ΡΟΜΠΟΤΙΚΟ ΠΟΔΟΣΦΑΙΡΟ

Σχολή Τεχνολογικών Εφαρμογών (Σ.Τ.Ε.Φ.)
Τμήμα Μηχανολόγων Μηχανικών ΤΕΙ Ηρακλείου ,
τομέας ρομποτική - μηχατρονική

ΠΕΡΙΕΧΟΜΕΝΑ

1. Σκοπός της εργασίας

Η συγκεκριμένη εργασία είναι μία πρωτη επαφή για την υλοποίηση του θέματος «ρομποτικό ποδόσφαιρο» και θα μπορούσε να αποτελέσει ένα μελλοντικό στόχο , για να λάβει μέρος το ΤΕΙ Ηρακλείου, στη διοργάνωση αγώνων που πραγματοποιείται σχεδόν κάθε χρόνο σε επίπεδο Ιδρυμάτων ή ομάδων ή διαφορετικών χωρών μεταξύ τους.

Κάτω από αυτό τον τίτλο , ρομποτικό ποδόσφαιρο , υπάρχει μία ποικιλία διαφορετικών εργασιών και διαφορετικής δυσκολίας κάτω από συγκεκριμένες προϋποθέσεις που έχει καθορίζει η ομοσπονδία ρομποτικού ποδοσφαίρου F.I.R.A που θα αναφερθεί αναλυτικότερα σε επόμενη θεματική ενότητα. Για την διεκπεραίωση της συγκεκριμένης εργασίας απαιτείται η γνώση μαθηματικών - γραμμική γεωμετρία, πληροφορικής – ικανότητας προγραμματισμού – γνώση οπτικής αναγνώρισης και γνώση μικροελεγκτών - αισθητήρων.

Ανέλαβα πρίν αρκετό καιρό την συγκεκριμένη εργασία με μεγάλο ενδιαφέρον χωρίς να έχω ασχοληθεί ποτέ με κάτι περεμφερές. Δεν υπάρχει μοναδικός τρόπος για την υλοποίηση της συγκεκιμένης εργασίας , καθως εξαρτάται καταρχήν από τον τύπο του υλικού που θα χρησιμοποιηθεί και τον τρόπο στρατηγικής της λειτουργίας του. Οι δεσμέυσεις – κανονισμοί που υπάρχουν αναφέρονται στα μηχανικά – ηλεκτρονκά χαρακτηριστικά , ώστε ο διαγωνισμός να γίνεται μεταξύ ισότιμων αντιπάλων .

Έγιναν πολλαπλές διαφορετικές σκέψεις και προσπάθειες για την επίλυση της εργασίας, όπως στον σχεδιασμό για την οπτική αναγώριση , στην αποστολή δεδομένων για την ενεργοποίηση κίνησης , στον τρόπο της κίνησης καθώς και σε άλλες διεργασίες που προέκυψαν. Ο περισσότερος χρόνος σπαταλήθηκε στις δοκιμαστικές προσπάθειες , καθώς εμφανίστηκαν πολλά προβλήματα τα οποία δεν ήταν ορατά εξ αρχής, σε διάφορα τμήματα ανάπτυξης της εργασίας τα οποία επιλύθηκαν , με κόστος αρκετές φορές , την αλλαγή μεγάλου μέρους του κώδικα.

Να αναφέρω ότι υπήρξε μία μικρη βοήθεια από το τμήμα , στην ενεργοποίηση του ρομποτικού παίκτη ώστε να μπορεί να λαμβάνει σειριακά ένα byte και να μπορεί να εκκινήσει έναν τροχό.

Οι γνώσεις που έχω αποκτήσει κατά την διάρκεια των σπουδών μου, από την τεχνολογία αισθητήρων, ρομποτική , εφαρμογές μικροελεγκτών, προγραμματισμό, μαθηματικά, μηχανική καθώς και τεχνικές της οπτικής αναγνώρισης από εξωτερικές πηγές εκμάθησης, δόθηκε η ευκαιρία να εφαρμοστούν συνολικά στην πράξη για την υλοποίηση της εργασίας , από τα πρώτα βήματα του σχεδιασμού μέχρι την επιτυχία τερμάτων, που είναι και ο τελικός στόχος της συγκεκριμένης εργασίας , αλλα στη συγκεκριμένη εργασία χωρίς αντίπαλο.

Μετά από αυτή την επαφή σε πρακτικό επίπεδο, δημιούργηθηκε ελάχιστη σχετική εμπειρία η οποία σε παρόμοια εφαρμογή μελλοντικά, θα μπορούσε να με βοηθήσει

προσωπικά , να αναπτύξω με πιο σωστό τρόπο, γρηγορότερα και να αποφύγω δυσλειτουργίες ή κακούς σχεδιασμούς τους οποίους να εχουν αντίκτυπο σε οικονομικούς παράγοντες, και χρονοτριβή.

Σαν πτυχιακή εργασία, αντικατοπτρίζει μία πραγματική εφαρμογή που σίγουρα συναντάται σημερα σε αυτόνομα ρομπότ. Ο βαθμός δυσκολίας της συγκεκριμένης εργασίας κατά την γνώμη μου , είναι αρκετά μεγάλος και επιβεβαιώνεται, από το ότι ασχολούνται οργανομένα ολόκληρες ομάδες ατόμων από τα καλύτερα πανεπιστήμια του κόσμου. Η μεγαλύτερη δυσκολία βρίσκεται στην λεπτομέρεια ή στις ειδικές περιπτώσεις , σαν παράδειγμα η γενική λειτουργία μία διαδικασίας περιγράφεται σε δέκα γραμμες κώδικα , ενώ αντιθέτως εάν θα πρέπει να συμπεριληφθεί μία ειδική περίπτωση στην γενική λειτουργία , μπορεί να χρειαστεί τρείς φορές περισσότερος κώδικας.

Θα ήθελα να ευχαριστήσω τον κο Καββουσανό Μανώλη , ο οποίος μου έδωσε την ευκαιρία να εκπονήσω αυτή την εργασία καθώς τον Κο Τουτουντζή που έκανε την βασική επικοινωνία μεταξυ των τροχών του ρομποτικού παίκτη και του μικροελεγκτή, και το τμήμα Μηχανολογίας της ΣΤΕΦ του ΤΕΙ Ηρακλείου , για τις γνώσεις που μου πρόσεφερε κατά την διάρκεια των σπουδών μου.

1.1 Project description

Last twenty years, Robotic soccer projects attracts many people of science and robotics specialists from all the world. It is organized in groups, categories , players act as national or local teams and they follow instructions and rules created by the FIRA (Federation of Internation Robotic Soccer) . The scope of competition that take place, is to have fun , but under the word “game” , there are tested new technologies and programming technics that have influence in the real world.

The project described here , contain some basic steps needed to cover this famous and future project named “robotic soccer”.

These first steps , will help the Technological Institute of Crete (T.E.I) to abtain ideas , creating questions about, how to develop better , such a project in a more professional way , in order to participate as a country or as a team , in the games that take place in the world and be organized by FIRA .

As a student in the Mechanical Engineering of this Insitute (Robotics – Mechatronics section), it was a great opportunity for me to combine the knowledge i learned during the studies and the experience i had in programming , to implement this project.

The technical details of this project , and the rules that required to be played , are described in the Mirosot football league category(3 x 3) and informations about this type of game can be found in the www.fira.net website.

The project that i attempted to finish and present here, includes only a robotic player , a PC system , a camera and the playground. The main goal of the project here, is to make the robotic player always to find the ball, to push it to the opposite side and finally to score the goal.

Without any experience in such a project that includes visual analysis , wireless serial communication and developing in Matlab interface , controling a robot in realtime with many events happened per second , it was a real difficult job for me.

During developing , there were many problems appeared , i have to report as example designing the head of the robotic players , find our robotic players and their characteristics , find the position of the robotic players – ball – enemies in the playground, differentiate between the undesired objects or image noise and real robotic players, sending the right commands to the proper robotic players , creating the path that will follow a robotic player to the ball , pushing the ball and follow another path to the way , in order to score the goal and finally celebrate the goal that is scored.

All these steps during developing reported above , help me to get some experience, that will help me in future projects, to perform faster , better and more professional.

2. Ιστορική εξέλιξη της Ρομποτικής τεχνολογίας

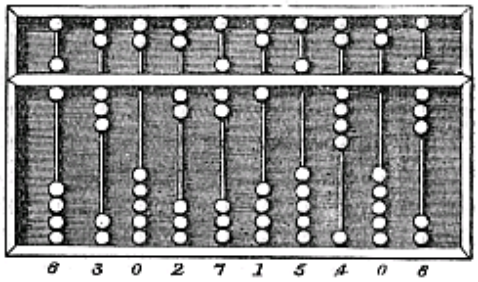
Η Ρομποτική , αποτελεί έναν από τους πλέον σύγχρονους τομείς της επιστήμης και της τεχνολογίας. Η Ρομποτική έχει εφαρμογές στην βιομηχανία, στην ιατρική, στην οικία, στην διαστημική τεχνολογία και γενικά όπου υπάρχει δυσκολία ή κίνδυνος ή αδύναμία της εργασίας απο τον άνθρωπο.

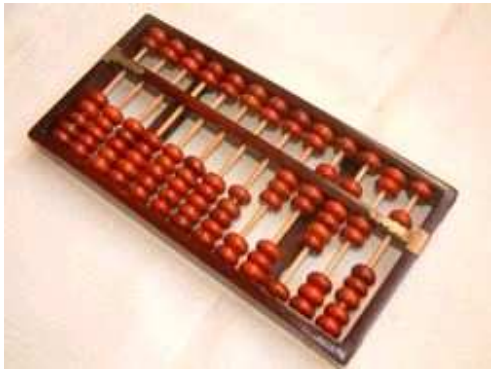
Συγκεκριμενα ο όρος ρομπότ παράγεται από την Τσέχικη λέξη «ρομπότε» που σημαίνει αγγαρεία και χρησιμοποιήθηκε για πρώτη φορά από τον Κ. Τσάπεκ στο θεατρικό έργο «RUR» το 1920, όπου ρομπότ ονομάζονταν οι μηχανικοί άνθρωποι. Η παλαιότερη ελληνική λέξη «αυτόματο», χρησιμοποιείται πλέον περισσότερο για μηχανισμούς που μιμούνται τον άνθρωπο ή κάποιο ζώο, χωρίς αναγκαστικά να παράγουν ωφέλιμο έργο. Επίσης ο νέωτερος όρος «ανδροειδές» αναφέρεται σε ανθρωπόμορφους αλλά όχι όμως σε ζωόμορφους μηχανισμούς.

Μία ιστορική αναδρομή της ρομποτικής και αυτοματισμών έως και πρίν ορισμένα χρόνια, που είναι γνωστοί μέχρι τα σημερινά χρόνια , φαίνεται παρακάτω με τις πληροφορίες που συλλέχθηκαν από διάφορες πηγές του διαδικτύου.

<u>Εξέλιξη σε αυτοματισμούς και ρομποτική σε παλαιότερα έως νεότερα χρόνια</u>

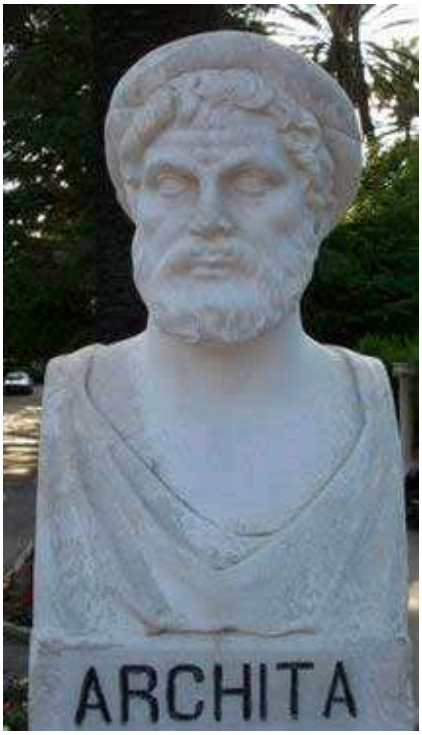
Άβακας, πρίν το 4000 π.Χ , ο θεωρείται σαν η πρώτη υπολογιστική μηχανή που κατασκεύασε ο άνθρωπος στην κοιλάδα της Μεσοποταμίας και στη σημερινή του μορφή το 2.600 π.Χ. από τους Κινέζους. Έχει χάντρες σε δύο τμήματα με τιμές αντίστοιχα 1, 10, 100, 1000 και 5, 50, 500, 5000. Μπορεί να κάνει και τις τέσσερις αριθμητικές πράξεις και κάποιος που είναι εξοικειωμένος με τη χρήση του μπορεί να κάνει πολύ γρήγορα υπολογισμούς.

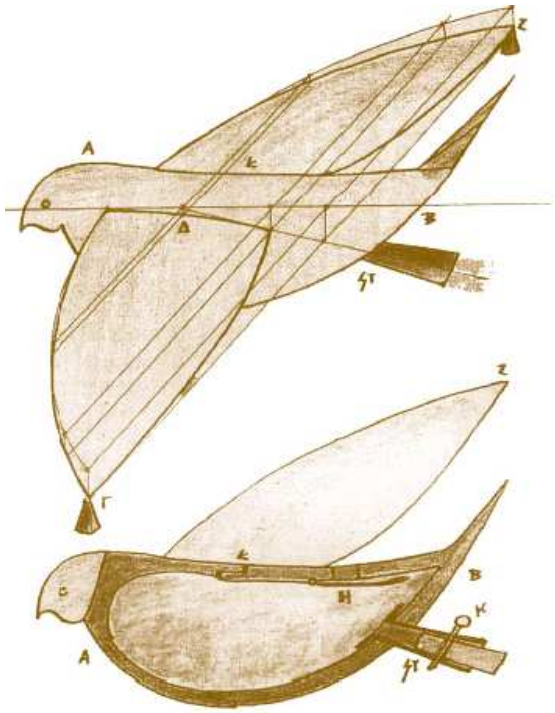




ε2.1

Περίπου το 350 π.Χ. ο Έλληνας μαθηματικός, Αρχύτας ο Ταραντίνος του Τάραντα χτίζει ένα μηχανικό πουλί με όνομα "το περιστέρι" που προωθείται από τον ατμό και μπορούσε να διανύσει περίπου 200m απόσταση αιωρώντας, το οποίο και αποτελεί μία από τις ιστορικά πρώτες μελέτες της πτήσης. Οπτικά πιθανόν να απεικονιζόταν σαν το παρακάτω σχεδιο το δημιουργημα αυτό.

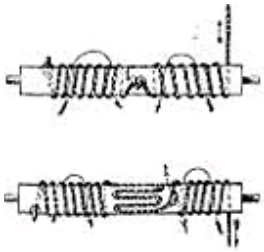




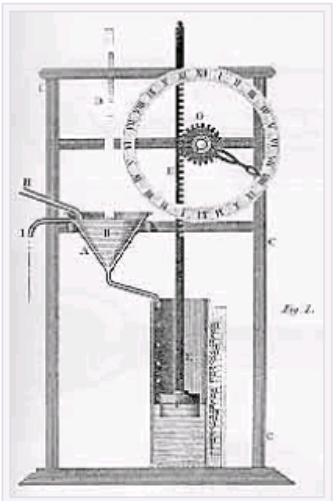
ε2.2

Στο 200 π.Χ. ο έλληνας εφευρέτης Κτησιβιος φυσικός της Αλεξάνδρειας σχεδιάζει ρολόγια του νερού που κινούνται σε σχέση με αυτό. Μετρούσε τον χρόνο σαν απότελεσμα της δύναμης της πτώσης του νερού σε ένα σταθερό ρυθμό και με μία κατάλληλη αναλογικά κατασκευή μπορούσε να μετρήσει τον χρόνο.

ε2.3



ε2.4

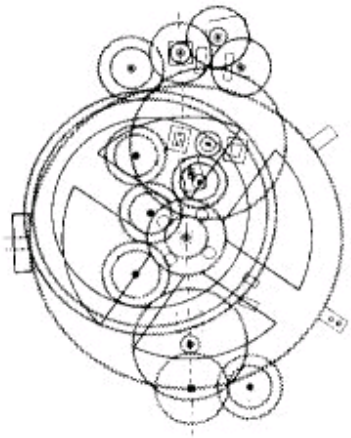


A diagram of Ctesibius's water clock.

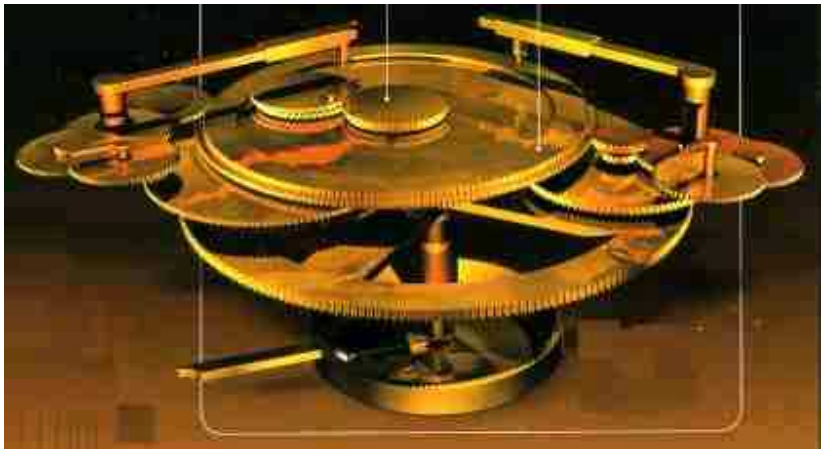
ε2.5

Περίπου 150 έως 100 π.Χ, δημιουργήθηκε ο μηχανισμός των ανικυθύρων , ο οποίος μπορούσε να προβλέψει την θέση των πλανητών , και εκτίθεται στο αρχαιολογικό μουσείο Αθηνών. Λέγεται ότι ο Αρχιμήδης (287 – 212 π.Χ) , έκανε μία πρώτη προσπάθεια να φτιάξει ένα παρόμοιο μηχανισμό και αμέσως μετά ακολούθησε ο συγκεκριμένος των ανιτκυθύρων.

ε2.6



ε2.7

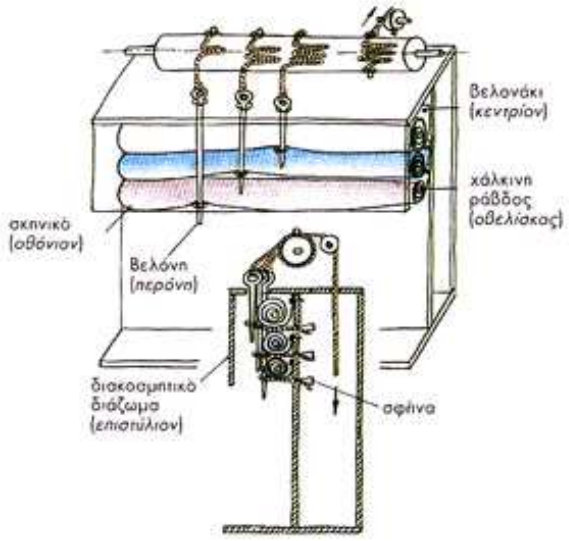


ε2.8

ε2.9

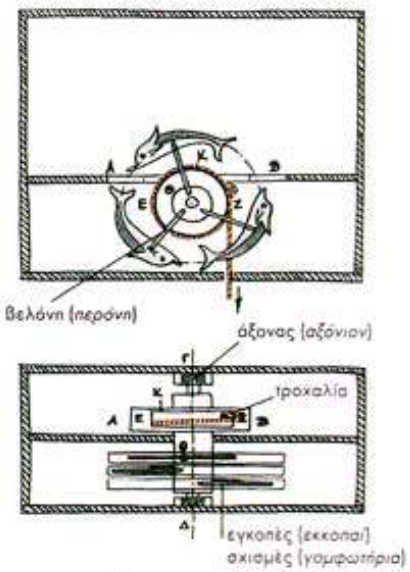
Μετά έχουμε διάφορους αυτοματισμούς από τον Ήρωνα τον Αλεξανδρινό, σοφός ,περίπου το 100 π.Χ , που θεωρείται ο πατέρας της σύγχρονης ρομποτικής όπου δίδαξε στο μουσείο της Αλεξάνδρειας και τα «αυτόματά του» περιγράφονται στο βιβλίο του «Πνευματικά και Αυτομοτοποιητική» και μεχρι να εμφανιστεί η λέξη robot , όλες οι ρομποτικές δημιουργίες ονομαζόνταν αυτόματα από τα αρχαία ελληνικά.

Κατάφερε να φτιάξει το κινούμενο αυτόματο θέατρο , όπου ρυθμίζονταν ανάλογα οι διαστάσεις του , σε ιδιοτικό η δημόσιο και ανέβαιναν αυτόματα οι σκηνές. Ακόμα ο Ηρων είχε αρκετές άλλες δημιουργίες οι οποίες στηρίχτηκαν στις γνώσεις των Κτησίβιου (300 π.Χ) και Φίλωνα (200 π.χ).



Ο μηχανισμὸς αλλαγὴς των σκηνικῶν.

ε2.10



Ο μηχανισμὸς γιὰ την κίνηση των δελφινιών.

ε2.11

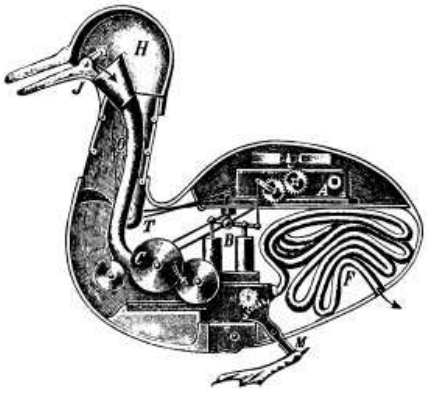
Μετα από το 1000 μ.Χ υπήρξαν περισσότερες δημιουργίες όπως περιγράφονται οι βασικότερες παρακάτω:

Ο άραβας Al-Jazari (1136-1206 μΧ) , κατασκεύασε το πρώτο ανθρωπόμορφο robot το οποίο ήταν ένας τυμπανιστής.

ε2.12

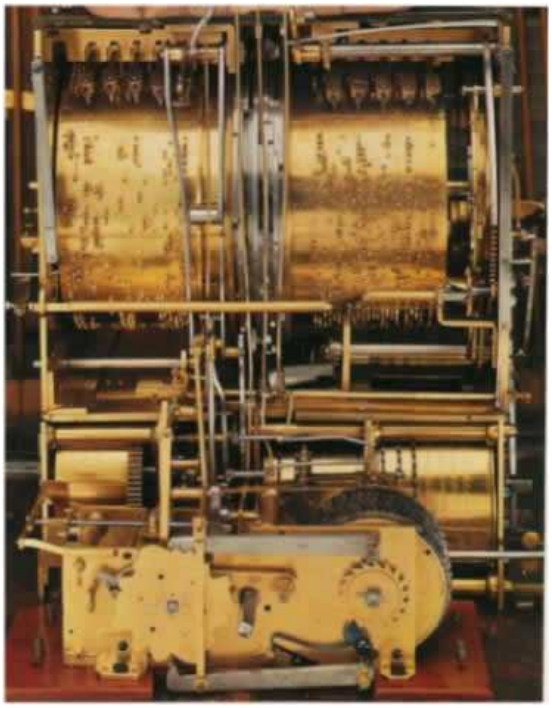
Ο Ιταλός λεονάρντο ντα Βίντσι (1452-1519) σχεδιάζει μια μηχανική συσκευή που μοιάζει με ένα θωρακισμένο ιππότη. Οι μηχανισμοί στο εσωτερικό του ρομπότ Λεονάρντο είναι "σχεδιασμένοι να κάνουν την κίνηση του ιππότη σαν να υπήρχε ένα πραγματικό πρόσωπο μέσα.

Το 1738 ο γάλλος Jacques de Vaucanson κατασκεύασε μια ρομποτική πάπια που είχε την δυνατοτητα να τρώει σπόρους και να κουνάει τα φτερά της. Ηταν ένα παράδειγμα το οποίο αποκαλούσε «κινητή ανατομία» ή ανατομία ζώων με μηχανική.

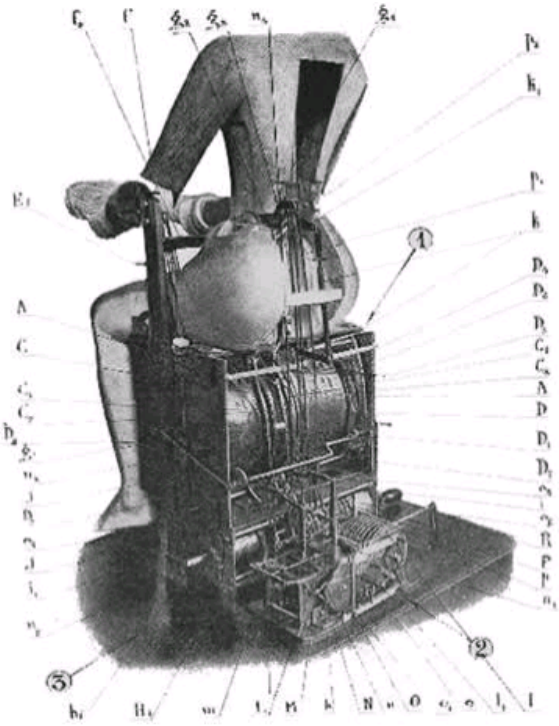


ε2.13

Το 1770, οι ελβετοί μηχανικοί ρολογιών και ο εφευρέτες του σύγχρονου ρολόιου χειρός Pierre Jaquet-Droz Θα δημιουργήσει τρεις κούκλες, η καθέ μία με μια μοναδική λειτουργία. Η μία θα μπορεί να γράψει, η άλλη να παίζει μουσική και η τρίτη να ζωγραφίζει.



ε2.14



ε2.15

ε2.16

Το 1796 Ο ιάπωνας Hishasige Tanaka, κατασκεύασε ρομποτικούς μηχανισμούς που μπορούσαν να σερβίρουν τσάι η να σχεδιάζουν γιαπωνέζικα ιδεογράμματα. Το αποκαλούσαν «Edison Ιαπωνίας» και ήταν και ένας από τους ιδρυτές της γνωστής εταιρίας Toshiba.

ε2.17

Το 1801 ο γάλλος Joseph Jacquard φτιάχνει ένα αυτόματο αργαλειό που ελέγχεται με διάτρητες κάρτες όπου αργότερα χρησιμοποιήθηκαν ως μια μέθοδο εισόδου για ορισμένους από τους πρώτους υπολογιστές του 20ου αιώνα.

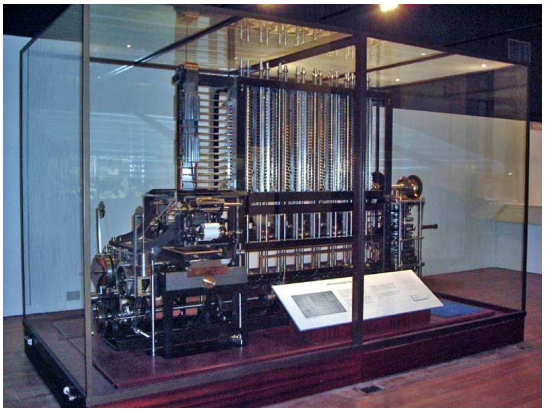
ε2.18

ε2.19

ε2.20

Το 1822 ο Charles Babbage ο Αγγλος εφευρέτης δημιούργησε την " Difference Engine ". Ηταν η πρώτη μηχανή που μπορούσε να υπολογίσει μηχανικά τιμές και συναρτήσεις. Συνεχίζοντας το έργο του, σχεδιάζοντας ένα ακόμη πιο φιλόδοξο σχέδιο "η Αναλυτική Μηχανή", που σύμφωνα με πληροφορίες χρησιμοποιούσε διάτρητες κάρτες που είχαν πάνω τους πρόγραμμα και έκανας τις επαναλήψεις. Συχνά αναφέρεται ως ο "Πατέρας των Υπολογιστών» και για το έργο του σχετικά με το δυαδικό σύστημα αρίθμησης που αποτελεί τη βάση των σύγχρονων υπολογιστών.

ε2.21

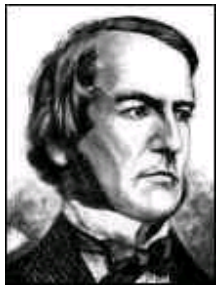


ε2.21

ε2.21

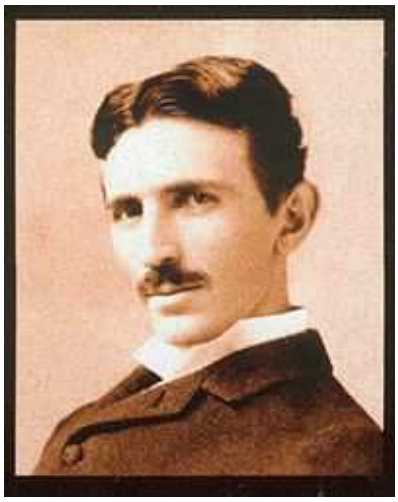
Το 1847 ο άγγλος μαθηματικός George Boole δημιουργεί την άλγεβρα Boole ή οποία είναι βασικός λίθος για την εξέλιξη των μελλοντικών εφευρέσεων σε ότι έχει να κάνει με αυτοματισμούς και λογικη.

\



ε2.22

Το 1898 ο γνωστος για τις εφευρεσεις του στον ηλεκτρομαγνητισμό , γεννημένος στην σημερινή κροατία , Nikola Tesla φτιάχνει τηλεκατευθυνόμενο σκάφος ρομπότ στο Madison Square Garden .



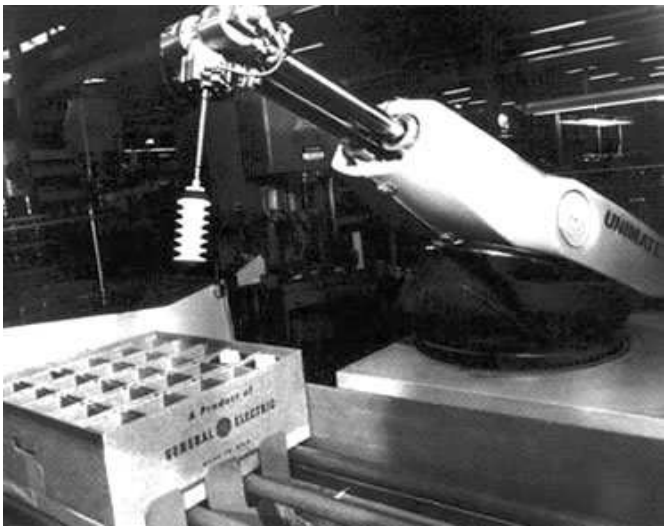
ε2.23

Το 1942 ο αμερικανος Ισαακ Ασιμοβ , προοώθησε την εκλαίκευση του όρου "Ρομποτική". Η συμβολη του ήταν να αναφερει τους τρείς νομους της ρομποτικής :
οτι ενα ρομπότ δεν μπορεί να βλάψει ένα ανθρώπινο ον, ενα ρομπότ πρέπει να υπακούει τις εντολές που δόθηκαν από τον άνθρωπο, και ενα ρομπότ δεν μπορεί να βλάψει την ανθρωπότητα.

ε2.24

Το 1959 ο John McCarthy και Marvin Minsky ειχαν ξεκινήσει το Εργαστηρίου Τεχνητής Νοημοσύνης στο γνωστό ινστιτουτο MIT.

Το 1961 ο Heinrich Ernst αναπτύσσει το MH-1, ένας υπολογιστής που λειτουργεί μηχανικά το χέρι στο MIT.

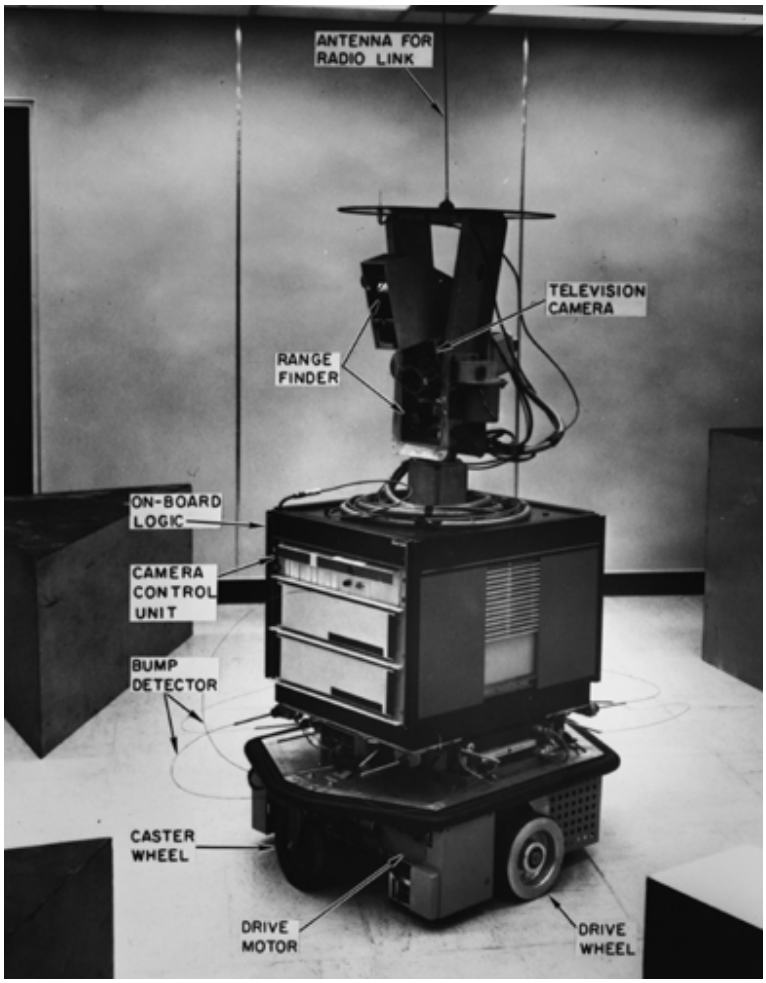


ε2.25

Το 1962 φτιάχνεται το πρώτο βιομηχανικό ρομπότ βραχίονας που χρησιμοποιείται σε επαναλαμβανόμενες ή επικίνδυνες εργασίες πάνω σε μια γραμμή συναρμολόγησης της General Motors.

Το 1963 ο John McCarthy leaves ξεκινάει το Εργαστήριο Τεχνητής Νοημοσύνης στο Πανεπιστήμιο του Στάνφορντ .

Το 1966 στο Stanford Research Institute δημιουργεί το πρώτο κινητό ρομπότ που αντιδράει με τη δικη του βούληση. Ονομάζεται “Shakey”, καταγράφει το περιβάλλον του , βρίσκει τον δρόμο του και ανασυντάσει σωστά διάφορα απλά αντικείμενα.



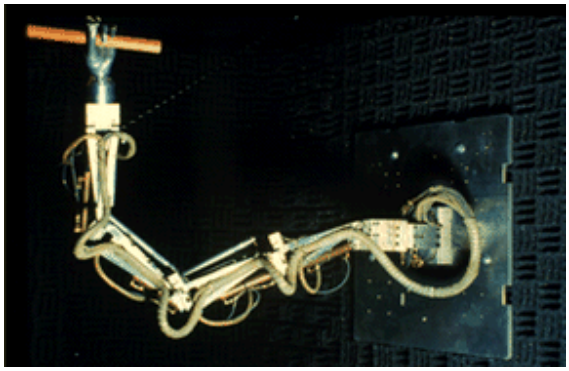
ε2.26

Επισης το 1966 δημιουργείτε το ELIZA πρόγραμμα τεχνητής νοημοσύνης στο MIT από τον Joseph Weizenbaum που λειτουργεί ως υπολογιστής ψυχολόγος

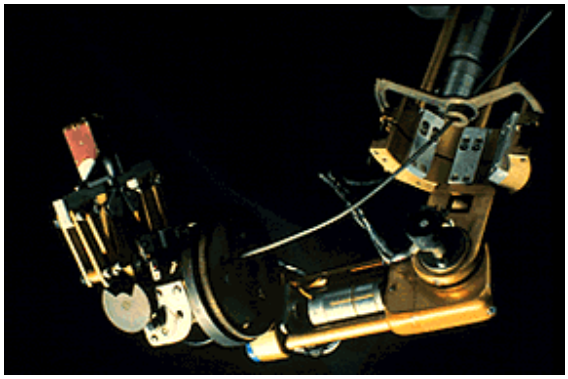
Το 1967 ο Richard Greenblatt γράφει το MacHack , ένα πρόγραμμα που παίζει σκάκι που εχει σχέση με την τεχνητή νοημοσύνη που είναι προγενέστερη προσπάθεια απο τον γνωστό Big Blue που ειχε αντίπαλο τον Gary Kasparov.

Το 1968 η εταιρια Kawasaki φτιάχνει ένα robot με υδραυλικά όπου προχωράει και στην παραγωγή του.

Το 1969 απο τον Marvin Minsky φτιάχνεται ένα άλλο ρομποτ-βραχίονας με 12 αρθρώσεις , με υδραυλικά υγρά , πολύ ευέλικτο και μπορει σηκώσει βάρος ανθρώπου.



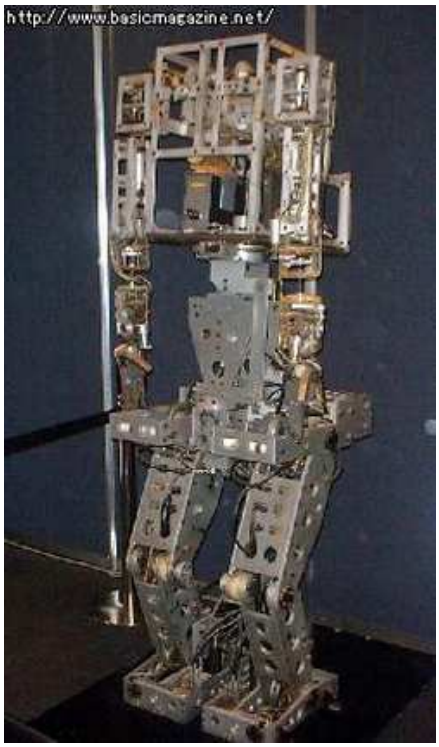
ε2.27



ε2.28

Το 1970 στο πανεπιστήμιο του Stanford φτιάχνεται ενα καλάθι το οποίο ακολουθεί ευθεία γραμμή και μπορεί να ελέγχεται από ηλεκτρονικό υπολογιστή μέσω ραδιοζεύξης.

Το 1973 στο πανεπιστήμιο του Waseda στην Ιαπωνία φτιάχτηκε το πρώτο ανθρωπόμορφο ρομπότ Wabot-1, το οποίο είχε σύστημα ελέγχου βηματισμού, Όρασης και ομιλίας.



ε2.29

το 1974 ο Victor Scheinman δημιουργεί έναν βραχίονα που ειναι ικανός για τη συναρμολόγηση μικρών κομματιών με αισθητήρες αφής.

Το 1976 ο Shigeo Hirose φτιαχνει μία ευλύγιστη αρπάγη που μπορεί να τυλιχτεί γύρω από ένα αντικείμενο.

Το 1979 αναβαθμίστηκε το project με το καλάθι του πανεπιστημίου του Στάντφορντ απο τον Hans Moravec και μπόρεσαν να γίνουν τα πρώτα πειράματα με χαρτογράφηση του τρισδιάστατου περιβάλλοντος.

Το 1981 ο Takeo Kanade δημιουργεί την άμεση κίνηση του βραχίονα , δηλαδή τοποθετήθηκαν κινητήρες απευθείας στις αρθρώσεις και είχε σαν αποτέλεσμα πιο γρήγορη και πολύ πιο ακριβή κίνηση απο τους άλλους βραχίονες.

Το 1989 στο MIT παρουσιάστηκε το Genghis ένα ρομπότ που βαδίζει.

Το 1992 ο Marc Thorpe κατασκεύασε μια ηλεκτρική σκούπα που ελέγχεται απο ραδιοκύματα.

Το 1992 ο Dr. John Adler χρησιμοποίησε ένα ρομπότ το οποίο έβλεπε με ακτίνες χ εναν ασθενή και αναλόγος αν έβλεπε κάποιο όγκο , μετέφερε μία υπολογισμένη δόση στο συγκεκριμένο σημείο.

Το 1993 δημιουργήθηκε ένα οχτάποδο ρομποτ ονομαζόμενο Dante και η αποστολή του είναι να συλλέγει δεδομένα από ένα σκληρό περιβάλλον παρόμοιο με αυτό που θα μπορούσαμε να βρούμε σε ένα κρατήρα ή άλλο πλανήτη.

ε2.30

Το 1996 οι Chris Campbell και Stuart Wilkinson έφτιαξαν ένα ρομπότ που χωνεύει τη βιολογική μάζα για να παράγουν διοξείδιο του άνθρακα, και στη συνέχεια χρησιμοποιείται για αυτοτροφοδότηση.

Το 1996 η εταιρία Honda φτιάχνει το μοντελο P3, που ηταν ο καρπός της δεκαετίας της πολυετούς προσπάθειας για την οικοδόμηση ενος ανθρωποειδές ρομπότ.

ε2.31

Το 1997 η διαστημική αποστολή Pathfinder προσεδαφίζει στον πλανήτη Άρη το ρομποτικό σύστημα Sojourner , το ποιο κυλάει και συλλέγει δεδομένα από την επιφάνεια.

ε2.32

Το 1998 η εταιρια Tiger Electronics παράγει το ρομποτ-παιχνιδι τον Furby ο οποίος έχει διάφορους αισθητήρες και μπορεί να αναπαράγει πολλές φράσεις αναλόγος τις καταστάσεις που βρίσκεται σαν άτομο.

ε2.33

Το 1999 η εταιρία SONY δημιουργεί το παιχνίδι-ρομπότ AIBO έκδοση 1, όπου μπορεί να αντιγράφει τους ήχους και έχει προγραμματισμένη συμπεριφορά.

Το 2000 η εταιρία Honda δημιουργεί το ανθρωποειδές ρομπότ ASIMO , πασίγνωστο για την ευελιξία του.

το 2001 η εταιρία LEGO δημιουργεί το ρομπότ MINDSTORMS.

ε2.34

Το 2001 η καναδική εταιρία MD robotics, φτιάχνει ενα ρομποτικό τμήμα, το οποίο μπαίνει σε τροχιά στο διάστημα και μπορεί να συναρμολογήσει τμήματα του διεθνή διαστημικού σταθμού.

ε2.35

Το 2003 η γνωστή εταιρία SONY , δημιουργεί την τρίτη γενια του παιχνιδιού-ρομπότ AIBO ERS-7, το οποιο κάνει πολυ καλή συντροφιά.

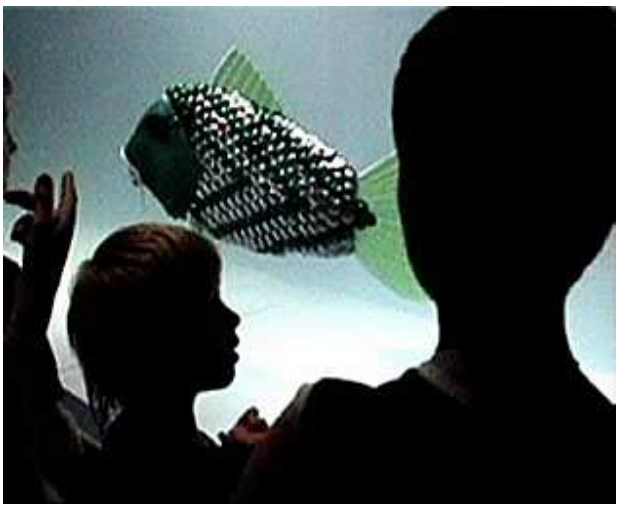
ε2.36

Το 2004 το ρομπότ rover Spirit προσγειώνεται στον Άρη, για εξερεύνηση του εδάφους και συλλογή πληροφοριών.

ε2.37

\

Το 2005 , απο το πανεπιστήμιο του Essex ο καθηγητής Hu και η ομάδα του έφτιαξαν ένα ψάρι για την έκθεση ψαριών στο Λονδίνο , το οποίο μιμείτε την κινηση των ψαριών και κινείτε αυτόνομα μέσα στο ειδικό γυάλινο δοχείο.



ε2.38

Απο εδώ και πέρα υπάρχουν αμέτρητες ειδικευμένες κατασκευές απο διάφορα ινστιτούτα και κατασκευαστικές εταιρίες , τα οποία βλεπουμε καθημερινά να αποτελούν βασικές λύσεις στην ανθρωπινη χρήση και για την εξέλιξη της ζωής του ανθρώπου.

Η μεγαλύτερες δημιουργίες χρησιμοποιούνται ήδη και είναι η ρομποτική συνδυασμένη με την απομακρυσμένη επικοινωνία δηλαδή επικοινωνία με κάποια σήματα ή με το ποιο απλό μέσο, την κινητή τηλεφωνία.

3. Γενικά για Ρομποτικό ποδόσφαιρο

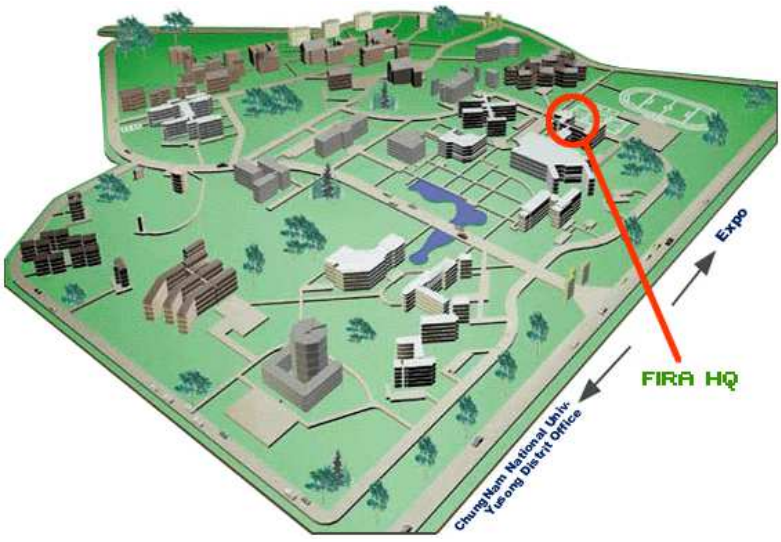
3.1. Οργανισμός ρομποτικου ποδοσφαίρου και η δομή του

Το ρομποτικό ποδόσφαιρο ξεκίνησε το 1995 από τον Jong-Hwan Kim , καθηγητής του πανεπιστημίου KAIST στην N. Κορέα, και το πρώτο πρωτάθλημα μεταξύ ομάδων έγινε στη Dageon της N.Κορέας το 1996.

Ο οργανισμος , που έχει δημιουργήσει αυτή την ροποτική κοινότητα και ορίζει τους κανονισμούς λειτουργίας του συγκεκριμένου θεσμού , ονομάζεται Fira (federation of international robotic-soccer association) και ουσιαστικά ιδρύθηκε από τον Jong-Hwan Kim το 1997. Εχει σαν έδρα κτηριακές υποδομές του πανεπιστιμίου KAIST στην N. Κορέα και πρόεδρος του οργανισμού, εξακολουθεί να είναι μέχρι σήμερα ο ίδιος ο ιδρυτής.

Η έδρα του οργανισμού βρίσκεται στην παρακάτω διεύθυνση καθώς και μια μακέτα της κτηριακής υποδομής του οργανισμού FIRA από μία τρισδιάσταση απεικόνηση (ε3.1) ανάμεσα στα κτηριακά συγκρότηματα του πανεπιστημίου KAIST(Korea Advanced Institute of Science and Technology).

#2348 Undergraduate Building 2, KAIST, 373-1 Guseong-dong, Yuseong-gu, Daejeon 305-701, Korea , Tel: +82-42-869-8871~9 Fax: +82-42-869-8870
E-mail: master@fira.net , fira@fira.net



ε3.1

Είναι η 19η χρονιά από την ίδρυση της ομοσπονδίας και έχουν γίνει συνεχείς αλλαγές στις διοργανώσεις. Το 2014 η διοργάνωση θα γινει στην Ν. Κορέα.

Η διοργάνωση έχει σαν σκοπό να μεταφέρει το πνεύμα της τεχνολογίας και της επιστήμης στους νέους, να κινήσει το ενδιαφέρον σε άσχετους με την τεχνολογια, διαμέσου ενός παιχνιδιού ποδοσφαίρου, να προωθήσει και να συνεισφέρει στην ανάπτυξη των αυτόνομων robot, να φέρει σε επικοινωνία ερευνητές και σπουδαστές που έχουν ειδικότητα σε διάφορες κατηγορίες όπως ρομποτική , τεχνολογία αισθητήρων, τεχνική νοημοσύνη , επικοινωνίες , επεξεργασία εικόνας, μηχατρονική, επιστήμη Η/Υ και να διοργανώνει το ρομποτικό παγκόσμιο πρωτάθλημα κάθε χρόνο και για διασκέδαση.

Το ρομποτικό ποδόσφαιρο , καθώς και άλλων παρεμφερεί τοπικών διαγωνισμών , απεικονίζει τον διαγωνισμό της εξέλιξης της ρομποτικής τεχνολογίας σε ένα περιορισμένο χώρο , δίδοντας την ευκαιρία στην νέα γενιά και στους ερευνητές να δουλέψουν σε αυτόνομα φορητά ρομποτικά συστήματα.

Συγχρόνως βοηθάει στην ανάπτυξη των γνώσεων , καθώς μέσα από αυτούς τούς ρομποτικούς διαγωνισμούς, τροφοδοτείτε ο επιστημονικός τομέας , ο οποίος με την σειρά του επιχειρεί για περαιτέρω εξέλιξη ώστε αυτές οι εφαρμογές να μπορούν να προσφέρουν με άλλους διαφορετικους τρόπους στην τεχνολογία και συνεπώς στην κοινωνία.

Το παγκόσμιο προτάθλημα ρομποτικού ποδοσφαίρου (FIRA Robot World Cup) έχει συμμετέχοντες από τα πανεπιστήμια με ειδικότητες σχετικές με ρομποτική, αλλα υπάρχει και η ρομποτική ολυμπιάδα Robot Olympiad , όπου μπορεί να λάβει μέρος ο οποισδήποτε.

Παράλληλα με τους διαγωνισμούς ποδοσφαίρου γίνονται και επιπλέων διαγωνισμοί (Hurocup), είναι διαγωνισμοί άλλων ειδών αναμετρήσεων όπως :
Σήκωμα φορτίου , αποφυγή εμποδίου , αρπαγή και μεταφορά , εκτέλεση πέναλτη , διαγωνισμός μπάσκετ , αναρρίχηση σε τοίχο , επιτάχυνση , διαγωνισμός αντοχής , κτλ.

Μετά από τόσους συνεχείς επιτυχημένους διαγωνισμούς , η διοργάνωση της FIRA έχει μία παγκόσμια αναγνώριση σαν ρομποτικό γεγονός.

Οι διαγωνισμοί είναι κατηγοριοποιημένοι , σε συγκεκριμένες αυτόνομες ομάδες , σε εθνικό ή και σε παγκόσμιο επίπεδο.

Ο οργανισμός είναι σε δύο μεγάλους τομείς χωρισμένος , τον Ευρωπαικό τομέα (European FIRA Chapter) και τον Ασιατικό τομέα(Asian FIRA Chapter).

Ο Ευρωπαικός τομέας , αποτελείτε από ολόκληρη την Ευρωπαική Ένωση καθώς και την Βουλγαρία , Τσεχία , Ουγγαρία , Πολωνία , Ρουμανία , Σλοβακία και Σλοβενία.

Εχει σαν σκοπό την χρήση των ρομπότ σαν απασχόληση στην ελεύθερη ώρα ή ώς χόμπυ και εστιάζεται γύρω από τους παρακατω θεματικούς κλάδους: Τα αυτόνομα φορητά ρομποτ (MAS) , την συνεργασία , την χάραξη δρομολογίου σε ένα άγνωστο χώρο , τον έλεγχο συντονισμού , την υιοθεσία και εκμάθηση , την χρησιμοποίηση ελλειπή και αντιφατικών πληροφοριών , τους έξυπνους αισθητήρες , την συνεργασία αισθητήρων , την εκπαίδευση με άποψη , την αλλαγή ρόλων σε μία ομάδα.

Ο Ασιατικός τομέας αποτελείτε από την Κίνα , Ινδία , Ιαπωνία, Ινδονησία , Κορέα , Φιλλιπίνες , Συγκαπούρη , Ταιβάν , Ην. Αρ. Εμιράτα , Σα. Αραβία , Ιράν , Μαλαισία , Ταιλάνδη και άλλα ασιατικά κράτη. Εχει σαν σκοπό την χρήση των ρομπότ σαν απασχόληση στην ελεύθερη ώρα ή ώς χόμπυ καθώς επίσης την πραγματοποίηση της επιπλέων ανάπτυξης της ρομποτικής τεχνολογίας. Εστιάζεται γύρω από τους παρακατω κλάδους: τα αυτονομα φορητά ρομποτ (MAS) , την συνεργασία , την χάραξη δρομολογιου σε ένα άγνωστο χώρο , τον έλεγχο συντονισμου , την υιοθεσία και εκμάθηση , την χρησιμοποιηση ελλειπή και αντιφατικών πληροφοριών , τους έξυπνους αισθητήρες , την συνεργασία αισθητήρων , την εκπαίδευση με άποψη , την αλλαγη ρόλων σε μία ομάδα , τα πολυρομποτικά συστήματα , τα κατασκοπικά συστήματα , την τεχνιτή νοημοσύνη , την μηχανική όραση και ρομποτική σκέψη , την μικρο και νανορομποτική.

Και στους δύο τομείς διοργανόνωνται συναντήσεις, ώστε να ελέγχουν το επίπεδο ανάπτυκης που βρίσκονται και να οργανώνουν τα μελλοντικά τους σχέδια.

3.2 Κατηγορίες παγκοσμίου κυπέλλου ρομποτικού ποδοσφαίρου

Υπάρχουν πολλά διαφορετικά πρωταθλήματα τα οποία διαφέρουν στον τύπο των ρομποτικών παικτών (ανδροειδές ή τροχήλατο), στον αριθμό τους, στο μέγεθος του γηπέδου, στους κανονισμούς, και στον τρόπο ελέγχου (αυτόνομα ή με χειροκίνητα).

Οι κατηγορίες Παγκόσμιου κυπέλλου ποδοσφραίρου ρομπότ (FIRA) που υπήρξαν και οι νέες που υπάρχουν μέχρι σήμερα, παρουσιάζονται πάρακάτω αναλυτικά στον πίνακα που ακολουθεί :

α/α	Κωδική Ονομασία	Πλήρη Ονομασία	Χαρακτηριστικά
1	HuroSot	Humanoid Robot World Cup Soccer Tournament	Ανδροιδές Ρομποτ. Πρέπει να έχουν μέγιστο ύψος 40 εκατοστών, 2 πόδια , διάμετρος μεταξύ των ποδίων 15 εκατοστών Ο αγώνας έχει δύο ομάδες με 3 ρομποτ / ομαδα , το ένα είναι τερματοφύλακας. Ο διαγωνισμός αποτελείτε από τρία άτομα στο γήπεδο (προπονητης , μαναγκερ , διαιτητής) . Αλλά οι κανόνισμοί παιχνιδιού δεν είναι έποιμοι ακόμη. Πρόκειτε για τη μελλοντική νέα διοργάνωση.
2	S-HuroSot	Single Humanoid Robot World Cup Soccer Tournament	Παρουσιάζει τα ιδια χαρακτηριστικά με το Hurosot , αλλα η κάθε ομάδα έχει ένα ρομπότ
3	RoboSot	Robot World Cup Soccer Tournament	Είναι ρομπότ επιπέδου 20x20 εκατοστών χωρίς οριο ύψους, χωρίς πόδια αλλά με τροχούς. Ο αγώνας έχει δύο ομάδες με 3 ρομπότ/ομάδα , το ένα είναι τερματοφύλακας. Τρία άτομα στο γήπεδο (προπονητης , μαναγκερ , διαιτητής) ,
4	S-RoboSot	Single Robot World Cup Soccer Tournament	Παρουσιάζει τα ίδια χαρακτηριστικά με το RoboSot, αλλά η κάθε ομάδα έχει ένα ρομπότ
5	MiroSot	Micro Robot World Cup Soccer Tournament	Το μέγεθος των ρομποό πρέπει να είναι 7.5x7.5x7.5 εκκατοστά , χωρίς να μετριέται το μέγεθος της κεραίας και κινούνται με τροχούς. Η μπάλλα πρέπει να είναι πορτοκαλί χρώμα. Ο αγώνας έχει δύο ομάδες με 3 ρομπότ/ομάδα , το ένα είναι τερματοφύλακας. Τρία άτομα στο γήπεδο (προπονητής , μανάγκερ , διαιτητής) .

