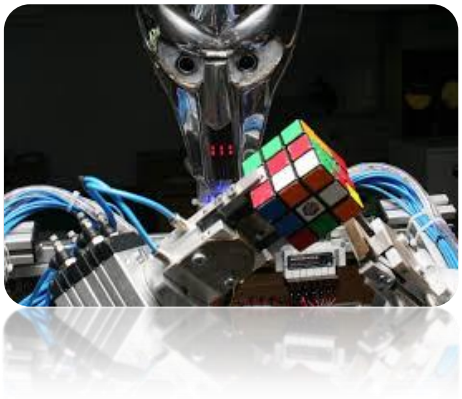
Πολυτεχνείο Κρήτης

Σχολή Μηχανικών Παραγωγής και Διοίκησης
Εργαστήριο Βιομηχανικών, Ενεργειακών
και Περιβαλλοντικών Συστημάτων

Διπλωματική εργασία

Επίλυση του κύβου του Rubik με χρήση του Lego-NXT και MATLAB



Ντουρτόγλου Γεώργιος

Αύγουστος 2018

Επιβλέπων καθηγητής:
Αναστάσιος Πουλιέζος

Ευχαριστίες

Η παρούσα διπλωματική υλοποιήθηκε στο εργαστήριο Βιομηχανικών, Ενεργειακών και Περιβαλλοντικών Συστημάτων υπό την επίβλεψη του καθηγητή Αναστάσιου Πουλιέζου τον οποίο θα ήθελα να ευχαριστήσω για την άριστη συνεργασία καθ' όλη την πορεία εκπόνησής της. Τέλος, ευχαριστώ την οικογένειά μου και τους φίλους μου για τη στήριξη και την υπομονή τους σε όλη την διάρκεια των σπουδών μου.

Ντουρτόγλου Γεώργιος,
Χανιά, 20/08/2018

Περίληψη

Στόχος της προτεινόμενης διπλωματικής είναι η εξοικείωση με τα συστήματα αυτοματισμού, που αποτελούνται από το σετ Lego mindstorm NXT,το περιβάλλον προγραμματισμού Matlab καθώς και η δυνατότητα για την μεταξύ τους αλληλεπίδραση. Επιλέχτηκε ο κύβος του rubik ως αντικείμενο μελέτης, μιας και από μηχανικής πλευράς αποτελεί πρόκληση η δυνατότητα κατασκευής ενός συστήματος κινητήρων και αισθητήρων, με σχετικά περιορισμένες δυνατότητες όπως αυτές του Lego-NXT, που να μπορεί να τον «δει» και να τον περιστρέψει. Μέσω της Matlab θα επιχειρηθεί η χρήση των κινητήρων και των αισθητήρων αυτών με μεγάλη ακρίβεια αλλά και η δημιουργία ενός βέλτιστου κώδικα για την επίλυση του κύβου με τις λιγότερες δυνατές κινήσεις. Τέλος, μακροπρόθεσμο στόχο αυτού του συστήματος NXT-Matlab, αποτελεί η δυνατότητα μεταγενέστεροι φοιτητές να αντιληφθούν ευκολότερα την επικοινωνία μεταξύ ρομπότ και υπολογιστή αλλά και να λύσουν δυσκολότερα προβλήματα.

Abstract

The aim of this thesis is to familiarize students with the automated systems, consisting of the NXT logical part, the Matlab's programming interface and the ability to react with each other. Rubik's cube was chosen as case of study, since mechanically is challenging to build all over a system of engines and sensors with relatively limited capabilities such as those of Lego-NXT, who can "see" the Rubik's cube and rotate it. Using Matlab's software, this thesis will try to use these motors and sensors with great precision but also create an optimal code to resolve the cube with the least possible moves. Finally, the long-term goal of this NXT-Matlab system is to inspire other students to better understand the communication between robots and computers and be able to solve more complex problems.

Ορολογία αναφοράς

Ορολογία	Επεξήγηση
Κύβος (Cube)	Ο τρισδιάστατος κύβος του Rubik ως σύνολο
Υποκύβος (Cubie)	Οι μικρότεροι κύβοι που συνθέτουν τον κύβο του Rubik
U(Up)	Η πάνω πλευρά του κύβου με κεντρικό υποκύβο χρώματος άσπρου
D(Down)	Η κάτω πλευρά του κύβου με κεντρικό υποκύβο χρώματος κίτρινου
L(Left)	Η αριστερή πλευρά του κύβου με κεντρικό υποκύβο χρώματος πράσινου
R(Right)	Η δεξιά πλευρά του κύβου με κεντρικό υποκύβο χρώματος μπλε
F(Front)	Η μπροστινή πλευρά του κύβου με κεντρικό υποκύβο χρώματος κόκκινου
B(Back)	Η πίσω πλευρά του κύβου με κεντρικό υποκύβο χρώματος πορτοκαλί
Μονή περιστροφή	Η περιστροφή μιας από τις πλευρές του κύβου κατά 90 μοίρες
Διπλή περιστροφή	Η περιστροφή μιας από τις πλευρές του κύβου κατά 180 μοίρες

Πίνακας Περιεχομένων

Λίστα Πινάκων

Λίστα Εικόνων

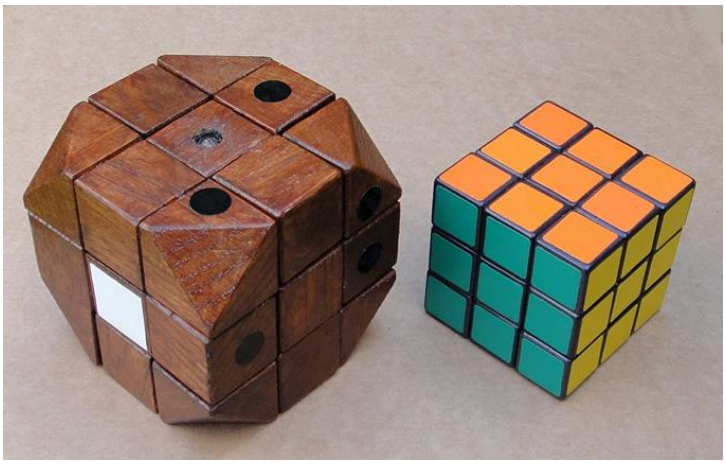
ΚΕΦΑΛΑΙΟ 1

1. Εισαγωγή

Στο κεφάλαιο αυτό αναλύονται τα δομικά στοιχεία της διπλωματικής και το περιβάλλον προγραμματισμού που χρησιμοποιήθηκε όπως ακόμα και τα απαραίτητα βοηθητικά λογισμικά.

1.1 Κύβος του Rubik

Ο κύβος του Ρούμπικ είναι ένα τρισδιάστατο μηχανικό παζλ [1]. Επινοήθηκε το 1974 από τον Ούγγρο γλύπτη και καθηγητή αρχιτεκτονικής Έρνο Ρούμπικ. Ο Ρούμπικ δημιούργησε τον κύβο σε μια προσπάθεια να βοηθήσει τους φοιτητές του να κατανοήσουν τρισδιάστατα προβλήματα. Όταν έφτιαξε τον πρώτο κύβο και ανακάτεψε τα χρώματα, του πήρε πάνω από έναν μήνα για να καταφέρει να τον επαναφέρει στην αρχική του κατάσταση.



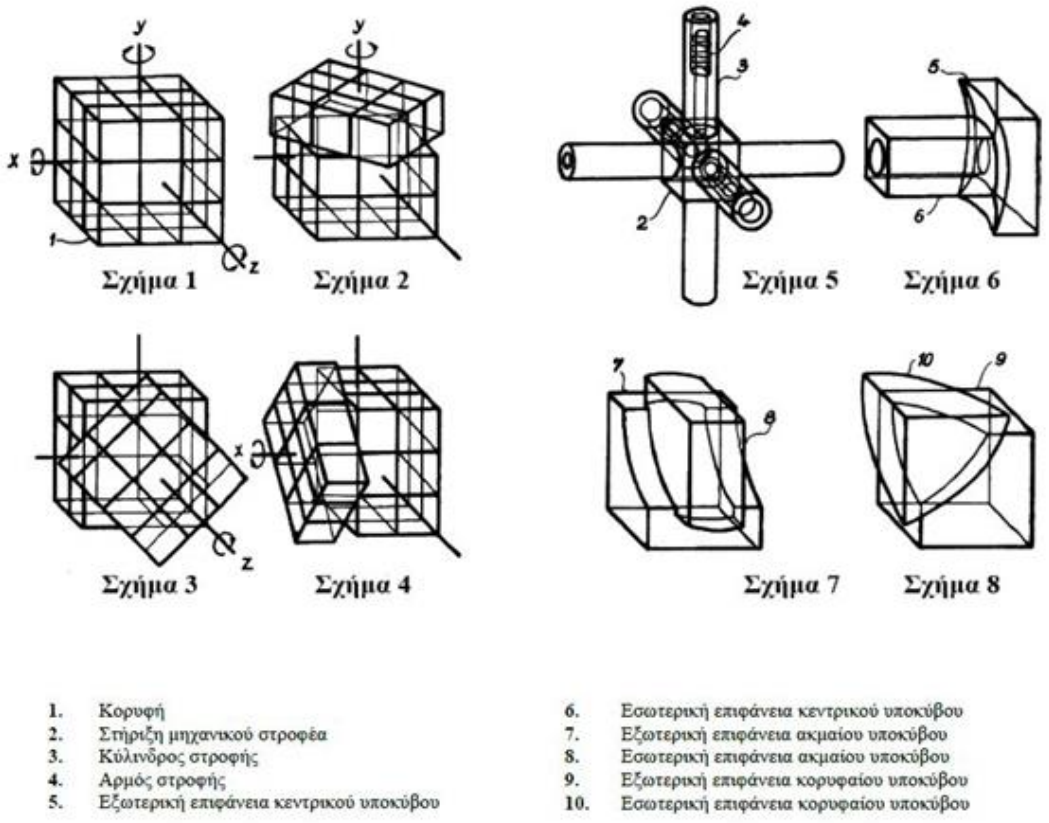
Εικόνα. 1 Κύβος του Ρούμπικ το 1974 (αριστερά) και σήμερα (δεξιά)

Έως τον Ιανουάριο του 2009, 350 εκατομμύρια κύβοι είχαν πουληθεί παγκοσμίως κάνοντας τον το εμπορικότερο παιχνίδι παζλ παγκοσμίως, ενώ θεωρείται ευρέως το εμπορικότερο παιχνίδι στον κόσμο. Σε έναν κλασσικό κύβο του Ρούμπικ, κάθε μία από τις έξι έδρες καλύπτεται από εννιά αυτοκόλλητα με έξι χρώματα (παραδοσιακά λευκό, κόκκινο, κίτρινο, πράσινο, μπλε και πορτοκαλί). Ένας μηχανισμός περιστροφής επιτρέπει σε κάθε έδρα να περιστρέφεται ανεξάρτητα από τις άλλες, με αποτέλεσμα να συγχέονται τα χρώματα. Τα κέντρα των εδρών, οι κεντρικοί υποκύβοι, μπορούν να

θεωρηθούν ακίνητα. Οι κινήσεις των εδρών είναι στροφές περί τα κέντρα U, L, F, R,B, D των εδρών του κύβου, ονόματα που προερχόμενα από τις αντίστοιχες αγγλικές έννοιες. Το ζητούμενο είναι, ξεκινώντας από μία τυχαία δυνατή κατάσταση να καταλήξουμε, μέσω επαλληλίας επιτρεπόμενων στροφών, σε 6 ομόχρωμες έδρες, κατάσταση την οποία ονομάζουμε λυμένη μορφή. [2]

1.2 Προηγούμενες διενέργειες επίλυσης

Οι πρώτες προσπάθειες για να λυθεί ο κύβος του Rubik με ρομπότ ευρείας κυκλοφορίας όπως το NXT ή ο προκάτοχός του RCX ξεκίνησαν το 2007. Μέχρι τότε είναι λίγες οι καταγεγραμμένες προσπάθειες λύσης του κύβου, με εξέχουσα αυτή της Kawasaki με τα δυο ρομπότ FS03. Παρόλα αυτά, όλες αυτές χρησιμοποιούσαν βιομηχανική τεχνολογία που δεν ήταν προσβάσιμη στο κοινό. Η πρώτη επιτυχημένη προσπάθεια σημειώθηκε με το JP Brown's Lego Rubik Solver που χρησιμοποιούσε όμως πολλούς κινητήρες και αισθητήρες, μια κάμερα web, ένα προσαρμοσμένο πρόγραμμα διασύνδεσης στον Η/Υ και έναν ήδη ημιβέλτιστο αλγόριθμο που βρισκόταν στον ιστό εκείνη την εποχή. Επακόλουθα, άρχισε να διαδίδεται η γνώση της ικανότητας των Bricks να λύνουν τον κύβο. Στο πέρασμα του χρόνου οι κατασκευές απλοποιήθηκαν και έγιναν πιο γρήγορες, δημιουργώντας ταυτόχρονα έναν ανταγωνισμό για τον καλύτερο και γρηγορότερο μηχανισμό. Σαν αποτέλεσμα, σε μία χρονική περίοδο το πιο γρήγορο ρομπότ επίλυσης κύβου ήταν κατασκευασμένο από το Lego ev3, τον απόγονο του NXT. Σήμερα το παγκόσμιο ρεκόρ από βιομηχανικά κατασκευασμένο ρομπότ είναι μόλις 0.38sec.



