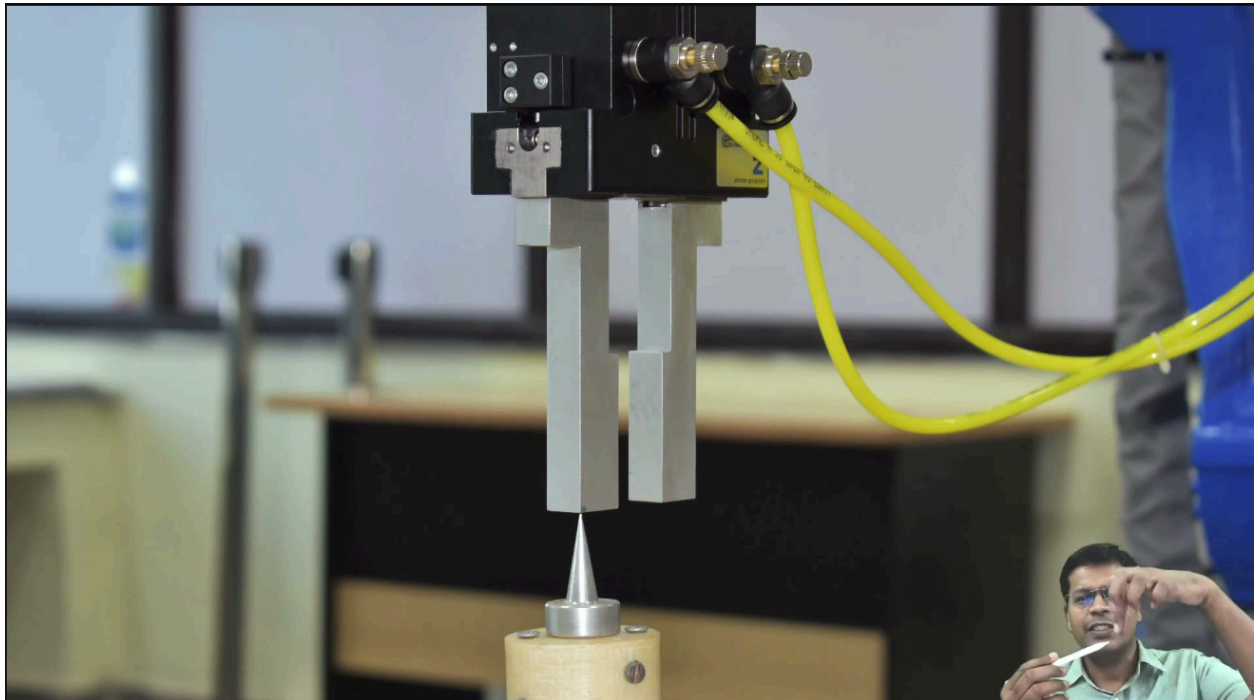


NPTEL Online Certification Courses
Industrial Robotics: Theories for Implementation
Dr Arun Dayal Udai
Department of Mechanical Engineering
Indian Institute of Technology (ISM) Dhanbad
Week: 06
Lecture: 29

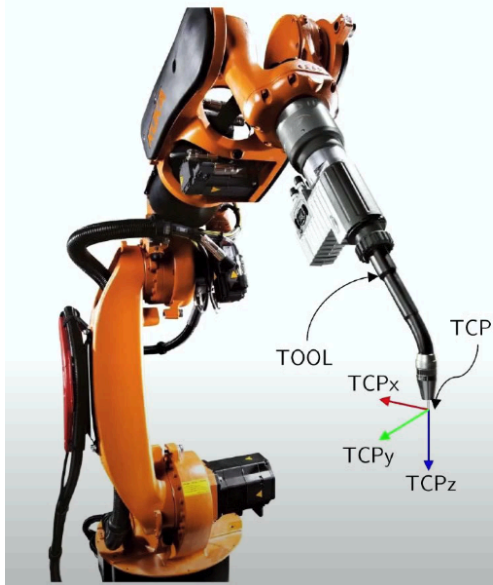
TCP Orientation Calibration using World Frame and Two-Point method

Hello everyone. So, in the last class, what did we do? We stored a gripper at the end of this robot, and we saw it requires tool calibration So we could successfully calibrate the origin of the tool. Okay, I saw you different methods, how it can be done. So, in today's class we will move back further in tool calibration. As we discussed in the last class, it requires two different things to be done. First is the orientation calibration, and next one is the tool end effector frame calibration that actually shows the origin to the controller, Got it? So, two different types of calibration are there as far as the tool is concerned.



I will just show you the representative image that clearly shows the tip of the tool. One corner I have marked it, and you see now I can locate precisely this tool to any reference mark which is there in the workspace of the robot. I can take my robot exactly to that point because that origin is. Now known. But now I want to orient the robot as well with reference to that particular TCP origin. So, that is what we are going to do today. So, let me just move further.

TCP Orientation Calibration



- ▶ It is the orientation of the tool frame with respect to the robot's tool flange.
- ▶ This allows precise estimation and/or control the end-effector tool orientation with respect to the robot base.
- ▶ TCP Orientation is calibrated after the TCP Origin calibration.
- ▶ There are two approaches for TCP orientation calibration that use the existing world frame as a reference **5D and 6D Method**.

