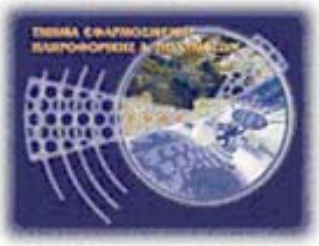
Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων



Πτυχιακή εργασία

“Ανίχνευση ίριδας και εφαρμογές”

Σπουδαστές :
Ζούζουλας Χαράλαμπος ΑΜ 1289, Θεοδώρου Βασίλειος ΑΜ 1770

Επιβλέπων καθηγητής : Τριανταφυλλίδης Γεώργιος

Επιτροπή Αξιολόγησης :

Ημερομηνία Παρουσίασης :

Ευχαριστίες

Θα θέλαμε να ευχαριστήσουμε τον επιβλέποντα καθηγητή κ. Τριανταφυλλίδη Γεώργιο καθώς και όλους όσους μας βοήθησαν στην εκπόνηση της πτυχιακής μας εργασίας.

Abstract

In this final year project, we describe ways of detecting the iris with the use of computer vision techniques. Then the development of an application is presented, where the cursor movement is achieved accordingly to the gaze direction. The goal of this project was to create the basis of an application, capable of running on low budget hardware, which could provide people with disabilities a pc interface, allowing them to improve their quality of life.

Σύνοψη

Σε αυτή τη πτυχιακή εργασία περιγράφονται τρόποι ανίχνευσης της ίριδας χρησιμοποιώντας τεχνικές όρασης υπολογιστών. Στη συνέχεια παρουσιάζεται η ανάπτυξη μίας εφαρμογής όπου ο κέρσορας κινείται σύμφωνα με τη κατεύθυνση του βλέμματος. Στόχος της πτυχιακής είναι η δημιουργία της βάσης μια διεπαφής Η/Υ, η οποία θα επιτρέπει σε άτομα με αναπηρία να αποκτήσουν μια καλύτερη σε ποιότητα ζωή .

Περιεχόμενα

Πίνακας Εικόνων

Λίστα Πινάκων

1. Εισαγωγή

1.1 Περίληψη

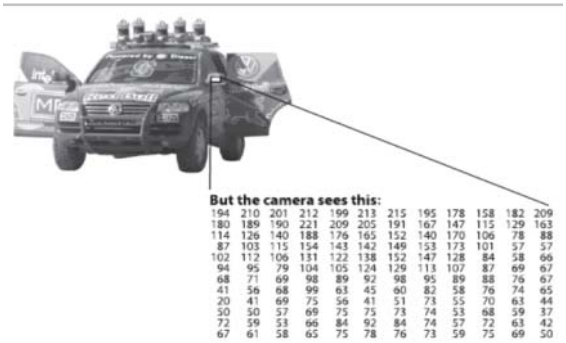
Σκοπός αυτής της πτυχιακής εργασίας είναι η ανίχνευση ίριδας, η παρακολούθηση της και ο χειρισμός του κέρσορα σύμφωνα με την κατεύθυνση του βλέμματος σε πραγματικό χρόνο, χρησιμοποιώντας τεχνικές όρασης υπολογιστών. Προσπαθήσαμε να επιτύχουμε τα παραπάνω χρησιμοποιώντας υλικό εύκολα διαθέσιμο (κοινές webcam) και όχι εξελιγμένες λύσεις (οπτικά laser, ακριβές κάμερες υψηλής ανάλυσης, ειδικούς σένσορες) ή ειδικούς συνδυασμούς (webcam τοποθετημένες πολύ κοντά στο μάτι χρησιμοποιώντας μηχανισμούς στήριξης στο κεφάλι). Επίσης θέλαμε το πρόγραμμα μας να μπορεί να τρέχει σε υπολογιστές χαμηλής τιμής (φτηνό μαθητικό notebook) και να μην είναι απαραίτητο για κάποιον που θέλει να το χρησιμοποιήσει να αποκτήσει την αιχμή της τεχνολογίας στους υπολογιστές. Απώτερος σκοπός μας ήταν να δημιουργήσουμε την βάση μίας εφαρμογής η οποία θα δώσει την δυνατότητα σε άτομα με κινητικές δυσκολίες μιας καλύτερης σε ποιότητα ζωής. Χρησιμοποιήθηκαν οι γλώσσες προγραμματισμού C++ και C# και η βιβλιοθήκη ανοιχτού κώδικα της OpenCV.

1.2 Τι είναι η τεχνητή όραση ?

Τεχνητή όραση είναι η μετατροπή των δεδομένων από μια φωτογραφική μηχανή ή μια βιντεοκάμερα, σε ένα συμπέρασμα ή μια νέα αναπαράσταση. Οι μετατροπές αυτές γίνονται για την επίτευξη ορισμένων στόχων. Τα συμπεράσματα μπορεί να είναι "υπάρχει ένα πρόσωπο σε αυτή τη εικόνα" ή "υπάρχουν 14 καρκινικά κύτταρα σε αυτήν τη διαφάνεια". Μια νέα αναπαράσταση μπορεί να είναι η μετατροπή μιας έγχρωμης εικόνας σε ασπρόμαυρη ή η αφαίρεση της κίνησης της κάμερας από μια ακολουθία εικόνων.

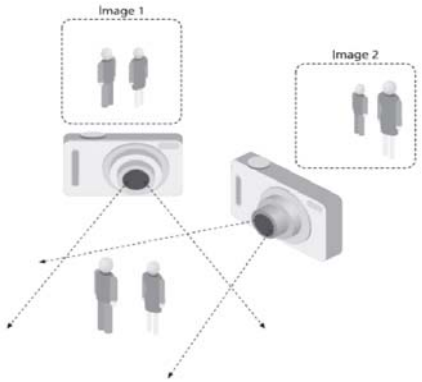
Επειδή είμαστε οπτικά πλάσματα, είναι εύκολο να μας δημιουργηθεί η λανθασμένη εντύπωση ότι η τεχνητή όραση είναι κάτι απλό. Πόσο δύσκολο είναι να βρεθεί, για παράδειγμα, ένα αυτοκίνητο σε μια εικόνα; Η αρχική σας διαίσθηση ίσως είναι αρκετά παραπλανητική. Ο ανθρώπινος εγκέφαλος διαχωρίζει το σήμα της όρασης σε πολλά κανάλια, τα οποία δίνουν διαφορετικά είδη πληροφοριών . Έχει ένα σύστημα αναγνώρισης το οποίο αναδεικνύει πια είναι τα σημαντικά τμήματα της εικόνας για εξέταση και μειώνει την σημασία των άλλων τμημάτων. Αυτό που είναι δεδομένο είναι ότι η διαδικασία της όρασης αποτελείται από πολύ σύνθετες παραμέτρους οι οποίες, μέχρι στιγμής, είναι ελάχιστα κατανοητές. Υπάρχουν διάσπαρτες συνειρμικές εισαγωγές από τους αισθητήρες ελέγχου των μυών των ματιών και όλες τις άλλες ανθρώπινες αισθήσεις που επιτρέπουν στον εγκέφαλο να λειτουργεί βάση συσχετίσεων, οι οποίες δημιουργήθηκαν έπειτα από χιλιάδες χρόνια εξέλιξης του ανθρώπινου είδους. Τα συστήματα ανατροφοδότησης πληροφοριών στον εγκέφαλο επενεργούν σε όλα τα στάδια της επεξεργασίας συμπεριλαμβανομένου του hardware των αισθητήρων, δηλαδή τα μάτια, τα οποία ελέγχουν μηχανικά το φωτισμό μέσω της ίριδας και την λήψη της εικόνας στην επιφάνεια του αμφιβληστροειδή.

Σε ένα σύστημα τεχνητής όρασης όμως, ο ηλεκτρονικός υπολογιστής λαμβάνει ένα σύνολο από αριθμούς από μια κάμερα και αυτό είναι όλο. Ως επί το πλείστον, δεν υπάρχει ενσωματωμένη αναγνώριση προτύπων, δεν υπάρχει αυτόματος έλεγχος εστίασης και διαφράγματος, και δεν υπάρχουν συσχετίσεις από αιώνες εξέλιξης. Τα συστήματα όρασης βρίσκονται ακόμη σε νηπιακό στάδιο.



1-1 Όραση υπολογιστή.

Στην εικόνα βλέπουμε ένα αυτοκίνητο και όπως μπορείτε να δείτε στην μεριά του οδηγού υπάρχει ένας καθρέπτης. Αυτό που βλέπει ένας υπολογιστής είναι ένας πίνακας από αριθμούς. Κάθε μεμονωμένος αριθμός του συγκεκριμένου πίνακα εμπεριέχει ένα αρκετά μεγάλο συστατικό θορύβου και έτσι από μόνος του μας δίνει λίγες πληροφορίες. Το έργο μας είναι να μετατραπεί αυτό το θορυβώδες σύνολο αριθμών στο συμπέρασμα «καθρέφτης».



1-2 Πρόβλημα 2Dαπεικόνισης ενός 3D κόσμου

Η εικόνα 1.2-2 μας δίνει μας δίνει μια παραπάνω ιδέα για την δυσκολία της τεχνητής όρασης. Λαμβάνοντας υπόψη μία δισδιάστατη προβολή ενός τρισδιάστατου κόσμου, δεν υπάρχει ένας και μοναδικός τρόπος για να ανακατασκευάσει το 3D σήμα, και τυπικά, ένα τέτοιας φύσεως πρόβλημα δεν έχει μοναδική ή οριστική λύση.

Η ίδια 2D εικόνα θα μπορούσε να αντιπροσωπεύει έναν άπειρο αριθμό συνδυασμών 3D σκηνών, ακόμα και αν τα δεδομένα ήταν τέλεια. Ωστόσο, όπως ήδη αναφέρθηκε, τα δεδομένα περιέχουν θόρυβο και παραμορφώσεις. Αυτή η διαφθορά μπορεί να πηγάζει από διακυμάνσεις στο περιβάλλον (καιρικές συνθήκες, φωτισμός, αντανακλάσεις, κινήσεις), από ατέλειες στο φακό, από το μηχανικό setup, το θόλωμα κίνησης, τον ηλεκτρικό θόρυβο στον αισθητήρα αλλά και από λάθη κατά την συμπίεση μετά την λήψη της εικόνας.

Με δεδομένες αυτές τις αποθαρρυντικές προκλήσεις, κατά το σχεδιασμό ενός πρακτικού συστήματος, γνώσεις που προϋπάρχουν μπορούν να χρησιμοποιούνται ώστε να εργαστούμε γύρω από τους περιορισμούς που μας επιβάλλονται από τους οπτικούς αισθητήρες. Εξετάστε το παράδειγμα ενός κινητού ρομπότ που η εργασία του είναι να μαζεύει συρραπτικά σε ένα κτίριο. Το ρομπότ θα μπορούσε να χρησιμοποιήσει το γεγονός ότι τα συρραπτικά βρίσκονται συνήθως πάνω σε γραφεία. Αυτό δίνει μια έμμεση αναφορά του μεγέθους. Τα συρραπτικά θα πρέπει να χωράνε πάνω στα γραφεία. Επίσης, βοηθά η εξάλειψη της πιθανότητας «αναγνώρισης» συρραπτικών σε απίθανους χώρους (π.χ. στην

οροφή ή σε ένα παράθυρο). Το ρομπότ θα αγνοήσει ένα αντικείμενο με σχήμα παρόμοιο ενός συρραπτικού εάν στερείται την προϋπόθεση του ξύλινου φόντου ενός γραφείου. Αντίθετα, με λειτουργίες όπως η ανάκτηση εικόνας, όλες οι εικόνες με συρραπτικά σε μια βάση δεδομένων μπορεί να είναι πραγματικά συρραπτικά και αντικείμενα τα οποία θα μπορούσαν να θεωρηθούν λανθασμένα συρραπτικά από το ρομπότ, πιθανώς να έχουν έμμεσα αποκλειστεί από τις υποθέσεις αυτών που πήραν τις φωτογραφίες. Ο φωτογράφος δηλαδή να πήρε εικόνες μόνο πραγματικών συρραπτικών. Οι άνθρωποι τείνουν επίσης να τοποθετούν στο κέντρο της εικόνας τα αντικείμενα που τους ενδιαφέρουν κατά τη λήψη των φωτογραφιών καθώς και να τους δίνουν χαρακτηριστικούς προσανατολισμούς. Κατά συνέπεια, υπάρχουν συχνά υπονοούμενες πληροφορίες στις φωτογραφίες που λαμβάνονται από ανθρώπους. Οι πληροφορίες μπορούν επίσης να διαμορφωθούν ρητά με τεχνικές machine learning.

Άλλες μεταβλητές όπως το μέγεθος, η φορά της βαρύτητας και άλλα, μπορούν να συσχετίζονται με τις τιμές τους σε ένα σύνολο κατάρτισης. Εναλλακτικά, κάποιος θα μπορούσε να προσπαθήσει να μετρήσει αυτές τις μεταβλητές με τη χρησιμοποίηση πρόσθετων αισθητήρων. Η χρήση ενός ανιχνευτή λέιζερ για να μετρήσει το βάθος μας επιτρέπει να μετρήσουμε ακριβώς το μέγεθος ενός αντικειμένου.

Το επόμενο πρόβλημα που αντιμετωπίζει η όραση υπολογιστών είναι ο θόρυβος. Ο θόρυβος εξετάζεται με τη χρησιμοποίηση στατιστικών μεθόδων. Για παράδειγμα, θα ήταν αδύνατο να ανιχνευτεί μία ακμή σε μία εικόνα συγκρίνοντας ένα σημείο με άλλα γειτονικά. Αλλά εάν χρησιμοποιήσουμε στατιστικές μεθόδους σε μία περιοχή, η ανίχνευση ακμών γίνεται ευκολότερη. Επίσης, είναι δυνατή η αντιστάθμιση του θορύβου παίρνοντας στατιστικά ανά χρονικά διαστήματα. Ακόμα, μια άλλη τεχνική για να αντιμετωπιστεί ο θόρυβος και η παραμόρφωση είναι η κατασκευή μοντέλων τα οποία εκπαιδεύονται από τα άμεσα διαθέσιμα δεδομένα. Για παράδειγμα, επειδή οι παραμορφώσεις των φακών έχουν μελετηθεί και κατανοηθεί, μπορεί κάποιος να μαθαίνοντας τις παραμέτρους για ένα απλό πολυωνυμικό μοντέλο, να καταφέρει να περιγράψει και να αναστρέψει το αποτέλεσμα των παραμορφώσεων.

Οι ενέργειες και οι αποφάσεις της όρασης υπολογιστών διενεργούνται για συγκεκριμένους σκοπούς. Μπορεί να θέλουμε να αφαιρέσουμε θόρυβο από μία εικόνα ώστε ένα σύστημα ασφάλειας να μπορεί να εντοπίσει μια απειλή. Τα λογισμικά τεχνητής όρασης για ρομπότ που περιπλανιούνται σε γραφεία πιθανότατα εφαρμόζουν διαφορετικές στρατηγικές από ακίνητες κάμερες ασφάλειας επειδή τα δύο αυτά συστήματα έχουν διαφορετικά πεδία και σκοπούς. Γενικός κανόνας: όσο περιορισμένο είναι το πεδίο της τεχνητής όρασης, τόσο πιο πιθανό είναι να ανταποκριθεί σε αυτό περιορίζοντας και απλοποιώντας το πρόβλημα, κάνοντας την επίλυση πιο αξιόπιστη [1].

1.3 Ιστορία ανίχνευσης ίριδας

Ο Robert L. Fantz, ψυχολόγος στο Western Reserve University, (1925 – 1981), διεξήγαγε μελέτες για τις οπτικές δυνατότητες αναγνώρισης σε νήπια, και είναι μεταξύ των συχνότερα αναφερμένων ερευνητών στον τομέα της αναπτυξιακής ψυχολογίας. Αποφάσισε να ακολουθήσει το βλέμμα των νηπίων για να διακρίνει τις οπτικές προτιμήσεις τους. Ένα μωρό θα τοποθετούταν σε ένα θάλαμο για απομονωθεί από εξωτερικά ερεθίσματα. Κατόπιν, καθώς το μωρό καθόταν στην πλάτη του μια σειρά οπτικών ερεθισμάτων εμφανιζόταν στην οροφή του θαλάμου. Ο Fantz κατόρθωσε να απεικονίσει το σχέδιο που η κόρη του νηπίου κοιτούσε και υπολόγιζε το χρονικό διάστημα που η κόρη του μωρού έπεφτε σε κάποιο οπτικό ερέθισμα. Η εκλεκτική προτίμηση του νηπίου παρουσιαζόταν από την αντανάκλαση στην επιφάνεια του ματιού του νηπίου.

Σύμφωνα με το Δρ Fantz, εάν ένα νήπιο κοίταξε περισσότερο σε ένα ερέθισμα από ένα άλλο σε συνεχή βάση, θα μπορούσε να ερμηνευθεί ως προτίμηση του νηπίου, και η έρευνά του οδήγησε στην κατανόηση των βασικών στοιχείων της οπτικής προτίμησης των νηπίων και σε μια μεθοδολογία για την τεκμηρίωση της ανάπτυξης της ανθρώπινης όρασης. Αποδείχτηκε ότι τα μωρά προτιμούν τα πρόσωπα από τα μη-πρόσωπα, μια παρατήρηση που μελέτες θεωρούν ακόμα αληθινή[2].

Το eye tracking είναι η διαδικασία μέτρησης είτε του σημείου εστίασης του βλέμματος "το που κοιτάζουμε" είτε της κίνησης του ματιού σχετικά με το κεφάλι. Ένας eye tracker είναι μια συσκευή μέτρησης για τις θέσεις ματιών και τη κίνηση ματιών. Το Eye tracking χρησιμοποιείται στην έρευνα των οπτικών συστημάτων, στην ψυχολογία, στη γνωστική γλωσσολογία, σε εφαρμογές υπολογιστών, συστήματα επικοινωνιών για τα άτομα με ειδικές ανάγκες, και στο σχεδιασμό προϊόντων [3].

Σήμερα στις ψηφιακές κάμερες η ανίχνευση ίριδας χρησιμοποιείται για την εστίαση. Η Canon EOS 5 ήταν μία ημι-επαγγελματική κάμερα με σύστημα αυτό-εστίασης που κυκλοφόρησε στην αγορά το Νοέμβρη του 1992. Η EOS 5/A2e παρείχε εστίαση ελεγχόμενη με το μάτι, που επέτρεπε στο χρήστη την επιλογή ενός εκ των πέντε σημείων εστίασης κοιτώντας μέσω του σκοπεύτρου. Η κάμερα χρησιμοποιούσε το συνδυασμό δύο IRED και υπέρυθρων αισθητήρων που ακολουθούν τις κινήσεις των ματιών του φωτογράφου και ενεργοποιούσαν έναν από τους πέντε αισθητήρες autofocus που αντιστοιχούσαν στην περιοχή όπου ο φωτογράφος κοίταζε [4].

Η μοναδικότητα της ίριδας επιτρέπει σε συστήματα ασφαλείας τη χρησιμοποίηση της για ταυτοποίηση. Η αναγνώριση ίριδας είναι μία βιομετρική μέθοδος που χρησιμοποιεί τεχνικές αναγνώρισης προτύπων βασισμένες σε υψηλής ανάλυσης εικόνες ίριδας ενός ατόμου, και μπορεί να χρησιμοποιηθεί για τη ταυτοποίηση του [5].

1.4 Δομή της εργασίας

Στα επόμενα κεφάλαια περιγράφονται τεχνικές και αλγόριθμοι για την επίτευξη της ανίχνευσης ίριδας. Το επόμενο κεφάλαιο αφορά τη βιβλιοθήκη που χρησιμοποιήθηκε για την επίτευξη των στόχων της εργασίας. Στο τρίτο κεφάλαιο παρουσιάζονται θεωρητικά οι μέθοδοι που εξετάστηκαν για την εύρεση της περιοχής ματιών καθώς και οι συναρτήσεις και οι αλγόριθμοι για την εύρεση της ίριδας. Στο κεφάλαιο ‘Εύρεση ίριδας χωρίς περιορισμό εικόνας’ γίνεται η απόπειρα εύρεσης ίριδας σε εικόνες οι οποίες δεν έχουν περιοριστεί, και αναφέρονται τα προβλήματα που παρουσιάζονται. Στο κεφάλαιο ‘Περιορισμός εικόνας σε περιοχή ματιών’ παρουσιάζονται πειραματικά αποτελέσματα των μεθόδων για την επίτευξη του περιορισμού της εικόνας σε σημεία μεγαλύτερης σημασίας για την εύρεση ίριδας. Στο επόμενο κεφάλαιο παρουσιάζονται τα αποτελέσματα συναρτήσεων και αλγόριθμών για την εύρεση της ίριδας. Στη συνέχεια περιγράφεται η ανάπτυξη μιας εφαρμογής όπου ο κέρσορας κινείται ανάλογα με τη κίνηση τις ίριδας.

2 OpenCV

Για την ολοκλήρωση της εφαρμογής ανίχνευσης ίριδας χρησιμοποιήθηκε η βιβλιοθήκη OpenCV. Η OpenCV είναι μια βιβλιοθήκη τεχνητής όρασης ανοικτού κώδικα η οποία είναι διαθέσιμη στην ιστοσελίδα της SourceForge, όπου υπάρχουν διάφορες εκδόσεις της. Η Βιβλιοθήκη είναι γραμμένη σε C και C++ και τρέχει σε Linux, Windows και MacOS. Είναι ενεργή η εξέλιξη διεπαφών για Python, Ruby, Matlab και άλλες γλώσσες. Η OpenCV σχεδιάστηκε ώστε να έχει υψηλή υπολογιστική απόδοση με στραμμένο το ενδιαφέρον στις εφαρμογές πραγματικού χρόνου. Η βιβλιοθήκη μπορεί να εκμεταλλευτεί τις δυνατότητες των νέων πολυπύρηνων επεξεργαστών. Εάν υπάρχει επιθυμία για βελτιστοποίηση της OpenCV υπάρχει δυνατότητα αγοράς της βιβλιοθήκης IPP (Intergraded Performance Primitives library) της Intel, η οποία αποτελείται από πιστοποιημένες υπορουτίνες χαμηλού επιπέδου που βελτιώνουν κατά πολύ την απόδοση της. Η OpenCV αυτόματα χρησιμοποιεί την βιβλιοθήκη IPP κατά το run-time εφόσον έχει εγκατασταθεί. Ένας από τους στόχους της OpenCV είναι να κάνει διαθέσιμη μια απλή στη χρήση υποδομή τεχνητής όρασης, η οποία θα βοηθά στο να κατασκευάζονται εξεζητημένες εφαρμογές γρήγορα. Η OpenCV διαθέτει πάνω από 500 συναρτήσεις οι οποίες αφορούν πολλά πεδία της τεχνητής όρασης περιλαμβάνοντας: εργοστασιακό έλεγχο προϊόντων, ιατρική όραση, ασφάλεια ,διεπαφή χρήστη, βαθμονόμηση κάμερας, στέρεο όραση, ρομποτική κτλ. Επειδή η τεχνητή όραση και η μάθηση μηχανής συχνά πάνε χέρι-χέρι, η OpenCV επίσης διαθέτει μια πλήρης γενικού σκοπού βιβλιοθήκη εκμάθησης μηχανής, Machine Learning Library (MLL), η οποία επικεντρώνεται στην στατιστική αναγνώριση προτύπων και στο clustering. Η MML είναι χρήσιμη για τις διαδικασίες όρασης που βρίσκονται στον πυρήνα της OpenCV αλλά μπορεί να χρησιμοποιηθεί και για οποιοδήποτε πρόβλημα εκμάθησης μηχανής.

2.1 Χρήστες της OpenCV

Οι περισσότεροι επιστήμονες της πληροφορικής και οι προγραμματιστές γνωρίζουν κάποια πτυχή του ρόλου της τεχνητής όρασης, αλλά λίγοι είναι εκείνοι γνωρίζουν όλους τους τρόπους με τον οποίο μπορεί να χρησιμοποιηθεί η τεχνική όραση. Πολλοί γνωρίζουν τη χρήση της στο πεδίο της ασφάλειας ή ότι χρησιμοποιείται όλο και περισσότερο σε εικόνες και βίντεο στο Web, κάποιοι ίσως έχουν δει χρήση τεχνητής όρασης σε διεπαφή παιχνιδιού. Οι περισσότερες εναέριες και street-map εικόνες (όπως το Street View της Google), κάνουν χρήση βαθμονόμησης κάμερας και τεχνικών συρραφής εικόνων. Υπάρχουν εξειδικευμένες εφαρμογές στην παρακολούθηση ασφάλειας, σε μη επανδρωμένα αεροπλάνα, στην βιοϊατρική ανάλυση. Σχεδόν κάθε τι που είναι μαζικής παραγωγής έχει αυτόματα επιθεωρηθεί κάποια στιγμή με την χρησιμοποίηση τεχνητής όρασης.

Η άδεια ανοικτού κώδικα της OpenCV έχει δομηθεί έτσι ώστε μπορεί να φτιαχτεί ένα εμπορικό προϊόν με το σύνολο ή μέρος της OpenCV. Εν μέρει λόγω αυτών των φιλελεύθερων όρων αδειοδότησης, υπάρχει μια μεγάλη κοινότητα χρηστών που περιλαμβάνει ανθρώπους από μεγάλες εταιρείες όπως οι IBM, Microsoft, Intel, SONY, Siemens, και Google, και ερευνητικά κέντρα όπως τα Stanford, MIT, CMU, Cambridge, INRIA. Επίσης υπάρχει το Yahoo Group φόρουμ της OpenCV στο οποίο οι χρήστες μπορούν να δημοσιεύσουν ερωτήσεις και να συζητήσουν και έχει περίπου 20.000 μέλη. Το

OpenCV είναι δημοφιλές σε όλο τον κόσμο, με μεγάλες κοινότητες χρηστών στην Κίνα, την Ιαπωνία, την Ρωσία, και την Ευρώπη.

Από την alpha έκδοση του Ιανουαρίου του 1999, η OpenCV έχει χρησιμοποιηθεί σε πολλές εφαρμογές, προϊόντα, και έρευνες. Οι εφαρμογές περιλαμβάνουν συρραφή εικόνων από δορυφόρους και web χάρτες, ευθυγράμμιση σαρωμένων εικόνων, μείωση του θορύβου σε ιατρικές εικόνες, ανάλυση αντικειμένων, συστήματα ασφάλειας και ανίχνευσης διείσδυσης, αυτόματα συστήματα παρακολούθησης, συστήματα ελέγχου παραγωγής, βαθμονόμηση κάμερας, στρατιωτικές εφαρμογές, καθώς και σε μη επανδρωμένα εναέρια, επίγεια και υποβρύχια οχήματα. Έχει ακόμη χρησιμοποιηθεί στην αναγνώριση ήχων και μουσικής, όπου οι τεχνικές αναγνώρισης εικόνων εφαρμόζονται στο φασματογράφημα του ήχου. Η OpenCV αποτελούσε ένα βασικό μέρος του συστήματος όρασης στο ρομπότ του Stanford «Stanley», το οποίο κέρδισε το βραβείο των δυο εκατομμυρίων δολαρίων στον αγώνα DARPA Grand Challenge Desert Robot Race.

2.2 Η προέλευση της OpenCV

Η OpenCV γεννήθηκε μέσα από μια έρευνα, πρωτοβουλία της Intel, για την προώθηση εφαρμογών με πολύ υψηλές απαιτήσεις από την CPU. Για τον σκοπό αυτό, η Intel εγκαινίασε πολλές εφαρμογές, συμπεριλαμβανομένων της ανίχνευσης ακτίνων σε πραγματικό χρόνο και τοίχων 3D απεικόνισης. Ένας από τους συντάκτες που εργάζονταν για την Intel εκείνη την εποχή επισκεπτόταν πανεπιστήμια και παρατήρησε ότι ορισμένες κορυφαίες ομάδες πανεπιστημίων, όπως το MIT Media Lab, είχε καλά ανεπτυγμένες και εσωτερικά ανοικτές υποδομές κώδικα τεχνητής όρασης που μεταβιβάζονταν από φοιτητή σε φοιτητή και οι οποίες έδιναν σε κάθε φοιτητή, ένα πολύτιμο προβάδισμα για την ανάπτυξη της δικής του εφαρμογής τεχνητής όρασης. Αντί να ανακαλύπτει τις βασικές λειτουργίες από το μηδέν, ένας νέος φοιτητής ξεκινούσε με προϋπάρχουσα πληροφορία και έτοιμο υλικό.

Η OpenCV σχεδιάστηκε ως ένας τρόπος για να γίνουν οι υποδομές τεχνητής όρασης παγκοσμίως διαθέσιμες. Με την ενίσχυση της ομάδας βιβλιοθήκης επιδόσεων της Intel, η OpenCV ξεκίνησε σαν ένας πυρήνας εφαρμόσιμου κώδικα και αλγοριθμικών προδιαγραφών που στέλνονταν στα μέλη της ρωσικής ομάδας βιβλιοθήκης της Intel. Αυτό είναι το «που» της OpenCV: ξεκίνησε στα εργαστήρια ερευνών της Intel με τη συνεργασία της ομάδας βιβλιοθήκης επιδόσεων της Intel μαζί με τις εφαρμογές και την βελτιστοποίηση από εμπειρογνώμονες στη Ρωσία.

Υπήρξαν πολλοί στόχοι για OpenCV από την αρχή:

• Προηγμένη έρευνα τεχνητής όρασης, παρέχοντας όχι μόνο ελεύθερο αλλά και βελτιστοποιημένο κώδικα για βασικές υποδομές.

• Διάδοση τη γνώσης, παρέχοντας μια κοινή υποδομή πάνω στην οποία οι προγραμματιστές μπορούν να χτίσουν, έτσι ώστε ο κώδικας να είναι πιο εύκολα αναγνώσιμος και μεταβιβάσιμος.

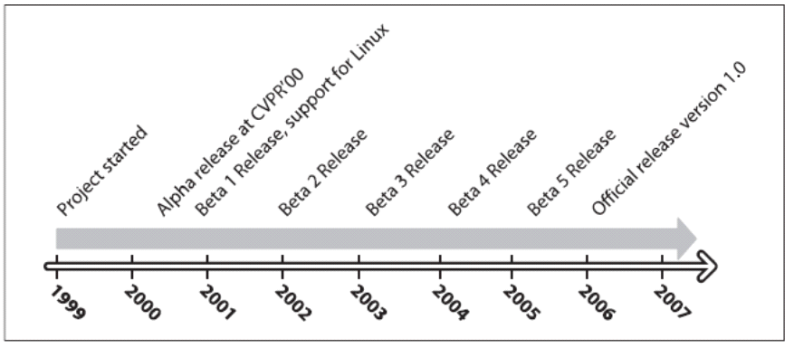
• Προηγμένες εμπορικές εφαρμογές τεχνητής όρασης, μέσω της διάθεσης φορητού, και βελτιστοποιημένου στην απόδοση δωρεάν κώδικα, με μια άδεια που δεν θα απαιτεί οι εμπορικές εφαρμογές να είναι δωρεάν ή ανοιχτού κώδικα.