			Ενας Η/Υ χρησιμοποιείται ανα ομάδα για την οπτική παρακολούθηση και την ανγνωρηση της θέσης τους Η εργασία γίνεται για τη συγκεκριμένη κατηγορία διοργάνωσης.
6	S-MiroSot	Single Micro Robot World Cup Soccer Tournament	Παρουσιάζει τα ίδια χαρακτηριστικά με το MiroSot, αλλα η κάθε ομάδα έχει ένα ρομπότ
7	NaroSot	Nano Robot World Cup Soccer Tournament	Το μέγεθος των ρομπότ πρέπει να είναι μέγιστο 3.75x3.75x3.75 εκατοστά και κινούνται με τροχούς. Ενας Η/Υ χρησιμοποιείται ανα ομάδα για την οπτική παρακολούθηση και την αναγνωρηση της θέσης τους
8	S-NaroSot	Single Nano Robot World Cup Soccer Tournament	Παρουσιάζει τα ίδια χαρακτηριστικά με το NaroSot, αλλα η κάθε ομάδα έχει ένα ρομπότ
9	KheperaSot	Khepera Robot World Cup Soccer Tournament	Ο αγώνας έχει δύο ομάδες με 3 ρομπότ/ομάδα , το ένα είναι τερματοφύλακας. Τρία άτομα στο γήπεδο (προπονητής , μάναγκερ , διαιτητής). Ενας Η/Υ χρησιμοποιείται ανα ομάδα για την οπτική παρακολούθηση και την ανγνωρηση της θέσης τους. Εκτος από το μέγεθος του ρομποτ και του γηπέδου οι κανόνες παιχνιδιου είναι παρόμοιοι με του τύπου MiroSot
10	Simurosot	Simuted World cup Robotic Soccer	Οι ομάδες δεν εχουν παίκτες ρομποτ , αλλα έχουν φτιάξει το παιχνίδι διεξάγεται σε Η/Υ με προσομοίωση παικτών
11	S-KheperaSot	Single Khepera Robot World Cup Soccer Tournament	Παρουσιάζει ίδια χαρακτηριστικά με το KheperaSot, αλλα η κάθε ομάδα έχει ένα ρομπότ

Να σημειωθεί ότι ο αριθμός των ρομποτικών παικτών μπορεί να είναι 3 , 5 και 11 ανάλογα την διοργάνωση και έχουν αντιστοιχες ονομασιες small league , medium league , large league και ανάλογα μεγάλο μέγεθος γηπέδου.

3.3 Αναλυτική κατάσταση παγκοσμίων διοργανώσεων έως σήμερα

Μία κατάσταση όλων των παγκόσμιων διοργανώσεων που έχουν γίνει μέχρι σήμερα παρουσιάζεται παρακάτω ταξινομημένες σύμφωνα με την ημερομηνία διεξαγωγής τους:

α/α	Ημερ/νιες διεξαγωγής	Χωρα διαγωνισμού / υπεύθυνος διοργάνωσης	Τύποι διαγωνισμού	Ομάδα- ν ικητής ανα τύπο
1	9–12 /Νοε / 1996	Korea / KAIST	MiroSot	Newton Research Labs. Seattle (U.S.A.)
2	2-5 /Ιουν / 1997	Korea / KAIST	MiroSot	Newton Research Labs. Seattle (U.S.A.)
3	1-3 /Ιουλ /1998	France / La cite des Sciences et de I'Industrie, Paris, France	MiroSot S-MiroSot NaroSot S-KheperaSot	The Keys, Human Interface Inc. (Korea) The Keys, Human Interface Inc. (Korea) MIRO III KAIST, (Korea) STATIC Univ. of Aarhus, (Denmark)
4	4-8 /Αυγ /1999	Brazil / Colegio Notre Dame School, Campinas	MiroSot S-MiroSot NaroSot S-KheperaSot	RobotIS, RobotIS Co., Ltd. (Korea) New NEU (China) RobotIS, RobotIS Co., Ltd. (Korea) -
5	18-24 / Σεπ /2000	Australia / Rockhampton, Gladstone, and Queensland /	MiroSot S-MiroSot	Robo-Coaster (Australia) Bandicoots (Australia)
6	01-03 / Αυγ /2001	Beijing / The Science and Technology Museum	MiroSot (3/3) MiroSot (5/5) SimuroSot(5/ 5) SimuroSot(11/11) RoboSot Tele-MiroSot	MASKARO, POSTECH, (Korea) HIT, (China) Guangdong Univ. of Technology, (China) Haier Group, (China) Shanghai Jiao Tong Univ., (China) SongLei High School, (China)
7	23-29 / Μια /2002	Korea / Federation of International Robot-soccer Association (FIRA)	HuroSot KheperaSot MiroSot(5/5) NaroSot RoboSot SimuroSot	HanSaRam-II, KAIST, (Korea) Danish Dynamite,Southern Univ.(Denmark) KingGo, SungKyunKwan Univ., (Korea) Y2K2, KAIST, (Korea) Teams of HIT, Harbin Inst.of Tech., (China) FORMOSA, Nat'l Taiwan Univ., (Taiwan)
8	28/Οκτ-03/Νοε /2003	Vienna Austria/Handling Devices and Robotics (IHRT) and Vienna University of	HuroSot / - KheperaSot / MiroSot (7/7)- NaroSot / - RoboSot /	RoboSapien (Singapore) Kheperoo (Australia & Thailand) Kinggo (Korea) RIT 1 (Korea) TKU_FORERUNNER (Chines Taipei)

		Technology (VUT).	SimuroSot(11/11)	WICT (China)
9	27-31 / Οκτ /2004	(Korea) Robot Soccer Association(KRSA), Busan National University and Intelligent Robot Research Center (KAIST ITRC-IRRC).	HuroSot KheperaSot MiroSot(11/11) NaroSot RoboSot SimuroSot(11/11)	Hei-Long (China) Kheperoo (Australia) Sparic (Singapore) Austro (Austria) Hei-Long (China) WIT (China)
10	11-16/ Δεκ /2005	(Singapore)	HuroSot KheperaSot MiroSot(11/11) NaroSot RoboSot SimuroSot(11/11)	manus (Singapore) Jumbo (Australia) Socrates (Singapore) AUSTRO (Austria) HIT (China) WIT (China)
11	30/06-03/07 / 2006	Dortmund, Germany	MiroSot (11/11) MiroSot (5/5) NaroSot RoboSot HuroSot KheperaSot	Socrates (Singapore) Socrates (Singapore) AUSTRO (Austria) TKU-MIRDC (Taiwan) manus (Singapore) HNI Devils (Germany)
12	14-17 /Ιουν /2007	San Francisco, USA	HuroCupMarathon AndroSot KheperaSot MiroSot (11/11) RoboSot SimuroSot(11/11)	Hansaram VII (Korea) WIT (China) SCT-Scooter1 (Germany) Socrates (Singapore) TKU (Taiwan) WUST (China)
13	22-25 /Ιουλ /2008	:Qingdao , (China)	Χωρίς πληροφορίες	Χωρίς πληροφορίες
14	18-20 /Αυγ /2009	Incheon / (Korea)	Χωρίς πληροφορίες	Χωρίς πληροφορίες
15	15-19 /Σεπ /2010	Bangalore / (India)	MiroSot(11/11) AndroSot SimuroSot(11/11)	SjF TUKE, (Slovakia) RIT, S. (Korea) CLIMBER, (China)
16	26-30 /Αυγ /2011	Kaohsiung , (Taiwan)	Χωρίς πληροφορίες	Χωρίς πληροφορίες
17	20-24 /Αυγ /2012	Bristol / (UK)	AndroSot MiroSot (11/11) RoboSot SimuroSot(11/11)	SIOR (Korea) EDRAGON (China) TK(Taiwan) NPU SOARU(China)
18	24-29 /Αυγ /2013	Shah Alam / Malaisia	Χωρίς πληροφορίες	Χωρίς πληροφορίες

3.4 Σύμβολα έναρξης ορισμένων διοργανώσεων

Ακολουθούν παρακάτω ορισμένα σύμβολα των διοργανώσεων που έχουν διεξαχθεί σε διαφορες ημερομηνίες ανα τον κόσμο:

α/α	Χώρα διεξαγωγής	Σύμβολο διοργάνωσης
1	Ν. Κορέα - 1996	

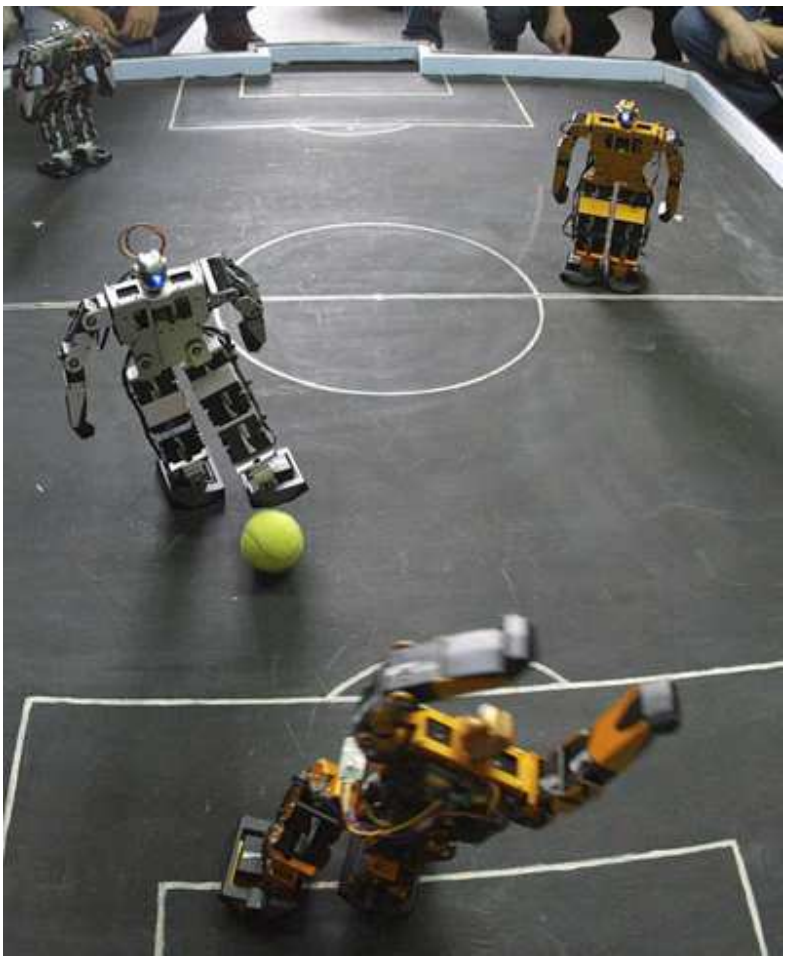
2	Γαλλία , 1998	
3	Βραζιλία , 1999	

4	Αυστραλία , 2000	
5	Ν. Κορέα , 2002	

6	Αυστρία , 2003	
7	N. Κορέα , 2004	

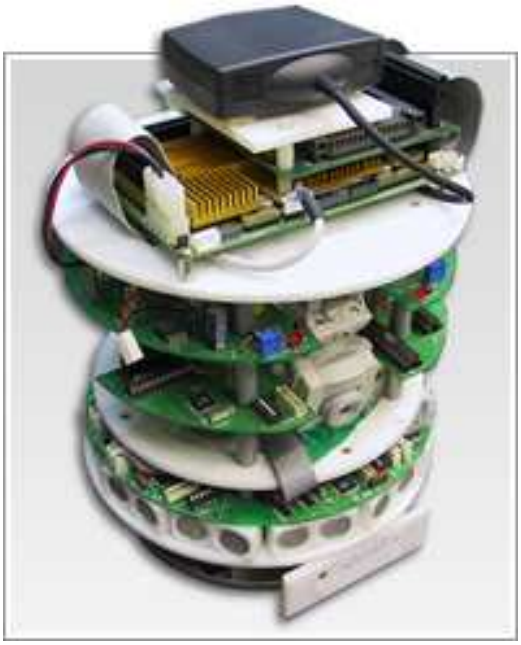
3.4 Παραδείγματα ρομποτικών παικτών ανα κατηγορία της διοργάνωσης

Hurosot , S-Hurosot



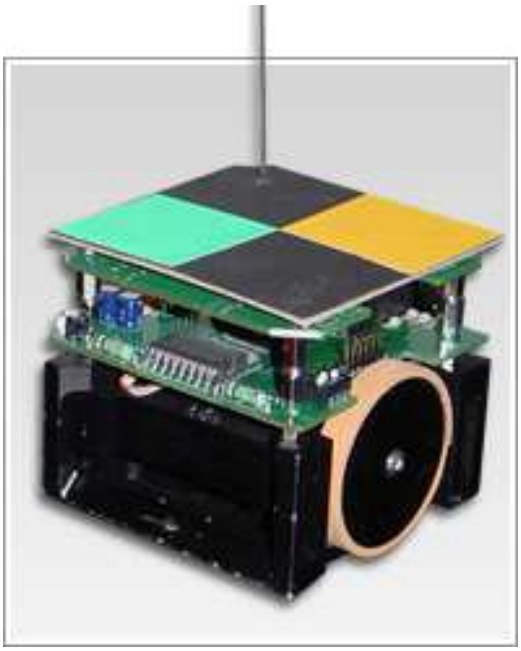
(ε3.2)

Robosot , S-RoboSot



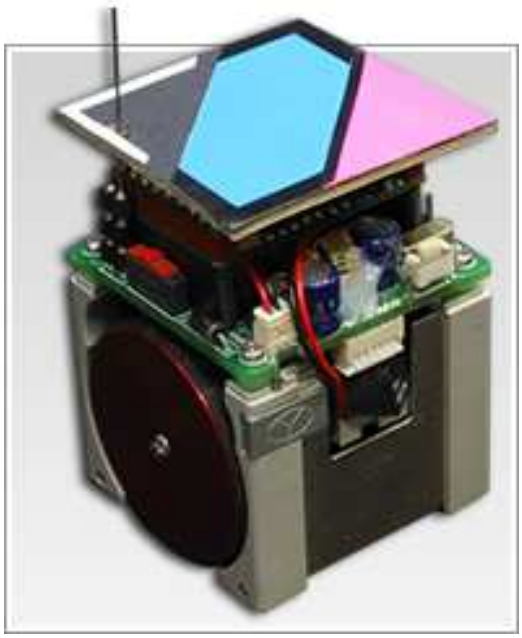
(ε3.3)

Mirosot , S-MiroSot



(ε3.4)

Narosot , NaroSot

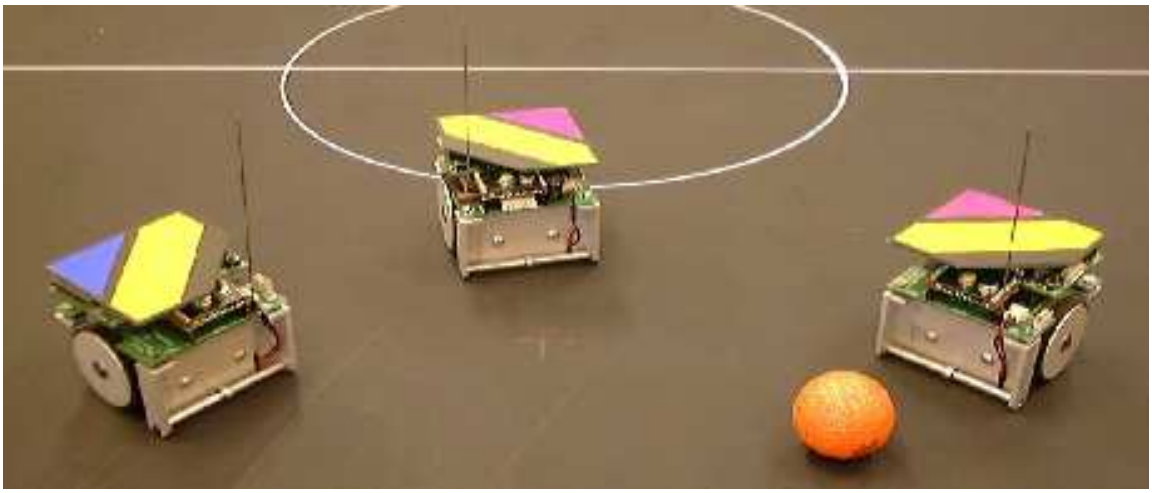


(ε3.5)

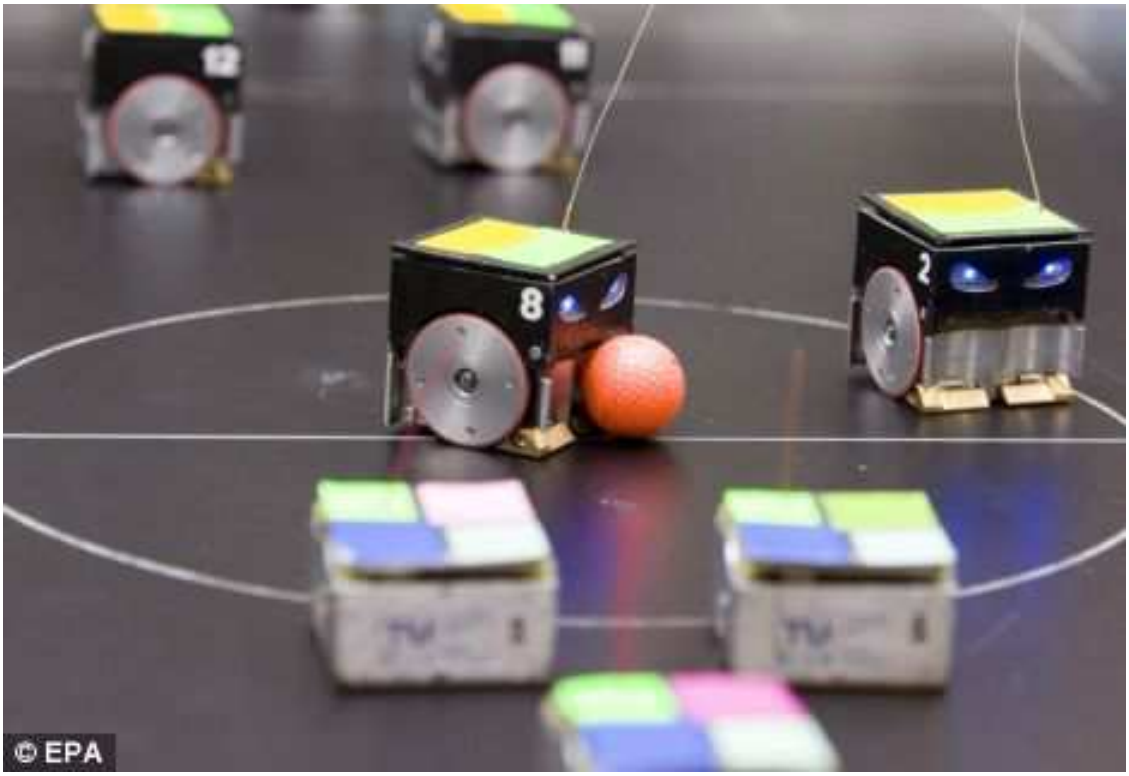
Simurosot

(ε3.6)

3.5 Στιγμιότηπα διαφορετικών διοργανώσεων

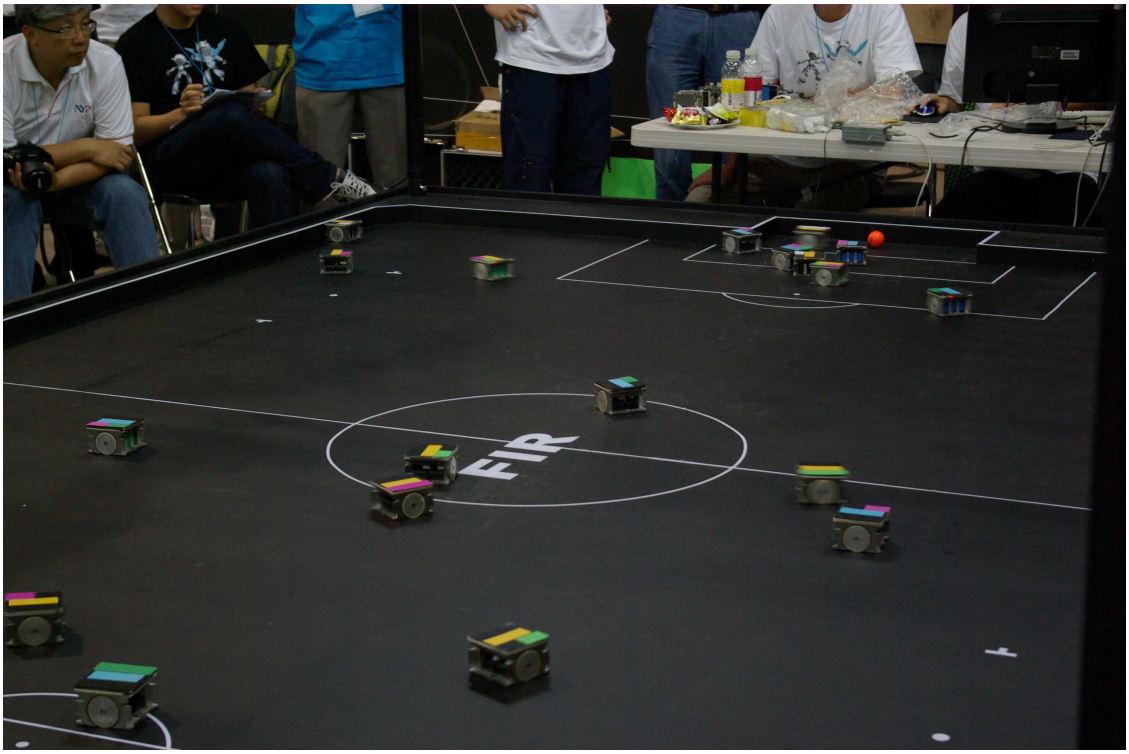


(ε3.7)



(ε3.8)

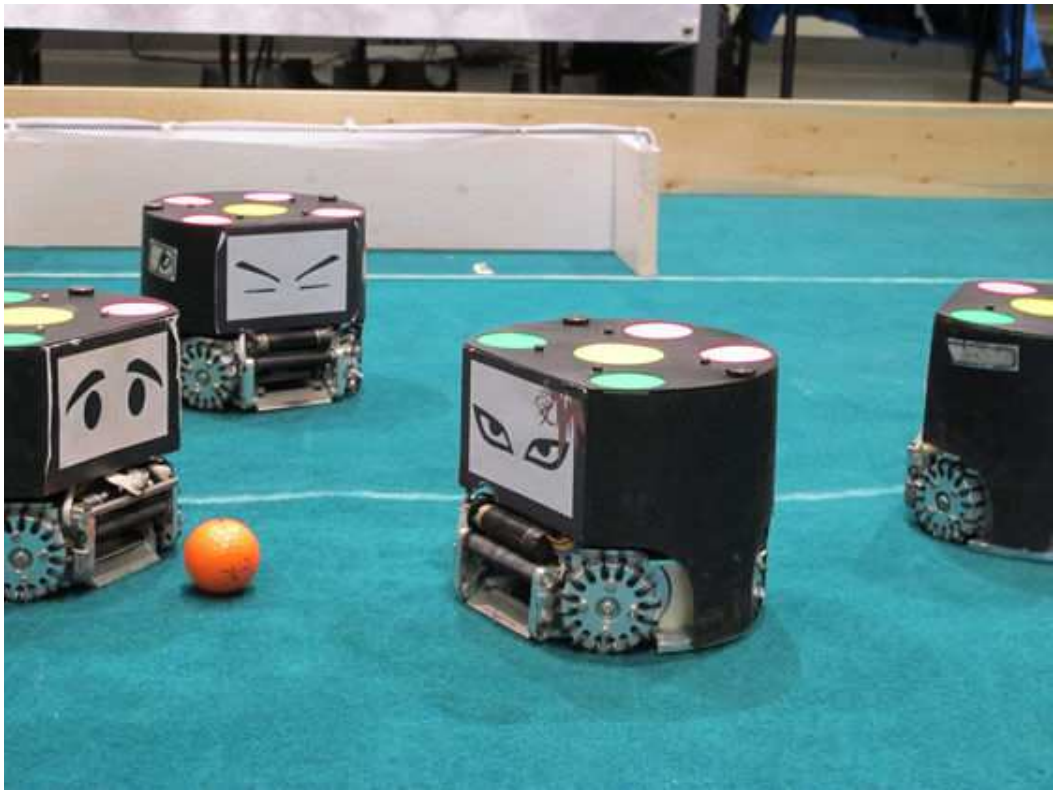
(ε3.9)



(ε3.10)

(ε3.11)

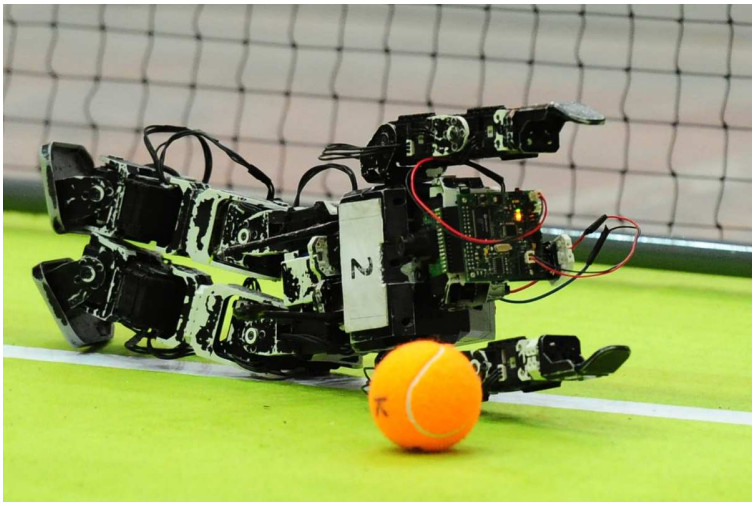
(ε3.12)



(ε3.13)

(ε3.14)

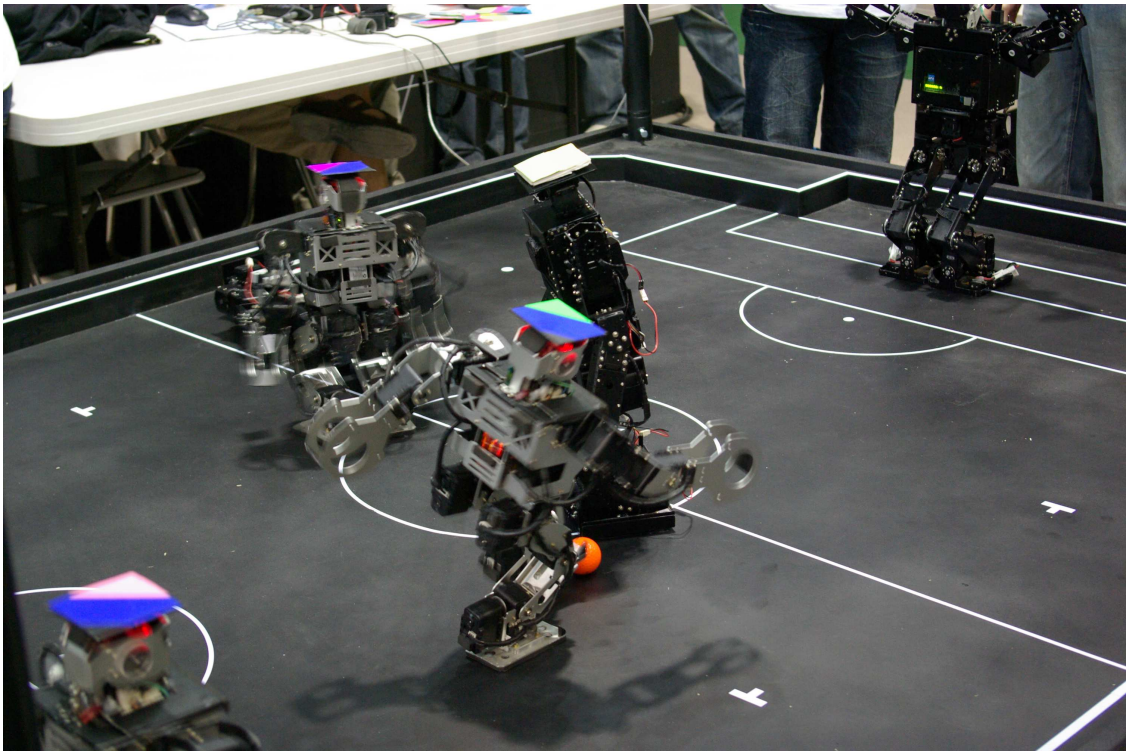
(ε3.15)



(ε3.16)

(ε3.17)

(ε3.18)



(ε3.19)

(ε3.20)

(ε3.21)

(ε3.22)

(ε3.23)

4. παρουσιάση – ανάλυση της λειτουργίας του ρομποτικού ποδοσφαίρου στην συγκεκριμένη εργασία.

4.1 Γενικά

Η συγκεκριμένη εργασία έχει δημιουργηθεί για να προσεγγίσει την κατηγορία πρωτάθληματος με το όνομα “MiroSot” με το μικρό μέγεθος γηπέδου (MiroSot small league).

Το συγκεκριμένο πρωτάθλημα χωρίζεται σε τρεις διαφορετικές κατηγορίες παιχνιδιού οι οποίες φαίνονται παρακάτω:

α/α	Όνομα κατηγορίας	Διαστάσεις γηπέδου	Αριθμός παικτών robots
1.	MiroSot large league	400cm x 280cm	11
2.	MiroSot middle league	220cm x 180cm	5
3.	MiroSot small league	150cm x 130cm	3

Η εργασία έχει σκοπό να λειτουργήσει σε συνθήκες της διοργάνωσης του MiroSot small league , η οποία χρησιμοποιεί το μικρότερο μέγεθος γηπέδου ώστε να είναι δυνατή και η προσομείωση της σε ένα δωμάτιο σπιτιού. Σε περίπτωση που θελήσουμε να λειτουργήσουμε σε μεγαλύτερη κατηγορία , ο βασικός κορμός του κώδικα και η αρχή λειτουργίας όλως των ελέγχων θα έιναι ίδιος απλά θα χρειαστούν ορισμένες μικρές προσαρμογές , για παράδειγμα θα έχουμε περισσότερους παίκτες στην στρατηγική να υπολογίζουμε και να κινήσουμε.

4.2 Επεξήγηση τμημάτων, περιγραφή συστήματος, και τρόπος λειτουργίας στο Matlab.

4.2.1 Κατάσταση αναλυτικών χαρακτηριστικών

Παρακάτω απεικονίζονται όλα τα τεχνικά χαρακτηριστικά της λειτουργίας σύμφωνα με τις προδιαγραφές της FIRA και η στήλη περιγραφής της πραγματικής λειτουργίας της συγκεκριμένης κατηγορίας πρωταθλήματος όπου χρησιμοποιείτε τύπος ρομποτικών παικτών Mirosot.

α/α	Χαρακτηριστικό	Προδιαγραφές Mirosot Small League	Πραγματική λειτουργία
		ΤΜΗΜΑ ROBOT	
1	Μέγιστο μέγεθος robot	7,5cm x 7,5cm x 7,5cm	8,5cm x 8,5cm x 9,5cm
2	Μέγεθος ένδυσης robot	8cm x 8cm x 8cm	9cm x 8,5cm x 10cm
3	Ακτινα τροχου	43.5 cm	43.5 cm
4	Αρθμός τροχών	2	2
5	Αριθμος robot	3	1
6	Βαρος robot	-	Αρκετά βαρύ
7	Τροφοδοσία robot	9v DC	9v DC
8	Τυπος τροφ/σιας	1 μπαταρία 9v	2 παρ/λες μπαταριες 9v
9	Επικοινωνία με Host PC	Ασύρματη σειριακή θύρα	Ασύρματη σειριακή θύρα
		με πομπό / δέκτη	με πομπό / δέκτη
10	Επεξεργαστης Robot	Atmel atmega8535 16Bit	Atmel atmega8535 16Bit
		ΤΜΗΜΑ ΜΠΑΛΑΣ	
1	διάμετρος μπάλας	42.7 mm	38 mm
2	Χρώμα μπάλας	Πορτοκαλί	Πορτοκαλί
		ΤΜΗΜΑ ΓΗΠΕΔΟΥ	
1	Μεγεθος γηπέδου	150cm x 130 cm	150cm x 130 cm
2	Ακτίνα κέντρου γηπέδου	200mm	200mm
3	Πάχος γραμμών	3mm	Όχι γραμμες
4	Πλάτος τέρματος	400mm	400mm
5	Χρώμα γηπέδου	Μαύρο	Μαύρο
		ΤΜΗΜΑ ΚΑΜΕΡΑΣ	
1	Ύψος κάμερας	Ελάχιστο 200 εκατοστά	210 εκατοστά
2	Frames/sec	-	21-31
3	Ανάλυση frame	-	640 x 480 pixel – Μεταβαλόμενο
		ΤΜΗΜΑ ΦΩΤΙΣΜΟΥ	
1	Φωτισμός	1000 lux ομοιογενής	~ 1000 lux μερικός ομοιογενής

		ΤΜΗΜΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ	
1	Περιβάλλον ελέγχου εφαρμογής	-	Matlab (ver 7-11)
2	Περιβάλλον εκτέλεσης κίνησης robot	-	AVR C
3	Βοηθητικό μέσω δειγματοληψίας	-	Βιβλιοθήκη για matlab vcapg2.dll της εταιρίας Gerox
4	Η/Υ λειτουργίας	-	Intel Core i7 4930, 4.7 Ghz (o.c)
		ΑΝΘΡΩΠΙΝΟ ΔΥΝΑΜΙΚΟ	
1	Μάνατζερ	1	
2	Διαιτητής	1	
3	Προπονητής	1	1

4.2.2 Γενική απεικόνηση εγκατάστασης και περιληπτική λειτουργία

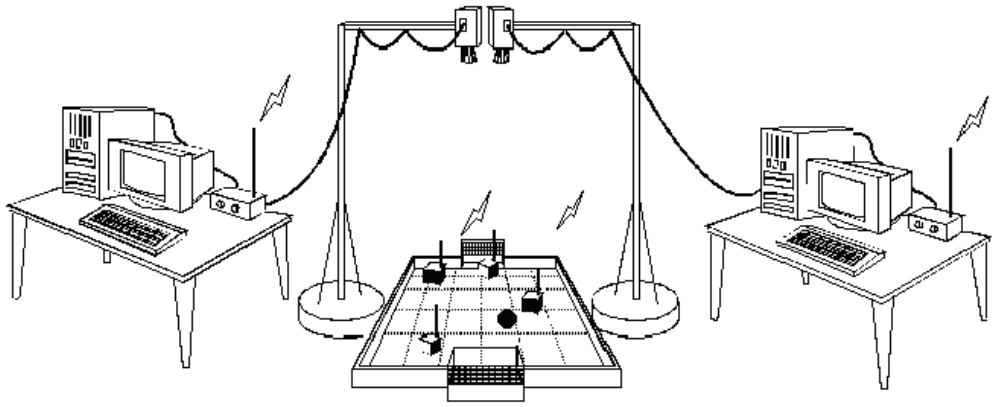
Το γενικό σενάριο της συγκεκριμένης εργασίας αποτελείτε από ένα αυτόνομο σύστημα ελέγχου και κίνησης .Ένας ρομποτκό παίκτης ακολουθεί την μπάλα προσπαθώντας να την μεταφέρει σε επιτρεπόμενη απόσταση και κλίση βολής προς το αντίπαλο τέρμα. Η εντολή που έχει ο ρομποτκό παίκτης, είναι να στείλει την μπάλλα προς το τέρμα. Δεν είναι σκοπός της εργασίας η λειτουργία και των υπόλοιπων ρομποτκών παίκτών ή η αποφυγή αντικειμένων ή η συνεργασία με υπόλοιπους παίκτες, προς το παρόν.

Ο κώδικας δεν έχει δοκιμαστεί για πλήρες παιχνίδι με αντιπάλους, αλλά περιλαμβάνει τις απαραίτητες λειτουργίες ώστε να μπορέσουν να ενσωματοθούν και οι υπόλοιποι ρομπότικοί παίκτες, με ορισμένες μικρές παρεμβάσεις στον κώδικα.

Η συγκεκριμένη εργασίας αποτελείτε από ένα σύνολο εργασιών , όπως η αναγνώριση αντικειμένων , διαχωρισμός φιλικών και αντίπαλων ρομποτικών παικτών , υπολογισμό αποστάσεων μεταξύ τους, υπολογισμό κλίσεων των αντικειμένων, υπολογισμό κινήσεων, αποστολή εντολών και φιλτράρισμα εισερχόμενων εντολών, αναγνώρηση διαφόρων καταστάσεων αντικειμένων (κίνηση , ακινησία) και δημιουργία στρατηγικής ανάλογα με την κατάσταση , θέση ή ανάλογα με τον ρόλο του κάθε ρομποτικού παίκτη .

4.2.2.1 Επεξήγηση χαρακτηριστικών της Διοργάνωσης MiroSot small league

Στο παρακάτω σχεδιάγραμμα (ε4.1) , απεικονίζεται η τυπική εγκατάσταση που απαιτείται για αυτο τον τύπο της διοργάνωσης .



(ε4.1)

Βλέπουμε στην απεικόνηση , ο κάθε παίχτης έχει μία κάμερα αναρτημένη πάνω από το γήπεδο , έναν Η/Υ και ένα ασύρματο σειριακό πομπό/δέκτη . Επίσης υπάρχει ένα ξύλινο δάπεδο , το γήπεδο ανάλογα με την διοργάνωση καθώς και τα αντίστοιχα φωτιστικά σώματα αναρτημένα περίπου στο ύψος που βρίσκονται οι κάμερες με συγκεκριμένη συνδυασμένη φωτεινότητα περίπου όση ορίζουν οι προδιαγραφές.

Η κάμερα χρησιμοποιείται για την δειγματοληψία της εικόνας , που στην ουσία αποτελούν τα μάτια των παικτών ρόμποτ. Οσο μεγαλύτερη είναι η ταχύτητα ανανέωσης της εικόνας από την κάμερα , τόσο καλύτερη είναι η απόκριση από το ολόκληρο το σύστημα.

Ενας Η/Υ ανα κάθε ομάδα , αναλαμβάνει να παραλαβει τις εικόνες για επεξεργασία , ώστε αυτός με την σειρά του να μπορέσει από την κινούμενη εικόνα να δημουργήσει τις πληροφορίες του περιβάλλοντος που χρειάζονται ,για τις παρακάτω λειτουργίες που εκτελούνται συνεχώς μέχρι να κλείσει το σύστημα :

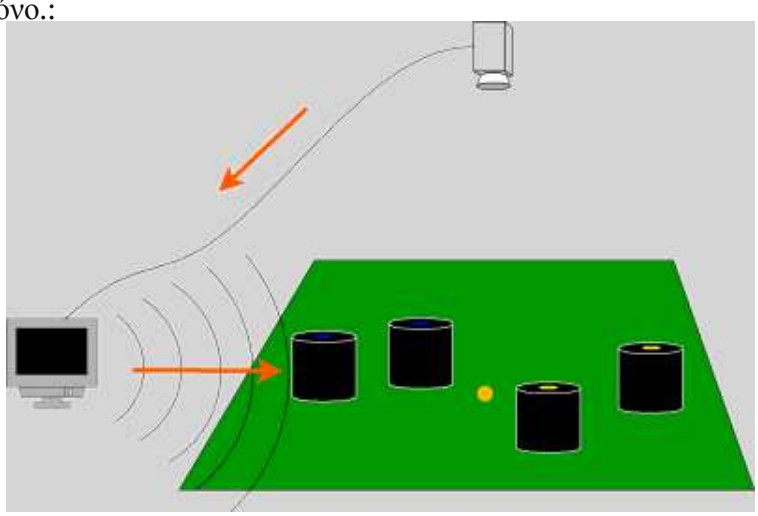
- Την εύρεση τις γεωγραφική θέσης του γηπέδου σε σχέση με το οπτικό παράθυρο της κάμερας (μόνο στην πρώτη εκτέλεση)
- την έυρεση των φιλικών και εχθρικών παικτών ρομπότ καθώς και το πώς φαίνονται τα διαχωριστικά διαγνωστικά χαρακτηριστικά τους ως προς το χρώμα και ως προς το μέγεθος,
- την θέση της μπάλλας , καθώς και το πώς φαίνεται στην εικόνα ως προς το χρώμα και ως προς το μέγεθος,

- να υπολογίσει την συνδιασμένη φωτεινότητα – ευκρίνεια (Threshold) που χρειάζεται ώστε τα αντίστοιχα διαχωριστικά διαγνωστικά τμήματα των φιλικών παικτών ρόμποτ να μπορούν να διαγνωσθούν σε όλα τα σημεία του γηπέδου,
- να μετατρέψει τις αντιστοιχες θέσεις όλως των αντικειμένων σε μαθηματικές τιμές
- να υπολογίσει από τις αντίστοιχες τιμές τις αποστάσεις π.χ μεταξύ μπάλλας και παικτών ρόμποτ ή αντίπαλου τέρματος ή μεταξύ των αντίπαλων ρόμποτ
- την γεωγραφική θέση του γηπέδου σε σχέση με το οπτικό παράθυρο της κάμερας
- να επιλέξει την σωστή στρατηγική για την αντίστοιχη συγκεκριμένη κατάσταση
- να αποστείλει τις εντολές κίνησης μέσω της ασύρματης σειριακής επικοινωνίας
- και να ελέγξει εάν η εντολή εκτελέσθηκε ώστε να υπάρξει η αντίστοιχη ανάδραση στην επόμενη αποστολή εντολών.

Ο σειριακός πομπός / δέκτης είναι συνδεδεμένος στον H/Y , πανω σε μία σειριακή θύρα , έχει ρυθμιστεί με ίδια χαρακτηριστικά όπως επίσης και στον κάθε ρομποτικό παίκτη , να λειτουργεί με συγκεκιμένη ταχύτητα .Αποτελεί το μέσο για την εκτέλεση της ανάδρασης του σύστηματος στο συστήμα κίνησης. Η κίνηση επιτυχγάνεται με συνδιασμένη απόστολή bytes προς τον κάθε ρομποτικό παίκτη , ο οποίος έχει ένα αντιστοιχο ασύρματο σειριακό δέκτη πάνω του. Όταν παραλειφθουν τα σωστά bytes , τοτε γίνεται η μετάφραση τους από το σύστημα κίνησης του κάθε ρομποτικού παίκτη και εκτελείτε η κίνηση.

4.2.2.2 Φορά μετάδοσης πληροφορίας

Ο επιτρεπόμενος τρόπος φοράς της πληροφορίας μπορεί να φανεί στο παρακάτω σχήμα. Η φορά της πληροφορίας δεν επιτρέπεται να είναι αμφίδρομη , αλλα προς μία κατεύθυνση μόνο.:



(ε4.2)

Όπως παρατηρουμε στο σκίτσο (ε4.2) , η φορά της πληροφορίας είναι από την κάμερα προς τον Η/Υ , μετά από τον Η/Υ προς τον κάθε ρομποτικό παίκτη.

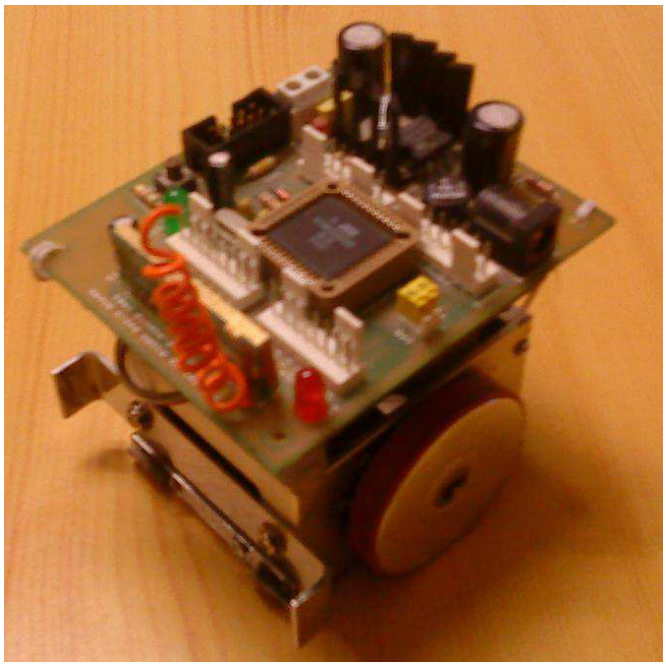
4.2.2.3 Περιβάλλον ανάπτυξης

Το περιβάλλον ανάπτυξης του κώδικα και ελέγχου της λειτουργίας όλου του συστηματος είναι το μαθηματικό πακέτο matlab, το οποίο εκτός από τις πολλαπλές δυνατότητες που έχει για μαθηματικές εφαρμογές, όπως επεξεργασία πινακων , ενσωματώνει και λειτουργίες οπτικής αναγνώρισης (vision) , ώστε με δυνατότητα ορισμένων παραμετροποιήσεων, χρησιμοποιούνται στο κώδικα επιτυχώς, με αποτέλεσμα να μην χρειάζεται να αναπτυχθεί επιπλέων κώδικας.

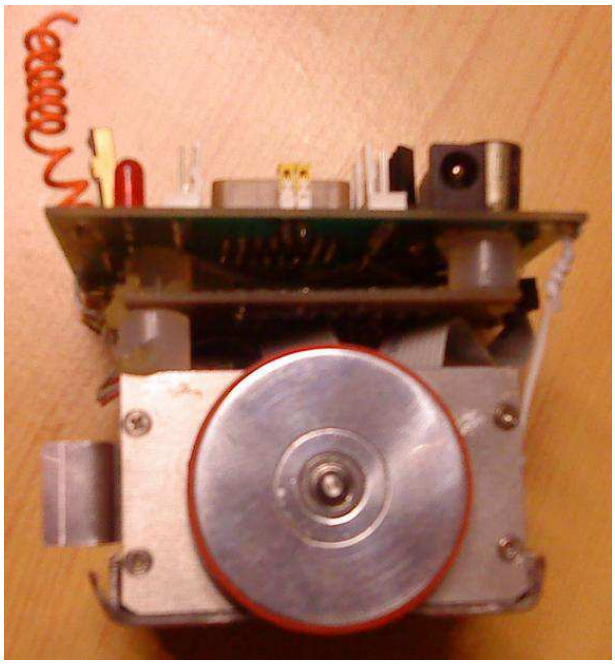
Ο κώδικας στο μικροελεγκτή είναι γραμμένος με το περοβάλλον ανάπτυξης AVR C.

4.2.2.4 Πραγματικές εικόνες δοκιμών της εφαρμογής

Ακολουθούν εικόνες (ε4.3 έως ε4.8) από τον ρομποτικο παίκτη

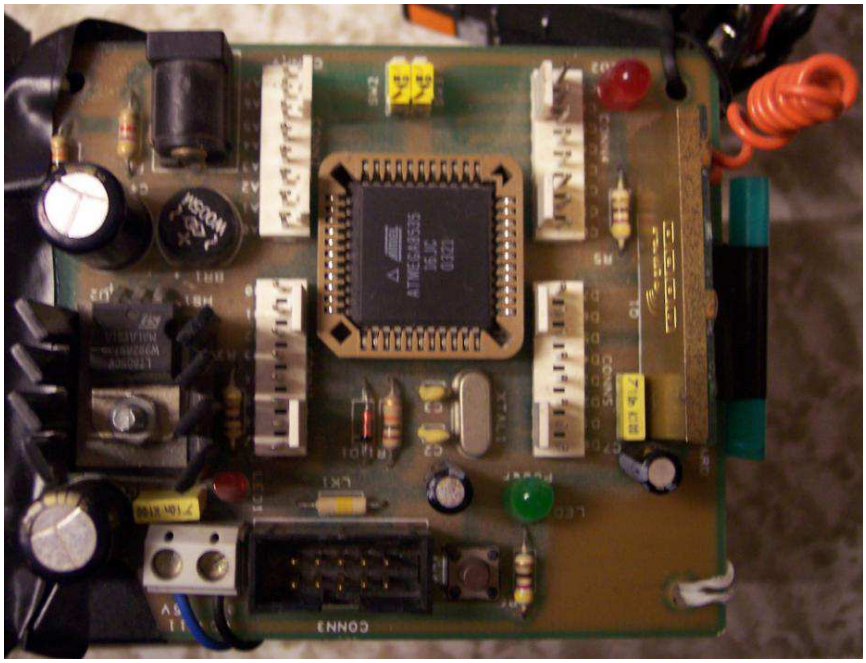


(ε4.3)



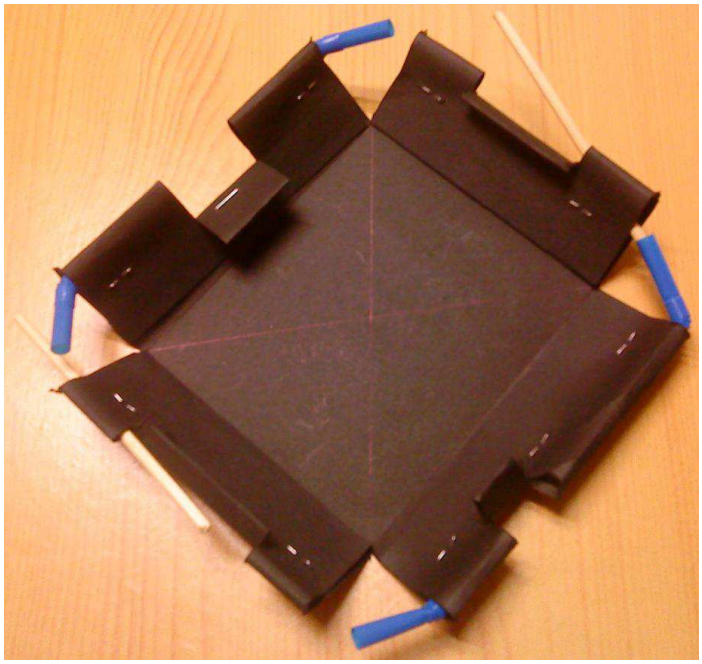
(ε4.4)

Ο ρομποτικός παίκτης χωρίς μπαταρίες, και κάλυμα.



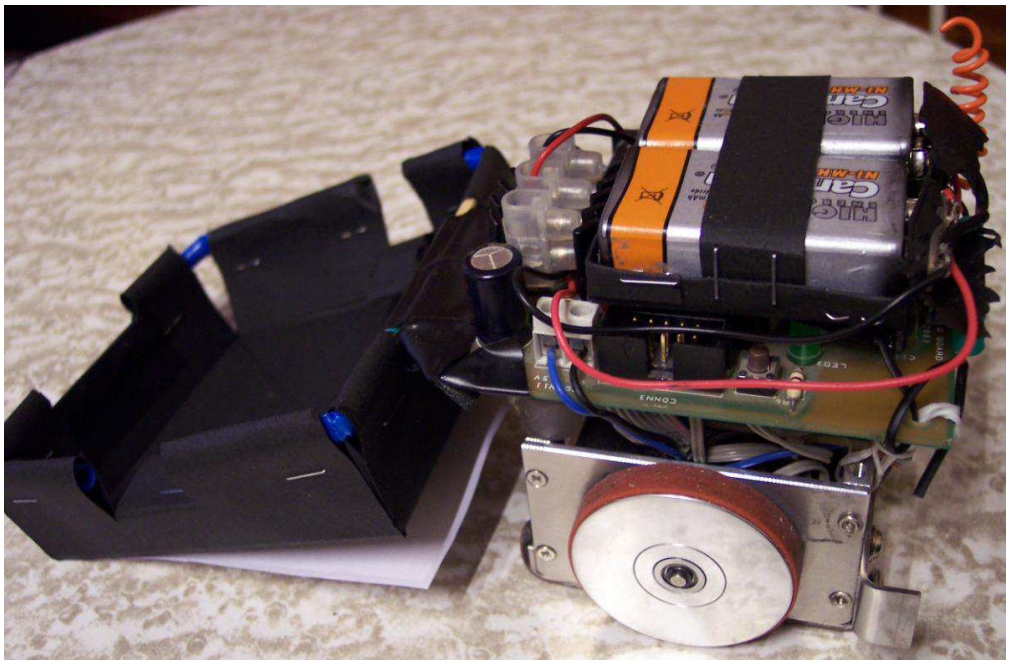
(ε4.5)

Στην κορυφή του ρομποτικού παίκτη, φαίνεται ο μικροελεγκτής , η θύρα επικοινωνίας για τον προγραμματισμό του , η είσοδος της τάσης , ο ασύρματος πομπός/δέκτης και από κάτω όπου δεν φαίνεται υπάρχει η ηλεκ/κή πλακέτα ισχύος των δύο τροχών.



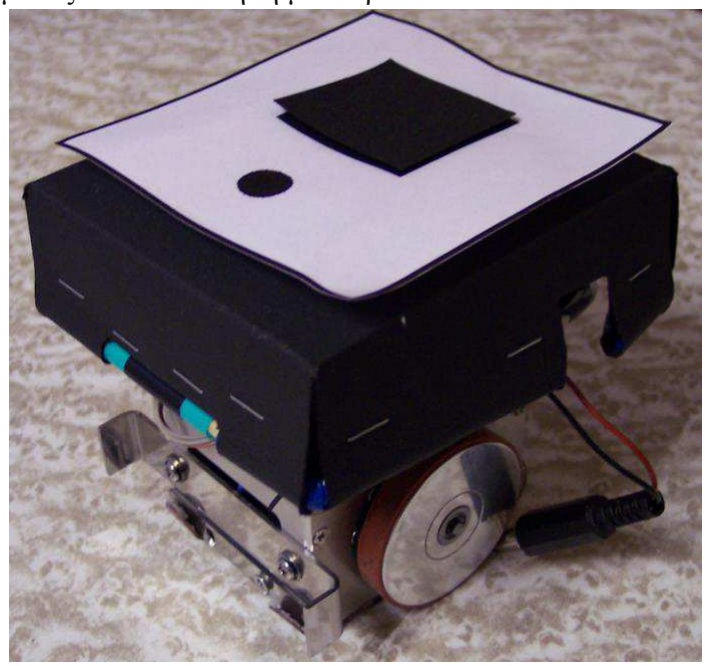
(ε4.6)

Το κάλυμμα του ρομποτικού παίκτη για να τοποθετείτε και να αφαιρείτε εύκολα για να εξυπηρετεί τη συχνή παρέμβαση στο πρόγραμμα ελέγχου του μικροελεκτή και για την προστασία του.



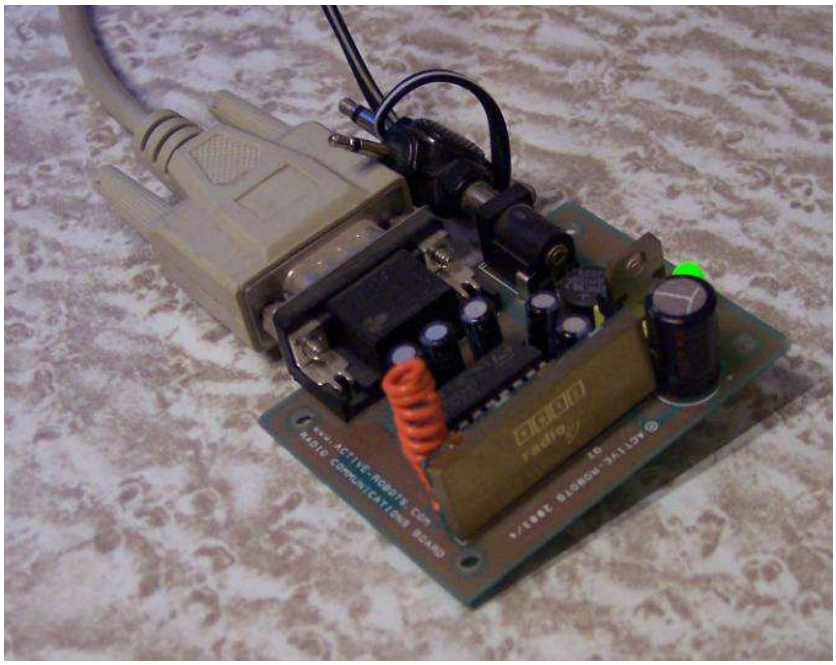
(ε4.7)

Εφαρμογή καλύματος και τοποθέτηση μπαταριών.



