Αλεξάνδρειο Τεχνολογικό
Εκπαιδευτικό Ίδρυμα
Θεσσαλονίκης

Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Πληροφορικής

Πτυχιακή Εργασία

Ψηφιακή Αφαίρεση Υποβάθρου σε Πραγματικό Χρόνο

Παναγιώτης Κουτσουράς

Επιβλέπων καθηγητής: Ιωάννης Παλιόκας

Περιεχόμενα

Ευρετήριο Εικόνων

Ευρετήριο Πινάκων

Περίληψη

Η Ρομποτική Όραση ή Μηχανική Όραση (Computer Vision) γνωρίζει ιδιαίτερη άνθηση και χρησιμοποιείται ευρύτατα σε ποικίλες εφαρμογές. Η OpenCV, μία βιβλιοθήκη ανοιχτού κώδικα με προσανατολισμό στις εφαρμογές Computer Vision, προσφέρει ένα μεγάλο αριθμό έτοιμων δομών και συναρτήσεων για την υποβοήθηση επίδοξων προγραμματιστών και ομάδων που θέλουν να ασχοληθούν με το αντικείμενο αυτό. Η παρούσα εργασία αποτελεί μία προσπάθεια μελέτης και παρουσίασης βασικών μεθοδολογιών μηχανικής αφαίρεσης υποβάθρου σε ροές βίντεο σε πραγματικό χρόνο. Ως περίπτωση χρήσης σχεδιάζεται και υλοποιείται ένα λογισμικό αντικατάστασης υποβάθρου με βάση τεχνικές που χρησιμοποιούνται ευρέως στη βιομηχανία του κινηματογράφου, έχοντας ως είσοδο την εικόνα μίας web camera και ένα αρχείο εικόνας ή video, που αποτελεί το νέο υπόβαθρο, και ως έξοδο την εικόνα της web camera με το αντικατεστημένο υπόβαθρο. Ταυτόχρονα, τέθηκε ως απαραίτητη προϋπόθεση για την ολοκλήρωση της εφαρμογής, η επίτευξη της προαναφερθείσας αντικατάστασης σε πραγματικό χρόνο (real time).

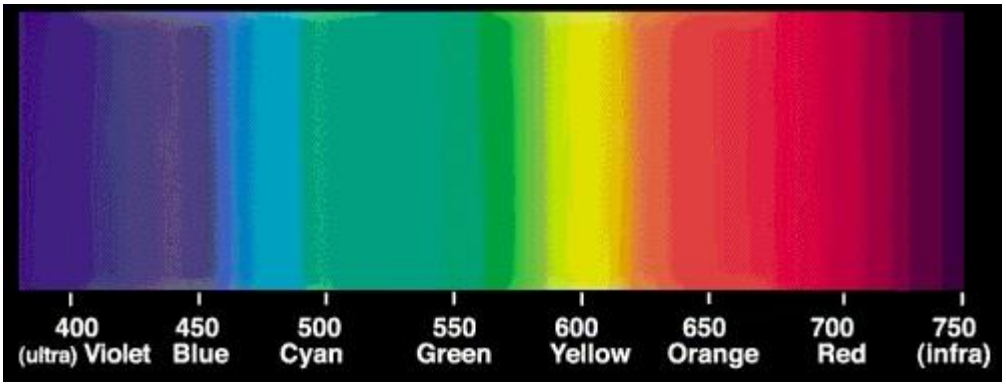
1 Ψηφιακή Εικόνα και Computer Vision

1.1 Ψηφιακή Εικόνα

Απαραίτητο στοιχείο για οποιαδήποτε εφαρμογή της Computer Vision είναι η ψηφιακή εικόνα. Η δημιουργία μίας ψηφιακής εικόνας μπορεί να υλοποιηθεί με δύο τρόπους: με την ψηφιοποίηση μίας ήδη υπάρχουσας, αναλογικής εικόνας με τη χρήση κατάλληλου εξοπλισμού, ή με την εκ του μηδενός δημιουργίας της με τη χρήση κατάλληλου λογισμικού (Δημητριάδης και άλλοι, 2004). Παρόλο που η δομή μίας ψηφιακής εικόνας είναι συγκεκριμένη και σχετική με την ψηφιακή τεχνολογία, δεν παύει να στηρίζεται στη θεωρία του χρώματος και στο σύνολο των χρωματικών μοντέλων που υπάρχουν.

1.1.1 Θεωρία χρώματος

Το χρώμα που αντιλαμβάνεται ο άνθρωπος είναι ένα μέρος του ηλεκτρομαγνητικού φάσματος, που περιλαμβάνει ακτινοβολίες με μήκη κύματος από χιλιοστά του δισεκατομμυριοστού του μέτρου, μέχρι χιλιόμετρα. Το κομμάτι του ηλεκτρομαγνητικού φάσματος που είναι ορατό στο ανθρώπινο μάτι βρίσκεται μεταξύ των 400 με 700 nm περίπου και αποτελεί την ορατή περιοχή του φάσματος, στη μία άκρη του οποίου (κοντά στα 400nm) βρίσκεται η ιώδης ακτινοβολία, ενώ στην άλλη άκρη (κοντά στα 700nm) η ερυθρή.



Εικόνα 1:1 Ορατό φάσμα ηλεκτρομαγνητικής ακτινοβολίας

Το λευκό ηλιακό φως προκύπτει από τη σύνθεση όλων των παραπάνω χρωμάτων. Το τι χρώμα θα έχει ένα αντικείμενο εξαρτάται από τον τρόπο που αλληλεπιδρά με το ηλιακό φως, δηλαδή από τις ακτινοβολίες που απορροφώνται και ανακλούνται. Η εκάστοτε χρωστική ουσία που καλύπτει το αντικείμενο απορροφά ορισμένα μήκη κύματος του ηλιακού φωτός και ανακλά τα υπόλοιπα, επομένως το φως που αντιλαμβάνεται ο άνθρωπος είναι αυτό που ανακλάται από τις χρωστικές ουσίες. Το λευκό, για παράδειγμα, το βλέπουμε όταν ανακλώνται όλα τα μήκη κύματος, ενώ το μαύρο στην ακριβώς αντίθετη περίπτωση, δηλαδή όταν όλα τα μήκη κύματος απορροφώνται. Στη θεωρία αυτή βασίζονται και τα χρωματικά μοντέλα.

1.1.2 Χρωματικά μοντέλα

Η ανάγκη για αυστηρή περιγραφή του χρώματος οδήγησε στη δημιουργία ποικίλων χρωματικών μοντέλων. Σκοπός των μοντέλων αυτών είναι η ακριβής, μαθηματική, περιγραφή των χρωμάτων, ώστε να υπάρχει μία άμεση αναφορά σε αυτά για την κωδικοποίηση και χρήση τους. Τα χρωματικά μοντέλα συνήθως αποτελούνται από τρεις ή τέσσερις συνιστώσες και όταν συνδεθούν με έναν τρόπο παρουσίασης, δημιουργούνται οι αντίστοιχοι χρωματικοί χώροι που καθορίζουν το σύνολο των χρωμάτων που μπορεί να αποδοθεί σε κάθε μέσο παρουσίασης (πχ οθόνες υπολογιστών, διαφάνειες, εκτυπώσεις κτλ) (Δημητριάδης και άλλοι, 2004).

Υπάρχουν πολλά διαφορετικά μοντέλα που το καθένα χρησιμοποιεί διαφορετικές παραμέτρους για την παρουσίαση του συνόλου των χρωμάτων. Τα κυριότερα των μοντέλων αυτών είναι τα:

- RGB (Red, Green, Blue) – προσθετικό μοντέλο
- HSL (Hue, Saturation, Lightness)
- HSV (Hue, Saturation, Brightness)
- CMY (Cyan, Magenta, Yellow) – αφαιρετικό μοντέλο

Μία παραλλαγή του τελευταίου αποτελεί το CMYK (Cyan, Magenta, Yellow, blacK) που χρησιμοποιείται για την καλύτερη δυνατή εκτύπωση του μαύρου σε ψηφιακές εκτυπώσεις.

Δύο βασικές έννοιες που συναντώνται στην περιγραφή των χρωματικών μοντέλων είναι το «αυξητικό» μοντέλο και το «αφαιρετικό». Οι έννοιες αυτές σχετίζονται με τον τρόπο που δημιουργείται η οπτική αίσθηση στον άνθρωπο. Στην περίπτωση του προσθετικού μοντέλου, οι ακτινοβολίες που δημιουργούν την αίσθηση αυτή προσπίπτουν άμεσα στο μάτι, ενώ στο αφαιρετικό μοντέλο υπάρχει πρώτα ανάκλαση των ακτινοβολιών σε κάποια επιφάνεια και ύστερα πρόσπτωση των ανακλώμενων ακτινοβολιών στο μάτι. Ως συνέπεια, ανάλογα με το μέσο που χρησιμοποιείται για την παρουσίαση μίας εικόνας (πχ οθόνη υπολογιστή ή εκτυπωμένη φωτογραφία), χρησιμοποιείται και το αντίστοιχο χρωματικό μοντέλο.

Το προσθετικό μοντέλο (RGB)

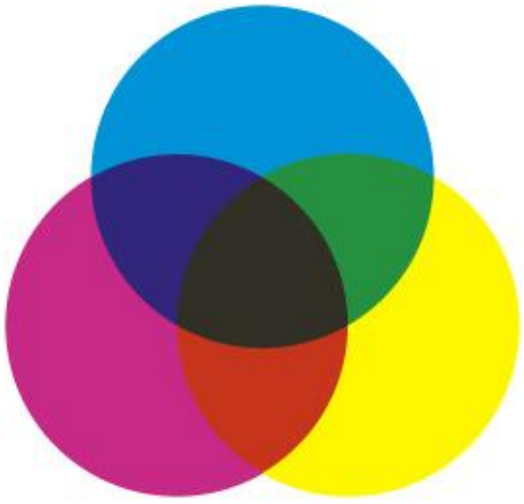
Το προσθετικό χρωματικό μοντέλο χρησιμοποιείται όταν η ακτινοβολία προσλαμβάνεται άμεσα από το μάτι μας (όπως συμβαίνει με την τηλεόραση). Το σύνολο των χρωμάτων προκύπτει ύστερα από συνδυασμό των βασικών τριών χρωμάτων που ορίζονται από το μοντέλο (RGB – Κόκκινο, Πράσινο, Μπλε).

Εικόνα 1:2 Πρωτεύοντα χρώματα στο Προσθετικό Μοντέλο

Ανάλογα με την ένταση του εκάστοτε χρώματος έχουμε και την αντίστοιχη απόχρωση. Στην περίπτωση που και τα τρία χρώματα έχουν τη μέγιστη έντασή τους, σαν αποτέλεσμα έχουμε το λευκό χρώμα. Στην αντίθετη περίπτωση έχουμε μαύρο.

Το αφαιρετικό μοντέλο (CMY)

Το αφαιρετικό μοντέλο έχει ως πρωτεύοντα χρώματα τα Cyan, Magenta και Yellow και χρησιμοποιείται σε περιπτώσεις που η ακτινοβολία φτάνει στο μάτι αφού ανακλαστεί σε μία επιφάνεια (όπως σε μία εκτυπωμένη φωτογραφία). Το σύνολο των χρωμάτων προκύπτει ανάλογα με την επιφάνεια όπου η ακτινοβολία προσπίπτει και ανακλάται. Μία μαύρη επιφάνεια απορροφά πλήρως τις ακτινοβολίες του λευκού φωτός, οπότε και δημιουργείται η αίσθηση του μαύρου. Αντίστοιχα, μία λευκή επιφάνεια ανακλά όλες τις ακτινοβολίες, οπότε και το μάτι αντιλαμβάνεται το άσπρο.



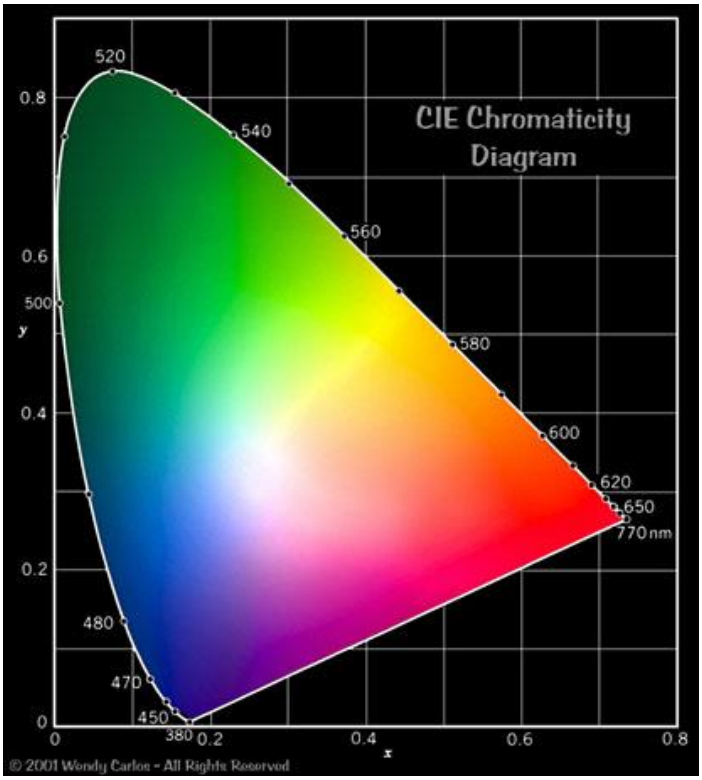
Εικόνα 1:3 Πρωτεύοντα χρώματα στο Αφαιρετικό Μοντέλο

HSL και HSV

Τα δύο αυτά μοντέλα αποτελούν μία διαφορετική αναπαράσταση του RGB και χρησιμοποιούνται κυρίως σε εφαρμογές γραφικών και προγραμμάτων επεξεργασίας εικόνων.

1.1.3 Χρωματικό Διάγραμμα CIE

Το 1931 η International Commission of Illumination (CIE – Commission Internationale de l' Eclairage) δημιούργησε το διάγραμμα CIE XYZ, το οποίο αποτέλεσε μία παρουσίαση της τριχρωματικής σύνθεσης των χρωμάτων. Σκοπός της δημιουργίας του διαγράμματος ήταν να υπάρχει μία κοινή αναφορά όταν ένα σύνολο ανθρώπων παρατηρούν το ίδιο χρώμα, δηλαδή το κάθε χρώμα να προσδιορίζεται από συγκεκριμένες αριθμητικές τιμές. Η CIE βασίστηκε στην έννοια του τυπικού παρατηρητή («standard observer») για τη σύνθεση του διαγράμματος, δηλαδή στο σύνολο των χρωμάτων που μπορεί να δει ο μέσος παρατηρητής. Για τον καθορισμό της έννοιας αυτής, σε ένα μεγάλο αριθμό παρατηρητών παρουσιάστηκαν πολλοί συνδυασμοί των τριών βασικών χρωμάτων κατά τη CIE (435.8 nm για το μπλε, 546.1 nm για το πράσινο, 700 nm για το κόκκινο) και με βάση τη χρωματική αντίληψη τους, δηλαδή ποια χρώματα θεωρούσαν ότι έβλεπαν, συντέθηκε το διάγραμμα CIE XYZ.



Εικόνα 1:4 Χρωματικό διάγραμμα CIE XYZ

1.1.4 Δομή της Ψηφιακής Εικόνας

Η καλύτερη περιγραφής μίας ψηφιακής εικόνας είναι η ταύτισή της με ένα δυσδιάστατο πίνακα, το κάθε στοιχείο του οποίου ονομάζεται «Εικονοστοιχείο» (pixel – Picture Element) και περιέχει την απαραίτητη πληροφορία φωτεινότητας και χρώματος. Το σύνολο των εικονοστοιχείων δημιουργεί στο μάτι του παρατηρητή της εικόνα. Όταν η εικόνα που προβάλλεται είναι ασπρόμαυρη, το κάθε εικονοστοιχείο παίρνει μία τιμή από το 0 (που αντιστοιχεί στο μαύρο) μέχρι το 255 (που αντιστοιχεί στο άσπρο). Οι τιμές αυτές εκφράζουν τη φωτεινότητα του εκάστοτε εικονοστοιχείου και δημιουργούν το σύνολο των πιθανών αποχρώσεων του γκρι. Σε μία έγχρωμη εικόνα, το κάθε εικονοστοιχείο λαμβάνει τιμές (από 0 μέχρι 255 επίσης) για τη φωτεινότητα των τριών βασικών χρωμάτων του προσθετικού μοντέλου (RGB) και έτσι δημιουργείται η πληθώρα των χρωμάτων που προβάλλονται και που μπορεί να δει ο άνθρωπος.

Δύο βασικά στοιχεία που χαρακτηρίζουν την εικόνα είναι η ανάλυσή της και το βάθος χρώματος που χρησιμοποιεί.

Ανάλυση εικόνας

Η ανάλυση της εικόνας έχει να κάνει με τον αριθμό των εικονοστοιχείων στη μονάδα μήκους και η μονάδα μέτρησης είναι το ppi (pixels per inch).

Βάθος χρώματος

Το βάθος χρώματος αναφέρεται στον αριθμό που δηλώνει πόσα bit χρησιμοποιούνται για την αποθήκευση της πληροφορίας του χρώματος που προβάλλεται στο εκάστοτε εικονοστοιχείο. Είναι προφανές ότι όσο περισσότερα bits χρησιμοποιούνται, τόσο μεγαλύτερη είναι η γκάμα χρωμάτων που μπορούν να παρουσιαστούν.

Στην απλούστερη περίπτωση, το βάθος χρώματος είναι μόλις 1 bit, κάτι που σημαίνει ότι το εικονοστοιχείο μπορεί να πάρει δύο τιμές, είτε άσπρο, είτε μαύρο. Τα 8 bit βάθος χρώματος προσφέρουν ένα εύρος 256 χρωμάτων (τιμές από 0 ως 255) και συνήθως χρησιμοποιούνται στην αναπαράσταση ασπρόμαυρων φωτογραφιών, μια που το εύρος αυτό καλύπτει το σύνολο των αποχρώσεων του γκρι. Αντίστοιχα, 16 bit βάθος χρώματος ισοδυναμούν με 65536 χρώματα, ενώ τα 24 bit σε 16.777.216. το νούμερο αυτό είναι υπεραρκετό για την κάλυψη όλων των πιθανών αναγκών που μπορεί να έχει ο άνθρωπος, μια που οι αποχρώσεις που μπορεί να ξεχωρίσει το ανθρώπινο μάτι περιορίζονται στις 300 με 350 χιλιάδες. Το βάθος χρώματος 24 bit χρησιμοποιείται κυρίως στο χρωματικό μοντέλο RGB, όπου 8 bit αντιστοιχούν στο κάθε ένα από τα τρία χρώματα-συνιστώσες του μοντέλου. Τέλος, παρατηρείται η χρήση περισσότερων bit βάθος χρώματος (πχ 32, 48, 96) κυρίως σε επαγγελματική επεξεργασία εικόνας.