Αυτοί οι στόχοι αποτελούν το «γιατί» της OpenCV. Οι εφαρμογές τεχνητής όρασης αυξάνουν την ανάγκη για γρήγορους επεξεργαστές. Η ανάγκη για αναβαθμίσεις με ταχύτερους επεξεργαστές παράγουν περισσότερο εισόδημα για την Intel από την πώληση κάποιου software. Ίσως για αυτό, ο ανοικτός και ελεύθερος κώδικας προέκυψε από έναν

προμηθευτή υλικού και όχι από μία εταιρία software. Υπό κάποια έννοια, υπάρχει περισσότερος χώρος για καινοτομίες στο software στο εσωτερικό μιας εταιρίας υλικού.

Σε οποιαδήποτε προσπάθεια ανοικτού κώδικα, είναι σημαντικό να επιτευχθεί μία μάζα χρηστών με την οποία το έργο θα καταστεί αύταρκες. Μέχρι στιγμής έχουν γίνει περίπου δύο εκατομμύρια downloads της OpenCV και ο αριθμός αυτός αυξάνεται κατά μέσο όρο με 26.000 downloads το μήνα. Η ομάδα χρηστών προσεγγίζει τώρα τα 20.000 μέλη. Η OpenCV λαμβάνει πολλές συνεισφορές από τους χρήστες, και η ανάπτυξη της έχει σε μεγάλο βαθμό μετακινηθεί εκτός της Intel.

Στην πορεία, η OpenCV επηρεάστηκε από την ραγδαία εξέλιξη του internet, καθώς και από τις πολλές αλλαγές στην διαχείριση και την κατεύθυνση της. Κατά τη διάρκεια αυτών των μεταβολών, υπήρξαν φορές που δεν υπήρχε κανένας στην Intel να εργάζεται πάνω σε αυτήν. Ωστόσο, με την έλευση των πολυπύρηνων επεξεργαστών και τις πολλές νέες εφαρμογές της τεχνητής όρασης, η αξία της OpenCV άρχισε να αυξάνεται. Σήμερα, η OpenCV είναι ένας ενεργός τομέας ανάπτυξης σε διάφορους οργανισμούς, και έτσι αναμένεται να δούμε πολλές ενημερώσεις π.χ. στη ταυτόχρονη βαθμονόμηση πολλών καμερών, στην αντίληψη βάθους, στις μεθόδους για την ανάμειξη τεχνητής όρασης με ανιχνευτές πεδίου λέιζερ, στην αναγνώριση προτύπων, καθώς και μεγαλύτερη υποστήριξη για τις ανάγκες της ρομποτικής όρασης.



2-1 Ανάπτυξη της Opencv χρονολογικά.

2.3 Επιτάχυνση OpenCV με IPP

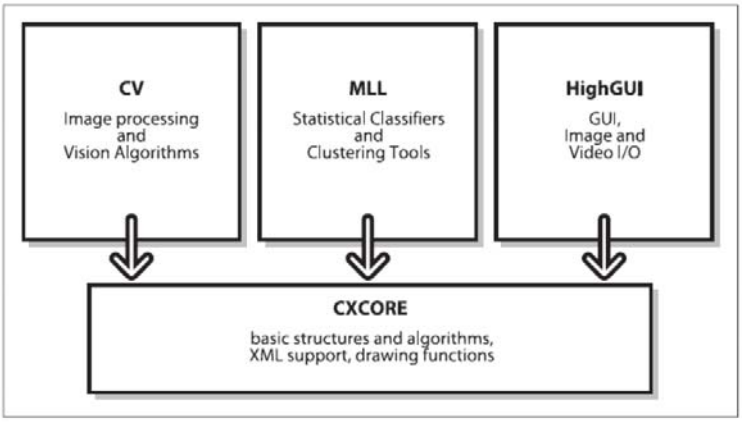
Επειδή η OpenCV «στεγάζονταν» στο πλαίσιο της ομάδας Intel Performance Primitives και αρκετοί από τους πρώτους προγραμματιστές έχουν φιλικές σχέσεις με αυτή την ομάδα, η OpenCV εκμεταλλεύεται τον ιδιαίτερα βελτιστοποιημένο κώδικα της IPP ώστε να επιταχυνθεί. Στο σχήμα φαίνεται η σύγκριση δύο άλλων βιβλιοθηκών τεχνητής όρασης, της LTI και της VXL, σε σχέση με την OpenCV και την OpenCV με IPP. Σημειώστε ότι οι επιδόσεις είναι ένας βασικός στόχος της OpenCV ώστε να μπορεί να εκτελεί κώδικα τεχνητής όρασης σε πραγματικό χρόνο.

Η OpenCV είναι γραμμένη σε C και C++ κώδικα για βέλτιστη απόδοση. Δεν εξαρτάται με οποιονδήποτε τρόπο από την IPP. Αν η IPP είναι παρούσα η OpenCV θα τη χρησιμοποιήσει αυτόματα με τη φόρτωση των βιβλιοθηκών δυναμικής σύνδεσης της IPP για να ενισχύσει περαιτέρω την ταχύτητά της.

2.4 Δομή και περιεχόμενο

Η OpenCV σε γενικές γραμμές διαρθρώνεται σε πέντε κύρια μέρη, τέσσερα από τα οποία παρουσιάζονται στο σχήμα. Η CV περιέχει την βασική επεξεργασία εικόνας και τους αλγόριθμους τεχνητής όρασης υψηλότερου επιπέδου. Η ML είναι η βιβλιοθήκη μάθησης μηχανής, που περιλαμβάνει πολλούς στατιστικούς ταξινομητές και εργαλεία ομαδοποίησης(clustering). Η HighGUI περιέχει ρουτίνες εισόδου/εξόδου και τις

συναρτήσεις για την αποθήκευση και φόρτωση βίντεο και εικόνων και η CXCore περιέχει τις βασικές δομές δεδομένων και το περιεχόμενο. Επίσης υπάρχει η CvAux, η οποία περιέχει περιοχές που δεν χρησιμοποιούνται (ενσωματωμένο HMM αναγνώρισης προσώπου) αλλά και πειραματικούς αλγόριθμους (κατάτμηση φόντου, πρόσοψης).



2-2 Βιβλιοθήκες της OpenCV. Το Σχήμα δεν περιλαμβάνει την CvAux, η οποία περιέχει τόσο περιοχές που δεν χρησιμοποιούνται (ενσωματωμένο HMM αναγνώρισης προσώπου) αλλά και πειραματικούς αλγόριθμους (κατάτμηση φόντου, πρόσοψης).

2.5 Φορητότητα

Η OpenCV σχεδιάστηκε για να είναι φορητή. Γράφτηκε αρχικά για compilation σε BorlandC++, MSVC++ και για τους compilers της Intel. Εννοείται ότι ο C και C++ κώδικας έπρεπε να είναι αρκετά 'στάνταρντ' για να κάνει την υποστήριξη πολλών πλατφορμών ευκολότερη. Η υποστήριξη για 32-bit αρχιτεκτονική Intel (IA32) στα Windows είναι η πιο ώριμη, ακολουθούμενη από τα Linux στην ίδια αρχιτεκτονική. Η φορητότητα σε Mac OS X έγινε προτεραιότητα μόνο αφού η Apple άρχισε να χρησιμοποιεί επεξεργαστές Intel (το port του OS X δεν είναι τόσο ώριμο όσο αυτό των Windows και των Linux, αλλά αυτό είναι ένα γεγονός που αλλάζει ραγδαία). Αυτές ακολουθούνται από την υποστήριξη 64-bit για εκτεταμένη μνήμη (EM64T), καθώς και την 64-bit αρχιτεκτονική της Intel (IA64). Μετά ακολουθούν τα SUN και άλλα λειτουργικά συστήματα. Αν μια αρχιτεκτονική ή OS δεν αναφέρθηκε, αυτό δεν σημαίνει ότι δεν υπάρχουν OpenCV ports για αυτά. Η OpenCV έχει μεταφερθεί σχεδόν σε κάθε εμπορικό σύστημα, από PowerPC Macs μέχρι και σε ρομποτικούς σκύλους. Η OpenCV λειτουργεί καλά με AMD επεξεργαστές, και η IPP μπορεί επωφεληθεί από τις επεκτάσεις πολυμέσων (MMX) σε επεξεργαστές AMD που ενσωματώνουν την τεχνολογία αυτή [1].

3 Συναρτήσεις και μέθοδοι που χρησιμοποιήθηκαν κατά την ανάπτυξη της εφαρμογής

3.1 Εισαγωγή

Στο κεφάλαιο αυτό παρουσιάζονται θεωρητικά οι συναρτήσεις οι τεχνικές που χρησιμοποιήθηκαν κατά την ανάπτυξη της εφαρμογής. Συγκεκριμένα παρουσιάζεται ο τρόπος εκπαίδευσης και λειτουργίας του Haar Cascade, γίνεται η επεξήγηση του αλγόριθμου για την επίτευξη του Blink Detection και τελικά οι αναφέρονται απαραίτητες συναρτήσεις και αλγόριθμοι για την εύρεση ίριδας.

3.2 Haar Classifier

Ο Haar classifier είναι μια τεχνική της OpenCV βασισμένη σε δέντρα η οποία αναπτύχθηκε ως μια ολοκληρωμένη εφαρμογή αναγνώρισης προσώπου. Παρόλα αυτά, μπορεί να εκπαιδευτεί για να αναγνωρίζει και άλλα αντικείμενα.

Η όραση υπολογιστών είναι ένας ευρύς και γρήγορα εξελισσόμενος τομέας, έτσι μέρη της OpenCV που εφαρμόζουν μία συγκεκριμένη τεχνική έχουν κίνδυνο να μην είναι εκσυγχρονισμένα. Ωστόσο ο haar classifier λειτουργεί αρκετά καλά και έχει χρησιμοποιηθεί από πολλές εταιρίες για την ανίχνευση σχετικά άκαμπτων αντικειμένων (π.χ. πρόσωπα, αυτοκίνητα, ποδήλατα), με την εκπαίδευση ανιχνευτών με χιλιάδες εικόνες που περιέχουν το αντικείμενο προς εκμάθηση. Αυτή η τεχνική έχει χρησιμοποιηθεί για να δημιουργηθούν ανιχνευτές, μίας όψης του αντικειμένου, πολύ υψηλής ποιότητας.

Η OpenCV εφαρμόζει μια έκδοση της τεχνικής ανίχνευσης προσώπου που αναπτύχθηκε αρχικά από τους Paul Viola και Michel Jones γνωστή ως Viola- Jones detector και αργότερα επεκτάθηκε από τους Rainer Lienhart και Jochen Maydt ώστε να χρησιμοποιεί διαγώνια χαρακτηριστικά. Η OpenCV αναφέρεται σε αυτόν τον ανιχνευτή ως "Haar classifier" επειδή χρησιμοποιεί Haar χαρακτηρίστηκα ή ακριβέστερα, όμοια με Haar μικρά κύματα (Haar-like) αποτελούμενα από την προσθήκη και αφαίρεση ορθογώνιων περιοχών εικόνας πριν από την εκτέλεση του threshold. Η OpenCV παρέχεται με ένα σύνολο ήδη εκπαιδευμένων αρχείων αναγνώρισης αντικειμένων, αλλά ο κώδικας επιτρέπει την εκπαίδευση και αποθήκευση νέων μοντέλων για τον ανιχνευτή. Να τονίσουμε ότι ο κώδικας για την εκπαίδευση createsamples(), haartraining() και την ανίχνευση cvHaarDetectObjects() λειτουργεί καλά σε οποιαδήποτε συμπαγή αντικείμενα, όχι μόνο πρόσωπα, των οποίων η υφή τους είναι συνεχής (έχουν ομοιόμορφη επιφάνεια).

3.2.1 Θεωρία εποπτευμένου Learning και Boosting

Ο Haar classifier της OpenCV είναι ένας εποπτευόμενος ταξινομητής. Τυπικά ως είσοδο στον ταξινομητή παρουσιάζουμε κομμάτια εικόνας με όμοιο ιστόγραμμα και μέγεθος, τα οποία μετά χαρακτηρίζονται για το αν περιέχουν η όχι το αντικείμενο ενδιαφέροντος το οποίο για το συγκεκριμένο ταξινομητή συνήθως είναι ένα πρόσωπο.

Ο ανιχνευτής Viola- Jones χρησιμοποιεί μια μορφή του AdaBoost αλλά το οργανώνει ως ένα cascade απόρριψης από τους κόμβους, όπου κάθε κόμβος είναι ένα πολύδεντρο AdaBoost classifier με σκοπό να έχει υψηλό ποσοστό ανίχνευσης (χαμηλά ψεύτικα αρνητικά, ή χαμένα πρόσωπα) με το κόστος ενός χαμηλού (κοντά 50%) ποσοστό απόρριψης (υψηλά ψεύτικα θετικά, ή "non-faces" λανθασμένα ταξινομημένα). Για κάθε έναν κόμβο, ένα αποτέλεσμα "nonclass" σε οποιοδήποτε στάδιο του cascade τερματίζει τον υπολογισμό, και ο αλγόριθμος έπειτα δηλώνει ότι κανένα πρόσωπο δεν υπάρχει σε εκείνη την θέση. Κατά συνέπεια, αληθινή ανίχνευση κατηγορίας δηλώνεται μόνο εάν ο υπολογισμός περάσει μέσω ολόκληρου καταρράκτη. Για τις περιπτώσεις όπου η αληθινή κατηγορία είναι σπάνια (π.χ., ένα πρόσωπο σε μια εικόνα), οι καταρράκτες απόρριψης μπορούν να μειώσουν πολύ το συνολικό υπολογισμό επειδή οι περισσότερες περιοχές που διερευνούνται για ένα πρόσωπο τερματίζονται γρήγορα με μια απόφαση nonclass.

3.2.2 Boosting classifiers

Στον καταρράκτη απόρριψης των Viola- Jones, οι αδύναμοι ταξινομητές που ενισχύουν (boost) σε κάθε κόμβο είναι δέντρα απόφασης συχνά ενός μόνο επίπεδου. Τους επιτρέπεται μια απόφαση μόνο της μορφής: "Η τιμή v ενός χαρακτηριστικού γνωρίσματος f βρίσκεται πάνω ή κάτω από το όριο t " . Απόφαση "πάνω" σημαίνει ότι υπάρχει αντικείμενο, ενώ απόφαση "κάτω" σημαίνει ότι δεν υπάρχει αντικείμενο:

$$f_i = \begin{cases} +1 & v_i \geq t_i \\ -1 & v_i < t_i \end{cases}$$

3-1 Haar classifier. Επιλογή weak classifier

Ο αριθμός των Haar-like features που χρησιμοποιούνται από τον Viola-Jones καταρράκτη σε κάθε αδύναμο ταξινομητή μπορεί να οριστεί στην εκπαίδευση, αλλά συνήθως χρησιμοποιούνται ένα έως τρία το πολύ χαρακτηριστικά γνωρίσματα. Με το boosting έπειτα μέσω επαναλήψεων ενισχύονται οι αδύναμοι ταξινομητές χρησιμοποιώντας τη συνάρτηση :

$$F = \text{sign}(w_1 f_1 + w_2 f_2 + \cdots + w_n f_n)$$

3-2 Haar classifier. Συνάρτηση επιλογής boosting

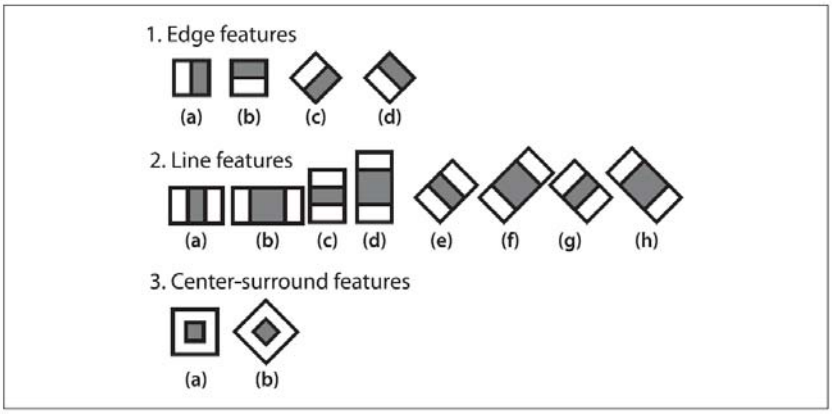
Η συνάρτηση επιστρέφει -1 εάν ο αριθμός είναι αρνητικός, 0 εάν ο αριθμός είναι ίσος με 0, και +1 εάν ο αριθμός είναι θετικός. Στο πρώτο πέρασμα μέσω του συνόλου στοιχείων, μαθαίνουμε το όριο $t1$ του γνωρίσματος f 1 που ταξινομεί καλύτερα την εισαγωγή. Το boosting έπειτα χρησιμοποιεί τα προκύπτοντα λάθη για τον υπολογισμό του βάρους $w1$. Κάθε διάνυσμα γνωρίσματος αποκτά καινούργιο βάρος χαμηλό ή υψηλό σύμφωνα με το εάν ήταν ταξινομημένο σωστά ή όχι σε εκείνη την επανάληψη του ταξινομητή. Μόλις ένας κόμβος εκπαιδευτεί με αυτόν τον τρόπο, τα επιζώντα στοιχεία από τα υψηλότερα σημεία του καταρράκτη χρησιμοποιούνται για να εκπαιδεύσουν τον επόμενο κόμβο. Η διαδικασία συνεχίζεται με αυτόν τον τρόπο μέχρι να ολοκληρωθεί η εκπαίδευση.

3.2.3 Viola-Jones Classifier Theory

Ο ταξινομητής των Viola-Jones υιοθετεί AdaBoost σε κάθε κόμβο του καταρράκτη για να αποκτήσει υψηλό ποσοστό ανίχνευσης με το κόστος ενός πολύδεντρου ταξινομητή χαμηλού ποσοστού απόρριψης σε κάθε κόμβο. Ο αλγόριθμος αυτός ενσωματώνει διάφορα καινοτόμα χαρακτηριστικά γνωρίσματα.

1. Χρησιμοποιεί Haar-like χαρακτηριστικά εισαγωγής: ένα threshold χρησιμοποιείται στα αθροίσματα και τις διαφορές ορθογώνιων περιοχών εικόνας.
2. Η τεχνική integral image της επιτρέπει το γρήγορο υπολογισμό της αξίας ορθογώνιων περιοχών ή τέτοιων περιοχών περιστρεμμένων κατά 45 μοίρες. Αυτή η δομή δεδομένων χρησιμοποιείται για να επιταχύνει τους υπολογισμούς των Haar-like χαρακτηριστικών της εισαγωγής.
3. Χρησιμοποιεί στατιστικό boosting για τη δημιουργία δυαδικών (face, non-face) ταξινομητών κόμβων που χαρακτηρίζονται από υψηλά ποσοστά ανίχνευσης και χαμηλά ποσοστά απόρριψης.
4. Οργανώνει τους αδύνατους κόμβους ταξινόμησης ενός καταρράκτη απόρριψης. Με άλλα λόγια: η πρώτη ομάδα ταξινομητών που επιλέγεται, ανιχνεύει καλύτερα περιοχές εικόνας που περιέχουν ένα αντικείμενο επιτρέποντας πολλές λανθασμένες ανιχνεύσεις, η επόμενη ομάδα ταξινομητών είναι η δεύτερη καλύτερη στην ανίχνευση με χαμηλό ποσοστό απόρριψης και ούτω καθ' εξής. Ένα αντικείμενο ανιχνεύεται μόνο εάν περάσει μέσω ολόκληρου του καταρράκτη.

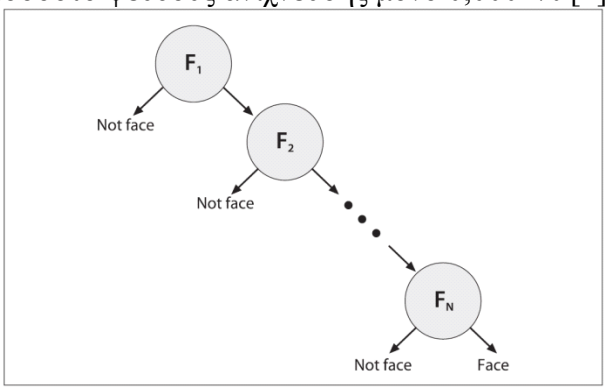
Τα Haar-like features που χρησιμοποιούνται από τον classifier παρουσιάζεται στο παρακάτω σχήμα. Σε όλες τις κλίμακες, αυτά τα χαρακτηριστικά γνωρίσματα διαμορφώνουν τη "πρώτη ύλη" που θα χρησιμοποιηθεί από τους boosted classifiers. Αυτοί υπολογίζονται γρήγορα από την integral image.



3-3 Haar classifier. Τα Haar-like features από τη διανομή πηγής OpenCV (η ορθογώνια περιοχή και οι περιστρεμμένες περιοχές υπολογίζονται εύκολα από την integral image): σε αυτήν την διαγραμματική αντιπροσώπευση των wavelets, η λευκή περιοχή ερμηνεύεται "προσθέστε αυτήν τη περιοχή" και η σκοτεινή περιοχή "αφαιρέστε αυτήν την περιοχή ".

Οι Viola και Jones οργάνωσαν κάθε boosted ομάδα ταξινομητών σε κόμβους ενός cascade απόρριψης, όπως φαίνεται στο επόμενο σχήμα. Κάθε ένας από τους κόμβους F περιέχει ένα ολόκληρο boosted καταρράκτη αποτελούμενο από ομάδες δέντρων απόφασης που εκπαιδεύτηκαν στα Haar-like χαρακτηριστικά γνωρίσματα προσώπων (ή άλλων αντικειμένων). Οι κόμβοι διατάσσονται από λιγότερο σε περισσότερο σύνθετους έτσι ώστε οι υπολογισμοί να ελαχιστοποιούνται (οι απλοί κόμβοι δοκιμάζονται πρώτα) κατά την απόρριψη των εύκολων περιοχών της εικόνας. Το boosting σε κάθε κόμβο είναι ρυθμισμένο ώστε να έχει ένα πολύ υψηλό ποσοστό ανίχνευσης. Κατά την εκπαίδευση στα πρόσωπα, σχεδόν όλα (99,9%) τα πρόσωπα βρίσκονται, αλλά πολλά (περίπου 50%) “μη

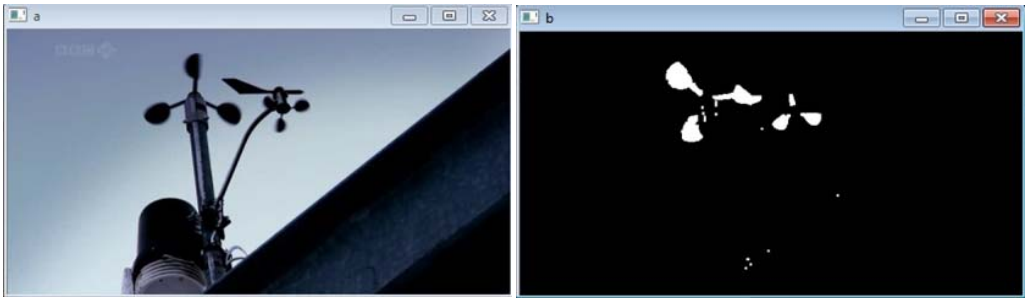
πρόσωπα" ανιχνεύονται λανθασμένα σε κάθε κόμβο. Αλλά αυτό δεν έχει σημασία επειδή η χρησιμοποίηση για παράδειγμα 20 κόμβων θα παράγει ένα ποσοστό ανίχνευσης προσώπου 98% με ένα ποσοστό ψευδούς ανίχνευσης μόνο 0,0001% [1].



3-.4 *Haar classifier.* Οργάνωση boosted ταξινομητή

3.3 Blink Detection

Ένας από τους τρόπους εύρεσης μίας περιοχής η οποία περιέχει την ίριδα είναι μέσω ανίχνευσης της κίνησης του βλέφαρου. Στην παρακάτω εικόνα βλέπουμε την έξοδο του προγράμματος. Στο παράθυρο a φαίνεται ένα στιγμιότυπο από βίντεο που έχουμε ως πηγή, ενώ στο παράθυρο b το αποτέλεσμα από την ανίχνευση κίνησης μέσω της χρησιμοποίησης της βιβλιοθήκης της OpenCV.



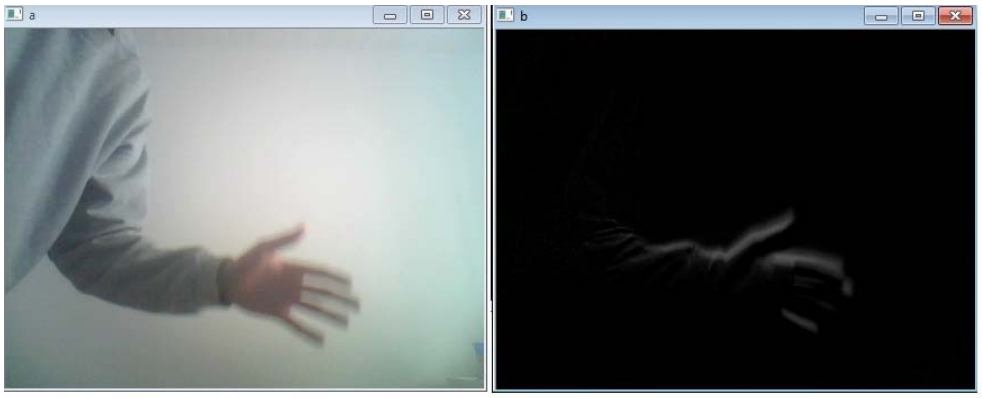
3-5 Blink Detection. Ανίχνευση κίνησης. Στο παράθυρο b φαίνεται η κίνηση που ανιχνεύτηκε από το βίντεο a.

Όπως βλέπουμε στις εικόνες η κίνηση δηλαδή οι μεταβολές ανάμεσα στα frame απεικονίζονται με άσπρο χρώμα, ενώ οι περιοχές που μένουν αμετάβλητες, εμφανίζονται με μαύρο χρώμα.

3.3.1 Συνάρτηση cvSub()

Η συνάρτηση cvSub() της OpenCV αφαιρεί τα pixel μιας εικόνας από μία άλλη. Για τη περίπτωση όπου οι εικόνες είναι δύο διαδοχικά frames ενός βίντεο το αποτέλεσμα περιέχει την κίνηση. Η ταχύτητα της κίνησης καθορίζει τον αριθμό των άσπρων pixels. Στην παρακάτω εικόνα φαίνονται περισσότερο στην εικόνα της διαφοράς των frames τα δάχτυλα, τα οποία έχουν αποκτήσει μεγαλύτερη ταχύτητα από τον αγκώνα παρόλο που και

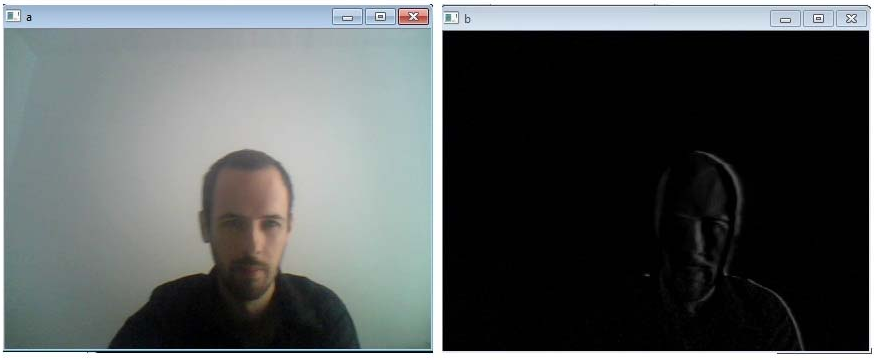
τα δύο μέλη του σώματος κινούνται. Επίσης φαίνονται περισσότερο τα περιγράμματα των κινούμενων αντικειμένων και χωρίς να είναι ξεκάθαρα όπως θα ήταν επιθυμητό .



3-6 Blink Detection. Διαφορά εικόνων. Το στην εικόνα b φαίνεται το αποτέλεσμα της διαφοράς δύο διαδοχικών frame.

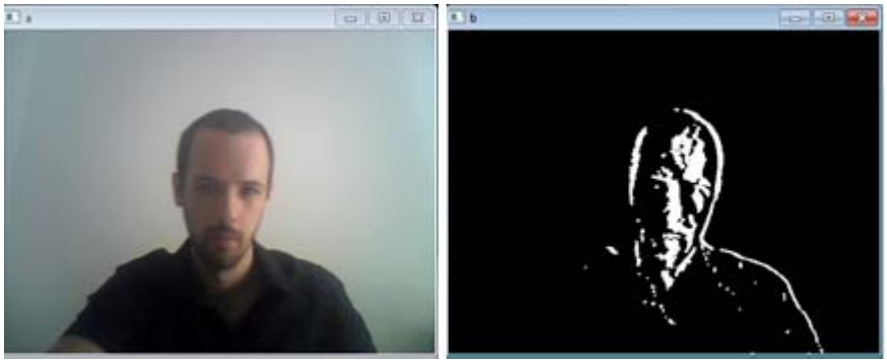
3.3.2 Συνάρτηση cvThreshold()

Επικεντρώνοντας την προσοχή μας στην ανίχνευση των ματιών το πρώτο θέμα που προκύπτει έχει να κάνει με το ότι, όπως προαναφέραμε, στην εικόνα της διαφοράς το αποτέλεσμα είναι πολύ θολό με αποτέλεσμα να μην είναι ευδιάκριτα τα χαρακτηριστικά του προσώπου όπως φαίνεται στην παρακάτω εικόνα. Αυτό δυσκολεύει τον εντοπισμό της συγκεκριμένης θέσης των ματιών εφόσον δεν αποκτήσει πιο ξεκάθαρες γραμμές η εικόνα της διαφοράς, αφού δεν θα είναι δυνατόν να ξεχωριστούν τα μάτια δίχως την χρησιμοποίηση τεχνικών που πιθανότατα θα επιβάρυναν υπερβολικά το πρόγραμμα με αποτέλεσμα να μην δουλεύει σε πραγματικό χρόνο (real- time). Για να αντιμετωπιστεί το θολό αυτό αποτέλεσμα χρησιμοποιήθηκε η συνάρτηση cvThreshold(), η οποία δίνει ως αποτέλεσμα άσπρα pixel για κάθε pixel μεγαλύτερης τιμής από ένα καθορισμένο όριο.



3-7 Blink Detection. Κίνηση χωρίς cvThreshold(). Στην εικόνα b, όπου απεικονίζεται η διαφορά των frames, δεν έχει εφαρμοστεί η συνάρτηση cvThreshold().

Η συνάρτηση αυτή μας δίνει το επιθυμητό αποτέλεσμα ,δηλαδή ξεκάθαρες γραμμές του προσώπου.



