

Zauberwürfel-Lösungs-Maschine

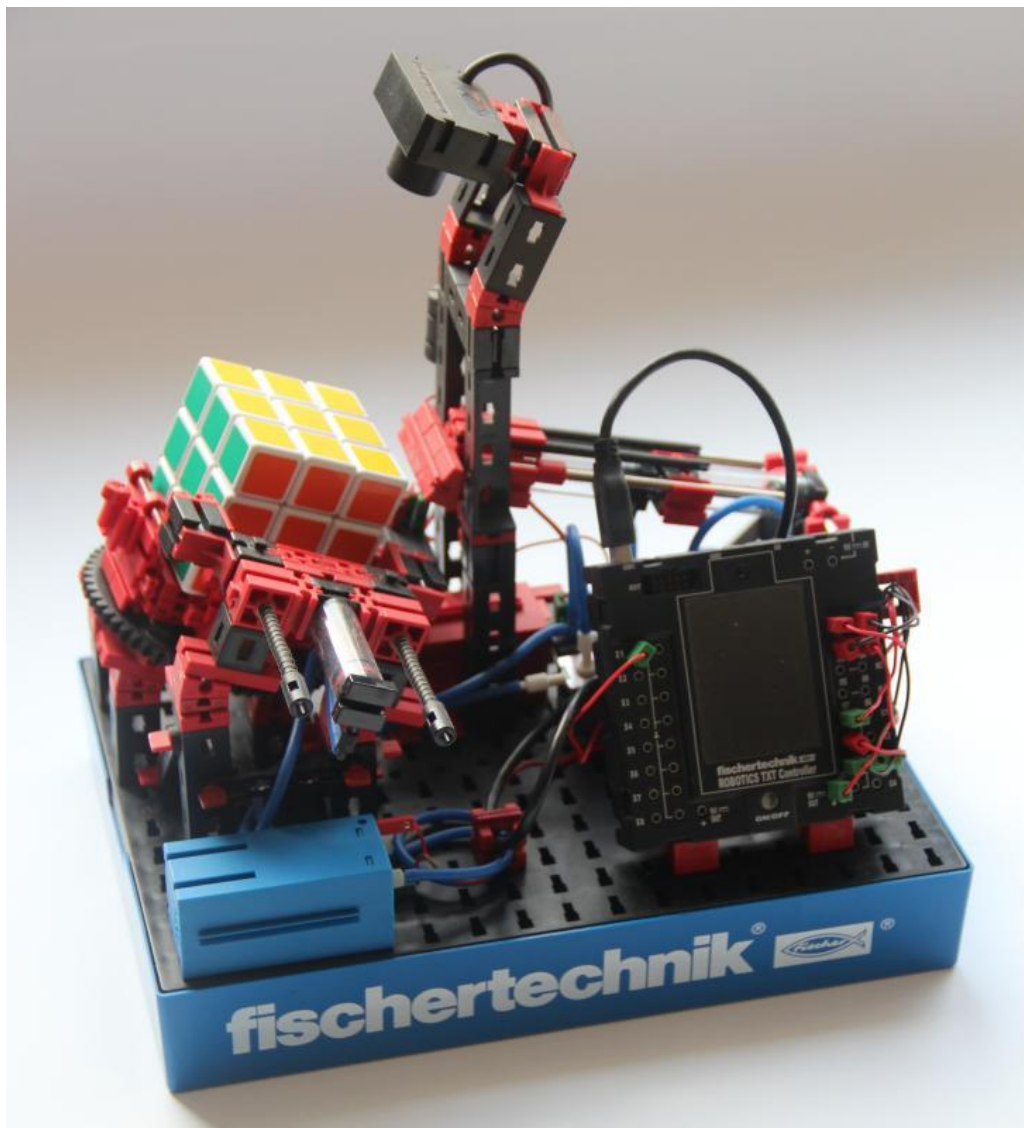
„Cube-Solver“

Modell: Till Harbaum <till@harbaum.org>

Grafische Umsetzung: Stephan Kallauch <stephan@kallauch.de>

Siehe auch www.ftcommunity.de

www.fischertechnik-designer.de



Version vom 7.11.2019

Einleitung

Der Cube-Solver-Roboter löst einen "Zauberwürfel" oder Rubiks-Cube vollautomatisch. Er besteht ausschließlich aus handelsüblichen fischertechnik-Bauteilen, nutzt aber auf dem TXT-Controller eine spezielle Software zum Betrieb, die sogenannte Community-Firmware.

Das Lösen des Würfels erfordert drei Schritte:

1. Einlesen des Ausgangszustands des Würfels
2. Berechnung des Lösungswegs
3. Ausführung der Lösung

Alle drei Schritte führt der Roboter automatisch aus. Die Steuerung aller Aufgaben übernimmt der im Modell integrierte fischertechnik-TXT-Controller. Zur Erkennung des Ausgangszustands wird eine fischertechnik-Kamera verwendet, die Berechnung der Lösung erfolgt auf dem TXT und schließlich steuert der TXT den Roboter so, dass der Würfel gelöst wird.

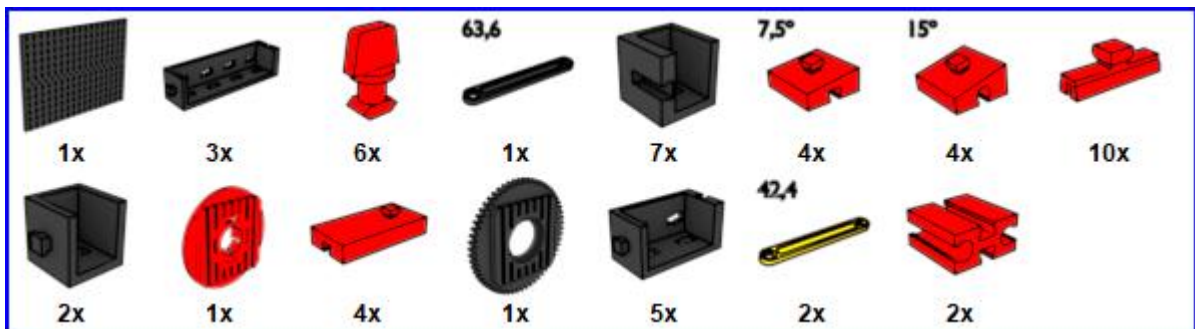
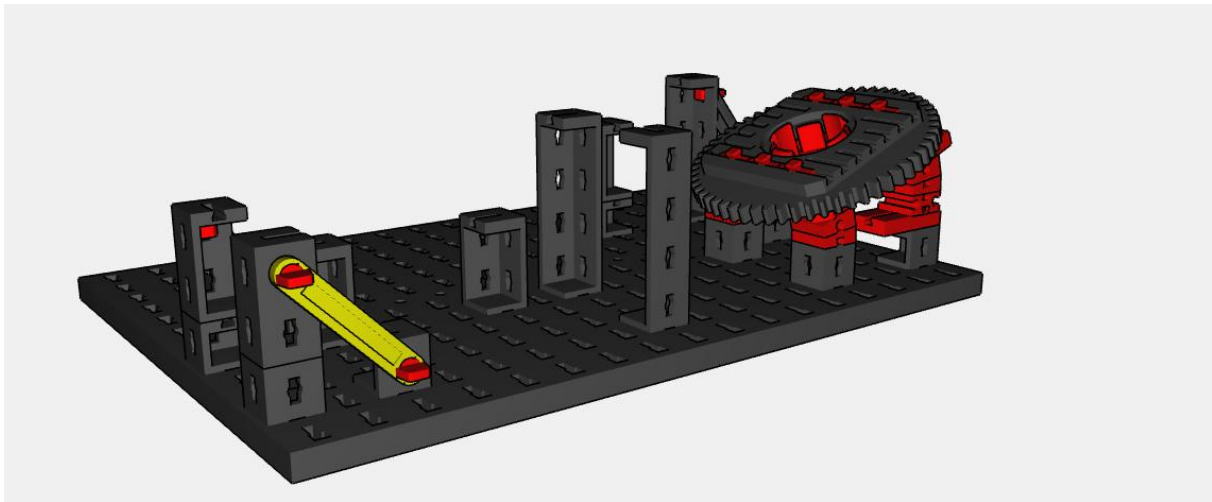
Zielgruppe

Dieser Roboter ist nicht „plug `n play“. Es erfordert etwas Geduld, die Mechanik und vor allem die Würfelerkennung so einzustellen, dass der Roboter seine Aufgabe tatsächlich fehlerfrei absolviert. Die Installation der Software erfordert den Einsatz der [Community-Firmware](#).