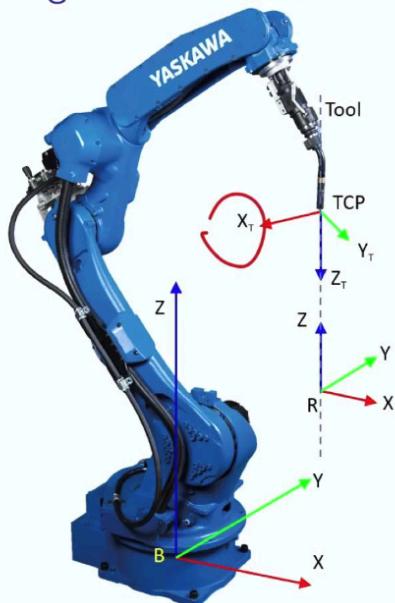


So, it is the orientation of the tool frame with respect to the robot's tool flange. Tool flange is something over here where you actually attach your tool. So, this is what is TCP orientation calibration. So, that will intend to find out the transformation matrix from here to this TCP. That clearly says the rotation along three different axes. And it takes me to the aligned orientation with respect to the base, to this tip. So, that is what orientation calibration is.

So, yes, this allows precise estimation and or control of the end effector tool orientation with respect to the robot base. So, with respect to the robot base, which is here now, I can also orient my end effector with respect to the two tips of this tool. So, the tip can be made stationary, and the whole of the end effector can now rotate about that. So, that the tip will remain stationary. So, after the completion of orientation calibration, we should expect our robot to do that.

So, the TCP orientation is calibrated after TCP origin calibration that is, we want to know this origin first. First, we will translate from this flange to this location, and then I will orient the frame so that it aligns with the tool orientation. So, this is the job that we want to do. So, there are two approaches for TCP orientation calibration that use one existing frame as the reference. So, these are the two methods (we will talk about this) that use world frame as a reference. So, what is the world frame? It is the frame which is attached to the workspace anywhere in the workspace where your robot is standing. If it is just a single robot, the world frame can be placed at the base of the robot, So it is exactly the zero frame of the robot itself. That also can be done. So, yes, this frame now is a free frame, about which I want to understand how my tip of the TCP is oriented. So, that is what our job is. So, these are the two methods. That uses one frame, and the next method is a two-point method for TCP orientation calibration.

Using World Frame: 5D Method



- ▶ The axes of the Tool Frame is aligned parallel to the axes of the World Frame using manual jogging.
- ▶ This communicates the Tool orientation to the robot controller.
- ▶ The tool direction i.e, the Z_T axis of the tool is aligned to the Z axis of the World frame. The axes X_T and Y_T are automatically assigned by the controller and is not important.

Example Applications: Robot welding (MIG/MAG), laser cutting, or Water jet cutting, Painting, Milling, Grinding, 3D Printing processes, etc.

5D?: The rotation about tool Z_T axis is allowed.



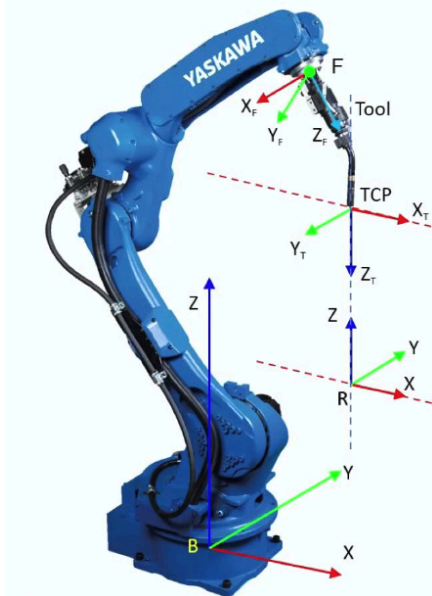
So, let us begin with one of them. The 5D method to begin with. So, what does it say? It uses one frame. So, now you see, this is the reference frame. B is your robot's base frame. Both are parallel to each other. You see, They are just translated by some distance along X, Y and Z, but both of them are parallel to each other. Now, the tool frame is this one. So, what are the steps? The axis of the tool frame is aligned parallel to the axis of the world frame using manual jogging. You jog, jog, jog till this one of the axes, that is, the tool frame axis. I call it the Z axis, normally the tool axis. Different robot manufacturers assign different axes to the tool. I prefer to put the Z axis along the tool axis. So, if it approaches, it will approach in that direction. If it goes away, it will go away along that direction.

This communicates the tool orientation to the robot controller. Now, record whatever the orientation of this flange with respect to this and this tool, how it is oriented. So, whatever this orientation is, this becomes the orientation of my tool with respect to the flange. So, using this as a reference, I understand Z is aligned with Z, so I leave the controller to decide upon X and Y directions. So, this whole of this is now totally constrained, and it is fully known. Once this is fully known, this frame is automatically known by using the robot's forward kinematics. So, from here to here, that transformation is identified by the controller, and that is saved as a tool frame orientation with respect to the robot flange frame. So, this is how it works inside. So, the tool orientation, that is, the Z_T . Z_T is for tool Z the axis of the tool is aligned to the Z axis of the world frame. The axis X_T and Y_T , that is, the tool axis, are automatically assigned by the controller. And it is not very significant here because I should be very, very happy with the Z axis only.

