Template of Manipulator Short project: Skull tumor surgery

Grup 11- Estudiants:

- **Pol Casacuberta Gil**
- **Marta Granero i Martí**

Link: https://drive.matlab.com/sharing/b8a7f88a-dcb6-4c09-a328-ba6c7e737807

I spect: 1) Pdf file, 2) Videos demostrating your successful task and your mlx file.

Notes. For better undestanding you can split the videos in the meaninful task.

Remember use the options of serial/link plot:

'workspace' for centering in the surgery task

'zoom' ... nice puma ratio aspect

'trail' .. to see the trajectory

etc..

See all at:

>> help SerialLink/plot

Table of Contents

The Robotic environment (10%)

Think that later on the enviroment will move to any place in a Univers Reference Frame {U}

Use: 'c = uisetcolor' to chose your prefered colors

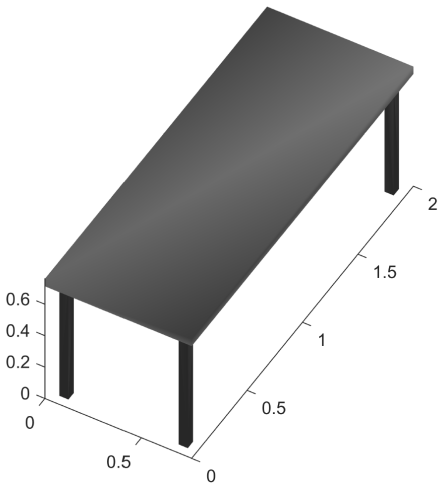
```
% c = uisetcolor
```

Operating table.

It can be raised, lowered, and tilted in any direction, and an auxiliary table for the tools. Define: Vertices and Faces and use 'patch' functions to model it. See help patch to find and example.

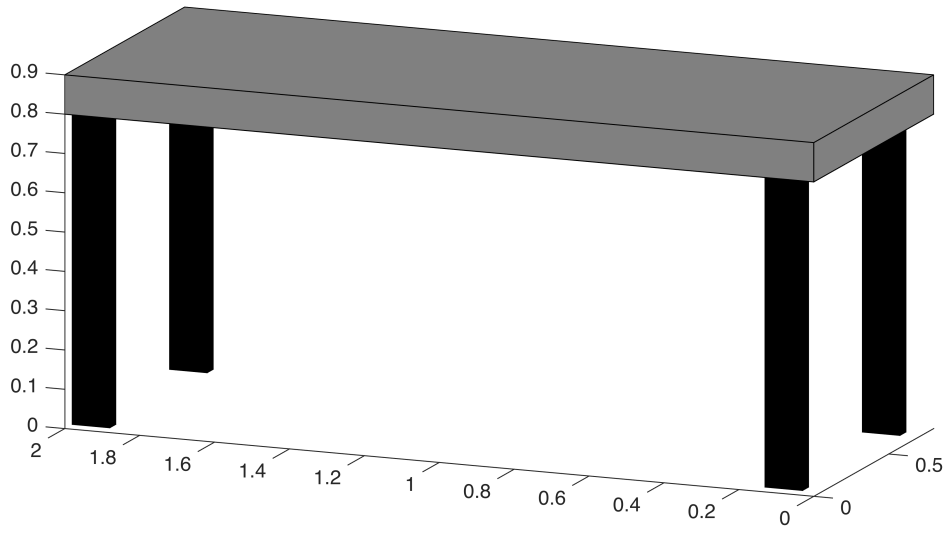
Think that later on the enviroment will move to any place in a Univers Reference Frame {U}

Expected results




```
clear
close all
clf
v = [-20 -8 5];
[caz,cel] = view(v);

% Define the dimensions of the flat board
board_width = 0.8; % width of the board (y)
board_length = 2; % length of the board (x)
board_thickness = 0.1; % thickness of the board

% Define the vertices of the board
v = [0, 0, 0;  % bottom left corner
     board_width, 0, 0; % bottom right corner
     board_width, board_length, 0; % top right corner
     0, board_length, 0; % top left corner
     0, 0, board_thickness; % bottom left corner of top surface
     board_width, 0, board_thickness; % bottom right corner of top surface
     board_width, board_length, board_thickness; % top right corner of top surface
     0, board_length, board_thickness]; % top left corner of top surface
```

```
% Define the faces of the board
f = [1, 2, 6, 5; % bottom face
     2, 3, 7, 6; % right face
     3, 4, 8, 7; % top face
     4, 1, 5, 8; % left face
     5, 6, 7, 8; % top surface
     1, 2, 3, 4]; % bottom surface
T_mesa_U = transl(0,0, 0.8);

v_pata = [0.05 0 0; 0 0.05 0; 0 0 8]*v'; % We scale the table in such a way we get a l
v_pata1 = v_pata + [0.05 0.05 -0.8]';
v_pata2 = v_pata + [ 0.70 0.05 -0.8]';
v_pata3 = v_pata + [ 0.05 1.90 -0.8]';
v_pata4 = v_pata + [ 0.70 1.90 -0.8]';

v_pata1 = v_pata1';
v_pata2 = v_pata2';
v_pata3 = v_pata3';
v_pata4 = v_pata4';

v_pata1(:,4) = 1;
v_pata2(:,4) = 1;
v_pata3(:,4) = 1;
v_pata4(:,4) = 1;

v_pata1 = T_mesa_U*v_pata1';
v_pata1 = v_pata1(1:3,:)';
v_pata2 = T_mesa_U*v_pata2';
v_pata2 = v_pata2(1:3,:)';
v_pata3 = T_mesa_U*v_pata3';
v_pata3 = v_pata3(1:3,:)';
v_pata4 = T_mesa_U*v_pata4';
v_pata4 = v_pata4(1:3,:)';

v(:,4) = 1;
v(:,4) = 1;
v = v';

v = T_mesa_U *v;
% Plot the flat board
v = v';
patch('Vertices', v(:,1:3), 'Faces', f, 'FaceColor', [0.5, 0.5, 0.5]);

patch('Vertices',v_pata1(:,1:3),'Faces',f,'FaceVertexCData',hsv(6),'FaceColor',[0.0 0.
patch('Vertices',v_pata2,'Faces',f,'FaceVertexCData',hsv(6),'FaceColor',[0.0 0.0 0.0])
patch('Vertices',v_pata3,'Faces',f,'FaceVertexCData',hsv(6),'FaceColor',[0.0 0.0 0.0])
patch('Vertices',v_pata4,'Faces',f,'FaceVertexCData',hsv(6),'FaceColor',[0.0 0.0 0.0])

axis equal; hold on;
```

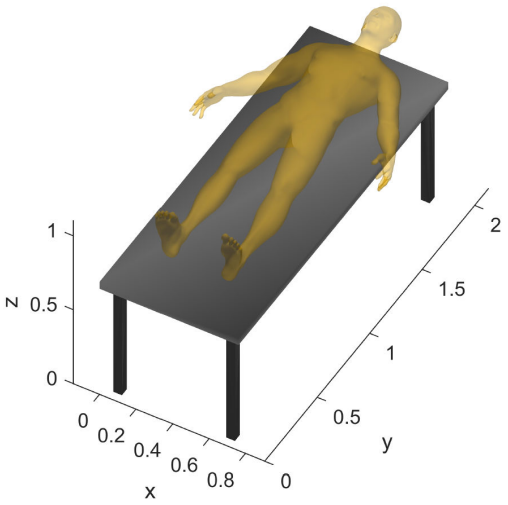



```
v1 = [0,0,0];
% % Table reference frame
% Tb_U_ref = transl(v1);
```

