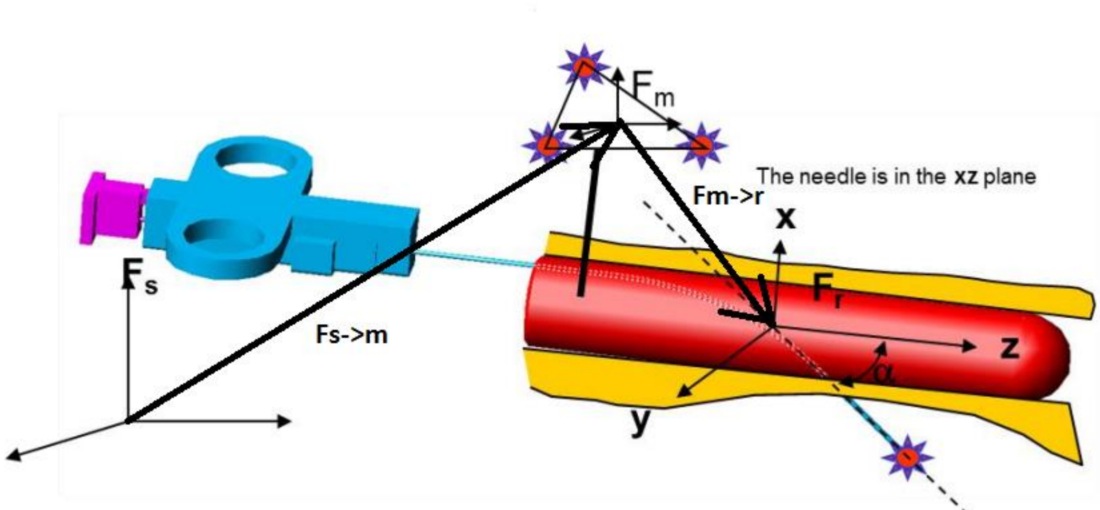
Eric Braun, 10121660

CISC 472

HW 3

March 18, 2016

1.



- The two frame transformations are marked on the diagram above. One for the transformation from scanner frame to marker from (Fs->m) and the other for marker frame to robot frame (Fm->r).
- The formula to transform a point from scanner frame (Fs) to robot frame (Fr) would be:

$$Pr = Fm\text{->}r * Fs\text{->}m * Ps$$

- Where Pr is the point found in robot frame and Ps is the point found in scanner frame.

2. To compute the marker frame (Fm) given the three markers (M1, M2, M3), we will need to define the 3 base vectors (x, y, z) and an origin (center).

- To compute x base vector, we will take the distance between M1 & M2 (M2 – M1).
- To compute z base vector, we will take the cross product of x base vector and the distance between M1 & M3 (M3 – M1).
- To compute y base vector, we will take the cross product of z and x base vectors.
- Take the normal of all three vectors.
- To compute the origin, we will find the center of gravity of three markers (M1+M2+M3)/3.

3. To compute the robot from (Fr) given the three degrees of freedom (1. translation on z-axis, 2. rotation about z-axis, 3. insertion/retraction of needle) and the tracker point on the tip of the needle we will need to define the three orthonormal base vectors and an origin for the system.