There are many applications which are here to use that particular kind of calibration. A few of them are welding operations. You know, welding can be done just by reaching a point. You can just reach there, and you can do it. It is irrelevant if you rotate along the axis. So, whatever the

axis rotation, axis rotation of the tool, axis rotation is not very relevant here. So, however, you align, it is all the same. Same with laser cutting. So, rotation along the axis of the laser line is irrelevant. It will cut along the same path, same direction, same position. So, again, water jet, cutting or painting. Also, you rotate your tool. Paint spray will go similarly. Okay, milling, grinding, 3D printing. So, all these operations can use this kind of calibration. Why is it 5D? Because rotation about the Z-axis is allowed. Rest, everything is constrained. So, rotation of the Z axis means that out of the 6 constraints that you can put in this tool frame, only one motion is allowed; the rest of everything gets constrained. So, this is allowed. You can rotate the tool about the axis that is X, and Y can have any position with respect to the Z axis. So, that means this rotation is allowed. And that is known as the 5D method of calibration, where these two are aligned.

Using World Frame: 6D Method



- ▶ All axes of the *Tool Frame* are aligned parallel to the axes of the *World Frame* using manual jogging.
- ▶ This communicates the Tool orientation to the robot controller.
- ▶ The tool frame of the *TCP* is aligned to an arbitrary reference marker frame *R* which is parallel to the world frame of the robot as:
 - Z_T is parallel to $-Z$ of the world frame
 - Y_T is parallel to $-Y$ of the world frame
 - X_T is parallel to X of the world frame.
- ▶ This effectively is rotation of the tool frame about X axis of the world frame by 180° .

Example Applications: 5D jobs, Pick-and-Place/Palletizing, Assembly task, etc.

So, let us look at the 6D method now. Here, we are perfectly aligning all the axes. So, all the axes of the tool frame are aligned parallel to the axis of the world frame using manual jobbing. So, you have aligned the Z axis to the Z axis of the world frame, and also you have aligned X to this, X, Y to this, and Y directions are different. So, now the whole of the tool frame is constrained. It is locked with the world frame. Once you define, you align. It is fully locked, Got it? So, that is why it is a 6D method. You have locked everything now. You have already defined the origin, and now you have defined all the axes of the frame. That means you have fully locked. So, it is a 6D method. So, how they are placed is something like this.

- Z_T is parallel to $-Z$ of the world frame
- Y_T is parallel to $-Y$ of the world frame
- X_T is parallel to X of the world frame.

So, you see, the Z axis is parallel to the minus Z of the world frame. The X axis is parallel to the axis of the world. The Y-axis is parallel to the minus Y of the world frame. This is how you have

to align, through manual jobbing only. So, this effectively is the rotation of the tool frame about the X-axis by an angle of 180 degrees. So, you see, this system of tool frame is actually, if you rotate the world frame about X axis by 180 degrees, you will get to this frame, Got it? So, this is an effective transformation that is there. Now, let us do some analysis on this and understand how, internally, it is doing all the calculations. What data is finally saved to the controller so that it gets to know the orientation of TCP with respect to the plans? So, yes, there are many applications. Now, a Few of them are 5D jobs. All the jobs which are 5D in nature, like welding jobs, laser cutting jobs, all those jobs are done with the 6D method of calibration also, Apart from the pick and place task or palletising task assembly task. These tasks specifically require a 6D kind of calibration. So, it is 6D because all the motions are defined, it is constrained, and it is controlled also.

Analysis of World Frame - 6D Method



- ▶ Reference frame \equiv Robot's base frame:
 $\Rightarrow {}^B Q_R \equiv {}^B Q_B \equiv [I]_{3 \times 3}$
- ▶ ${}^B Q_F$ is evaluated using forward kinematics.
- ▶ Upon aligning the TCP to the Reference frame $Rot(X, 180^\circ)$:

$${}^B Q_T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(180^\circ) & \sin(180^\circ) \\ 0 & -\sin(180^\circ) & \cos(180^\circ) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$
- ▶ As ${}^B Q_T = {}^B Q_F {}^F Q_T$ and for rotation matrix $Q^{-1} = Q^T$
- ▶ Identified TCP Orientation with respect to the flange is:

$${}^F Q_T = {}^B Q_F^{-1} {}^B Q_T = [{}^B Q_F]^T \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$
- ▶ This method is quick. **Manual alignment is not very precise.**

So, what does the robot controller do inside once you align it, and how does it find out the transformation from here to here? Your reference frame, with respect to the base-okay, is the same as the base with respect to the base, that is, I3. There is no rotation about any of the axes. So, this is what can be written very well. So, the transformation of reference with respect to the base is equal to I3. That is this matrix, 1110000. So, this is your I3 matrix. Now, based on the plans from here to here, that transformation is evaluated using forward kinematics. So, forward kinematics, what does it have? It has a position of this as well as orientation. That is the if you see the forward kinematic transformation matrix. So, this is your base for the plans. Transformation. This is your end effector position. This is 1. These three are 0. This is your transformation. So, this is already there in your forward kinematics transformation matrix, So you extract it directly from there. You know the joint angles, and You know the d-h parameters of your system, So this can be very well known. Done.