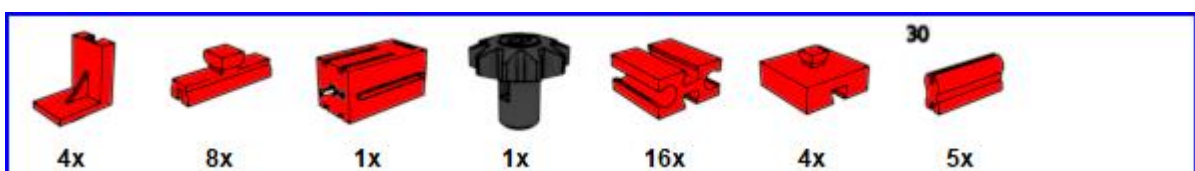
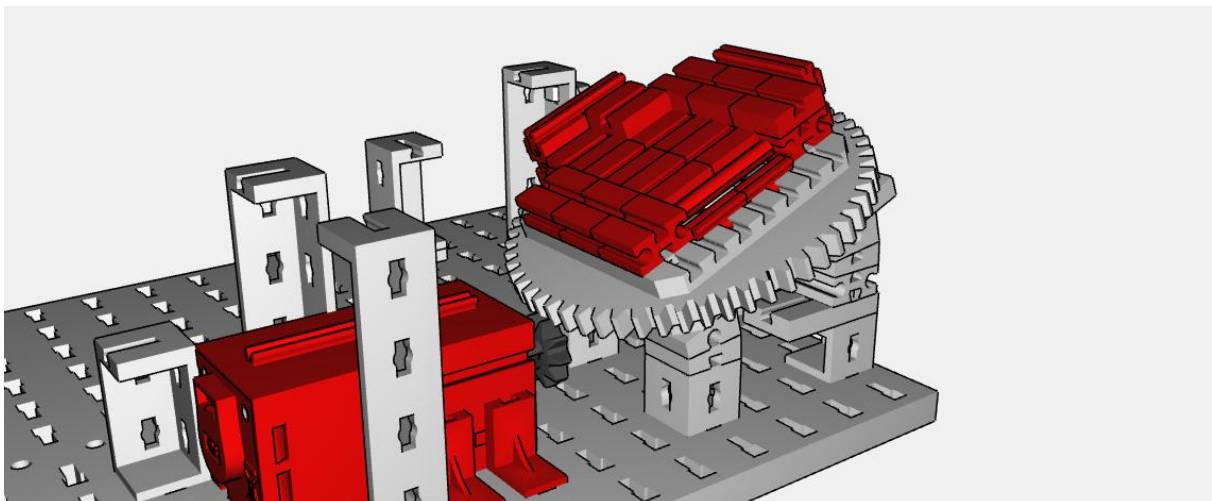
Eine Steuerung des Roboters durch „RoboPro“ ist nicht vorgesehen.

Bauanleitung

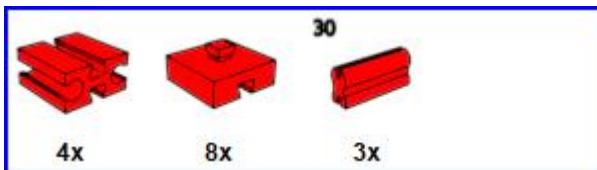
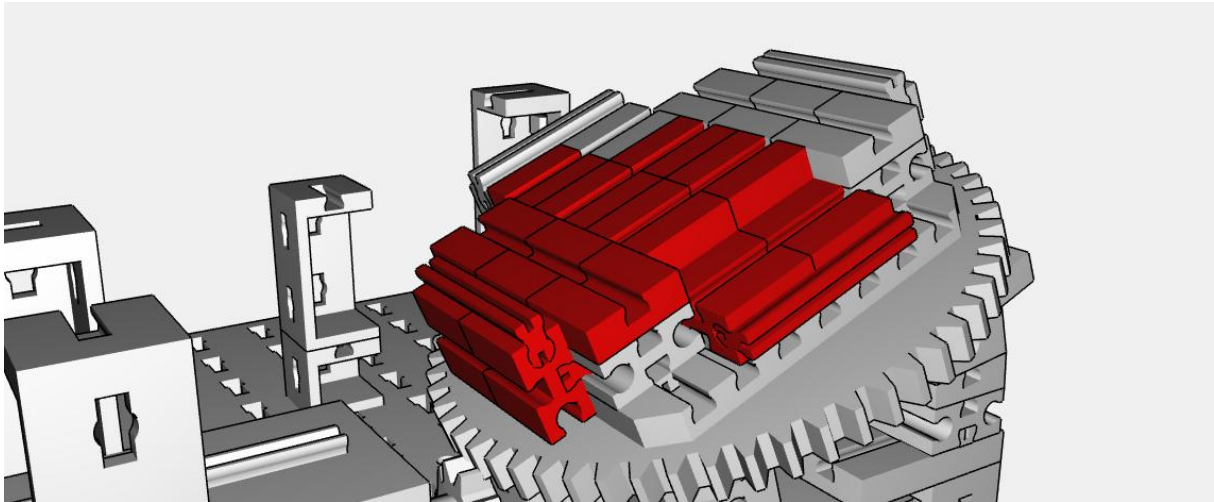
Bauphase 1



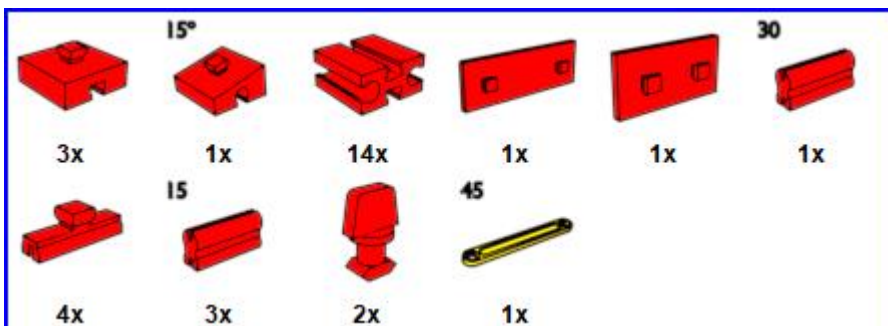
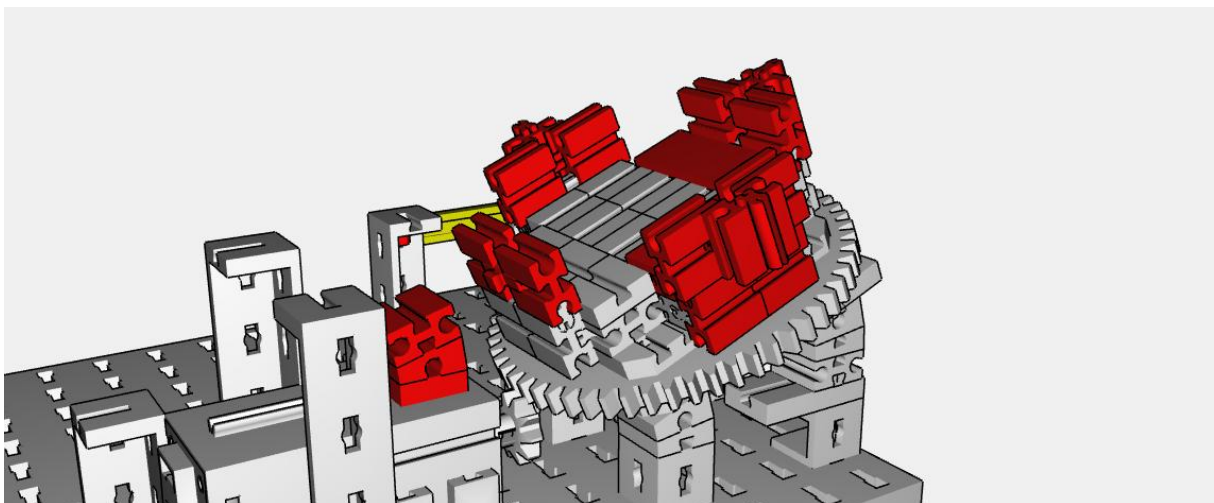
Bauphase 2