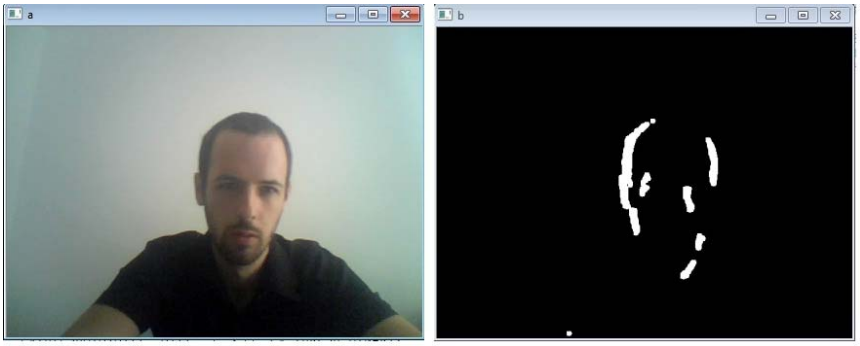
3-8 Blink Detection. Κίνηση σώματος κεφαλιού με threshold.

3.3.3 Συνάρτηση cvMorphologyEx() και CreateStructuringElementEx()

Χρησιμοποιώντας την συνάρτηση cvMorphologyEx μπορούμε να πειράξουμε την μορφολογία στην εικόνα της διαφοράς των frame με τρόπο τέτοιο, ώστε να πάρουμε τα επιθυμητά αποτελέσματα δηλαδή τα μάτια ως δύο ενιαία αντικείμενα.

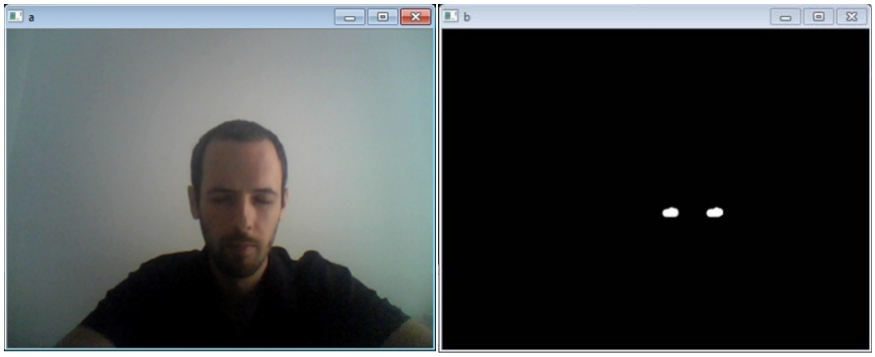
Η συνάρτηση CreateStructuringElementEx() δεσμεύει μνήμη και γεμίζει την δομή IplConvKernel, η οποία μπορεί να χρησιμοποιηθεί ως δομικό στοιχείο σε μορφολογικούς μετασχηματισμούς.

Ουσιαστικά αυτό που αλλάζει στην μορφολογία της εικόνας της διαφοράς των frame είναι η ενοποίηση των λεύκων pixel με μικρό μέγεθος με άλλα, έτσι ώστε να αποκτήσουν ενιαίο όγκο και στρογγυλή μορφή. Η στρογγυλή αυτή μορφή επιτυγχάνεται με την επιλογή ελλειπτικού δομικού στοιχείου. Το αποτέλεσμα αποθηκεύεται ως ένα ενιαίο αντικείμενο (contour). Στην εικόνα 3-9 παρατηρείται ότι σε σχέση με την εικόνα 3-8 που δείχνει και πάλι κίνηση κεφαλιού-σώματος, ότι πολλές από τις λεπτομέρειες χάνονται, αλλά τα μέρη που έχουν αρκετά μεγάλο όγκο παραμένουν με μεγαλύτερη πλέον περιφέρεια και έχουν στρογγυλοποιηθεί στις άκρες τους.



3-9 Blink Detection. Κίνηση σώματος κεφαλιού με cvMorphologyEx. Τα άσπρα σημεία πλέον λογίζονται ως ένα ενιαίο αντικείμενο.

Όταν ο χρήστης ανοιγοκλείσει τα μάτια του, λόγω του μετασχηματισμού μορφολογίας που χρησιμοποιείται παράγονται δύο ενιαία αντικείμενα (contour) για μάτια τα οποία μπορούμε με τον κατάλληλο κώδικα να τα χρησιμοποιήσουμε ώστε να βρεθεί η ίριδα. Για να ελεγχτεί αν τα contour είναι περιοχές ματιών και όχι οποιαδήποτε άλλη κίνηση, τα contour εξετάζονται για το αν είναι δύο, για το μέγεθος τους και την απόσταση τους στους x και y άξονες. Εφόσον πληρούν τις προϋποθέσεις που έχουν τεθεί τότε υποθέτουμε ότι τα αντικείμενα αυτά αντιπροσωπεύουν τα μάτια και πλέον συνεχίζουμε τις ενέργειες μας για να βρούμε συγκεκριμένα την ίριδα.

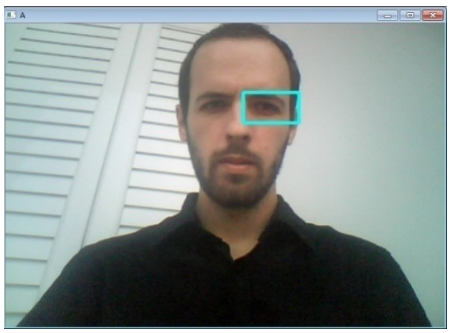


3-10 Blink Detection. Εύρεση contours ματιών. Αν τα contours έχουν παρόμοιο όγκο, συγκεκριμένη απόσταση x και βρίσκονται περίπου στο ίδιο ύψος y τότε θεωρούνται περιοχές ματιών.

3.3.4 Template Matching

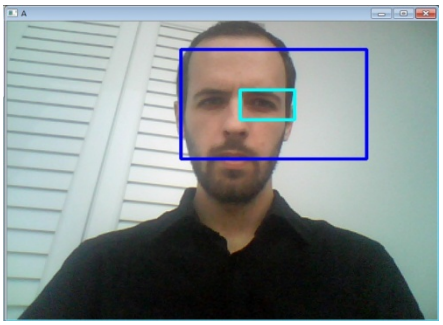
Αφού έχει βρεθεί πλέον η περιοχή των ματιών, αποθηκεύονται δεδομένα από την αρχική εικόνα στη περιοχή όπου βρέθηκε ένα contour (με μία σχετική καθυστέρηση έτσι ώστε τα μάτια να έχουν ανοίξει), και με χρήση template matching γίνεται το tracking της περιοχής αυτής.

Μέσω της συνάρτησης cvfindMinEnclosingRect() η οποία βρίσκεται το μικρότερο ορθογώνιο που περικλείει ένα contour, και βάση αυτού του ορθογωνίου δημιουργείται ένα ακόμη μεγαλύτερο ορθογώνιο το οποίο, χρησιμοποιείται για το template matching. Η συνάρτηση σκανάρει όλη την εικόνα προσπαθώντας να εντοπίσει μια περιοχή όμοια με αυτήν που περιέχει το ορθογώνιο που περιγράψαμε. Στην εικόνα βλέπουμε το περίγραμμα αυτού του ορθογώνιου με γαλάζιο χρώμα.



3-11 Blink Detection. Template matching. Στο γαλάζιο περίγραμμα φαίνεται το αποτέλεσμα του Template matching στην εικόνα.

Επειδή όμως η ανίχνευση συγκεκριμένου ορθογωνίου σε ολόκληρη την εικόνα δεν είναι πολύ αποδοτική δημιουργείται και ένα δεύτερο ορθογώνιο μεγαλύτερο από το πρώτο, στο οποίο θα γίνεται πλέον το template matching. Είναι σημαντικό το μέγεθος του νέου ορθογώνιου να είναι τέτοιο ούτως ώστε με τις συνήθεις κινήσεις του κεφαλιού να μην χαθεί το ορθογώνιο του ματιού. Στην παρακάτω εικόνα φαίνεται το νέο ορθογώνιο με μπλε περίγραμμα μέσα στο οποίο γίνεται το template matching [6].



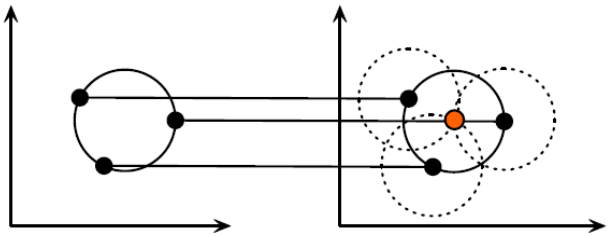
3-12 *Blink Detection.* Ανίχνευση σε περιορισμένη εικόνα. Πλέον η ανίχνευση γίνεται αποκλειστικά μέσα στο μπλε περίγραμμα.

3.4 cvHoughCricles()

Η συνάρτηση HoughCicles δίνει τη δυνατότητα εύρεσης κύκλων συγκεκριμένων ακτινών σε μία grayscale εικόνα χρησιμοποιώντας μία παραλλαγή του Hough transform. Η HoughCicles εκτελεί canny edge detection στην εικόνα για την ανίχνευση των ακμών.

3-13 cvHoughCircles(). Canny Edge Detection. Η δεύτερη εικόνα είναι το αποτέλεσμα ανίχνευσης ακμών.

Η Hough transform μπορεί να χρησιμοποιηθεί για να καθορίσει τις παραμέτρους ενός κύκλου όταν διάφορα σημεία που αφορούν την περίμετρο είναι γνωστά. Εφόσον οι ακτίνες των κύκλων που μας ενδιαφέρουν είναι γνωστές ο στόχος είναι να βρεθούν οι (x ,y) συντεταγμένες των κέντρων. Κάθε ακμή από το αποτέλεσμα του canny edge detector είναι ένα πιθανό περίγραμμα ενός κύκλου. Για κάθε λευκό pixel ζωγραφίζεται ένας νοητός κύκλος με ακτίνα R. Από ένα πραγματικό κέντρο ενός κύκλου με ακτίνα R είναι το κοινό pixel των νοητών κύκλων και υπολογίζονται από το συσσωρευτή threshold.



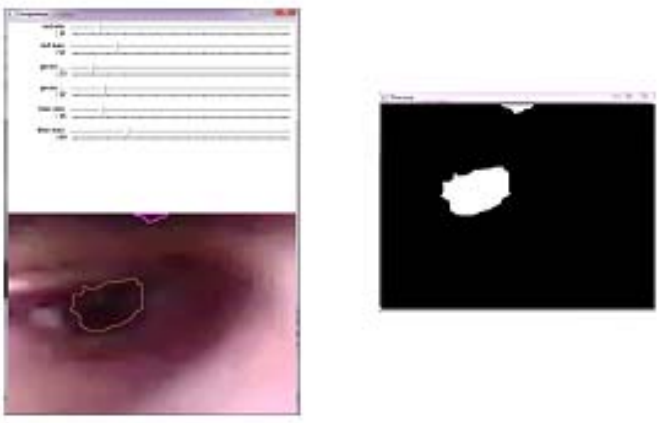
3-14 *cvHoughCircles()*. Ανίχνευση κύκλων. Το κοινό σημείο των νοητών κύκλων είναι ένα πραγματικό κέντρο κύκλου.

Όσο μικρότερη είναι η τιμή του συσσωρευτή threshold τόσο περισσότεροι εσφαλμένοι κύκλοι θα επιστραφούν από τη συνάρτηση. Οι κύκλοι αποθηκεύονται σε ένα storage μνήμης με βάση το πόσο καλά ανταποκρίθηκαν στο συσσωρευτή threshold [7].

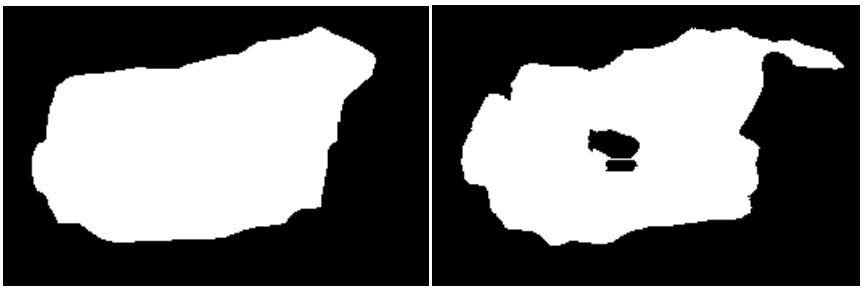
3.5 Δημιουργία Contour βάση χρώματος

Στη παρακάτω μέθοδο μία εικόνα RGB χωρίζεται σε τρείς grayscale, μία για κάθε χρωματικό κανάλι. Ο χρήστης μπορεί να επιλέξει ένα κατώτερο και ένα ανώτερο threshold για κάθε μία από τις grayscale εικόνες. Ο αλγόριθμος ελέγχει όλα τα pixel κάθε grayscale εικόνας. Αν ένα pixel σε θέση (x,y) μίας εικόνας έχει τιμή ανάμεσα στα δύο threshold και τα αντίστοιχα pixel των άλλων δύο εικόνων έχουν τιμή ανάμεσα στα δικά τους όρια, τότε το pixel (x,y) της εικόνας αποτελέσματος παίρνει τιμή 255(άσπρο). Διαφορετικά παίρνει τιμή 0.

Αφού ολοκληρωθεί ο έλεγχος, γίνεται μορφολογικός μετασχηματισμός (closing) στην εικόνα αποτελέσματος και στη συνέχεια ελέγχεται για contours. Τα όποια contours βρεθούν ζωγραφίζονται στην αρχική εικόνα [8].



3-15 Contour βάση χρώματος. Δημιουργία contour από επιλογή χρωματικού συνδυασμού. Τα contour είναι το αποτέλεσμα των χρωματικών επιλογών της πρώτης εικόνας.



3-16 Contour βάση χρώματος. Μορφολογικός μετασχηματισμός. Η δεύτερη εικόνα είναι το αποτέλεσμα της πρώτης μετά από μορφολογικό μετασχηματισμό closing.

3.6 Εύρεση σκοτεινότερου σημείου

Μία μέθοδος που δουλεύει πολύ καλά με εικόνα υψηλής ανάλυσης είναι η ανίχνευση του ποιο σκοτεινού σημείου στην εικόνα. Στη μέθοδο αυτή εκμεταλλευόμαστε το γεγονός του ότι η κόρη του ματιού στις περισσότερες περιπτώσεις αποτελεί το ποιο σκούρο σημείο του ματιού αλλά και της περιοχής γύρο από αυτό. Στην περίπτωση μας, όπου η εικόνα δεν είναι ιδιαιτέρα υψηλής ανάλυσης κάθε frame σκανάρεται ολόκληρο ώστε να βρεθεί το pixel με την πιο μικρή τιμή. Αφού αυτό βρεθεί, ξανασκανάρεται η εικόνα για να εντοπιστούν τα pixel τα οποία έχουν ίδια τιμή χρώματος με το pixel με τη μικρότερη τιμή χρώματος, για τη περίπτωση που υπάρχουν περισσότερα από ένα pixel με αυτή τη τιμή φωτεινότητας. Για να φανούν τα pixel αυτά απεικονίζονται με γκρίζο χρώμα.

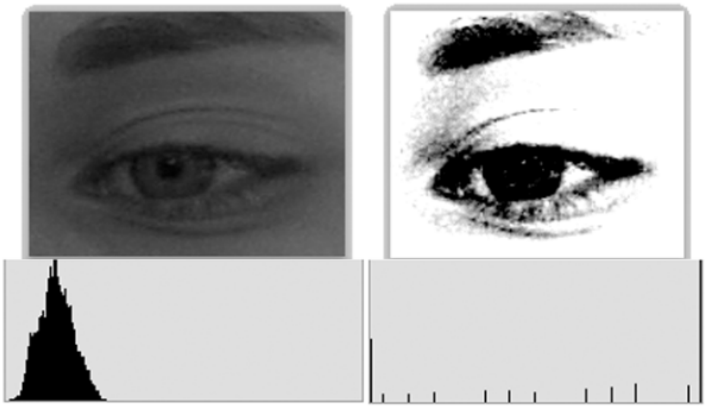
3-17 Εύρεση σκοτεινότερου σημείου. Ο αλγόριθμος αρχικά επιστρέφει τα pixel με τη μικρότερη τιμή φωτεινότητας.

Στη συνέχεια ο αλγόριθμος, ξεκινώντας από κάθε σκοτεινό σημείο που έχει βρεθεί, ψάχνει για δύο σημεία σε αντίθετες κατευθύνσεις με μεγάλη χρωματική αντίθεση (χρωματική διάφορα ίριδας-λευκού μέρους του ματιού). Αν η απόσταση των δύο αυτών σημείων είναι περίπου ίδια από το σκοτεινό pixel, τότε το σκοτεινό σημείο θεωρείται κέντρο ίριδας.

3.7 Contrast-Brightness

Η αντίθεση είναι το ποσοστό διαφοράς μεταξύ των τόνων σε μια εικόνα. Μετά την εφαρμογή του contrast, τα σκοτεινά γκρίζα pixel της εικόνας στρέφονται, ανάλογα με το βαθμό που θα επιλεχτεί, προς το μαύρο και ανοιχτά γκρι προς το λευκό. Η χρήση του για την εύρεση ίριδας, είναι ο διαχωρισμός της ίριδας από άλλα σημεία της εικόνας.

Με το φίλτρο brightness επηρεάζεται η φωτεινότητα μιας εικόνας. Τα λευκά και μαύρα pixel μπορούν να γίνουν λευκότερα αν η εικόνα είναι πολύ σκοτεινή, ή το αντίθετο αν η εικόνα είναι πολύ φωτεινή. Αυτό είναι μπορεί να φανεί χρήσιμο αφού οι περισσότερες μέθοδοι που χρησιμοποιήθηκαν για την εύρεση ίριδας, αλλά και το περιορισμό εικόνας, επηρεάζονται από τη φωτεινότητα.[9]



3-18 Εικόνες και ιστογράμματα μετά από την εφαρμογή των φίλτρων contrast-brightness. Πλέον η ίριδα μπορεί να αναγνωριστεί ευκολότερα από τον υπολογιστή.

4. Ανίχνευση ίριδας χωρίς περιορισμό εικόνας

4.1 Εισαγωγή

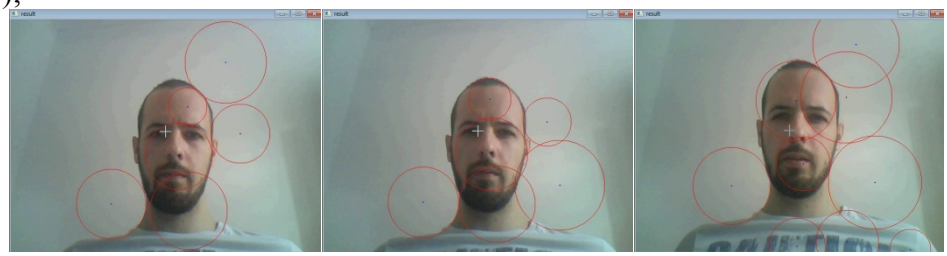
Σκοπός του κεφαλαίου αυτού είναι η ανάδειξη της αναγκαιότητας του περιορισμού της εικόνας. Παρακάτω γίνεται η απόπειρα εύρεσης της ίριδας, χωρίς πρώτα να έχει γίνει κάποιος περιορισμός της εικόνας σε μία περιοχή μεγαλύτερης σημασίας. Για την εύρεση χρησιμοποιείται η σηνάρτηση Hough Circles της OpenCV. Στο τέλος του κεφαλαίου παρουσιάζονται τα αποτελέσματα της απόπειρας αυτής.

4.2 Αποτελέσματα Hough Circles χωρίς περιορισμό εικόνας

Χρησιμοποιώντας τη συνάρτηση cvHoughCircles() με διαφορετικά ορίσματα και σε διαφορετικές συνθήκες πήραμε τα παρακάτω αποτελέσματα.

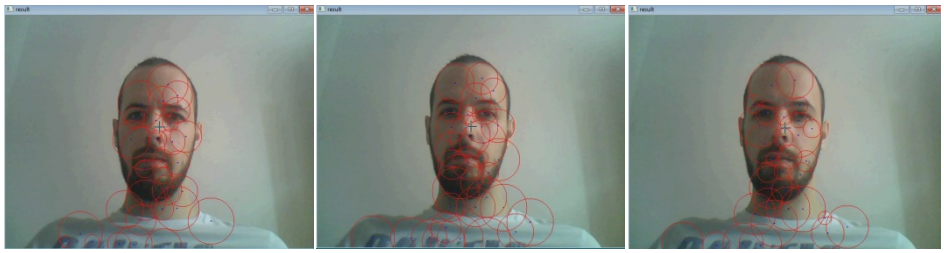
Αποτελέσματα με λευκό φόντο και καλό φωτισμό

1. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,1, 100, 200, 3, 1, 100);



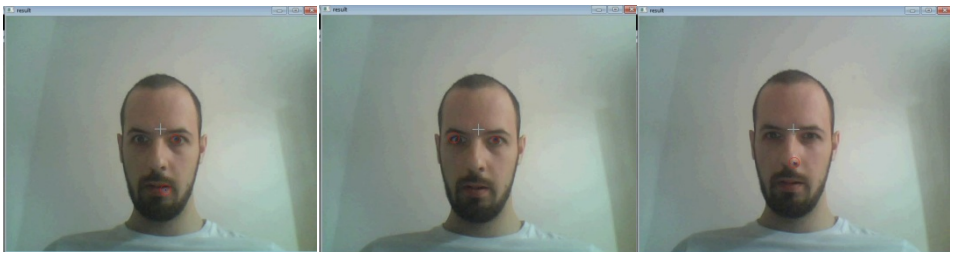
4-1 Εύρεση ίριδας χωρίς περιορισμό. Πρώτο σετ ορισμάτων σε ιδανικές συνθήκες. Η συνάρτηση επιστρέφει πολύ μεγάλους κύκλους πολλοί από τους οποίους είναι ψεύτικοι.

2. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,1, 30, 50, 18, 1, 50);



4-2 Εύρεση ίριδας χωρίς περιορισμό. Δεύτερο σετ ορισμάτων σε ιδανικές συνθήκες. Η συνάρτηση συνεχίζει να επιστρέφει μεγάλους κύκλους.

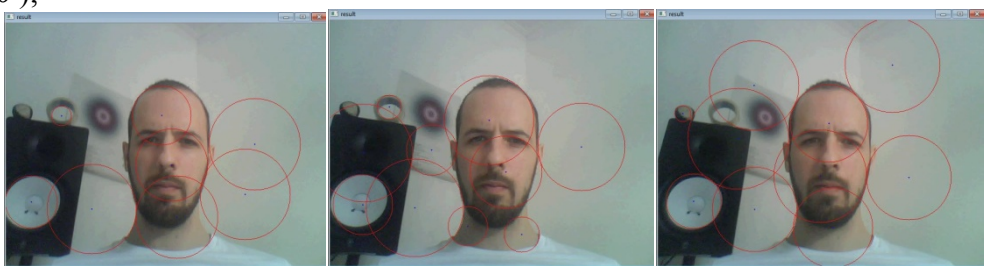
3. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,3, 80, 100, 80, 1, 13);



4-3 Εύρεση ίριδας χωρίς περιορισμό. Τρίτο σετ ορισμάτων σε ιδανικές συνθήκες. Υπάρχει δυνατότητα εύρεσης κύκλων, ωστόσο υπάρχουν ακόμα ψεύτικοι κύκλοι και η εύρεση γίνεται δύσκολα.

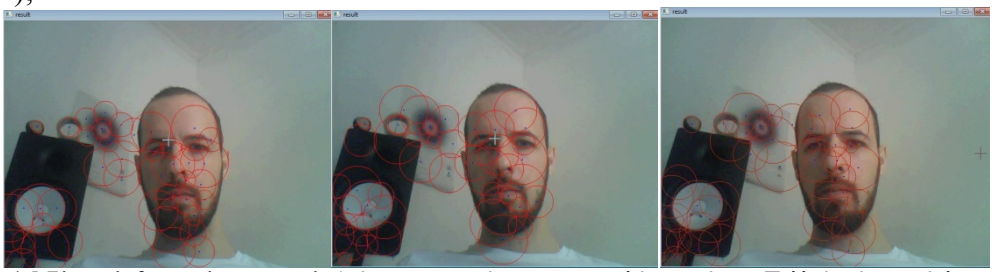
Αποτελέσματα με φόντο που περιέχει στρογγυλά αντικείμενα σε χώρο με καλό φωτισμό.

1. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,1, 100, 200, 3, 1, 100);



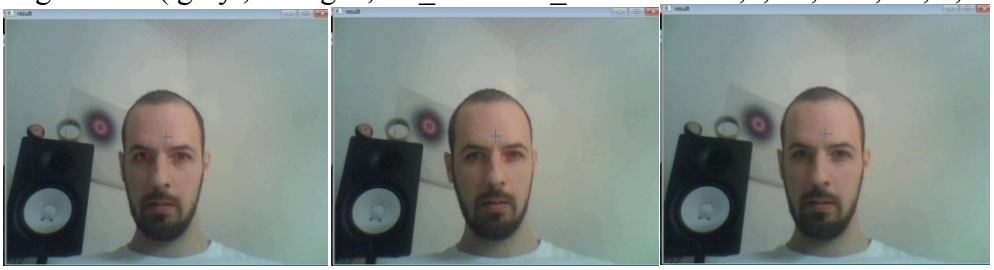
4-4 *Εύρεση ίριδας χωρίς περιορισμό.* Πρώτο σετ σε φόντο με στρογγυλά αντικείμενα. Οι κύκλοι που επιστρέφονται είναι είτε ψεύτικοι ή ανήκουν στο φόντο.

2. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,1, 30, 50, 18, 1, 50);



4-5 *Εύρεση ίριδας χωρίς περιορισμό.* Δεύτερο σετ σε φόντο με στρογγυλά αντικείμενα. Πολλοί ψεύτικοι κύκλοι.

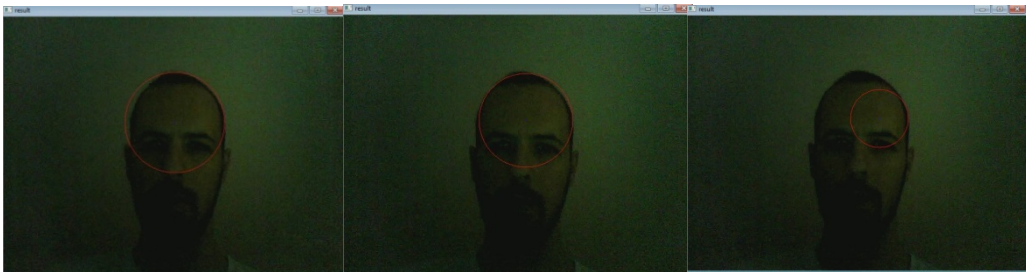
3. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,3, 80, 100, 80, 1, 13);



4-6 *Εύρεση ίριδας χωρίς περιορισμό.* Τρίτο σετ σε φόντο με στρογγυλά αντικείμενα. Υπάρχει δυνατότητα εύρεσης της ίριδας αλλά υπάρχουν ακόμα ψεύτικοι κύκλοι.

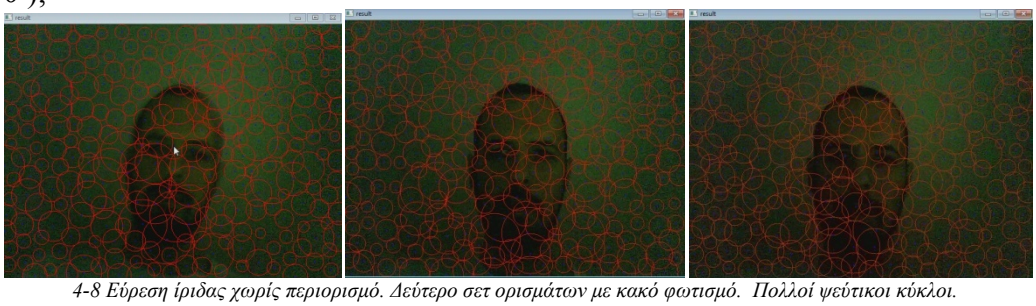
Με φόντο χωρίς στρογγυλά αντικείμενα σε χώρο με κακό φωτισμό.

1. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,1, 100, 200, 3, 1, 100);



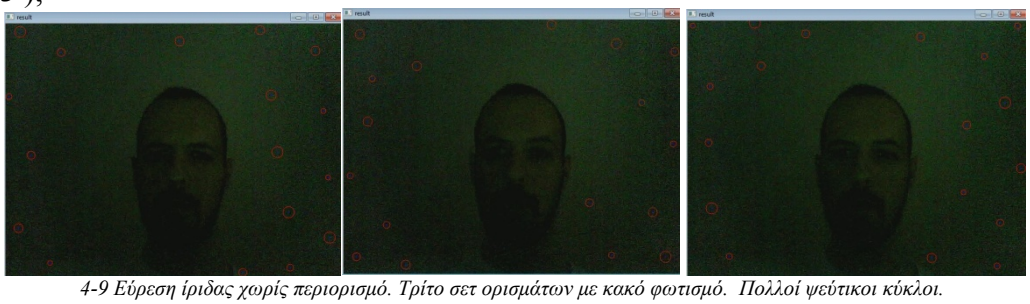
4-7 Εύρεση ίριδας χωρίς περιορισμό. Πρώτο σετ ορισμάτων με κακό φωτισμό. Η ίριδα δεν μπορεί να βρεθεί.

2. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,1, 30, 50, 18, 1, 50);



4-8 Εύρεση ίριδας χωρίς περιορισμό. Δεύτερο σετ ορισμάτων με κακό φωτισμό. Πολλοί ψεύτικοι κύκλοι.

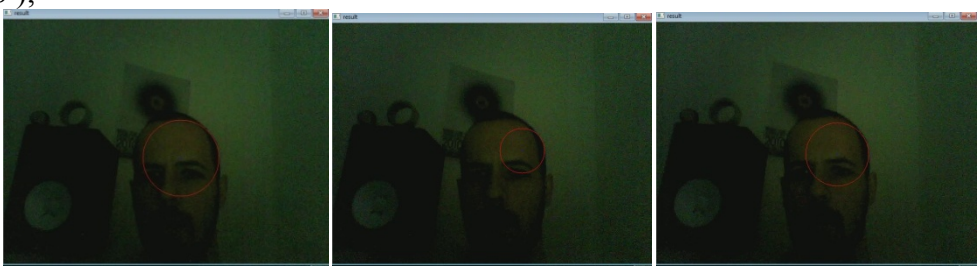
3. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,3, 80, 100, 80, 1, 13);



4-9 Εύρεση ίριδας χωρίς περιορισμό. Τρίτο σετ ορισμάτων με κακό φωτισμό. Πολλοί ψεύτικοι κύκλοι.

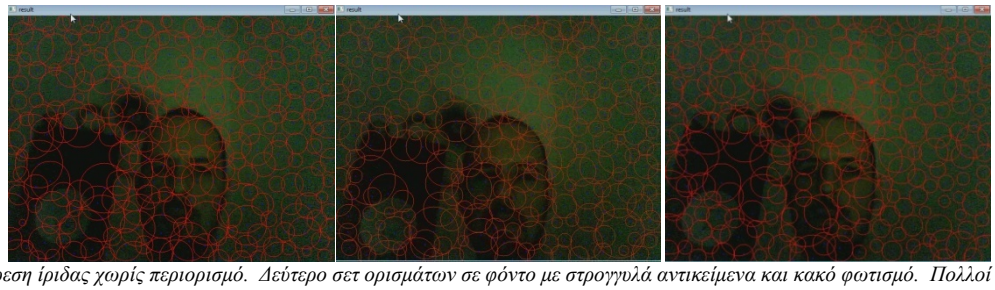
Με φόντο με στρογγυλά αντικείμενα σε χώρο με κακό φωτισμό.

1. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,1, 100, 200, 3, 1, 100);



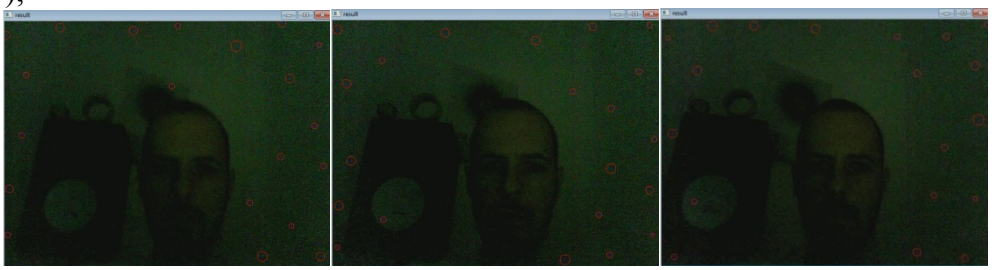
4-10 Εύρεση ίριδας χωρίς περιορισμό. Πρώτο σετ ορισμάτων σε φόντο με στρογγυλά αντικείμενα και κακό φωτισμό. Η ίριδα δεν μπορεί να βρεθεί.

2. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,1, 30, 50, 18, 1, 50);



4-11 Εύρεση ίριδας χωρίς περιορισμό. Δεύτερο σετ ορισμάτων σε φόντο με στρογγυλά αντικείμενα και κακό φωτισμό. Πολλοί ψεύτικοι κύκλοι.

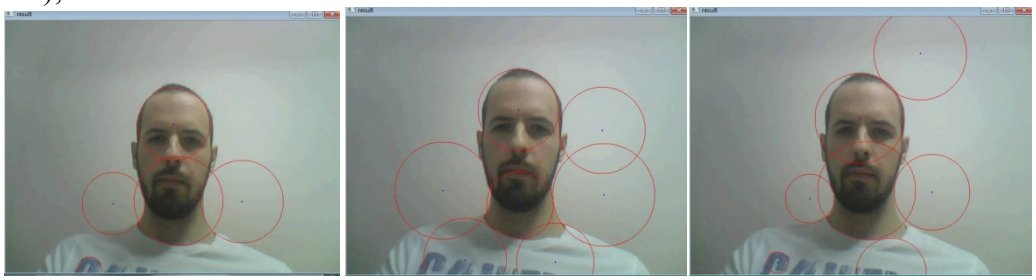
3. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,3, 80, 100, 80, 1, 13);



4-12 Εύρεση ίριδας χωρίς περιορισμό. Δεύτερο σετ ορισμάτων σε φόντο με στρογγυλά αντικείμενα και κακό φωτισμό. Πολλοί ψεύτικοι κύκλοι.

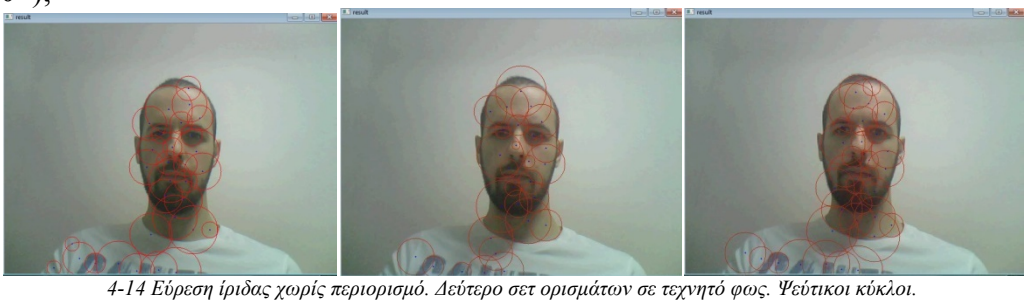
Με φόντο χωρίς στρογγυλά αντικείμενα σε χώρο με τεχνητό φωτισμό.

1. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,1, 100, 200, 3, 1, 100);



4-13 Εύρεση ίριδας χωρίς περιορισμό. Πρώτο σετ ορισμάτων σε τεχνητό φως. Ψεύτικοι κύκλοι.

2. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,1, 30, 50, 18, 1, 50);



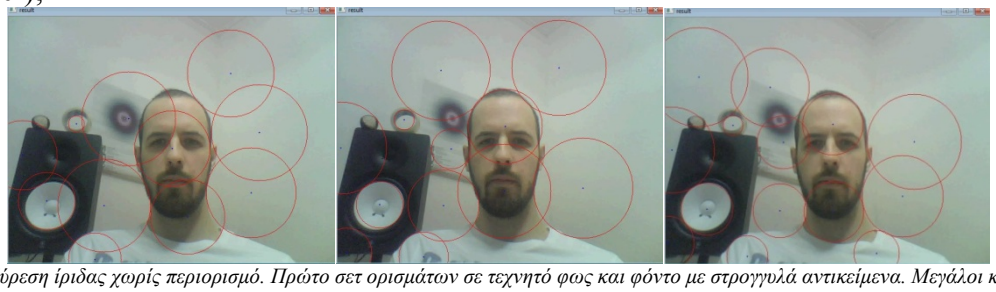
4-14 Εύρεση ίριδας χωρίς περιορισμό. Δεύτερο σετ ορισμάτων σε τεχνητό φως. Ψεύτικοι κύκλοι.

3. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,3, 80, 100, 80, 1, 13);

4-15 Εύρεση ίριδας χωρίς περιορισμό. Τρίτο σετ ορισμάτων σε τεχνητό φως. Η ίριδα μπορεί να βρεθεί αλλά υπάρχουν και ψεύτικοι κύκλοι.

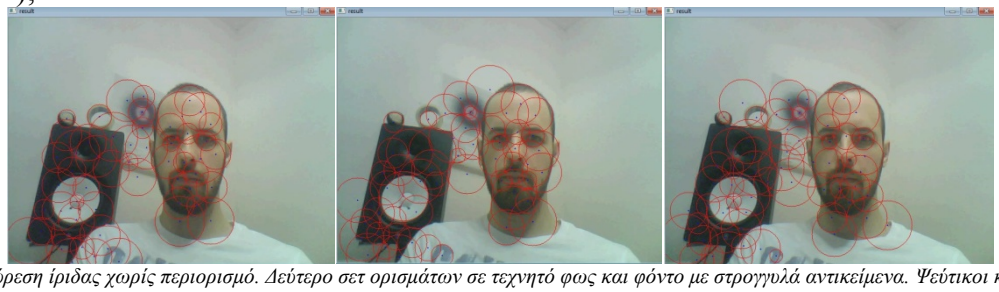
Με φόντο με στρογγυλά αντικείμενα σε χώρο με τεχνητό φωτισμό.

1. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,1, 100, 200, 3, 1, 100);



4-16 Εύρεση ίριδας χωρίς περιορισμό. Πρώτο σετ ορισμάτων σε τεχνητό φως και φόντο με στρογγυλά αντικείμενα. Μεγάλοι κύκλοι, ψεύτικοι.

2. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,1, 30, 50, 18, 1, 50);



4-17 Εύρεση ίριδας χωρίς περιορισμό. Δεύτερο σετ ορισμάτων σε τεχνητό φως και φόντο με στρογγυλά αντικείμενα. Ψεύτικοι κύκλοι.

3. cvHoughCircles(gray , storage1,CV_HOUGH_GRADIENT ,3, 80, 100, 80, 1, 13);

4-18 Εύρεση ίριδας χωρίς περιορισμό. Τρίτο σετ ορισμάτων σε τεχνητό φως και φόντο με στρογγυλά αντικείμενα. Η ίριδα μπορεί να βρεθεί αλλά επιστρέφονται και κύκλοι που δεν είναι ίριδα.

Με φόντο χωρίς στρογγυλά αντικείμενα		Φωτισμός		
		Καλός	Κακός	Τεχνητός
Ορίσματα συνάρτησης	(1, 500, 200, 10, 18, 200)	X	X	X
	(1, 100, 200, 3, 1, 100)	X	X	X
	(1,30, 50, 18, 1, 50)	X	X	X
	(3,80, 100, 80, 1, 13)	Η ίριδα ανιχνεύτηκε	X	Η ίριδα ανιχνεύτηκε

Πίνακας 1 Αποτελέσματα cvHoughCircles()χωρίς περιορισμό εικόνας σε ιδανικό φόντο.

Με φόντο με στρογγυλά αντικείμενα		Φωτισμός		
		Καλός	Κακός	Τεχνητός
Ορίσματα συνάρτησης	(1, 500, 200, 10, 18, 200)	X	X	X
	(1, 100, 200, 3, 1, 100)	X	X	X
	(1,30, 50, 18, 1, 50)	X	X	X
	(3,80, 100, 80, 1, 13)	Η ίριδα ανιχνεύτηκε	X	Η ίριδα ανιχνεύτηκε

Πίνακας 2 Αποτελέσματα cvHoughCircles( )χωρίς περιορισμό εικόνας σε μη ιδανικό φόντο.

4.3 Συμπεράσματα

Όπως μπορεί να διακρίνει κάποιος στις προηγούμενες εικόνες, για να εντοπιστεί η ίριδα πρέπει ο χρήστης να ανοίξει αφύσικα τα μάτια του ώστε να φανεί η ίριδα ολόκληρη. Το τεχνητό φως του συγκεκριμένου δωματίου είναι πολύ ισχυρό και η ανίχνευση δουλεύει αρκετά καλά κάτι που δεν ισχύει συνήθως υπό συνθήκες τεχνητού φωτισμού. Επίσης παράλληλα με τον εντοπισμό της ίριδας των ματιών χαρακτηρίζονται και άλλα μέρη της εικόνας ως κύκλοι (είτε ψεύτικοι είτε αληθινοί), που κάνουν δύσκολη την εύρεση της ίριδας.

Σε αυτήν την φάση των δοκιμών θεωρήθηκε ότι είναι απαραίτητος ο περιορισμός της εικόνας σε μία κρίσιμη περιοχή στην οποία θα γίνεται η αναζήτηση των κύκλων ώστε οι κύκλοι που εντοπίζονται να είναι μόνο αυτοί που σχηματίζονται από τις ίριδες των ματιών. Στο παρακάτω κεφάλαιο παρουσιάζουμε δυο τεχνικές επίτευξης του περιορισμού της εικόνας.

4-19 Εύρεση ίριδας χωρίς περιορισμό. Περιορισμός εικόνας. Στο λευκό περίγραμμα φαίνεται μία κρίσιμη περιοχή για την εύρεση της ίριδας. Με το περιορισμό της εικόνας σε τέτοιες περιοχές η εύρεση γίνεται ευκολότερη.

5. Περιορισμός εικόνας σε περιοχή ματιών

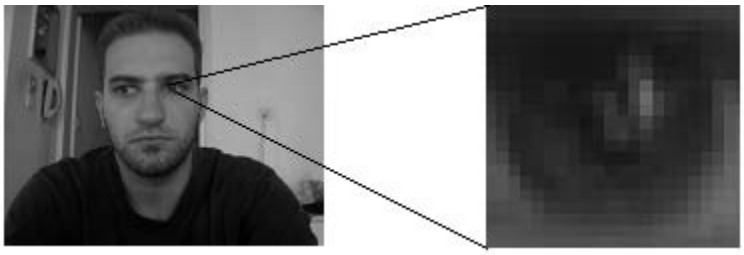
5.1 Εισαγωγή

Από τα συμπεράσματα του προηγούμενου κεφαλαίου πριν την ανίχνευση της ίριδας θα πρέπει να γίνει η εστίαση σε σημεία της εικόνας που είναι σημαντικότερα από άλλα. Η ανίχνευση ματιών μπορεί να χρησιμοποιηθεί για την καλύτερη ανίχνευση της ίριδας, αφού αυτές οι περιοχές είναι οι μοναδικές στις οποίες θα μπορούσε να ανιχνευτεί. Ο στόχος είναι η καλύτερη δυνατή επεξεργασία σε αυτά τα σημεία τις εικόνας και η εξοικονόμηση πόρων από την ανίχνευση αφού η ανίχνευση ίριδας πλέον θα γίνεται μόνο στις περιοχές των ματιών (περιοχές ενδιαφέροντος). Το αποτέλεσμα είναι ένα ή περισσότερα περιγράμματα τα οποία περιβάλουν περιοχές ματιών.

Στο κεφάλαιο αυτό παρουσιάζονται τα πειραματικά αποτελέσματα των μεθόδων ανίχνευσης ματιού που εξετάστηκαν για την ανίχνευση περιοχών ματιών κατά την ανάπτυξη της εφαρμογής.

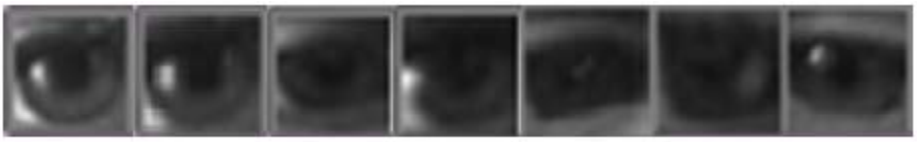
5.2 Αποτελέσματα από εκπαιδευμένο cascade για την εύρεση ίριδας

Για την εκπαίδευση του παρακάτω classifier έχουν χρησιμοποιηθεί samples από εικόνες 640x480. Οι φωτογραφίες τραβήχτηκαν καθώς η ίριδα κοίταζε σε διάφορα σημεία της οθόνης του υπολογιστή.



5-1 Εκπαίδευση classifier. Συγκέντρωση δεδομένων.

Με τη λειτουργία createsamples δημιουργήσαμε ένα vec αρχείο με τα samples που θα χρησιμοποιηθούν για την εκπαίδευση του classifier. Τα samples έχουν όσο το δυνατό περισσότερη χρήσιμη πληροφορία και μέγεθος 20x20.



5-2 Εκπαίδευση classifier. Δείγμα samples ίριδας για την εκπαίδευση του cascade.

Μετά την εκπαίδευση του classifier πήραμε τα παρακάτω στοιχεία από τη λειτουργία performance της OpenCV. Το testing set αποτελείται από εικόνες που δεν χρησιμοποιήθηκαν στο training και έχουν την ίδια ανάλυση και μέγεθος. Παρότι ο

classifier δεν έχει εκπαιδευτεί για συνθήκες κακού φωτισμού, το testing set περιέχει τέτοιου είδους φωτογραφίες.

		Basic			
Max False Alarm	**n of Stages**	**n of Splits**	**Hits**	**Missed**	**False**
0,5	14	1	11	7	135
0,5	18	1	9	9	18
0,5	10	1	14	4	388
0,4	14	1	10	8	65
0,3	14	1	2	16	8
0,2	14	1	6	12	5
0,5	14	2	8	10	2
0,5	14	3	2	16	12

		Core			
Max False Alarm	**n of Stages**	**n of Splits**	**Hits**	**Missed**	**False**
0,5	14	1	11	7	51
0,5	18	1	7	11	9
0,5	10	1	16	2	177
0,4	14	1	9	9	30
0,3	14	1	10	8	14
0,2	14	1	4	14	4
0,5	14	2	6	12	10
0,5	14	3	4	14	10

		All			
Max False Alarm	**n of Stages**	**n of Splits**	**Hits**	**Missed**	**False**
0,5	14	1	6	12	11
0,5	18	1	5	13	1
0,5	10	1	10	8	145
0,4	14	1	7	11	12
0,3	14	1	7	11	7
0,2	14	1	7	11	7
0,5	14	2	2	16	1
0,5	14	3	8	10	10

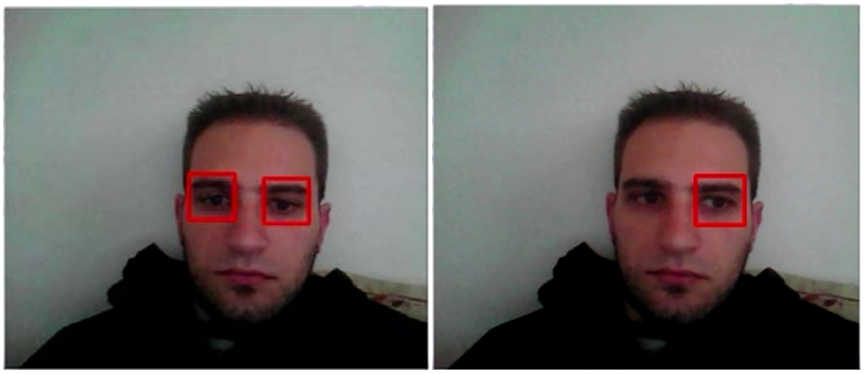
Πίνακας 3 Αποτελέσματα classifier από τη λειτουργία performance της OpenCV. Στους παραπάνω πίνακες φαίνονται οι τιμές maxfalsealarm, nofstages, και nofsplits, που δόθηκαν στα mode basic, core και all της λειτουργίας haartraining και τα αντίστοιχα αποτελέσματα hits, missed false της λειτουργίας performance. Για τις τιμές του minxitrate δε παρουσιάστηκαν διαφορές και για αυτό δεν παρουσιάζονται.

Τα δύο καλύτερα δυνατά αποτελέσματα για αυτό το training set παρουσιαστήκαν με τις επιλογές core, Max False Alarm 0,3,no Of stages 14, no. Of splits 1 με αποτελέσματα Hits 10, Missed 8, False 14 και All, Max False Alarm 0,3 ή 0,2, no Of stages 14, no. Of

splits 1 με αποτελέσματα Hits 7, Missed 11, False 7. Και στις δύο περιπτώσεις το ποσοστό του Hit δεν είναι αρκετά ικανοποιητικό αλλά αυτό μπορεί να οφείλεται στις φωτογραφίες κακού φωτισμού στον οποίο δεν έχει εκπαιδευτεί ο classifier. Επίσης το False Alarm θα μπορούσε να μειωθεί με τη χρησιμοποίηση περισσότερων αρνητικών φωτογραφιών. Παρακάτω παρουσιάζονται τα αποτελέσματα από video για τις επιλογές core, Max False Alarm 0,3, no Of stages 14, no. Of splits 1.

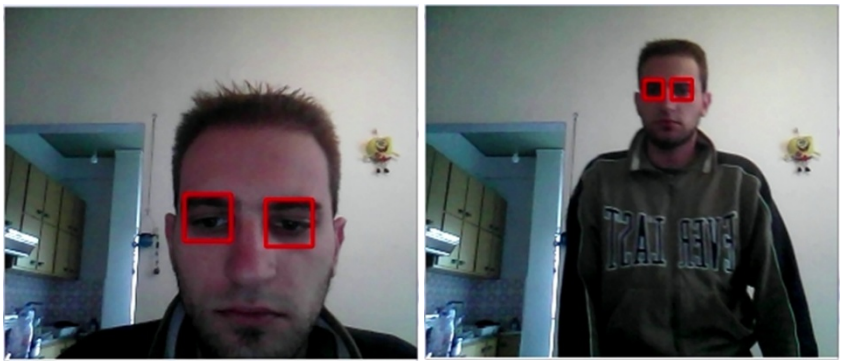
Συμπεράσματα αποτελεσμάτων από ανίχνευση σε video

Σε γενικές γραμμές η απόδοση του classifier, για τις συνθήκες για τις οποίες έχει εκπαιδευτεί, είναι καλή. Το αποτέλεσμα της ανίχνευσης είναι ένα rectangle στις περιοχές όπου έχει ανιχνευτεί ίριδα. Παρά την επιτυχή ανίχνευση της ίριδας όμως, από το αποτέλεσμα δεν μπορεί να βγει κάποιο συμπέρασμα για την ακριβή θέση της και για την επίτευξη κάτι τέτοιου χρειάζεται ανίχνευση μέσα στα rectangle με άλλες τεχνικές. Επίσης, ο classifier παρουσιάζει δυσκολία στην εύρεση όταν η ίριδα δεν αντικρίζει την κάμερα, αλλά δεν υπάρχουν False Alarms.



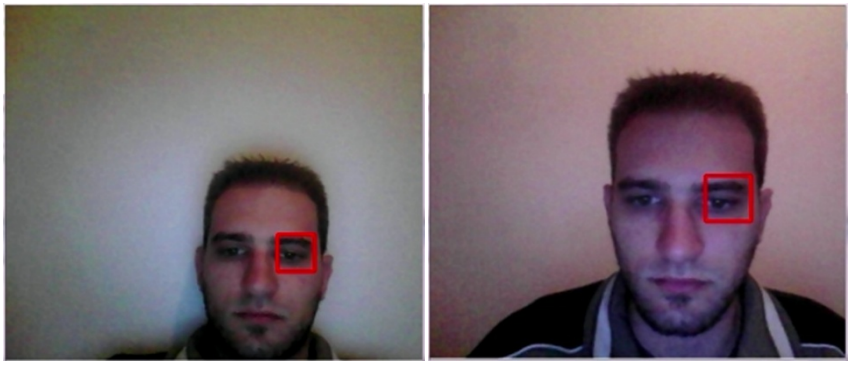
5-3 Haar Classifier. Αποτελέσματα από video σε φώς ημέρας. Ο classifier έχει καλές επιδόσεις με φως ημέρας . Τα rectangles που δημιουργούνται βρίσκουν την ευρύτερη περιοχή όπου βρίσκετε η ίριδα άλλα δεν μπορεί να εξαχθεί κάποιο συμπέρασμα για την θέση της, οπότε το cascade μπορεί να χρησιμοποιηθεί μόνο για τη κατάτμηση εικόνας. Επίσης δυσκολεύεται να ανιχνεύσει την ίριδα όταν αυτή στρέφεται προς το πλάι.

Όσον αφορά την εμβέλεια λειτουργίας ο classifier έχει αποδεκτά αποτελέσματα μεταξύ των αποστάσεων 0,40 με 1,15 μέτρων περίπου από τη κάμερα, συνθήκες για τις οποίες έχει εκπαιδευτεί.



5-4 Haar Classifier. Αποτέλεσμα classifier από διαφορικές αποστάσεις. Αποδεκτά αποτελέσματα υπήρξαν μεταξύ των αποστάσεων 0,40m ως 1,15m περίπου.

Σε συνθήκες κακού φωτισμού ο classifier δυσκολεύεται στην εύρεση αληθινής ίριδας ενώ παράλληλα αυξάνεται ο αριθμός εύρεσης ψεύτικων ιρίδων. Τα αποτελέσματα μπορούν να βελτιωθούν με την εκπαίδευση του classifier σε αυτές τις συνθήκες.



5-5 Haar Classifier. Αποτελέσματα με κακό φωτισμό. Χωρίς το φως της ημέρας και με το φώς της λάμπας ο classifier βρίσκει σπάνια θετικό δείγμα. Τα αποτελέσματα δεν είναι καθόλου καλά, οπότε χρειάζεται περεταίρω εκπαίδευση για αυτές τις συνθήκες.

Δεδομένου ότι μετά την εκπαίδευση δεν υπάρχει χώρος για βελτιώσεις, ο συγκεκριμένος classifier δεν μπορεί να χρησιμοποιηθεί λόγο τις κακής απόδοσης του σε συνθήκες κάκου φωτισμού. Για την βελτίωση της απόδοσης του classifier είναι απαραίτητη η επανεκπαίδευση του με επιπλέον θετικές εικόνες κακού φωτισμού, καθώς και αρνητικών εικόνων.

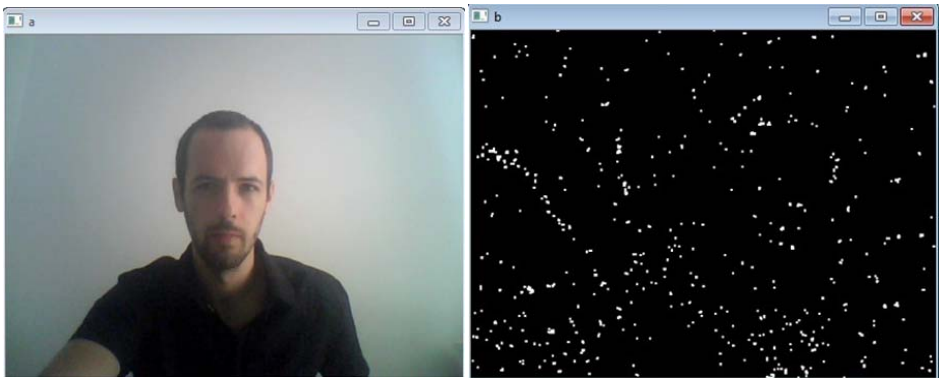
Ένα επιπλέον πρόβλημα που παρουσιάστηκε ήταν η κακή απόδοση του σε εικόνες υψηλής ανάλυσης, αφού η εκπαίδευση του περιορίστηκε σε εικόνες 640x480, ενώ ο χρόνος απόκρισης του είναι γίνεται πολύ μεγαλύτερος.

Haar Classifier	
Εκπαίδευση	Η εκπαίδευση του ενός Haar classifier είναι μία χρονοβόρα και δύσκολη διαδικασία.
Προγραμματισμός	Απλό στο προγραμματισμό του.
Καλός φωτισμός	Παρά τη καλή απόδοση σε καλό φωτισμό δωματίου, υπάρχει δυσκολία εύρεσης της ίριδας όταν αυτή δεν αντικρίζει τη κάμερα, και χρειάζεται περεταίρω εκπαίδευση.
Κακός φωτισμός	Κακή απόδοση.
Εμβέλεια Λειτουργίας	Καλή λειτουργία σε αποστάσεις μεταξύ 0,40 - 1,15 μέτρων (αποστάσεις για τις οποίες έχει εκπαιδευτεί).
Απόκριση	Υπάρχει δυσκολία απόκρισης για εικόνες υψηλής ανάλυσης και εύρεση αντικειμένου από βίντεο.

Πίνακας 4 Πίνακας πειραματικών αποτελεσμάτων haar classifier.

5.3 Συμπεράσματα αποτελεσμάτων Blink detection από ανίχνευση σε video

Μεγάλη σημασία στην σωστή λειτουργία της μεθόδου Blink detection έχει η σωστή ρύθμιση της συνάρτησης cvThreshold(). Στις δυο παρακάτω εικόνες φαίνεται το αποτέλεσμα για διαφορετικά ρυθμισμένο threshold. Στην πρώτη περίπτωση ακόμα και όταν ο χρήστης είναι ακίνητος παρατηρείται ότι η εικόνα έχει θόρυβο (χιόνι) το οποίο οφείλετε στο ότι το threshold είναι ρυθμισμένο με πολύ μικρή τιμή.

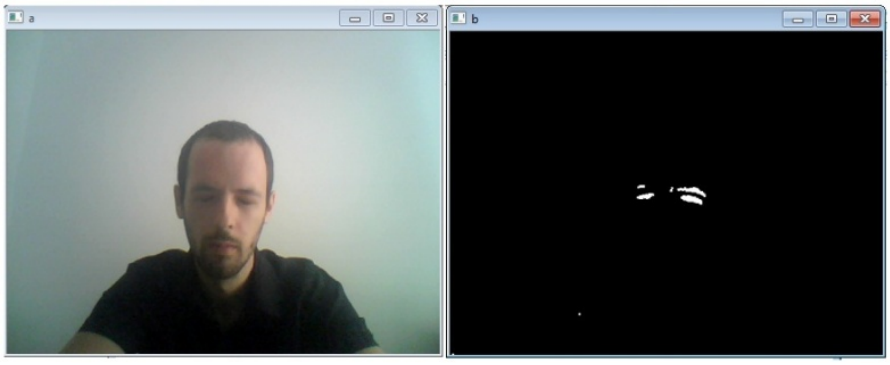


5-6 Blink detection. Χαμηλό threshold. Έχοντας ρυθμίσει το όριο του Threshold χαμηλά η εικόνα αποτελέσματος b παρουσιάζει λευκά pixel που δεν οφείλονται σε κίνηση αλλά σε θόρυβο.

Στην εικόνα που ακολουθεί φαίνεται ένα σωστά ρυθμισμένο threshold ενώ ο χρήστης παραμένει ακίνητος, παρατηρείται ότι ο θόρυβος είναι σχεδόν μηδενικός. Είναι πολύ σημαντικό η τιμή του threshold να μην είναι πολύ μεγάλη, διότι τότε η κίνηση θα πρέπει να είναι πολύ έντονη για να γίνει αντιληπτή από τον υπολογιστή, και οι μικρότερες σε ταχύτητα κινήσεις θα χαθούν από την εικόνα του αποτελέσματος. Είναι προτιμότερο να υπάρχει ένα ποσοστό θορύβου στην περίπτωση μας από το να χαθεί μέρος της κίνησης, αφού η κίνηση που προσπαθούμε να εντοπίσουμε παρόλο που είναι γρήγορη (κάποια εκατοστά του δευτερολέπτου) είναι πολύ μικρή σε σχέση με το μέγεθος της εικόνας.

5-7 Blink detection. Σωστό threshold. Με το όριο του Threshold σε ικανοποιητική τιμή δεν υπάρχει θόρυβος και λευκά pixels παρουσιάζονται μόνο όταν υπάρχει κάποια κίνηση.

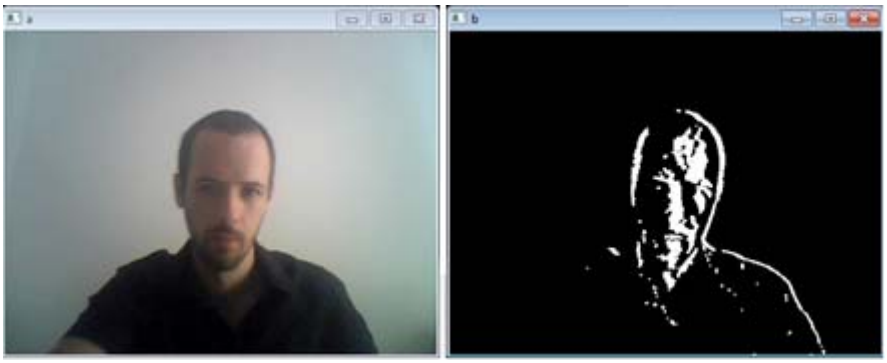
Το ποσοστό θορύβου, καθώς και η προβολή της κίνησης στην εικόνα αποτελέσματος μπορεί να διαφέρει ανάλογα με τη φωτεινότητα των αρχικών εικόνων. Αυτό σημαίνει ότι το Threshold θα πρέπει να ρυθμίζεται διαφορετικά σε συνθήκες κακού φωτισμού από όταν λειτουργεί με καλό φωτισμό.



5-8 Blink detection. ανοιγοκλείσιμο των ματιών με σωστό threshold

Ακόμα και όταν ο χρήστης μένει ακίνητος, ανοιγοκλείνοντας τα μάτια, παρατηρείται ότι τα αποτελέσματα δεν είναι τα επιθυμητά. Εκτός από τα μάτια, εντοπίζεται κίνηση σε σημεία που ασυναίσθητα κινούνται όταν ανοιγοκλείνουν τα μάτια. Ο λόγος που το αποτέλεσμα δεν είναι επιθυμητό είναι το ότι χρειάζεται να ξεχωρίσει το πρόγραμμα συγκεκριμένο αριθμό κινούμενων αντικειμένων .