Next, upon aligning the TCP to the reference frame, what you saw is effectively rotation about the x-axis by 180 degrees. That means this is now rotated about the x-axis by an angle of 180 degrees. So, now, how is your TCP oriented? Basically, TCP, with respect to the base, is rotation about the x-axis by an angle of 180 degrees, \cos , \sin minus \sin \cos . This makes it exactly this. So, this is your BQT, That is, the transformation of the tool frame with respect to the base. This is now known. So, now let us come back to your 5-D method. So, once you align this, you don't bother with whatever orientation you are aligning to your tool frame. So, you have made your tool frame in whatever way. So, you have aligned. So, whatever is the position of your tool with respect to this reference. So, it will automatically assign x and y like this, and it will take those data and it will save it. So, it won't let you know that it has placed x and y exactly like this. But, yes, internally, it does it in a similar way. So, now, coming back to this. So, now ${}^B Q_T$, that is transformation, that is, the rotation matrix of the tool with respect to the base. So, the rotation matrix is equal to the transpose of the rotation matrix. So, these two things should be noted. So, moving ahead. So, the identified TCP orientation with respect to the plans now will be: if you say plans to the tool. So, you just have to bring this one to this side, taking inverse, and you are remaining with plans for the tool. That is here. So, it becomes this, Got it?

$${}^B Q_F^{-1} {}^B Q_T$$

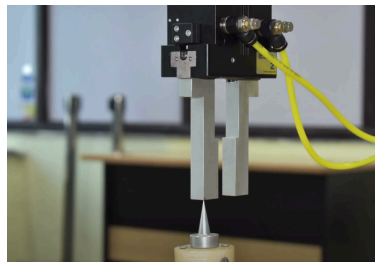
So, what was this? Base to plans, transpose or inverse? The base of a plan matrix is known, and the base of the tool is here. That is also known. You just copy it here and use these two. You get to know what are plans for the tool are. So, this is what is the orientation matrix we were interested in, and it can quickly now be found out. The controller can do it by using this calculation. So, this method is very, very quick. You can quickly align. You can do it. But, because it is a manual alignment, it is not very precise. You have to depend on your eye estimation when you do it. So, you are manually aligning it by some jogging method, So it may not be very precise, but yes, it is quite good for quite a number of applications like tick and place using two-finger grippers. A bit of half a degree here and there is okay. Still, you can hold your object.

Two Point Method for TCP Orientation Calibration



- ▶ Performed after **TCP Origin** is calibrated.
- ▶ By jogging the tool, the external reference point is brushed to the calibrated **TCP Origin**, a point on the Z_T Axis, and a point in the $X_T Z_T$ plane by gradually jogging the tool.
- ▶ Using this procedure the Z_T Axis is assigned by the user, whereas X and Y axes are automatically calculated by the controller with the two points (the point in the $X_T Z_T$ plane and the known TCP position).

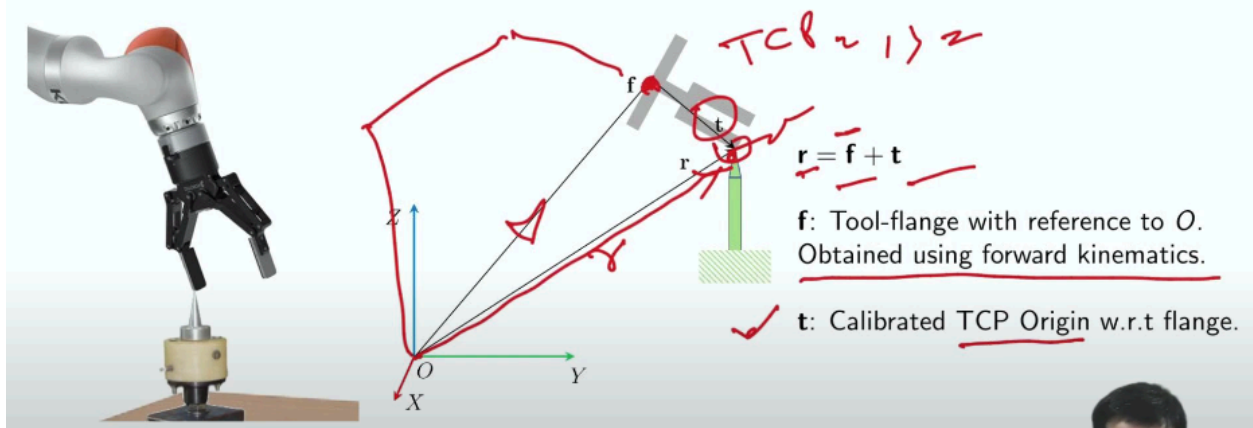
Now, let us do a different method which is more precise. It is known as a two-point method for TCP orientation calibration, So it is performed immediately after TCP origin calibration. Now, you already know the origin of the tool center point.



So, if this is the image of the marked point which is making contact with the reference point that is the origin, now, I have to find out the orientation. So, something that we did just on the last slide. The last two methods: we want to find out the same matrix, Okay. So, now how to do it, how to use this method. By jogging the tool, the external reference point is brushed to the calibrated TCP origin. Okay. So, external reference point, you have to bring it closer to almost touching the TCP origin. Okay, an External reference point, and this should match a point on the Z axis is taken, and you also have to brush the reference point to a point on the Z axis and a point on the $X-Z$ plane of the tool by gradually jogging the tool. So, three points that you have to locate. First, TCP, that is, the TCP origin, should brush against the reference point. Next, you have to brush the reference point to one of the points along the Z axis that you want to make it and another point, which is somewhere on the $X-Z$ plane. Using these three pieces of information, it calculates the unit vectors. Those are aligned along the TCP frame axis. So, that is what will be found out. So, using this procedure, the Z axis is actually assigned by the user itself, where X and Y are automatically calculated by the controller with the two points that you, additionally have made contact with, the reference point, that is, a point on the $X-Z$ plane of the tool, and the known point, TCP position, that is, TCP origin. So, how does it? Let us come to that.

