2010

Πτυχιακή εργασία – Χαρτογράφηση χώρου με αυτοκινούμενο Ρομπότ

Μανιάς Νίκος

Πτυχιακή Εργασία



Μανιας Νίκος

Επιβλεπων Καθηγητης : Γιώργος Παλαμάς

Πίνακας περιεχομένων

Περιεχόμενα....2

Κεφάλαιο 1- Εισαγωγή....4

1.1 Σκοπός της άσκησης....4

1.2 Τι είναι λοιπον το ρομπότ?....4

1.3 Αυτοκινούμερα ρομπότ....6

1.4 Το Lego Mindstorms NXT....9

1.5 Το Ρομπότ....11

Κεφάλαιο 2- Το πρόβλημα της πλοήγησης και εντοπισμού θέσης....14

2.1 Σημάδια (Landmarks)....15

2.2 Ανάλυση εικόνας....17

2.3 Ιστογράμματα....20

2.4 Feature Based vs Appearance Based....25

Κεφάλαιο 3- Θεωρητική προσέγγιση της εργασίας....26

3.1 Η διαδικασία που θα ακολούθησουμε....26

3.1.2 Μετρικές....31

3.2 Κατηγοριοποίηση Εικόνων....32

3.2.1 Κατηγοριοποίηση βαση «κοντινότερου γείτονα»....33

3.2.2 Κατηγοριοποίηση με πρότυπα....34

3.3 Τεχνητά Νευρωνικά δίκτυα....36

3.3.1 Ιστορία Νευρωνικών Δικτύων....38

3.3.2 Τα πρώτα μοντέλα....41

3.3.3 Η ωρίμανση....42

3.3.4 Η κατάσταση σήμερα....45

Κεφάλαιο 4- Εφαρμογή σε Matlab και OpenCv....48

4.1 Εισαγωγή....48

4.2 Υλοποίηση με Matlab....50

4.3 OpenCv – C# - Emgu CV....54

4.3.1 Η φόρμα μας....55

Κεφάλαιο 5- Προοπτικες- Επίλογος……58

5.1 Προοπτικες……59

5.2 Επίλογος……59

Ευχαρηστήρια……59

Βιβλιογραφία……59

Παράρτημα……60

ΚΕΦΑΛΑΙΟ 1
ΕΙΣΑΓΩΓΗ

1.1 Σκοπός της άσκησης

Στην σημερινή εποχή η ανάγκη για αυτοματοποίηση πολλών διαδικασιών γίνεται όλο και πιο έντονη. Συνεχεία ακούγονται λέξεις όπως : βελτιστοποίηση ποιότητας ζωής, αυτοματοποίηση στον βιομηχανικό τομέα, έξυπνο σπίτι και άλλα..

Στο μέλλον υπολογίζεται ότι κάθε νοικοκυριό η χώρος εργασίας θα έχει αυτοκινούμενες μηχανές οι οποίες θα πραγματοποιούν εργασίες που είναι αδύνατες, κουραστικές η και βαρετές για τον άνθρωπο. Αυτές οι μηχανές, οι αποκαλούμενες Ρομπότ θα πρέπει να έχουν αναπτύξει ένα δικό τους τρόπο πλοήγησης, απόφασης και δράσης. Σε αυτή την εργασία θα πειραματιστούμε με το ανοικτό ακόμα πρόβλημα της πλοήγησης ενός ρομπότ μέσα σε κλειστό χώρο, καθώς και στον διαχωρισμό περιπτώσεων και αποφάσεων για κάθε ξεχωριστή περίπτωση.

Τι είναι λοιπόν το Ρομπότ?

Η λέξη ρομπότ μπορεί να αναφέρεται σε μηχανικά ρομπότ η εικονικά προγράμματα, τα οποία συνήθως αποκαλούνται bots. Μιας και δεν υπάρχει διαχωρισμός μέσω κάποιου κανόνα ανάμεσα σε ρομπότ και μηχάνημα, κατά γενική ομολογία ένα ρομπότ μπορεί να κάνει κάτι η κάποια από τα παρακάτω :

- να κινείται,
- να χειρίζεται ένα μηχανικό άκρο (βραχίονα),
- να αισθάνεται και να αντιδρά στο περιβάλλον και να
- μιμείται κάποια νοημοσύνη, ιδιαίτερα ανθρωπινή η κάποιου ζώου.

Γενικά, υπάρχει κάποια σύγκρουση στον αν ο όρος Ρομπότ μπορεί να αποδοθεί σε μηχανήματα που λειτουργούν με τηλεχειρισμό, η αν είναι αποκλειστικά με μηχανές που διαθέτουν την δίκη τους νοημοσύνη και είναι αυτοδιατηρουμενα. Αυτές οι απόψεις διαφέρουν από πολιτισμό σε πολιτισμό, από χωρά σε χωρά και από άτομο σε άτομο.

Ιστορίες από τεχνητούς βοηθούς και συντρόφους η και προσπαθείες να δημιουργηθούν υπάρχουν από παλιά, όμως τα πρώτα πλήρως αυτόνομα ρομπότ δεν ήρθαν στο φως πριν από $20^{ο}$ αιώνα. Το πρώτο ψηφιακό και προγραμματιζόμενο ρομπότ το Unimate, εγκαταστάθηκε το 1961 και η εργασία του ήταν να σηκώνει καυτά κομμάτια σιδηρού και να τα στοιβάζει. Σήμερα, τα ρομπότ χρησιμοποιούνται στην βιομηχανία και παντού οπού πρέπει να γίνουν εργασίες οι οποίες είναι επικίνδυνες, δύσκολες η ακόμα και ανιαρές και τα καταφέρνουν με μεγαλύτερη ταχύτητα και με μικρότερο κόστος.



Τάλως, το πρώτο ρομπότ που έχει αναφερθεί, στην ελληνική μυθολογία

Φυσικά, είναι σχεδόν αδύνατον να συγκρίνει κάποιος τον αριθμό των ρομπότ που υπάρχουν από χώρα σε χώρα, καθώς όπως είπαμε υπάρχουν πολλοί ορισμοί για τον ορό ρομπότ. Ο Διεθνής Οργανισμός Προτυποποίησης έδωσε ένα ορισμό στο ISO 8373: «ένα αυτοελεγχόμενο, αναπρογραμματιζόμενο, πολλαπλών χρήσεων μηχάνημα τριών η περισσότερων αξόνων το όποιο μπορεί είτε να είναι σταθερό σε μια βάση είτε αυτοκινούμενο για εργασία σε βιομηχανικά περιβάλλοντα».

Τα ρομπότ χωρίζονται σε διαφορές κατηγορίες. Στην δίκη μας έρευνα, χρησιμοποιήσαμε ένα αυτοκινούμενο ρομπότ.

Αυτοκινούμενα Ρομπότ

Ένα αυτοκινούμενο ρομπότ είναι ένα ρομπότ το όποιο μπορεί να κινηθεί σε ένα οποιοδήποτε περιβάλλον, αντιθέτως με τα βιομηχανικά ρομπότ οπού είναι συνήθως ένας βραχίονας πάνω σε μια βάση και στην άκρη υπάρχει ένας γάντζος για να εκτελεί εργασίες. Υπάρχουν διάφορα ειδή αυτοκινούμενων ρομπότ, όπως

- ρομπότ εδάφους,

- ιπτάμενα ρομπότ που συνήθως αναφέρονται ως μη επανδρωμένα ιπτάμενα οχήματα καθώς και

- υποβρύχια ρομπότ για κίνηση μέσα σε νερό (μη επανδρωμένα υποβρύχια οχήματα).



Ρομπότ Εδάφους



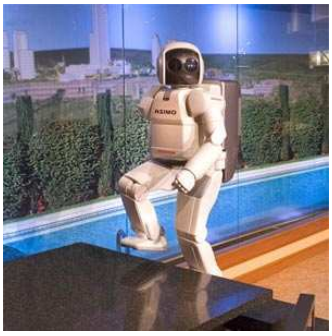
Ιπτάμενο Ρομπότ



Υποβρύχιο Ρομπότ

Για την κίνηση τους υπάρχουν διάφοροι τρόποι όπως :

- ανθρωποειδή πόδια
- πόδια που μιμούνται πόδια εντόμων η ζώων
- Έντροχα ρομπότ
- Ρομπότ που χρησιμοποιούν ράγες η ερπύστριες.



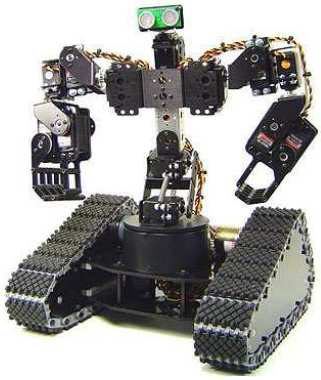
Ανθρωποειδή ποδιά



Εξαποδό Ρομπότ



Έντροχο Ρομπότ



με ερπύστριες

Στην δίκη μας περίπτωση, χρησιμοποιήσαμε έντροχο ρομπότ.

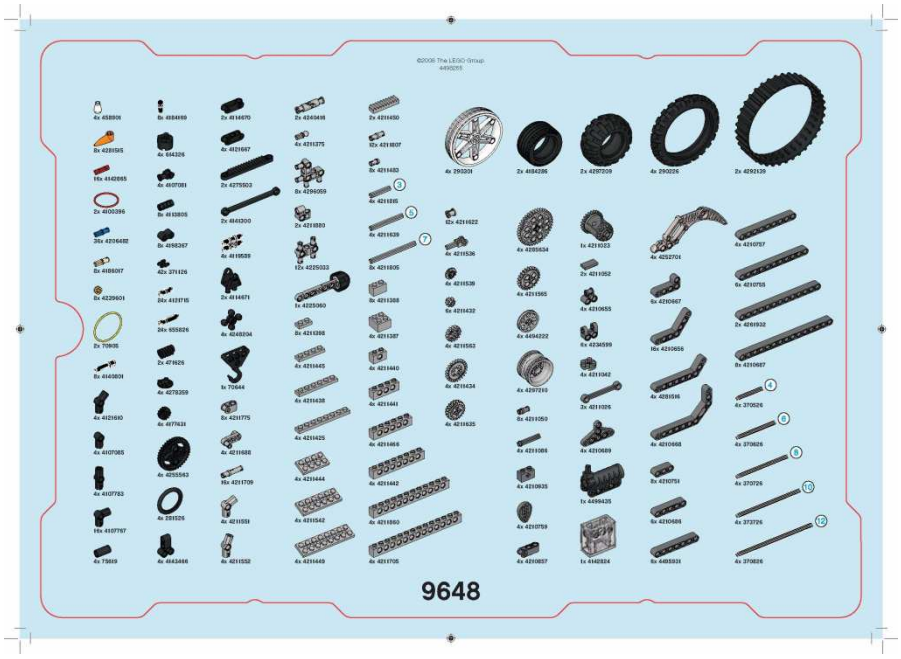
Το Lego Mind storms NXT 2.0

Στο πείραμα μας, μας ενδιέφερε το ρομπότ που θα χρησιμοποιήσουμε να είναι εύκολα προγραμματίσιμο, αρκετά ανθεκτικό σε κρούσεις και τρανταγμους και το πιο σημαντικό, να είναι εύκολα συναρμολογήσιμο.

Έτσι, αποφασίσαμε ότι το μοντέλο της Lego Mind storms NXT 2.0 είναι η καλύτερη επιλογή που ένας μη επαγγελματίας θα μπορούσε να έχει.



Το mind storms αποτελείτε από μικρά κομμάτια (όπως τα κλασσικά παιχνίδια της ίδιας εταιρείας) σε παρά πολλά σχήματα όπου είναι πολύ εύκολο να συνδεθούν μεταξύ τους δίνοντας έτσι την δυνατότητα να συναρμολογηθεί ένα οποιοδήποτε ρομπότ.

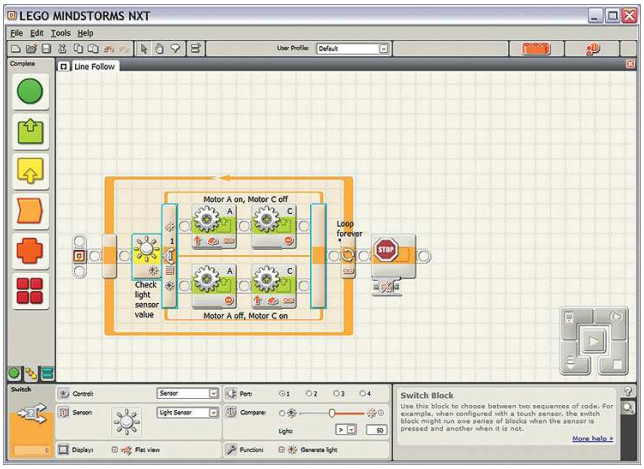


Τα κομμάτια που περιέχει μέσα ένα μοντέλο της Lego

Περιέχει ρόδες, γρανάζια και οτιδήποτε μπορεί κάποιος να χρησιμοποιήσει για να φτιάξει ένα εντροχο, βαδίζον η στατικό ρομπότ. Στην δική μας περίπτωση χρησιμοποιούμε 4 ρόδες με 2 κινητήρες, αλλά χρησιμοποιούμε ερπύστριες για να μεταφέρουμε την κίνηση στον κάθετο άξονα. Μας ενδιέφερε πολύ το ρομπότ να μπορεί να κάνει στροφή επί τόπου, καθώς με την πανοραμική όραση που θα του προσθέσουμε, αν θεωρητικά περιστραφεί όσες μοίρες και να είναι η πληροφορία που θα παίρνουμε από την κάμερα μας θα είναι ίδια. Περισσότερα όμως για αυτά θα πούμε παρακάτω.

Σε κάτι άλλο που έπρεπε να δώσουμε σημασία, είναι η ευκολία στον προγραμματισμό. Κάθε ρομπότ, για να εκτελέσει κινήσεις να πάρει αποφάσεις και να αναπτύξει μια «νοημοσύνη» πρέπει να προγραμματιστεί από άνθρωπο. Το mind storms λοιπόν είναι μια καλή επιλογή για τέτοιου είδους πειράματα καθώς είναι προγραμματισιμο σε πολλές διαφορετικές γλώσσες προγραμματισμού. Η ίδια η Lego έχει αναπτύξει μια δική της γλώσσα συγκεκριμένα για το mind storms,την

NXT-G, αλλά υπάρχει δυνατότητα να προγραμματιστεί σε C, C++, Java, .Net και σχεδόν οποιαδήποτε άλλη γλώσσα κάποιος επιθυμεί.



Nxt-g Η γραφική γλώσσα προγραμματισμού της Lego

Αρχικά, εμείς προγραμματίσαμε σε Matlab, ένα πρόγραμμα από την Mathworks, αλλά στην συνέχεια αποφασίσαμε ότι υπάρχουν κάποια αρνητικά σε αυτό. Έτσι, αναγκαστήκαμε να αλλάξουμε γλώσσα προγραμματισμού και τρόπο που τραβάμε και αποθηκεύουμε τις φωτογραφίες που χρησιμοποιούμε.

Το ρομπότ μας

Το ρομπότ μας αποτελείτε από ένα κεντρικό «τούβλο» της Lego πάνω στο οποίο συνδέονται και μεταδίδουν κίνηση δυο βηματικοι κινητήρες (σερβοκινητηρες-servomechanism). Οι σερβοκινητηρες η σέρβο είναι αυτόματες συσκευές που χρησιμοποιούν ανίχνευση σφάλματος για να διορθώσουν τυχόν προβλήματα στην απόδοση του μηχανισμού. Παρέχουν έλεγχο θέσης, ο οποίος συνήθως πραγματοποιείται με σύγκριση της εξόδου που πρέπει να παράγει ο κινητήρας με την πραγματική θέση του κινητήρα. Έτσι, αν χρειαστεί το σήμα που παράγεται διορθώνεται έτσι ο κινητήρας να έρθει στην θέση που πρέπει. Οι κινητήρες της Lego έχουν ακρίβεια μιας μοίρας (1^{o}).



Ένας σέρβο του εμπορίου



Ένας σέρβο της Lego

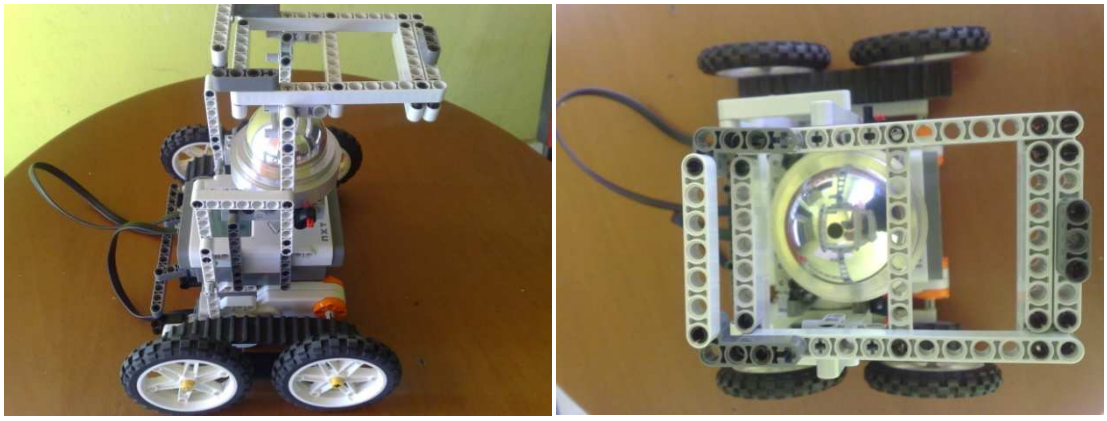
Επιπλέον, μιας και θέλαμε η κίνηση να μεταδίδεται κάθετα και όχι οριζόντια χρησιμοποιούμε 2 ερπύστριες για να μεταφέρουν την κίνηση από τον μπροστινό τροχό στον πίσω. Έτσι, αν η μια πλευρά του ρομπότ κινείτε αντίστροφα από την άλλη το ρομπότ μπορεί να κάνει στροφή 360 μοίρες επί τόπου.

Πάνω στο ρομπότ μας, έχουμε εγκαταστήσει μια μεταλλική, γυαλιστερή σφαίρα, η καλυτέρα ημισφαίριο. Ακριβώς πάνω από αυτό το ημισφαίριο, μια φωτογραφική μηχανή (μια οποιαδήποτε μηχανή) θα τρίβει φωτογραφίες το ημισφαίριο.

Πάνω λοιπόν στο ημισφαίριο έχουμε την πληροφορία που θα είχαμε αν είχαμε φωτογραφίσει τον χώρο που βρισκόμαστε σε 360 μοίρες. Ουσιαστικά, έχουμε τον αντικατοπτρισμό ολοκλήρου του δωματίου που φαίνεται από το σημείο που βρίσκεται το ρομπότ πάνω στην σφαίρα και φωτογραφίζοντας την είναι σαν να έχουμε την φωτογραφία ολοκλήρου το δωματίου.

Εδώ τονίζουμε το γεγονός ότι επειδή ακριβώς ο τρόπος φωτογράφισης είναι τέτοιος, σε περίπτωση που η σφαίρα περιστραφεί, όσες μοίρες και να είναι αυτί, η πληροφορία που βρίσκεται πάνω της είναι πάντα ακριβώς η ίδια. Για αυτό ακριβώς τον λόγο αναφέραμε πριν ότι το ρομπότ θέλαμε να μπορεί να κάνει επί

τόπου στροφή χωρίς να μετακινηθεί καθόλου από το σημείο που βρίσκεται το κέντρο της σφαίρας.



