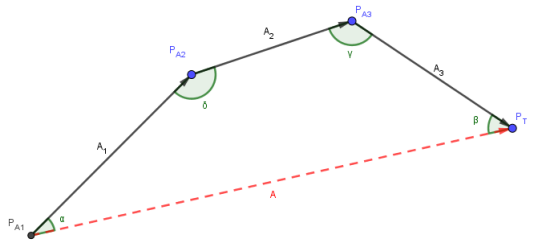
SyArm Positioning

Samuel Nösslböck

April 2022

1 Introduction

The SyArm is build up from two arms and a tong, represented by three vectors.



The robot is controlled by a Raspberry PI 3 with a simple systemd service which keeps the main python script alive.

The main positioning function just receives the three dimensional vector $\vec{A}$ and has to calculate all for the positioning relevant servo angles.

So the control script knows:

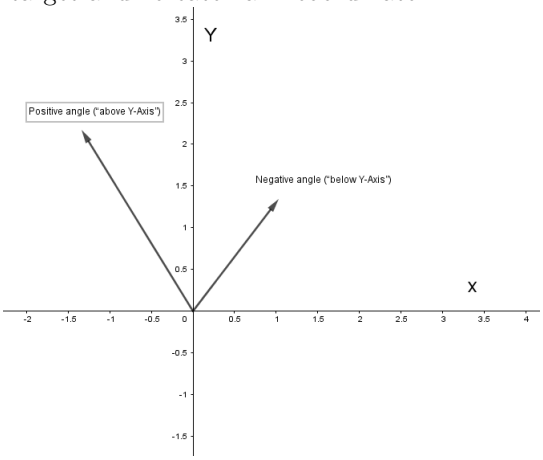
- All lengths $\vec{A}, A_{10}, A_{20}, A_{30}$
- Another property $\delta = \gamma$

An should calculate

- The servo angles $\phi_B, \phi_{A1}, \phi_{A2}, \phi_{T1}$
- The arm vectors $\vec{A_1}, \vec{A_2}, \vec{A_3}$

2 Base Angle ϕ_B

As the unmodified vectors all point along the Y-Axis, the base angle is the only angle that can move all arms towards the target and "create" a X-coordinate.

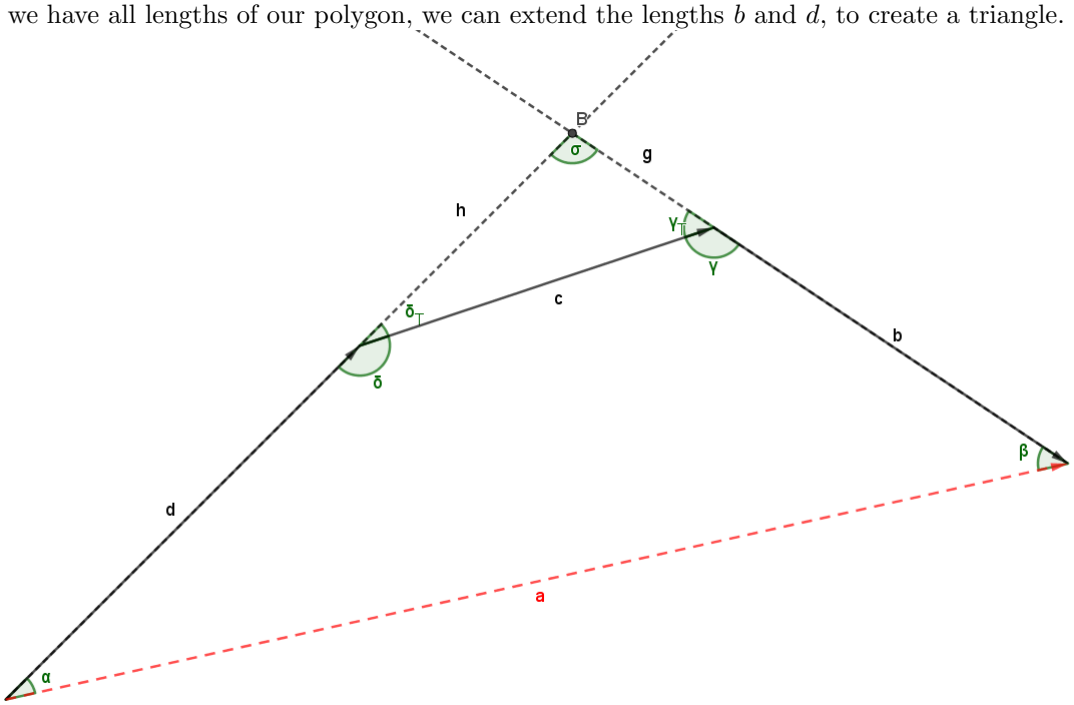


Because of this default Y-Axis, the system for a "positive" base angle is rotated by 90 degrees. Which means $\arcsin(y)$ cannot be applied here, the fully defined arc-function for this case is

