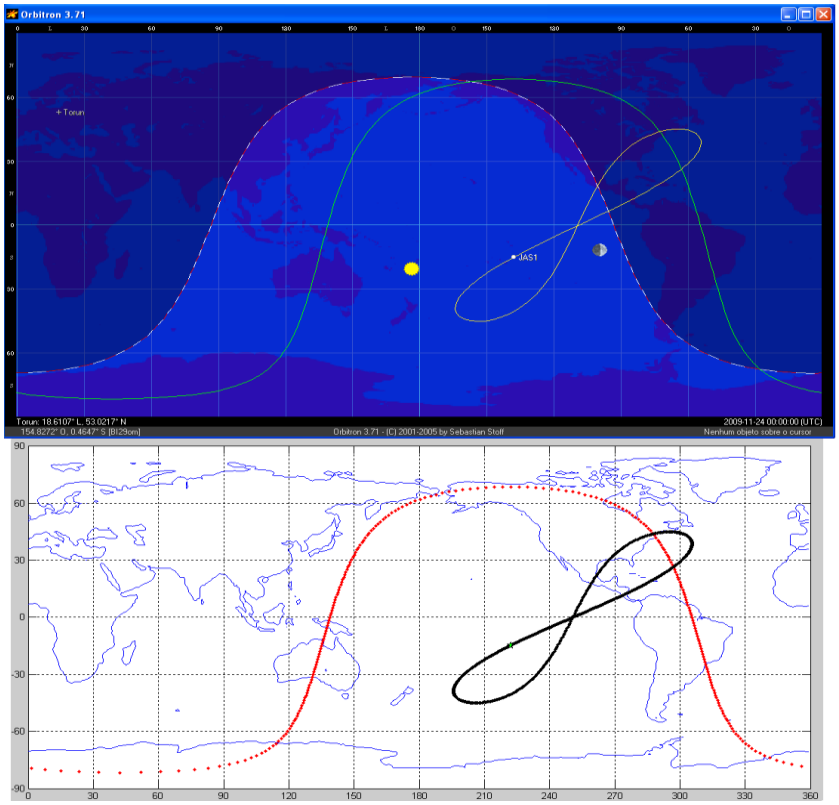


Figura 38 - Órbita geossíncrona Test3. (a) Orbitron; (b) *View_IAA*.