- First, we need to define two points to create the z base vector
    - To do this we will spin the device 360 degrees while tracking the markers.
    - We will then map concentric circles of best fit to these 3 sets of data points.
    - Next, we will find the center of these three circles.
        - This will give us a point on the z-axis (ZPoint1).
    - We will shift the device forward in the z direction then repeat this process to get a second point on the axis (ZPoint2).
        - To find the **z-basis vector** we will take the difference of these two points z = (ZPoint2 - ZPoint1).
- Since we are also tracking the tip of the needle, we will take two readings of the needle tip. We will insert the needle a little and take the point (NPoint1), then insert it further to obtain the second point (NPoint2).
    - Taking the difference of these two points will give us the direction vector of the needle NeedleVector = (NPoint2 – Npoint1).
    - Because the needle is on the x-z plane, we can compute the **y-basis vector** by taking the cross produce of the needle direction vector and the z-basis vector.
      y = (cross (NeedleVector, z)).
- Now, we will simply take the cross product of the z basis vector and x basis vector to compute the **x basis-vector.** x = cross (z,y).
- Now take the **norm** of the three basis vectors.
- For the **center point** we will use the center of the robot where the needle comes out. To obtain this we will find the intersection point of the z-axis and the NeedleVector.
    - First we need to create line formulas for the two defined by a point on the line and direction vector:
        - NeedleLine = NPoint1 + t1 * NeedleVector
        - ZLine = ZPoint1 + t2 * z
    - If there is an intersection, the two should meet so set them equal to each other
        - NPoint1 + t1 * NeedleVector = ZPoint1 + t2 * z
    - Now simplify:
        - t1 * NeedleVector = (ZPoint1 – Npoint1) + t2* z
    - Take the cross products to take t2 out:
        - t1 * cross(NeedleVector, z) = cross((ZPoint1 – Npoint1), z)
        - t1 = (cross((ZPoint1 – Npoint1), z)) / (cross(NeedleVector, z)
    - Now insert this value into original equations:
        - Intersection = NPoint1 + t1 * NeedleVector.
- To find the **exit angle** of the needle we will use the dot product and the vectors from the needle and z-axis
    - dot (NeedleVector, z) = norm(NeedleVector) * norm(z) * cos(Angle).
    - cos(Angle) = (dot (NeedleVector, z)) / (norm(NeedleVector) * norm(z)).
    - Angle = arccos ((dot (NeedleVector, z)) / (norm(NeedleVector) * norm(z))).

4. To make each of these processes more robust we can do them multiple times while eliminating outliers.

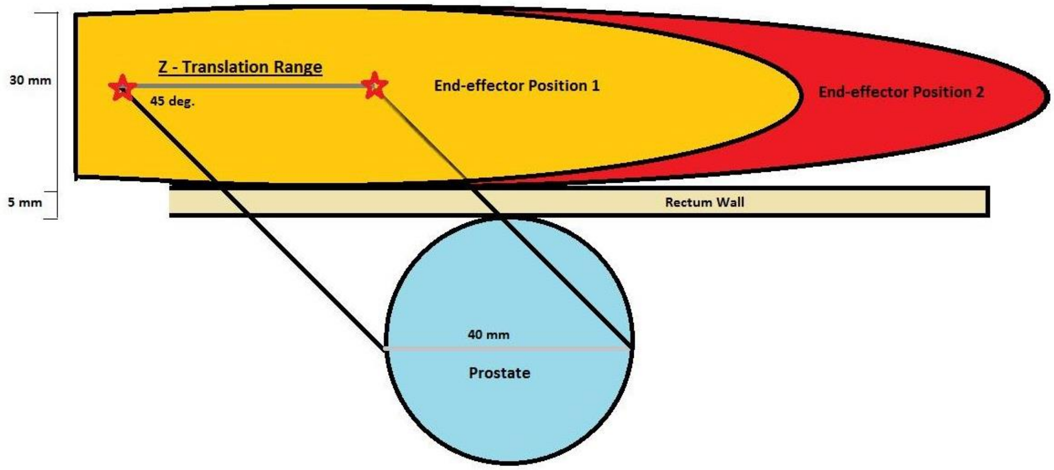
- For finding the points on the z-axis, we could continue making concentric circles with each of the markers then eliminate points or even whole circles that deviated too far away from the average, then take the average without the outliers.
    - We can also move the circles to different z magnitudes by translating the whole robot either in the positive or negative direction.
- For computing the needle vector, we could take multiple points at different insertion/retraction lengths and compare each of the points to each other eventually excluding any out of place values.

