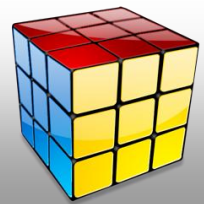


Informatikprojekt: Cube Solver

Dokumentation

Von Jakob Seiwert und Felix Kirchmann

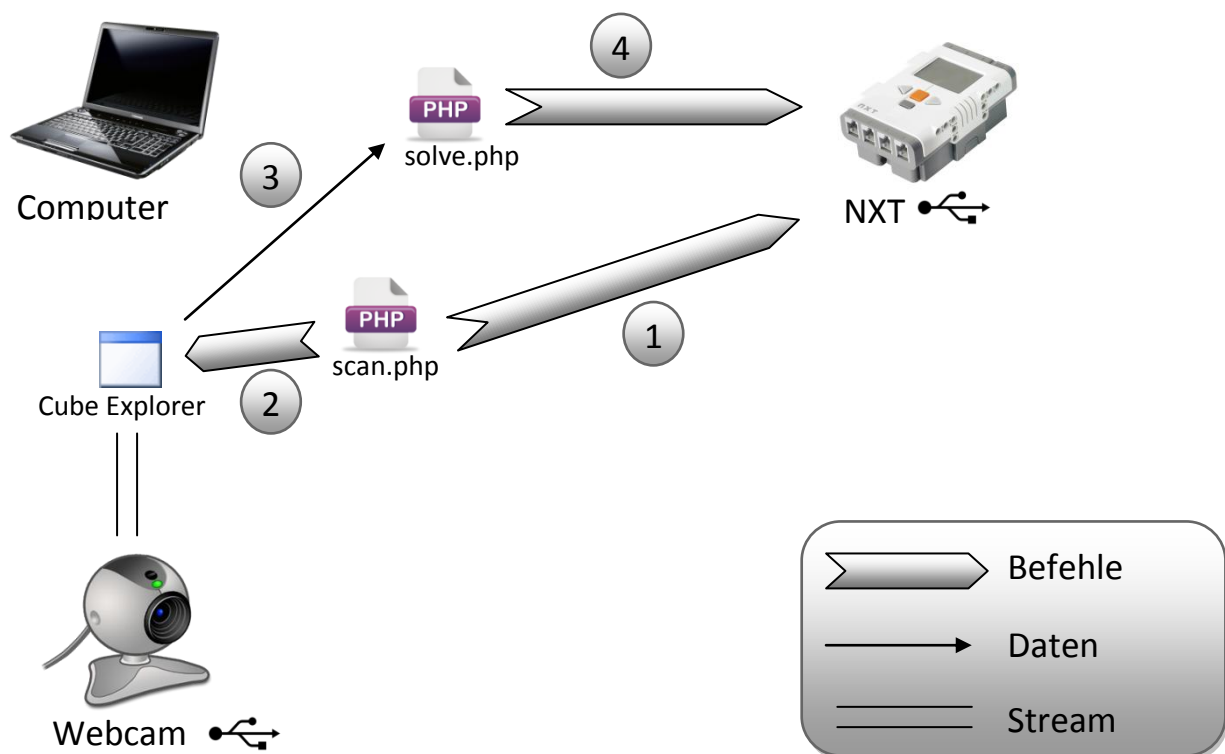


Aufgabenstellung

Der Roboter soll einen Rubik's Cube einscannen und lösen.

Umsetzung / Funktionsweise

→ Diagramm



→ Erklärung

1	Das Scan-Skript befiehlt dem Roboter, den Würfel von allen Seiten sichtbar zu machen
2	Der Cube Explorer scannt die Würfelseiten ein und berechnet den Lösungsalgorithmus, der dann in einer Textdatei gespeichert wird
3	Das Lösungsskript öffnet die Textdatei, liest den Inhalt und berechnet die Roboterbewegungen
4	Der Roboter empfängt die Befehle vom Lösungsskript