1.2 Computer vision

Αποτελώντας ένα ιδιαίτερα σύνθετο πεδίο της επιστήμης της πληροφορικής, η Computer Vision δεν μπορεί να οριστεί επακριβώς, καθώς αποτελεί συνιστώσα μελετών και συνεργασιών

πολλών κλάδων τόσο της Πληροφορικής, όσο και άλλων επιστημών. Ως εκ τούτου, η αναφορά στους στόχους που τίθενται και στα μέσα που χρησιμοποιούνται για την επίτευξη αυτών, είναι ένα πρώτο βήμα για τον καθορισμό του τι είναι Computer Vision.

Ο άνθρωπος γνωρίζει τι είναι αυτά που βλέπει, για αυτό και μπορεί να καθορίσει τη θέση του στο χώρο και να πράξει ανάλογα με το τι υπάρχει γύρω του. Έχοντας αυτό σα βάση, γίνεται η υπόθεση ότι με ένα παρόμοιο τρόπο θα μπορούσε να «σκέφτεται» και ο υπολογιστής, όταν του δοθεί μία εικόνα. Το πρόβλημα εμφανίζεται στο γεγονός ότι ο υπολογιστής δεν μπορεί να αναγνωρίσει σχήματα και αντικείμενα, πρότυπα σε ένα γενικότερο πλαίσιο, μέσα σε μία εικόνα, παρά μόνο μία συστοιχία αριθμών που αντιστοιχούν στο εκάστοτε pixel της εικόνας και καθορίζουν το χρώμα ή/ και τη φωτεινότητα αυτού. Ως αποτέλεσμα, ο υπολογιστής είναι ανίκανος να εξάγει το οποιοδήποτε συμπέρασμα, ακριβώς γιατί μία έγχρωμη εικόνα για αυτόν δεν είναι τίποτα παραπάνω από ένα δυσδιάστατο πίνακα με αριθμούς. Αυτό είναι και το βασικότερο πρόβλημα από το οποίο ξεκινάει και προσπαθεί να «λύσει» η Computer Vision: επιδιώκεται η μεταμόρφωση μίας εικόνας ή ενός βίντεο σε μία απόφαση ή σε μία νέα αναπαράσταση της ήδη υπάρχουσας πληροφορίας, ανάλογα πάντα με την εφαρμογή και τους στόχους της. Η χρήση της Computer Vision για εξαγωγή συμπερασμάτων και αποφάσεων θα μπορούσε να αναφέρεται σε ένα ρομπότ που πρέπει να προχωράει μέσα σε έναν τρισδιάστατο χώρο, αποφεύγοντας εμπόδια. Αντίστοιχα, μία πιθανή νέα αναπαράσταση της αρχικής εικόνας είναι η μετατροπή της σε ασπρόμαυρη.

Τα παραπάνω προβλήματα περιπλέκονται ακόμα περισσότερο από το γεγονός ότι ο πραγματικός κόσμος είναι τρισδιάστατος, ενώ η εικόνα που λαμβάνει και επεξεργάζεται ένας υπολογιστής είναι δισδιάστατη. Ο υπολογιστής καλείται λοιπόν να εξάγει συμπεράσματα από αυτήν χωρίς να μπορεί να γνωρίζει τι απεικονίζεται μέσα της, αλλά και χωρίς να έχει την αίσθηση του βάθους και της γεωμετρίας που υφίσταται στον πραγματικό κόσμο. Με βάση λοιπόν τα προβλήματα αυτά, η Computer Vision θα μπορούσε να οριστεί ως:

'Ένα σύνολο υπολογιστικών τεχνικών που στοχεύουν στην πρόβλεψη ή συγκεκριμενοποίηση των γεωμετρικών και δυναμικών ιδιοτήτων του τρισδιάστατου κόσμου μέσω μίας ψηφιακής εικόνας' (Trucco et al., 1998).

Στο σημείο αυτό γίνεται κατανοητό ότι η Computer Vision δεν καθορίζεται μόνο από το στόχο της, αλλά και από τον τρόπο που αυτός θα μπορέσει να ικανοποιηθεί. Για την ολοκλήρωση αυτού του στόχου είναι απαραίτητη η χρήση εργαλείων, τα οποία χωρίζονται σε δύο μεγάλες κατηγορίες: τους αλγορίθμους και το υλικό. Οι αλγόριθμοι είναι απαραίτητοι για την επεξεργασία της εικόνας και την αναγνώριση προτύπων μέσα σε αυτή, με βάση τα οποία θα παρθούν και οι ανάλογες αποφάσεις. Το υλικό αναφέρεται στις συσκευές που θα χρησιμοποιηθούν για τη λήψη και αποθήκευση των εικόνων (ή του video), στις μονάδες επεξεργασίας αυτών και φυσικά στο χώρο που θα λειτουργεί η εφαρμογή. Παρόλο που οι αλγόριθμοι είναι ο πυρήνας όλης της επεξεργασίας της πληροφορίας και η πηγή των συμπερασμάτων, το υλικό και ο χώρος είναι εξίσου σημαντικά στοιχεία για την ορθή λειτουργία των αλγορίθμων και για τα βέλτιστα αποτελέσματα. Η πληροφορία που φέρει μία εικόνα εξαρτάται από πάρα πολλούς παράγοντες, όπως από την τεχνολογία της κάμερας που τραβάει τις

εικόνες ή το video, τον υπάρχοντα φωτισμό, την ποιότητα των φακών, τη γεωμετρία του χώρου, την συμπίεση των εικόνων σε μικρότερα μεγέθη (που οδηγεί και σε χάσιμο πληροφορίας). Πάνω σε αυτήν την πληροφορία θα δράσει ο εκάστοτε αλγόριθμος και θα βγάλει τα απαιτούμενα συμπεράσματα, οπότε είναι πολύ βασικό να υπάρχουν όλες οι απαραίτητες συνθήκες για να έχουμε την καλύτερη δυνατή ποιότητα και ποσότητα πληροφορίας, ανάλογα πάντα με την εφαρμογή που εξετάζεται. Σε πολλές εφαρμογές, ο χώρος και ο φωτισμός είναι στοιχεία πλήρως καθορισμένα και δεν υπάρχει καμία διαφοροποίηση κατά τη διάρκεια εκτέλεσής της, ενώ ο αλγόριθμος, βάσει του οποίου εκτελείται η εφαρμογή, είναι προσαρμοσμένος στα δεδομένα της διαρρύθμισης του χώρου.

2 Chroma Key και τεχνικές αφαίρεσης υποβάθρου

Η αφαίρεση υποβάθρου είναι μία διαδικασία απαραίτητη για την επίτευξη πολλών εφέ που σχετίζονται με την τοποθέτηση ηθοποιών και αντικειμένων σε άλλο περιβάλλον, πέραν του φυσικού. Αναπτύχθηκαν διάφορες τεχνικές με την απλούστερη να είναι η Chroma Key, η οποία βασίζεται στη χρήση μονοχρωματικού φόντου που αντικαθίσταται με μία νέα εικόνα ή βίντεο. Συνήθεις εφαρμογές των διαφόρων τεχνικών αφαίρεσης υποβάθρου συναντώνται στον κινηματογράφο και στα δελτία καιρού της τηλεόρασης, όπου ηθοποιοί και παρουσιαστές τοποθετούνται μπροστά από ένα ομοιόμορφο, μονοχρωματικό, φόντο το οποίο αντικαθίσταται αντίστοιχα με ετερογενείς σκηνές ή μετεωρολογικούς χάρτες.

Στο κεφάλαιο αυτό δίνεται μία σύντομη ιστορική αναδρομή στην τεχνική αυτή και αναλύεται ο τρόπος υλοποίησής της. Ταυτόχρονα, αναφέρονται τροποποιήσεις και ολοκληρώσεις της, με στόχο το ακριβέστερο και αποδοτικότερο αποτέλεσμα

2.1 Chroma Key: τεχνική και ιστορική αναδρομή

Το Chroma Key έκανε την εμφάνισή του τη δεκαετία του 1930, πατώντας σε ήδη υπάρχουσες τεχνολογίες αφαίρεσης υποβάθρου και αντικαθιστώντας τες. Η βασική ιδέα πίσω από την υλοποίησή της είναι η τοποθέτηση ενός ηθοποιού, ή ενός αντικειμένου γενικότερα, μπροστά από ένα ομοιόμορφο φόντο (συνήθως ένα μπλε ή πράσινο πανί) και η εκ νέου αφαίρεση αυτού και αντικατάστασή του με ένα άλλο φόντο. Η λογική του βασίστηκε στο travelling matte, μία τεχνική αφαίρεσης υποβάθρου που χρησιμοποιούνταν μέχρι την εμφάνιση της έγχρωμης φωτογραφίας, οπότε η αναφορά στο πώς αυτή υλοποιείται κρίνεται απαραίτητη για την ορθότερη κατανόηση του Chroma Key.

2.1.1 Travelling Matte

Αυτό που ανέκαθεν ενδιέφερε τους κινηματογραφιστές ήταν η δυνατότητα συνδυασμού εικόνων βίντεο που τραβήχτηκαν σε διαφορετικό χρόνο ή σε διαφορετικά μέρη. Το αποτέλεσμα θα ήταν η αποτύπωση μίας κατάστασης που, υπό άλλες συνθήκες, θα ήταν αδύνατη, όπως για παράδειγμα η προσθήκη ενός ιπτάμενου αντικειμένου σε μία σκηνή ενός λιβαδιού. Σε πολύ πρώιμο στάδιο, για την επίτευξη αυτού του οπτικού «εφέ» κομμάτια του καρέ του film καλύπτοντας, ώστε να προστεθεί στο σημείο που έμενε κενό το στοιχείο που επιθυμούσε ο κινηματογραφιστής. Η τεχνική αυτή έδινε την ικανότητα συνδυασμού δύο ετερογενών εικόνων, με τον περιορισμό όμως ότι το αντικείμενο που προσθέτονταν έπρεπε να παραμένει ακίνητο ή μέσα σε ένα πολύ συγκεκριμένο πλαίσιο, χωρίς να υπάρχει η δυνατότητα αυξομείωσης του κομματιού που καλύπτονταν από το αρχικό καρέ, ούτε η δυνατότητα αλληλεπίδρασης του «ξένου» αντικειμένου με το αρχικό πλάνο (Rickitt Richard, 2007).

Ο παραπάνω περιορισμός αποδείχτηκε βασικότατος με το πέρασμα των χρόνων, δεδομένου ότι υπήρξε, πολύ άμεσα μάλιστα, η ανάγκη τοποθέτησης ενός κινούμενου μοντέλου (κυρίως ηθοποιού) μπροστά από μία σκηνή που τραβήχτηκε σε άλλο τόπο και χρόνο. Η τεχνική που χρησιμοποιήθηκε τα πρώτα χρόνια του κινηματογράφου έδινε τη δυνατότητα συνδυασμού δύο σκηνών, με την προϋπόθεση όμως ότι η δεύτερη, αυτή που θα «φιλοξενούνταν» σε ένα άλλο καρέ, θα ήταν περιορισμένη σε ένα συγκεκριμένο και προκαθορισμένο πλαίσιο, αδυνατώντας να «βγει» έξω από αυτό. Η βασική ιδέα για την κάλυψη της νέας ανάγκης ήταν σχετικά απλή: το κάθε ένα από τα δύο γεγονότα που θέλουμε (ένα κινούμενο μοντέλο και ένα υπόβαθρο) θα φωτογραφίζονταν σε διαφορετικό χρόνο και τόπο, ανάλογα με τις ανάγκες, και μετά τα δύο διαφορετικά film θα ενώνονταν σε ένα. Ένας ηθοποιός, για παράδειγμα, θα φωτογραφίζονταν μπροστά από ένα μαύρο πανί και το φόντο, στο οποίο έπρεπε να «μπει» και να δράσει αυτός, θα φωτογραφίζονταν ξεχωριστά. Ύστερα, αυτές οι δύο εικόνες θα ενώνονταν για την παραγωγή του επιθυμητού αποτελέσματος. Η φύση όμως του κινηματογραφικού film (celluloid film) δεν επέτρεπε κάτι τέτοιο να έχει το βέλτιστο αποτέλεσμα, γιατί ο ηθοποιός, ή το οποιοδήποτε εν κινήσει μοντέλο, θα φαινόταν διαφανές, μέσω του οποίου θα εμφανίζονταν και το φόντο (εικόνα 2:1) (Rickitt Richard, 2007).

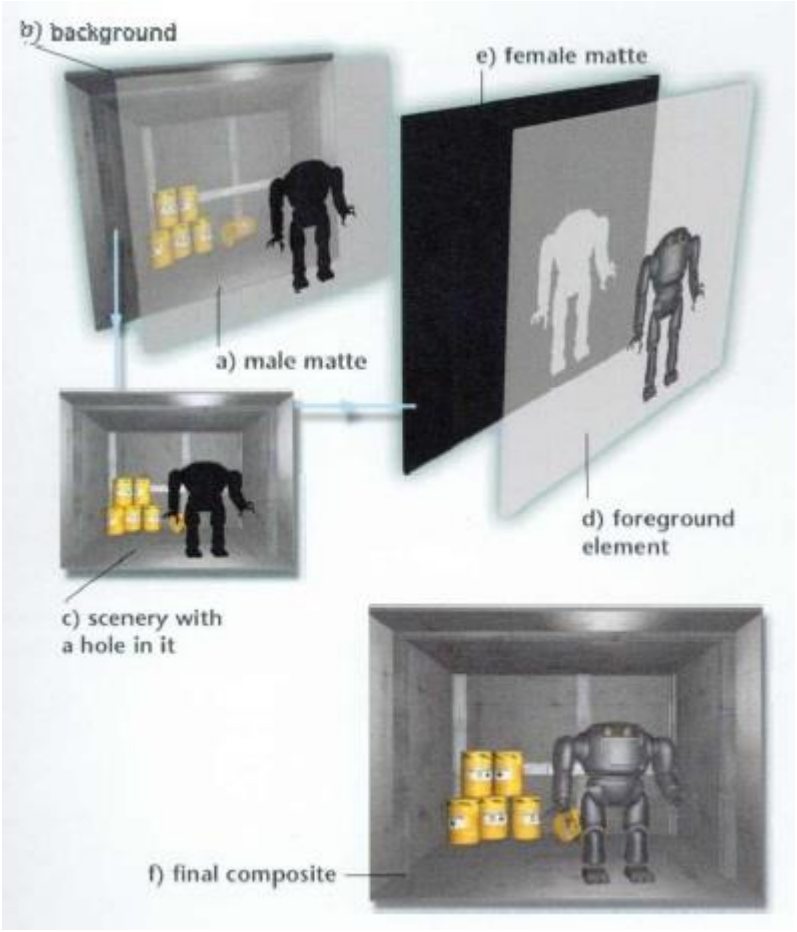
Εικόνα 2:1 Διαφάνεια προσκηνίου (Rickitt Richard, 2007)

Ένας ιδανικός τρόπος για την παραγωγή του απαιτούμενου αποτελέσματος είναι η προσθήκη μίας κινούμενης «οπής» - δηλαδή μίας σκιάς που ανταποκρίνεται πλήρως στο μέγεθος και στις κινήσεις του μοντέλου – πάνω στο φόντο. Η οπή αυτή θα καλυφθεί αργότερα από την εικόνα του μοντέλου και έτσι θα επιτευχθεί η τελική εικόνα, όπου το μοντέλο θα βρίσκεται στο επιλεγμένο φόντο. Η ολοκλήρωση αυτής της διαδικασίας απαιτεί πολλά στάδια και το συνδυασμό πολλών ενδιάμεσων στοιχείων που δημιουργούνται στα στάδια αυτά (Rickitt Richard, 2007).

Το πρώτο στοιχείο, που ονομάζεται και «male matte,» αποτελείται από τη σιλουέτα του μοντέλου (εικόνα 2:2a) που τοποθετείται μπροστά από το επιθυμητό φόντο (εικόνα 2:2b). Αυτά τα δύο στοιχεία ενώνονται σε ένα μη εμφανισμένο film, όπου είναι όλο κανονικά εκθεμένο, εκτός της περιοχής που καλύπτει το male matte (εικόνα 2:2c). Το αποτέλεσμα είναι δηλαδή το

φόντο με μία μαύρη τρύπα, στην οποία θα μπει η εικόνα του μοντέλου. Στη συνέχεια το κομμάτι του film που έχει το κινούμενο μοντέλο (εικόνα 2:2d) πρέπει να τοποθετηθεί στη μαύρη οπή που του αντιστοιχεί. Για να γίνει αυτό χρησιμοποιείται ένα female matte (εικόνα 2:2e), η επιφάνεια του οποίου είναι μαύρη εκτός από ένα κομμάτι που ταυτίζεται πλήρως με τη σιλουέτα του κινούμενου μοντέλου. Το female matte τοποθετείται πάνω στο μη εμφανισμένο film που είναι ήδη εκθεμένο στο φόντο, ώστε να εμποδίσει την παραπάνω έκθεση αυτού, αφήνοντας μόνο τη σιλουέτα του κινούμενου μοντέλου «ευάλωτη» για έκθεση. Στη συνέχεια τοποθετείται το film με το κινούμενο μοντέλο στην ανάλογη θέση. Με την εμφάνιση του film, το αποτέλεσμα είναι το κινούμενο μοντέλο να βρίσκεται μπροστά στο επιλεγμένο φόντο, δηλαδή ο επιθυμητός συνδυασμός (εικόνα 2:2f) (Rickitt Richard, 2007).

Η παραπάνω διαδικασία ονομάστηκε «travelling mattes,» επειδή για κάθε καρέ του τελικού film πρέπει να χρησιμοποιηθούν ξεχωριστά male και female mattes, τα οποία διαφέρουν στη θέση, το σχήμα και το μέγεθός τους και ανταποκρίνονται σε διαφορετικές στάσεις του κινούμενου μοντέλου.