(ε4.8)

Τελική μορφή Ρομποτικού παίκτη με την τοποθέτηση του συγκεκριμένου διαχωριστικού αναγνωριστικού καλύματος για το κάθε μέλος του γηπέδου.



(ε4.9)

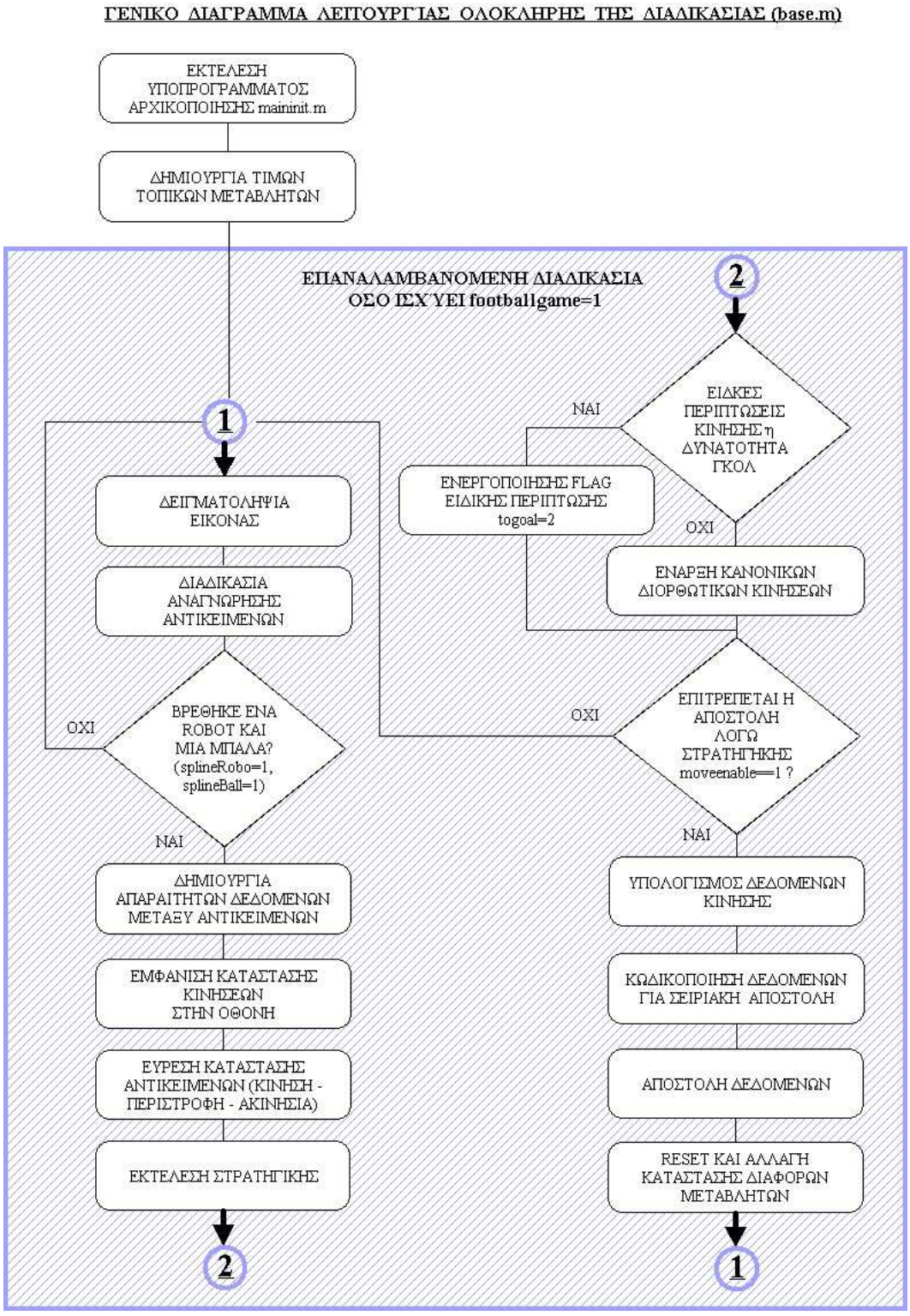
Σειριακός πομπός και δέκτης συνδεδεμένος στον Η/Υ για την αποστολή των δεδομένων στον αντίστοιχο δέκτη του κάθε ρομποτικού παίκτη.

(ε4.6)

Παράδειγμα της δοκιμαστικής εγκατάστασης του φωτισμού , ομοιόμορφα κατά μήκος στο γήπεδο όπως επίσης και η αναρτημένη κάμερα στο κέντρο του γηπέδου

4.2.3 Αναλυτική επεξήγηση λειτουργίας

Το γενικό διάγραμμα της λειτουγίας περιγράφεται παρακάτω.



4.2.3.1 Δημιουργία των απαραίτητων υποδομών την πρώτη φορά για την σωστή λειτουργία

Η διαδικασία αυτή πρέπει να εκτελείτε, έστω για έλεγχο , κάθε πρώτη φορά που γίνεται η εγκατάσταση σε νέο χώρο , ή όταν τοποθετείτε το γήπεδο ή όταν μετακινουνται τα φωτιστικά σώματα ή κάμερα.

Ο έλεγχος περιλαμβάνε τα εξής βήματα :

- Η κάμερα πρέπει να βρίσκεται περίπου σε ύψος 2m ή μεγαλύτερο από την επιφάνεια του γηπέδου,
- Να βρίσκεται στο κέντρο του γήπεδου, ο νοητός άξονας Χ του παραλληλογράμου του γηπέδου, να είναι παράλληλος, με τον άξονα Χ του παραλληλογράμμου του γηπέδου της δειγματοληψίας.
- ο φωτισμός να είναι όσο πιο ομοιόμορφος γίνεται σε όλη την επιφάνεια του γηπεδου
- η κάμερα πρέπει να βρίσκεται όσο περισσοτερο στο κέντρο του άξονα Χ και άξονα Υ του γηπέδου , ώστε να έχει την λιγότερη δυνατη κλιση προς τα άκρα και για να γίνεται ποιο γρήγορα η αναγνωρηση των αντικειμένων και ιδιαίτερα των δικών μας ρομποτικών παικτών
- Ρυθμίζουμε την κάμερα με τις ρυθμίσεις που αναφέρονται στις αναλυτικες τις ρυθμίσεις όπως φάινονται στο 4.2.3.2 τμήμα.

Όταν σταθεροποιηθεί η κάμερα και το γήπεδο έρθει σε σχετικά καλή θέση, το πρώτο βήμα που γίνεται εδώ είναι να τοποθετήθει το ο ρομποτικός παίκτης (χωρίς τάση) και η μπάλλα , περίφερειακα του κέντρου του γηπέδου και όχι στα άκρα για λογους ρυθμίσεων. Το κέντρο του γηπέδου , επίσης χρησιμοποιείτε για να γινει ένας δειγματοληπτικος έλεγχος του χρώματος του γηπέδου και να διαβαστουν οι R,G,B τιμές του , που χρησιμοποιουνται αργοτερα στον κωδικα.

- Εκτελούμε το πρόγραμμα “Maininit”, κάθε φορά από το Matlab για να πάρουμε μία ή περισσότερες δειγματοληψίες από την κάμερα τοποθετόντας τον ρομποτικό παίκτη περιφερειακά του κέντρου και περιφερειακά στις 4 γωνίες του γηπέδου , και καταγράφουμε τις τιμές Thresdhold που μας επιστρέφει το πρόγραμμα στην οΘόνη.

ώστε με την συγκεκριμένη φωτεινότητα να μας δώσει τα threshold που χρειαζόμαστε.

Εάν ρυθμίσουμε την κάμερα και τοποθετήσουμε τον φωτισμό όπως πρέπει , οι τιμές του Threshold που εμφανιστούν για ττον ρομποτικό παίκτη που τοποθετούμε , πρέπει να είναι από 0,2 < Threshold < 0,4.

Σε περίπτωση που έχουμε διαφορετικές τιμές , τότε δεν είναι κατι ρυθμισμένο σωστά από θέμα κάμερας η φωτεινότητας η ευθυγράμμισης. Το πρόγραμμα έχει την δυνατότητα να αλλάζει αυτόματα το threshold δυναμικά κατά την λειτουργία του

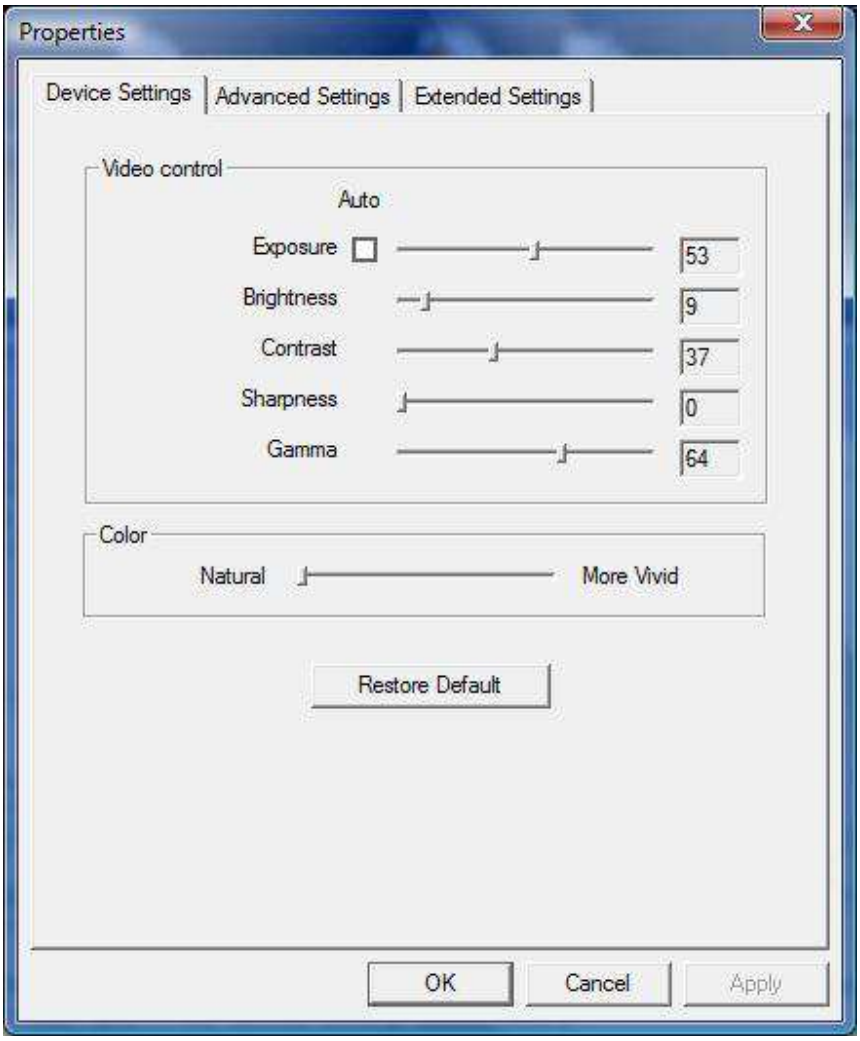
προγράμματος , απλά πρέπει να ορίσουμε τα όρια ξανά , εάν δεν είναι ανάμεσα στις τιμές που αναφερα παραπάνω.

Αυτό αργότερα θα βοηθήσει ώστε να εχουμε με μεγαλύτερη ακρίβεια , για τις υπάρχουσες συνθήκες φωτισμού, τις λιγότερες επιθυμητές αλοιώσεις της εικόνας που παρουσιάζονται όσο περισσότερο απέχουμε από το κέντρο του γηπέδου (ακτινικά) όπως για παράδειγμα ελλειπτική μορφή εικόνας προς τα μέσα η προς τα έξω .

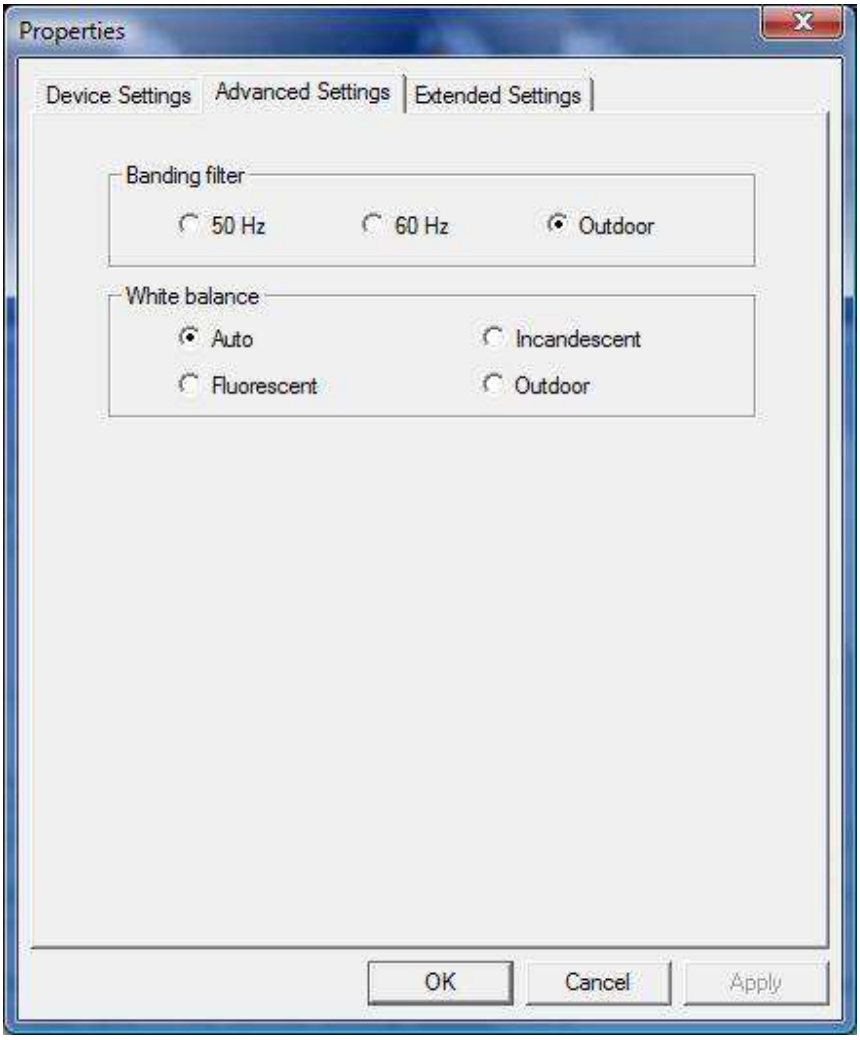
Μετά τα παραπάνω βήματα, εαν ορισμένες τιμές threshold είναι εντάξει ή είναι ρυθμισμένες αναλογα, τοτε μπορούμε να τροφοδήσουμε με τάση (μπαταρία) τον ρομποτικό παίκτη και τον πομπό της ασύρματης επικοινωνίας. Στην συνέχεια μπορούμε να εκκινήσουμε το βασικό πρόγραμμα “Base”.

4.2.3.2 Ρυθμίσεις κάμερας

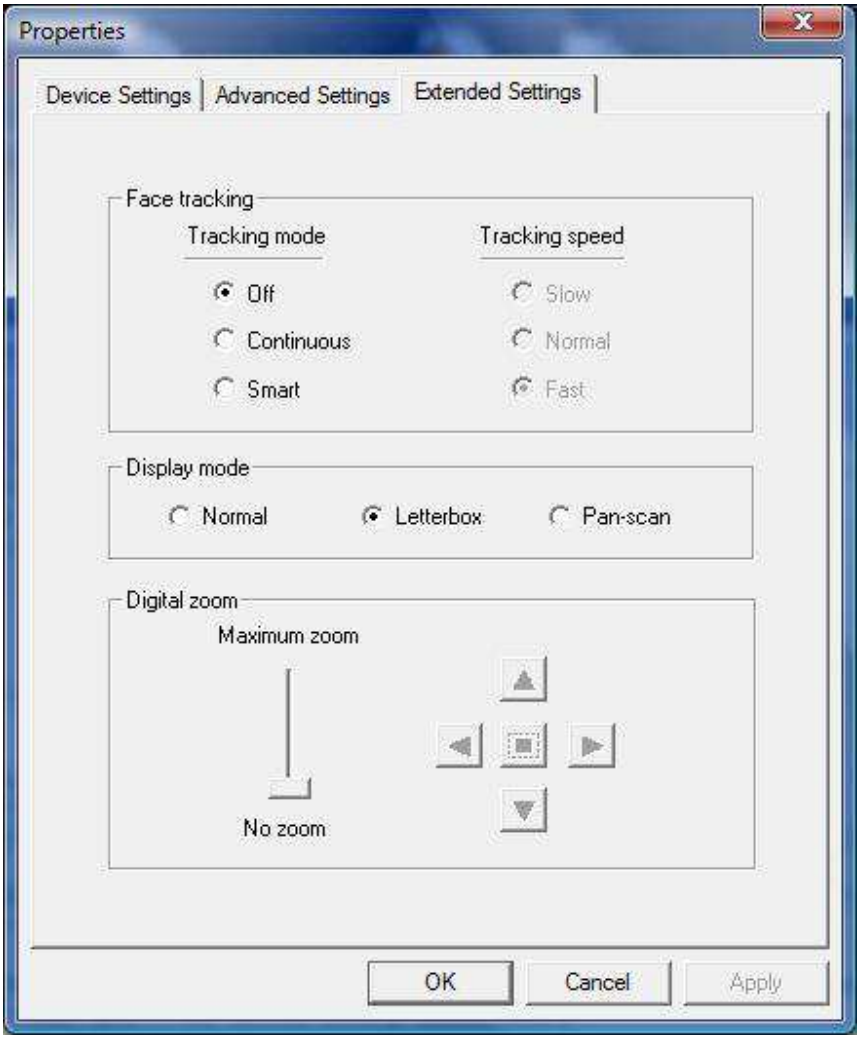
Οι ρυθίσεις της συγκεκριμένης κάμερας αναλυτικά βρίσκονται παρακάτω (ε4.10 έως ε4.13) , όπως δίδονται από τον κατασκευαστή. Οι ρυθμίσεις είναι ανάλογες της φωτεινοτητας και σε περιπτωση που υπάρχουν πολύ διαφορετικές τιμές threshold κατά την εγκατασταση, συνήθως αυξομειώνουμε το Exposure , Gamma και λιγότερο Sharpness, Brighsness και contrast.



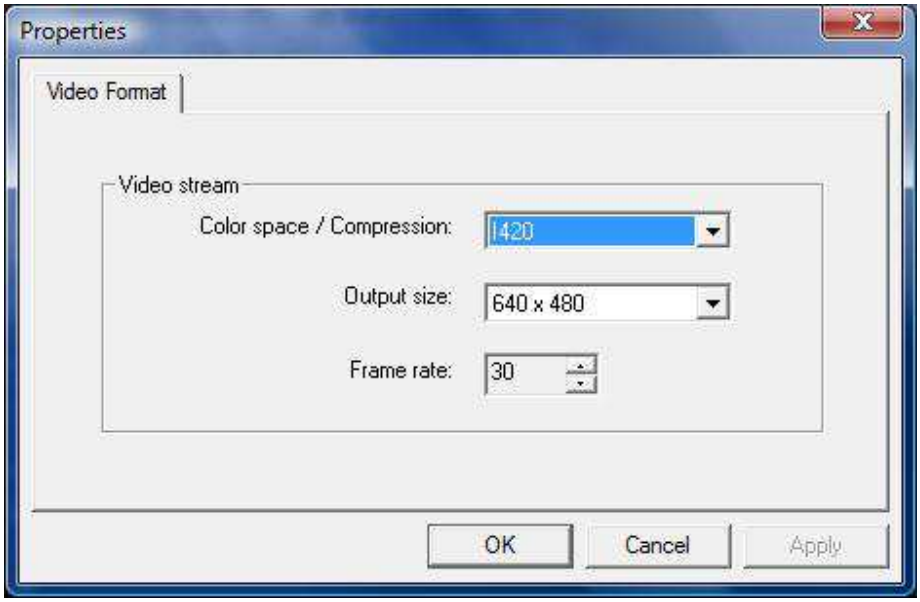
(ε4.10)


Properties
Device Settings
Advanced Settings
Extended Settings
Banding filter
50 Hz
60 Hz
Outdoor
White balance
Auto
Incandescent
Fluorescent
Outdoor
OK
Cancel
Apply

(ε4.11)


Properties
Device Settings
Advanced Settings
Extended Settings
Face tracking
Tracking mode
Off
Continuous
Smart
Tracking speed
Slow
Normal
Fast
Display mode
Normal
Letterbox
Pan-scan
Digital zoom
Maximum zoom
No zoom
OK
Cancel
Apply

(ε4.12)



(ε4.13)

4.2.3.3 Εκκίνηση και τερματισμός λειτουργίας προγράμματος.

Όταν επιτευχθεί η εγκατάσταση στις σωστές θέσεις και συνθήκες, το σωστό φωτισμό, και η σύνδεση των τάσεων στον ρομποτικό παίκτη και στον πομπό, τοποθετείται ο ρομποτικός παίκτης και η μπάλα μέσα στο χώρο του γηπέδου εκτός δίπλα στα άκρα για την εκκινηση .

Η εκκίνηση της λειτουργίας επιτυγχάνεται καλώντας για εκτέλεση το πρόγραμμα base.m από το περιβάλλον του Matlab. Αυτό το πρόγραμμα καλεί με τη σειρά του το υποπρόγραμμα “maininint.m” , το οποίο δημιουργεί ορισμένες τιμές του περιβάλοντος και τις επιστρέφει στο βασικό πρόγραμμα , ώστε η διαδικασία να αρχίσει , χωρίς να χρειάζεται να γίνει κάτι επιπλέων από τον χρήστη.

Ο ρομποτικός παίκτης προσπαθεί να ωθήσει την μπάλλα προς το αντίπαλο τέρμα που έχει οριστεί, το οποίο γίνεται με την δηλωμένη μεταβλητή που βρίσκεται μέσα στο πρόγραμμα base.m, myterm=1 ή 2 , όπου myterm=1 σημαίνει ότι το αριστερό τέρμα όπως εμφανίζεται στην εικόνα είναι το δικό μας ενώ το δεξιό είναι το αντίπαλο.

Η κίννηση σταματάει , όταν για κάποιο λόγο δεν αναγνωρίζεται ο ρομποτικός παίκτης ή η μπάλλα ή όταν έχει επιτευχθεί γκόλ.

Επειδη η λειτουργία του ελέγχου βρίσκεται σε μία μεγάλη χωρίς τέλος επαναληπτική διαδικασία , για να διακοπεί η λειτουργία πρέπει να πατηθεί το ctrl+c μέσα στο παραθυρο εντολών του Matlab , οπου φαινονται οι τιμές.

Για την επανεκκίνηση της διαδικασίας , μπορουμε να ξαναπατήσουμε το F5 , και το πρόγραμμα θα εκκινησει όταν θα βρεί έστω ένα ρομποτικό παίκτη και μία σωστή μπάλλα σε χρώμα και μέγεθος.

4.2.3.4 Αρχικοποίηση τιμών (υποπρόγραμμα maininit.m)

Το υποπρόγραμμα “Maininit.m”εκκινεί αυτόματα , στην αρχή μέσω του βασικού προγράμματος “Base.m” , μόνο για μία φορά , και είναι το τμήμα της αρχικοποίησης τιμών των χαρακτηριστικών. Το λογικό διάγγραμα φαίνεται παρακάτω.

Για να λειτουργήσει , χρειάζεται ένα ρομποτικό παίκτη και μία μπάλλα τουλάχιστον μέσα στο γήπεδο.

Αυτό το τμήμα περιλαμβάνει τις παρακάτω λειτουργίες

- το έλεγχο πόσο ευθύγραμμα είναι το γήπεδο σε σχέση με την κάμερα
  - τον έλεγχο όλων των σωστών Threshold που μπορούν να αναγνωρήσουν το ρομποτικό παίκτη

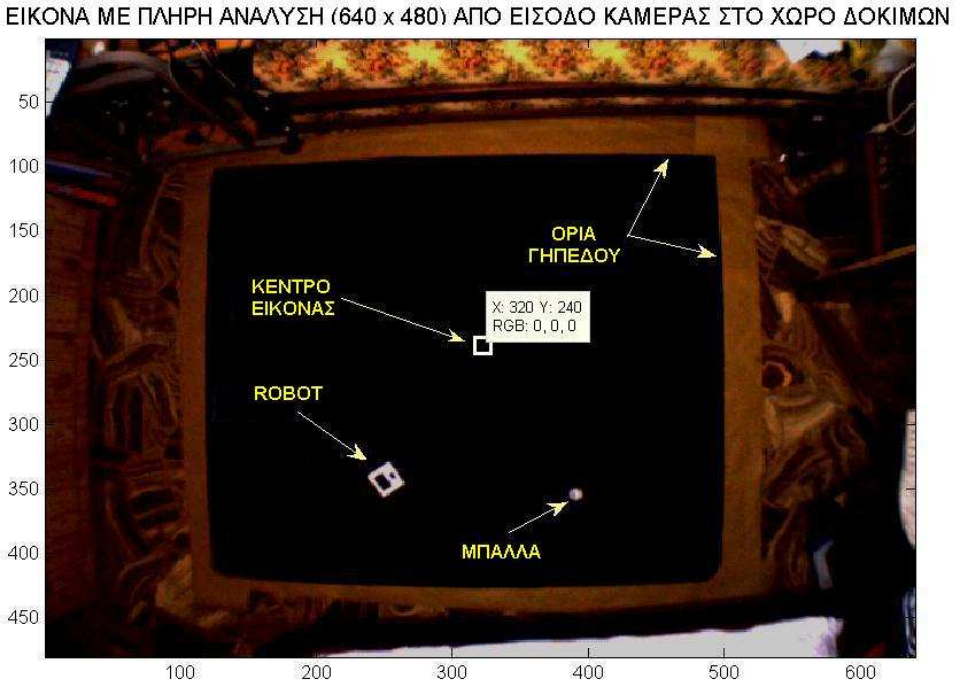
Σε περίπτωση μη επιτυχίας του προγράμματος , δηλαδή ότι το προγραμμα δεν δίνει τον έλεγχο στο βασικό πρόγραμμα “base.m” και συνεχώς επμφανίζει το παράθυρο του γηπέδου , σημαίνει ότι κάποια παράμετρος δεν είναι σωστή σύμφωνα με τις προδιαγραφές.

Σε περιπτωση επιτυχίας , το πρόγραμμα επιστρέφει αρκετές πληροφορίες που φαίνονται αναλυτικά στο τμήμα 4.3.2.5 Προετοιμασία τιμών για επιστροφή στο κυρίως πρόγραμμα.

ΓΕΝΙΚΟ ΔΙΑΓΡΑΜΜΑ ΤΟΥ ΤΜΗΜΑΤΟΣ ΑΡΧΙΚΟΠΟΙΗΣΗΣ -INITIALIZE (maininit.m)

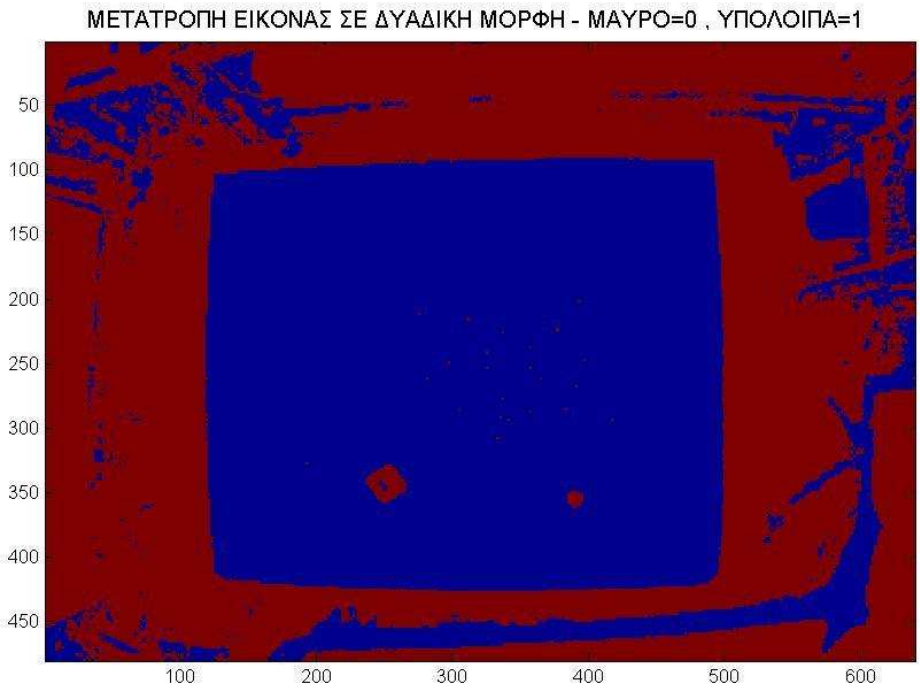
4.2.3.4α. Εύρεση γηπέδου.

Για την εύρεση των διαστάσεων του γηπέδου και της θέσης του παραλληλογράμου του στην εικόνα, χρησιμοποιούμε την πλήρη εικόνα που έρχεται από την κάμερα που αποτελείτε από τον πίνακα (640 χ 480) εικονοστοιχείων, και αποικονίζεται παρακάτω . στο υποπρόγραμμα maininint η εικόνα περιλαμβάνεται στον πίνακα helppic με την εντολή orpic=vcapg2(1), και αργότερα helppic= orpic που φαίνεται παρακάτω (ε4.20).



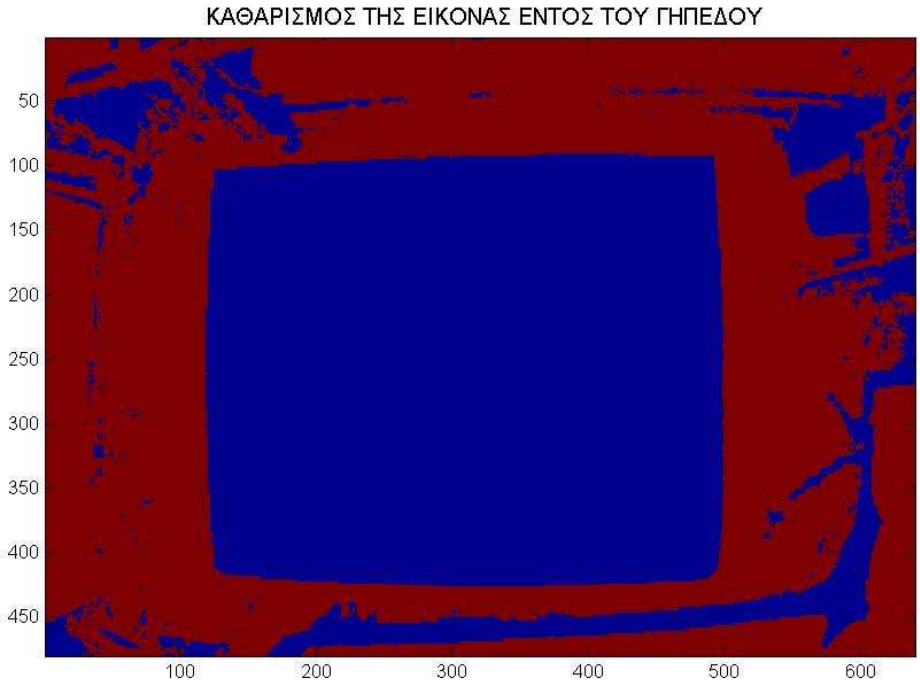
(ε4.20)

Για επισήμανση οι φωτογραφίες του χωρόυ του γηπέδου είναι από την επιφάνεια δοκιμών που αποτελείτε από μαύρο χαρτί στις ίδιες διαστάσεις με το γήπεδο, και ήταν ο χώρος που εφαρμόστηκαν τα περισσότερα τμήματα της εφαρμογής. Από την εικόνα παίρνουμε ένα δείγμα του χρωματος (rgb) του κέντρου δηλαδή του σημείου (320,240), και υποθέτουμε ότι είναι το σωστό χρώμα του γηπέδου, της μορφής (r,g,b) και τιμές στην συγκεκριμένη περίπτωση (0,0,0). Προϋπόθεση ότι έχουμε μεταφέρει το ρομποτικό παίκτη και την μπάλλα σε λίγο διαφορετικό σημείο από το κέντρο του γηπέδου. Στην συνέχεια ελέγχουμε σειριακά τον πίνακα της εικόνας κάθε ένα εικονοστοιχείο και στην περίπτωση που αμφανιστέι ένα εικονοστοιχείο διαφορετικό με το rgb=(0,0,0) τότε το μετατρέπουμε σε rgb(254,254,254), ώστε να έχουμε στο τέλος ένα δυαδικό πίνακα, όπως φαίνεται παρακάτω που περιλαμβάνεται στο HpicBW πίνακα (ε4.21)



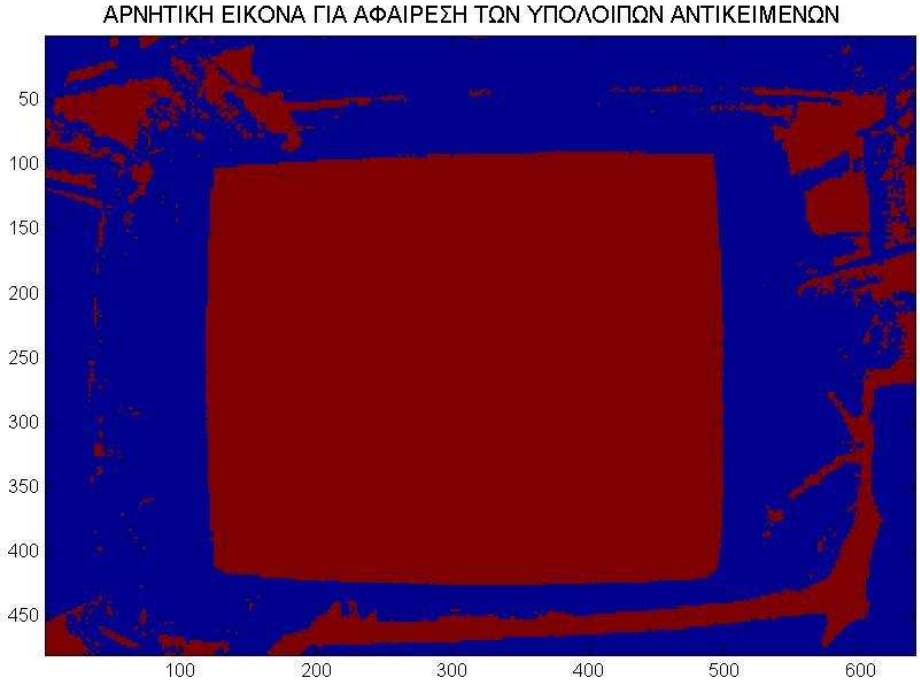
(ε4.21)

Μετά ακολουθεί καθαρισμός της εικόνας από μικρά τμήματα εντός του γηπέδου που φαίνεται στον πίνακα Hpic (ε.4.22)



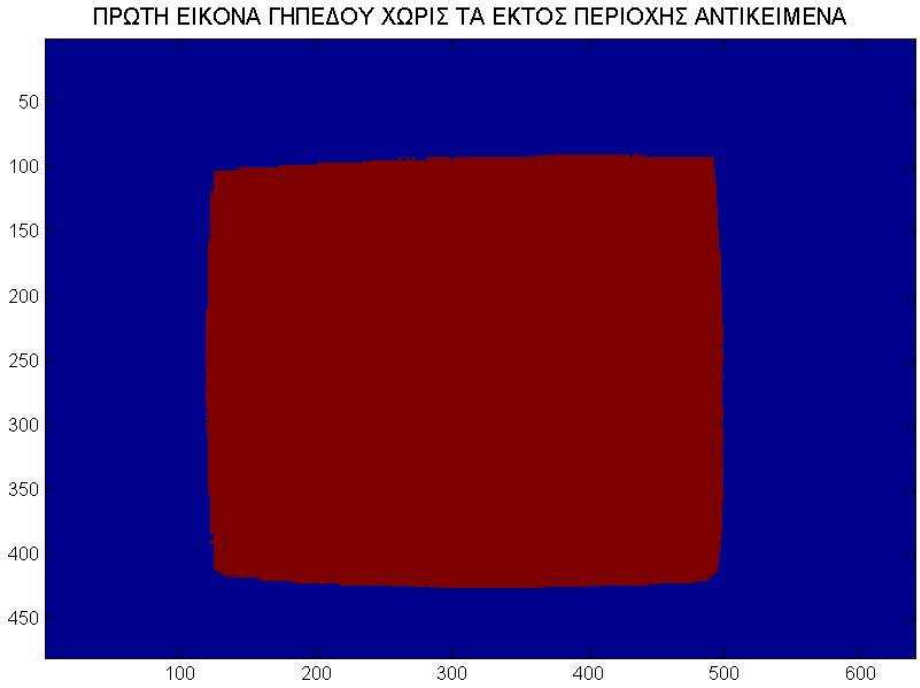
(ε4.22)

Σε αυτό το σημείο πρέπει να κάνουμε την εικόνα αρνητική για να μπορέσουμε να αφαιρέσουμε τα “σκουπίδια” με το επόμενο βήμα και φαίνεται στον πίνακα Hpic2 (ε.4.23)



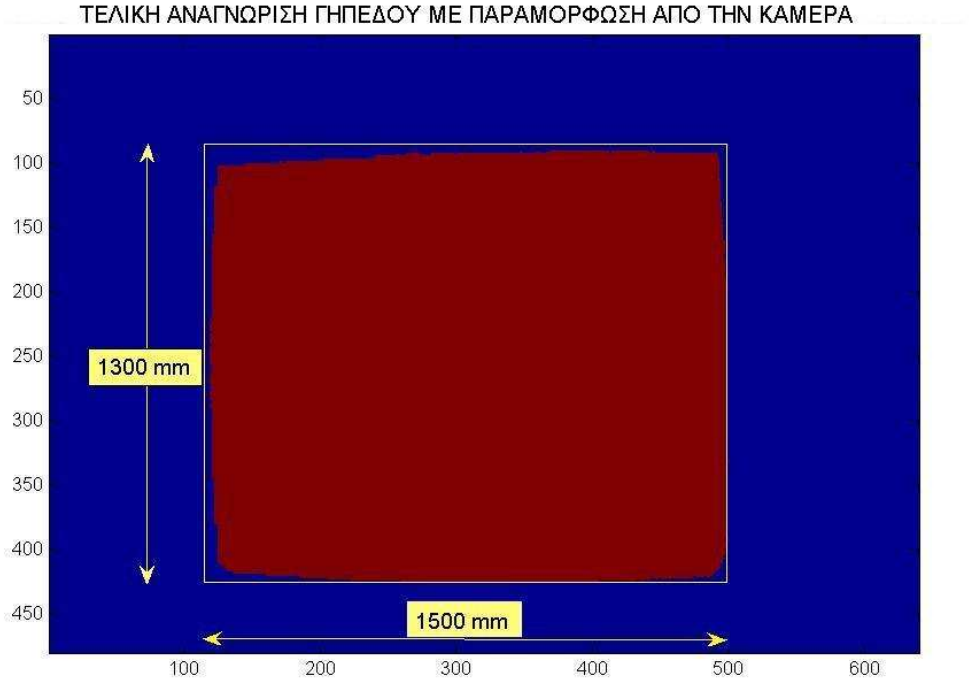
(ε4.23)

Όπως φαίνεται εδώ , αρχίζει να ξεχωρίζει το γήπεδο στο οποίο θα μας ενδιαφέρει πλέον οτιδήποτε υπάρχει μόνο μέσα σε αυτό. Στην συνέχεια πρέπει να αφαιρέσουμε όλα τα μικρότερα τμήματα σε σχέση με το γήπεδο που έχουν απομείνει και αυτό φαίνεται στον πίνακα Hpic3 (ε.4.24).



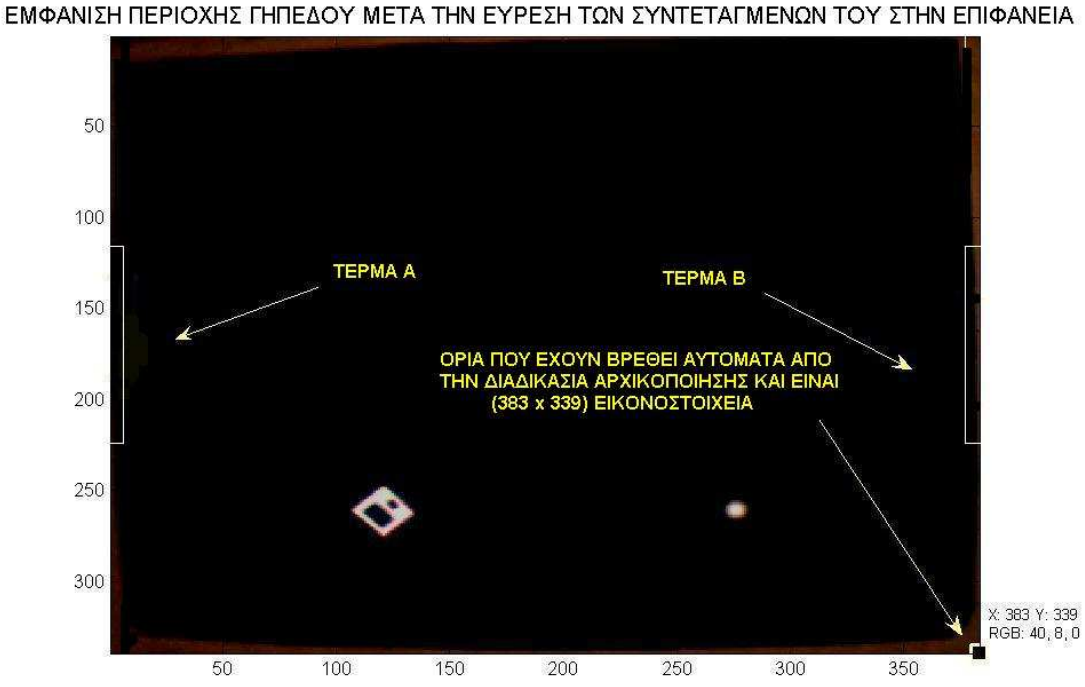
(ε4.24)

Ακολουθεί μία προσπάθεια με την εντολή bwboundaries, για να βρεθούν τα χαρακτηριστικά του γηπέδου μέσα στην εικόνα. Μετά την εκτέλεσή της έχουμε βρεί το περίγραμμα του γηπέδου που αποθηκεύεται στον πίνακα crtable, και το εμβαδόν της σε εικονοστοιχεία, το οποίο χρειάζεται κατά την λειτουργία του προγράμματος. Λόγω των παραμορφώσεων από την κάμερα έχει δεσμευτεί η επιφάνεια που βρίσκεται στο μέσα πλαίσιο όπως φαίνετε παρακάτω (ε.4.25) και περιγράφετε στο τμήμα 4.3.3.4 προετοιμασία τιμών για επιστροφή στο κυρίως πρόγραμμα.



(ε4.25)

Από αυτό το σημείο και μετά, όλες οι εικόνες που θα ακολουθήσουν , θα έχουν μόνο το γήπεδο σαν περιεχόμενο , χωρίς να μας ενδιαφέρει από έξω τι γίνεται και θα είναι για παράδειγμα της παρακάτω μορφής (ε.4.26). Το καλύτερο αποτέλεσμα για εμάς θα ήταν το παραλληλόγραμμο του γηπέδου να εφάπτεται ακριβώς στο σχεδιασμένο πλαίσιο της παραπάνω εικόνας (ε4.25), αλλά όπως φαίνεται αυτό σχύει και εξήγειτε στο τμήμα γενικών προβλημάτων.



(ε4.26)

4.2.3.4β Ευρεση - δημιουργία default Threshold στο υποπρόγραμμα

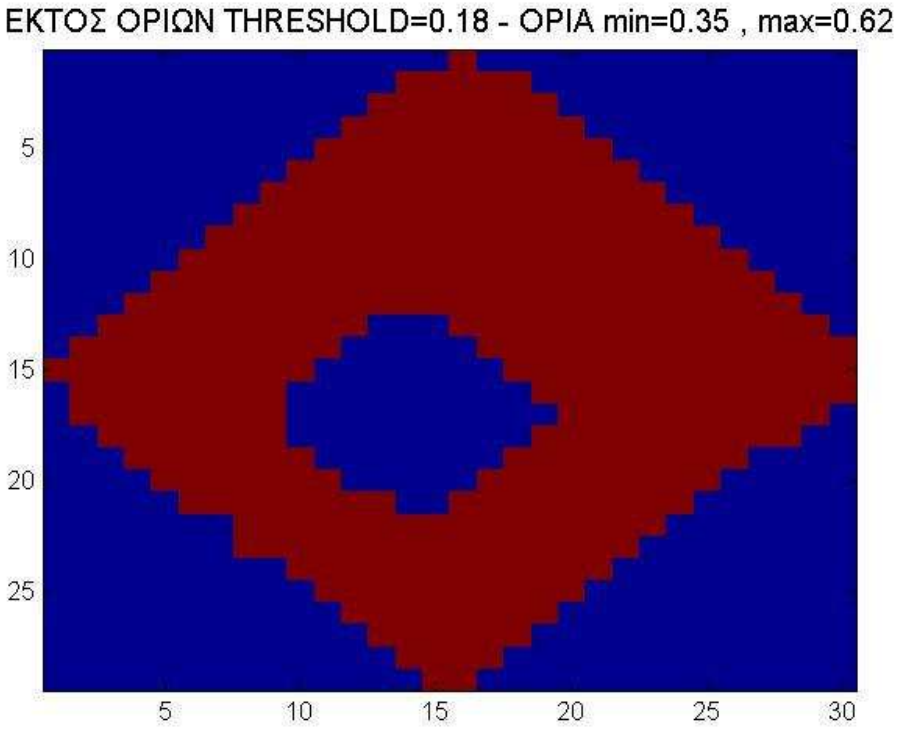
Η εύρεση του threshold γίνεται αυτόματα όπως περιγράφεται εδώ. Γνωρίζοντας την περιοχή του γηπέδου, για να λειτουργήσει αυτό το τμήμα του προγράμματος , χρειάζεται να υπάρχουν ένα ρομποτικός παίκτης και η μπάλα μέσα στον χώρο του κατά προτίμηση κοντά στο κέντρο, για να δοκιμαστούν όλες οι τιμές του threshold (φωτεινότητας στη συγκεκριμένη περίπτωση) και να επιλεχθεί η καλύτερη.

Σε αυτό το τμήμα προγράμματος υπάρχει κώδικας ο οποίος επαναλαμβάνεται από τιμή threshold=0 μέχρι threshold=1 με αυξητικό βήμα κατά 0.01, οπότε συνολικά γόνονται 100 επαναλήψεις. Σε κάθε επανάληψη γίνεται προσπάθεια αναγνώρισης του ρομποτικου παίκτη σαν μέτρο ελέγχου.

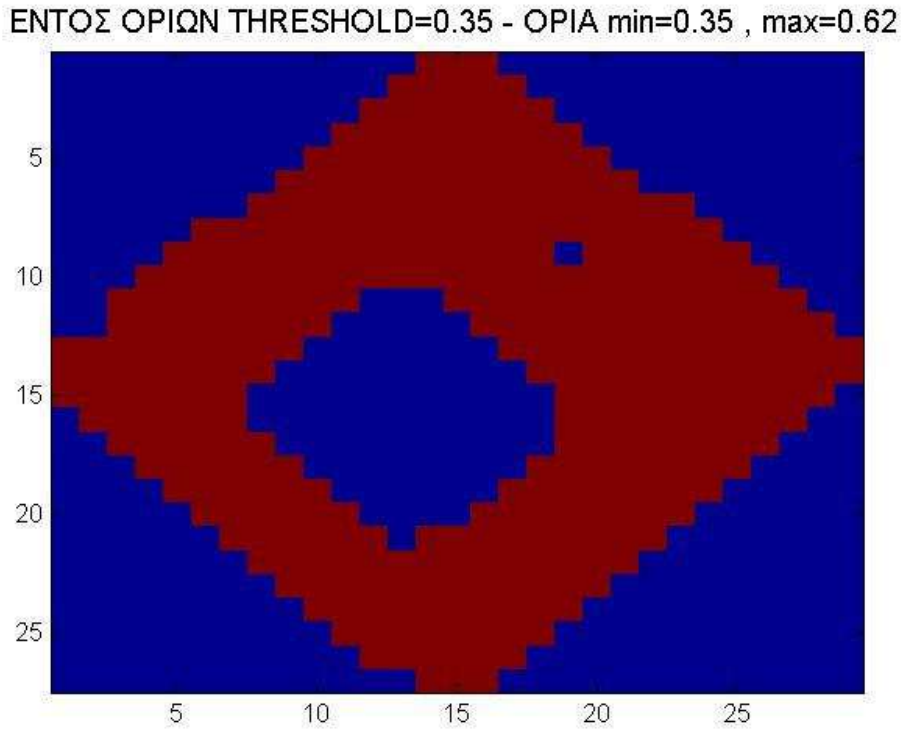
Σε κάθε επιτυχή αναγνώριση το συγκεκριμένο threshold κρατείτε σε έναν πίνακα. Η πρώτη φορά σωστής αναγνώρισης , είναι η πρώτη τιμή στον πίνακα γιατί οι υπόλοιπες είναι στην σειρά μέχρι να υπάρξει έστω μία μη σωστή αναγνώριση. Σε περίπτωση που δεν υπάρξει σωστή αναγνώρηση λόγο μεγέθους και χρωματικου ελέγχου του ρομποτικου παίκτη που θα αναφερουμε παρακάτω , τότε η συγκεκριμένη τιμή threshold δεν είναι αποδεκτή.

Παρακάτω φαίνονται κάποιες διαφορετικές τιμές του threshold και το αποτελέσμα που έχουν στην εικόνα. Παρατηρείτε ότι όσο πιο έξω είμαστε από τα όρια του threshold, τόσο ποιο πολύ πληροφορία χάνεται ή παραμορφόνεται. Όπως εικονίζεται

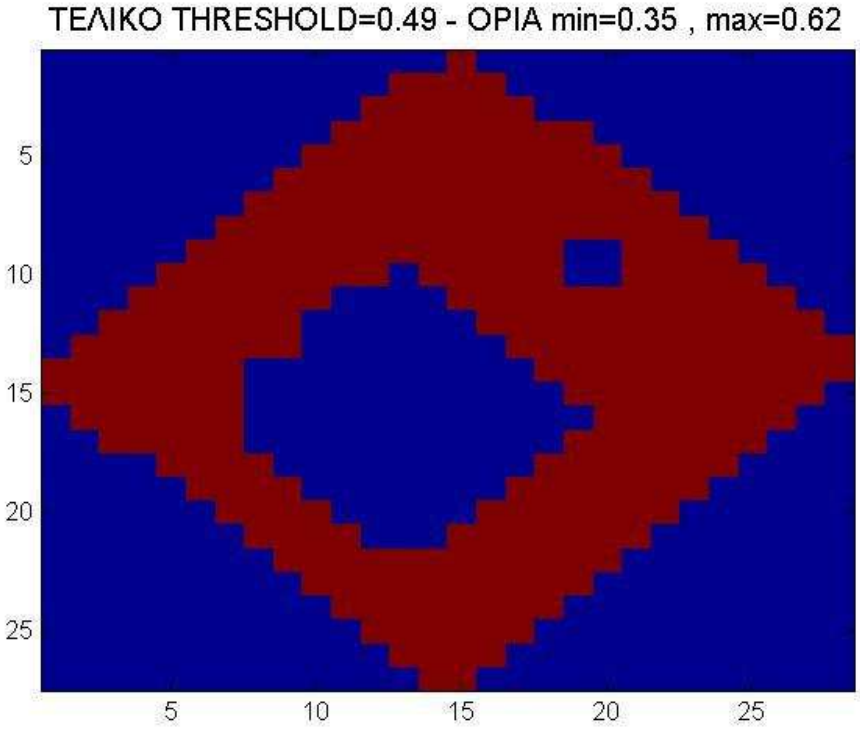
στα παρακάτω παραδείγματα (ε.4.27 έως ε.4.30) έχοντας έγκυρες τιμές 0,35<threshold<0,62, ενώ οποιαδήποτε άλλη τιμή παρουσιάζει προβλήματα.



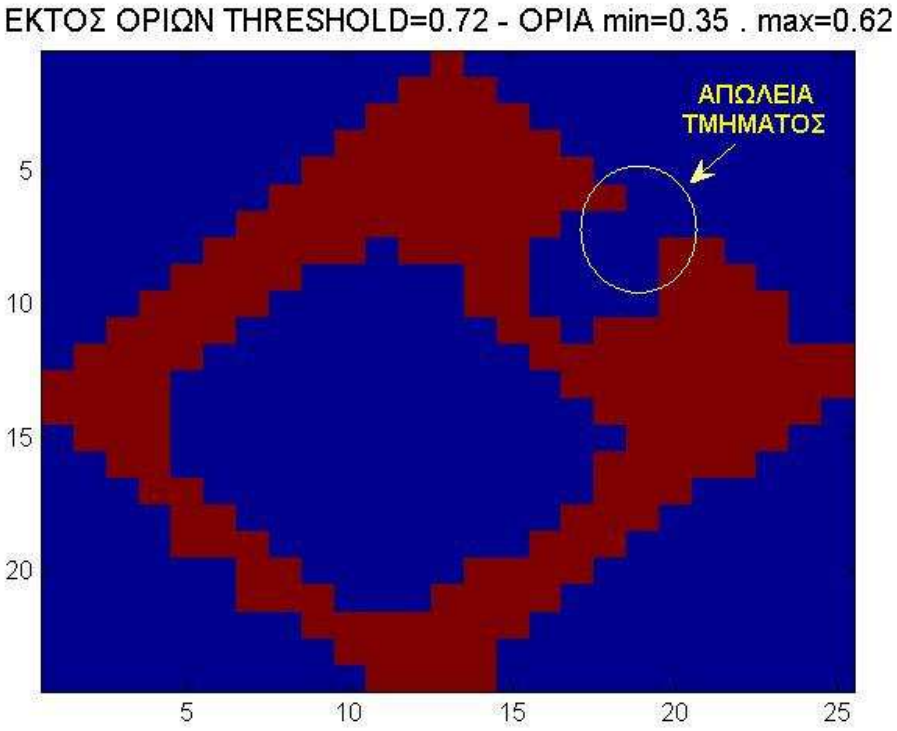
(ε4.27)


ΕΝΤΟΣ ΟΡΙΩΝ THRESHOLD=0.35 - OPIA min=0.35 , max=0.62
5
10
15
20
25
5
10
15
20
25

(ε4.28)


ΤΕΛΙΚΟ THRESHOLD=0.49 - OPIA min=0.35 , max=0.62
5
10
15
20
25
5
10
15
20
25

(ε4.29)



(ε4.30)

Πρέπει να αναφέρουμε ότι το κατάλληλο threshold είναι το κεντρικότερο έγκυρο threshold ανάμεσα στα έγκυρα όρια threshold που έχουν βρεθεί, το οποίο βέβαια εξαρτάται από το πόσο ομοιογενή είναι η φωτεινότητα.

Επίσης η τελική τιμή threshold μας δίνει τη δυνατότητα να μπορούμε γρηγορότερα και πάντα να αναγνωρίζουμε το αντικείμενο σε οποιαδήποτε μικρή σχετικά αλλαγή φωτισμού, επειδή έχει την καλύτερη αναλογία υλικού με τα δύο κενά μέρη που έχει σχεδιαστεί το ρομποτικός παίκτης σε σχέση με τα οριακά threshold, και να αποφύγουμε τα προβλήματα που εμφανίστηκαν παραπάνω σε όσα ήταν εκτός ορίων.

