Manual Cube\_Solver\_Robot

Inhalt

[Hardware 2](#_Toc187499097)

[Würfel tauschen 2](#_Toc187499098)

[Reset 2](#_Toc187499099)

[State-Machine 3](#_Toc187499100)

[Allgemeine Funktionen und Modi 4](#_Toc187499101)

[Funktionen im Automatik-Modus 4](#_Toc187499102)

[Random moves 4](#_Toc187499103)

[Change Cube 4](#_Toc187499104)

[Scan Cube 4](#_Toc187499105)

[Solve 4](#_Toc187499106)

[Fast/Slow 5](#_Toc187499107)

[Funktionen im extern\_PC-Modus 5](#_Toc187499108)

[Random moves 5](#_Toc187499109)

[Change Cube 5](#_Toc187499110)

[Scan Cube 5](#_Toc187499111)

[Receive Cube from Robot 5](#_Toc187499112)

[Show Cube 5](#_Toc187499113)

[Python-Skript 5](#_Toc187499114)

[Solve 6](#_Toc187499115)

[Darstellung des Würfels für den extern\_PC Modus 6](#_Toc187499116)

[Darstellung der Drehungen für den extern\_PC Modus 7](#_Toc187499117)

# Hardware

* Sechs Schrittmotoren im Closed-Loop System + Servo Motor
* FRDM-MCXA153 Development Board (MCUXpresso IDE)
* Farbsensor: TCS34725 RGB
* Würfel: QiYi M Pro 3x3 MagLev Ball-Core UV Coated (Achtung nicht jeder Würfel ist geeignet)

# Würfel tauschen

Der Würfel ist über Magnete mit den Motoren verbunden. Dafür muss dieser jedoch leicht modifiziert werden. An jeder Seite des Würfels wurden vier kleine Magnete verklebt. Dabei ist die Ausrichtung der Magnete zu beachten. Sie sind alle in einen Nord-Süd Wechsel angebracht.

Dies ist ebenfalls beim einbauen des Würfels zu beachten und es müssen **ggf. Motoren um 90° gedreht werden**, damit die Magnete des Würfels und der Motoren passend übereinstimmen.

**ACHTUNG:**

**Beim Einsetzten MUSS die Gelbe Fläche nach oben und die Rote Fläche nach vorne zeigen!**

Der Würfel ist frei entnehmbar und sollte entweder **vor dem Anschalten** des Roboters eingesetzt werden oder in dem dafür vorgesehenen **Modus: „Change Cube“**.

**Wichtig ist, dass sich der Roboter nach dem Einschalten und in dem oben genannten Modus kalibriert. Daher ist es wichtig, dass der Würfel annähernd passend und nicht verkanntet eingesetzt wird.**

# Reset

Falls sich der Roboter sich aufhängen sollte gibt es auch einen Reset Knopf, welcher mit einem passenden Stift betätigt werden kann. Auch hier wird sich der Roboter neu kalibrieren. Außerdem wird der Kontroller zurückgesetzt und verliert jegliche gespeicherte Informationen.

# 

# State-Machine

# 

# Allgemeine Funktionen und Modi

Der Roboter verfügt über zwei Hauptmodi: Automatik und external\_PC

Die Funktionen des Roboters sind dabei sehr ähnlich, jedoch ändert sich die Bedienung. In dem Automatik Modus wird der Roboter über vier Taster und den Bildschirm gesteuert. Bei dem externen Modus läuft diese Bedienung über die USB-Schnittstelle. Diese Auswahl muss nach dem Anschalten des Roboters über Taster ausgewählt werden. Für den externen Modus existiert im **/src** Ordner eine Datei für die Kommunikation.

## Funktionen im Automatik-Modus

### Random moves

In dieser Funktion verdreht der Roboter zufällig 20 Seiten nacheinander. Der „Zufall“ wird hier über eine interne rand() Funktion gelöst, welchen den aktuellen Systemtick des Kontrollers als Seed nutzt.

