# Lösen des Rubik Cube mit Methoden der AI / Machine Learning

Die Erforschung des Problems erfolgt zunächst am PC unter Verwendung von Python, da es hier bereits fertige Frameworks für Machine-Learning gibt.

Es sollen möglichst einfache Programmiertechniken verwendet um den entstehenden Python Code auf ein Microcontroller-System übertragen zu können (Portierung in C). Bei der MCU wird ausgegangen von einem ARM Cortex mit ca 1Mbyte RAM / 512kByte Flash

Am Rechner werden einige Lösungsmöglichkeiten / AI Methodiken versucht und die geeignete dann auf ein MCU Sytem übertragen

## Definition des Datenmodells

Es wird ein möglichst einfaches Datenmodell gewählt. Der Rubiks Cube besteht aus 3x3x3 Mikrowürfeln.

Denkbar sind folgende 2 Darstellungsarten

1. Eine 3D Representation der Mikrowürfel in einem (3x3x3) Array. Der Wert ist eine zusammengesetzte Datenstruktur, welche soviele Informationen enthalten muss um die relevante Info extrahieren zu können
2. Eine 2D Representation der Sichbaren Seiten in einem (6x3x3) Array. Der Wert ist die Farbinformation (z.b. als Farb-Index)

Die Veranschaulichung für b) ist einfach skizierbar für den gelösten Würfel.

2/0

2/1

2/2

1/2

1/1

1/0

0/2

0/1

0/0

2/0

2/1

2/2

1/2

1/1

1/0

0/2

0/1

0/0

2/0

2/1

2/2

1/2

1/1

1/0

0/2

0/1

0/0

2/0

2/1

2/2

1/2

1/1

1/0

0/2

0/1

0/0

2/0

2/1

2/2

1/2

1/1

1/0

0/2

0/1

0/0

2/0

2/1

2/2

1/2

1/1

1/0

0/2

0/1

0/0

Die Variante b) wird gewählt da sie einfacher Umzusetzen ist.

## Definition der Aktionen

Die Aktionen werden beschränkt auf diese welche auch vom Menschen durchgeführt werden können. Geht man davon aus der Würfel wird niemals vollständig rotiert (weiß ist immer oben, orange ist immer vorne) so kann man sich rein auf die Rotations-Aktion pro Seite konzentrieren.

Es können pro Achse 3 Seiten rotiert werden, entweder im (CW) oder gegen (CCW) den Uhrzeigersinn. Die Rotation der Inneren Reihe ist zwar denkbar würde aber dazu führen, dass der Würfel vollständig rotiert wird (weiß wäre nicht mehr oben), Daher wird diese zunächst ausgeschlossen und muss impliziert hergestellt werden durch 2 Rotation an den Außenreihen, welche zum selben Ergebnis führen.

Mit dieser Einschränkung gibt es dann

3 Achsen \* 2 Seiten \* 2 Richtung = 12 Aktionen

## Defintion der Rotations-Achsen

Folgende Bilder veranschaulichen:

2/0

2/1

2/2

1/2

1/1

1/0

0/2

0/1

0/0

2/0

2/1

2/2

1/2

1/1

1/0

0/2

0/1

0/0

**X**

**Y**

**X**

**Z**

**0**

**1**

**2**

**2**

**1**

**0**

**2**

**1**

**0**

Exemplarisch: Rotation Z=0, CW

2/0

2/1

2/2

1/2

1/1

1/0

0/2

0/1

0/0

2/2

2/1

2/2

1/2

1/1

2/1

0/2

0/1

2/0

2/0

2/1

2/2

1/2

1/1

1/0

0/2

0/1

0/0

2/2

1/2

0/2

1/2

1/1

1/0

0/2

0/1

0/0

2/0

2/1

2/2

1/2

1/1

1/0

0/0

1/0

2/0

2/0

2/1

0/2

0/1

1/1

1/0

0/0

0/1

0/0

### Array Indizierung

Wird die Array Indizierung genutzt um daraus auf die X/Y/Z Koordinate zu kommen so hat dieses Modell seine Schwäche. Egal wie die Indizierung des 2-D Arrays für die Darstellung der Sichtbaren Seite gewählt wird, so kann es nicht für jede der 12 möglichen Rotationen einfach sein. Im Bild veranschaulicht sind die Original Koordination vor der Rotation. Dennoch kann dieses Modell genutzt werden. Aufgrund der beschränkten Anzahl an Aktionen (12) kann für jede Aktion hartkodiert hinterlegt werden wie die Indizes getauscht werden müssen.