FUNCTIONS PAPER

Project Matlab – Math II

Aquest document inclou informació sobre totes les funcions implementades per tal
d'acomplir l'objectiu requerit en el Projecte.
Proporcionem una breu descripció sobre cada funció, els seus inputs i outputs.
https://github.com/MayKoder/MATH_2_Matlab_Project

Eudald Garrofé, Miquel Suau, Bernat Casañas

FUNCTIONS PAPER

PROJECT MATLAB – MATH II

UPDATE PURPOSE FUNCTIONS

The following described functions share the same purpose, they are called each time the cube gets redrawn during the Redraw function. Its purpose is to modify its own type of representation of the matrix of the cube and show the values in screen.

Update RotMat

```matlab
function UpdateRotMat(handles,rot_mat)
set(handles.mat_11,'String',rot_mat(1,1));
set(handles.mat_12,'String',rot_mat(1,2));
set(handles.mat_13,'String',rot_mat(1,3));
set(handles.mat_21,'String',rot_mat(2,1));
set(handles.mat_22,'String',rot_mat(2,2));
set(handles.mat_23,'String',rot_mat(2,3));
set(handles.mat_31,'String',rot_mat(3,1));
set(handles.mat_32,'String',rot_mat(3,2));
set(handles.mat_33,'String',rot_mat(3,3));
```

➢ Purpose: The purpose of this function is to modify the rotation matrix panel each time we redraw the cube through dragging it with our mouse.

➢ Inputs: Gets the current handles and the rotation matrix contained in the cube.

➢ Outputs: It has no outputs, simply modifies the static numbers contained in the rotation matrix panel in order that we can see them.

Update Quaternion

```matlab
function UpdateQuaternion(handles,rot_mat)
quaternion=rotM2Quat(rot_mat);
set(handles.q_0,'String',quaternion(1));
set(handles.q_1,'String',quaternion(2));
set(handles.q_2,'String',quaternion(3));
set(handles.q_3,'String',quaternion(4));
```

➢ Purpose: The purpose of this function is to modify the quaternion angular and vectorial part each time we redraw the cube through dragging it with our mouse. Gives use of the rotM2Quat function in order to perform the transformation from a rotation matrix to a quaternion.

➢ Inputs: Gets the current handles and the rotation matrix contained in the cube.

➢ Outputs: It has no outputs, simply modifies the editable numbers contained in the quaternion panel in order that we can see them.

Update EAA

```
function UpdateEAA(handles,rot_mat)
[euler_angle,euler_axis]=rotMat2Eaa(rot_mat);
set(handles.eu_angle,'String',euler_angle);
set(handles.eu_x,'String',euler_axis(1));
set(handles.eu_y,'String',euler_axis(2));
set(handles.eu_z,'String',euler_axis(3));
```

- ➢ Purpose: The purpose of this function is to using the rotMat2Eaa function transform the matrix contained in the cube into the principal Euler angle and axis and showing them each time we Redraw the cube.

- ➢ Inputs: Gets the current handles and the rotation matrix contained in the cube.

- ➢ Outputs: It has no outputs; it modifies the editable text strings for the Principal Euler angle and axis panel in order to see them.

Update RotVec

```
function UpdateRotVec(handles,rot_mat)
r_vec=rotM2rotVec(rot_mat);
set(handles.vec_x,'String',r_vec(1));
set(handles.vec_y,'String',r_vec(2));
set(handles.vec_z,'String',r_vec(3));
```

- ➢ Purpose: The purpose of this function is to using the rotM2rotVec function transform the matrix contained in the cube into the Rotation Vector and assigning this values to the Rotation Vector panel each time we modify the cube.

- ➢ Inputs: Gets the current handles and the rotation matrix contained in the cube.

- ➢ Outputs: It has no outputs; it modifies the editable text strings for the Rotation Vector panel in order to see them.

UpdateEA

```
function UpdateEA(handles,rot_mat)
[y,p,r]=rotM2eAngles(rot_mat);
set(handles.yaw,'String',y);
set(handles.pitch,'String',p);
set(handles.roll,'String',r);
```

- ➢ Purpose: The purpose of this function is to using the rotM2reAngle function transform the matrix contained in the cube into the three Euler angles and assigning this values to the Euler angles panel each time we modify the cube.

