MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÕES
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

CMC-233-4 – Trabalho Final

Descrição

Criar um programa que simule o modo de aquisição do apontamento para o Sol de um satélite. O conjunto de atuadores disponíveis são **3 bobinas magnéticas** e **3 rodas de reação**. Como sensores, estão disponíveis **um conjunto de sensores solares**, **um giroscópio** e **um magnetômetro**.

Objetivo

O objetivo deste trabalho é exercitar os conceitos apresentados na disciplina CMC-233-4.

Detalhamento

A seguir, é apresentado o detalhamento do simulador a ser desenvolvido.

Sistema Inercial

O sistema inercial (**ECI**) a ser utilizado no simulador é o **TEME** (*true equator, mean equinox*). Todos os arquivos TLE (*two line elements*) disponibilizados pelo NORAD são baseados nesse sistema de referência. Portanto, a órbita inicial considerada a seguir também estará baseada nesse sistema de referência. Ele é definido como:

- O eixo X aponta para o equinócio Vernal médio;
- O eixo Z aponta na direção verdadeira da rotação da Terra na data ou época escolhida;
- O eixo Y completa o sistema de coordenadas dextrogiro.

A conversão entre o TEME e o sistema de referência fixo na Terra, que será necessário devido às medidas do campo magnético conforme mostrado a seguir, deve levar em consideração alguns movimentos terrestres. Para simplificar, **apenas o tempo sideral médio de Greenwich** será levado em conta. Com isso, o algoritmo que converte o ECI para o ECEF utilizado nesse trabalho pode ser visto no Apêndice. Dessa forma, estaremos utilizando o sistema de referência PEF (*Pseudo Earth Fixed*) como o sistema fixo na Terra.

Propagação Orbital

Deve-se simular a órbita do satélite com algum algoritmo de propagação orbital. Recomenda-se utilizar um propagador simples baseado nas iterações Newtonianas entre dois corpos sem considerar perturbações, ou seja, assumindo a Terra uma esfera perfeita. Esse tipo de propagador é chamado de *two-body orbit propagator* na literatura. Basicamente, considera-se que o semi-eixo maior, a inclinação, o argumento do perigeu e a ascensão reta do nodo ascendente são constantes, bastando, portanto, propagar a anomalia média.

Esboço

- Propagar a anomalia média utilizando

$$M(t) = n \cdot t\,,$$

onde $M(t)$ é a anomalia média e $n = \sqrt{\mu/a^3}$ é a velocidade angular orbital.
- Utilizar algum algoritmo numérico para resolver a equação de Kepler e obter a anomalia excêntrica $E(t)$:

$$M(t) = E(t) - e \cdot \sin\big(E(t)\big)\,,$$

onde e é a excentricidade da órbita.
- Finalmente, encontrar a anomalia verdadeira $\nu(t)$ que conclui a propagação da órbita:

$$\tan\frac{\nu(t)}{2} = \sqrt{\frac{1+e}{1-e}} \cdot \tan\frac{E(t)}{2}$$

Nota: no cálculo da anomalia verdadeira, deve-se observar o quadrante devido à ambiguidade da função atan(). Para isso, basta notar que $\nu(t)/2$ e $E(t)/2$ sempre estarão no mesmo quadrante.

Exemplo: Utilizando **Julia** e o **SatelliteToolbox.jl** para propagar uma órbita por 60s.

```
julia> using SatelliteToolbox

julia> JD = date_to_jd(2020, 6, 4, 11, 0, 0)
2.4590049583333335e6

julia> orbp = init_orbit_propagator(
           Val(:twobody),
           KeplerianElements(
           JD,                 # Época da órbita.
           7130.982e3,         # Semi-eixo maior [m].
           0.001111,           # Excentricidade.
           98.405*pi/180,      # Inclinação [rad].
           230.297*pi/180,     # RAAN [rad].
           90*pi/180,          # Arg. do perigeu [rad].
           305*pi/180          # Anom. verdadeira [rad].
       ))

julia> r,v = propagate!(orbp, 60)

julia> o = get_mean_elements(orbp)
KeplerianElements{Float64, Float64}:
           Epoch :    2.459e6 (2020-06-04T11:01:00)
 Semi-major axis : 7130.98     km
    Eccentricity :    0.001111
     Inclination :   98.405    °
            RAAN :  230.297    °
 Arg. of Perigee :   90.0      °
    True Anomaly :  308.609    °

julia> r
3-element StaticArrays.SVector{3, Float64} with indices SOneTo(3):
 -4.057224160306575e6
 -3.8689614024157017e6
  4.398910700921908e6

julia> v
3-element StaticArrays.SVector{3, Float64} with indices SOneTo(3):
 2328.5452068171385
 4141.245423283889
 5779.497949833172
```