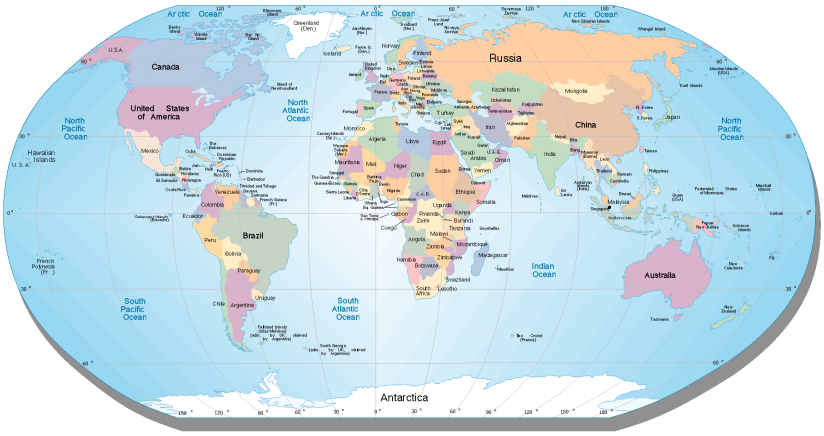
Το Lego Mindstorms Nxt σε σχημα έντροχου ρομπότ



Ο κρυσταλλος που φωτογραφίζουμε

Κεφαλαιο 2
Το προβλημα της πλοηγησης και του εντοπισμου θεσης.

Έχετε ποτέ αναρωτηθεί πως αναγνωρίζετε που βρίσκεστε? Για παράδειγμα, αν κάθεστε στον υπολογιστή μέσα στο δωμάτιο σας το οποίο βρίσκεται στην Οδό Γ.Γεωργιαδου 70, στο Ηράκλειο Κρήτης, στην Ελλάδα, στην Ευρώπη, στην Γη στο σύμπαν..



Πως ξέρουμε σε ποιο σημείο της γης βρισκόμαστε?

Από ότι καταλαβαίνουμε, ο εντοπισμός θέσης είναι ένα ιεραρχικό πρόβλημα το οποίο ξεκίνα από μικρότερες κλίμακες, όπως είναι το δωμάτιο μας, και επεκτείνετε σε μεγαλύτερες όπως είναι η Γη και το σύμπαν.

Στις μικρές κλίμακες βασιζόμαστε σε στοιχεία όρασης και βραχείας μνήμης, όπως πχ η διαρρύθμιση του χώρου του δωματίου η τα χρώματα. Σε μεγαλύτερη κλίμακα βασιζόμαστε σε εννοιολογική η

σημασιολογική γνώση. Πιο δύσκολη κλίμακα είναι η μικρή. Για παράδειγμα, βρίσκομαι στο δικό μου δωμάτιο η στου αδελφού μου? Είμαι στο δικό μου σπίτι η στου φίλου μου? Και αν θέλω, πως μπορώ να μετακινηθώ από το ένα σπίτι στο άλλο?

Στην Ρομποτική, τα παραπάνω ερωτήματα αποτελούν το Πρόβλημα πλοήγησης και εντοπισμού θέσης και περιλαμβάνει και την γνώση του «που βρίσκομαι» και την γνώση του «πως θα πάω από το σημείο Α στο σημείο Β». Μελέτες με ανθρώπους και ζώα δείχνουν ότι χρησιμοποιούνται 2 στοιχειά που χρησιμοποιούμε για να διαχωρίσουμε ένα χώρο από ένα άλλο, τα «σημάδια» και την «ανάλυση πληροφορίας από εικόνες- στατιστική εικόνας».



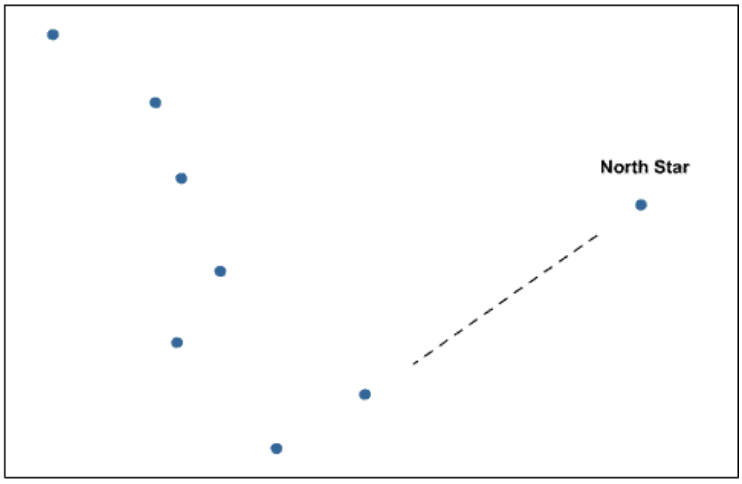
Πάνω σε τι βασιζόμαστε για να ξεχωρίσουμε το ένα δάσος από το άλλο?

Σημάδια (landmarks)

Τα σημάδια είναι διαχωρίσιμα στοιχειά στο περιβάλλον που λειτουργούν ως «πινακίδες». Για παράδειγμα, αν ξέρω πως είναι ένας τοίχος του δωματίου του δικού μου σπιτιού και έτσι μπορώ να το ξεχωρίσω από το υπόλοιπο σπίτι. Επιπλέον, αν δεν έχω την ίδια

εικόνα για κάποιο τοίχο του σπιτιού του φίλου μου, τότε μπορώ να ξεχωρίσω το δικό μου σπίτι από το δικό του.

Παλαιοτέρα, τέτοια «σημάδια» χρησιμοποιούσαν εξερευνητές, ναυτικοί και πολλοί άλλοι που ταξίδευαν σε άγνωστα μέρη. Από όπου περνούσαν, σημείωναν πράγματα και τοποθεσίες που ήταν μοναδικά με σκοπό αν κάποια στιγμή θελήσουν να επιστρέψουν κακολογώντας τα σημάδια αυτά να γυρίσουν σε κάποιο σημείο το οποίο είναι λιγότερο επικίνδυνο να περάσουν την νύχτα η να εξερευνήσουν που οδηγεί κάποιος άλλος δρόμος.



Η Βόρεια Άρκτος, το πιο συνηθισμένο σημάδι που χρησιμοποιούσαν για πολλούς αιώνες οι ναυτικοί για να μπορούν να υπολογίσουν που είναι ο Βορράς και ο Νότος.

Τώρα πλέον, στην σημερινή εποχή, τέτοια σημάδια θα μπορούσαν να είναι κτήρια, λόφοι, σταυροδρόμια η ακόμα και άστρα- σίγουρα όμως κάτι που ξέρουμε ότι δεν θα μετακινηθεί η δεν θα βρούμε κάτι ακριβώς το ίδιο κάπου αλλού.

Επίσης, τέτοιου είδους σημάδια χρησιμοποιούμε και όταν δίνουμε οδηγίες για κάπου, πχ «στο δεύτερο περίπτερο θα στρίψετε δεξιά, στο τρίτο στενό αριστερά και όταν δείτε την ταμπέλα του φαρμακείου θα καταλάβετε ότι φτάσατε». Σε αυτό το παράδειγμα, χρησιμοποίησα 3

σημάδια : α) το περίπτερο β) το στενό γ) την ταμπέλα του φαρμακείου.

Φυσικά, αν και είναι αποτελεσματική μέθοδος, έχει και αυτή τα μειονεκτήματα της.

Ανάλυση εικόνας (image statistics)

Η ανάλυση εικόνας και η εξαγωγή πληροφορίας γινεται ολοένα και πιο διαδεδομένη στην εποχή μας. Με βάση το ρητό «μια εικόνα ίσον χίλιες λέξεις» οι επιστήμονες και οι ερευνητές του τομέα της τεχνητής όρασης, της αναγνώρισης προτύπων αλλά και την ρομποτικής στρέφουν προς τα εκεί τις έρευνες τους.

Μια εικόνα περιέχει παρά πολλή πληροφορία και μπορεί να μας διαφωτίσει για πολλά πράγματα που μπορεί να θέλουμε αλλά έχουμε μερικά προβλήματα:

1. Η φωτογραφίες συνήθως περιέχουν πολλή περισσότερη πληροφορία από αυτή που πραγματικά χρειαζόμαστε. Έτσι, οι έρευνες για την εξαγωγή χαρακτηριστικών γίνονται ολοένα και πιο συγκεκριμένες και λεπτομερείς για να μπορούμε να ξεχωρίσουμε τα αντικείμενα που θέλουμε από αυτά που δεν χρειαζόμαστε.



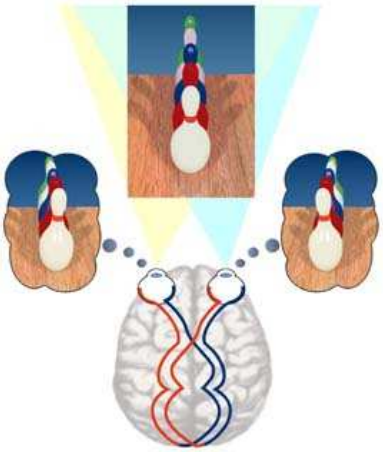
Αν μας ενδιέφερε μόνο το βάζο σε αύτη την εικόνα, όλα τα υπόλοιπα pixel αυτής της φωτογραφίας είναι αδιάφορη πληροφορία!

2. Ακόμα και δυο φωτογραφίες οι όποιες έχουν τραβηχτεί με ένα δευτερόλεπτο διάφορα μπορεί το περιεχόμενο τους να είναι πολύ διαφορετικό μιας και οι εξωτερικοί παράγοντες που επηρεάζουν μια φωτογραφία είναι παρά πολλοί:
    - Ο Φωτισμός
    - Η Κίνηση του αντικείμενου
    - Η ποιότητα του φακού (καμπυλότητα, ευκρίνεια)
    - Ακόμα και η μετατροπή από αναλογικό σε ψηφιακό σήμα μπορεί να παρουσιάσει προβλήματα αν οι συνθήκες είναι διαφορετικές
3. Φυσικά, οι φωτογραφίες (τουλάχιστον μέχρι τώρα, στον μέλλον ποιος ξέρει τι μπορεί να γίνει, ήδη βγαίνουν στην αγορά τρισδιάστατες κάμερες) αναγκαστικά είναι σε 2 διαστάσεις. Αυτό σημαίνει ότι χάνεται οποιαδήποτε πληροφορία περιέχεται στην διάσταση του βάθους.



Μια πολύ έξυπνη αναπαράσταση της ψευδαίσθησης βάθους που δημιουργείται με χρωματισμό σε εικόνες

Ο άνθρωπος καταφέρνει να έχει στερεοσκοπική όραση διότι χρησιμοποιεί 2 φακούς (τα ματιά του) μέσα από τους οποίους παίρνει την ίδια πληροφορία αλλά από μια διαφορετική γωνιά (γύρω στις 6 μοίρες). Έτσι ανάλογα με την απόσταση που έχει ένα σημείο από το άλλο ο εγκέφαλος μπορεί να καταλάβει πόσο πιο «μέσα» είναι ένα αντικείμενο από ένα άλλο. Κάτι τέτοιο όμως στις εικόνες είναι αδύνατο. Μπορεί να μην το καταλαβαίνουμε αμέσως γιατί έχουμε συνηθίσει σε κάτι τέτοιου είδους προβλήματα, αλλά όταν προσπαθούμε να εντοπίσουμε πχ ένα τραπέζι για να μην το χτυπήσει το ρομπότ μας τότε έχουμε ίσως να μη τα καταφέρει τόσο καλά.



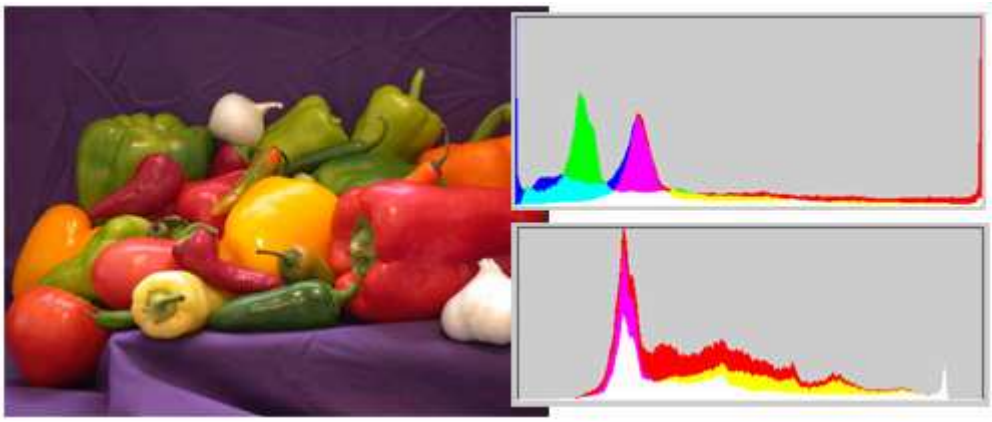
Σχεδιάγραμμα πως αντιλαμβάνεται ο άνθρωπος την αίσθηση του βάθους.



Στην ίδια φωτογραφία με πριν, μπορούμε να είμαστε σίγουροι ότι το πράσινο φωτιστικό μαζί με το πράσινο τραπεζάκι που βρίσκεται αριστερά στην φωτογραφία είναι πάνω στο τραπέζι η πίσω του?

Ιστογράμματα

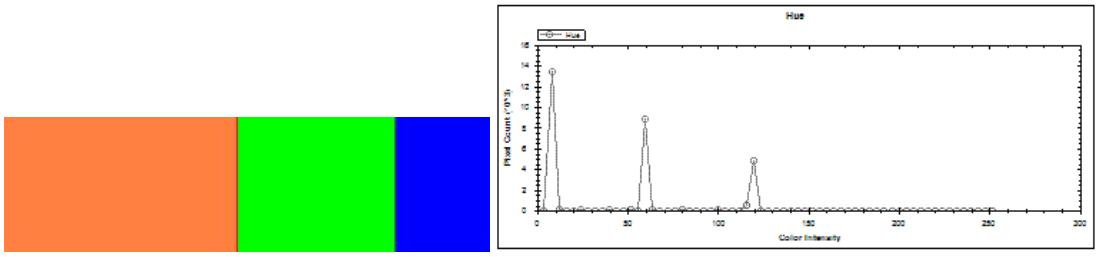
Η πιο διαδεδομένη ανάλυση εικόνας, σχετικά γρήγορη και απλή, είναι η εξαγωγή ιστογράμματος της εικόνας. Ένα *ιστόγραμμα* μέτραει πόσες φόρες εμφανίζεται ένα χαρακτηριστικό σε διαφορετικές τιμές. Για παράδειγμα, ένα ιστόγραμμα χρώματος μέτριοι τον αριθμό των πήξε σε κάθε κατηγορία χρώματος : Αν χωρίσουμε τον χρωματικό χώρο σε χρώματα, τότε κάθε φορά που ένα πήξε αντιστοιχεί σε ένα από τα χρώματα, ο μετρητής αυξάνεται κατά ένα. Εάν το κάνουμε αυτό σε μια ολόκληρη εικόνα τότε παίρνουμε ένα είδος «συχνότητας» χρώματος το οποίο αντιπροσωπεύει των αριθμό των πήξε που αντιστοιχούν σε κάθε κατηγορία. Αυτός ο πινάκας από στοιχειά λοιπόν είναι αυτό που αποκαλούμε ιστόγραμμα μιας εικόνας και συνήθως μας δίνει ένα χαρακτηριστικό πινάκα που αντιστοιχεί σε μια φωτογραφία.



Μια εικόνα από το Mat lab και δίπλα τα ιστογράμματα χρωματων.

Για παράδειγμα, εάν η κουζίνα έχει πολύ χρώμα που είναι κοντά στην απόχρωση του κίτρινου ενώ το καθιστικό δωμάτιο είναι κατά κύριο λόγο άσπρο, από το ιστόγραμμα και μόνο μπορούμε να ξεχωρίσουμε αυτά τα δυο δωμάτια.

Από ότι φαίνεται, η κατανόηση της χρήσης της πληροφορίας που αντλούμε από μια εικόνα είναι το κλειδί για την μεγαλυτέρου επίπεδου αναγνώριση αντικειμένων, η οποία με την σειρά της θα βοηθήσει για την αναγνώριση με βάση τα σημάδια. Έτσι, το να αρχίσουμε με την στατιστική της εικόνας είναι μια καλύτερη ιδέα από την αναγνώριση η ανίχνευση αντικειμένων.



Ακριβώς από πάνω, βλέπουμε μια φωτογραφία (μπορεί να μην έχει πολύ πληροφορία αλλά δεν παύει να είναι εικόνα) και δίπλα της το ιστόγραμμα χρωμάτων. Αν κάποιος παρατηρήσει στο ιστόγραμμα διακρίνονται 3 κορυφές, όπου η κάθε μια αντιστοιχεί στο χρώμα που βλέπουμε στην εικόνα. Μάλιστα, βλέποντας το ύψος της κορυφής

μπορούμε να καταλάβουμε ότι η πρώτη κορυφή ανήκει στο πορτοκαλί μέρος της εικόνας, αφού καταλαμβάνει το μεγαλύτερο μέρος της εικόνας. Δεύτερο έρχεται το πράσινο και τρίτο το μπλε.

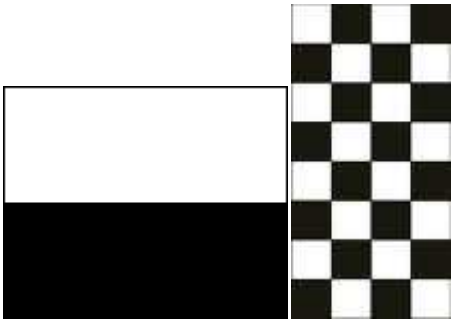
Πάνω σε αυτές τις διαφορές θα βασιστούμε για να έχει το ρομπότ μας την ικανότητα να διακρίνει μέσα από φωτογραφίες σε ποιο σημείο του δωματίου βρίσκεται.

Να τονίσουμε ότι. Όπως κάθε τι τα ιστογράμματα εχουν τα αρνητικά και τα θετικά τους.

- ✓ Οπτικα, είναι πολύ εύκολα και δύνατα σαν τρόπο επίδειξης πληροφορίας.
- ✓ Είναι εύκολα συγκρίσιμο με την κανονική καμπυλη.
- ✓ Συνήθως, ο κάθετος άξονας παρουσιάζει την μέτρηση της συχνότητας αντικειμένων που ανήκουν στην ίδια διαδικασία.

Και όπως παντα, υπάρχουν και τα αρνητικα τους.

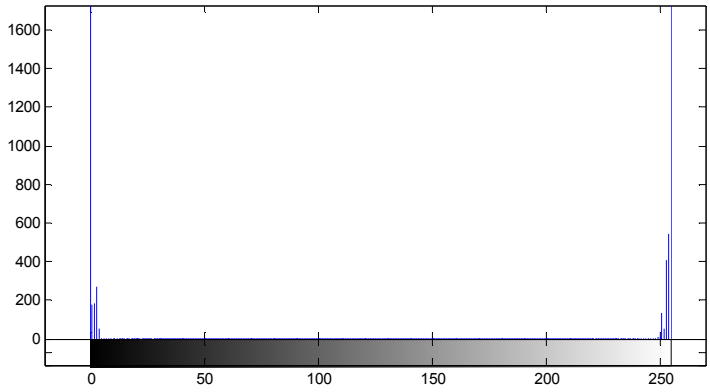
- ☒ Τα ιστογράμματα παρουσιάζουν το έξης πρόβλημα στην φάση της κατηγοριοποίησης: Αν και περιέχουν την πληροφορία, αποτυπώνουν μόνο την ποσότητα της. Δεν μας δείχνουν τίποτα απολύτως όσον αφορά το που βρίσκονται αυτά τα κόκκινα πήξε πχ.



Αυτές οι δυο φωτογραφίες έχουν το ίδιο ιστόγραμμα!