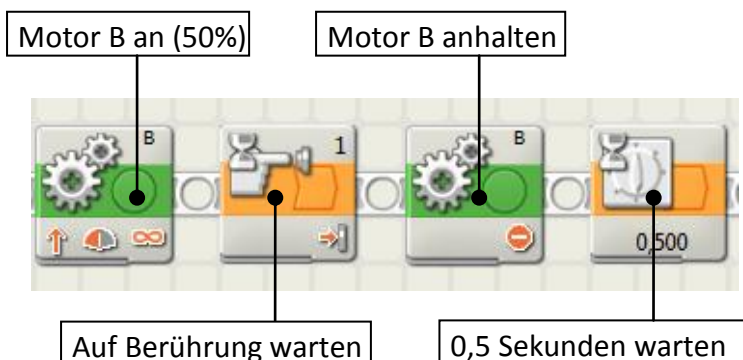
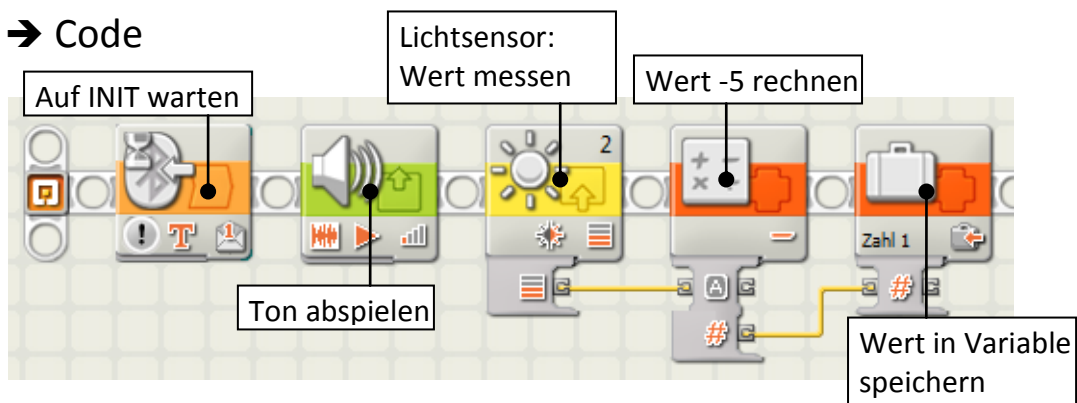
CubeClient.rbt

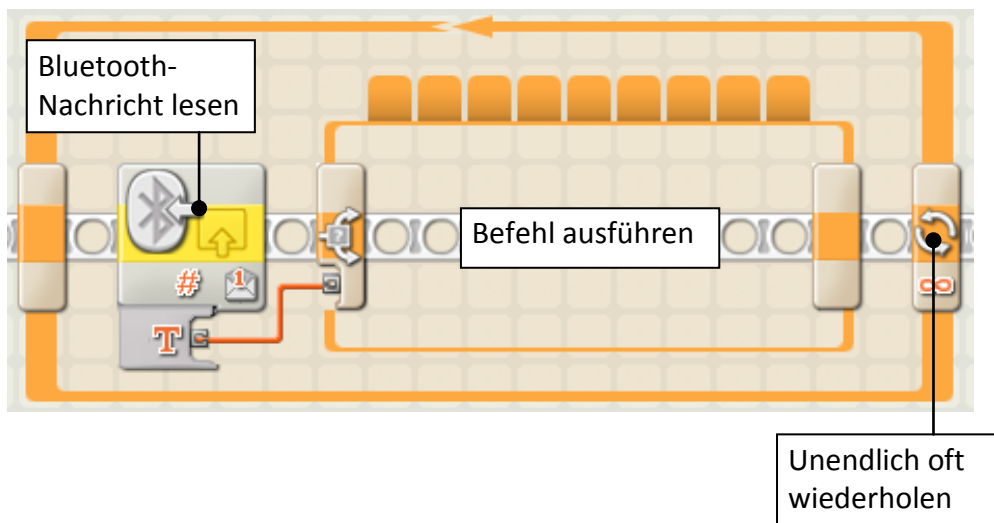
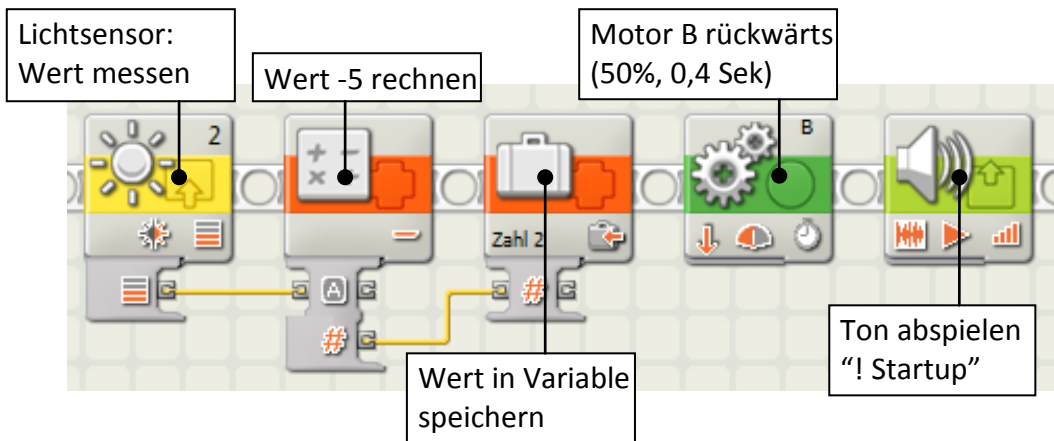
Dieses Programm, geschrieben in NXT-G, empfängt Befehle vom Computer (via USB oder Bluetooth, Inbox 1) und führt entsprechende Bewegungen aus.