Θα πρέπει να βρεθούν δύο ενιαία αντικείμενα (contour) τα οποία θα πρέπει να πληρούν κάποιες προϋποθέσεις που θα αποκλείουν λανθασμένα αποτελέσματα. Οι προϋποθέσεις αυτές είναι η απόσταση μεταξύ των contour, η υψομετρική διαφορά τους, καθώς και το μέγεθος τους. Η προϋποθέσεις αυτές, μπορεί να είναι διαφορετικές ανάλογα την απόσταση από τη κάμερα ή ανάλογα τη φωτεινότητα κάθε πλευράς του προσώπου. Για παράδειγμα αν υπάρχει φως μόνο από τη μία πλευρά του προσώπου το μέγεθος των contour μπορεί να είναι διαφορετικό.



5-9 Blink detection. Ανίχνευση κίνησης που δεν οφείλεται σε ανοιγόκλεισμα ματιών. Για τον αποκλεισμό των κινήσεων που δεν οφείλεται σε ανοιγόκλεισμα ματιών τα παραγόμενα contours θα πρέπει ελέγχονται ως προς την απόσταση τους την υψομετρική διαφορά τους και το μέγεθος τους.

Blink detection	
Εκπαίδευση	Δεν απαιτείται εκπαίδευση.
Προγραμματισμός	Πολύπλοκο στο προγραμματισμό του.
Καλός φωτισμός	Καλή απόδοση, με τη δυνατότητα ρύθμισης της συνάρτησης cvThreshold().
Κακός φωτισμός	Καλή απόδοση, με τη δυνατότητα ρύθμισης της συνάρτησης cvThreshold().
Εμβέλεια Λειτουργίας	Η επίδοση με βάση την απόσταση μπορεί να ρυθμιστεί από τη συνάρτηση cvCreateStructuringElementEx() και την cvThreshold.
Απόκριση	Καλή απόκριση ακόμα και για αντικειμένου από βίντεο υψηλής ανάλυσης.

Πίνακας 5 Πίνακας πειραματικών αποτελεσμάτων Blink Detection

5.4Σύγκριση Haar Classifier - Blink motion

Προγραμματισμός – Εκπαίδευση : Η διαδικασία εκπαίδευσης ενός Haar classifier είναι μία χρονοβόρα διαδικασία η οποία περιλαμβάνει τη συγκέντρωση θετικών και αρνητικών φωτογραφιών, την εκπαίδευση του με τα καλύτερα δυνατά ορίσματα, και το testing. Αν ο classifier αποτύχει το testing τότε η διαδικασία θα πρέπει να επαλειφθεί. Η βιβλιοθήκη της OpenCV παρέχει όμως κάποιους έτοιμους classifiers για εύρεση αντικειμένων. Το blink detection δεν απαιτεί εκπαίδευση αλλά είναι πιο σύνθετο ως προς τον προγραμματισμό του.

Απόκριση : Ο haar classifier αργή απόκριση όταν πρέπει να ανιχνεύσει αντικείμενα σε εικόνες μεγάλης ανάλυσης ή εικόνες video. Με το blink detection δεν υπάρχουν τέτοια προβλήματα απόκρισης.

Χρησιμότητα : Ένας haar classifier μπορεί να εκπαιδευτεί για να ανιχνεύσει οποιοδήποτε άκαμπτο αντικείμενο, ενώ, το blink detection μπορεί να βρει μόνο τη περιοχή στην οποία έχει γίνει ένα κλείσιμο ματιών εκτός εάν τροποποιηθεί ο κώδικας. Επίσης ένας classifier μπορεί να έχει αποδεκτά αποτελέσματα για τις συνθήκες που έχει εκπαιδευτεί (φωτεινότητα, ποιότητα εικόνων), ενώ το Blink Detection απλά χρειάζεται να ρυθμιστεί.

Για την εύρεση περιοχής όπου υπάρχει ίριδα : Το μεγαλύτερο πρόβλημα ενός haar classifier είναι η δυσκολία εκπαίδευσης του για την εύρεση αντικειμένου από διαφορετικές γωνίες και σε διαφορετικές συνθήκες φωτεινότητας. Για την εκπαίδευση του για αναγνώριση ίριδας το πρόβλημα αυτό δεν είναι τόσο μεγάλο αφού οι πιθανές γωνίες δεν είναι τόσες πολλές, αλλά υπάρχει ένα επιπλέον πρόβλημα καθώς η ίριδα μπορεί να καλύπτεται από το βλέφαρο ή γυαλιά. Ένα πρόβλημα με το blink detection είναι στη περίπτωση που υπάρχει διαφορά φωτεινότητας στις περιοχές των ματιών (πχ το φώς στην μία πλευρά είναι περισσότερο από την άλλη). Επίσης εάν υπάρχει κίνηση στο φόντο το blink detection δεν μπορεί να λειτουργήσει. Βέβαια επειδή πιθανότατα θα χρησιμοποιείται σε κλειστούς και ελεγχόμενους χώρους αυτό δεν αποτελεί μεγάλο πρόβλημα.

6.Ανίχνευση ίριδας σε περιορισμένη εικόνα

Αφού η ανίχνευση ματιού έχει επιτευχθεί, οι περιοχές ενδιαφέροντος μπορούν να εξεταστούν για ανίχνευση ίριδας γρηγορότερα και πιο αποτελεσματικά αφού πλέον η ανίχνευση περιορίζεται στα σημεία αυτά και η επεξεργασία της εικόνας μπορεί να αναδείξει την ίριδα με λιγότερες, αν όχι καθόλου, ψεύτικες ίριδες. Σε αυτό το κεφάλαιο παρουσιάζονται τα πειραματικά αποτελέσματα των τεχνικών που δοκιμάστηκαν κατά την ανάπτυξη της εφαρμογής.

6.1 Πειραματικά αποτελέσματα cvHoughCircles()

Η cvHoughCricles() είναι μία συνάρτηση εύρεσης κύκλων βάση του σχήματος. Για την περίπτωση της εύρεσης ίριδας, το πρόβλημα είναι ότι η ίριδα τις περισσότερες φορές καλύπτεται από τα βλέφαρα και μπορεί να μην λαμβάνεται ως κύκλος. Αυτό σημαίνει ότι η ανίχνευση θα πρέπει να γίνεται με χαλαρά κριτήρια. Αυτό όμως έχει κόστος όσον αφορά την ανίχνευση ψεύτικων κύκλων.

Ακόμα, η ανίχνευση επηρεάζεται από τη φωτεινότητα. Αυτό γίνεται διότι σε συνθήκες χαμηλού φωτισμού ο μετασχηματισμός canny edge detection δεν μπορεί να αναγνωρίσει ακμές το ίδιο καλά στο σκοτάδι επηρεάζοντας έτσι και την ανίχνευση του κύκλου. Μία επίλυση σε αυτό το πρόβλημα είναι η μείωση του canny threshold για τη καλύτερη ανίχνευση ακμών και η μείωση του συσσωρευτή threshold με το κόστος όμως εύρεσης ψεύτικων κύκλων.

6-1 Hough circles με κοινά ορίσματα σε διαφορετικές εντάσεις φωτός. Το αποτέλεσμα του canny edge detection δεν μπορεί να αναγνωρίσει ακμές το ίδιο καλά στο σκοτάδι επηρεάζοντας έτσι και την ανίχνευση του κύκλου.

Ο αλγόριθμος της houghCircles() βασίζεται και στην ακτίνα των κύκλων που πρέπει να ανιχνευτούν. Όταν η απόσταση του χρήστη από τη κάμερα είναι διαφορετική, αλλάζει το μέγεθος της ίριδας όπως το αντιλαμβάνεται ο υπολογιστής και κατά συνέπια η ακτίνα της. Άρα οι επιθυμητές ακτίνες στα ορίσματα της houghCircles() θα πρέπει να δίνονται ανάλογα με την απόσταση από τη κάμερα.

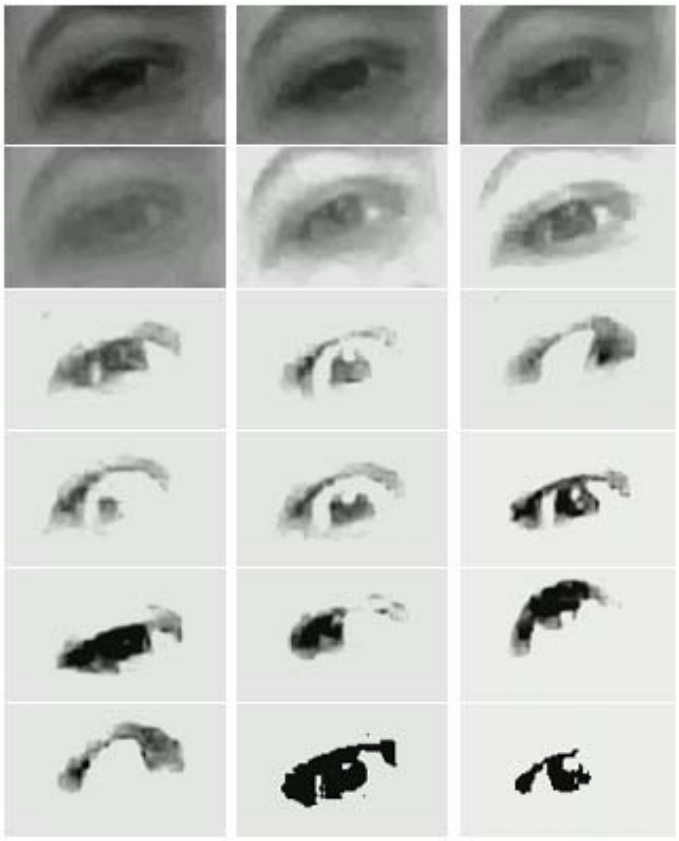
6-2 Hough circles με κοινά ορίσματα από διαφορετικές αποστάσεις. Δοκιμάζοντας τη HoughCircles με διαφορετικές αποστάσεις εκατοστά από τη κάμερα το τελικό αποτέλεσμα επηρεάζεται, αφού η ακτίνα της ίριδας στην εικόνα αλλάζει, έτσι πρέπει να γίνουν διαφορετικές ρυθμίσεις.

Hough Circles	
Απόδοση	Καλή απόδοση όταν η ίριδα αντικρίζει τη κάμερα (δεν καλύπτεται από βλέφαρο ή άλλο αντικείμενο). Σε αντίθετη περίπτωση μπορεί να γίνει ρύθμιση του συσσωρευτή threshold.
Απόσταση	Η επίδοση σε διαφορετικές απόστασης μπορεί να ρυθμιστεί από τα ορίσματα min_radious - max_radius
Φωτεινότητα	Η επίδοση σε διαφορετικές συνθήκες φωτεινότητας επηρεάζεται από το όρισμα του canny edge detector και το όρισμα του συσσωρευτή threshold

Πίνακας 6 Πίνακας πειραματικών αποτελεσμάτων HoughCircles().

6.2 Contrast-brightness

Σε εικόνες που περιέχουν ίριδα, για να γίνει η ίριδα πιο ευδιάκριτη εφαρμόστηκαν φίλτρα αντίθεσης και φωτεινότητας. Στις παρακάτω φωτογραφίες φαίνονται στιγμιότυπα από το πρόγραμμα ξεκινώντας έχοντας τα sliders φωτεινότητας και αντίθεσης σε ουδέτερη θέση, αυξάνοντας σταδιακά τις τιμές τους, καταλήγοντας στις θέσεις όπου η εικόνα έχει ελάχιστη πληροφορία.



6-3`Εφαρμογή φίλτρου contrast-brightness. Με την εφαρμογή του φίλτρου είναι δυνατός ο διαχωρισμός τις ίριδας από την υπόλοιπη εικόνα, η ίριδα όμως αλλοιώνεται.

Όπως μπορεί κάποιος να παρατηρήσει, με τη μέθοδο αυτή πράγματι η ίριδα γίνεται αρκετά πιο ευδιάκριτη. Δυστυχώς όμως αυτό ισχύει μόνο όταν η ίριδα απέχει από τα βλέφαρα. Όταν η ίριδα πλησιάζει τα βλέφαρα ή το δέρμα στις άκρες του ματιού στην επεξεργασμένη εικόνα γίνεται ουσιαστικά ένα με αυτά και αλλοιώνεται το σχήμα της. Ακόμα η επιλογή του κατάλληλου βαθμού του contrast και του brightness εξαρτάται από το χρήστη. Για τους λόγους αυτούς θεωρήθηκε ότι η συγκεκριμένη τεχνική δεν είναι κατάλληλη από μόνη της για ανίχνευση ίριδας. Παρόλα αυτά όμως θα μπορούσε να χρησιμοποιηθεί μαζί με άλλες τεχνικές.

Contrast-brightness	
Απόδοση	Μπορεί να διαχωρίσει την ίριδα, μέχρι ενός σημείου. Η ίριδα αλλοιώνεται ως σχήμα σε συγκεκριμένες θέσεις. Θα μπορούσε να συνδυαστεί με άλλες τεχνικές. Μη αυτόματη μέθοδος.
Απόσταση	Επηρεάζεται από την απόσταση.
Φωτεινότητα	Μπορεί να ρυθμιστεί από το φίλτρο Brightness.

Πίνακας 7 Πίνακας πειραματικών αποτελεσμάτων Contrast-brightness.

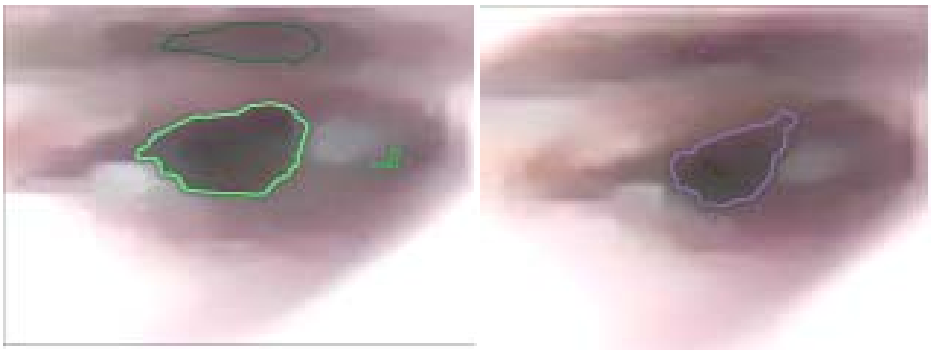
6.3 Δημιουργία Contour βάση χρώματος

Με τη τεχνική αυτή δίνεται η δυνατότητα επιλογής χρωματικών συνδυασμών με σκοπό την επιλογή όλων των pixel με συγκεκριμένο χρώμα στην εικόνα. Σε μία εικόνα που περιέχει ίριδα, ο χρήστης μπορεί να την επιλέξει διαλέγοντας το χρώμα της ίριδας. Το πρόβλημα είναι ότι άλλα αντικείμενα μέσα στην εικόνα ενδέχεται να έχουν ίδιο χρώμα με την ίριδα, και αυτό σημαίνει ότι θα επιλεχτούν και αυτά.

Για τη περίπτωση που η τεχνική αυτή εφαρμόζεται σε βίντεο, όταν ο χρήστης κοιτά σε διαφορετικά σημεία, αλλάζει και το χρώμα της ίριδας έτσι όπως το αντιλαμβάνεται ο υπολογιστής. Αυτό σημαίνει ότι χρειάζονται διαφορετικές ρυθμίσεις.

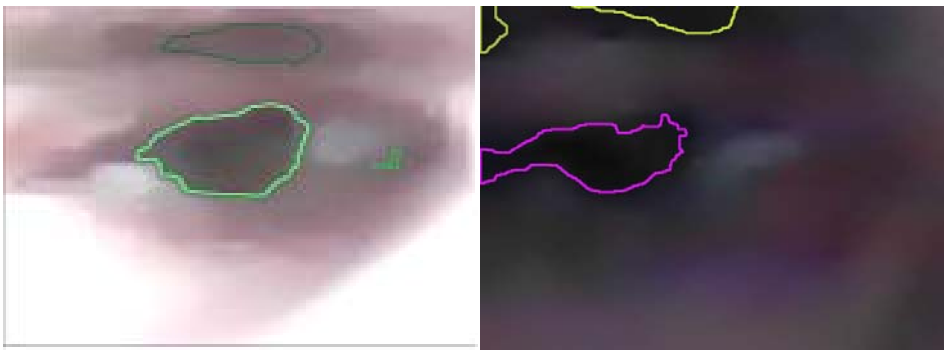
6-4 Contour βάση χρώματος. Αποτελέσματα κοιτώντας σε διαφορετικά σημεία. Με την αλλαγή της θέσης της ίριδας ενδέχεται το χρώμα της να αλλάξει, έτσι όπως το αντιλαμβάνεται ο υπολογιστής, και χρειάζεται καινούρια επιλογή χρωματικού συνδυασμού.

Η απόσταση δεν επηρεάζει το αποτέλεσμα αρκεί να μην αλλάζει ο φωτισμός. Αυτό γίνεται επειδή η εύρεση βασίζεται στο χρώμα και σε διαφορετικές αποστάσεις υπάρχει περίπτωση να επηρεάζεται από το φως.



6-5 Contour βάση χρώματος. Αποτελέσματα σε διαφορετικές αποστάσεις. Η απόσταση δεν επηρεάζει την εύρεση της ίριδας.

Σε διαφορετικό φωτισμό, η ίριδα έχει διαφορετικό χρώμα έτσι όπως την αντιλαμβάνεται ο υπολογιστής και απαιτείται διαφορετική επιλογή χρωματικού συνδυασμού. Με χαμηλότερο φωτισμό η αντίθεση της εικόνας μειώνεται και αυτό έχει ως συνέπια την επιλογή περισσότερης άχρηστης πληροφορίας.



6-6 Contour βάση χρώματος. Αποτελέσματα σε διαφορετικές συνθήκες φωτισμού. Σε διαφορετικό φωτισμό, η ίριδα έχει διαφορετικό χρώμα έτσι όπως το την αντιλαμβάνεται ο υπολογιστής.

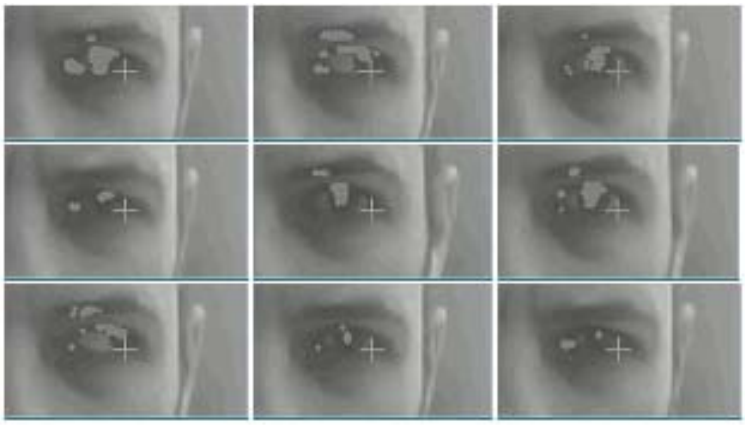
Το κύριο πρόβλημα αυτής της τεχνικής είναι ότι δεν είναι αυτόματη. Αυτό απαιτεί από το χρήστη να επιλέξει μόνος του ένα κατάλληλο συνδυασμό για την εύρεση της ίριδας.

Επιλογή contour βάση χρώματος	
Απόδοση	Μη αυτόματη μέθοδος, ο χρήστης πρέπει να επιλέξει μόνος του το σωστό χρωματικό συνδυασμό. Ακόμα και με τι καλύτερη δυνατή επιλογή χρωμάτων, θα υπάρξει επιλογή άχρηστης πληροφορίας. Αν η ίριδα αλλάξει θέση μετά την επιλογή χρώματος η επανεπιλογή χρώματος είναι απαραίτητη.
Απόσταση	Δεν επηρεάζεται από την απόσταση.
Φωτεινότητα	Σε διαφορετικές συνθήκες φωτεινότητας χρειάζεται νέα επιλογή χρωματικού συνδυασμού. Ένα contour σε εικόνα χαμηλής φωτεινότητας έχει περισσότερη άχρηστη πληροφορία από ένα σε μια εικόνα κανονικής φωτεινότητας.

Πίνακας 8 Πίνακας πειραματικών αποτελεσμάτων επιλογής contour βάση χρώματος.

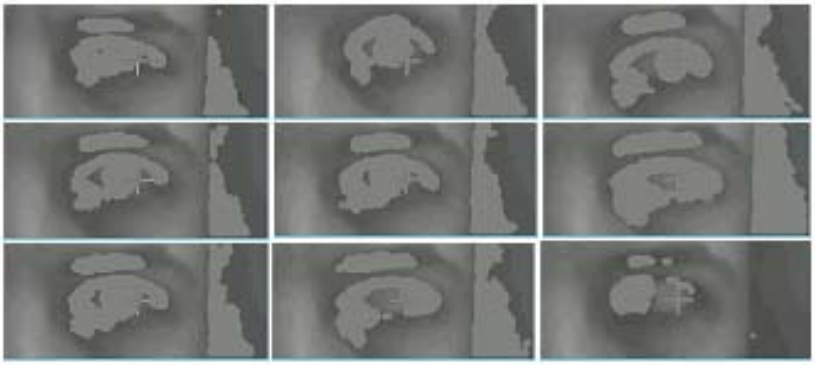
6.4 Εύρεση του σκοτεινότερου pixel (κόρης του ματιού)

Με την τεχνική αυτή πολλές φορές το σκοτεινότερο σημείο δεν είναι το κέντρο ίριδας, είτε λόγο του φωτισμού, της χαμηλής ανάλυσης, είτε λόγο της αντανάκλασης της ίριδας στο φως. Αυξάνοντας τον αριθμό των pixel που επιστρέφει ο αλγόριθμος αυξάνεται και η πιθανότητα εύρεσης του κέντρου μιας ίριδας. Αυτό πετυχαίνεται με την επιστροφή όλων των pixel που έχουν μέχρι και δέκα βαθμίδες περισσότερη φωτεινότητα από το σκοτεινότερο pixel.



6-7Ευρεση σκοτεινότερου pixel. Με την αύξηση των pixel που επιστρέφονται από τη συνάρτηση, αυξάνεται η πιθανότητα να επιστραφεί και το κέντρο της ίριδας.

Σε περίπτωση που τα pixels που επιστρέφει η συνάρτηση είναι πολλά, τότε το πρόγραμμα γίνεται αναποτελεσματικό . Αυτό σημάνει ότι χρειάζεται σωστή επιλογή της βαθμίδας φωτεινότητας που θα επιστρέφεται από τη συνάρτηση. Παρακάτω φαίνεται το αποτέλεσμα με επιστροφή pixels μέχρι και είκοσι βαθμίδες φωτεινότητας περισσότερο από το σκοτεινότερο pixel.



6-8 Εύρεση σκοτεινότερου pixel. Πολλά επιστρεφόμενα pixel.

Εύρεση σκοτεινότερου pixel	
Απόδοση	Σε περιπτώσεις όπου το σκοτεινότερο σημείο δεν είναι το κέντρο της ίριδας η εύρεση είναι δύσκολη. Καλή λειτουργία με κάμερες μεγάλης ευκρίνειας.
Απόσταση	Δεν επηρεάζεται από την απόσταση
Φωτεινότητα	Χρειάζεται τον κατάλληλο φωτισμό, έτσι ώστε να υπάρχει η καλύτερη δυνατή αντίθεση μεταξύ της ίριδας και του λευκού του ματιού.

Πίνακας 9 Πίνακας συμπερασμάτων εύρεσης σκοτεινότερου pixel

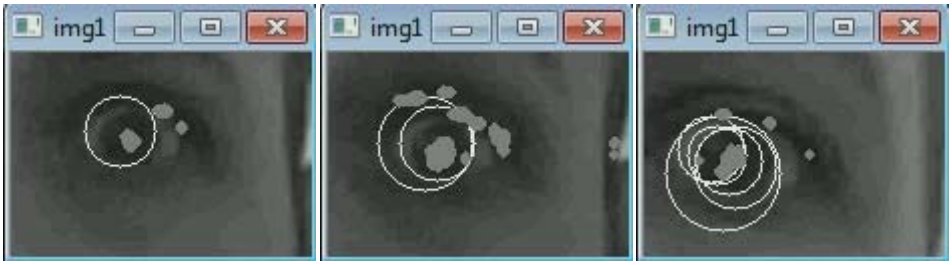
6.5 Εύρεση της ίριδας με τον συνδυασμό ανίχνευσης κύκλων και της εύρεσης της κόρης του ματιού.

Η cvHoughCircles() είναι μία συνάρτηση εύρεσης κύκλων που βασίζεται στο σχήμα. Για την περίπτωση της εύρεσης ίριδας το αντικείμενο προς εύρεση καλύπτεται από τα βλέφαρα, με αποτέλεσμα η ίριδα να μην φαίνεται ως κύκλος και να χρειάζεται αναζήτηση με χαμηλά κριτήρια. Ο αλγόριθμος του σκοτεινότερου σημείου από την άλλη βασίζεται στην φωτεινότητα (τη τιμή από 0 ως 255 των pixels μίας grayscale εικόνας). Το πρόβλημα για την εύρεση ίριδας είναι πως pixels που δεν ανήκουν στην ίριδα μπορεί να έχουν την ίδια φωτεινότητα. Για να έχουμε τα καλύτερα δυνατά αποτελέσματα στην ανίχνευση ίριδας δοκιμάστηκε ο συνδυασμός των δύο παραπάνω τεχνικών.

Αρχικά βρίσκονται κύκλοι στην περιορισμένη εικόνα (στην περιοχή γύρο από το μάτι), και ύστερα θεωρείται ότι απεικονίζουν την ίριδα μόνο οι κύκλοι των οποίων το κέντρο τους βρίσκεται ένα από τα ποιο σκούρα pixel. Για τα καλύτερα δυνατά αποτελέσματα με τη συγκεκριμένη τεχνική, τα ορίσματα της cvHoughCircles() θα πρέπει να έχουν ρυθμιστεί έτσι ώστε να επιστρέφονται μόνο κύκλοι που απεικονίζουν την ίριδα, και τα σκούρα pixel που εντοπίζονται να ανείκουν στην κόρη του ματιού.

6-9 Συνδυασμός ανίχνευσης κύκλων και της εύρεσης της κόρης του ματιού. Μηδέν επιστρεφόμενοι κύκλοι από τη cvHoughCircles(). Το πρόγραμμα θεωρεί ότι δεν υπάρχει ίριδα.

6-*10 Συνδυασμός ανίχνευσης κύκλων και της εύρεσης της κόρης του ματιού. Εύρεση κύκλου με κέντρο ένα από τα σκοτεινότερα pixel. Το πρόγραμμα θεωρεί ότι υπάρχει ίριδα.*



6-11 Συνδυασμός ανίχνευσης κύκλων και της εύρεσης της κόρης του ματιού. Εύρεση κύκλων με κέντρο μη σκοτεινό σημείο, το πρόγραμμα θεωρεί ότι δεν υπάρχει ίριδα.

Συνδυασμός cvHoughCircles και εύρεσης σκοτεινότερου pixel	
Απόδοση	Μειώνει τη περίπτωση εύρεσης ψεύτικης ίριδας συνδυάζοντας τεχνικές εύρεσης που βασίζονται στο σχήμα και στο χρώμα.
Απόσταση	Για διαφορετικές αποστάσεις χρειάζεται ρύθμιση του ορίσματος των ακτινών της cvHoughCircles().
Φωτεινότητα	Χρειάζεται τον κατάλληλο φωτισμό, έτσι ώστε να υπάρχει η καλύτερη δυνατή αντίθεση μεταξύ της ίριδας και του λευκού του ματιού. Ακόμα χρειάζεται και ρύθμιση του ορίσματος του cany edge detector.

Πίνακας 10 Πίνακας αποτελεσμάτων συνδυασμού cvHoughCircles και εύρεσης σκοτεινότερου pixel

7. Ανάπτυξη εφαρμογής

7.1 Εισαγωγή

Σε αυτή την ενότητα παρουσιάζεται η ανάπτυξη μίας εφαρμογής ελέγχου του δείκτη του ποντικιού (κέρσορα) σύμφωνα με την κίνηση της ίριδας, καθώς και παραλλαγές της που αυξάνουν τη λειτουργικότητα. Για την ανάπτυξη της εφαρμογής αυτής χρησιμοποιήθηκαν τεχνικές και αλγόριθμοι που παρουσιάστηκαν και δοκιμάστηκαν πειραματικά στα προηγούμενα κεφάλαια.

Αρχικά στην εφαρμογή, χρησιμοποιείται Blink Detection για το περιορισμό της εικόνας και μία προ-αποθηκευμένη φωτογραφία ίριδας ματιού για το template matching (εύρεση παρόμοιας εικόνας), ώστε να υπολογιστεί η κατάλληλη θέση του κέρσορα στην οθόνη.

Στην πρώτη παραλλαγή του προγράμματος, ακολουθείται μια παρόμοια διαδικασία με αυτή της αρχικής, μόνο που αυτή τη φορά η ίριδα που χρησιμοποιείται για το template matching είναι μία ίριδα που συλλέγεται από το χρήστη. Για την εύρεση της ίριδας του χρήστη σε αυτή τη παραλλαγή, χρησιμοποιείται πάλι αρχικά μια αποθηκευμένη εικόνα ίριδας. Στην δεύτερη παραλλαγή χρησιμοποιείται για το template matching η ίριδα του χρήστη, άλλα αφού αυτή έχει πρώτα βρεθεί με το συνδυασμό των cvHoughCircles() και σκοτεινότερου σημείου, ο οποίος προσφέρει καλύτερα αποτελέσματα.

Στην επόμενη παραλλαγή παρουσιάζεται μια τεχνική που βελτιώνει την απόδοση της εφαρμογής κάνοντας την πιο σταθερή μέσω της τοποθέτησης του δείκτη του ποντικιού σε συγκεκριμένες θέσεις.

Τέλος παρουσιάζεται μία αυτοματοποιημένη αρχικοποίηση που καθιστά την εφαρμογή πολύ πιο χρήσιμη για χρήστες με κινητικές δυσκολίες.

7.2 Έλεγχος κέρσορα Η/Υ με τα μάτια χρησιμοποιώντας αποθηκευμένη φωτογραφία ίριδας.

Στο εδάφιο αυτό περιγράφονται οι βασικές λειτουργίες της αρχικής εφαρμογής για τον έλεγχο του δείκτη του ποντικιού. Συγκεκριμένα αναφέρονται οι τεχνικές για το περιορισμό της εικόνας, για την εύρεση της ίριδας, για την εξάλειψη του τρέμουλου και του τρόπο υπολογισμού της θέσης του δείκτη στην οθόνη. Στη συνέχεια παρουσιάζεται η εκτέλεση του προγράμματος και τέλος τα συμπεράσματα.