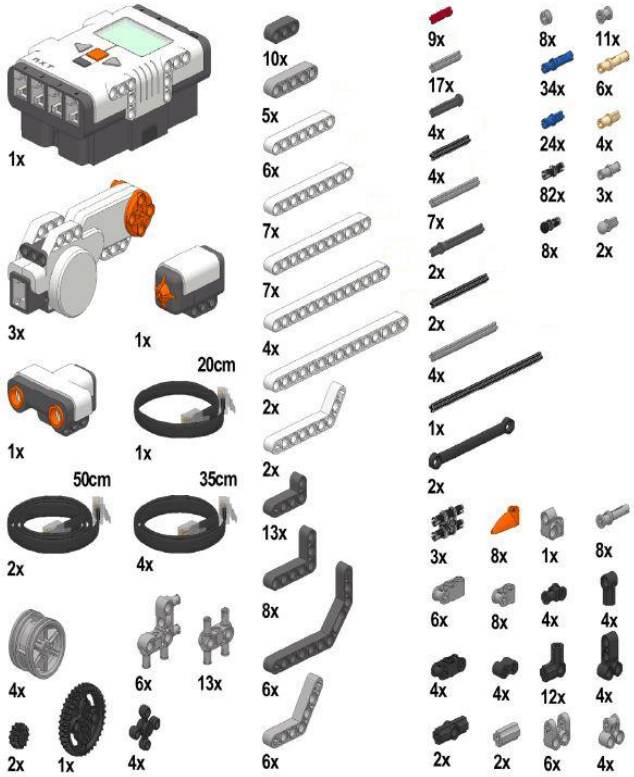
Εικόνα 2 Η συνδεσιμότητα και ο μηχανισμός του κύβου

1.3 Lego Mindstorms NXT

Τα Lego Mindstorms είναι μια γραμμή παραγωγής της Lego που συνδυάζει προγραμματιζόμενα τούβλα με ηλεκτρικές μηχανές, αισθητήρες, τούβλα Lego, και τεχνικά κομμάτια Lego (όπως εργαλεία, άξονες, ακτίνες, και υδραυλικά μέρη) κατάλληλα για να χτίσει ο χρήστης ρομπότ και άλλα αυτοματοποιημένα ή αλληλεπιδραστικά συστήματα. [3]

Εικόνα 3 Το Lego Mindstroms

Η πρώτη λιανική έκδοση των Lego Mindstorms κυκλοφόρησε το 1998 και πωλήθηκε εμπορικά με την επωνυμία Robotics Invention System (RIS). Η τρέχουσα έκδοση κυκλοφόρησε το 2006 ως Lego Mindstorms NXT.



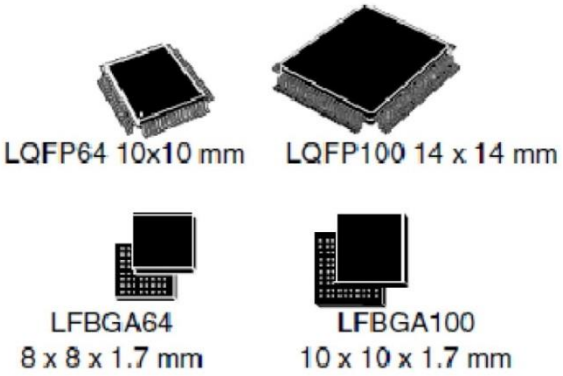
Εικόνα. 4 Τα εξαρτήματα του Lego Mindstorms

Η αρχική έκδοση Mindstorms Robotics Invention System περιείχε δύο μηχανές, δύο αισθητήρες αφής και έναν αισθητήρα φωτός. Η έκδοση NXT έχει τρεις σερβοκινητήρες και τέσσερις αισθητήρες, δύο για την αφή, το φως και την απόσταση. Τα Lego Mindstorms μπορούν να χρησιμοποιηθούν για να κατασκευαστεί ένα μοντέλο ενσωματωμένου συστήματος με ηλεκτρομηχανικά μέρη ελεγχόμενα από υπολογιστή. Πολλά είδη πραγματικών ενσωματωμένων συστημάτων, από ελεγκτές ανελκυστήρων έως βιομηχανικά ρομπότ, μπορούν να διαμορφωθούν χρησιμοποιώντας τα Mindstorms. Τα Mindstorms kits πωλούνται επίσης και χρησιμοποιούνται ως εκπαιδευτικά εργαλεία, έχοντας ως αφετηρία μια συνεργασία μεταξύ της Lego και του Εργαστηρίου Πολυμέσων του MIT (MIT Media Laboratory). Η εκπαιδευτική έκδοση των προϊόντων καλείται Lego Mindstorms for Schools, και έρχεται με το γραφικό λογισμικό προγραμματισμού ROBOLAB, που αναπτύχθηκε στο Πανεπιστήμιο Tufts χρησιμοποιώντας ως μηχανή το LabVIEW της National Instruments.

1.3.1 Λειτουργικά χαρακτηριστικά

Ο εγκέφαλος του NXT Intelligent Brick αποτελείται από: [4]

- 32-bit Atmel AT91SAM7S256 main microcontroller (256 KB flash memory, 64 KB RAM)
- 8-bit Atmel ATmega48 microcontroller @ 4 MHz (4 KB flash memory, 512 Bytes RAM)



Εικόνα. 5 Ο εγκέφαλος του NXT Intelligent Brick

Το κουτί που συνδέει το NXT με το περιβάλλον έχει προσαρμοσμένα πάνω του:

- Μια LCD οθόνη με ανάλυση 100×64 pixel
- Τέσσερις 6-pin θύρες εισόδου (θύρες 1-4)
- Τρεις 6-pin θύρες εξόδου (θύρες A-C)
- USB θύρα
- Ενσωματωμένο Bluetooth Class II V2.0
- Ηχεία – 8 kHz sound quality, 8-bit resolution, 2–16 kHz sample rate
- Τέσσερα κουμπιά :
  - Πορτοκαλί κουμπί: On/Enter
  - Πράσινα κουμπιά: κίνηση δεξιά και αριστερά στο μενού
  - Γκρι κουμπί: Clear/Go back
- Τροφοδοτείται από έξι AA batteries ή από την επαναφορτιζόμενη μπαταρία του.

Εικόνα. 6 Το κουτί του NXT

1.3.2 Βασικοί αισθητήρες

Το αρχικό πακέτο περιέχει τέσσερις αισθητήρες. Έναν αισθητήρα χρώματος, δυο αισθητήρες αφής και έναν υπερηχητικό αισθητήρα. Οι υπόλοιποι αισθητήρες μπορούν να αγοραστούν ξεχωριστά.

1. Αισθητήρα χρώματος (9694), για την ανίχνευση 6 διαφορετικών χρωμάτων: μπλε, πράσινο, κόκκινο, κίτρινο, λευκό, μαύρο

Εικόνα. 7 Ο αισθητήρας χρώματος

2. Αισθητήρα φωτός (9844), για την ανίχνευση επιπέδων φωτός. (Συμπεριλαμβάνεται στην πρώτη έκδοση, αλλά στην v2.0, έχει αντικατασταθεί από τον αισθητήρα χρώματος.)

3. Αισθητήρα αφής (9843), ένα απλό κουμπί που ανιχνεύει αν κάτι πιέζεται σε αυτόν.

Εικόνα. 8 Ο αισθητήρας αφής

4. Υπερηχητικό αισθητήρα (9846), για τη μέτρηση αποστάσεων χρησιμοποιώντας ηχητικά κύματα.

Εικόνα. 9 Ο υπερηχητικός αισθητήρας

5. Αισθητήρα ήχου (9845), για βασική "ακρόαση". Έχει την δυνατότητα μέτρησης της έντασης, αλλά δεν μπορεί να καταγράψει τους πραγματικούς ήχους.

Εικόνα. 10 Ο αισθητήρας ήχου

6. Αισθητήρα πυξίδας (MS1034), για ανίχνευση της κατεύθυνσης. Διαθέτει ενσωματωμένο βαθμονομητή για τη μείωση των παρεμβολών από άλλα μαγνητικά στοιχεία.

Εικόνα. 11 Ο αισθητήρας πυξίδας

7. Αισθητήρα επιταχυνσιομέτρου (MS1040), για την ανίχνευση της γενικής κατεύθυνσης στην οποία κινείται. Μπορεί επίσης να μετρήσει τη force-g.

8. Αισθητήρα RFID, για επικοινωνία μεταξύ πολλαπλών ρομπότ.

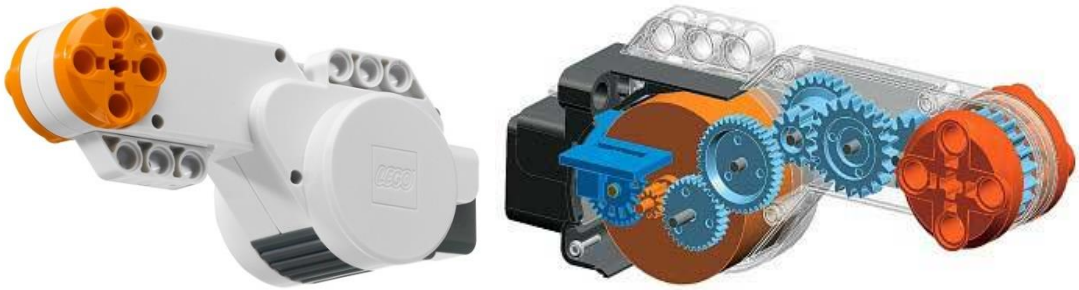
Εικόνα. 12 Ο αισθητήρας RFID

9. Αισθητήρα περιστροφής (ενσωματωμένο στους σερβοκινητήρες), για τη μέτρηση του βαθμού στον οποίο έχει γυρίσει. Η ιδιαιτερότητά του είναι ότι μετρά με βάση τη στροφή των εσωτερικών γραναζιών, αντί για τον ίδιο τον κινητήρα.

10. Επικοινωνία Bluetooth (ενσωματωμένη στο "Intelligent brick"), για επικοινωνία με άλλες συσκευές. Μπορεί να χρησιμοποιηθεί στο μέσο του προγράμματος και για λήψη νέων προγραμμάτων και δεδομένων.

1.3.3 Σερβοκινητήρες

Το πακέτο περιέχει ακόμα τρεις σερβοκινητήρες και παρέχει τη δυνατότητα συγχρονισμένης χρήσης. Τα χαρακτηριστικά των κινητήρων δεν προσδιορίζονται από την εταιρία.



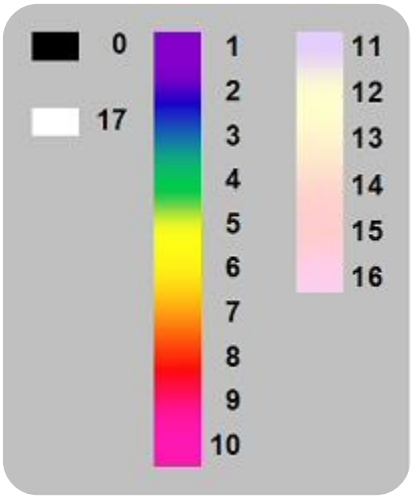
Εικόνα. 13 Οι σερβοκινητήρες του NXT

1.3.4 Αισθητήρες άλλων εταιριών

Για την εκπόνηση της διπλωματικής εργασίας πέραν του βασικού πακέτου του Lego NXT, χρησιμοποιήθηκε και ένας αισθητήρας χρώματος διαφορετικής εταιρίας, ο HiTechnic NXT Color Sensor Version 2. [5]

Εικόνα. 14 Ο αισθητήρας HiTechnic NXT Color Sensor Version 2

Ο συγκεκριμένος αισθητήρας δίνει την δυνατότητα εύρεσης περισσότερων χρωμάτων, επιστρέφει πιο ακριβή αποτελέσματα και έχει μεγαλύτερη εμβέλεια λήψης.