3D model of a human body

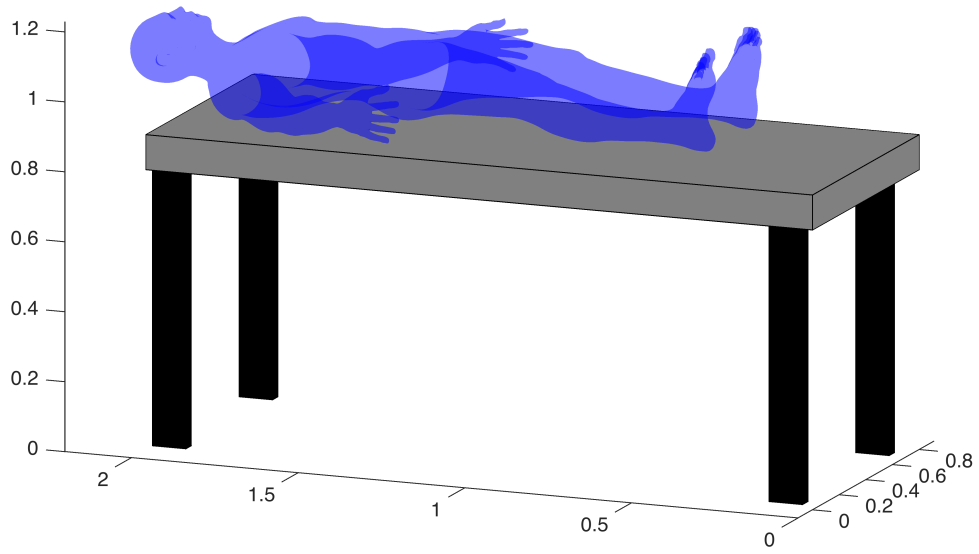
Situate the human model on the operating table.

Expected results



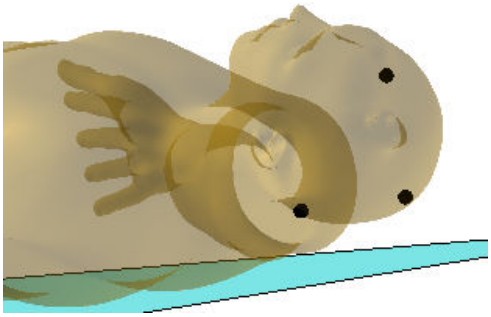
put your code Here

```
load('F_V_HumanBody.mat');
Vh(:,4) = 1;
Vh = Vh';
% We add a little offset between the man and the table to avoid z-fighting
T_human_mesa = transl(0,0,board_thickness+0.00001)*trotx(-pi/2)*transl(board_width/2, -
T_U_H =T_mesa_U*T_human_mesa;
Vh = T_U_H*Vh;
Vh = Vh';
patch('Vertices',Vh(:,1:3),...
    'Faces',Fh,...
    'FaceColor',['b'],...
    'FaceAlpha',0.3,...
    'Linestyle','none');
```



Fiducials

The Radiology Department before to take a Computer Tomography (CT) of the brain, fix three fiducials in the head of the patient for registering purpose, visit: https://en.wikipedia.org/wiki/Fiducial



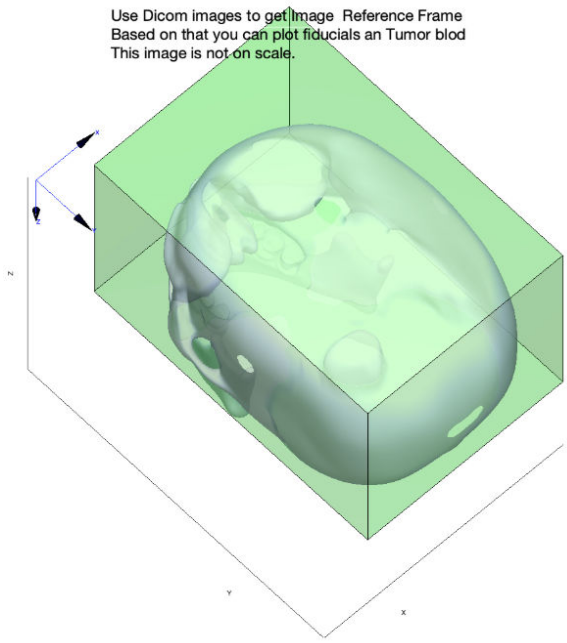
Dicom image vs Image Reference frame {I}

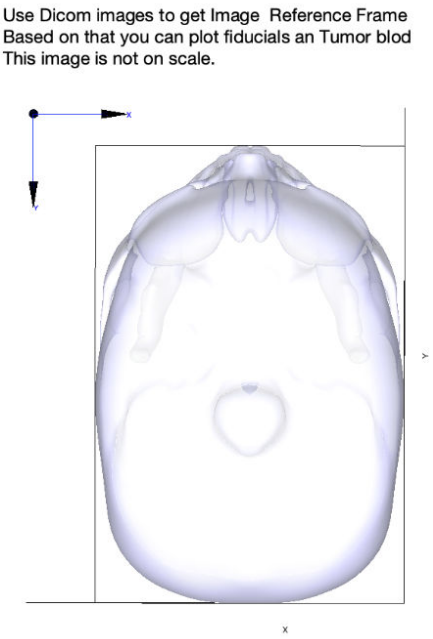
Get familiar with Dicom Images, Visit: https://www.imaios.com/en/Imaios-Dicom-Viewer#!

Use a container Box of the skull to infer the Image Reference Frame {I}

See: 6_Plot_Box_Cone.mlx and 7_Help_Image_RF_Containig_Box.fig to inspire yourselt

Expected results



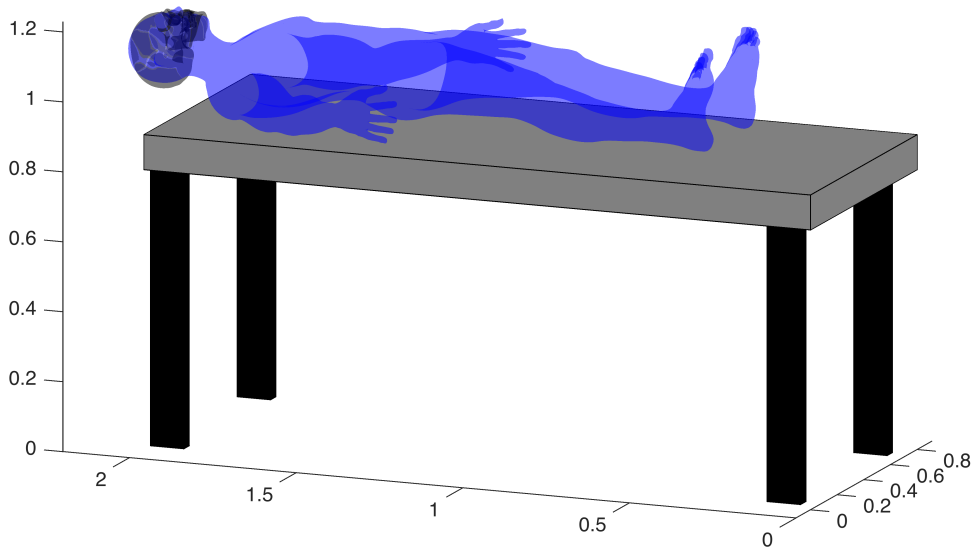


put your code Here

```
% clear
% close all
% clf
```

```
load('F_V_Skull.mat');
```

```
Vs(:,4) = 1;

% Skull reference frame with translation and rotation
S_U_ref = transl(0.4,1.98,0.25)*trotx(-90,'deg');

S_U_ref = T_mesa_U *S_U_ref;
% Patch skull with new vertices
Vs = (S_U_ref* Vs')';
patch('Vertices',Vs(:,1:3),...
    'Faces',Fs,...
    'FaceColor','black',...
    'FaceAlpha',0.3,...
    'Linestyle','none');
```




```
% Max and min dichotomy with y and z changed to change axis
maxDichotomy  = [206 1 236];
minDichotomy  = [43 112 25];
min(Vs(:,2))
```

```
ans = 1.9804
```

```
max(Vs(:,2))
```

```
ans = 2.1927
```

```
% Containing box min and max point (notice that y (which is the new z) is inverted)
minSkullSpace = [min(Vs(:,1)) max(Vs(:,2))           min(Vs(:,3))];
```