$$\phi_B(\vec{A}) = \begin{cases} \arcsin(-\hat{x}) & \textit{if } \hat{y} > 0 \\ \pi + \arcsin(-\hat{x}) & \textit{otherwise} \end{cases} \qquad \hat{x} = \frac{\vec{A}_x}{\sqrt{\vec{A}_x^2 + \vec{A}_y^2}}, \quad \hat{y} = \frac{\vec{A}_y}{\sqrt{\vec{A}_x^2 + \vec{A}_y^2}} \tag{1}$$

3 Arm angles

To simplify the following equations simpler labels have been assigned to the following labels:
$a = \vec{A}, \quad b = \vec{A_3}, \quad c = \vec{A_2}, \quad d = \vec{A_1}$
As we have all lengths of our polygon, we can extend the lengths b and d, to create a triangle.



Considering the property $\delta = \gamma$ the following equations form

$$360 = \alpha + \beta + \gamma + \delta x \tag{2}$$

$$180 = \alpha + \beta + \sigma = \delta_T + \gamma_T + \sigma \tag{3}$$

$$180 = \delta + \delta_T = \gamma + \gamma_T \tag{4}$$

$$\delta_T = \gamma_T \quad g = h \tag{5}$$

As there are now two triangles, one of them being an isosceles one, the law of cosines can be applied

$$c^2 = h^2 + g^2 - 2gh\cos(\sigma) = 2g^2(1 - \cos(\sigma)) \tag{6}$$

$$a^2 = (d+g)^2 + (b+g)^2 - 2(d+g)(b+g)\cos(\sigma) \tag{7}$$

Out of equation (6), the cosine of sigma can be calculated with

$$\cos(\sigma) = 1 - \frac{c^2}{2g^2} \tag{8}$$

When this result is inserted into equation (7) we can solve the unknown length g

$$\begin{gathered} a^2 = (d+g)^2 + (b+g)^2 - 2(d+g)(b+g)(1 - \tfrac{c^2}{2g^2}) \\ 0 = d^2 + 2dg + g^2 + b^2 + 2bg + g^2 - 2db + \tfrac{dbc^2}{g^2} - 2dg + \tfrac{dc^2}{g} - 2gb + \tfrac{bc^2}{g} - 2g^2 + c^2 \\ 0 = (-a^2 + b^2 + c^2 + d^2 - 2db) + \tfrac{1}{g}(dc^2 + bc^2) + \tfrac{1}{g^2}(dbc^2) \\ 0 = g^2(-a^2 + b^2 + c^2 + d^2 - 2db) + g(dc^2 + bc^2) + dbc^2 \end{gathered} \tag{9}$$

For calculation, some helper variables are introduced

$$\begin{gathered} a_C = -a^2 + b^2 + c^2 + d^2 - 2db \\ b_C = dc^2 + bc^2 \\ c_C = dbc^2 \end{gathered} \tag{10}$$

$$g_{1,2} = -\frac{b_C}{2a_C} \pm \sqrt{(\frac{b_C}{2a_C})^2 - \frac{c_C}{a_C}} \tag{11}$$

When having the first result for g [g_1], equation (8) can be used to calculate σ, γ and δ

$$\begin{aligned} \sigma &= \arccos(1 - \tfrac{c^2}{2g^2}) \\ \gamma_T &= \tfrac{180-\sigma}{2} \\ \gamma = \delta &= 180 - \gamma_T \end{aligned} \tag{12}$$

When now having σ and the length of g, the law of sines can be applied

$$\frac{a}{\sin(\sigma)} = \frac{d+h}{\sin(\beta)} = \frac{b+g}{\sin(\alpha)} \tag{13}$$