Εικόνα. 15 Το χρωματικό εύρος του αισθητήρα

1.4 Matlab

1.4.1Γενικά

Το MATLAB (matrix laboratory) είναι ένα περιβάλλον αριθμητικής υπολογιστικής και μια προγραμματιστική γλώσσα τέταρτης γενιάς. Αποθηκεύει και κάνει τις πράξεις με βάση την άλγεβρα μητρών. [6]

Εικόνα. 16 Το λογότυπο του λογισμικού Matlab

Επιτρέπει χειρισμούς σε πίνακες, σχεδίαση λειτουργιών και δεδομένων, υλοποίηση αλγορίθμων, δημιουργία διεπαφών χρήστη και διασύνδεση με προγράμματα γραμμένα σε άλλες γλώσσες, όπως C, C ++, C #, Java, Fortran και Python. Παρόλο που το MATLAB προορίζεται κυρίως για αριθμητική υπολογιστική, μια προαιρετική εργαλειοθήκη χρησιμοποιεί το συμβολικό μηχανισμό MuPAD, επιτρέποντας την πρόσβαση σε συμβολικές υπολογιστικές ικανότητες. Ένα πρόσθετο πακέτο, η Simulink, προσθέτει γραφική προσομοίωση πολλαπλών τομέων και σχεδιασμό με βάση το μοντέλο για δυναμικά και ενσωματωμένα συστήματα. Από το 2018, το MATLAB έχει περισσότερους από 3 εκατομμύρια χρήστες παγκοσμίως . Οι χρήστες του MATLAB προέρχονται από διάφορα περιβάλλοντα μηχανικής, επιστημών και οικονομίας.

1.4.2 Ιστορία

Ο Cleve Moler, πρόεδρος του τμήματος πληροφορικής του Πανεπιστημίου του Νέου Μεξικού, άρχισε να αναπτύσσει το MATLAB στα τέλη της δεκαετίας του 1970. Το σχεδίασε για να δώσει στους μαθητές του πρόσβαση σε LINPACK και EISPACK χωρίς να χρειάζεται να μάθουν Fortran. Σύντομα εξαπλώθηκε σε άλλα πανεπιστήμια και βρήκε ένα ισχυρό κοινό στο πλαίσιο της εφαρμοσμένης μαθηματικής κοινότητας. Ο Jack Little, ένας μηχανικός, ‘’εκτέθηκε’’ σε αυτό κατά τη διάρκεια μιας επίσκεψης του Moler που έγινε στο Πανεπιστήμιο του Στάνφορντ το 1983. Αναγνωρίζοντας το εμπορικό του δυναμικό, ενώθηκε με τον Moler και τον Steve Bangert. Επανέγραψαν το MATLAB στη C και ίδρυσαν την MathWorks το 1984 για να συνεχίσουν την ανάπτυξή του. Το MATLAB υιοθετήθηκε πρώτη φορά από ερευνητές και

επαγγελματίες στον τομέα της μηχανικής ελέγχου, ειδικότητα του Little, αλλά γρήγορα εξαπλώθηκε σε πολλούς άλλους τομείς. Σήμερα χρησιμοποιείται επίσης στην εκπαίδευση, ιδιαίτερα στη διδασκαλία της γραμμικής άλγεβρας, της αριθμητικής ανάλυσης, και είναι δημοφιλής στους επιστήμονες που ασχολούνται με την επεξεργασία εικόνων. [7]

1.5 Εγκατάσταση λογισμικού

Για την λειτουργία του κώδικα αλλά και για οποιαδήποτε χρήση του Lego NXT είναι απαραίτητα τα τρία παρακάτω λογισμικά.

1.5.1 Matlab

Όλες οι εκδόσεις του Matlab έχουν τη δυνατότητα επικοινωνίας και χρήσης του NXT. Για την παρούσα διπλωματική χρησιμοποιήθηκε η 2015a. Για την απόκτησή του υπάρχουν τρεις διαφορετικοί τρόποι. Ο πρώτος είναι να χρησιμοποιηθεί με την δωρεάν έκδοση για 30 μέρες, μια επιλογή που εξασφαλίζει την δοκιμαστική χρήση του προϊόντος. Ο δεύτερος είναι η κανονική αγορά του προϊόντος και τέλος η δωρεάν χρήση για φοιτητές που ποικίλει ανάλογα με τα ιδρύματα και τις συμφωνίες που έχουν συνάψει.

1.5.2 RWTH Aachen MINDSTORMS NXT Toolbox

Η βιβλιοθήκη που παρέχει το πανεπιστήμιο του Aachen δίνει την δυνατότατα χρήσης όλων των αισθητήρων του βασικού πακέτου Lego NXT αλλά και αισθητήρων από άλλες εταιρίες όπως η Hitechnic. Με την εγκατάσταση του, ο χρήστης μπορεί να χρησιμοποιήσει όλες τις λειτουργιές του NXT (και των αισθητήρων του) μέσα από την Matlab χωρίς να χρειάζονται περαιτέρω προγράμματα. Οι εντολές του θα αναλυθούν σε επόμενο κεφάλαιο. Προτεινόμενη έκδοση v4.07. [8]

1.5.3 LEGO MINDSTORMS NXT Driver

Τέλος, χρειάζεται ο LEGO MINDSTORMS NXT Driver για να μπορέσει ο υπολογιστής να επικοινωνήσει αποτελεσματικά με το NXT μέσω του καλωδίου USB. Δυστυχώς, το NXT δεν υποστηρίζεται πλέον από την Lego, οπότε δεν υπάρχει driver από την ίδια την εταιρία. Προτεινόμενος driver NXT Fantom Driver. [9]

Εικόνα. 17 Ο NXT Fantom Driver

1.6 Συνδεσμολογία

Για την επίλυση του κύβου χρησιμοποιήθηκε ένα brick NXT, τρεις σερβοκινητήρες και ένας αισθητήρας χρώματος. Ο αισθητήρας χρώματος τοποθετήθηκε στην τρίτη input θέση του NXT, ενώ οι σερβοκινητήρες κατέλαβαν και τις τρεις διαθέσιμες θέσεις. Στην θέση A τοποθετήθηκε ο κινητήρας για την περιστροφή του κύβου, στην θέση B ο κινητήρας για την κίνηση του αισθητήρα χρώματος, ενώ στην τελευταία, τη θέση C, ο κινητήρας της δαγκάνας. Όλες οι παραπάνω συνδέσεις έγιναν με καλώδια που παρέχονται στο βασικό πακέτο του Lego mindstorm και έχουν βύσματα RJ12. Το σύστημα brick, αισθητήρων και κινητήρων συνδέεται σε ηλεκτρονικό υπολογιστή με καλώδιο usb type A to type B, που επίσης βρίσκεται στο βασικό σετ. Για να επικοινωνήσει ο Η/Υ με το NXT πέρα από την φυσική σύνδεση, χρησιμοποιεί ένα πρόγραμμα NXC που ονομάζεται MotorControl και περιλαμβάνεται στην εργαλειοθήκη RWTH - Mindstorms NXT. Αυτό το πρόγραμμα γίνεται compiled (χρησιμοποιώντας τον μεταγλωττιστή NXC / NBC) σε ένα δυαδικό αρχείο RXE που μεταφέρεται στο brick NXT. Κατά την εκτέλεση των προγραμμάτων MATLAB, το MotorControl συνεχίζει να τρέχει στο NXT και λαμβάνει εντολές. Το πρόγραμμα NXC στη συνέχεια φροντίζει για τους κινητήρες και τους ελέγχει με ακρίβεια, ενώ το πρόγραμμα MATLAB μπορεί να συνεχίσει να εκτελείται. Η τελευταία έκδοση είναι το MotorControl 2.2 (περιλαμβάνεται στην έκδοση 4.04 του RWTH - Mindstorms NXT).

ΚΕΦΑΛΑΙΟ 2

2. Μηχανική κατασκευή

Στο κεφάλαιο αυτό γίνεται παρουσίαση της μηχανικής κατασκευής για την επίλυση του κύβου.

2.1 Εισαγωγή

Για να λυθεί ο κύβος το πρώτο πρόβλημα που αντιμετωπίστηκε ήταν να δημιουργηθεί η μηχανική κατασκευή. Το ίδιο το πρόβλημα διαχωρίζεται σε δύο υποπροβλήματα. Το πρώτο είναι η σάρωση του κύβου, διαδικασία κατά την οποία ο αισθητήρας χρώματος περνάει πάνω από κάθε υποτετράγωνο του κύβου, έτσι ώστε να μπορέσει να αποθηκεύσει την συνολική εικόνα του κύβου. Το δεύτερο είναι η διαδικασία επίλυσης του κύβου. Πρέπει λοιπόν να δημιουργηθεί ένα σύστημα κινητήρων με το όποιο μπορεί ο κύβος να περιστραφεί σε όλους τους άξονες και ο αισθητήρας χρώματος να έχει την δυνατότητα κίνησης έτσι ώστε να μην εμποδίζει οποιαδήποτε κίνηση του κύβου. [10]

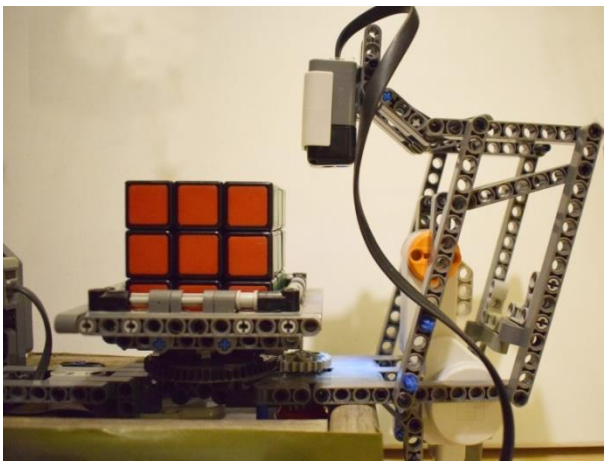
2.2 Σερβοκινητήρες και η κίνησή τους στον χώρο.

Οι σερβοκινητήρες που έχουμε στην διάθεσή μας έχουν την δυνατότητα περιστροφικής κίνησης, οπότε για οποιαδήποτε κίνηση είναι απαραίτητη η δημιουργία βραχιόνων, τέτοιων ώστε η περιστροφική κίνηση να μετατρέπει σε γραμμική ή οποιαδήποτε άλλη κίνηση. Ένας ακόμα περιορισμός που υπάρχει είναι η κατοχή μόνο τριών κινητήρων, οπότε για την κίνηση του κύβου σε όλους τους άξονες πρέπει να χρησιμοποιηθεί ένας συνδυασμός τους και όχι ένας ξεχωριστός αισθητήρας για κάθε κίνηση.

2.3 Κίνηση αισθητήρα χρώματος

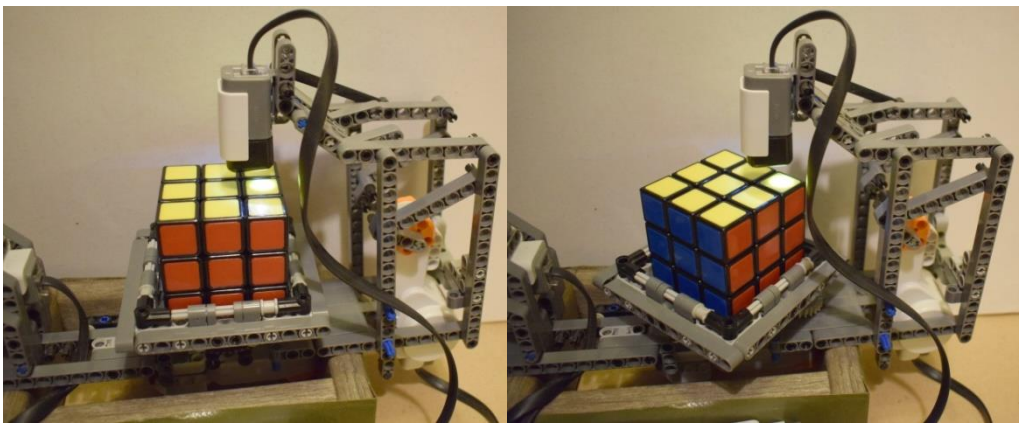
Ο μηχανικός βραχίονας για την κίνηση του αισθητήρα χρώματος είναι τοποθετημένος στο δεξί μέρος του ρομπότ. Πρέπει να έχει την δυνατότητα να μετακινείται προς το

κέντρο, εκεί που είναι τοποθετημένος ο κύβος και στην συνέχεια να επιστρέφει μακριά από αυτόν έτσι ώστε να δίνει την δυνατότητα στην δαγκάνα να κινείται χωρίς να την εμποδίζει. Απαραίτητο είναι ακόμα, να έχει τη δυνατότητα να κάνει μικρές κινήσεις έτσι ώστε να μπορεί να βρίσκεται πάνω από τα πλάγια κομμάτια του κύβου αλλά και τα γωνιακά, μιας και αυτά τα δυο βρίσκονται σε διαφορετικές θέσεις, καθώς ο κύβος περιστρέφεται.