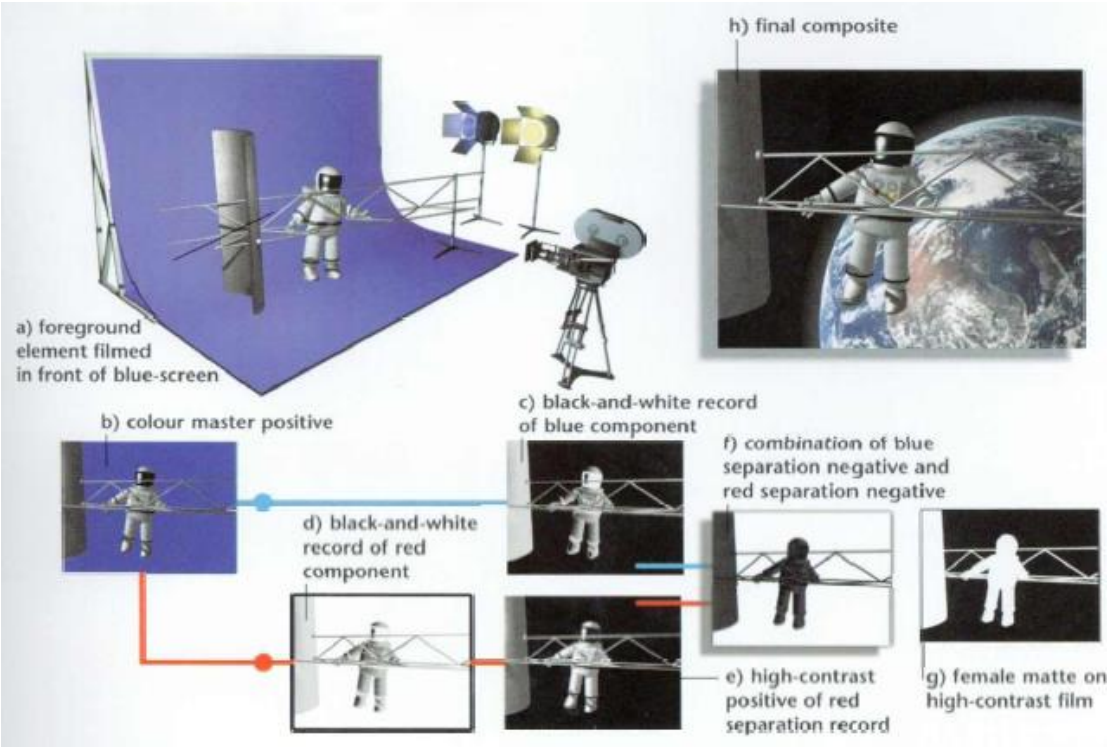
Εικόνα 2:2 Διαδικασία υλοποίησης του travelling matte (Rickitt Richard, 2007)

Παρόλη την αποτελεσματικότητά της, η μέθοδος αυτή γνώρισε μία σειρά τροποποιήσεων και εξελίξεων έως την εμφάνιση της πλέον χρησιμοποιούμενης τεχνικής, το διαχωρισμό υποβάθρου με χρήση ενός μπλε πανιού (Blue-Screen), κύριος στόχος της οποίας ήταν η κάλυψη της ανικανότητας των προηγούμενων τεχνικών να παράγουν έγχρωμο αποτέλεσμα.

2.1.2 Blue Screen

Η εμφάνιση της έγχρωμης φωτογραφίας κατέστησε ανεπαρκείς της ως τότε τεχνικές αντικατάστασης υποβάθρου, εξαιτίας της ανικανότητάς τους να καλύψουν το έγχρωμο μέσο στο παράγωγο film τους. Ως συνέπεια ήρθε μία νέα τεχνική που βασίζεται στη χρήση ενός μονοχρωματικού μπλε πανιού ως φόντο και αποτελεί προέκταση των διαφόρων εξελίξεων της travelling matte.

Όπως σε όλες τις εφαρμογές του travelling matte, το πλάνο χωρίζεται σε δύο βασικά στοιχεία, το φόντο και το εν κινήσει μοντέλο (συνήθως ηθοποιός), τα οποία αποτελούν και τη βάση για τη δημιουργία των δύο mattes – αρσενικού και θηλυκού. Αρχικά τοποθετείται το μοντέλο μπροστά από ένα μπλε φόντο και φωτογραφίζεται (εικόνα 2:3a). Σε αυτή τη διαδικασία πρέπει να προσεχθεί ιδιαίτερα ο φωτισμός που πέφτει πάνω στο μοντέλο: οποιαδήποτε απόχρωση του μπλε θα αφαιρεθεί στα επόμενα βήματα της διαδικασίας και θα αντικατασταθεί από το νέο φόντο, οπότε είθισται να φωτίζεται το μοντέλο με ένα διακριτικό κίτρινο προβολέα, ενώ το φόντο με ένα μπλε, για τον καλύτερο διαχωρισμό τους (Rickitt Richard, 2007).



Εικόνα 2:3 Διαδικασία υλοποίησης του Chroma Key (Rickitt Richard, 2007)

Το αρνητικό της φωτογραφίας εκτυπώνεται για τη δημιουργία ενός έγχρωμου θετικού (εικόνα 2:3b) το οποίο, με τη σειρά του, τυπώνεται σε ασπρόμαυρο film μέσω μπλε φίλτρου ώστε να παραχθεί ένα αρνητικό – διαχωριστικό των χρωμάτων – το οποίο παριστάνει ασπρόμαυρα τα μπλε στοιχεία της φωτογραφίας (εικόνα 2:3c). Σε αυτό το αρνητικό, το φόντο, που είναι το μοναδικό μπλε στοιχείο, είναι μαύρο. Ένα ακόμα αντίγραφο του αρχικού θετικού γίνεται σε ασπρόμαυρη ταινία, αυτή τη φορά περνώντας το από ένα κόκκινο φίλτρο. Μέσα από αυτή τη διαδικασία παράγεται ένα αρνητικό που διαχωρίζει τα κόκκινα του αρχικού θετικού (εικόνα 2:3d), στο οποίο αρνητικό το φόντο είναι καθαρό (άσπρο). Τυπώνοντας το αρνητικό αυτό σε υψηλής αντίθεσης ασπρόμαυρο χαρτί παράγεται ένα ακόμα θετικό, όπου το φόντο είναι μαύρο (εικόνα 2:3e). Με τον τρόπο αυτό, το αρνητικό που διαχωρίζει τα μπλε στοιχεία και το θετικό που διαχωρίζει τα κόκκινα έχουν και τα δύο μαύρο φόντο. Τυπώνονται και τα δύο στο ίδιο κομμάτι film υψηλής αντίθεσης, παράγοντας το θετικό matte που έχει άσπρο φόντο και μαύρο το μοντέλο (εικόνα 2:3f). Το θετικό αντίγραφό του είναι το female matte (εικόνα 2:3g), που έχει μαύρο φόντο και άσπρο το μοντέλο. Τα δύο τελευταία παράγωγα συνδυάζονται με το αρχικό θετικό και με το επιθυμητό φόντο σε έναν ειδικό εκτυπωτή που παράγει και την τελική εικόνα, όπου το μοντέλο βρίσκεται μπροστά από το επιλεγμένο φόντο (εικόνα 2:3h) (Rickitt Richard, 2007).

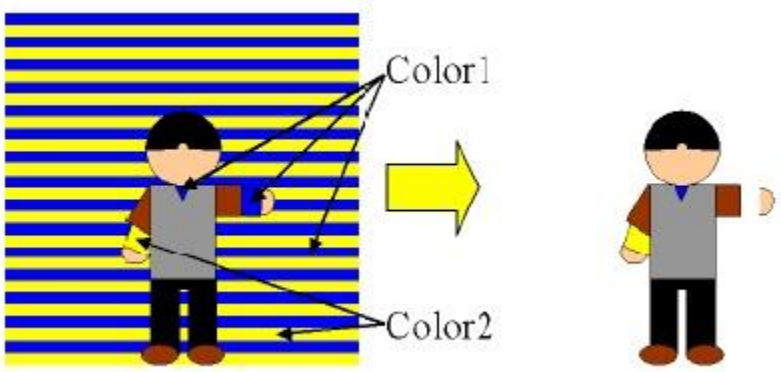
Η τεχνική αυτή αποτελεί την πρώτη απόπειρα ολοκλήρωσης της αφαίρεσης υποβάθρου με τη χρήση μονοχρωματικού φόντου και παρουσίαζε διάφορα προβλήματα, όπως τη λανθασμένη αντικατάσταση σημείων που χαρακτηρίζονταν από κάποια διαφάνεια (ποτήρια, καπνοί τσιγάρου κτλ). Επίσης, κάποιες συμβάσεις σχετικά με την ενδυμασία των ηθοποιών ακολουθούνται, ώστε να μην ταυτίζονται τα ρούχα τους με το χρώμα του φόντου. Ρούχα με μπλε αποχρώσεις αποφεύγονται, ενώ, σε μετέπειτα στάδια, προτιμήθηκε η αντικατάσταση του μπλε φόντου με πράσινο λόγω της μικρότερης συγγένειας του με πιο καθημερινά χρώματα και αποχρώσεις.

Με την είσοδο της ψηφιακής τεχνολογίας, η επεξεργασία των παραπάνω δεδομένων για την αντικατάσταση ενός μονοχρωματικού υποβάθρου έγινε αρκετά πιο εύκολη και ελεγχόμενη, δεδομένου ότι μπορούν να χρησιμοποιηθούν αλγόριθμοι για την ορθότερη αναγνώριση των μπλε (ή πράσινων) στοιχείων ενός καρέ και τον καθορισμό αν αυτά ανήκουν ή όχι στο φόντο.

2.2 Ολοκληρώσεις της Chroma Key

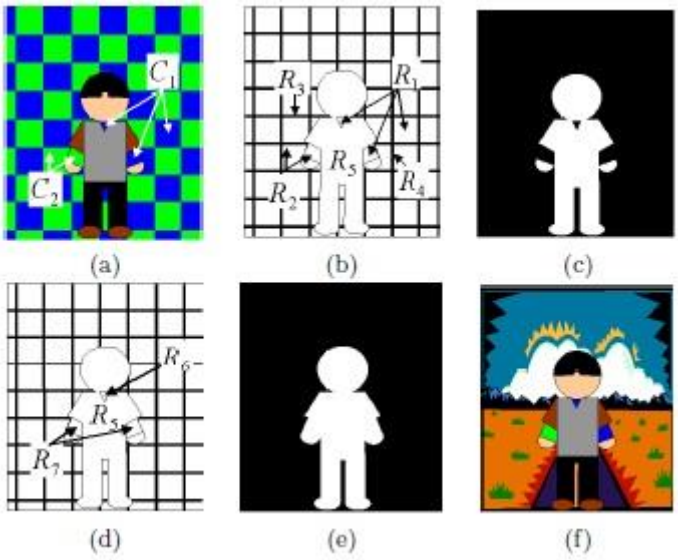
Ένα από τα βασικότερα προβλήματα που παρατηρείται στην εφαρμογή της τεχνολογίας αυτής είναι η αδυναμία διαχωρισμού μέρους του προσκηνίου που έχει το ίδιο χρώμα με το υπόβαθρο. Στην περίπτωση αυτή, κομμάτι του προσκηνίου συμμετέχει κανονικά στη διαδικασία της αντικατάστασης, γεγονός που επιφέρει μη θεμιτά αποτελέσματα. Έχουν προταθεί διάφορες τεχνικές ορθότερης λειτουργίας σε τέτοιες περιπτώσεις που βασίζονται κυρίως στη συνδρομή της ψηφιακής τεχνολογίας στην όλη διαδικασία. Ο Hiroki Agata και άλλοι υλοποίησαν διάφορες λύσεις Chroma Key για το διαχωρισμό του υποβάθρου από το προσκήνιο με αποδοτικότερο τρόπο, δίνοντας λύση στο προαναφερθέν πρόβλημα.

Η αρχική υλοποίηση (Hiroki Agata και άλλοι, 2008) είχε να κάνει με τη χρήση δύο χρωμάτων στο υπόβαθρο, τα οποία χωρίζονται μεταξύ τους με μία γραμμή. Όσον αφορά στο διαχωρισμό υποβάθρου από προσκήνιο, δημιουργείται το περίγραμμα του τελευταίου και ανιχνεύονται οι περιοχές όπου έχουν ίδιο ή παρόμοιο χρώμα με το υπόβαθρο. Για την ανίχνευση των περιοχών αυτών απαραίτητο είναι να συνορεύουν με το υπόβαθρο. Αν το χρώμα του προσκηνίου και του παρασκηνίου είναι το ίδιο και το περίγραμμα του πρώτου είναι παράλληλο με τις γραμμές που χωρίζουν τα χρώματα του δεύτερου, τότε η περιοχή αυτή του προσκηνίου αφαιρείται κατά την αφαίρεση του παρασκηνίου (εικόνα 2:4).



Εικόνα 2:4 Αφαίρεση κομματιού του προσκηνίου λόγω χρωματικής ταύτισης με το υπόβαθρο (Hiroki Agata και άλλοι, 2008)

Για την επίλυση του παραπάνω προβλήματος, υλοποιήθηκε ένα πρότυπο που βασίζεται σε διχρωματικό έλεγχο του υποβάθρου. Το πρότυπό αυτό χωρίζεται σε τέσσερις φάσεις: στην εξαγωγή του χρώματος του υποβάθρου (εικόνα 2:5a), στην εξαγωγή των γραμμών που διαχωρίζουν τα δύο χρώματα του υποβάθρου (εικόνα 2:5b,c), στην εξαγωγή του προσκηνίου (εικόνα 2:5d,e) και στη σύνθεση των δύο ετερογενών εικόνων (εικόνα 2:5f) (του προσκηνίου και του νέου υποβάθρου).



Εικόνα 2:5 Φάσεις αφαίρεσης και αντικατάστασης υποβάθρου (Hiroki Agata και άλλοι, 2008)

Με τον τρόπο που υλοποιείται η παραπάνω μέθοδος αποφεύγεται η αντικατάσταση μέχρι και πιθανών σημείων του προσκηνίου που έχουν τα ίδια χρώματα με του φόντου. Συγκεκριμένα, η μέθοδος χρησιμοποιεί την πληροφορία των δύο χρωμάτων του υποβάθρου και τις διαχωριστικές γραμμές αυτών για το διαχωρισμό του προσκηνίου από το παρασκήνιο.

Ο David Yamnitsky δημιούργησε μία άλλη τεχνική υλοποίησης Chroma Key, η λειτουργία της οποίας βασίζεται στη χρήση της Graphics Processing Unit (GPU) του υπολογιστή. Η λογική είναι ότι λόγω των απαιτήσεων σε γραφική δύναμη, οι κάρτες γραφικών εξελίχθηκαν ραγδαία και ταυτόχρονα έπεσε αισθητά η τιμή τους, ενώ ο μεγάλος αριθμός πυρήνων που διαθέτουν προωθεί ακόμα περισσότερο την παράλληλη επεξεργασία δεδομένων των διαφόρων φάσεων της τεχνικής Chroma Key. Η τεχνική που προτείνει χωρίζεται σε τέσσερα βήματα: στο σχεδιασμό του Chroma Key Matte (εικόνα 2:6), στην τελειοποίηση των ακρών του matte που δημιουργείται (εικόνα 2:7), στον περιορισμό των αποχρώσεων του φόντου που, λόγω του φωτός, αντανακλώνται στο προσκήνιο και, τέλος, στην εξομάλυνση των φωτιστικών συνθηκών του προσκηνίου και του νέου φόντου (εικόνα 2:8). Το τελευταίο στάδιο είναι ίσως και η βασικότερη παραλλαγή της όλης διαδικασίας, μια που δημιουργείται μία ομοιομορφία μεταξύ των δύο εικόνων που ενώνονται, έτσι ώστε το αποτέλεσμα να είναι όσο πιο ρεαλιστικό γίνεται.

Εικόνα 2:6 Δημιουργία του Chroma Key Matte (David Yamnitsky, 2009)

Εικόνα 2:7 Τελειοποίηση των ακρών (David Yamnitsky, 2009)

Εικόνα 2:8 Εξομάλυνση των φωτιστικών συνθηκών (David Yamnitsky, 2009)

3 Εργαλεία Λογισμικού που Χρησιμοποιήθηκαν

Τα εργαλεία που χρησιμοποιήθηκαν για την ολοκλήρωση της εφαρμογής ήταν τα παρακάτω:

- η βιβλιοθήκη OpenCV 2.1: Χρησιμοποιήθηκε λόγω της πληθώρας έτοιμων συναρτήσεων που προσφέρει για θέματα που σχετίζονται με την computer vision.
- η γλώσσα προγραμματισμού Visual C++: Η γλώσσα προγραμματισμού στην οποία γράφτηκε ο κώδικας όλης της εφαρμογής. Ένας από τους λόγους που προτιμήθηκε η συγκεκριμένη γλώσσα είναι επειδή η βιβλιοθήκη OpenCV είναι γραμμένη σε C και C++ (Bradski και άλλοι, 2008).
- το Microsoft Visual Studio 2008: Χρησιμοποιήθηκε για τη δημιουργία του γραφικού περιβάλλοντος της εφαρμογής, καθώς και για τη συγγραφή και αποσφαλμάτωση του κώδικα της.
- CMake: Απαραίτητο εργαλείο για τη μετατροπή του πηγαίου κώδικα της OpenCV σε project του Visual Studio 2008.
- το λειτουργικό σύστημα Windows 7 Ultimate: Το λειτουργικό σύστημα στο οποίο γράφτηκε και εκτελέστηκε η εφαρμογή ήταν το Windows 7 Ultimate, 64bit.