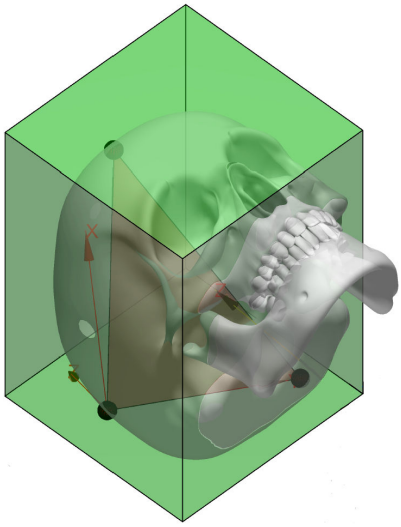
```
% We do not want the lower part of the skull so we start a little over the
% lowest point (by eye)
maxSkullSpace = [max(Vs(:,1)) min(Vs(:,2))+0.06  max(Vs(:,3))];

% v= [0 0 0;1 0 0;1 1 0;0 1 0;0 0 1;1 0 1;1 1 1;0 1 1]
% f = [1 2 6 5;2 3 7 6;3 4 8 7;4 1 5 8;1 2 3 4;5 6 7 8]
%
% % Scale and translate the box to match the skull bounding box
% scaleFactor = (maxSkullSpace - minSkullSpace) ./ [1 1 1];
% boxCenter = minSkullSpace + scaleFactor ./ 2;
% % Compute center point of skull bounding box
% centerSkullSpace = (maxSkullSpace + minSkullSpace) / 2;
%
% v = v .* scaleFactor + boxCenter;
% v(:,4) = 1;
% v = T_mesa_U * S_U_ref * v';
% v = v';
%
%
% patch('Vertices',v(:,1:3),'Faces',f,'FaceVertexCData',hsv(6),'FaceColor','g','FaceAl


% Relation between dichotomy and matlab coordinates
dichotomy2Space = abs((maxSkullSpace-minSkullSpace)./(maxDichotomy-minDichotomy));

% Calculate the position of the image reference frame
referenceFrameSkullOrigin = maxSkullSpace + minDichotomy.*dichotomy2Space;

% Compute the image referenc frame with tranlations and rotations
I_U_ref = transl(referenceFrameSkullOrigin)*trotx(pi/2)*trotz(pi);
```

Fiducials wrt {I}

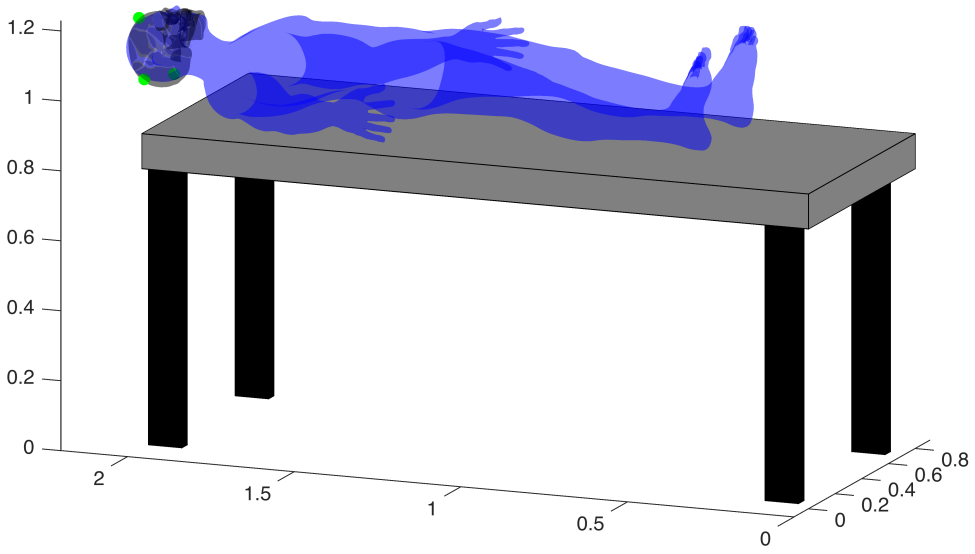
Use the Dicom images to place the fiducial relative to Image Reference Frame {I}.

See: 5_Skull_pose_estimation.mlx and use the skull to make the exercise.

Expected results



```
% We set by hand (using dicom viewer) the fiducial coordinates
fiducial1Coord = [70 214 32];
fiducial2Coord = [186 210 65];
fiducial3Coord = [122 62 94];

% Transformation matrix that transforms points from dichotomy to univers
d2s = [1 0 0 0; 0 0 1 0; 0 1 0 0; 0 0 0 1]*...
      diag([dichotomy2Space 1])*...
      [1 0 0 0; 0 0 -1 112; 0 1 0 0; 0 0 0 1];

% Compute and patch the 3 fiducials
fiducial1Space = I_U_ref*d2s*[fiducial1Coord 1]';
scatter3(fiducial1Space(1),fiducial1Space(2),fiducial1Space(3),'green','filled');
fiducial2Space = I_U_ref*d2s*[fiducial2Coord 1]';
scatter3(fiducial2Space(1),fiducial2Space(2),fiducial2Space(3),'green','filled');
fiducial3Space = I_U_ref*d2s*[fiducial3Coord 1]';
scatter3(fiducial3Space(1),fiducial3Space(2),fiducial3Space(3),'green','filled');
```

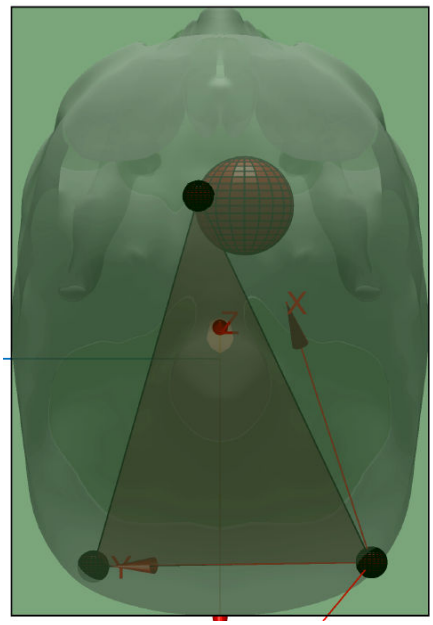


Tumor points wrt {I}

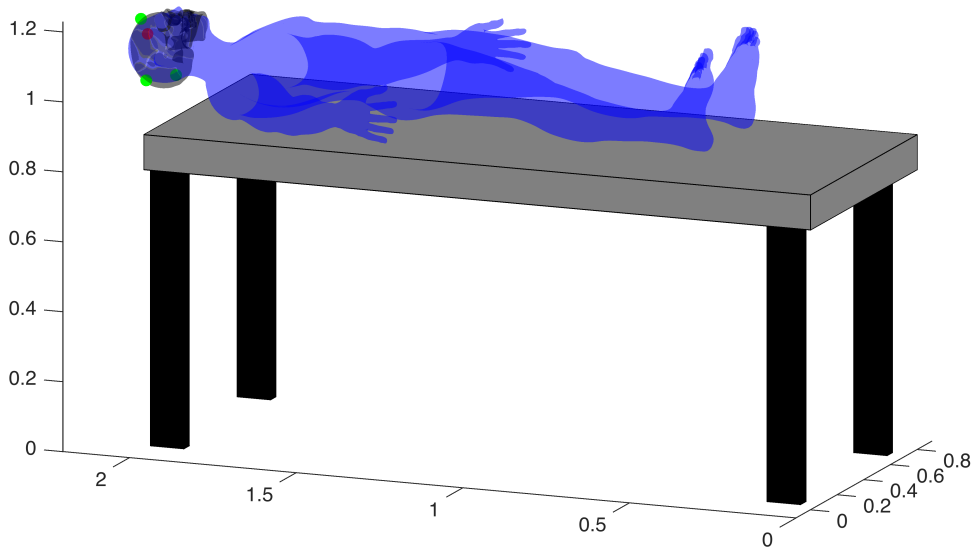
Use the Dicom images to get the points of the outer perimeter of the tumor relative to Image Reference Frame {I}.