Εικόνα. 18 Η πειραματική διάταξη με τον αισθητήρα χρώματος

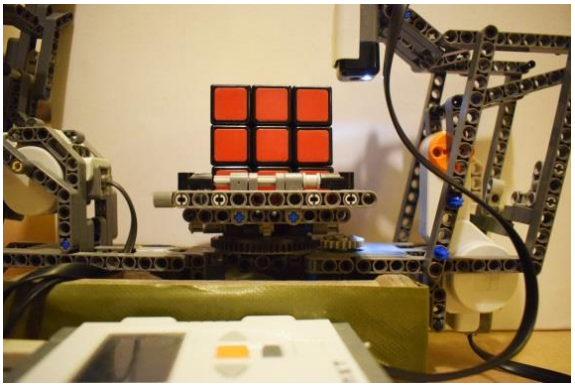
Συνεπώς, για τον αισθητήρα χρώματος η κίνηση που είναι απαραίτητη είναι να μπορεί να κινείται μπρος και πίσω δηλαδή με φορά προς τον κύβο και ανάποδα.



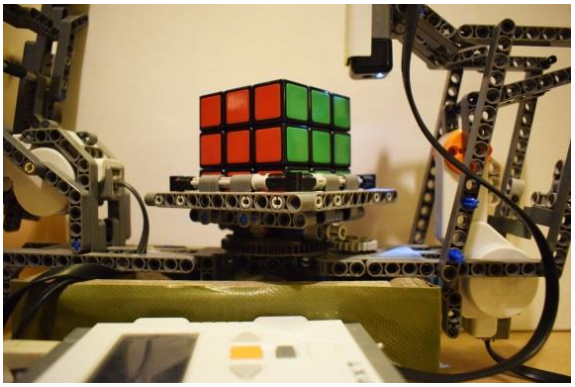
Εικόνα. 19 Ανάγνωση χρώματος από τον αισθητήρα

2.4 Περιστροφή του κύβου

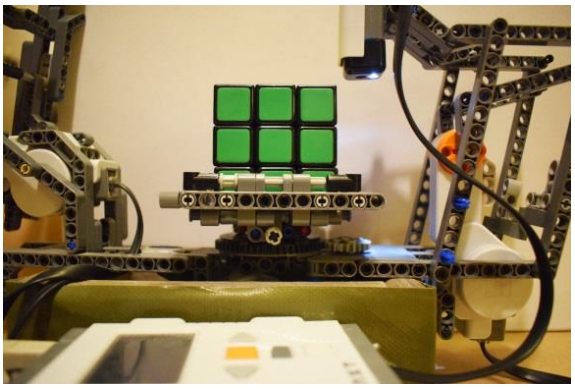
Η κίνηση αυτή του κύβου είναι η πιο απλή από τις τρεις. Το μόνο που χρειάζεται είναι η σύνδεση του κύβου με έναν από τους κινητήρες, έτσι ώστε αυτός να μπορεί να περιστρέφεται. Αυτό πραγματοποιείται με ένα σύστημα γραναζιών και μια βάση στην οποία πάνω τοποθετείται ο κύβος. Αυτός μας έδωσε την δυνατότητα να περιστρέφουμε τον κύβο δεξιόστροφα ή αριστερόστροφα.



Εικόνα. 20 Περιστροφική κίνηση του κύβου 1^{ο} στάδιο



Εικόνα. 21 Περιστροφική κίνηση του κύβου 2^{ο} στάδιο



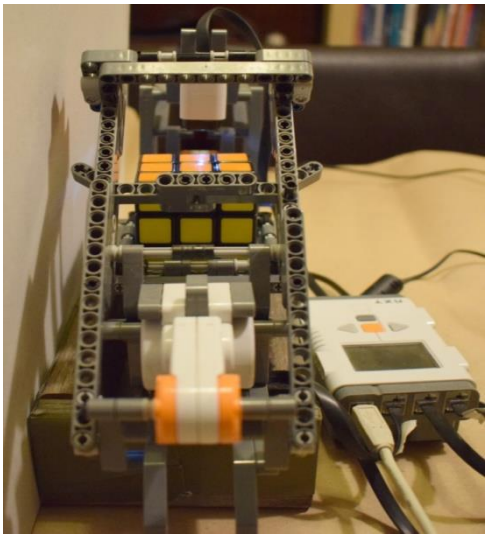
Εικόνα. 22 Περιστροφική κίνηση του κύβου 3^{ο} στάδιο

2.5 Βάση του κύβου

Η βάση του κύβου δημιουργήθηκε από ένα τετράγωνο που η χρησιμότητα του είναι να στηριχτεί ο κύβος και ένα πλαίσιο πάνω και γύρω από αυτό. Η χρήση του τελευταίου έχει ως σκοπό όταν η δαγκάνα πιάνει τον κύβο και προσπαθεί να τον περιστρέψει στον εγκάρσιο άξονα, να μπορεί να βρίσκει αντίσταση ώστε η περιστροφή του να είναι δυνατή.

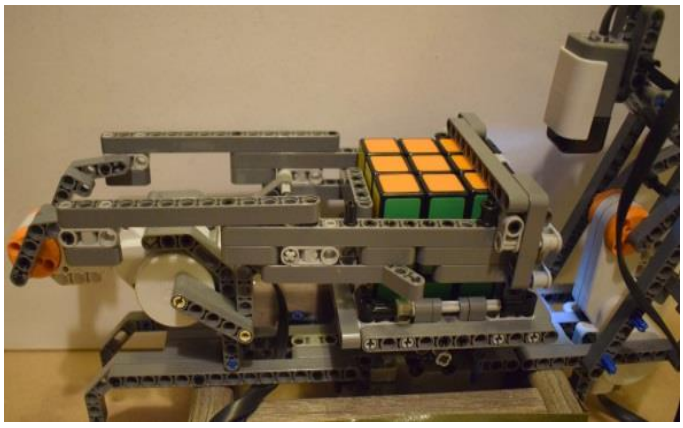
2.6 Δαγκάνα περιστροφής

Το τελευταίο και πιο απαιτητικό μέρος ήταν η δυνατότητα περιστροφής του κύβου στο εγκάρσιο άξονα, αλλά και η ικανότητα σταθεροποίησής του, έτσι ώστε να μπορούν με την περιστροφή της βάσης να αλλάζουν θέσεις οι πλευρές του. Αυτό επιτυγχάνεται με την δημιουργία μιας δαγκάνας που μπορεί να περιστρέφεται πάνω από τον κύβο.

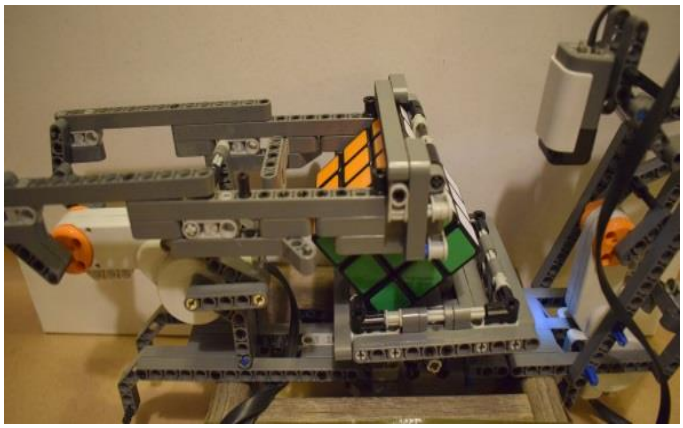


Εικόνα. 23 Η δαγκάνα περιστροφής

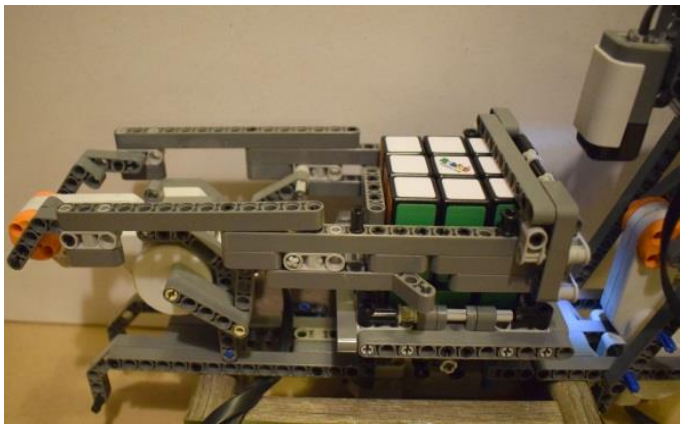
Για την εγκάρσια μετατόπιση του κύβου η δαγκάνα τοποθετείται στο πάνω μέρος του, τον σέρνει προς το μέρος της και στην συνέχεια τον απωθεί, οδηγώντας τον σε μια περιστροφική κίνηση, αλλάζοντάς του τις πλευρές. Ακόμα, έχει την δυνατότητα να εφάπτεται στο πάνω μέρος του και με το πλαίσιο της να τον εμποδίζει να κινηθεί περιστροφικά, έτσι ώστε να μπορεί να αλλάξει θέση μόνο η κάτω του πλευρά (η πλευρά που χρειάζεται να μετακινηθεί ώστε να λυθεί ο κύβος).



Εικόνα. 24 Εγκάρσια μετατόπιση του κύβου 1^{ο} στάδιο



Εικόνα. 25 Εγκάρσια μετατόπιση του κύβου 2^{ο} στάδιο



Εικόνα. 26 Εγκάρσια μετατόπιση του κύβου 3^{ο} στάδιο

ΚΕΦΑΛΑΙΟ 3

3. Προγραμματισμός στο Matlab

Στο κεφάλαιο αυτό αναλύεται ο κώδικας που δημιουργήθηκε για την επίλυση του κύβου στο περιβάλλον του Matlab.

3.1 Εισαγωγή

Κατά την αρχική προσέγγιση του προγραμματιστικού προβλήματος εντοπίζονται τρία μεγάλα υποπροβλήματα που πρέπει να επιλυθούν. Έχοντας ήδη κατασκευασμένο το μηχανικό κομμάτι, πρέπει καταρχάς να περάσει ο αισθητήρας χρώματος πάνω από όλα τα υποτετράγωνα του κύβου, σε όλες τις πλευρές , και να αποθηκεύσει τα δεδομένα σε πίνακες. Το δεύτερο μεγάλο υποπρόβλημα είναι αυτό της ίδιας της λύσης του κύβου, δηλαδή η ορθή ανάγνωση των προηγούμενων πινάκων και η εύρεση της σωστής αλληλουχίας κινήσεων έτσι ώστε ο κύβος να μπορεί να λυθεί. Τέλος, πρέπει από τις κινήσεις που έχουν βρεθεί στο δεύτερο βήμα να σχεδιαστεί και να εκτελεστεί μια σειρά κινήσεων από τους κινητήρες, έτσι ώστε και ο πραγματικός κύβος να φτάσει στην λυμένη του κατάσταση.

3.2 Επικοινωνία με το NXT

Το πρώτο που πρέπει να γίνει είναι να εκτελεστούν οι εντολές που θα ανοίξουν τους διαύλους επικοινωνίας μεταξύ του Matlab και του NXT. Αυτό συμβαίνει με τις εντολές:

```
COM_SetDefaultNXT()
COM_GetDefaultNXT
```

Αφού έχει εγκαθιδρυθεί η σύνδεση του Matlab και του NXT μπορεί να προχωρήσει ο χειρισμός των αισθητήρων και των κινητήρων.

3.3 Κινήσεις κινητήρων και αισθητήρα

Για την λειτουργία των κινητήρων γράφτηκαν κάποιες εντολές οι οποίες χρησιμοποιούνται επανειλημμένα και θα αναλυθούν εδώ.

```
%% orismos parametrwn kinitira peristrofis kubou
Power = 30;
port  = MOTOR_B;
dist  = 69;    % apostasi peristrofis
%% kataskeusi peristrofikis kinisis kubou
kinisiproskubo      =      NXTMotor(port1,      'Power',-power1,
'ActionAtTachoLimit', 'HoldBrake');

%% proetimasia peirstofis
kinisiproskubo.Stop('off');
kinisiproskubo.ResetPosition();

    % apostasi peristrofis
     kinisiproskubo.TachoLimit = dist1 ;
    % kinisi
    kinisiproskubo.SendToNXT();
    %stamatima
    kinisiproskubo.WaitFor()
```

Ο παραπάνω κώδικας είναι ένα παράδειγμα, της κίνησης του αισθητήρα προς τον κύβο. Στο πρώτο κομμάτι αρχικοποιούνται οι παράμετροι που θα χρησιμοποιηθούν με το power να είναι το ποσοστό επί της εκατό δύναμης του κινητήρα, δηλαδή στην συγκεκριμένη περίπτωση το 30% της δύναμης του κινητήρα. Στο port φαίνεται σε ποιόν κινητήρας θέλουν να απευθυνθούν οι εντολές, σε αυτήν την περίπτωση τον κινητήρα που βρίσκεται στην θέση B του NXT. Στις θέσεις του port μπορεί να υπάρχει είτε motor A, είτε motor B, είτε motor C. Τέλος, στο dist αποθηκεύεται η απόσταση που θέλουμε να περιστραφεί ο κινητήρας. Στην συνέχεια, δημιουργείται η εντολή που θα δέχεται το NXT (δεν είναι η εντολή που θα καλείται η συνάρτηση παρά μόνο το όνομα της εντολής που θα στέλνεται στο NXT). Τέλος, όπως φαίνεται και στον κώδικα δίνονται οι εντολές για προετοιμασία, για την απόσταση που πρέπει να διανυθεί, για την εκκίνηση και την παύση του κινητήρα.