Analysis of Two Point Method

Step 1: The TCP is moved to any arbitrary fixed reference point. This records the position vector or simply, position of the reference point r with respect to the robot's base O .



So the first step is what we did. The TCP is moved to any arbitrary point. This point is the tip of this reference point. So, that point is contracted with the origin of your TCP. This records the position vector, or simply the position of the reference point r with respect to the robot's base O . How does it? Let us just look at., You made a contact over here, Got it This point? So, the robot already knew this location. Using forward kinematics It will quickly calculate the location of this flange. So, it has this information, that is, the tool flange with respect to the origin, that is obtained using forward kinematics. And because this is an origin-calibrated tool for which you have already done the origin calibration, you also know this location with respect to this, that is, the TCPx, TCPy, and TCPz are known. That means this vector is known. With respect to this, that means t is known. So, finally, r . This r vector is calculated using f plus t .

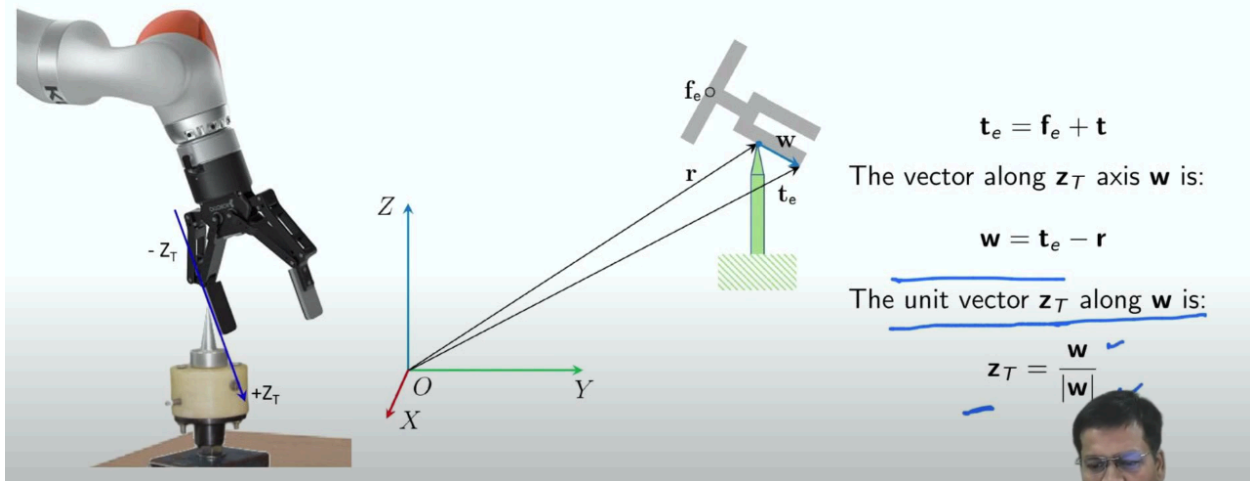
$$r = f + t$$

So, now you know the location of the position vector of your tip of the reference tool. So, now I know the location of this robot base frame because you are doing forward kinematics over here. So, with respect to that, this vector is known. The first job is done.

Analysis of Two Point Method



Step 2: The TCP is now moved so that the reference point is brushed to a point on the Z_T axis which has negative value. The negative values of Z_T commonly lies on the tool, and hence it is more convenient.



Then, your second job. The TCP is now moved so that the reference point is brushed to a point on the Z_T axis. Any line along the line where you want to make it as a Z axis. You have to brush it against one such point on the tool. First, you can contact it here. The second point is here where you are making brushing your reference tool to that particular point which has a negative value. You took it backwards why I will tell you. Because the negative value of Z_T commonly lies on the tool itself, it is easy to locate and hence it is more convenient. So, what you did is something like this. You make contact somewhere here, so now again, t_e is known, that is, the end effector of the tool, and the position is known. Why? Because it is a calibrated tool and uses forward kinematics, you have this information. Forward kinematics you can calculate this. This was a known TCP, so T was known. So, finally, you know what your is this vector t_e . t_e vector is known, so this is known. So, this is the first job that you do when you make contact over here. As soon as you make contact over here, this is the first job you do. Now, the vector along Z_T , that is, the w vector, which is marked as shown in this figure, is basically how much is that w is equal to t_e minus r? Because you know w plus r is equal to t_e .