4.2.3.4.γ Εύρεση – δημιουργία χαρακτηριστικών μπάλλας και ρομποτικού παίκτη

Η αναγνώρηση της μπάλλας κατά την διάρκεια της αρχικοποίησης , γίνεται με την χρησιμοποίηση ενός Threshold=0.1 το οποίο δεν χρησιμοπιείτε για αναγνώρηση των ρομποτικών παικτών.

Συγκεκριμένα , μέσα σε μία επαναληπτική διαδικασία , μέσα στο γήπεδο αναζητούνται όλα τα αντικείμενα με μέγεθος περίπου ίσο με τις εμπειρικές τιμές που έχουν δηλωθεί ήδη για σωστά μεγέθη μπάλλας σε εικονοστοιχία και συγχρόνως γίνεται χρεωματικός έλεγχος, όπου οι τιμές πρέπει να βρίσκονται μέσα στα όρια των τιμων RGB που έχουν δηλωθεί για το συγκεκριμένο χρώμα μπάλλας με το συγκεκριμένο Threshold.

Στις τιμές RGB που βρέθηκαν από την μπάλλα , δημιουργουται ανοχές +-33 για λόγους πιθανής ανομειογένειας φωτεινότητας, από την κάθε τιμή RGB που βρέθηκε και αποθηκευονται στον πίνακα ColorArray.

Η αναγνώρηση του ρομποτικού πάικτη κατά την διάρκεια της αρχικοποίησης γίνειται με τον ίδιο τρόπο ελέγχοντας τα ίδια χαρακτηριστικά , αλλα με άλλη διαδικασία.

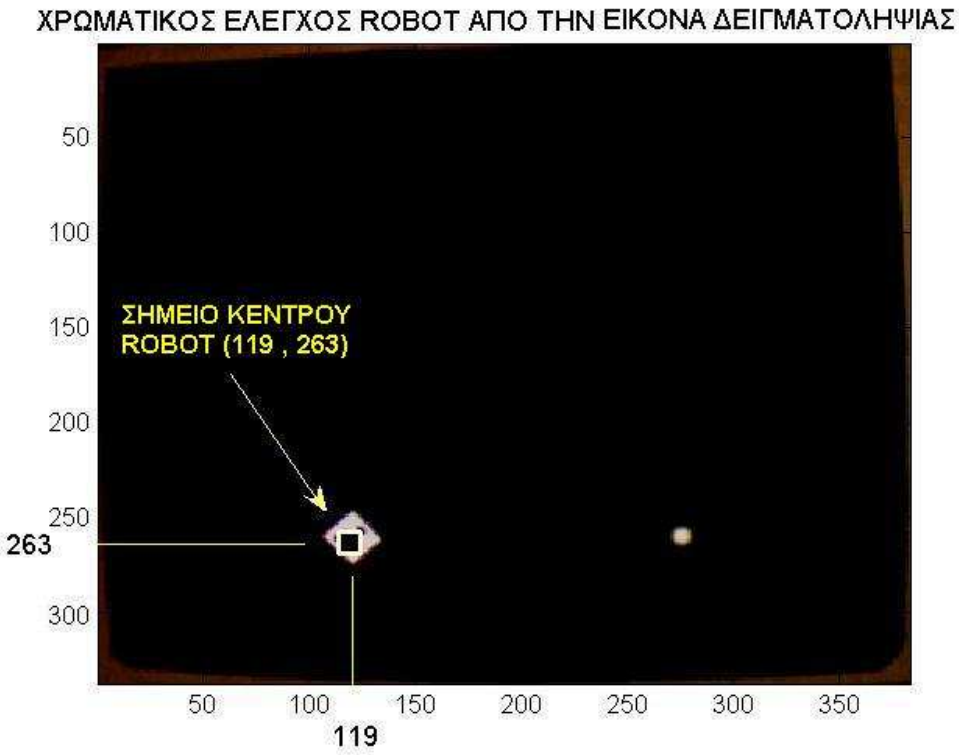
Κατά την διάρκεια της εύρεσης του Threshold , γινεται τοπική δειγματοληψία σε κάθε αντικέιμενο μεγέθους ρομποτικού παίκτη και ελέγχεται μέσα στο περίγραμμα του, εάν υπάρχουν τα δύο τετράγωνα με το αντίστοιχο χρώμα της ομάδας μας. Το εικονοστοιχίο που γίνεται ο χρωματικός έλεγχος φαίνεται στην παρακάτω εικόνα (ε4.31) και επιλέγεται αυτό το σημείο, για συγκεκριμένους λόγους που περιγράφεται στο τμήμα ανάλυσης του robot.

Το καλυτερο threshold που έχει βρεθεί ήδη από την διαδικασία της εύρεσης του είναι ο Μ.Ο των τιμών των σωστών threshold.

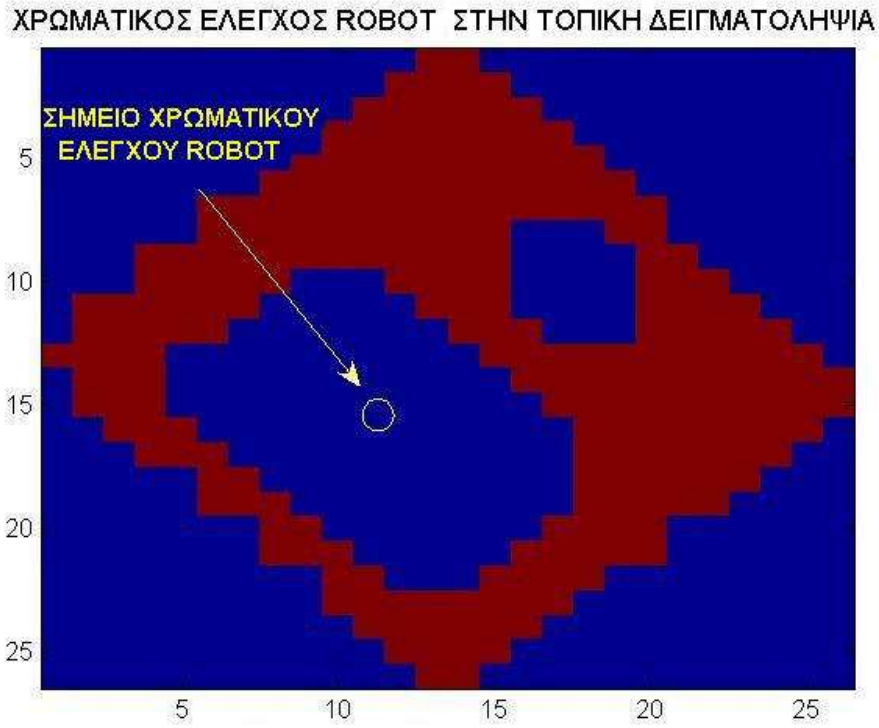
Επίσης οι τιμές μεγέθους και οι τιμές χρώματος RGB στο κέντρο του μεγάλου τετράγώνου καταγράφονται , δημιουργούνται ανοχές +-33 σε κάθε χρώμα , και αποθηκεύονται στον πίνακα ColorArray.

Το χρωματικό δείγμα παραλαμβάνεται με την παρακάτω εντολή και η διαδικασία γενικά της αναγνώρησης , φάινεται αναλυτικότερα στα σχήματα που ακολουθουν.

impixel(pic,ImageCentroidX,ImageCentroidY)-ColorAnnot;


ΧΡΩΜΑΤΙΚΟΣ ΕΛΕΓΧΟΣ ROBOT ΑΠΟ ΤΗΝ ΕΙΚΟΝΑ ΔΕΙΓΜΑΤΟΛΗΨΙΑΣ
ΣΗΜΕΙΟ ΚΕΝΤΡΟΥ
ROBOT (119 , 263)
263
119
50
100
150
200
250
300
350

(ε4.31)


ΧΡΩΜΑΤΙΚΟΣ ΕΛΕΓΧΟΣ ROBOT ΣΤΗΝ ΤΟΠΙΚΗ ΔΕΙΓΜΑΤΟΛΗΨΙΑ
ΣΗΜΕΙΟ ΧΡΩΜΑΤΙΚΟΥ
ΕΛΕΓΧΟΥ ROBOT
5
10
15
20
25

(ε4.32)

4.2.3.4.δ. Προετοιμασία τιμών για επιστροφή στο κυρίως πρόγραμμα.

Τελικό στάδιο της αρχικοποίησης είναι η τοποθέτηση σε ένα πίνακα όλων των στοιχειών που έχουν βρεθεί με αυτό το υποπρόγραμμα ώστε να μπορούν να μεταφέρθουν στο κυρίως προγραμμα “Base.m” , όπου χρειάζονται για την λειτουργίας του, όπως εμφανίζονται παρακάτω

α/α	χαρακτηριστικά
1	πίνακας διαστάσεων γηπέδου
2	Εύρος χρώματος γηπέδου
3	Threshold μέσος όρος
4	μέγεθος ρομποτικού παίκτη σύμφωνα με το ύψος της κάμερας
5	μέγεθος μπάλας σύμφωνα με το ύψος της κάμερας
6	Εύρος χρώματος ρομποτικού παίκτη σύμφωνα με την φωτεινίτητα σε RGB τιμές
7	Εύρος χρώματος μπάλας σύμφωνα με την φωτεινίτητα σε RGB τιμές

4.2.3.5 Ανάλυση σχήματος ρομποτικού παίκτη και χαρακτηριστικά του

Μετά από πολλές προσπάθειες σχεδιασμού για την αναγνώριση των ρομποτικών παικτών , και σύμφωνα με τους κανονισμούς του συγκεκριμένου πρωταθλήματος κατέληξα στο παρακάτω σχήμα το οποίο τοποθετείτε στην κορυφή του ρομποτικού παίκτη, το οποίο με παραλαγές μεγέθους των δύο τετραγώνων γίνεται ο διαχωρισμός του κάθε παίκτη και σίγουρα ξεχωρίζει από τα αντίπαλα.

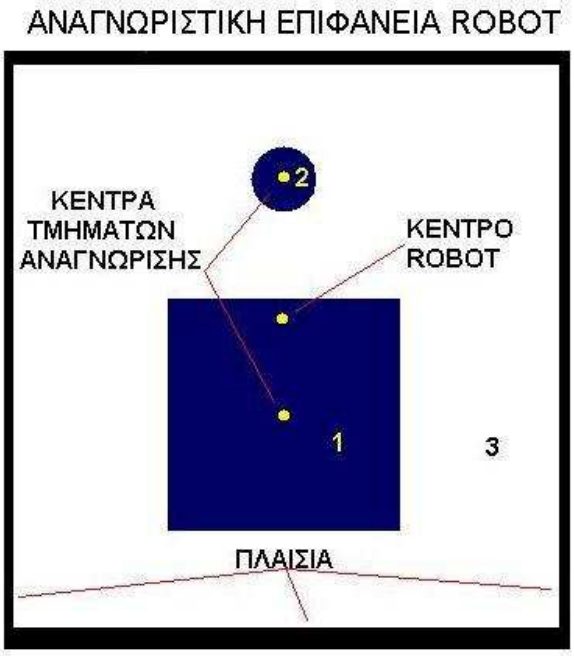
Βασικός σκοπός είναι , από ένα σχήμα να μπορούμε εύκολα να πάρουμε τις πληροφορίες που θέλουμε γρηγορότερα και με μεγαλύτερη ακρίβεια τόσο σε στάση όσο και σε κίνηση, κερδίζοντας χρόνο εκτέλεσης κώδικα .

Το κάθε στοιχείο πάνω στο κεφάλη του ρομποτικού παίκτη εξυπηρετεί ένα σκοπό, και είναι σχεδιασμένο σε απλό χαρτί, το οποίο δεν πρέπει να αντανακλάει το φώς, ώστε να παρουσιάζονται διάφορες αλοιώσεις στην εικόνα του σχήματος , καθώς καταγράφεται από την κάμερα. Αποτελείτε από ένα άσπρο πλαίσιο συγκεκριμένων διαστάσεων που αναφέρεται στους πίνακες πληροφοριών, έχει δύο τετράγωνα ίδιων χρώμάτων εσωτερικά αλλά διαφορετικών διαστάσεων σε σχέση μεταξύ τους, εκ των οποίον το μεγαλύτερο τετράγωνο (3.5 x 3.5 εκατοστών) είναι υποχρεωτικό και έχει τουλάχιστον την μικρότερη επιτρεπτή διάστασηλόγο επισης κανονισμών, καθώς και ένα μαύρο πλαίσιο περιμετρικά.

Επειδή η ομάδα αποτελείτε από τρείς ρομποτικούς παίκτες, οι υπόλοιποι δύο είναι παραλαγές του παρακάτω εικονιζόμενου, με μικρές διαφορές στα εσωτερικά τετράγωνα. Τα δύο τετράγωνα ανάλογα με τον λόγο του εμβαδού τους , χρησιμοποιούνται για τον διαχωρισμό των δικών μας ρομποτικών παικτών. Το κέντρο του μεγαλύτερου τετραγώνου, χρησιμοποιείτε για την δειγματοληψία χρώματος, απαραίτητη για τον διαχωρισμό των δικών μας από τα εχθρικά robot καθώς και για τον διαχωρισμό του καθε δικου μας παίκτη , ώστε να παραλάβει την κατάλληλη εντολή κίνησης.

Αποτελεί απαραίτητο μέτρο ελέγχου επειδή , λογικά όλοι οι ρομποτικοί παίκτες έχουν το ίδιο μέγεθος και θα μπορούσαν να έχουν ένα παρόμοιο αριθμό τετραγώνων, ή

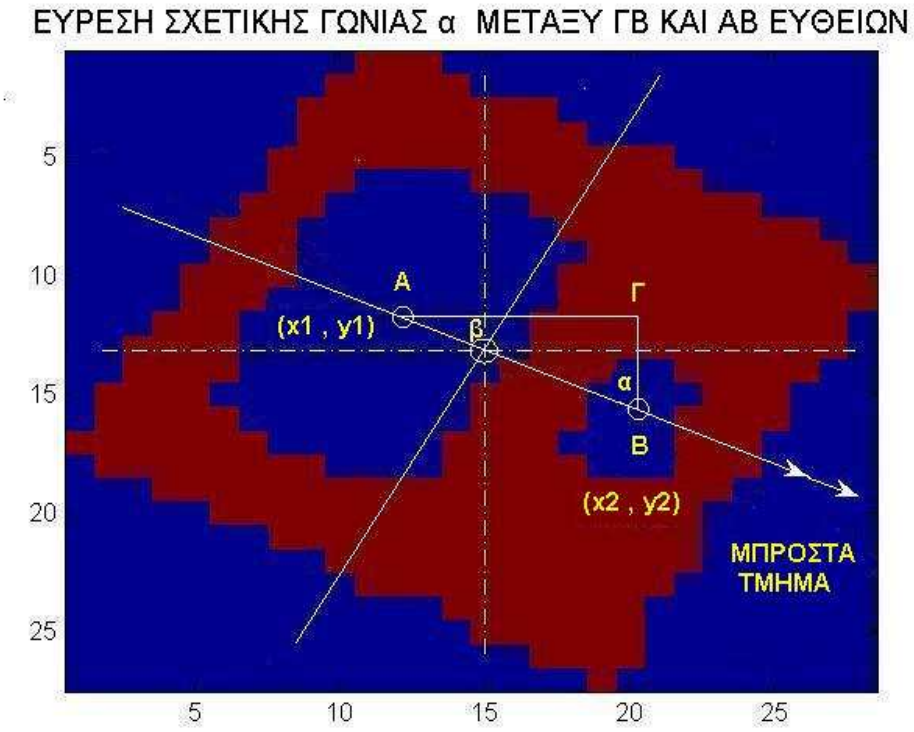
οποιδήποτε άλλο σχέδιο που θα έκανε ο αντίπαλος θα μπορούσε να μας προκαλέσει πρόβλημα στην αναγνώριση. Το περιμετρικό πλαίσιο χρημοποιείται σε περίπτωση που δύο robot ακουμπήσουν το ένα με το άλλο με συνέπεια να εμφανίζεται ένα ενιαιο σχήμα, που δεν θα ήταν αναγνωρίσιμο (ε4.33)



(ε4.33)

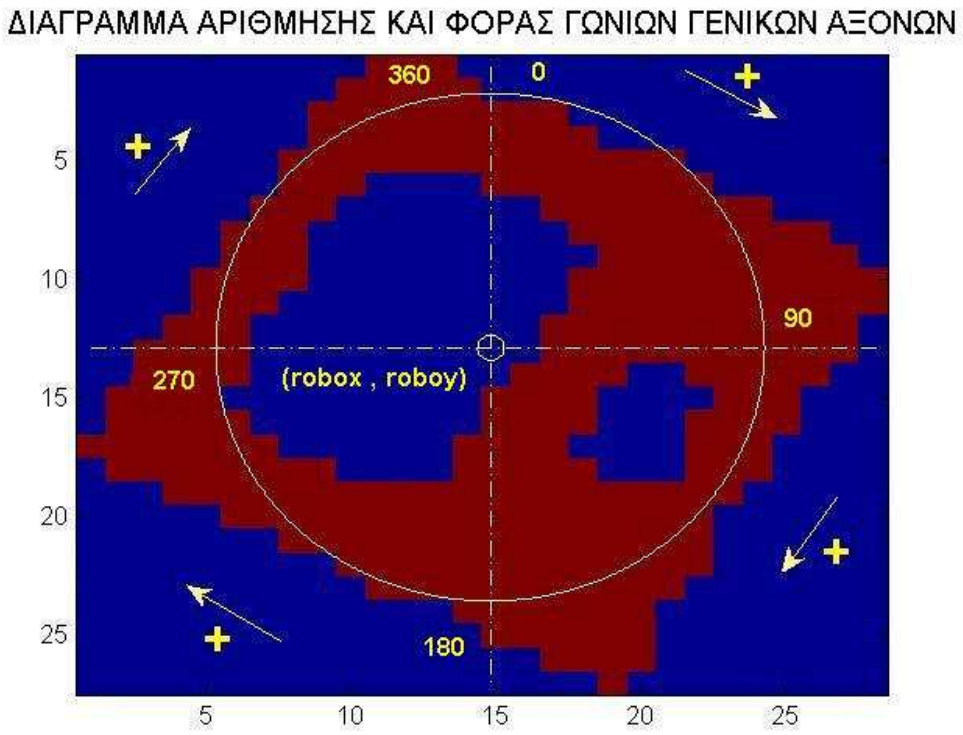
Η θέση των σημειακών κέντρων των δύο τετραγώνων μεταξύ τους, μας δίνει την δυνατότητα να υπολογίσουμε ποια είναι η κλίση του ρομποτικού παίκτη σε σχέση με τους γενικούς άξονες που έχουμε δημιουργήσει και φαίνονται σε παρακάτω σχεδιάγραμμα (ε4.34). Ανάλογα τις διαφορές x=(x2-x1) , y=(y2-y1) και την γωνία α, γχρησιμοποιείτε ο ανάλογος κώδικας για να μετατραπεί η γωνία, σε γωνία γενικών αξόνων. Παρακάτω ακολουθεί στο σχέδιάγραμμα ένα παράδειγμα , όπου στη συγκεκριμένη περίπτωση έχω α=61º ,δηλαδή 180 º -90 º -61 º => β=29 º και γωνία γενικών αξόνων του robot 90 º +β =119 º.

Με αντίστοιχες αυξομοιώσεις των γωνιών α και β και ανάλογα τις τέσερεις περιπτώσεις που υπάρχουν x>0, x<0, y>0, y<0, βρίσκουμε την τελική γωνία κλίσης του κάθε ρομποτικού παίκτη σύμφωνα με τους γενικούς άξονες που έχουμε ορίσει. Η γωνία αναφέρεται στο μπροστά τμήμα με το οποίο παραλαμβάνει την μπάλα, και η φορά του απεικονίζεαι με την γραμμή που έχει διπλά βέλη στο παρακάτω σχέδιο (ε.4.34).



(ε4.34)

Επίσης η φορά και αρίθμηση των γωνιών που χρησιμοποιούμε εδώ μετατρέπονται σύμφωνα με την παρακάτω εικόνα (ε4.35). Ο κάθε ρομποτικός παίκτης έχει πάντα το κέντρο του (robox,roboy) , όπως ονομάζεται στον κώδικα και απεικονίζεται σαν σχήμα μέσα στο γήπεδο.



(ε4.35)

4.2.3.6 Πώς γίνεται η αναγνώριση των αντικειμένων

Γενικά σε μία αρχική δειγματοληψία απλά έχουμε γνωστό τον αριθμό των αντικειμένων μέσα στο γήπεδο (δικά μας, αντίπαλα και την μπάλα) και η εικόνα έρχεται από την κάμερα. Τα αντικείμενα πρέπει να είναι 7, γιατί έχουμε 3 ρομποτικούς παίκτες δικούς μας , 3 εχθρικούς και μία μπάλα.

Υπαρχει μία μεγάλη επαναλαμβανόμενη διαδικασία (while fullgame=1) έχουμε σε κάθε επανάληψη την νέα καταγραφή από την κάμερα του τμήματος που μας ενδιαφέρει. Από τον πίνακα orpic που τοποθετείτε η εικόνα , παραλαμβάνουμε την περιοχή μου μας ενδιαφέρει σύμφωνα με τις τιμές ττης περιμέτρου που βρίσκεται στον πίνακα crtable που έχουμε αυτοματα ορίσει κατά την εκκινηση του προγράμματος

Αρχικά χρησιμοπιείται το βασικό threshold και διαβάζεται η εικόνα καθώς σημειώνονται τα αντίστοιχα αντικείμενα στον πινακα B με τις παρακάτω εντολές:

```
[B,L] = bwboundaries(im2bw(pic,Thold),'noholes');
stats = regionprops(L,'Image','Centroid','BoundingBox','Eccentricity');
```

4.2.3.6.α Αναγνώρηση δικών μας και εχθρικών ρομποτικών παικτών

Ακολουθεί έλεγχος για όλα τα αντικείμενα που βρέθηκαν και δημιουργείτε πίνακας temprobo με όσα αντικείμενα έχουν εξωτερικό μέγεθος περίπου ίσο και σχήμα περίπου ίδιο ενός ρομποτικού παίκτη και τοποθετουνται τα αποτελέσματα εκεί.

Ο έλεγχος γινεται με την εντολή
abs(stats(R).Eccentricity-0.5)<0.25) && (length(B{R})>40) && (length(B{R})<100)

Στην συνέχεια γίνεται τοπική δειγματοληψία κάθε αντικειμένου , δηλαδή αντιγραφή από την αρχική εικόνα το τμήμα τις περιμέτρου του, όπου ελέγχεται κάθε φορά για κάθε αντικείμενο μέχρι το τέλος, εάν ανοίκει στους δικούς μας ρομποτικούς παίκτες ή στους αντιπάλους. Σε περίπτωση που δεν βρεθεί ότι είναι δικό μας , γίνεται προσπάθεια με το επόμενο σωστό threshold η αναγνώριση για το συγκεκριμένο αντικείμενο. Εάν τελειώσουν όλα τα υποψήφια threshold και δεν έχει αναγνωριστεί ότι είναι δικό μσας , τότε το συγκεκριμένο αντικείμενο κατατάσετε στα αντίπαλα.

Όταν κατηγοριοποιηθεί το αντικείμενο ότι είναι αντίπαλο ή ότι είναι δικός μας ρομποτικός παίκτης , τότε ο έλεγχος συνεχίζεται στο επόμενο υποψήφιο αντικείμενο και αυτό γινεται μέχρι το τέλος του αριθμού των υποψηφίων αντικειμένων.

Κατά την διαδικασία της τοπικής δειγματοληψίας , έχουμε σαν κριτήριο ελέγχου τα στοιχεία που έχουμε συλλέξει από το τμήμα της αρχικοποίησης, δηλαδή το μέγεθος του ρομποτικού παίκτη , της μπάλας, και τα χρώματα τους.

(ε4.36)

Ας υποθέσουμε ότι το αντικείμενο N=2 (ε4.36) είναι ένας δικός μας ρομποτικός παίκτης. τότε με την τοπική δειγματοληψία έχουμε καταφέρει να έχουμε την πλήρη εικόνα του και τα χαρακτηριστικά του , δηλαδή συνολικό μέγεθος, μεγέθη των τετραγώνων 1,2, χρώμα, όπως φαίνεται παρακάτω στο σχήμα (ε4.37). Αυτό γίνεται με τις παρακάτω εντολές.

frame=stats(N).Image;

ο πίνακας frame περιλαμβάνει την εικόνα του αντικειμένου (2) που υποθέτουμε ότι είναι ο ρομποτικός παίκτης.

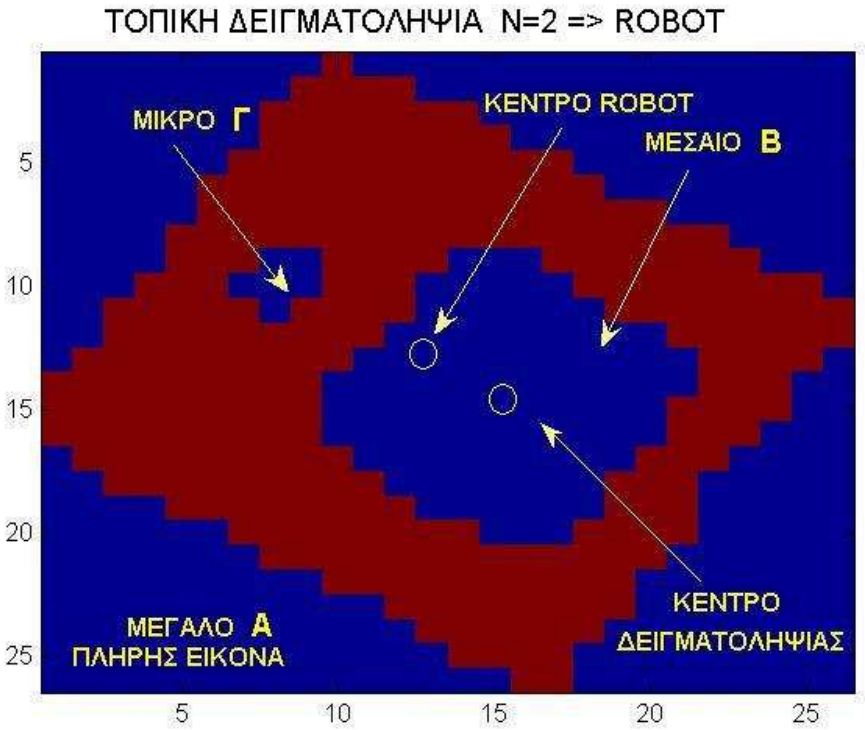
[HB,HL]=bwboundaries(frame,'holes');

Γίνεται αναζήτηση για αντικείμενα μέσα στην εικόνα του πίνακα του ρομποτικού παίκτη.

Hstats =regionprops(HL,'Centroid','FilledArea');

επιστρέφει τα δεδομένα που υπάρχουν στο κέντρο του κάθε τετραγώνου (1,2,3) καθως και τις περιμετρου τους.

Η τιμή 'FilledArea' αποτελεί το εμβαδόν/μέγεθος σε εικονοστοιχεία του ρομποτικου παίκτη..



(ε4.37)

Δηλαδή η τοπική δειγματοληψία περιλαμβάνει τον παρακάτω τρόπο ελέγχου:

α) Εδώ παίρουμε ορισμένες τιμές που είναι ο λόγος του μεγέθους του μεσαίου τετραγώνου προς το μεγάλο Β/Α και ο λόγος του μεγέθους του μικρού τετραγώνου προς το μεσαίο τετραγώνο Γ/Β. Σάν μέτρο σύγκρισης έχω τελικά τον πρώτο λόγο δια τον δεύτερο λόγο. Για παράδειγμα εάν είναι ο τελικός λόγος είναι 0.3 <2 τότε έχουμε τον ρομποτικό παίκτη 3 που είναι επιθετικός για παράδειγμα, ενώ αν είναι > 6 τότε έχουμε το ρομποτικό παίκτη 2 που είναι τερματοφύλακας , διαφορετικά έχω τον ρομποτικό παίκτη 1 που είναι αμυντικός και επιτρέπεται ο κάθε ένας να εκτελεί τον δικό τους ρόλο σύμφωνα με τους κανονισμούς και την στρατηγική μας.

β) Επιπλέον πραγματοποιείται η χρωματική δειγματοληψία στο κέντρο δειγματοληψίας και συγκρίνεται με τις τιμές που εχουμε βρεί στην αρχικοποίηση. Αυτοί οι έλεγχοι γίνονται όπως αναφέραμε παραπάνω κατά την διάρκεια ελέγχου του κάθε αντικειμένου που έχει μεγεθος και σχήμα ένα ρομποτικό παίκτη.

Για να έχουμε την δυνατότητα διαχωρισμού των ρομποτικών παικτών, πρέπει να υπάρχουν οι τιμές σύγκρισης που βρεθηκαν εμπειρικά με την ακόλουθη διαδικασία. Εγινε δειγματοληψία σε κάθε ρομποτικό παίκτη σε διάφορετικές θέσεις αποθηκεύοντας τις αντίστοιχες τιμές των τετραγώνων τους, και απο αυτές στο τέλος δημιουργήθηκαν τα όρια μεγεθών του κάθε ρομποτικού παίκτη, που αποτελεί το μέτρο σύγκρισης. Ο πίνακας διαχωρισμού φτιάχτηκε με σταθερό threshold, για να έχουμε ανάλογες τιμές, και παριστάνεται παρακάτω πίνακα.

Αναλόγος το threshold , οι τιμές μπορεί να αλλάζουν , αλλα θα αλλάζουν όλες αναλογικά , οπότε θα έχουμε σταθερους λόγους.

Πίνακας αναφοράς αναγνώρισης ρομποτικών παικτών

ROBOT ID	**ΡΟΜΠΟΤΙΚΟΣ ΠΑΙΚΤΗΣ 1**					
	ΣΧΗΜΑ (ΠΑΡΑΛΛΗΛΟΓΡΑΜΜΟ)			ΤΙΜΕΣ ΠΟΥ ΠΡΟΚΥΠΤΟΥΝ		ΤΙΜΕΣ
ΣΧΗΜΑ (ΠΑΡΑΛΛΗΛΟΓΡΑΜΟ)	ΜΕΓΑΛΟ	ΜΕΣΣΑΙΟ	ΜΙΚΡΟ	a1	a2	
ΦΥΣΙΚΗ ΔΙΑΣΤΑΣΗ (mm)	85 x 95	35 x 35	Φ10			
ΘΕΣΗ ΡΟΜ/ΚΟΥ ΠΑΙΚΤΗ ΔΕΙΓΜΑΤΟΛΗΨΙΑΣ	(εικονοστοιχεια)					
ΚΕΝΤΡΟ ΕΥΘΕΙΑ	342	66	3	0.1930	0.0455	4.2456
>> ΔΙΑΓΩΝΙΑ	349	71	3	0.2034	0.0423	4.8147
ΠΑΝΩ ΔΕΞΙΑ ΓΩΝΙΑ	362	85	6	0.2348	0.0706	3.3264
>> ΔΙΑΓΩΝΙΑ	364	92	7	0.2527	0.0761	3.3218
>> ΑΝΑΣΤΡΟΦΑ 180 ΜΟΙΡΕΣ	357	88	7	0.2465	0.0795	3.0988
ΚΑΤΩ ΔΕΞΙΑ ΓΩΝΙΑ	362	100	6	0.2762	0.0600	4.6041
>> ΔΙΑΓΩΝΙΑ	368	95	8	0.2582	0.0842	3.0656
ΟΡΙΑ ΤΙΜΩΝ ΜΕ THRESHOLD=0.64						**>3 και <4.8**

ROBOT ID	**ΡΟΜΠΟΤΙΚΟΣ ΠΑΙΚΤΗΣ 2**					
	ΣΧΗΜΑ (ΠΑΡΑΛΛΗΛΟΓΡΑΜΜΟ)			ΤΙΜΕΣ ΠΟΥ ΠΡΟΚΥΠΤΟΥΝ		ΤΙΜΕΣ
ΣΧΗΜΑ (ΠΑΡΑΛΛΗΛΟΓΡΑΜΟ)	ΜΕΓΑΛΟ	ΜΕΣΣΑΙΟ	ΜΙΚΡΟ	a1	a2	
ΦΥΣΙΚΗ ΔΙΑΣΤΑΣΗ (mm)	85 x 95	50 X 35	Φ10			
ΘΕΣΗ ΡΟΜ/ΚΟΥ ΠΑΙΚΤΗ ΔΕΙΓΜΑΤΟΛΗΨΙΑΣ	(εικονοστοιχεια)					
ΚΕΝΤΡΟ ΕΥΘΕΙΑ	349	102	3	0.2923	0.0294	9.9370
>> ΔΙΑΓΩΝΙΑ	342	95	2	0.2778	0.0211	13.1944
ΠΑΝΩ ΔΕΞΙΑ ΓΩΝΙΑ	358	124	6	0.3464	0.0484	7.1583
>> ΔΙΑΓΩΝΙΑ	362	118	6	0.3260	0.0508	6.4107
>> ΑΝΑΣΤΡΟΦΑ 180 ΜΟΙΡΕΣ	359	117	6	0.3259	0.0513	6.3552
ΚΑΤΩ ΔΕΞΙΑ ΓΩΝΙΑ	363	131	6	0.3609	0.0458	7.8792
>> ΔΙΑΓΩΝΙΑ	365	130	6	0.3562	0.0462	7.7169
ΟΡΙΑ ΤΙΜΩΝ ΜΕ THRESHOLD=0.64						**>6 και <20**

ROBOT ID	**ΡΟΜΠΟΤΙΚΟΣ ΠΑΙΚΤΗΣ 3**					
	ΣΧΗΜΑ (ΠΑΡΑΛΛΗΛΟΓΡΑΜΜΟ)			ΤΙΜΕΣ ΠΟΥ ΠΡΟΚΥΠΤΟΥΝ		ΤΙΜΕΣ
ΣΧΗΜΑ (ΠΑΡΑΛΛΗΛΟΓΡΑΜΟ)	ΜΕΓΑΛΟ	ΜΕΣΣΑΙΟ	ΜΙΚΡΟ	a1	a2	
ΦΥΣΙΚΗ ΔΙΑΣΤΑΣΗ (mm)	85 x 95	50 X 35	30 x 10			
ΘΕΣΗ ΡΟΜ/ΚΟΥ ΠΑΙΚΤΗ ΔΕΙΓΜΑΤΟΛΗΨΙΑΣ	(εικονοστοιχεια)					
ΚΕΝΤΡΟ ΕΥΘΕΙΑ	347	63	13	0.1816	0.2063	0.8798
>> ΔΙΑΓΩΝΙΑ	356	67	15	0.1882	0.2239	0.8406
ΠΑΝΩ ΔΕΞΙΑ ΓΩΝΙΑ	375	79	21	0.2107	0.2658	0.7925
>> ΔΙΑΓΩΝΙΑ	365	81	23	0.2219	0.2840	0.7815
>> ΑΝΑΣΤΡΟΦΑ 180 ΜΟΙΡΕΣ	363	79	20	0.2176	0.2532	0.8596
ΚΑΤΩ ΔΕΞΙΑ ΓΩΝΙΑ	372	87	23	0.2339	0.2644	0.8846
>> ΔΙΑΓΩΝΙΑ	378	91	24	0.2407	0.2637	0.9128
ΟΡΙΑ ΤΙΜΩΝ ΜΕ THRESHOLD=0.60						**<2**

4.2.3.6.β Αναγνώριση μπάλας.

Η ποιο απλή αναγνώριση είναι της μπάλας όπου δεν μοιάζει καθόλου λόγο μεγέθους με τους ρομποτικούς παίκτες λόγω κανονισμών.

Χρησιμοποιείται μία σάρωση όλως των αντικειμένων από την ήδη εισερχόμενη εικόνα με =0.1, όπου εμπειρικά φαίνεται η μπάλλα με το πορτοκαλί χρώμα στις ανάλογες συνιθηκες φωτεινότητας με την παρακάτω εντολή .

[B,L] = bwboundaries(im2bw(pic,0.1),'noholes');

Εδώ για την μπάλλα πρέπει να κάνουμε σύγκριση για το χρώμα και για το μέγεθος που έχουν βρεθεί από το τμήμα αρχικοποίησης. Επειδή η μπάλλα είναι κυκλική, ελέγχουμε το σχήμα της με την παρακάτω εντολή

abs(bstats(objind).Eccentricity-0.5)<0.2)

ενώ για τον έλεγχο μεγέθους της χρησιμοποιούμε την παρακάτω εντολή

(bstats(objind).BoundingBox(3) > Ballsize*0.5) && (bstats(objind).BoundingBox(3) < Ballsize*2))

4.2.3.7. Εύρεση χαρακτηριστικών μεταξύ ρομποτικού παίκτη, μπάλας και τερμάτων

Σε αυτή την παράγραφο θα εξηγήθεί ο ρόλος στις παρακάτω χαρακτηριστικές μεταβλητές καθώς και ο τρόπος εύρεσης τους: α) αποσταση ρομποτικού παίκτη από την μπάλλα, β) αποσταση ρομποτικού παίκτη από το αντίπαλο τέρμα, γ) απόσταση ρομποτικού παίκτη από το επιθυμητό σημείο τελικης θέσης της κίνησης, δ) απόσταση μπάλας από το τέρμα, στ) γωνία κλίσης ρομποτικού παίκτη σε σχέση με το τέρμα, ζ) γωνία κλίσης μπάλλας σε σχέση με το τέρμα, η) γωνία κλίσης ρομποτικού παίκτη και μπάλλας.

Η απόσταση **ρομποτικού παίκτη από την μπάλα** ονομάζεται στον κώδικα dest_rb. Είναι η απόσταση του κέντρου του ρομποτικού παίκτη (robox, roboy) μέχρι την απόσταση του κέντρου της μπάλλας (ballx,bally) αντίστοιχα. Βρίσκεται από την υποτείνουσα (dest_br) που σχηματίζεται το οθρογώνιο τρίγωνο μεταξύ τους , καθώς μπορεί να βρεθεί η σχετική τους γωνία ang_br. Αυτή η απόσταση χρησιμοποιείται για την ευθεία κίνηση προς τα εμπρός καθώς έχει και άλλες χρήσεις που θα αναφερθούμε στο **τμήμα κινήσεων,** ,όπως επίσης και η μεταβλητή γωνία ang_br, που αναλόγα την περίπτωση έχει σχέση με το είδος της επόμενης εντολής κίνησης που θα δωθεί.

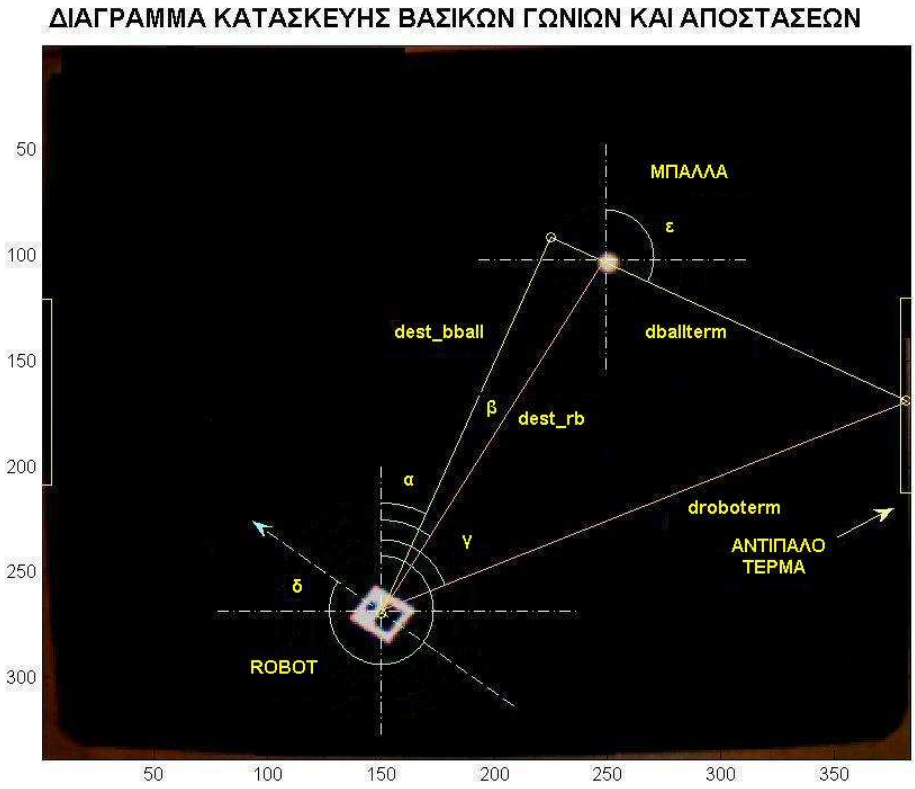
Η απόσταση **ρομποτικού παίκτη με το τέρμα** (στην εικόνα θεωρητική). Ονομάζεται στον κώδικα droboterm. Έιναι η απόσταση του κέντρου του ρομποτικού παίκτη (robox, roboy) μέχρι την απόσταση του κέντρου του δηλωμένου αντιπάλου τέρματος (termx, termy). Βρίσκεται από την υποτείνουσα (droboterm) που σχηματίζεται το οθρογώνιο τρίγωνο μεταξύ τους , καθώς μπορεί να βρεθεί και η σχετική τους γωνία angr. Αυτή η απόσταση χρησιμοποιείται για την ευθεία κίνηση προς τα εμπρός καθώς έχει και άλλες χρήσεις που θα αναφερθώ στο **τμήμα κινήσεων.**

Η απόσταση **μπάλλα με το τέρμα** (στην εικόνα θεωρητική) ονομάζεται στον κώδικα dballterm. Έιναι η απόσταση του κέντρου της μπάλλας (ballx, bally) μέχρι την απόσταση του κέντρου του δηλωμένου αντιπάλου τέρματος (termx, termy). Βρίσκεται από την υποτείνουσα (dballterm) που σχηματίζεται το οθρογώνιο τρίγωνο μεταξύ τους , καθώς μπορεί να βρεθεί η σχετική τους γωνία angb. Αυτή η απόσταση χρησιμοποιείται για την ευθεία κίνηση προς τα εμπρός καθώς έχει και άλλες χρήσεις που θα αναφερθώ στο **τμήμα κινήσεων,** ,όπως επίσης και η angb που αναλόγα την περίπτωση έχει σχέση με το είδος της επόμενης εντολής κίνησης που θα δοθεί.

Η απόσταση **ρομποτικού παίκτη από το σημείο προετοιμασίας της επόμενης εντολής κίνησης** ονομάζεται στον κώδικα dest_bball. Είναι η απόσταση του κέντρου του robot (robox, roboy) μέχρι την απόσταση του σημείου που σχηματίζεται από το (hhx, hhy). Βρίσκεται από την υποτείνουσα (dest_bball) που σχηματίζεται το οθρογώνιο τρίγωνο μεταξύ τους , καθώς μπορεί να βρεθεί η σχετική τους γωνία angn από τον άμεσο υπολογισμό. Η γωνία angs , μας δίνει την κλίση που πρέπει να έχει ο ρομποτικός παίκτης , ώστε να εκτελεστεί η εντολή της ευθείας κίνησης , σύμφωνα πάντα με την θέση της μπάλλας , την θέση του αντίπαλου τέρματος και την θέση που βρίσκεται ήδη ο ρομποτικός παίκτης μέσα στο γήπεδο.

Αυτή η απόσταση είναι η επιθυμητή απόσταση , που πρέπει να διανύσει ο ρομποτικός παίκτης, ώστε να βρεθεί σε εκείνο το σημείο που ορίζεται με (hhx , hhy) μέσα στον κώδικα. Αυτό το σημείο είναι συνήθως πίσω από την μπάλα και σε σπάνειες περιπτωσεις σε σχετική ευθεία με αυτή κατά τον άξονα το y, όπου θέση πίσω από την μπάλλα , ορίζεται η θέση η οποία βρίσκεται και πίσω από το αντίπαλο τέρμα σύμφωνα με τις τιμές των κέντρων τους μέσα στο γήπεδο, δηλαδή hhx < ballx < termx σύμφωνα με την αρίθμηση των εικονοστοιχείων της εικόνας.

Ανάλογα του μεγέθους και άλλων παραμέτρων, μπορεί να γίνει κίνηση ευθείας ή περιστροστροφής. Επίσης η γωνία angs χρησιμοποιείται πάντα και έμεσα για τον υπολογισμό της κλίσης της επόμενης κίνησης που έχει το όνομα angn. Περισσότερες λεπτομέρειες θα αναφερθούν στο **τμήμα κινήσεων.** Στην παρακάτω εικόνα (ε4.38) κατασκευάζονται οι γωνίες και οι αποστάσεις που αναφέρθηκαν.



(ε4.38)

Στους παρακάτω πίνακες φαίνονται περιληπτικά τα ονόματα των παραπάνω μεταβλητών, γωνιών και αποστάσεων.

Όνομα μεταβλητής στην εικόνα	Όνομα μεταβλητής στην εφαρμογή	σύντομη επεξήγηση μεταβήτής	Τιμή μεταβλητής στο συγκεκριμένο παράδειγμα (σε μοίρες σύμφωνα με τους γενικούς άξονες)
α	angs	επιθυμητή κλίση robot / μπάλας	22.88º
β	ang_br	πραγματική διαφορα γωνιας robot/μπάλας	31.18º
γ	angr	κλίση robot / αντίπαλο τέρμα	66.52º
δ	klisi	πραγματική κλήση robot σε σχεση με τους γενικους αξονες	305.57º
ε	angb	κλίση μπάλας / αντίπαλο τέρμα	116.23º

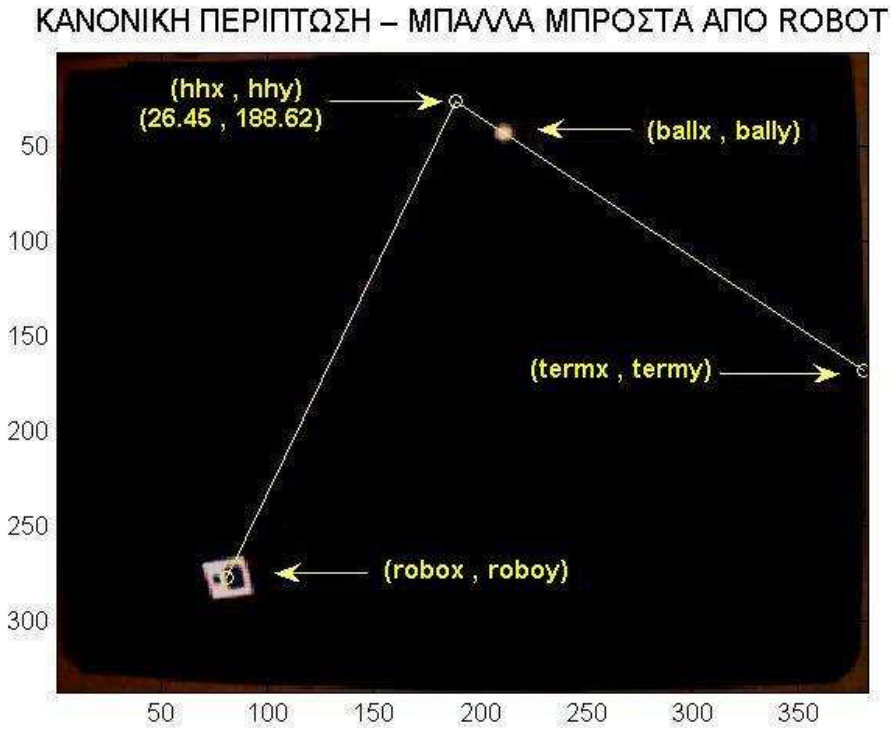
Όνομα μεταβλητής στην εφαρμογή	σύντομη επεξήγηση μεταβήτής	Τιμή μεταβλητής στο συγκεκριμένο παράδειγμα (σε εικονοστοιχεία)
dest_bball	απόσταση robot από την επόμενη επιθυμητή θέση του	193.28
dballterm	απόσταση μπάλλας από το κέντρο του αντίπαλου τέρματος	146.88
droboterm	απόσταση robot από το κέντρο του αντίπαλου τέρματος	252.95
dest_rb	πραγματική απόσταση robot με μπάλα	193.67

Σάν επισήμανση όλες οι τιμές σε εικονοστοιχεία αποστέλλονται στον κάθε ρομποτικό παίκτη και μετατρέπονται σε mm πρίν την εκτέλεση κίνησης , κατά τη ροή του προγράμματος.

4.2.3.8. Αναφορά στα είδη κινήσεων και τρόπος υπολογισμού στο πρόγραμμα ελέγχου

Το πόσο και προς τα πού θα κινηθεί ο ρομποτικός παίκτης , εξαρτάται από τα δεδομένα που έχουμε συλέξει κατά την διάρκεια της αναγνώρισης των αντικειμένων και από την στρατηγικη της συγκεκριμένη ςεφαρμογής που έχουμε ορίσει ανάλογα τι συνθήκες παιχνιδιού.

Ο πρώτος υπολογισμός που γίνεται, και ακολουθεί πάντα τους κανόνες της στρατηγικής της εφαρμογής, είναι η εύρεση μίας καλής θέσης πίσω από την μπάλλα ή η εκτέλεση κάποιας ειδικής περίπτωσης στρατηγικής. Η επιθυμητή θέση κίνησης , του ρομποτικού παίκτη , αποθηκεύετα στις συντεταγμένες (hhx, hhy). Αυτές για να υπολογιστούν είναι συνάρτηση, της γεωγραφικής θέσης του ρομποτικού παίκτη , της θέσης της μπάλας, της θέσης τους σε σχέση με το αντίπαλο τέρμα , και συνεπάγεται όλων των κλίσεων που έχουν αυτά μεταξύ τους. Πρίν από τον υπολογισμό τους υπάρχουν κάποιοι βοηθητικού συντελστές που υπολογίζονται και ονομάζονται mlx , mly και mmlx, mmly, οι οποίοι περιλαμβάνουν μία τιμή για τον κάθε γενικό άξονα της εφαρμογής. Παράδειγμα κανονικής περίπτωσης του υπολογισμού των (hhx, hhy) της παρακάτω εικόνας (ε4.39) εμφανίζεται στον επόμενο πίνακα.



(ε4.39)

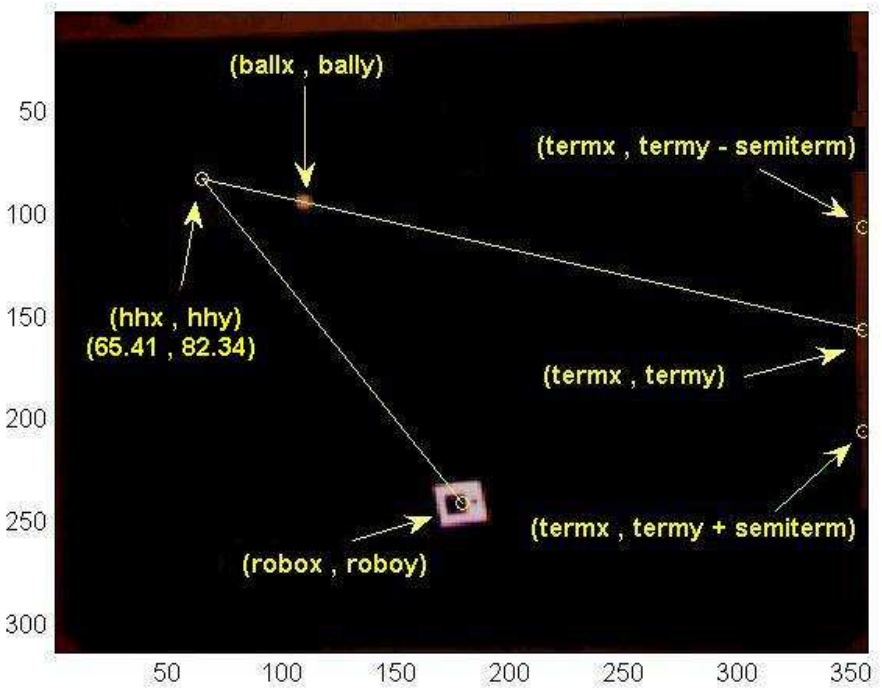
ΠΑΡΆΔΕΙΓΜΑ ΥΠΟΛΟΓΙΣΜΟΥ ΓΙΑ ΤΗΝ ΕΥΡΕΣΗ ΤΩΝ ΕΠΙΘΥΜΗΤΩΝ ΣΗΜΕΙΩΝ (hhx, hhy) ΚΑΝΟΝΙΚΗΣ ΠΕΡΙΠΤΩΣΗΣ – ΜΠΑΛΑ ΜΠΡΟΣΤΑ ΑΠΟ ROBOT		
Έλεγχοι	Διαδικασίες που εκτελούνται	παρατηρησεις

προηγούμενα δεδομένα	RoboLength, (robox, roboy) , (ballx , bally) ,
	bx=abs(ballx-termx);
	by=abs(bally-termy);	

επειδή MyTerm=1	Βρίσκουμε αντίστοιχα τις γωνίες angb , angr	Στο παράδειγμα αυτό δεν χρησιμοποιούνται άμεσα
ballx>robox	mlx=-1	
Ang_br>35	Mly=-1	
Λόγο των παραπάνω υπολογιζονται όπως δεξια	hhy=bally+((RoboLength*cosd(atand(bx/by)))*mly) => **hhy=26.45** hhx=ballx+((RoboLength*sind(atand(bx/by)))*mlx) => **hhx=188.62**	

Η αντίστροφη περίπτωση που έχω αναφέρει είναι όταν η μπάλλα είναι πίσω από το ρομποτικό παίκτη σε σχέση με το αντίπαλο τέρμα που φαίνεται στην παρακάτω εικόνα (ε4.40) και υπολογίζεται στον επόμενο πίνακα.

ΕΠΕΞΗΓΗΣΗ ΒΑΣΙΚΩΝ ΟΝΟΜΑΤΩΝ ΣΥΝΤΕΤΑΓΜΕΝΩΝ ΕΦΑΡΜΟΓΗΣ



(ε4.40)

ΠΑΡΆΔΕΙΓΜΑ ΥΠΟΛΟΓΙΣΜΟΥ ΓΙΑ ΤΗΝ ΕΥΡΕΣΗ ΤΩΝ ΕΠΙΘΥΜΗΤΩΝ ΣΗΜΕΙΩΝ (hhx, hhy) ΑΝΤΙΣΤΡΟΦΗΣ ΠΕΡΙΠΤΩΣΗΣ – ΜΠΑΛΑ ΠΙΣΩ ΑΠΟ ROBOT		
Έλεγχοι	Διαδικασίες που εκτελούνται	παρατηρησεις
προηγούμενα δεδομένα	Όπως παραπανω πίνακα	
	mlx=-1	
ballx<robox και bally<roboy	mmly=(cosd(angb)*2*RoboLength) mmlx=(sind(angb)*2*RoboLength) mmlz=+1	

KK=0.6782	Αν abs(cosd(angb)-cosd(angr)) < 0.55 τότε
	mmly=RoboLength*mmlz;
	mmlx=RoboLength*1.5;	

Λόγο των παραπάνω υπολογιζονται όπως δεξια	hhy=bally+mmly => **hhy=65.41** hhx=ballx-abs(mmlx) => **hhx=82.34**	

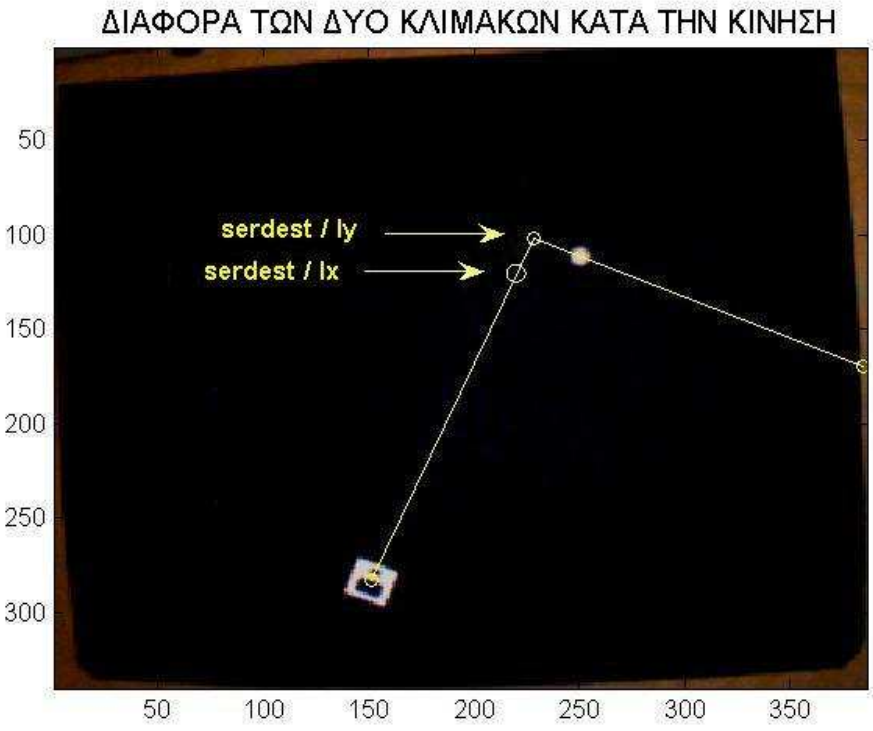
Μετά έχοντας βρεί τα επιθυμητά σημεία που θα κινηθούμε ακολουθεί ο υπολογισμός της φυσικής κίνησης. Οι αποστάσεις μετρουνται σε εικονοστοιχεία πάντα και πολλαπλασιάζονται με έναν συντελεστή ο οποίος αλλάζει όπως φαίνεται παρακάτω στην εικόνα (ε4.41).