→ Befehle

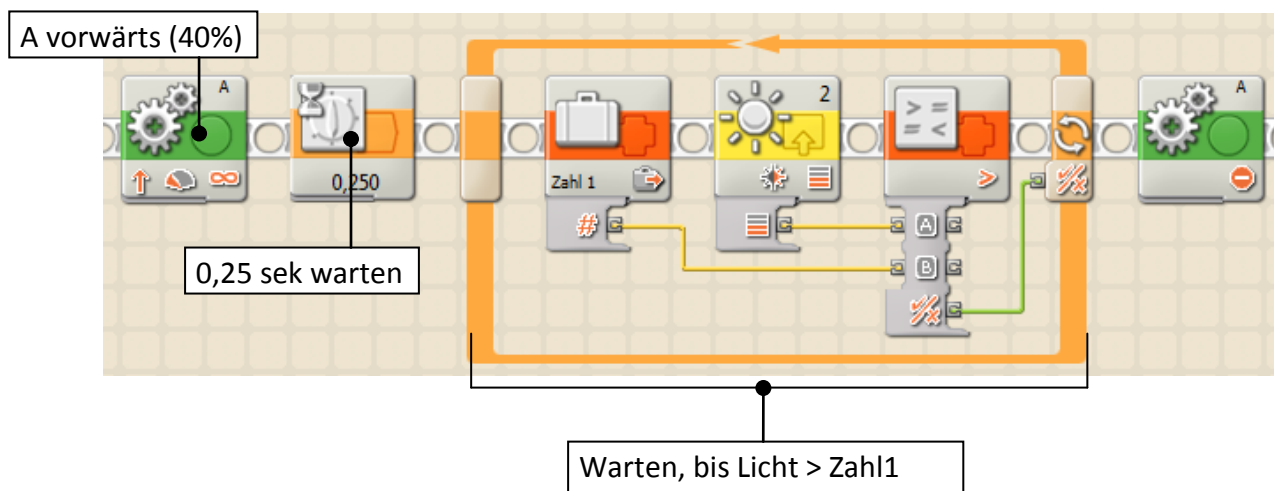
Nachricht	Befehl
INIT	Kalibriert die Sensoren und initialisiert das Programm, muss immer als erstes aufgerufen werden.
10.	Pod 90° nach links drehen
11.	Pod 90° nach rechts (im Uhrzeigersinn) drehen
30.	Aktiviert den Schubser
40.	Fährt den Greifer herunter
41.	Fährt den Greifer hoch
100.	Zeigt ein Häkchen auf dem Display und beendet das Programm
CALIB	Dreht die Plattform schnell nach links und rechts, um festgesteckte Würfelsegmente zu lösen