5. To create the 4x4 homogenous transformation matrix, we would create an orthonormal basis vector system for marker frame from the 3 tracked markers on the coil. Then, we would define the robot frame using the above process.
    - We would compute the translation and rotation from scanner frame to marker frame.
    - We would compute the translation and rotation from marker to robot frame.
    - The whole transformation matrix would be in the order of the translation from scanner to marker, rotation from scanner to marker, inverse rotation from robot to marker, then finally negative translation from robot to marker.
    - Transformation = -(Tr->m) * (Rr->m)’ * (Rs->m) * (Ts->m)

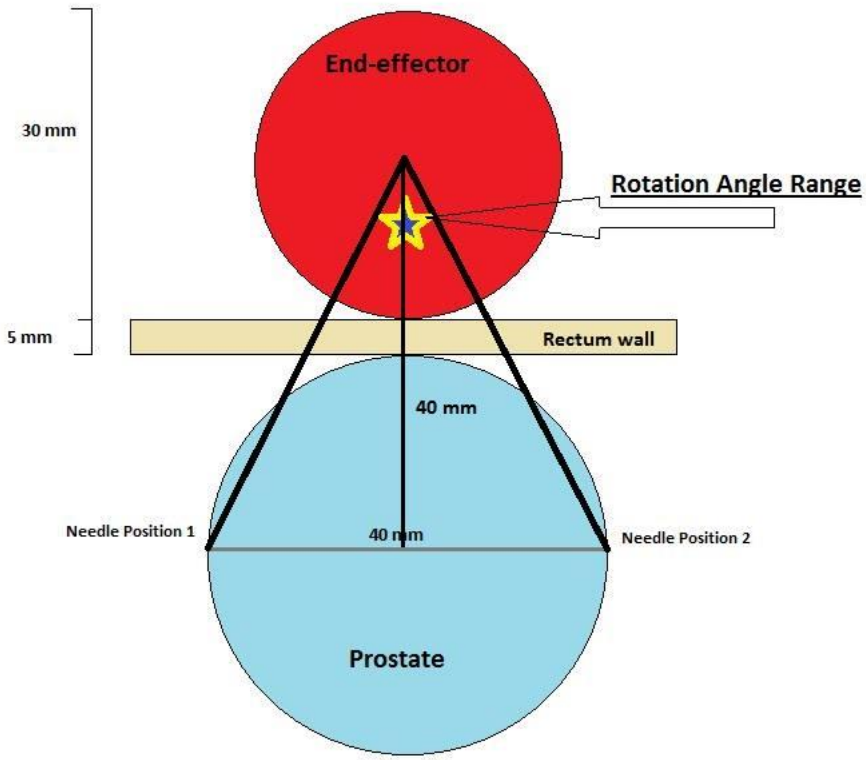
6. Workspace Analysis

- We have to find the minimum range of motion for
    a. Translation about the Z-axis
    b. Rotation about the Z axis
    c. Needle insertion/retraction

    a. The translation about the z-axis should allow us to reach the furthest most point on prostate to the furthest point on the prostate in the z direction.
        i. Because the needle remains at the same angle, this will be the diameter of the prostate. Diameter prostate = 40 mm.
        ii. **Range of z translation = 40 mm**



b. The rotation about the z-axis should allow us to reach the two most extreme points on the prostate sphere in the y direction.
   i. To compute this, we will think of the needle creating a triangle connecting each ends of the prostate to the center of the robot.
   ii. This will create two smaller triangles sharing the bottom diameter of prostate.
      1. Bottom of each = radius of prostate = 40 mm/2 = 20 mm
   iii. The height of both triangles will be the distance from center of end-effector to center of prostate.
      1. Radius of end-effector = 30/2 = 15 mm
      2. Rectum wall = 5 mm
      3. Radius of prostate = 20 mm
         a. 40 mm
   iv. Now, we can take the angle of the tip of one of the triangles.
      1. Tan (angle) = Opposite/Adjacent
      2. Tan (angle) = 20 mm/40 mm
      3. Tan (angle) = 0.5
      4. angle = 26.56 degree
   v. Range of angles around z -axis = -26.56 – +26.56 degrees