Οι κινήσεις που έχουμε ενεργοποιήσει στην εφαρμογή εδώ είναι η ευθεία κίνηση και η περιστροφική κίνηση που χωρίζεται στην δεξιά και αριστερή.

4.2.3.8α Δύο κλίμακες κατά την ευθεία κίνηση

Στην διαδικασία αρχεικοποίησης λαμβάνουμε τις μέγιστες τιμές του άξονα x και y του γηπέδου σε mm και διαιρώντας τους δύο άξονες x και y αντίστοιχα των εικονοστοιχείων που απεικονιζεται το γήπεδο , θα έχουμε δύο λόγους . Για παράδειγμα , στο ύψος των 2m που αναρτάται η κάμερα έχουμε όρια γηπέδου από την εικόνα περίπου με limx=382 και limy=340 εικονοστοιχεία.

Οπότε διαιρώντας της διαστάσεις που έχουμε βρεί σαν όρια με τις φυσικές διαστάσεις που μετράμε ανα άξονα, προκύπτουν lx=limx/150=2.55 και ly=limy/130=2.60 εικονοστοιχεία / cm, όπου παρατηρούμε ότι οι δύο κλιμακες δεν είναι ακριβώς οι ίδιες και υπάρχει μία μικρη διαφορά που φαίνεται για παράδειγμα στην παρακάτω εικόνα (ε4.41):



(ε4.41)

Αν υπήρχε μόνο μία κλιμακα η lx, όπως φαίνεται στο παράδειγμα , ο ρομποτικός παίκτης δεν θα πήγαινε στην καθορισμένη θέση που βρίσκεται σε σχετική ευθεία με το τέρμα αλλα λίγο ποιο χαμηλά με αποτέλεσμα να χρειαζόταν άλλη μία εκτέλεση εντολής για την διόρθωση της τελικής θέσης . Αυτό το έχουμε λάβει υπόψη μας, οπότε χρησιμοποιούμε και τις δύο κλιμακες αναλόγως την γωνία κλίσης. Αυτό μπορεί να οφείλετε στην κάμερα η οποία κατά τον άξονα τον x επειδή έχει μέγαλυτερη δυνατότήτα στην ανάλυση 640 παρά στον y που έχει 480. Μπορεί επίσης λόγο σφάλματος η κάμερα να μην είναι ακριβώς κάθετη με την επιφάνεια, όπως επίσης έχουμε παρατηρήσει ότι για μία ίδια απόσταση εάν μετρηθεί 5 φορές έχουμε συνήθως διαφορετικές μικρες αποκλίσεις.

Αυτή η μικρή τιμή διαφοράς δεν παίζει σημασία σε μικρές τιμές αλλα σε μεγάλες κινήσεις και σίγουρα σε κλισεις μεγαλυτερες από 20 μοίρες σε σχέση με το τέρμα (δηλ 90 ή 270 αναλόγος το τέρμα μοίρες +-20).

4.2.3.8.β Ευθεία κίνηση

Πρέπει να διαχωρήσουμε ότι η ευθεία έχει διαφορετικές τιμές όταν ο ρομποτικός παικτης είναι μπροστά από την μπάλλα ή πίσω από την μπάλλα και επομένως ακόμα ποιο μπροστά από το τέρμα.

Η ευθεία κίνηση όταν ο παίκτης είναι μπροστά από την μπάλλα, έχει κλίση προς την υπολογισμένη θέση (hhx,hhy) , δηλαδή προώηση ώστε να βρεθεί ο ρομποτικός παίκτης πίσω και σε μία συγκεκριμένη θέση κοντά ή ακριβώς πάνω στην μπάλλα, εκτός εάν υπάρχει κάποια διαφορετική στρατηγική.

Η ευθεία κίνηση όταν ο παίκτης είναι πίσω από την μπάλλα , έχει τέτοια κλίση ώστε , να βρεθεί σε μία νοητή ευθεία με την μπάλλα και το τέρμα , για να μπορεί επιτευχθεί γκόλ , εάν απλά περιστραφεί ο ρομποτικός παίκτης ορισμένες μοίρες. Εξαίρεση αποτελεί επίσης, εάν η στρατηγική έχει διαφορετική κίνηση.

Για την κίνηση της ευθείεας , η μέγιστη κίνηση που μπορεί να γίνει στο συγκεκριμένο μέγεθος γηπέδου είναι η διαγώνια του που βρίσκεται από

$$\sqrt{(150cm)^2 + (130cm)^2} = 198.5cm$$

και η ελάχιστη επιτρεπόμενη μπορεί να είναι το 1cm., αλλα χρησιμοποιούμε ως ελάχιστη κίνηση ευθείας την απόσταση Robolength/2 , δηλαδή το μήκος του ρομποτικού παίκτη / 2 .

Το πρόγραμμα πρέπει να μπορεί να διαχειριστεί αυτή την μέγιστη κίνηση για να μην βρεθούμε εκτός ορίων στοιυς υπολογισμοούς, και πρέπει να φτιάξουμε μια κλίμακα αντιστοίχησης εινονοστοιχείων / cm.

Για να ωθείτε η μπάλλα μπροστά , χρειάζεται ο ρομποτικός παίκτης να προωθείτε λίγο παραπάνω από την απόσταση της μπάλας. Στον κώδικα υπάρχει το τμήμα όπου υπολογίζονται το πόσο θα έιναι ένα βήμα της ευθείας ή της προώθησης ανάλογα την περιπτωση.

Ο παρακάτω πίνακας εμφανίζει τις περιπτώσεις και ποια είναι αντίστοιχα η προώθηση του κάθε ρομποτικού παίκτη ανάλογος σε εικονοστοιχεία .

Περιπτώσεις ευθείας κίνησης	ώθηση μπάλας (σε εικονοστοιχεία επιπλέων) min_strmov
Ρομποτικός παίκτης σε σχετική ευθεία με μπάλλα και τέρμα, με κλίση προς την μπάλλα κοντά με το αντίπαλο τέρμα - Επίτευξη γκόλ – mul=1	(dest_rb/2)+Robolength;
Ρομποτικός παίκτης σε σχετική ευθεία με την μπάλλα και το αντίπαλο τέρμα, με κλίση προς την μπάλλα αλλα μακρυά από το τέρμα mul=1	Robolength/2
Ρομποτικός παίκτης όχι σε σχετική ευθεία με την μπάλλα , αλλα σε σχετική ευθεία με προς τα σημεία (hhx,hhy) mul=1	Robolength/2
Ρομποτικός παίκτης όχι σε σχετική ευθεία με την μπάλλα , και όχι σε σχετική ευθεία με με τα σημεία (hhx,hhy) mul=1	dest_bball
Ρομποτικός παίκτης κοντά από τα σημεία (hhx,hhy) mul=0	abs(robox-ballx)
Ρομποτικός παίκτης μακρυά από τα σημεία (hhx,hhy) mul=0	abs(robox-hhx)*0.7

Υπάρχει στον κώδικα το τμήμα που η τελική εντολή κίνησης θα είναι serdest= min_strmov σε εικονοστροιχεία και μετά με την σειρά του μεταφράζεται σε mm με διαφορετικό αλγόριθμο.

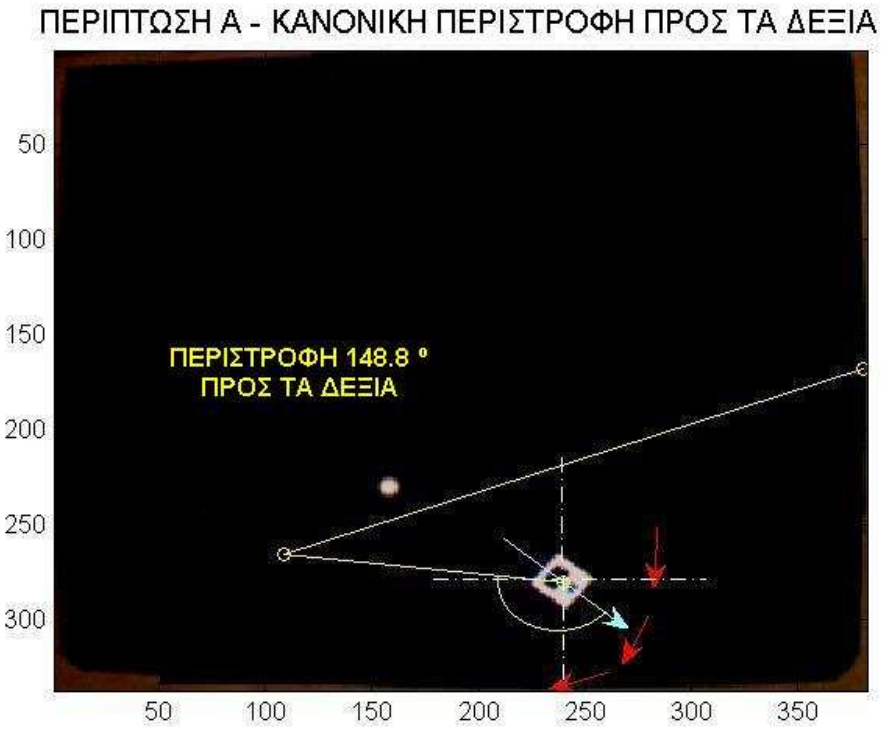
Σε κάθε αύξηση του strmov ο ρομποτικός παίκτης χρυπάει ή ωθεί την μπάλλα μεμεγαλύτερη ταχύτητα, αλλα πρέπει να είναι ελεγχόμενο επειδη υπάρχουν και τα όρια του γηπέδου ή τα τέρματα ή η συγκεκριμένη στρατηγική.

Η ώθηση χρησιμοποιείται μόνο στην κίνηση ευθείας και όχι στην περιστροφική. Στο τέλος η τιμή που μεταφέρω στην μεταβλητή serdest, διαιρεμένη με τον συντελεστή που έχω επιλέξει μετατρέπεται σε mm και είναι αυτή που θα αποσταλλέι στν ρομποτικό παίκτη, δηλαδή στον κώδικα έχουμε serdest/angdiv, όπου angdiv=lx ή angdiv=ly, ανάλογα.

4.2.3.8.γ Περιστροφική κίνηση

Για την περιστροφική κίνηση η μέγιστη κίνηση που μπορεί να γίνει είναι οι 180 μοίρες, δηλαδή 180 δεξιόστροφα ή 180 αριστερόστροφα. Πρίν την εκτέλεση περιστροφικής κίνησης ελέγχεται ποια είναι η ποιο κοντινή κίνηση , δηλαδή με τις λιγότερες μοίρες περιστροφής για την επιθυμητή κλίση και επιλέγεται να εκτελεστεί η συγκεκριμένη.

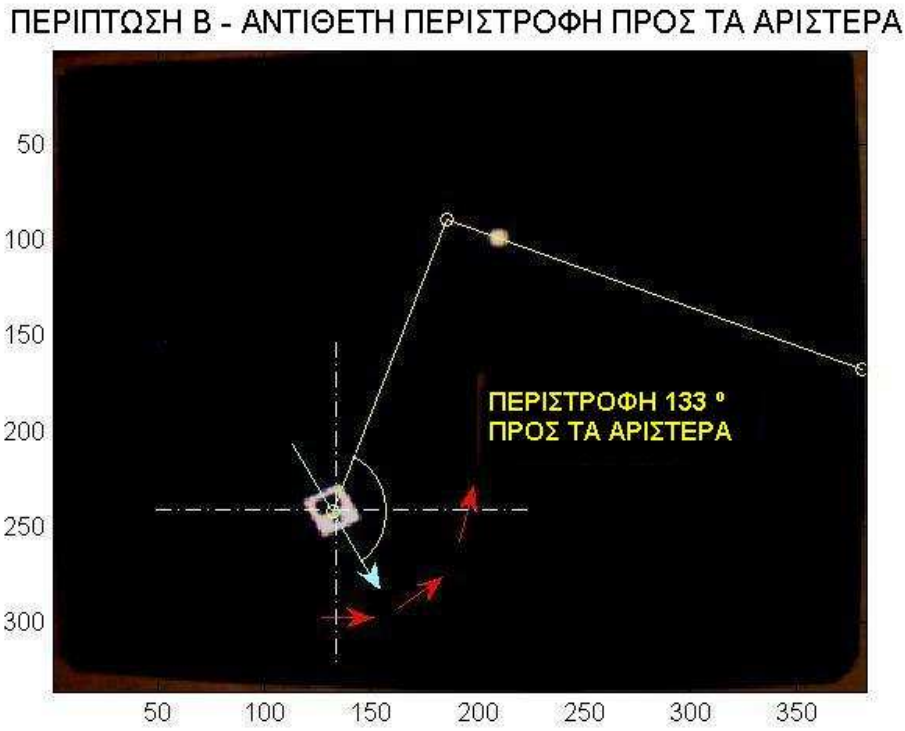
Η πρώτη περίπτωση είναι η κανονική περιστροφή , δηλαδή περιστρφή προς τα δεξιά και παράδειγμα φαίνεται στην παρακάτω εικόνα (ε4.42).



(ε4.42)

Εδώ έχουμε κλίση ρομποτικού παίκτη klisi=127.4 º , επιθυμητή κλίση ρομποτικού παίκτη angs=276.2 º, οπότε η διαφορα που έχουν είναι angd=148,8 º, και η εντολή περιστοφής είναι δηλαδή serang=148,8 º. Αυτή είναι η τιμή που θα κωδικοποιηθεί για να αποσταλεί στον αντίστοιχο ρομποτικό παίκτη..

Στην δεύτερη περίπτωση έχουμε αντίθετη περιστροφή, την αρνητική κίνηση σε σχέση με τους γενικούς άξονες και παράδειγμα φαίνεται παρακάτω στην εικόνα (ε4.43).



(ε4.43)

Εδώ έχουμε κλίση ρομποτικού παίκτη klisi=152.3 º , επιθυμητή κλίση ρομποτικού παίκτη angs=19.3 º, οπότε η διαφορα που έχουν είναι είναι angd=133 º, και η εντολή περιστοφής είναι 360-angd -> δηλαδή serang=227 º. Σε αυτές τις περιτπώσεις χρησιμοποιείτε η αρίθμηση από 360 º – 180 º, που σημαίνει 0 – 180 μοίρες αντίστοιχα αριστερόστροφα, για λόγους ευκολίας στην αναπτυξη του κώδικα. Αυτή είναι η τιμή που θα κωδικοποιηθεί για να αποσταλεί στον ρομποτικό παίκτη .

Ο παρακάτω πίνακας εμφανίζει τις περιπτώσεις και ποια είναι αντίστοιχα η προώθηση του κάθε ρομποτικού παίκτη ανάλογος σε εικονοστοιχεία .

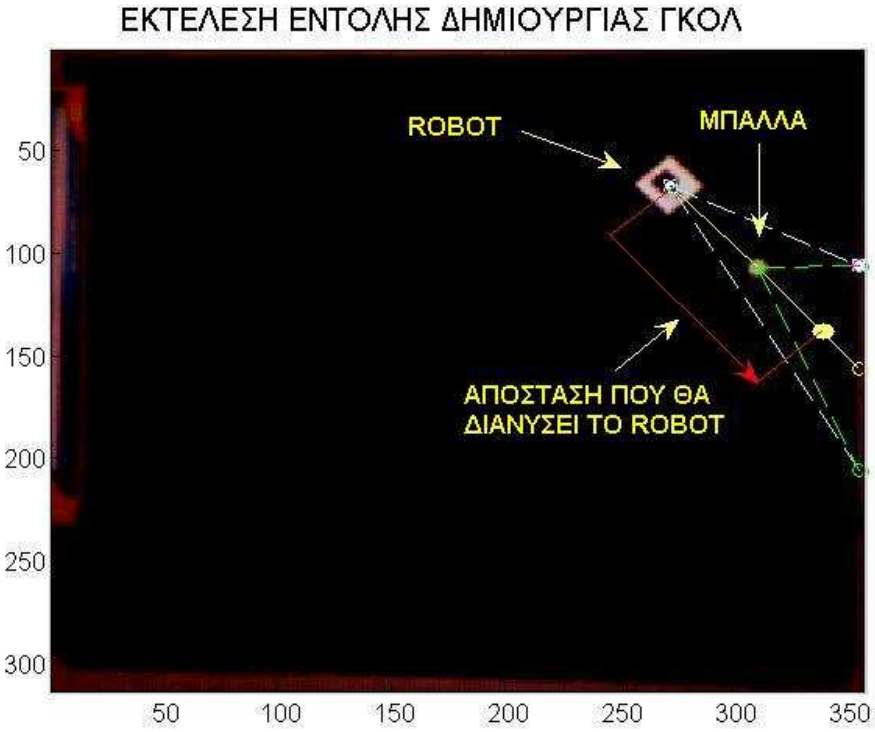
Περιπτώσεις περιστροφικής κίνησης	Περιστροφή (σε εικονοστοιχεία)
Σε όλες τις περιπτωσεις με mul=1 ή mul=0	Αν διαφορα επιθυμητής περιστροφής <60 τοτε serang=8 , διαφορετικά όση είναι η διαφορά serang=διαφορα

4.2.3.8.δ Πότε εχουμε εντολή εκτέλεσης γκόλ , πώς επιτυγχάνεται και πανηγυρισμός του

Ο στόχος του παιχνιδιού είναι τα γκόλ. Αυτή η εντολή ανοίκει στις ειδικές πρτιπτώσεις που σημαίνει ότι εκτελείτε άσχετα από τις υπόλοιπες καταστάσεις που συντρέχουν στο γήπεδο και για να έχουμε εντολή εκτέλεσης για γκόλ πρέπει να πληρύνται τα παρακάτω κρητήρια που βρίσκονται στον πίνακα.

Επεξήγηση Κρητήριων	Αντίστοιχα κρητηρία στο πρόγραμμα
Εάν η φορά του robot είναι προς το αντίθετο τέρμα	mul=1
Η νοητή ευθεία του ρομποτικού παίκτη , , της μπάλλας , και του τέρματος να είναι οκ	ang_obj_ok
Η κλιση του ρομποτικού παίκτη να είναι προς το το τέρμα	(klisi<ang_temp(1)) && (klisi>ang_temp(2))
Η μπάλα να είναι σχετικά στην επιτρεπόμενη περιοχή εκτέλεσης σούτ δηλαδή σχετικά κοντά	dballterm<goaldistance

Όταν γίνει η εκκίνηση αυτής της εντολής , η τιμή της ευθείας που θα διανήσει το robot είναι η min_strmov=(dest_rb/2)+Robolength;, δηλαδή θα κινηθεί όσο τα μισό της απόστασης του ρομποτικου παίκτη + το μέγεθος του συνολικά σε απόσταση.ώστε λίγο πριν το αντιπαλο τέρμα , χτυπώντας και συμπαρασύροντας την μπάλλα προς τα εκεί. Στην παρακάτω εικόνα (ε.4.44) φαίνεται περιπου η κίνηση που θα επιτευχθεί.



(ε4.44)

Οι άσπρες γραμμες δείχνουν την γωνία που σχηματίζεται από το κέντρο του ρομποτικού παίκτη, με τα άκρα του τέρματος και η πράσινη διακεκομένη γραμμή δείχνει ότι είναι σε απόσταση βολής και έχει ενεργοποιηθεί η εκτέλεση για γκολ. Οι τιμές στο παραπάνω παράδειγμα είναι : droboterm=121.91 , dest_bball=53.99 , min_strmov=38.38 , serdest=93.78 εικονοστοιχεία. Δηλαδή η απόσταση που έχει ο ρομποτικός παίκτης από το τέρμα droboerm, η απόσταση που θα αποσταλεί για εκτέλεση είναι μικρότερη όπως φαινεται στην serdest.

Η τιμή ενεργοποιίησης αποστολής κινήσεων εντολών θα γίνει moveenable=1 και η μεταβλητή togoal=2, που δείχνει ότι εκτελείτε η ειδική περίπτωση επίτευξης γκόλ. Στην επόμενη επανάληψη κάνουμε το moveenable=0 για να μην γίνει καμία κίνηση και περιμένουμε να μπεί η μπάλα μέσα στο τέρμα για να γίνει το splineBall=0, που σημαίνει ότι δεν υπάρχει μπάλλα στο γήπεδο, όταν δεν αναγνωριστεί μετά από λίγο χρόνο. Όταν γίνει αυτό και μετά απο 20 επαναλήψεις συνεχής μη αγννώρισης μπάλλας, τότε γίνεται το togoal=3, το οποίο με την σειρά του στην επόμενη επανάληψη εκτελεί μία περιστροφική κίνηση σαν πανυγηρισμό, και αθροίζεται το σκορ.

Το παιχνίδι ξαναρχίζει όταν εισάγουμε μία μπάλλα μέσα στον χώρο του γηπέδου και επαναλαμβάνεται η γενική διαδικασία επανάληψης.

4.2.3.9 Ανοχές και χρήση τους.

Στους περισσότερους ελέγχους χρησιμοποιούμε άνοχές, επειδή κατά την κίνηση αλλάζουν γρήγορα πολλά δεδομένα

1. ColorArray (1-4) που περιλαμβάνουν τις τιμές των χρωματικών παλετών του ρομποτικού παίκτη και της μπάλας. Επειδή κατά την διαδικασία της αρχικοποίησης λαμβάνουμε μία τιμή (rgb) για τον ρομποτικό παίκτη όπως επίσης και για την μπάλλα, σε ένα συγκεκριμένο σημείο του γηπέδου, δεν μπορώ να έχω σε όλο το εύρος του, τις ίδιες τιμές λόγο ανομοιογένειας φωτινότητας , το οποίο θα έχει σαν αποτέλεσμα, να μην μπορώ να αναγνωρίσω τα αντικείμενα σε χρωματικό έλεγχο. Για αυτό το λόγο χσησιμοποιειτε η ανοχή ColorAnnot=33 , που δημιουργεί ένα εύρος χρωμάτων +/- ColorAnnot, από την τιμή που αντιστοιχεί στο αντικείμενο. Για παράδειγμα εάν είχα ένα χρώμα (30,40,150) με την επεξεργασία, ο έλεγχος που θα είχα θα ήταν (0,0,100) < (30,40,150) < (80,90,200), οπότε θα μπορούσα να αναγνωρίσω τον ρομποτικό παίκτη.

2. RoboLength και BallLength είναι τα μεγέθη που έχουν τα γεωμετρικά στοιχεία των αντίστοιχων αντικειμενων. Πάλι κατά την αναγνώριση στην αρχικοποίηση λαμβάνουμε τα γεωμετρικά μεγέθη του κάθε αντικειμενου σε εικονοστοιχεία στο συγκεκριμένο σημείο. Εάν στο σημείο αυτό είναι το κέντρο που βρίσκεται θεωρητικά η κάμερα εντελώς κάθετα, τότε τα αντικείμενα έχουν τις μέγιστες διαστάσεις που μπορούν να πάρουν σε όλο το χώρο του γηπέδου. Όσο αποκλίνουν από το κέντρο , τότε οι διαστάσεις μικραίνουν αναλογικά σε κάθε γεωμετρικό τους στοιχείο. Αυτές τις ανοχές τις εκφράζουμε με σταθερούς αριθμούς ([πολαπλασιαστές) π.χ 1.2. Για παράδειγμα εάν έχουμε μέγεθος ρομποτικού παίκτη robolength=28, τότε με την εφαρμογή της ανοχής θα έχουμε τον έλεγχο (RoboLength/1.2 < RoboLength < RoboLength*1.2), αντίστοιχα για την μπαλα (BallLength/1.2 < BallLength < BallLength*1.2). Στην περίπτωση που δεν χρησιμοποιουσαμε τις ανοχές δεν θα μπορούσαμε να αναγνωρίσουμε τα αντικείμενα , για αυτό τον λόγο προτείνεται κατά την αναγνώριση να βρίσκονται τα αντικείμενα περιμετρικα περίπου στο κέντρο του γηπέδου.

3. Ανοχές χρησιμοποιούνται επίσης κατά την αναγνώρηση των ρομποτικών παικτών οσον αφορά ποιος είναι ο κάθε δικός μας παίκτης. Κάθε φορά που χρησιμοποιείται το Threshold και ανάλογα με το αν είναι μεγαλύτερο ή μικρότερο σε τιμή , τα εσωτερικά τετράγωνα αναγνωρισης των ρομποτικών παικτών αλλάζουν , οπότε χρησιμοποιούνται ανοχές για να δούμε εάν είναι μέσα στα όρια.

Καθώς το ρομποτικός παίκτης κινείτε μπορεί να παρεκλίνει λίγο της πορείας του για λόγους που αναφέρω στο τμήμα γενικών προβλήματων και δεν μπορεί να υπάρξει η απόλυτη ακρίβεια, οπότε συγχρόνως παρεμβάινει η εντολή κλίσης , ώστε να διορθώσει την κλίση.

Γενικά αν δεν θα χρησιμοποιούσαμε ανοχές πιθανό να υπήρχε μεγάλη καθυστέρηση στην ανάδραση όλου του συστήματος.

4.2.3.10 καθορισμός της επόμενης εντολής ανάλογα της κατάστασης των αντικειμένων

Επειδή το πρόγραμμα εκτελείτε γρήγορα, τουλάχιστον 30 φορές το δευτερόλεπτο, η ροή πληροφοριών είναι μεγάλη και όσο ποιο γρήγορα γίνεται τόσο ποιο άμεσα πρέπει να επεμβαίνουμε με εντολές για να κάνουμε ανάδραση.για παράδειγμα όταν βρίσκεται ο ρομποτικός παίκτης σε ακτίνα βολής, η αμέσως επόμενη εντολή είναι η ευθεία κίνηση εάν πληρεί τις απαιτούμενες προυποθέσεις για εκτέλεση προς το τέρμα και να επιτύχει

γκόλ. Το πρόγραμμα έχει δημιουργηθεί με τετοιο τρόπο ώστε να δίνεται προτεραιότητα στις ειδικές περιπτώσεις , ως παράδειγμα την εκτέλεση γκολ.

Σε ολες τις περιπτώσεις , όποτε πληρήται μία κατάσταση , τότε αποστέλλεται μία εντολή σειριακή , άσχετα εάν έχει κινηθει από παρόποια , πριν από ελάχιστο χρόνο.

Περίπτωση που αλλάζει η προκαθορισμένη θέση του (hhx,hhy) είναι όταν οι τιμές πάνε να βγαινουν οριακά έκτός γηπέδου ή όταν είναι πολύ οριακές και δεν μπορεί να προχωρήσει ο ρομποτικός παίκτης.

4.2.3.11 Τρόπος αποστολής δεδομένων προς τον ρομποτικό παίκτη και κωδικοποίηση τους.

Σε αυτή την εφαρμογή τα δεδομένα που στέλνωνται μεσω της σειριακής θύρας ασύρματα είναι ένας πίνακας ο serdata που παριλαμβάνει: την τιμή serdest που είναι η επιθυμητή απόσταση κήνησης της εντολής ευθείας και η τιμή serang που είναι η διαφορά κλήσης που πρέπιει να περιστραφεί ο ρομποτικός πάικτης για να επιτύχει την επιθυμητή γωνία. Κατά την κατασκαευή της κωδικοποίησης έγινε η σκέψη να χρησιμοποιηθεί και αλλαγή ταχύτητας οπότε έχει προστεθεί η επιπλέων μεταβλητη που μπορεί να αποσταλλεί , αλλα δεν εχει αξιοποιηθεί κατά την λειτουργία.

Οι έγκυρες τιμές που μπορεί να έχουν η serdest είναι από 0 - 2000 mm, αλλα το δυαιρουμε δια 2 για λόγους κωδικοποίησης ορίων και επαναφέρεται σωστά στην αποκωδικοποίηση στον ελεγκτή του ρομποτικού παίκτη, οπότε έχουμε τιμές από 0 – 999.

Στις γωνίες η μέγιστη τιμή που μπορεί να μεταφερθεί είναι και εδώ 999 μοίρες αλλά οι έγκυρες τιμές είναι από 0 μέχρι 360. Οπότε δεν υπάρχει τιμή που να μην μπορούμε να παραστήσουμε για την συγκεκριμένη εγκατασταση. Επειδη η σειριακή θύρα λαμβάνει μόνο bytes που μπορούν να αποθηκεύσουν τιμές από 0 – 256 (2^8) , είναι προφανές ότι έχουμε πρόβλημα με την αποστολή μεγαλύτερων τμών. Εδώ χησιμοποιουμε ένα αλγόριθμο που διαχωρίζει την επιθυμητή τιμή που πρέπει να σταλέι σε τέσσερα διαφορετικά bytes τα οποία στέλλωνται σαν γκρούπ και ξανασυνθέτωνται με ένα αντίστοιχο αλγόριθμο στην σωστή σειρά δημιουργώντας την σωστή τιμή από τον ελεγκτή του κάθε ρομποτικού παίκτη.

Η κωδικοποίηση περιγράφεται ως εξής: Στέλνονται πάντα σε απροσδιόριστη σειρά μία ολοκληρωμένη σειρά από bytes, τα τρία byte δεδομένων και ένα byte το οποίο ελέγχει την αποστολή. Όλα τα bytes δεδομένων είναι μικρότερα από 100, ενώ τα Bytes ελέγχου είναι πάντα μεγαλύτερα ή ίσα απο 100. Στους παρακάτω πίνακες φαίνεται η κωδικοποίηση που χρησιμοποιείται.

Ο πίνακας αυτός δείχνει ποιά είναι η τιμή που δίδεται σε κάποια εντολή κίνησης που χρειάζεται για την κατανόηση των παρακάτω:

Α/Α	ΚΑΤΗΓΟΡΙΑ ΚΙΝΗΣΗΣ	ΠΑΡΑΣΤΑΣΗ ΤΙΜΗΣ	ΒΟΗΘΗΤΙΚΟ ΧΑΡΑΚΤΗΡΙΣΤΙΚΟ
1	DEST= ευθεία κίνηση	0	Α
2	SPD= αλλαγή ταχύτητας	1	Β
3	ANG=περιστροφική κίνηση	2	Γ

Παράσταση πρώτου ψηφίου των byte που στέλνονται.
Το πρώτο ψηφίο είναι πάντα = 0 στα DB's (data bytes) και πάντα =1 στα CB's (control bytes)

Παράσταση δεύτερου ψηφίου των byte που στέλνονται.

Για την παράσταση του δεύτερου ψηφίου των DB's είναι απαραίτητος ο παρακάτω πίνακας :

ΠΙΝΑΚΑΣ DATA BYTES (DB's)			
Α/Α	ROBO ID	ΣΕΙΡΑ ΑΠΟΣΤΟΛΗΣ BYTE	ΠΑΡΑΣΤΑΣΗ ΤΙΜΗΣ
1	0	$1^{η}$	0
2	0	$2^{η}$	1
3	0	$3^{η}$	2
4	1	$1^{η}$	3
5	1	$2^{η}$	4
6	1	$3^{η}$	5
7	2	$1^{η}$	6
8	2	$2^{η}$	7
9	2	$3^{η}$	8

Να σημειώσουμε ότι η σειρά αποστολής byte είναι ποιο ψηφίο ενοούμε, π.χ για 586 σαν $1^{η}$ σειρά=5, $2^{η}$ σειρά=8, $3^{η}$ σειρά=6.

Για την παράσταση του δεύτερου ψηφίου των CB's είναι απαραίτητος ο παρακάτω πίνακας :

ΠΙΝΑΚΑΣ CONTROL BYTES (CB's)			
Α/Α	ROBO ID	ΤΥΠΟΣ (DEST,SPD,ANG)	ΠΑΡΑΣΤΑΣΗ ΤΙΜΗΣ
1	0	Α	0
2	1	Β	1
3	2	Γ	2
4	0	Α	3
5	1	Β	4
6	2	Γ	5
7	0	Α	6
8	1	Β	7
9	2	Γ	8

Να σημειώσουμε ότι ο τύπος αναφέρεται σε τι είδους κίνηση ανήκει αυτό το ένα byte ελέγχου. Δηλαδή τύπου Α σημαίνει ότι είναι byte ελέγχου αποστολής που ανοίκει σε κίνηση ευθείας DEST, οπότε γίνεται έλεγχος με τα υπόλοιπα bytes δεδομένων αν είναι σωστή η αποστολή.

Παράσταση τρίτου ψηφίου των byte που στέλνονται.

Αυτό το ψηφίο είναι στην ουσία το ψηφίο που μας ενδιαφέρει , δηλαδή στα τρία DB's έχουμε την τριψήφια τίμή που ανοίκει στην κίνηση ευθείας ή στην περιστροφική κίνηση. Το τελευταίο byte των CB's συμπληρώνεται από την απόλυτη τιμή της διαφοράς του $2^{ου}$ στην σειρά και του $3^{ου}$ στην σειρά DB's πρίν την αποστολή. Μετά την αποστολή αυτό το νούμερο του CB ελέγχεται με την διαφορά των άλλων δύο και εάν είναι σωστή , τότε η αποστολή έγινε επιτυχής και προχωράει η διαδικασία στον ελεγκτή του κάθε ρομποτικού παίκτη που θα περιγραφεί σε διαφορετικό σημείο.

Γενικά υπάρχουν πολλές τιμές οι οποίες δεν χρησιμοποιούνται. Μπορουν να χρησιμοποιειθούν για ειδικές χρήσεις όπως για απράδειγμα για τον πανυγιρισμό του ρομποτικού παίκτης ή για ένα reset στα δεδομένα η και για άλλες λειτουργίες.

Ένα παράδειγμα :

Εχουμε μια κίνηση ευθείας (DEST) δηλαδή serdest=118.78 εικονοστοιχείων. Η αποστολή αυτής της τιμής στο ρομποτικό πάικτη γίνεται από τα βήματα:

Μετατροπή τιμής σε mm και προετοιμασία τιμής για την κωδικοποίηση, όπου angdiv είναι η υποδιαίρεση εικονοστοιχεία/cm που χρησιμοποιώ ανάλογα την περίπτωση lx ή ly και έχουμε περγράψει σε προηγούμενη παράγραφο.

Serdata=round(((serdest/angdiv)*10)/2) -> 234

Η τιμή που θα κωδικοποιηθεί και θα αποσταλεί είναι η serdata=234.

Δημιουργούνται τα παρακάτω bytes από τον αλγόριθμο και αποθηκεύονται στον πίνακα serbyte:

$1^{ο}$ data byte=002 ($1^{ος}$ αριθμός παντα είναι = 0, $2^{ος}$ αριθμός από τον πίνακα DB's για ROBO ID=0 και βίσκεται στην πρωτη σειρά -> τότε = 0, ο $3^{ος}$ είναι αυτό που μας ενδιαφέρει δηλαδή το **2**.

$2^{ο}$ data byte=013 ($1^{ος}$ αριθμός παντα είναι = 0, $2^{ος}$ αριθμός από τον πίνακα DB's για ROBO ID=0 και βίσκεται στην δεύτερη σειρά -> τότε = 1, ο $3^{ος}$ είναι αυτό που μας ενδιαφέρει δηλαδή το 3.

$3^{ο}$ data byte=024 ($1^{ος}$ αριθμός παντα είναι = 0, $2^{ος}$ αριθμός από τον πίνακα DB's για ROBO ID=0 και βίσκεται στην τρίτη σειρά -> τότε = 2, ο $3^{ος}$ είναι αυτό που μας ενδιαφέρει δηλαδή το 4.

$4^{ο}$ control byte=101 ($1^{ος}$ αριθμός παντα είναι = 1, $2^{ος}$ αριθμός από τον πίνακα CB's για ROBO ID=0 , τύπο κίνησης = A -> τότε = 0, ο $3^{ος}$ είναι αυτό που μας ενδιαφέρει δηλαδή η απόλυτη τιμή της διαφοράς του τελευτείου ψηφίου των $2^{ου}$ =013 και $3^{ου}$=024 data byte δηλαδή | 3 – 4 | = 1.

Οπότε συνοπτικά η παράσταση της τιμής της κίνησης της ευθείας του ROBO ID=0 , για κωδικοποιημένη απόσταση 234 είναι η σειρά των bytes : 002 013 024 και 101.

Η αναπαράσταση αυτής της κίνησης φαίνεται στην εικόνα (ε.4.45):

(ε4.45)

Ο πινακας αποθηκεύσης των κωδικοποιημένων bytes που δημιουργούνται από τις παραπάνω διαδικασίες ονομάζεται serbyte και έχει πάντα μία σειρά από bytes ευθείας και μία σειρά περιστροφικής κίνησης.

Υπάρχει μία μεταβλητή που ονομάζεται sdc , η οποία παραλαμβάνει τον τιμή τέτοια ώστε να αναγνωρίζεται ποια κίνησης πρόκειτε να λειτουργήσει.

Αναλόγος την τιμή sdc, εκτέλείτε η αντίστοιχη κίνηση. Αν είναι sdc=2 τότε αποστέλεται η ευθεία κίνηση, διαφορετικά αν είναι sdc=4, τότε αποστέλεται η περιστροφική κίνηση. Για να αποσταλεί πληροφορία σειριακά πρέπει η σειριακή θύρα να έχει ίδια χαρακτηριστικά με την ρύθμιση που έχει γινει στον κάθε ρομποτικό παίκτη το οποίο γινεται στην την εκκίνηση του προγράμματος μία φορά με τηνεντολή !mode com1:9600,n,8,1, η οποία δίνει στο λειτουργικό σύστημα τους αντίστοιχους παραμέτρους για να ρυθμιστεί η σειριακή θύρα. Εδώ επιλεγουμε bitrate=9600, paritybit=none, dtatabit=8 και Η αποστολή μίας κίνησης χρειάζεται τις εντολές:
για το άνοιγμα της θύρας για εγγραφή που γίνεται πάντα σε κάθε αποστολή κάθε σειράς bytes

```
port = fopen('com1:','w');
```

μετά ακολουθεί η αποστολή κάθε byte με μία εντολη το κάθε ένα και είναι οι ακολουθες:

```
fwrite(port,serbyte(2,4));  control byte
```

```
fwrite(port,serbyte(2,1));  %  1ο data byte
fwrite(port,serbyte(2,2));  %  2ο data byte
fwrite(port,serbyte(2,3)); %  3ο data byte
```

Η όλη διαδικασία τελειώνει με το κλεισιμο της θύρας που γίνεται με την εντολή: fclose(port); Με τοκελίσιμο της θύρας αρχίζει η επεξεργασίας στο robot το οποίο θα αναύσω σε άλλο τμήμα.

4.2.3.12 Εμφάνιση εικόνας με τις ενδείξεις κινήσεων

Η εμφάνιση των υπολογισμένων επιθυμητών εμφανιζομένων κινήσεων γίνεται συνεχώς σε κάθε επανάληψη του προγράμματος όταν έχουμε το showstatus=1. Πάντα εμφανίζεται η επόμενη κλίση που πρέπει να έχει ο ρομποτικός παίκτης, όπως και η κλίση που έχει η μπάλα με το αντίπαλο τέρμα και είναι πολύ χρήσιμη η εμφάνιση τους για να ελεγχθεί εάν ο ρομποτικός παίκτης συμπεριφέρετα σωστά , σύμφωνα με τις εντολές και τα δεδομένα που φαίνονται σην οθόνη. Η επεξεργασία και η εμφάνιση γίνεται με γρήγορο H/Y σε συχνότητα 30 εικόνες τουλάχιστον το δευτερόλεπτο περίπου που είναι μία απαραίτητη ταχύτητα για την εμφάνιση των διαφόρων καταστάσεων θεωρητικά σε πραγματικό χρόνο. Σε περίπτωση που δεν θέλουμε εμφάνιση, κάνουμε το showstatus=0, με συνέπεια να αυξάνεται ελάχιστα η συχνότητα εκτέλεσης του κώδικα

Στο matlab υπάρχει επίσης η δυνατότητα να διάβαζονται οι συντεταγμένες καθώς και οι τιμές των χρωμάτων σε σημεία πάνω στην εικόνα, πράγμα που βοηθάει σε ορισμένες περιπτώσεις.

4.3 Επεξήγηση τμημάτων, περιγραφή συστήματος, και τρόπος λειτουργίας στον μικροελεγκτή

4.3.1 Περιγραφή μικροελεγκτή

Σε κάθε ρομποτικό παίκτη υπάρχουν δύο ηλεκτρονικές πλακέτες. Η μία είναι η ηλεκτρονική πλακέτα της εταιρίας active-robots atmega8535 micro radio board και η άλλη είναι η πλακέτα ισχύος και κίνησης των μοτέρ που κινούν τους τροχούς. Η πρώτη περιλαμβάνει την είσοδο της τάσης 9V, τον μικροελεγκτή 8535, την υποδοχή προγραμματισμού του, το ολοκληρωμένο κύκλωμα παραλαβής δεδομένων μέσω ασύρματης σειριακής επικοινωνίας, και το τμήμα διανομής της τάσης στην δεύτερη ηλεκτρονική πλακέτα. Η δεύτερη είναι η ηλεκτρονική πλακέτα της ισχύος και ελέγχου των μοτέρ που υπάρχει ένα στον κάθε τροχό.

Ο συγκεκριμένος μικροελεγκτής είναι Atmega8535 είναι της εταιρίας atmel που χρησιμοποιείται σε πολλές ρομποτικές εφαρμογές. Κάποια στοιχεία που μας ενδιαφέρουν εδώ είναι ότι αυτός ο μικροελεγκτής είναι της τάξης των 8 bit, με μέγιστη συχνότητα χρονισμού 16MHZ. Εχει δυο timers/counter των 8 Bit και ένα των 16 bit με διαφορετικά prescallers, είναι 8 καναλιών με 10 bit A/D μετατροπέα, 32 I/O ports, τάση λειτουργίας 4-5 Volt μέσω του κυκλώματος που είναι εγκατεστημένος , έχει λειτουργία

παραλαβής προτοκόλου USART σειριακών δεδομένων, και προγραμματίζεται μέσω της παράλληλης θύρας. Το περιβάλλον εργασίας και προγραμματισμού του είναι η AVR C, και χρησιμοποιείται συμβατότητα με ansi c.

4.3.2 Γενική λειτουργία προγράμματος μικροελεγκτή

Η λειτουργία του μικροελεγκτή είναι συνεχής όταν συνδεθεί τάση , οπότε περιμένει να δεχτεί κάποια σειριακά δεδομένα. Όταν καταλάβει ότι έρχονται σειριακά δεδομένα , αναλαμβάνει να τα αποθηκεύσει για επεξεργασία. Όταν παραλάβει τα 4 bytes που έχουμε ορίσει για την κάθε περιγραφή μίας κίνησης , τότε γινεται έλεγχος στο τμήμα ακοκωδικοποίησης , για το εάν είναι σωστά τα δεδομένa και γι αοτν συγκεκριμένο ρομποτικό παίκτη.

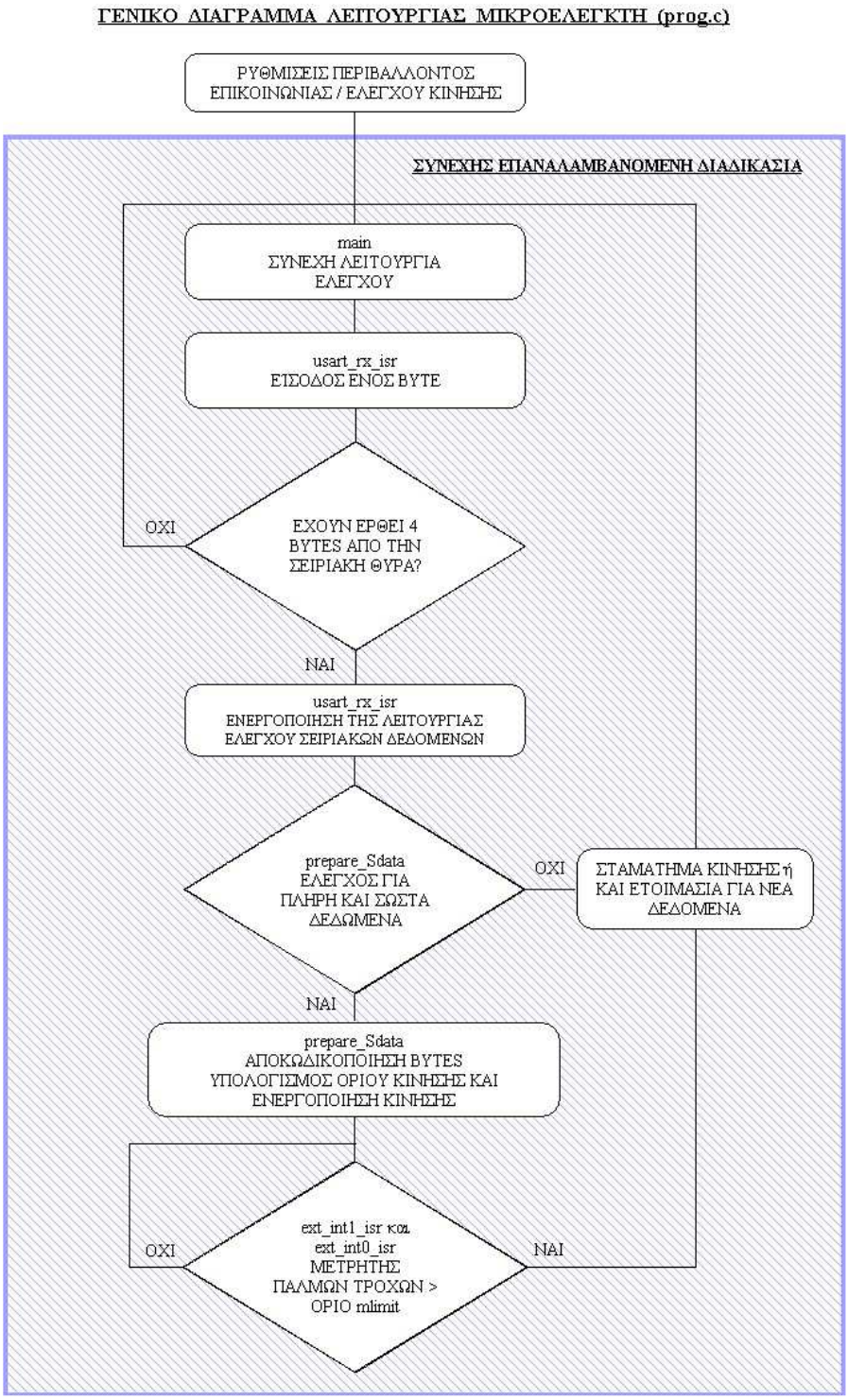
Σε περίπτωση που δεν εχουν παραληφθεί σωστά με τον έλεγχο που έχει περιγραφεί παρακάτω αναλυτικά, τότε τα δεδομένα δέν λαμβάνονται υπόψιν και προετοιμάζεται η διαδικασία για τα νέα 4 bytes. . Εάν παραληφθούν σωστά τότε εκτελείτε ο διαχωρισμός για το τι κίνηση πρόκειτε να εκτελεσθεί και για πόσο μεγάλη απόσταση (mlimit). Ενεργοποιούνται οι αντίστοιχες μεταβλητές με τις τιμές που έχουν αποκωδικοποιηθεί και αρχίζει η κίνηση.

Όπως αναφέραμε οι κινήσεις είναι δύο ειδών , ευθεία και περιστροφική. Εάν αποσταλλούν ειδικές τιμές σε bytes π.χ 450 κινηση πανηγυρισμού , 458 περιστροφή μερικής αριστερής στροφής , 459 περιστροφή μερικής δεξιάς στροφής , 800 σταμάτημα ρομποτικού παίκτη, ενεργοποιούνται τα αντίστοιχα τμήματα για να εκτελεστον αυτές οι ειδικές περιπτώσεις.

Επειδή ο τρόπος αποστολής της ασύρματης επικοινωνίας , έχει αποδέκτες όλους τους ρομποτικούς παίκτες συγχρόνως, και πιθανό να δέχονται σήματα και από τον αντίπαλο ασύρματο πομπο, πρέπει ο κάθε ο ρομποτικός παίκτης να μπορεί να αναγνωρίσει εάν τα δεδομένα που παίρνει , είναι εντολές για αυτόν ή για κάποιον άλλο ρομποτικό παίκτη σε περίπτωση που υπάρξει ίδια συχνότητα αποστολής.

Για αυτό τον λόγο στο κάθε τμήμα ελέγχου εισερχόμενων bytes, του κάθε ρομποτικού παίκτη οι τιμές ελέγχου αλλάζουν και είναι σύμφωνες με τις τιμές τις κωδικοποίησης που έχουμε δημιουργήσει στο περιβάλλον του ελέγχου του συστήματος στο Matlab. Οπότε στην περίπτωση που έχουμε τρεις ρομποτικούς παίκτες, πρέπει να υπάρχουν και τρία διαφορετικά προγράμματα γραμμένα στην AVR C, ώστε ο κάθε ένας ρομποτικός παίκτης να παλαμβάνει τα δικά τις δικές του εντολές.

Στο παρακάτω διάγραμμα , φαίνεται ένα γενικό σχεδιάγραμμα του προγράμματος που βρίσκεται στον μικροελεγκτή του κάθε ρομποτικού παίκτη.



4.3.3 Αρχικοποίηση τιμών στον atmega8535 και περιγραφή τμημάτων προγράμματος

Ο μικροελεγκτής από μόνος του έχει δυνατότητα λειτουργίας 16MHZ. Εμείς δεν μπορούμε να δουλέψουμε με αυτή την συχνότητα επειδή είναι αρκετά μεγάλη οπότε χρησιμοποιούμε του prescalers (διαιρέτες συχνότητας) για να βρούμε μία αξιοπρεπή αναλογία χρόνου/διαιρέτη. Στο συγκεκριμένο πρόγραμμα έχουμε τους timer/clock ορίσει σε 62.500 HZ που εξηπηρετεί της ανάγκες μας για την κίνηση, που μας εξασφαλίζει μία διακριτική ικανότητα 2^{16} παλμών σε ένα σύνολο αποδεκτών περιστροφών των τροχων. . Επίσης χρησιμοποιούμε ένα ανάλογο διαιρέτη συχνότητας που ρυθμίζουμε την συχνότητα ταχύτητας λήψης σειριακών δεδομένων της USART σε 9600 bit/sec, που έιναι μία αποδεκτή τιμή για την εφαρμογή αυτή, και είναι ρυθμίσμένη όπως και στο περιβάλλον ελέγχου Matlab.

Όλες αυτές οι ρυθμίσεις γίνονται με την καταχώρηση συγκεκριμένων δυαδικών ή δεκαεξαδικών τιμων στις αντίστοιχες μεταβλητές του μικροελεγκτή και αυτές με την σειρά τους σε επίπεδο bits, ενεργοποιούν και απενεργοποιούν τις αντίστοιχα λειτουργίες που χρειάζονται, και εμφανίζονται στην συνάρτηση setup μεσα στο πρόγραμμα.

Επίσης εδώ δημιουργούνται οι αναλογίες της ευθύγραμμης και της περιστροφικής κίνησης που χρειάζονται για την δημιουργία των ορίων κίνησης κατά την διάρκεια του προγράμματος, οι οποίες τιμές έχουν βρεθεί εμπειρικά. Διάφορα χωριστά τμήματα παρουσιάζονται παρακάτω:

4.3.3.1 Τμήμα μέτρησης παλμων κατά την κίνηση συνάρτηση (ext_int1_isr και ext_int0_isr)

Ο κάθε τροχος χρησιμοποιεί μία διακοπή (interrupt) η οποία αναπτύσεται μέσα στο πρόγραμμα με συγκεκριμένο κώδικα. Εδώ κάθε φορά αυξάνεται ο μετρητής left_enc ή ο right_enc ανάλογα για τον αριστερό ή δεξιό τροχό, μέχρι να φτάσει το όριο mlimit και εάν περάσει το όριο αυτό, τότε εκτελεί το τμήμα που υπάρχει σε κάθε interrupt και σταματάει την κίνηση με την εκχώρηση των κατάληλων τιμών στις μεταβλητές. Το όριο αυτό είναι το ίδιο και στα δύο interrupts , επειδή έχω πάντα ίδια κίνηση των τροχών όπως το έχουμε προγραμματίσει..

Έχει εμφανιστεί ένα προβλήμα που έχει σχέση με μη σωστή κατανομή του βάρους στο KB του ρομποτικού παίκτη ή δυσλειτουργία του μοτέρ, στον αριστερό τροχό. Υπάρχει ένα τμήμα το οποίο διορθώνει το πρόβλημα της εκκίνησης που αναφέρεται στον τμήμα προβλημάτων. Εδώ όταν έχουμε το left_enc > spdchg, δηλαδή όταν εκκινήσει, μετά από περίπου 5000 παλμούς, κάνουμε ίσες τις τιμές των ταχυτήτων των δύο τροχών για να έχουμε συνέχή ευθύγραμμη κίνηση.

4.3.3.2 Τμήμα εισόδου σειριακών δεδομένων και αποθήκευσης (συνάρτηση usart_rx_isr)

Αυτή είναι μία άλλη interrupt, η οποία εκτελείτε κάθε φορά που έρχεται ένα byte. Στην συγκεκριμένη συνάρτηση, κάθε byte αποθηκεύεται στον πίνακα serbyte και αυξάνεται ο μετρητή του sercount, ο οποίος όταν γίνει ίσος με τον αριθμό τέσσερα, τότε σημαίνει ότι έχει έρθει όλη η σειρά των bytes που περιμένουμε. Σε αυτή την περίπτωση αλλάζει κατάσταση η μεταβλητή recfl=10, η οποία με την σειρά της ενεργοποιεί το τμήμα

ελέγχου των εισερχόμενων bytes καθώς και ο μετρητής sercount γίνεται μηδέν , για να μπορούν αν εισαχθούν νέες εντολές.

4.3.3.3 Τμήμα ελέγχου εισερχόμενων bytes (συνάρτηση prepare_Sdata) , αποκωδικοποίησης τους και έναρξη εκτέλεσης εντολών.

Αυτό το τμήμα αναλαμβάνει να αναγνωρίσει και να κρίνει εάν τα εισερχομενα δεδομένα είναι πλήρη, σωστά και εάν αφορούν το συγκεκριμένο robot id. Σε περίπτωση λάθους απλά γίνεται μηδενισμός των μετρητών των interrupts, για τον λόγο μήπως έχουν κινηθεί από εξωτερικό αίτιο. Εάν είναι τα δεδομένα σωστά τότε τα δεδομένα που βρίσκονται στον πίνακα serbyte, μεταφέρονται για αποκωδικοποίηση. Η απόκωδικοποίηση είναι ανάλογη με την κωδικοποίηση που έγινε στο περιβάλλον του Matlab πρίν να σταλλούν τα bytes, αλλα με ανάποδα βήματα.