3.1 OpenCV 2.1

Η OpenCV αποτελεί μία open source βιβλιοθήκη με προσανατολισμό την computer vision. Δεδομένου ότι οι εφαρμογές που σχετίζονται με αυτόν τον κλάδο γνωρίζουν ιδιαίτερη ανάπτυξη τα τελευταία χρόνια (video games, ιατρικές εφαρμογές, κάμερες ασφαλείας), μία βιβλιοθήκη τέτοιου τύπου αποτελεί βασικό και χρήσιμο εργαλείο για τον εκάστοτε προγραμματιστή, ή ομάδα προγραμματιστών, που έχει σκοπό να ασχοληθεί με την ανάπτυξη εφαρμογών ή με την έρευνα. Διαθέτοντας πάνω από πεντακόσιες συναρτήσεις που επεκτείνονται σε πολλούς τομείς της computer vision, η OpenCV προσφέρει έτοιμα εργαλεία για απλές και σύνθετες λειτουργίες, ώστε ο προγραμματιστής να γλυτώνει χρήσιμο και απαραίτητο χρόνο για τη δουλειά του. Κύριος στόχος της είναι η υψηλή απόδοση και επίδοση εφαρμογών που τρέχουν σε πραγματικό χρόνο. Επίσης, επειδή η Μηχανική Εκμάθηση (Machine Learning) συνεργάζεται άμεσα με την Computer Vision, η OpenCV διαθέτει και μία βιβλιοθήκη Μηχανικής Εκμάθησης (Machine Learning Library – MLL).

Οι απαρχές της OpenCV μπορούν να εντοπιστούν στις αρχές του 1999, στα εργαστήρια της Intel. Η Intel, για την καλύτερη δυνατή απόδοση του υλικού (Hardware) που κατασκεύαζε, ανέπτυξε κάποιες εφαρμογές που «πίεζαν» τον επεξεργαστή με σκοπό να μετρήσουν την απόδοσή του. Μέσα στις εφαρμογές αυτές, κάποιες σχετίζονταν με την Computer Vision, όπως

ανίχνευση ακτινών σε πραγματικό χρόνο. Έχοντας ως δεδομένο τις εφαρμογές αυτές και σε συνεργασία με την ομάδα Software Performance Libraries και με ειδικούς βελτιστοποίησης της Intel στη Ρωσία, γεννήθηκε η OpenCV.

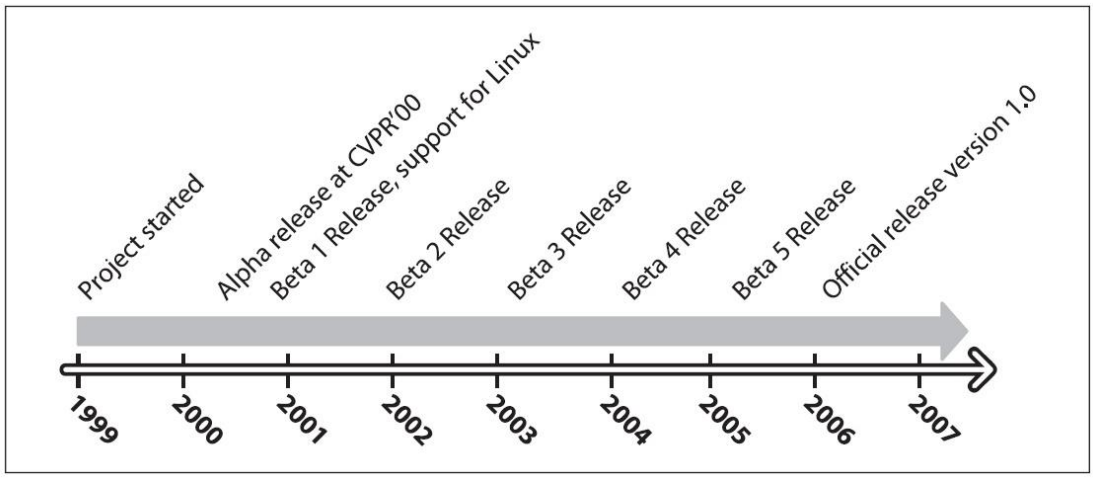
Οι αρχικοί στόχοι που τέθηκαν για την OpenCV, οι οποίοι αποτέλεσαν και τον κύριο λόγο δημιουργίας και έκδοσής της ως ελεύθερη βιβλιοθήκη, είναι τρεις:

1. Δε θα εφεύρει ο κάθε νέος προγραμματιστής τον τροχό. Η OpenCV προσφέρει έτοιμες λειτουργίες και εργαλεία, έτσι ώστε να εξοικονομείται σημαντικός χρόνος για την ανάπτυξη της εφαρμογής και να μη σπαταλιέται σε τετριμμένες και ήδη υλοποιημένες εργασίες.

2. Ευανάγνωστος και μεταφέρσιμος κώδικας. Έχοντας μία υποδομή, ο κάθε προγραμματιστής θα δουλεύει με συγκεκριμένες συναρτήσεις και δομές. Με τον τρόπο αυτό θα υπάρχει μία σχετική ομοιογένεια στην πληθώρα των εργασιών και μελετών. Επίσης, επειδή η OpenCV αποτελεί μία έτοιμη βιβλιοθήκη, το σύνολο των προγραμμάτων και εφαρμογών που γράφονται με βάση αυτήν μπορούν εύκολα να μεταφερθούν, να επεξεργαστούν και να τρέξουν σε άλλα συστήματα, με την προϋπόθεση να είναι εγκατεστημένη η βιβλιοθήκη αυτή.

3. Ανάπτυξη εμπορικών εφαρμογών που αξιοποιούν την Computer Vision. Ο κώδικας που εμπεριέχεται στη βιβλιοθήκη είναι ελεύθερος και μπορεί να χρησιμοποιηθεί από τον καθένα, όμως οι εφαρμογές που αναπτύσσονται δεν είναι απαραίτητο να είναι επίσης ελεύθερες ή/ και ανοιχτές.

Η OpenCV έχει χιλιάδες χρήστες αυτή τη στιγμή και ο αριθμός αυτών αυξάνεται συνεχόμενα. Αυτή είναι ίσως και η μεγαλύτερη επιτυχία της, κυρίως γιατί μία ανοιχτού τύπου βιβλιοθήκη επωφελείται και εμπλουτίζεται από τη συνδρομή των χρηστών της. Επίσης, η OpenCV αποτελεί βασικό εργαλείο έρευνας σε πολλά Ιδρύματα, οπότε η περεταίρω ανάπτυξη της σε διάφορους τομείς της Computer Vision είναι πιθανή.



Εικόνα 3:1: Χρονοδιάγραμμα της OpenCV

Η OpenCV, με βάση όλα τα παραπάνω και ιδιαίτερα των τριών στόχων της, είναι χρήσιμο εργαλείο για την παρούσα εργασία. Βασική λειτουργία της εφαρμογής είναι η απόκτηση και προβολή βίντεο από μία web camera, η αφαίρεση του ομοιόμορφου υποβάθρου και η αντικατάστασή του με μία άλλην εικόνα, σε πραγματικό χρόνο. Η OpenCV προσφέρει έτοιμες δομές και συναρτήσεις που διευκολύνουν τη συνεργασία με μία web camera, την απόκτηση και προβολή των εικόνων που αυτή λαμβάνει, καθώς και την αντικατάσταση των pixels της μίας εικόνας με αυτών μίας άλλης. Επομένως, η εξοικονόμηση χρόνου από την ανάπτυξη αυτών των εργαλείων θα αναλωθεί στην ανάπτυξη του συνόλου της εφαρμογής, κάτι που συνάδει και με τους στόχους που τέθηκαν για την ανάπτυξη της βιβλιοθήκης. Ο βελτιστοποιημένος κώδικας και η υψηλή απόδοση είναι ένας ακόμα παράγοντας που προτρέπει στη χρήση της βιβλιοθήκης, αφού η εκτέλεση της εφαρμογής σε πραγματικό χρόνο είναι πρωτίστης σημασίας.

3.2 Microsoft Visual Studio 2008

Για τις ανάγκες της εφαρμογής έπρεπε να αναπτυχθεί ένα γραφικό περιβάλλον με το οποίο θα γεφυρώνεται η επικοινωνία του χρήστη με το σύνολο των λειτουργιών της εφαρμογής. Για το σκοπό αυτό, δηλαδή την ανάπτυξη του γραφικού περιβάλλοντος, χρησιμοποιήθηκε το Microsoft Visual Studio 2008, ένα IDE (Integrated Development Environment) της Microsoft, το οποίο προσφέρει στο χρήστη μία συλλογή εργαλείων για σχεδιασμό και ανάπτυξη γραφικού περιβάλλοντος για διάφορες πλατφόρμες, όπως τα Windows και το .NET Framework. Ταυτόχρονα, υποστηρίζει πολλές γλώσσες προγραμματισμού, όπως οι C, C++, VB .NET, C#, ενώ με ειδικές επεκτάσεις μπορεί να συνεργαστεί και την Python, την Ruby κ.α.

Ο βασικός λόγος επιλογής του συγκεκριμένου έναντι άλλων IDEs ήταν το λειτουργικό σύστημα στο οποίο αναπτύχθηκε και εκτελέστηκε η εφαρμογή, δηλαδή τα Windows 7. Για τη Visual C++, το Visual Studio 2008 προσθέτει τη νέα έκδοση του Microsoft Foundation Classes (MFC 9.0), το οποίο υποστηρίζει το νέο γραφικό περιβάλλον που εισήχθηκε με τα Windows Vista. Επίσης, το MFC 9.0 γεφυρώνει το χάσμα που υπάρχει μεταξύ των στοιχείων του .NET Framework και του MFC (*Wikipedia*) και δεδομένου ότι το Visual Studio είναι ένα IDE με πλήρη υποστήριξη .NET Framework, θεωρήθηκε και το κατάλληλο για την ανάπτυξη της εφαρμογής. Η εξοικείωση με το συγκεκριμένο IDE, καθώς και η ποιότητα και η αξιοπιστία του ήταν δύο ακόμα σημαντικοί λόγοι για την επιλογή του.

3.3 Visual C++

Η Visual C++ αποτελεί ένα IDE της Microsoft για την C++. Έχει εργαλεία για ανάπτυξη και αποσφαλμάτωση C++ κώδικα, καθώς και κώδικα γραμμένο για το Windows API, DirectX API και το .NET Framework. Οι λόγοι για τους οποίους επιλέχτηκε η Visual C++ ως γλώσσα προγραμματισμού για τη συγγραφή του κώδικα είναι δύο:

- Το Visual Studio 2008 υποστηρίζει πλήρως την Visual C++ και το MFC 9.0, που ενθυλακώνει κομμάτι του Windows API σε C++ classes.

- Η OpenCV είναι γραμμένη σε C και C++.

3.4 OpenCV και συμβατά video file formats

Πέραν της λήψης και άμεσης επεξεργασίας video σε πραγματικό χρόνο, η OpenCV δίνει τη δυνατότητα εισαγωγής και επεξεργασίας ενός ήδη υπάρχοντος video. Στην περίπτωση αυτή μπορεί να υπάρξουν θέματα ασυμβατότητας, που για την καλύτερη δυνατή καταπολέμησή τους ο χρήστης οφείλει να γνωρίζει τις εναλλακτικές που του προσφέρονται όσον αφορά στους codecs που υποστηρίζονται από τη βιβλιοθήκη.

Η σωστότερη τακτική είναι τα προς χρήση video να είναι ασυμπίεστα. Στην περίπτωση που ένα video είναι συμπιεσμένο, συνίσταται η μετατροπή του σε ασυμπίεστη μορφή και η εκ των υστέρων επεξεργασία του με τα διάφορα εργαλεία της OpenCV. Στον παρακάτω πίνακα παρουσιάζονται μερικά από τα codecs που είναι πλήρως συμβατά με την OpenCV.

Container	FourCC	Name	Description
AVI	'DIB '	RGB(A)	Uncompressed RGB, 24 or 32 bit
AVI	'I420'	RAW I420	Uncompressed YUV, 4:2:0 chroma subsampled
AVI	'IYUV'	RAWI420	Identical to I420

Πίνακας 1 Συμβατά codecs με OpenCV

Οι παραπάνω επιλογές λειτουργούν σε όλες τις πλατφόρμες (Windows, Linux και Mac OS X), υπάρχει όμως ένα πιθανό πρόβλημα χώρου, δεδομένου ότι οι ασυμπίεστες μορφές video έχουν πάρα πολύ όγκο. Για εξοικονόμηση χώρου μπορούν να χρησιμοποιηθούν ειδικοί συμπιεστές που δεν οδηγούν σε μειωμένη ποιότητα video, δεν είναι όμως συμβατοί με όλες τις πλατφόρμες. Κάποιοι από αυτούς τους συμπιεστές είναι οι HuffYUV, CorePNG, Motion PNG or Motion JPEG2000.

Για τη μετατροπή ενός συμπιεσμένου αρχείου video σε ασυμπίεστο, μπορούν να χρησιμοποιηθούν τα εξής εργαλεία:

- VirtualDub για Windows
- ffmpeg για Linux και Mac OS X
- QuickTime Pro για Windows και Mac OS X (εμπορικό λογισμικό)

4 Η λειτουργία της εφαρμογής

Η βασική λειτουργία της εφαρμογής είναι η αντικατάσταση του υποβάθρου στην εικόνα που αναπαράγεται μέσω μίας web camera με βάση ένα ή περισσότερα pixels που θα επιλέξει ο χρήστης. Για την υλοποίηση αυτών των λειτουργιών χρησιμοποιούνται συναρτήσεις και δομές της OpenCV και του MFC, ενώ ήταν απαραίτητη η ανάπτυξη ενός γραφικού περιβάλλοντος για την καλύτερη δυνατή επικοινωνία του χρήστη με το σύστημα και η δημιουργία ορισμένων συναρτήσεων με εξειδικευμένη λειτουργία. Σε αυτήν την ενότητα γίνεται μία αναφορά σε όλες τις δομές και τις συναρτήσεις της OpenCV και του MFC που χρησιμοποιήθηκαν, καθώς και στις λοιπές συναρτήσεις που έπρεπε να δημιουργηθούν, ενώ περιγράφεται και ο τρόπος λειτουργίες της εφαρμογής. Για την αναλυτικότερη περιγραφή του πώς επικοινωνούν οι συναρτήσεις μεταξύ τους και για τη ροή εκτέλεσης του προγράμματος, ο αναγνώστης μπορεί να ανατρέξει στα σχόλια του πηγαίου κώδικα που βρίσκονται στο Παράρτημα 2.

4.1 Δομές και συναρτήσεις της OpenCV

Όπως έχει ήδη αναφερθεί, με την OpenCV πολλές λειτουργίες, σχετικές με την Computer Vision, επιτυγχάνονται εύκολα και αποδοτικά. Στην παρούσα εφαρμογή, η OpenCV διευκολύνει στη δημιουργία αντικειμένων που φιλοξενούν εικόνες και video, στην ανίχνευση κάμερας και αναπαραγωγή της εικόνας που αυτή «βλέπει,» όπως και στην αντικατάσταση των pixels του video μίας web camera με μία άλλη εικόνα ή video. Στην ενότητα αυτή γίνεται εκτενής αναφορά στις δομές και στις συναρτήσεις της OpenCV που χρησιμοποιήθηκαν, καθώς και στη λειτουργία τους.

4.1.1 Δομές

- *CvCapture**: Τύπος αντικειμένου για τη φόρτωση και επεξεργασία video. Πέραν των αρχείων video που μπορούν να φορτωθούν, υπάρχει δυνατότητα σύνδεσης μίας web camera με ένα CvCapture* αντικείμενο. Παρόλο που μπορεί να χρησιμοποιηθεί σε δύο περιπτώσεις, ο τρόπος επεξεργασίας και αναπαραγωγής του video αρχείου ή της εικόνας της web camera είναι ενιαίος, προσφέροντας έτσι αρκετή ευελιξία. Σε ένα CvCapture*: αντικείμενο μπορούν να καθοριστούν τα διάφορα χαρακτηριστικά του video, όπως πχ η ανάλυση (ύψος και πλάτος).
- CvPoint: Τύπος αντικειμένου που χρησιμοποιείται για την αποθήκευση των συντεταγμένων x και y ενός σημείου. Έχει διάφορες χρήσης, στην παρούσα εφαμοργή όμως βοηθάει στον προσδιορισμό της ακριβής θέσης των pixels που επιλέγει ο χρήστης και τον υπολογισμό του μέσου όρου των RGB τιμών αυτών.
- *IplImage**: Τύπος αντικειμένου για τη φόρτωση και επεξεργασία εικόνων. Περιέχει πληθώρα συναρτήσεων για την αλλαγή διαφόρων χαρακτηριστικών που διέπουν μία

εικόνα, όπως το βάθος χρώματος (depth), ανάλυση (ύψος και πλάτος), κανάλια (nChannels) κτλ.

4.1.2 Συναρτήσεις

- CvCapture* cvCreateCameraCapture(int index): Καταχωρεί ένα CvCapture* αντικείμενο, το οποίο φορτώνει και διαβάζει το stream της κάμερας που ορίζεται από το index που δίνεται ως πέρασμα στο κάλεσμα της συνάρτησης. Αν δοθεί το -1, τότε βρίσκεται είτε η μοναδική κάμερα του συστήματος, είτε μία οποιαδήποτε από τις υπάρχουσες.
- CvCapture* cvCreateFileCapture(const char* filename): Δημιουργεί και επιστρέφει ένα CvCapture* αντικείμενο, το οποίο φορτώνει και διαβάζει το stream από ένα αρχείο video, το οποίο καθορίζεται από την char* παράμετρο που δίνεται ως όρισμα.
- IplImage* cvCreateImage(CvSize size, int depth, int channels): Καταχωρεί τον απαραίτητο χώρο για ένα IplImage* αντικείμενο, καθορίζοντας ταυτόχρονα την ανάλυσή του (size), το βάθος χρώματός του (depth) και τα κανάλια (channels) μέσω των ορισμάτων που περνιόνται στο κάλεσμά της.
- void cvDestroyWindow(const char* name): Καταστρέφει ένα ήδη υπάρχον παράθυρο. Το όρισμα name καθορίζει το παράθυρο προς καταστροφή.
- double cvGetCaptureProperty(CvCapture* capture, int property_id): Ανασύρει μία ιδιότητα του δοθέντος CvCapture* αντικειμένου. Η ιδιότητα καθορίζεται από το property_id που περνιέται ως όρισμα.
- IplImage * cvLoadImage (const char* filename, int iscolor=CV_LOAD_IMAGE_COLOR): Φορτώνει μία εικόνα από ένα αρχείο (καθορίζεται από την παράμετρο filename) και επιστρέφει έναν pointer τύπου IplImage* που «δείχνει» στην εικόνα που φορτώθηκε.
- IplImage* cvQueryFrame(CvCapture* capture): Λαμβάνει το αμέσως επόμενο frame από ένα video file ή camera που ορίζεται από το CvCapture* αντικείμενο που περνιέται στο κάλεσμα της συνάρτησης. Αποτελεί ουσιαστικά ένα συνδυασμό δύο άλλων συναρτήσεων της OpenCV, της cvGrabFrame και cvRetrieveFrame, που, αντίστοιχα, παίρνουν το επόμενο frame ενός video file ή camera και το ανασύρουν. Επιστρέφεται ένας pointer (IplImage*) στο επόμενο frame που ανασύρθηκε.
- void cvReleaseCapture(CvCapture** capture): Απελευθερώνει τη μνήμη που πιάνει ένα CvCapture* αντικείμενο. Συνίσταται να χρησιμοποιείται όταν πλέον αυτό το αντικείμενο δε χρησιμοποιείται.
- void cvReleaseImage(IplImage** img): Ίδια λειτουργία με την cvReleaseCapture, μόνο που ενεργεί πάνω σε IplImage* αντικείμενα.
- void cvResize(const CvArr* I, CvArr* J, int interpolation=CV_INTER_LINEAR): Η συνάρτηση αυτή χρησιμοποιείται για την αλλαγή του μεγέθους μίας εικόνας στο μέγεθος μίας δεύτερης. Δίνονται δύο ορίσματα στο κάλεσμα της, όπου το πρώτο είναι η πρωτότυπη εικόνα, ενώ το δεύτερο είναι η εικόνα στην οποία θα αποθηκευθεί η πληροφορία της πρώτης, προσαρμοσμένη όμως στο μέγεθος της δεύτερης.