You can simplify the tumor information by defining the center of mass and estimate an equivalent diameter.

Expected results



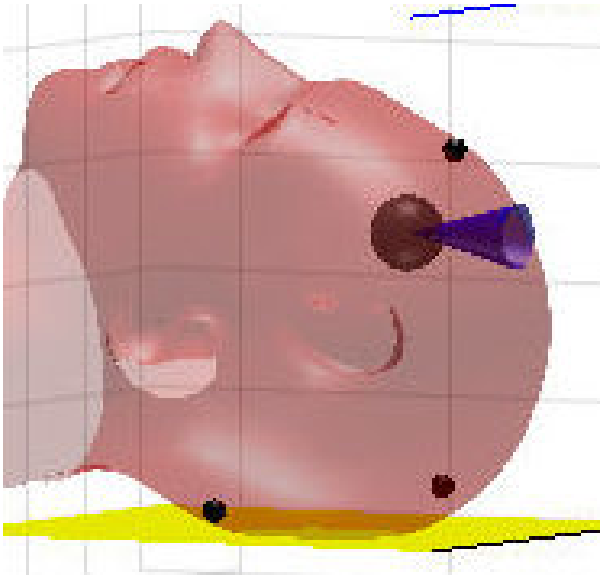
```
% Dichotomy tumor center coordinates
tumorCenterCoords = [132 98 77];

% Tumor radius in mm
tumorRadius = 0.016;

% Tumor center position in univers
tumorCenterUnivers = I_U_ref*d2s*[tumorCenterCoords 1]';

% Plot tumor as sphere
plot_sphere(tumorCenterUnivers,tumorRadius,[1 0 0],0.5);
```



Fiducials and Tumor wrt Human Reference Frame

Place fiducial and tumor in the head of the human. You will have to re-do the containing box secction.

Expected results



First approach (10%)

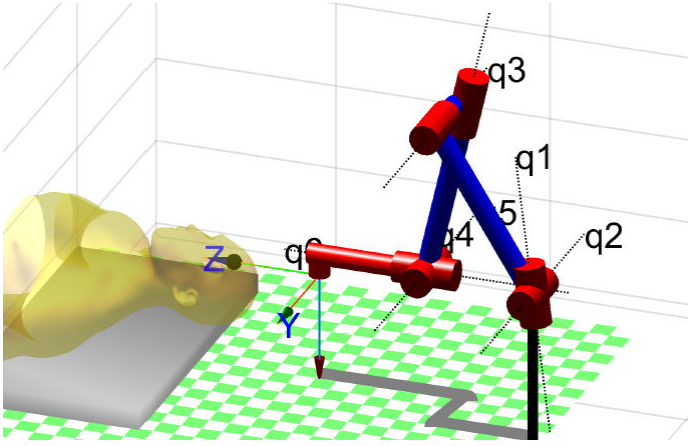
Asume that the ZX plane of the Robot is aligned with the plane of symmetry of the human body.

Robot manipulator

Consider the best position of manipulator to be nearby the operating table to warranty that the head is in the reachable work space. Use a Puma 560. Use p560.teach to play.

Use: p560.base & p560.tool to locate the Puma and add the tools.

Expected results



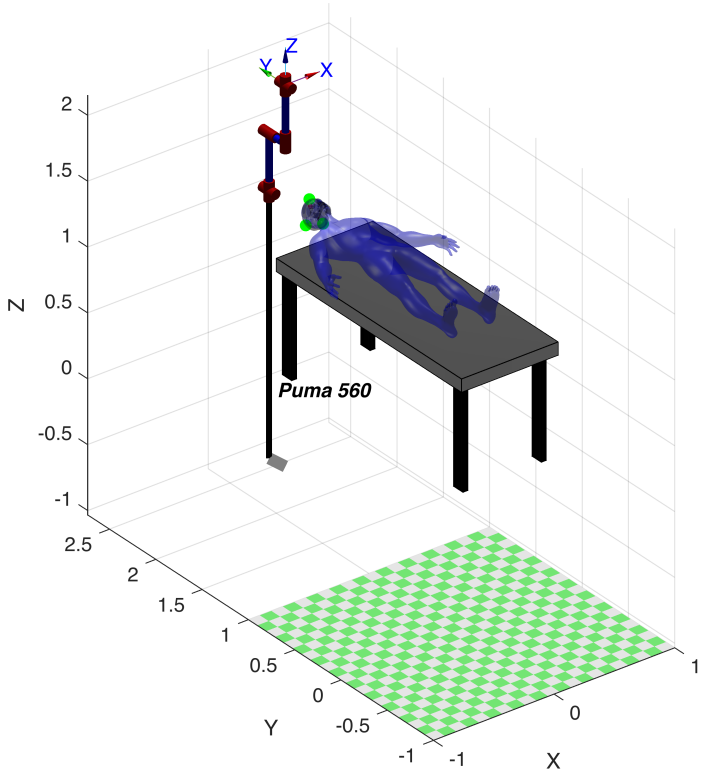
put your code Here

```
mdl_puma560;
```

```
% p560.base=transl(5.75,9.75,0)*trotz(-pi/4)
p560.base=transl(0.4, 2.6, 1)
```

```
p560 =

Puma 560 [Unimation]:: 6 axis, RRRRRR, stdDH, fastRNE
 - viscous friction; params of 8/95;
+---+-----------+-----------+-----------+-----------+-----------+
| j |     theta |         d |         a |     alpha |    offset |
+---+-----------+-----------+-----------+-----------+-----------+
|  1|         q1|          0|          0|     1.5708|          0|
|  2|         q2|          0|     0.4318|          0|          0|
|  3|         q3|    0.15005|     0.0203|    -1.5708|          0|
|  4|         q4|     0.4318|          0|     1.5708|          0|
|  5|         q5|          0|          0|    -1.5708|          0|
|  6|         q6|          0|          0|          0|          0|
+---+-----------+-----------+-----------+-----------+-----------+
base:    t = (0.4, 2.6, 1), RPY/xyz = (0, 0, 0) deg
```

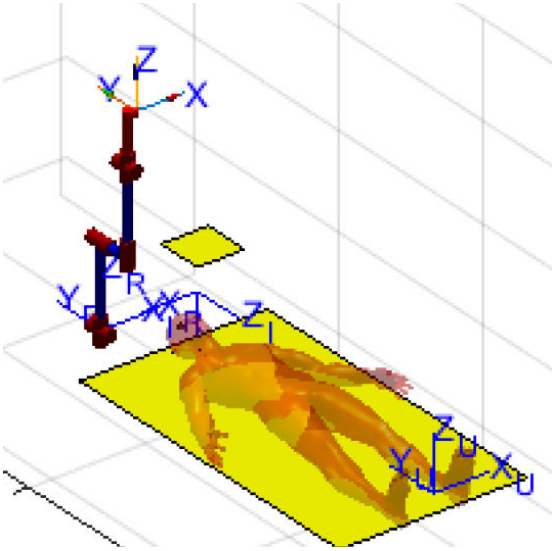
```
 p560.plot(qr,'zoom',1.5)
% axis([5 6.5 8.8 10 0 1.8])
T_R_U=p560.base.T;
hold on
axis equal;
```