Για παράδειγμα, οι δυο φωτογραφίες αυτές αποτελούνται από δυο χρώματα, άσπρο και μαύρο. Ένα ιστόγραμμα τους θα μετρήσει ποσά λεύκα και ποσά μαύρα

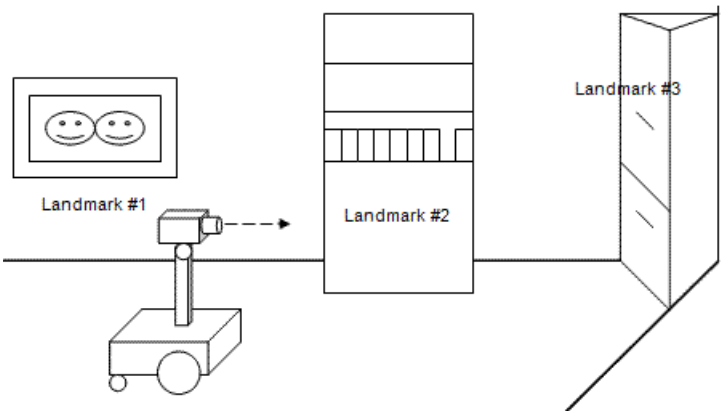
πήξε έχει κάθε εικόνα και για κάθε ένα θα προστίθεται μια μονάδα στην αντίστοιχη φωτεινότητα (αυτές οι δυο φωτογραφίες έχουν κορυφές στο 0 και στο 255). Έτσι, για κάθε μια από τις φωτογραφίες έχουμε ένα ιστόγραμμα της μορφής:



- ☒ Ακριβείς πληροφορίες δεν είναι πάντα ορατές καθως τα δεδομένα είναι ομαδοποιημένα σε κατηγορίες.
- ☒ Είναι δυσκολότερο να συγκρίνεις δύο σετ δεδομένων
- ☒ Λειτουργούν μόνο με συνεχή δεδομένα

Όμως, αρκετοί θα αναρωτηθούν: Γιατί απλά δεν χρησιμοποιούμε κάποια τεχνητά σημάδια τα οποία το ρομπότ μπορεί να ανιχνεύει και βάση αυτού να καταλάβει που βρίσκεται μέσα στο δωμάτιο?

Όντως, αν είχαμε ένα δωμάτιο το οποίο περιέχει κάποια μοναδικά σημάδια για να μπορεί το ρομπότ να τα ξεχωρίζει από το υπόλοιπο περιβάλλον, ανάλογα με την παραμόρφωση και το μέγεθος του, μπορεί να υπολογίσει πόσο μακριά βρίσκεται από αυτό η σε ποια γωνιά βρίσκεται.

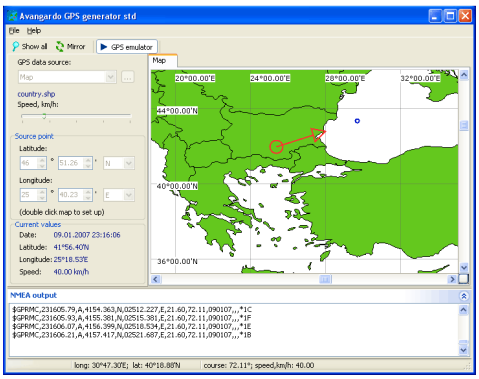


Ένα ρομπότ το οποίο κινείται μέσα σε ένα δωμάτιο με σημάδια μπορεί να καταλάβει που βρίσκεται.

Για παράδειγμα, αν τοποθετούσαμε στις τέσσερις γωνίες του ταβανιού στο δωμάτιο 4 διαφορετικού χρώματος χαρτιά και διδάσκαμε στο ρομπότ ποιο χρώμα είναι σε ποια γωνιά βρίσκεται πιο κοντά. Επίσης, αν ήξερε το μέγεθος του χαρτιού από μια απόσταση, έστω ένα μετρό, θα μπορούσε να συγκρίνει αυτό που βλέπει εκείνη την στιγμή με αυτό που ξέρει με σκοπό να βρει την απόσταση του από το σημάδι.

Βεβαία, δεν θα μπορούσαμε να χρησιμοποιήσουμε σημάδια με διαφορετικό σχήμα μόνο, καθώς όπως είπαμε οι φωτογραφίες είναι δυο διαστάσεων. Έτσι, το σημάδι φαίνεται πολύ διαφορετικό από διαφορετικές γωνίες, κάτι που θα μπορούσε να αποτρέψει το ρομπότ στο να το αναγνωρίσει, η ακόμα χειρότερα να νομίζει ότι βρίσκεται κάπου αλλού.

Επίσης, κάποιοι ίσως αναρωτηθούν γιατί δεν χρησιμοποιούμε το όλο και πιο διαδεδομένο GPS (global positioning system).



Λογισμικό GPS

Το GPS χρησιμοποιεί δορυφόρους για να εντοπίσει το σήμα που εκπέμπει μια συσκευή. Ανάλογα με την ισχύς του σήματος που δέχεται ο δέκτης, υπολογίζει την απόσταση του πομπού με του δεκτή. Όταν πλέον η συσκευή έχει καταφέρει να συγχρονιστεί με μερικούς δορυφόρους μπορεί να προβάλει το σημείο που έχει υπολογιστεί πάνω σε ένα χάρτη.

Παρόλα αυτά, δεν χρησιμοποιήσαμε κανένα από τους παραπάνω τρόπους! Γιατί? Σε αυτό που δώσαμε ιδιαίτερη σημασία στην εργασία μας είναι να έχει όσο το δυνατόν γενικότερη εφαρμογή. Η τοποθέτηση σημείων σε ένα δωμάτιο

προϋποθέτει ότι το ρομπότ θα κινείται μέσα σε αυτό το δωμάτιο και μόνο. Σε οποιοδήποτε άλλο δωμάτιο το ρομπότ θα αναγνωρίζει σαν σημάδια κάτι τελείως διαφορετικό από αυτό που θα θέλαμε εμείς κάτι που κάνει το ρομπότ μας τίποτα περισσότερο από μια μηχανή που κάνει μια συγκεκριμένη δουλεία σε ένα βιομηχανικό περιβάλλον. Επίσης, σε περίπτωση που θα θέλαμε να κάνουμε το πείραμα μας σε εξωτερικό περιβάλλον, η τοποθέτηση σημαδιών δεν είναι πάντα εύκολη υπόθεση.

Αντιθέτως, το GPS θα μπορούσε πολύ εύκολα να χρησιμοποιηθεί για ανοιχτούς χώρους όπως αυλές η δρόμους, όμως σε κλειστό χώρο (δωμάτιο σπίτι) το σήμα δεν είναι αρκετά δυνατό ώστε να έχουμε εντοπισμό θέσης. Και για εμάς τουλάχιστον, το πείραμα εκτελείτε μέσα σε ένα κλειστό δωμάτιο.

Featured Based Vs Appearance based

Η σύγκριση εικόνων και η εξαγωγή αποτελεσμάτων μπορει να γίνει με δυο τρόπους: με βάση τα χαρακτηριστικά και με βάση την εμφάνιση της εικονας.

Η κύρια διαφορά τους είναι ότι στα προγράμματα που βασίζονται στα χαρακτηριστικα μιας φωτογραφίας ψάχνουν για γράμμες, κύκλους η οποιαδήποτε άλλο τροπο να εξάγουν πληροφορίες και να παίρνουν αποφάσεις.Αντίθετα, τα προγραμματα που βασίζονται στην εμφάνιση μιας φωτογραφίας κοιτάζουν γενικότερα την φωτογραφία σαν συνολο και όχι σαν άθροισμα αντικειμένων του χώρου.

Τα feature based συστήματα ψάχνουν για περισσοτερη λεπτομερεια, κατι που σε εφαρμογες σαν την δικη μας είναι δυσκολο να την επεξεργαστεις. Επίσης, η χρηση τους βασίζονται σε συγκεκριμένα πραγματα μεσα σε ένα χωρο. Κάτι τετοιο καθιστά την ερευνα μας πολύ περιορισμενη (μεσα σε ένα δωματιο ουσιαστικα) και εμεις όπως εχουμε αναφέρει θελαμε το ρομποτ μας να μπορει να πλοηγηθει μεσα σε ένα οποιοδήποτε δωμάτιο. Επίσης, με τον πρώτο τρόπο υπάρχει μεγαλυτερη πιθανοτητα λαθους καθως από διαφορετικές γωνιες είναι πολύ δυσκολο να ξεδιακρινει καποιος σχηματα μεταξυ τους. Επειδη ακριβως θέλαμε να βασιστούμε στην αξιοπιστία του προγραμματος, επίλεξαμε να χρησιμοποιήσουμε την εμφάνιση

μιας εικόνας για να δουλέψουμε- και τα αποτελέσματα μας δεν ήταν καθόλου αποθαρρυντικά.

Έτσι, έχοντας πλέον μια ιδέα πως θα γίνει η ερευνά μας, προχώρημα στο επόμενο κεφάλαιο.

ΚΕΦΑΛΑΙΟ 3

Θεωριτικη Προσεγγιση της εργασιας

Η διαδικασία που θα ακολουθήσουμε

Περνώντας στο θεωρητικό μέρος της εργασίας μας, να τονίσουμε ότι όπως σε κάθε ερευνά που γινεται σε ένα τομέα, μετά από την θεωρητική ανάλυση του κάθε προβλήματος έρχεται η εφαρμογή της θεωρίας στην πράξη. Εκεί ακριβώς γίνονται οι συγκρίσεις με την θεωρία και βγάζουμε συμπεράσματα για παράγοντες που δεν είχαμε υπολογίσει η δεν είχαμε δώσει την πρέπουσα σημασία.

Η χαρτογράφηση του χώρου θα πραγματοποιηθεί στα έξης βήματα:

1. Το ρομπότ αφήνεται ελεύθερο να κινηθεί μέσα στον χώρο ταυτόχρονα τραβώντας φωτογραφίες. Έτσι, χτίζου μια βάση δεδομένων με φωτογραφίες.
2. Γινεται επεξεργασία αυτών των φωτογραφιών έτσι ώστε να έχουμε τα ιστογράμματα που χρειαζόμαστε.
3. Ανίχνευση θέσης ρομπότ.
4. Υπολογισμός πλοήγησης, πως θα μετακινηθούμε από το ένα σημείο στο άλωσε αυτή την εργασία δεν θα ασχοληθούμε με αυτό το βήμα.

Τώρα, ας τα δούμε αναλυτικά.

ΒΗΜΑ $1^{ο}$:

Το ρομπότ αφήνεται ελεύθερο μέσα στο δωμάτιο μας, σε ένα τυχαίο σημείο. Μέσω του κεντρικού «τούβλου» της Lego δίνεται κίνηση στους τροχούς και το ρομπότ αρχίζει να κινητέ μέσα στο χώρο. Ταυτόχρονα, η φωτογραφική μηχανή κάθε ένα χρονικό διάστημα τρίβει φωτογραφίες το ημισφαίριο που είναι τοποθετημένο στο ρομπότ μας. Μετά από κάποιο χρόνο, όσο εμείς θέλουμε, σταματάμε την χαρτογράφηση και περνάμε στην πλοήγηση.

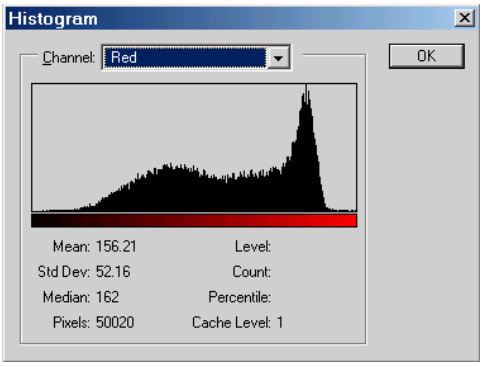


Δωμάτιο χαρτογραφημένο από αυτοκινούμενο ρομπότ

ΒΗΜΑ 2^{ο}:

Αυτό το βήμα γινεται ταυτόχρονα με το βήμα 1. Μόλις τραβηχτεί μια φωτογραφία γινεται η ανάλυση της φωτογραφίας σε ιστογράμματα. Τα ιστογράμματα που μας ενδιαφέρουν στην δική μας περίπτωση είναι 6 : τα τρία ιστογράμματα του χρωματικού πρότυπου RGB(red,green,blue, τα οποία είναι τα βασικά χρώματα) καθώς και τα τρία του πρότυπου HSV (hue, saturation, value η αλλιώς απόχρωση, κορεσμός και φωτεινότητα).

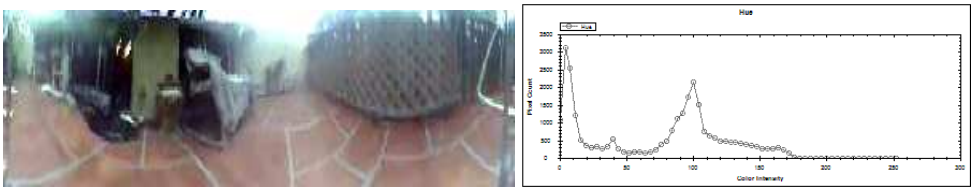
Σε κάθε ένα από αυτά τα ιστογράμματα βρίσκεται ένα κομμάτι πληροφορίας : στο κόκκινο, πράσινο και μπλε αθροίζεται ο αριθμός των πήξε που έχουν αυτή την τιμή. Για παράδειγμα, αν φωτογραφίζαμε ένα μήλο τα κόκκινα πήξε θα ήταν πολύ περισσότερα από του μπλε.



Το ιστόγραμμα του κόκκινου καναλιού μιας φωτογραφίας

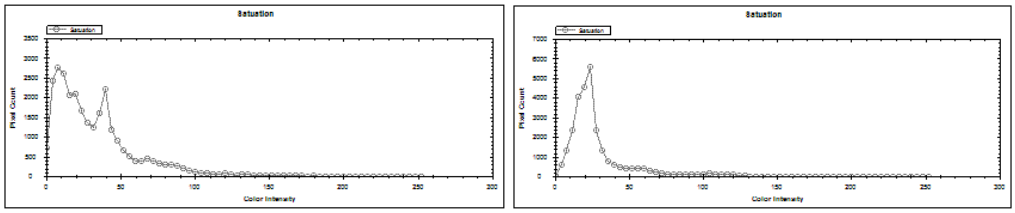
Το hue (απόχρωση) είναι περίπου πως υπολογίζουμε ποιο βασικό χρώμα είναι ένα αντικείμενο. Οι τιμές κοντά στο 0 του ιστογράμματος αντιστοιχούν στο κόκκινο χρώμα ενώ οι τιμές προς το δεξί άκρο

αντιστοιχούν στο μπλε χρώμα. Στην φωτογραφία που φαίνεται από κάτω, αλλά και στο ιστόγραμμα απόχρωσης βλέπουμε ότι υπάρχει πλειονότητα κόκκινου χρώματος.



Μια φωτογραφία και δίπλα το ιστόγραμμα της απόχρωσης. Όπως βλέπουμε και στα δυο, υπάρχει πλειονότητα κόκκινου χρώματος.

Το saturation(διάβρωση) μας δίνει την πληροφορία του πόσο ανοιχτόχρωμο εμφανίζεται το χρώμα στην φωτογραφία. Για παράδειγμα, αν είχαμε τα δυο ιστογράμματα που εμφανίζονται παρακάτω, μπορούμε με ευκολία να διακρίνουμε ότι το αριστερό ιστόγραμμα εμφανίζει ότι τα χρώματα της εικόνας είναι περισσότερο σκουρόχρωμα από την δεξιά.

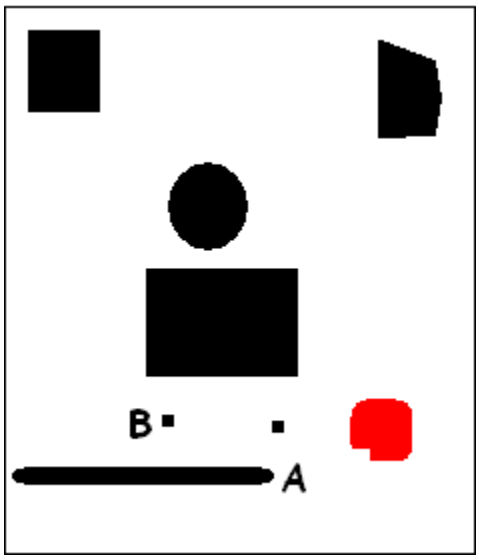


Το value, πολλές φόρες γνωστό και σαν lightness, προσδιορίζει την φωτεινότητα της φωτογραφίας. Μια σκούρα φωτογραφία έχει περισσότερες τιμές κοντά στο 0.

<u>Γιατί όμως χρησιμοποιούμε τόσα πολλά ιστογράμματα?</u>

Αυτό το επιλέξαμε για τον έξης λογο :

Οι αποστάσεις που διανύει το ρομπότ ανάμεσα στις φωτογραφίες που θα τραβήξει πολλές φόρες απέχουν πολύ λίγο από την μια στην άλλη. Ίσως και μερικά εκατοστά. Αυτό σημαίνει ότι πολύ εύκολα το ρομπότ να μπερδευτεί και να αποφασίσει ότι βρίσκεται σε λάθος σημείο μέσα στο δωμάτιο. Κάτι τέτοιο μπορεί να οδηγήσει σε μικρά αλλά συνεχόμενα και αθροιζόμενα σφάλματα και στο τέλος θα έχουμε χάσει τελείως την χαρτογράφηση που θέλουμε να κάνουμε.



Αν αυτό είναι η κάτοψη ενός δωματίου με το κόκκινο να είναι το ρομπότ μας, μια φωτογραφία στο σημείο Α δεν θα έχει μεγάλες διαφορές από μια φωτογραφία στο σημείο Β

Όπως βλέπετε, δυο τέτοιες φωτογραφίες θα παρουσίαζαν περίπου το ίδιο ιστογραμμα . Για να αντιμετωπίσουμε τέτοια προβλήματα, χρησιμοποιούμε διάφορα ιστογράμματα τα οποία με την σειρά τους παρέχουν διαφορετική πληροφορία. Συνδυάζοντας αυτές τις πληροφορίες ελπίζουμε ότι θα έχουμε καλυτέρα αποτελέσματα μιας και αν πχ τα δυο κόκκινα ιστογράμματα μοιάζουν πολύ μεταξύ τους, μπορεί τα ιστογράμματα διάβρωσης να μας βοηθήσουν να ξεχωρίσουμε τις εικόνες που απεικονίζουν!

Ας προχωρήσουμε στο επόμενο βήμα!

ΒΗΜΑ $3^{ο}$:

Τελειώνοντας το $2^{ο}$ βήμα, έχουμε στην διάθεση μας μια βάση δεδομένων που αποτελείτε από ιστογράμματα εικόνων. Τώρα, θα τοποθετήσουμε το ρομπότ σε μια τυχαία θέση στο δωμάτιο και θα το αφήσουμε να τραβήξει μια φωτογραφία. Σκοπός μας είναι να καταφέρει το πρόγραμμα που οδηγεί το ρομπότ να ξεχωρίσει σε ποια από όλες τις φωτογραφίες που είχε τραβήξει κατά την διάρκεια της χαρτογράφησης βρίσκεται. Έτσι, έρχεται στο προσκήνιο η σύγκριση ιστογραμμάτων.

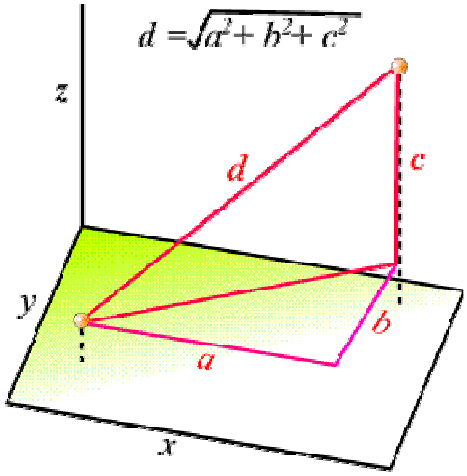
Μετρικές

Οι μετρικές στα μαθηματικά, η αλλιώς συναρτήσεις απόστασης, είναι συναρτήσεις που προσδιορίζουν τις αποστάσεις μεταξύ στοιχειων μιας ομάδας. Επίσης, μετρικές χρησιμοποιούνται για αποστάσεις μεταξυ πινάκων. Γι'αυτό ακριβώς τον λόγο υπάρχων πάρα πολλών ειδών μετρικές, από τις πιο περίπλοκες μεχρι τις πιο απλές.