- int cvSetCaptureProperty(CvCapture* capture, int property_id, double value): Θέτει μία τιμή στη δοθείσα ιδιότητα του CvCapture* αντικειμένο που περνιέται στο κάλεσμα της συνάρτησης.
- void cvSetMouseCallback(const char* window_name, CvMouseCallback on_mouse, void* param=NULL): Θέτει μία συνάρτηση που θα χειρίζεται τα διάφορα mouse events που συμβαίνουν σε ένα παράθυρο. Το παράθυρο στο οποίο υπάρχει ανάγκη ελέγχου των mouse events δίνεται μέσω του πρώτου ορίσματος, ενώ με το δεύτερο όρισμα ορίζεται ποια συνάρτηση θα χειρίζεται τα προαναφερθέντα mouse events.
- void cvShowImage(const string& winname, const Mat& image): Εμφανίζει μία εικόνα σε ένα παράθυρο που ορίζεται από το string αντικείμενο που περνιέται ως όρισμα στο κάλεσμα της συνάρτησης. Η δεύτερη παράμετρος της συνάρτησης αναφέρεται στο IplImage* αντικείμενο που θα προβληθεί. Να σημειωθεί εδώ ότι στην OpenCV η δημιουργία ενός παραθύρου γίνεται με τον καθορισμό απλώς ενός string που αναφέρεται στο όνομα που θα έχει το παράθυρο. Για την εμφάνιση μίας εικόνας, για παράδειγμα, σε ένα παράθυρο, αρκεί να δοθεί ένα όνομα για το παράθυρο αυτό. Για την εμφάνιση άλλων στοιχείων στο ίδιο παράθυρο αρκεί να δοθεί πάλι το ίδιο όνομα (για αυτό είναι καλή τακτική να κρατιέται το όνομα του εκάστοτε παραθύρου σε ξεχωριστά string αντικείμενα, ώστε να υπάρχει μία αναφορά).
- int cvWaitKey(int delay): Η συνάρτηση περιμένει το πάτημα ενός κουμπιού για το χρονικό διάστημα που δίνεται ως πέρασμα στο κάλεσμά της και επιστρέφει τον κωδικό που αντιστοιχεί στο κουμπί που πατήθηκε. Αν δεν πατηθεί κάποιο κουμπί, η συνάρτηση επιστρέφει την τιμή -1. Αν δοθεί αριθμός μικρότερος ή ίσως με το 0, η εκτέλεση παγώνει έως ότου πατηθεί κάποιο κουμπί από το χρήστη, αλλιώς η εκτέλεση συνεχίζεται μετά το πέρασμα του χρονικού διαστήματος που ορίστηκε (μετριέται σε milliseconds). Μία συχνή χρήση της είναι για τον καθορισμό του refresh rate ενός video που προβάλλεται. Περνώντας, για παράδειγμα, τον αριθμό 33 στη cvWaitKey, η εκτέλεση θα «παγώσει» για 33 ms.

```
while(i)

{

        frame = cvQueryFrame(cam;  //Πέρασμα του επόμενου frame.;

        c = cvWaitKey(33);  //Πάγωμα για 33ms

        cvShowImage(frame);  //Εμφάνιση του τρέχοντος frame.

        if (c == 27)  //Αν πατηθεί το escape...

                break;

}//while(i)
```

Με την cvWaitKey μπορούν ακόμα να καθοριστούν συγκεκριμένες εκτελέσεις με το πάτημα κάποιου κουμπιού. Στο παραπάνω παράδειγμα, όταν πατηθεί το πλήκτρο Esc (escape), που ισοδυναμεί με τον αριθμό 27, η εκτέλεση του ατέρμονος βρόχου σταματάει (break).

4.2 Συναρτήσεις της εφαρμογής

Για τις ανάγκες τις εφαρμογές αναπτύχθηκαν κάποιες συναρτήσεις για την υλοποίηση βασικών λειτουργιών που απαιτούνται για την ψηφιακή αντικατάσταση υποβάθρου. Η βασική λειτουργία αυτών των συναρτήσεων δίνεται παρακάτω, ενώ αναλυτική περιγραφή του σκοπού του κάθε ορίσματος και της μεταφοράς των δεδομένων μεταξύ των μεθόδων βρίσκεται στα σχόλια του πηγαίου κώδικα της εφαρμογής στο Παράρτημα 2.

- void createCamera (CvCapture* cam, char* camWindow, int i, bool bIoV, CWnd* mainWindow, int btnCamID, int txtSxID, int txtFxID, int txtSyID, int txtFyID, int comboID): Είναι η πρώτη συνάρτηση που χρησιμοποιείται για το ξεκίνημα της διαδικασίας αναπαραγωγής της εικόνας της camera και για την αντικατάσταση του υποβάθρου. Καλείται όταν ο χρήστης πατήσει το κουμπί «Start webcamera» και περνιόνται τα απαραίτητα ορίσματα για τη σωστή λειτουργία της εφαρμογής.
- void mouseOnCallback (int event, int x, int y, int flags, void* param): Χειρίζεται τα διάφορα mouse events που συμβαίνουν. Δηλώνεται μέσω της cvSetMouseCallback για την επεξεργασία αυτών και η υπογραφή της είναι προκαθορισμένη από την OpenCV. Το πρώτο όρισμα χρησιμοποιείται για τον προσδιορισμό του event που προκάλεσε το κάλεσμά της, τα επόμενα δύο int ορίσματα αναφέρονται στις συντεταγμένες του παραθύρου που έγινε το mouse event που προκάλεσε την κλήση της, το flags αναφέρεται σε πιθανά κουμπιά που ήταν πατημένα κατά την κλήση της συνάρτησης, ενώ το τελευταίο όρισμα είναι μία πιθανή παράμετρος που μπορεί να περαστεί προαιρετικά και είναι οποιουδήποτε είδους. Μέσω αυτής της συνάρτησης προσδιορίζονται οι ακριβείς συντεταγμένες του/ των pixel/s που επιλέχθηκε/αν από το χρήστη και καλείται η colourCalculation, η λειτουργία της οποίας αναφέρεται παρακάτω.
- void replacement (CvPoint sPt, CvPoint fPt, IplImage* camFrame, char* window, CWnd* mainWindow, int txtSxID, int txtFxID, int txtSyID, int txtFyID): Η πρώτη replacement χρησιμοποιείται για την αντικατάσταση του υποβάθρου της camera με μία εικόνα.
- void replacement (CvPoint sPt, CvPoint fPt, IplImage* camFrame, IplImage* videoFrame, char* camWindow, CWnd* mainWindow, int txtSxID, int txtFxID, int txtSyID, int txtFyID): Αυτή η replacement χρησιμοποιείται για την αντικατάσταση του υποβάθρου της camera με ένα video.
- void colourCalculation (int sX, int sY, int fX, int fY, IplImage* picture): Η colourCalculation χρησιμοποιείται για τον υπολογισμό του μέσου όρου των τιμών RGB των pixels που επιλέχθηκαν. Με τα πρώτα τέσσερα ορίσματα καθορίζονται τα pixels αυτά, ενώ το τελευταίο όρισμα είναι το frame της camera τη στιγμή που έγινε το mouse event που προκάλεσε την κλήση της mouseOnCallback. Ο μέσος όρος που υπολογίζεται χρησιμοποιείται από την replacement για την αντικατάσταση υποβάθρου.

- void appendEditText(CEdit& edit, LPCTSTR pszText): Η συνάρτηση αυτή χρησιμοποιείται για την προσθήκη περιεχομένου κειμένου στο edit box που βρίσκεται στο κεντρικό παράθυρο της εφαρμογής.

4.3 Δομές και συναρτήσεις του MFC

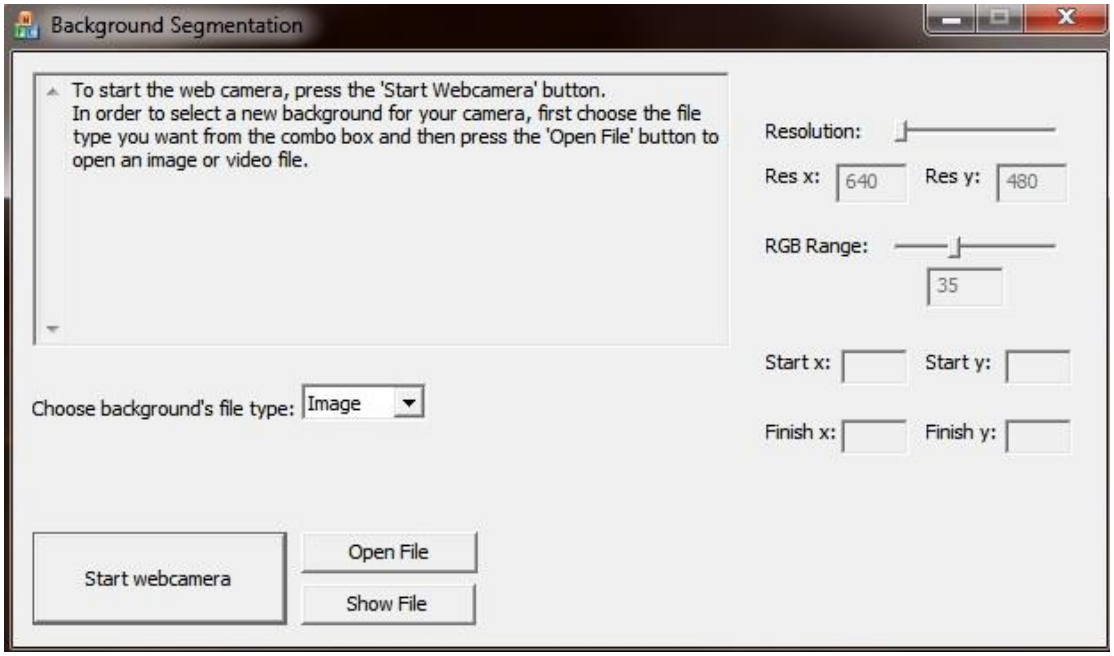
Δεδομένου ότι η εφαρμογή αναπτύχθηκε στο MFC, για τη σωστή και αποτελεσματική λειτουργία του γραφικού περιβάλλοντος ήταν απαραίτητη η χρήση δομών και συναρτήσεων που ανήκουν στο MFC. Οι δομές που χρησιμοποιήθηκαν αναφέρονται παρακάτω:

- CButton: Χρησιμοποιείται για τη δημιουργία αντικειμένων για το χειρισμό των διαφόρων button controls που υπάρχουν στο κεντρικό παράθυρο διαλόγου της εφαρμογής.
- CSliderCtrl: Χρησιμοποιείται για τη δημιουργία αντικειμένων για το χειρισμό των slider controls που υπάρχουν στο κεντρικό παράθυρο διαλόγου της εφαρμογής..
- CEdit: Χρησιμοποιείται για τη δημιουργία αντικειμένων για την πρόσβαση και επεξεργασία του περιεχομένου των text controls που υπάρχουν στο κεντρικό παράθυρο διαλόγου της εφαρμογής.
- CComboBox: Χρησιμοποιείται για τη δημιουργία αντικειμένου για το χειρισμό του combo box που υπάρχει στο κεντρικό παράθυρο διαλόγου της εφαρμογής.
- CWnd*: Χρησιμοποιείται για τη δημιουργίας ενός αντικειμένου που αναφέρεται σε παράθυρα της εφαρμογής. Η χρήση του περιορίζεται στην αναφορά του κεντρικού παράθυρου της εφαρμογής, ώστε να υπάρχει δυνατότητα πρόσβασης και επεξεργασίας των controls που βρίσκονται μέσα σε αυτό, όταν το focus είναι σε κάποιο άλλο παράθυρο.
- CToolTipCtrl: Χρησιμοποιείται για τη δημιουργία αντικειμένου για την ανάθεση tooltips στα διάφορα controls της εφαρμογής.

Ταυτόχρονα, χρησιμοποιείται μία πληθώρα συναρτήσεων για την απόκτηση ή/ και την επεξεργασία των τιμών των παραπάνω controls. Αναφορά και αναλυτική περιγραφή της λειτουργίας των συναρτήσεων αυτών βρίσκονται στα σχόλια του πηγαίου κώδικα της εφαρμογής στο Παράρτημα 2.

4.4 Λεπτομερής περιγραφή της λειτουργίας της εφαρμογής

Με την έναρξη της εφαρμογής εμφανίζεται το παρακάτω παράθυρο διαλόγου:



Εικόνα 4:1 Βασικό παράθυρο διαλόγου

Στο παράθυρο αυτό βρίσκονται όλα τα απαραίτητα controls για τη λειτουργία της εφαρμογής και την εκτέλεση των διαφόρων λειτουργιών της. Το combo box καθορίζει τον τύπο αρχείου του νέου υποβάθρου (εικόνα ή video), το «Open File» κουμπί ανοίγει ένα παράθυρο για φόρτωση ενός αρχείου εικόνας ή video (ανάλογα με το τι έχει επιλεγεί στο combo box), το «Show File» κουμπί εμφανίζει το αρχείο που επιλέχθηκε, ενώ το «Start webcamera» κουμπί ανοίγει την εικόνα της web camera. Με τα δύο sliders καθορίζονται, αντίστοιχα, η ανάλυση της web camera, και κατ' επέκταση του αρχείου που επιλέχθηκε, και το RGB range που θα προσθαφαιρείται στις RGB τιμές των pixels που θα επιλέξει ο χρήστης. Τέλος, υπάρχει ένα μεγάλο edit box που περιέχει οδηγίες για το τι μπορεί να κάνει ο χρήστης στην τρέχουσα φάση εκτέλεσης του προγράμματος.

Για την έναρξη της διαδικασίας της αντικατάστασης υπάρχουν δύο προϋποθέσεις:

1. Να επιλεγεί κάποιο υπόβαθρο μέσω του «Open File.»
2. Να ξεκινήσει η web camera μέσω του «Start webcanera.»

Με το πάτημα του τελευταίου, εμφανίζεται μία στατική εικόνα που αποτελεί και το πρώτο frame που λαμβάνει η web camera. Η αντικατάσταση υποβάθρου ξεκινάει με το που επιλέξει ο χρήστης με το ποντίκι του κάποιο (ή κάποια) pixel της εικόνας. Τη στιγμή της επιλογής ανακτάται η ακριβής τιμή RGB του (ή των) επιλεγμένου pixel και στην τιμή αυτή προσθαφαιρείται το RGB range που καθορίζεται από το σχετικό slider του βασικού παραθύρου. Με τον τρόπο αυτό δημιουργείται ένα εύρος τιμών RGB, βάσει του οποίου υλοποιείται και η αντικατάσταση υποβάθρου στο video της web camera. Πιο συγκεκριμένα, έστω ότι το pixel που επιλέχθηκε έχει τιμές RGB 25, 38, 55 αντίστοιχα (δηλαδή Red = 25, Green = 38 και Blue = 55) και το RGB range έχει καθοριστεί να είναι 50. Αυτομάτως, δημιουργείται το εύρος τιμών R 0-75, G 0-88, B 5-105.

Στη συνέχεια γίνεται μία ένα προς ένα σάρωση όλων των pixels του τρέχοντος frame της web camera και κάθε pixel που έχει τιμές RGB μέσα στο προαναφερθέν εύρος, αντικαθίσταται με το αντίστοιχο pixel του επιλεγμένου υποβάθρου. Αν, για παράδειγμα, το pixel του τρέχοντος frame της web camera που βρίσκεται στη θέση (10,15) έχει τιμές RGB 20, 15 και 100, θα αντικατασταθεί με το αντίστοιχο pixel (δηλαδή αυτό που βρίσκεται στη θέση (10,15)) του νέου υποβάθρου, ενώ αν έχει τιμές 20, 15 και 108, δε θα αντικατασταθεί.

