# ROBOTIK_HTM_Scara

November 20, 2024

# 1 Herleitung der HTM für Standardmanipulatoren

## 1.1 (1) SCARA

Folgende Bibliotheken, Funktionen und Konstanten werden genutzt:

[1]:
```
from matplotlib import pyplot as plt
from numpy.linalg import inv
from IPython import display
import numpy as np
import math

PI = 3.1415926535

def plot4x4Mat(m):
    print("[%6.2f\t%6.2f\t%6.2f\t%6.2f]"%(m[0,0],m[0,1],m[0,2],m[0,3]))
    print("[%6.2f\t%6.2f\t%6.2f\t%6.2f]"%(m[1,0],m[1,1],m[1,2],m[1,3]))
    print("[%6.2f\t%6.2f\t%6.2f\t%6.2f]"%(m[2,0],m[2,1],m[2,2],m[2,3]))
    print("[%6.2f\t%6.2f\t%6.2f\t%6.2f]"%(m[3,0],m[3,1],m[3,2],m[3,3]))

def rotX(phi):
    m = np.array([[1,0,0,0],
                  [0,math.cos(phi),math.sin(phi),0],
                  [0,-math.sin(phi),math.cos(phi),0],
                  [0,0,0,1]])
    return m

def rotY(phi):
    m = np.array([[math.cos(phi),0,-math.sin(phi),0],
                  [0,1,0,0],
                  [math.sin(phi),0,math.cos(phi),0],
                  [0,0,0,1]])
    return m

def rotZ(phi):
    m = np.array([[math.cos(phi),math.sin(phi),0,0],
                  [-math.sin(phi),math.cos(phi),0,0],
                  [0,0,1,0],
```

```
                    [0,0,0,1]])
    return m
```

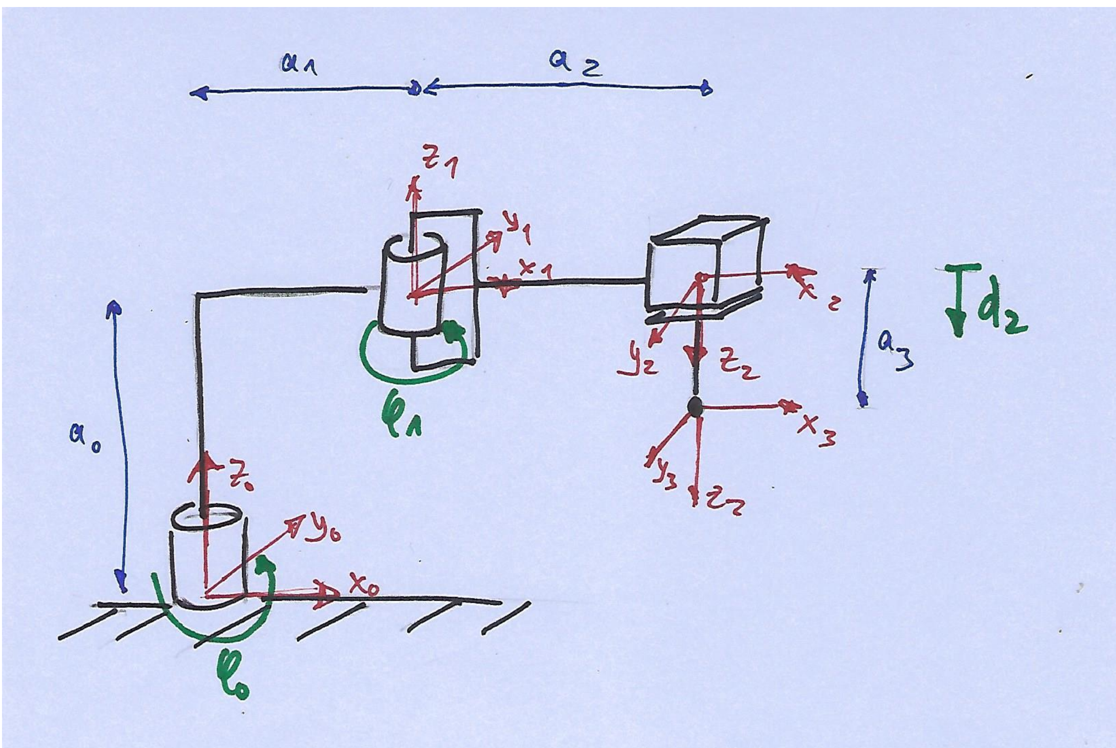
### 1.1.1 (I) Kinematisches Model des Roboters

**(I.1) Der Roboter**

**(I.2) Kinematische Struktur und Parameter** Die wesentlichen Parameter des SCARA-Roboters sind zwei Rotations und ein Translationsgelenk. Im folgenden ist die Kinematische Kette des Systems angegeben. Neben den Gelenkparametern $\phi_0$, $\phi_1$ und $d_2$ sind noch die Parameter zu Gliederabständen ($a_0$, $a_1$, $a_2$ und $a_3$) gegeben. Weiterhin sind die Koordinatensysteme KS0, KS1, KS2, KS3 für jedes Gelenk und den TCP eingetragen. Die Koordinatensysteme drehen oder bewegen sich nicht mit, wenn die entsprechenden Gelenkwerte verändert werden.