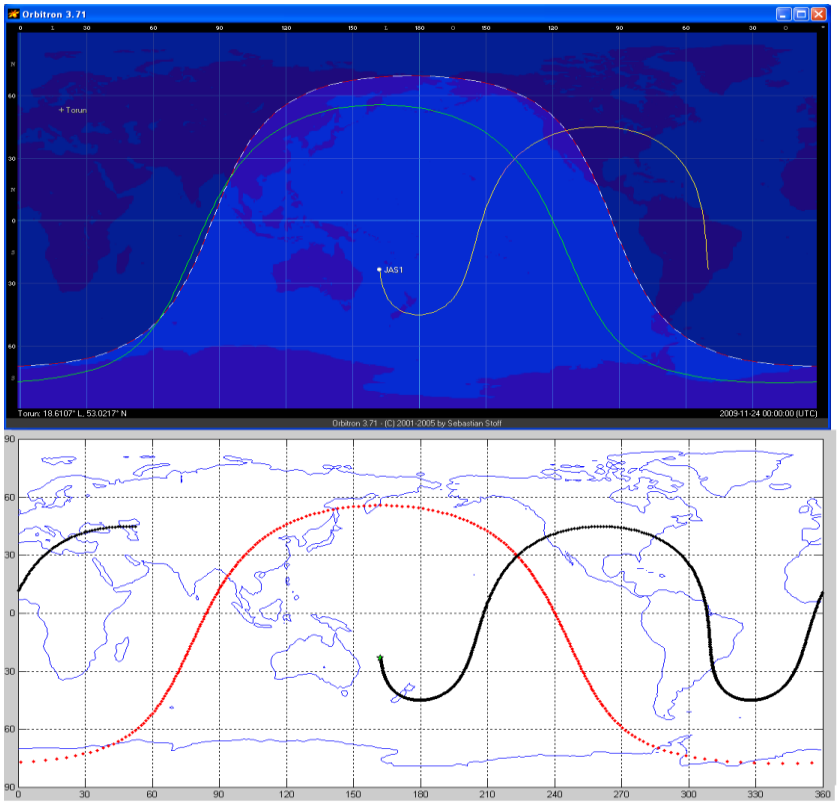
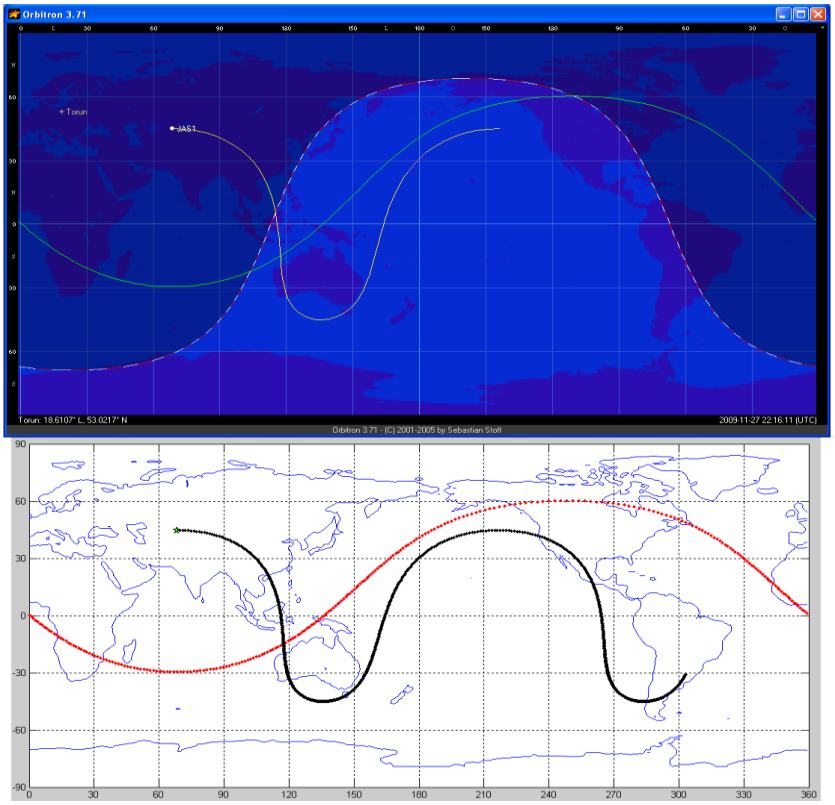


Figura 39 - Órbita Test4 no dia 24 de novembro. (a) Orbitron; (b) *View_IAA*.



(a)

(b)

Figura 40 - Órbita Test4 no dia 27 de novembro. (a) Orbitron; (b) *View_IAA*.

Os testes mostrados nas duas últimas seções demonstram o correto funcionamento das funções que definem o posicionamento do satélite no espaço e, também, a posição relativa do satélite sobre a superfície da Terra.

Os testes expostos até aqui são apenas uma parcela do grupo total de casos de teste [31]. Para validação completa das funções foram executados primeiramente testes variando cada parâmetro orbital individualmente, feito isso, foram testados uma série de TLE's escolhidos de forma aleatória, esses TLE's variavam de um a todos os parâmetros orbitais. Nos testes individuais, para cada parâmetro que pode variar de 0° a 360°, foram escolhidos os valores limites dos quadrantes e seus pontos médios (0°, 45°, 90°, 135°, 180°, 225°, 270°, 315° e 360°) e para os que variam de 0° a 180°, apenas os dois primeiros quadrantes (0°, 45°, 90°, 135°, 180°).

