On the alignment of a tool attached to a UR3e cobot by using RoboDK software

Jean-Louis Boimond
University of Angers

This tutorial deals with the alignment of a tool attached to a UR3e cobot (built by Universal Robots) by using the simulation software RoboDK. Most industrial robot manufacturers propose such a functionality under the form of a software instruction: tool alignment consists of orienting the Tool frame, denoted Tool Center Point (TCP) hereafter, so that each of its axes is aligned with one of the axes of the base frame (denoted R_0) of the cobot. More precisely, each axis is aligned with the ***nearest*** to one of the axes of R_0, whether in its positive or negative direction. Note that only the *orientation* of the TCP can change during an alignment, unlike its *position* (in space R^3) which remains unchanged. An example realized with RoboDK is shown in Figure 2 (where X, Y, Z axes are drawn in red, green, blue, respectively) with an arbitrary TCP orientation (before its alignment) at the left of the figure and the TCP orientation after its alignment at the right.

To simplify (but without losing generality), we consider in the following that the TCP corresponds directly to the frame associated with the flange (denoted R_6) of the cobot (it is not equipped with a tool), see Figure 1 (knowing that X, Y, Z axes are drawn in **red**, **green**, **blue**, respectively):

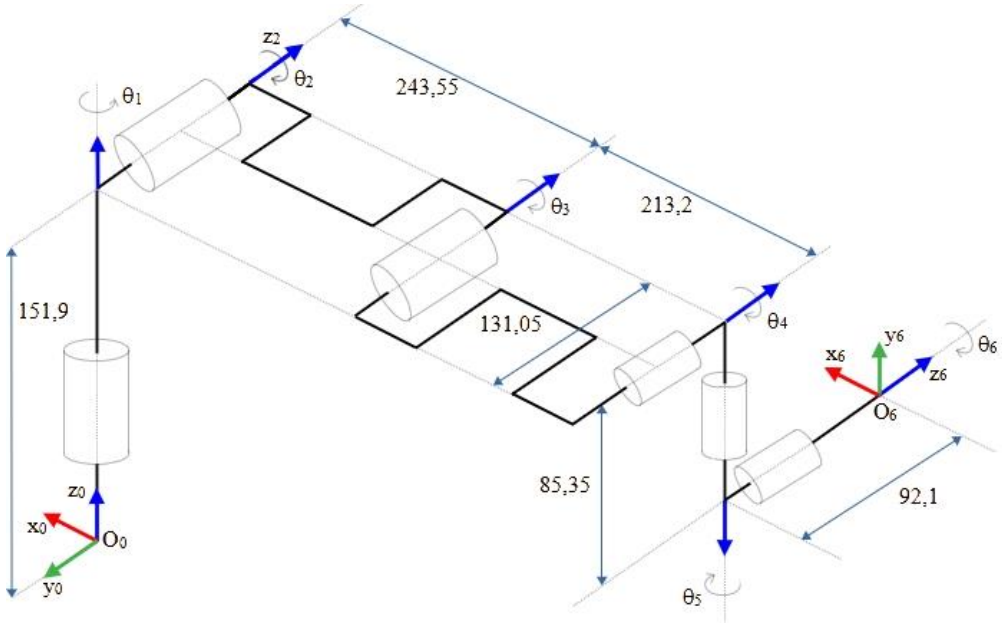


Figure 1: Representation of frames R_0 and R_6 when the UR3e cobot is in its initial configuration (obtained when $\theta_1 = \cdots = \theta_6 = 0$).

