Модель орбитальной ступени

Динамика твёрдого тела и систем тел

Юдинцев В. В.

Кафедра теоретической механики
Самарский университет

1 декабря 2017 г.

Файл-функция Ax.m

Файл-функция матрицы поворота вокруг оси x:

```
function A = Ax(angle)

c = cos(angle);
s = sin(angle);

A = [1,  0,  0;
     0,  c, -s;
     0,  s,  c];


end
```

Файл-функция Ay.m

Файл-функция матрицы поворота вокруг оси y:

```
function A = Ay(angle)

c = cos(angle);
s = sin(angle);

A = [c,  0,  s;
     0,  1,  0;
    -s,  0,  c];


end
```

Файл-функция Az.m

Файл-функция матрицы поворота вокруг оси z:

```
function A = Az(angle)

c = cos(angle);
s = sin(angle);

A = [c, -s,  0;
     s,  c,  0;
     0,  0,  1];


end
```

Файл-функция get_frustum.m

Файл-функция координат точек поверхности усечённого конуса:

```
function [x, y, z, c] = get_frustum(R1,R2,L,cl)

n = 32;
k = linspace(0,2*pi,n);

y = [R1*cos(k);R2*cos(k)];
z = [R1*sin(k);R2*sin(k)];
x = [zeros(1,n); L*ones(1,n)];

c = zeros(2,n,3);
c(:,:,1) = cl(1);
c(:,:,2) = cl(2);
c(:,:,3) = cl(3);

end
```

Файл-функция affine_transform.m

Поворот и перемещение точек:

```
function [x1,y1,z1]=affine_transform(x,y,z,r,A)
data = zeros(size(x,1),size(x,2),3);
data(:,:,1) = x;
data(:,:,2) = y;
data(:,:,3) = z;
data = permute(data,[3 2 1]);
r = reshape(r,3,1);
data(:,:,1) = A*data(:,:,1) + ...
                repmat(r,1,size(data(:,:,1),2));
data(:,:,2) = A*data(:,:,2) + ...
                repmat(r,1,size(data(:,:,1),2));
data = permute(data,[3 2 1]);
x1 = data(:,:,1);
y1 = data(:,:,2);
z1 = data(:,:,3);
```

Файл-функция draw_upper_stage.m

```
function draw_upper_stage(bc, rc, A)

% Nozzle
L1 = 1; R11 = 1; R12 = 0.3;
[x,y,z,c] = get_frustum(R11,R12,L1,[1,0,0]);
[x,y,z   ] = affine_transform(x-bc(1), ...
                              y-bc(2), ...
                              z-bc(3), rc, A);
surf(x,y,z,c);
```

Файл-функция draw_upper_stage.m

Двигательный отсек

```
L2 = 1; R21 = 1.5; R22 = 2;

[x,y,z,c] = get_frustum(R21,R22,L2,[0.5,0.5,0.5]);

x = x + L1;

[x,y,z  ] = affine_transform(x-bc(1),...
                             y-bc(2),...
                             z-bc(3), rc, A);
surf(x,y,z,c);

fill3(x(1,:),y(1,:),z(1,:),c(1,:,:));
```

Файл-функция draw_upper_stage.m

Корпус

```
L3 = 6; R3 = 2;

[x,y,z,c] = get_frustum(R3,R3,L3,[0.0,0.5,0.0]);

x = x + L1 + L2;

[x,y,z  ] = affine_transform(x-bc(1), ...
                             y-bc(2), ...
                             z-bc(3), rc, A);

surf(x,y,z,c);

fill3(x(2,:),y(2,:),z(2,:),c(2,:,:));
```

Файл-функция draw_upper_stage.m

Переходный отсек

```
L4 = 1; R41 = 2; R42 = 2.5;

[x,y,z,c] = get_frustum(R41,R42,L4,[0.5,1.0,0.5]);

x = x + L1 + L2 + L3;

[x,y,z  ] = affine_transform(x-bc(1), ...
                             y-bc(2), ...
                             z-bc(3), rc, A);

surf(x,y,z,c);
```

Файл-функция draw_upper_stage.m

Оси координат

```
xc = A*[0, 7;
        0, 0;
        0, 0] + repmat(rc,1,2);
yc = A*[0, 0;
        0, 5;
        0, 0] + repmat(rc,1,2);
zc = A*[0, 0;
        0, 0;
        0, 5] + repmat(rc,1,2);

line(xc(1,:),xc(2,:),xc(3,:),'Color',[1,0,0]);
line(yc(1,:),yc(2,:),yc(3,:),'Color',[0,1,0]);
line(zc(1,:),zc(2,:),zc(3,:),'Color',[0,0,1]);
```

Файл-скрипт view_rocket.m

Оси координат

```
figure;
axis([-10 10 -10 10 -10 10]);
hold on;
box;
axis vis3d;
for i=1:360
    cla;
    draw_upper_stage([4;0;0], ...
                     [0;0;0], Ay(i*pi/180.0));
    shading flat;
    lighting gouraud;
    light('Position',[0 0 10]);
    getframe;
end
```

Кватернионы

Файл-скрипт quat.m

Определение координат кватерниона по заданному направлению и углу поворота

```
function res = quat(e, phi)

res = [cos(phi*0.5), reshape(e,1,3)*sin(phi*0.5)];
```

$$\mathbf{\Lambda} = \cos\frac{\varphi}{2} + \boldsymbol{e}\sin\frac{\varphi}{2},$$

Результат (res) содержит 4 компоненты кватерниона.