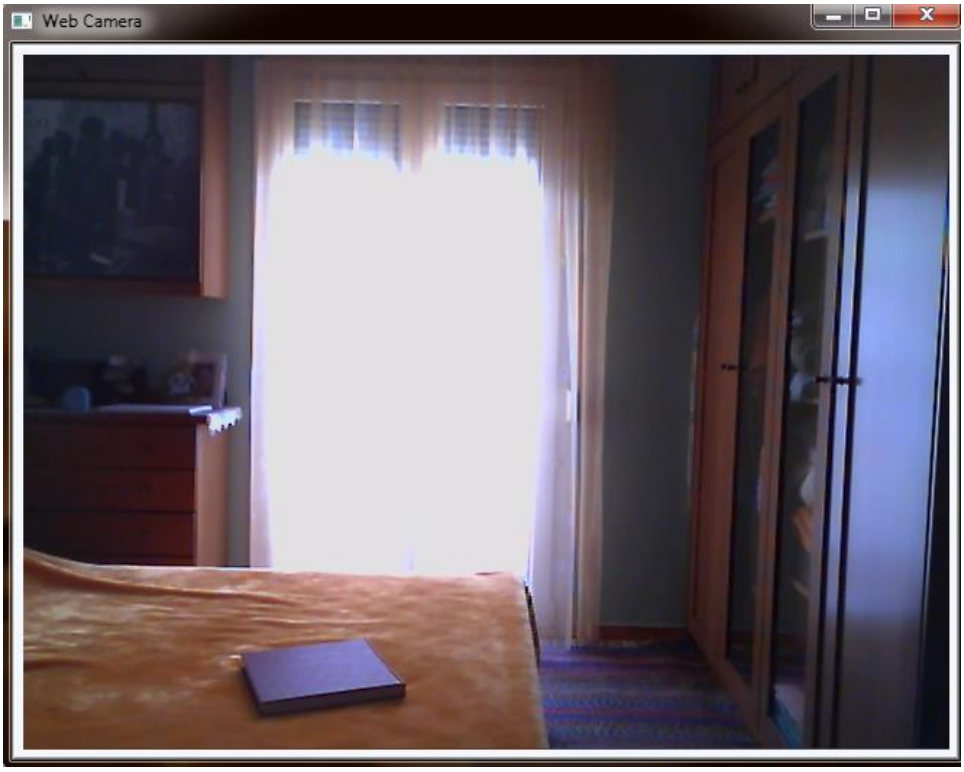
Με τον τρόπο αυτό, pixels που έχουν συγγενικές RGB τιμές αντικαθίστανται με τα αντίστοιχα pixels ενός άλλου αρχείου εικόνας ή video. Η λογική της αντικατάστασης υποβάθρου απαιτεί την ύπαρξη ενός ομοιόμορφου φόντου, οπότε με την παραπάνω μεθοδολογία επιτυγχάνεται εύκολα ο στόχος αυτός.

Στην περίπτωση που το νέο υπόβαθρο είναι video, λαμβάνεται το επόμενο frame αυτού όταν έχει ολοκληρωθεί η σάρωση και η αντικατάσταση του αμέσως προηγούμενου frame της web camera, ώστε να υπάρχει μία λογική συνέχεια στο video που προβάλλεται ως υπόβαθρο.

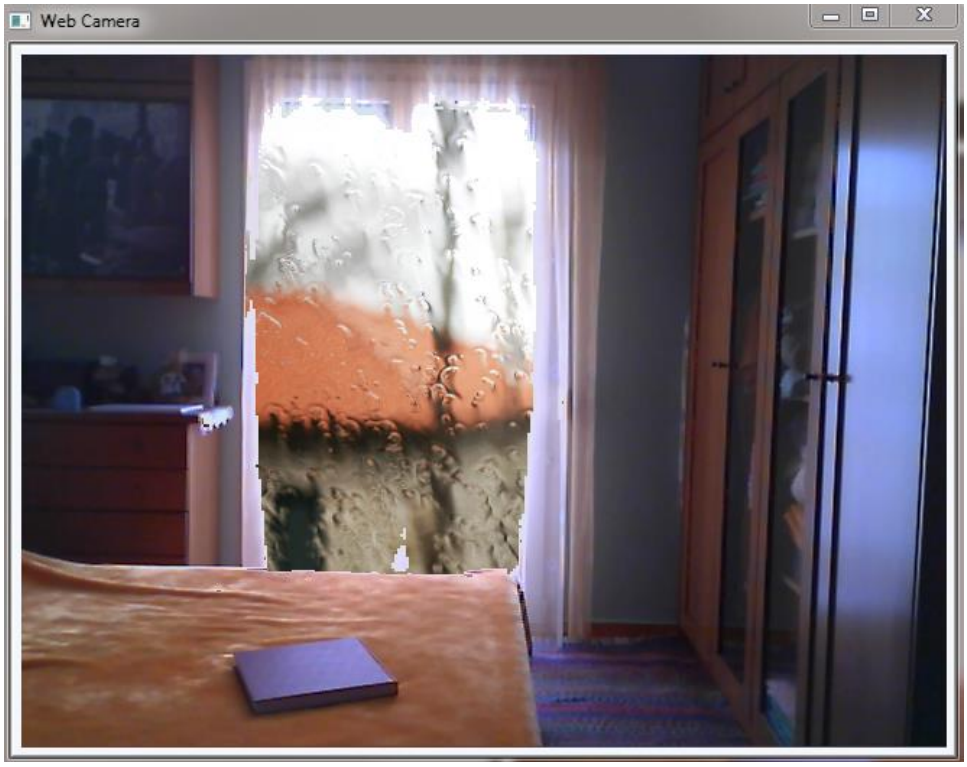
Οποιαδήποτε στιγμή, ο χρήστης μπορεί να αυξομειώσει το RGB range και την ανάλυση της web camera μέσω των αντίστοιχων sliders. Με την αύξηση ή τη μείωση του RGB range θα αυξάνονται ή θα μειώνονται αντίστοιχα το σύνολο των pixels που αντικαθίστανται. Με τον τρόπο αυτό μπορεί να υπάρξει έλεγχος της ακρίβειας της αντικατάστασης υποβάθρου που επιτυγχάνεται, ώστε να υπάρξει και το καλύτερο δυνατό αποτέλεσμα. Ταυτόχρονα, με την αύξηση ή τη μείωση της ανάλυσης μπορεί η εφαρμογή να προσαρμοστεί σε πιθανώς πιο αδύναμα συστήματα. Η διαδικασία της αντικατάστασης είναι σχετικά «βαριά,» δεδομένου ότι σαρώνονται ένα προς ένα όλα τα pixels τόσο της εικόνας της web camera, όσο και του νέου υποβάθρου. Στην περίπτωση που η ανάλυση έχει τεθεί στα 640x480, σαρώνονται 307200 pixels, τόσο για την web camera, όσο και για το νέο υπόβαθρο, συνολικά δηλαδή 614400 pixels ανά frame της web camera, δηλαδή περίπου 18432000 pixels το δευτερόλεπτο, με όλες τις συνθήκες ελέγχου και τις λοιπές διαδικασίες που εκτελούνται ταυτόχρονα. Με τη μείωση της ανάλυσης, μειώνεται αισθητά ο αριθμός των ελέγχων αυτών, οπότε και αυξάνεται η επίδοση του προγράμματος στο εκάστοτε σύστημα. Εξάλλου, βασικός προσανατολισμός της εφαρμογής είναι η επίτευξη αντικατάστασης υποβάθρου σε πραγματικό χρόνο.

Εδώ να αναφερθεί ότι με το άνοιγμα μίας εικόνας ή ενός video, προσαρμόζονται στην ανάλυση που έχει οριστεί για την web camera. Αν, για παράδειγμα, ανοιχτεί μία εικόνα που έχει ανάλυση 1024x768, θα μετατραπεί αμέσως στην τρέχουσα ανάλυση της web camera (πχ σε 640x480). Το ίδιο ισχύει και για το video, η ανάλυσή του προσαρμόζεται πάντα στην ανάλυση της web camera.

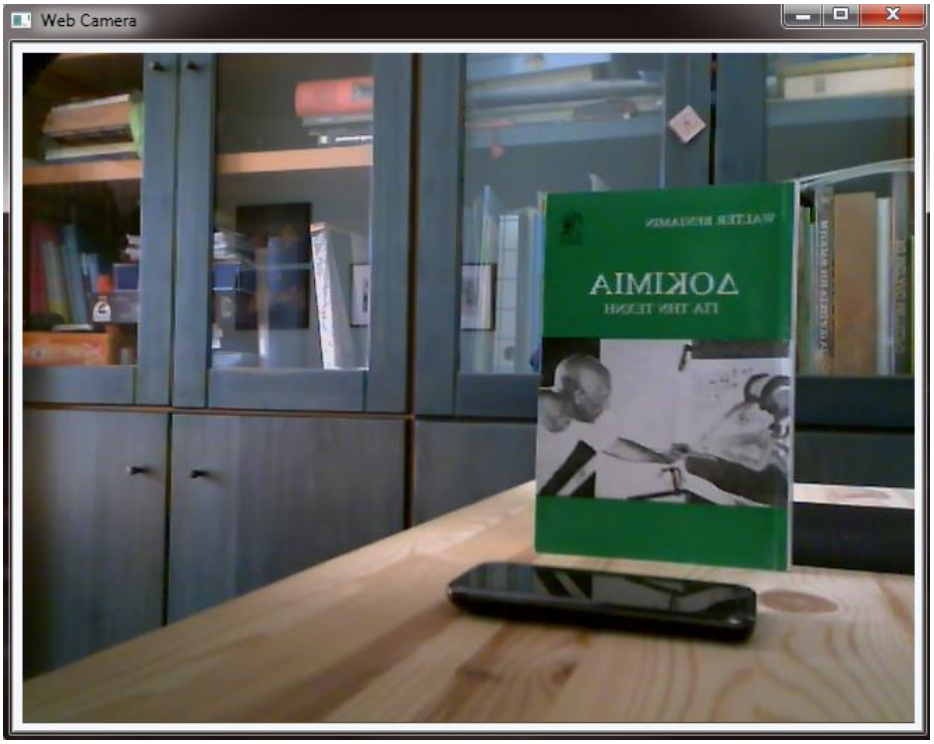
Συμπληρωματικά στο κεντρικό παράθυρο της εφαρμογής έχουν συμπεριληφθεί και τέσσερα text boxes που περιέχουν τις συντεταγμένες του πρώτου και του τελευταίου pixel που επιλέχθηκε από το χρήστη. Είναι καθαρά ενημερωτική η χρήση τους και συμμετέχουν κάπως στην όλη διαδικασία. Επίσης, υπάρχει δυνατότητα αλλαγής του υποβάθρου κατά τη διάρκεια της αντικατάστασης. Τέλος, να αναφερθεί ότι για το κλείσιμο του παραθύρου της web camera πρέπει να πατηθεί το πλήκτρο Esc (escape).



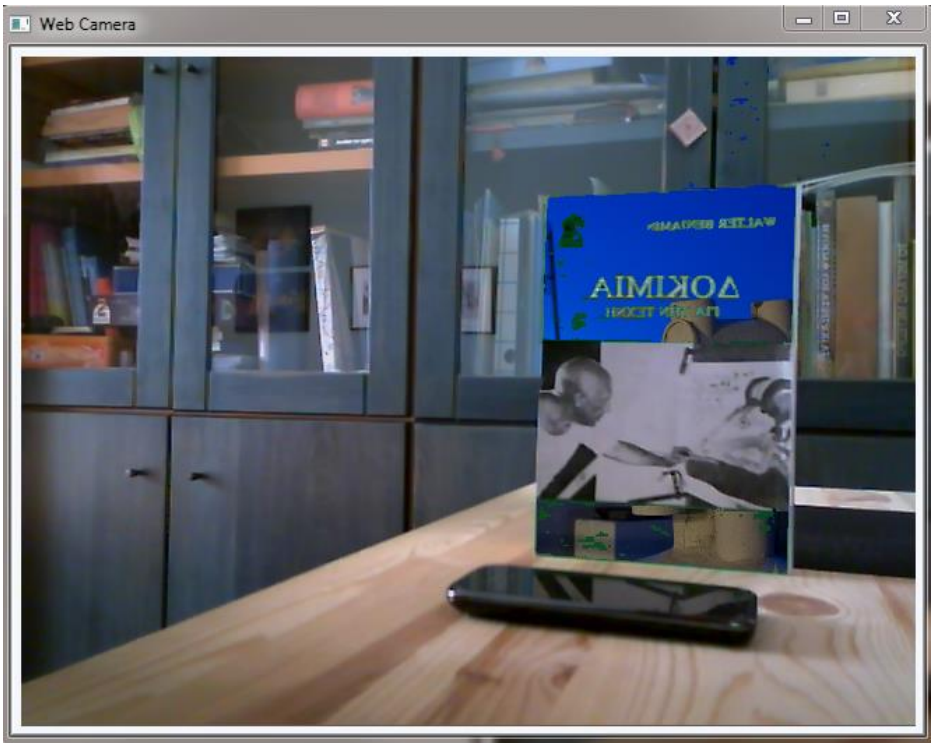
Εικόνα 4:2 Πριν την αντικατάσταση



Εικόνα 4:3 Μετά την αντικατάσταση



Εικόνα 4:4 Πριν την αντικατάσταση



Εικόνα 4:5 Μετά την αντικατάσταση

4.5 Στάδια ανάπτυξης της εφαρμογής

Η εφαρμογή πέρασε από διάφορα στάδια ανάπτυξης για να καταλήξει στην τελική της μορφή. Υπήρξαν ιδέες οι οποίες απορρίφθηκαν και αλγόριθμοι σύγκρισης και αντικατάστασης pixels που έδωσαν τη θέση τους σε άλλους. Η ενότητα αυτή λειτουργεί σαν ένα ιστορικό της εφαρμογής, όπου καταγράφονται οι διάφορες φάσεις της ανάπτυξης και οι αλλαγές που έγιναν κατά τη διάρκειά της.

1. Υλοποίηση της τεχνικής αφαίρεσης και αντικατάστασης υποβάθρου σε κονσόλα: Στη φάση αυτή η εφαρμογή έτρεχε σε κονσόλα, χωρίς να υπάρχει κάποιο γραφικό περιβάλλον. Η όποια εντολή δινόταν από το πληκτρολόγιο, πέραν της περίπτωσης που άνοιγε η εικόνα της web camera και ο χρήστης έπρεπε να επιλέξει ένα pixel για την εύρεση των RGB τιμών αυτού. Στο στάδιο αυτό δε γινόταν υπολογισμός ενός εύρους, βάσει του οποίου θα γινόταν η αντικατάσταση. Αντιθέτως, σαρώνονταν όλα τα pixels της κάμερας και όποια είχαν ακριβώς την ίδια RGB τιμή με το pixel αυτό, αντικαθίσταντο από τα αντίστοιχα pixels μίας άλλης εικόνας. Επίσης, η εικόνα που αποτελούσε το νέο υπόβαθρο ήταν προκαθορισμένη και δεν μπορούσε να αλλάξει, ενώ η ανάλυσή της έπρεπε να ταιριάζει ακριβώς στην ανάλυση της κάμερας για να λειτουργεί σωστά η αντικατάσταση.
2. Προσθήκη δυνατότητας επιλογής πολλών pixels για τον υπολογισμό ενός εύρους RGB για την αντικατάσταση: Το πρώτο πράγμα που προστέθηκε σε αυτό το στάδιο ήταν η δυνατότητα επιλογής πολλών pixels για τον υπολογισμό ενός εύρους για την αντικατάσταση. Στο πρώτο στάδιο ο χρήστης μπορούσε να επιλέξει μόνο ένα pixel, εδώ όμως καθορίζονται δύο σημεία (το ένα με το push down mouse event και το δεύτερο με το push up), τα οποία δημιουργούν ένα νοητό τετράγωνο επιλογής. Αμέσως μετά υπολογίζεται το ελάχιστο και το μέγιστο R,G,B όλων των pixels που βρίσκονται σε αυτό το τετράγωνο επιλογής και πραγματοποιείται η λειτουργία της αντικατάστασης βάσει αυτών των min και max R,G,B τιμών. Σαρώνεται όλη η εικόνας της web camera και όποιο pixel βρίσκεται μέσα στο εύρος αυτό, αντικαθίσταται με το αντίστοιχο μίας άλλης εικόνας.
3. Αλλαγή του τρόπου υπολογισμού του εύρους: Η παραπάνω τακτική οδηγούσε σε αντικατάσταση πάρα πολλών pixels, που πολλές φορές δεν είχαν καν συγγενικές τιμές. Για το λόγο αυτό άλλαξε ο τρόπος υπολογισμού του εύρους, ώστε να περιοριστεί το εύρος. Πλέον δεν κρατούνται οι ελάχιστες και μέγιστες R,G,B τιμές των επιλεγμένων pixels, αλλά υπολογίζεται ο μέσος όρος του R, του G και του B του συνόλου των pixels που επιλέχθηκαν και στους μέσους όρους προσθαφαιρείται ένας αριθμός. Μετά από τα παραπάνω, σαρώνεται όλη η εικόνα και τα pixels που έχουν RGB που ανήκει μέσα στο εύρος που δημιουργήθηκε με την παραπάνω μέθοδο, αντικαθίσταντο με τα αντίστοιχα pixels μίας άλλης εικόνας.
4. Δημιουργία γραφικού περιβάλλοντος και προσαρμογή όλων των παραπάνω σε αυτό: Έχοντας τεθεί οι βασικές λειτουργίες της εφαρμογές, δημιουργήθηκε ένα γραφικό περιβάλλον με το οποίο θα εκτελούνταν όλα όσα ήταν ήδη υλοποιημένα. Με το πάτημα ενός κουμπιού ξεκινούσε η προβολή της εικόνας από την web camera. Η διαδικασία της αντικατάστασης και ο υπολογισμός του εύρους γίνονται με τον ίδιο ακριβώς τρόπο.

5. Προσθήκη δυνατότητας επιλογής εικόνας που θα αποτελέσει το νέο υπόβαθρο: Δημιουργήθηκε ένα κουμπί, με το πάτημα του οποίου ανοίγει ένα παράθυρο διαλόγου για την εξερεύνηση και το άνοιγμα μίας εικόνας που θα αποτελέσει το νέο υπόβαθρο. Με τον τρόπο αυτό ο χρήστης, πλέον, μπορεί να επιλέγει το νέο υπόβαθρο, βασική λειτουργία που έλειπε από τα παλιότερα στάδια.
6. Προσθήκη sliders για τον καθορισμό της ανάλυσης της web camera και της τιμής που θα προσθαφαιρείται στους μέσους όρους των τιμών R, G και B: Με την προσθήκη αυτών των sliders ο χρήστης έχει τη δυνατότητα να προσαρμόζει την ανάλυση της web camera, κάτι που βοηθάει πιο αδύναμα συστήματα στο να πραγματοποιήσουν τη διαδικασία της αντικατάστασης σε πραγματικό χρόνο. Ταυτόχρονα, με τη δυνατότητα καθορισμού της τιμής που προσθαφαιρείται στους μέσους όρους των R, G και B, μπορεί να επιτευχθεί η επιθυμητή ακρίβεια. Στο στάδιο αυτό, η εικόνα που επιλέγεται για υπόβαθρο προσαρμόζεται αυτομάτως στην ανάλυση της web camera, ώστε να μη δημιουργούνται λάθη κατά τη διαδικασία της αντικατάστασης.
7. Προσθήκη δυνατότητας επιλογής video για νέο υπόβαθρο: Ως τώρα το νέο υπόβαθρο μπορούσε να είναι μόνο εικόνα. Στο στάδιο αυτό προστέθηκε ένα combo box, στο οποίο ο χρήστης επέλεγε ποιος θα είναι ο τύπος του νέου υποβάθρου. Ανάλογα με την επιλογή που έκανε (image ή video), το κουμπί επιλογής του νέου υποβάθρου προσάρμοζε τη λειτουργία του και γινόταν η ανάλογη αναζήτηση. Οι δυνατότητες που έχει ο χρήστης παραμένουν οι ίδιες, δηλαδή μπορεί να αυξομειώνει την ανάλυση της web camera (και, κατ᾽ επέκταση, του νέου υποβάθρου, ανεξαρτήτως τύπου) και να καθορίσει την τιμή που προσθαφαιρείται στους μέσους όρους των R, G και B.
8. Ολοκλήρωση της εφαρμογής, επίλυση τυχών bugs, προσθήκη συμπληρωματικών λειτουργιών και λοιπές διαδικασίες βελτιστοποίησης: Στο στάδιο αυτό ολοκληρώθηκε η εφαρμογή, διορθώνοντας τα διάφορα bugs που εκκρεμούσαν και βελτιστοποιώντας τον κώδικα.