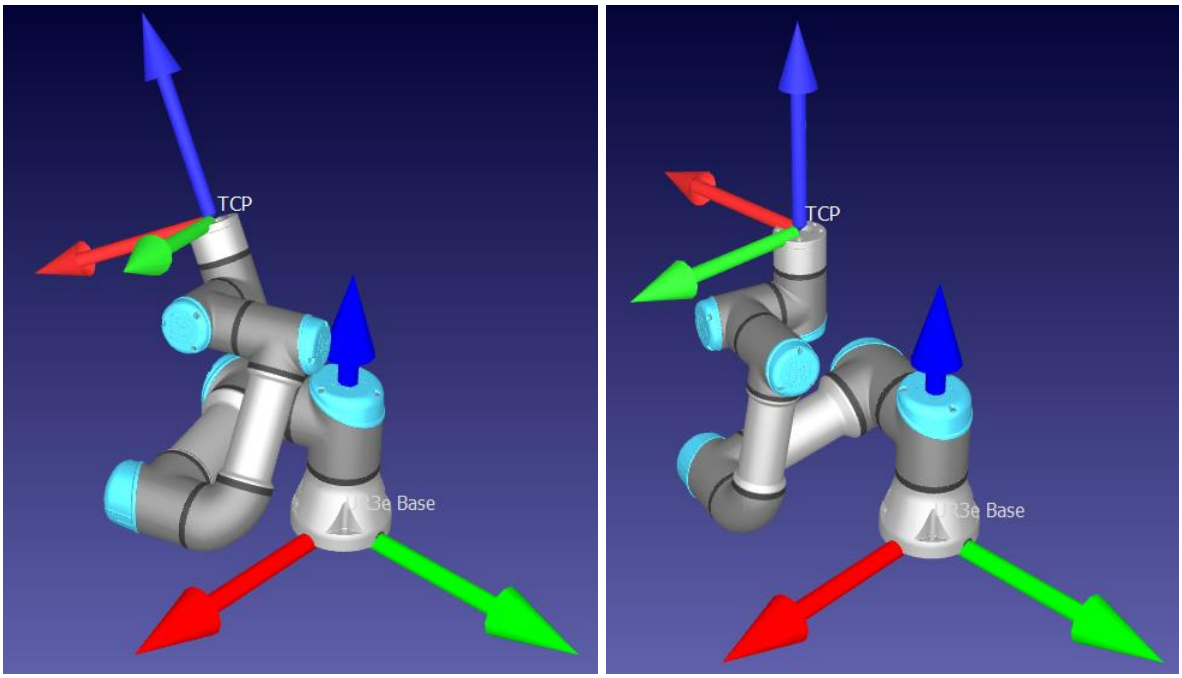


Figure 2: On the left, an arbitrary TCP (corresponding to R_6 frame) situation before its alignment; on the right, TCP situation after its alignment (we see that axes X_6, Y_6, Z_6 are aligned to the axes $-Y_0, X_0, Z_0$ respectively).

Let us reproduce as an example the alignment of the TCP considered in Figure 2. First, let us give some basics of RoboDK software:

- the import of the UR3e cobot is done through the **File>Open robot library** menu, which allows to select the cobot and then, by directing the mouse over the image of the cobot, to click on the **Open** key to realize the import (authorize, if necessary, the request to open the link robodk);
- the **View** menu allows to optimize the display of the cobot, for example:
  - the cobot can be zoomed in or out by pressing and moving the left mouse button or by rolling the mouse wheel;
  - the orientation of the cobot can be changed by pressing and moving the right mouse button;
  - the cobot can be translated by pressing the mouse wheel.

  Note the existence of items **Isometric**, **Top**, **Front**, **Right**, **Left**, **Back** to view in different ways the cobot;
- the **UR3e panel** window that appears (to the right of the main window) by double-clicking on one of the cobot bodies allows to:
  - access (to read or write) the **joint values** $(\theta_1, \cdots, \theta_6)$ of the cobot in the box '*Joint axis jog*', as well as the **TCP situation** (X, Y, Z, u, v, w) in the '*Tool Frame with respect to Reference Frame*' box (knowing that, by default, the Reference Frame corresponds to the cobot base as indicated in the '*Reference Frame with respect to robot base*') box;
  - display **the different frames associated with the cobot** in the '*Show Frames*' box, in particular *via* the checkboxes: '*Base'* to represent the base frame (*i.e.*, R_0) and '*Tool Frame*' to represent the TCP frame (*i.e.*, R_6).

  Remember that we consider that the cobot is not equipped with a tool, as indicated in the '*Tool Frame with respect to robot flange*' box.

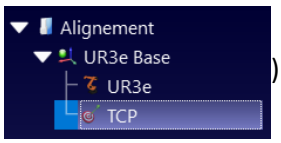
✓ Open the '*Alignement.rdk*' script available <here> in order to visualize (with an isometric view) the cobot as described in Figure 2 on the left. The point reached (denoted `TCP` in the script) corresponds to the TCP before the alignment and has the following joint coordinates:

$$\theta_1 = 17, \theta_2 = -182, \theta_3 = 127, \theta_4 = -27, \theta_5 = 65, \theta_6 = 8 \quad (°),$$

which gives rise to the following coordinates of the TCP situation:

$$X = 73.583, Y = -155.243, Z = 388.824 \ (mm)$$
$$u = 23.345, v = 14.584, w = -62.075 \ (°).$$

✓ Check the validity of this information through the **UR3e panel** window.