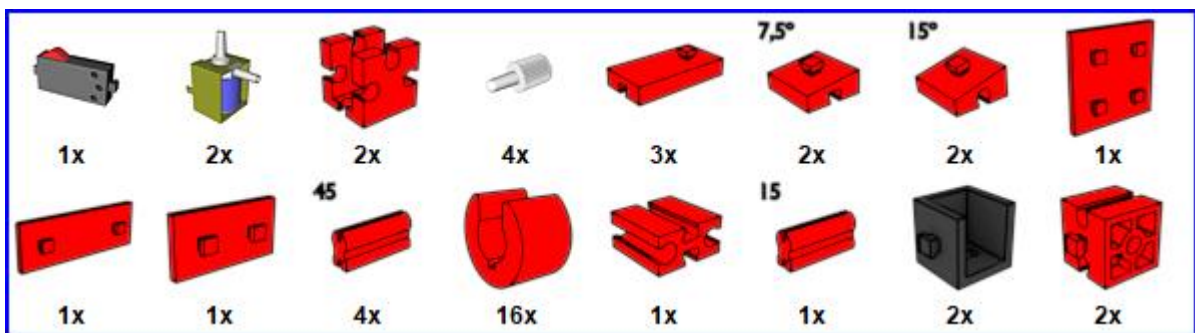
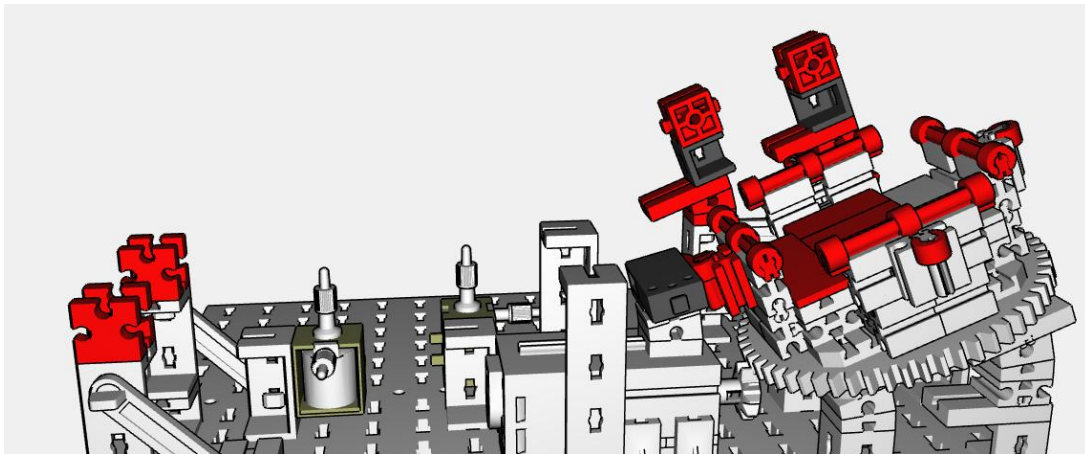
Bauphase 3



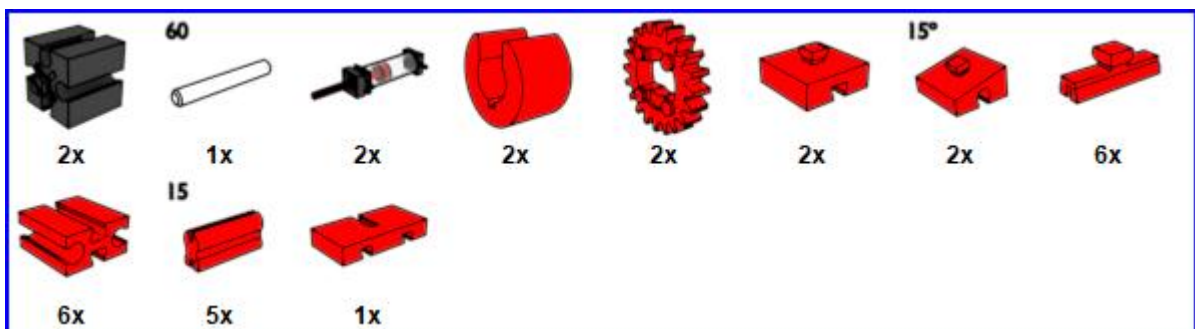
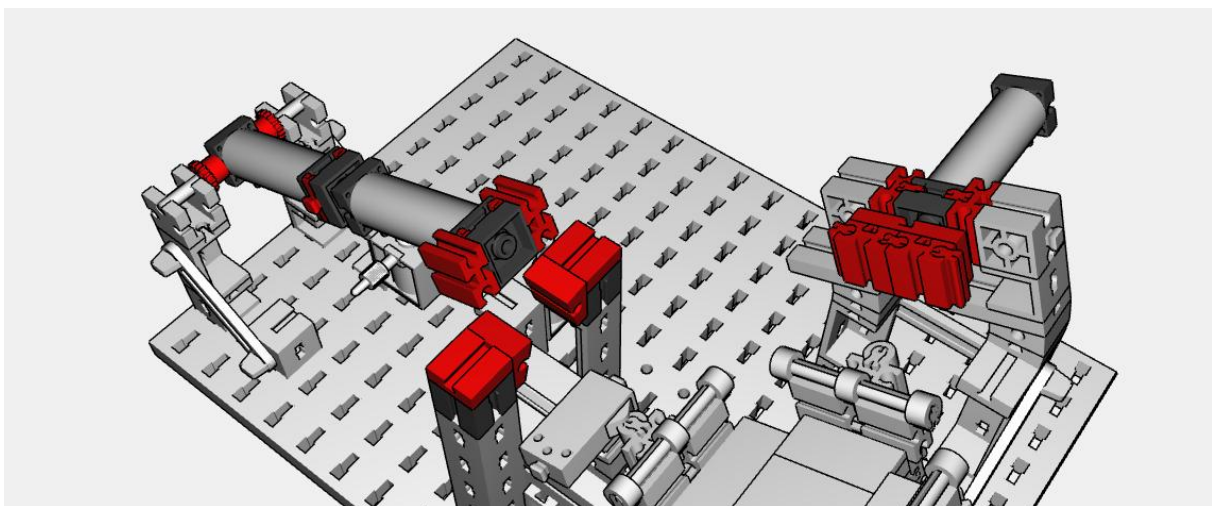
Bauphase 4