So α and β result in

$$\alpha = \arcsin(\frac{\sin(\sigma)(b+g)}{a}) \tag{14}$$

$$\beta = \arcsin(\frac{\sin(\sigma)(d+h)}{a}) \tag{15}$$

4 Arm vectors

The points of each joint and the final position vector can be calculated by using the default vectors

$$\vec{A_{10}} = \vec{A_1}(\phi_B = 0, \phi_{A1} = 0) = \begin{bmatrix} 0 \\ A_{10} \\ 0 \end{bmatrix} \tag{16}$$

$$\vec{A_{20}} = \vec{A_2}(\phi_B = 0, \phi_{A1} = 0, \phi_{A2} = 0) = \begin{bmatrix} 0 \\ A_{20} \\ 0 \end{bmatrix} \tag{17}$$

$$\vec{A_{30}} = \vec{A_3}(\phi_B = 0, \phi_{A1} = 0, \phi_{A2} = 0, \phi_{A3} = 0) = \begin{bmatrix} 0 \\ A_{30} \\ 0 \end{bmatrix} \tag{18}$$

and rotating around the servo axis using rotation matrices. The base rotation is represented by a Z-axis rotation, the joints by a X-axis rotation

$$\vec{A_1}(\phi_B, \phi_{A1}) = R_Z(\phi_B) \cdot R_X(\phi_{A1}) \cdot \vec{A_{10}}; \quad P_{A1} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad P_{A2}(\phi_B, \phi_{A1}) = P_{A1} + \vec{A_1}(\phi_B, \phi_{A1}) \tag{19}$$

$$\vec{A_2}(\phi_B, \phi_{A1}, \phi_{A2}) = R_Z(\phi_B) \cdot R_X(\phi_{A1}) \cdot R_X(\phi_{A2}) \cdot \vec{A_{20}} \quad P_{A3}(\phi_B, \phi_{A1}, \phi_{A2}) = P_{A2} + \vec{A_2}(\phi_B, \phi_{A1}, \phi_{A2}) \tag{20}$$

$$\begin{aligned} \vec{A_3}(\phi_B, \phi_{A1}, \phi_{A2}, \phi_{A3}) &= R_Z(\phi_B) \cdot R_X(\phi_{A1}) \cdot R_X(\phi_{A2}) \cdot R_X(\phi_{A3}) \cdot \vec{A_{30}} \\ P_{T1}(\phi_B, \phi_{A1}, \phi_{A2}, \phi_{A3}) &= P_{A3} + \vec{A_3}(\phi_B, \phi_{A1}, \phi_{A2}, \phi_{A3}) \end{aligned} \tag{21}$$

Taking all of these vectors, the position vector $\vec{A}$ can be calculated

$$\begin{aligned} \vec{A}(\phi_B, \phi_{A1}, \phi_{A2}, \phi_{A3}) &= \vec{A_1}(\phi_B, \phi_{A1}) + \vec{A_2}(\phi_B, \phi_{A1}, \phi_{A2}) + \vec{A_3}(\phi_B, \phi_{A1}, \phi_{A2}, \phi_{A3}) \\ \vec{A}(\phi_B, \phi_{A1}, \phi_{A2}, \phi_{A3}) &= R_Z(\phi_B) \cdot R_X(\phi_{A1}) \cdot \left(\vec{A_{10}} + R_X(\phi_{A2}) \cdot \left(\vec{A_{20}} + R_X(\phi_{A3}) \cdot \vec{A_{30}}\right)\right) \end{aligned} \tag{22}$$

5 Coordinate systems and conversions

The control script uses multiple coordinate systems listed here

$$Kath = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad Rot = \begin{bmatrix} r \\ \phi_Z \\ \phi_X \end{bmatrix} \tag{23}$$

Kath and Rot both have their advantages and disadvantages, Rot is easier to use for the control system (it always converts to this system), while Kath is easier for human use. The conversion functions are the following

$$Kath(Rot) = R_Z(\phi_Z) \cdot R_X(\phi_X) \cdot \begin{bmatrix} 0 \\ r \\ 0 \end{bmatrix} \tag{24}$$

$$Rot(Kath) = \begin{bmatrix} |Kath| \\ \arctan_F(y, -x) \\ \arctan_F(\sqrt{x^2 + y^2}, z) \end{bmatrix} \quad \arctan_F(x, y) = \begin{cases} \arctan(y/x) & if\ x > 0 \\ \pi + \arctan(y/x) & if\ x < 0 \\ \begin{cases} \frac{\pi}{2} & if\ y > 0 \\ \frac{-\pi}{2} & if\ y < 0 \\ 0 & otherwise \end{cases} & otherwise \end{cases} \tag{25}$$