✓ Align the TCP by right-clicking on `TCP` (located at the top left in the tree 

) to select '`Align Target orientation`' item. Double-click (again) on `TCP` so that the cobot arm adopts a posture corresponding to the TCP once aligned, as shown in Figure 2 on the right.

✓ Verify that only TCP orientation has changed, note the u, v, w values obtained as a result of the TCP alignment (we will use them later).

Now, let us look at the calculation allowing the alignment of the TCP.

1) Orientation: quaternion, rotation matrix

Before performing a TCP alignment on the UR3e cobot, note that the Universal Robots (UR) company (as well as ABB) uses **quaternions**, rather than the usual (3×3) rotation matrices, to represent the orientation of a point, such as the TCP, defined in task space. So, let us quickly describe quaternions.

Let us first express the expression for the (3×3) rotation matrix, denoted $R(U, \theta)$, by an angle θ about an arbitrary axis carried by a **unit vector** U defined by coordinates (u_x, u_y, u_z) in the base frame R_0, see Figure 3 which follows:

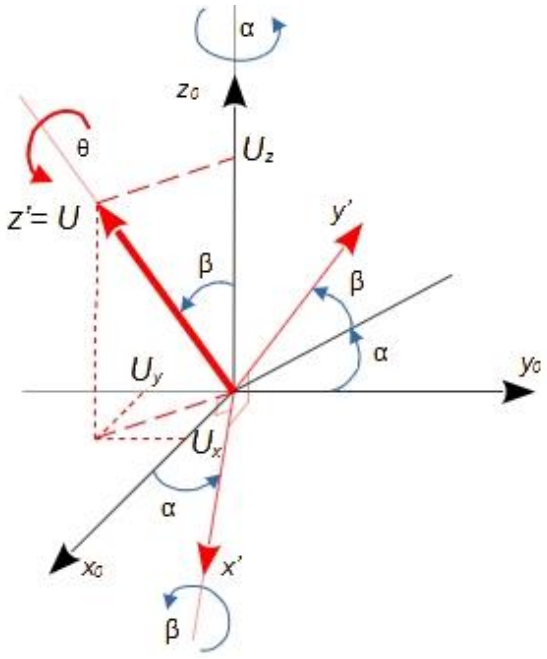


Figure 3: Definitions of vector U and angle θ.

The matrix $R(U, \theta)$ is equal to:

$$\begin{pmatrix} u_x^2(1-\cos(\theta)) + \cos(\theta) & u_x u_y(1-\cos(\theta)) - u_z \sin(\theta) & u_x u_z(1-\cos(\theta)) + u_y \sin(\theta) \\ u_x u_y(1-\cos(\theta)) + u_z \sin(\theta) & u_y^2(1-\cos(\theta)) + \cos(\theta) & u_y u_z(1-\cos(\theta)) - u_x \sin(\theta) \\ u_x u_z(1-\cos(\theta)) - u_y \sin(\theta) & u_y u_z(1-\cos(\theta)) + u_x \sin(\theta) & u_z^2(1-\cos(\theta)) + \cos(\theta) \end{pmatrix}. \quad \text{(eq. 1)}$$

Note: We use the following relation to calculate this matrix:

$$R(U,\theta) = R(z,\alpha)\ R(x,\beta)\ R(z,\theta)\ R(x,-\beta)\ R(z,-\alpha),$$

knowing that:

- $R(x,-\beta)\ R(z,-\alpha)$ enables the frame $(x',y',z'(=U))$ to be confused with the base frame R_0,
- thus $R(z,\theta)$ enables the rotation by an angle θ around the axis z_0,
- then, $R(z,\alpha)\ R(x,\beta)$ enables the modification of the frame (x',y',z') such that z' returns to its initial position (U).

The details of the calculation are given in TP 4 '*Génération de trajectoires rectilignes du Tool Center Point dans l'espace des tâches – Application au robot KUKA KR3*' accessible on my website.

The quaternions Q_1, Q_2, Q_3, Q_4 that describe the orientation resulting from rotation by an angle θ about the axis carried by the (unit) vector U, are given by the following relations:

$$\begin{aligned} Q_1 &= Cos(\theta/2), \\ Q_2 &= u_x\ Sin(\theta/2), \\ Q_3 &= u_y\ Sin(\theta/2), \\ Q_4 &= u_z\ Sin(\theta/2). \end{aligned}$$

Note: It is possible to do the sum, the product of quaternions; for example, the quaternion $(Q_1, Q_2, Q_3, Q_4) = (1,0,0,0)$ corresponds to a rotation by zero angle about any axis.