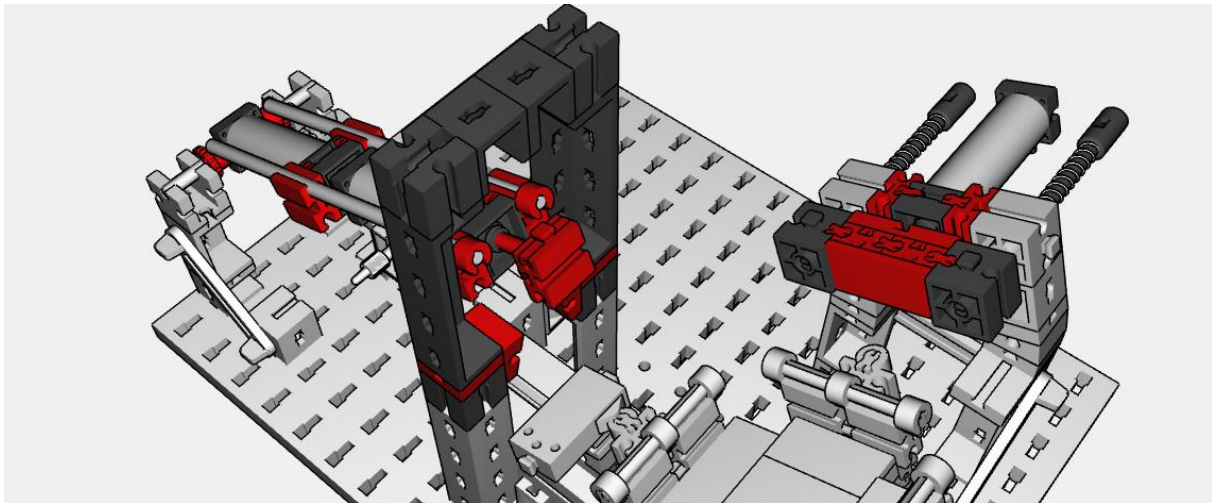
Bauphase 5



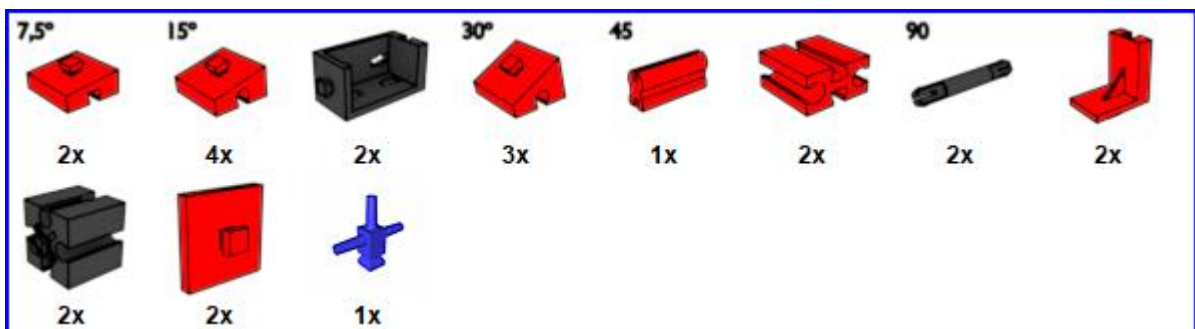
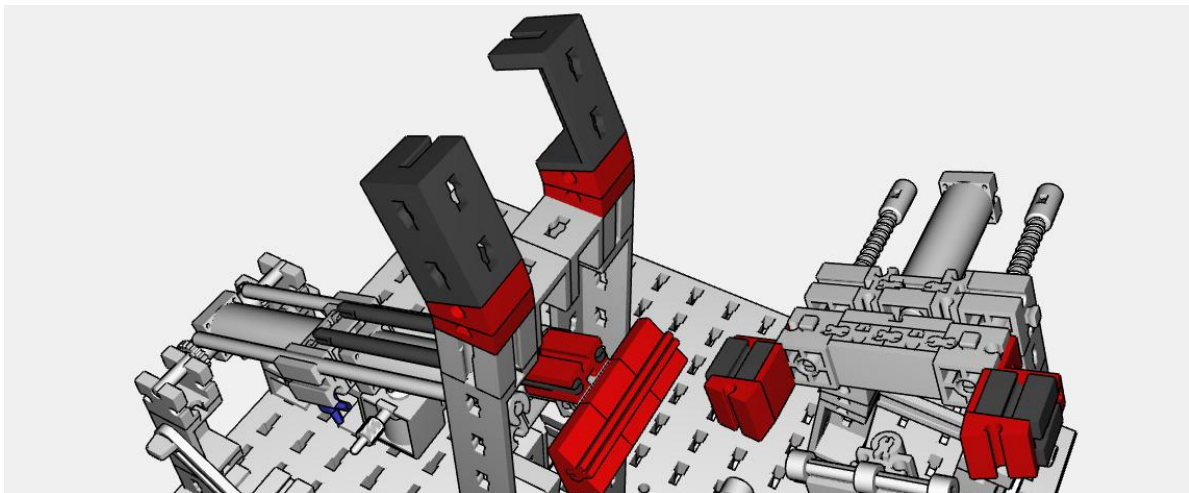
Bauphase 6