Εμείς, όπως θα δούμε παρακάτω, χρησιμοποιήσαμε στην πρώτη φάση του πειράματος την Earth Movers Distance (EMD,Απόσταση των Περιηγητών στην Γη) και στην συνέχει την μέθοδο

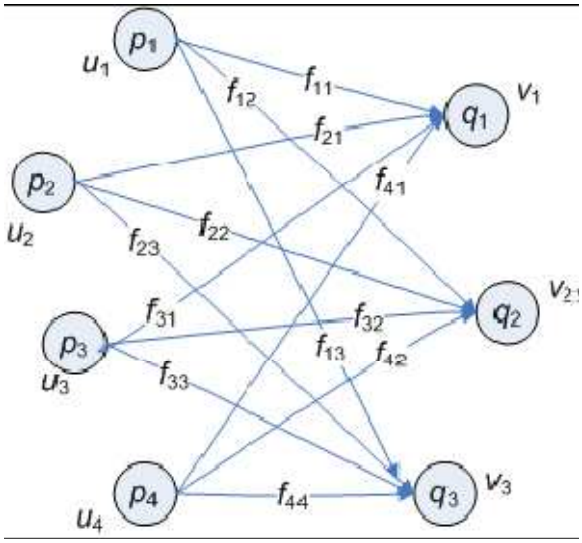
Ας δουμε μερικές

i. Η πιο διαδεδομένη είναι η **Ευκλείδεια απόσταση**, η οποία βασίζεται στα μαθηματικά. Ουσιαστικά, η Ευκλείδεια απόσταση είναι ο «κλασσικός» τρόπος που ο άνθρωπος μέτριοι αποστάσεις μεταξύ δυο σημείων με ένα χάρακα. Βασίζεται στον Πυθαγόρειο μαθηματικό τύπο μετατρέποντας έτσι τον Πυθαγόρειο χώρο σε μετρικό χώρο.



Η Ευκλείδεια απόσταση

ii. Το EMD στους νομούς των πιθανοτήτων, είναι η μέτρηση απόστασης μεταξύ δυο κατανομών πιθανοτήτων σε μια περιοχή Δ. Σε απλά Ελληνικά, υποθέστε ότι έχουμε δυο σωρούς από χώμα, ίδιας περιεκτικότητας αλλά με διαφορετικό τρόπο στοιβάσματος. Το EMD μέτριοι το ελάχιστο κόστος της μετατροπής του ενός σωρού στο άλλο, με κόστος να σημαίνει την μεταφορά χώματος από το ένα στο άλλο πολλαπλασιασμένο επί την απόσταση που χρειάζεται για να μετακινηθεί.



EMD μεταξύ δυο υπογραφών, (m = 4, n = 3) σαν πρόβλημα μεταφοράς

iii. Επισης μια χρησιμη μετρικη είναι η μέθοδος Bhattacharyya. Στην στατιστική, η απόσταση Bhattacharyya μετράει τον ομοιότητα δύο διακριτών η συνεχόμενων κατανομών πιθανοτητων.

Οι δύο τελευταίες μετρικές είναι και αυτές που χρησιμοποιήσαμε και στην εργασία μας, καθως είναι παρα πολύ αποτελεσμάτικες για σύγκριση ιστογραμμάτων.

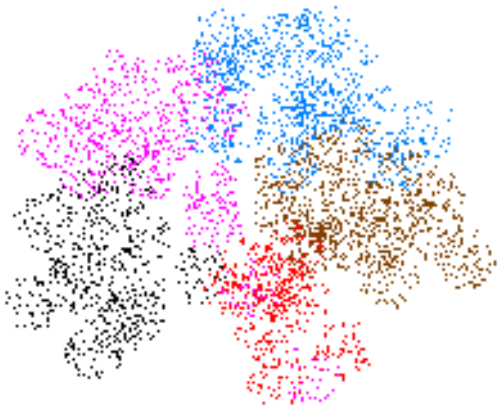
Κατηγοριοποίηση εικόνων

Για να κάνουμε λίγο πιο κατανοητό το επόμενο βήμα, ας μεγαλώσουμε λίγο την κλίμακα πάνω στην οποία δουλεύουμε.

Ας υποθέσουμε ότι έχουμε ένα σπίτι με 5 δωμάτια, και θέλουμε να μάθουμε να ξεχωρίζουμε τα δωμάτια βάση των ιστογραμμάτων κάποιων εικόνων που έχουμε από αυτά. Έχοντας λοιπόν την βάση δεδομένων μας χρειαζόμαστε κάποιον τρόπο να κατηγοριοποιήσουμε τις επόμενες φωτογραφίες σε διαφορετικές ομάδες ανάλογα με το δωμάτιο με το οποίο τραβηχτήκαν.

Αν φανταστούμε τον εαυτό μας να το κάνει αυτό, θα έπρεπε να μπορεί να κοιτάξει ένα ιστόγραμμα και να καταλάβει σε σύγκριση με την βάση δεδομένων μας σε ποιο δωμάτιο βρίσκεται ανάλογα με κάποια χαρακτηριστικά του ιστογράμματος τα οποία ο ίδιος προσπαθεί να διακρίνει. Κάτι τέτοιο φυσικά δεν είναι εύκολο, καθώς πρέπει να βρεις ομοιότητες και διαφορές σε κάτι που δεν είναι ιδιαίτερα ξεκάθαρο. Όμως με λίγη προσπάθεια ίσως καταφέρουμε να έχουμε καλυτέρα αποτελέσματα.

Λοιπόν, πως θα μπορούσαμε να διακρίνει ένα υπολογιστής τέτοιες διαφορές? Ευτυχώς για μας πολλοί άνθρωποι δουλεύουν εδώ και δέκατες πάνω σε αυτό το πρόβλημα και μας έχουν δώσει αρκετές ικανοποιητικές λύσεις, ανάλογα φυσικά με τον τύπο των δεδομένων που έχουμε. Το γενικότερο πρόβλημα είναι στην αντιστοίχηση ενός σετ δεδομένων που χρησιμοποιούμε σαν είσοδο σε ένα σετ δεδομένων που ορίζεται σαν έξοδος.



Αν κάθε σημείο θα αντιστοιχούσε σε μια εικόνα, μια σωστή κατηγοριοποίηση θα κατάφερνε να ξεχωρίσει όσο το δυνατόν καλυτέρα τα χρώματα μεταξύ τους

Στην δική μας περίπτωση, το σετ εισόδων είναι ο αριθμός που έχει μετρήσει ένα ιστόγραμμα στο κάθε κανάλι και το σετ εξόδων είναι ένας άσσος στο δωμάτιο που βρισκόμαστε και ένα μηδενικό στις φωτογραφίες που αντιστοιχούν στα δωμάτια τα οποία δεν βρισκόμαστε. Μαθηματικά , τα δεδομένα εισόδου και εξόδου αντιπροσωπεύονται με πρότυπα και οι σχέσεις μεταξύ τους ως πολυδιάστατοι πίνακες. Σε εμάς, τα πρότυπα εισόδου έχουν 256 χαρακτηριστικές τιμές (μια για κάθε τιμή που μπορεί να πάρει ένα πήξε σε κάθε κανάλι) ενώ τα δεδομένα εξόδου έχουν 5 χαρακτηριστικές τιμές, μια για κάθε δωμάτιο.

Παρακάτω αναφέρονται τρεις τρόποι κατηγοριοποίησης:

a) Κατηγοριοποίηση με πρότυπα,
b) με τον «κοντινότερο γείτονα» και τέλος,
c) με νευρωνικά δίκτυα.

Κατηγοριοποίηση βάση «κοντινότερου γείτονα»

Ο πρώτος τρόπος κατηγοριοποίησης είναι αυτή με τον «κοντινότερο γείτονα». Όπως είναι προφανές και το όνομα της, κάθε φωτογραφία αντιστοιχίζεται σε αυτή που το σύστημα θεώρει ότι βρίσκεται πιο κοντά.

Η ιδέα είναι αρκετά απλή : Μετά από την φάση της χαρτογράφησης και της τοποθέτησης του ρομπότ σε μια τυχαία θέση μέσα στο δωμάτιο, το ρομπότ τρίβει μια φωτογραφία, την οποία θα ονομάζουμε φωτογραφία δόκιμης.

Έχοντας στον υπολογιστή μας μια βάση δεδομένων με τις εικόνες του δωματίου, αρχίζει να δουλεύει ένας αλγόριθμος οποίος δουλεύει ως έξης:

- ☒ Ανάλυση της φωτογραφίας δόκιμης στα 6 κανάλια που αναφέραμε προηγουμένως.
- ☒ Υπολογισμό ιστογραμμάτων για κάθε τέτοιο κανάλι.
- ☒ Υπολογισμό διαφοράς (με όποιον τρόπο επιλέξει κάποιος) μεταξύ του κάθε ιστογράμματος της φωτογραφίας δόκιμης με το αντίστοιχο ιστόγραμμα που είναι αποθηκευμένο στην βάση δεδομένων μας, για κάθε φωτογραφία ξεχωριστά.
- ☒ Αποθήκευση αυτής της διαφοράς σε ένα πινάκα, ο οποίος στην συνέχεια χρησιμοποιείται για την εύρεση της ελαχίστης διαφοράς ανάμεσα στις φωτογραφίες. Με αυτόν τον τρόπο ουσιαστικά βρίσκουμε σε ποια φωτογραφία πιο κοντά βρίσκεται η φωτογραφία δόκιμης μας.
- ☒ Κάθε ιστόγραμμα ψηφίζει με βάση το παραπάνω βήμα σε ποια φωτογραφία ταιριάζει πιο πολύ, με περισσότερους ψήφους να παίρνει η μικρότερη απόσταση. Οι υπόλοιπες φωτογραφίες παίρνουν λιγότερους ψήφους ανάλογα πάντα με την διάφορα τους από την φωτογραφία δόκιμης
- ☒ Εύρεση του μεγίστου αριθμού ψήφων και στη συνέχεια αντιστοίχηση της φωτογραφίας δόκιμης και της φωτογραφίας που έχει τους περισσότερους ψήφους.
- ☒ Απόδοση νικήτριας φωτογραφίας την ετικέτα του σημείου που βρίσκεται το ρομπότ.

Κατηγοριοποίηση με πρότυπα

Ο πιο εύκολος , πιο εύκολα κατανοητός και πιο εύκολα προγραμματισιμος είναι ο τρόπος της κατηγοριοποίησης κατά πρότυπα.

Συμφώνα με αυτόν τον τρόπο, κατά την διάρκεια της χαρτογράφησης σε κάθε σημείο λαμβάνονται περισσότερες από μια φωτογραφίες. Μάλιστα, αν αυτές οι φωτογραφίες είναι και σε μια μικρή ακτίνα γύρω από το σημείο ενδιαφέροντος αυτό είναι ακόμα καλύτερο.

Στην συνέχεια, όποιες φωτογραφίες αφορούν ένα σημείο μέσα στο δωμάτιο αθροίζονται τα ιστογράμματα τους και βγαίνει ένας μέσος ορός. Αυτό το ιστόγραμμα ουσιαστικά είναι το πρότυπο που ανήκει στο συγκεκριμένο σημείο.

Μόλις περάσουμε στην φάση του εντοπισμού θέσης, κάθε φωτογραφία θα συγκρίνεται με το πρότυπο κάθε σημείου, μειώνοντας έτσι τον αριθμό των συγκρίσεων σε σχέση με την τεχνική του «κοντινότερου γείτονα».

Βήμα προς βήμα ο αλγόριθμος δουλεύει ως έξης:

- Κατά την διάρκεια της χαρτογράφησης, στην βάση δεδομένων μας αποθηκεύουμε περισσότερες φωτογραφίες από ότι στην τεχνική με τον κοντινότερο γειτονα.Αυτο συμβαίνει γιατί ενώ πριν είχαμε μια φωτογραφία για κάθε σημείο, τώρα έχουμε δυο, τρεις η και παραπάνω για ένα σημείο.
- Η βάση δεδομένων μας επεξεργάζεται αθροίζοντας και βγάζοντας τον μέσο όρο για τις φωτογραφίες που αντιστοιχούν σε ένα σημείο του δωματίου.
- Το ρομπότ μας παίρνει βρίσκεται σε μια τυχαία θέση στο δωματίου, όπου και τρίβει μια φωτογραφία

- Διαχωρισμός της εικόνας δόκιμης στα έξι κανάλια που χρησιμοποιούμε.
- Υπολογισμός ιστογραμμάτων εικόνας δόκιμης
- Υπολογισμός διαφοράς ιστογραμμάτων εικόνας δόκιμης και αποθηκευμένων προτύπων.
- Εύρεση της μικρότερης διαφοράς ανάμεσα στα ιστογράμματα
- Ψηφοφορία κάθε ιστογράμματος με περισσότερους ψήφους να παίρνει η φωτογραφία με την μικρότερη διάφορα
- Εύρεση νικήτριας φωτογραφίας, η οποία παίρνει και ετικέτα με το όνομα του σημείου στο οποίο αντιστοιχήθηκε.

Αυτός ο τρόπος, είναι λιγότερο πολύπλοκος και ταχύτερος με αυτόν που αναλύσαμε παραπάνω.

Ο επόμενος τρόπος, είναι ο πιο αποδοτικός, ο πιο πολύπλοκος και ο πιο εντυπωσιακός. Μιλάμε για τα νευρωνικά δίκτυα, τα οποία βασίζονται σε μια προσπάθεια της προσομοίωσης των λειτουργιών του εγκέφαλου με την βοήθεια των μαθηματικών και της πληροφορικής. Λογού της μεγάλης προοπτικής που έχουν τα νευρωνικά δίκτυα θα αναφερθούμε λεπτομερώς σε αυτά.

ΤΕΧΝΗΤΑ ΝΕΥΡΩΝΙΚΑ ΔΙΚΤΥΑ

Παραδοσιακά, ο ορός νευρωνικό δίκτυο χρησιμοποιούνταν για αναφορά ένα δίκτυο η κύκλωμα βιολογικών νευρώνων. Η μοντέρνα χρήση του ορού συχνά αναφέρεται στα τεχνητά νευρωνικά δίκτυα, τα οποία αποτελούνται από τεχνητούς νευρώνες.

Αρά, ο ορός έχει δυο διαφορετικές χρήσεις:

- Βιολογικά νευρωνικά δίκτυα τα οποία αποτελούνται από πραγματικούς βιολογικούς νευρώνες οι οποίοι συνδέονται η είναι σχετικοί με την λειτουργιά του περιφερειακού η κεντρικού

νευρικού συστήματος. Στο πεδίο της νευρολογίας, συχνά αναγνωρίζονται ως ομάδες νευρώνων που πραγματοποιούν μια συγκεκριμένη λειτουργιά σε εργαστηριακή ανάλυση.

- Σε τεχνητά νευρωνικά δίκτυα, τα οποία αποτελούνται από τεχνητούς νευρώνες (προγραμματιστικά κατασκευάσματα τα οποία μιμούνται τις ιδιότητες των βιολογικών νευρώνων) οι οποίοι είναι διασυνδεμένοι μεταξύ τους. Τα τεχνητά νευρωνικά δίκτυα χρησιμοποιούνται είτε για να βοηθήσουν στην κατανόηση των βιολογικών νευρωνικών δικτύων, η για την λύση ενός προβλήματος τεχνητής νοημοσύνης χωρίς την δημιουργία ενός πραγματικού βιολογικού μοντέλου. Τα πραγματικά βιολογικά μοντέλα είναι ιδιαίτερα πολύπλοκα και περιλαμβάνουν μερικές ιδιότητες που πολλές φόρες είναι αχρείαστες βασιζόμενοι στην κατανόηση των τεχνητών νευρωνικών δικτύων.

Επισκόπηση

Γενικά, ένα βιολογικό νευρωνικό δίκτυο αποτελείται από μια ομάδα η μια ομάδα ομάδων νευρώνων, συνδεμένων χημικά η λειτουργικά μεταξύ τους. Ένας νευρώνας μπορεί να είναι συνδεμένος με πολλούς άλλους νευρώνες και ο συνολικός αριθμός νευρώνων και συνδέσεων μπορεί να είναι πολύ μεγάλος.

Έκτος από τα ηλεκτρικά σήματα, υπάρχουν και άλλες μορφές σήματος που εμφανίζονται σε νευρώνες-πομπούς που μοιάζουν σε ηλεκτρικά σήματα. Έτσι, καταλαβαίνουμε ότι τα νευρωνικά δίκτυα είναι εξαιρετικά πολύπλοκα.

Η τεχνητή νοημοσύνη προσπαθεί να εξομοιώσει τις ιδιότητες των νευρωνικών δικτύων. Στόχος της είναι η επίλυση προβλημάτων, ενώ στόχος του μοντελισμού νοημοσύνης είναι η δημιουργία μαθηματικών μοντέλων βιολογικών νευρωνικών δικτύων.

Στο πεδίο της τεχνητής νοημοσύνης, τα τεχνητά νευρωνικά δίκτυα έχουν χρησιμοποιηθεί με μεγάλη επιτυχία σε συστήματα αναγνώρισης φωνής, σε ανάλυση εικόνας καθώς και σε έλεγχο προσαρμογής με σκοπό την δημιουργία προγραμμάτων (σε υπολογιστές και ηλεκτρονικά παιχνίδια) η και ακόμα στην δημιουργία αυτόνομων ρομπότ. Στην πλειονότητα τους τα νευρωνικά βασίζονται στην στατιστική, στην βελτιστοποίηση και στην θεωρία του έλεγχου.

Στο πεδίο του μοντελισμού νοημοσύνης περιλαμβάνεται η δημιουργία φυσικών η και μαθηματικών μοντέλων της συμπεριφοράς των νευρωνικών δικτύων. Το βεληνεκές τους μπορεί να είναι σε ένα μοναδικό νευρωνικό επίπεδο η και σε clusters (ομάδες) νευρώνων η ακόμα και σε ολοκληρωμένους οργανισμούς.

Η τεχνητή νοημοσύνη και ο μοντελισμός νοημοσύνης είναι και τα δυο πεδία επεξεργασίας πληροφορίας που εμπνευστήκαν από τον τρόπο που επεξεργάζονται τα δεδομένα τα βιολογικά νευρωνικά συστήματα.

Ιστορία των νευρωνικών δικτύων

Το πρώτο μμοντέλο νευρωνικού δικτύου το οποίο προτείνει ότι οι νευρώνες είναι η βασική μμονάδα του δικτύου παρουσιάσθηκε το 1943 από τους McCulloch και Pitts.

Σε μία πρώτη εργασία τους οι ερευνητές αυτοί παρουσίασαν για πρώτη φορά την ιδέα ότι ένα νευρωνικό δίκτυο αποτελείται από μία συλλογή ενός μΕγέλου αριού νευρώνων και έδειξαν πώς θα

μπερούσαν να λειτουργούν οι νευρώνες με τις διασυνδέσεις τους. Αυτή θεωρείται ιστορικά ότι είναι η πρώτη εικόνα ενός νευρωνικού δικτύου. Μάλιστα οι συγγραφείς θεώρησαν ότι οι νευρώνες και οι συνδέσεις τους είναι ένα πρότυπο, ανάλογο ενός ηλεκτρικού κυκλώνατε. Ο McCulloch ήταν νευροφυσιολόγος και ο Pitts ένας 18χρονος πρωτοετής φοιτητής των Μαθηματικών.

Οι ίδιοι συγγραφείς προχώρησαν το 1947 σε πιο εξελιγμένο πρότυπο για την αναγνώριση σχημάτων. Το πρότυπο αυτό περιέχει πολλά χαρακτηριστικά από τα μεταγενέστερα πρότυπα. Ο νευρώνας θεωρείται ότι μπορεί να έχει δύο μόνον καταστάσεις.Μπορεί να δέχεται πολλές εισόδους αλλά δίνει μία μόνον έξοδο. Οι έξοδοι από διαφορετικούς νευρώνες δεν επιτρέπεται να ενώνονται, αλλά πρέπει υποχρεωτικά να οδηγούν σε είσοδο άλλου νευρώνα. Οι απολήξεις των νευρώνων είναι δύο ειδών: διεγερτικές και ανασταλτικές. Οι δύο καταστάσεις του νευρώνα είναι ότι είτε πυροδοτεί ή βρίσκεται σε ηρεμία. Η ροή της πληροφορίας μέσα στον νευρώνα ελέγχεται από πύλες, οι οποίες επίσης είναι διεγερτικές ή ανασταλτικές. Όταν ο νευρώνας πυροδοτεί, στέλνει ένα παλμό. Οι λειτουργίες αυτές πάντα γίνονται σε διακριτό χρόνο και υποτίθεται ότι όλοι οι νευρώνες αποκρίνονται ταυτόχρονα, δηλ. το σύστημα δρα συγχρονισμένα. Η κατάσταση ενός νευρώνα σε χρόνο t+ 1 εξαρτάται από την κατάστα-σή του σε χρόνο t και από τις εισόδους που εισέρχονται στην χρονική αυτή στιγμή.