6.3 Validação da função *Attitude_Estimation*

A função *Attitude_Estimation* necessitou de um tratamento especial, não só por ser a principal função deste trabalho de conclusão, mas também porque não foi encontrado nenhum programa que pudesse ser utilizado pelo grupo para validar sua saída.

Para resolver este problema, o grupo utilizou o fato de que a posição do satélite em relação ao ECI e em relação à superfície da Terra já estava validada por meio das ferramentas Celestia e Orbitron, como vimos nas seções 5.1 e 5.2. A posição do satélite em relação ao ECI pode ser então representada como um vetor no espaço, da mesma forma como a posição do alvo na superfície da Terra. Logo, ao fazer a subtração desses dois vetores, o resultado obtido é um vetor que inicia na origem do ECI e tem a mesma direção do vetor que aponta da posição do satélite até o alvo na superfície da Terra[27], a esse vetor é dado o nome VetorAlvo.

Como visto na seção 4.5, os eixos do *Body Frame* estão inicialmente alinhados com os Eixos do *Orbital Frame*, ou seja, o eixo de interesse (Eixo-X) aponta da posição do satélite diretamente para o centro da Terra. Feitas todas as rotações calculadas pela função sobre os eixos do *Body Frame*, subtraímos o vetor que aponta para a posição do satélite do outro que representa o Eixo-X do *Body Frame e*, com isso, teremos um vetor que inicia na origem do ECI e tem a mesma direção do vetor que representa o eixo-X do satélite. É dado a esse vetor o nome VetorBodyX.

De posse desses dois vetores, pode-se calcular o quanto o eixo X do satélite está defasado do alvo, para isso utiliza-se a equação da Figura 41.

```
ERRO = acosd( dot(Alvo,Body-X) / ( norm(Alvo)*norm(Body-X) ) )
```

Figura 41 - Fórmula para cálculo do Erro.

Na equação da Figura 41, acosd(X) é a função que retorna o arco-cosseno de X em graus, dot(X,Y) retorna o produto escalar entre os dois vetores X e Y, norm(X) é a função que retorna a norma do vetor X, Alvo é o VetorAlvo e Body-X é o VetorBodyX. Como o objetivo

da função é garantir que o satélite esteja apontando diretamente para o alvo, o resultado dessa equação deve ser o mais próximo possível de zero, qualquer valor acima de 0,1 graus de diferença já acarretam grandes erros de direcionamento, principalmente para satélites com órbitas altas (acima de 35786 km) [32].

Os testes foram realizados com 10 TLE's escolhidos de forma aleatória, onde para cada um foram feitos cinco testes. O primeiro com o alvo diretamente sob o SSP do satélite, seguido de um alvo em cada um dos quatro quadrantes vistos pelo satélite. A ordem de grandeza dos erros encontrados foi de 10^{-3} graus, esses erros são decorrentes do acúmulo de pequenos erros de cada rotação, quanto menor o número de rotações necessárias, menor o erro encontrado. Um erro de ordem 10^{-3} pode ser considerado nulo, uma vez que mesmo atuadores de alta precisão usualmente trabalham com ângulos mínimos de 0,025 graus [33].

Devemos lembrar que o erro referente ao fato de a terra não ser uma esfera perfeita, não é levado em consideração na análise exposta no parágrafo anterior. No presente trabalho esta simplificação foi necessária para manter a complexidade da geometria envolvida em um nível aceitável.

7. Conclusão e trabalhos futuros

Para o desenvolvimento do presente trabalho de conclusão de curso, foi necessário ao grupo um estudo aprofundado de temas complexos relativos não apenas à Engenharia, mas também à Matemática e à Física, temas estes abordados de forma mais superficial no curso de Engenharia de Computação. Por esse motivo, cerca de 60% do tempo disponível destes quatro meses de trabalho foi utilizado no estudo teórico necessário para a assimilar esses conhecimentos.