Modelo do Campo Magnético

Conforme será visto a seguir, deve-se utilizar as bobinas magnéticas para remover o momento embarcado no conjunto de rodas de reação. Para isso, o campo magnético deverá ser simulado. Existem diversos modelos disponíveis, mas o mais comum na literatura é o *International Geomagnetic Reference Field* (**IGRF-13**). Este é o modelo que deverá ser utilizado neste trabalho.

Notas

- O pacote **SatelliteToolbox.jl** já possui uma implementação do modelo IGRF13. Para isso, veja a documentação das funções `igrf13` e `igrf13syn`.
- Uma versão em FORTRAN do modelo pode ser encontrada aqui:
  https://www.ngdc.noaa.gov/IAGA/vmod/igrf.html
  Vide a rotina chamada `igrf13syn()`.
- É possível também encontrar implementações desse modelo em outras linguagens.
- Para utilizar esse modelo, é necessário converter a posição inercial do satélite para **latitude/colatitude**, **longitude** e **altitude**. Para isso, primeiramente deve-se obter a DCM que rotaciona o sistema inercial (**ECI**) no sistema de referência fixo à Terra (**ECEF**). Para simplificar, isso deverá ser feito no algoritmo apresentado no apêndice. Na sequência, a posição do satélite no sistema **ECEF** deve ser convertida para **LLA** (latitude, longitude e altitude). Isso deve ser feito utilizando o **WGS84**. Para maiores informações, consulte o documento:
  https://microem.ru/files/2012/08/GPS.G1-X-00006.pdf
  Finalmente, a colatitude pode ser facilmente obtida da latitude utilizando:

$$colat = 90^\circ - lat$$

- A rotina `igrf13syn` retorna o vetor do campo magnético em nT representado no sistema local NED (Norte, Leste e Vertical para Baixo). **Esse vetor deve ser convertido para o sistema do corpo a fim de simular a medida do magnetômetro.** Para fazer isso, deve-se computar as seguintes transformações:

$$\text{NED} \Rightarrow \text{ECEF} \Rightarrow \text{ECI} \Rightarrow \text{Sistema do Corpo}$$

Posição do Sol

Deve-se codificar um algoritmo para obter o vetor unitário solar representado no sistema inercial. Um exemplo pode ser encontrado no **Astronomical Almanac**. Note que, para fins desse exercício, pode-se considerar que o versor solar **<u>quando representado no sistema inercial</u>** é constante durante toda a simulação.

Sensores

Considere que o conjunto de sensores são ideais, portanto:

- Os sensores solares fornecem o versor solar representado no sistema do corpo. Como simplificação, não é necessário simular o eclipse e considera-se que o campo de visão do conjunto de sensores é de 4π sr.
- O giroscópio fornece a velocidade angular do satélite representada no sistema do corpo.

Atuadores

O modelo das rodas de reação será simplificado. Considera-se que não existe nenhum tipo de atrito e que todo o torque comandado pelo sistema de controle é efetivamente aplicado ao rotor. Entretanto, **deve-se considerar que o torque que cada roda consegue aplicar é limitado e que sua velocidade de rotação também é limitada**.