Τα δίκτυα McCulloch–Pitts προσπαθούν να εξηγήσουν για πρώτη φορά πως δουλεύει η μνήμη. Θεωρούν ότι ένας πιθανός μηχανισμός μνήμης μπορεί να είναι ηύπαρξη κλειστών διαδρομών του σήματος μέσα στο δίκτυο. Αν δεν υπάρχει καμία τέτοια διαδρομή και χωρίς νέο εξερχόμενο σήμα, τότε το δίκτυο θα μείνει μόνιμα σεκατάσταση ηρεμίας. Έτσι, μια ίνα ενώνει την έξοδο ενός κυττάρου με το σημείο εισόδου στο ίδιο κύτταρο, δημιουργώντας έναν μηχανισμό ανάδρασης (feedback).Μόλις πυροδοτεί ένα τέτοιο κύτταρο θα συνεχίσει να

πυροδοτεί μέχρι να έλθει σήμα από ανασταλτική ίνα. Καθόλη τη διάρκεια της λειτουργίας αποστέλλονται παλμοί στην πύλη των κυττάρων και μεταδίδεται το σήμα και η πληροφορία.

Αυτός ο κύκλος του σήματος θα μπορούσε να θεωρηθεί ως μία πρώτη ιδέα για ένα μηχανισμό μνήμης.Οι εργασίες αυτές πιθανόν να χάνονταν στην βιβλιογραφία αν δεν τις χρησιμοποιούσε λίγα χρόνια αργότερα ο J. von Neumann ως παράδειγμα για υπολογιστικές μηχανές την δεκαετία που διαδόθηκε ο ηλεκτρονικός υπολογιστής, δηλ.την δεκαετία του πενήντα. Τότε έγιναν και οι πρώτες προσπάθειες να αντλήσου-με πληροφορίες από τα βιολογικά δίκτυα και να δημιουργηθούν τα πρώτα τεχνητά δίκτυα.

Ενα άλλο έργο της πρώτης αυτής εποχής που αφήνει ακόμα και σήμερα την επιρ-ροή του είναι το βιβλίο του D. Hebb , «The organisation of behavior» (1949),το οποίο εισάγει τον κανόνα μάθησης του Hebb. Το μοντέλο του Hebb έχει ως κεντρική ιδέα τις συνδέσεις μεταξύ μονάδων του συστήματος, δηλαδή τους νευρώνες. Έφτασε στα συμπεράσματα αυτά μετά από σωρεία πειραμάτων νευροφυσιολογίας. Ο κανόνας αυτός λέγει ότι κάθε φορά που το δίκτυο χρησιμοποιεί τις νευρωνικές του συνδέσεις, οι συνδέσεις αυτές ενισχύονται και το δίκτυο πλησιάζει περισσότερο στο να μάθει το πρότυπο το οποίο παρουσιάζεται. Όταν ο νευρώνας I επανειλημμένα διεγείρει τον νευρώνα J , τότε συμβαίνει να αναπτύσσεται μια μετα-βολική σύνδεση στον ένα ή και στους δύο νευρώνες, έτσι ώστε η απόδοση του φαινομένου (το i διεγείρει το j) να αυξάνεται. Αν wij είναι το βάρος της σύνδεσης μεταξύ i και j, xi η είσοδος στον νευρώνα j από τον νευρώνα i,xj η έξοδος του νευρώνα j, τότε ισχύει ότι:

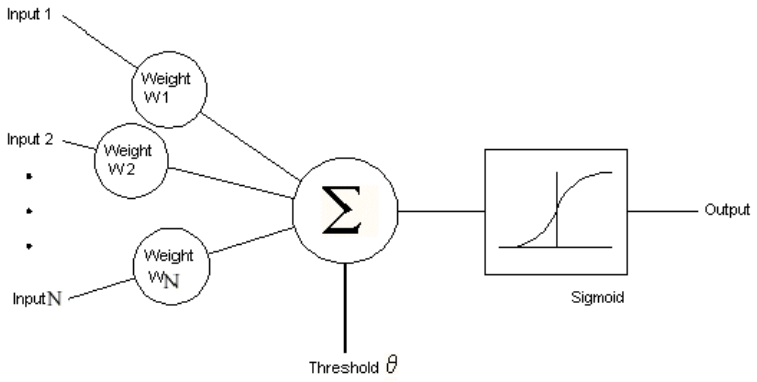
wij(new) =wij(old) + axixj.

Εδώ a είναι μία θετική σταθερά που λέγεται παράμετρος του ρυθμού εκπαίδευσης.Το νέο λοιπόν βάρος wij θα είναι ίσο με το παλαιό ενισχυμένο κατά μία ποσότητα axixj. Ο κανόνας αυτός έχει τοπικό

χαρακτήρα, ισχύει δηλαδή μόνο για την σύνδεσητου νευρώνα i και j και όχι για άλλες συνδέσεις του δικτύου.

Τα πρωτα μοντελα

Το μοντέλο του αισθητήρα (perceptron) παρουσιάσθηκε για πρώτη φορά το 1957από τον F. Rosenblatt , ο οποίος αρχικά έφτιαξε το πρώτο δίκτυο με hardware που μπορούσε να κάνει πολλές και διάφορες διεργασίες. Είναι ένα πολύ απλό που έχει μόνο δύο επίπεδα, αυτά της εισόδου και της εξόδου. Το σήμα προχωρά μονοδρομικά από την είσοδο στην έξοδο.



Μια αναπαρασταση του perceptron

Το μοντέλο αυτό στην αρχή είχε πολλές επιτυχίες, δημιούργησε μεγάλο ενθουσιασμό και μάλιστα ήδη αρχίζει να συζητείται η ιδέα ότι πιθανόν τα νευρωνικά δίκτυα να είναι η ανώτερη τεχνική που λύνει όλα τα προβλήματα που μέχρι τότε παρέμεναν άλυτα. Οι πρώτες λοιπόν επιτυχίες μεγαλοποιήθηκαν, αλλά γρήγορα φάνηκε ότι τα μοντέλα αυτά είχαν πολλούς περιορισμούς.

Μια συνολική και εμπεριστατωμένη εικόνα του προτύπου αυτού παρουσιάσθηκε το 1969 στοβιβλίο «Perceptrons» των Minsky και Papert. Αποδεικνύεται με αναλυτικά μαθηματικά ότι υπάρχουν συγκεκριμένοι περιορισμοί στο πρότυπο αυτό. Έτσι, δεν μπορεί να λύσει,λ.χ. το σχετικά απλό πρόβλημα του X–OR. Οι αρχικές προσδοκίες που είχαν δημιουργηθεί ήδη φαίνεται ότι δεν επαληθεύονται και προς

το παρόν τα νευρωνικά δίκτυα χάνουν την δημοτικότητα τους, με αποτέλεσμα ο κόσμος να στρέφεται σε μιά νέα παρεμφερή περιοχή που τότε άρχισε να γίνεται γνωστή, την Τεχνητή Νοημοσύνη.

Την ίδια περίπου εποχή με την ανάπτυξη του μοντέλου του αισθητήρα οι Widrowκαι Hoff ανέπτυξαν το 1959 δύο νέα μοντέλα, το Adaline και το Madaline, τα οποία όπως ήταν από τα πρώτα μοντέλα που χρησιμοποιήθηκαν επιτυχώς για πρακτικά προβλήματα: Χρησιμοποιήθηκαν ως φίλτρα για να εξαλείψουν την ηχώ σε τηλεφωνικές γραμμές.Τα επόμενα είκοσι χρόνια, μέχρι περίπου το 1980, μικρή μόνο πρόοδος επιτελέσθηκε στα νευρωνικά δίκτυα, διότι οι περιορισμοί που αναφέρθηκαν παραπάνω αποθάρρυναν τους περισσότερους στο πεδίο αυτό, το οποίο, όπως φάνηκε λίγο αργότερα, έψαχνε να βρεί μία διέξοδο και να κάνει νέα σημαντικά βήματα.

Η ωριμανση

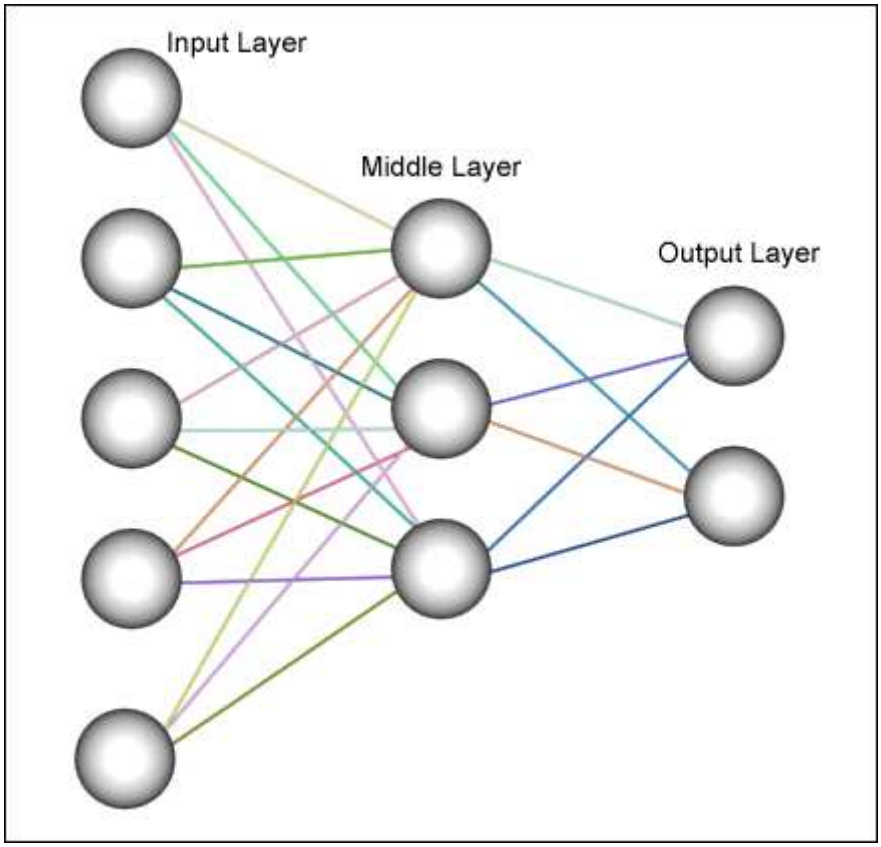
Η διέξοδος αυτή ήρθε με ένα μνημιώδες έργο που παρουσιάσθηκε το 1982 από τον J. Hopfield, ο οποίος είναι βιολόγος, και το οποίο έδωσε μεγάλη ώθηση στην ανάπτυξη των δικτύων. Σε μία εργασία του μόλις 5 σελίδων ο Hopfield έδειξε με αυστηρά μαθηματική απόδειξη πώς ένα νευρωνικό δίκτυο μπορεί να χρησιμοποιηθεί ως αποθηκευτικός χώρος (storage device) και πώς επίσης μπορεί ένα δίκτυο να επανακτήσει όλη την πληροφορία ενός συστήματος έστω και αν του δοθούν μερικά τμήματα μόνο και όχι ολόκληρο το σύστημα.

Αμέσως εκτιμήθηκε η σπουδαιότητα της ιδιότητας αυτής και ως εκ τούτου η εργασία αυτή αποτέλεσε έμπνευση για πολλές άλλες ιδέες που ακολούθησαν.Ενα επόμενο σημαντικό βήμα ήταν η πρόοδος που επιτελέσθηκε στην διαδικασία εκπαίδευσης των δικτύων όταν επινοήθηκε ο κανόνας της διόρθωσης του σφάλματος (error correction learning). Έγινε κατανοητό ότι κατά την εκπαίδευση ενός δικτύου, σε όποια κατάσταση και αν βρίσκεται αυτό σε μιά δεδομένη στιγμή, σημασία έχει η απόκλιση που δίνει στην την έξοδο του το δίκτυο από

την αναμενόμενη τιμή ή τον στόχο που έχουμε θέσει. Η διαφορά αυτή δίνει το σφάλμα που παράγει το δίκτυο την δεδομένη στιγμή n και δίνεται από:δ(n)=t(n)–o(n) όπου «t» είναι ο στόχος, «ο» είναι η έξοδος και «δ» είναι το σφάλμα.

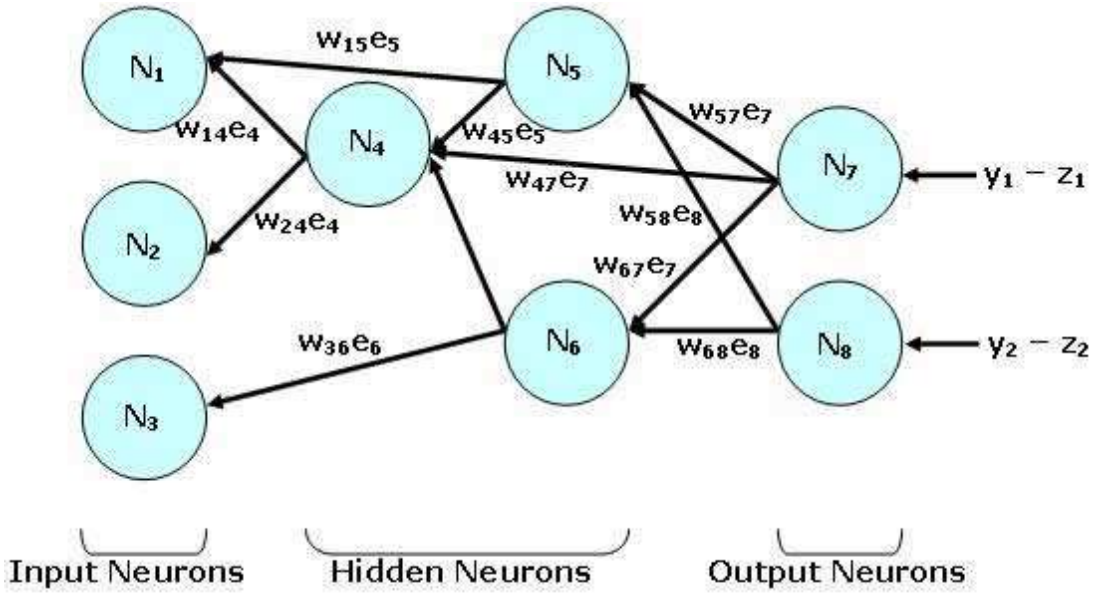
Το δ τώρα ενεργοποιεί ένα μηχανισμό ελέγχου με σκοπό να επιφέρει μιά σειρά από διορθωτικές αλλαγές στα βάρη w, πράγμα που θα φέρει το δίκτυο πλησιέστερα στην εκπαίδευση του. Αυτό γίνεται με τον υπολογισμό της ποσότητας Δ:Δi=ηδxi οπου η είναι μιά σταθερά, και xi είναι η τιμή του σήματος εισόδου.

Τέλος το Δw (n),δηλ. η διόρθωση του βάρους w στην δεδομένη στιγμή n , δίνεται κατ’ ευθείαν από το Δ:Δw(n) =Δi.Ο κανόνας αυτός λέγεται «κανόνας Δ» ή κανόνας «Widrow–Hoff» και χρησιμοποιείται ευρύτατα σε διάφορες μεθόδους εκπαίδευσης δικτύων.Το 1986 δημοσιεύεται ένα άλλο σημαντικό έργο από τους McClelland καιRumelhart, το «Parallel Distributed Processing», το οποίο ανοίγει νέους δρόμους στην εκπαίδευση των νευρωνικών δικτύων. Παρουσιάζεται η ιδέα πώς ένα νευρωνικό δίκτυο μπορεί να θεωρηθεί και να χρησιμοποιηθεί ως παράλληλος επεξεργαστής.



Ενα νευρωνικο δικτυο με παραπανω επιπεδα απο οτι το perceptron

Το έργο αυτό κάνει ένα σημαντικό βήμα πέρα από το Perceptron, με το να επιτρέπει την ύπαρξη και άλλων επιπέδων νευρώνων, εκτός από την είσοδο και την έξοδο, που αποτελούν την εσωτερική δομή του δικτύου. Προτείνουν μία νέα διαδικασία εκπαίδευσης, την μέθοδο της οπισθοδιάδοσης (back–propagation), η οποία χρησιμοποιεί τις εξισώσεις που αναφέρθηκαν παραπανω και κατέληξε να είναι η πιο χρήσιμη σήμερα τεχνική εκπαίδευσης δικτύων. Η μέθοδος αυτή είχε συζητηθεί και από άλλους νωρίτερα, αλλά για πρώτη φορά το 1986 παρουσιάσθηκε ολοκληρωμένα και με αυστηρό μαθηματικό τρόπο.



Σχηματίκη αναπαράσταση ενος back- propagation νευρωνικού δικτύου

Η κατασταση σημερα

Μετά την πρόοδο σε τόσα πολλά σημεία που παρουσιάσθηκε ιδιαίτερα την δεκαετία του 1980, τα τελευταία δέκα χρόνια παρατηρούμε ότι αρχίζουν να εμφανίζονται πολλά σημεία που δείχνουν ότι η περιοχή των νευρωνικών δικτύων έχει πλέον ωριμάσει και αναπτυχθεί σε ένα ανεξάρτητο πεδίο της επιστήμης με δικά του στοιχεία,δικό του χαρακτήρα σαφώς καθορισμένο και τέλος με μεγάλο αριθμό επιστημόνων που ασχολούνται αποκλειστικά τώρα με την νέα αυτή περιοχή.

Τα στοιχεία αυτά είναι:Από το 1985 και μετά αρχίζουν τα πρώτα συνέδρια που είναι αφιερωμένα αποκλειστικά σε νευρωνικά δίκτυα, από την American Physical Society και από την IEEE.Παρακολουθούνται από περισσότερους από χίλιους συνέδρους. Ταυτόχρονα δημιουργούνται ειδικές επαγγελματικές εταιρίες νευρωνικών δικτύων με χιλιάδες μέλη,όπως η International Neural Network Society με τρεις πόλους: Αμερική (με διευθυ-ντή τον Grossberg), Ευρώπη (Kohonen) και Ιαπωνία (Amari). Προς τα τέλη της δεκαετίας του ογδόντα παρουσιάζονται τουλάχιστον πέντε νέα

περιοδικά αφιερωμένα αποκλειστικά στα νευρωνικά δίκτυα, ενώ πριν λίγα χρόνια δεν υπήρχε ούτε ένα.

Τα τελευταία χρόνια μετά το 1990 εκδίδονται και άλλα 3–4νέα, με συνέπεια να υπάρχουν σήμερα περίπου 10 επιστημονικά περιοδικά αφιερωμένα στα νευρωνικά δίκτυα. Φυσικά, και τα γνωστά περιοδικά της Επιστήμες Υπο-λογιστών, της Φυσικής και των Ηλεκτρολόγων Μηχανικών επίσης περιλαμβάνουν πλειάδα άρθρων με νέα αποτελέσματα στα νευρωνικά δίκτυα. Κάθε μήνα πλέον δημοσιεύονται εκατοντάδες εργασίες με αποκλειστικό θέμα κάποια άποψη των νευρωνικών δικτύων. Μερικά από τα εξειδικευμένα νέα περιδικά είναι:

• Neural Networks: The Official Journal of the International Neural NetworkSociety (Pergamon Press).

• Network: Computation in Neural Systems (Institute of Physics Publishing).

• International Journal of Neural Systems (World Scientific).

• Neural Computation.

• Connection Science: Journal of Neural Computing, Artificial Intelligence and Cognitive Research (Carfax Publishing)

.• Neural Network World: Neural and Massively Parallel Computing and Information Systems (Computer World, Prague).