[2]:
```
display.Image("./data/SCARA_KinStrk.jpg")
```

[2]:



[3]:
```
a0 = 1
a1 = 1
a2 = 1
a3 = 1
phi0 = 0
phi1 = 0
d2   = 0
```

[4]:
```
#Ausgabe der Ergebnisse
print("a0 = ",end=""); print(a0)
print("a1 = ",end=""); print(a1)
print("a2 = ",end=""); print(a2)
print("a3 = ",end=""); print(a3)
print("phi0 = ",end=""); print(phi0)
print("phi1 = ",end=""); print(phi1)
print("d2 = ",end=""); print(d2)
```

```
a0 = 1
a1 = 1
a2 = 1
a3 = 1
phi0 = 0
phi1 = 0
d2 = 0
```

#### (I.3) Koordinatensysteme

### 1.1.2 (II) Herleitung der HTM

Kinematische Kette…

**(II.1) HTM ${}^2_3T$** Die Koordinatensysteme KS3 und KS2 haben die gleiche Orientierung. Die Orientierung verändert sich auch nicht, wenn beliebige Gelenkwerte verändert werden. Daher hat die HTM ${}^2_3T$ keine Rotationsanteil.

Ein Translationsanteil ist jeodch vorhanden. Dieser ist einmal fest, definert durch die Link-Länge $a_3$, und einmal variabel durch das Translationsgelenk, repäsentiert durch den Parameter $d_2$.

Der parametrisierte Translationsvektor ${}^2V$ bzgl. Zielkoordinatensystem KS2 beträgt:

$$
{}^2V = \begin{pmatrix} 0 \\ 0 \\ a_3 + d_2 \\ 1 \end{pmatrix}
$$

Die finale HTM ${}^2_3T(d_2)$ beträgt somit:

$$
{}^3_2T \quad = \quad \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & (a_3 + d_2) \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

[5]:
```
T_3_2 = np.array([[1,0,0,0],[0,1,0,0],[0,0,1,(a3+d2)],[0,0,0,1]]);
```

[6]:
```
#Ausgabe der Berechnungsergebnisse
print("T_3_2 = ",end="\n"); plot4x4Mat(T_3_2)
```

```
T_3_2 =
[  1.00   0.00    0.00    0.00]
[  0.00   1.00    0.00    0.00]
[  0.00   0.00    1.00    1.00]
[  0.00   0.00    0.00    1.00]
```

**(II.2) HTM $^1_2T$** In der Grundstellung (siehe Kinematische Struktur oben) habe die beiden Koordinatensysteme K2 und K1 eine unterschiedliche Orientierung. Die Orientierung der Koordinatensysteme ist ausserdem noch bestimmt durch die Gelenkstellung $\phi_1$. In der Grundstellung ($\phi_1 = 0$) können die beiden Koordinatensysteme in die gleiche Orientierung gebracht werden, wenn KS2 entlang der X-Achse eine Rotation um 180° bzw. $\pi$(rad) erfährt.

Eine weitere Rotation ist notwendig, wenn $\phi \neq 0$ gilt. Dann muss KS2 entlang der Z-Achse zusätzlich rotiert werden. Der Betrag dieser Rotation ist leicht aus der Kinematischen Struktur abzulesen. Der Betrag der Rotation beträgt $|\phi|$. Fraglich ist das Vorzeichen.

Bzgl. der Definitionen aus der Vorlesung betrachten wir die Drehung von KS2 (Ursprungskoordinatensystem) zu KS1 (Zielkoordinatensystem) hin. Wird nun zuerst die Rotation von KS2 um die X-Achse durchgeführt und danach die Rotation um die Z-Achse, dann beträgt der Wert des Rotationswinkels $-\phi$. Denn KS2 müsste dann entgeben der postiven Drehrichtung Rotiert werden.

Wird hingegen zuerste die Drehung um die Z-Achse ausgeführt und als zweites die Rotation um die X-Achse, dann ist der Rotationswinkel $+\phi$.