### Change Cube

Hier werden alle Motoren deaktiviert und der Würfel kann entnommen und/oder eingesetzt werden.

Nach dem Tauschen muss ein Taster auf dem Roboter gedrückt werden, um den Wechsel zu bestätigen. Im Folgenden kalibrieren sich alle Seiten wieder nacheinander und der Roboter ist wieder einsatzbereit.

### Scan Cube

In dieser Funktion wird der Würfel eingelesen. Dafür wird jede Fläche nacheinander von einem Farbsensor gelesen dann interpretiert und abgespeichert. Um alle Flächen auf die untere Seite zu bekommen wird eine bestimmte Zugabfolge genutzt, welche den Würfel nicht verändert. Somit ist am Ende nach ca. 35 Sekunden der gesamte Würfel eingelesen und wurde dennoch nicht verändert.

Die einzelnen Steine werden nach einem bestimmten System in einem Array abgespeichert, welches später genauer erläutert wird.

### Solve

Hier löst ein von uns selbst geschriebener Algorithmus den eingelesenen Würfel und speichert die benötigten Züge in ein Array auf dem Kontroller ab. Dieser Algorithmus ist auf der GitHub Seite des Repo verlinkt.

### Fast/Slow

In diesen Funktionen wird das in „Solve“ berechnete Lösungsarray mit der entsprechenden Geschwindigkeit ausgeführt. Nach dem Erfolgreichen Lösen des Würfels werden die benötigten Züge und Zeit auf dem Bildschirm angezeigt.

## Funktionen im extern\_PC-Modus

### Random moves

In dieser Funktion verdreht der Roboter zufällig 20 Seiten nacheinander. Der „Zufall“ wird hier über eine interne rand() Funktion gelöst, welchen den aktuellen Systemtick des Kontrollers als Seed nutzt.

### 

### Change Cube

Hier werden alle Motoren deaktiviert und der Würfel kann entnommen und/oder eingesetzt werden.

Nach dem Tauschen muss ein Taster auf dem Roboter gedrückt werden, um den Wechsel zu bestätigen. Im Folgenden kalibrieren sich alle Seiten wieder nacheinander und der Roboter ist wieder einsatzbereit.

### Scan Cube

In dieser Funktion wird der Würfel eingelesen. Dafür wird jede Fläche nacheinander von einem Farbsensor gelesen dann interpretiert und abgespeichert. Um alle Flächen auf die untere Seite zu bekommen wird eine bestimmte Zugabfolge genutzt, welche den Würfel nicht verändert. Somit ist am Ende nach ca. 35 Sekunden der gesamte Würfel eingelesen und wurde dennoch nicht verändert.

Die einzelnen Steine werden nach einem bestimmten System in einem Array abgespeichert, welches später genauer erläutert wird.

### Receive Cube from Robot

Hier wird der eingelesene Würfel vom Roboter über die USB-Schnittstelle an den externen PC gesendet und dort abgespeichert.

**Wichtig!**

In den Bedienoptionen gibt es noch zwei weitere Funktionen, welche nicht auf dem Bild oben aufgeführt sind.

### Show Cube

In dieser Funktion wird der zuletzt Empfangene Würfen in der Console ausgegeben.

### Python-Skript

Hier wird vom C Programm über die Funktion fork() ein Python-Skript ausgeführt. Dieses berechnet dann die benötigten Züge um den Würfel zu lösen. Die Kommunikation zwischen dem C Programm und dem Python Skript wird über Inter-Prozess-Communication (IPC) realisiert.

Passend dafür existiert im /src/external\_PC Ordner eine Vorlage in welcher nur noch der eigene Lösungsalgorithmus implementiert werden muss. In dieser Vorlage wird der Würfel aus dem C Programm eingelesen und am Ende wird das Lösungsarray mit maximal 1024 Züge zurück an das C Programm gesendet.