→ Code

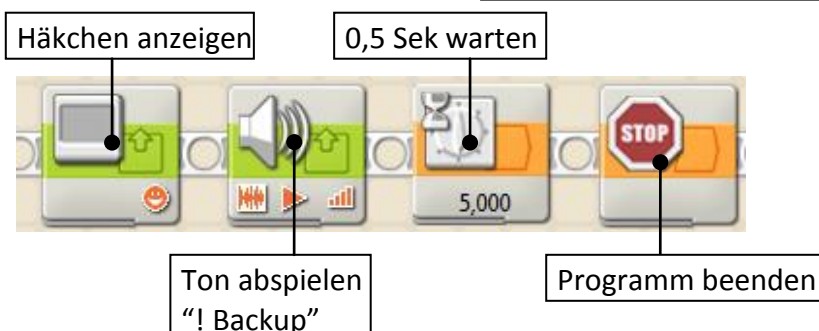




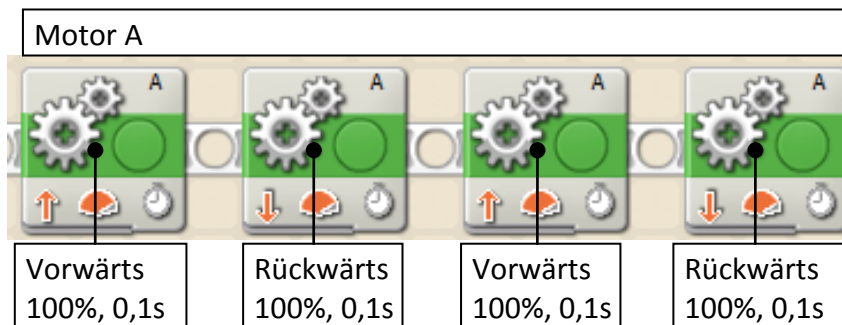
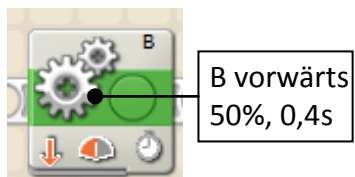
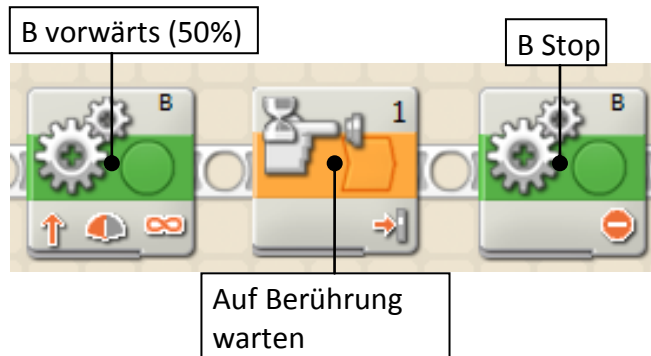
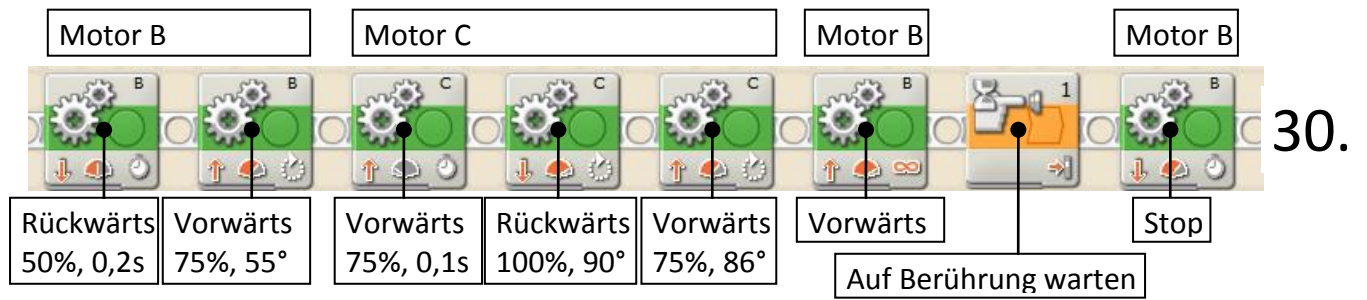
➔ Befehle: Code



10.



100.





lib/librobo.php

librobo stellt grundlegende Funktionalitäten bereit, um Nachrichten bzw. Befehle via Bluetooth oder USB an den NXT zu übertragen. Benötigt wird eine intakte BricxCC Installation bzw. das NeXTTool.

➔ Funktionen

robomsg (\$message)	Sendet die Nachricht \$message auf Inbox 1 an den Roboter
---------------------	---



lib/libmouse.php

libmouse ermöglicht das Steuern der Cursorposition und Maustasten via PHP.

➔ Funktionen

mouse_moveto (\$X, \$Y)	Bewegt den Cursor zu Position (\$X / \$Y)
mouse_click ()	Löst einen Linksklick aus
mouse_rightclick ()	Löst einen Rechtsklick aus
mouse_posclick (\$X, \$Y)	Bewegt den Cursor zu Position (\$X / \$Y) und löst einen Linksklick aus



lib/libcugo.php

libcugo berechnet für jede Bewegung des Würfels den optimalen Weg und führt ihn aus.

➔ Funktionen

cugo_process (\$from, \$to)	Bewegt den Würfel von Position \$from zu Position \$to
-----------------------------	--



scan.php

Das Scan-Skript bewegt die einzelnen Seiten des Würfels in das Blickfeld der Webcam und sorgt dafür, dass der Cube Explorer sie scannt. Aufgrund der Unzuverlässigkeit des Web API's wird libmouse zur Steuerung des Cube Explorers verwendet.



solve.php

Das Solve-Skript liest den in Solvestring.txt gespeicherten Lösungsalgorithmus aus und lässt den Roboter sie lösen. Zum Beschleunigen wird libcugo verwendet.