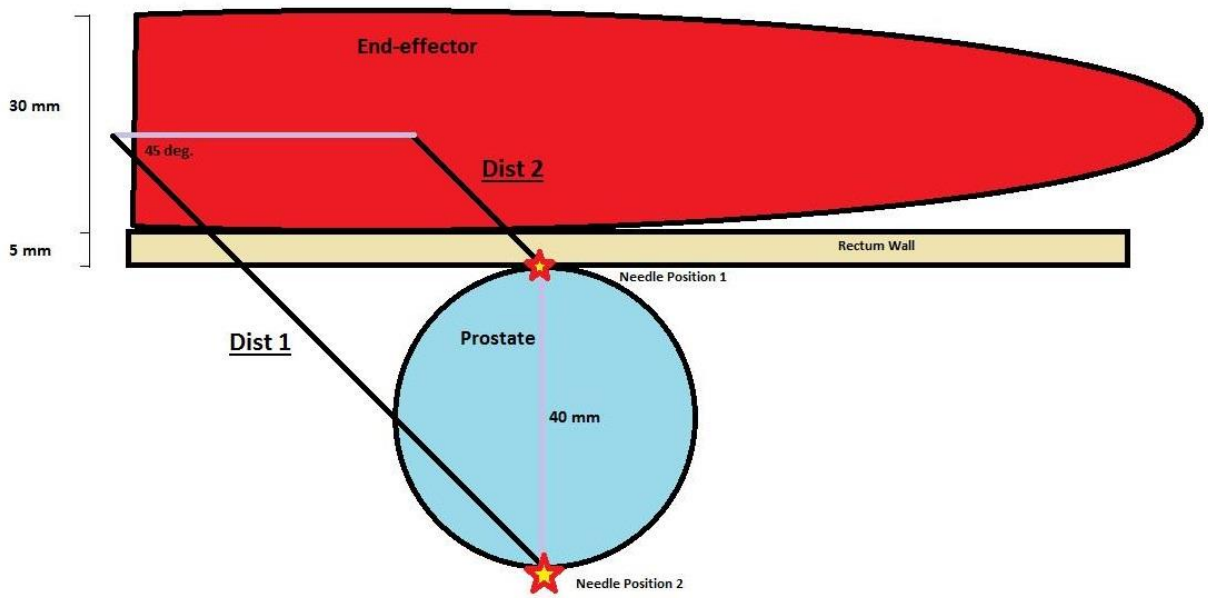


c. The insertion/retraction of the needle should allow us to reach the most extreme points on the prostate in the x direction.
    i. To calculate this, we will move the needle to the top and bottom of the prostate while maintaining 45 degrees.
    ii. This will create two triangles, with a known angle in the top left corner and easily to compute distance of the right side.
        1. The smaller triangle's right side will be the length from the middle of the end-effector to the top of the prostate.
            a. Radius end-effector = 15 mm
            b. Rectum wall = 5 mm
                i. 20 mm
            c. Sin (angle) = Opposite / Hypotenuse
            d. Sin (45) = 20 mm / Length1
            e. Length1 = 20 mm / Sin (45)
            f. Length1 = 28.28 mm
        2. The larger triangle's right side will be the length from the middle of the end-effector to the bottom of the prostate.
            a. Radius end-effector = 15 mm
            b. Rectum wall = 5 mm
            c. Diameter prostate = 40 mm

i. 60 mm

d. Sin (angle) = Opposite / Hypotenuse
e. Sin (45) = 60 mm / Length2
f. Length2 = 60 mm / Sin(45)
g. Length2 = 84.85 mm

iii. **Range of insertion**: **28.28 – 84.85 mm**



7. Forward Kinematics:

To compute the needle position relative to the home position we can use the combination of scalar values given for translation about z-axis, rotation about z-axis and length of needle insertion.