Es gibt also mindestens zwei Möglichkeiten die Rotationen darzustellen. Erste Variante, erst Rotation um die X-Achse und dann um die Z-Achse:

$$^1_2R_{XZY}(\pi, -\phi_1, 0) = R_Y(0) \cdot R_Z(-\phi_1) \cdot R_X(\pi) \; =$$

$$\begin{pmatrix} cos(0) & 0 & -sin(0) & 0 \\ 0 & 1 & 0 & 0 \\ sin(0) & 0 & cos(0) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} cos(-\phi_1) & sin(-\phi_1) & 0 & 0 \\ -sin(-\phi_1) & cos(-\phi_1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\pi) & sin(\pi) & 0 \\ 0 & -sin(\pi) & cos(\pi) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} =$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} cos(\phi_1) & -sin(\phi_1) & 0 & 0 \\ sin(\phi_1) & cos(\phi_1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} =$$

$$\begin{pmatrix} cos(\phi_1) & sin(\phi_1) & 0 & 0 \\ sin(\phi_1) & -cos(\phi_1) & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

[7]:
```
R1_2_1 = rotY(0) @ rotZ(-phi1) @ rotX(PI);
```

[8]:
```python
#Ausgabe der Berechnungsergebnisse
print("R1_2_1 = ",end="\n"); plot4x4Mat(R1_2_1)
```

```
R1_2_1 =
[  1.00   0.00    0.00    0.00]
[  0.00  -1.00    0.00    0.00]
[  0.00  -0.00   -1.00    0.00]
[  0.00   0.00    0.00    1.00]
```

Bei der zweite Variante wird erst um die Z-Achse und dann um die X-Achse rotiert:

$$^1_2R_{ZXY}(\phi_1, \pi, 0) = R_Y(0) \cdot R_X(\pi) \cdot R_Z(\phi_1) \ =$$

$$\begin{pmatrix} cos(0) & 0 & -sin(0) & 0 \\ 0 & 1 & 0 & 0 \\ sin(0) & 0 & cos(0) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\pi) & sin(\pi) & 0 \\ 0 & -sin(\pi) & cos(\pi) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} cos(\phi_1) & sin(\phi_1) & 0 & 0 \\ -sin(\phi_1) & cos(\phi_1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} =$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} cos(\phi_1) & sin(\phi_1) & 0 & 0 \\ -sin(\phi_1) & cos(\phi_1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} =$$

$$\begin{pmatrix} cos(\phi_1) & sin(\phi_1) & 0 & 0 \\ sin(\phi_1) & -cos(\phi_1) & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

[9]:
```python
R2_2_1 = rotY(0) @ rotX(PI) @ rotZ(phi1);
```

[10]:
```python
#Ausgabe der Berechnungsergebnisse
print("R2_2_1 = ",end="\n"); plot4x4Mat(R2_2_1)
```

```
R2_2_1 =
[  1.00   0.00    0.00    0.00]
[  0.00  -1.00    0.00    0.00]
[  0.00  -0.00   -1.00    0.00]
[  0.00   0.00    0.00    1.00]
```

Mit beiden Varianten werden die Koordinatensystem KS2 und KS1 in die gleiche Orientierung gebracht. Beide Varianten führen auch zu den gleichen Rotationsmatrizen.

Auch der Translationsvektor $^1V$ ist nicht direkt aus der Kinematischen Struktur abzulesen. Befindet sich der Roboter nicht in der Grundstellung, dann hat dieser Vektor x- und y-Anteile, abhängig von der Gelenkstellung $\phi_1$.

$$^1V = \begin{pmatrix} a_2 \cdot cos(\phi_1) \\ a_2 \cdot sin(\phi_1) \\ 0 \\ 1 \end{pmatrix}$$

Damit ergibt sich folgende paremetrisierte HTM ${}^{1}_{2}T(\phi_1)$:

$$
{}^{1}_{2}T(\phi_1) \;=\; \begin{pmatrix} cos(\phi_1) & sin(\phi_1) & 0 & a_2 \cdot cos(\phi_1) \\ sin(\phi_1) & -cos(\phi_1) & 0 & a_2 \cdot sin(\phi_1) \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

[11]:
```
T_2_1 = R2_2_1;
T_2_1[0,3] = a2*math.cos(phi1)
T_2_1[1,3] = a2*math.sin(phi1)
```

[12]:
```
#Ausgabe der Berechnungsergebnisse
print("T_2_1 = ",end="\n"); plot4x4Mat(T_2_1)
```

```
T_2_1 =
[  1.00   0.00    0.00    1.00]
[  0.00  -1.00    0.00    0.00]
[  0.00  -0.00   -1.00    0.00]
[  0.00   0.00    0.00    1.00]
```

**(II.3) HTM ${}^{0}_{1}T$** Die HTM kann genauso hergeleitet werden wie ${}^{1}_{2}T$ oben.Der einzige Unterschied ist, dass es keine konstante Rotation um $\pi$ gibt. Daher gibt es auch nur eine Rotation um die Achse Z des Zielkoordinatensystems KS0 mit negativen Betrag. Der Rotationswinkel ist also $-\phi_0$.