Πολύ σημαντικό είναι επίσης το γεγονός ότι τα τελευταία δεκαπέντε χρόνια δημιουργήθηκαν και οι πρώτες εμπορικές εταιρίες οι οποίες ασχολούνται αποκλειστικάμε νευρωνικά δίκτυα. Βρίσκονται σχεδόν όλες στις ΗΠΑ, συνήθως έχουν μικρό αριθ-μό εργαζομένων (λ.χ. 20 άτομα), και παράγουν εξειδικευμένα προγράμματα για την λύση συγκεκριμένων προβλημάτων. Μερικά από αυτά έχουν επιτυχία, αλλά διαφαίνεται με το πέρασμα του χρόνου ότι οι αρχικές προσδοκίες για

ραγδαία αύξηση των εμπορικών εφαρμογών δεν επαληθεύονται. Έχουν φθάσει πλέον σε ένα σταθερό επίπεδο ανάπτυξης, ενώ η ετήσια αύξηση είναι μικρή.

Μετά από αυτή την αναφορά στα νευρωνικά δίκτυα ας επιστρέψουμε πίσω όπου είχαμε μείνει.

Το νευρωνικό δίκτυο θα μπορούσε να χρησιμοποιηθεί στην εργασία μας για να εκπαιδεύσουμε το δίκτυο να ξεχωρίζει της φωτογραφίες μεταξύ τους διαδικασία θα είχε ως έξης:

- Εκπαιδεύουμε το νευρωνικό μας διαλέγοντας σαν εισόδους τις τιμές των ιστογραμμάτων για κάθε κανάλι και θέτοντας σαν έξοδο (θα είχε τόσες εξόδους όσα τα σημεία που έχουμε) 1 στο σημείο του δωματίου που αντιστοιχεί και 0 σε όλα τα υπόλοιπα.
- Όταν τελειώσει η εκπαίδευση του έχουμε την δυνατότητα να βάλουμε το ρομπότ μας σε οποιαδήποτε σημείο του δωμάτιο και το νευρωνικό έχει εκπαιδευτεί να ξεχωρίσει αυτά τα σημεία.

Η μέθοδος κατηγοριοποίησης με νευρωνικά είναι ο πιο αποδοτική αλλά ταυτόχρονα η πιο χρονοβόρα(τουλάχιστον κατά την διάρκεια εκπαίδευσης του δικτύου) και περιπλοκή.

Έχοντας αναλύσει θεωρητικά την εργασία μας περνάμε στην εφαρμογή της θεωρίας στην πράξη.

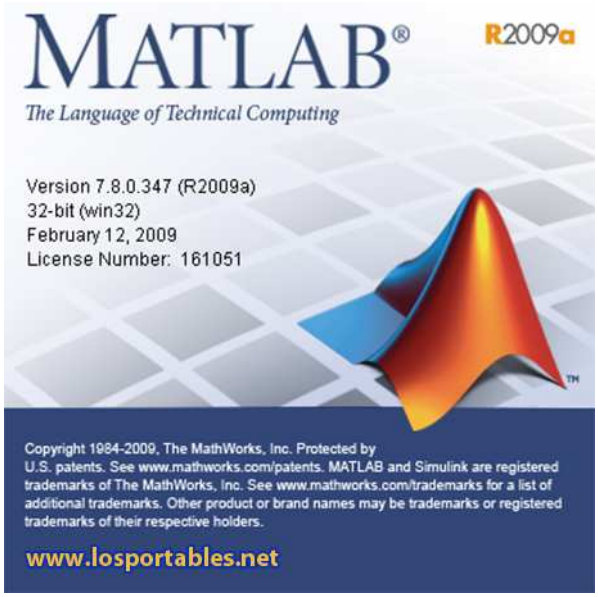
ΚΕΦΑΛΑΙΟ 4
ΕΦΑΡΜΟΓΗ ΣΕ MATLAB ΚΑΙ OPENCV

ΕΙΣΑΓΩΓΗ

Περνώντας στην πρακτική εφαρμογή της θεωρίας, θα αναφερθούμε στα προγράμματα που χρησιμοποιήσαμε για να φέρουμε σε πέρας την εργασία μας.

Το πείραμα τελικά διεξήχθητε σε δυο φάσεις.

1) Μια στο Mat lab, ένα πρόγραμμα που χρησιμοποιείται κατά κύριο λόγο για την επίλυση μαθηματικών προβλημάτων, ωστόσο είναι πολύ "ισχυρό" και μπορεί να χρησιμοποιηθεί και για προγραμματισμό.



Το Mat lab είναι ένα παρά πολύ ισχυρό πρόγραμμα για την επίλυση μαθηματικών προβλημάτων αλλά και χειρισμό πινάκων.

2) Μετά από την αντιμετώπιση κάποιων προβλημάτων με το Mat lab η εργασία μας μεταφέρθηκε σε περιβάλλον .NET. Συγκεκριμένα χρησιμοποιήσαμε για γλώσσα προγραμματισμού την C Sharp (C#) και για την απόκτηση φωτογραφιών την βιβλιοθήκη OPENCV. Συγκεκριμένα, δουλέψαμε με ένα wrapper από C++ σε C#, το Emgu CV.

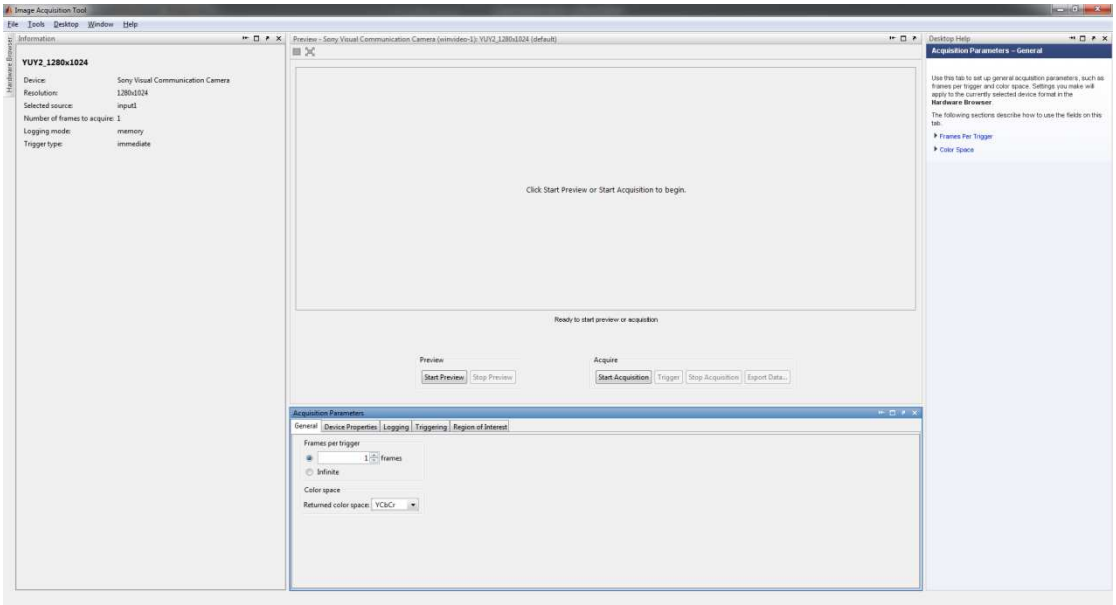
Να αναφέρουμε ότι και στις δυο περιπτώσεις η μέθοδος κατηγοριοποίησης των φωτογραφιών είναι η κατηγοριοποίηση κατά «κοντινότερο γείτονα», καθως είναι μια γρηγορη μέθοδος κατηγοριοποιήσης, καθως και πολύ αξιόπιστη.

Η μέθοδος με τα νευρωνικά δίκτυα δοκιμάστηκε αλλα αποφασίστικε ότι η φάση της εκπαίδευσης του δικτύου είναι παρα πολύ αργή και καταναλωτική σε επεξεργαστική ισχυ και δεν την χρησιμοποιήσαμε τελικά. Επίσης, τα νευρωνικά δίκτυα προυποθέτουν πολλές δοκιμές πάνω στην δομη του δικτύου ώστε να βγάζει σωστά αποτελέσματα, κατι το οποιο είναι και αυτό χρονοβορο.

Υλοποίηση με Matlab

Για την λήψη φωτογραφιών στο Mat lab χρησιμοποιήσαμε το ενσωματωμένο με το Mat lab toolbox, το ονομαζόμενο Image Acquisition.

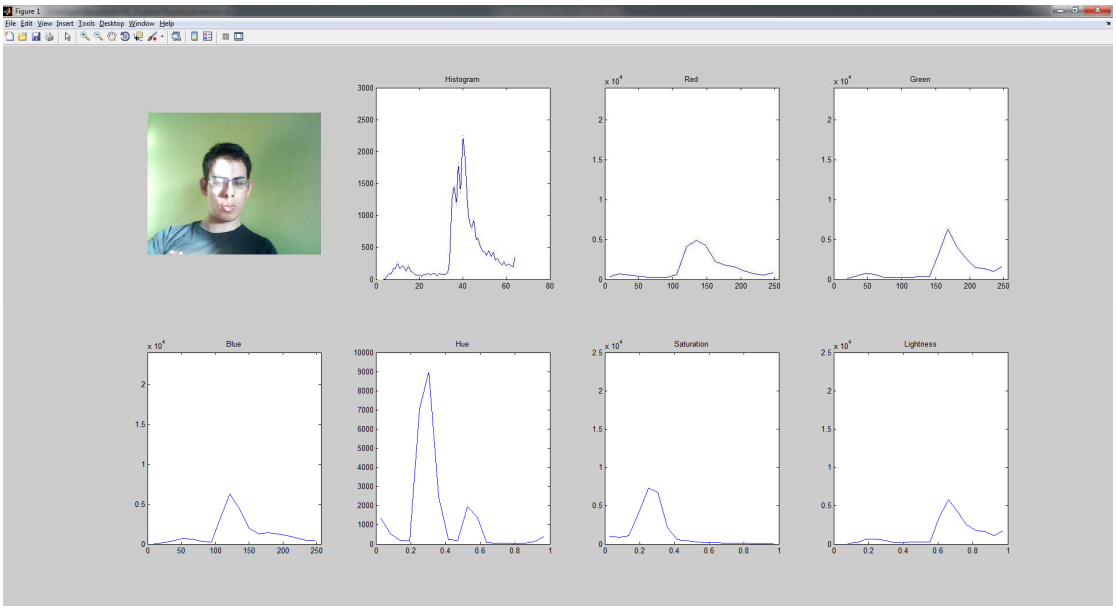
Η εργαλειοθήκη αυτή περιέχει ότι χρειαζόμαστε για να ρυθμίσουμε ότι παραμέτρους χρειαζόμαστε για την λήψη φωτογραφιών. Ανιχνεύει συσκευές λήψης εικόνας και βίντεο καθώς και τις αναλύσεις που υποστηρίζει κάθε κάμερα.



To Image acquisition Toolbox

Αρχικά στο Mat lab για να δούμε τον τρόπο που αναπαριστούνται οι φωτογραφίες και τα ιστογράμματα τους δημιουργήσαμε ένα πρόγραμμα το οποίο τρίβει ένα αριθμό φωτογραφιών και παρουσιάζει τα ιστογράμματα του κάθε καναλιού που δουλεύουμε, καθώς και το ιστόγραμμα ασπρόμαυρης εικόνας.

Τα ιστογράμματα ανανεώνονται σε πραγματικό χρόνο όπως και οι φωτογραφίες, αρά είναι σαν να έχουμε ένα βίντεο μαζί με το ιστόγραμμα του.



Το interface στο Mat lab.Από πάνω αριστερά προς κάτω δεξιά βλέπουμε την φωτογραφία που τραβάμε και τα έξης ιστογράμματα : ασπρόμαυρης εικόνας, κόκκινου, πράσινου , μπλε, αποχρωσης,διαβρωσης και φωτεινότητας.

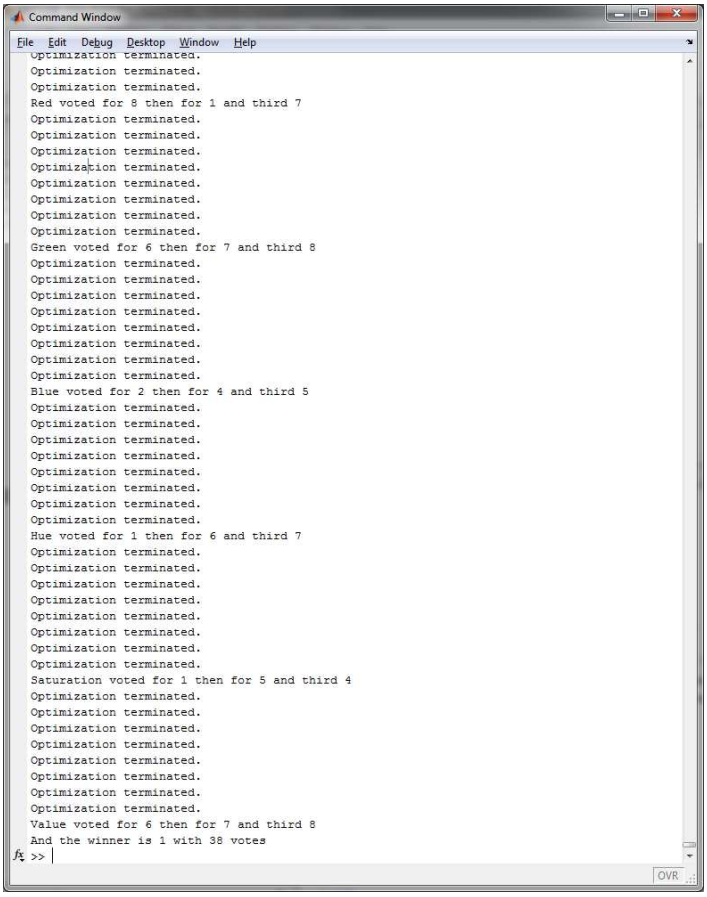
Συνεχίζοντας, ακλουθώντας τα βήματα που αναφέραμε στην θεωρητική προσέγγιση της εργασίας, δημιουργήσαμε ένα πρόγραμμα το οποίο εκτελεί την έξης διαδικασία:

- Τρίβει μια φωτογραφία με ανάλυση εικόνας 176X144 στον χρωματικό χώρο RGB
- Μετατρέπει την φωτογραφία στον χρωματικό χώρου του HSV και grayscale.
- Αποθηκεύει την γνησία φωτογραφία με το όνομα photo# όπου # ο αύξων αριθμός της φωτογραφίας που τραβήχτηκε
- Διαχωρισμός των φωτογραφιών στα 3 κανάλια που του αντιστοιχούν
- Υπολογισμός ιστογραμμάτων κάθε καναλιού της εκάστου φωτογραφίας. Για μικρότερο όγκο δεδομένων και γρηγορότερο υπολογισμό το ιστόγραμμα αποτελείτε από 8 τιμές αντί 256 που είναι το προεπιλεγμένο
- Αποθήκευση όλων αυτών των δεδομένων σε μεταβλητές με ονόματα συμφώνα με τον αριθμό της φωτογραφίας για ευκολότερη αναζήτηση και αντιστοίχηση
- Κίνηση του ρομπότ σε μια ευθεία γραμμή απλοποιώντας το πρόβλημα μας από δισδιάστατο σε μονοδιάστατο.
- Επανεκκίνηση του αλγορίθμου από το πρώτο βήμα.

Μέχρι εδώ έχουμε υλοποιήσει τα δυο πρώτα βήματα που αναφέραμε στην θεωρία.

Στην συνέχεια, δημιουργήσαμε ένα .m file το οποίο εκτελεί τις έξης διαδικασίες:

- Τρίβει μια φωτογραφία ιδίου μεγέθους με αυτή που τραβηχτικέ κατά την διάρκεια της χαρτογράφησης
- Μετατρέπει την φωτογραφία στους κατάλληλους χρωματικούς χώρους
- Υπολογίζει τα ιστογράμματα της νέας φωτογραφίας και τα αποθηκεύει σε προσωρινές μεταβλητές
- Με την μέθοδο Earth movers Distance υπολογίζει την διάφορα του κάθε ιστογράμματος κάθε καναλιού με κάθε αντίστοιχο ιστόγραμμα των φωτογραφιών της βάσης δεδομένων.
- Κάθε ιστόγραμμα δίνει ψήφο στο κάθε ιστόγραμμα που βρέθηκε να αντιστοιχίζει καλυτέρα
- Ταξινομεί τα αποτελέσματα της παραπάνω διαδικασίας
- Τυπώνει στην οθόνη για κάθε κανάλι που έδωσε τις 3 μεγαλύτερες βαθμολογίας και
- Βρίσκει την νικητήρια εικόνα και την γραφεί στην οθόνη.



Τα αποτελέσματα όπως φαίνονται στην γραμμή εντολών του Mat lab

Τα αποτελέσματα που είχαμε από το Mat lab δεν ήταν ιδιαίτερα εντυπωσιακά αλλά σίγουρα έδειχναν ότι οι προοπτικές εξέλιξης του έργου ήταν παρά πολλές.

Σε δόκιμες που έγιναν σε ευθεία γραμμή και με τον ίδιο προσανατολισμό η επιτυχία της εύρεσης της κατάλληλης φωτογραφίας άγγιζε το 90%. Δυστυχώς, τέτοιες συνθήκες δεν είναι πάντα εύκολο να υπάρχουν. Επίσης, το πρόβλημα που προσπαθούμε να λύσουμε αφορά ένα ολόκληρο δωμάτιο όχι μόνο μια ευθεία γραμμή.

Καθώς εργαζόμασταν με το mat lab συναντήσαμε κάποια προβλήματα τα οποία κάποια ήταν σημαντικά κάποια αλλά όχι. Όμως αυτά τα προβλήματα μας οδήγησαν στο να αλλάξουμε περιβάλλον εργασίας.

- Το Mat lab είναι ένα πολύ απαιτητικό πρόγραμμα σε θέματα μνήμης. Αυτό το καθίστα στα περισσότερα μηχανήματα αργό και ίσως σε κάποιους παλιούς υπολογιστές αδύνατο να λειτουργήσει. Σε μια εργασία όπως η δική μας η ταχύτητα παίζει πολύ σημαντικό ρολό καθώς θα θέλαμε να πλησιάσουμε όσο τον δυνατόν γινόταν την ταχύτητα πραγματικού χρόνου.
- Αν και το παραπάνω ήταν ο βασικός λόγος που αλλάξαμε περιβάλλον εργασίας αυτό που μας δημιούργησε επίσης πρόβλημα ήταν

αναποτελεσματικότητα του αλγορίθμου μας. Σκεφτήκαμε ότι ίσως με πιο γρήγορη επεξεργασία θα έχουμε καλυτέρα αποτελέσματα

Για τους παραπάνω λογούς, αποφασίσαμε να δουλέψουμε σε ένα διαφορετικό περιβάλλον, εξίσου δυνατό και ευκολόσβηστο με το Mat lab αλλά και πολύ πιο γρήγορο.