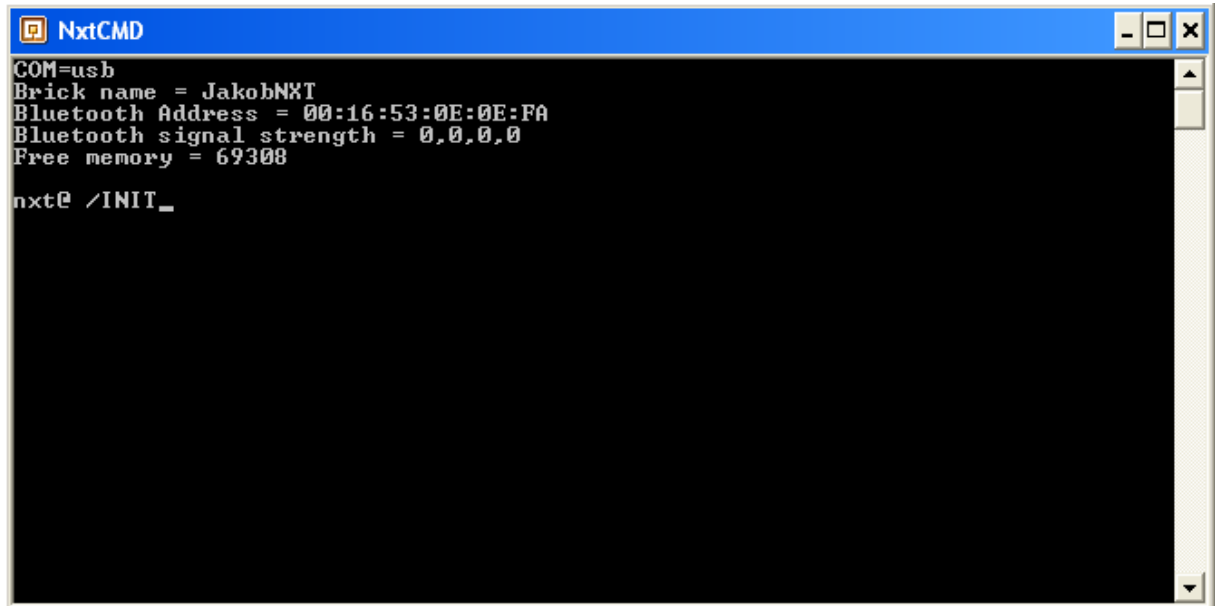
Verwendete PHP-Befehle

<code>include (\$file);</code>	Inkludiert ein in \$file angegebenes PHP-Skript
<code>echo (\$text);</code>	Gibt \$text aus
<code>if (condition) { command(); }</code>	Führt <code>command()</code> aus, wenn <code>condition == true</code>
<code>foreach (\$array) { command(); }</code>	Führt <code>command()</code> für jede Datenreihe in \$array aus
<code>exec (\$cmd);</code>	Führt den in \$cmd angegebenen Kommandozeilen-Befehl aus
<code>sleep (\$secs);</code>	Wartet \$secs Sekunden, bis das Skript weiter ausgeführt wird
<code>usleep (\$usecs);</code>	Wartet \$usecs Mikrosekunden, bis das Skript weiter ausgeführt wird
<code>str_replace (\$find, \$replace, \$string);</code>	Ersetzt jedes Vorkommen von \$find in \$string durch \$replace
<code>explode(\$divisor, \$string)</code>	Teilt \$string in durch \$divisor Stücke auf, und gibt diese Stücke als Array zurück
<code>fopen (\$file, \$mode)</code>	Öffnet die Datei \$file im Modus \$mode (z.B. „r“ für Lesezugriff) und gibt einen Handle zurück
<code>fread (\$handle, \$length)</code>	Liest \$length Bytes aus \$handle und gibt diese als String zurück
<code>fclose (\$handle)</code>	Schließt den Dateihandle \$handle