Reference Frames

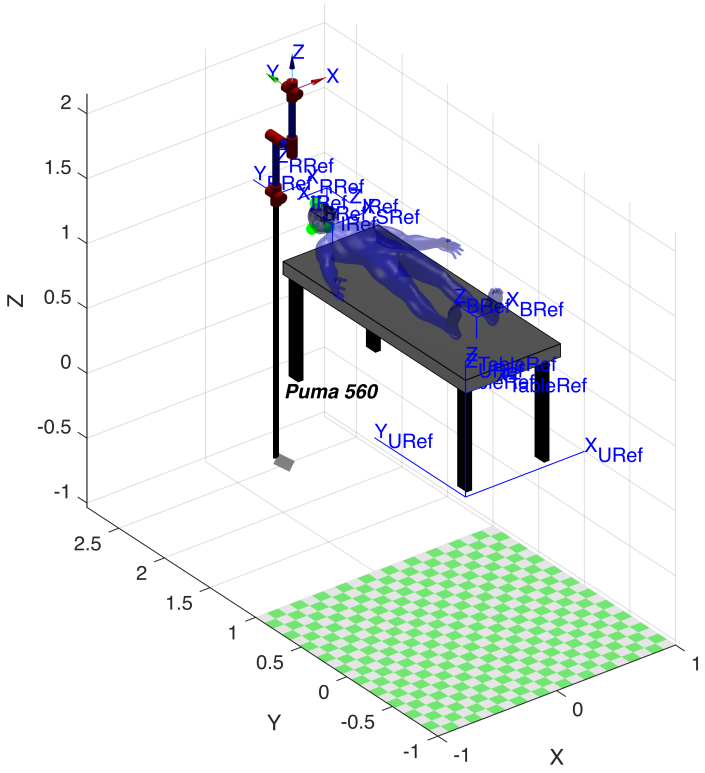
Display all necessary reference frame. Use best scale to see it.

- {U} Univers [0 0 0]
- {R} Robot
- {I} Image
- {Tb} Table_body
- {Tt} Table tool
- {EE} End Efector
- others
- ...



put your code Here

```
plotScale = diag([0.25, 0.25, 0.25, 1]);
trplot(eye(4)              , 'frame', 'URef'    );
trplot(T_mesa_U * plotScale, 'frame', 'TableRef');
trplot(T_U_H  * plotScale, 'frame', 'BRef'    );
trplot(S_U_ref  * plotScale, 'frame', 'SRef'    );
trplot(T_R_U  * plotScale, 'frame', 'RRef'    );
trplot(I_U_ref  * plotScale, 'frame', 'IRef'    );
```




```
```

Transformations

Enumerate the transformation you will need.

put your code Here

```
```

Tumor points in Robot Frame.

Remember the Transform compound exercise

```
%% put your code Here
```

Second approach: (25%)

Modify your code to repeat the exercise if the table with the patient is given as happend in the Rosa video.

To know the head relative pose with respect to the Puma Robot ...

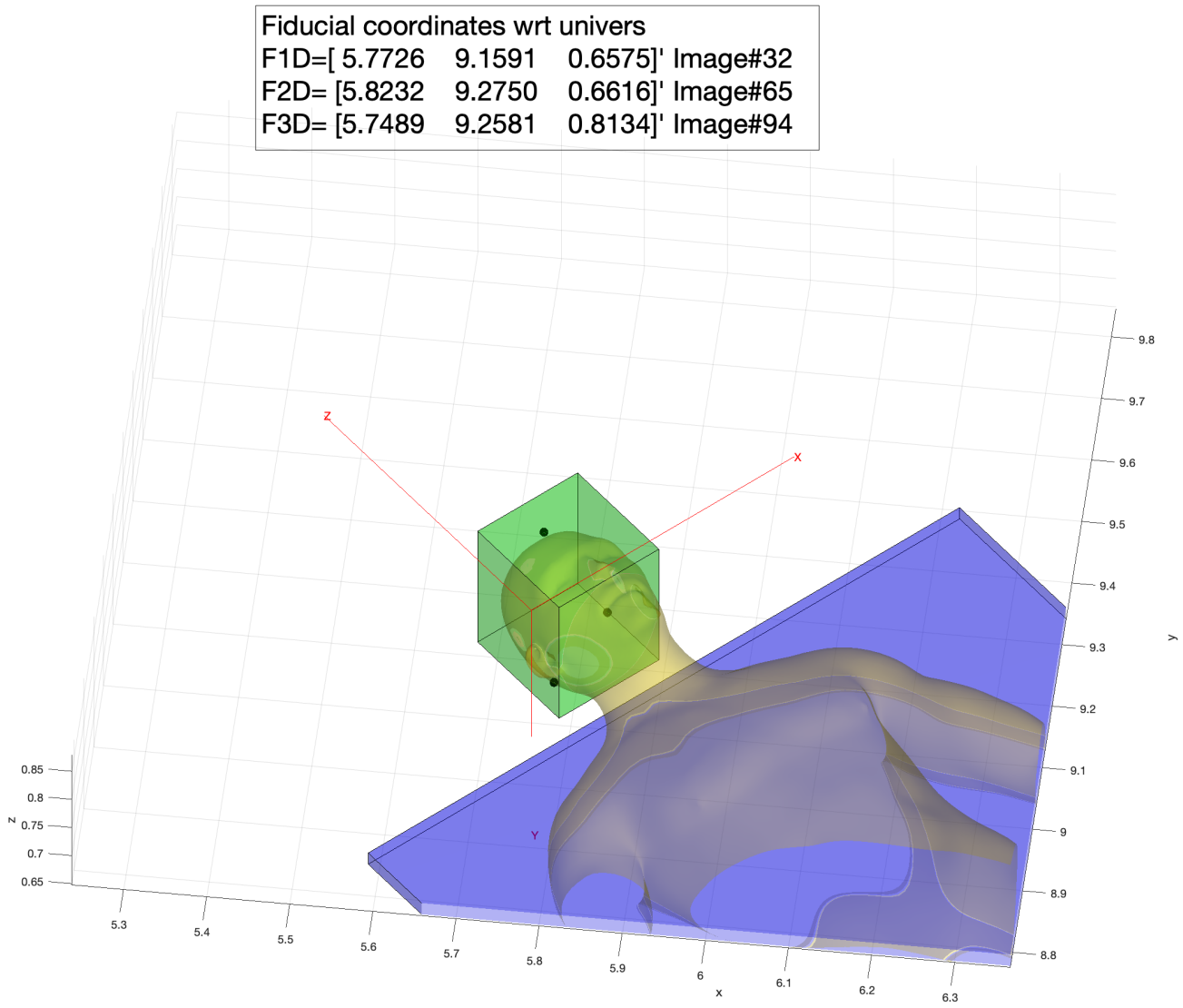
```
clear
clf
```

```
close all
```

```
% Open the new starting figure
open('3_Second_approach_Patient_pose.fig')
patient = openfig('3_Second_approach_Patient_pose.fig','visible')
```

```
patient =
  Figure (13: Figure) with properties:

      Number: 13
        Name: 'Figure'
       Color: [1 1 1]
    Position: [1001 99 1193 1052]
       Units: 'pixels'

  Show all properties
```

```
hold on
```