$$w = t_e - r$$

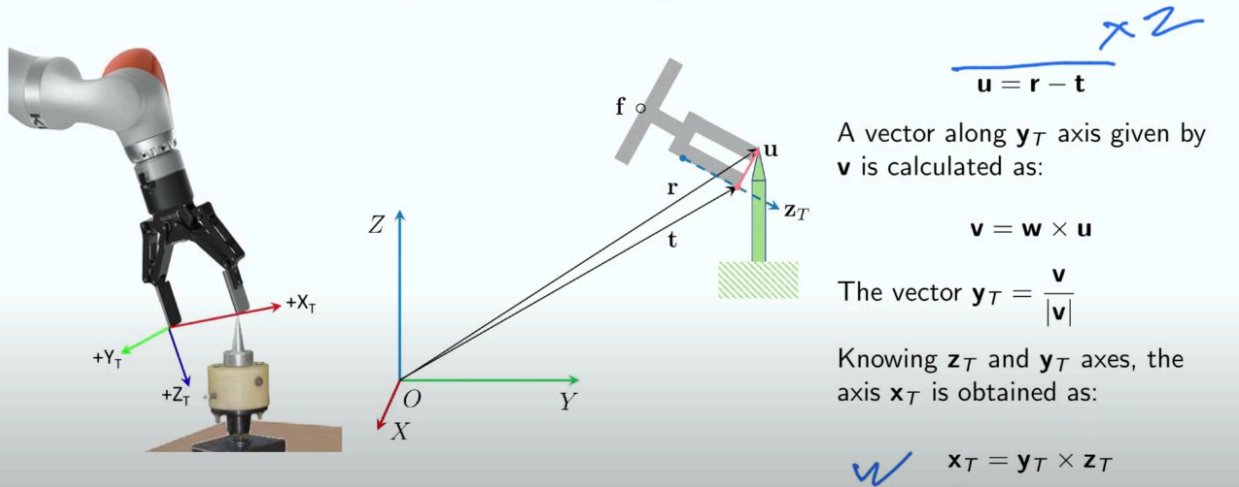
So, just by playing with vector sum you can find out this vector, that is, W and where is this orientation aligned? It is aligned along the Z_T tool Z axis. So, now you can calculate the unit vector Z_T along what that is the tool Z is this W by the magnitude of W. So, it is the unit vector along the Z axis of the tool. The first thing that is found out is step 2.

$$z_T = \frac{w}{|w|}$$

Analysis of Two Point Method



Step 3: The TCP is moved so as to brush the reference point to any point in the $X_T Z_T$ plane. This step is to create u which is vector in $X_T Z_T$ plane, and passes through the TCP.



Now, next, what did you do? The TCP is moved to brush the reference point to any point on the XZ plane, any point on the XZ plane of the tool. So, this is your XZ plane, the plane which is formed by this. So, this is your XZ plane. You can see this. So, this is your plane. So, you have to make contact anywhere on this plane. So, how is it easy to do: you can comfortably take one of the points in the XZ plane which is lying on the tool itself. It is easy to make that reference. So, conveniently, I place it on the other gripper finger, which is lying along the X-axis. I want to make that as an X axis somewhere along that. So, not exactly on the X axis, it will be. It may be somewhere on the XZ plane. So, approximately, I placed it here $+X_T$. Maybe it is a little different because of a bit of misalignment of the two fingers. It may not be exactly along the X-axis, but with two tips, it can be a little like this. So, that is the reason. So, what effectively does it do now? So, this was your Z_T , that is, you calculated in the earlier step, and the u vector is r minus t .

$$u = r - t$$

So, now again, the tooltip is known using forward kinematics and this t vector you calculated the tooltip vector. So, that is using the known tooltip that is here. how much is your r ? r is also known because your reference vector is always known with respect to this. So, r minus t gives you u . u is a vector that lies somewhere near to your X axis of the tool, but it is on the ZX plane. Approximately, it will be somewhere near to the X-axis, but not necessarily that. So, it will be on the ZX plane. So, the vector along the Y_T axis, the tool Y axis is given by v , which is calculated as v is equal to w cross u .

$$v = w \times u$$

So, the u vector is in the ZX plane, and you already have the Z vector. So, a cross product of those two will give you a vector, which is perpendicular to the ZX plane. Okay, perpendicular to the ZX plane, any vector will be parallel to Y_T , that will be parallel to Y_T . So, now I will use v to find out Y_T , that is the tool, Y axis. So, how is that calculated? v by the magnitude of v , that is, that gives you a wound vector parallel to Y_T , so that is there.

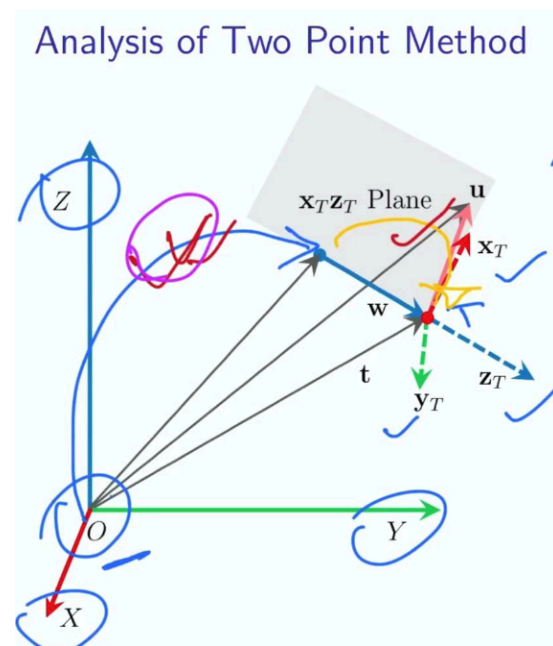
$$Y_T = \frac{v}{|v|}$$

So, this is how Y_T is calculated. We calculated Z_T in the previous step. In this step, you have calculated Y_T . So, X_T can now be obtained precisely. u was not along X_T . So, U was somewhere near to X_T . It was on the XZ plane. But now it is X_T , which is taken as a cross product of Y and Z will exactly give you the X_T unit vector. The unit vector, which is X_T , got it. You can see this equation clearly now.

$$X_T = Y_T \times Z_T$$

So, yes, this is what is obtained. So, you have obtained X_T , Y_T and Z_T , all of them.