Apesar de o tempo necessário de estudo teórico ter sido bem mais longo do que o planejado, o grupo conseguiu desenvolver e validar com êxito todas as ferramentas propostas. Estas se mostraram eficientes e de fácil utilização. As ferramentas visuais possibilitam ao usuário um rápido entendimento dos complexos parâmetros orbitais, enquanto que os dados calculados se mostraram precisos a ponto de serem aplicados de maneira prática.

À medida que o assunto foi sendo compreendido, percebeu-se que a etapa de integração com o hardware e fechamento da malha de controle não poderia ser concluída, pois os dados provenientes do periférico que possuíamos [19] não seriam suficientes para a determinação de atitude do satélite, sendo necessária a inclusão de mais hardware, ao qual não temos acesso na PUCRS.

Como trabalhos futuros, podemos citar o desenvolvimento de uma plataforma para determinação de atitude, utilizando inclinômetros, magnetômetros e/ou sensores solares. Assim como a plataforma, um algoritmo de determinação de atitude também é de grande interesse.

Outro sistema que poderia ser desenvolvido é o de determinação de órbita. O presente trabalho pressupõe que a órbita é conhecida e descrita por um TLE, porém um sistema de determinação de órbita fornece dados em tempo real, o que vem a ser extremamente útil, uma vez que as órbitas podem variar mais rapidamente do que a atualização do TLE.

Referências Bibliográficas

[1] National Aeronautics and Space Administration. **The Hubble Space Telescope.** Disponível em: http://hubble.nasa.gov/. Acesso em: 07 set 2009.

[2] Intelsat Resources. **Satellite Keplerian Data**. Disponível em: http://www.intelsat.com/resources/satellitedata/keplerian.asp. Acesso em: 07 set 2009.

[3] Chris Hall. **Spacecraft Attitude Dynamics and Control - Lecture Notes**. Virginia Tech, 2003, 122p.

[4] Futron. **Space Transportation Costs: Trends Price Per Pounds to Orbit.** Disponível em: http://www.futron.com/pdf/resource_center/white_papers/FutronLaunchCostWP.pdf Acesso em: 02 set 2009.

[5] CCSDS, **TC Space Data Link Protocol. Blue Book. Issue 1**. Washington, DC, USA, September 2003, CCSDS Recommended Standard.

[6] CCSDS, **Communications Operation Procedure-1. Blue Book. Issue 1**. Washington, DC, USA, September 2003, CCSDS Recommended Standard.

[7] CCSDS, **Space Packet Protocol. Blue Book. Issue 1**. Washington, DC, USA, September 2003, CCSDS Recommended Standard.

[8] CCSDS, **Telecommand Summary of Concept and Rationale. Green Book. Issue 6**. Washington, DC, USA, January 1987, CCSDS Recommended Standard.

[9] INPE, **Multi-mission Platform Attitude Control and Data Handling (ACDH) Subsystem Specification**, Documento de Especificação A822700-SPC-01/06, INPE, São José dos Campos, 129p, Agosto 2001.

[10] ALMEIDA G. M., **Códigos Corretores de Erros em Hardware para Sistemas de Telecomando e Telemetria em Aplicações Espaciais**, Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Ciência da Computação, FACIN, PUCRS, Porto Alegre, Março 2007, 97 p.

[11] I. S. Reed and G. Solomon, **Polynomial codes over certain finite fields**, SIAM Journal of Applied Mathematics. JSTOR. Society for Industrial and Applied Mathematics. Philadelphia, PA, vol. 8, no. 2, pp. 300-304, June 1960.

[12] A. Hocquenghem, **Codes correcteurs d'erreurs**, Chiffres (Paris), September 1959.

[13] R. C. Bose and D. K. Ray-Chaudhuri, **On a class of error correcting binary group codes, Information and Control**. MIT Computer Science & AI Laboratory. Elsevier. Atlanta, GA, vol. 3, no. 1, pp. 68-79, March 1960.