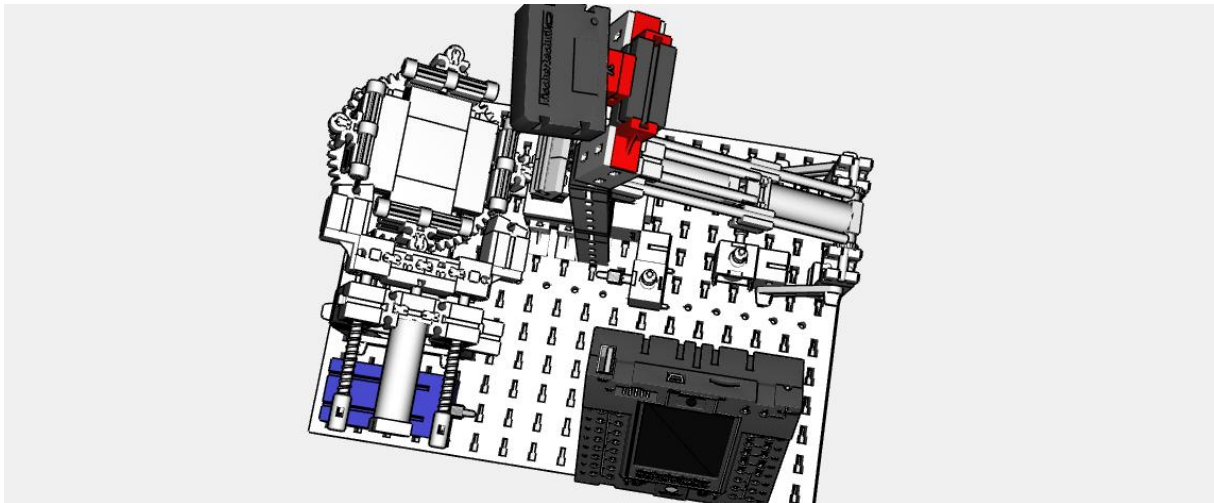
Bauphase 7



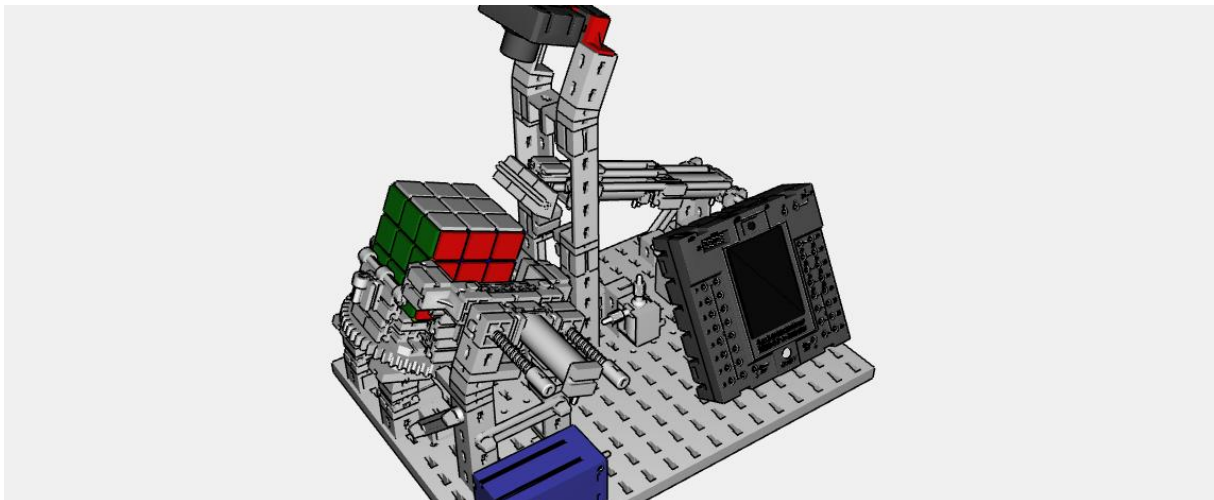
Bauphase 8



Bauphase 9



Bauphase 10



Inbetriebnahme der Mechanik

Die Mechanik des Roboters besteht aus drei wesentlichen Komponenten:

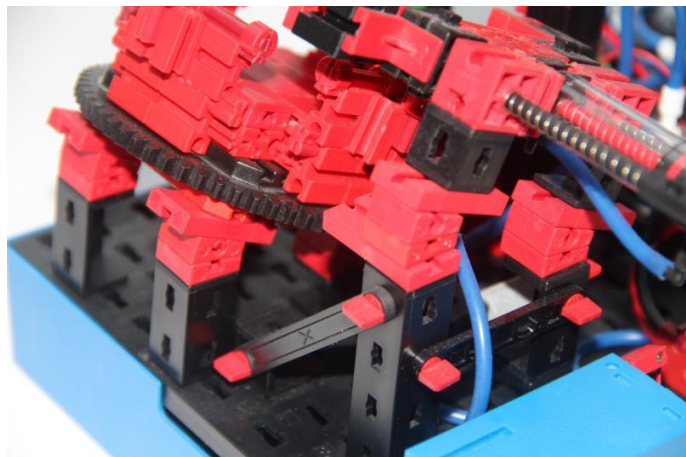
1. dem Drehteller, der den ganzen Würfel um die vertikale Achse dreht
2. dem Greifer, der die mittlere Ebene ggf. fixiert
3. dem Drücker, der den ganzen Würfel kippt

Alle drei Komponenten können einzeln aufgebaut und unabhängig ohne TXT-Controller getestet werden.

Drehteller

Der Drehteller dreht den Würfel um die vertikale Achse. Sein mechanischer Aufbau ist recht einfach zu verstehen. Sein elektrischer Anschluss erfordert allerdings etwas Sorgfalt und ein paar Erklärungen.

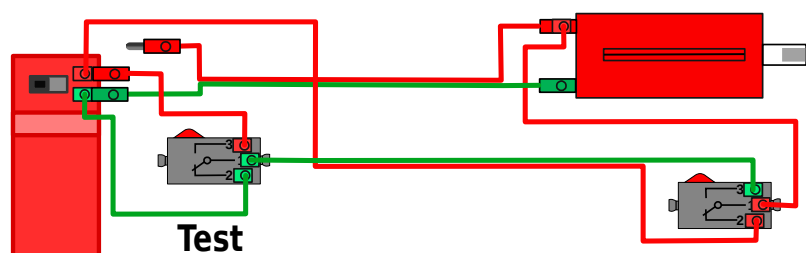
Der Drehteller wird von einem Motor angetrieben und ein Taster erkennt jeweils, wenn eine Drehung um 90° beendet wurde. Der Taster ist so geschaltet, dass sich der Motor automatisch dreht, wenn der Taster nicht gedrückt ist.



Ist der Taster am Motor gedrückt, dann hängt es vom TXT-Controller ab, was passiert. Schaltet der TXT den Ausgang O4 auf Masse (-), so wird der Motor gebremst. Normalerweise dreht sich der Motor noch ein wenig weiter, wenn der TXT-Controller seinen Ausgang ausschaltet. In diesem Fall wird aber der Motor bewusst mit beiden Ausgängen auf Masse (-) verbunden und so aktiv gebremst, bleibst also nahezu schlagartig stehen. Schaltet der TXT-Controller den Ausgang O4 dagegen auf 9V (+), dann dreht sich der Motor. Der Motor soll so angeschlossen sein, dass er sich gegen den Uhrzeigersinn dreht, wenn man von oben auf den Roboter schaut.

In der Praxis dreht der Motor sich immer bis der Taster gedrückt wird. Soll sich der Drehteller um 90° drehen, dann gibt der TXT-Controller einen kurzen Impuls und der Motor fängt an sich zu drehen. Sobald der Motortaster verlassen wurde dreht sich der Drehteller unabhängig vom TXT-Controller bis er in der nächsten 90°-Position ankommt.

Man kann das ganze zunächst auch ohne TXT-Controller mit einer Batterie und einem weiteren Taster ausprobieren wie in nebenstehender Abbildung zu sehen.

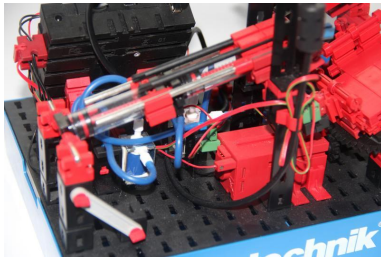


Man kann die genaue Stoppposition des Würfels durch Verschieben des Tasters am Motor einstellen. Der Taster ist korrekt eingestellt, wenn der Würfel sich nach einem kurzen Druck auf den Testtaster einmal 90° dreht und dann stoppt. Dabei soll der Würfel nicht ganz gerade zum Stehen kommen sondern ein paar Grad über das Ziel hinausschiessen. Warum das? Der Würfel liegt etwas lose im Drehteller. Wird er in der Mitte von Greifer festgehalten, dann muss der Drehteller sich ein paar Grad zu weit drehen, damit die untere Ebene des lose darin liegende Würfels so weit gedreht wird, dass der Würfel wieder exakt "glatt" wird. Mehr dazu im Abschnitt „Greifer“.

Die Kamera ist aus dem gleichen Grund mit einem 7,5°-Winkelstein leicht nach links gedreht befestigt, so dass sie den etwas zu weit gedrehten Würfel exakt im Bild hat.

Der Drücker

Der Drücker ist vor allem mechanisch etwas schwieriger. Er besteht aus recht wenigen Teilen, die aber nicht sehr robust verbunden sind.

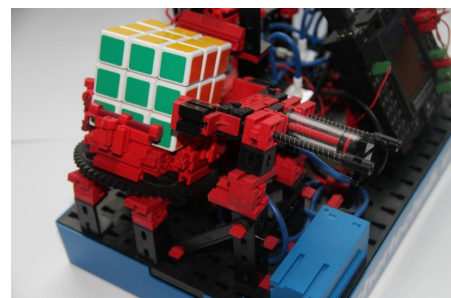


Der Drücker liegt lose im Kameragerüst. Wird er aktiviert, so soll er den Würfel einmal umwerfen, so dass er um 90° kippt. Das funktioniert nur, wenn der Würfel am Ende wieder genau im Drehteller liegt. Um das sicherzustellen ist der ganze Aufbau um 22,5 Grad gekippt. Der kippende Würfel rutscht so wieder zurück in den Drehteller.