Open Cv – C# -Emgu CV

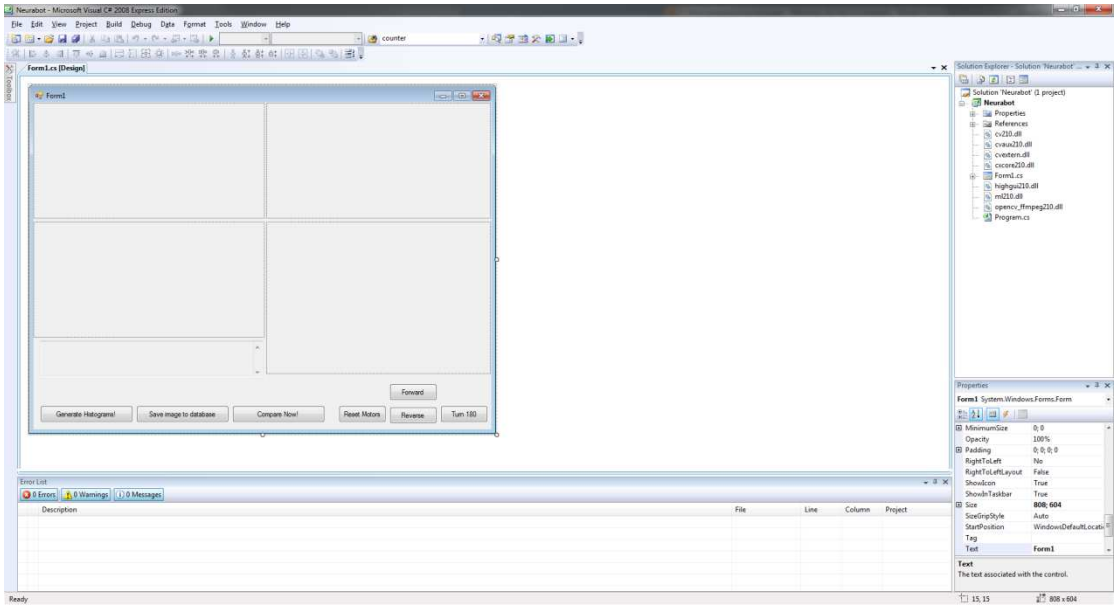
Το opencv είναι μια βιβλιοθήκη για πολλές πλατφόρμες. Είναι παρά πολύ ισχυρή καθώς περιέχει εύκολες συναρτήσεις για οτιδήποτε αφορά λήψη, επεξεργασία, αποθήκευση και χειρισμό εικόνας. Διαθέτει και ανίχνευση κίνησης και πολλά πράγματα τα οποία καθένας μπορεί να χρησιμοποιήσει για να πραγματοποιήσει ότι αφορά Τεχνητή Όραση και επεξεργασία εικόνων.

Οι άμεσες κλήσεις προσδίδουν πολύ μεγάλη ταχύτητα στον υπολογισμό καθώς και μια πλήρη γκάμα από συναρτήσεις και ιδιότητες. Η μεγάλη επιτυχία του οφείλετε επίσης στην πολύ καλή βοήθεια που άρχετε μαζί με το πακέτο και όλα αυτά σε ελεύθερη χρήση, καθώς είναι βιβλιοθήκη ανοιχτού κώδικα.

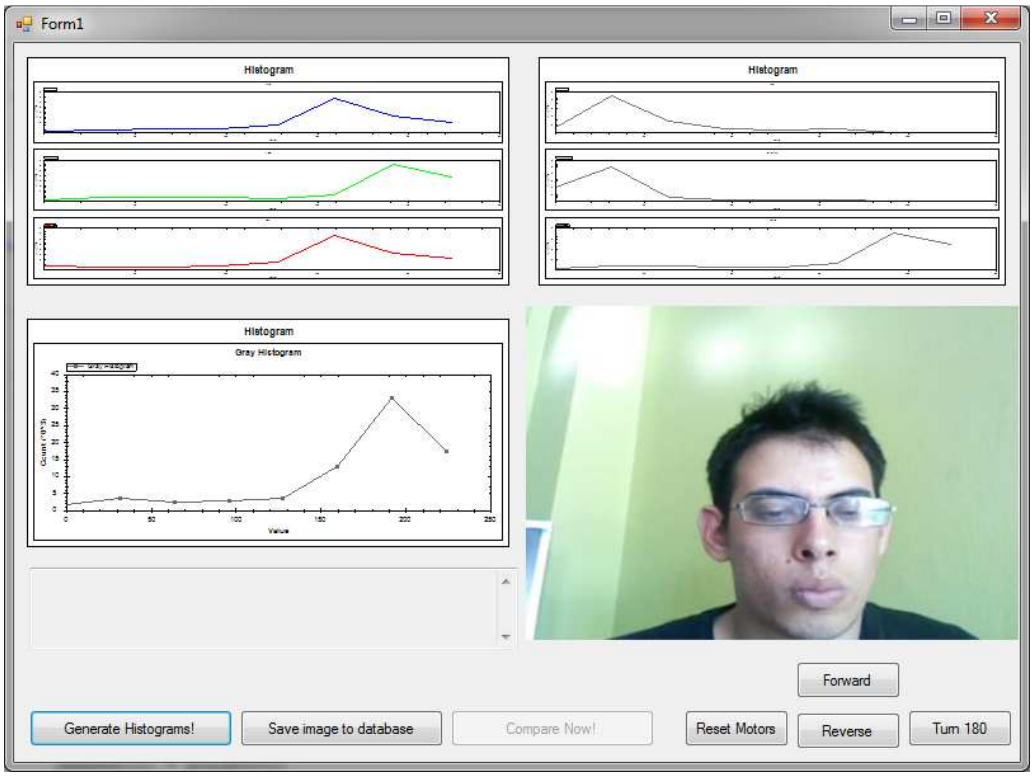
Ενώ το Open Cv ήταν φτιαγμένο για την γλώσσα προγραμματισμού C++, εμείς αποφασίσαμε να χρησιμοποιήσουμε την επίσης .NET γλώσσα C#.

Η C Sharp είναι μια αντικειμενοστραφής γενικής χρήσης γλώσσα προγραμματισμού η οποία αναπτύχτηκε από την Microsoft με σκοπό να βοηθήσει τους όχι και τόσο σκληροπυρηνικούς προγραμματιστές που χρησιμοποιούν την C++. Επειδή το OPEN CV είναι φτιαγμένο για άλλη γλώσσα προγραμματισμού, θα χρησιμοποιήσουμε μια παραλλαγή του Open CV, το Emu CV το οποίο ουσιαστικά είναι ένας κλώνος σε άλλη γλώσσα προγραμματισμού. Επίσης, παρέχει μερικά επιπλέον στοιχειά έλεγχου για να ικανοποιήσει τις ανάγκες οποιουδήποτε προβλήματος.

Η φόρμα μας



Η φόρμα μας αποτελείτε από 3 histogram boxes τα οποία χρησιμοποιούνται για να προβάλλονται τα ιστογράμματα που υπολογίζουμε για τις φωτογραφίες, ένα picture box για την προβολή της φωτογραφίας που έχει λάβει το ρομπότ μας, ένα textbox για την εκτύπωση αποτελεσμάτων καθώς και μερικά κουμπιά που χρησιμοποιούνται για τον έλεγχο των λειτουργιών της φόρμας.



Το interface στη C# Express

Ας αναλύσουμε την λειτούργει κάθε κουμπιού:

1. Generate Histograms: Είναι το βασικό κουμπί που πατάμε και αρχίζει το ρομπότ την χαρτογράφηση του χώρου. Σε αυτή την διαδικασία το ρομπότ τρίβει μια φωτογραφία, την στέλνει στον υπολογιστή όπου και θα αποθηκευτεί για περαιτέρω χρήση. Στην συνέχεια δίνει εντολή στο ρομπότ να κάνει μια κίνησε σε ευθεία γραμμή. Να τονίσουμε ότι και σε αυτό το κομμάτι της εργασίας η χαρτογράφηση θα γίνει σε μια ευθεία γραμμή.

Επίσης, ο προσανατολισμός είναι ίδιος κάθε φορά , καθώς ενώ είχαμε υπολογίσει να έχουμε την ίδια εικόνα σε όσες μοίρες και αν περιστραφεί το ρομπότ δεν υπολογίσαμε τις σκιές που δημιουργούνται από το ίδιο το ρομπότ και την φωτογραφική μηχανή που βρίσκεται πάνω.

2. Save image to Database: Ενώ αρχικά είναι απενεργοποιημένο, μετά από το πρώτο πάτημα του πρώτου κουμπιού, αυτό ενεργοποιείτε και τώρα πλέον μπορεί να πατηθεί. Δίνει εντολή στην υπολογιστή να αποθηκεύσει τις φωτογραφίες που είναι αποθηκευμένες προσωρινά από το Generate Histograms στον σκληρό δίσκο με σκοπό να υποστούν επεξεργασία και σύγκριση όταν αυτό είναι απαραίτητο.
3. Compare now! Και αυτό το κουμπί είναι απενεργοποιημένο αρχικά αλλά όταν αποθηκεύσουμε τις φωτογραφίες μας στον σκληρό δίσκο τότε πλέον γινεται διαθέσιμο. Η λειτουργιά του : τρίβει μια φωτογραφία στο σημείο που βρίσκεται αυτή την στιγμή το ρομπότ και προσπαθεί να καταλάβει σε ποιο σημείο του δωματίου βρίσκεται.

Όπως είχαμε αναφέρει πριν, σε αυτοί εδώ ακριβώς το βήμα αλλάξαμε τρόπο που συγκρίνουμε τα ιστογράμματα μας. Ενώ πριν χρησιμοποιούσαμε EMD, τώρα χρησιμοποιούμε την μέθοδο `BHATTACHARYYA`.

Επισης, να προσθεσουμε ότι μετα από δοκιμες αποφασισαμε να μην μετραμε στην ψηφοφορια τα ιστογραμματα Value καθως τα αποτελεσματα ηταν πολλες φορες λανθασμενα λογο της φωτεινοτητας καθως σε διαφορετικες ωρες της ημερας ειχαμε διαφορετικο φωτισμο η ακομα και η φωτογραφια ηταν πιο σκουροχρωμη από ηταν αποθηκευμενη στον σκληρο δισκο.

4. Τα κουμπιά που βρίσκονται στο δεξί μέρος της φόρμας είναι κουμπιά τηλεχειρισμού του ρομπότ για να το κατευθύνουμε εμείς όπου χρειάζεται.

Τελειώνοντας, παραθέτουμε τα αποτελέσματα και τις παρατηρήσεις αυτής της φάσης.

Λογού του ότι ποτέ στην ερευνά δεν είναι δυνατόν να υπολογίσεις όλες τις πιθανότητες πριν δοκιμάσεις, είδαμε ότι το ρομπότ μας είχε περισσότερο βάρος από ότι μπορούσε να σηκώσουν οι ρόδες του. Το αποτέλεσμα ήταν να είχαμε

παραμόρφωση στις ρόδες και το ρομπότ μας μετά από μερικά μετρά που διανύει αρχίζει και έχει μόνιμο σφάλμα στην κίνηση του.

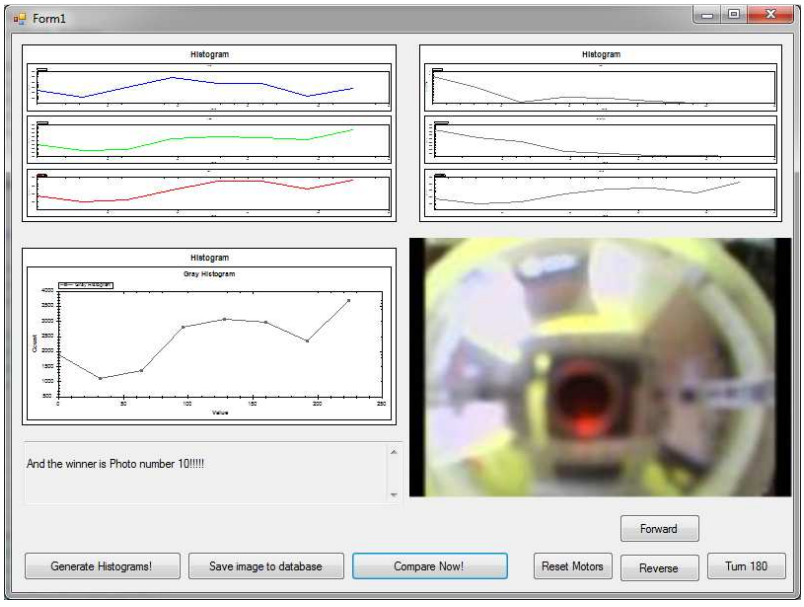
Αυτό συμβαίνει γιατί οι ρόδες «ανοίγουν» προς τα έξω από το βάρος της μεταλλικής σφαίρας και της φωτογραφικής μηχανής με αποτέλεσμα να χάνεται η ακρίβεια στην κίνηση.

Στην 2η αυτή εφαρμογή τα αποτελέσματα ήταν καλυτέρα, καθώς με την διαφοροποίηση του τρόπου σύγκρισης των ιστογραμμάτων τα αποτελέσματα μας ήταν πιο ικανοποιητικά.

Σε ένα συγκεκριμένο πείραμα είχαμε 12 σημεία τα οποία είχαμε αποθηκεύσει και τοποθετώντας το ρομπότ ακριβώς στο σημείο που είχε τραβηχτεί η φωτογραφία της βάσης δεδομένων το αποτέλεσμα ήταν σχεδόν 100%.

Αφήνοντας όμως το ρομπότ να κινηθεί μόνο του μπροστά και πίσω, συναντήσαμε το πρόβλημα που μόνιμου σφάλματος που αναφέραμε πριν λίγο, με συνέπεια το ρομπότ μας να βγει από την ευθεία που ήξερε πολύ καλά.

Και πάλι σε απόσταση 10 εκατοστών από το σημείο και με ίδιο πάντα προσανατολισμό (καθώς είχαμε προβλήματα με τις σκιές που δημιουργούνταν από τα φώτα στην οροφή του κτηρίου που βρισκόμασταν) είχαμε αρκετά καλά αποτελέσματα!



Ένα σωστό αποτέλεσμα από το ρομπότ μας!

ΚΕΦΑΛΑΙΟ 5
ΠΡΟΟΠΤΙΚΕΣ
ΚΑΙ ΕΠΙΛΟΓΟΣ

Προοπτικές

Η χαρτογράφηση χώρου είναι ακόμα ένα ανοιχτό θέμα πάνω στο τομέα της ρομποτικής. Ερευνά γινεται ακόμα πάνω στον τομέα αυτό.

Όπως αναφέραμε κατά την διάρκεια της εργασίας μας υπάρχουν πολλοί τρόποι για καλύτερη κατηγοριοποίηση εικόνων, λήψη εικόνων, τρόποι επεξεργασίας και πολλά αλλά πράγματα που μπορούν να βελτιωθούν και να έχουμε καλυτέρα αποτελέσματα.

Το σίγουρο είναι ότι το μέλλον υπολογίζεται λαμπρό καθώς η χαρτογράφηση είναι πολύ σημαντική για ένα ρομπότ.

Χαρτογράφηση μπορεί να πραγματοποιηθεί και με ιπτάμενα ρομπότ ανεβάζοντας έτσι το επίπεδο έρευνας που μπορεί να πραγματοποιηθεί κατά πολύ.

Ρομπότ ξεναγοί, ρομπότ που κάνουν δουλείες του σπιτιού και ακόμα και τα πιο εξελιγμένα ρομπότ που μπορούν να σταλούν σε άλλους πλανήτες πρέπει να έχουν ένα τρόπο χαρτογράφησης, πλοήγησης και εντοπισμού θέσης.

Με τα νευρωνικά δίκτυα μπορούμε να έχουμε ικανοποιητικά αποτελέσματα σε πολύπλοκα προβλήματα. Αρχικά, θέλαμε και εμείς να ενεργοποιήσουμε ένα νευρωνικό δίκτυο αλλά τα κακά αποτελέσματα και η αργή εκτέλεση των βημάτων στο mat lab μας καθυστέρησε και έτσι αποφασίσαμε να το αφήσουμε για μια επομένη εργασία.

Υπάρχει πολύ ερευνά να γίνει ακόμα και δυνατότητες που ούτε εμείς τις φανταζόμαστε.

Επίλογος

Σε αυτή την εργασία αναλύσαμε τα ρομπότ, το πρόβλημα πλοήγησης και εντοπισμού θέσης με δικά μας μέσα και επιλογές. Τα αποτελέσματα για μας ήταν ικανοποιητικά αλλά το πιο σημαντικό ήταν η γνώση που αποκομίσαμε.

Τέτοιες εργασίες μας αφυπνίζουν και μας προτρέπουν να ψάξουμε στο άγνωστο πτυχές που μπορεί να βοηθήσουν να κάνουμε την ποιότητα ζωής μας καλύτερη.

Πάνω από όλα όμως πρέπει να σεβόμαστε τον άνθρωπο, το περιβάλλον και να μην ξεχνάμε ότι όση τεχνολογία και αν αναπτύξουμε οι αξίες μας δεν πρέπει να αλλάξουν ποτέ.

Ευχαριστήρια

- ✓ Πάνω από όλα στην οικογενεια μου για την στηριξη της
- ✓ Την Μιχαέλα Βρυθια επισης για την στηριξή της
- ✓ Το Γ. Παλαμα, επιβλεπων καθηγητη
- ✓ Τον Ιασωνα Πουλη καθως και τον Μακη Μαριμπι για την βοηθεια τους
- ✓ Γενικα ολους οσοι με βοηθησαν ο κάθε ενας από την δική του μεριά

Βιβλιογραφία

Ιστοσελίδες

- www.wikipedia.com
- www.wikipedia.gr
- http://www.pirobot.org/
- www.in.gr
- http://www.sphoto.com/techinfo/histograms/histograms2.htm

Βιβλία

- **Νευρωνικά Δίκτυα και Εφαρμογές , ΠΑΝΟΣ ΑΡΓΥΡΑΚΗΣ**

Καθηγητής Τμήματος Φυσικής Αριστοτελείου Πανεπιστημίου Θεσσαλονίκης

ΠΑΡΑΡΤΗΜΑ

Κωδικας σε Matlab

Histograms.m

```
%Gia to Image Building Database



%cam=videoinput('winvideo',1,'RGB24_320x240')
;
% set(cam, 'FramesPerTrigger', 1);
% set(cam, 'ReturnedColorSpace', 'rgb');

cam=videoinput('winvideo',1,'YUY2_176x144');
set(cam, 'FramesPerTrigger', 1);
set(cam, 'ReturnedColorSpace', 'rgb');
triggerconfig(cam,'manual');
%********check**********
if ((exist('red','var'))~=1)
    red=0;
end
if ((exist('green','var'))~=1)
    green=0;
end
if ((exist('blue','var'))~=1)
    blue=0;
end
if ((exist('hue','var'))~=1)
    hue=0;
end
if ((exist('saturation','var'))~=1)
    saturation=0;
end
if ((exist('lightness','var'))~=1)
    lightness=0;
end

if ((exist('imagecount','var'))~=1)
    imagecount=0;
end
if ((exist('histogram','var'))~=1)
    histogram=0;
end
%*********End Checking******
start(cam);
for i=0:50
photo2=getsnapshot(cam);
end
grayphoto=rgb2gray(photo2);
hsvimage=rgb2hsv(photo2);
stop(cam);
imwrite(photo2,sprintf('%d.jpg',imagecount+1)
);
 v = genvarname('histogram', who);
 eval([v ' = imhist(grayphoto,16);']);
imagecount=imagecount+1;
red=double(photo2(:,:,1));
green=double(photo2(:,:,2));
blue=double(photo2(:,:,3));
h=hsvimage(:,:,1);
s=hsvimage(:,:,2);
v=hsvimage(:,:,3);
q=genvarname('red',who);
w=genvarname('green',who);
e=genvarname('blue',who);
r=genvarname('hue',who);
t=genvarname('saturation',who);
a=genvarname('lightness',who);
eval([q ' =photo2(:,:,1)']);
eval([w ' =photo2(:,:,2)']);
eval([e ' =photo2(:,:,3)']);
eval([r ' =hsvimage(:,:,1)']);
eval([t ' =hsvimage(:,:,2)']);
eval([a ' =hsvimage(:,:,3)']);
```