5 Συμπεράσματα

Με τη χρήση της ψηφιακής τεχνολογίας, η αφαίρεση του υποβάθρου σε ένα video είναι δυνατή με αρκετά καλά και ακριβή αποτελέσματα. Με την ένα προς ένα σάρωση όλων των pixels της εικόνας του video, αντικαθίσταται οποιοδήποτε pixel βρίσκεται μέσα στο καθορισμένο εύρος RGB. Αυτό οδηγεί σε μία πιθανή αντικατάσταση κάποιων pixels που αντιστοιχούν στη φιγούρα ενός κινούμενου αντικείμενου (πχ ενός ανθρώπου) που βρίσκεται μέσα στο κάδρο, κάτι που προφανώς αποτελεί μειονέκτημα, δεδομένου ότι με την αναγνώριση προτύπων ή κίνησης θα μπορούσε να αποφευχθεί η οποιαδήποτε λανθασμένη αντικατάσταση. Η ύπαρξη, βέβαια, των απαραίτητων ελέγχων για την προαναφερθείσα αναγνώριση θα προσέθετε παραπάνω πολυπλοκότητα και ίσως οδηγούσε σε μία απαιτητικότερη εφαρμογή. Αυτό που επιτυγχάνεται είναι μία πλήρως λειτουργική αφαίρεση υποβάθρου, τα αποτελέσματα της οποίας είναι ιδιαίτερα ικανοποιητικά, ειδικά αν κρατηθούν κάποιες τυπικές συμβάσεις που συνήθως ακολουθούνται σε τέτοιες τεχνικές. Συνήθως, οι συμβάσεις αυτές αφορούν στην επιλογή συγκεκριμένων χρωμάτων για υπόβαθρο (δηλαδή χρωμάτων που δε χρησιμοποιούνται τόσο πολύ και η RGB τιμή τους δε «συγκρούεται» με άλλες τιμές, όπως αναφέρεται και στο Κεφάλαιο 2) και στην προσαρμογή της ενδυμασίας στο χρώμα του υποβάθρου, ώστε να μην ταυτίζεται με αυτό. Η επιλογή ενός μαύρου υποβάθρου, για παράδειγμα, είναι σχεδόν αναπόφευκτο να οδηγήσει σε λανθασμένη αντικατάσταση, γιατί γειτονικές RGB τιμές του απόλυτου μαύρου (0,0,0) είναι ιδιαίτερα συχνές. Ταυτόχρονα, η δυνατότητα καθορισμού ενός αριθμού που προσθαφαιρείται στο μέσο όρο των τιμών RGB των pixels που επιλέγονται από το χρήστη οδηγεί σε ακόμα καλύτερη ακρίβεια, ανάλογα με τις απαιτήσεις και τις ανάγκες που ορίζονται από τις παρούσες συνθήκες (φωτισμός, χρώματα κτλ).

Εδώ πρέπει να αναφερθεί ένα ακόμα σημαντικό πλεονέκτημα της εφαρμογής. Όντας γραμμένη σε MFC, με μοναδική απαίτηση μία web camera, ο οποιοσδήποτε χρήστης μπορεί να την εγκαταστήσει στο σύστημά του (με λειτουργικό Windows) και να την τρέξει, χωρίς να αντιμετωπίσει κάποιο πρόβλημα. Η επίτευξη μίας τέτοιας διαδικασίας με τη χρήση ενός τόσο καθημερινού και φτηνού υλικού είναι κάτι ιδιαίτερα θετικό και με τη μεταφορά σε άλλα λειτουργικά συστήματα (όπως Linux και MacOS), πράγμα σχετικά εύκολο, δεδομένου ότι υπάρχουν εκδόσεις της OpenCV για αυτά, θα μπορούσε να αποτελέσει βάση για πιο σύνθετα προγράμματα.

6 Μελλοντικά Σχέδια και ιδέες

Οι εφαρμογές που έχουν ως αντικείμενο την Computer Vision είναι ανοικτές σε πολλές βελτιώσεις και προσθήκες, δεδομένης της φύσης του αντικειμένου και της πληθώρας νέων ιδεών και εξελίξεων στο χώρο. Η παρούσα εφαρμογή αναπτύχθηκε για την κάλυψη των βασικών λειτουργιών αφαίρεσης και αντικατάστασης υποβάθρου, αφήνοντας έτσι ένα σημαντικό περιθώριο για περαιτέρω προσθήκες, τόσο στη λειτουργία της, όσο και στην απόδοσή της. Στην ενότητα αυτή εξετάζεται το πλήθος των προσθηκών αυτών, χωριζόμενες σε δύο κατηγορίες: τις βελτιώσεις που μπορούν να γίνουν σε προγραμματιστικό επίπεδο και στο γραφικό περιβάλλον, για την αποδοτικότερη εκτέλεσή της και την καλύτερη δυνατή επικοινωνία του χρήστη με αυτήν αντίστοιχα, καθώς και στις προσθήκες νέων λειτουργιών, για τον εμπλουτισμό του τι μπορεί να επιτευχθεί μέσω αυτής.

6.1 Βελτιώσεις

1. Χρήση multithreading. Η εφαρμογή είναι έτσι υλοποιημένη, ώστε το σύνολο των λειτουργιών της να εκτελείται σειριακά. Αυτό προϋποθέτει ότι θα υπάρχει μία πληθώρα συνθηκών ελέγχου, για τη σωστή μετάβαση από τη μία διαδικασία στην άλλη, καθώς και την ολοκλήρωση της μίας, για την έναρξη της άλλης. Επόμενο είναι λοιπόν να υπάρχει ο παράγοντας της καθυστέρησης για την προαναφερθείσα μετάβαση και σε ορισμένες περιπτώσεις γίνεται ιδιαίτερα αισθητή, όπως όταν η διαδικασία της αφαίρεσης και αντικατάστασης μπει σε μία κατάσταση «αδράνειας,» έως ότου κάποια άλλη διαδικασία (πχ αλλαγής αναλύσεως) ολοκληρωθεί. Με τη χρήση multithreading, τέτοια προβλήματα θα λύνονταν αρκετά εύκολα, κάνοντας την εφαρμογή αποδοτικότερη, αλλά και λογικά αρτιότερη. Ταυτόχρονα θα μειώνονταν οι εξαρτήσεις μεταξύ των διαφόρων διαδικασιών και του συνόλου των συνθηκών ελέγχου.
2. Προσθήκη αναγνώρισης προτύπων και κίνησης. Η δυνατότητα αναγνώρισης προτύπων και κίνησης θα έκανε πιο ευέλικτη την αφαίρεση υποβάθρου, δεδομένου ότι η εφαρμογή θα μπορούσε να «καταλάβει» ποια αντικείμενα βρίσκονται σε κίνηση και ποια όχι, όπως και να αναγνωρίσει σχήματα και πρότυπα, αποφεύγοντας την όποια αντικατάσταση pixels που βρίσκονται μέσα στην περιφέρειά αυτών. Τέτοιοι αλγόριθμοι είναι αρκετά πολύπλοκοι και απαιτητικοί.
3. Ολοκλήρωση της εφαρμογής σε άλλα λειτουργικά συστήματα. Η εφαρμογή αναπτύχθηκε σε MFC, οπότε μπορεί να εγκατασταθεί και να τρέξει μόνο σε περιβάλλον Windows. Μία σημαντική βελτίωση θα ήταν η μεταφορά του σε άλλα περιβάλλοντα, όπως σε Linux ή MacOS, ώστε να μπορούν χρήστες αυτών των λειτουργικών συστημάτων να την τρέξουν. Αυτό προϋποθέτει βέβαια το να υπάρχει έκδοση της OpenCV για το εκάστοτε λειτουργικό σύστημα.

6.2 Προσθήκες

1. Επιλογή μεταξύ διαφορετικών χρωματικών μοντέλων.

2. Προσθήκη δυνατότητας αποθήκευσης του παραγόμενου video με το νέο υπόβαθρο. Το αποτέλεσμα της εφαρμογής είναι η προβολή, σε πραγματικό χρόνο, της εικόνας που «βλέπει» η web camera του χρήστη με ένα νέο υπόβαθρο, όμως δεν υπάρχει κάποιο παράγωγο. Ιδανική προσθήκη θα ήταν η δυνατότητα αποθήκευσης του αποτελέσματος της αντικατάστασης σε ένα αρχείο video στον υπολογιστή.
3. Η ταυτόχρονη υποστήριξη πολλών web cameras ταυτόχρονα, η δυναμική επεξεργασία εικόνων με απλούς ελέγχους, η διαχείριση του συνόλου των frames διαφόρων videos κτλ είναι διαδικασίες που υποστηρίζονται. Ο πειραματισμός και η προσθήκη νέων τεχνικών αντικαταστάσεις περιορίζεται μόνο από τη φαντασία του εκάστοτε προγραμματιστή. Μερικές απλές στην υλοποίηση, με ενδιαφέρον όμως αποτέλεσμα, τεχνικές αφαίρεσης και αντικατάστασης υποβάθρου είναι:
    a. Χρήση μίας δεύτερης web camera, η εικόνα της οποίας θα λειτουργεί ως το υπόβαθρο που θα αντικαθιστά το υπόβαθρο της εικόνας που «βλέπει» η πρώτη web camera.
    b. Αντικατάσταση πολλαπλών υποβάθρων στην ίδια web camera. Αν το υπόβαθρο μίας σκηνής αποτελείται από δύο διακριτά χρώματα (πχ ένα πράσινο και ένα μπλε), θα μπορούσε να αναπτυχθεί μία μέθοδος για την αντικατάσταση του ενός με μία εικόνα και του άλλου με κάποια άλλη.
    c. Κάτι που θα προσέθετε αρκετό ενδιαφέρον θα ήταν, ταυτόχρονα με την αντικατάσταση του υποβάθρου, η αντικατάσταση των pixels που δεν ανήκουν σε αυτό με κάποια άλλη εικόνα ή βίντεο, ανάλογα με το αποτέλεσμα που επιδιώκει ο καθένας.
4. Προσθήκη διαφορετικών τεχνικών αντικατάστασης υποβάθρου και δυνατότητα επιλογής. Η τεχνική που ακολουθήθηκε σαρώνει ολόκληρο το τρέχον frame της web camera και το νέο υπόβαθρο και κάνει τη σχετικά αντικατάσταση. Σε περίπτωση που προστεθεί δυνατότητα αναγνώρισης κίνησης και προτύπων, θα μπορούσε να αλλάξει τελείως ο τρόπος που πραγματοποιείται η αντικατάσταση, δεδομένου ότι από τη στιγμή που η εφαρμογή «γνωρίζει» ποια αντικείμενα δεν ανήκουν στο υπόβαθρο, θα μπορούσε αυτομάτως να αντικαταστήσει τα pixels εκτός των αντικειμένων αυτών με τα αντίστοιχα pixels μίας άλλης εικόνας ή video.

7 Βιβλιογραφία

Bradski και άλλοι. (2008). Learning OpenCV. O' Reilly, pp. 1-10

Δημητριάδης και άλλοι. (2004). Τεχνολογία Πολυμέσων Θεωρία και Πράξη. Τζιόλα, pp. 152-175

Trucco και άλλοι. (1998). Introduction Techniques for 3-D Computer Vision. Prentice Hall, pp. 1-5

Rickitt, Richard. (2007). Special Effects – The History and Techniques. Billboard Books, pp. 44-54

Βλαχάβας και άλλοι (2006). Τεχνητή Νοημοσύνη. Β. Γκιούρδας Εκδοτική, pp. 706-709.

Hoffmann, G. (2006). CIE Color Space, http://www.fho-emden.de/~hoffmann.

Hiroki και άλλοι. (2007). Chroma Key Using a Checker Pattern Background. IEICE TRANS. INF. & SYST., VOL.E90–D, NO.1, pp. 1-6.

David Yamnitsky. (2009). Real-Time Chroma Keying on the GPI. Boris FX, pp. 2-4.

Hugh D. Young. (1992). Πανεπιστημιακή Φυσική Τόμος Β. Εκδόσεις Παπαζήση, pp. 944-947.

Παραρτήματα

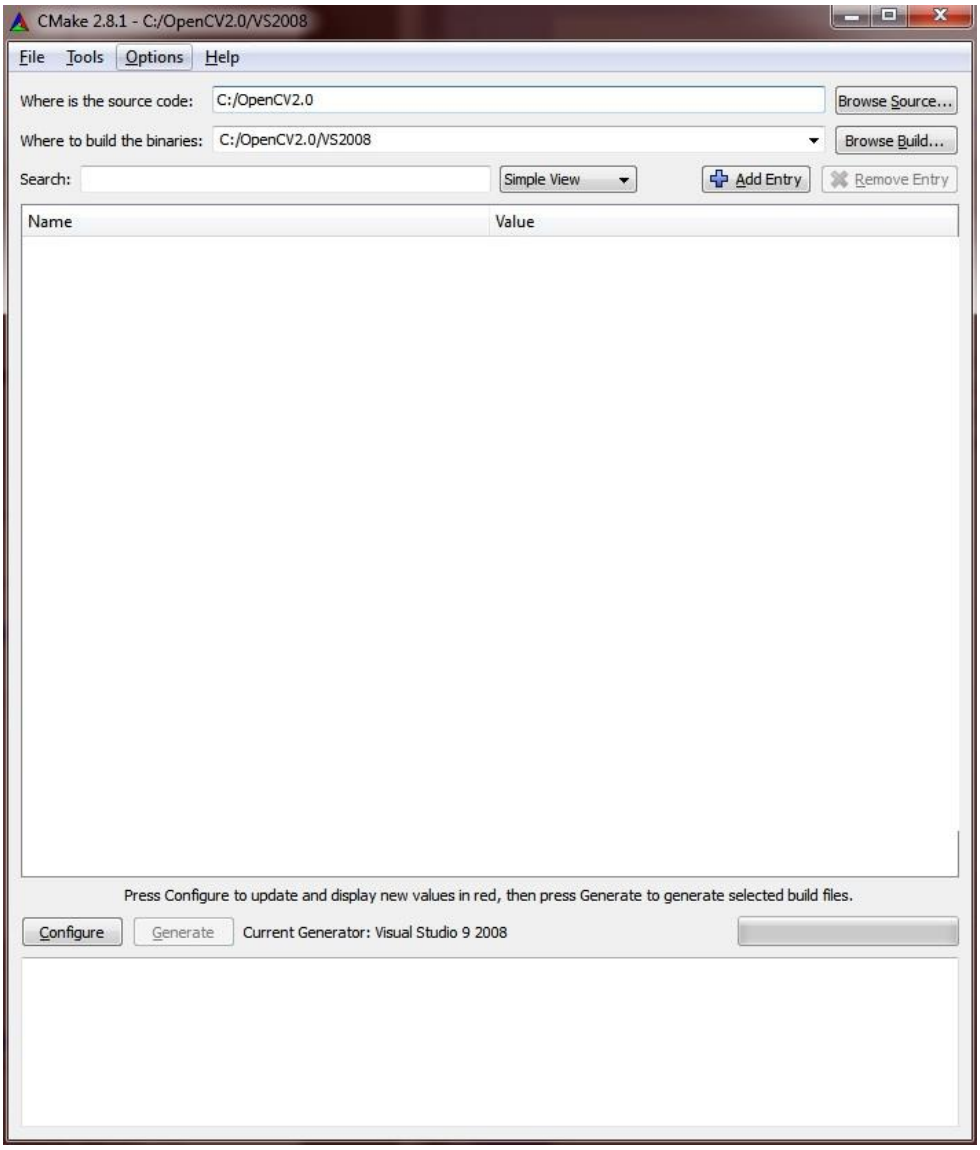
Παράρτημα 1: Εγκατάσταση των εργαλείων

Για να μπορέσει κάποιος να χρησιμοποιήσει τα εργαλεία της OpenCV προς όφελος του, μέσω του Visual Studio 2008, πρέπει πρώτα να μετατρέψει τον κώδικα της βιβλιοθήκης σε project του Visual Studio 2008 και μετά να το κάνει compile ώστε να παραχθούν τα απαιτούμενα DLLs. Για να γίνει αυτό, χρειάζονται τα παρακάτω εργαλεία και αρχεία:

- CMake 2.8.2 (http://www.cmake.org/cmake/resources/software.html)
- OpenCV 2.1 (http://sourceforge.net/projects/opencvlibrary/files/opencv-win/2.1/)
- Microsoft Visual Studio 2008 (Professional ή Express Edition)

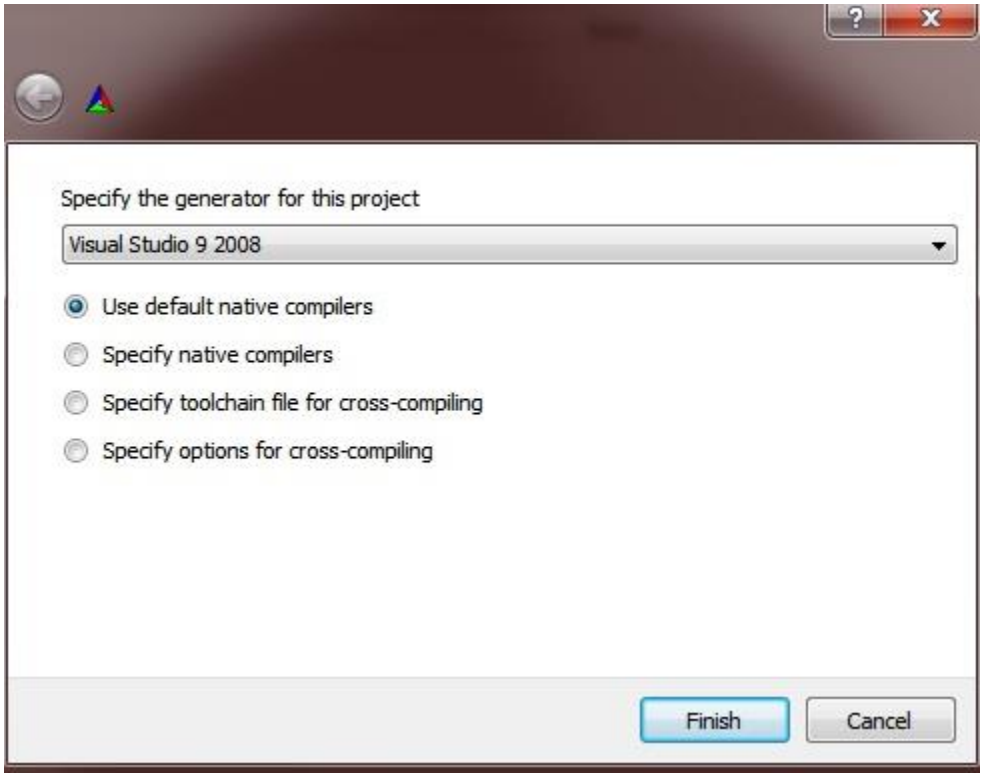
Αφού κατεβαστούν τα παραπάνω αρχεία, πρέπει να εγκατασταθούν στον υπολογιστή του χρήστη. Η σειρά με την οποία θα εγκατασταθούν δεν παίζει κάποιο ρόλο, είναι όμως απαραίτητο στην εγκατάσταση της OpenCV 2.1 να επιλεγεί το «Add OpenCV to the System PATH for all users» ή «Add OpenCV to the System PATH for current user,» όταν αυτό ερωτηθεί.