- ➢ Inputs: Gets the current handles and the rotation matrix contained in the cube.

- ➢ Outputs: It has no outputs; it modifies the editable text strings for the Euler Angles panel in order to see them.

TRANSFORMATION PURPOSE FUNCTIONS

For these functions its purpose is to transform an inputted matrix into another kind of rotation method.

These have been used in the functions explained before.

RotM2Quat

```matlab
function quaternion = rotM2Quat(rotation_matrix)
%ROTM2QUAT This function returns the quaternion given a rotation matrix.
%    Input: rotation matrix
%    Output: quaternion, with dimensions [4, 1]

quaternion(1, 1) = sqrt(1 + rotation_matrix(1, 1) + rotation_matrix(2, 2) + rotation_matrix(3, 3)) / 2;
quaternion(2, 1) = (rotation_matrix(3, 2) - rotation_matrix(2, 3)) / (4 * quaternion(1, 1));
quaternion(3, 1) = (rotation_matrix(1, 3) - rotation_matrix(3, 1)) / (4 * quaternion(1, 1));
quaternion(4, 1) = (rotation_matrix(2, 1) - rotation_matrix(1, 2)) / (4 * quaternion(1, 1));

end
```

➢ Purpose: This function modifies a rotation matrix method into a quaternion based method.

➢ Inputs: Gets the matrix to be transformed.

➢ Outputs: Returns the quaternion constructed from the matrix.

RotMat2Eaa

```matlab
function [a,u] = rotMat2Eaa(R)
% [a,u] = rotMat2Eaa(R)
% Computes the angle and principal axis of rotation given a rotation matrix R.
% Inputs:
%    R: rotation matrix
% Outputs:
%    a: angle of rotation
%    u: axis of rotation

%Return angle in rads
preAngle = (trace(R)-1)/2;

%Make sure angle is between -1 and 1
if preAngle >= -1 && preAngle <= 1
    a=acosd((trace(R)-1)/2);
else
    a = 0;
end

%Calculate u matrix
test = (R-R.');

if test == zeros(length(R), length(R))
    u = [1; 1; 1] / sqrt(3);
else
    mat2=(test/(2*sin(a)));

    %Return u axis
    u=[-mat2(2,3);mat2(1,3);-mat2(1,2)];
end

end
```

➢ Purpose: This function modifies a rotation matrix method into a Principal euler angle and axis based method.

➢ Inputs: Gets the matrix to be transformed.

➢ Outputs: Returns the Euler's principal angle and axis constructed from the matrix.

RotM2rotVec

```
function rotation_vector = rotM2rotVec(rotation_matrix)
%ROTM2ROTVEC  This function returns the rotation vector given a rotation
%matrix.
%    Input: rotation matrix
%    Output: rotation vector in dimensions [3, 1]

rotation_vector(2) = asind(rotation_matrix(3, 1));
rotation_vector(1) = atan2d((rotation_matrix(3, 2) / cosd(rotation_vector(2))), (rotation_matrix(3, 3) / cosd(rotation_vector(2))));
rotation_vector(3) = atan2d((rotation_matrix(2, 1) / cosd(rotation_vector(2))), (rotation_matrix(1, 1) / cosd(rotation_vector(2))));

end
```

➢ **Purpose:** This function modifies a rotation matrix method into a Rotation Vector based method.

➢ **Inputs:** Gets the matrix to be transformed.

➢ **Outputs:** Returns the Rotation Vector constructed from the matrix.

RotM2eAngles

```
function [yaw, pitch, roll] = rotM2eAngles(R)
% [yaw, pitch, roll] = rotM2eAngles(R)
% Computes the Euler angles (yaw, pitch, roll) given an input rotation matrix R.
% Inputs:
%    R: rotation matrix
% Outputs:
%    yaw: angle of rotation around the z axis
%    pitch: angle of rotation around the y axis
%    roll: angle of rotation around the x axis

local_R = R;

if abs(R(3, 1)) == 1
    yaw = 0;
    if R(3, 1) == 1
        pitch = 90;
    else
        pitch = 270;
    end

    roll = atan2d((local_R(3, 2) / cos(pitch)), ((local_R(3, 3) / cos(pitch))));

else

    pitch = asind(-R(3, 1));

    yaw = atan2d((local_R(2, 1) / cos(pitch)), ((local_R(1, 1) / cos(pitch))));
    roll = atan2d((local_R(3, 2) / cos(pitch)), ((local_R(3, 3) / cos(pitch))));

end
end
```

➢ **Purpose:** This function modifies a rotation matrix method into a Euler Angles based method.