3.4 Κύριο κομμάτι κώδικα για την αποτύπωση του κύβου

Το πρώτο που πρέπει να γίνει είναι να δημιουργηθούν οι συνθήκες, έτσι ώστε να περιστραφεί ο κύβος κυκλικά. Αφού είναι έτοιμη η περιστροφή του κύβου αρχίζει η αποτύπωσή του. Πρέπει ο αισθητήρας να περάσει πάνω από τα εννέα τετράγωνα και στις έξι πλευρές. Λόγω του ότι οι πραγματικές θέσεις που βρίσκονται τα γωνιακά κομμάτια είναι διαφορετικές από τις θέσεις των πλευρών, προκύπτει η ανάγκη της ελάχιστης κίνησης του αισθητήρα έτσι ώστε να βρίσκεται ακριβώς πάνω από το εκάστοτε κομμάτι. Στο σημείο του κώδικα που ενεργοποιείται ο αισθητήρας χρώματος, έτσι ώστε να μπορέσει να καταγράψει το ακριβές χρώμα που βρίσκεται κάτω από αυτόν, αποθηκεύει επίσης και το ποσοστό των βασικών χρωμάτων που το αποτελούν (κόκκινο, μπλε και πράσινο), έτσι ώστε να μπορούν να ελεγχτούν τυχόν αστοχίες. Για να βελτιστοποιηθεί ο αλγόριθμος εύρεσης χρωμάτων, αφού έχει αποθηκευτεί μια πλήρη αποτύπωση του κύβου, ελέγχεται και αν χρειαστεί αλλάζει το χρώμα των κεντρικών υποκύβων κάθε πλευράς και τοποθετείται το σωστό τους χρώμα. Αυτό είναι δυνατόν να συμβεί διότι σε κάθε πλευρά αντιστοιχεί ένα συγκεκριμένο χρώμα π.χ. στην πλευρά F αντιστοιχεί το κόκκινο κ.ο.κ. Τέλος, μετατρέπονται παρεμφερή χρώματα στο αντίστοιχο βασικό που χρησιμοποιείται, π.χ. μπορντό σε κόκκινο ή γαλάζιο σε μπλε.

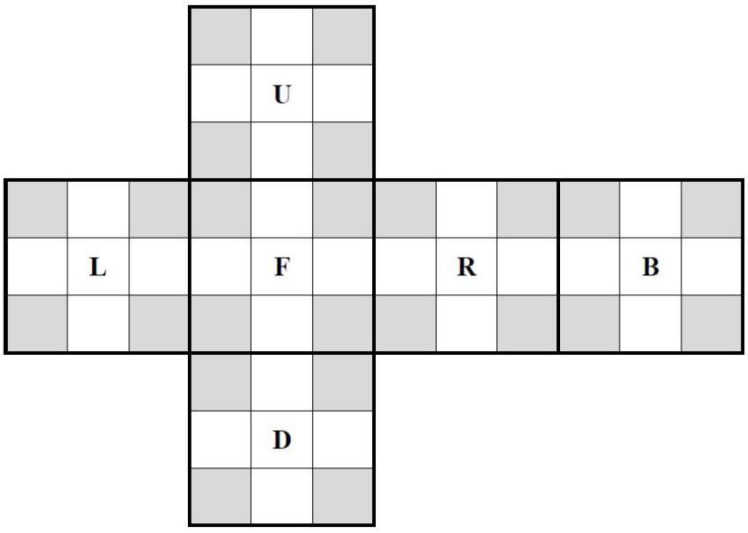
```
%% orismos parametrwn kinitira peristrofis kubou
power = 100;
port  = MOTOR_A;
dist  = 105;    % apostasi peristrofis

%% kataskeusi peristrofikis kinisis kubou
peristrofi      =      NXTMotor(port,      'Power',      -power,
'ActionAtTachoLimit', 'HoldBrake');

%% proetimasia peirstofis
peristrofi.Stop('off');
peristrofi.ResetPosition();


%peristrofi kubou egarsia
for i=1:6
%%kuria kinisi gia tis 8 pleures

%% aisthitiras paei pros ton kubo
kinisiaisthitira()
for j=1:8

if mod(j,2)==0
kinisiaisthitiragiagonies();
```

```
%% times apo aisthitira xromatos
 OpenColor(SENSOR_3)
 [index r g b] = GetColor(SENSOR_3, 0);
 CloseSensor(SENSOR_3)
 %%pinakes gia xromata
 telikoxrwma(i,j) = index;
 telikokkino(i,j) = r;
 telikoble(i,j) = b;
 telikoprasino(i,j) = g;
  finalcolor(i,j) =index;
    kinisiaisthitiragiagoniesanapoda();
    % apostasi peristrofis
     peristrofi.TachoLimit = dist ;
    % kinisi
    peristrofi.SendToNXT();
    %stamatima
    peristrofi.WaitFor();

else
%% times apo aisthitira xromatos
 OpenColor(SENSOR_3)
 [index r g b] = GetColor(SENSOR_3, 0);
 CloseSensor(SENSOR_3)
 %%pinakes gia xromata
 telikoxrwma(i,j) = index;
 telikokkino(i,j) = r;
 telikoble(i,j) = b;
 telikoprasino(i,j) = g;
  finalcolor(i,j) =index;

    % apostasi peristrofis
     peristrofi.TachoLimit = dist ;
    % kinisi
    peristrofi.SendToNXT();
    %stamatima
    peristrofi.WaitFor();
end
 end
%%sarosei kedriko komatiou
kinisiproskedro()
OpenColor(SENSOR_3)
 [index r g b] = GetColor(SENSOR_3, 0);
 CloseSensor(SENSOR_3)

%Apothikeysi xromatwn kedrikou tetrawnoyu apothikeuete stin
teleutaia thesi
telikoxrwma(i,j+1) = index;
 telikokkino(i,j+1) = r;
 telikoble(i,j+1) = b;
 telikoprasino(i,j+1) = g;
 finalcolor(i,j+1) =index;
 %%epistrofi tou aisthtira stin arxiki thesi
kinisiproskedroanapoda()

 %peristrofi kubou ston katheto aksona gia oles tis pleures
 if i<=3
     kinisidaganas()
 elseif i==4
     peristrofi90moires()
     kinisidaganas()
 elseif i==5
     kinisidaganas()
     kinisidaganas()
```

```
 end
end
%%kanoume ola ta xromata idia gia na apalipsoume diafores opws
mpordo me
%%kokkino
for i=1:6
    for j=1:9
        if finalcolor(i,j)==9
            finalcolor(i,j)=8;
        end
         if finalcolor(i,j)==17
            finalcolor(i,j)=12;
         end
         if finalcolor(i,j)==13
            finalcolor(i,j)=4;
         end
          if finalcolor(i,j)==3
            finalcolor(i,j)=2;
         end
        if finalcolor(i,j)==14
            finalcolor(i,j)=4;
        end
         if finalcolor(i,j)==11
            finalcolor(i,j)=2;
         end
         if finalcolor(i,j)==10
            finalcolor(i,j)=8;
         end
        if finalcolor(i,j)==5
            finalcolor(i,j)=6;
        end
    end
end
```

3.5 Ετοιμασία για λύση του κύβου

Αφού τα χρώματα από όλα τα κομμάτια του κύβου έχουν αποθηκευτεί, πρέπει να ετοιμαστούν έτσι ώστε να μπορεί ο αλγόριθμος επίλυσης του κύβου να τα ¨αντιληφθεί σωστά¨. Αυτό συμβαίνει διότι τα αποτελέσματα καταγράφονται με τη σειρά κατά την οποία πέρασε ο αισθητήρας πάνω από τις υποπλευρές και δεν αντιστοιχούν στην ανάπτυξη του κύβου. Θα πρέπει, λοιπόν, ο αρχικός πίνακας με διαστάσεις 6x9 να μετατραπεί σε έξι πίνακες με διαστάσεις 3x3. Στη συνέχεια είναι απαραίτητο να αλλάξει η σειρά των πινάκων, έτσι ώστε να αντιστοιχούν στη σειρά ανάπτυξης του κύβου, δηλαδή F,R,B,L,U,D. Τέλος, απαιτείται η μετάθεση των στοιχείων του κάθε πίνακα έτσι ώστε να ταιριάζει στον προσανατολισμό του αναπτυγμένου κύβου.



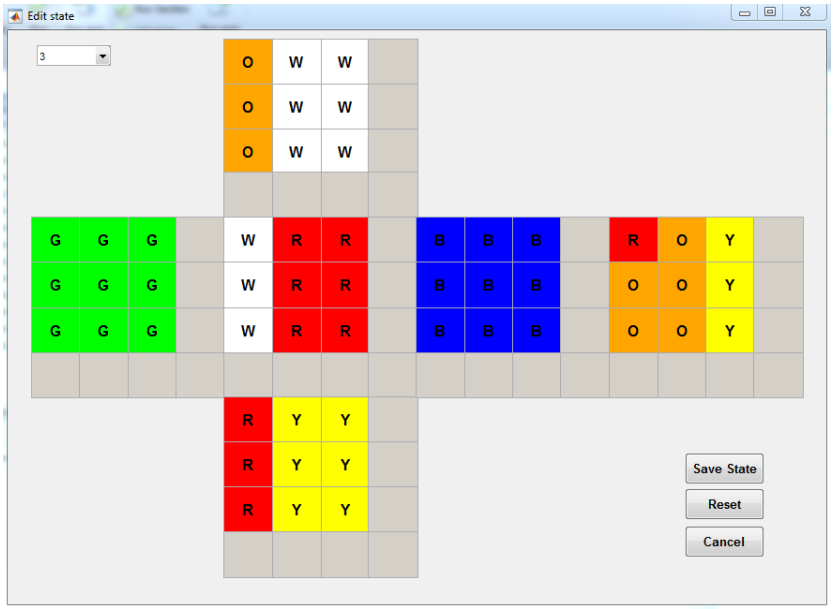
Εικόνα. 27 Ανάπτυγμα του κύβου για την επίλυση του μέσω του κώδικα

```
%%etimasia xromatos gia lusi apo solve45
for i=1:6
    for j=1:9
        if finalcolor(i,j)==12
            finalcolor(i,j)=5;
        end
         if finalcolor(i,j)==7
            finalcolor(i,j)=3;
         end
         if finalcolor(i,j)==8
            finalcolor(i,j)=1;
         end
         if finalcolor(i,j)==4
            finalcolor(i,j)=4;
         end
        if finalcolor(i,j)==2
            finalcolor(i,j)=2;
        end
    end
end
%%etimasia pinaka gia lusi apo solve45
R1=zeros(3,3,6);
%%bazoume tis theseis stis theseis pou katalabenei to solve45

 for k=1:6
    T(k,1)=finalcolor(k,2);
    T(k,2)=finalcolor(k,1);
    T(k,3)=finalcolor(k,8);
    T(k,4)=finalcolor(k,3);
    T(k,5)=finalcolor(k,9);
    T(k,6)=finalcolor(k,7);
    T(k,7)=finalcolor(k,4);
    T(k,8)=finalcolor(k,5);
    T(k,9)=finalcolor(k,6);

 end
 %%kanoume ton pinaka 6x9 se pinakes 3x3x6
 l=0;
```