Στην περίπτωση σωστής αποκωδικοποίησης του τελικού byte που έγινε ανασύνθεση, αποθηκεύεται στον πίνακα datbyte[] που αποτελείτε από τρία στοιχεία. Σε περίπτωση που αφορά ευθύγραμμη κίνηση η αποθήκευση γίνεται στην θέση 1,δηλαδή datbyte[0] και η τιμή του περιεχομένου , είναι η τιμή της απόστασης της ευθυγραμμης κίνησης όπως ακριβώς υπήρχε πρίν να διασπαστεί σε byte στο τμήμα κωδικοποίησης του matlab. Σε πρίπτωση που αφορά αλλαγή ταχύτητας, τότε έχουμε αντίστοιχα αποθήκευση στη θέση 2, δηλαδή datbyte[1] και τοποθετείτε επίσης η τιμή που είχε στο matlab. Σε περίπτωση που αφορά περιστροφική κίνηση, η αποθήκευση γίνεται στην θέση 3 δηλαδή datbyte[2], και ισχύει το ίδιο πράγμα για την τιμή όπως στις άλλες περιπτώσεις.

Αναλόγος τίς τιμές που περιλαμβανοντα μέσα στον πίνακα datbyte[], γίνεται και η ανάλογη εκτελεση. Όταν περιέχει τιμή το datbyte[0] τότε δίδονται τιμές για την εκκίνηση της ευθύγραμμης κίνησης. Εδώ επιπλέων, λόγω του πρόβληματος που υπάρχει όπως αναφέρθηκε, υπάρχει ένας έλεγχος που αποδίδει διαφορετικές τιμές ταχυτήτων, αναλόγος της απόστασης που προκειτε να διανύσει το robot, για λόγους συγκράτησης της ευθύγραμμης κίνησης. Στην περίπτωση που περιέχει τιμή το datbyte[2] , κάνουμε έναν επιπλέων έλεγχο για το εάν το περιεχόμενο του datbyte[2]<180 ή datbyte[2]>180. Η πρώτη περίπτωση ανοίκει στην περιστροφική κίνηση (αριστερόστροφα) που ονομάζεται κανονική. Η δεύτερη περίπτωση είναι η ανάποδη περιστροφική κίνηση (δεξιόστροφα) που έχει περιγραφεί σε παραπάνω τμήμα. Μετά τον ελεγχο αυτό, το πόσο θα περιστραφεί ο ρομποτικός παίκτης εξαρτάται από την τιμή του πίνακα. Επίσης πάντα λίγο πρίν αρχίσει η κίνηση, αναλόγος, υπάρχει η συνάρτηση υπολογισμού του ορίου mlimit, καθώς μετά εκμηδενίζονται τα περιεχόμενα του datbyte[] και δημιουργούνται οι συνθήκες για την επόμενη παραλαβή σειριακών δεδομένων.

4.3.3.4 Βασικό τμήμα της ροής του προγράμματος (συνάρτηση main)

Αυτή είναι η ποιο βασική συνάρτηση της γλώσσας C, η οποία εκτελείτε συνεχώς εάν δεν συμβαίνει τίποτα , δηλαδή δεν έρχονται σειριακά δεδομένα ή δεν κινείται ρομποτικός παίκτης. Είναι έτσι γραμμένη ώστε να εκτελείτε συνέχεια , καθώς και να δίνει την εντολή έναρξης του ελέγχου των σειριακών δεδομένων εάν έρθει μια πλήρη σειρά τεσσάρων bytes. Επίσης χρησιμοποιείται και για να μπορεί να φρενάρει ο ρομποτικός παίκτης σε περίπτωση που δημιουργηθεί κάποιο πρόβλημα στα interrupts.

4.3.4 Πώς επιτυγχάνεται η κίνηση σε επίπεδο μικροελεγκτή

Οι δυνατότητες γενικά που μπορεί να εκτελέσει ο κάθε τροχός είναι η κίνηση προς τη δεξιά φορα , η κίνηση της αριστέρης φοράς και η αλλαγή ταχυτήτητας. Με συνδυασμό αυτών των απλών κινήσεων μπορούμε να επιύχουμε τις κινήσεις της ευθείας και της περιστροφής και οποιασδήποτε άλλης συνδυασμενης κίνησης. Δηλαδή για να επιτύχουμε την ευθεία κίνηση απλά δίνουμε την ίδια φορά θεωρητικά (πρακτικά αντίθετη) από ένα εξωτερικό σημείο παρατήρησης και την ίδια ταχύτητα. Για να επιχύχουμε την δεξιά περιστροφή ή την αριστερή περιστροφή εδώ χρησιμοποιούμε την στροφή γύρω από το κέντρο του κάθε ρομποτικού παίκτη. Οπότε για να γίνει αυτό δύνουμε θεωρητικά αντίθετη φορά (πρακτικά την ίδια) και στους δύο τροχούς καθώς και ίδια τχύτητα.

Η φορά της κινησης για να επιτευχθεί, χρησιμοπιείτε η πολικότητα ενώ η ταχύτητα για να επιτευχθεί είναι συναρτηση της τάση. Αυτές οι παράμετροι αναφέρονται μέσα στον πρόγραμμα σαν: α) για τις δύο διευθύνσεις MOTOR1_DIR ή MOTOR2_DIR που η κάθε μία αντιστοιχεί σε ένα μοτέρ πάιρνουν τιμές 0 ή 1, της πολικότητας. Β) για τις δύο ταχύτητες OCR0 και OCR2 που η κάθε μία αντιστοιχεί σε ένα μοτέρ και οι τιμές τους είναι από 0 – 256.

Για παράδειγμα για να κινηθεί ο ρομποτικός παίκτης ευθεία με ταχύτητα x m/s για πρέπει να δώσουμε τις τιμές MOTOR1_DIR=0 , MOTOR2_DIR=1 (ανάποδες πολικότητες) και OCR0=45, OCR2=256-45. Επίσης κατα την διάρκει της κίνησης πρέπει να έχουμε την δυνατότητα να γνωρίζουμε πόσο χρόνο κινείτε η πόσα βήματα ή παλμούς έχει κάνει ο κάθε τροχός για να μπορέσουμε να κινηθούσε σε συγκεκριμένες αποστάσεις. Σε αυτό το σημείο εάν εκτελεστούν οι εντολές αυτές, η κίνηση θα είναι συνεχής χωρίς να σταματήσει.

Για αυτό το λόγο η κάθε κίνηση τροχού έχει συνδεθεί με μία διακοπή (interrupt) η οποία εκτελείτε κάθε φορά που υπάρχει μία αναγνώριση βήματος , δηλαδή κίνησης του τροχού. Το πόσο πρέπει να κινηθεί ο τροχός για να καταλάβουμε ένα παλμό έχει να κάνει με τις αρχικές ρυθμίσεις του timer/clock που θέλουμε σε κάθε τροχό. Αυτό σημαίνει θεωρητικά ότι κάθε φορά που αναγνωρίζεται κίνηση, ο έλεγχος του προγράμματος μεταβιβάζεται μέσα στη συγκεκριμένη interrupt, εκτελώντας τις εντολές της και μετά επιστρέφοντας στο σημείο που ήταν πριν από αυτό. Αυτό είναι ότι χρειαζόμαστε γιατί σε εκείνο το σημείο μπορούμε να αθροίζουμε όλες τις εκτελέσεις του κάθε interrupt δηλαδή για κάθε τροχό, οπότε αυτομάτα εκεί μπορούμε να ελέγξουμε των αριθμό των παλμών και έχοντας θέσει ένα όριο, μπορούμε να σταματήσουμε την κίνηση του, δίδοντάς τους παραμέτορους την ταχύτητα = 0 ή 255 , αναλόγος τον τροχό.

Για την ευθύγραμμη κίνηση το μόνο που χρειαζόμαστε μετά το γνωστό αριθμό των βημάτων που έχει κάνει ένας τροχός είναι να βρούμε μία αντιστοιχεία βήμα/mm.Στο πρόγραμμα χρησιμοποιούμε σταθερή ταχύτητα για την κίνηση της ευθύγραμμης κίνησης και σταθερή ταχύτητα για την περιστροφική. Εμπειρικά μετρώντας την ακτίνα του τροχού ρ=22mm και χρησιμοποιώντας τον τύπο της περίμετρου Π=2*π*ρ => βρίσκουμε ότι η περίμετος του κάθε τροχού είναι W_PER=138.2mm. Μετακινώντας αρκετές φορές

για επιβεβαίωση το ρομποτικό παίκτη σε ευθεία με μία πλήρη περιστροφή βρίσκουμε ότι οι παλμοί είναι 3870, με την βοήθεια του terminal εμφάνιζωντας τις στην οθόνη.

Οπότε έχουμε 3870 παλμοί / 138.2 mm => 28 παλμοί / mm ή 1 mm=28 παλμοί. Οπότε εάν θέλουμε να μετακινηθούμε 1000mm , πρέπει να χρησιμοποιήσω το όριο mlimit=1000*28 παλμοί = 28000 παλμοί.

Δηλαδή καθώς κινείτε ο ρομποτικός παίκτης, σε κάθε μικρή κίνηση του, αθροίζοντας τις φορές που εισέρχεται μέσα στην interrupt , θα ελέγχεται πάντα το όριο mlimit, το οποίο εάν το περάσει μπορεί να δωθεί η εντολή να σταματήσει. Τώρα μπορούμε να ελέγχουμε εάν η ρύθμιση που έχουμε κάνει στην αρχικοποίηση του προγράμματος timer/clock μας επιτρέπει να μπορούμε να εκτελέσουμε και την μέγιστη ευθύγραμμη κίνηση.

Όπως έχουμε μελετήσει παραπάνω, η μέγιστη ευθύγραμμη κίνηση που μπορούμε να έχουμε μέσα σε αυτές τις διαστάσεις του γηπέδου είναι περίπου 2000mm. Οπότε μεγιστο όριο σε πλαμούς μπορούμε να έχουμε mlimit=2000*28 παλμοί => mlimit=56000 παλμούς, και είναι ένας αιτθμός που μπορεί να παρασταθεί χωρίς να γίνει κάποια υπερχείληση του ορίου των integer μετρητών των παλμών ανά κάθε τροχο (2^{16})=65536 παλμοί.

Για την περιστροφική κίνηση , εμπειρικά παρατήρούμε πολλές φορές ότι για την περιστροφή του ρομποτικού παίκτη γύρω από τον εαυτό του 90^{o}, περιστρέφεται και ο κάθε τροχος κατά ¼ της περιμέτρου του. Οπότε σύμφωνα με αυτό, επειδή περιστρέφεται κατά ¼ ο τροχός θα έκανε ¼ * 3870 παλμούς = 967 παλμούς. Αυτό σημαίνει ότι για 90^{o}=967 παλμοί => ότι 967/90=10.75 περίπου ή όπως χρησιμοποιούμε στο πρόγραμμα η points_rot =11 παλμοί ανά 1^{o}.

Δηλαδή για παράδειγμα εάν θέλουμε να κινηθούμε 120^{o}, το όριο των παλμών που πρέπει να ελέγχεται σε κάθε εκτέλεση της interrupt, είναι το mlimit=120 * 11 => mlimit=1320 παλμοί.

Γενικά η κανονική εύρεση των αναλογιών είναι όπως περίγραφηκε παραπάνω αλλα υπάρχουν ορισμένες διαφοροποιήσεις σε αυτές τις τιμές και θα αναφερθεί στους λόγους στα γενικά προβλήματα. Επίσης οι περισσότερες μεταβλητές και ειδικότερα οι μεταβλητές που έχουν σχέση με την κίνηση , είναι δηλωμένες global, δηλαδή ευρείας εμβέλιας , οπότε με μία αλλαγή τους σε συνάρτηση, έχουν αποτέλεσμα σε όλο το υπόλοιπο πρόγραμμα. Ακόμα πρέπει να αναφέρθεί ότι καθώς γίνεται η κίνηση, και εκτελούνται τα interrupts, μπορεί οποιαδήποτε στιγμή να παραληφθούν νέα σειριακά δεδομένα και να αλλάξει συμπεριφορά ο ρομποτικός παίκτης, χωρίς να χρειάζεται να τελειώσει πρώτα η κίνηση αυτή, όπως το παράδειγμα που ο ρομποτικός παίκτης βρεθεί σε σημείο βολής εκτέλεσης γκολ.

5. Υπόλοιπα χαρακτηριστικά εφαρμογής και αναφορές κατά την υλοποίηση της.

5.1 Χρόνοι εκτέλεσης εντολών

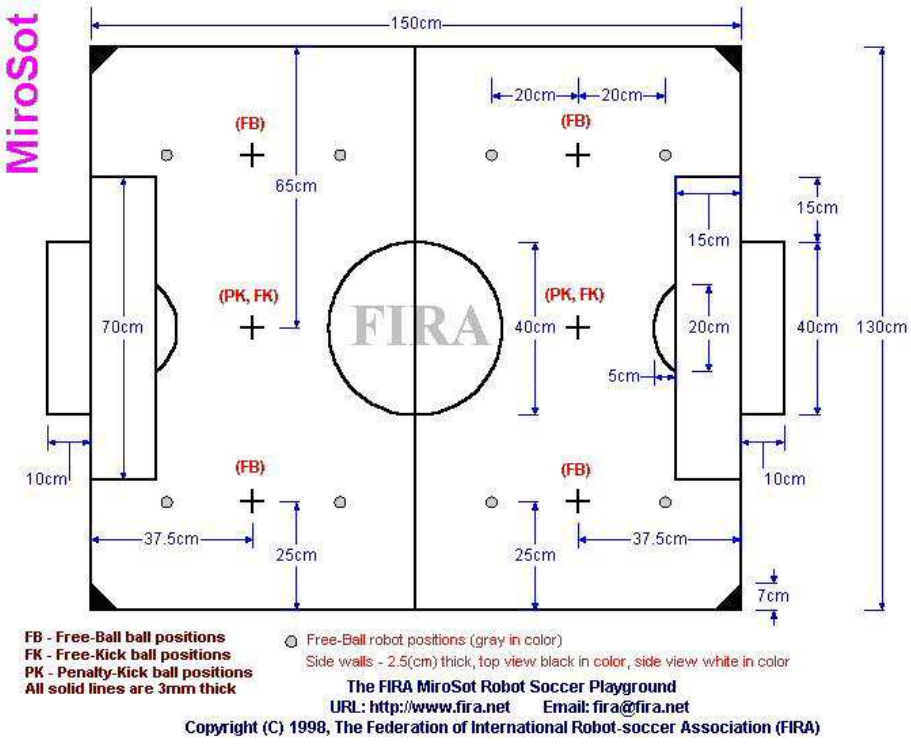
Οι ποιο χρονοβόρες εντολές σε εκτέλεση σε όλη την εφαρμογή είναι το τμήμα της δειγματοληψίας και το τμήμα της αποστολής δεδομένων. Το τμήμα της δειγματοληψίας χρειάζεται περίπου 0.015 s για να μπορέσει να πάρει την εικόνα και να αριθμήσει τα αντικείμενα που αναγνωρίζει με την χρήση του κατάλληλου threshold. Η αποστολή δεδομένων χρειάζεται περίπου 0.030 s χρονο για να ανοίξει την θύρα , να εκτελεστούν οι εντολές αποστολής με τα τέσσερα byte και να κλείσει η σειριακή θύρα,

Απο τις δοκιμές που έγιναν ο χρόνος αυτός έχει σχέση με τον αριθμό των δεδομένων που στέλνονται και μεχρι να κλείσει η σειριακή θύρα , γιατί τότε καταλαβαίνει το matlab ότι η αποστολή τελείωσε.

Γενικά η υπόλοιπη λειτουργία κατά μέσο όρο χρειάζεται μικρό χρόνο σε σχέση με τις παραπάνω περιπτώσεις οπότε έχουμε περίπου 21-31 επαναλήψεις / δευτερόλεπτο αλλα για τον συγκεκριμένο Η/Υ που χρησιμοποιείται. Σε Η/Υ αργό βέβαια οι χρόνοι που αναφερα διαφέρουν αρκετά.

5.2 Στρατηγική της συγκεκριμένης εφαρμογής

Η Στρατηγική στο κανονικό πρωτάθλημα είναι μία απλή έως μία σύνθετη διαδικασία , διότι προυπόθεση είναι να εφαρμόζονται οι κανονισμοί και μετά από αυτό να εφαρμόζεται η στρατηγική μας. Μπορεί να αποτελείτε από μία απλή κίνηση , έως μία συνδυασμένη. Μία εικόνα παρακάτω εμφανίζει τις διαστάσεις που έχει το κανονικό γήπεδο και κάθε λεπτομέρεια δημιουργεί ένα κανονισμό, όπως μας το δίνει ο οργανισμός της FIRA.



(ε5.1)

Επειδή εχουμε μόνο ένα ρομποτικό παίκτη στην εφαρμογή αλλά βασίζεται στην μικρή κατηγορία των τριών robot, πρέπει να χρησιμοποιήσουμε μία απλή εστω στρατηγική για λόγους παρουσίασης της εργασίας. Οπότε η στρατηγική που ακολουθείτε είναι η παρακάτω: Πρέπει να ισχύει η προυπόθεση, να υπάρχουν τα αντικείμενα και να έχουν αναγνωριστεί επιτυχώς και τα δύο, δηλαδή να εχουμε splineRobo=1 για ρομποτικό παίκτη και να έχουμε splineBall=1 για την μπάλλα.

Σαν σημείωση κατά την εκκπινηση της εκτέλεσης του προγράμματος δηλώνεται χειροκίνητα ποιο είναι το δικό μας τέρμα οπότε το άλλο είναι το αντίπαλο, γιατί παίζει σημασία σε όλους του υπολογισμούς γωνιών και αποστάσεων. Μετά από αυτό, το παιχνίδι αρχίζει με τον ρομποτκό παικτη και την μπάλλα σε τυχαία θέση. Στόχος του ρομποτικού παίκτη , είναι απλά να παρακολουθεί και να ωθεί την μπάλλα όποτε είναι δυνατό προς το αντίπαλο τέρμα, μεχρι να καταφέρει να βάλει το γκόλ. Εδώ υπάρχουν δύο περιπτώσεις που έχουν διαχωριστεί, αναλόγος των θέσεων robot , μπάλας και αντίπαλου τέρματος ως προς τον άξονα τον x.

Στην περίπτωση που ο ρομποτικός παίκτης βρίσκεται ανάμεσα στο αντίπαλο τέρμα και την μπάλλα, τότε πρέπει να υπολογιστούν οι κινήσεις οι οποίες θα φέρουν το ρομποτικό παίκτη πίσω από την μπάλλα χωρίς να την χτυπήσει κατα την διάρκεια αυτή, δηλαδή να βρεθεί η κατάλληλη κλίση, ωστε να είναι η ποιο γρήγορη, και το αποτέλεσμά της να έιναι το καλύτερο επιθυμητό. Σαν παράδειγμα για αυτή την κίνηση γίνεται ελεγχος για να επιλεγεί η καλύτερη πλευρά, από που θα περάσει το robot παράλληλα

από την μπάλλα , πάνω ή κάτω συμφωνα με τον άξονα Y. Επίσης η ποιο απλή περίπτωση είναι όταν το robot βρίσκεται ήδη πίσω από την μπάλα και το αντίπαλο τέρμα βρίσκεται στο αντίθετο άκρο.

Κατά τις κινήσεις αυτές, αναλόγος τις συνθήκες που αναφέρονται στα είδη κινήσεων με λεπτομέρειες, γίνονται οι αντίστοιχοι υπολογισμοί όπου παίζει σημασία η γεωγραφική θέση των αντκειμένων. Στην περίπτωση που είναι πίσω από την μπάλλα, και εάν είναι πρίν την θέση που έχει δηλωθεί για εκτέλεση σούτ, απλά σπρώχνει την μπάλλα με μικρές κινησεις, προσπαθώντας να λάβει κλίση ίση με την κλίση του τέρματος και απόσταση ίση με για εκτέλεση σούτ. Σε περίπτωση που βρεθεί στη συγκεκριμένη απόσταση εκτέλεσης βολής και εάν βρίσκεται σε σχετική ευθεία μεταξύ ρομποτικού παίκτη, μπάλλας και αντίπαλου τέρματος, μετά απο τον έλεγχο των γωνιών, τότε προχωράει περίπου μέχρι το τέρμα, συμπαρασύροντας την μπάλλα μαζί του, και αφήνοντας την να καταλήξει σε γκόλ.

Εάν επιτύχει γκόλ κάνει μία μικρή στροφή σαν πανηγυρισμό, και μετά ο έλεγχος του προγράμματος μεταφέρεται στο σημείο που γίνεται η δειγματοληψία, όπου περιμένει να αναγνωρισθεί μία έγκυρη μπάλλα για να ξαναρχίσει η λειτουργία από την αρχή.

5.3 Ιδέες και δυνατότητας αναβάθμησης της εφαρμογής

Ο συγκεκριμένος τρόπος αντιμετώπισης αυτής της εργασίας για το ρομποτικό ποδόσφαιρο δεν είναι ο μοναδικός και ο ποιο σωστός που υπάρχει. Επειδή είναι αρκετά πολύπλοκη η εφαρμογή όταν πρέπει να υπολογιστούν οι λεπτομέρειες , τότε μπορεί ο κάθε ένας να έφαρμόσει πολλές ιδεές οπότε πιστεύω ότι πάντα θα υπάρχουν δυνατότητες αναβάθμησης.

Η εφαρμογή όπως δημιουργήθηκε , υποστηρίζει σε διάφορα σημεία την πλήρη λειτουργία για τρείς ρομποτικούς παίκτες, και σε διαφορετικά , μόνο για να μην παρουσιαστεί πρόβλημα κατα την λειτουργία παρουσιάσης. Δεν υπήρχε η δυνατότητα να δοκιμάστουν τρείς ρομποτικοί παίκτες, οπότε δεν γνωρίζουμε πόσο αξιόπιστος και πόσο καλά θα λειτουργήσει σε πραγματιές συνθήκες ο κώδικας.

Σαν Ιδέα αναβάθμισης θα πρότεινα αντί να υπάρχουν απλές συντεταγμένες και να κινείτε το robot σε κάποιες από αυτές , δηλαδή σαν παράδειγμα goto (134,56), θα μπορούσαμε να έχουμε ορίσει και περιοχές συντεταγμένων και να τις ονομάζαμε για παράδειγμα κέντρο, όπου θα περιλάμβανε ένα εύρος συντεταγμένων π.χ από (100 – 200 , 140 – 180) και η εντολή θα ήταν goto center ή goto term κτλ, και θα αποτελούσε τμήμα της στρατηγικής.

Επίσης στο πρόγραμμα αυτό θα μπορούσε να αλλαχτεί λίγο ο τρόπος μεταφοράς των δεδομένων, δηλαδή να ανοιγει και να κλέινει μία φορά η σειριακή θύρα, επειδή είναι χρονοβόρα, και να αποστέλλονται οι τιμές ενδιάμεσα, πράγμα που δεν μπόρεσε να επιτευχθεί πιθανό από μη σωστή λειτουργία του matlab ή κακή ρύθμιση δική μας στον μικροελεγκτή.

Άλλη μία ρύθμιση για την χρονοτριβή, θα μπορούσε να χρησιμοποιηθεί ένας άλλος τρόπος εισαγωγής της δειγματοληψίας, με απευθείας τρόπο προσπέλασης της μνήμης των δεδομένων του προγράμματος οδήγησης της κάμερας, και όχι μέσω του τρίτου παροχέα (.dll) που χρησιμοποιούμε στην συγκεριμένη εφαρμογή.

Κατασκευαστικά στο θέμα του ρομποτικού παίκτη , επειδή είναι πολύ ευαίσθητος στην ολίσθηση , θα πρότεινονταν ένας ρομποτικός παίκτης με 3 τροχούς παρά με δύο τροχούς και την συνεχή τριβή της επιφάνειας αλλα θα βρισκόμασταν σε άλλη κατηγορία διοργάνωσης.

5.4 Προβλήματα που αντιμετοπίστηκαν σε όλη την διάρκεια της αναπτυξης της εφαρμογής

Ένα πρώτο πρόβλημα που υπήρξε, ήταν να βρεθεί ένα σωστό σχέδιο για την αναγνώριση του κάθε ρομποτικού παίκτη, και σύμφωνα με τους κανόνες της συγκεκριμένης διοργάνωσης. Δημιούργηθηκαν πολλά σχέδια , με διάφορα χρώματα και έγιναν πολλές δοκιμές για τι πληροφορίες μπορούμε να έχουμε από το κάθε σχέδιο, καθώς επίσης το κάθε σχέδιο να μπορούσε να δώσει συνολικά τρείς παραλαγές με λίγες αλλαγές ώστε να εφαρόοζόνταν σε κάθε μέλος της ομάδας.

Ένα άλλο πρόβλημα ήταν η αντανακλάσεις που υπήρχαν από τα αντικείμενα, οπότε ή έπρεπε να χρησιμοποιηθεί χαμηλότερο threshold αλλα θα χανόταν πληροφορία ή να χρησιμοποιούταν υλικά που δεν θα είχαν ανατανακλαση.

Επειδή η εγκατάσταση σύμφωνα με τους κανονισμούς αναφέρει ότι η κάμερα πρέπει να είναι τουλάχιστον 2 m, αυτό είχε σαν αποτέλεσμα ότι από την σχετικά καλή διακριτική ικανότητα της κάμερας που είναι 640x480 εικονοστοιχεία, σε αυτό το ύψος, η χρήσιμη περιοχή του γήπεδου σύμφωνα με τις διαστάσεις της συγκεκριμένης διοργάνωσης, απεικονιζόταν με περίπου 380x340 εικονοστοιχεία. Σύμφωνα με αυτό, και με την σχεδίαση που έχει γινει το μικρό τετράγωνο ορισμένες φορές φαίνεται πολύ μικ΄ρο αναλ΄γος το Threshold, και η η μικρότερη απόσταση που είναι διακριτή είναι περίπου ένα εκατοστό. Σε περίπτωση μεγαλύτερου ύψους της κάμερας και σε συνδυασμό με ορισμένη φωτεινότητα δεν μπορούσε πάντα να εμφανιστεί η γραμμή, κατ ατην αναγνώρηση μέχρι που χρησιμοποιήθηκε το δυναμικό Threshold.

Πρόβλημα υπάρχει επίσης σε όλόκληρη την εικόνα γιατί εάν προσέξουμε σε όλες τις δειγματοληψίες , η επιφάνεια που δεσμεύεται έχει ένα ελλειπτικό σχήμα και όχι παραλληλόγραμμο, που περιγράφουν οι διαστασεις του γηπέδου. Η συγκεκριμένη κάμερα έχει κάποιες ρυθμίσες παραπάνω στο πρόγραμμα της στης “extended settings” την α) normal, β) letterbox και γ) pan-scan, οι οποίες έχουν κάποια διαφορά σε αυτό το πρόβλημα, αλλά πάλι όλες έχουν σαν αποτέλεσμα το ελλειπτικό σχήμα το οποίο ορισμένες φορές στα πολύ άκρα να μην φαινεται ολόκληρο το αντικέιμενο.

Πολύ βασικό σημείο είναι η ταχύτητα της όλης διασικασίας δηλαδή από τη στιγμή που θα παραληφθεί η εικόνα μέχρι τη στιγμή σημείο που θα υπάρξει εντολή κίνησης. Η κάμερα σαν ηλεκτρονικό εξάρτημα έχει την δυνατότητα να δώσει 30 εικόνες το δευτερόλεπτο στο πρόγραμμά της. Εδώ στο matlab, έγινε χρήση μίας εξωτερικής «βιβλιοθήκης» “.dll”, δηλαδή υποπρογράματος, που δίνει την δυνατότητα στους χρήστες να μπορούν να πάρουν εικόνα από μια usb κάμερα και να μπορούν να την επεξεργαστούν, καλώντας την μέσα σε ένα τμήμα κώδικα.

Η αρχική ανάπτυξη της εφαρμογής έγινε με ένα απλό H/Y P4 (3.4GHZ 1 core) το οποίο έδινε ρυθμό δειγματοληψίας 5-8 εικόνες το δευτερόλεπτο με παράθυρο εμφάνισης . Στο παράθυρο εμφάνισης (figure) απεικονίζεται ότι βλέπει η κάμερα και πως το βλέπει καθώς επίσης εμφανίζεται και η πορεία που έχει υπολογιστεί για τον ρομποτικό παίκτη, Αυτό είναι πολύ βασικό κατά την ανάπτυξη της εφαρμογής γιατί πρέπει να φάινεται η υπολογισμένη πορεία επειδή οι πληροφορίες κατά την διάρκεια τις κίνησης είναι πολλές και μόνο από το αποτέλεσμα φαίνεται εάν κάτι χρειάζεται διόρθωση.

Η αργή δειγματοληψία είχε σαν αποτέλεσμα , όταν η εφαρμογή αντιλαμβανόταν ότι ο ρομποτικός παίκτης βρισκόταν για παράδειγμα σε θέση για ευθυγραμμη κίνηση την χρονική στιγμή t1 , στην πραγματικότητα είχε περάσει απο αυτό το στάδιο και βρισκόταν εκτός ευθείας κίνησης, οπότε δεν υπήρχε σωστός συγχρονισμός και συνεπώς σαν επιπλέον αποτέλεσμα , έγινε λάθος προγραμματισμός κώδικα. Με αλλαγή σε αρκετά γρήγοροτερο H/Y καταφέραμε 25-30 εικόνες ανα δευτερόλεπτο με παράθυρο εμφάνισης κατάστασης σε νεότερο.

Πρέπει να αναφέρουμε ότι επειδή έχουμε σχετικά μικρότερη διακριτική ικανότητα η οποία αυτή συνεπάγεται μικρότετη ανάλυση , το μόνο καλό που έχει αυτό είναι ότι έχουμε και μεγαλύτερη ταχύτητα λόγο μικρότερης ποσότητας στοιχείων.σε ένα δυσδιάστατο πίνακα με (X x Y)

Άλλο πρόβλημα που υπήρξε και υπάρχει σε κάποιο βαθμό είναι ο ομοιόμορφος φωτισμος. Επειδή στη δειγματοληψία αποθηκεύεται κάθε εικονοστοιχείο με την παλέτα του RGB τιμή, από αριθμο χρωμάτων (256 x 256 x 256), είναι πολύ πιθανό κάτι που μας φαίνεται ίδιο με τα μάτια , να είναι λίγο διαφορετικό στην αποθήκευση σε μία από αυτές τις τιμές. Κατά τη δειγματοληψία και κατά την αναγνώρηση επειδή χρησιμοποιείτε και χρωματικός έλεγχος, το ίδιο χρώμα σε διαφορετικές θέσεις του γηπέδου , έχει διαφορετικές τιμές . Αυτό το πρόβλημα έει λύθει σε καλό βαθμό με την χρήση ανοχών , όπως έχει αναφέρθει σε άλλη παράγραφο.

Το μεγαλύτερο πρόβλημα ήταν ο προγραμματισμός του μικροελεγκτή, για τις κινήσεις και για να φτιαχτεί η επκοινωνία , που έπρεπε να ήταν σταθερή και έπρεπε να είμαστε σίγουροι ότι οι τιμές που ζητούσαμε να μετατραπούν σε κίνηση , αποκοδικοποιούταν σωστά. Επειδή ο μικροελεγκτής δεν έχει οθόνη για να μπορούμε να κάνουμε debugging στις τιμές που έχουμε παραλάβει για να έχουμε γρήγορα την λύση , όπως σε ένα πρόγραμμα που απευθύνεται σε Η/Υ με οθόνη, χάθηκε αρκετός χρόνος εκεί.
Με μόνη βοήθεια το terminal, και ένός βοηθητικού υποπρόγραμματος, ,μπορεσαν να εμφανίστουν ορισμένες τιμές μεταβλητων εκεί, άλλα όχι συγχρόνος με την αποστολή σειριακών δεδομένων, γιατί υπήρξε κάποια διένεξη μεταξύ αυτής και της usart_rx_isr .

Πρόβλημα χρόνου υπήρχε συνεχώς κατά την διάρκεια ελέγχου της κίνησης , γιατί σε κάθε αλλαγή έστω μίας τιμής έπρεπε να τοποθετηθεί ξανά το καλώδιο προγραμματισμού στον μικροελεγκτή , και να αλλάζεται η πηγή τάσης στον σταθερό μετασχηματιστή , διότι οι μπαταρίες τελείωναν αρκετά γρήγορα και ήταν μόνο για την κίνηση. Επίσης κατά την διάρκεια της κίνησης χρησιμοποιού'ηθηκαν δύο μπαταρίες παρράληλα συνδεδεμένες γιατί η απώλεια της ισχύος τους , ήταν πολύ γρήγορη πολλές φορές δεν αντιδρούσε ο ρομποτικός παίκτης σωστά, πράγμα το οποίο δημιουργόύσε σε λάθος συμπεράσμαστα.

Επειδή λόγο κατασκευής ο ρομποτικός παίκτης έχει πάντα επαφή εκτός από τους τροχούς επίσης με την μπροστά ή με την πίσω πλευρά του , υπήρχαν και υπάρχουν ακόμη προβλήματα με την ευθεία κίνηση, το οποίο έχει να κάνει με το κεντρο βάτους, καθώς τοποθετώντας την μία ή και τις δύο μπαταρίες τότε αυτό αλλάζει. Επίσης κατά την εκκίνηση του ρομποτικού παίκτη, σε μία ευθύγραμμη εντολή κίνησης , τείνει να περιστρέφεται για λίγο προς τα αριστερά ή τα δεξιά. Βέβαια αυτό δεν μπορεί να υπολογίστεί ακριβως το κέντρο βάρους, διότι η επιφάνειά της κορυφής του ρομποτικού παίκτη δεν είναι αρκετά λεία λόγω ηλεκτρονικών, καθώς επίσης υπάρχει περιορισμός του ύψους του, λόγο κανονισμών.

Επιπλέον πρόβλημα λόγο ολισθησης , έχουμε επειδή υπαρχει ευαίσθησια μεγαλυτερη στην περιστροφή και οπου υπάρξει τριβή με πολύ μικρή αντίσταση σε μία

πλευρά , τότε τείνει να περιστραφεί προς αυτή. Μία εμπειρική παρατήρηση σε αυτό το θέμα είναι ότι εάν έχουμε μεγαλύτερη ταχύτητα συνήθως στην εκκίνηση, τότε έχουμε περισσότερες πιθανότητες να μην πάει ευθεία το robot. Με αρκετές αλλαγές θέσεων στις μπαταρίες , έχουμε αντίστοιχες διαφορετικές συμπεριφορές στις κίνησεις του ρομποτικού παίκτη.

Αυτό το πρόβλημα το έχει λύθει σε σχετικά καλό βαθμό, τοποθετώντας τις μπαταρίες εντελώς στο μπροστά μέρος, και σε μία ευθύγραμμη εντολή κίνησης, χρησιμοποιείται μικρότερη ταχύτητα και λίγο διαφορετική σε κάθε τροχό για να αποφευγθεί η περιστροφή στην αρχή κίνησης, μέχρι μία τιμή (spdchg στον AVR), και μετά γίνονται ίσες τις ταχύτητες για να έχουμε μία τελική ευθεία.

Ορισμένες φορές σε κάποιες αποστάσεις , λόγω ορμής του ρομποτικού παίκτη μπορεί να παρασυρθεί λίγο παραπάνω από ότι χρειάζεται, για αυτό το λόγο έέχει χρησιμοποιειθεί η μείωση του λόγου παλμών τροχού/mm. Γενικα έχουν λυθεί τέτοια προβλήματα με την χρήση των ανοχών όπως έχω αναφέρει με λεπτομέρειες σε παρπάνω παράγραφο.

6. Πηγές πληροφοριών – Βιβλιογραφία

Θεωρεία από την Τεχνολογία μικροελεγκτών της Σχολής Μηχανολογίας

Απειροστικός Λογισμός 1 – Πανεπιστημιακές εκδόσεις Κρήτης

Ηλεκτρονική βοήθεια Matlab

http://www.fira.net

http://www.flickr.com

http://www.robolab.tuc.gr/ASSETS/PAPERS_PDF/ROBOTICS/2_INTRO.pdf - Εισαγωγη στην ρομποτικη - Νικος Βλάσσης

http://www.tzi.de

http://www.promitheasblog.com/2009/12/blog-post_30.html

http://images.google.gr/images

http://en.wikipedia.org/wiki/Tanaka_Hisashige

http://el.wikipedia.org/wiki/Μηχανισμός_των_Αντικυθήρων

http://en.wikipedia.org/wiki/Charles_Babbage

http://www.thocp.net/reference/robotics/robotics2.htm

https://docs.google.com/file/d/0B6ck1zsLRnkeM2VmNzc2MWItZTdkZC00MmMzLTlmOGMtZWY1ZTMyNWU5ZmFl/edit?hl=en&pli=1

http://www.acceleratingfuture.com/people-blog/2008/ai-and-agi-past-present-and-future/

http://www.washingtonpost.com/blogs/wonkblog/files/2013/03/robots-soccer.jpg

http://www.natureworldnews.com/articles/2728/20130630/robocup-training-robots-beats-human-soccer-players-2050-video.htm

http://vancouver.24hrs.ca/News/local/2012/06/12/19870211.html

http://www.huffingtonpost.com/2013/07/01/robocup-2013-photos-video_n_3529350.html#slide=2642544

http://thechronicleherald.ca/artslife/1139126-electric-dreams-soccer-playing-robots-aim-for-world-cup

http://www.acceleratingfuture.com/people-blog/wp-content/uploads/2008/04/7_principles_24.png

http://usatoday30.usatoday.com/tech/news/robotics/2008-04-22-robot-soccer_N.htm?csp=34