- **Rotation matrix corresponding to a quaternion:**

The tool alignment method presented here - applied to the UR3e cobot - is based on rotation matrices to represent the orientation of points located in task space, so it is first necessary to establish the rotation matrix corresponding to a quaternion.

To do this, consider a rotation matrix, denoted A, that defines the orientation of a point P in task space, such that:

$$A = (s \quad n \quad a) = \begin{pmatrix} s_x & n_x & a_x \\ s_y & n_y & a_y \\ s_z & n_z & a_z \end{pmatrix},$$

where s, n, a represent the orientation vectors of the point P (s_x, s_y, s_z being the projections of the vector s in the base frame R_0, and the same for the vectors n and a).

Question 1: Consider that the rotation matrix A corresponds to the quaternion defined earlier, *i.e.*, $A = R(U,\theta)$. Knowing that $Cos(\theta) = Cos^2\left(\frac{\theta}{2}\right) - Sin^2\left(\frac{\theta}{2}\right)$, show that:

$$\begin{aligned} Q_1^2 &= \frac{1}{2}(1 + Cos(\theta)), \\ Q_2^2 &= \frac{1}{2}u_x^2(1 - Cos(\theta)), \\ Q_3^2 &= \frac{1}{2}u_y^2(1 - Cos(\theta)), \\ Q_4^2 &= \frac{1}{2}u_z^2(1 - Cos(\theta)). \end{aligned}$$

From equation 1, we can deduce the following expression for the rotation matrix A:

$$A = \begin{pmatrix} 2\left(Q_1^2 + Q_2^2\right) - 1 & 2(Q_2Q_3 - Q_1Q_4) & 2(Q_2Q_4 + Q_1Q_3) \\ 2(Q_2Q_3 + Q_1Q_4) & 2\left(Q_1^2 + Q_3^2\right) - 1 & 2(Q_3Q_4 - Q_1Q_2) \\ 2(Q_2Q_4 - Q_1Q_3) & 2(Q_3Q_4 + Q_1Q_2) & 2\left(Q_1^2 + Q_4^2\right) - 1 \end{pmatrix}.$$

Question 2: Verify that this expression is correct for the elements $A(1,1)$ and $A(2,1)$ (given that the other elements of the matrix A are calculated in a similar way).

- **Special case of quaternion representation for Universal Robots:**

In fact, Universal Robots represents a quaternion, corresponding to the matrix $R(U,\theta)$, using only three values, denoted u,v,w (instead of four, *i.e.*, Q_1,Q_2,Q_3,Q_4). To do this, the magnitude of the vector $U=(u,v,w)$ is equal to the value of the angle θ (it is no longer necessarily unitary), *i.e.*:

$$\theta = \|U\|,$$

which leads to the following values of u_x,u_y,u_z:

$$u_x = u/\|U\|, u_y = v/\|U\|, u_z = w/\|U\|.$$

Question 3: Program a MatLab function called '`CosinusDirecteur_a_partir_des_quaternions_uvw`' with the following header:

```
function A = CosinusDirecteur_a_partir_des_quaternions_uvw(u,v,w),
```

that calculates the rotation matrix A corresponding to the quaternion u,v,w.

Verify that you obtain $A=\begin{pmatrix} 0.4499 & 0.8931 & 0.0029 \\ -0.8010 & 0.4049 & -0.4410 \\ -0.3951 & 0.1961 & 0.8975 \end{pmatrix}$ when $(u,v,w) = (23.345, 14.584, -62.075)$ in degree, which corresponds to TCP orientation considered in the example corresponding to Figure 2 on the left.

2) Tool alignment procedure

Two steps are considered:

2.1) Calculation of the TCP orientation matrix *after alignment* from an arbitrary (unaligned) TCP orientation

To simplify the notations, the point representing the TCP *before its alignment* is denoted '*initial point*', it is denoted *'final point' after its alignment*. Let A^{ini}, A^{fin} be the orientation matrices of the *initial* and *final points*, respectively.

The calculation of the orientation matrix A^{fin} is done by considering each of its three columns separately. Consider, for example, the first column of the matrix A^{ini} (the reasoning being the same for the other two columns): let i (between 1 and 3) be the number of the row whose content (a real number between -1 and 1) has the largest absolute value, then the matrix A^{fin} is such that:

- $A^{fin}(i,1) = \begin{cases} 1 & \text{if the real number is positive,} \\ -1 & \text{otherwise,} \end{cases}$
- The value of the content of the 2 other rows (in the first column of A^{fin}) is equal to 0.

