In dem folgenden Abschnitt werden verschiedene Ansätze zur Regelung des Systems erläutert und evaluiert. Die Aufgabe des Reglers besteht darin aus der Ausgangsgröße des Systems, dem Winkel des Würfels, die erforderliche Eingangsgröße, das Motormoment, zu ermitteln. Mit Hilfe eines Simulinkmodells können die verschiedenen Regler konfiguriert und erprobt werden.

Simulation in Simulink

Das dynamische Verhalten des Würfels kann mit den folgenden Differentialgleichungen bestimmt werden.

$$
\begin{aligned}
(\Theta_b^{(A)}+m_w\cdot l^2)\cdot\ddot{\phi}_b &= (m_b\cdot l_b+m_w\cdot l)\cdot g\cdot\sin(\phi_b)-C_b\cdot\dot{\phi}_b+C_w\cdot \dot{phi}_w-T_M \\
\ddot{\phi}_w &= \frac{(\Theta_b^{(A)}+\Theta_w^{(B)}+m_w\cdot l^2)\cdot(T_M-C_w\cdot\dot{phi}_w)}{\Theta_w^{(B)}\cdot(\Theta_b^{(A)}+m_w\cdot l^2)}-\frac{(m_b\cdot l_b+m_w\cdot l)\cdot g\cdot\sin(\phi_b)-C_b\cdot\dot{phi}_b}{\Theta_b^{(A)}+m_w\cdot l^2}
\end{aligned}
$$

Folgich kann ein Simulinkmodell entworfen werden, welches dieses Verhalten simuliert. Bei dem Sollwert

$$\phi_{Soll} = 0$$

handelt es sich um einen instabilen Gleichgewichtspunkt. Um dieses Verhalten auf das Simulink-modell zu übertragen muss ein zufällige Störgröße zur Winkelbeschleunigung des Körpers addiert werden. Der maximale Störfaktor beträgt das durch Gravitation verursachte Moment bei einem Ausfallwinkel von fünf Grad.

PID-Regler

Der erste Ansatz zur Kontrolle des Würfels besteht darin einen PID-Regler zu entwerfen. Da die Regelstrecke bekannt und somit simulierbar ist, bietet es sich an Methoden des maschinellen Lernen zu verwenden um die Regelparameter zu bestimmen.

Evolutionäre Algorithmen

Evolutionäre Algorithmen gehören zu den Methoden des maschinellen Lernens. Das Ziel besteht darin einen optimalen Lösungskandidaten zu ermitteln in dem Prinzipien der Evolution, wie z.B. Mutation oder Selektion eingesetzt werden.
Für einen evolutionären Algorithmus muss zuerst die Kodierung der Lösungskandidaten festgelegt werden. In diesem Anwendungsfall setzt sich ein Lösungskandidat aus jeweils einem Wert für die Verstärkung des proportionalen, des differenzierenden und des integrierenden Anteils zusammen.

$$k_i=\{K_P^{(i)},K_D^{(i)},K_I^{(i)}\}$$

Zu Beginn des Verfahrens wird eine Anfangspopulation der Größe *p* von Lösungskandidaten erzeugt.

$$P_0=\{k_1,k_2,\ldots,k_p\}$$

Mit Hilfe einer Fitness-Fuktion werden die einzelnen Lösungskandidaten bewertet. Hierbei soll die Fitness-Funktion so gestaltet werden, dass gute Kandidaten eine höhere Fitness erzielen. In diesem Fall wird das Integral der quadratischen Regelabweichung verwendet.

$$f_{fit}(k_i):k_i\rightarrow fitness_{k_i} \quad | \quad fitness_{k_i}\in \mathbb{R}$$

$$f_{fit}(k_i)=\left(\int_0^{t_{end}} e_{k_i}^2\,dt\right)^{-1}$$

Der Algorithmus basiert darauf die aktuelle Population zu verändern und dadurch die nächste Generation zu erzeugen. Dies erfolgt durch die Manipulation von Kandidaten und das Ersetzen von Kandidaten durch neue. Hierfür werden s.g. genetische Operatoren eingeführt. Mit Hilfe einer Selektionsmethode wird bestimmt welche Kandidaten aus der alten Population in die neue übernommen werden. Der Selektionsparameter *r* bestimmt den Anteil der zu ersetzenden Kandidaten.

$$f_{sel}(P_i) : P_i \rightarrow P_{i,selected} \quad | \quad |P_{i,selected}| = p \cdot r$$

In dieser Anwendung wird eine warscheinlichkeitsbasierte Selektionsmethode verwendet um die Übernahmekandidaten zu bestimmen. Zuerst wird die relative Fitness eines Lösungskandidaten berechnet.

$$fitness_{rel}(k_i) \quad = \quad \frac{f_{fit}(k_i)}{\sum_{j=1}^{p} f_{fit}(k_j)}$$

Die relative Fitness eines Kandidaten wird als Wahrscheinlichkeit interpretiert. Mittels dieser Wahrscheinlichkeiten wird ein Kandidat bestimmt, welcher in die nächste Generation übernommen wird. Dieser Prozess wird so oft wiederholt bis die gewünschte Anzahl an Kandidaten erreicht wurde.

Anschließend wird die selektierte Population mit Hilfe der Crossover-Funktion auf die ursprüngliche Größe *p* gebracht.

$$f_{cross}(P_{i,selected}) : P_{i,selected} \rightarrow P_{i,crossed} \quad | \quad |P_{i,crossed}| = p$$

Die Crossover-Funktion wählt zwei zufällig Kandidaten und erzeugt aus diesen einen neuen Kandidaten. Die Parameter des neuen Kandidaten werden bestimmt, in dem der jeweilige Wert des zufällig gewählten Elternkandidaten übernommen wird.

$$f_{cross}(k_i, k_j) : \{k_i, k_j\} \rightarrow k_{new} \quad | \quad k_i, k_j \in P_{i,selected}$$

Die Population wird anschließend manipuliert. Die Manipulationsrate *m* bestimmt die Wahrscheinlichkeit, dass ein Kandidat verändert wird. Zur Veränderung eines Kandidaten wird eine Mutationsfunktion verwendet.

$$f_{mut}(P_{i,crossed}) : P_{i,crossed} \rightarrow P_{i,mutated}$$

$$P_{i,mutated} = P_{i+1}$$

Die Mutationsfunktion addiert zu den Parametern eines Kandidaten einen zufälligen Wert, der in einem vorgegebenen Bereich liegen.

$$f_{mut}(k_i) = \{K_P^{(i)} + M_P^{(i)}, K_D^{(i)} + M_D^{(i)}, K_I^{(i)} + M_I^{(i)}\} \quad | \quad M_{min} < M^{(i)} < M_{max}$$

Dieser Ablauf wird zyklisch wiederholt bis entweder ein Kandidat die gewünschte Fitness erreicht hat oder eine vorgegebene Anzahl von Generation durchlaufen worden ist.

Regler-Ergebnisse
Mit dem so entworfenen Regler wurden bereits zufriedenstellende Ergebnisse in der Simulation erreicht. Exemplarisch werden die Verläufe der Ausfallwinkel bei einem Startwinkel von null und fünf Grad gezeigt.