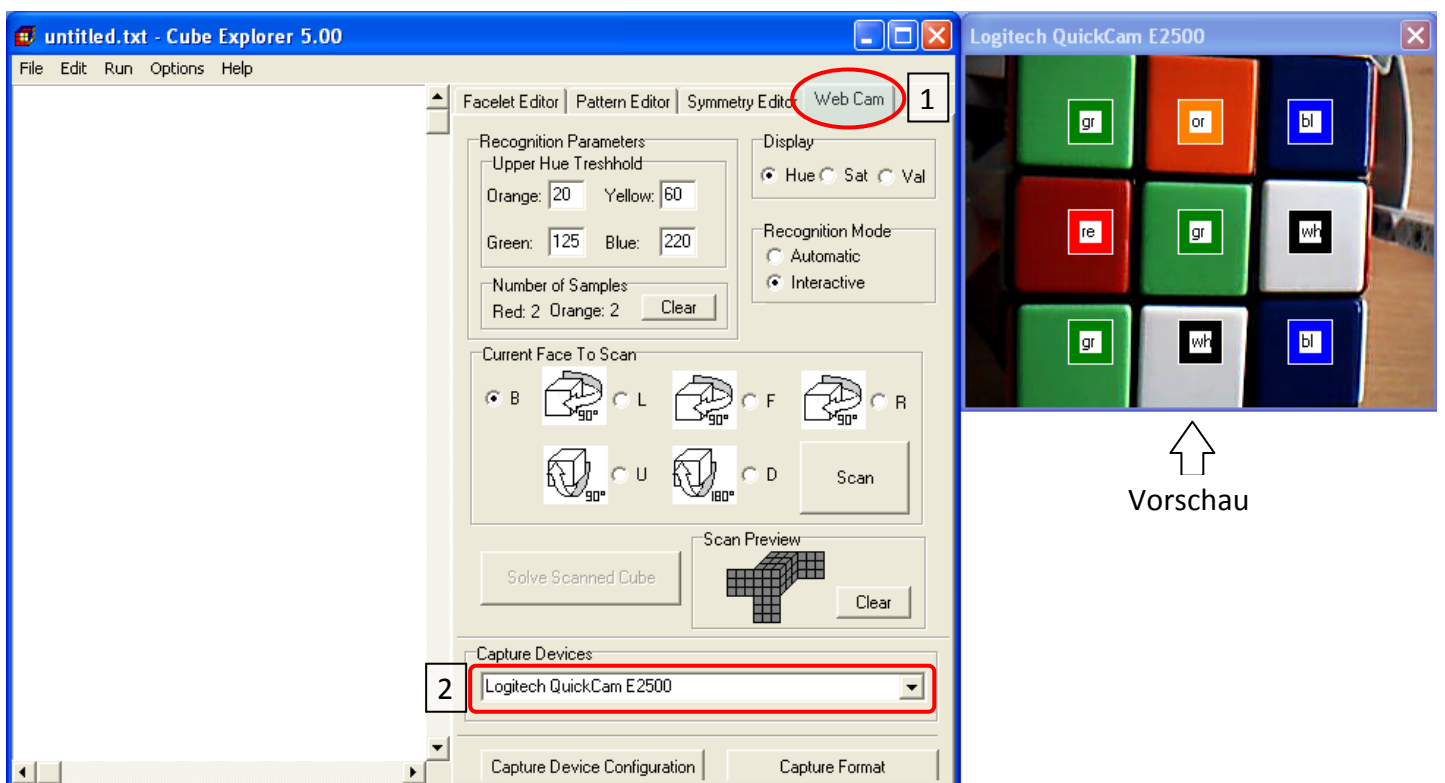


Den Rubik's Cube lösen

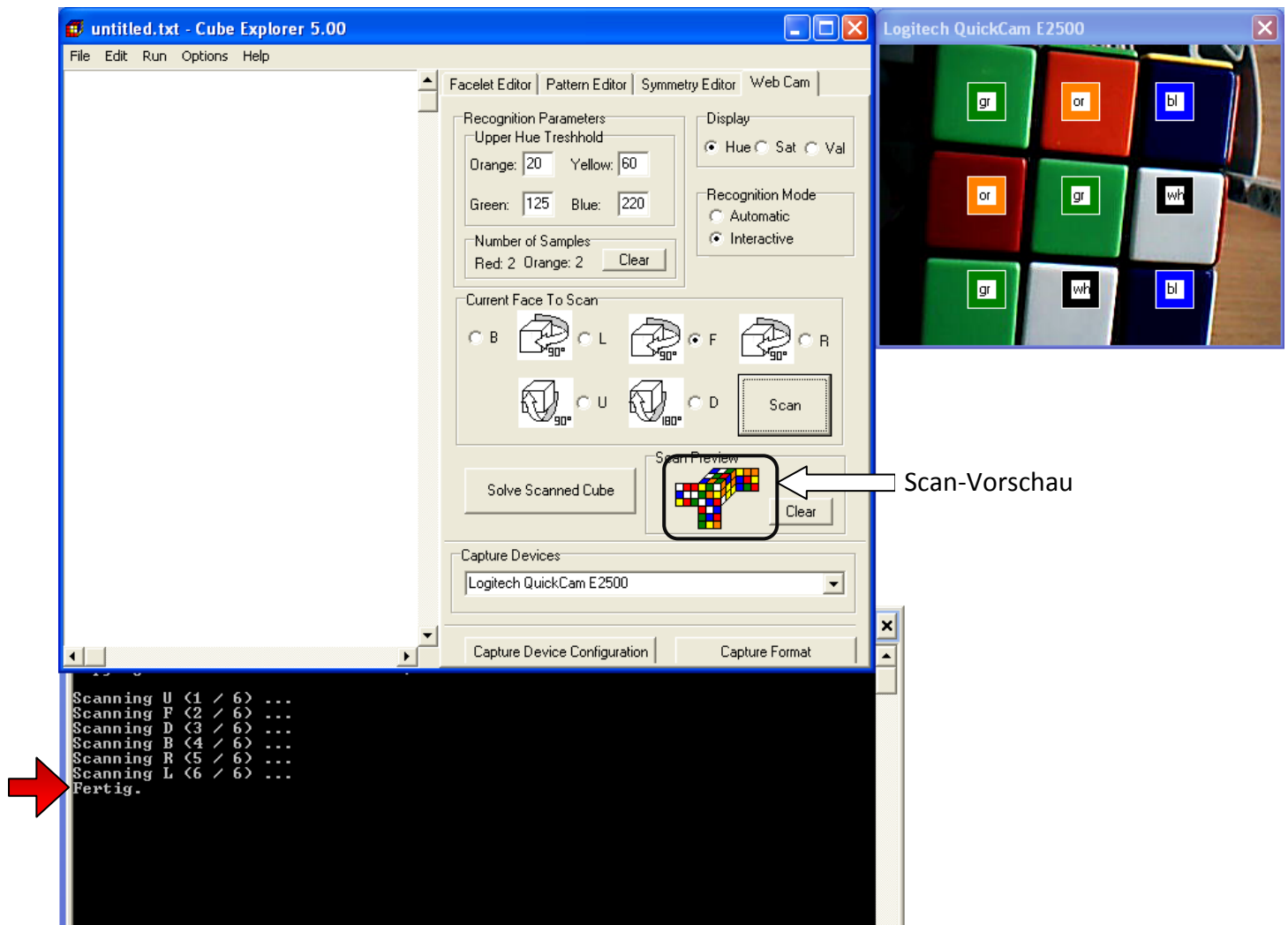
1. Man schließt den NXT via USB an den Computer an und startet beide.
2. Nun sendet man den Befehl zur Initialisierung an den NXT:



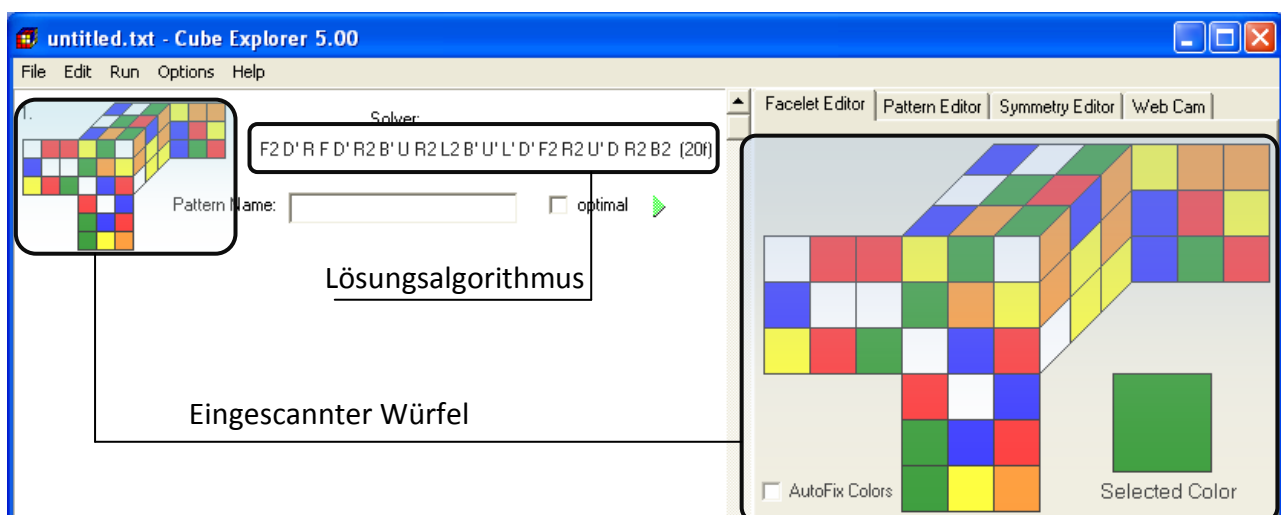
3. Wenn der Roboter seinen Lichtsensor fertig kalibriert hat, kann man den Cube Explorer starten, den Webcam-Tab anklicken und seine Kamera auswählen:



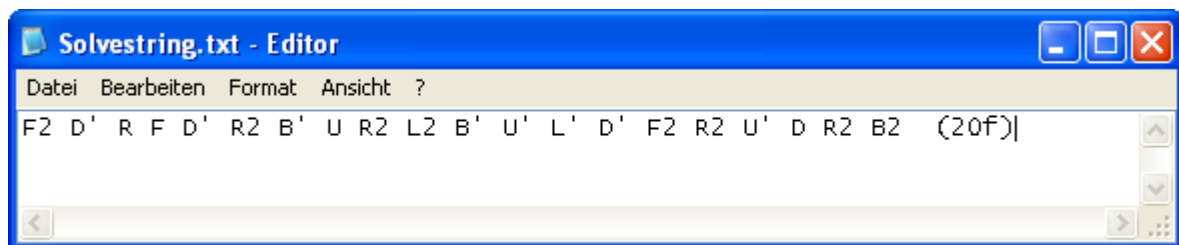
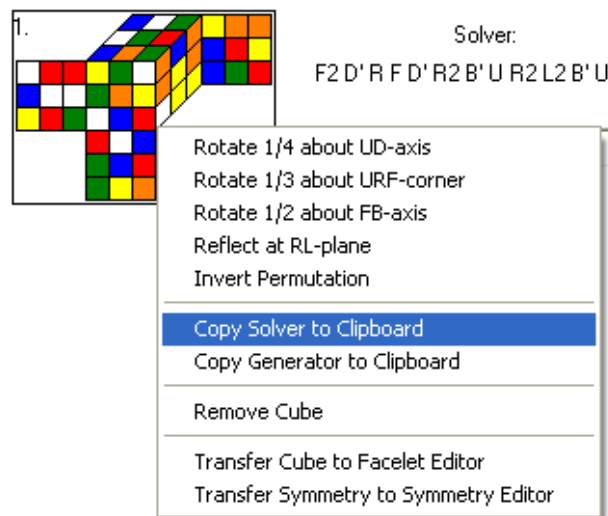
4. Jetzt kann man den Würfel mithilfe des Roboters scannen. Dazu zieht man das Scan-Skript (scan.php) per Drag'n'Drop auf die run.bat. Es sollte sich ein weiteres Fenster öffnen, welches „Fertig.“ ausgeben wird, sobald der Scanprozess abgeschlossen ist.