Auch den Drücker kann man ohne TXT testen. Dazu schließt man das Magnetventil einfach an einen Taster und eine Batterie an.

Der Greifer

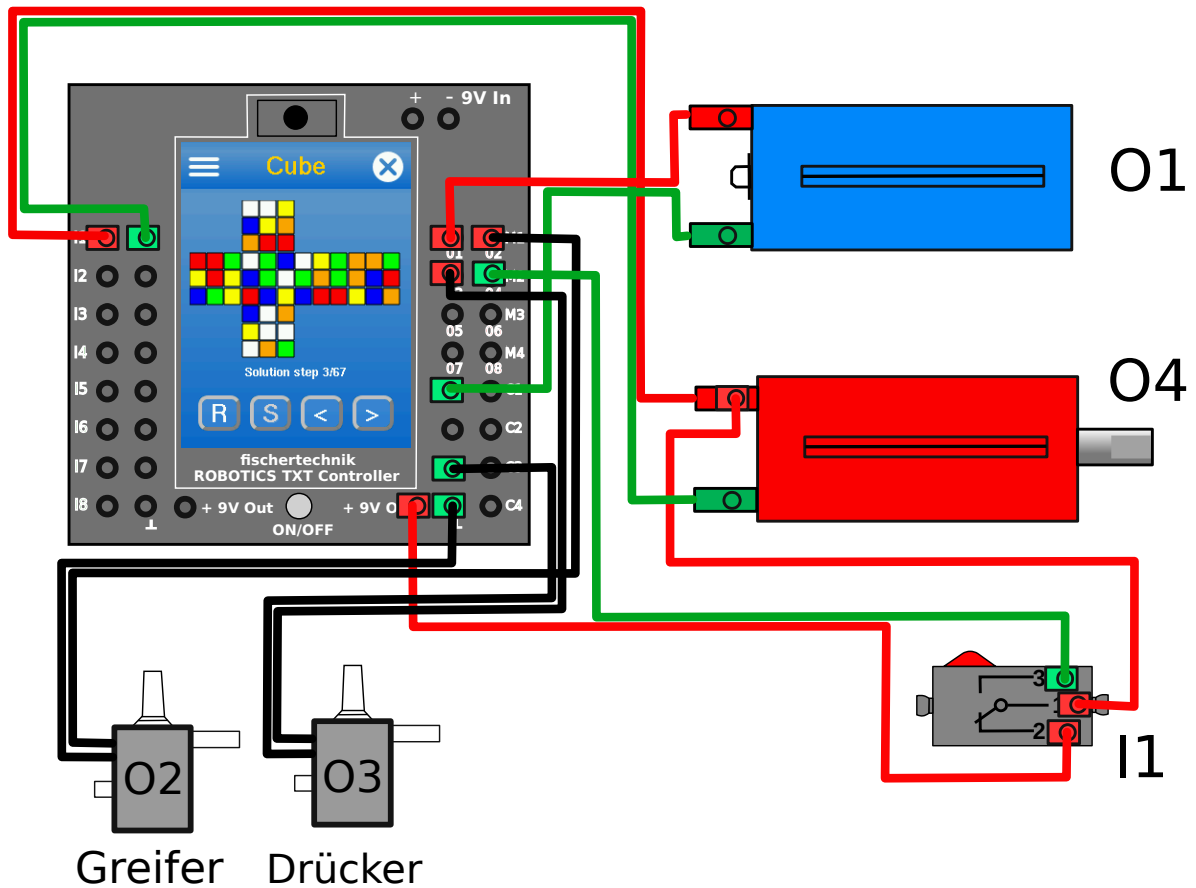
Der Greifer erfüllt seine Funktion im Zusammenspiel mit dem Drehteller. Normalerweise würde der Drehteller immer den ganzen Würfel drehen. Um den Würfel zu lösen muss der Roboter aber eine einzelne Seite des Würfels drehen. Dazu kann der Greifer die mittlere Ebene des Würfels greifen und fixieren. Dreht sich dann der Drehteller, so wird nur die untere im Drehteller liegende Ebene gedreht.



Auch das kann man wieder mit Batterie und Taster testen. Es ist hilfreich, auch die Tastersteuerung des Drehtellers noch angeschlossen zu haben. So kann man das Zusammenspiel von Greifer und Drehteller ausprobieren. Dabei sollte nun auch offensichtlich werden, warum der Drehteller sich etwas zu weit drehen muss, wenn der Greifer verwendet wird.

Verdrahtungsschema

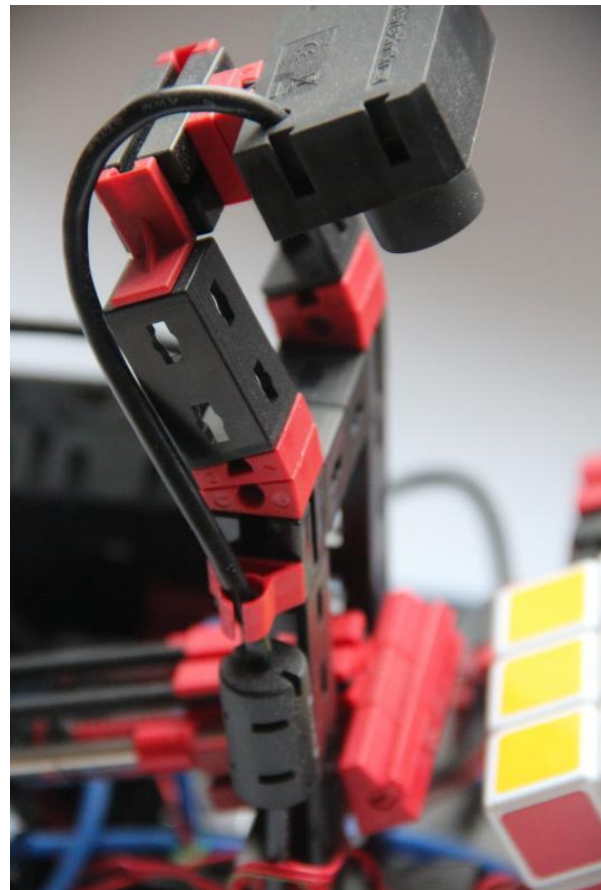
Ist der mechanische Aufbau abgeschlossen und die Mechanik getestet, dann kann der TXT verkabelt werden.



Die Kamera

Die größte Hürde bei der Inbetriebnahme des Cube-Solvers ist die Erkennung durch die Kamera. Dazu muss die Kamera insgesamt sechs Einzelbilder des Würfels aufnehmen und exakt erkennen, welche Farben sich an welcher Stelle des Würfels befinden.

Kritisch ist vor allem die Erkennung der Farbtöne. Die Farben werden durch Umgebungslicht, die Art der Lichtquelle und den übrigen Bildinhalt beeinflusst. Normalerweise führen Kameras dazu einen automatischen Weissabgleich durch ("automatic white balance"), bei dem die Kamera die Farben anhand weisser Elemente im Bild selbst einstellt. Das funktioniert aber nur dann halbwegs zuverlässig, wenn sich tatsächlich weisse Elemente im Bild befinden. Das ist bei einem Zauberwürfel im überwiegend roten Cube-Solver-Roboter nicht immer gegeben. Im direkten Vergleich sind Farben wie Rot und Orange auf dem Kamerabild oft noch halbwegs zu unterscheiden. Bei mehreren Aufnahmen, bei denen ggf. auch der Weissanteil im Bild schwankt, können die Unterschiede sehr groß werden. Das kann in einigen Fällen so weit gehen, dass eine komplett rote Würfelseite zusammen mit dem weitgehend roten Roboter zu einem nahezu farblos-grauen Bild führt. Die Kamera geht



schlicht davon aus, dass der Rotton in der Menge wohl unerwünscht ist. Aus diesem Grund kann es helfen, einen Würfel aus weissem Plastik zu verwenden. So ist immer etwas weisses Plastik im Bild, an dem sich die Kamera orientieren kann. Es stört dabei nicht, dass sich das weisse Plastik ausserhalb der Flächen befindet, die in der späteren Bildverarbeitung zur Farbanalyse herangezogen werden. In der "Calibration"-Ansicht des Cube-App sollte dieser weisse Teil komplett ausgeblendet sein. Trotzdem nutzt die Kamera diesen Teil.