7.2.1 Περιγραφή εφαρμογής

Η εφαρμογή χρησιμοποιεί το Blink Detection για το περιορισμό της εικόνας σε μία περιοχή γύρο από το μάτι. Σε αυτήν την περιορισμένη εικόνα θα χρησιμοποιηθεί σε template matching μία προ-αποθηκευμένη εικόνα της ίριδας του ματιού. Όπως φαίνεται

στην παρακάτω εικόνα, δεν περιέχεται μόνο η ίριδα αλλά και λίγο από το άσπρο του ματιού αριστερά και δεξιά της , καθώς και λίγη από την περιοχή των βλεφάρων.

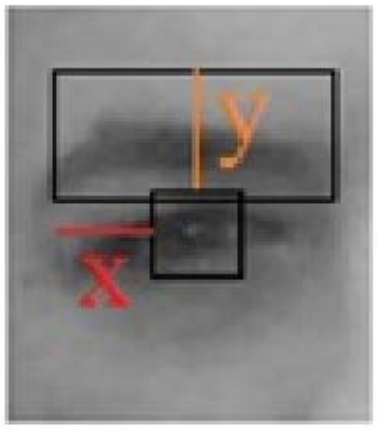
7-1 Προ-αποθηκευμένη εικόνα ίριδας. Η εικόνα είναι αποθηκευμένη πριν την έναρξη του προγράμματος και χρησιμοποιείται για την εύρεση της ίριδας του χρήστη.

Εικόνες με τέτοιου είδους χαρακτηριστικά είχαν πειραματικά τα καλύτερα αποτελέσματα. Αυτό οφείλετε στην έντονη αντίθεση που υπάρχει, η οποία ανιχνεύεται πιο εύκολα από την template matching της OpenCV, σε σχέση με εικόνες όπου υπάρχει μόνο η ίριδα.

Εκτός από την εικόνα της ίριδας, χρησιμοποιείται ένα ακόμα template matching μιας εικόνας του χρήστη της ευρύτερης περιοχής γύρω από το μάτι, η οποία δημιουργείται αυτόματα μετά την ολοκλήρωση του Blink Detection, και που χρησιμοποιείται για την σταθερότητα στη κίνηση του δείκτη του ποντικιού. Για ευκολία χρησιμοποιείται η περιοχή του φρυδιού. Ουσιαστικά, σκοπός του δεύτερου template είναι η εξάλειψη του τρέμουλου που υπάρχει στην περιοχή που ορίζεται από τη τεχνική Blink Detection. Επειδή με το template matching που χρησιμοποιείται για την εύρεση της περιορισμένης εικόνας κινείται η θέση της ίριδας, το δεύτερο template χρησιμοποιείται επίσης ως σημείο αναφοράς για να μην επηρεάζεται το τελικό αποτέλεσμα. Η δεύτερη αυτή εικόνα θα πρέπει να απεικονίζει ένα μεγάλο μέρος του φρυδιού, για να εντοπίζεται εύκολα χωρίς λάθη, και να απέχει από τα άκρα της περιορισμένης εικόνας, για καλύτερη απόσβεση. Επίσης, το μάτι θα πρέπει να αποτελεί όσο το δυνατόν μικρότερο μέρος της εικόνας για το δεύτερο template, διότι πρέπει η εικόνα να μην αποτελείτε από κινούμενα μέρη του προσώπου. Στην παρακάτω φωτογραφία φαίνεται η εικόνα που δημιούργησε το πρόγραμμα αυτόματα βασισμένο στα contour που αντιπροσωπεύουν τα μάτια.

7-2 Εικόνα χρήστη για την εξάλειψη του τρέμουλου. Όταν η περιορισμένη εικόνα αλλάζει, αλλάζει και η θέση τις ίριδας. Ένα δεύτερο template χρησιμοποιείται για την εξάλειψη της κίνησης αυτής.

Μετά τον καθορισμό των templates, γίνονται οι αρχικοποιήσεις που χρειάζονται για τον έλεγχο το δείκτη του ποντικιού με το βλέμμα. Οι τιμές αυτές προέρχονται από την διαφορά που υπάρχει ανάμεσα στη πάνω αριστερή γωνία του παραλληλόγραμμου που απεικονίζει την ίριδα, και στη πάνω αριστερή γωνία του παραλληλόγραμμου η οποία απεικονίζει τα περιεχόμενα της εικόνας που χρησιμοποιούμε για την εξάλειψη του τρέμουλου.

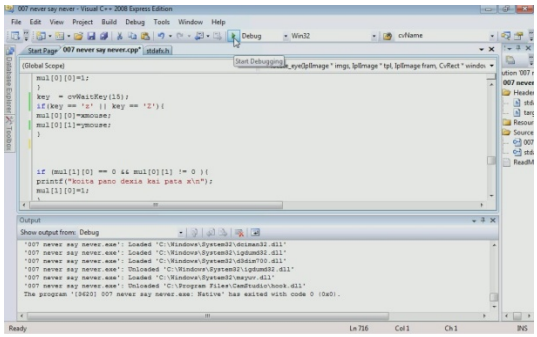


7-3 Υπολογισμός θέσης x, y του δείκτη του ποντικιού. Η θέση του δείκτη στην οθόνη εξαρτάται από τη διαφορά των δύο πάνω αριστερών γωνιών των rectangles.

Ακόμα, για την συσχέτιση τις θέσης της ίριδας στην εικόνα και του σημείου της οθόνης όπου κοιτά ο χρήστης χρειάζεται η διαδικασία της αρχικοποίησης. Κατά τη διάρκεια της αρχικοποίησης ζητείται από το χρήστη να κοιτάξει στις τέσσερις γωνίες της οθόνης και στο κάτω κέντρο της. Αφού ο χρήστης κοιτάξει σε μία από αυτές τις θέσεις χρειάζεται το πάτημα κουμπιού για να καταχωρηθεί η τιμή της στη θέση αυτή. Αφού ολοκληρωθεί η αρχικοποίηση, η τελική θέση του δείκτη είναι ανάλογη της θέσης της ίριδας στην εικόνα, και της ανάλυσης της οθόνης του υπολογιστή.

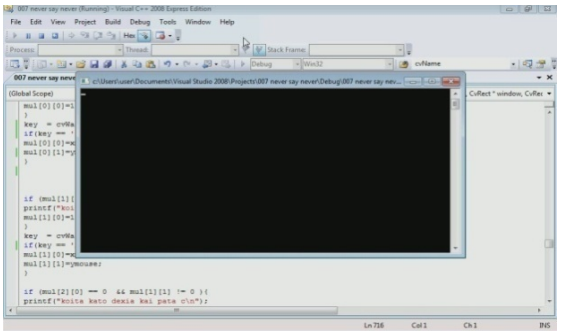
7.2.2 Εκτέλεση της εφαρμογής.

Παρακάτω θα περιγραφεί η ροή του προγράμματος πριν από την αρχικοποίηση των τιμών. Στην εικόνα φαίνεται το περιβάλλον της visual C++ express την οποία χρησιμοποιήσαμε για να δημιουργήσουμε τα προγράμματα μας.



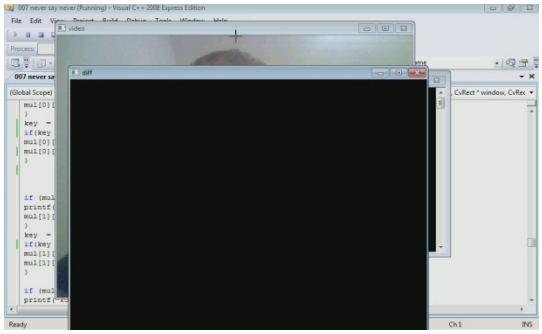
7-4 Περιβάλλον C++.

Στην επόμενη εικόνα βλέπουμε ότι έχει ξεκινήσει το πρόγραμμα να τρέχει. Μπορούμε να δούμε το command line το οποίο έχει εμφανιστεί και θα το χρησιμοποιήσουμε αργότερα για την αρχικοποίηση. Το command line χρησιμοποιήθηκε πολύ στο debugging των προγραμμάτων μας και λιγότερο στην τελική τους μορφή.



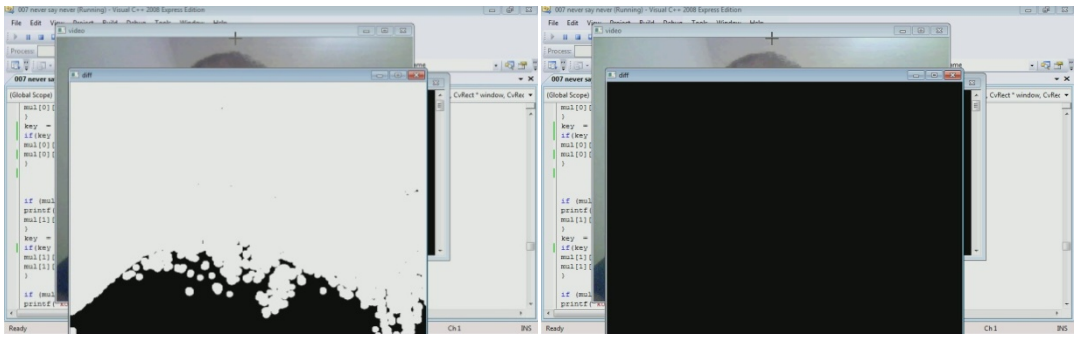
7-5 Εκκίνηση προγράμματος.

Στην εικόνα 7-6 βλέπουμε το παράθυρο της εικόνας της διαφοράς και από πίσω την εικόνα κατευθείαν από την web-cam του Η/Υ .



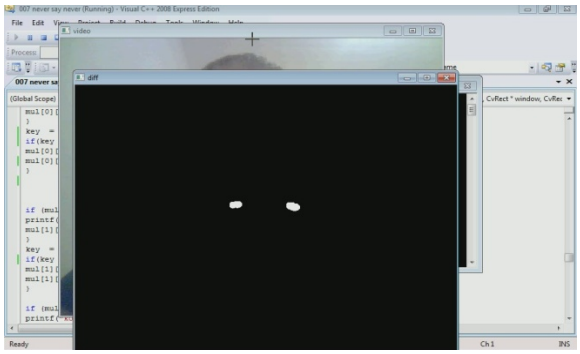
7-6 Παράθυρο εικόνας διαφορών διαδοχικών frames.

Στην εικόνα 7-7 φαίνεται μία από τις λάμψεις στην εικόνα της διαφοράς που πάντα παρατηρούμε τα πρώτα δύο τρία δευτερόλεπτα αφού ξεκινήσει το πρόγραμμα και οφείλεται στην αυτόματη ρύθμιση της κάμερας στο φως. Μετά τα πρώτα δευτερόλεπτα και αφού δεν υπάρχει κίνηση η εικόνα της διαφοράς εμφανίζεται προβλεπόμενα μαύρη .



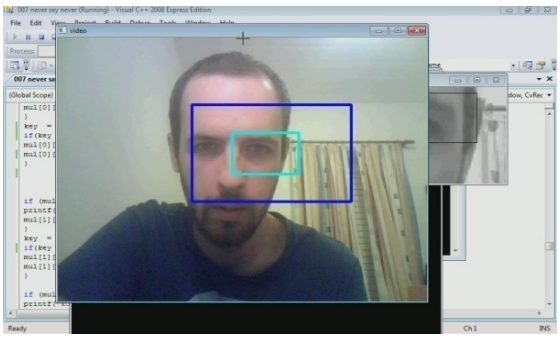
7-7 Αυτόματη ρύθμιση κάμερας. Η αυτόματη ρύθμιση της κάμερας στο φως δημιουργεί λανθασμένη εύρεση κίνησης.

Στην εικόνα 7-8 έχοντας το κεφάλι σταθερό κλείνουμε τα μάτια, και έχουμε επιτυχές αποτέλεσμα αφού βρέθηκαν τα δυο contour που αντιπροσωπεύουν τα μάτια χωρίς πολλές προσπάθειες. Αυτό σημαίνει ότι το threshold και το kernel μορφοποίησης είναι σωστά ρυθμισμένα για την συγκεκριμένη περίπτωση, καθώς επίσης και ο κώδικας που ελέγχει αν τα contour αντιπροσωπεύουν μάτια.



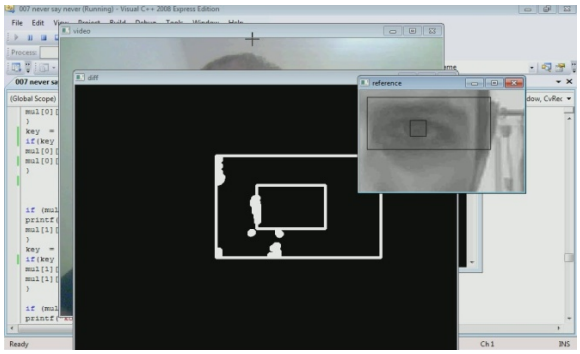
7-8 Εικόνα διαφοράς μετά από κλείσιμο ματιών.

Στην εικόνα 7-9 φαίνεται η εικόνα της web-cam με τα περιγράμματα των παραλληλόγραμμων που ορίζουν το αρχικό template matching στην εικόνα για την επικέντρωση στην περιοχή γύρο από το μάτι, το οποίο γίνεται για την μείωση του κόστους επεξεργασίας και για να μην χάνεται η περιοχή του ματιού όταν μετακινείται ο χρήστης.



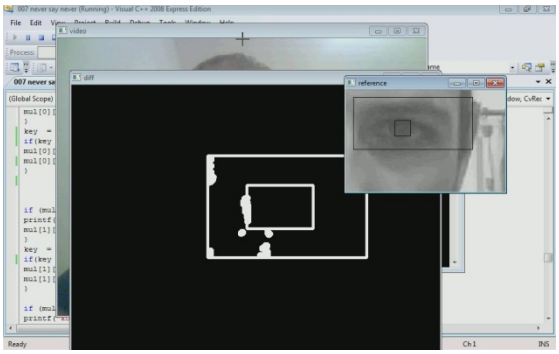
7-9 Αποτέλεσμα του Blink Detection.

Στην εικόνα 7-10 φαίνεται η δεύτερη φάση του προγράμματος όπου έχει εντοπιστεί η ίριδα και η περιοχή του προσώπου πάνω από το μάτι η οποία χρησιμοποιείται για την απόσβεση του τρέμουλου.



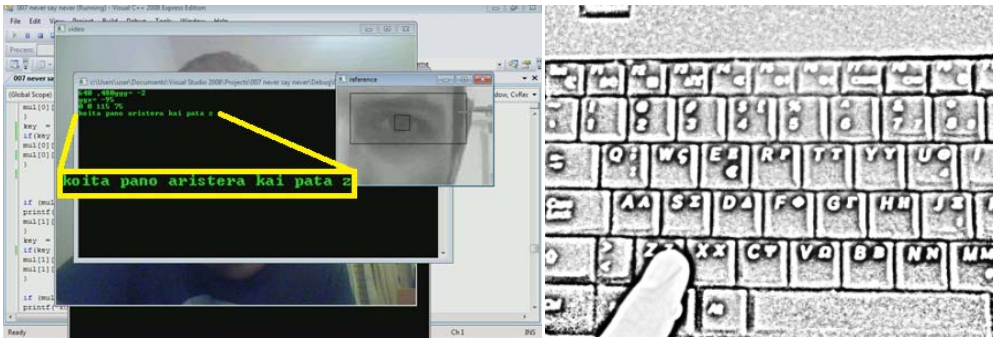
7-10 Εύρεση ίριδας και εικόνας σταθερότητας.

Στην εικόνα 7-11 παρουσιάζεται το πρόγραμμα πριν από την αρχικοποίηση. Όπως ξεκάθαρα φαίνεται το μεγάλο παραλληλόγραμμο μέσα στο παράθυρο έχει αρκετή απόσταση από τα άκρα της εικόνας και είναι αρκετά μεγάλο ώστε να μην επηρεάζεται η κίνηση του από την κίνηση του ματιού. Επίσης βλέπουμε ότι η ίριδα έχει εντοπιστεί αρκετά καλά.



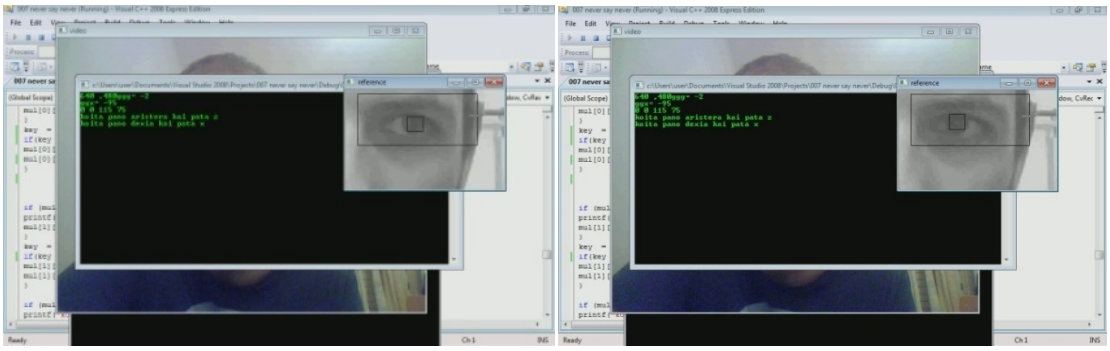
7-11 Η εικόνα του προγράμματος πριν την αρχικοποίηση.

Πλέον η εφαρμογή μπαίνει στην διαδικασία της αρχικοποίησης. Στο προσκήνιο υπάρχει το παράθυρο του command line όπου φαίνεται το μήνυμα "κοίτα πάνω αριστερά και πάτα z". Ο χρήστης το μόνο που έχει να κάνει είναι ακολουθήσει τις οδηγίες δηλαδή να κοιτάξει στην πάνω αριστερή γωνία της οθόνης και να πατήσει το κουμπί z. Είναι πολύ σημαντικό κατά την διάρκεια της αρχικοποίησης αλλά και για το υπόλοιπο της λειτουργίας του προγράμματος το κεφάλι του χρήστη να είναι σταθερό. Στο πρόγραμμα αυτό δεν λαμβάνεται υπόψη κατά και μετά την αρχικοποίηση των τιμών η κίνηση του κεφαλιού. Αν η κίνηση πραγματοποιηθεί μετά την αρχικοποίηση, τότε θα υπάρξει κίνηση του δείκτη του ποντικιού αλλά η συντεταγμένες του θα απέχουν από τις επιθυμητές.



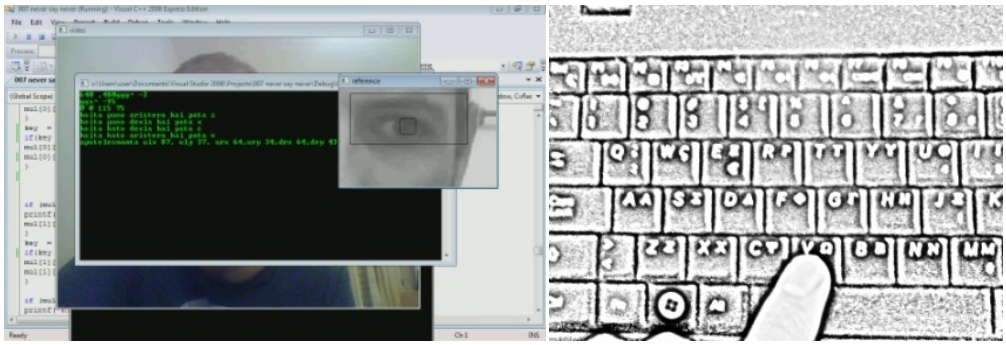
7-12 Βήματα αρχικοποίησης. Όσο ο χρήστης κοιτάζει πάνω αριστερά πατά το κουμπί 'z' και η θέση της ίριδας αποθηκεύεται ως μέγιστη y ελάχιστη x τιμή.

Αφού πατηθεί το κουμπί "z" ,θα ζητηθεί από τον χρήστη να κοιτάξει πάνω δεξιά και να πατήσει το κουμπί "x".Το ίδιο θα πρέπει να γίνει για κάτω δεξιά και κάτω αριστερά πατώντας "c" και "v" αντίστοιχα.

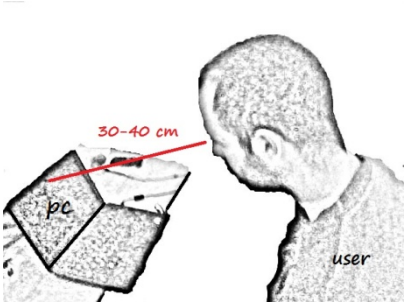


7-13 Βήματα αρχικοποίησης 2. Με τον ίδιο τρόπο βρίσκονται και οι τιμές x max και y min.

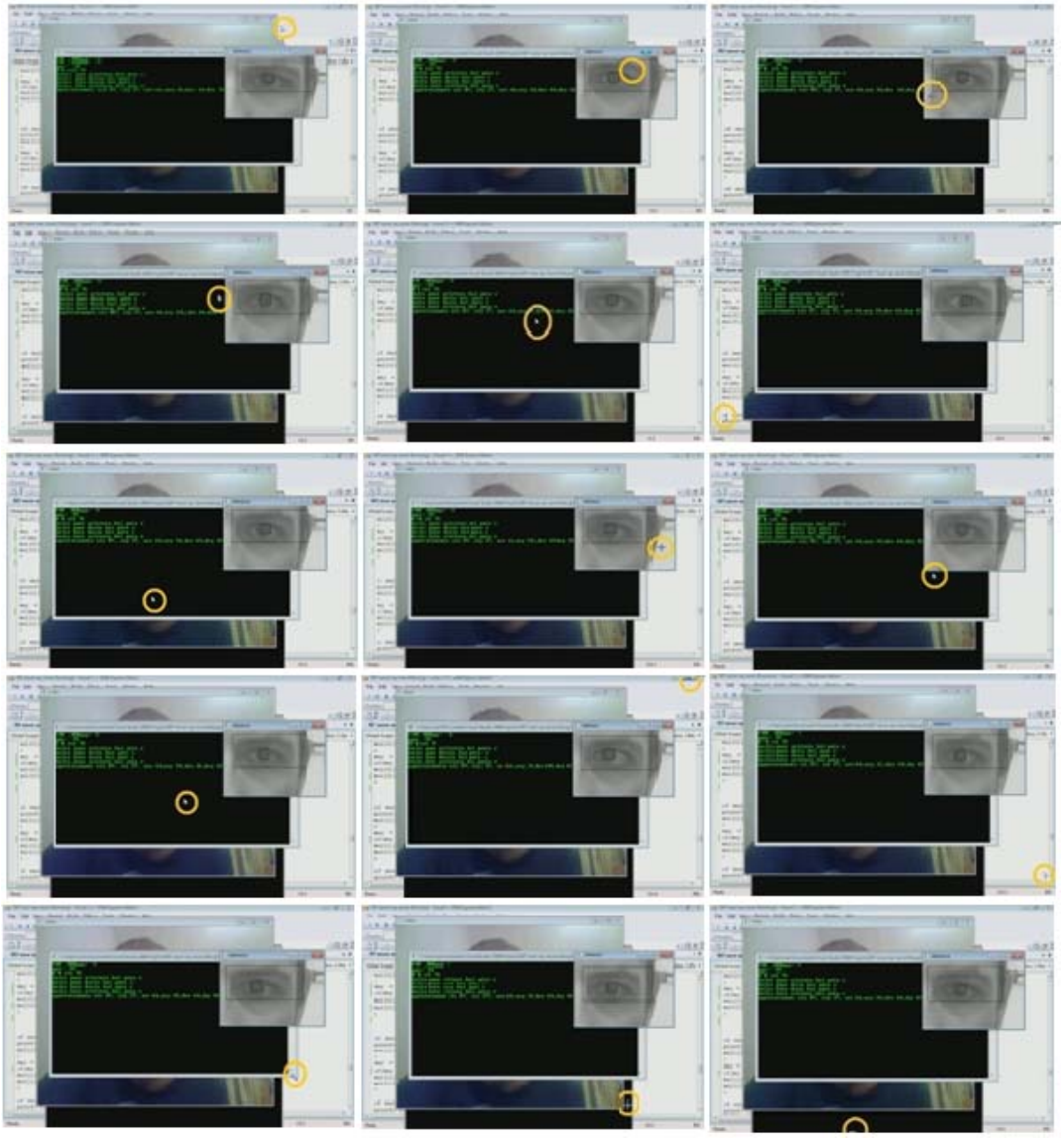
Με το πάτημα του κουμπιού "v" η αρχικοποίηση ολοκληρώνεται και ο δείκτης κινείται πλέον με βάση την κατεύθυνση του βλέμματος. Εφόσον η αρχικοποίηση έχει γίνει σωστά η κίνηση του δείκτη είναι καλή, όμως δεν θα πλησιάζει την ακρίβεια που παρέχει ένα "ποντίκι".



7-14 Βήματα αρχικοποίησης 3. Με το πάτημα του κουμπιού 'v' η διαδικασία ολοκληρώνεται.

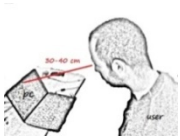
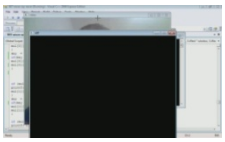
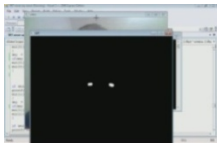
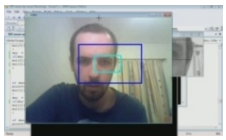
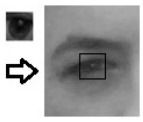
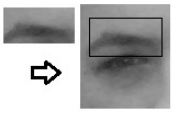
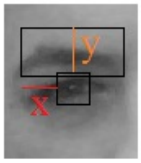
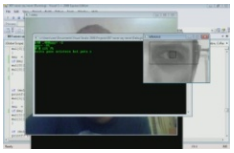
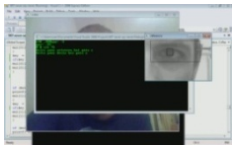
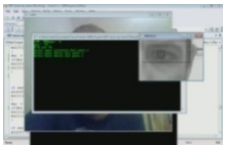
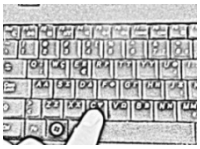
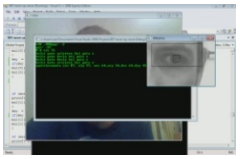
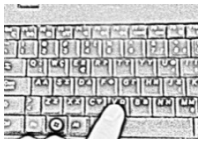
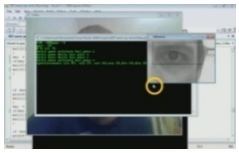
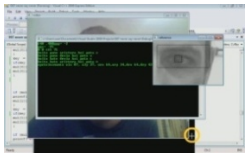
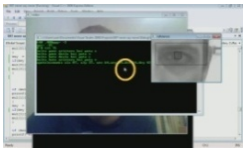


Με ένα net-book με οθόνη 10.1 ιντσών και τα μάτια μας σε απόσταση 30 εκατοστών ο δείκτης κινούταν μέσα σε χώρο η περιφέρεια του οποίου βρισκόταν 2 εκατοστά από το σημείο που κοιτούσαμε. Σε φορητό υπολογιστή με οθόνη 17 ιντσών και τα μάτια σε απόσταση 40 εκατοστών υπήρχε ακρίβεια 3 εκατοστών. Στις επόμενες εικόνες βλέπουμε τη θέση του δείκτη σε σχέση με τη θέση της ίριδας.Για ευκολία έχουμε κυκλώσει τον δείκτη του ποντικιού με έντονο χρώμα.



7-15 Κίνηση δείκτη με την ίριδα. Κρατώντας το κεφάλι σταθερό είναι δυνατή η κίνηση του δείκτη του ποντικιού ανάλογα με το που κοιτά ο χρήστης.

7.2.3 Συνοπτικός πίνακας αρχικής εφαρμογής

1 Χρήστης-ΗΥ . 	2 Έναρξη προγράμματος. 	3 Παράθυρο εντοπισμού κίνησης. 	4 Αρχικά frame. 
5 Εντοπισμός κίνησης ματιών. 	6 Δημιουργία περιορισμένου παραθύρου γύρο από το μάτι. 	7 Περιορισμένο παράθυρο περιοχής ματιού. 	8 Αποθηκευμένη εικόνα ίριδας. 
9 Template matching ίριδας. 	10 Εικόνα για απόσβεση τρεμοπαίγματος. 	11 Template matching εικόνας απόσβεσης τρεμοπαίγματος. 	12 Πρόγραμμα πριν την αρχικοποίηση. 
13 Απεικόνιση τρόπου λήψης δεδομένων 	14 Μήνυμα αρχικοποίησης για πάνω αριστερά. 	15 Πάτημα πλήκτρου ‘z’ 	16 Μήνυμα αρχικοποίησης για πάνω δεξιά. 
17 Πάτημα πλήκτρου ‘x’ 	18 Μήνυμα αρχικοποίησης για κάτω δεξιά. 	19 Πάτημα πλήκτρου ‘c’ 	20 Μήνυμα αρχικοποίησης για πάνω αριστερά. 
21 Πάτημα πλήκτρου ‘v’  \	22 Απεικόνιση ελέγχου του κέρσορα με τα μάτια. 	23 Απεικόνιση ελέγχου του κέρσορα με τα μάτια. 	24 Απεικόνιση ελέγχου του κέρσορα με τα μάτια. 

Πίνακας 11 Συνοπτικός πίνακας αρχικής εφαρμογής.

7.2.4 Συμπεράσματα από την εκτέλεση της αρχικής εφαρμογής

Η χρήση μίας προ-αποθηκευμένης εικόνας για την εύρεση της ίριδας είναι καλή λύση μόνο στη περίπτωση που η προ-αποθηκευμένη ίριδα είναι του χρήστη που χρησιμοποιεί το πρόγραμμα. Αν ο χρήστης είναι διαφορετικός η υπάρχει περίπτωση η ίριδα να μην βρεθεί σωστά ή και καθόλου. Ακόμα υπάρχει η περίπτωση να αλλάξουν οι συνθήκες φωτισμού ή η απόσταση του χρήστη, οπότε υπάρχουν και πάλι προβλήματα.

Η διαδικασία της αρχικοποίησης εξαρτάται από το χρήστη. Αν κατά τη διάρκεια της παρθούν λάθος τιμές, τότε ο δείκτης του ποντικιού θα παίρνει λάθος θέσεις.

Η θέση του δείκτη στην οθόνη βρίσκεται με σχετική ακρίβεια που εξαρτάται από την απόσταση του χρήστη, την ανάλυση της οθόνης και το μέγεθος της περιορισμένης εικόνας. Παρά τη χρησιμοποίηση της εικόνας σταθερότητας, ο δείκτης συνεχίζει να τρεμοπαίζει αν και το τρεμόπαιγμα αυτό μειώνεται.

Έλεγχος κέρσορα Η/Υ με τα μάτια χρησιμοποιώντας προ-αποθηκευμένη φωτογραφία ίριδας.	
Εύρεση ίριδας	Στην περίπτωση που η προ-αποθηκευμένη εικόνα ίριδας είναι του χρήστη, η εύρεση γίνεται σωστά. Σε περιπτώσεις που ο χρήστης είναι διαφορετικός υπάρχουν προβλήματα εύρεσης. Ακόμα υπάρχουν προβλήματα αν η προ-αποθηκευμένη εικόνα έχει διαφορετική φωτεινότητα από την περιορισμένη εικόνα .
Αρχικοποίηση	Η διαδικασία δεν είναι αυτόματη, και χρειάζεται τη σωστή εκτέλεσή της από το χρήστη.
Σταθερότητα δείκτη	Η εικόνα που χρησιμοποιείται για την εξάλειψη του τρεμουλιάσματος μειώνει το τρεμούλιασμα αλλά δεν το εξαφανίζει πλήρως.