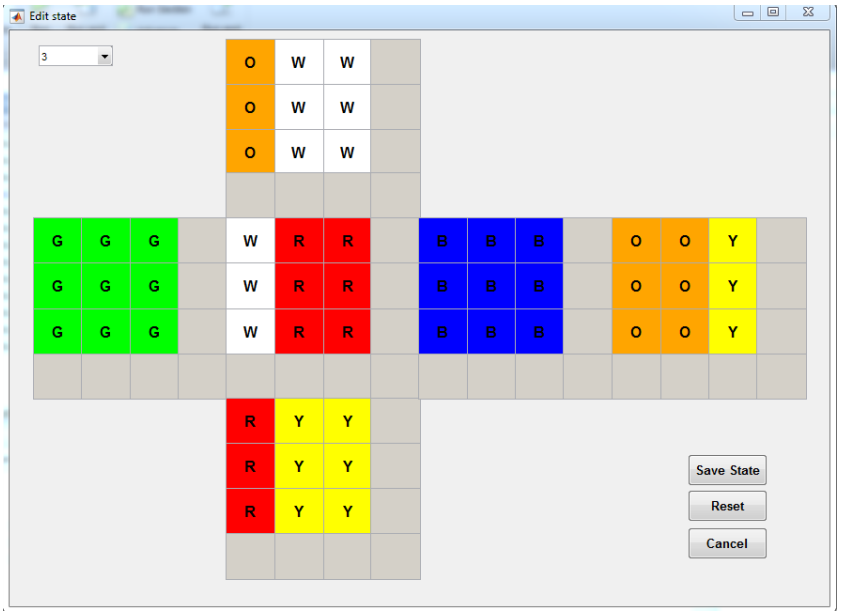
```
 for k=1:6
   for i=1:3
     for j=1:3
         l=l+1;
         R(i,j,k)=T(k,l);
     end

   end
    l=0;
 end

 %%alazoume tin kateuthinsi ton pinakwn etsi wste na
teriazoun sto pws ta pernei
 %%to solve45
R(:,:,1)=rot90(R(:,:,1),3);
R(:,:,2)=rot90(R(:,:,2),3);
R(:,:,3)=rot90(R(:,:,3),3);
R(:,:,4)=rot90(R(:,:,4),3);
R(:,:,5)=rot90(R(:,:,5),3);
R(:,:,6)=rot90(R(:,:,6));
```

3.6 Διόρθωση αστοχιών

Ένα βήμα πριν την λύση του κύβου ο χρήστης έχει την δυνατότητα να δει τον κύβο και να διορθώσει τυχόν λάθη από την προηγούμενη αποτύπωση. Αυτή η δυνατότητα δόθηκε διότι ο αισθητήρας χρώματος που χρησιμοποιήθηκε έδινε διαφορετικά αποτελέσματα ανάλογα με τα επίπεδα φωτός του χώρου. Ακόμα οι διαφορές μεταξύ κάποιων χρωμάτων, όπως το πορτοκαλί και κόκκινο, στον κύβο είναι μικρές με αποτέλεσμα ο αισθητήρας πολλές φορές να τα συγχέει. Έτσι, δόθηκε η δυνατότητα στον χρήστη να κάνει έναν τελευταίο έλεγχο αν τα χρώματα και οι θέσεις τους είναι σωστές, αλλά και να διορθώσει τυχόν λάθη.



Εικόνα. 28 Λάθος αναγνώριση του ενός πορτοκαλί υποκύβου ως κόκκινο (στην πλευρά B)



Εικόνα. 29 Χειροκίνητη διόρθωση του κόκκινου χρώματος σε πορτοκαλί

3.7 Επίλυση του κύβου

Για την λύση του κύβου χρησιμοποιείται ο αλγόριθμος του Thistlethwaite. Ο συγκεκριμένος δίνει την δυνατότητα λύσης του κύβου με το μέγιστο αριθμό κινήσεων να είναι οι σαράντα πέντε, αν και συνήθως χρειάζονται πολύ λιγότερες. Επίσης, διαφέρει από τις στρατηγικές επίλυσης των ανθρώπινων αλγορίθμων, καθώς δεν αποκαθιστά τον κάθε επιμέρους κύβο ξεχωριστά στη σωστή του θέση, αλλά

επεξεργάζεται τις θέσεις όλων των υποκύβων ταυτόχρονα, περιορίζοντας ολοένα και περισσότερο τις διαθέσιμες στροφές κάθε επιπέδου. Προϋπόθεση, λοιπόν, είναι η κατάτμησή του σε υποπροβλήματα. Αυτό συμβαίνει με την εξής διαδικασία: αρχικά εκτελούνται στροφές κάθε είδους μέχρις ότου ο κύβος να μεταπέσει σε μια κατάσταση που επιλύεται χωρίς να απαιτούνται μονές στροφές για τις έδρες U και D, αν και μπορεί να απαιτούνται διπλές περιστροφές. Η ίδια διαδικασία επαναλαμβάνεται, αλλά αυτή τη φορά με τις πλευρές F και B. Καταλήγει, έτσι, σε μία κατάσταση που επιλύεται χωρίς τη χρήση μονών στροφών, παρά μόνο με διπλές στροφές, οπότε και επιλύεται. Τα τέσσερα αυτά στάδια επίλυσης είναι αρκετά περίπλοκα, εφόσον χρησιμοποιούν πίνακες αναζήτησης (lookup tables). [11]

Οι πίνακες αναζήτησης

Ακολουθία υποομάδων	Καταστάσεις
$G_0 = \langle L, R, F, B, U, D \rangle$	$4.33 \cdot 10^{19}$
$G_1 = \langle L, R, F, B, U^2, D^2 \rangle$	$2.11 \cdot 10^{16}$
$G_2 = \langle L, R, F^2, B^2, U^2, D^2 \rangle$	$1.95 \cdot 10^{10}$
$G_3 = \langle L^2, R^2, F^2, B^2, U^2, D^2 \rangle$	$6.63 \cdot 10^{5}$
$G_4 = \langle I \rangle$	1

Στο πρώτο στάδιο διορθώνεται ο προσανατολισμός των γωνιακών υποκύβων. Οι συγκεκριμένοι, είναι αδύνατο να αναστραφούν (flip), αν επιτρέπονται μόνο διπλές στροφές για τις έδρες U και D. Στο δεύτερο στάδιο διορθώνονται οι προσανατολισμοί όλων των κορυφαίων υποκύβων, τοποθετώντας ακόμα τους γωνιακούς υποκύβους στην αντίστοιχη έδρα που ανήκουν. Στο συγκεκριμένο στάδιο επιτρέπονται μόνο διπλές στροφές για τις έδρες F, B, U, D, οπότε οι έδρες L,R μπορούν να έχουν έως δύο διαφορετικά χρώματα. Στο τρίτο στάδιο διορθώνεται η θέση των γωνιακών υποκύβων των εδρών L και R. Στην τελευταία ομάδα θέσεων υπάρχει μόνο μία θέση, η λυμένη κατάσταση του κύβου.

Πίνακας σταδίων επίλυσης

	Στάδιο 1:	Στάδιο 2:	Στάδιο 3	Στάδιο 4	Σύνολο
Μέγιστο πλήθος στροφών	$G_0 \rightarrow G_1$	$G_1 \rightarrow G_2$	$G_2 \rightarrow G_3$	$G_3 \rightarrow G_4$	
Αρχικός αλγόριθμος	7	13	15	17	52
Βελτιωμένος αλγόριθμος	7	12	14	17	50
Βέλτιστο	7	10	13	15	45

3.7.1 Ανάλυση του αλγορίθμου

Μετάβαση από το G0 στο G1: Αυτό το κομμάτι περιλαμβάνει μονό κομμάτια ακμών , και είναι αρκετά απλό, για αυτό το λόγο δεν δίνονται πίνακες. Ένα κομμάτι ακμής μπορεί να θεωρηθεί είτε καλό είτε κακό. Θεωρείται κακό εάν για να βρεθεί στην σωστή του πλευρά χρειάζονται περιττός αριθμός στροφών του ενός τεταρτημόριου U και D (μονές στροφές) διαφορετικά είναι καλό. Επομένως, για να γίνουν όλα τα κομμάτια ακμής καλά, πρέπει να μετακινηθούν στις πλευρές U ή D χωρίς να χρησιμοποιηθούν στροφές του ενός τεταρτημόριου U και D και στην συνεχεία αφού έχουν πάει στις προηγούμενες πλευρές να φτάσουν τελικά στις αντίστοιχες πλευρές τους χρησιμοποιώντας μονές στροφές U και D. Για παράδειγμα, αν και οι δώδεκα υποκύβοι ανήκουν στην κατηγορία κακό, η αλληλουχία D,B,F,U,R,'L',D θα τους διορθώσει όλους.

Μετάβαση από το G1 στο G2. Αυτό που έχει επιτευχθεί μέχρι στιγμής (αν και δεν μοιάζει με αυτό) είναι ο σωστός προσανατολισμός των υποκύβων των ακμών. Στο παρόν στάδιο, το ίδιο γίνεται για τις γωνίες, και για τα κομμάτια ακμής FU,FD,BU,BD που έρχονται στις πλευρές τους. Όπως είναι γνωστό, οι γωνίες γενικά δεν έχουν προσανατολισμό, αλλά στην προκειμένη περίπτωση, κατά προσέγγιση, θα τοποθετηθούν όλες με τον ίδιο τρόπο. Πιο συγκεκριμένα, κάθε γωνιακό κομμάτι έχει είτε L-όψη είτε R-όψη: Μετά την ολοκλήρωση αυτού του σταδίου κάθε μία από αυτές τις όψεις θα βρίσκεται ή στην πλευρά L ή στην πλευρά R. Το ίδιο θα γίνει και για τα οκτώ κομμάτια ακμών με L ή R όψη. Υπάρχουν 1082565 περιπτώσεις που πρέπει να εξετασθούν εδώ, αριθμός που προέρχεται από την δύναμη 3^7 (συνολικός αριθμός γωνιακών προσανατολισμών) και 12^4 (συνολικός αριθμός διατάξεων του συνόλου {FU,FD,BU,BD} μεταξύ των δώδεκα θέσεων ακμής). Εντυπωσιακό είναι το γεγονός, πως είναι δυνατή η υλοποίηση αυτού του σταδίου με το πολύ δεκαεπτά κινήσεις χωρίς τη χρήση πινάκων, γεγονός που δεν ισχύει στο επόμενο στάδιο παρά τις λιγότερες περιπτώσεις (μονάχα 29400). Ωστόσο, με τη χρήση προκαθορισμένων πινάκων, οι δεκαεπτά κινήσεις μπορούν να μειωθούν σε δεκατρείς, ενώ με περισσότερους υπολογισμούς είναι εφικτό να μειωθούν σε δέκα. Προκειμένου να αποδειχθεί, ότι οι δέκα κινήσεις είναι επαρκείς, θα έπρεπε να ελεγχθεί πως μέσα από τις ακολουθίες των δέκα κινήσεων στον υπολογιστή, θα προέκυπταν 1082565 διαφορετικές περιπτώσεις, υπολογισμός που θα απαιτούσε αρκετή υπολογιστική ισχύ και χρόνο.

Σχετικά με την επίλυση, η στροφή μιας γωνίας υπολογίζεται εξετάζοντας την L ή R όψη και παρατηρώντας πως αυτή περιστρέφεται σε σχέση με την προσκείμενη επιφάνεια L ή R. Εδώ αξίζει να σημειωθεί, πως τα τεταρτημόρια των πλευρών F και B, μεταβάλλουν την συστροφή των γωνιών, ενώ αντιθέτως οι υπόλοιπες κινήσεις της ομάδας G1 δεν έχουν καμία επίδραση. Το στάδιο G1 υλοποιείται το πολύ σε τέσσερις

κινήσεις, όπου τα κομμάτια των ακμών {FU,FD,BU,BD } έρχονται όλα στην UD-πλευρά. Στην περίπτωση που αυτό δεν είναι εφικτό να συμβεί, δηλαδή να έρθουν στην UD-πλευρά, έρχονται στην U-πλευρά το πολύ σε τέσσερις κινήσεις. Στη συνέχεια εξετάζεται ο αντίστοιχος, λεπτομερής πίνακας.

Μετάβαση από το G2 στο G3: Το συγκεκριμένο στάδιο θεωρείται το πιο περίπλοκο και γι' αυτό αναλύεται σε δύο υποστάδια. Στο πρώτο υποστάδιο, οι γωνίες λαμβάνουν τις φυσικές του τροχιές, ενώ στο δεύτερο μεταβάλλονται οι γωνίες μέσα στις τροχιές τους έτσι ώστε να αποκτήσουν μία από τις 96 γωνιακές μεταβολές από την ομάδα τετραγώνων (G3), ενώ παράλληλα διαχωρίζονται τα κομμάτια ακμής στις σωστές πλευρές του κύβου. Ο πίνακας των αρχικών κινήσεων της πρώτης σελίδας του 3ου Σταδίου, υλοποιεί μέρος από αυτά τα δύο υποστάδια, ενώ οι λεπτομερείς πίνακες υλοποιούν το υπόλοιπο κομμάτι. Μετά την εκτέλεση των αρχικών κινήσεων, το σύνολο των γωνιών που θα είναι εκτός τροχιάς θα βρίσκονται σε μία από τις τρεις περιπτώσεις (συμμετρία μοντέλου):

i. στο κενό σύνολο

ii. στο σύνολο {1,8,2,7}

iii. στο σύνολο {1,5}

Στη συνέχεια, κατά την ανάγνωση του κύβου πρέπει να προσδιοριστεί σε ποιά υπο-μορφή του σταδίου G3αβ βρίσκεται η μετάθεση των γωνιών, όπου α είναι ένα από τα παρακάτω: (14)(68), (24)(58), (12), (14), (24), και το β είναι ξανά ένα από τα: (18)(27), (15).