### Solve

In dieser Funktion wird das aus dem Python-Skript berechnete Lösungsarray an den Kontroller geschickt und dort auch sofort ausgeführt. Am Ende werden die benötigten Züge und die Zeit auf dem Bildschirm angezeigt.

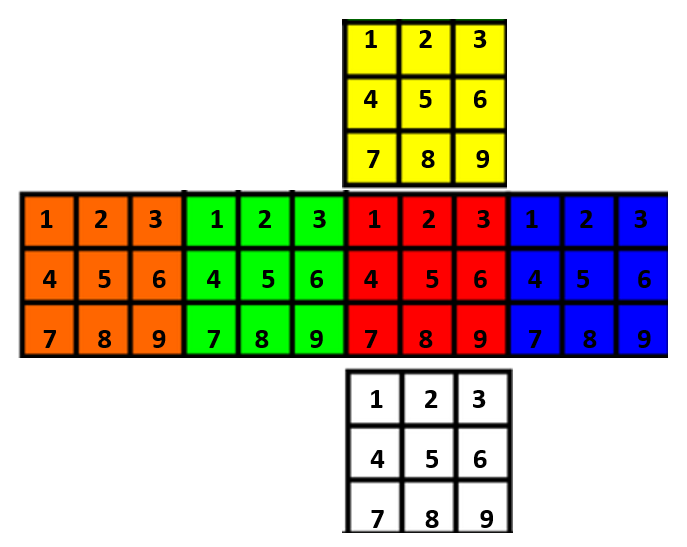
# Darstellung des Würfels für den extern\_PC Modus

Jede Farbe ist einer Zahl zugeordnet:

|  |  |
| --- | --- |
| **Color** | **Number** |
| White | 1 |
| Yellow | 2 |
| Red | 3 |
| Green | 4 |
| Orange | 5 |
| Blue | 6 |

Das eingelesene Array besteht aus 54 Zahlen und jede repräsentiert eine farbige Seite eines Steins auf dem Würfel. Das Array besteht so theoretisch aus 6 Blöcken, welche jeweils eine Fläche darstellen. Jeder dieser Blöcke beinhaltet 9 Zahlen. Die Reihenfolge der eingelesenen Blöcke ist:

White -> Red -> Blue -> Orange -> Green -> Yellow

Jeder Stein auf dieser Fläche wird dabei nach dem folgenden Muster eingelesen:

Würfel = [

W1, W2, W3, W4, W5, W6, W7, W8, W9,

R1, R2, R3, R4, R5, R6, R7, R8, R9,

B1, B2, B3, B4, B5, B6, B7, B8, B9,

O1, O2, O3, O4, O5, O6, O7, O8, O9,

G1, G2, G3, G4, G5, G6, G7, G8, G9,

Y1, Y2, Y3, Y4, Y5, Y6, Y7, Y8, Y9

]

**Tipp: Die Zahlen an der Position 5 des Blockes repräsentieren die Farbe der Fläche, da sich die Mittelsteine nicht verschieben lassen!**

# Darstellung der Drehungen für den external\_PC Modus

Jede Drehung ist folgenden Zahlen codiert (Genormte Notation als Vergleich).

Right beschreibt die Drehung im Uhrzeigersinn und left gegen den Uhrzeigersinn.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| side | right | | left | | Notation |
| white | 1 | D | 7 | D‘ | Down |
| yellow | 2 | U | 8 | U‘ | Up |
| red | 3 | F | 9 | F‘ | Front |
| green | 4 | R | 10 | R‘ | Right |
| orange | 5 | B | 11 | B‘ | Back |
| blue | 6 | L | 12 | L‘ | Left |

Das Ergebnisarray, welches an den Roboter geschickt wird, darf somit **nur Zahlen zwischen 1 und 12** beinhalten. Außerdem ist die **Länge auf 1024 Züge begrenzt**.