Πίνακας 12 Συμπεράσματα έλεγχου κέρσορα με χρήση προ-αποθηκευμένης εικόνας ίριδας.

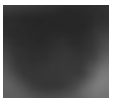
7.3 Βελτίωση της εφαρμογής με την χρησιμοποίηση της ίριδας του εκάστοτε χρηστή

Επιδιώκοντας την βελτίωση της απόδοσης του προγράμματος έγινε προσπάθεια χρησιμοποίησης της ίριδας του εκάστοτε χρήστη καθώς είναι δύσκολο να υπάρχουν πάντα καλά αποτελέσματα με μια αποθηκευμένη ίριδα, μιας και μπορεί να διαφέρει αρκετά από αυτή του χρήστη. Σίγουρα, υπάρχει δυνατότητα αποθήκευσης μιας φωτογραφίας για κάθε νέο χρήστη επεξεργασμένη με το χέρι ώστε να περιλαμβάνει μόνο την επιθυμητή περιοχή της ίριδας, αλλά αυτό θα άλλαζε την λειτουργικότητα του προγράμματος ως προς τη δυνατότητα αυτοματοποίησης.

Το πρόβλημα αυτό επιλύθηκε με τον εξής τρόπο. Χρησιμοποιώντας σε template matching την εικόνα της περιοχής ενός από τα contour που βρέθηκαν με το κλείσιμο των βλεφάρων, και της εικόνας της ίριδας που χρησιμοποιήθηκαν στην προηγούμενη εφαρμογή αφού πρώτα μετασχηματιστεί έτσι ώστε να έχει ύψος ελάχιστα μικρότερο από το ύψος του ορθογώνιου του contour .

7-16 Η περιοχή που αυτόματα επιλέχτηκε γύρο από το contour που δημιουργήθηκε με το ανοιγοκλείσιμο του ματιού.

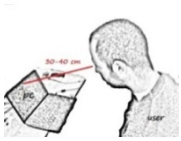
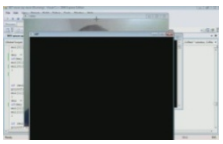
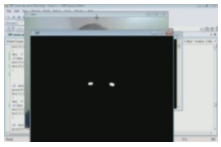
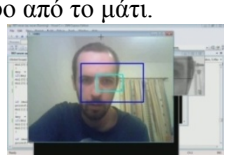
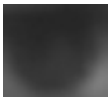
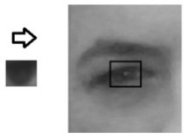
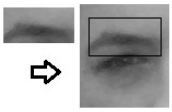
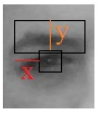
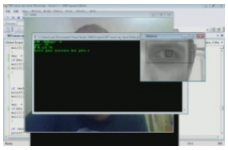
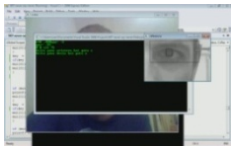
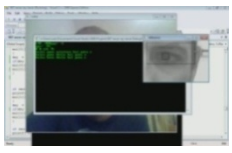
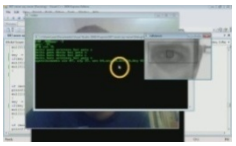
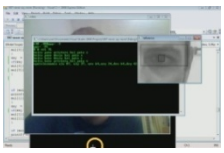
Το αποτέλεσμα της σύγκρισης αυτής φαίνεται στην εικόνα 7-17. Εύκολα διακρίνεται ότι η εικόνα είναι εικόνα ίριδας αλλά η ποιότητα της είναι πολύ κακή. Το θετικό όμως είναι ότι η εικόνα αν και πιο χαμηλής ποιότητας από την ίριδα της εικόνας 7-1, πλέον αντιπροσωπεύει την ίριδα του εκάστοτε χρήστη προσφέροντας καλύτερα και πιο σταθερά αποτελέσματα από το να χρησιμοποιηθεί κατευθείαν η αποθηκευμένη ίριδα. Βέβαια αυτό δεν ισχύει στην περίπτωση που χρήστης είναι αυτός από τον οποίο πήραμε το αρχικό δείγμα της ίριδας. Τότε θα ήταν προτιμότερο να χρησιμοποιηθεί η εικόνα της αποθηκευμένης ίριδας η οποία θα είχε πολύ καλύτερα αποτελέσματα.



7-17 Το αποτέλεσμα από την εφαρμογή template matching μεταξύ των εικόνων 7-1 και 7-16 το οποίο αντιπροσωπεύει πλέον την ίριδα.

Εφόσον το contour που εντοπίστηκε με την ανίχνευση της κίνησης έχει παραπλήσιο μέγεθος με αυτό του ματιού τότε η αρχική εικόνα της ίριδας που χρησιμοποιήθηκε θα μετασχηματιστεί σωστά και θα έχουμε πολύ καλά αποτελέσματα. Αν όμως το contour δημιουργηθεί με βάση την κίνηση ενός μόνο μέρους του ματιού ή δημιουργηθεί από μια ταυτόχρονη κίνηση και άλλου μέρους του προσώπου και ο κώδικας μας δεν το αποτρέψει, τότε θα χρειαστεί να ξεκινήσουμε πάλι την διαδικασία ανίχνευσης της κίνησης του κλεισίματος των ματιών .

7.3.1 Συνοπτικός πίνακας πρώτης βελτίωσης εφαρμογής.

1 Χρήστης-ΗΥ . 	2 Έναρξη προγράμματος. 	3 Παράθυρο εντοπισμού κίνησης. 	4 Αρχικά frame. 
5 Εντοπισμός κίνησης ματιών. 	6 Δημιουργία περιορισμένου παραθύρου γύρο από το μάτι. 	7 Περιορισμένο παράθυρο περιοχής ματιού. 	8 Αποθηκευμένη εικόνα ίριδας. 
9 Εικόνα περιοχής contour. 	10 Template matching περιοχής contour-αποθηκευμένης ίριδας. 	11 Αποτέλεσμα template matching. 	12 Template matching ίριδας χρήστη. 
13 Εικόνα για απόσβεση τρεμοπαίγματος. 	14 Template matching εικόνας απόσβεσης τρεμοπαίγματος. 	15 Πρόγραμμα πριν την αρχικοποίηση. 	16 Απεικόνιση τρόπου λήψης δεδομένων 
17 Μήνυμα αρχικοποίησης για πάνω αριστερά. 	18 Πάτημα πλήκτρου ‘z’ 	19 Μήνυμα αρχικοποίησης για τάνω δεξιά. 	20 Πάτημα πλήκτρου ‘x’ 
21 Μηνύματα αρχικοποίησης για κάτω δεξιά και κάτω αριστερά. 	22 Πάτημα πλήκτρων ‘c’και ‘v’. 	23 Απεικόνιση ελέγχου του κέρσορα με τα μάτια 	24 Απεικόνιση ελέγχου του κέρσορα με τα μάτια. 

Πίνακας 11 Συνοπτικός πίνακας πρώτης βελτίωσης εφαρμογής.

7.4 Βελτίωση εφαρμογής χρησιμοποιώντας τον συνδυασμό cvHoughCircles() και των σκοτεινότερων pixel.

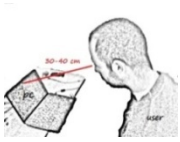
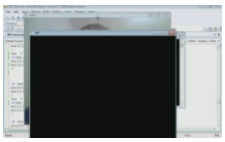
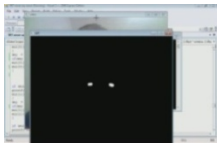
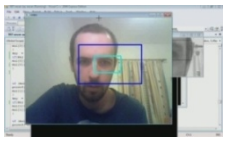
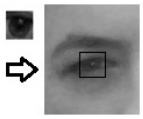
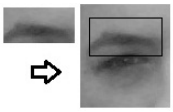
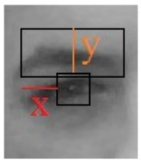
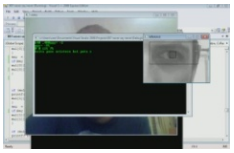
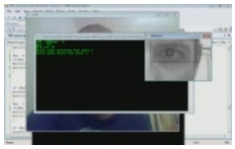
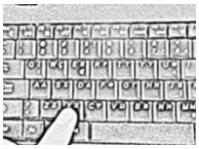
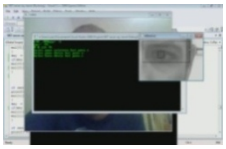
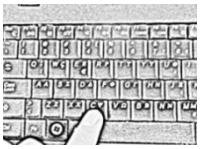
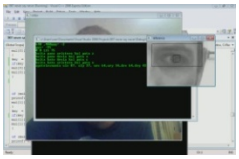
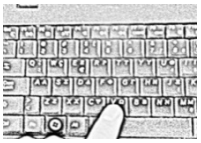
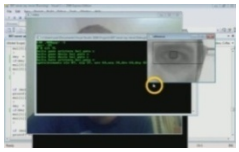
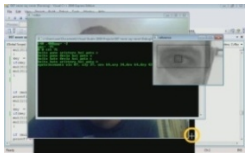
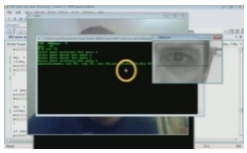
Για την επίλυση των προβλημάτων των παραπάνω μεθόδων χρησιμοποιήθηκε ο συνδυασμός της ανίχνευσης κύκλων cvHoughCircles() με την ανίχνευση της κόρης του ματιού ψάχνοντας τα σκουρότερα pixel. Σε αντίθεση με τις άλλες μεθόδους, αυτή η παραλλαγή δεν βασίζεται σε μία προ-αποθηκευμένη εικόνα. Αρχικά βρίσκονται κύκλοι στην περιορισμένη εικόνα (στην περιοχή γύρο από το μάτι), και ύστερα επιλέγονται προς εμφάνιση μόνο οι κύκλοι των οποίων στο κέντρο τους βρίσκεται ένα από τα σκουρότερα pixel. Για τα καλύτερα δυνατά αποτελέσματα με τη συγκεκριμένη τεχνική, είναι σημαντικό τα ορίσματα της cvHoughCircles() να έχουν ρυθμιστεί έτσι ώστε να βρίσκονται μόνο κύκλοι οι οποίοι απεικονίζουν την ίριδα, και τα σκούρα pixel που εντοπίζονται να αντιπροσωπεύουν όσο γίνεται στην κόρη του ματιού.

Αφού η ίριδα του χρήστη βρεθεί τότε αποθηκεύεται ως εικόνα και χρησιμοποιείται για το template matching, ώστε να βρίσκεται η θέση της στην εικόνα. Για την εύρεση της θέση του κέρσορα χρησιμοποιείται εικόνα σταθερότητας όπως και στις προηγούμενες περιπτώσεις.

Εύρεση ίριδας και template matching			
Μέθοδος	**Έλεγχος κέρσορα Η/Υ με τα μάτια χρησιμοποιώντας προ-αποθηκευμένη φωτογραφία ίριδας.**	**Έλεγχος κέρσορα Η/Υ με τα μάτια χρησιμοποιώντας την ίριδα του χρηστή.**	**Εύρεση ίριδας χρησιμοποιώντας τον συνδυασμό cvHoughCircles() και των σκοτεινότερων pixel.**
Σχόλια	Στην περίπτωση που η προ-αποθηκευμένη εικόνα ίριδας είναι του χρήστη, η εύρεση γίνεται σωστά. Σε περιπτώσεις που ο χρήστης είναι διαφορετικός υπάρχουν προβλήματα εύρεσης. Ακόμα υπάρχουν προβλήματα αν η προ-αποθηκευμένη εικόνα έχει διαφορετική φωτεινότητα από την περιορισμένη εικόνα .	Για την εύρεση της ίριδας χρησιμοποιείται μία προ-αποθηκευμένη εικόνα, αλλά για το template matching εικόνα ίριδας του χρήστη. Με τη μέθοδο αυτή λύνονται τα προβλήματα φωτεινότητας και απόστασης.	Η εικόνα της ίριδας που χρησιμοποιείται για το template matching είναι αποτέλεσμα της ανίχνευσης ίριδας. Αυτό λύνει τα προβλήματα διαφορετικών χρηστών, φωτεινότητας, και απόστασης χρήστη.

Πίνακας 14 Συγκριτικός πίνακας αποτελεσμάτων μεθόδων εύρεσης ίριδας.

7.4.1 Συνοπτικός πίνακας δεύτερης βελτίωσης εφαρμογής.

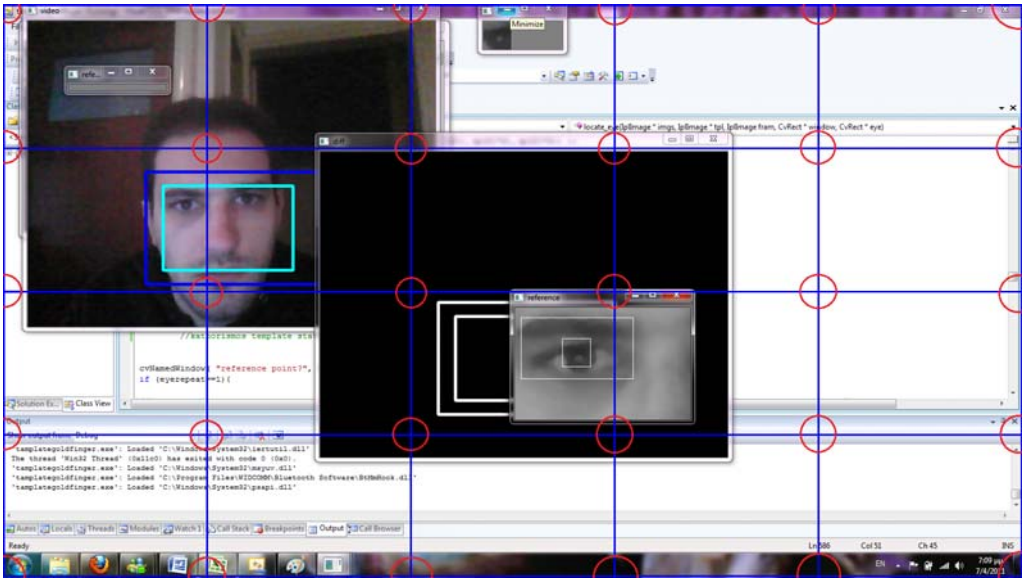
1 Χρήστης-ΗΥ . 	2 Έναρξη προγράμματος. 	3 Παράθυρο εντοπισμού κίνησης. 	4 Αρχικά frame. 
5 Εντοπισμός κίνησης ματιών. 	6 Δημιουργία περιορισμένου παραθύρου γύρο από το μάτι. 	7 Περιορισμένο παράθυρο περιοχής ματιού. 	8 Συνδυασμός cvHoughCircles() και των σκοτεινότερων pixel. 
9 Template matching ίριδας. 	10 Εικόνα για απόσβεση τρεμοπαίγματος. 	11 Template matching εικόνας απόσβεσης τρεμοπαίγματος. 	12 Πρόγραμμα πριν την αρχικοποίηση. 
13 Απεικόνιση τρόπου λήψης δεδομένων 	14 Μήνυμα αρχικοποίησης για πάνω αριστερά. 	15 Πάτημα πλήκτρου ‘z’ 	16 Μήνυμα αρχικοποίησης για πάνω δεξιά. 
17 Πάτημα πλήκτρου ‘x’ 	18 Μήνυμα αρχικοποίησης για κάτω δεξιά. 	19 Πάτημα πλήκτρου ‘c’ 	20 Μήνυμα αρχικοποίησης για πάνω αριστερά. 
21 Πάτημα πλήκτρου ‘v’ 	22 Απεικόνιση ελέγχου του κέρσορα με τα μάτια. 	23 Απεικόνιση ελέγχου του κέρσορα με τα μάτια. 	24 Απεικόνιση ελέγχου του κέρσορα με τα μάτια. 

Πίνακας 11 Συνοπτικός πίνακας δεύτερης βελτίωσης εφαρμογής.

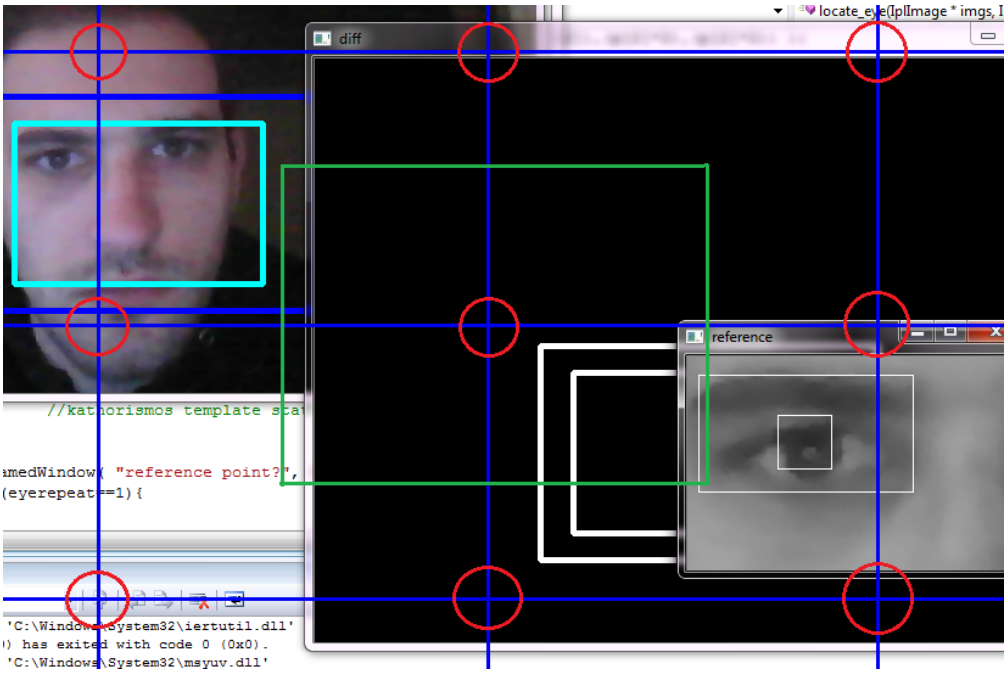
7.5 Δημιουργία συγκεκριμένων θέσεων για τον κέρσορα.

Παρότι η χρησιμοποίηση μίας δεύτερης εικόνας για την αντιμετώπιση του τρεμουλιάσματος που οφείλεται στο template matching της περιορισμένης εικόνας έχει αποτέλεσμα, το τρεμούλιασμα συνεχίζει να υπάρχει. Για την αντιμετώπιση του προβλήματος αυτού δίνονται συγκεκριμένες θέσεις για τον δείκτη του ποντικιού.

Η οθόνη χωρίζεται σε κομμάτια ίδιων μεγεθών για τον άξονα x και το ίδιο γίνετε για τον άξονα y, δημιουργώντας έτσι ένα φανταστικό πλέγμα πιθανών θέσεων για το δείκτη. Η τελική θέση του δείκτη είναι η πιο κοντινή θέση σε σχέση με την πραγματική του. Η επιλογή του αριθμού των πιθανών θέσεων πρέπει να γίνεται βάση το μέγεθος της οθόνης. Για τον ίδιο αριθμό πιθανών θέσεων σε μία μικρή οθόνη υπάρχει λιγότερη ακρίβεια από ότι σε μία μεγαλύτερη.



7-18 Πιθανές θέσεις δείκτη ποντικιού μετά από το καθορισμό τους. Οι κόκκινοι κύκλοι είναι πιθανές θέσεις για το δείκτη.



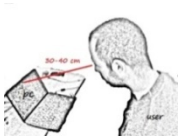
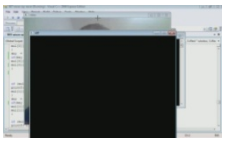
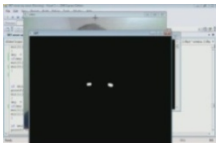
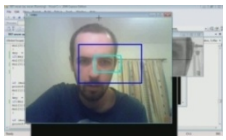
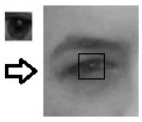
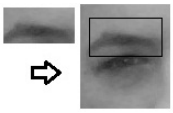
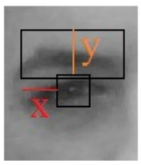
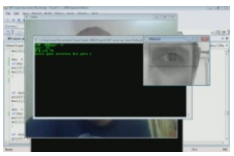
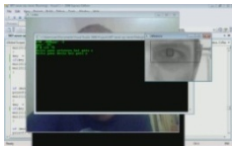
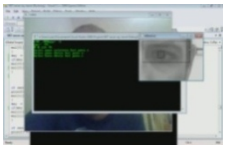
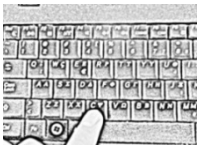
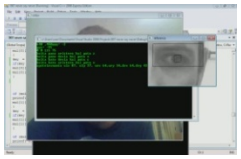
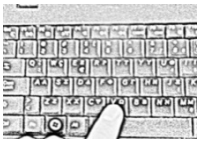
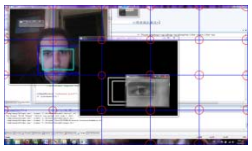
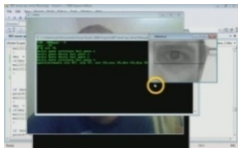
7-19 Τελική θέση δείκτη ποντικιού σε καθορισμένες θέσεις. Για να βρεθεί ο δείκτης στο κεντρικό κύκλο θα πρέπει η πραγματική του θέση να είναι μέσα στο πράσινο ορθογώνιο.

Χρησιμοποιώντας συγκεκριμένες θέσεις για το δείκτη του ποντικιού πετυχαίνεται ο εξαφανισμός του τρεμουλιάσματος, αλλά η χρήση του προϋποθέτει μια διεπαφή στην οποία οι πιθανές θέσεις του δείκτη θα βρίσκονται σε σημεία τέτοια έτσι ώστε να μπορούν να εκτελεστούν όλες οι βασικές λειτουργίες.

Θέση Δείκτη	**Χωρίς περιορισμό**	**Συγκεκριμένες θέσεις**
Σχόλια	Ο δείκτης του ποντικιού παρουσιάζει αστάθεια, ενώ η κίνηση του σε σχέση με το βλέμμα του χρήστη έχει σχετική ακρίβεια.	Ο δείκτης του ποντικιού είναι σταθερότερος. Οι πιθανές θέσεις του δείκτη θα πρέπει να αντιστοιχούν σε λειτουργίες.

Πίνακας 16 Πίνακας συμπερασμάτων για συγκεκριμένες θέσεις του δείκτη και θέσεις χωρίς περιορισμούς.

7.5.1 Συνοπτικός πίνακας τρίτης βελτίωσης εφαρμογής.

1 Χρήστης-ΗΥ . 	2 Έναρξη προγράμματος. 	3 Παράθυρο εντοπισμού κίνησης. 	4 Αρχικά frame. 
5 Εντοπισμός κίνησης ματιών. 	6 Δημιουργία περιορισμένου παραθύρου γύρο από το μάτι. 	7 Περιορισμένο παράθυρο περιοχής ματιού. 	8 Συνδυασμός cvHoughCircles() και των σκοτεινότερων pixel. 
9 Template matching ίριδας. 	10 Εικόνα για απόσβεση τρεμοπαίγματος. 	11 Template matching εικόνας απόσβεσης τρεμοπαίγματος. 	12 Πρόγραμμα πριν την αρχικοποίηση. 
13 Απεικόνιση τρόπου λήψης δεδομένων 	14 Μήνυμα αρχικοποίησης για πάνω αριστερά. 	15 Πάτημα πλήκτρου ‘z’ 	16 Μήνυμα αρχικοποίησης για πάνω δεξιά. 
17 Πάτημα πλήκτρου ‘x’ 	18 Μήνυμα αρχικοποίησης για κάτω δεξιά. 	19 Πάτημα πλήκτρου ‘c’ 	20 Μήνυμα αρχικοποίησης για πάνω αριστερά. 
21 Πάτημα πλήκτρου ‘v’ 	22 Τοποθέτηση του κέρσορα σε συγκεκριμένα σημεία, λίγες θέσεις. 	23 Τοποθέτηση του κέρσορα σε συγκεκριμένα σημεία, πολλές θέσεις. 	24 Απεικόνιση ελέγχου του κέρσορα με τα μάτια. 

Πίνακας 17 Συνοπτικός πίνακας τρίτης βελτίωσης εφαρμογής.

7.6 Αυτοματοποιημένη αρχικοποίηση εφαρμογής.

Στην αρχική εφαρμογή η αρχικοποίηση γίνεται χειροκίνητα από το χρήστη. Παρότι τα αποτελέσματα είναι αποδεκτά, θα ήταν δύσκολο να χρησιμοποιηθεί από άτομα με κινητικές δυσκολίες δίχως την βοήθεια τρίτου. Έτσι κρίθηκε απαραίτητη η ανάπτυξη μεθόδου με αυτόματη αρχικοποίηση.

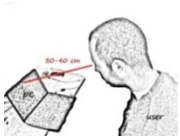
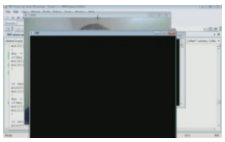
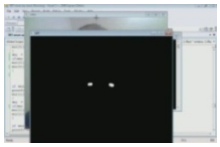
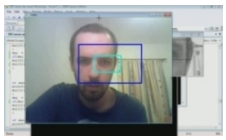
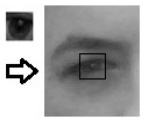
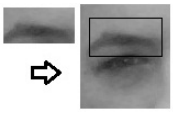
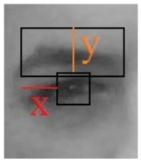
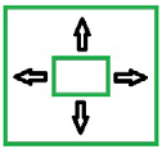
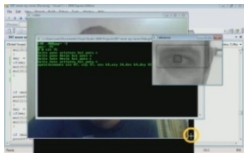
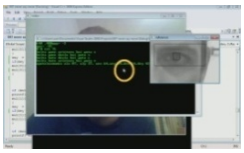
Η λειτουργία της αυτόματης αρχικοποίησης της εφαρμογής, ξεκινά μόλις βρεθεί η ίριδα του χρήστη με την τεχνική που περιγράφεται στο κεφάλαιο 6.5. Τότε για ένα χρονικό διάστημα δέκα περίπου δευτερολέπτων, μετρώντας frame, δημιουργούνται τα όρια ενός υποθετικού παραθύρου από το πρόγραμμα τα οποία μεγαλώνουν ανάλογα με το πόσο μακριά από το κέντρο κοιτάει ο χρήστης.

Αρχικά, το υποθετικό παράθυρο είναι μηδενικού μεγέθους και βρίσκεται στο κέντρο της οθόνης. Ο χρήστης, μέσα στο χρονικό όριο της αρχικοποίησης, πρέπει να κοιτάξει στις 4 γωνίες της οθόνης δίχως να κοιτάξει έξω από αυτήν. Κοιτώντας εκτός οθόνης κατά την διάρκεια της αρχικοποίησης θα καθιστούσε το υποθετικό παράθυρο μεγαλύτερο της οθόνης ανάλογα με το πόσο έξω από αυτήν κοίταξε και ο κέρσορας θα κουνιόταν μετά λανθασμένα. Μόλις τελειώσει ο χρόνος της αρχικοποίησης που έχουμε ορίσει εμφανίζεται μήνυμα ‘αρχικοποίηση τέλος’ και ο χρήστης ελέγχει πλέον τον κέρσορα με τα μάτια. Μπορεί πλέον να κοιτάξει και εκτός οθόνης χωρίς να υπάρχει πρόβλημα.

Αρχικοποίηση προγράμματος	
Μη αυτόματη	Χρειάζεται πάτημα πλήκτρων και σωστή εκτέλεση από το χρήστη.
Αυτόματη	Επιτυγχάνεται με τη κίνηση της ίριδας, χρειάζεται σωστή εκτέλεση από το χρήστη.

Πίνακας 18 Πίνακας συμπερασμάτων αυτόματης και μη αυτόματης αρχικοποίησης του προγράμματος.

7.6.1 Συνοπτικός πίνακας τέταρτης βελτίωσης εφαρμογής

1 Χρήστης-ΗΥ . 	2 Έναρξη προγράμματος. 	3 Παράθυρο εντοπισμού κίνησης. 	4 Αρχικά frame. 
5 Εντοπισμός κίνησης ματιών. 	6 Δημιουργία περιορισμένου παραθύρου γύρο από το μάτι. 	7 Περιορισμένο παράθυρο περιοχής ματιού. 	8 Συνδυασμός cvHoughCircles() και των σκοτεινότερων pixel. 
9 Template matching ίριδας. 	10 Εικόνα για απόσβεση τρεμοπαίγματος. 	11 Template matching εικόνας απόσβεσης τρεμοπαίγματος. 	12 Πρόγραμμα πριν την αρχικοποίηση. 
13 Απεικόνιση τρόπου λήψης δεδομένων. 	14 Αυτοματοποιημένη αρχικοποίηση. 	15 Απεικόνιση ελέγχου του κέρσορα με τα μάτια. 	16 Απεικόνιση ελέγχου του κέρσορα με τα μάτια. 

Πίνακας 19 Συνοπτικός πίνακας τέταρτης βελτίωσης εφαρμογής.