Καθώς μερικά από τα υποσύνολα είναι όμοια με άλλα, δεν κρίνεται απαραίτητο να παράγονται πίνακες για όλες τις έξι πιθανές τιμές του α.

Μετάβαση από το G3 στη λυμένη κατάσταση. Στο τελευταίο αυτό στάδιο χρησιμοποιούνται μόνο διπλές στροφές. Οι πίνακες για αυτό το στάδιο δίνουν λέξεις για την παραγωγή καθεμιάς από τις 6912 ακραίες θέσεις με σταθερές γωνίες. Επομένως πρέπει πρώτα να επαναφερθούν οι γωνίες (σε 4 κινήσεις το πολύ) και στη συνέχεια να χρησιμοποιηθούν οι πίνακες για να επαναφερθούν οι άκρες

Η συνάρτηση που χρησιμοποιήθηκε λέγεται solve45() και μαζί με τις σχετικές υποσυναρτήσεις παρέχονται από το MATLAB Central File Exchange submission και γραφτήκαν από τον Joren Heit.

3.7.2 Άλλοι αλγόριθμοι και μέγιστος αριθμός κινήσεων

Μετά από έρευνες των Tomas Rokicki, Herbert Kociemba, Morley Davidson, and John Dethridge σε συνδυασμό με τους υπερυπολογιστές της Google βρέθηκε ότι ο μέγιστος αριθμός κινήσεων που χρειάζεται ο κύβος από οποιαδήποτε κατάσταση για να λυθεί είναι είκοσι. Αυτό επιταχύνθηκε χρησιμοποιώντας υπολογιστική δύναμη που αντιστοιχεί σε τριανταπέντε συνεχόμενα χρόνια χρήσης του επεξεργαστή ενός οικιακού υπολογιστή. Σε αυτή τη διαδικασία λυθήκαν και οι 43,252,003,274,489,856,000 δυνατές καταστάσεις του κύβου βρίσκοντας πέρα από κάθε αμφιβολία το μέγιστο αριθμό κινήσεων. Ο αριθμός αυτός ονομάστηκε αριθμός του θεού για τον κύβο του Rubik. [12]

Πίνακας 1: Ιστορική ανάδρομη μέγιστων κινήσεων επίλυσης του κύβου

Date	Lower bound	Upper bound	Gap	Notes and Links
July, 1981	18	52	34	Morwen Thistlethwaite proves 52 moves suffice.
December, 1990	18	42	24	Hans Kloosterman improves this to 42 moves.
May, 1992	18	39	21	Michael Reid shows 39 moves is always sufficient.
May, 1992	18	37	19	Dik Winter lowers this to 37 moves just one day later!
January, 1995	18	29	11	Michael Reid cuts the upper bound to 29 moves by analyzing Kociemba's two-phase algorithm.
January, 1995	20	29	9	Michael Reid proves that the "superflip" position (corners correct, edges placed but flipped) requires 20 moves.
December, 2005	20	28	8	Silviu Radu shows that 28 moves is always enough.
April, 2006	20	27	7	Silviu Radu improves his bound to 27 moves.
May, 2007	20	26	6	Dan Kunkle and Gene Cooperman prove 26 moves suffice.
March, 2008	20	25	5	Tomas Rokicki cuts the upper bound to 25 moves.
April, 2008	20	23	3	Tomas Rokicki and John Welborn reduce it to only 23 moves.
August, 2008	20	22	2	Tomas Rokicki and John Welborn continue down to 22 moves.
July, 2010	20	20	0	Tomas Rokicki, Herbert Kociemba, Morley Davidson, and John Dethridge prove that God's Number for the Cube is exactly 20.

Δεν έχει βρεθεί ακόμα ο αλγόριθμος που με απολυτή ακρίβεια βρίσκει πάντα τις βέλτιστες κινήσεις(Gods algorithm), παρόλα αυτά παρακάτω παρουσιάζονται δύο παραδείγματα που τον πλησιάζουν. [13]

Ο Two-Phase-Algorithm που δίνει υποβέλτιστα αποτελέσματα. Ο αλγόριθμος αυτός ενώ σε όσες δοκιμές έχουν γίνει δεν έχει ξεπεράσει ποτέ τις είκοσι κινήσεις, έχει δύο βασικά μειονεκτήματα. Το πρώτο είναι ότι λύνει πολύ μεγαλύτερο αριθμό κύβων με είκοσι κινήσεις από αυτόν που θα έπρεπε. Δηλαδή επιλύει κύβους με περισσότερες κινήσεις από ότι είναι το βέλτιστο. Το δεύτερο είναι ότι για κάποιες δύσκολες καταστάσεις του κύβου χρειάζεται αρκετή ώρα για να βρει την λύση. Χαρακτηριστικό παράδειγμα είναι ότι για τον συνδυασμό κινήσεων L, R2, U2, B', D2, L, D2, F', U', R, U2, L', F', D', R', B2, D2, R', U', F2 χρειάζεται περίπου δεκαέξι λεπτά σε έναν οικιακό υπολογιστή για να βρει την λύση του.

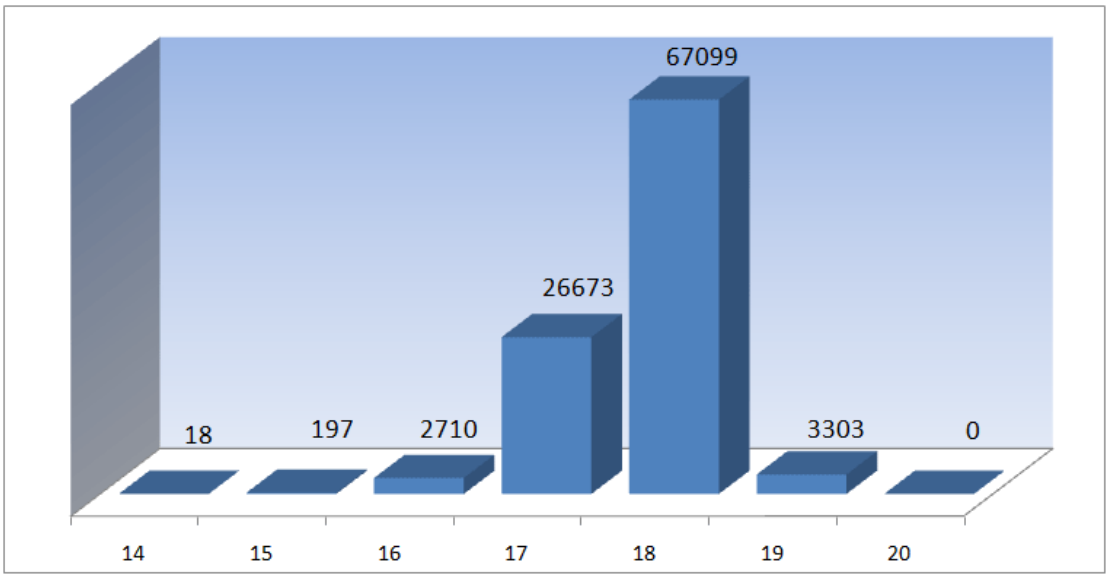
Ο Cube Solver θεωρείται ο βέλτιστος αλγόριθμος και παρόλο που τα αποτελέσματά του πλησιάζουν κατά πολύ την κατανομή που έχουν οι γενικές λύσεις του κύβου δεν έχει αποδειχτεί ότι λύνει όλες τις καταστάσεις με το βέλτιστο αριθμό κινήσεων.

Παρακάτω βλέπουμε τον γενικό πινάκα λύσεων του κύβου αλλά και παραδείγματα της συμπεριφοράς των αλγορίθμων.

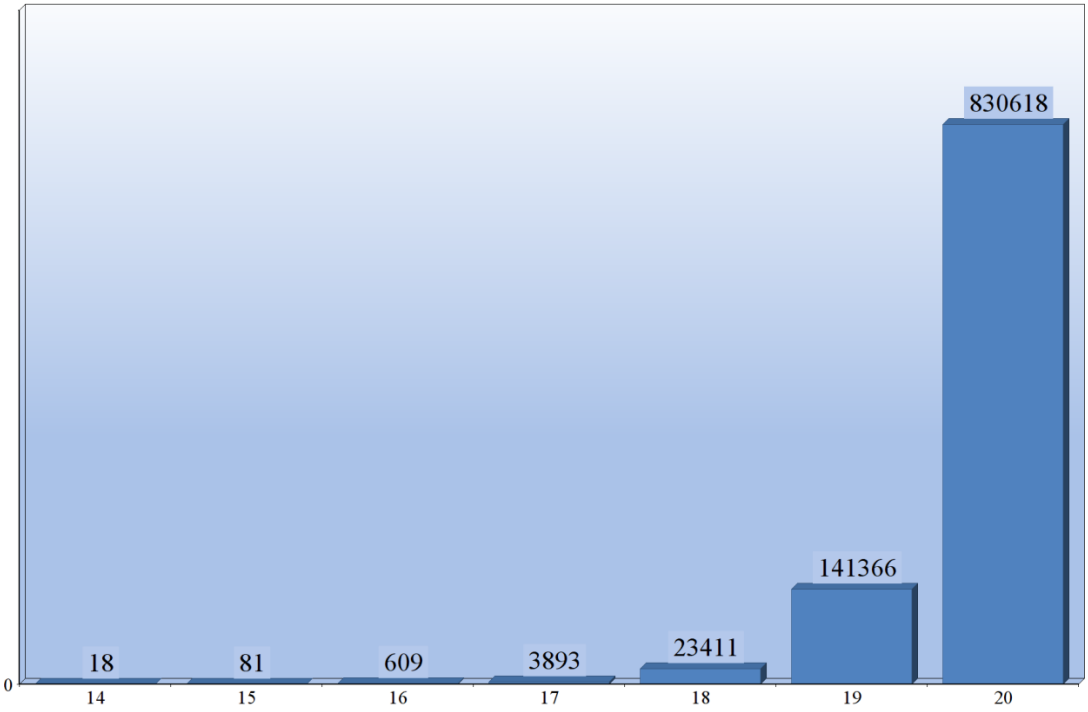
Πίνακας 2:Κινήσεων απαιτούμενων για λυση- Καταστάσεων κύβου

Distance	Count of Positions
0	1
1	18
2	243
3	3,240
4	43,239
5	574,908
6	7,618,438
7	100,803,036
8	1,332,343,288
9	17,596,479,795
10	232,248,063,316
11	3,063,288,809,012
12	40,374,425,656,248
13	531,653,418,284,628
14	6,989,320,578,825,358
15	91,365,146,187,124,313
16	about 1,100,000,000,000,000,000
17	about 12,000,000,000,000,000,000
18	about 29,000,000,000,000,000,000
19	about 1,500,000,000,000,000,000
20	about 490,000,000