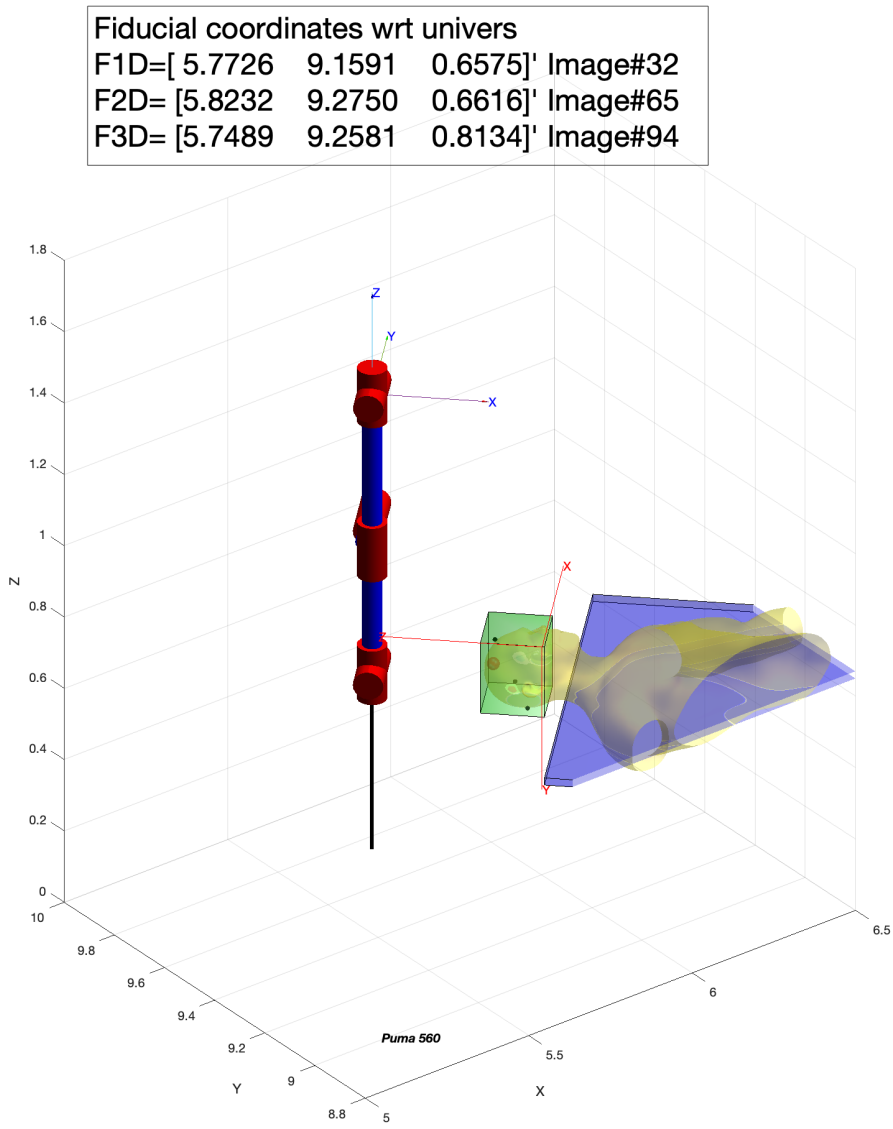



```
% New fiducials
F1 = [5.7726 9.1591 0.6575]';
F2 = [5.8232 9.2750 0.6616]';
F3 = [5.7489 9.2581 0.8134]';

F1D = [0.067 0.200 0.112]';
F2D = [0.159 0.200 0.058]';
F3D = [0.107 0.076 0.010]';
```

```
% Compute the universe fiducials reference frame
Zf = (F2-F1)/norm(F2-F1);
b = (F3-F1)/norm(F3-F1);
```

```
Yf = cross(Zf,b)/norm(cross(Zf,b));
Xf = cross(Zf,Yf)/norm(cross(Zf,Yf));
UTF = [[Xf;0] [Zf;0] [Yf;0] [F1;1]];

% Compute the dicom fiducials reference frame
ZfD = (F2D-F1D)/norm(F2D-F1D);
bD = (F3D-F1D)/norm(F3D-F1D);
YfD = cross(ZfD,bD)/norm(cross(ZfD,bD));
XfD = cross(ZfD,YfD)/norm(cross(ZfD,YfD));
UTFD = [[XfD;0] [ZfD;0] [YfD;0] [0;0;0;1]];

% Compute the image reference frame and the universe reference frame
AUX = UTF*inv(UTFD);
IRefFrame = AUX*transl(-F1D);
URefFrame = [[t2r(AUX) [0;0;0]]; [0 0 0 1]]*trotx(90,'deg')*trotz(180,'deg');
```

```
% Define robot
mdl_puma560;
p560.base=transl(5.75,9.75,0.5)*trotz(-pi/4);
p560.plot(qr,'zoom',1.5)
% Define tumor information
tumorRadius = 0.016;
tumorDicomPosition = [0.123 0.092 0.049];
tumorPosition2A = IRefFrame*[tumorDicomPosition 1]';

% Plot the tumor
plot_sphere(tumorPosition2A(1:3),tumorRadius,[1 0 0],0.3);
axis([5 6.5 8.8 10 0 1.8])
```




```
% Tool definition
p560.tool = transl(0.05, 0, 0.25);
```

Surgery (55%)

Biospy

Prepare a script that perform a biopsy. Zoom in the scene and record a video with the best view.

Take into account:

- Use a tool that has the following Transformation: transl(0.05 0 0.25)
- Use 'trail' option of plot to visualize the trajectory.

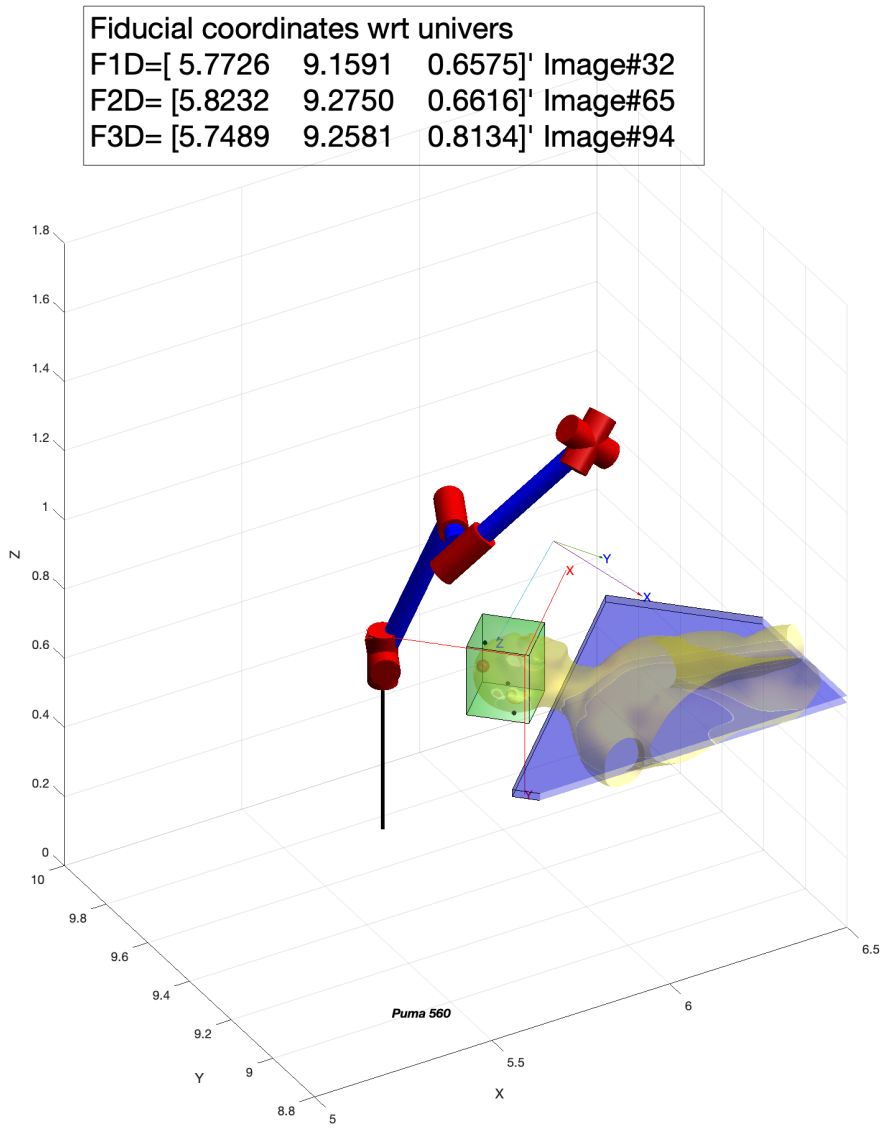
- The speed of biopsy function that you design ought to be a parameter to satisfy the surgeons.