➢ **Inputs:** Gets the matrix to be transformed.

➢ **Outputs:** Returns the three Euler Angles pitch, roll and yaw constructed from the matrix.

RotQua2M

```matlab
function [R] = rotQua2M(q)
%QUAT2ROTMAT This function returns the conversion of a quaternion to a
%rotation matrix.
%Input: quaternion
%Output: rotation matrix

q = q/norm(q);

R = [q(1)^2 + q(2)^2 - q(3)^2 - q(4)^2, 2*q(2)*q(3) - 2*q(1)*q(4), 2*q(2)*q(4) + 2*q(1)*q(3);
    2*q(2)*q(3) + 2*q(1)*q(4), q(1)^2 - q(2)^2 + q(3)^2 - q(4)^2, 2*q(3)*q(4) - 2*q(1)*q(2);
    2*q(2)*q(4) - 2*q(1)*q(3), 2*q(3)*q(4) + 2*q(1)*q(2), q(1)^2 - q(2)^2 - q(3)^2 + q(4)^2];

end
```

> ➢ Purpose: This function creates a rotation matrix from a quaternion.

> ➢ Inputs: Gets the quaternion.

> ➢ Outputs: Returns the rotation matrix.

eAngles2rotM

```matlab
function [R] = eAngles2rotM(yaw, pitch, roll)
% [R] = eAngles2rotM(yaw, pitch, roll)
% Computes the rotation matrix R given the Euler angles (yaw, pitch, roll).
% Inputs:
%    yaw: angle of rotation around the z axis
%    pitch: angle of rotation around the y axis
%    roll: angle of rotation around the x axis
% Outputs:
%    R: rotation matrix

r_yaw = [cosd(yaw), sind(yaw), 0;
         -sind(yaw), cosd(yaw), 0;
         0, 0, 1];

r_pitch = [cosd(pitch), 0, -sind(pitch);
           0, 1, 0;
           sind(pitch), 0, cosd(pitch)];

r_roll = [1, 0, 0;
          0, cosd(roll), sind(roll);
          0, -sind(roll), cosd(roll)];

R = transpose(r_roll * r_pitch *r_yaw);

end
```

> ➢ Purpose: This function creates a rotation matrix from the three Euler angles.

> ➢ Inputs: Gets three angles

> ➢ Outputs: Returns the rotation matrix.

Eaa2RotMat

```matlab
function [R] = Eaa2rotMat(a,u)
% [R] = Eaa2rotMat(a,u)
% Computes the rotation matrix R given an angle and axis of rotation.
% Inputs:
%    a: angle of rotation
%    u: axis of rotation
% Outputs:
%    R: generated rotation matrix

%Normalize matrix
u = u/ norm(u);

len = length(u);
s_cos =cos(a);
s_sen =sin(a);

I = eye(len);
uuT = u * transpose(u);

Ux = [0,-u(3),u(2);
      u(3),0,-u(1);
      -u(2),u(1),0];

R = (I*s_cos)+((1-s_cos)*uuT) +(Ux*s_sen);

end
```

➤ Purpose: This function creates a rotation matrix from the three the principal Euler angle and axis

➤ Inputs: Gets an angle and a vector.

➤ Outputs: Returns the rotation matrix.

rotVec2rotMat

```matlab
function rotation_matrix = rotVec2rotMat(rot_vector)
%ROTVEC2ROTMAT This functions returns the rotation matrix of a given
%rotation vector.

%    Input: rotation vector
%    Output: rotation matrix

normalized_vector = norm(rot_vector);
rot_vector = rot_vector / normalized_vector;

rot_mat = [0 -rot_vector(3) rot_vector(2);
           rot_vector(3) 0 -rot_vector(1);
           -rot_vector(2) rot_vector(1) 0];

rotation_matrix = eye(3) * cosd(normalized_vector) + (1 - cosd(normalized_vector)) * (rot_vector * rot_vector') + sind(normalized_vector) * rot_mat ;

end
```

➤ Purpose: This function creates a rotation matrix from the three a rotation vector.