Analysis of Two Point Method



All the axes vectors for the TCP frame, i.e., x_T , y_T , z_T are now known in the robot's base frame O .

The orientation of the TCP may now be given by the 3×3 rotation matrix

$${}^O Q_T = \begin{bmatrix} x_T & y_T & z_T \end{bmatrix} \equiv {}^O Q_F {}^F Q_T$$

Which gives the **tool orientation with reference to the flange** as:

$${}^F Q_T = {}^O Q_F^{-1} {}^O Q_T = [{}^O Q_F]^T {}^O Q_T$$

${}^O Q_F$: Obtained using forward kinematics.

Assumption: The orientation of the tool didn't change while jogging during the entire procedure.

So, all the axis vectors X_T , Y_T and Z_T are now known in the robot's base frame. In this frame, all the vectors are known. So, that orientation of TCP may now be written as 3 cross 3 rotation matrices. You know, it is a projection of the tool frame along X, along Y, along Z. Similarly, tool frame Y along X, along Y along Z. Tool frame Z along X, along Y along Z. So, it is nothing but your tool vector projections along X, Y and Z.

So, it is by the fundamentals that you have understood about the rotation matrix and effector rotation matrix. It is exactly this one, okay, so you can quickly, whatever you have calculated, you can put it as X_T along X, X_T along Y, X_T along Z. So, that becomes the first column of this. Similarly, the second column will be Y_T along X, Y_T along Y, Y_T along Z or similarly this one. So, this is your rotation matrix 0 to the tool frame, that is, the tool frame expressed with respect to 0. So, that is the complete transformation now you know, and how much is that? It is 0 to flange, 0 to flange and flange to this. 2 transformations are there. The first one is here. The second one is here. So, using those 2, it can be obtained, what is already known to you because you know X_T , Y_T and Z_T . So, now I will make use of that. So, I will make use of forward kinematics again, 0 to this is already known to you, using forward kinematics. So, this is known. So, you can bring it again using the inverse. So, how much is the flange to a tool? That is this

one (${}^F Q_T$) flange to the tool. It is the inverse of this, the inverse of this (${}^O Q^{-1}_F$) into this (${}^O Q_T$). 0 to the tool is known. This is using known forward kinematics. So, this is calculated. again, the inverse is equal to the transpose of the matrix, so this is there. So, now 0 to F. You know it is obtained using forward kinematics, so this is calculated. So, the only assumption which is here is the orientation of the tool didn't change while jogging during the entire procedure. So, the tool frame should be kept, but one can make this algorithm-you can work on this, make it rugged against that also, but this could be one of the simplest approaches, which can be called a two-point method. TCP origin was already known. You brushed at two additional points, that is, this one and a point over here. So, that is why it is there.

Numeric Input of TCP data



The tool data can be fed manually to the controller.

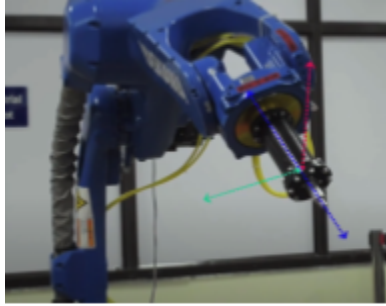
The possible sources of data could be:

- ▶ CAD.
- ▶ External calibration.
- ▶ Manufacturer's technical specifications of the tool.

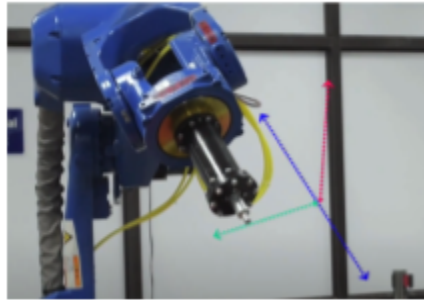
Video Demonstration: Tool frame motions.

So, now there is yet another method, but that is you cannot call it a method here because it is directly fed to the controller using the known data. if the tool is well known to you, you know it is. It is having some shape which is very well known, and you also know the orientation of that, the precise location of your tip with the flange dimensions. You know where it is going to center with the flange and also the tip where you want to work with. So, with that vector, that orientation is well known. You can directly fit those data to the controller. The common source is CAD. If you have a CAD design of your tool. Even if it is a gripper or it is a welding tool, if you know, you can very well know to a very good clarity how your tooltip is oriented and located.

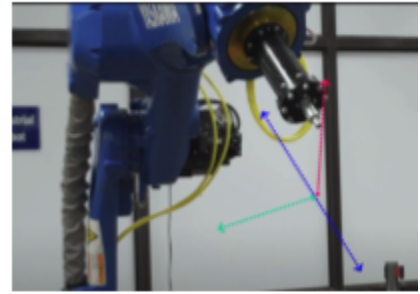
Through some external calibration tool like KUKA track, as we showed you, or there is a tool with ABB also, which is known as a bullseye. They do the same thing. It directly takes the data from the external calibration and feeds it to the controller, or using any manufacturer's technical specification of the tool. If your tool is manufactured by any manufacturer. So,metimes, they do also provide how my tooltip is located with respect to the flange going to place this. So, exactly this information you may be having. So, I will show you some video demonstration now for this. What is the result of doing TCP calibration? That is the tooltip origin calibration and tooltip orientation calibration.



z-axis translation

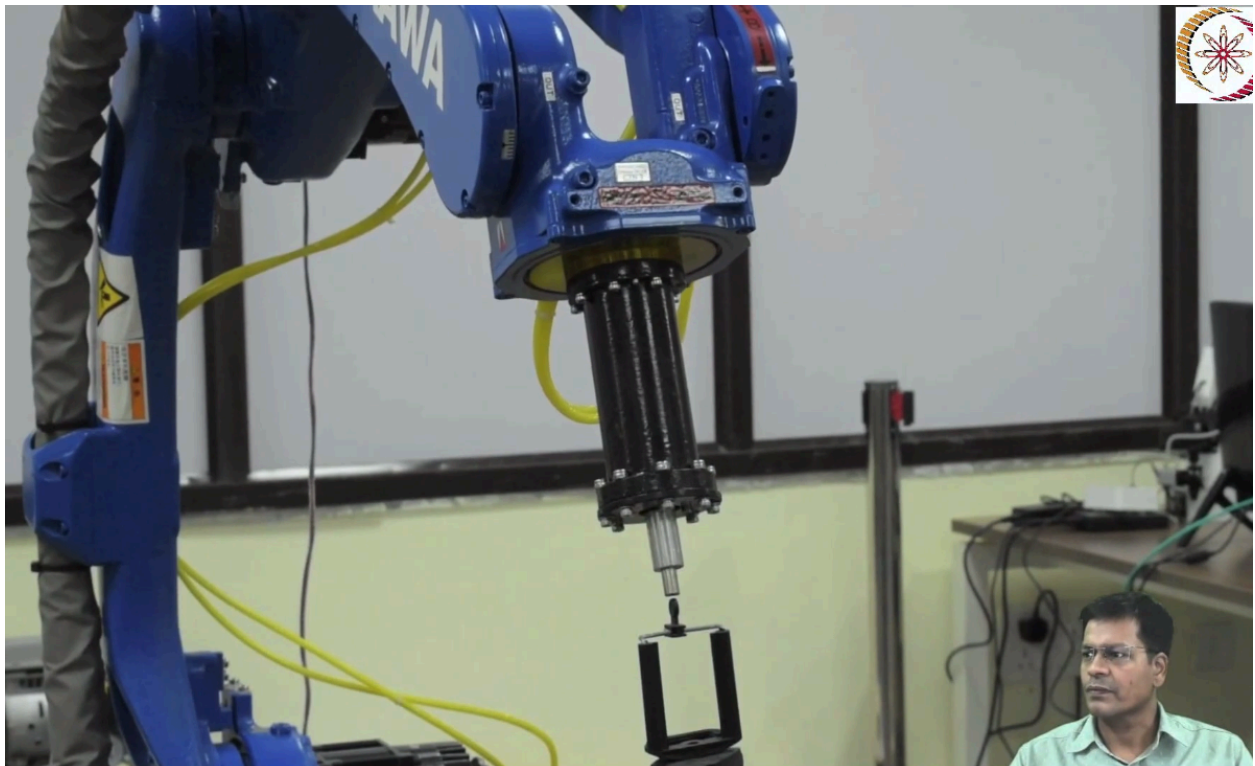


x-axis translation



y-axis translation

So, the first video here will show you a motion in a tool frame. So, now it is moving along the tool centreline axis. It can be the z-axis of the tool, and the other two orthogonal axes are also. I will be moving it that is, along x and along y. Those are orthogonal to this z-axis. This is the second axis motion. These are the translation motions along the tool frame axis. This is yet another axis which is perpendicular to both of them. So, the z axis. You have seen that was the first one. So, these were the three, and also I'll show you one more video which shows rotation about tooltips.



Intentionally, I have kept reference so that you can understand that it is rotating about that. Actually, it is not rotating; and it is trying to make a roll motion and a pitch motion about the tool axis. We'll do that. It will come back. There are some errors over here that are visible as a gap here. Because you know these methods are manually calibrated, you cannot go very close to the reference point also. Now, it is moving like the pitch motion along the tool. Now, I'll do the

z-axis rotation about the tool axis itself. It almost rotates on the axis of the tool itself. So, you see, there is some misalignment, but yes, it is very close to the z-axis.

This is what is the result of doing tool tip calibration and tool frame calibration. You find out the position, you find out the orientation and if that gives the additional capability for the robot to move it in Cartesian space, move the tooltip in the Cartesian space.

So that's all. Thanks a lot. So, in the next class, we'll start with the next model, that is, the calibration of the industrial robot systems, the things which are also there in the workspace that is the object which is there. You have a fixed tool which is there in the workspace: your workpiece, your work surface, your turntable, if it is there. So, the frames which are attached to those external objects should also be known to the robot so that it can move with respect to that also. So, that is what your next module is all about. So, that's all for this lecture. Thanks a lot.