http://robotsimulators.8m.com/

http://hackedgadgets.com/2008/11/13/robofoot-robot-football/

http://swordrock.wordpress.com/tag/robot-soccer/

http://www.youtube.com/watch?v=kHaDVhUJA0s

7. Παράθεση Κώδικα

7.1 Βασικό πργραμμα διαχείρησης εφαρμογής (Base.m)

```
% PROJECT:      ROBOSOCCER
% PART:         VISION SECTION
% LAST UPDATE:  11/05/2014

clear;

serenable=0;
plotsize=[50, 200, 150, 120];
angs=0; % ΕΠΙΘΥΜΗΤΗ ΓΩΝΙΑ ΚΙΝΗΣΗΣ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ
angd=0; % ΒΟΗΘΗΤΙΚΗ ΓΩΝΙΑΣ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ

splineRobold=0; % ΑΡΧΙΚΟΠΟΙΗΣΗ FLAG ΠΟΥ ΔΕΙΧΝΕΙ ΕΑΝ ΥΠΑΡΧΕΙ ΕΝΑΣ ΑΝΑΓΝΩΡΙΣΜΕΝΟΣ ΡΟΜΠΟΤΙΚΟΣ ΠΑΙΚΤΗΣ
                % ή ΟΧΙ. ΜΕ ΤΙΜΗ=1 ΕΧΟΥΜΕ ΣΩΣΤΗ ΑΝΑΓΝΩΡΙΣΗ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ

splineRobo=0; % ΑΡΧΙΚΟΠΟΙΗΣΗ FLAG ΠΟΥ ΔΕΙΧΝΕΙ ΕΑΝ ΕΧΕΙ ΑΝΑΓΝΩΡΙΣΤΕΙ Η ΜΠΑΛΑ ή ΟΧΙ.
              % ΜΕ ΤΙΜΗ=1 ΔΕΙΧΝΕΙ ΟΤΙ ΈΧΕΙ ΓΙΝΕΙ ΣΩΣΤΗ ΑΝΑΓΝΩΡΙΣΗ

minang=5; % ΟΡΙΟ ΜΕΓΙΣΤΗΣ ΕΠΙΣΤΡΕΠΟΜΕΝΗΣ ΑΝΟΧΗΣ ΓΙΑ ΕΛΕΓΧΟ ΓΩΝΙΩΝ ΚΙΝΗΣΕΩΝ

error=1; % ΔΗΛΩΝΕΙ ΛΑΘΟΣ ΣΤΗΝ ΑΝΑΓΝΩΡΙΣΗ ΑΝΤΙΚΕΙΜΕΝΩΝ (ΔΗΛ ΕΚΤΟΣ ΜΠΑΛΑΣ , ROBOID, ENEMY)
st_thres=0; % ΒΟΗΘΗΤΙΚΟ threshold
end_thres=0;% ΒΟΗΘΗΤΙΚΟ threshold

% ============================================================================
%% ΤΜΗΜΑ ΕΝΕΓΟΠΟΙΗΣΗΣ ΤΜΗΜΑΤΟΣ ΥΠΟΠΡΟΓΡΑΜΜΑΤΟΣ (maininint.m) ΚΑΙ ΕΠΙΣΤΡΟΦΗ
%% ΤΙΜΩΝ
% ============================================================================

!mode com1:9600,n,8,1 % ΕΝΕΡΓΟΠΟΗΣΗΣ ΣΕΙΡΙΑΚΗΣ ΘΥΡΑΣ ΜΕ ΤΟΥΣ ΑΝΑΟΓΟΥΣ ΠΑΡΑΜΕΤΡΟΥΣ
while (error==1) % ΕΠΑΝΑΛΗΨΗ ΔΙΑΔΙΚΑΣΙΑΣ (LOOP) ΜΕΧΡΙ Η ΑΝΑΓΝΩΡΙΣΗ ΑΝΤ/ΝΩΝ ΑΠΟ
       % ΤΟ ΥΠΟΠΡΟΓΡΑΜΜΑ ΝΑ ΕΙΝΑΙ ΣΩΣΤΗ (ΣΤΗΝ ΣΥΓΚΕΚΡΙΜΕΝΗ ΠΕΡΙΠΤΩΣΗ ΝΑ ΒΡΕΘΕΙ ΕΠΙΤΥΧΩΣ
       % ΤΟΥΛΑΧΙΣΤΟΝ  Η ΜΠΑΛΑ ΚΑΙ ΕΝΑΣ ΡΟΜΠΟΤΙΚΟΣ ΠΑΙΚΤΗΣ ΣΥΜΦΩΝΑ ΜΕ ΤΑ ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ ΠΟΥ ΕΧΕΙ
       % ΠΕΡΙΓΡΑΦΕΙ Ο ΚΑΘΕ ΈΝΑΣ

       Initvar=MainInit(1); % ΜΕΤΑΒΛΗΤΗ ΤΥΠΟΥ ΠΙΝΑΚΑ ΠΟΥ ΠΕΡΙΛΑΜΒΑΝΕΙ ΤΙΣ ΤΙΜΕΣ ΠΟΥ ΕΠΙΣΤΡΕΦΟΝΤΑΙ
                            % ΑΠΟ ΤΗΝ ΕΚΤΕΛΕΣΗ ΤΟΥ ΥΠΟΠΡΟΓΡΑΜΜΑΤΟΣ (maininit.m).
                            % ΑΠΟ ΕΚΕΙ ΕΠΙΣΤΡΕΦΟΝΤΑΙ ΟΙ ΑΡΧΙΚΕΣ ΤΙΜΕΣ ΤΟΥ ΠΕΡΙΒΑΛΛΟΝΤΟΣ
                            % Π.Χ (THRESHOLD, ΜΕΓΕΘΟΣ ΓΗΠΕΔΟΥ - ΜΠΑΛΑΣ - ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ ΣΕ PIXEL,
                            % ΕΥΡΟΣ ΧΡΩΜΑΤΩΝ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ - ΜΠΑΛΑΣ -ΓΗΠΕΔΟΥ ΣΕ RGB ΤΙΜΕS)
                            % ΜΕΓΕΘΟΣ ΑΝΤ/ΜΕΝΩΝ ΚΤΛ.)

       sizeparams=size(Initvar);
       if (sizeparams(1)==8)
           error=Initvar(8,1); % ΕΝΕΡΓΟΠΟΙΕΙΤΕ ΟΤΑΝ ΥΠΑΡΧΕΙ ΑΓΝΩΣΤΟ ΑΝΤΙΚΕΙΜΕΝΟ ΣΤΗΝ ΑΝΑΓΝΩΡΙΣΗ
       end
end;

MainThreshold=Initvar(1,1);   % ΕΠΙΣΤΡΟΦΗ THRESHOLD ( ΤΟ ΚΑΛΥΤΕΡΟ ΔΥΝΑΤΟ ΓΙΑ ΤΗΝ ΑΝΑΓΝΩΡΙΣΗ)
Threshold=MainThreshold;
BallLength=Initvar(1,2); % ΕΠΙΣΤΡΟΦΗ ΜΕΓΕΘΟΥΣ ΜΠΑΛΑΣ
RoboLength=Initvar(1,3); % ΕΠΙΣΤΡΟΦΗ ΜΕΓΕΘΟΥΣ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ

ColorArray(1,:)=Initvar(2,:); % ΕΠΙΣΤΡΟΦΗ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ ΧΡΩΜΑ (ΕΛΑΧΙΣΤΗ ΤΙΜΗ)

%   ColorArray(1,1)=0; ColorArray(1,2)=0; ColorArray(1,3)=0;   % mono gia test !!!!!!!

ColorArray(2,:)=Initvar(3,:); % ΕΠΙΣΤΡΟΦΗ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ ΧΡΩΜΑ (ΜΕΓΙΣΤΗ ΤΙΜΗ)
ColorArray(3,:)=Initvar(4,:); % ΕΠΙΣΤΡΟΦΗ ΜΠΑΛΑ ΧΡΩΜΑ (ΕΛΑΧΙΣΤΗ ΤΙΜΗ)
ColorArray(4,:)=Initvar(5,:); % ΕΠΙΣΤΡΟΦΗ ΜΠΑΛΑ ΧΡΩΜΑ (ΜΕΓΙΣΤΗ ΤΙΜΗ)
crtable(1:2)=Initvar(6,1:2);  % ΕΠΙΣΤΡΟΦΗ ΣΥΝΤΕΤΑΓΜΕΝΗΣ (X,Y) ΠΟΥ ΔΗΛΩΝΕΙ ΠΟΙΟ ΕΙΝΑΙ ΤΟ
                              % ΣΗΜΕΙΟ ΠΟΥ ΑΡΧΙΖΕΙ ΤΟ ΓΗΠΕΔΟ ΑΠΟ ΤΟ ΤΕΡΜΑ ΑΡΙΣΤΕΡΑ ΠΑΝΩ ΣΤΗΝ ΕΙΚΟΝΑ
                              % ΑΠΟ ΤΟΝ ΠΙΝΑΚΑ ΟΛΟΚΛΗΡΗΣ ΤΗΣ ΕΙΚΟΝΑΣ (ORPIC)

crtable(3:4)=Initvar(7,1:2);  % ΕΠΙΣΤΡΟΦΗ 2 ΤΙΜΩΝ X+,Y+ ΠΟΥ ΔΗΛΩΝΟΥΝ ΚΑΤΑ ΠΟΣΟ ΑΠΟ ΤΟ
                              % ΠΡΟΗΓΟΥΜΕΝΟ ΣΗΜΕΙΟ (X,Y) ΕΠΙΜΗΚΥΝΕΤΑΙ Ο ΚΑΘΕ ΑΞΟΝΑΣ, ΔΗΛΑΔΗ ΜΑΣ ΔΙΝΕΙ
```

```
66                                % ΤΟ ΜΗΚΟΣ ΚΑΙ ΤΟ ΠΛΑΤΟΣ ΚΑΙ ΤΟΥ ΓΗΠΕΔΟΥ ΣΕ PIXEL.
67
68 -  error==Initvar(8,1);          % FLAG ΠΟΥ ΜΑΣ ΔΗΛΩΝΕΙ ΕΑΝ Η ΠΡΟΗΓΟΥΜΕΝΗ ΑΝΑΓΝΩΡΙΣΗ ΗΤΑΝ ΕΠΙΤΥΧΗΣ
69
70 -  tt=[0 0 0 0 0 0 0 0 0 0]; % ΒΟΗΘΗΤΙΚΟΣ ΠΙΝΑΚΑΣ ΠΑΡΑΚΟΛΟΥΘΗΣΗΣ ΤΙΜΩΝ
71
72
73    % ΜΕΓΙΣΤΗ ΚΙΝΗΣΗ ΣΕ mm ΒΡΙΣΚΕΤΑΙ ΑΠΟ : sqrt((150^2)+(130^2))= 198,49 ~ 200cm = 2000 mm megisth
74
75    %%%%     1-> ΩΣ ΠΡΟΣ X -> 640cm/1500mm = 0.42 -> 1 cm = 2.34 pixel
76    %%%%     1-> ΩΣ ΠΡΟΣ Y -> 480cm/1300mm = 0.37 -> 1 cm = 2.70 pixel
77    %%%%    ------------------------------------------------------------
78    %%%%    M.O    (2.34 + 2.7) /2 = 1 cm -> 2.52 pixel
79    %%%%     ΟΠΟΤΕ ΓΙΑ 200 CM -> 200 CM * 2.52 = 504 PIX MAX ΜΕΓΙΣΤΗ ΠΑΡΑΣΤΑΣΗ
80
81    % initialize function
82
83    % ROBOT INFORMATION STRUCTURE - MATRIX
84    % roboinfo(1) -> Robo_id (1=ball, 2-3-4=our, 5-6-7=enemy)
85    % roboinfo(2) -> X
86    % roboinfo(3) -> Y
87    % roboinfo(4) -> Direction (0-360 depends on kick side)
88    % roboinfo(5) -> Type (0=ball, 1=our, -1=enemy)  ?
89    % roboinfo(6) -> Hasball 1=have ball, 0=does not have the ball
90
91
92 -  limx=crtable(3); % ΠΛΑΤΟΣ ΓΗΠΕΔΟΥ ΠΟΥ ΕΧΕΙ ΑΝΑΓΝΩΡΙΣΤΕΙ ΑΥΤΟΜΑΤΑ ΑΠΟ ΤΗΝ ΔΙΑΔΙΚΑΣΙΑ
93                     % (maininit.m) ΜΕΣΑ ΑΠΟ ΤΗΝ ΑΡΧΙΚΗ ΕΙΚΟΝΑ ΔΕΙΓΜΑΤΟΛΗΨΙΑΣ ΤΩΝ ΔΙΑΣΤΑΣΕΩΝ
94                     % (640 x 480)
95 -  limy=crtable(4); % ΜΗΚΟΣ ΠΟΥ ΕΧΕΙ ΑΝΑΓΝΩΡΙΣΘΕΙ ΑΥΤΟΜΑΤΑ ΑΠΟ ΤΗΝ ΔΙΑΔΙΚΑΣΙΑ
96                     % (maininit.m) ΜΕΣΑ ΑΠΟ ΤΗΝ ΑΡΧΙΚΗ ΕΙΚΟΝΑ ΔΕΙΓΜΑΤΟΛΗΨΙΑΣ
97                     % ΤΩΝ ΔΙΑΣΤΑΣΕΩΝ
98                     % (640 x 480)
99 -  lx=limx/150; % ΒΡΙΣΚΕΙ ΑΝΑΛΟΓΙΑ CM / PIXEL ΩΣ ΠΡΟΣ X ΑΞΟΝΑ  Π.Χ ΓΙΑ X = 400 PIXEL/150 cm -> 1cm = 2.66 pix
100 -  ly=limy/130; % ΒΡΙΣΚΕΙ ΑΝΑΛΟΓΙΑ CM / PIXEL ΩΣ ΠΡΟΣ Y ΑΞΟΝΑ ΓΙΑ ΤΗΝ ΕΚΤΕΛΕΣΗ ΤΗΣ ΠΟΙΟ ΑΚΡΙΒΗΣ ΚΙΝΗΣΗΣ
101
102 -  hcol=[0 0 0];             % ΑΡΧΙΚΟΠΟΙΗΣΗ ΤΙΜΩΝ ΚΑΙ ΔΗΜΙΟΥΡΓΙΑ ΤΟΥ ΤΥΠΟΥ ΤΗΣ ΜΕΤΑΒΛΗΤΗΣ
103                           % ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΕΙΤΑΙ ΓΙΑ ΤΗΝ ΑΠΟΘΗΚΕΥΣΗ ΤΗΣ
104                           % ΔΕΙΓΜΑΤΟΛΗΨΙΑΣ ΧΡΩΜΑΤΩΝ ΚΑΤΑ ΤΗΝ ΔΙΑΔΙΚΑΣΙΑ ΤΗΣ
105                           % ΑΝΑΓΝΩΡΙΣΗΣ ΑΝΤΙ/ΝΩΝ
106 -  footballgame=1;           % FLAG ΓΙΑ ΤΗΝ ΕΝΕΡΓΟΠΟΙΗΣΗ ΤΗΣ ΕΠΑΝΑΛΑΜΒΑΝΟΜΕΝΗΣ ΔΙΑΔΙΚΑΣΙΑΣ ΕΚΤΕΛΕΣΗΣ
107                           % ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ
108
109 -  angb=0;   % ΜΕΤΑΒΛΗΤΗ ΑΠΟΘΗΚΕΥΣΗΣ ΓΩΝΙΑΣ ΜΠΑΛΑΣ ΣΕ ΣΧΕΣΗ ΜΕ ΤΟΥΣ ΓΕΝΙΚΟΥΣ ΑΞΟΝΕΣ (0 - 360)
110             % ΣΥΜΦΩΝΑ ΜΕ ΤΗΝ ΜΕΤΑΒΛΗΤΗ MyTerm.
111 -  angr=0;   % ΜΕΤΑΒΛΗΤΗ ΑΠΟΘΗΚΕΥΣΗΣ ΓΩΝΙΑΣ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ ΣΕ ΣΧΕΣΗ ΜΕ ΤΟΥΣ ΓΕΝΙΚΟΥΣ ΑΞΟΝΕΣ (0 - 360)
112 -  yesball=0;  % ΜΕΤΡΗΤΗΣ ΕΠΑΝΑΛΗΨΕΩΝ (ΟΛΟΚΛΗΡΟΥ ΤΟΥ LOOP) ΓΙΑ ΑΡΙΘΜΟ ΕΜΦΑΝΙΣΗΣ ΤΗΣ ΜΠΑΛΑ ΣΤΟ
113               % ΓΗΠΕΔΟ , ΜΕΤΑ ΤΗΝ ΕΚΚΙΝΗΣΗ ΕΚΤΕΛΕΣΗΣ ΤΗΣ ΕΙΔΙΚΗΣ ΠΕΡΙΠΤΩΣΗΣ spec_case=1
114               % (goal), Η ΟΠΟΙΑ ΜΑΣ ΔΗΛΩΝΕΙ ΟΤΙ ΕΧΕΙ ΓΙΝΕΙ ΕΚΤΕΛΕΣΗ ΓΙΑ GOAL.
115 -  noball=0; % ΜΕΤΡΗΤΗΣ ΕΠΑΝΑΛΗΨΕΩΝ (ΟΛΟΚΛΗΡΟΥ ΤΟΥ LOOP) ΓΙΑ ΑΡΙΘΜΟ ΜΗ ΕΜΦΑΝΙΣΗΣ ΤΗΣ ΜΠΑΛΑΣ ΣΤΟ
116               % ΓΗΠΕΔΟ , ΜΕΤΑ ΤΗΝ ΕΚΚΙΝΗΣΗ ΕΚΤΕΛΕΣΗΣ ΤΗΣ ΕΙΔΙΚΗΣ ΠΕΡΙΠΤΩΣΗΣ spec_case=1
117               % (goal), Η ΟΠΟΙΑ ΜΑΣ ΔΗΛΩΝΕΙ ΟΤΙ ΕΧΕΙ ΓΙΝΕΙ ΕΚΤΕΛΕΣΗ ΓΙΑ GOAL.
118
119 -  frame=0; % ΠΙΝΑΚΑΣ ΠΟΥ ΠΕΡΙΕΧΕΙ ΤΗΝ ΕΙΚΟΝΑ ΜΙΑΣ ΤΟΠΙΚΗΣ ΔΕΙΓΜΑΤΟΛΗΨΙΑΣ ΠΑΝΩ ΣΤΗΝ ΕΠΙΦΑΝΕΙΑ
120            % ΕΝΟΣ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ, ΓΙΑ ΠΕΡΑΙΤΕΡΩ ΕΛΕΓΧΟ
121 -  Rplayer=0; % ΠΕΡΙΕΧΕΙ ΤΟ ROBOID(1 - 3) ΤΗΣ ΤΕΛΕΥΤΑΙΑΣ ΕΙΚΟΝΑΣ ΠΟΥ ΕΧΕΙ ΑΠΟΘΗΚΕΥΤΕΙ ΣΤΟΝ
122              % ΠΙΝΑΚΑ FRAME ΚΑΙ ΕΧΕΙ ΑΝΑΓΝΩΡΙΣΤΕΙ ΕΠΙΤΥΧΩΣ.
123 -  klisi=0;  % ΠΕΡΙΕΧΕΙ ΤΗΝ ΠΡΑΓΜΑΤΙΚΗ ΚΛΙΣΗ ΤΟΥ ΤΕΛΕΥΤΑΙΟΥ ΑΝΑΓΝΩΡΙΣΜΕΝΟΥ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ
124             % ΣΥΜΦΩΝΑ ΜΕ ΤΟΥΣ ΓΕΝΙΚΟΥΣ ΑΞΟΝΕΣ ( ΑΠΟ 0 - 360 ΜΟΙΡΕΣ)
125
126 -  B=1; % ΠΕΡΙΛΑΜΒΑΝΕΙ ΜΙΑ ΤΙΜΗ ΠΟΥ ΜΑΣ ΔΕΙΧΝΕΙ ΠΟΣΑ ΑΝΤ/ΜΕΝΑ ΕΧΟΥΝ ΑΝΑΓΝΩΡΙΣΤΕΙ ΜΕΤΑ ΤΗΝ
127        % ΕΚΤΕΛΕΣΗ ΤΗΣ ΕΝΤΟΛΗΣ BWBOUNDARIES ΣΥΜΦΩΝΑ ΜΕ ΤΟΥΣ ΠΑΡΑΜΕΤΡΟΥΣ ΠΟΥ ΕΧΟΥΝ ΔΟΘΕΙ (THRESHOLD κτλ.).
128
129 -  errcnt=0;
130
```

```
min_strmov=10; % ΒΟΗΘΗΤΙΚΗ ΜΕΤΑΒΛΗΤΗ ΠΟΥ ΑΛΛΑΖΕΙ ΑΝΑΛΟΓΑ ΜΕ ΤΗΝ ΠΕΡΙΠΤΩΣΗ ΤΙΜΕΣ ΚΑΙ ΔΙΝΕΙ ΤΗΝ ΑΠΟΣΤΑΣΗ
                % ΤΟΥ ΒΗΜΑΤΟΣ ΤΗΣ ΕΥΘΕΙΑΣ ΚΙΝΗΣΗΣ
dest_bball=0; % ΠΕΡΙΛΑΜΒΑΝΕΙ ΤΗΝ ΜΙΚΡΟΤΕΡΗ ΕΥΘΕΙΑ ΑΠΟΣΤΑΣΗ ΑΠΟ ΤΗΝ ΠΡΑΓΜΑΤΙΚΗ ΘΕΣΗ ΤΟΥ
              % ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ ΜΕΧΡΙ ΤΗΝ ΝΕΑ ΥΠΟΛΟΓΙΖΟΜΕΝΗ ΕΠΙΘΥΜΗΤΗ ΘΕΣΗ ΠΟΥ ΠΡΕΠΕΙ ΝΑ ΛΑΒΕΙ

splineRobo=0; % ΠΕΡΙΛΑΜΒΑΝΕΙ ΤΙΣ ΤΙΜΕΣ 0 ή 1 ΠΟΥ ΔΕΙΧΝΟΥΝ ΑΝΤΙΣΤΟΙΧΑ ΕΑΝ ΈΧΕΙ ΑΝΑΓΝΩΡΙΣΤΕΙ
              % ΕΠΙΤΥΧΩΣ=1 ΈΣΤΩ ΈΝΑΣ ΡΟΜΠΟΤΙΚΟΣ ΠΑΙΚΤΗΣ.
splineBall=0; % ΠΕΡΙΛΑΜΒΑΝΕΙ ΤΙΣ ΤΙΜΕΣ 0 ή 1 ΠΟΥ ΔΕΙΧΝΟΥΝ ΑΝΤΙΣΤΟΙΧΑ ΕΑΝ ΈΧΕΙ ΑΝΑΓΝΩΡΙΣΤΕΙ
              % ΕΠΙΤΥΧΩΣ=1 Η ΜΠΑΛΑ.

plotform='--om'; % ΕΠΙΛΟΓΗ ΜΟΡΦΗΣ ΕΜΦΑΝΙΣΗΣ ΤΗΣ ΓΡΑΜΜΗΣ ΣΤΟ PLOT WINDOW ,
                % ΓΙΑ ΤΗΝ ΑΠΕΙΚΟΝΙΣΗΣ ΤΩΝ ΕΠΙΘΥΜΗΤΩΝ ΚΙΝΗΣΕΩΝ Η ΕΜΦΑΝΙΣΗΣ ΤΩΝ ΓΩΝΙΩΝ.

Loopok=1; % ΒΟΗΘΗΤΙΚΗ ΜΕΤΑΒΛΗΤΗ ΓΙΑ ΤΗΝ ΔΡΟΜΟΛΟΓΗΣΗ ΤΗΣ ΡΟΗΣ ΤΗΣ ΛΕΙΤΟΥΡΓΙΑΣ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ

Thresholdfix=0.15;  % ΒΟΗΘΗΤΙΚΗ ΜΕΤΑΒΛΗΤΗ THRESHOLD
Thresholdfixdistance=40;
Thold=0;

%%term = [1 (crtable(4)-crtable(2))/2 crtable(3) (crtable(4)-crtable(2))/2];
%        x1         y1                x2               y2
term = [1 crtable(4)/2 crtable(3) crtable(4)/2]; % ΔΗΜΙΟΥΡΓΙΑ ΤΩΝ ΣΥΝΤΕΤΑΓΜΕΝΩΝ ΤΩΝ ΤΕΡΜΑΤΩΝ ΑΠΟ ΤΙΣ
                                                 % ΑΡΧΙΚΕΣ ΓΕΩΜΕΤΡΙΚΕΣ ΜΕΤΑΒΛΗΤΕΣ ΤΟΥ ΥΠΟΠΡΟΓΡΑΜΜΑΤΟΣ
                                                 % ΑΡΧΙΚΟΠΟΙΗΣΗΣ

semiterm=50; % ΒΟΗΘΗΤΙΚΗ ΜΕΤΑΒΛΗΤΗ ΠΟΥ ΠΕΡΙΕΧΕΙ ΤΟ ΜΙΣΟ ΤΟΥ ΠΛΑΤΟΥΣ ΤΟΥ ΤΕΡΜΑΤΟΣ
MyTerm=1; % ΒΟΗΘΗΤΙΚΗ ΜΕΤΑΒΛΗΤΗ ΓΙΑ ΤΗΝ ΕΠΙΛΟΓΗ ΤΟΥ ΕΧΘΡΙΚΟΥ ΤΕΡΜΑΤΟΣ ΑΡΙΣΤΕΡΟ=1, ΔΕΞΙΟ=2,
          % ΠΟΥ ΕΧΕΙ ΣΗΜΑΣΙΑ ΓΙΑ ΤΗΝ ΔΡΟΜΟΛΟΓΗΣΗ ΤΗΣ ΛΕΙΤΟΥΡΓΙΑΣ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ

serdata(1)=0; % ΠΙΝΑΚΑΣ ΠΟΥ ΠΕΡΙΛΑΜΒΑΝΕΙ ΤΙΣ ΤΙΜΕΣ ΤΩΝ ΚΙΝΗΣΕΩΝ. ΤΟ ΣΤΟΙΧΕΙΟ 1 ΠΕΡΙΛΑΜΒΑΝΕΙ
              % ΤΗΝ ΠΕΡΙΣΤΡΟΦΙΚΗ ΚΙΝΗΣΗ ΣΕ ΜΟΙΡΕΣ, ΚΑΙ ΑΝΑΛΟΓΑ ΜΕ ΤΗΝ ΠΕΡΙΠΤΩΣΗ ΔΙΑΜΟΡΦΩΝΟΝΤΑΙ ΚΑΙ ΟΙ
              % ΤΙΜΕΣ. ΕΠΙΣΗΣ ΤΟ ΣΤΟΙΧΕΙΟ 2 ΠΕΡΙΛΑΜΒΑΝΕΙ ΤΗΝ ΤΙΜΗ ΤΗΣ ΕΥΘΥΓΡΑΜΜΗΣ ΚΙΝΗΣΗΣ
              % ΔΙΑΜΟΡΦΩΝΜΕΝΕΣ ΑΝΑΛΟΓΑ ΜΕ ΤΙΣ ΥΠΟΛΟΓΙΣΜΕΝΕΣ ΚΛΙΜΑΚΕΣ.

sdc=-1; % FLAG ΠΟΥ ΕΝΕΡΓΟΠΟΙΕΙ ΤΗΝ ΚΙΝΗΣΗ ΤΗΣ ΠΕΡΙΣΤΡΟΦΗΣ Η ΤΗΝ ΚΙΝΗΣΗ ΤΗΣ ΕΥΘΕΙΑΣ ΚΙΝΗΣΗΣ
        % Π.Χ ΓΙΑ ΚΙΝΗΣΗ ΠΕΡΙΣΤΡΟΦΗΣ Η ΕΝΕΡΓΟΠΟΙΗΣΗ=2 ,ΕΝΩ ΓΙΑ ΚΙΝΗΣΗ

serang=0; % ΜΕΤΑΒΛΗΤΗ ΠΟΥ ΠΕΡΙΛΑΜΒΑΝΕΙ ΤΗΝ ΤΙΜΗ ΤΗΣ ΠΕΡΙΣΤΡΟΦΙΚΗΣ ΚΙΝΗΣΗΣ. ΣΕ ΠΕΡΙΠΤΩΣΗ
          % ΔΕΞΙΟΣΤΡΩΦΗΣ  ΚΙΝΗΣΗΣ ΟΙ ΤΙΜΕΣ ΕΙΝΑΙ ΑΠΟ 0 - 180 ΜΟΙΡΕΣ, ΔΗΛΑΔΗ ΜΕΓΙΣΤΗ
          % ΠΕΡΙΣΤΡΟΦΗ 180. ΣΕ ΠΕΡΙΠΤΩΣΗ ΑΡΙΣΤΕΡΟΣΤΡΩΦΗΣ ΚΙΝΗΣΗΣ ΟΙ ΤΙΜΕΣ ΕΙΝΑΙ ΑΠΟ
          % 181 - 360 ΜΟΙΡΕΣ. ΓΙΑ ΠΑΡΑΔΕΙΓΜΑ ΕΑΝ ΕΧΩ 250 ΜΟΙΡΕΣ ΤΙΜΗ ΣΗΜΑΙΝΕΙ
          % 360-250 = 110 ΜΟΙΡΕΣ ΑΡΙΣΤΕΡΟΣΤΡΟΦΑ.

showstatus=1; % ΜΕΤΑΒΛΗΤΗ  ΠΟΥ ΕΝΕΡΓΟΠΟΙΕΙ=1 ΤΟ ΠΑΡΑΘΥΡΟ PLOT ΚΑΙ ΤΗΝ ΕΜΦΑΝΙΣΗ ΤΩΝ ΓΡΑΜΜΩΝ ΚΙΝΗΣΕΩΝ
debugmode=1;  % ΜΕΤΑΒΛΗΤΗ  ΠΟΥ ΕΝΕΡΓΟΠΟΙΕΙ=1 ΕΠΙΠΛΕΩΝ ΒΟΗΘΗΤΙΚΕΣ ΓΡΑΜΜΕΣ ΓΙΑ ΟΠΤΙΚΟ ΒΟΗΘΗΜΑ
              % ΣΤΟ ΠΑΡΑΘΥΡΟ ΤΟΥ PLOT

objstatus=0; % FLAG ΠΟΥ ΕΠΙΤΡΕΠΕΙ ΤΗΝ ΛΕΙΤΟΥΡΓΙΑ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ ΟΤΑΝ ΥΠΑΡΧΕΙ
             % ΑΝΑΓΝΩΡΙΣΜΕΝΟΣ ΡΟΜΠΟΤΙΚΟΣ ΠΑΙΚΤΗΣ ΚΑΙ ΜΠΑΛΑ, ΕΠΑΝΑΛΑΜΒΑΝΕΙ ΤΟ ΓΕΝΙΚΟ LOOP ΜΕΧΡΙ ΝΑ
             % ΥΠΑΡΞΕΙ ΣΩΣΤΗ ΑΝΑΓΝΩΡΙΣΗ

moveenable=0; % FLAG ΠΟΥ ΕΠΙΤΡΕΠΕΙ (ΤΙΜΗ=1) ΤΗΝ ΑΠΟΣΤΟΛΗ ΣΕΙΡΙΑΚΩΝ ΔΕΔΟΜΕΝΩΝ
              % ΑΠΟ ΤΟΝ ΠΟΜΠΟ ΣΤΟ ΡΟΜΠΟΤΙΚΟ ΠΑΙΚΤΗ.

togoal=0; % FLAG ΓΙΑ ΕΛΕΓΧΟ ΚΑΤΑΣΤΑΣΗ GOAL. ΜΕ ΤΙΜΗ=1 ΕΧΕΙ ΔΟΘΕΙ ΕΝΤΟΛΗ ΓΙΑ
          % ΕΚΚΙΝΗΣΗ ΕΚΤΕΛΕΣΗΣ GOAL, ΜΕ ΤΙΜΗ=2 ΕΧΕΙ ΕΚΤΕΛΕΣΤΕΙ ΤΟ ΣΟΥΤ ΚΑΙ ΕΧΕΙ
          % ΕΝΕΡΓΟΠΟΙΗΘΕΙ Ο ΕΛΕΓΧΟΣ ΓΙΑ ΕΠΙΤΥΧΙΑ GOAL. ΜΕ ΤΙΜΗ=3 ΕΧΕΙ ΕΝΕΡΓΟΠΟΙΗΘΕΙ Η
          % ΛΕΙΤΟΥΡΓΙΑ ΕΚΤΕΛΕΣΗΣ ΠΑΝΗΓΥΡΙΣΜΩΝ.

motime=0; % ΒΟΗΘΗΤΙΚΗ ΜΕΤΑΒΛΗ

timcnt=0.00001; % ΒΟΗΘΗΤΙΚΗ ΜΕΤΑΒΛΗ
tim=0; % ΒΟΗΘΗΤΙΚΗ ΜΕΤΑΒΛΗ

goals=0; % ΜΕΤΡΗΤΗΣ ΤΩΝ GOAL ΡΟΥ ΕΧΟΥΝ ΕΠΙΤΥΧΕΙ ΤΑ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗS
```

```
196
197 - ser_commands=1; % ΜΕΤΡΗΤΗΣ ΣΕΙΡΙΑΚΩΝ ΑΠΟΣΤΟΛΩΝ ΠΡΟΣ ΤΑ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗΣ
198
199 - spec_case=0; % ΕΙΔΙΚΗ ΜΕΤΑΒΛΗΤΗ ΠΟΥ ΕΛΕΓΧΕΙ ΤΙ ΕΙΔΟΥΣ ΚΙΝΗΣΗ ΕΧΕΙ ΕΤΕΛΕΣΤΕΙ ΑΝΑΛΟΓΑ ΤΗΝ
200           % ΕΙΔΙΚΗ ΠΕΡΙΠΤΩΣΗ. ΕΚΤΕΛΟΥΝΤΑΙ ΑΣΧΕΤΑ ΣΕ ΤΙ ΚΑΤΑΣΤΑΣΗ ΒΡΙΣΚΕΤΑΙ ΤΟ ΣΥΣΤΗΜΑ ΜΕ ΤΗΝ
201           % ΟΙ ΠΕΡΙΠΤΩΣΕΙΣ ΕΙΝΑΙ:
202           %  ΤΙΜΗ=1  ΕΙΝΑΙ Η ΕΚΚΙΝΗΣΗ ΕΚΤΕΛΕΣΗΣ ΣΟΥΤ (ΕΥΘΥΓΡΑΜΜΗΣ ΚΙΝΗΣΗΣ) ΓΙΑ GOAL
203           %  ΤΙΜΗ=4  ΕΚΤΕΛΕΣΗ ΠΡΟΩΘΗΣΗΣ ΜΠΑΛΑΣ, (ΕΥΘΥΓΡΑΜΜΗ ΚΙΝΗΣΗ), ΑΠΛΑ ΓΙΑ ΝΑ ΠΡΟΧΩΡΗΣΕΙ ΠΡΟΣ ΤΑ ΕΜΠΡΟΣ
204           %  ΤΙΜΗ=5  ΠΕΡΙΣΤΡΟΦΗ ΣΕ ΣΥΓΚΕΚΡΙΜΕΝΗ ΘΕΣΗ , ΓΙΑ ΝΑ ΕΤΟΙΜΑΣΤΕΙ ΜΕΤΑ ΓΙΑ ΤΗΝ ΕΚΤΕΛΕΣΗ ΣΟΥΤ
205           %  ΤΙΜΗ=3  ΔΕΝ ΕΚΤΕΛΕΙΤΕ ΚΑΠΟΙΑ ΕΙΔΙΚΗ ΠΕΡΙΤΠΩΣΗ ΛΟΓΩ ΤΩΝ ΘΕΣΕΩΝ ΤΟΥΣ ΟΠΟΤΕ ΧΡΗΣΙΜΟΠΟΙΟΥΝΤΑΙ
206           %          ΟΙ ΚΑΝΟΝΙΚΟΙ ΕΛΕΓΧΟΙ ΓΙΑ ΤΙΣ ΚΙΝΗΣΕΙΣ.
207           %  ΤΙΜΗ=0  ΔΕΝ ΕΚΤΕΛΕΙΤΕ ΚΑΠΟΙΑ ΕΙΔΙΚΗ ΠΕΡΙΤΠΩΣΗ ΕΠΕΙΔΗ ΤΟ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ ΔΕΝ
208           %          ΒΡΙΣΚΕΤΑΙ ΠΙΣΩ ΑΠΟ ΤΗΝ ΜΠΑΛΛΑ ΟΠΟΤΕ ΧΡΗΣΙΜΟΠΟΙΟΥΝΤΑΙ ΟΙ ΚΑΝΟΝΙΚΟΙ ΕΛΕΓΧΟΙ ΕΚΤΕΛΕΣΗΣ ΚΙΝΗΣΕΩΝ
209
210 - roboy=0;
211 - robox=0;
212 - bally=0;
213 - ballx=0;
214 - enemy=[0 0 0];
215 - temprobo=[0];
216 - tempthreshold=Threshold;
217 - lastthreshold=0.50;
218 - goaldistance=250;
219
220
221 - comtext=['S T R' 'A N G']; % ΒΟΗΘΗΤΙΚΗ ΜΕΤΑΒΛΗΤΗ
222
223 - if (MyTerm==1)  % ΑΡΧΙΚΟΠΟΙΗΣΗ ΠΙΝΑΚΑ term(..) ΤΗΣ ΘΕΣΗΣ ΤΟΥ ΕΧΘΡΙΚΟΥ ΤΕΡΜΑΤΟΣ ΣΤΟΝ (X,Y)
224                   % ΑΞΟΝΑ ΑΝΑΛΟΓΑ ΜΕ ΤΟ FLAG MyTerm (ΑΡΙΣΤΕΡΟ ή ΔΕΞΙΟ)
225 -     termx=term(3)+0.0001;
226 -     termy=term(4)+0.0001;
227 - else
228 -     termx=term(1)+0.0001;
229 -     termy=term(2)+0.0001;
230 - end
231
232 - X=[1 10000;2 10000;3 10000]; % ΑΡΧΙΚΟΠΟΙΗΣΗ ΒΟΗΘΗΤΙΚΟΥ ΠΙΝΑΚΑ X, ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΕΙΤΑΙ ΓΙΑ
233                               % ΑΠΟΘΗΚΕΥΣΗ ΤΩΝ ΠΕΡΙΜΕΤΡΩΝ ΤΗΣ ΚΑΘΕ ΠΕΡΙΟΧΗΣ ΠΟΥ
234                               % ΒΡΙΣΚΕΤΑΙ ΣΤΗΝ ΕΠΙΦΑΝΕΙΑ ΤΟΥ ΚΑΘΕ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ ΜΕΤΑ ΑΠΟ
235                               % ΕΠΙΤΥΧΗ ΑΓΝΩΡΙΣΗ ΓΙΑ ΤΟΝ ΔΙΑΧΩΡΙΣΜΟ ΤΟΥΣ
236
237
238 - for i=1:2 % ΕΚΚΙΝΗΣΗ ΔΙΑΔΙΔΑΚΑΣΙΑΣ ΛΗΨΗΣ ΕΙΚΟΝΑΣ (2 ΔΕΙΓΜΑΤΟΛΗΨΙΕΣ) ΓΙΑ ΝΑ ΕΚΚΙΝΗΣΕΙ ΤΟ ΕΞΩΤΕΡΙΚΟ
239             % ΠΡΟΓΡΑΜΜΑ ΕΠΙΚΟΙΝΩΝΙΑΣ ΜΕ ΤΗΝ ΚΑΜΕΡΑ ΓΙΑ ΑΠΟΦΥΓΗ ΚΑΘΥΣΤΕΡΗΣΗΣ
240 -      pic=vcapg2(1); % ΛΗΨΗ ΕΙΚΟΝΑΣ ΣΕ ΜΟΡΦΗ RGB ΜΕ (X,Y) ΣΥΝΤΕΤΑΓΜΕΝΕΣ ΑΠΟΘΗΚΕΥΜΕΝΗ ΣΤΟΝ ΠΙΝΑΚΑ pic.
241 - end
242
243 -  figure(1); % ΕΜΦΑΝΙΖΕΤΕ ΕΝΑ ΠΑΡΑΘΥΡΟ ΜΕ ΟΝΟΜΑ FIGURE 1 ΤΟ ΟΠΟΙΟ ΕΙΝΑΙ ΕΤΟΙΜΟ ΝΑ ΔΕΧΤΕΙ
244               % ΤΟ ΕΠΟΜΕΝΟ PLOT
245
246   % ##########################################################################
247   % ##########################################################################
248   %
249   %                              L   O   O  P
250   %
251   % ##########################################################################
252   % ##########################################################################
253
254 - i=0;
255 - while (footballgame==1)     % ΕΚΚΙΝΗΣΗ ΑΔΕΙΑΛΗΠΤΗΣ ΛΕΙΤΟΥΡΓΙΑΣ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ
256 -      tic;
257
258 -        if ( (roboy<Thresholdfixdistance) || (abs(roboy-crtable(4))) <Thresholdfixdistance)
259 -            Threshold=Thresholdfix;  % ΕΝΕΡΓΟΠΟΙΗΣΗ THREHOLD ΑΝΑΛΟΓΑ ΤΗΝ ΠΕΡΙΠΤΩΣΗ , ΠΡΩΤΗ ΦΟΡΑ Ή ΕΠΑΝΑΛΗΨΗ ΤΗΣ ΑΝΑΓΝΩΡΗΣΗ
260 -        else
```

```
            Threshold=MainThreshold;
        end

        orpic=vcapg2; % ΛΗΨΗ ΕΙΚΟΝΑΣ

        pic=imcrop(orpic,crtable); % ΑΝΤΙΓΡΑΦΗ ΣΤΟΝ ΠΙΝΑΚΑ pic ΟΛΟΚΛΗΡΗΣ ΤΗς ΠΕΡΙΟΧΗΣ ΤΟΥ ΓΗΠΕΔΟ ΣΥΜΦΩΝΑ ΜΕ ΤΑ ΓΕΩΜΕΤΡΙΚΑ
                                   % ΤΟΥ ΣΤΟΙΧΕΙΑ ΠΟΥ ΒΡΙΣΚΟΝΤΑΙ ΣΤΟΝ ΠΙΝΑΚΑ crtable,
                                   % ΚΑΙ ΥΠΑΡΧΟΥΝ ΣΤΗΝ ΑΡΧΙΚΗ ΕΙΚΟΝΑ

        % =================  1. ΕΥΡΕΣΗ ΣΟΣΤΟΥ ΑΡΙΘΜΟΥ ΑΝΤΙΚΕΙΜΕΝΩΝ

        obj=0;  % ΒΟΗΘΗΤΙΚΟΣ ΠΙΝΑΚΑΣ ΑΠΟΘΗΚΕΥΣΗΣ ΤΩΝ INDEX ΤΩΝ ΣΩΣΤΩΝ ΑΝΤ/ΝΩΝ ΜΕΤΑ ΑΠΟ ΕΛΕΓΧΟ ΜΕΓΕΘΟΥΣ
        i=1; % ΒΟΗΘΗΤΙΚΟΣ ΜΕΤΡΗΤΗΣ
        objcnt=1000; % ΠΕΡΙΕΧΕΙ ΤΟ ΠΛΗΘΟΣ ΤΩΝ ΑΝΑΓΝΩΡΙΣΜΕΝΩΝ ΑΝΤ/ΝΩΝ ΚΑΙ ΧΡΗΣΙΜΟΠΟΙΕΙΤΑΙ ΓΙΑ ΤΗΝ
                     % ΛΕΙΤΟΥΡΓΙΑ ΔΙΑΧΩΡΙΣΜΟΥ ΑΝΤ/ΝΩΝ ΣΕ ΚΑΤΗΓΟΡΙΕΣ (ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ, ΜΠΑΛΑΣ, ΕΝΕΜΥ) .

        if (obj(1)~=0) % ΕΛΕΓΧΟΣ ΓΙΑ ΤΟ ΑΝ ΕΧΟΥΝ ΒΡΕΘΕΙ ΑΝΤ/ΜΕΝΑ
            objcnt=1; % ΜΕ ΤΙΜΗ=1 ΤΟ FLAG ΕΝΕΡΓΟΠΟΙΕΙ ΤΟΝ ΔΙΑΧΩΡΙΣΜΟ ΤΩΝ ΑΝΤΙΚΕΙΜΕΝΩΝ
        end

        splineBall=0;
        splineRobo=0;

        enmcnt=1; % ΜΕΤΡΗΤΗΣ ΠΛΗΘΟΥΣ ΕΧΘΡΙΚΩΝ ΑΝΤ/ΝΩΝ

        % ========================================================================
        %                         objects - ΑΝΤΙΚΕΙΜΕΝΑ
        % ========================================================================

        % ΒΟΗΘΗΤΙΚΕΣ ΜΕΤΑΒΛΗΤΕΣ
        N=1;
        objind=1;
        F=1;
        Z=B;

        % ΔΗΜΙΟΥΡΓΙΑ ΔΥΝΑΜΙΚΟΥ THRESHOLD ΚΑΙ ΟΡΙΩΝ ΤΟΥ ΠΟΥ ΠΙΘΑΝΟ ΝΑ ΧΡΕΙΑΣΤΟΥΝ
        temprobo(1)=0;
        tempthreshold=Threshold;

        if ( Threshold<0.6)
            tempthreshold=Threshold;
        else
            Threshold=MainThreshold;
            tempthreshold=Threshold;
        end;

        if (Threshold*2<0.6)
            lastthreshold=Threshold*2;
        else
            lastthreshold=0.6;
        end

        Thold=abs(tempthreshold);
        thres_step=0.01;

        % ΑΝΑΖΗΤΗΣΗ ΑΝΤΙΚΕΙΜΕΝΩΝ (ΡΟΜΠΟΤΙΚΩΝ ΠΑΙΚΤΩΝ) ΣΥΜΦΩΝΑ ΜΕ ΤΟ ΣΧΗΜΑ ,
        % ΤΟ ΜΕΓΕΘΟΣ ΚΑΙ ΤΟΠΟΘΕΤΗΣΗ ΧΑΡΑΚΤΗΡΙΣΙΤΚΩΝ ΤΟΥΣ ΣΤΟΝ ΠΙΝΑΚΑ  temprobo()

        [B,L] = bwboundaries(im2bw(pic,0.1),'noholes'); % ΕΥΡΕΣΗ ΑΝΤΙΚ/ΝΩΝ ΑΠΟ ΤΗΝ ΕΙΚΟΝΑ pic ΣΥΜΦΩΝΑ ΜΕ ΤΟ ΔΕΔΟΜΕΝΟ
                                                        % THRESHOLD ΚΑΙ ΑΡΙΘΜΗΣΗ ΤΟΥΣ .
        stats = regionprops(L,'Image','Centroid','BoundingBox','Eccentricity');
                                                            % ΕΥΡΕΣΗ ΧΑΡΑΚΤΗΡΙΣΤΙΚΩΝ ΚΑΘΕ ΑΝΤ/ΝΟΥ
                                                            % (ΜΕΓΕΘΟΣ ΣΕ PIXEL , ΚΕΝΤΡΟ (X,Y), ΠΕΡΙΜΕΤΡΟΣ ΣΕ PIXEL)
        R=1;
        i=0;
```

```
        objmax=length(B);

        while ( R<=objmax )
               if (abs(stats(R).Eccentricity-0.5)<0.3)
                   if ( (length(B{R})>40) && (length(B{R})<100) )
                       i=i+1;
                       temprobo(i)=R;
                   else
                       s1=( (stats(R).BoundingBox(3) > BallLength*0.5) && (stats(R).BoundingBox(3) < BallLength*2) );
                       s2=( (stats(R).BoundingBox(4) > BallLength*0.5) && (stats(R).BoundingBox(4) < BallLength*2) );
                       if (s1 || s2)
                           hcol=impixel(pic,stats(R).Centroid(1),stats(R).Centroid(2));     % ΕΛΕΓΧΟΣ ΓΙΑ ΤΟ ΧΡΩΜΑ
                           if ((hcol>=ColorArray(3,:)) & (hcol<=ColorArray(4,:))) % ΜΠΑΛΑ ΠΟΡΤΟΚΑΛΙ
                               splineBall=R;
                               hasball=0;
                           end
                       end
                   end
               end
               R=R+1;
        end

        if (temprobo(1)==0)
            if ( (abs(tempthreshold)>0.05) && (abs(tempthreshold)<=lastthreshold) )
                tempthreshold=((-1)*tempthreshold)+thres_step;
                thres_step=thres_step+0.01;
                Thresholdfix=abs(tempthreshold); % ΓΙΑ ΑΝΤΙΚΕΙΜΕΝΑ ΔΥΣΚΟΛΑ ΣΤΟ THRESHOLD
            else
                Thold=0.6; % ΓΙΑ ΕΞΟΔΟ ΑΠΟ ΤΟ LOOP
                Thresholdfix=MainThreshold;
            end
        else
            Thresholdfix=abs(tempthreshold);
        end

        objmax=i; % ΑΡΙΘΜΟΣ ΙΣΟΣ ΜΕ ΤΑ ΑΝΤΙΚΕΙΜΕΝΑ ΜΕΓΕΘΟΥΣ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ
        R=1;

        % % ==================  2. LOOP ΓΙΑ ΔΙΑΧΩΡΙΣΜΟ ΑΝΤ/ΝΩΝ ΣΕ ΚΑΤΗΓΟΡΙΕΣ

        while ( (R<=objmax) && (splineRobo==0) && (temprobo(1)>0)   )

            stats = regionprops(L,'Image','Centroid','BoundingBox','Eccentricity');
             % ΕΥΡΕΣΗ ΧΑΡΑΚΤΗΡΙΣΤΙΚΩΝ ΚΑΘΕ ΑΝΤ/ΝΟΥ (ΜΕΓΕΘΟΣ ΣΕ PIXEL , Κ.Β (X,Y), ΠΕΡΙΜΕΤΡΟΣ ΣΕ PIXEL)
            if (R<=objmax)
                st_thres=Threshold;
                thres_step=0.01;
                thres_end=Threshold+Thresholdfix;
                akk=2;
                helpstat=stats(temprobo(R));
                i=i+1;

                while ( (splineRobo==0) && (thres_step<thres_end)) % ΕΠΑΝΑΛΗΦΗ ΜΕ ΔΙΑΦΟΡΕΤΙΚΟ THRESHOLD

                   if (length(Z)>0) % ΕΛΕΓΧΟΣ ΑΠΟ ΤΗΝ ΑΡΧΙΚΗ ΕΙΚΟΝΑ pic
                       frame=helpstat.Image; %  ΤΟΠΙΚΗ ΔΕΙΓΜΑΤΟΛΗΨΙΑ ΕΝΟΣ ΑΝΤΙΚΕΙΜΕΝΟΥ ΓΙΑ ΠΕΡΑΙΤΕΡΩ ΕΛΕΓΧΟ
                       [HB,HL]=bwboundaries(frame,'holes'); % ΚΑΤΑΜΕΤΡΗΣΗ ΚΕΝΩΝ ΤΜΗΜΑΤΩΝ -
                                                            % ΑΝΤ/ΝΩΝ ΣΤΗΝ ΤΟΠΙΚΗ ΔΕΙΓΜΑΤΟΛΗΨΙΑ
                                                            % ΜΕ ΑΠΟΘΗΚΕΥΣΗ ΤΟΥ ΑΡΙΘΜΟΥ ΤΟΥΣ ΣΤΟ HB ΚΑΙ
                                                            % ΤΟΥ ΣΧΗΜΑΤΟΣ ΤΟΥΣ ΣΤΟ HL
                       Hstats =regionprops(HL,'Centroid','FilledArea'); % ΑΝΑΓΝΩΣΗ ΧΑΡΑΚΤΗΡΙΣΤΙΚΩΝ ΤΟΥ ΤΡΕΧΟΝΤΟΣ ΑΝΤ/ΜΕΝΟ
                                        % ΚΑΙ ΑΠΟΘΗΚΕΥΣΗ ΤΟΥΣ ΣΤΟΥΣ ΑΝΤΙΣΤΟΙΧΟΥΣ ΠΑΡΑΜΕΤΡΟΥΣ ΤΗΣ ΕΝΤΟΛΗΣ regionprops
                       akk=length(HB); % ΕΥΡΕΣΗ ΠΛΗΘΟΥΣ ΑΝΤ/ΝΩΝ.


```

```
                % ΤΜΗΜΑ ΕΥΡΕΣΗΣ ΚΑΙ ΑΝΑΛΥΣΗΣ ΠΕΡΙΟΧΩΝ ΤΟΥ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ

                if (akk==3) % ΕΑΝ ΙΣΧΥΕΙ ΤΟΤΕ ΕΧΕΙ ΒΡΕΙ ΣΙΓΟΥΡΑ ΤΟ ΜΕΓΑΛΟ ΚΑΙ ΤΟ ΜΕΣΑΙΟ ΤΜΗΜΑ,
                                                        % ΟΠΟΤΕ ΠΡΟΧΩΡΑΕΙ ΣΕ ΑΝΑΓΝΩΡΙΣΗ.
                    X=[0 0]; % ΑΡΧΙΚΟΠΟΙΗΣΗ ΤΟΥ ΒΟΗΘΗΤΙΚΟΥ ΠΙΝΑΚΑ ΑΠΟΘΗΚΕΥΣΗΣ ΠΕΡΙΟΧΩΝ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ

                    for i=1:akk
                        X(i,2)=Hstats(i).FilledArea; % ΑΝΑΓΝΩΣΗ ΠΕΡΙΜΕΤΡΟΥ ΣΧΗΜΑΤΟΣ
                    end

                    [Y,NDX] = sort(X(:,:)); % ΚΑΤΑΧΩΡΗΣΗ ΑΠΟ ΜΙΚΡΟ ΣΕ ΜΕΓΑΛΟ ΤΜΗΜΑ
                          % NDX(1,2) -> MIN
                          % NDX(2,2) -> MIDDLE
                          % NDX(3,2) -> MAX

                          % ΠΑΡΑΚΑΤΩ ΒΡΙΣΚΟΝΤΑΙ ΤΟ ΚΕΝΤΡΟ (X,Y) ΤΟΥ ΜΕΣΑΙΟΥ (ΣΕ ΜΕΓΕΘΟΣ)
                          % ΑΝΑΓΝΩΡΙΣΜΕΝΟΥ ΤΜΗΜΑΤΟΣ ΤΟΥ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ
                          % ΚΑΙ ΑΠΟΘΗΚΕΥΕΤΑΙ ΣΤΙΣ ΑΝΤΙΣΤΟΙΧΕΣ ΜΕΤΑΒΛΗΤΕΣ (ImageCentroidX, ImageCentroidY)
                          % ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΟΥΝΤΑΙ ΓΙΑ ΤΟΝ ΧΡΩΜΑΤΙΚΟ ΕΛΕΓΧΟ.

                    ImageCentroidX=Hstats(NDX(end-1,2)).Centroid(1)+helpstat(1).BoundingBox(1); % ΑΠΟ 3,5x3,5 +
                                                                                                % ΟΡΙΑ ΠΑΟ ΤΗΝ ΕΙΚΟΝΑ pic
                    ImageCentroidY=Hstats(NDX(end-1,2)).Centroid(2)+helpstat(1).BoundingBox(2);
                    hcol=impixel(pic,ImageCentroidX,ImageCentroidY); % ΔΕΙΓΜΑΤΟΛΗΨΙΑ ΧΡΩΜΑΤΟΣ ΑΠΟ ΤΙΣ ΣΥΝΤΕΤΑΓΜΕΝΕΣ
                                                                     % (ImageCentroidX,ImageCentroidY)

                    if (hcol<=ColorArray(2,:)) % ΤΙΜΕΣ ΑΝΑΜΕΣΑ ΑΠΟ 0,0,0 ΕΩΣ COLORARRAY(2) ΧΡΩΜΑΤΙΚΟΣ ΕΛΕΓΧΟΣ
                                               % ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ (ΑΝΑΜΕΣΑ ΣΤΙΣ ΑΝΟΧΕΣ)

                        a1=Y(end-1)/Y(end); % ΔΗΜΙΟΥΡΓΙΑ ΛΟΓΟΥ ΤΜΗΜΑΤΩΝ ΜΕΣΟΥ/ΜΕΓΑΛΟΥ
                        a2=Y(end-2)/Y(end-1); % ΔΗΜΙΟΥΡΓΙΑ ΛΟΓΟΥ ΤΜΗΜΑΤΩΝ ΜΙΚΡΟΥ/ΜΕΣΟΥ
                        loga1a2=a1/a2; % ΔΗΜΙΟΥΡΓΙΑ ΛΟΓΟΥ ΑΝΑΓΝΩΡΙΣΗΣ ID

                        if (loga1a2<2) % ΕΛΕΓΧΟΣ ΤΟΥ ΛΟΓΟΥ loga1a2 ΚΑΙ ΣΥΜΦΩΝΑ ΜΕ ΑΥΤΟΝ ΔΙΔΕΤΑΙ ΤΟ ID ΣΤΟ
                                       % ΣΥΓΚΕΚΡΙΜΕΝΟ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ
                            Rplayer=1; % ΔΗΛΩΣΗ ID=1 ΣΤΟ ΣΥΓΚΕΚΡΙΜΕΝΟΥ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ

                        else
                            if (loga1a2>6)
                                Rplayer=2;
                            else
                                Rplayer=3;
                            end;
                        end;

                        splineRobo=temprobo(R);
                        x=Hstats(NDX(2,2)).Centroid(1)-Hstats(NDX(1,2)).Centroid(1); % ΕΥΡΕΣΗ ΑΠΟΣΤΑΣΗΣ X
                                                                                     % ΜΙΚΡΟΥ-ΜΕΓΑΛΟΥ ΤΕΤΡΑΓΩΝΟΥ
                        y=Hstats(NDX(2,2)).Centroid(2)-Hstats(NDX(1,2)).Centroid(2); % ΕΥΡΕΣΗ ΑΠΟΣΤΑΣΗΣ
                                                                                     % ΜΙΚΡΟΥ-ΜΕΓΑΛΟΥ ΤΕΤΡΑΓΩΝΟΥ
                        r=sqrt((x^2)+(y^2)); % ΥΠΟΤΕΙΝΟΥΣΑ ΓΙΑ x ΚΑΙ y
                        sin_a=x/r; % ΗΜΙΤΟΝΟ

                        if (sin_a<0)
                            sin_a=sin_a*(-1); % ΔΗΜΙΟΥΡΓΙΑ ΠΑΝΤΑ ΘΕΤΙΚΟΥ ΗΜΙΤΟΝΟΥ
                        end;

                        a=asind(sin_a)-1; % ΕΥΡΕΣΗΣ ΓΩΝΙΑΣ ΑΠΟ 0-90 ΜΟΙΡΕΣ

                        % ΤΜΗΜΑ ΕΥΡΕΣΗΣ ΓΩΝΙΑΣ a , ΔΗΜΙΟΥΡΓΙΑ ΚΑΙ ΚΑΤΑΧΩΡΗΣΗ ΤΗΣ
                        % ΓΩΝΙΑΣ ΚΛΙΣΗΣ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ ΣΤΗ ΜΕΤΑΒΛΗΤΗ klisi

                        if (x>0)
                            if (y>0) %270-360
                                % t=4;
```

```
                                klisi=360-a;
                                klisidif=klisi-360;
                            else %180-270
                                % t=3;
                                klisi=180+a;
                                klisidif=180-klisi;
                            end;
                        else
                            if (y>0) %0-90
                                % t=2;
                                klisi=abs(a);
                                klisidif=klisi;
                            else % 90-180
                                % t=3;
                                klisi=180-a;
                                klisidif=180-klisi;
                            end;
                        end;
                    else
                        enemy=1; % logo xromatikoy llegxoy
                    end;
                else % logo sxhmatos
                    enemy=1;
                end

                if (splineRobo==0)
                    st_thres=((-1)*st_thres)+thres_step;
                    localobject=imcrop(pic,stats(temprobo(R)).BoundingBox);
                    [Z,F] = bwboundaries(im2bw(localobject,abs(st_thres)),'noholes'); % ΕΛΕΓΧΟΣ ΤΟΥ ΤΟΠΙΚΟΥ
                                                                                      % ΑΝΤΙΚΕΙΜΕΝΟΥ  ΜΕ ΤΟ ΝΕΟ THREHOLD
                    % ΕΥΡΕΣΗ ΧΑΡΑΚΤΗΡΙΣΤΙΚΩΝ ΚΑΘΕ ΑΝΤ/ΝΟΥ (ΜΕΓΕΘΟΣ ΣΕ PIXEL , Κ.Β (X,Y), ΠΕΡΙΜΕΤΡΟΣ ΣΕ PIXEL)
                    Rstats = regionprops(F,'Image','Centroid','BoundingBox','Eccentricity');

                    helpstat=Rstats;
                    thres_step=thres_step+0.01;
                end
            else
                 thres_step=thres_end; % ΓΙΑ ΝΑ ΒΓΕΙ ΕΚΤΟΣ LOOP ΟΤΑΝ ΔΕΝ ΕΧΕΙ ΒΡΕΙ
                                       % ΑΠΟ ΣΦΑΛΜΑ ΚΑΝΕΝΑ ΑΝΤΙΚΕΙΜΕΝΟ ΚΑΙ ΝΑ ΜΗΝ ΚΟΛΛΗΣΕΙ
            end
        end
    end
    R=R+1;
end

     % ============================================================
     % ΠΡΟΕΤΟΙΜΑΣΙΑ ΜΕΤΑΒΛΗΤΩΝ ΑΠΟ ΤΑ ΔΕΔΟΜΕΝΑ ΠΟΥ ΕΧΟΥΝ ΒΡΕΘΕΙ
     % ΜΕΧΡΙ ΤΩΡΑ ΓΙΑ ΝΑ ΠΡΟΕΤΟΙΜΑΣΤΟΥΝ ΟΙ ΑΝΤΙΣΤΟΙΧΕΣ ΚΙΝΗΣΕΙΣ
     % ============================================================

if ((splineBall>0) && (splineRobo>0))
     objstatus=1;
else
     objstatus=0;
end

if (objstatus==1)
    bally=stats(splineBall).Centroid(2);
    roboy=stats(splineRobo).Centroid(2);
    ballx=stats(splineBall).Centroid(1);
    robox=stats(splineRobo).Centroid(1)+0.0001;

    Loopok=3;
    sercount=0;
    sa=RoboLength;
```

```
521
522 -            if (Loopok==3)
523
524 -               Loopok=5;
525 -               sercount=3;
526
527               % ΕΥΡΕΣΗ ΠΡΑΓΜΑΤΙΚΗΣ ΚΛΙΣΗΣ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ / ΑΝΤΙΠΑΛΟΥ ΤΕΡΜΑΤΟΣ
528
529 -               rx=abs(robox-termx);
530 -               ry=abs(roboy-termy);
531 -               droboterm=sqrt(rx^2+ry^2);
532 -               angr=asind(rx/droboterm); % ΦΥΣΙΚΗ ΓΩΝΙΑ ΜΕΤΑΞΥ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ ΚΑΙ ΤΕΡΜΑΤΟΣ
533
534               % ΕΥΡΕΣΗ ΠΡΑΓΜΑΤΙΚΗΣ ΚΛΙΣΗΣ ΜΠΑΛΑΣ / ΑΝΤΙΠΑΛΟΥ ΤΕΡΜΑΤΟΣ
535 -               bx=abs(ballx-termx);
536 -               by=abs(bally-termy);
537 -               dballterm=sqrt(bx^2+by^2);
538 -               angb=asind(bx/dballterm);
539
540               % APOSTASI ROBO/TERM - BALL/TERM
541 -               dterm=abs(droboterm-dballterm);
542
543               % EYRESH PRAGMATIKIS KLISIS BALL/ROBO
544 -               x=abs(ballx-robox);
545 -               y=abs(bally-roboy);
546 -               dest_rb=sqrt(x^2+y^2); % APOSTASI ROBO - BALL
547 -               ang_br=asind(x/(dest_rb+0.0001));
548
549               % ΕΥΡΕΣΗ ΣΩΣΤΩΝ ΓΩΝΙΩΝ (MIN-MAX), ΣΥΜΦΩΝΑ ΜΕ ΤΗΝ ΚΛΙΣΗ ΤΟΥ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ
550
551 -               stx1=robox-termx;
552 -               stx2=stx1;
553 -               sty1=roboy-termy-semiterm;
554 -               sty2=roboy-termy+semiterm;
555 -               dest_st1=sqrt(stx1^2+sty1^2);
556 -               dest_st2=sqrt(stx2^2+sty2^2);
557 -               ang_temp(1)=asind(abs(stx1)/(dest_st1+0.0001)); % ΛΟΓΟ divide zero
558 -               ang_temp(2)=asind(abs(stx2)/(dest_st2+0.0001)); % ΛΟΓΟ divide zero
559
560
561               % ΕΥΡΕΣΗ ΣΩΣΤΩΝ ΓΩΝΙΩΝ (MIN-MAX), ΣΥΜΦΩΝΑ ΜΕ ΤΗΝ ΚΛΙΣΗ ΤΗΣ ΜΠΑΛΑΣ
562 -               ang_obj_temp(1)=(roboy-termy+semiterm)/(robox-termx)+0.001;
563 -               ang_obj_temp(2)=(roboy-termy-semiterm)/(robox-termx)+0.001;
564 -               ang_obj_temp(3)=(bally-termy+semiterm)/(ballx-termx)+0.001;
565 -               ang_obj_temp(4)=(bally-termy-semiterm)/(ballx-termx)+0.001;
566
567
568               % ΕΛΕΓΧΟΣ ΑΝ ΕΙΝΑΙ ΣΕ ΣΧΕΤΙΚΗ ΕΥΘΕΙΑ Η ΜΠΑΛΑ , Ο ΡΟΜΠΟΤΙΚΟΣ ΠΑΙΚΤΗΣ ΚΑΙ ΜΠΟΡΕΙ ΝΑ ΜΠΕΙ ΓΚΟΛ
569
570 -               ang_obj_ok=xor(((ang_obj_temp(1)>ang_obj_temp(3))),((ang_obj_temp(2)>ang_obj_temp(4))));
571
572 -               if (dest_rb<RoboLength)
573 -                   offang=8;
574 -               else
575 -                   if (dest_rb<100)
576 -                       offang=4; % ΑΝΑΛΟΓΟΣ ΤΗΝ ΑΠΟΣΤΑΣΗ ΘΑ ΕΙΝΑΙ ΚΑΙ Η ΚΙΝΗΣΗ ΣΕ ΜΟΙΡΕΣ
577 -                   else
578 -                       offang=300/dest_rb;
579 -                   end
580 -               end
581
582 -               switch(MyTerm)
583 -                      case 2 % ΤΕΡΜΑ ΑΡΙΣΤΕΡΟ
584
585 -                             if (roboy<term(2)) % ΥΠΟΛΟΓΙΣΜΟΣ ΣΧΕΤΙΚΗΣ ΓΩΝΙΑΣ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ
```

```
586 -                       angr=180+angr;
587 -                   else
588 -                       angr=360-angr;
589 -                   end
590
591 -                   if (bally<term(2))  % ΕΛΕΓΧΟΣ ΜΕ ΤΟ Υ ΤΗΣ ΜΠΑΛΑΣ
592 -                       angb=180+angb;
593 -                       mly=1;
594 -                   else
595 -                       angb=360-angb;
596 -                       mly=-1;
597 -                   end
598
599 -                   if (sty1<0) % ΥΠΟΛΟΓΙΣΜΟΣ ΣΩΣΤΩΝ ΓΩΝΙΩΝ ΠΟΥ ΜΠΟΡΕΙ ΝΑ ΒΑΛΕΙ
600                                 % ΓΚΟΛ Ο ΡΟΜΠΟΤΙΚΟΣ ΠΑΙΚΤΗΣ ΣΥΜΦΩΝΑ ΜΕ ΤΟΥΣ ΣΓΕΝΙΚΟΥΣ ΑΞΟΝΕΣ
601 -                       ang_temp(1)=180+ang_temp(1);
602 -                       if (sty2<0)
603 -                           ang_temp(2)=180+ang_temp(2);
604 -                       else
605 -                           ang_temp(2)=360-ang_temp(2);
606 -                       end
607 -                   else
608 -                       ang_temp(1)=360-ang_temp(1);
609 -                       ang_temp(2)=360-ang_temp(2);
610 -                   end
611
612 -                   if (ballx>robox) % ΕΥΡΕΣΗ ΠΡΑΓΜΑΤΙΚΗΣ ΚΛΙΣΗΣ ΣΕ ΣΧΕΣΗ ΜΕ ΤΟΥΣ ΓΕΝΙΚΟΥΣ ΑΞΟΝΕΣ
613                                         % ΓΙΑ ΤΗΝ ΔΗΜΙΟΥΡΓΙΑ ΗΗΧ,ΗΗΥ
614
615 -                       if (bally>roboy) % ΚΑΝΟΝΙΚΗ ΠΕΡΙΠΤΩΣΗ , ΡΟΜΠΟΤΙΚΟΣ ΠΑΙΚΤΗΣ ΠΙΣΩ ΑΠΟ ΤΗΝ ΜΠΑΛΑ
616 -                           mmlx=(sind(angb)*2*RoboLength);
617 -                           mmly=(cosd(angb)*2*RoboLength);
618 -                           mmlz=-1;
619 -                       else
620 -                           mmly=(cosd(angb)*2*RoboLength);
621 -                           mmlx=(sind(angb)*2*RoboLength);
622 -                           mmlz=+1;
623 -                       end;
624
625 -                       KK=abs(cosd(angb)-cosd(angr));
626
627 -                       if (KK<0.55) % ΕΙΔΙΚΗ ΠΕΡΙΠΤΩΣΗ ΕΥΘΕΙΑΣ ΜΠΑΛΑΣ ΚΑΙ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ
628 -                           mmly=RoboLength*mmlz;
629 -                           mmlx=RoboLength*1.5;
630 -                       end
631
632 -                       hhy=bally+mmly;
633 -                       hhx=ballx+abs(mmlx);
634 -                       mul=0;
635 -                   else  % (ballx<robox)
636 -                       mlx=1;
637 -                       mul=1;
638
639 -                       if ((abs(ang_br_gl-klisi)<offang) && (klisi>min(ang_temp)) && (klisi<max(ang_temp)))
640 -                           hhy=bally;   % ΠΕΡΙΠΤΩΣΗ ΟΤΑΝ ΗΗΥ,ΗΗΧ ΒΡΙΚΟΝΤΑΙ ΠΙΣΩ ΑΠΟ ΤΟ ROBOX
641 -                           hhx=ballx;
642 -                       else
643                             % ΝΕΟ Υ ΓΙΑ ΤΕΛΙΚΗ ΘΕΣΗ ΡΟΜΟΤΙΚΟΥ ΠΑΙΚΤΗ
644 -                           hhy=bally-((RoboLength*cosd(atand(bx/by)))*mly);
645                             % ΝΕΟ Χ ΓΙΑ ΤΕΛΙΚΗ ΘΕΣΗ ΡΟΜΟΤΙΚΟΥ ΠΑΙΚΤΗ
646 -                           hhx=ballx+((RoboLength*sind(atand(bx/by)))*mlx);
647 -                       end
648 -                   end
649
650 -               case 1 % ΤΕΡΜΑ ΔΕΞΙΟ (ΙΣΧΥΟΥΝ ΟΙ ΙΔΙΕΣ ΠΕΡΙΓΡΑΦΕΣ ΜΕΤΑΒΛΗΤΩΝ
```

```
651                 % ΜΕ ΔΙΑΦΟΡΕΤΙΚΕΣ ΤΙΜΕΣ, ΟΠΩΣ ΤΟ ΠΑΡΑΠΑΝΩ ΑΡΙΣΤΕΡΟ)
652
653 -               if (roboy<term(4))
654 -                   angr=180-angr;
655 -               else
656 -                   angr=angr;
657 -               end
658
659 -               if (roboy>bally)
660 -                   ang_br_gl=ang_br;
661 -               else
662 -                   ang_br_gl=180-ang_br;
663 -               end
664
665 -               if (bally>term(4))
666 -                   angb=angb;
667 -                   mly=1;
668 -               else
669 -                   angb=180-angb;
670 -                   mly=-1;
671 -               end
672
673 -               if (sty1<0)
674 -                   ang_temp(1)=180-ang_temp(1);
675 -                   if (sty2<0)
676 -                       ang_temp(2)=180-ang_temp(2);
677 -                   else
678 -                       ang_temp(2)=ang_temp(2);
679 -                   end
680 -               else
681 -                   ang_temp(1)=ang_temp(1);
682 -                   ang_temp(2)=ang_temp(2);
683 -               end
684
685 -               tankb=tand(klisi)/(tand(angb)+0.0001);
686
687 -               if (ballx<robox)
688
689 -                   if (bally>roboy)
690 -                       mmlx=(sind(angb)*2*RoboLength);
691 -                       mmly=(cosd(angb)*2*RoboLength);
692 -                       mmlz=-1;
693 -                   else
694 -                       mmly=(cosd(angb)*2*RoboLength);
695 -                       mmlx=(sind(angb)*2*RoboLength);
696 -                       mmlz=+1;
697 -                   end;
698
699 -                   KK=abs(cosd(angb)-cosd(angr));
700
701 -                   if (KK<0.55)
702 -                       mmly=RoboLength*mmlz;
703 -                       mmlx=RoboLength*1.5;
704 -                   end
705
706 -                   hhy=bally+mmly;
707 -                   hhx=ballx-abs(mmlx);
708 -                   mul=0; % ΔΕΙΧΝΕΙ ΟΤΙ Η ΚΙΝΗΣΗ ΕΙΝΑΙ ΑΝΑΠΟΔΑ ,
709                            % ΡΟΜΠΟΤΙΚΟΣ ΠΑΙΚΤΗΣ ΠΙΣΩ ΑΠΟ ΜΠΑΛΑ
710 -               else % (ballx>=robox)
711 -                   mlx=-1;
712 -                   if ((abs(ang_br_gl-klisi)<offang) && (klisi>min(ang_temp)) && (klisi<max(ang_temp)))
713 -                       hhy=bally;
714 -                       hhx=ballx;
715 -                   else
```

```
                                hhy=bally+((RoboLength*cosd(atand(bx/by)))*mly);
                                hhx=ballx+((RoboLength*sind(atand(bx/by)))*mlx);
                            end
                            mul=1;
                        end
                end % end switch

                % ΔΙΟΡΘΩΣΗ ΓΙΑ ΤΙΣ ΕΚΤΟΣ ΟΡΙΩΝ ΤΙΜΕΣ Η ΤΙΜΕΣ ΣΤΑ ΑΚΡΑ ΤΟΥ ΓΗΠΕΔΟΥ
                if (hhx<RoboLength/2)
                    hhx=RoboLength/2;
                else
                    if (hhx>limx-RoboLength/2)
                      hhx=limx-RoboLength/2;
                    end
                end

                if (hhy<RoboLength/2)
                    hhy=RoboLength/2;
                else
                    if (hhy>limy-RoboLength/2)
                        hhy=limy-RoboLength/2;
                    end
                end

                x=abs(hhx-robox);
                y=abs(hhy-roboy);
                dest_bball=sqrt(x^2+y^2); % ΑΠΟΣΤΑΣΗ ΚΙΝΗΣΗΣ ΑΠΟ ΡΟΜΠΟΤΙΚΟ ΠΑΙΚΤΗ ΜΕΧΡΙ ΤΟ ΗΗΧ,ΗΗΥ.
                angn=asind(x/(dest_bball+0.0001));
                maxserdest=dest_bball;

                angdif=angb-klisi;

                % ΔΙΟΡΘΩΣΗ ΚΛΙΣΗΣ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ

                if (robox>hhx)
                    if (roboy>hhy)
                        angs=360-angn;
                        angsdif=angs-360;
                    else
                        angs=180+angn;
                        angsdif=180-angs;
                    end;
                else
                    if (roboy>hhy)
                        angs=angn;
                        angsdif=angs;
                    else
                        angs=180-angn;
                        angsdif=180-angs;
                    end
                end
             end        % end Loopok==3)

            drawnow;
            imagesc(pic); % ΕΜΦΑΝΙΣΗ ΕΙΚΟΝΑΣ

            if (showstatus==1) % ΕΜΦΑΝΙΣΗ ΒΟΗΘΗΤΙΚΩΝ ΓΡΑΜΜΩΝ ΣΤΗΝ ΕΙΚΟΝΑ
                hold on
                xx=[hhx robox];
                yy=[hhy roboy];
                plot(x,y,'',xx,yy,'-oy');
                xx=[hhx termx];
                yy=[hhy termy];
                plot(x,y,'',xx,yy,'-oy');
                plot(termx,termy-semiterm,'',termx,termy-semiterm,'-oy');
```

```
781 -               plot(termx,termy+semiterm,'',termx,termy+semiterm,'-oy');
782 
783 -               if (debugmode==1)
784 -                   xx=[robox termx];
785 -                   yy=[roboy termy-semiterm];
786 -                   plot(robox,roboy,'',xx,yy,'--ob');
787 
788 -                   xx=[robox termx];
789 -                   yy=[roboy termy+semiterm];
790 -                   plot(robox,roboy,'',xx,yy,'--ob');
791 
792                     % plot ball term
793 -                   xx=[ballx termx];
794 -                   yy=[bally termy-semiterm];
795 -                   plot(ballx,bally,'',xx,yy,plotform);
796 
797 -                   xx=[ballx termx];
798 -                   yy=[bally termy+semiterm];
799 -                   plot(ballx,bally,'',xx,yy,plotform);
800 -               end
801 -               hold off
802 -           end;
803 
804             % ============================================================
805             %      ΕΝΕΡΓΟΠΟΙΗΣΗ ΚΙΝΗΣΗΣ ΑΝΑΟΓΑ ΜΕ ΤΗΝ ΠΕΡΙΠΤΩΣΗ
806             %      ΚΑΙ ΤΗΝ ΣΤΡΑΤΗΓΙΚΗ
807             % ============================================================
808 
809 -           moveenable=1;
810 -           angd=abs(klisi-angs);
811 -           tminang=sqrt((10^2)+(dest_rb^2)); % ΕΥΡΕΣΗ minang ΑΝΑΛΟΓΑ ΜΕ ΑΠΟΣΤΑΣΗ ΚΑΙ ΜΕΓΕΘΟΣ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ
812 -           minang=asind(10/tminang);
813 -           RoboLength=sa;
814 
815 -           if  (mul==1)
816 -               if (ang_obj_ok)
817 -                   if ( (klisi<ang_temp(1)) && (klisi>ang_temp(2)) ) % ΠΕΡΙΠΟΥ ΣΕ ΣΧΕΤΙΚΗ ΕΥΘΕΙΑ ΓΙΑ ΓΚΟΛ Η ΚΛΙΣΗ
818 -                       if (dballterm<goaldistance)    % ΠΡΟΟΘΗΣΗ ΜΠΑΛΑΣ ΠΡΟΣ ΤΕΡΜΑ (ΓΚΟΛ)
819 -                           togoal=1;
820 -                           spec_case=1;
821 -                           min_strmov=(dest_rb/2)+RoboLength;
822 -                           sdc=4;
823 -                           plotform='--or';
824 -                       else % ΑΠΛΗ ΠΡΟΟΘΗΣΗ
825 
826 -                           if  (dest_rb>RoboLength*3) % ΓΡΗΓΟΡΗ ΚΙΝΗΣΗ
827 -                               sdc=4;
828 -                               min_strmov=RoboLength/2;
829 -                               spec_case=4;
830 -                               togoal=1;
831 -                           else
832 -                               sdc=4;              % ΑΡΓΗ ΚΙΝΗΣΗ
833 -                               min_strmov=RoboLength;
834 -                               spec_case=4;
835 -                               togoal=1;
836 -                           end
837 -                       end
838 -                   else % ΟΤΑΝ ΔΕΝ ΕΙΝΑΙ ΣΕ ΣΧΕΤΙΚΗ ΕΥΘΕΙΑ Η ΚΛΙΣΗ ΤΟΥ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ
839                         % ΓΙΝΕΤΑΙ ΠΕΡΙΣΤΡΟΦΙΚΗ ΚΙΝΗΣΗ ΠΡΟΣ ΚΛΙΣΗ ΜΠΑΛΑΣ
840 -                           sdc=2;
841 -                           spec_case=3;
842 -                           angs=angb;
843 -                           angd=abs(klisi-angs);
844 -                   end
845 -               else
```

```
                    if (angd<5)
                            sdc=4; % ΑΡΓΗ ΚΙΝΗΣΗ
                            min_strmov=dest_bball*0.9;
                            spec_case=4;
                    else % ΠΕΡΙΣΤΡΟΦΙΚΗ ΚΙΝΗΣΗ ΠΡΟΣ ΗΗΧ,ΗΗΥ
                        sdc=2;
                        spec_case=3;
                    end
            end
        else % ΜΗ  ΚΑΝΟΝΙΚΗ ΠΕΡΙΠΤΩΣΗ , ΟΛΕΣ ΟΙ ΚΙΝΗΣΕΙΣ ΕΥΘΕΙΑΣ ΚΑΙ ΠΕΡΙΣΤΡΟΦΗΣ ΕΙΝΑΙ ΠΡΟΣ ΓΩΝΙΑ ANGS
            if ( abs(angsdif-klisidif)<10)
                if ( ballx<RoboLength ) % ΑΝ ΕΙΝΑ ΜΑΚΡΙΑ ΑΠΟ ΤΟ ΤΕΡΜΑ ΝΑ ΓΙΝΕΙ ΜΕΓΑΛΗ ΕΥΘΕΙΑ ΚΙΝΗΣΗ
                    if (dest_rb>3*RoboLength)
                        sdc=4;
                        min_strmov=abs(robox-hhx)*0.7;
                        spec_case=4;
                    else % ΑΠΟΣΤΑΣΗ ΚΟΝΤΑ
                        sdc=4;
                        min_strmov=abs(robox-ballx);
                        spec_case=4;
                    end
                else
                    sdc=4;
                    min_strmov=abs(robox-hhx);
                    spec_case=4;
                end
            else
                sdc=2;
                spec_case=3;
            end
        end

        angm=angd;

        % ΕΠΙΠΛΕΟΝ ΡΥΘΜΙΣΕΙΣ ΠΕΡΙΣΤΡΟΦΙΚΗΣ ΚΑΙ ΕΥΘΕΙΑΣ ΚΙΝΗΣΗΣ
        if (klisi>angs)
            if (angd>180)
                angf=1; % ΚΙΝΗΣΗ ΠΡΟΣ ΔΕΞΙΑ
                angm=360-angd;
            else
                angf=2; % ΚΙΝΗΣΗ ΠΡΟΣ ΑΡΙΣΤΕΡΑ
            end
        else
            if (angd>180)
                angf=2; % ΚΙΝΗΣΗ ΠΡΟΣ ΑΡΙΣΤΕΡΑ
                angm=360-angd;
            else
                angf=1;
            end
        end

        if (angf==1) % ΓΙΑ ΝΑ ΓΙΝΕΤΑΙ Η ΜΕΓΑΛΥΤΕΡΗ ΚΙΝΗΣΗ ΤΗΣ ΓΩΝΙΑΣ >60 ΜΟΙΡΕΣ ΔΙΑΦΟΡΑ , ΜΕ ΜΕΓΑΛΥΤΕΡΗ ΤΑΧΥΤΗΤΑ
            if (angm>60)
                serang=angm;
            else
                serang=8;
            end
        else
            if (angm>60)
                serang=360-angm;
            else
                serang=360-8;
            end
        end