Es gibt mindestens zwei Kameras von fischertechnik, die fatalerweise identisch aussehen. Beide Kameras wurden erfolgreich im Cube-Solver eingesetzt. Allerdings ist der ältere Kamerateyp wesentlich anfälliger für Falschfarberkennung und einen übertriebenen Weissabgleich. Die neuere Kamera ist dagegen sehr viel robuster und ist z.B. in der Lage einen schwarzen Original-Rubiks-Cube zu erkennen. Die Farbschwankungen unter schlechten Bedingungen sind sehr viel geringer.

Eine Möglichkeit, die beiden Kamerateypen zu unterscheiden ist ihre sogenannte USB-ID. Sie lässt sich z.B. unter Windows im Gerätemanager in Erfahrung bringen oder unter Linux mit dem lsusb-Kommando. Die alte Kamera nutzt die USB-ID 1b71:0056, während die neue und besser geeignete Kamera die USB-ID 0c45:6340 verwendet.

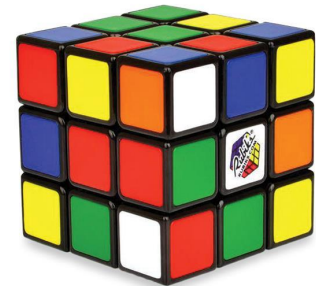
Der Würfel

Der Zauberwürfel selbst ist für dieses Projekt von entscheidender Bedeutung. Drei Faktoren haben sich als wichtig herausgestellt:

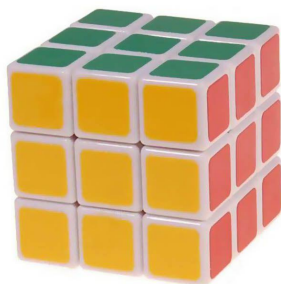
1. Der Würfel muss leichtgängig sein
2. Die Farben müssen für die Kamera unterscheidbar sein
3. Das Plastik sollte weiss sein

Die Leichtgängigkeit wird dadurch bedingt, dass die Robotermechanik nicht sehr kraftvoll ist. Außerdem passiert es immer mal wieder, dass der Roboter eine Seiten-Drehung nicht perfekt ausführt und der Würfel nicht ganz "glatt" ist. Wenn dann eine andere Drehung ausgeführt wird verkantet sich die ganze Würfelmechanik etwas. Je leichtgängiger der Würfel ist, desto eher rutschen die Würfelteile in so einem Fall doch noch an die richtige Stelle. Sogenannte "Speed Cubes" sind hier in der Regel besser. Vor allem haben solche Cubes Schrauben unter den abnehmbaren Deckeln der jeweiligen Mittelfelder. Diese Schrauben lassen sich etws lösen und der Würfel dreht leichter. Alternativ kann der Einsatz von Schmiermitteln wie Silikonfett helfen. Der Original-Rubiks-Cube z.B. hat keine Schrauben, die man lockern könnte, und Silikonfett kann ggf. helfen.

Die Farben der Aufkleber des Würfels müssen möglichst dem Original entsprechen. Das betrifft die Anordnung der Farben zueinander wie auch die Farben an sich. So muss z.B. Rot gegenüber Orange sein, Weiss gegenüber Gelb und Blau gegenüber Grün. Kritisch haben sich bei den Farben an sich vor allem Rot und Orange herausgestellt. Auch wenn das menschliche Auge diese Alarm-Farben gut unterscheidet liegen sie oft sehr nah beieinander. Die "Farbkorrektur" der Kamera kann hier fatale Auswirkungen zeigen. Speziell Neonfarbtöne haben sich als problematisch herausgestellt. Die klassischen etwas dezenteren Farben des Originalwürfels sind dagegen besser. Einige Würfel tragen ein Logo auf dem zentralen weissen Feld. In einigen Fällen wird der Würfel trotzdem erkannt. Alternativ muss man das Logo abkleben. Nicht getestet wurden bisher Würfel, die durchgefärbte Seiten statt der Aufkleber haben.



Schließlich spielt die Plastikfarbe des eigentlichen Würfels eine Rolle. Vor allem bei der Verwendung der älteren fischertechnik-Kamera hilft das weisse Plastik der Kamera beim sogenannten Weissabgleich. Findet eine solche Kamera überhaupt kein Weiss im Bild, dann kann es zu deutlichen



Farbabweichungen iom Bild kommen und speziell die Erkennung der Rot- und Orangetöne gelingt nur selten. Alternativ kann man auch ein weisses Objekt im seitlichen Bildbereich der Kamera unterbringen. Der Original-Rubiks-Cube ist aus schwarzem Plastik gefertigt und bietet der alten Kamera wenig Chancen zur korrekten Farberkennung. Neuere Kameras können auch den Original-Würfel korrekt erfassen, i.d.R. sogar mit dem Logo auf dem zentralen weissen Feld.

Die Farberkennung lässt sich in der Cube-App im Menüpunkt "Calibration" testen. Dort kann man sehen, welche Farben der Roboter zu erkennen glaubt und in gewissen Grenzen kann man die Farberkennung anpassen.

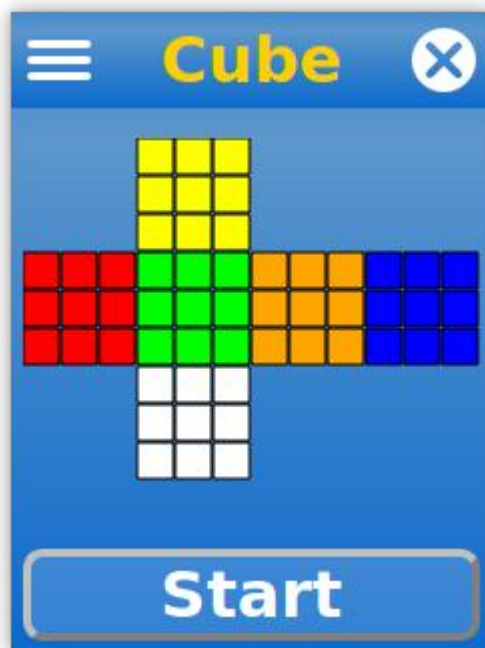
Cube-App

Die Software, die auf dem TXT zum Einsatz kommt, basiert auf der sogenannten [Community-Firmware](#), einem alternativen Betriebssystem für den TXT-Controller, das u.a. die Programmierung von Anwendungen (Apps) in der Programmiersprache Python unterstützt und auch die Einbindung sogenannter nativer Anwendungen erlaubt.

Die eigentliche Cube-App lässt sich in der Community-Firmware in der mitgelieferten Store-App leicht nachinstallieren. Unter <https://github.com/harbaum/cfw-apps> findet sich u.a. die Cube-App inklusive einer Beschreibung der Installation.