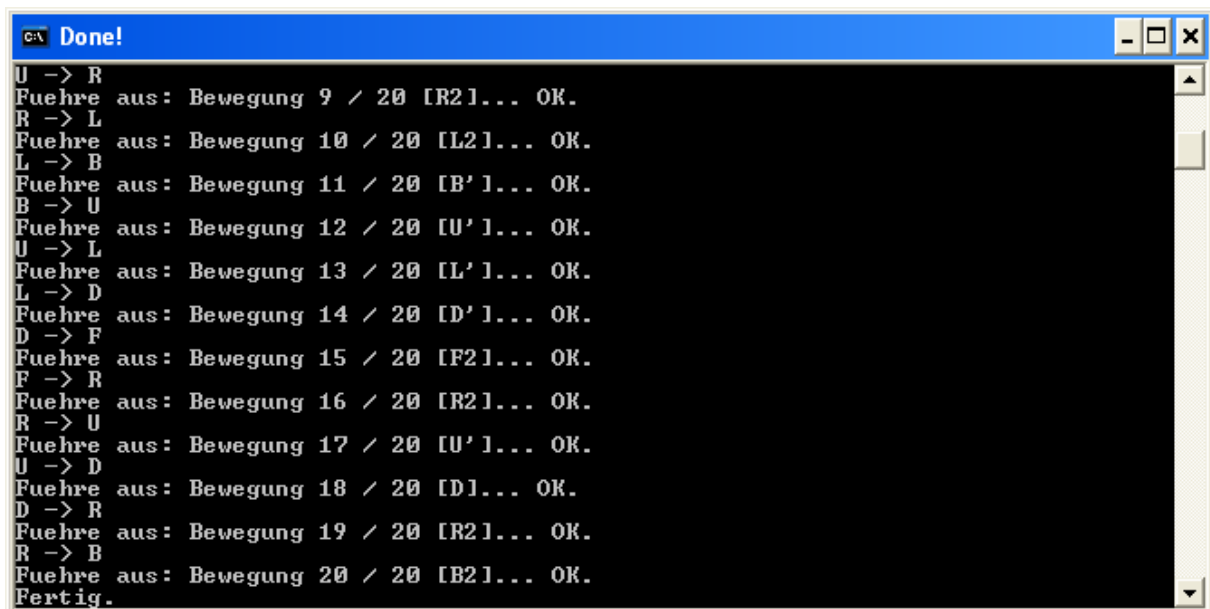
5. Wurde der Würfel korrekt gescannt, so kann man auf „Solve Scanned Cube klicken. Der Würfel wird nun ins Hauptfenster übertragen. Gleichzeitig berechnet der Cube Explorer einen Lösungsalgorithmus.



6. Man kopiert den Algorithmus in Solvestring.txt,



7. und lässt den Roboter den Würfel lösen, indem man solve.php ausführt.



Arbeitsprotokoll

16.03.2010:	Auswahl der Aufgabe, Abgabe des Projektbogens
25.03.2010:	Fertigstellung des ersten Roboterprototyps, anfängliche Programmierarbeit
30.03.2010:	Client weiter verbessert und getestet
08.04.2010:	Client weiter verbessert und getestet
13.04.2010:	PHP-Skript überarbeitet, NXT-Kommunikation funktioniert nun
22.04.2010:	Trennung des Backends in zwei Teile: Skripte und lib's
27.04.2010:	Reprogrammierung des Clients aufgrund notwendiger Modifikationen am Roboter
29.04.2010:	Client weiter verbessert und getestet
04.05.2010:	Client weiter verbessert und getestet
13.05.2010:	Lichtsensoren eingebaut, automatische Kalibrierung implementiert
18.05.2010:	Stabilitätstests
27.05.2010:	Problem: Unzuverlässige Würfelmodifikation, behoben durch zusätzliche Kalibrierung
01.06.2010:	Erste Arbeiten am Webcam-Scan, libmouse v 0.1
15.06.2010:	Bugfix und Testing des Webcam-Scan
24.06.2010:	<i>(Vorstellung)</i>

Heimarbeit ist nicht aufgelistet.