O modelo das bobinas magnéticas também será simplificado. A atuação das bobinas deverá ser feita pelo seguinte algoritmo:

- Calcular o torque desejado $\hat{\boldsymbol{t}}_{mtr}$ para remoção de momento nas rodas de reação:

  $$\hat{\boldsymbol{t}}_{mtr,b} = k_{mtr} \cdot \boldsymbol{h}_b^{rw},$$

  onde k_{mtr} é um ganho **que deve ser sintonizado** e $\boldsymbol{h}_b^{rw}$ é o momento angular do conjunto de rodas de reação.
- Converter o torque $\hat{\boldsymbol{t}}_{mtr}$ para um momento de dipolo magnético $\boldsymbol{d}_b$ utilizando:

  $$\boldsymbol{d}_b = \frac{\left(\boldsymbol{b}_b \times \hat{\boldsymbol{t}}_{mtr,b}\right)}{b_{médio}^2},$$

  onde $\boldsymbol{b}_b$ é a medida dos magnetômetros e $b_{médio}$ é o valor médio do campo magnético na órbita selecionada. Note que $b_{médio} \approx 30{,}000\ nT$ para a órbita que será utilizada. **Lembre-se de verificar as unidades desse cálculo!** A unidade de $\boldsymbol{d}_b$ deverá ser A.m^2, que é obtido utilizando o S.I. em todas as variáveis.
- Assumir que cada bobina magnética fornecerá o dipolo calculado em $\boldsymbol{d}_b$ **considerando os valores de saturação** (máximo dipolo permitido pela bobina).
- Finalmente, o torque magnético aplicado ao corpo do satélite devido ao dipolo $\boldsymbol{d}_b$ gerado pelas três bobinas será:

  $$\boldsymbol{t}_{mtr,b} = \boldsymbol{d}_b \times \boldsymbol{b}_b.$$

  Esse é o valor que deverá ser utilizado na equação dinâmica do modelo.

Dinâmica da Atitude

A dinâmica de atitude do satélite deverá ser simulada calculando a representação do sistema de referência do corpo em relação ao sistema inercial. A representação e propagação da atitude poderá ser feita utilizando qualquer método que não possua singularidades.

Algoritmo de Controle da Atitude do Satélite

Deve-se equacionar uma lei de controle que, utilizando os dados dos sensores (posição do Sol e velocidade angular representadas no sistema do corpo), comande as rodas de reação de tal forma que o eixo +X do satélite seja apontado em direção ao Sol. Ou seja, as referências para o algoritmo de controle serão:

$$\hat{\boldsymbol{s}}_b = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \boldsymbol{\omega}_b = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} rad/s.$$

Parâmetros e Condições Iniciais

A tabela a seguir contém o conjunto de parâmetros e condições iniciais que devem ser utilizados na simulação. Esses valores foram baseados em estudos já realizados no INPE.

Parâmetro	Valor
Instante inicial da simulação	04/06/2022 às 11:00 GMT
Parâmetros Orbitais	
Semi-eixo maior	$7130{,}092\ km$
Excentricidade	$0{,}001111$
Inclinação	$98{,}405^\circ$
Argumento do perigeu	90°
Ascenção reta do nodo ascendente	$230{,}327^\circ$
Anomalia média inicial	305°
Condições Iniciais da Atitude do Satélite	
Apontamento inicial	A atitude inicial do sistema do corpo em relação ao sistema inercial deve ser calculada utilizando os seguintes ângulos de Euler: • *Yaw = 180,0°;* • *Pitch = 0,0°;* • *Roll = 0,0°.*
Velocidade inicial	A velocidade inicial do satélite será dada pelo seguinte vetor: $\begin{bmatrix} 0 \\ 0 \\ 0{,}024 \end{bmatrix} rad/s$
Outros Dados	
Matriz de inércia do satélite	$\begin{bmatrix} 310{,}0 & 1{,}11 & 1{,}01 \\ 1{,}11 & 360{,}0 & -0{,}35 \\ 1{,}01 & -0{,}35 & 530{,}7 \end{bmatrix} kg.m^2$
Inércia das rodas de reação	$0{,}01911\ kg.m^2$
Máximo torque das rodas de reação	$0{,}075\ Nm$
Máxima velocidade das rodas de reação	$4000\ e\ 6000\ RPM^1$
Máximo momento de dipolo magnético nas bobinas	$30\ \ A.m^2$

[1]: Devem ser apresentadas duas simulações, uma com a velocidade máxima das rodas em 4000 RPM e outra com a velocidade máxima das rodas em 6000 RPM.