Answer these questions:

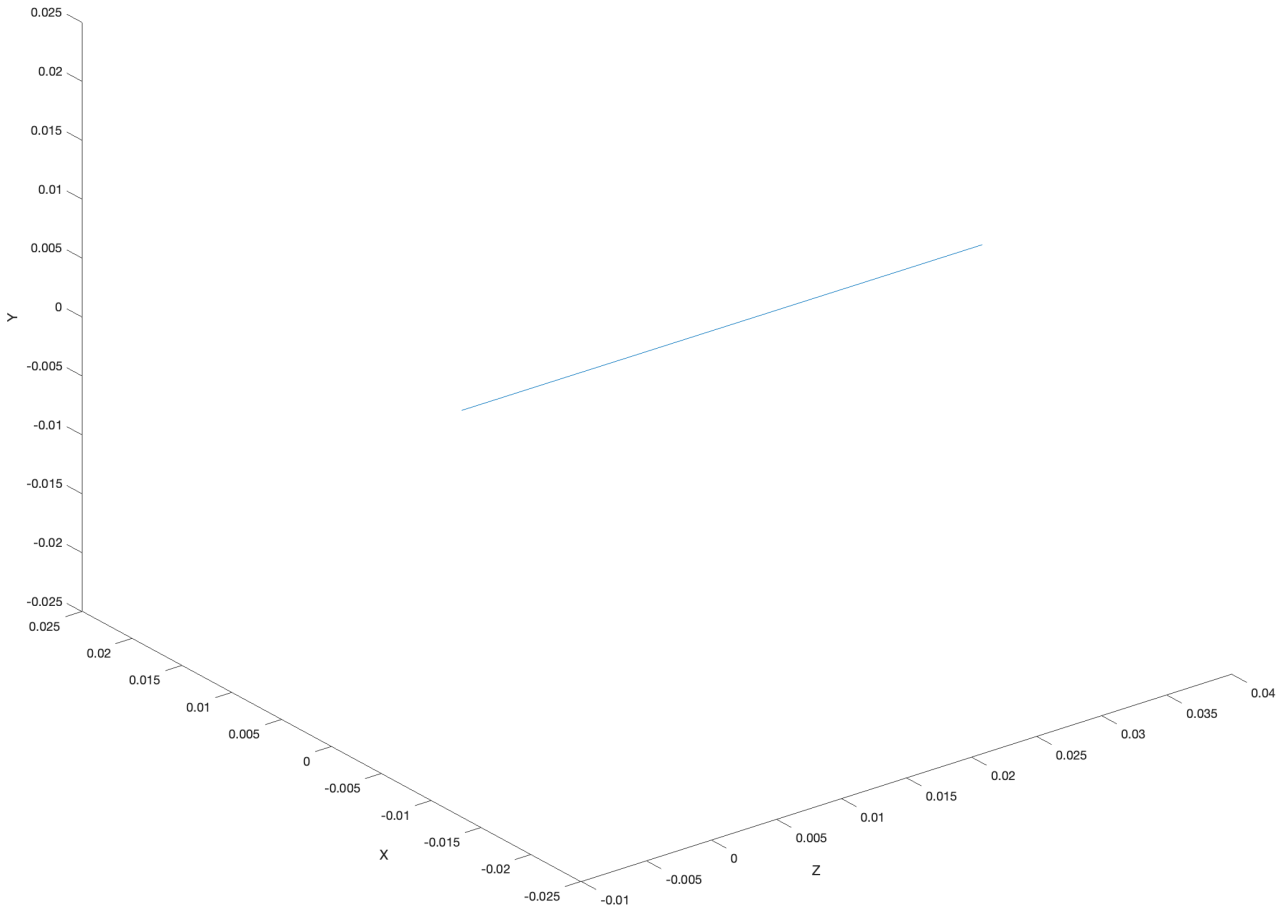
- Display in a figure the displacement, velocity and aceleration of the tool i End Efector Reference frame.
- How much enter the tool in the patient brain.
- What are the speed of the tool in Wordl Reference Frame.

```
% Calculate the tumor reference frame and the initial position of it.
tumorReference2A = transl(tumorPosition2A(1:3))*URefFrame*...
    trotz(-90,'deg')*troty(90,'deg');
initRobotFrame2A = transl(tumorPosition2A(1:3))*URefFrame*...
    transl([0 0.35 0])*trotz(-90,'deg')*troty(90,'deg');

% Define approach and recede trajectories
t = tpoly(0,1,300);
approach2A = ctraj(initRobotFrame2A, tumorReference2A, t);
recede2A = ctraj(tumorReference2A, initRobotFrame2A, t);
trajPoints = cat(3,approach2A,recede2A);

% combine approach and recede
trajectory = p560.ikine6s(trajPoints,'run');

% Plot results
% xlim([2.8 3.8]); ylim([2.5 3.7]); zlim([1 1.7]);
p560.plot(trajectory,'zoom',2.5,'workspace',[3 4 2.4 3.5 1 1.7],...
 'view',[44.463 8.2778],'trail', {'r', 'LineWidth', 1.5} );
%p560.plot(trajectory,'zoom',2.5,'movie','Biopsy2A.mp4','workspace',...
 %[3 4 2.4 3.5 1 1.7],'fps',40,'trail',"-",'jaxes','view',[44.463 8.2778]);
```