Beta.m

```
% sigrisi

%cam=videoinput('winvideo',2,'RGB24_320x240')
;
cam=videoinput('winvideo',1,'YUY2_176x144');
set(cam, 'FramesPerTrigger', 1);
set(cam, 'ReturnedColorSpace', 'rgb');

triggerconfig(cam,'manual');
start(cam);
for i=0:50
temp=getsnapshot(cam);
end
result=zeros(1,8);
tempgray=rgb2gray(temp);
temphsv=rgb2hsv(temp);
%imhist(photo2);
stop(cam);
tempred=temp(:,:,1);
tempgreen=temp(:,:,2);
tempblue=temp(:,:,3);
temph=temphsv(:,:,1);
temps=temphsv(:,:,2);
tempv=temphsv(:,:,3);
```

```
temphistogram=imhist(tempgray);
for i=1:8
result(i)=0;
end



[ca ha]=imhist(tempred,16);
[cb hb]=imhist(red1,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(1)=fval;

[cb hb]=imhist(red2,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(2)=fval;
[cb hb]=imhist(red3,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(3)=fval;
[cb hb]=imhist(red4,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(4)=fval;
[cb hb]=imhist(red5,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(5)=fval;
[cb hb]=imhist(red6,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(6)=fval;
[cb hb]=imhist(red7,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(7)=fval;
[cb hb]=imhist(red8,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(8)=fval;
sorted=sort(x);
for X=1:8
    if sorted(1)==x(X)
        result(X)=(result(X)+10);
       poleposition=X;
    end
    if sorted(2)==x(X)
        result(X)=(result(X)+8);
        second=X;
    end
    if sorted(3)==x(X)
        result(X)=(result(X)+6);
        third=X;
    end
    if sorted(4)==x(X)
        result(X)=(result(X)+4);
    end
    if sorted(5)==x(X)
        result(X)=(result(X)+2);
    end
    if sorted(6)==x(X)
        result(X)=(result(X)+1);
    end

end
fprintf(['Red            voted            for
',num2str(poleposition),'       then       for
',num2str(second),'          and          third
',num2str(third),'\n']);
 [ca ha]=imhist(tempgreen,16);
[cb hb]=imhist(green1,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(1)=fval;

[cb hb]=imhist(green2,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(2)=fval;
[cb hb]=imhist(green3,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(3)=fval;
[cb hb]=imhist(green4,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(4)=fval;
[cb hb]=imhist(green5,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(5)=fval;
[cb hb]=imhist(green6,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(6)=fval;
[cb hb]=imhist(green7,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(7)=fval;
[cb hb]=imhist(green8,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(8)=fval;
sorted=sort(x);
for X=1:8
    if sorted(1)==x(X)
        result(X)=(result(X)+10);
       poleposition=X;
    end
    if sorted(2)==x(X)
        result(X)=(result(X)+8);
        second=X;
    end
    if sorted(3)==x(X)
        result(X)=(result(X)+6);
        third=X;
    end
    if sorted(4)==x(X)
        result(X)=(result(X)+4);
    end
    if sorted(5)==x(X)
        result(X)=(result(X)+2);
    end
    if sorted(6)==x(X)
        result(X)=(result(X)+1);
```

```
    end

end
fprintf(['Green           voted           for
',num2str(poleposition),'       then       for
',num2str(second),'          and          third
',num2str(third),'\n'],'\n');
 [ca ha]=imhist(tempblue,16);
[cb hb]=imhist(blue1,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(1)=fval;

[cb hb]=imhist(blue2,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(2)=fval;
[cb hb]=imhist(blue3,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(3)=fval;
[cb hb]=imhist(blue4,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(4)=fval;
[cb hb]=imhist(blue5,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(5)=fval;
[cb hb]=imhist(blue6,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(6)=fval;
[cb hb]=imhist(blue7,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(7)=fval;
[cb hb]=imhist(blue8,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(8)=fval;
sorted=sort(x);
for X=1:8
    if sorted(1)==x(X)
        result(X)=(result(X)+10);
       poleposition=X;
    end
    if sorted(2)==x(X)
        result(X)=(result(X)+8);
        second=X;
    end
    if sorted(3)==x(X)
        result(X)=(result(X)+6);
        third=X;
    end
    if sorted(4)==x(X)
        result(X)=(result(X)+4);
    end
    if sorted(5)==x(X)
        result(X)=(result(X)+2);
    end
    if sorted(6)==x(X)
        result(X)=(result(X)+1);
```

```
    end

end
fprintf(['Blue           voted           for
',num2str(poleposition),'       then       for
',num2str(second),'          and          third
',num2str(third),'\n']);
 [ca ha]=imhist(temph,16);
[cb hb]=imhist(hue1,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(1)=fval;

[cb hb]=imhist(hue2,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(2)=fval;
[cb hb]=imhist(hue3,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(3)=fval;
[cb hb]=imhist(hue4,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(4)=fval;
[cb hb]=imhist(hue5,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(5)=fval;
[cb hb]=imhist(hue6,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(6)=fval;
[cb hb]=imhist(hue7,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(7)=fval;
[cb hb]=imhist(hue8,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(8)=fval;
sorted=sort(x);
for X=1:8
    if sorted(1)==x(X)
        result(X)=(result(X)+10);
       poleposition=X;
    end
    if sorted(2)==x(X)
        result(X)=(result(X)+8);
        second=X;
    end
    if sorted(3)==x(X)
        result(X)=(result(X)+6);
        third=X;
    end
    if sorted(4)==x(X)
        result(X)=(result(X)+4);
    end
    if sorted(5)==x(X)
        result(X)=(result(X)+2);
    end
    if sorted(6)==x(X)
        result(X)=(result(X)+1);
```

```
    end

end
fprintf(['Hue            voted            for
',num2str(poleposition),'       then       for
',num2str(second),'          and          third
',num2str(third),'\n']);
 [ca ha]=imhist(temps,16);
[cb hb]=imhist(saturation1,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(1)=fval;

[cb hb]=imhist(saturation2,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(2)=fval;
[cb hb]=imhist(saturation3,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(3)=fval;
[cb hb]=imhist(saturation4,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(4)=fval;
[cb hb]=imhist(saturation5,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(5)=fval;
[cb hb]=imhist(saturation6,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(6)=fval;
[cb hb]=imhist(saturation7,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(7)=fval;
[cb hb]=imhist(saturation8,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(8)=fval;
sorted=sort(x);
for X=1:8
    if sorted(1)==x(X)
        result(X)=(result(X)+10);
       poleposition=X;
    end
    if sorted(2)==x(X)
        result(X)=(result(X)+8);
        second=X;
    end
    if sorted(3)==x(X)
        result(X)=(result(X)+6);
        third=X;
    end
    if sorted(4)==x(X)
        result(X)=(result(X)+4);
    end
    if sorted(5)==x(X)
        result(X)=(result(X)+2);
    end
    if sorted(6)==x(X)
        result(X)=(result(X)+1);
```

```
    end

end
fprintf(['Saturation         voted         for
',num2str(poleposition),'       then       for
',num2str(second),'          and          third
',num2str(third),'\n']);
 [ca ha]=imhist(tempv,16);
[cb hb]=imhist(lightness1,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(1)=fval;

[cb hb]=imhist(lightness2,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(2)=fval;
[cb hb]=imhist(lightness3,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(3)=fval;
[cb hb]=imhist(lightness4,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(4)=fval;
[cb hb]=imhist(lightness5,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(5)=fval;
[cb hb]=imhist(lightness6,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(6)=fval;
[cb hb]=imhist(lightness7,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(7)=fval;
[cb hb]=imhist(lightness8,16);
f1=ha;
f2=hb;
w1 = ca / sum(ca);
w2 = cb / sum(cb);
[~, fval] = emd(f1, f2, w1, w2, @gdf);
x(8)=fval;
sorted=sort(x);
for X=1:8
    if sorted(1)==x(X)
        result(X)=(result(X)+10);
       poleposition=X;
    end
    if sorted(2)==x(X)
        result(X)=(result(X)+8);
        second=X;
    end
    if sorted(3)==x(X)
        result(X)=(result(X)+6);
        third=X;
    end
    if sorted(4)==x(X)
        result(X)=(result(X)+4);
    end
    if sorted(5)==x(X)
        result(X)=(result(X)+2);
    end
    if sorted(6)==x(X)
        result(X)=(result(X)+1);
```

```
    end

end
fprintf(['Value           voted           for
',num2str(poleposition),'       then       for
',num2str(second),'          and          third
',num2str(third),'\n']);
 [votes,president]=max(result);
fprintf(['And     the     winner     is     ',
num2str(president),'  with  ',num2str(votes),'
votes\n']);
```

Κώδικας C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using Emgu.CV;
using Emgu.CV.ML;
using Emgu.Util;
using Emgu.CV.UI;
using Emgu.CV.Structure;
using NKH.MindSqualls;
using Emgu.CV.CvEnum;

namespace Neurabot
{

    public partial class Form1 : Form
    {
        NxtBrick brick = new NxtBrick(7);
        NxtMotor motor = new NxtMotor();
        NxtMotor motor2 = new NxtMotor();

        HistogramBox     histbox     =     new
HistogramBox();
        HistogramBox     histbox2     =     new
HistogramBox();
        HistogramBox     histbox3     =     new
HistogramBox();
        Capture capture = new Capture(1);
        public       Image<Gray,       Byte>[]
RedDatabase = new Image<Gray, Byte>[101];
        public       Image<Gray,       Byte>[]
GreenDatabase = new Image<Gray, Byte>[101];
        public       Image<Gray,       Byte>[]
BlueDatabase = new Image<Gray, Byte>[101];
        public       Image<Gray,       Byte>[]
HueDatabase = new Image<Gray, Byte>[101];
        public       Image<Gray,       Byte>[]
SaturationDatabase    =    new    Image<Gray,
Byte>[101];
        public       Image<Gray,       Byte>[]
ValueDatabase = new Image<Gray, Byte>[101];
        public       Image<Gray,       Byte>[]
GrayDatabase = new Image<Gray, Byte>[101];
        public int counter = 0;
        Matrix<Byte>[]     matrix     =     new
Matrix<Byte>[1];
        Image<Bgr, byte>[] captured_images =
new Image<Bgr, byte>[300];
        DenseHistogram hist;
        Image<Gray, Byte>[] imagear;
        Image<Hsv, Byte> hsvphoto;
        Image<Gray, Byte> grayphoto;
        Image<Bgr, Byte> photo;
        public double compareResult;
        public Double[,] compareMatrix = new
Double[100, 6];
        public  Double[,]  sortMatrix  =  new
Double[100, 6];
        public NxtMotorSync motorPair;
        public Form1()
        {
            InitializeComponent();

        }

        private     void     Form1_Load(object
senders, EventArgs er)
        {
System.IO.Directory.CreateDirectory("../Image
Database");
            histbox.Size = panel1.Size;
            histbox2.Size = panel2.Size;
            histbox3.Size = panel3.Size;
            panel1.Controls.Add(histbox);
            panel2.Controls.Add(histbox2);
            panel3.Controls.Add(histbox3);
            button2.Enabled = false;

            button3.Enabled = false;
            brick.MotorB = motor;
            brick.MotorC = motor2;
            motorPair          =          new
NxtMotorSync(motor, motor2);
            brick.Connect();

motorPair.ResetMotorPosition(false);
        }




        private   void   button1_Click(object
sender, EventArgs e)
        {

            counter = 0;

            //capture  some  photos,  resize
them and show them
            for (int i = 0; i < 10; i++)
            {
                button3.Enabled = false;
                counter++;
                captureImage();

pictureBox1.Load("photo.jpg");
                pictureBox1.Refresh();

                //Create the images, bgr and
hsv
                hsvphoto  =  new  Image<Hsv,
Byte>(photo.Size);
                grayphoto  =  new  Image<Gray,
Byte>(photo.Size);

                //convert gbr to hsv and gray

CvInvoke.cvCvtColor(photo.Ptr,  hsvphoto.Ptr,
COLOR_CONVERSION.CV_BGR2HSV);

CvInvoke.cvCvtColor(photo.Ptr, grayphoto.Ptr,
COLOR_CONVERSION.CV_BGR2GRAY);

                //Gray histogram
                imagear  =  new  Image<Gray,
Byte>[1];
                imagear[0] = grayphoto;

                //calculate gray histogram
                hist = new DenseHistogram(8,
new RangeF(0, 256));
                hist.Calculate(imagear,
false, null);

                //Split  the  photos  to  3
channels
                photosplit(photo);
                photosplit(hsvphoto);

                //setup the histogram boxes
                histbox.ClearHistogram();
                histbox2.ClearHistogram();
                histbox3.ClearHistogram();

                //Bgr and HSV histograms

histbox.GenerateHistograms(photo, 8);

histbox2.GenerateHistograms(hsvphoto, 8);
                histbox3.AddHistogram("Gray
Histogram", Color.DimGray, hist);

                //paint the histograms
```

```
                histbox3.Refresh();
                histbox.Refresh();
                histbox2.Refresh();
                //    saveImage();
                button2.Enabled = true;
                if (i != 9)
                {
                    brick.Program           =
"Forward.rxe";
                }
               //  MessageBox.Show("Click  Ok
to continue",  string.Format("Photo  No  {0}",
i+1));

System.Threading.Thread.Sleep(3000);
            }

        }
        //method for splitting Bgr images
        void photosplit(Image<Bgr, Byte> h)
        {
            Image<Gray,  Byte>[]  channels  =
new Image<Gray, byte>[3];
            channels = h.Split();
            BlueDatabase[counter]           =
channels[0];
            GreenDatabase[counter]          =
channels[1];
            RedDatabase[counter]            =
channels[2];
        }

        //method for splitting Hsv images
        void photosplit(Image<Hsv, Byte> h)
        {
            Image<Gray,  Byte>[]  channels  =
new Image<Gray, byte>[3];
            channels = h.Split();
            // return channels;
            HueDatabase[counter]            =
channels[0];
            SaturationDatabase[counter]     =
channels[1];
            ValueDatabase[counter]          =
channels[2];
        }
        //method for capturing
        void captureImage()
        {
            for (int i = 0; i < 30; i++)
            {
                captured_images[counter]    =
capture.QuerySmallFrame();
            }
            photo = captured_images[counter];
            photo.Resize(176,            144,
INTER.CV_INTER_LINEAR);
            photo.Save("photo.jpg");

        }

        private   void   button2_Click(object
sender, EventArgs e)
        {
            saveImage();
            button3.Enabled = true;
        }
        //saving image to database
        void saveImage()
        {
            if (counter != 0)
            {
                MessageBox.Show("Please  wait
until your images are being saved!\nClick Ok
to    continue!",    "Saving    images...",
MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);
                for (int i = 1; i < counter +
1; i++)
                {


captured_images[i].Save(String.Format("../Ima
ge Database/image{0}.jpg", i));
                }
                MessageBox.Show("Done!",
"Success!",            MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);
            }
            else
            {
                MessageBox.Show("No  photo  to
save  yet!",  "Error",  MessageBoxButtons.OK,
MessageBoxIcon.Hand);
            }
        }
        //compare histograms method
        double  compareHistograms(Image<Gray,
Byte> h, Image<Gray, Byte> i)
        {
            Image<Gray,  Byte>[]  imar  =  new
Image<Gray, byte>[1];
            Image<Gray,  Byte>[]  imar2  =  new
Image<Gray, byte>[1];
            imar[0] = h;
            DenseHistogram    hist1    =    new
DenseHistogram(8, new RangeF(0, 256));
            DenseHistogram    hist2    =    new
DenseHistogram(8, new RangeF(0, 256));
            hist1.Calculate(imar,       false,
null);
            imar2[0] = i;
            hist2.Calculate(imar2,      false,
null);
            compareResult                   =
CvInvoke.cvCompareHist(hist1.Ptr,  hist2.Ptr,
HISTOGRAM_COMP_METHOD.CV_COMP_BHATTACHARYYA);
            return compareResult;
        }

        void    bestMatch(Image<Bgr,    Byte>
image)
        //method  for  finding  closest  photo
match
        {
            Image<Bgr, Byte> temptestingPhoto
= new Image<Bgr, byte>(176, 144);
            Image<Bgr,  Byte>  testingPhoto  =
image;
            Image<Hsv,  Byte>  hsvTestingPhoto
= new Image<Hsv, byte>(testingPhoto.Size);
            Image<Gray,  Byte>[]  channels  =
new Image<Gray, byte>[3];
            Image<Gray, Byte>[] hsvchannels =
new Image<Gray, byte>[3];
            Image<Gray,  Byte>[]  tempchannels
= new Image<Gray, byte>[3];
            Image<Gray,                Byte>[]
temphsvchannels = new Image<Gray, byte>[3];
            Image<Hsv,                  Byte>
temphsvTestingPhoto    =    new    Image<Hsv,
byte>(image.Size);
            CvInvoke.cvCvtColor(image.Ptr,
temphsvTestingPhoto.Ptr,
COLOR_CONVERSION.CV_BGR2HSV);
            channels = testingPhoto.Split();
            hsvchannels                     =
hsvTestingPhoto.Split();
            for (int i = 0; i < counter; i++)
            {

                temptestingPhoto     =     new
Image<Bgr,      Byte>(string.Format("../Image
Database/image{0}.jpg", i + 1));
                tempchannels                =
temptestingPhoto.Split();

CvInvoke.cvCvtColor(temptestingPhoto.Ptr,
temphsvTestingPhoto.Ptr,
COLOR_CONVERSION.CV_BGR2HSV);
                temphsvchannels             =
temphsvTestingPhoto.Split();
                compareMatrix[i,     0]     =
compareHistograms(channels[0],
tempchannels[0]);
                compareMatrix[i,     1]     =
compareHistograms(channels[1],
tempchannels[1]);
                compareMatrix[i,     2]     =
compareHistograms(channels[2],
tempchannels[2]);
```

```
                compareMatrix[i,      3]      =
compareHistograms(hsvchannels[0],
temphsvchannels[0]);
                compareMatrix[i,      4]      =
compareHistograms(hsvchannels[1],
temphsvchannels[1]);
                compareMatrix[i,      5]      =
compareHistograms(hsvchannels[2],
temphsvchannels[2]);

            }
            //sorting compareMatrix
            Array.Copy(compareMatrix,
sortMatrix, compareMatrix.Length);
           // sortMatrix = compareMatrix;
            double temp;
            for (int e = 0; e < counter-1;
e++)
            {
                for (int w = 0; w < 6; w++)
                {

                    for (int q = 0; q <
counter-1; q++)
                    {
                        if (sortMatrix[q, w]
> sortMatrix[q + 1, w])
                        {
                            temp            =
sortMatrix[q, w];
                            sortMatrix[q,   w]
= sortMatrix[q + 1, w];
                            sortMatrix[q + 1,
w] = temp;

                        }
                    }
                }

            }

        }

        private   void   button3_Click(object
sender, EventArgs e)
        {
           // textBox1.Text = "";
            int[]     imageVotes     =     new
int[counter];
            int president = 0;
            int votes;

            Image<Bgr,   Byte>   snap   =   new
Image<Bgr, byte>(176, 144);
            for (int q = 0; q < 30; q++)
            {
                snap                        =
capture.QuerySmallFrame();
            }
            snap.Resize(176,             144,
INTER.CV_INTER_LINEAR);
            bestMatch(snap);
            for (int channel = 0; channel <
5; channel++)
            {
                for (int photoNr = 0; photoNr
< counter; photoNr++)
                {
                    if
(compareMatrix[photoNr,      channel]      ==
sortMatrix[0, channel])
                    {
                        imageVotes[photoNr] =
imageVotes[photoNr] + 10;
                    }
                    else                    if
(compareMatrix[photoNr,      channel]      ==
sortMatrix[1, channel])
                    {
                        imageVotes[photoNr] =
imageVotes[photoNr] + 8;
                    }
                    else                    if
(compareMatrix[photoNr,      channel]      ==
sortMatrix[2, channel])
                    {
                        imageVotes[photoNr] =
imageVotes[photoNr] + 6;
                    }
                    else                    if
(compareMatrix[photoNr,      channel]      ==
sortMatrix[3, channel])
                    {
                        imageVotes[photoNr] =
imageVotes[photoNr] + 4;
                    }
                    else                    if
(compareMatrix[photoNr,      channel]      ==
sortMatrix[4, channel])
                    {
                        imageVotes[photoNr] =
imageVotes[photoNr] + 2;
                    }
                }
            }


            votes = imageVotes.Max();
            for (int i = 0; i < counter; i++)
            {
                if (votes == imageVotes[i])
                {
                    president = i;
                }
            }
            textBox1.Text                   =
String.Format(textBox1.Text                 +
Environment.NewLine  +  "And  the  winner  is
Photo number {0}!!!!!!", president + 1);
        }

        private   void   button4_Click(object
sender, EventArgs e)
        {
            brick.Program = "Forward.rxe";
        }

        private   void   button5_Click(object
sender, EventArgs e)
        {
            brick.Program = "Reverse.rxe";
        }

        private   void   button6_Click(object
sender, EventArgs e)
        {


motorPair.ResetMotorPosition(false);
        }

        private   void   button7_Click(object
sender, EventArgs e)
        {

motorPair.ResetMotorPosition(false);
            motorPair.Run(100, 500, 100);

motorPair.ResetMotorPosition(false);
        }


    }
}
```