$$
{}^{Z}_{U}R_{XYZ}(0,0,-\phi_0) = R_Z(-\phi_0) \cdot R_Y(0) \cdot R_X(0) \;=
$$

$$
\begin{pmatrix} cos(-\phi_0) & sin(-\phi_0) & 0 & 0 \\ -sin(-\phi_0) & cos(-\phi_0) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \;=
$$

$$
\begin{pmatrix} cos(\phi_0) & -sin(\phi_0) & 0 & 0 \\ sin(\phi_0) & cos(\phi_0) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

$$
{}^{0}V = \begin{pmatrix} a_1 \cdot cos(\phi_0) \\ a_1 \cdot sin(\phi_0) \\ a_0 \\ 1 \end{pmatrix}
$$

$$
{}^{0}_{1}T(\phi_1) \;=\; \begin{pmatrix} cos(\phi_0) & -sin(\phi_0) & 0 & a_1 \cdot cos(\phi_0) \\ sin(\phi_0) & cos(\phi_0) & 0 & a_1 \cdot sin(\phi_0) \\ 0 & 0 & 1 & a_0 \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

[13]:
```
T_1_0 = rotZ(-phi0)
T_1_0[0,3] = a1*math.cos(phi0)
T_1_0[1,3] = a1*math.sin(phi0)
T_1_0[2,3] = a0
```

[14]:
```
#Ausgabe der Berechnungsergebnisse
print("T_1_0 = ",end="\n"); plot4x4Mat(T_1_0)
```

```
T_1_0 =
[  1.00   0.00    0.00    1.00]
[ -0.00   1.00    0.00    0.00]
[  0.00   0.00    1.00    1.00]
[  0.00   0.00    0.00    1.00]
```

**(II.4) Die finale HTM $^0_3T$**

$$^0_3T(\phi_0, \phi_1, d_2) \;=\; ^0_1T(\phi_0) \cdot ^1_2T(\phi_1) \cdot ^2_3T(d_2) \;=$$

$$\begin{pmatrix} cos(\phi_0) & -sin(\phi_0) & 0 & a_1 \cdot cos(\phi_0) \\ sin(\phi_0) & cos(\phi_0) & 0 & a_1 \cdot sin(\phi_0) \\ 0 & 0 & 1 & a_0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} cos(\phi_1) & sin(\phi_1) & 0 & a_2 \cdot cos(\phi_1) \\ sin(\phi_1) & -cos(\phi_1) & 0 & a_2 \cdot sin(\phi_1) \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & (a_3 + d_2) \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

[15]:
```
T_3_0 = T_1_0 @ T_2_1 @ T_3_2
```

[16]:
```
#Ausgabe der Berechnungsergebnisse
print("T_3_0 = ",end="\n"); plot4x4Mat(T_3_0)
```

```
T_3_0 =
[  1.00   0.00    0.00    2.00]
[  0.00  -1.00    0.00    0.00]
[  0.00  -0.00   -1.00    0.00]
[  0.00   0.00    0.00    1.00]
```

### 1.1.3 (III) Übung

[17]:
```
# Parameter der Kinematischen Struktur
a0 = 1
a1 = 1
a2 = 1
a3 = 1
# Gelenkparameter
#phi0 = -PI/2
#phi1 = PI/2
#d2   = 2
print("phi0 = ", end="")
```

```python
phi0 = float(input());
print("phi1 = ", end="")
phi1 = float(input());
print("d2 = ", end="")
d2 = float(input());
# HTM: T_3_2
T_3_2 = np.array([[1,0,0,0],[0,1,0,0],[0,0,1,(a3+d2)],[0,0,0,1]]);
# HTM: T_2_1
T_2_1 = rotX(PI) @ rotZ(phi1);
T_2_1[0,3] = a2*math.cos(phi1)
T_2_1[1,3] = a2*math.sin(phi1)
# HTM: T_1_0
T_1_0 = rotZ(-phi0)
T_1_0[0,3] = a1*math.cos(phi0)
T_1_0[1,3] = a1*math.sin(phi0)
T_1_0[2,3] = a0
# HTM: T_3_0 = T_1_0 * T_2_1 * T_3_2
T_3_0 = T_1_0 @ T_2_1 @ T_3_2
#Ausgabe der Berechnungsergebnisse
print("T_3_0 = ",end="\n"); plot4x4Mat(T_3_0)
```

```
phi0 =

 0

phi1 =

 0

d2 =

 1

T_3_0 =
[   1.00   0.00    0.00    2.00]
[   0.00  -1.00    0.00    0.00]
[   0.00  -0.00   -1.00   -1.00]
[   0.00   0.00    0.00    1.00]
```

[ ]:

[ ]:

[ ]: