Robotik-Praktikum: Ballwurf mit dem Roboterarm Lynx6

– Modellbeschreibung –

Julia Ziegler, Jan Krieger

Modell zur Optimierung

Doppelpendel-Modell

Zur Optimierung einer Wurfbewegung wurde ein physikalisches Modell des Armes benötigt. Als einfachstes Modell wurde ein umgekehrtes Doppelpendel gewählt (siehe 1). An beiden Gelenken (rote Kreise) sind Motoren angebracht, die die Drehmomente $U_1(t)$ und $U_2(t)$ ausüben. Um die Bewegung mit dem Tool MUSCOD optimieren zu können, benötigt man die Bewegungsgleichungen des Systems, also die Differentialgleichungen, die den Roboter bewegen. Um diese aufzustellen, wurde der Lagrange-Formalismus angewendet. Dazu wird zunächst die Lagrangefunktion $\mathcal{L} = T - V$ aufgestellt. Dabei ist T die gesamte kinetische und V die gesamte potentielle Energie des Systems. Die kinetische Energie setzt sich aus zwei Teilen zusammen, der Rotationsenergie T_{rot} und der Energie durch Linearbewegung T_{kin}. Zur zweiteren zählt das unterste Glied („Unterarm") nicht hinzu, weil dieser eine Rotation um einen raumfesten Drehpunkt ausführt.

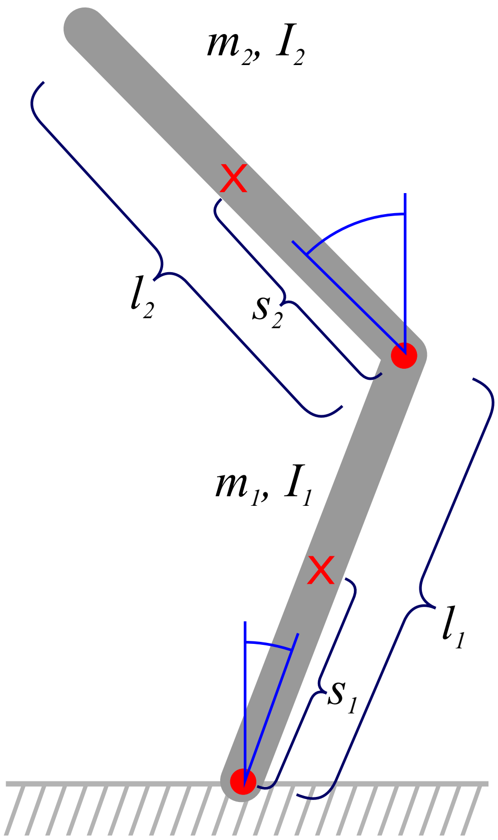

Abb. 1: Modell des Armes als Doppelpendel

Jedes Glied i wird durch eine Masse m_i und ein Trägheitsmoment I_i beschrieben. Das Trägheitsmoment bezieht sich dabei auf eine Drehung im Gelenk (also nicht im Schwerpunkt). Die Gelenke sind im wesentlichen längliche Quader, die zusätzlich durch eine Länge l_i beschrieben werden. Der Schwerpunkt befindet sich im Abstand s_i com Gelenk. Das Schultergelenk sitzt im Punkt $(0, 0)$ fest.

Das Ellbogengelenk sitzt bei $\vec{x}_1$ und die „Hand“ bei $\vec{x}_2$. Die Koordinaten der Schwerpunkte von Ober- und Unterarm sind $\vec{x}_{s1}$ und $\vec{x}_{s2}$.

Das Modell wird durch Minimalkoordinaten beschrieben. Da es sich um ein ebenes Problem handelt reduzieren sich die 6 Freiheitsgrade der Positionen Gelenke mit den Nebenbedingungen auf zwei. Die zwei unabhängigen Variablen sind $\varphi_{1,2}$, die Winkel an den Gelenken.

Die Bewegungsgleichungen ergeben sich über die Euler-Lagrange-Gleichung 2. Art

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\varphi}_i}\right) - \frac{\partial \mathcal{L}}{\partial \varphi_i} = 0. \tag{0.0.1}$$

Um die Drehmomente $U_i(t)$ zu berücksichtigen wird (0.0.1) erweitert (Euler-Lagrange-Gleichungen 1. Art):

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\varphi}_i}\right) - \frac{\partial \mathcal{L}}{\partial \varphi_i} = U_i. \tag{0.0.2}$$

Es bleibt also noch die kinetische und die potentielle Energie zu bestimmen. Für die letztere hat man nur Beiträge durch die Gravitation, also:

$$V = m_1 g y_{s1} + m_2 g y_{s2} \tag{0.0.3}$$

Die kinetische Energie ist:

$$T = \frac{1}{2} I_1 \dot{\varphi}_1^2 + \frac{1}{2} I_2 \dot{\varphi}_2^2 + \frac{1}{2} m_2 \dot{\vec{x}}_{s2}^2 \tag{0.0.4}$$

Um die Ausdrücke (0.0.3) und (0.0.4) zu berechnen, benötigt man noch die Koordinaten der Schwerpunkte der Glieder:

$$\vec{x}_{s1} = s_1 \cdot \begin{pmatrix} \sin\varphi_1 \\ \cos\varphi_1 \end{pmatrix} \tag{0.0.5}$$

$$\vec{x}_1 = l_1 \cdot \begin{pmatrix} \sin\varphi_1 \\ \cos\varphi_1 \end{pmatrix} \tag{0.0.6}$$

$$\vec{x}_{s2} = \vec{x}_1 + s_2 \cdot \begin{pmatrix} \sin\varphi_2 \\ \cos\varphi_2 \end{pmatrix} \tag{0.0.7}$$

Das Aufstellen der Bewegungsgleichungen lässt sich über (0.0.1) auch von Hand erledigen, was allerdings sehr aufwändig wird. Deswegen wird Mathematica eingesetzt, was es auch ermöglicht die Bewegungsgleichungen gleich numerisch zu integrieren und damit zu testen.

Aus Mathematica erhält man mit dem Befehl `EulerEquations[L, q[t], t]` die Euler-Lagrange-Gleichungen 2. Art. Durch einfache Manipulation kann man diesen eine generalisierte Kraft $U_i(t)$ hinzufügen. Diese wird ja einfach statt der 0 auf die rechte Seite der Gleichung (0.0.1) geschrieben. Dies erreicht man über die folgende Befehlsabfolge:

```
el1 = EulerEquations[L, phi1[t], t][[1]] == u1[t]
```

Diese berechnet die Euler-Lagrange-Gleichungen erster Art und selektieren die linke Seite. Diese linke Seite wird dann der aufgeprägten Kraft `u1[t]` gleichgesetzt. Danach werden die Winkelgeschwindigkeiten `omega1[t]=phi1'[t]` als unabhängige Variablen eingeführt (Variablenersetzungen):

```
el1 = ReplaceAll[el1, {phi1''[t] -> omega1'[t],  phi2''[t] -> omega2'[t
    ], phi1'[t] -> omega1[t],  phi2'[t] -> omega2[t] }];
```

Nun müssen die so erstellten Bewegungsgleichungen so umgestellt werden, dass sie Gleichungen der Standardform

$$\frac{d\vec{x}}{dt} = f(\vec{x}, t) \tag{0.0.8}$$

ergeben, da MUSCOD diese erwartet. Dazu wird aus den zwei Differentialgleichungen erster Ordnung (die sich oben ergeben haben) die Ableitung der zweiten Winkelgeschwindigkeit eliminiert. Man erhält daraus eine Differentialgleichung erster Ordnung, in der nur noch die Ableitung `omega1'[t]` auftaucht. Diese muss dann nur noch nach dieser Größe aufgelöst werden:

```
1 el11 = Collect[Eliminate[{el1, el2}, {omega2'[t]}], omega1'[t]];
2 els1 = Solve[el11, omega1'[t]] // FullSimplify
3 Format[els1, CForm]
```

Der letzte Befehl erzeugt aus der Mathematica-Formel aus Zeile 2 eine Formel in C-Syntax (unter Benutzung von `math.h`-Methoden).

Es bleibt zum Schluss noch die Umrechnung der absoluten Winkel des Mathematica-Modells auf die relativen Winkel des Roboters und der Simulation. Dazu betrachtet man die folgende Abbildung 2.

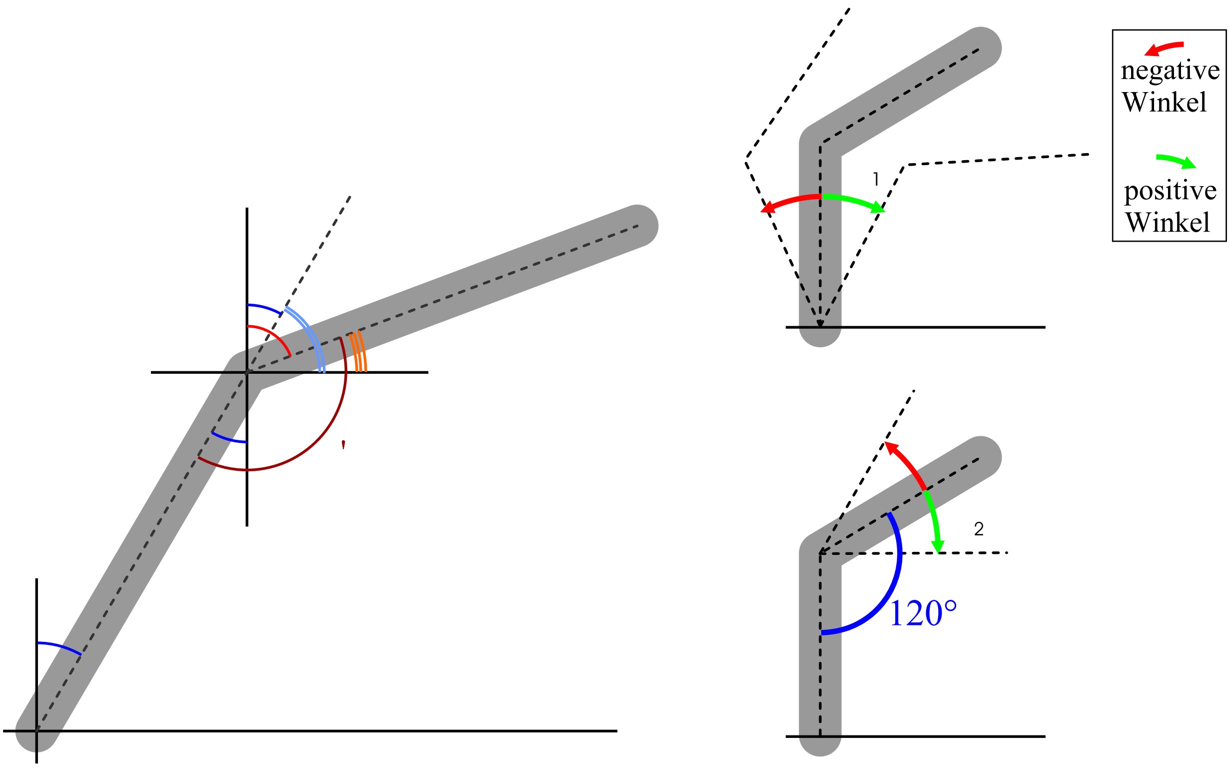

Abb. 2: Zur Umrechnung der absoluten Winkel φ_1, φ_2 auf die absoluten Winkel α_1, α_2. Die grau eingezeichnete Stellung ist die Ausgangsstellung in der $\alpha_1 = \alpha_2 = 0°$ gilt.

Nun kann man Umrechnungformeln aufstellen:

$$\alpha_1 = \varphi_1 \tag{0.0.9}$$

$$\alpha_2 = -\left(60° + \varphi_1 - \varphi_2\right) \tag{0.0.10}$$

Das eben beschriebene Mathematica-Tool kann sehr einfach auch auf ein Dreifachpendel erweitert werden (siehe `dreifachpendel.m`). Dazu müssen nur zusätzliche Definitionen für die weiteren Glieder eingefügt werden.

Berechnung der Trägheitsmomente

Der Roboter wird als physikalisches Pendel simuliert. Zur Aufstellung der Differentialgleichungen (siehe) benötigt man die Trägheitsmomente der einzelnen Gliedmaßen, bei Rotation um die Gelenkachsen. Um diese zu berechnen wurde ein C++-Programm geschrieben, in dem die Glieder aus rudimentären Formen (Quader und Kugeln) zusammengesetzt werden können. Zum Schluss wird dann der Trägheitstensor berechnet:

$$I = \sum_i m_i \begin{pmatrix} y_i^2 + z_i^2 & -x_i y_i & -x_i z_i \\ -y_i x_i & x_i^2 + z_i^2 & -y_i z_i \\ -z_i x_i & -z_i y_i & x_i^2 + y_i^2 \end{pmatrix} \tag{0.0.11}$$

Um diesen zu Berechnen wird der der durch Objekte belegte Raum in 100×100×100 Volumenelemente i zerlegt, denen aufgrund von Masse und Volumen der Quader/Kugeln ein Massenelement m_i zugeordnet werden kann. Damit lässt sich (0.0.11), sowie der Schwerpunkt

$$\vec{r}_S = \frac{\sum_i m_i \vec{r}_i}{M}, \qquad M = \sum_i m_i \tag{0.0.12}$$

berechnen, indem alle Volumenelemente i durchgegegangen werden. Zusätzlich bietet das Programm die Möglichkeit aus Quadern andere Quader aus zuschneiden.

Mit dem beschriebenen Programm ergaben sich die folgenden Trägheitstensoren und Schwerpunkt. Die Objekte sind relativ zur Drehachse (parallel zur z-Achse) der Gelenke ausgerechnet.

- Oberarm (enthält keine Motoren):

$$I_{\mathrm{OA}} = \begin{pmatrix} 204.256 & -1.74247 \cdot 10^{-13} & 4.41657 \cdot 10^{-14} \\ -1.74247 \cdot 10^{-13} & 2080.43 & -1.37353 \cdot 10^{-16} \\ 4.41657 \cdot 10^{-14} & -1.37353 \cdot 10^{-16} & 1930.92 \end{pmatrix} \cdot \underbrace{10^{-7}\ \mathrm{kg\ m}^2}_{=1\ \mathrm{g\ cm}^2}$$

$$I_{\mathrm{OAz}} = 1930 \cdot 10^{-7}\ \mathrm{kg\ m}^2 \qquad \vec{r}_{\mathrm{OA}} = \begin{pmatrix} 5.95 \\ 3.69579 \cdot 10^{-16} \\ -2.37106 \cdot 10^{-16} \end{pmatrix} \mathrm{cm}$$

- Unterarm (enthält zwei Standard-Servos):

$$I_{\mathrm{UA}} = \begin{pmatrix} 120.915 & -4.62527 \cdot 10^{-13} & 43.0685 \\ -4.62527 \cdot 10^{-13} & 5998.36 & 8.36703 \cdot 10^{-15} \\ 43.0685 & 8.36703 \cdot 10^{-15} & 5952.79 \end{pmatrix} \cdot 10^{-7}\ \mathrm{kg\ m}^2$$

$$I_{\mathrm{UAz}} = 5995 \cdot 10^{-7}\ \mathrm{kg\ m}^2 \qquad \vec{r}_{\mathrm{UA}} = \begin{pmatrix} 5.24296 \\ 2.73313 \cdot 10^{-16} \\ -0.0865433 \end{pmatrix} \mathrm{cm}$$

- Greifer (enthält zwei Micro-Servos und einen Ball mit Durchmesse 50 mm und Gewicht 50 g):

$$I_{\mathrm{GR}} = \begin{pmatrix} 487.189 & 54.6525 & -2.75152 \cdot 10^{-12} \\ 54.6525 & 12680.8 & 1.38671 \cdot 10^{-14} \\ -2.75152 \cdot 10^{-12} & 1.38671 \cdot 10^{-14} & 12653.1 \end{pmatrix} \cdot 10^{-7}\ \mathrm{kg\ m}^2$$

$$I_{\mathrm{GRz}} = 12653 \cdot 10^{-7}\ \mathrm{kg\ m}^2 \qquad \vec{r}_{\mathrm{GR}} = \begin{pmatrix} 8.69287 \\ -0.0443085 \\ 2.53626 \cdot 10^{-15} \end{pmatrix} \mathrm{cm}$$

- Unterarm und Greifer zusammen (für das Modell mit zwei Gelenken):

$$I_{\mathrm{UA+GR}} = \begin{pmatrix} 611.829 & 130.308 & 38.7404 \\ 130.308 & 64457.4 & -0.0171811 \\ 38.7404 & -0.0171811 & 64381.3 \end{pmatrix} \cdot 10^{-7}\ \mathrm{kg\ m}^2$$

$$I_{\mathrm{UA+GRz}} = 64412 \cdot 10^{-7}\ \mathrm{kg\ m}^2 \qquad \vec{r}_{\mathrm{UA+GR}} = \begin{pmatrix} 14.0568 \\ -0.03093 \\ -0.0335507 \end{pmatrix} \mathrm{cm}$$

Zusätzlich sind die Positionen der Schwerpunkte $\vec{r}_*$ der Glieder angegeben. Sie sind jeweils relativ zum Gelenk angegeben. Die Modellierung erfolgt so, dass die Bewegung in der xy-Ebene ausgeführt wird. Damit finden alle Rotationen um die z-Achse statt. Man hat also einen Winkelgeschwindigkeitsvektor $\vec{\omega} = (0, 0, \omega)^T$. Dieser führt auch zu den angegebenen z-Trägheitsmomenten $I_{*,\mathrm{z}}$. Bei der Berechnung wurden nur Größen mitgenommen, die in der Größenordnung vergleichbar waren. Die Matrixeinträge mit der Größenordnung 10^{-13} und kleiner wurden vernachlässigt.