8. Βιβλιογραφία

[1] Gary Bradski,Adrian Kaebler(Σεπτέμβριος 2008) <<Learning OpenCV Computer Vision with the OpenCV Library>> O'REILLY MEDIA

[2] http://eyetrackingupdate.com/2010/03/02/eye-tracking-history-early-eye-tracking-apparatus/

[3] http://en.wikipedia.org/wiki/Eye_tracking

[4] [http://en.wikipedia.org/wiki/Canon_EOS_5, http://www.mir.com.my/rb/photography/hardwares/classics/eos/eoscamera/EOS5A2EQD/index.htm

[5] http://en.wikipedia.org/wiki/Iris_recognition

[6] http://nashruddin.com/

[7] OpenCV Reference Manualv2.1 (Μάρτιος 2010) διαθέσιμο από: http://public.cranfield.ac.uk/c5354/teaching/dip/opencv/manual/opencv.pdf

[8] http://dasl.mem.drexel.edu/~noahKuntz/openCVTut4.html

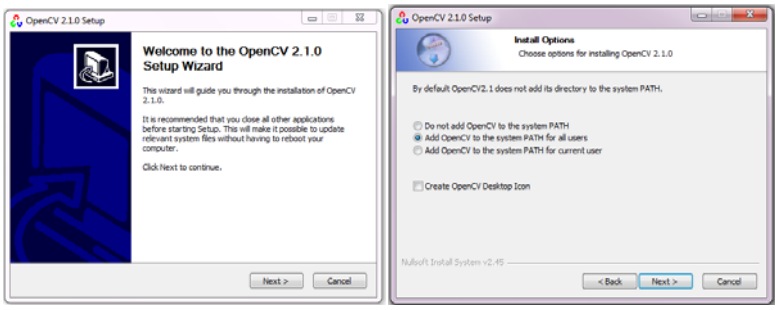
[9]https://code.ros.org/trac/opencv/browser/trunk/opencv/samples/c/demhist.c?rev=1429
http://visca.com/ffactory/archives/5-99/msg00021.html

9. Παράρτημα

Λήψη και Εγκατάσταση της OpenCV

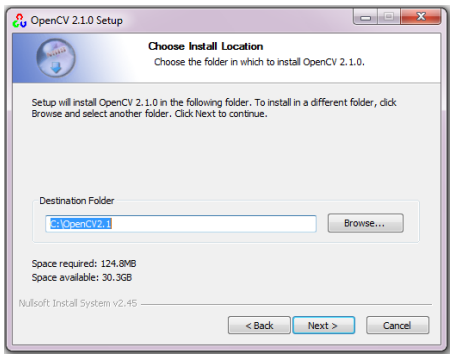
Εγκατάσταση OpenCV για τη γλώσσα C++

Η βιβλιοθήκη της Open Computer Vision (μέχρι στιγμής τελική έκδοση 2.1), που χρησιμοποιείται στη γλώσσα προγραμματισμού C++, μπορεί να κατεβαστεί από την ηλεκτρονική διεύθυνση ‘ http://sourceforge.net/projects/opencvlibrary/files/opencv-win/2.1/ ’ για τα windows ενώ για τα Linux από τη διεύθυνση ‘ http://sourceforge.net/projects/opencvlibrary/files/opencv-unix/2.1/ ’. Αφού γίνει η λήψη του .exe αρχείου, για τη περίπτωση των windows, μπορούμε να κάνουμε την εγκατάσταση της OpenCV.



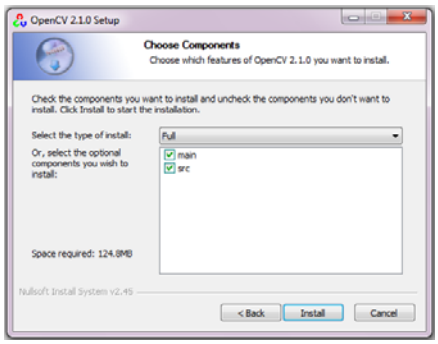
βήματα εγκατάστασης opencv

Στο παράθυρο Install Options επιλέγουμε Add OpenCV to the system PATH for all users και πατάμε Next >.



βήματα εγκατάστασης opencv

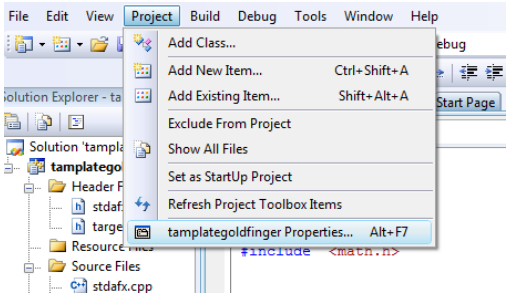
Στο επόμενο παράθυρο καθορίζουμε τον φάκελο εγκατάστασης. Ο προεπιλεγμένος φάκελος είναι ο ‘C:\OpenCV2.1’.



βήματα εγκατάστασης opencv

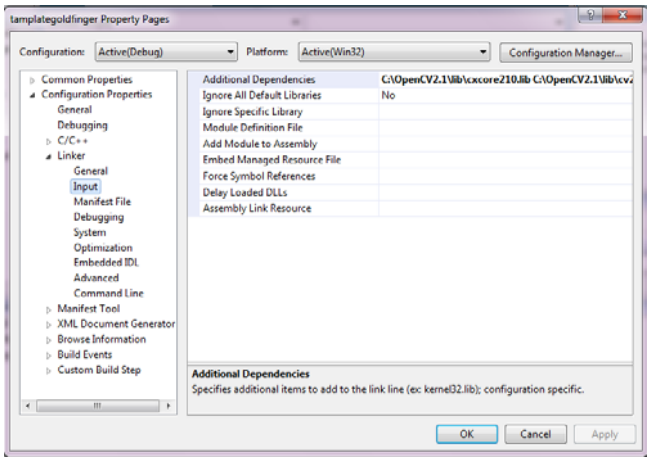
Στο παράθυρο Choose Components επιλέγουμε type of install: Full, και κάνουμε την εγκατάσταση.

Αφού ολοκληρωθεί η εγκατάσταση θα πρέπει να συνδέσουμε τις βιβλιοθήκες της OpenCV που χρειαζόμαστε στο πρόγραμμα μας. Οι βιβλιοθήκες βρίσκονται στο φάκελο ‘ C:\OpenCV2.1\lib\ ’ και για να τις συνδέσουμε με ένα project της C++, αφού ανοίξουμε το project ,θα πρέπει να επιλέξουμε Project -> Properties, ή να πατήσουμε Alt+F7.

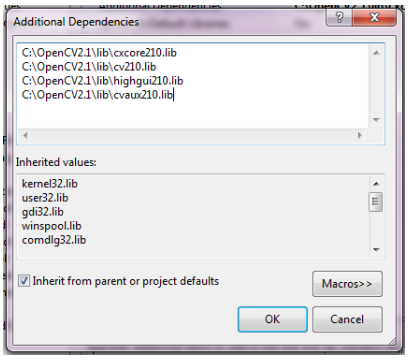


βήματα εγκατάστασης opencv

Στο παράθυρο που θα εμφανιστεί επιλεγούμε τη καρτέλα Input από το ‘Configuration Properties-> Linker’ και στην επιλογή Additional Dependencies να προσθέσουμε της βιβλιοθήκες που χρειαζόμαστε.



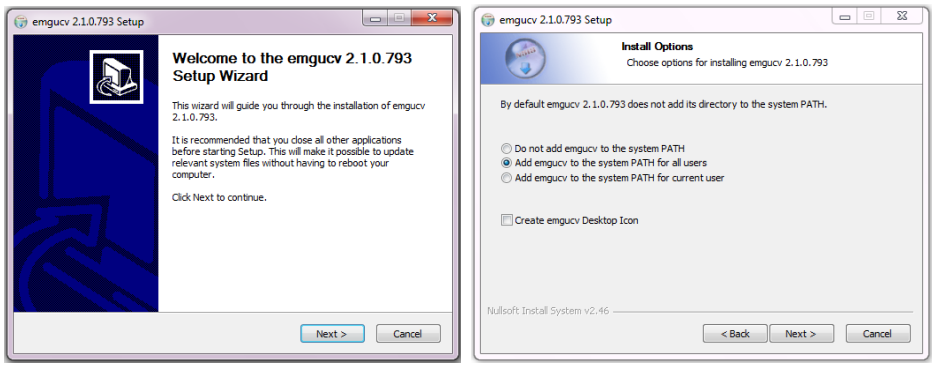
βήματα εγκατάστασης opencv



βήματα εγκατάστασης opencv

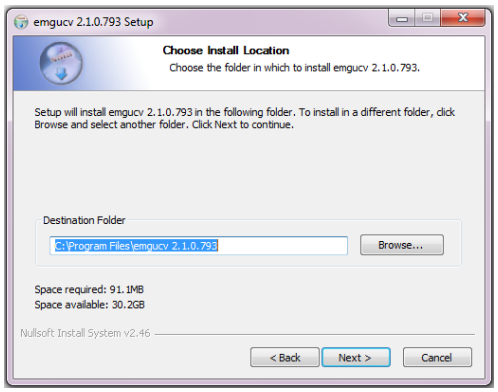
Εγκατάσταση OpenCV για τη γλώσσα C#

Η αντίστοιχη βιβλιοθήκη της OpenCV για τη γλώσσα C# είναι η Emgu CV, η οποία επιτρέπει τη κλήση εντολών της OpenCV από γλώσσες συμβατές με τη .NET όπως C#, VB, VC++ ή η IronPython. Η EmguCV μπορεί να κατεβαστεί από τη διεύθυνση ‘ http://sourceforge.net/projects/emgucv/files/ ’. Αφού γίνει η λήψη του αρχείου setup, μπορούμε να κάνουμε την εγκατάσταση της EmguCV.



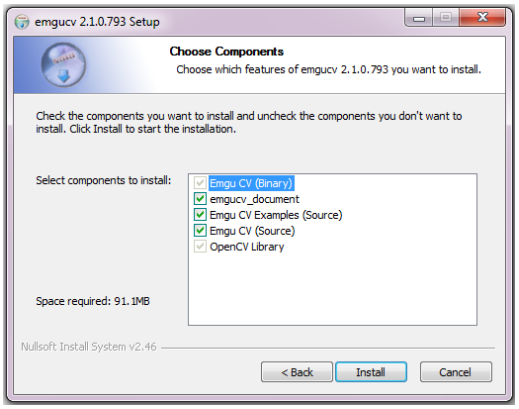
βήματα εγκατάστασης Emgu CV

Στο παράθυρο Install Options επιλέγουμε Add OpenCV to the system PATH for all users και πατάμε Next >.



βήματα εγκατάστασης Emgu CV

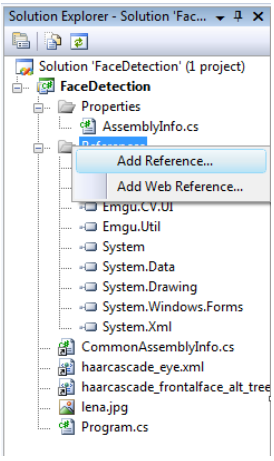
Στο επόμενο παράθυρο καθορίζουμε τον φάκελο εγκατάστασης. Ο προεπιλεγμένος φάκελος είναι ο 'C:\Program Files\emgucv 2.1.0.793'.



βήματα εγκατάστασης Emgu CV

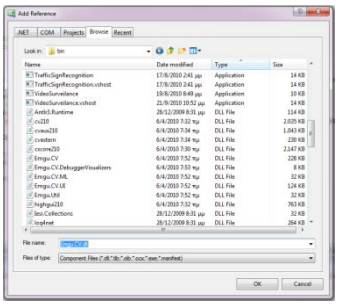
Στο παράθυρο Choose Components επιλέγουμε τα Emgucv_document , Emgu CV Examples και Emgu CV , και κάνουμε την εγκατάσταση.

Για να γίνει η σύνδεση μίας βιβλιοθήκης με ένα project της C# θα πρέπει να τη προσθέσουμε στα References του project. Αυτό μπορεί να γίνει πατώντας δεξί click στα References και επιλέγοντας Add reference .



βήματα εγκατάστασης Emgu CV

Στο παράθυρο που θα εμφανιστεί επιλέγουμε Browse επιλέγουμε τη βιβλιοθήκη που θέλουμε. Οι βιβλιοθήκες της Emgu CV βρίσκονται στο φάκελο 'C:\Program Files\emgucv 2.1.0.793\bin\'.



βήματα εγκατάστασης Emgu CV

OpenCV HaarTraining

Η OpenCV παρέχει έναν τρόπο εκπαίδευσης και δημιουργίας classifier, χρησιμοποιώντας Haar-like χαρακτηριστικά σε HaarTraining. Το αποτέλεσμα της εκπαίδευσης είναι ένα αρχείο xml που περιέχει τους ορισμούς του classifier. Η διαδικασία δημιουργίας ενός classifier χωρίζεται σε τέσσερα βήματα :

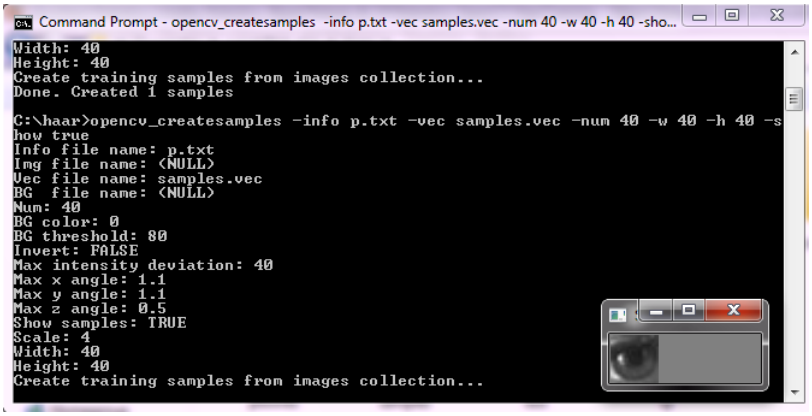
1. Προετοιμασία στοιχείων (απόκτηση εικόνων/βίντεο).
2.Δημιουργία δειγμάτων από τις εικόνες και παραγωγή ενός αρχείου .vec (χρησιμοποιώντας την εφαρμογή createsamples.exe).
3.Εκπαίδευση του classifier.
4.Εκτίμηση απόδοσης του classifier.

1. Η πρώτη φάση της διαδικασίας περιλαμβάνει την απόκτηση των εικόνων ή των βίντεο που περιέχουν το αντικείμενο του classifier. Οι εικόνες αυτές αναφέρονται ως θετικές εικόνες. Θετικές εικόνες που δε θα χρησιμοποιηθούν στην εκπαίδευση του classifier θα χρησιμοποιηθούν για το testing. Περιλαμβάνει επίσης τη συλλογή των "αρνητικών" εικόνων, που δεν περιέχουν το αντικείμενο.

path/neg_img_name_1
path/neg_img_name_2
...
Π.χ. C:\Haar\ Negatives\iris_0.jpg
όπου path η διεύθυνση της εικόνας και neg_img_name το όνομά της.
Οι θετικές εικόνες πρέπει να συνταχθούν σε ένα .txt αρχείο με τη παρακάτω σύνταξη:
<path>/img_name_1 count_1 x11 y11 w11 h11 x12 y12 . . .
<path>/img_name_2 count_2 x21 y21 w21 h21 x22 y22 . . .
. . .

Όπου count ο αριθμός των σημαντικών αντικειμένων που υπάρχουν στην εικόνα x και y το pixel που ξεκινά το αντικείμενο σε χ και y άξονα και w και h το μέγεθος του.
Π.χ. *C:\Haar\ Positives\iris_0.jpg 1 288 167 111 111*
C:\Haar\ Positives\iris_0.jpg 2 288 167 111 111 350 169 110 110

2. Αφότου διαχωρίζονται όλα τα θετικά και τα αρνητικά στους κατάλληλους φακέλους χρησιμοποιείται η εφαρμογή της OpenCV createsamples.exe για τη παραγωγή του αρχείου .vec . Το αρχείο .vec είναι ένα vector θετικών δειγμάτων.



Δημιουργία Samples

Command line arguments:

- vec <vec_file_name> : Όνομα του αρχείου που θα περιέχει τα θετικά δείγματα για εκπαίδευση.

- img <image_file_name>: Εικόνα αντικειμένου (π.χ., ίριδα)

- bg <background_file_name> : Background description file, περιέχει έναν κατάλογο εικόνων στον οποίο τυχαία διαστρεβλωμένες εκδόσεις του αντικειμένου κολλιούνται για τη θετική παραγωγή δειγμάτων

- num <number_of_samples> : Αριθμός θετικών εικόνων

- bgcolor <background_color>: Background color (υποθέτει grayscale εικόνες) το background color δείχνει το διαφανές χρώμα. Αφού μπορούν να υπάρξουν συμπιεσμένα artifacts, το ποσό ανοχής χρώματος μπορεί να διευκρινιστεί από το - bgthresh. Όλα τα pixels μεταξύ bgcolor-bgthresh και bgcolor+bgthresh θεωρούνται διαφανής.

- inv : Αν καθοριστεί, τα χρώματα αντιστρέφονται

- randinv : Αν καθοριστεί, τα χρώματα αντιστρέφονται τυχαία

- maxidev <max_intensity_deviation>: μέγιστη απόκλιση έντασης των δειγμάτων pixel foreground

- maxxangle <max_x_rotation_angle>,
-maxyangle <max_y_rotation_angle>,
- maxzangle <max_z_rotation_angle> : μέγιστες γωνίες περιστροφής σε ακτίνια

-show : Αν καθοριστεί, κάθε sample θα παρουσιαστεί. Πατώντας 'Esc' θα συνεχίσει τη δημιουργία χωρίς παρουσίαση. Χρήσιμο για το debugging.

- w <sample_width> : Μήκος (σε pixels) των εξαγόμενων samples

- h <sample_height> : Ύψος (σε pixels) των εξαγόμενων samples

Η createsamples μπορεί να χρησιμοποιηθεί για να συνθέσει δεδομένα εφαρμόζοντας γεωμετρικούς μετασχηματισμούς, προσθέτοντας θόρυβο, μετατρέποντας χρώματα, κ.α. Η διαδικασία αυτή μπορεί να (π.χ.) μάθει ένα logo, με μία μόνο εικόνα περνώντας από διάφορες μετατροπές.

Πχ. *createsamples -img face.img -num 10 -bg negatives.txt -vec samples.vec -maxxangle 0.6 -maxyangle 0 -maxzangle 0.3 -maxidev 100 -bgcolor 0 -bgthresh 0 -w 20 -h 20*

3. Εκπαίδευση του Classifier using haartraining.exe χρησιμοποιώντας τη εφαρμογή haartraining.exe.

Command line arguments :
- data <dir_name> : Το όνομα του φακέλου όπου ο εκπαιδευμένος classifier θα αποθηκευτεί

- vec <vec_file_name> : Ονομασία των θετικών sample (δημιουργηθήκαν από την εφαρμογή trainingsamples)

- bg <background_file_name> : Αρνητικές εικόνες

- npos <number_of_positive_samples>,
- nneg <number_of_negative_samples> : Αριθμός θετικών/αρνητικών samples που θα χρησιμοποιηθούν σε κάθε στάδιο εκπαίδευσης του classifier.

- nstages <number_of_stages> : Αριθμός σταδίων εκπαίδευσης

- nsplits <number_of_splits> : Καθορίζει τον αδύναμο classifier. Αν 1, simple stump classifier, αν 2 ή παραπάνω, CART classifier με αριθμό n_splits εσωτερικών (split) nodes

- mem <memory_in_MB> : Διαθέσιμη μνήμη MB για επεξεργασία. Όσο περισσότερη μνήμη τόσο γρηγορότερη εκπαίδευση

- sym (default),
- nonsym : Καθορίζει αν το αντικείμενο υπό εκπαίδευση έχει κάθετη συμμετρία ή όχι. Η κάθετη συμμετρία επιταχύνει τη διαδικασία εκπαίδευσης .

- minhitrate <min_hit_rate> : Ελάχιστο επιθυμητό hit rate για κάθε στάδιο του classifier. Το ολικό hit rate μπορεί να υπολογιστεί ως (min_hit_rate^number_of_stages)

- maxfalsealarm <max_false_alarm_rate> : Μέγιστο επιθυμιτό false alarm για κάθε στάδιο του classifier. Το ολικό false alarm rate μπορεί να υπολογιστεί ως (max_false_alarm_rate^number_of_stages)

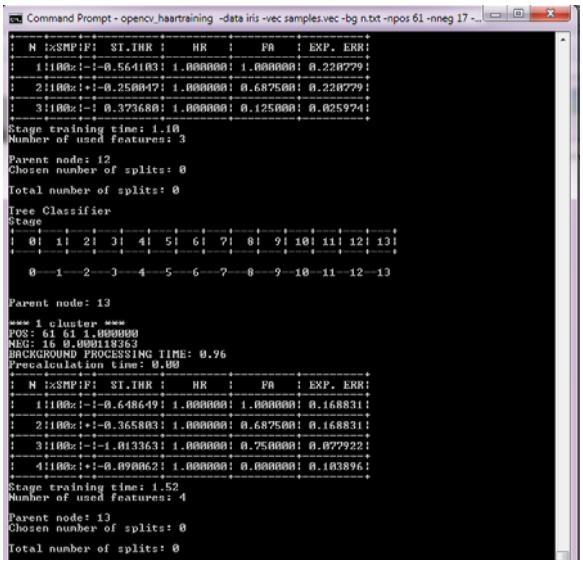
- weighttrimming <weight_trimming> : Καθορίζει αν και πόσο weight trimming πρέπει να χρησιμοποιηθεί.

- eqw

- mode <BASIC (default) | CORE | ALL> : Επιλεγει το τύπο των haar features που χρησιμοποιούνται στην εκπαίδευση. Το BASIC χρησιμοποιεί μόνο upright features, ενώ το ALL χρησιμοποιεί πλήρες set upright και περιστρεμμένα κατά 45 βαθμούς feature.

- w <sample_width>,
- h <sample_height> : Μέγεθος των εκπαιδευμένων samples (in pixels). Must πρέπει να έχουν της ίδιες τιμές που χρησιμοποιήθηκαν για τη δημιουργία samples creation (trainingsamples)

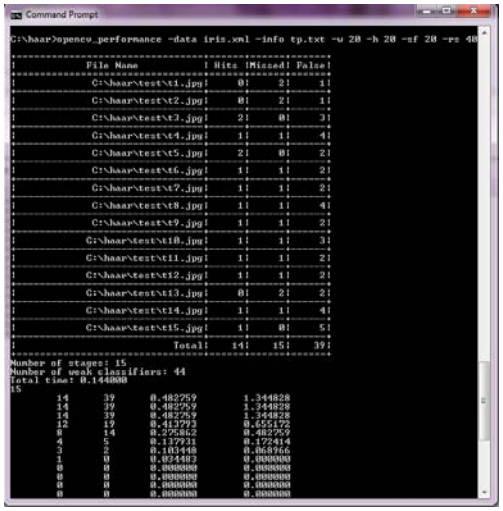
Πχ. *haartraining.exe -data HaarCascade_ιρισ -vec Positives.vec -bg Negatives.txt -nstages 18 -nsplits 2 -minhitrate 0.998 -maxfalsealarm 0.45 -npos 1291 -nneg 3074 -w 20 -h 20 -nonsym -mem 512*



Εκπαίδευση cascade

Το haartraing δημιουργεί ένα xml αρχείο όταν η διαδικασία ολοκληρωθεί .

5. Για την απόδοση ενός classifier μπορεί να χρησιμοποιηθεί η εφαρμογή performance. Παίρνει μια συλλογή χαρακτηρισμένων εικόνων, εφαρμόζει τον ταξινομητή και εξάγει την απόδοση, δηλαδη: αριθμός αντικειμένων, αριθμός ευρεθέντων αντικειμένων, αριθμός εσφαλμένων συναγερμών και άλλες πληροφορίες.



Δοκιμή του Cascade

Command line arguments:
- data <dir_name> : Φάκελος στον οποίο έχει αποθηκευτεί ο εκπαιδευμένος classifier

- info <collection_file_name> : Αρχείο καθορισμού των test samples

- maxSizeDiff <max_size_difference>,
- maxPosDiff <max_position_difference> : Καθορισμός του κριτήριου αναφοράς και ανιχνευμένης σύμπτωσης ορθογωνίων. Οι προκαθορισμένες τιμές είναι 1.5 και 0.3 αντίστοιχα.

- sf <scale_factor> : Παράμετρος εύρεσης. Η προκαθορισμένη τιμή είναι 1.2.

- w <sample_width>,
- h <sample_height> : Μέγεθος των εκπαιδευμένων samples (σε pixels). Θα πρέπει να έχουν τις ίδιες τιμές με αυτές που χρησιμοποιήθηκαν στην εκπαίδευση (haartraining).

Συναρτήσεις

cvSub()

void cvSub(const CvArr* src1, const CvArr* src2, CvArr* dst, const CvArr*mask=NULL);

src1
ο πρώτος πινάκας πηγή
src2
ο δεύτερος πινάκας πηγή
dst
ο πινάκας του αποτελέσματος
mask
μάσκα λειτουργίας, 8-bit πίνακας ενός καναλιού. Προσδιορίζει τα στοιχεία του πίνακα προορισμού τα οποία θα αλλαχτούν.

dst(I)=src(I)-value **if** mask(I)!=0

cvThreshold()

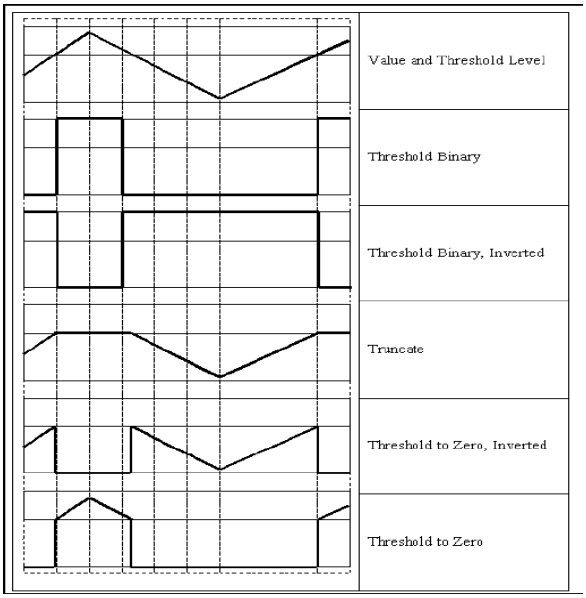
double cvThreshold(const CvArr* src,CvArr* dst,double threshold,double maxValue, int thresholdType);

src
πίνακας πηγή (ενός καναλιού, 8-bit ή 32-bit floating point)
dst
πίνακας αποτελέσματος πρέπει να είναι του ίδιου τύπου μετον src ή 8-bit
threshold
η τιμή του Threshold
maxValue
Η μέγιστη τιμή οταν χρησιμοποιούνται οι CV THRESH BINARY και CV THRESH BINARY INV τύποι Threshold
thresholdType
ο τύπος του Threshold ο οποίος μπορεί να είναι ένας από τους επόμενους:
CV THRESH BINARY
dst(x; y)= maxValue if src(x; y) > threshold,0 otherwise
CV THRESH BINARY INV
dst(x; y) = 0 if src(x; y) > threshold,maxValue otherwise
CV THRESH TRUNC
dst(x; y) = threshold if src(x; y) > threshold,src(x; y) otherwise
CV THRESH TOZERO

dst(x; y) = src(x; y) if src(x; y) > threshold,0 otherwise

CV THRESH TOZERO INV

dst(x; y) = 0 if src(x; y) > threshold,src(x; y) otherwise



3.3.2-1 επιλογές threshold

cvMorphologyEx()

void cvMorphologyEx(const CvArr* src,CvArr* dst,CvArr* temp,IplConvKernel* element,int operation,int iterations=1);

src

Η εικόνα πηγή

dst

Η εικόνα αποτελέσματος

temp

Προσωρινή εικόνα ,είναι απαραίτητη σε κάποιες περιπτώσεις .

Element

δομικό στοιχείο

operation

Τύπος του μορφολογικού μετασχηματισμού ο οποίος μπορεί να είναι ένας από τους επόμενους:

CV MOP OPEN
CV MOP CLOSE
CV MOP GRADIENT
CV MOP TOPHAT
CV MOP BLACKHAT

iterations

Αριθμός που εκφράζει το πόσες φορές θα εφαρμοστούν η διάβρωση και η διαστολή στην εικόνα του αποτελέσματος

cvCreateStructuringElementEx()

```
IplConvKernel* cvCreateStructuringElementEx(int cols,int rows,int anchorX,int
anchorY,int shape,int* values=NULL );
```

cols Ο αριθμός των στύλων στο δομικό στοιχείο
rows Ο αριθμός των γραμμών στο δομικό στοιχείο
anchorX Σχετική οριζόντια μετατόπιση του κέντρου του σημείου ενδιαφέροντος
anchorY Σχετική κάθετη μετατόπιση του κέντρου του σημείου ενδιαφέροντος
shape Το σχήμα του δομικού στοιχείου ,μπορει να πάρει μία από τις παρακάτω τιμές:
CV SHAPE RECT παραλληλόγραμμο στοιχείο
CV SHAPE CROSS στοιχείο σε σχήμα σταυρού
CV SHAPE ELLIPSE στοιχείο σε σχήμα έλλειψης
CV SHAPE CUSTOM στοιχείο με σχήμα καθοριζόμενο από τον χρήστη.

values Δείκτης ως προς το δομικό στοιχείο.

cvHoughCircles()

```
CvSeq* cvHoughCircles( CvArr* image, void* circle_storage,
                int method, double dp, double min_dist,
                double param1=100, double param2=100,
                int min_radius=0, int max_radius=0 );
```

image
Grayscale εικόνα πηγής 8-bit ενός καναλιού.
circle_storage
Η «αποθήκη» για τους κύκλους που θα ανιχνευτούν. Μπορεί να είναι memory storage (σε περίπτωση που οι κύκλοι που θα δημιουργηθούν στο storage και θα επιστρεφτούν από τη συνάρτηση) ή matrix μίας γραμμής και μίας στήλης (CvMat*) τύπου CV_32FC3, στο οποίο θα γραφτούν τα στοιχεία των κύκλων. Η επικεφαλίδα του matrix τροποποιείτε από τη συνάρτηση έτσι ώστε οι γραμμές ή οι στήλες του να περιέχουν τον αριθμό των γραμμών που θα ανιχνευτούν. Αν το circle_storage είναι matrix και ο αριθμός των γραμμών ξεπερνά το μέγεθος του matrix, επιστρέφεται ο μέγιστος δυνατός αριθμός κύκλων. Κάθε κύκλος κωδικοποιείται ως 3 floating-point αριθμοί: κεντρικές συντεταγμένες (x,y) και η ακτίνα.
method
Προς το παρόν, η μόνη ενσωματωμένη μέθοδος είναι η CV_HOUGH_GRADIENT, η οποία είναι 21HT.
dp
Η ανάλυση του συσσωρευτή που χρησιμοποιείται για τη ανίχνευση των κέντρων των κύκλων. Π.χ., εν είναι 1, ο συσσωρευτής θα έχει την ίδια ανάλυση με την εικόνα εισαγωγής, αν είναι 2 – θα έχει υποδιπλασιασμένο μήκος και πλάτος.
min_dist

Η ελάχιστη απόσταση μεταξύ των ανιχνευμένων κύκλων. Αν η παράμετρος είναι πολύ μικρή, μπορεί να ανιχνευτούν εσφαλμένα πολλοί γειτονικοί κύκλοι επιπρόσθετα με τον πραγματικό. Αν είναι πολύ μεγάλο, κάποιοι πραγματικοί κύκλοι μπορεί να χαθούν.
param1

Η πρώτη παράμετρος της μεθόδου. Στη περίπτωση του CV_HOUGH_GRADIENT είναι το μεγαλύτερο threshold του Canny edge detector (το μικρότερο threshold θα είναι δύο φορές μικρότερο).
param2

Η δεύτερη παράμετρος της μεθόδου. Στη περίπτωση του CV_HOUGH_GRADIENT είναι ένας συσσωρευτής threshold στο κέντρο της διαδικασίας ανίχνευσης. Όσο πιο μικρό είναι, τόσο περισσότεροι εσφαλμένοι κύκλοι θα ανιχνευτούν. Οι κύκλοι που ανταποκρίνονται στις μεγαλύτερες τιμές του συσσωρευτή ,επιστρέφονται πρώτοι.
min_radius

Η ελάχιστη ακτίνα των κύκλων.
max_radius

Η μέγιστη ακτίνα των κύκλων. Το default για τη μέγιστη ακτίνα είναι max(image_width, image_height).