Πίνακας 3: Αποτελέσματα 10.000 λύσεων με την χρήση του Cube Solver



Πίνακας 4:Αποτέλεσμα 1.000.000 λύσεων για Two Phace Algorithm



3.8 Κώδικας για την μηχανική λύση του κύβου

Το τελευταίο μέρος του κώδικα μετατρέπει τις θεωρητικές κινήσεις του προηγούμενου αλγορίθμου σε κινήσεις στο NXT. Ο προηγούμενος αλγόριθμος επιστρέφει συνολικά δεκαοκτώ διαφορετικές κινήσεις. Η κίνηση U σημαίνει ότι πρέπει να περιστραφεί η πάνω πλευρά(Up) κατά 90 μοίρες δεξιόστροφα, η U' 90 μοίρες αριστερόστροφα και η U2 180 μοίρες δεξιόστροφα. Αντίστοιχες είναι και οι κινήσεις για τις υπόλοιπες πέντε πλευρές, R, F, D, L και B. Το πρώτο που αποθηκεύεται είναι ο αριθμός των κινήσεων που επέστρεψε ο αλγόριθμος, έτσι ώστε να υπάρξει βρόγχος με κατάλληλο μέγεθος. Στην συνέχεια, ανάλογα με την εκάστοτε κίνηση που χρειάζεται, τοποθετείται η απαιτούμενη πλευρά να ''κοιτάει'' τη βάση περιστροφής, οι υπόλοιπες πλευρές μένουν ακίνητες από τη δαγκάνα και περιστρέφεται η πλευρά κατά την φορά και το μέτρο που ζητείται.

```
t=numel(sol);
for i=1:t
    %%kinisi gia lusi asprou
    if isequal(sol{i},'U')
        kinisidaganasgiakratima();
        kinisiU();
        kinisidaganasgiaafima();
    elseif isequal(sol{i},'U''')
        kinisidaganasgiakratima();
            kinisiU1();
            kinisidaganasgiaafima();
    elseif isequal(sol{i},'U2')
        kinisidaganasgiakratima();
```

```
    kinisiU2();
    kinisidaganasgiaafima();
%%kinisi gia lusi mple
elseif isequal(sol{i},'R')
    kinisidaganas();
    kinisidaganasgiakratima();
        kinisiU();
        kinisidaganasgiaafima();
         kinisidaganas();
         kinisidaganas();
          kinisidaganas();
elseif isequal(sol{i},'R''')
     kinisidaganas();
    kinisidaganasgiakratima();
    kinisiU1();
    kinisidaganasgiaafima();
     kinisidaganas();
      kinisidaganas();
       kinisidaganas();
elseif isequal(sol{i},'R2')
     kinisidaganas();
    kinisidaganasgiakratima();
        kinisiU2();
        kinisidaganasgiaafima();
         kinisidaganas();
          kinisidaganas();
           kinisidaganas();
%%kinisi  gia lusi tou kitrinou

 elseif isequal(sol{i},'D')
     kinisidaganas();
     kinisidaganas();
     kinisidaganasgiakratima();
     kinisiU();
     kinisidaganasgiaafima();
     kinisidaganas();
     kinisidaganas();
 elseif isequal(sol{i},'D''')
     kinisidaganas();
     kinisidaganas();
     kinisidaganasgiakratima();
     kinisiU1();
     kinisidaganasgiaafima();
     kinisidaganas();
     kinisidaganas();
 elseif isequal(sol{i},'D2')
     kinisidaganas();
     kinisidaganas();
     kinisidaganasgiakratima();
     kinisiU2();
     kinisidaganasgiaafima();
     kinisidaganas();
     kinisidaganas();

     %%kinisi  gia lusi tou prasino

 elseif isequal(sol{i},'L')
     kinisidaganas();
     kinisidaganas();
     kinisidaganas();
     kinisidaganasgiakratima();
     kinisiU();
     kinisidaganasgiaafima();
```

```
    kinisidaganas();
elseif isequal(sol{i},'L''')
    kinisidaganas();
    kinisidaganas();
    kinisidaganas();
    kinisidaganasgiakratima();
    kinisiU1();
    kinisidaganasgiaafima();
    kinisidaganas();
elseif isequal(sol{i},'L2')
    kinisidaganas();
    kinisidaganas();
    kinisidaganas();
    kinisidaganasgiakratima();
    kinisiU2();
    kinisidaganasgiaafima();
    kinisidaganas();

    %%kinisi gia lusi tou portokali

elseif isequal(sol{i},'B')
    kinisiU();
    kinisidaganas();
    kinisidaganasgiakratima();
    kinisiU();
    kinisidaganasgiaafima();
    kinisidaganas();
    kinisidaganas();
    kinisidaganas();
    kinisiaristera();
elseif isequal(sol{i},'B''')
    kinisiU();
    kinisidaganas();
    kinisidaganasgiakratima();
    kinisiU1();
    kinisidaganasgiaafima();
    kinisidaganas();
    kinisidaganas();
    kinisidaganas();
    kinisiaristera();
 elseif isequal(sol{i},'B2')
    kinisiU();
    kinisidaganas();
    kinisidaganasgiakratima();
    kinisiU2();
    kinisidaganasgiaafima();
    kinisidaganas();
    kinisidaganas();
    kinisidaganas();
    kinisiaristera();

   %%kinisi gia lusi kokkino

   elseif isequal(sol{i},'F')
    kinisiaristera();
    kinisidaganas();
    kinisidaganasgiakratima();
    kinisiU();
    kinisidaganasgiaafima();
    kinisidaganas();
    kinisidaganas();
    kinisidaganas();
    kinisideksia();
```

```
            elseif isequal(sol{i},'F''')
            kinisiaristera();
            kinisidaganas();
            kinisidaganasgiakratima();
            kinisiU1();
            kinisidaganasgiaafima();
            kinisidaganas();
            kinisidaganas();
            kinisidaganas();
            kinisideksia();
            elseif isequal(sol{i},'F2')
            kinisiaristera();
            kinisidaganas();
            kinisidaganasgiakratima();
            kinisiU2();
            kinisidaganasgiaafima();
            kinisidaganas();
            kinisidaganas();
            kinisidaganas();
            kinisideksia();
    end
end
```

4. Προβλήματα

Τα προβλήματα που αντιμετωπίστηκαν ήταν αρκετά και ποίκιλαν ανάλογα με το στάδιο επίλυσης. Το πρώτο που παρουσιάστηκε ήταν η αδυναμία του αισθητήρα χρώματος να διαχωρίσει με ακρίβεια τα χρώματα του κύβου. Αυτό συνέβη κατά κύριο λόγο μεταξύ χρωμάτων παρεμφερή μεταξύ τους, όπως το μπλε και το γαλάζιο, αλλά ακόμα και σε κύρια χρώματα που χρειάζονταν, όπως το πορτοκαλί και το κόκκινο. Αντιμετωπίστηκε με βελτιστοποίηση του αλγορίθμου εύρεσης χρώματος αλλά και με ένα user interface που προστέθηκε, έτσι ώστε ο χρήστης να έχει την δυνατότητα να ελέγξει τα χρώματα που αποθήκευσε ο αισθητήρας σε σχέση με τα πραγματικά.

Στην συνέχεια, εμφανίστηκαν προβλήματα σχετικά με την κίνηση των σερβοκινητήρων. Οι κινητήρες του NXT δεν έχουν σχεδιαστεί για να μπορούν να κάνουν κινήσεις με τόση μεγάλη ακρίβεια, μιας και η πρωταρχική τους χρήση είναι να κινούν ρόδες ή απλά συστήματα δαγκανών ή βραχιόνων. Το συγκεκριμένο πρόβλημα δεν λύθηκε σε απόλυτο βαθμό αλλά βελτιστοποιήθηκε δημιουργώντας ένα σετ μικρών κινήσεων έτσι ώστε να επανέρχεται το ρομπότ στις απαραίτητες θέσεις και η λειτουργία του να είναι ικανοποιητική. Το τελευταίο μεγάλο πρόβλημα που συναντήθηκε ήταν η επίλυση του ίδιου του κύβου. Παρόλο που ο αλγόριθμος λύσης του κύβου επέστρεφε σωστές αλληλουχίες κινήσεων, κατέστη αδύνατη η δημιουργία κίνησης του κύβου ώστε να μπορεί ο πραγματικός κύβος να φτάσει σίγουρα στην λυμένη του κατάσταση. Αυτό συνέβη κατά κύριο λόγο, εξαιτίας του ότι η συνδεσμολογία του μηχανικού μέρους του ρομπότ δεν επέτρεπε κινήσεις τόσο μεγάλης ακρίβειας, οι οποίες ήταν απαραίτητες. Συνοψίζοντας τα κύρια προβλήματα είναι:

i. Οι σερβοκινητήρες δεν έχουν την απαραίτητη ακρίβεια
ii. Τα Lego δεν ενδείκνυνται για την δημιουργία τόσο περίπλοκων κατασκευών
iii. Ο αισθητήρας χρώματος ενώ επιστρέφει ικανοποιητικά τα αποτελέσματα, έχει αστοχίες σε παρεμφερή χρώματα

5. Συμπεράσματα

Σύμφωνα με το τελικό αποτέλεσμα η απόδοση του ρομπότ είναι ικανοποιητική. Ο μέσος χρόνος επίλυσης ενός κύβου βρίσκεται πολύ κοντά στα αποτελέσματα άλλων κατασκευαστών που ακολούθησαν πανομοιότυπη αρχιτεκτονική. Οι χρόνοι που απαιτούνται είναι περίπου ένα λεπτό και τριάντα δευτερόλεπτα για την σάρωση του κύβου και περίπου δυο λεπτά για την επίλυση του αν και αυτό εξαρτάται σε πολύ μεγάλο βαθμό από την κατάσταση του κύβου. Επιπλέον, πλεονέκτημα αποτελεί ότι δεν χρησιμοποιήθηκαν εξαρτήματα πέραν του ενός Lego Mindstorms NXT πακέτου και του αισθητήρα της Hitechnic , όπως για παράδειγμα η χρήση κάποιας webcam ή κάποιου δεύτερου πακέτου. Παράλληλα, διαπιστώθηκε η ορθότητα της θεωρητικής ανάλυσης που προηγήθηκε. Από την προσπάθεια, λοιπόν, δημιουργίας του ρομπότ πρόεκυψαν και κάποια θεωρητικά συμπεράσματα. Καταρχάς, η ικανότητα επικοινωνίας του Matlab με το NXT δίνει την δυνατότητα να μπορούν να προγραμματιστούν αρκετά περίπλοκα συστήματα, τα οποία μέσω των συμβατικών προγραμμάτων κωδικοποίησης του brick θα ήταν πολύ δύσκολο έως και αδύνατο να δημιουργηθούν. Ακόμα, έγιναν αντιληπτές οι μαθησιακές δυνατότητες του πακέτου Mindstorm μιας και η πληθώρα αισθητήρων και κινητήρων επαρκούν για την κατανόηση και την αντίληψη αρκετών προβλημάτων μηχανικής και προγραμματισμού. Τέλος, συμπεραίνεται ότι παρόλο που είχαμε ένα αρκετά ικανοποιητικό αποτέλεσμα, το πακέτο NXT δεν προτείνεται για κατασκευές που απαιτούν χειρισμούς ακριβείας.

6. Προοπτικές

Οι προοπτικές της παρούσας διπλωματικής μπορούν να χωριστούν σε δυο κατηγορίες. Η πρώτη κατηγορία είναι η χρήση της ίδιας της διπλωματικής για μαθησιακούς σκοπούς. Δηλαδή η χρήση του κατασκευασμένου συστήματος κινητήρων και αισθητήρα, έτσι ώστε οι φοιτητές και λοιποί εκπαιδευόμενοι, να μπορούν να αντιληφθούν ευκολότερα πως μικρές αλλαγές στον κώδικα ή στο μηχανικό τμήμα, μπορούν να βελτιώσουν ή να καταστρέψουν τελείως την λειτουργία. Ακόμα, βοηθάει στην αντίληψη ότι αισθητήρες, κινητήρες και λοιπά μηχανικά αντικείμενα που έχουν σχεδιαστεί για έναν αρχικό σκοπό, όταν είναι πλέον παρωχημένα, αντί να αχρηστευτούν μπορούν να χρησιμοποιηθούν για διαφορετικούς σκοπούς. Η δεύτερη είναι η προοπτική βελτίωσης της ίδιας της διπλωματικής. Αρχικά, ο αλγόριθμος που χρησιμοποιήθηκε για την θεωρητική επίλυση του κύβου είναι υποβέλτιστος. Θα μπορούσε, λοιπόν, να αντικατασταθεί από έναν βέλτιστο ή ακόμα και από έναν ανθρώπινο αλγόριθμο επίλυσης, ώστε να συγκριθούν τα αποτελέσματα. Είναι δυνατή, επίσης, η προσθήκη παραπάνω αισθητήρων έτσι ώστε να δημιουργηθεί ένα σύστημα ανάδρασης που θα μπορούσε να διορθώνει αστοχίες σε πραγματικό χρόνο. Τέλος, πρόκληση αποτελεί η δημιουργία ενός καλύτερου μηχανικού συστήματος για το στάδιο επίλυσης του κύβου, αλλά και του προγραμματισμού του.

Βιβλιογραφία

[1] [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Rubik%27s_Cube.

[2] [Ηλεκτρονικό]. Available: https://eu.rubiks.com/.

[3] [Ηλεκτρονικό]. Available: https://el.wikipedia.org/wiki/Lego_Mindstorms.

[4] [Ηλεκτρονικό]. Available: https://www.Lego.com/en-us/mindstorms.

[5] [Ηλεκτρονικό]. Available: https://www.hitechnic.com/cgi-bin/commerce.cgi?key=NCO1038&preadd=action.

[6] [Ηλεκτρονικό]. Available: https://www.mathworks.com/products/Matlab.html.

[7] [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/MATLAB.

[8] [Ηλεκτρονικό]. Available: http://www.mindstorms.rwth-aachen.de/.

[9] [Ηλεκτρονικό]. Available: https://www.Lego.com/en-us/mindstorms/downloads.

[10] [Ηλεκτρονικό]. Available: http://mindcuber.com/.

[11] [Ηλεκτρονικό]. Available: https://www.jaapsch.net/puzzles/thistle.htm.

[12] [Ηλεκτρονικό]. Available: https://www.cube20.org/.

[13] [Ηλεκτρονικό]. Available: http://kociemba.org/moves20.htm.