Thus the matrix A^{fin} is composed of $-1, 0$ or $+1$.

Question 4: Program a MatLab function called '`Orientation_finale`' with the following header:

```
function A_finale = Orientation_finale(A_initiale),
```

that calculates matrix A^{fin} from a matrix A^{ini}. Take care in the function to test if the matrix A^{fin} represents a rotation matrix (*i.e.*, is orthogonal and has a determinant equal to 1).

In relation to the example shown in Figure 2, verify that the TCP orientation *after alignment* corresponds to the matrix $A^{fin}=\begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ when $A^{ini}=\begin{pmatrix} 0.4499 & 0.8931 & 0.0029 \\ -0.8010 & 0.4049 & -0.4410 \\ -0.3951 & 0.1961 & 0.8975 \end{pmatrix}$.

Check that the values u,v,w obtained after the TCP alignment (using the '*Alignment.rdk*' script, see pages 2, 3) match the matrix A^{fin}.

2.2) Calculation of the rotation matrix used to move from the orientation matrix of the *initial* point to that of the *end* point

The aim is to determine the rotation matrix, denoted $R(V,\gamma)$, by an angle γ about an axis carried by a **unit vector** V of coordinates (v_x, v_y, v_z) in R_0 (like the matrix $R(U,\theta)$ described by eq. 1) which allows the transition from the matrix A^{ini} to the matrix A^{fin}, *i.e.*, from the frame representing the *initial* orientation of the TCP to the one representing its *final orientation*.

Question 5: Determine the matrix $R(V,\gamma)$. To do this, let us recall the difference in interpretation between $R \times A$ and $A \times R$ when R is a rotation matrix and A is a matrix representing the orientation of a frame:

- concerning the expression $R \times A$, for example, $R_{0,1} \times (s \quad n \quad a)$ allows to make a **change of frame** in the sense that it allows to calculate in the frame R_0 the coordinates of the vectors s, n, a expressed in the frame R_1 (in my course, $R_{0,1} \times \overrightarrow{O_1P}$ is used to express in the frame R_0 the coordinates of the vector $\overrightarrow{O_1P}$ expressed in the frame R_1 (denoted vector $\overrightarrow{O_1P}_{|R_1}$ in my course));
- concerning the expression $A \times R$, for example, $(s \quad n \quad a) \times R(U,\theta)$ allows the vectors s, n, a **to be rotated** by an angle θ about the axis carried by the vector U (unlike the previous case, the vectors s, n, a and p, q, r with $(p \quad q \quad r) = (s \quad n \quad a) \times R(U,\theta)$ are expressed in the same frame).

Question 6: Program a MatLab function called '`Rotation_V_gamma`' with the following header:

```
R_V_gamma = Rotation_V_gamma(A_initiale,A_finale),
```

that calculates matrix $R(V,\gamma)$ from matrices A^{ini} and A^{fin}.

In relation to the example shown in Figure 2, calculate the rotation matrix $R(V,\gamma)$ from matrices $A^{ini} = \begin{pmatrix} 0.4499 & 0.8931 & 0.0029 \\ -0.8010 & 0.4049 & -0.4410 \\ -0.3951 & 0.1961 & 0.8975 \end{pmatrix}$ and $A^{fin} = \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$. Verify that the matrix after the rotation by $R(V,\gamma)$ corresponds to the matrix $A^{fin} = \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ when matrix $A^{ini} = \begin{pmatrix} 0.4499 & 0.8931 & 0.0029 \\ -0.8010 & 0.4049 & -0.4410 \\ -0.3951 & 0.1961 & 0.8975 \end{pmatrix}$.

To go further regarding the generation of the trajectory between the *initial* point and the *end* point (reached at the instant t_{final}), it would be necessary to have the matrices $A(t)$ corresponding to the TCP situation at the instant t with $A(0) = A^{ini}$ (for the *initial* point) and $A(t_{final}) = A^{fin}$ (for the *final* point). To do this, it is necessary to compute the vector V and the angle γ (using equation 1 and the matrix expression $R(V,\gamma)$ from question 5) in order to perform an interpolation based on the variation of the angle γ over time, *i.e.*, $\gamma(t)$; refer to TP 4 '*Génération de trajectoires rectilignes du Tool Center Point dans l'espace des tâches – Application au robot KUKA KR3*' (accessible on my web page) for calculation details.