[14] REIF C. E. C.; AZEVEDO L. R.; BAI T. X., **Sistema de Telecomando CCSDS em FPGAs**, Relatório Final de Trabalho de Conclusão de Curso de Engenharia de Computação, FACIN/FENG, PUCRS, Porto Alegre, Novembro 2007, 77 p.

[15] MADEIRA A. G., **Verificação da Implementação do Protocolo CCSDS do PUC#SAT usando PROMELA/SPIN**, Relatório Final de Trabalho de Conclusão de Curso de Engenharia de Computação, FACIN/FENG, PUCRS, Porto Alegre, Julho 2008, 57 p.

[16] Gerard J. Holzmann, **The Spin Model Checker: Primer and Reference Manual**. Addison Wesley, September 2003.

[17] Gerard J. Holzmann, **Design And Validation Of Computer Protocols**. Prentice Hall, November 1990.

[18] **On-The-Fly LTL Model Checking With Spin**. <http://spinroot.com/spin/whatispin.html>. Acesso em: 07 dez 2009.

[19] FERREIRA C. G.; SILVA F. A.; VILLA P. R. C., **Concepção do On-Board Data Handling com Sensor Inercial para Aplicações Espaciais**, Relatório Final de Trabalho de Conclusão de Curso de Engenharia de Computação, FACIN/FENG, PUCRS, Porto Alegre, Julho 2009, 74 p.

[20] Emadi, Ali. **Vehicular electric power systems : land, sea, air, and space vehicles**, Extraído de http://www.engnetbase.com/ejournals/books/book_km.asp?id=1527 em 29 de Julho de 2009.

[21]Wikipédia. **MATLAB History**. Disponível em: http://en.wikipedia.org/wiki/MATLAB. Acesso em: 09 set 2009.

[22] MathWorks. **MATLAB**. Disponível em: http://www.mathworks.com/products/matlab/. Acesso em: 07 set 2009.

[23] Celestia. **Celestia Documentation**. Disponível em: http://www.shatters.net/celestia/. Acesso em: 09 set 2009.

[24] Orbitron. **Orbitron Details**. Disponível em: http://www.stoff.pl/. Acesso em: 07 set 2009.

[25] Lay, David C. **Álgebra Linear e Suas Aplicações**. 2ª Edição. Rio de Janeiro: Editora LTC, 1999. 504 p.

[26] Satellite Technology. **Keplerian Elements**. Disponível em: http://sattechnology.wordpress.com/. Acesso em: 07 ago 2009.

[27] ANTON, Howard. **Cálculo: Um novo horizonte**. 6ª Edição. Porto Alegre: Editora Bookman, 2006. Volume 2.

[28] PY4ZBZ. **Satélite Diferente**. Disponível em: http://www.qsl.net/py4zbz/satdif.htm. Acesso em: 07 set 2009.

[29] Enrico Russo, Eberhard Gill**. A GNSS-based Access Scheme for Mobile Satellite Communications**. Disponível em: http://www.weblab.dlr.de/rbrt/pdf/KaBCC_05088.pdf. Acesso em: 07 set 2009.

[30] Kelson T. S., **Satellite Times**. Disponível em: http://celestrak.com/columns/v04n03/. Acesso em: 07 dez 2009.

[31]Pezzè, Mauro.**Teste e análise de software : processo, princípios e técnicas**. Porto Alegre, Bookman, 2008. 512 p.

[32]Roger R. Bate, Donald D. Mueller, and Jerry E. White. **Fundamentals of Astrodynamics**. Dover Publications Inc., New York, 1971. 455p.

[33]Sinclair, Doug; Grant, C. Cordell; Zee, Robert E., **Enabling Reaction Wheel Technology for High Performance Nanosatellite Attitude Control**, Space Flight Laboratory, Toronto, 2007. Disponível em: http://www.sinclairinterplanetary.com/SSC07-X-3.pdf. Acesso em: 09 dez 2009.