```

```
911
912             % angs = ΕΠΙΘΥΜΗΤΗ ΚΛΙΣΗ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ
913             % angd = ΔΙΑΦΟΡΑ ΓΩΝΙΩΝ ΑΠΟ ΤΩΡΙΝΗ ΘΕΣΗ ΣΕ ΕΠΙΘΥΜΗΤΗ
914             % angm = ΕΥΡΕΣΗ ΜΙΚΡΟΤΕΡΗΣ ΔΙΑΦΟΡΑΣ ΤΩΡΙΝΗΣ ΚΑΙ ΕΠΙΘΥΜΗΤΗΣ
915             % serang = ΤΙΜΗ ΓΩΝΙΑΣ ΠΟΥ ΘΑ ΜΕΤΑΤΡΑΠΕΙ ΣΕ BYTES ΓΙΑ ΑΠΟΣΤΟΛΗ ΣΤΗ ΣΕΙΡΙΑΚΗ ΘΥΡΑ.
916
917             % ============================================================
918             % ΓΙΑ ΤΗΝ ΑΠΟΣΤΟΛΗ ΔΕΔΟΜΕΝΩΝ ΜΕ ΤΗΝ ΣΕΙΡΙΑΚΗ ΘΥΡΑ ΣΤΟΝ AVR
919             %
920             % ΓΙΑ ΚΙΝΗΣΗ ΠΡΟΣ ΑΡΙΣΤΕΡΑ Π.Χ 50 ΜΟΙΡΕΣ ΤΟΤΕ ΣΤΕΛΝΕΤΑΙ 360-50=310,
921             % ΔΗΛΑΔΗ ΣΗΜΑΙΝΕΙ 50ο ΑΡΙΣΤΕΡΑ  ΑΠΟ ΕΚΕΙ ΠΟΥ ΒΡΙΣΚΕΤΑΙ Ο
922             % ΡΟΜΠΟΤΙΚΟΣ ΠΑΙΚΤΗΣ
923             % ΓΙΑ ΚΙΝΗΣΗ ΠΡΟΣ ΔΕΞΙΑ  Π.Χ 50 ΜΟΙΡΕΣ ΤΟΤΕ ΣΤΕΛΝΕΤΑΙ 50,
924             % ============================================================
925
926 -           if ((moveenable==1) && (togoal<=1))
927 -               moveenable=0;
928
929 -               if (abs(cosd(angs))<cosd(20)) % ΕΠΙΛΟΓΗ ΚΛΙΜΑΚΑΣ mm/pixel
930 -                   angdiv=ly;
931 -               else
932 -                   angdiv=lx;
933 -               end
934
935 -               serdest=min_strmov; % ΥΠΟΛΟΓΙΣΜΟΣ ΕΝΤΟΛΗΣ ΕΥΘΕΙΑΣ ΚΙΝΗΣΗΣ
936
937             % ΓΕΜΙΣΜΑ ΠΙΝΑΚΑ ΑΠΟΣΤΟΛΗΣ ΕΥΘΕΙΑΣ ΚΙΝΗΣΗΣ ΚΑΙ ΕΤΟΙΜΑΣΙΑ ΤΙΜΗΣ ΓΙΑ ΔΗΜΙΟΥΡΓΙΑ BYTES
938 -           serdata(sercount)=round((((serdest/angdiv)*10)/2));
939
940 -           serdata(sercount-1)=round(serang); % ΥΠΟΛΟΓΙΣΜΟΣ ΕΝΤΟΛΗΣ ΠΕΡΙΣΤΡΟΦΙΚΗΣ ΚΙΝΗΣΗΣ
941
942
943 -           if (serang>180)
944 -               tempserdata=458; % ΕΝΕΡΓΟΠΟΙΗΣΗ ΑΡΙΣΤΕΡΗΣ ΠΕΡΙΣΤΡΟΦΗΣ
945 -           else
946 -               tempserdata=459;  % ΔΕΞΙΑ ΠΕΡΙΣΤΡΟΦΗ
947 -           end
948
949
950             % ====================================
951             %   ΚΩΔΙΚΟΠΟΙΗΣΗ BYTES
952             % ====================================
953
954             % ΓΙΑ DBs (BYTES)
955             %  1ο ΑΡΙΘΜΟΣ ΠΑΝΤΑ = 0
956             %  2ο ΣΕΙΡΑ ΑΝΑΛΟΓΑ ΤΟ ID ΑΠΟ ΤΟΝ AVR : ΓΙΑ ID=0 ->:    0-> 1ο ΑΡΙΘΜΟΣ  ,
957             %       1-> 2ο ΑΡΙΘΜΟΣ, 2-> 3ο ΑΡΙΘΜΟΣ
958             %  3ο Ο ΑΡΙΘΜΟΣ
959
960             % ΓΙΑ CBs (BYTES)
961             %  1ο ΑΡΙΘΜΟΣ ΠΑΝΤΑ = 1
962             %  2ο ΤΥΠΟΣ ΑΠΟ ΤΟΝ AVR ΑΝΑΛΟΓΑ ΤΟ ID:  ΓΙΑ ID=0 -> :   0-> DEST  , 1-> SPD , 2-> ANG
963             %  3ο Η ΔΙΑΦΟΡΑ ΜΕΤΑΞΥ ΤΟΥ 1 ΚΑΙ ΤΟΥ 2ου ΤΟΥ DB
964
965 -           if (serenable==0)
966 -                 !mode com1:9600,n,8,1
967 -                 serenable=1;
968 -           end
969
970 -           if (sdc==2) % ΠΕΡΙΣΤΡΟΦΙΚΗ ΚΙΝΗΣΗ
971               % ΔΗΜΙΟΥΡΓΙΑ BYTES ΠΕΡΙΣΤΡΟΦΙΚΗΣ ΚΙΝΗΣΗΣ
972 -               serbyte(1,1)=0+(0*10)+(fix(mod(serdata(sercount-1)/100,10)));     % 1ο byte ΚΑΙ 1ος ΑΡΙΘΜΟΣ
973 -               serbyte(1,2)=0+(1*10)+(fix(mod(serdata(sercount-1)/10,10)));      % 2ο byte ΚΑΙ 2ος ΑΡΙΘΜΟΣ
974 -               serbyte(1,3)=0+(2*10)+(fix(mod(serdata(sercount-1),10)));         % 3ο byte ΚΑΙ 3ος ΑΡΙΘΜΟΣ
975
```

```
                serbyte(1,4)=(1*100)+(2*10)+fix(abs(mod(serbyte(1,2),10)-mod(serbyte(1,3),10)));    % MAKE CHECKBYTE

                % ΔΗΜΙΟΥΡΓΙΑ ΒΥΤΕΣ ΓΡΗΓΟΡΗΣ ΔΙΟΡΘΩΣΗΣ ΠΕΡΙΣΤΡΟΦΙΚΗΣ ΚΙΝΗΣΗΣ
                serbyte(3,1)=0+(0*10)+(fix(mod(tempserdata/100,10)));    % 1Ο byte kai 1ο num
                serbyte(3,2)=0+(1*10)+(fix(mod(tempserdata/10,10)));     % 2Ο byte kai 2ο num
                serbyte(3,3)=0+(2*10)+(fix(mod(tempserdata,10)));        % 3Ο byte kai 3ο num

                serbyte(3,4)=(1*100)+(2*10)+fix(abs(mod(serbyte(3,2),10)-mod(serbyte(3,3),10))); % ΔΗΜΙΟΥΡΓΙΑ ΒΥΤΕ ΕΛΕΓΧΟΥ

                sport = fopen('com1:','w'); % ΑΝΟΙΓΜΑ ΘΥΡΑΣ ΣΕΙΡΙΑΚΗΣ

                % ΑΠΟΣΤΟΛΗ ΒΥΤΕΣ ΠΕΡΙΣΤΡΟΦΙΚΗΣ ΚΙΝΗΣΗΣ
                fwrite(sport,serbyte(1,4));  %  cb
                fwrite(sport,serbyte(1,1)); %  1ο BYTE
                fwrite(sport,serbyte(1,2)); %  2ο BYTE
                fwrite(sport,serbyte(1,3));  %  3ο BYTE
                fclose(sport); % ΚΛΕΙΣΙΜΟ ΘΥΡΑΣ

                errcnt=0;
                Loopok=5; % ΚΑΝΕΙ ΡΕΣΕΤ ΤΑ VALUES ΚΑΙ ΠΕΡΝΕΙ ΝΕΑ ΔΕΔΟΜΕΝΑ
                tloop=0;

                ser_commands=ser_commands+1;
                sdc=7;
            end

            if (sdc==1)

                sport = fopen('com1:','w');

                 % ROT_cont
                fwrite(sport,serbyte(1,4));  %  cb
                fwrite(sport,serbyte(1,1)); %  1ο BYTE
                fwrite(sport,serbyte(1,2)); %  2ο BYTE
                fwrite(sport,serbyte(1,3));  %  3ο BYTE

                fclose(sport);

                errcnt=0;

                Loopok=5;
                tloop=0;

                ser_commands=ser_commands+1;
                sdc=7;
            end


            if (sdc==4) % ΕΥΘΕΙΑ ΚΙΝΗΣΗ
                % ΔΗΜΙΟΥΡΓΙΑ ΒΥΤΕΣ ΕΥΘΕΙΑΣ ΚΙΝΗΣΗΣ
                serbyte(2,1)=0+(0*10)+(fix(mod(serdata(sercount)/100,10)));    % 1ο byte ΚΑΙ 1ος ΑΡΙΘΜΟΣ
            serbyte(2,2)=0+(1*10)+(fix(mod(serdata(sercount)/10,10)));     % 2ο byte ΚΑΙ 2ος ΑΡΙΘΜΟΣ
            serbyte(2,3)=0+(2*10)+(fix(mod(serdata(sercount),10)));        % 3ο byte ΚΑΙ 3ος ΑΡΙΘΜΟΣ

            serbyte(2,4)=(1*100)+(0*10)+fix(abs(mod(serbyte(2,2),10)-mod(serbyte(2,3),10))); % ΔΗΜΙΟΥΡΓΙΑ ΒΥΤΕ ΕΛΕΓΧΟΥ
                sport = fopen('com1:','w');

                % ΑΠΟΣΤΟΛΗ ΒΥΤΕΣ ΕΘΥΕΙΑΣ ΚΙΝΗΣΗΣ
                fwrite(sport,serbyte(2,4));  %  cb
                fwrite(sport,serbyte(2,1));  %  1ο BYTE
                fwrite(sport,serbyte(2,2));  %  2ο BYTE
                fwrite(sport,serbyte(2,3)); %  3ο BYTE
                fclose(sport);

                strobox=robox;
```

```
                stroboy=roboy;
                errcnt=0;

                Loopok=5;
                tloop=0;
                ser_commands=ser_commands+1;

                if (togoal==1)
                    togoal=2;
                end
                sdc=7;
            end
        end

    end % objstatus


    % ΠΑΝΗΓΥΡΙΣΜΟΣ ΓΚΟΛ
    if (togoal==3) % ΕΑΝ ΕΧΕΙ ΕΝΕΡΓΟΠΟΙΗΘΕΙ ΠΟΡΕΙΑ ΓΙΑ GOAL ΚΑΙ ΔΕΝ ΦΑΙΝΕΤΑΙ Η ΜΠΑΛΑ %
                   % ->ΣΗΜΑΙΝΕΙ ΟΤΙ ΕΧΕΙ ΜΠΕΙ Η ΜΠΑΛΑ ΜΕΣΑ ΚΑΙ ΓΙΝΕΤΑΙ ΕΝΕΡΓΟΠΟΙΗΣΗ ΠΑΝΗΓΥΡΙΣΜΟΥ

        sport = fopen('com1:','w');

        % ID=1 , ANGLE = 450 -> KODIKOS PANIGIRISMOS  -> goal
        fwrite(sport,15);%  2o BYTE
        fwrite(sport,20); %  3o BYTE
        fwrite(sport,4); %  1o BYTE
        fwrite(sport,125);%  CB

        fclose(sport);

        togoal=0;
        Loopok=0;

       ser_commands=ser_commands+1; % ΑΘΡΟΙΣΜΑ ΑΠΕΣΤΑΛΜΕΝΩΝ ΕΝΤΟΛΩΝ
       sdc=7;
    end

    if (togoal==2)    % ΕΛΕΓΧΟΣ ΑΝ ΠΡΕΠΕΙ ΝΑ ΓΙΝΕΙ ΠΑΝΗΓΥΡΙΣΜΟΣ
        moveenable=0;
        if (splineBall==0) % ΕΑΝ ΔΕΝ ΦΑΙΝΕΑΙ Η ΜΠΑΛΑ ΣΥΝΕΧΟΜΕΝΑ ΟΡΙΣΜΕΝΕΣ ΦΟΡΕΣ ΝΑ ΕΚΤΕΛΕΣΤΕΙ Ο ΠΑΝΗΓΥΡΙΣΜΟΣ
            noball=noball+1;
            plotform='--og';
        else
            yesball=yesball+1;
            noball=0;
        end;

        if (noball>20) % ΚΑΘΥΣΤΕΡΗΣΗ ΓΙΑ ΝΑ ΜΕΤΡΗΘΕΙ Η ΑΠΟΥΣΙΑ ΤΗΣ ΜΠΑΛΑΣ
            togoal=3; % ΕΝΕΡΓΟΠΟΗΣΗ ΕΚΤΕΛΕΣΗΣ ΠΑΝΗΓΥΡΙΣΜΟΥ
            goals=goals+1;
            yesball=0;
            noball=0;
            plotform='--om';
        else
            if (yesball>20) % RESET ΕΑΝ Η ΜΠΑΛΑ ΔΕΝ ΜΠΕΙ ΓΚΟΛ ΚΑΙ ΕΠΙΣΤΡΟΦΗ ΣΤΗΝ ΚΑΝΟΝΙΚΗ ΛΕΙΤΟΥΡΓΙΑ
                moveenable=1;
                togoal=0;
                yesball=0;
                plotform='--om';
            end
        end
    end

     % ΜΕΤΑΒΛΗΤΕΣ ΜΕΤΡΗΣΗΣ ΧΡΟΝΟΥ ΕΚΤΕΛΕΣΗΣ LOOP
```

```
1106 -          tim=tim+toc;
1107 -          timcnt=timcnt+1;
1108 -          motime=tim/timcnt;
1109 -          lps=1/motime;
1110
1111 -          tt=[timcnt motime lps splineBall splineRobo]
1112 -  end %% END main while
```

7.2 Υποπρόγραμμα Αρχικοποίησης τιμών (maininit.m)

```
% PROJECT:      maininit.m (ROBOSOCCER)
% PART:         VISION SECTION
% LAST UPDATE:  11/05/2014

function comp_data=MainInit(a) %  ΕΝΕΡΓΟΠΟΙΗΣΗ ΥΠΟΠΡΟΓΡΑΜΜΑΤΟΣ

% ΓΙΑ ΝΑ ΛΕΙΤΟΥΡΓΗΣΕΙ ΤΟ ΥΠΟΠΡΟΓΡΑΜΜΑ ΚΑΙ ΤΟ ΠΡΟΓΡΑΜΜΑ ΣΤΗΝ ΣΥΝΕΧΕΙΑ ΠΡΕΠΕΙ
% ΝΑ :
%     - ΥΠΑΡΧΕΙ ΜΟΝΟ ΕΝΑΣ ΡΟΜΠΟΤΙΚΟΣ ΠΑΙΚΤΗΣ ΣΤΟ ΓΗΠΕΔΟ
%     - ΥΠΑΡΧΕΙ ΜΙΑ ΜΠΑΛΑ ΣΤΟ ΓΗΠΕΔΟ
%     - ΚΑΝΕΝΑ ΑΠΟ ΤΑ ΔΥΟ ΔΕΝ ΠΡΕΠΕΙ ΝΑ ΕΙΝΑΙ ΑΡΚΕΤΑ ΣΤΑ ΑΚΡΑ ΚΑΙ  ΑΚΡΙΒΩΣ
%       ΣΤΟ ΚΕΝΤΡΟ ΤΟΥ ΓΗΠΕΔΟΥ.
%     - ΟΤΑΝ ΑΝΑΓΝΩΡΙΣΤΟΥΝ ΣΩΣΤΑ , ΕΠΙΣΤΡΕΦΕΤΑΙ Ο ΕΛΕΓΧΟΣ ΣΤΟ ΒΑΣΙΚΟ
%     ΠΡΟΓΡΑΜΜΑ

% ΡΥΘΜΙΣΗ ΣΕΙΡΙΑΚΗΣ ΘΥΡΑΣ
 !mode com1:9600,n,8,1

 % ΒΟΗΘΗΤΙΚΕΣ ΜΕΤΑΒΛΗΤΕΣ

thresst=0.01;
thresen=-1;
ok=1;
error=0;
RoboId=1;
errortext=' ΔΕΝ ΒΡΕΘΗΚΕ Η ΜΠΑΛΑ η Ο ΡΟΜΠΟΤΙΚΟΣ ΠΑΙΚΤΗΣ - or ROBOPLAYER MISSING ';

ColorArray=[0 0 0; 0 0 0];
TempballColorArray=[170 75 75];
treshdata(1,1)=0;
comp_data(8,1)=0;
comp_data=ColorArray;
ColorAnnot=33;
Threshold=0.0;

indx=1;
x=0;
pixlim=640*480;
offsR=0;
offsG=pixlim;
offsB=pixlim;


fbground_width=8;
treshdata=[0,0,0,0];
objsId=0;
HVV=1;
Ips=0;
ground_align=0;
threshold_tune=0;
lengtharray=0;
dok=0;
tempballmin=10; % ΕΛΑΧΙΣΤΗ ΕΜΠΕΙΡΙΚΗ ΤΙΜΗ ΜΕΓΕΘΟΥΣ ΜΠΑΛΑΣ ΣΥΜΦΩΝΑ ΜΕ ΤΟ ΥΨΟΣ ΤΗς ΚΑΜΕΡΑΣ
tempballmax=40; % ΜΕΓΙΣΤΗ
temproboarray=[0 , 0 , 0 , 0, 0 , 0]; % index , threshold , ΑΡΙΘΜΟΣ ΑΝΤΙΚΕΙΜΕΝΟΥ ΑΠΟ ΤΟΝ ΠΙΝΑΚΑ B

for (i=1:2)
    orpic=vcapg2(1);
end

% ΡΥΘΜΙΣΗ ΚΕΝΤΡΟΥ ΓΗΠΕΔΟΥ ΜΕ ΚΕΝΤΡΟ ΤΗΣ ΚΑΜΕΡΑΣ. ΕΝΕΡΓΟΠΟΙΕΙΤΕ ΚΑΙ
% ΣΤΑΜΑΤΑΕΙ ΧΕΙΡΟΚΙΝΗΤΑ ΤΗΝ ΦΟΡΑ ΤΟΠΟΘΕΤΗΣΗΣ ΤΟΥ ΓΗΠΕΔΟΥ.

 if (ground_align==1)
      while (1)

```

```
            drawnow;

            orpic=vcapg2(1);
            imagesc(orpic);
            hold on

            xx=[146 494];
            yy=[80 390];
            plot(1,1,'',xx,yy,'-oy');

            xx=[146 494];
            yy=[390 80];
            plot(1,1,'',xx,yy,'-oy');

            xx=[146 146];
            yy=[1 480];
            plot(1,1,'',xx,yy,'-oy');

            xx=[494 494];
            yy=[1 480];
            plot(1,1,'',xx,yy,'-oy');

            xx=[1 640];
            yy=[80 80];
            plot(1,1,'',xx,yy,'-oy');

            xx=[1 640];
            yy=[390 390];
            plot(1,1,'',xx,yy,'-oy');


            xx=[295 345];
            yy=[240 240];
            plot(1,1,'',xx,yy,'-oy');

            xx=[320 320];
            yy=[215 265];
            plot(1,1,'',xx,yy,'-oy');
            hold off
      end
 end
         hold off;

% ================================================================================

% ΕΥΡΕΣΗ ΓΗΠΕΔΟΥ ΚΑΙ ΧΑΡΑΚΤΗΡΙΣΤΙΚΩΝ ΑΝΤΙΚΕΙΜΕΝΩΝ ΜΠΑΛΑΣ ΚΑΙ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ
% ================================================================================

m=0;

while (objsld<0.95)%
    m=m+1;
    orpic=vcapg2(1);
    help_color=impixel(orpic,320,240); % ΔΕΙΓΜΑ ΦΩΤΕΙΝΟΤΗΤΑΣ ΚΑΙ ΧΡΩΜΑΤΩΝ ΣΤΟ ΚΕΝΤΡΟ ΤΟΥ ΓΗΠΕΔΟΥ
    colpic=orpic;
    pground(1:pixlim*3)=0; % 3-> ΕΠΕΙΔΗ (3=R+G+B)
    helppic=orpic;   % ΓΙΝΕΤΑΙ ΑΝΑΠΟΔΑ Η ΕΙΚΟΝΑ  ΠΑΝΩ/ΚΑΤΩ
    pground(1,find(orpic(1:end)>2))=1;
    i=1;

    while (i<pixlim) % ΡΥΘΜΙΣΗ ΠΙΝΑΚΑ ΜΕ ΤΙΜΕΣ (R,G,B) ΓΙΑ ΈΝΑ ΧΡΩΜΑ
        if  ( (pground(i)==0) && (pground(offsG+i)==0) && (pground(offsB+i)==0) )
            helppic(i)=0;
            helppic(offsG+i)=0;
            helppic(offsB+i)=0;
        else
```

```
            helppic(i)=254;
            helppic(offsG+i)=254;
            helppic(offsB+i)=254;
        end
        i=i+1;
    end

    HpicBW=im2bw(helppic,0.10); % ΤΟ BACKGROUND ΓΙΝΕΤΑΙ ΜΠΛΕ
    Hpic=bwareaopen(HpicBW,1000); % ΑΦΑΙΡΕΣΗ ΘΟΡΥΒΟΥ ΑΠΟ ΔΙΑΦΟΡΑ ΑΝΤΙΚΕΙΜΕΝΑ
    Hpic2=not(Hpic); % ΑΡΝΗΤΙΚΗ ΕΙΚΟΝΑ
    Hpic3=bwareaopen(Hpic2,20000); % BACKGROUND ΚΟΚΚΙΝΟ ΜΕ ΣΒΗΣΙΜΟ ΑΣΧΕΤΩΝ ΑΝΤΙΚΕΙΜΕΝΩΝ
    [B,L] = bwboundaries(Hpic3,'noholes');
    stats = regionprops(L,'Image','Area','Centroid','Solidity','BoundingBox');

    lenobj=length(B);

    %ΕΥΡΕΣΗ ΜΙΚΡΟΤΕΡΟΥ ΤΕΤΡΑΓΩΝΟΥ ΚΑΙ ΤΕΤΡΑΓΩΝΟΥ 3,5Χ3,5
    for i=1:lenobj
        x(i)=stats(i).Area;
    end

    if (x>0) % x=1 -> ΓΗΠΕΔΟ, x=2 -> ΡΟΜΠΟΤΙΚΟΣ ΠΑΙΚΤΗΣ
        [Cvl,Ips] = max(x(1,:)); % ΕΥΡΕΣΗ ΤΟΥ ΜΕΓΑΛΥΤΕΡΟΥ ΤΜΗΜΑΤΟΣ ΔΗΛΑΔΗ ΤΟ ΓΗΠΕΔΟ.
        if (Ips>0)
            objsld=stats(Ips).Solidity; % ΕΛΕΓΧΟΣ ΣΧΗΜΑΤΟΣ ΓΗΠΕΔΟΥ
            crtable=stats(Ips).BoundingBox;
            crtable(3)=crtable(3)-fbground_width;
            crtable(4)=crtable(4)-fbground_width;
            HVV=stats(Ips).BoundingBox(4)/stats(Ips).BoundingBox(3);
        end

        figure(1);
        imagesc(Hpic3);
        rectangle('Position',[crtable(1),crtable(2),crtable(3),crtable(4)]);
    end
    hold on; drawnow;
end


% ============================================================
% THRESHOLD section. -> ΕΥΡΕΣΗ THRESHOLD
% ============================================================

close (1);

nindx=1;

while ( (thresst<1) || (threshold_tune==1) )
    orpic=vcapg2(1);
    pic=imcrop(orpic,crtable);
    [B,L] = bwboundaries(im2bw(pic,thresst),'noholes');
    stats = regionprops(L,'Image','Centroid','BoundingBox','Eccentricity');

    % ΑΝΑΓΝΩΡΙΣΗ ΜΠΑΛΑΣ ΜΕ thresst= 0.1
    if ( (thresst>=0.09)  &&  (thresst<=0.11) )  % Η ΤΙΜΗ THRESHOLD=0.1 ΘΑ ΧΡΗΣΙΜΟΠΟΙΗΘΕΙ ΙΔΙΑ ΣΤΟ ΒΑΣΙΚΟ ΠΡΟΓΡΑΜΜΑ
        for sa=1:length(B) % ΕΥΡΕΣΗ ΜΠΑΛΑΣ ΑΠΟ ΟΛΑ ΤΑ ΑΝΤΙΚΕΙΜΕΝΑ
                           % ΕΛΕΓΧΟΣ ΜΕΓΕΘΟΥΣ ΣΧΗΜΑΤΟΣ
            if ( (length(B{sa})>tempballmin) && (length(B{sa})<tempballmax) && (abs(stats(sa).Eccentricity-0.5)<0.3))
                tempcolorarray=impixel(pic,stats(sa).Centroid(1),stats(sa).Centroid(2) ); % ΑΝΑΓΝΩΣΗ ΧΡΩΜΑΤΟΣ
                if (TempballColorArray>tempcolorarray) % ΧΡΩΜΑΤΙΚΗ ΚΑΡΑΓΡΑΦΗ ΜΠΑΛΑΣ ΜΑΣΙ ΜΕ ΑΝΟΧΕΣ
                    sx=stats(sa).BoundingBox(1)+(stats(sa).BoundingBox(3)/2);
                    sy=stats(sa).BoundingBox(2)+(stats(sa).BoundingBox(4)/2);
                    tmpballcolorread=impixel(pic,sx,sy);
                    ColorArray(3,:)=tmpballcolorread-ColorAnnot; % Ballcolor MIN
                    ColorArray(4,:)=tmpballcolorread+ColorAnnot; % Ballcolor MAX
                    ColorArray(3,find(ColorArray(3,:)<0))=0; % ΔΙΩΡΘΩΣΗ ΤΙΜΩΝ ΕΚΤΩΣ ΟΡΙΩΝ (ΑΡΝΗΤΙΚΕΣ ΤΙΜΕΣ ΧΡΩΜΑΤΩΝ)
```

```
                    ColorArray(4,find(ColorArray(4,:)>254))=254; % ΔΙΩΡΘΩΣΗ ΤΙΜΩΝ ΕΚΤΩΣ ΟΡΙΩΝ ΠΑΛΕΤΑΣ ΧΡΩΜΑΤΩΝ
                    comp_data(1,2)=stats(sa).BoundingBox(3); % ΜΕΓΕΘΟΣ ΜΠΑΛΑΣ ΜΕ thress=0.1
                    BallId=sa;
                    error=0;
                end
            end
        end
    end

    % ΑΝΑΓΝΩΡΙΣΗ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ ΑΝΑ ΚΑΘΕ THRESHOLD
    for sa=1:length(B)
        if ((abs(stats(sa).Eccentricity-0.5)<0.2) && (thresst<1)) % ΕΛΕΓΧΟΣ ΣΧΗΜΑΤΟΣ
            RoboId=0;
            if ( (length(B{sa})>50) && (length(B{sa})<100) ) % ΜΕΓΕΘΟΣ ΣΥΜΦΩΝΑ ΜΕ ΕΝΑΝ ΡΟΜΠΟΤΙΚΟ ΠΑΙΚΤΗ
                        RoboId=sa; % ΕΥΡΕΣΗ ROBO ID
            end

            if (RoboId>0)
                frame=stats(RoboId).Image;
                imagesc(frame);
                [HB,HL]=bwboundaries(frame,'holes');
                Hstats =regionprops(HL,'Image','Centroid','FilledArea');

                if ( length(HB)==3) % ΕΑΝ ΕΙΝΑΙ ΚΑΙ ΤΑ ΤΡΙΑ ΟΚ , ΤΟΤΕ ΝΑ ΚΑΤΑΓΡΑΨΕΙ ΤΑ ΔΕΔΟΜΕΝΑ ΤΟΥ ΑΝΤΙΚΕΙΜΕΝΟΥ
                    treshdata(indx,4)=thresst;
                    treshdata(indx,1)=Hstats(1).FilledArea;
                    treshdata(indx,2)=Hstats(2).FilledArea;
                    treshdata(indx,3)=Hstats(3).FilledArea;
                    temproboarray(indx,3)=RoboId; % roboId ΑΠΟ ΤΟΝ ΠΙΝΑΚΑ B
                    temproboarray(indx,2)=stats(RoboId).BoundingBox(3); % ΜΕΓΕΤΗΟΣ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ
                    temproboarray(indx,1)=thresst; % ΤΡΕΧΟΝ THRESOLD

                    [Y,NDX] = sort(treshdata(indx,:)); % ΤΑΞΙΝΟΜΗΣΗ
                    a1=Y(end-1)/Y(end); % ΕΛΕΓΧΟΣ ΤΕΤΡΑΓΩΝΩΝ
                    a2=Y(end-2)/Y(end-1);
                    loga1a2=a1/a2;
                    dok=dok+1;

                    % ΧΡΩΜΑΤΙΚΟΣ ΕΛΕΓΧΟΣ ΣΤΟ 3.5x3.5 ΤΕΤΡΑΓΩΝΟ.
                    ImageCentroidX=Hstats(NDX(end-1)).Centroid(1)+stats(RoboId).BoundingBox(1);
                    ImageCentroidY=Hstats(NDX(end-1)).Centroid(2)+stats(RoboId).BoundingBox(2);
                    % ΧΡΩΜΑΤΙΚΟΣ ΕΛΕΓΧΟΣ ΣΤΟ ΜΕΓΑΛΟ ΤΕΤΡΑΓΩΝΟ
                    temproboarray(indx,4:6)=impixel(pic,ImageCentroidX,ImageCentroidY)-ColorAnnot;

                    indx=indx+1;
                    imagesc(stats(RoboId).Image);

                    drawnow;
                end
            end
        end
    end % for

    nindx=nindx+1;

    if ( (threshold_tune==1) && (thresst>0.98) ) % ΧΡΗΙΜΟΠΟΙΕΙΤΕ ΜΟΝΟ ΣΤΗΝ ΕΓΚΑΤΑΣΤΑΣΗ ,
                                                 % ΓΙΑ ΕΥΡΕΣΗ ΤΩΝ ΤΙΜΩΝ THRESHOLD ΣΕ ΔΙΑΦΟΡΕΣ ΘΕΣΕΙΣ.
        thresst=0.01;
        treshdata=[0,0,0,0];
        dok=0;
    else
        thresst=thresst+0.01;
    end;
end

[m,n]=(size(treshdata)); % ΜΕΓΕΘΟΣ TRESHDATA
```

```
265 -  if ((treshdata(1)>300) && (m>=1) ) % ΕΛΕΓΧΟΣ ΓΙΑ ΣΩΣΤΕΣ ΑΝΑΓΝΩΡΙΣΕΙΣ ΡΟΜΠΟΤΙΚΩΝ ΠΑΙΚΤΩΝ
266 -      Thold=treshdata(round(m/2),4); % ΜΕΣΗ ΤΙΜΗ THRESHOLD
267 -  else
268 -      error=1;
269 -  end
270
271    % ==========================
272    % OBJECT COLORS -> ΕΥΡΕΣΗ ΧΡΩΜΑΤΩΝ
273    % ==========================
274
275
276 -  if (error==1)
277 -      errortext
278 -      comp_data(8,1)=error;
279 -  else
280 -      treshdata=[0];
281
282        % ΔΗΜΙΟΥΡΓΙΑ ΑΝΟΧΩΝ ΧΡΩΜΑΤΟΣ ΤΟΥ ΡΟΜΠΟΤΙΚΟΥ ΠΑΙΚΤΗ ΚΑΙ ΤΗΣ ΜΠΑΛΑΣ
283 -      tmproboobj=find((temproboarray)==Thold);
284 -      ColorArray(1,:)=temproboarray(tmproboobj,4:6);
285 -      ColorArray(1,find(ColorArray(1,:)<0))=0;
286 -      ColorArray(2,:)=ColorArray(1,:)+ColorAnnot; % Robocolor MAX
287 -      ColorArray(2,find(ColorArray(2,:)>254))=254; % ΔΙΩΡΘΩΣΗ ΤΙΜΩΝ ΕΚΤΩΣ ΟΡΙΩΝ ΠΑΛΕΤΑΣ ΧΡΩΜΑΤΩΝ
288
289        % ΜΕΤΑΦΟΡΑ ΤΙΜΩΝ ΣΤΟΝ ΠΙΝΑΚΑ ΕΠΙΣΤΡΟΦΗΣ ΠΡΟΣ ΤΟ ΒΑΣΙΚΟ ΠΡΟΓΡΑΜΜΑ
290 -      comp_data(1,1)=Thold;
291 -      comp_data(1,3)=temproboarray(tmproboobj,2); % RoboId megethos sto sosto threshold
292 -      comp_data(2,:)=ColorArray(1,:); % Roboid color min
293 -      comp_data(3,:)=ColorArray(2,:); % Roboid color max
294 -      comp_data(4,:)=ColorArray(3,:); % Ballid color min
295 -      comp_data(5,:)=ColorArray(4,:); % Ballid color max
296 -      comp_data(6,1:2)=crtable(1:2); % X , Y  (αρχη αριστερης γωνιας γηπεδου στο φυσικο μεγεθος foto)
297 -      comp_data(7,1:2)=crtable(3:4); % X+ , Y+ (μαγεθος playground)
298 -      comp_data(8,1)=error;
299 -  end
300
301 -  end
```

7.3 Κώδικας μικροελεγκτή (prog.c)

```
////*****************************************************/
// program was produced by the
//CodeWizardAVR V1.24.6 Professional
//Automatic Program Generator//
//© Copyright 1998-2005 Pavel Haiduc, HP InfoTech s.r.l.
//http://www.hpinfotech.com
//e-mail:office@hpinfotech.com

//Project : Robotic Soccer
//Version :
//Date    : 11/05/2014
//Author  : J. ch.
//Company : TEI Crete
//Comments: Mechanical engineer -


//Chip type           : ATmega8535
//Program type        : Application
//Clock frequency     : 16,000000 MHz
//Memory model        : Small
//External SRAM size  : 0
//Data Stack size     : 128
//*****************************************************/

//              OCR2 = encoder1 = MOTOR2 =
//              OCR0 = encoder2 = MOTOR1
//
//

#include <mega8535.h>
#include <stdio.h>
#include <delay.h>
#include <math.h>

#define ENC1_CHA  PINB.0  //Encoder 1 Channel A input
#define ENC2_CHA  PINB.1  //Encoder 2 Channel A input

#define MOTOR1_DIR  PORTB.2  //Direction for motor1
#define MOTOR2_DIR  PORTD.6  //Direction for motor2


#define max_str_move    200 // mm logo sqrt((150^2)+(130^2))= 198,49 ~ 200cm = 2000 mm

#define W_PER 138.2      // wheel perimeter = 2*PI*r = 2*3,14159*2,2 cm; = 138.2mm
                         // points (encoder)
#define points_str 21    // 3870 points -> 138.2 mm, -> 1mm = 28 points

#define points_rot 15    // 90o -> 1/4 cycle -> 3870 * 0,25=967 points ->
                           // 967/90o = 10.75 points / moira   -> 11
#define wpoints 3870


```

```
53  void setup(void);
54  //void printdec(unsigned int, unsigned char , unsigned char );
55  //void print_new_line (void);
56  void prepare_Sdata(void);
57
58
59  unsigned int points,mlimit=0;
60  unsigned char mvs=0;
61  unsigned char init=0,c1,c2,d1,d2;
62
63  unsigned char com=0,ocb,a,b,c,d,sum,cb,n1,n2;
64  unsigned char m=0,oldcb=198,angle=0;
65  unsigned char i;
66  unsigned char cc,recfl=0,j,sercount,mm=100;
67
68  unsigned char serbyte[4]={0,0,0,0};
69  unsigned char fsfd[3]={0,0,0};
70  signed int datbyte[3]={0,0,0};
71  unsigned int h_enc=0,left_enc=0,right_enc=0,spdchg=5000,ttt=1;
72
73
74  // encoder 1 -> MOTOR2
75  // encoder 2 -> MOTOR1
76  // c1 right
77  // c2 left
78
79  // External Interrupt 0 service routine left motor
80  interrupt [EXT_INT0] void ext_int0_isr(void)
81  {
82      //    if (mvs==1)
83              if (left_enc++>mlimit)
84                  d1=d2=c1=c2=0;
85
86
87  }
88
89  // External Interrupt 1 service routine right motor
90  interrupt [EXT_INT1] void ext_int1_isr(void)
91  {
92       //    if (mvs==1)
93              if (left_enc++>mlimit)
94                  d1=d2=c1=c2=0;
95
96  }
97
98
99  // Timer 1 overflow interrupt service routine
100 interrupt [TIM1_OVF] void timer1_ovf_isr(void)
101 {
102
103 }
104
```

```
105 //---------------- UART RECEIVE INTERRUPT ----------------------
106 interrupt [USART_RXC] void usart_rx_isr(void)
107 {
108
109 // if (UDRE) com=UDR;
110
111 //delay_ms(50);   //   ROBOT ID=0
112
113  if (0b10000000 & UCSRA)       // INCOMING BYTES
114  {
115     com=UDR;
116
117     serbyte[sercount]=com;
118
119     if (sercount>=3)  // after 4 bytes is enabled
120     {
121        recfl=10;
122        sercount=0;
123     }
124     else sercount++;
125
126   }
127  }
128
129 /*
130 void print_new_line (void)
131 {
132     UDR=10; delay_ms(25); //print new line
133     UDR=13; delay_ms(25); //print enter
134 }
135   */
136
137 //-----------------------------------------------------
138 void setup (void)
139 {
140 DDRA=0x00;PORTA=0x00;
141 DDRB=0x0C;PORTB=0x03;
142 DDRC=0x00;PORTC=0x00;
143 DDRD=0xC0;PORTD=0x0C;
144
145 // Timer/Counter 0 initialization
146 // Clock source: System Clock
147 // Clock value: 62,500 kHz
148 // Mode: Phase correct PWM top=FFh
149 // OC0 output: Non-Inverted PWM
150 TCCR0=0x64;//  1  TCCR0=0x65; // bazo ck/1024 giati htan ck/256
151 TCNT0=0x00;
152 OCR0=0x00;
153
154 // Timer/Counter 1 initialization
155 // Clock source: System Clock
156 // Clock value: 250,000 kHz
157 // Mode: Normal top=FFFFh
```

```
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: On
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
//TCCR1B=0x00;
TCCR1B=0x03;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: 62,500 kHz
// Mode: Phase correct PWM top=FFh
// OC2 output: Non-Inverted PWM
ASSR=0x00;
TCCR2=0x66;//TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;


// External Interrupt(s) initialization
// INT0: On
// INT0 Mode: Rising Edge
// INT1: On
// INT1 Mode: Rising Edge
// INT2: Off
GICR|=0xC0;
MCUCR=0x0F;
MCUCSR=0x00;
GIFR=0xC0;

// Timer(s)/Counter(s) Interrupt(s) initialization
//TIMSK=0x00;
TIMSK=0x04;


// USART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART Receiver: On
// USART Transmitter: On
// USART Mode: Asynchronous
```

```
210
211
212 // USART Baud rate: 19200 (single Speed Mode)    SOSTO
213
214 UCSRA=0x02;      //     MALLON PAIZEI STATHERA H EPIKOINONIA ETSI ALLA PITHANO
215 //UCSRA=0x00;    //     NA EPHREAZEI KAI TA DYO ENCODER MAZI. NA KANO TEST  ?
216 UCSRB=0x98;
217 UCSRC=0x86;
218 UBRRH=0x00;
219 UBRRL=0x68;//UBRRL=0x67; -> (CLK / 16 /( 4800-1))  -> 208d = D0h
220                                 // --> (CLK / 16 /( 9600-1))  -> 104d = 68h
221
222
223 //UBRRL=0x67;
224 //UBRRH=0x00 >> 8;
225
226
227 // Analog Comparator initialization
228 // Analog Comparator: Off
229 // Analog Comparator Input Capture by Timer/Counter 1: Off
230 ACSR=0x80;
231 SFIOR=0x00;
232
233 // Global enable interrupts
234 #asm("sei")
235
236  }
237
238
239 /*
240 //--------------- PRINT DECIMAL NUMBER OF (n) CHARACTERS --------
241  void printdec(unsigned int x, unsigned char n, unsigned char fillch)
242
243 {
244     unsigned char i;
245     unsigned char s[10];
246     for (i = 0; i < n; i++) {
247         s[n - i - 1] = '0' + (x % 10);
248         x /= 10;
249     }
250     for (i=0; i<(n - 1); i++) {
251         if (s[i] == '0') s[i] = fillch; else break;
252     }
253     for (i=0; i<n; i++) {
254       UDR=s[i]; delay_ms(25);
255     }
256     //while (!CHECKBIT(UCSRA,5)); UDR=10; //print new line
257     //while (!CHECKBIT(UCSRA,5)); UDR=13; //print enter
258 }
259   */
```

```
//-----------------------------------------------------
//-----------------------------------------------------

void prepare_Sdata(void)
{
   unsigned char k,idstart,mm;
   signed char hcb,n1,n2;

   i=0;sum=0;


hcb=-1; i=0; k=0;
                // find CB byte if exist
while ((i<=3) && (hcb==-1))  // reset robot id=0 -> 199, id=1 -> 200, id=2 -> 201
{

      if (serbyte[i]>=100) //  all bytes checked and the last valid CB byte is kept

      {
         k=(serbyte[i]/10)%10;       // CB 2o number checked for reset

         if (k<3)   // if is a legal CB
         {
             hcb=i;
         }
      }
      i++;

}

if (hcb>=0)  // check for legal cb for this robot -> id (0,1,2)
{
   if (serbyte[hcb]==199)
   {
       datbyte[oldcb]=0;  // 0 for possible problem in case of direction change
       oldcb=0;
       hcb=-1;
   }
   else
   {
       if (serbyte[hcb]!=199) //  if a old CB then ser_reset
       {
          cb=hcb;  // position in array



          oldcb=serbyte[cb];
          a=k;                    // CB 2o -> id CB + type of the DB bytes
          b=(serbyte[cb]%10);     // CB 3o -> chksum
          datbyte[a]=0;           // reset dest , spd , ang

       }
   }
```

```
313     sum=0;
314     fsfd[0]=fsfd[1]=fsfd[2]=0;
315  }
316
317
318  if ((serbyte[cb]!=199) && (hcb>=0)) // id check
319  {
320
321   idstart=0;  // depend on id for  id=1 -> 3 gia id=2 ->6
322   i=0;
323
324
325   while ((i<=3))
326   {
327   if ((serbyte[i]/100)<1)    // CB 1o byte // check for right CB's
328      {
329          c=(serbyte[i]/10)%10;  //DB 2o -> for id=0 depend of the series TYPE=0,1,2
330                                 // -> c=0,1,2
331
332          if ((c<3) )    //  ->  1o,2o,3o number
333          {
334
335             d=(serbyte[i]%10);   // DB 3o -> Number
336
337             switch(c)   //  type (dest=0, spd=1, ang=2)
338             {
339                 case 0:  // example for id=1 -> datbyte[c-3] and  id=2 datbyte[c-6]
340                          datbyte[a]=datbyte[a]+ ((int) d*100);//1o number
341                          fsfd[c]=1;
342                          break;
343                 case 1:
344                          n1=d;
345                          datbyte[a]=datbyte[a]+ ((int) d*10);//2o number
346                          fsfd[c]=1;
347                          break;
348                 case 2:
349                          n2=d;
350                          datbyte[a]=datbyte[a]+d;   // 3o number
351                          fsfd[c]=1;
352                          break;
353             }
354             fsfd[c]=1;
355             sum=1*fsfd[0]+2*fsfd[1]+4*fsfd[2];
356
357            if (n2>=n1) mm=n2-n1;
358            else mm=n1-n2;
359
360             if ((mm==b) && (sum==7)) // check like checksum thath all  bytes came and are right
361             {
362                 i=100;
363             }
364          }
365
```

```
366         } // if
367         i++;
368     }
369
370
371     if (mm==b)
372     {
373
374       oldcb=198;
375       fsfd[0]=fsfd[1]=fsfd[2]=0;
376       recfl=1;
377       right_enc=0;
378       left_enc=0;
379
380
381       if (datbyte[0]>0) // straight movement
382       {
383             mvs=1;
384             mlimit=(datbyte[0])*points_str*2;
385             if (mlimit>10000) // because of big distance robot will be travel a bit more
386             {
387                     c1=35;      // speed 35 from 0-254
388                     d1=0;
389                     d2=1;
390                     c2=255-35; // opposite speed 35 from 0-254, in order to move forware
391             }
392             else
393             {
394                     c1=35 ;
395                     d1=0;
396                     d2=1;
397                     c2=255-35;
398             }
399
400
401             datbyte[0]=0;
402
403       }
404       else
405       {
406           if (datbyte[2]>0)      // rotational movement
407           {
408               mvs=2;
409               if (datbyte[2]>180)
410               {
411                         if (datbyte[2]<=360) //  left turn ( 181 - 360 )
412                         {
413                             c1=20;
414                             d1=0;
415                             d2=0;
416                             c2=20;
417                             mlimit=(360-datbyte[2])*points_rot;
418
419                             datbyte[2]=0;
```

```
420
421                     recfl=3; // enable robot stoping
422                 }
423                 else        // special cases values ( 361 - 999 )
424                 {
425                     switch(datbyte[2])
426                     {
427                         case 450:  // after goal movement                                    c1=25;
428                                 d1=0;
429                                 d2=0;
430                                 c2=25;
431                                 mlimit=wpoints*3; // perform 3 turns
432
433                                 datbyte[2]=0;
434
435                                 recfl=3; // enable robot stoping
436
437                         break;
438                         case 458:  // left continues rotation
439                                 c1=80;
440                                 d1=0;
441                                 d2=1;
442                                 c2=255-35;
443                                 mlimit=2000;
444
445                                 datbyte[2]=0;
446
447                                 recfl=3;
448
449                         break;
450                         case 459:  // // right continues rotation
451                                 c1=35;
452                                 d1=0;
453                                 d2=1;
454                                 c2=255-80;
455                                 mlimit=2000;
456
457                                 datbyte[2]=0;
458
459                                 recfl=3;
460
461                         break;
462                         case 800: // stop moving
463                                 c1=0;
464                                 d1=0;
465                                 d2=0;
466                                 c2=0;
467
468
469                                 datbyte[2]=0;
470
471                                 recfl=3; // enable rot stoping
472
473             break;
```

```
                break;
                                case 502: // goal movement ?
                                          c1=30;
                                          d1=0;
                                          d2=0;
                                          c2=30;

                                          datbyte[2]=0;

                                          recfl=3;

                break;
                                } // switch

                        } // if > 360


                    }
                else   // < 180
                {
                   if (datbyte[2]<=180) //  right turn ( 0 - 180 )
                    {
                           c1=255-20;
                            d1=1;
                            d2=1;
                            c2=255-20;
                            mlimit=(datbyte[2]*points_rot);
                            datbyte[2]=0;
                            recfl=4;
                    }
                } // }  // >180

            } // [2]  >0
        } // [0] >0
    }
    else
    {           // in case of wrong byte/characters
        recfl=0;
        h_enc=65500;move encoder to max
                // performed movement interrupt but with h_enc<mlimit
        right_enc=65500;
        left_enc=65000;
    } // mm==bb
} //  (serbyte[cb]!=199)
} // func


void main(void)
{

 setup();

```

```
525   recfl=0; sercount=0; cc=255;
526   d1=1; c1=255;
527   d2=0; c2=0; h_enc=0;
528
529
530   while (1)
531   {
532           MOTOR1_DIR=d1;                  /* variables are global  */
533           MOTOR2_DIR=d2;
534           OCR0=c1;
535           OCR2=c2;
536
537
538            if (recfl==10)      /*  check for valid nu,ber and type of bytes in order to enable movement*/
539            {
540                  prepare_Sdata();
541                  recfl=0;
542            }
543   }
544   }
545
```