➤ Inputs: Gets a vector.

➤ Outputs: Returns the rotation matrix.

PUSH BUTTON FUNCTIONS

The following functions are the ones that get triggered once a tagged button is pushed.

Reset

```matlab
function reset_Callback(hObject, eventdata, handles)
% hObject     handle to reset (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)
R=eye(3);
handles.Cube=RedrawCube(R,handles);
```

> ➢ Purpose: This function has the purpose of restarting the matrix of the cube using a defined identity matrix and sending it to redraw the cube from it.

> ➢ Inputs: Recibes the needed parameters that Matlab requires to perform a callback with a button.

> ➢ Outputs: It has no outputs, simply performs a reset on the cube, including the whole other rotation methods on the panels that directly depend on the cube matrix.

Quaternion

```matlab
function q_rot_Callback(hObject, eventdata, handles)
% hObject     handle to q_rot (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)
q=zeros(1,4);
q(1)=str2double(get(handles.q_0,'String'));
q(2)=str2double(get(handles.q_1,'String'));
q(3)=str2double(get(handles.q_2,'String'));
q(4)=str2double(get(handles.q_3,'String'));

R= rotQua2M(q);
handles.Cube=RedrawCube(R,handles);
```

> ➢ Purpose: This function gets the current values of the editable text boxes of the quaternion panel and creates a rotation matrix from them, later it passes the matrix to redraw de cube.

> ➢ Inputs: Recibes the needed parameters that Matlab requires to perform a callback with a button.

> ➢ Outputs: It has no outputs.

Principal Euler Angle and Axis

```
function eu_angle_Callback(hObject, eventdata, handles)
% hObject      handle to eu_angle (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
a= str2double(get(handles.eu_angle,'String'));
u=zeros(3,1);
u(1)= str2double(get(handles.eu_x,'String'));
u(2)= str2double(get(handles.eu_y,'String'));
u(3)= str2double(get(handles.eu_z,'String'));

R=Eaa2rotMat(a,u);
handles.Cube=RedrawCube(R,handles);
```

➢ Purpose: This function gets the current values of the editable text boxes of the euler principal angle and axis panel and creates a rotation matrix from them, later it passes the matrix to redraw de cube.

➢ Inputs: Recibes the needed parameters that Matlab requires to perform a callback with a button.

➢ Outputs: It has no outputs.

Euler Angle

```
function angles_rot_Callback(hObject, eventdata, handles)
% hObject      handle to angles_rot (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
yaw=str2double(get(handles.yaw,'String'));
pitch=str2double(get(handles.pitch,'String'));
roll=str2double(get(handles.roll,'String'));

R=eAngles2rotM(yaw,pitch,roll);
handles.Cube=RedrawCube(R,handles);
```

➢ Purpose: This function gets the current values of the editable text boxes of the euler angles panel and creates a rotation matrix from them, later it passes the matrix to redraw de cube.

➢ Inputs: Recibes the needed parameters that Matlab requires to perform a callback with a button.

➢ Outputs: It has no outputs.

Rotation Vector

```matlab
function vec_rot_Callback(hObject, eventdata, handles)
% hObject    handle to vec_rot (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

vec=zeros(3,1);
vec(1)=str2double(get(handles.vec_x,'String'));
vec(2)=str2double(get(handles.vec_y,'String'));
vec(3)=str2double(get(handles.vec_z,'String'));

R=rotVec2rotMat(vec);
handles.Cube=RedrawCube(R,handles);
```

> ➤ Purpose: This function gets the current values of the editable text boxes of the rotation vector panel and creates a rotation matrix from them, later it passes the matrix to redraw de cube.

> ➤ Inputs: Recibes the needed parameters that Matlab requires to perform a callback with a button.

> ➤ Outputs: It has no outputs.