Файл-скрипт quat_conj.m

Вычисление сопряженного кватерниона

```
1 function res = quat_conj(q)
2
3 res = ...
```

$$\Lambda = \lambda_0 + \boldsymbol{\lambda},\ \overline{\Lambda} = \lambda_0 - \boldsymbol{\lambda}$$

Файл-скрипт quat_mul.m

Умножение кватернионов

```
function AB = quat_mul(A, B)

AB = [A(1)*B(1)-dot(...) , A(1)*B(2:end) +...];
```

$$\boldsymbol{\Lambda} \circ \boldsymbol{B} = (\lambda_0 + \boldsymbol{\lambda}) \circ (b_0 + \boldsymbol{b})$$
$$= (\lambda_0 + \lambda_1 \boldsymbol{i}_1 + \lambda_2 \boldsymbol{i}_2 + \lambda_3 \boldsymbol{i}_3) \circ (b_0 + b_1 \boldsymbol{i}_1 + b_2 \boldsymbol{i}_2 + b_3 \boldsymbol{i}_3) =$$
$$\lambda_0 b_0 + \lambda_1 b_1 \boldsymbol{i}_1 \circ \boldsymbol{i}_1 + \lambda_2 b_2 \boldsymbol{i}_2 \circ \boldsymbol{i}_2 + \lambda_3 b_3 \boldsymbol{i}_3 \circ \boldsymbol{i}_3 +$$
$$+ \lambda_0 \boldsymbol{b} + b_0 \boldsymbol{\lambda} + \underbrace{\lambda_1 b_2 \boldsymbol{i}_3 - \lambda_1 b_3 \boldsymbol{i}_2 - \lambda_2 b_1 \boldsymbol{i}_3 + \lambda_2 b_3 \boldsymbol{i}_1 + \lambda_3 b_1 \boldsymbol{i}_2 - \lambda_3 b_2 \boldsymbol{i}_1}_{\boldsymbol{\lambda} \times \boldsymbol{b}} =$$
$$= \underbrace{\lambda_0 b_0 - \boldsymbol{\lambda} \cdot \boldsymbol{b}}_{\text{скалярная часть}} + \underbrace{\lambda_0 \boldsymbol{b} + \boldsymbol{\lambda} b_0 + \boldsymbol{\lambda} \times \boldsymbol{b}}_{\text{векторная часть}}.$$

Использовать функции cross (векторное произведение) и dot (скалярное произведение)

Файл-скрипт quat_transform.m

Поворот вектора r при помощи кватерниона

```
function rp = quat_transform(q, r)

rp = quat_mul(quat_mul(q,[0, r]),quat_conj(q));
rp = rp(2:end);
```

$$\boxed{\boldsymbol{R}' = \boldsymbol{\Lambda} \circ \boldsymbol{R} \circ \overline{\boldsymbol{\Lambda}}} \quad |\boldsymbol{\Lambda}| = 1.$$

Файл-скрипт quat_to_mat.m

Матрица поворота, соответсвующая кватерниону

```
function a = quat_to_mat(q)

a = [  2*(q(1)*q(1)+q(2)*q(2))-1,  ...,  ...;
       ...,  ...,  ...;
       ...,  ...,  ...  ];
```

$$\boldsymbol{A} = \begin{bmatrix} 2(\lambda_0^2+\lambda_1^2)-1 & 2(\lambda_1\lambda_2-\lambda_0\lambda_3) & 2(\lambda_1\lambda_3+\lambda_0\lambda_2) \\ 2(\lambda_1\lambda_2+\lambda_0\lambda_3) & 2(\lambda_0^2+\lambda_2^2)-1 & 2(\lambda_2\lambda_3-\lambda_0\lambda_1) \\ 2(\lambda_1\lambda_3-\lambda_0\lambda_2) & 2(\lambda_2\lambda_3+\lambda_0\lambda_1) & 2(\lambda_0^2+\lambda_3^2)-1 \end{bmatrix}$$

Файл-скрипт view_rocket.m

Оси координат

```
figure;
axis([-10 10 -10 10 -10 10]);
hold on;
box;
axis vis3d;
for i=1:360
 cla;
 q = quat([1 0 0], i*pi/180.0);
 draw_upper_stage([4;0;0],[0;0;0],quat_to_mat(q));
 shading flat;
 lighting gouraud;
 light('Position',[0 0 10]);
 getframe;
end
```