Resultados Esperados

Deve-se compilar um relatório detalhado sobre a implementação do simulador. Os resultados esperados são:

- O satélite deverá ser capaz de apontar para o Sol e manter esse apontamento indefinidamente.
- As bobinas magnéticas deverão ser capazes de remover momento angular das rodas de reação. Portanto, a velocidade das rodas deverá tender a zero para tempos de simulação altos.

Duas simulações devem ser feitas: uma com a limitação da velocidade máxima das rodas em 4000 RPM e outra com a limitação da velocidade das rodas em 6000 RPM. Para cada uma dessas simulações, devem ser apresentados os seguintes gráficos:

1. O vetor unitário solar representado no sistema inercial no início da simulação;
2. A anomalia verdadeira em função do tempo;
3. O vetor solar representado no sistema do corpo;
4. O erro de apontamento para o Sol;
5. A velocidade angular do satélite representada no sistema do corpo;
6. O torque de controle aplicado às rodas de reação;
7. A velocidade das rodas de reação;
8. A norma do vetor de momento do conjunto de rodas de reação;
9. O vetor do campo magnético representado no sistema do corpo;
10. A norma do vetor do campo magnético;
11. O torque de controle aplicado às bobinas magnéticas;
12. O gráfico do dipolo gerado pelas bobinas magnéticas;
13. Os ângulos de Euler que representam a atitude do satélite com relação ao sistema inercial (ECI);
14. Os ângulos de Euler que representam a atitude do satélite com relação ao sistema fixo na Terra utilizado (ECEF).

Pode-se adicionar qualquer outro gráfico desejado que promova um melhor entendimento sobre o sistema simulado.

Junto com o relatório, deverá ser entregue em **formato eletrônico** (**.zip**) o código-fonte do simulador criado.

Informações Gerais

Os relatórios e código-fonte devem ser enviados para o e-mail ronan.arraes@inpe.br até o dia **24/05/2022**. Se o arquivo final for maior do que **10 MiB**, por favor, me procure pessoalmente para entregar o trabalho, pois existe uma possibilidade do servidor de e-mails do INPE bloquear o envio.

Todo e-mail será respondido como forma de verificação do recebimento.

Apêndice

Algoritmo de conversão entre o sistema ECI e o sistema ECEF utilizados

```julia
"""
    j2000_to_gmst(j2000_ut1::Number)

Compute the Greenwich Mean Sideral Time (GMST) \\[rad] given the instant
`j2000_ut1` in J2000.0 reference [UT1].

!!! info
    The algorithm is based in **[1]**.

# References

- **[1]** http://www.navipedia.net/index.php/CEP_to_ITRF, accessed 2015-12-01.
"""
function j2000_to_gmst(j2000_ut1::Number)
    # Julian centuries elapsed from the epoch J2000.0.
    t_ut1 = j2000_ut1 / 36525

    # Greenwich Mean Sideral Time at t_ut1 [s].
    θ_gmst = @evalpoly(
        t_ut1,
        + 67310.54841,
        + (876600.0 * 3600 + 8640184.812866),
        + 0.093104,
        - 6.2e-6
    )

    # Reduce to the interval [0, 86400]s.
    θ_gmst = mod(θ_gmst, 86400)

    # Convert to radian and return.
    return θ_gmst * π / 43200
end

"""
    jd_to_gmst(jd_ut1::Number)

Compute the Greenwich Mean Sideral Time (GMST) \\[rad] for the Julian Day
`jd_ut1` [UT1].

!!! info
    The algorithm is based in **[1]**(p. 188).

# References
- **[1]** Vallado, D. A (2013). Fundamentals of Astrodynamics and Applications.
    Microcosm Press, Hawthorn, CA, USA.
"""
jd_to_gmst(jd_ut1::Number) = j2000_to_gmst(jd_ut1 - 2.451545e6)

"""
    function eci_to_ecef(JD::Real)

Compute the DCM that rotates the ECI reference frame into alignment with ECEF
reference frame at Julian Day `JD`.
"""
function eci_to_ecef(JD::Real)
    gmst  = jd_to_gmst(JD)
    D_e_i = [ +cos(gmst)  +sin(gmst)  0;
              -sin(gmst)  +cos(gmst)  0;
                  0           0       1;]
    return D_e_i
end
```