- We will compute our point by the formula: Translation * Rotation * Point.
- First we have to find the needle tip placement from the insertion and angle of the needle.
    - Thinking of these as a triangle we can use the cosine * insertion value to find these coordinates. The x value will be negated because it does down.
        - Point = [-cos(90 – angle) * insertion;
          0 ;
          cos(angle) * insertion ;
          1]
- Now we will construct the rotation matrix for the rotation about the z-axis (R):
        - R =[cos(angle), sin(angle), 0;
          -sin(angle, cos(angle), 0;
          0 , 0 , 1]

      - Add buffer:

        R = [cos(angle), sin(angle), 0, 0;
        -sin(angle, cos(angle), 0 0;
        0 , 0 , 1 0;
        0 , 0 , 0 1]

- Now, we construct the translation matrix about the z-axis
  - T = [1, 0, 0, 0;
    0, 1, 0, 0;
    0, 0, 1, Translation;
    0, 0, 0, 1];
- Finally, our needle tip is made using the formula Translation * Rotation * Point

8. Inverse Kinematics:

To find the series of transformations given a needle point we will reverse the steps we did in the above process.

- First, we find the insertion given an insertion angle. This is done by first computing the distance between the point and the z-axis.
  - distance = square root(Point(x)^2 + Point(y) ^2).
    - Ignore the z-component because they will be the same value.
- Then, using the triangle created by the z-axis, insertion (hypotenuse), and the distance between the point and line we can compute the distance of the hypotenuse using sin
  - sin(angle) = opposite/hypotenuse
  - sin (NeedleAngle) = distance/Insertion
  - **Insertion = distance/sin(NeedleAngle)**
- Next, we can use the same triangle to find the z-translation. The amount that the robot has moved in the z-direction will be the z value of the needle point – the z value of the top of the triangle
  - Cos(angle) = adjacent/hypotenuse
  - cos(NeedleAngle) = Zdistance/Insertion
  - Zdistance = cos(NeedleAngle) * Insertion
  - **Translation = NeedlePoint(z) – Zdistance**
- Finally, we will find the rotation between the needle point translated back by the translation value and a point in the x-z plane where the point would be with no rotation by the same amount of insertion
      - newPoint = [NeedlePoint(1);
        NeedlePoint(2);
        NeedlePoint(3) – Translation;
        1];
  - We will use the same equation as 7 to create a point on a triangle by some distance

        - 0rotationPoint = [-cos(90-NeedleAngle) * Insertion;
                                                   0   ;
                           cos(NeedleAngle) * Insertion   ;
                                                   1   ]
    - Now, we will find the angle between the two vector's x and y components using:
        - dot(0rotationPoint, newPoint) = ||0rotationPoint|| * ||newPoint|| * cos(Rotation).
        - cos(Rotation) = dot(0rotationPoint, newPoint) / (||0rotationPoint|| * ||newPoint||)
            - use the inverse of cosine (arccosine)
        - **Rotation = arccosine(dot(0rotationPoint,newPoint)/(norm(0rotationPoint) * norm(newPoint))**

10. Testing:

I constructed three ground truths to show each of the uncoupled motions.

a. Z-translation
    i. I used the point [0;0;40;1] to show only translation in the z-direction.
b. Z-rotation
    i. I choose the point [0;1;0;1] to show only rotation about the z-axis: this will be rotated 90 degrees so it only lies in the y-z plane. Even though this is not in the workspace this is the easiest example to show.
c. Insertion
    i. I used the point [-40;0;40;1] to show the insertion of the needle. This will be the hypotenuse of the triangle created from these values. It will be the square root of 40^2 + 40^2.

The Assignment3test.m shows the testing of the points as well as the 10 random scalar values of the degrees of freedom and their calculated inverse kinematics values.

As you can see in the output, all the values are the same meaning the InverseKinematics.m code works correctly.

This is the equation I was using to compute the angle between two vectors if it was not clear above:

$$\mathbf{v} \cdot \mathbf{w} = \|\mathbf{v}\| \, \|\mathbf{w}\| \cos \theta$$