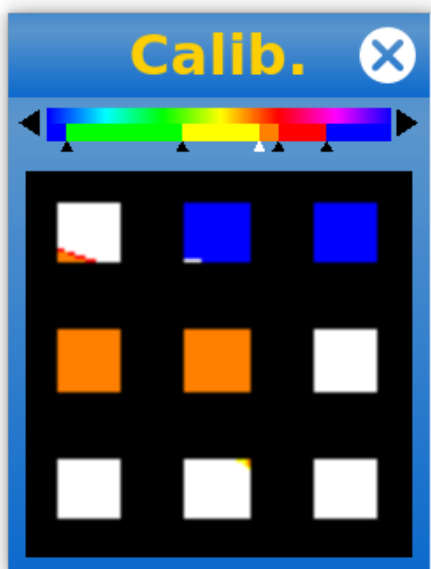
Die grafische Oberfläche der Cube-App ist in Python geschrieben. Der interessierte Nutzer findet sie unter <https://github.com/harbaum/cfw-apps/tree/master/packages/cube>. Die App nutzt intern das [PyCuber-Projekt](#) zur Darstellung des Zauberwürfelzustandes in Python-Programmen.

Der Lösungsalgorithmus wurde aus dem [Kociemba-Projekt von Maxim Tsoy](#) übernommen.



Kalibrierungsbildschirm

Wichtig für die Inbetriebnahme ist der Kalibrierungsbildschirm, der im Menü der App unter „Calibration“ zu finden ist. Dieser Bildschirm zeigt die Farben so, wie sie der Erkennungsalgorithmus erfasst.



Die Grenzen zwischen den fünf Farben sind im oberen Bereich des Bildschirms einstellbar. Dazu wählt man einen der kleinen Trennpfeile unterhalb des Farbbalkens aus. Der ausgewählte weisse Trenner kann dann mit den beiden Pfeilen links und rechts des Farbbalkens bewegt werden.

Wird z.B. ein oranges Feld fälschlicherweise als rot erkannt, dann kann man den Trennpfeil zwischen orange und rot auswählen und mit dem rechten Pfeil weiter nach rechts bewegen. Der als orange akzeptierte Bereich wird dadurch etwas in den roten Bereich hinein erweitert.

Die Erkennung von weissen Flächen kann nicht beeinflusst werden, da diese Flächen nicht am Farbwert erkannt werden sondern mit Hilfe der bei Weiss fehlenden Farbsättigung.

Im Kalibrierungsbildschirm ist auch erkennbar, ob der Würfel exakt in der Erkennungsmaske liegt. Ggf. muss die Kamera leicht geneigt oder gedreht werden, damit die Farbfelder der Erkennung vollständig gefüllt sind.

Die Lösungsberechnung

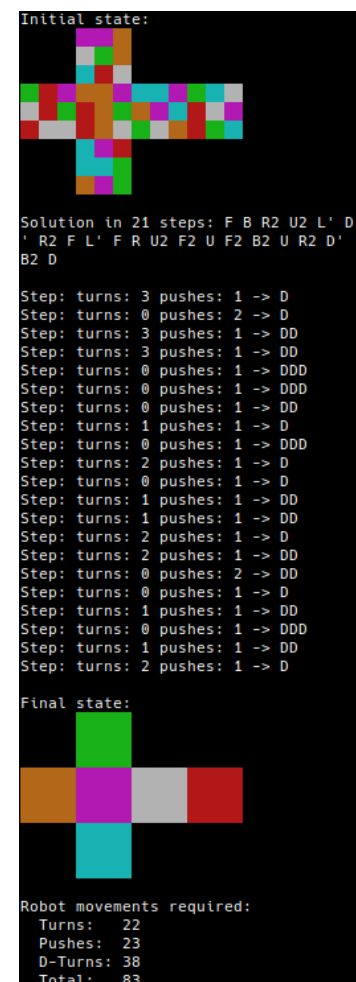
Der "magische" Teil des Cube-Solvers steckt sicher in der eigentlichen Berechnung der Lösung. Sie hat entscheidenden Einfluss auf die Laufzeit des Roboters, denn es gibt nicht einfach "die" Lösung. Bekannt wurde in den 80ern die [Lösung aus dem "Spiegel"](#). Mit ein paar auswendig zu lernenden Zügen konnte man so den Würfel lösen. Je nach Ausgangsstellung benötigt man damit gut 100 Einzelzüge, um den Würfel zu lösen. Unser Roboter braucht für 100 Züge (kippen oder drehen) weniger als zwei Minuten.

Leider bleibt es nicht dabei. Während ein Mensch problemlos jede Seite des Würfels einzeln drehen kann beherrscht der Roboter nur Drehungen der Unterseite (die Drehung des sogenannten "D-Face" in Cuber-Fachsprache). Soll eine andere Seite gedreht werden, so muss der Roboter diese Seite erst nach unten bekommen. Zusätzlich kann er diese Seite nur in eine Richtung drehen. Soll eine Drehung in die Gegenrichtung erfolgen, so muss stattdessen dreimal gedreht werden. In der Regel vervierfacht sich dadurch die Zahl der nötigen Züge und der Roboter bräuchte über 400 Züge bzw. rund acht Minuten zur Lösung.

An dieser Stelle lohnt sich der Einsatz von Rechenleistung. Lange war die Frage offen, was die maximale Zahl an Zügen ist, die nötig ist, um jeden Würfel lösen zu können. Erst 2010 wurde bewiesen, dass diese auch als "God's Number" bezeichnete Zahl der Züge 20 ist (mehr unter <http://kociemba.org/cube.htm>). Der Two-Phase-Algorithmus von Herbert Kociemba kann eine Lösung mit maximal ca 20 Zügen für jeden beliebig verdrehten Würfel in wenigen Sekunden errechnen. Auch der Cube-Solver beinhaltet den Two-Phase-Algorithmus.

Aus dieser maximal 20 Schritte langen Folge berechnet der Cube-Solver die von ihm aufgrund der mechanischen Beschränkungen nutzbare Zugfolge. In der Regel besteht das Ergebnis aus knapp 100 Zügen, die der Roboter in ca. zwei Minuten ausführt.

Das Bild auf der rechten Seite zeigt einen Test des Algorithmus. Der Test startet mit einem simulierten verdrehten Würfel, dessen Lösung durch den Two-Face-Algorithmus berechnet wird. Die Lösung hat in diesem Fall 21 Schritte. Diese Schritte werden dann in die drei möglichen Bewegungen des Roboters umgewandelt („turn“, „push“ und „D-turn“, also drehen, kippen und das Drehen der einzelnen Unterseite). Aus den ursprünglich 21 Schritten werden so nun insgesamt 83 Bewegungen des Roboters.



```
Initial state:
[Diagram of a scrambled 3x3x3 cube]

Solution in 21 steps: F B R2 U2 L' D
' R2 F L' F R U2 F2 U F2 B2 U R2 D'
B2 D

Step: turns: 3 pushes: 1 -> D
Step: turns: 0 pushes: 2 -> D
Step: turns: 3 pushes: 1 -> DD
Step: turns: 3 pushes: 1 -> DD
Step: turns: 0 pushes: 1 -> DDD
Step: turns: 0 pushes: 1 -> DDD
Step: turns: 0 pushes: 1 -> DD
Step: turns: 1 pushes: 1 -> D
Step: turns: 0 pushes: 1 -> DDD
Step: turns: 2 pushes: 1 -> D
Step: turns: 0 pushes: 1 -> D
Step: turns: 1 pushes: 1 -> DD
Step: turns: 1 pushes: 1 -> DD
Step: turns: 2 pushes: 1 -> D
Step: turns: 2 pushes: 1 -> DD
Step: turns: 0 pushes: 2 -> DD
Step: turns: 0 pushes: 1 -> D
Step: turns: 1 pushes: 1 -> DD
Step: turns: 0 pushes: 1 -> DD
Step: turns: 1 pushes: 1 -> DD
Step: turns: 2 pushes: 1 -> D

Final state:
[Diagram of a solved 3x3x3 cube]

Robot movements required:
Turns: 22
Pushes: 23
D-Turns: 38
Total: 83
```