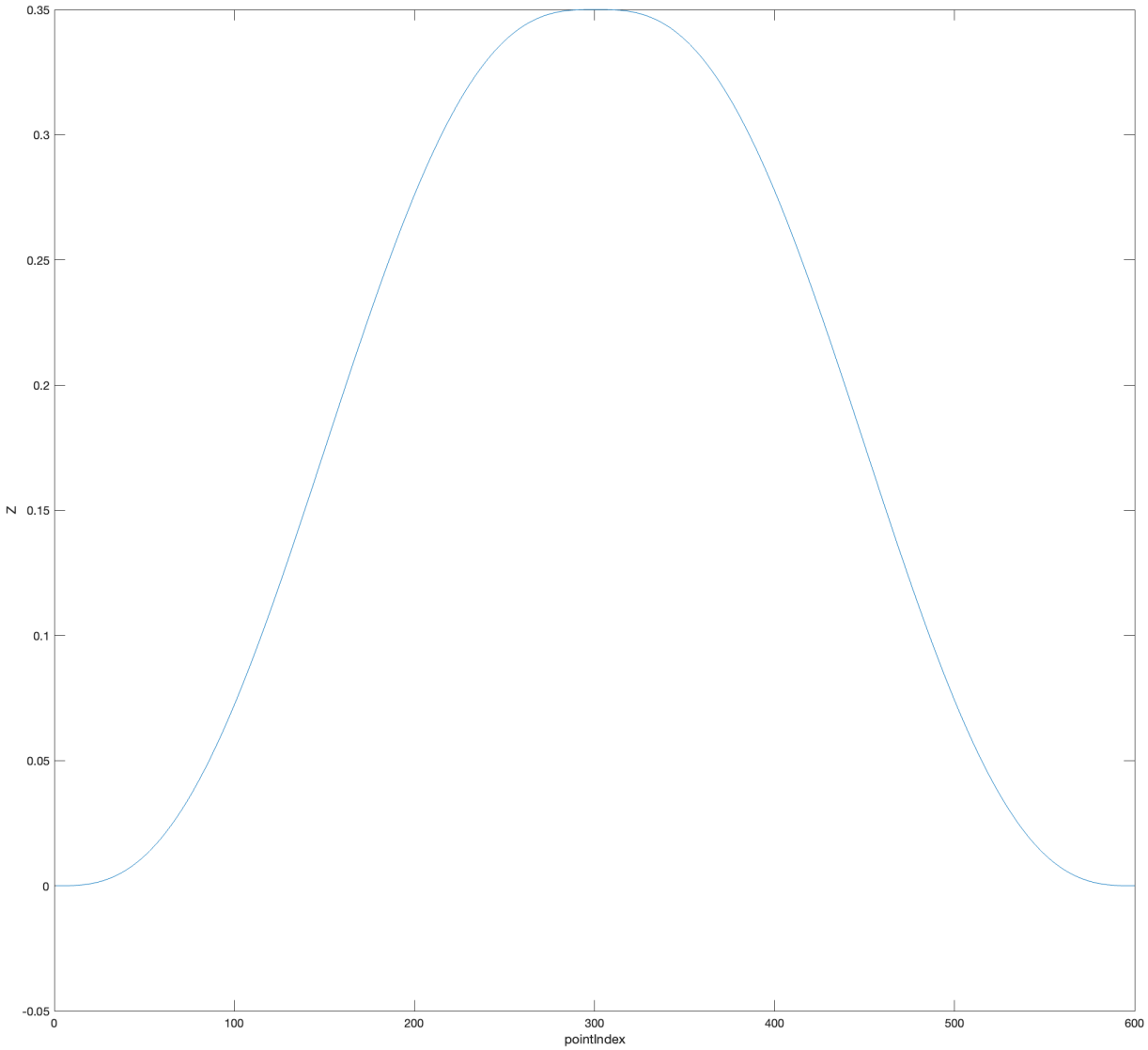
```
hold off
```



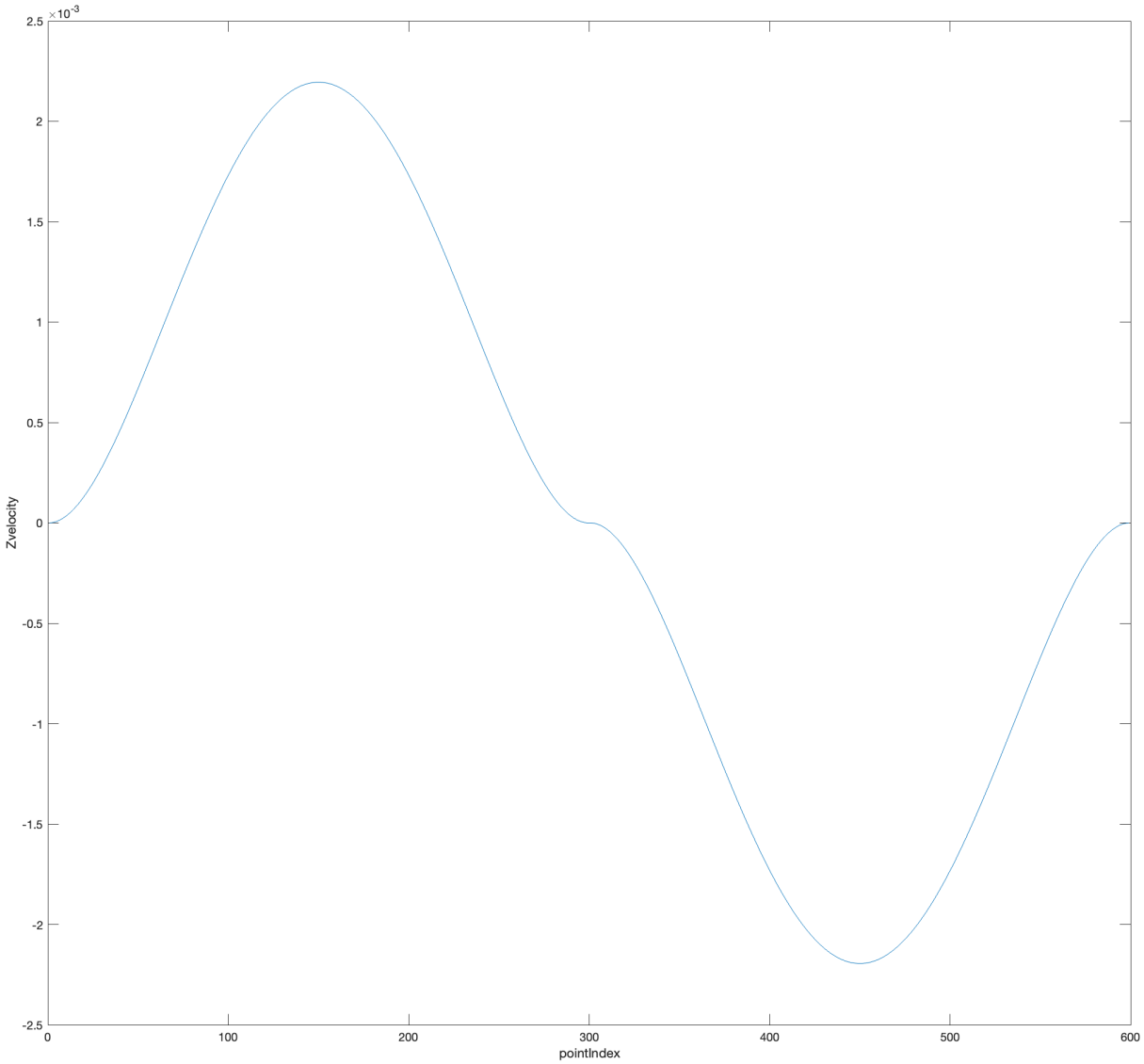

```
 %We calculate every position from the EE reference frame
for i=1:600
    TProf(i,:) = inv(trajPoints(:,:,1)) * [transl(trajPoints(:,:,i)); 1];
end

%trajectory position from EE_RF
plot3(TProf(:,3), TProf(:,1), TProf(:,2))
xlabel('Z')
ylabel('X')
zlabel('Y')
axis([-0.01 0.04 -0.025 0.025 -0.025 0.025])
hold off
```




```
%Z_EE over time
plot([1:600], TProf(:,3))
xlabel('pointIndex')
ylabel('Z')
hold off
```




```
%Z_EE velocity ove time
plot([1:599], diff(TProf(:,3)))
xlabel('pointIndex')
ylabel('Zvelocity')
hold off
```



Trepanation

Prepare a script that perform trepanation. Zoom in the scene and record a video with the best view.

Take into account:

- Use a tool that has the following Transformation: transl(0 0 0.2)
- Use 'trail' option of plot to visualize the trajectory.
- Place a 45º cone on top of the trepanation to better understand.

- See: 6_Plot_Box_Cone_fiducials.mlx. You will have to scale it. Play with transparency.

Answer these questions:

- Display in a figure the lineal displacement, velocity and aceleration of the tool in End Efector Reference frame.
- Display in a figure the manipulabitlity of the trepanation function either for translation and rotation.
- Find an alternate robot location for improving your manipulabitity.

```
%% put your code Here
```

Tumor burning

Prepare a script that perform tumor burning with the laser. Zoom in the scene and record a video with the best view.

You ought to think in an algorithm, that in order, fill up the tumor's equivalent sphere with small burning spheres of 4mm diameter.

Use a tool that has the following Transformation: transl(0 0 0.2)

Answer these questions:

- Display in a figure the lineal displacement, velocity and aceleration of the tool in End Efector Reference frame.
- How long it takes your burning function to burn the tumor

```
%% put your code Here
```