Στη συνέχεια ο χρήστης πρέπει να ανοίξει το CMake και συγκεκριμένα το cmake-gui (το γραφικό περιβάλλον του CMake). Εδώ πέρα πρέπει να οριστεί από πού θα διαβάσει το CMake τον πηγαίο κώδικα (source code) και που να αποθηκευτούν τα binaries που θα δημιουργηθούν.



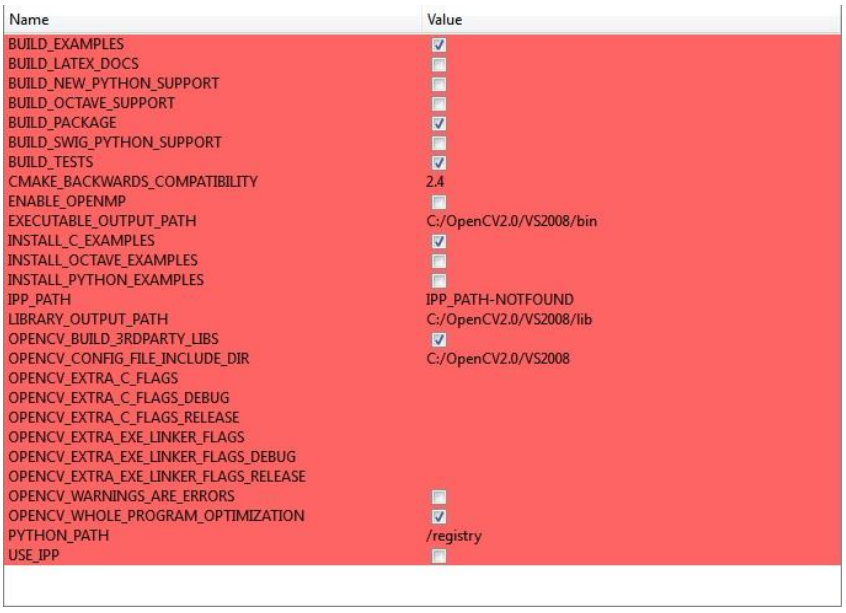
Εικόνα 0:1 Κεντρικό παράθυρο του CMake

Στο πρώτο πεδίο («Where is the source code») ορίζεται ο φάκελος που έχει εγκατασταθεί η OpenCV , ενώ στο δεύτερο πεδίο («Where to build the binaries») καθορίζεται που θα αποθηκευτούν τα binaries. Στο παραπάνω παράδειγμα, τα binaries αποθηκεύονται στο φάκελο C:\OpenCV2.0\VS2008. Αφού καθοριστούν αυτά, γίνονται ο χρήστης επιλέγει το IDE και τους compilers που θέλει να χρησιμοποιήσει, πατώντας το κουμπί «Configure.»



Εικόνα 0:2 Configure παράθυρο του CMake

Στο παράθυρο αυτό πρέπει να επιλεγεί το Visual Studio 9 2008 (το IDE που χρησιμοποιήθηκε για την ανάπτυξη της εφαρμογής). Αφού ο χρήστης πατήσει το κουμπί «Finish,» στο αρχικό παράθυρο θα εμφανιστεί μία λίστα επιλογών σε κόκκινο πλαίσιο.



Εικόνα 0:3 Απαραίτητες επιλογές για τα binary αρχεία που θα δημιουργηθούν

Τα πεδία που πρέπει να επιλεγούν είναι τα παρακάτω:

- BUILD_EXAMPLES
- BUILD_PACKAGES
- BUILD_TEST
- INSTALL_C_EXAMPLES
- OPENCV_BUILD_3RDPARTY_LIBS
- OPENCV_WHOLE
- PROGRAM_OPTIMIZATION

Τέλος, το απαιτούμενο Visual Studio Project δημιουργείται με το πάτημα του κουμπιού «Generate.» Με το πέρας της διαδικασίας, το CMake δεν είναι πλέον απαραίτητο και μπορεί να κλείσει.

Το επόμενο βήμα είναι η φόρτωση του project που δημιουργήθηκε με το Visual Studio 2008. Για τη φόρτωση του project πρέπει να επιλεχθεί το OpenCV.sln (βρίσκεται στο φάκελο που ορίστηκε για την αποθήκευση των binaries, στο παράδειγμα ο φάκελος αυτός είναι ο C:\OpenCV2.0\VS2008). Στη συνέχεια γίνεται build το Project και σε Debug και σε Release. Αφού ολοκληρωθούν και τα δύο builds χωρίς errors (τα warnings δεν παίζουν ρόλο), πρέπει να ελεγχθεί αν στο φάκελο των Debug και Release (με βάση το παράδειγμα, C:\OpenCV2.0\VS2008\lib\Debug και C:\OpenCV2.0\VS2008\lib\Release) έχουν δημιουργηθεί τα απαραίτητα .lib (cv200d, cvaux200d, cxcore200d, cxts200d, highgui200d, ml200d, opencv_ffmpeg200d).

Στη συνέχεια προστίθενται στο system path (My Computer->Properties->Advanced->Environment variable->PATH->edit) τα παρακάτω:

- C:\OpenCV2.0\VS2008\lib\Debug
- C:\OpenCV2.0\VS2008\lib\Release

(και τα δύο είναι με βάση το δοθέν παράδειγμα)

Ένα πολύ βασικό βήμα είναι να ορίσει ο χρήστης που θα βρίσκει το Visual Studion 2008 τις απαραίτητες βιβλιοθήκες και source files, ώστε να αναγνωρίζονται οι δομές και οι μέθοδοι της OpenCV. Ο καθορισμός αυτός γίνεται μέσω των Options του Visual Studio 2008 (Tools->Options->Projects and Solutions-> VC++ Directories). Στο drop down menu «Show directories for:» πρέπει να επιλεγεί το «Include Files» και να εισαχθεί ο φάκελος:

- C:\OpenCV2.0\include\opencv.

Στο ίδιο drop down menu πρέπει να επιλεγεί το «Library files option:» και να εισαχθούν τα:

- C:\OpenCV2.0\VS2008\lib\Debug
- C:\OpenCV2.0\VS2008\lib\Release

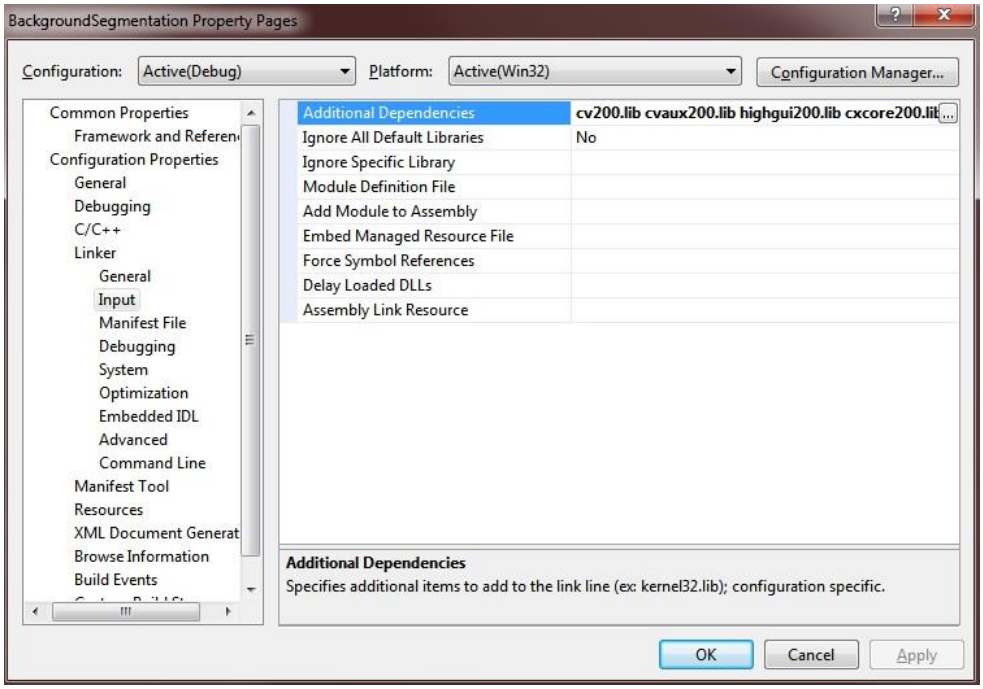
Τέλος, πρέπει να επιλεγεί το «Source files options:» και να εισαχθούν τα:

- C:\OpenCV2.0\VS2008\src\cxcore\cxcore.dir\Debug
- C:\OpenCV2.0\VS2008\src\ml\ml.dir\Debug
- C:\OpenCV2.0\VS2008\src\highgui\highgui.dir\Debug
- C:\OpenCV2.0\src
- C:\OpenCV2.0\VS2008\src\cvaux\cvaux.dir\Debug
- C:\OpenCV2.0\VS2008\src\cv\cv.dir\Release
- C:\OpenCV2.0\VS2008\src\cv\cv.dir\Debug

(όλες οι παραπάνω διευθύνσεις βασίζονται στο παράδειγμα)

Στο σημείο αυτό, το περιβάλλον είναι έτοιμο για την ανάπτυξη εφαρμογών βασισμένες στην OpenCV. Σε κάθε νέο project που δημιουργεί ο προγραμματιστής πρέπει να πηγαίνει στο Project-> "Project name" Properties-> Configuration Properties-> Linker και στο πεδίο «Additional Dependencies» ναι προσθέτει τα παρακάτω:

- cv200.lib cvaux200.lib highgui200.lib cxcore200.lib ml200.lib



Εικόνα 0:4 Καθορισμός των Additional Dependencies σε ένα νέο Visual Studio Project

Φυσικά η εφαρμογή που αναπτύχθηκε έχει ως απαραίτητη προϋπόθεση την ύπαρξη μίας web camera, η οποία χρησιμοποιείται για την είσοδο του video που υπόκειται στη διαδικασία της αντικατάστασης του υποβάθρου. Είναι και η μοναδική απαίτηση από πλευράς hardware.

Παράρτημα 2: Κώδικας

```
// BackgroundSegmentation.cpp : Καθορίζεται η συμπεριφορά των classes της εφαρμογής.

#include "stdafx.h"
#include "BackgroundSegmentation.h"
#include "BackgroundSegmentationDlg.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#endif

// CBackgroundSegmentationApp

BEGIN_MESSAGE_MAP(CBackgroundSegmentationApp, CWinApp)
	ON_COMMAND(ID_HELP, &CWinApp::OnHelp)
END_MESSAGE_MAP()
```

```
// CBackgroundSegmentationApp construction

CBackgroundSegmentationApp::CBackgroundSegmentationApp()
{
// TODO: προσθήκη κώδικα για το construction εδώ
// Όλες οι απαραίτητες αρχικοποιήσεις να γίνουν στο InitInstance
}

// Το μοναδικό CBackgroundSegmentationApp object

CBackgroundSegmentationApp theApp;

// CBackgroundSegmentationApp αρχικοποίηση

BOOL CBackgroundSegmentationApp::InitInstance()
{
// Το InitCommonControlsEx() είναι απαραίτητο σε Windows XP, αν η δήλωση της εφαρμογή
// υποδηλώνει τη χρήση του ComCtl32.dll version 6 ή μετέπειτα για την ενεργοποίηση των
// visual styles.  Αλλιώς, οποιαδήποτε δημιουργία windows θα αποτύχει.
	INITCOMMONCONTROLSEX InitCtrls;
	InitCtrls.dwSize = sizeof(InitCtrls);
// Αυτό χρησιμοποιείται για να γίνουν inclune όλα τα common control classes που θα
// χρησιμοποιηθούν στην εφαρμογή
	InitCtrls.dwICC = ICC_WIN95_CLASSES;
	InitCommonControlsEx(&InitCtrls);

	CWinApp::InitInstance();

	AfxEnableControlContainer();

// Βασική αρχικοποίηση
// Αν δε χρησιμοποιούνται αυτά τα χαρακτηριστικά και υπάρχει η διάθεση για μείωση του
// εκτελέσιμου κομματιού του τελικού executable, τότε πρέπει να αφαιρεθούν από τα ακόλουθα
// συγκεκριμένες ρουτίνες αρχικοποίησης που δε χρειάζονται
// Πρέπει να αλλαχθεί το registry key όπου αποθηκεύοντιαι οι ρυθμίσεις
	SetRegistryKey(_T("Local AppWizard-Generated Applications"));

	CBackgroundSegmentationDlg dlg;
	m_pMainWnd = &dlg;
	INT_PTR nResponse = dlg.DoModal();
	if (nResponse == IDOK)
	{
```

```
// TODO: Εδώ πρέπει να τοποθετηθεί κώδικας όταν πατιέται το OK
	}
	else if (nResponse == IDCANCEL)
	{
// TODO: Εδώ πρέπει να τοποθετηθεί κώδικας όταν πατιέται το Cancel
	}

// Από τη στιγμή που κλείνει ο διάλογος, πρέπει να γίνει return FALSE ώστε να υπάρξει έξοδος από την εφαρμογή
	return FALSE;
}

// BackgroundSegmentationDlg.h : header file

#pragma once
#include "afxwin.h"
#include "Resource.h"
#include "afxcmn.h"
#include "highgui.h"
#include "cv.h"
#include "Utils.h"
#include "Globals.h"

// CBackgroundSegmentationDlg dialog
class CBackgroundSegmentationDlg : public CDialog
{
// Construction
public:
// Βασικός constructor
	CBackgroundSegmentationDlg(CWnd* pParent = NULL);

// Dialog Data
	enum { IDD = IDD_BACKGROUNDSEGMENTATION_DIALOG };

	protected:
// DDX/DDV υποστήριξη
	virtual void DoDataExchange(CDataExchange* pDX);

// Υλοποίηση
protected:
	HICON m_hIcon;

// Παραγώμενες  message map functions
```

```
        virtual BOOL OnInitDialog();
        afx_msg void OnSysCommand(UINT nID, LPARAM lParam);
        afx_msg void OnPaint();
        afx_msg HCURSOR OnQueryDragIcon();
        DECLARE_MESSAGE_MAP()
//Δημιουργούμε μία αναφορά σε ένα CvCapture* αντικείμενο.
        CvCapture* Video;
//Δημιουργία μίας αναφοράς σε ένα CWnd* αντικείμενο. Χρησιμοποιείται για την πρόσβαση σε ορισμένα controls του κεντρικού παραθύρου.
        CWnd* window;
//Το αντικείμενο για τον έλεγχο των tooltips.
        CToolTipCtrl c_toolTip;
//Κανονικό πλάτος του video (της web camera).
        int videoSWidth;
//Κανονικό ύψος του video (της web camera).
        int videoSHeight;
//bool μεταβλητή. Όταν είναι false, γίνεται αντικατάσταση με εικόνα, ενώ όταν είναι true, γίνεται αντικατάσταση με video.
        bool bImgOrVid;  public:
        afx_msg void OnTcnSelchangeTab1(NMHDR *pNMHDR, LRESULT *pResult);
        afx_msg void OnBnClickedbtnCam();
        afx_msg void OnBnClickedbtnshowfile();
        afx_msg void OnBnClickedbtnopenfile();
        afx_msg void OnHScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar);
        afx_msg void OnClose();
        afx_msg void OnCancel();
        afx_msg void OnCbnSelchangecombovidorimg();
        BOOL CBackgroundSegmentationDlg::PreTranslateMessage(MSG* pMsg);
//Control για το Star webcamera button (btnCam).
        CButton c_btnCam;
//Control για το Show Image button (btnShowImg).
        CButton c_btnShowFile;
//Control για το Open Image button (btnOpenImg).
        CButton c_btnOpenFile;
//Control για το slider Resolution (slideRes).
        CSliderCtrl c_slideRes;
//Control για το slider Range (slideRange).
        CSliderCtrl c_slideRange;
//Control για το txtRangeV.
        CEdit c_txtRange;
//Control για το txtInfo.
        CEdit c_txtInfo;
//Control για το comboVidOrImg;
```

```
        CComboBox c_comboVidOrImg;
};

//Globals.h
//Εδώ δηλώνονται κάποιες global μεταβλητές που θα χρησιμοποιηθούν σε διαφορετικά αρχεία του project.

#include "BackgroundSegmentationDlg.h"

/*Κρατάει την εικόνα που επιλέχτηκε για background χωρίς να έχει υποστή κάποιο resize. Η είκονα αυτή χρησιμοποιείται για να γίνεται το resize της πληροφορίας που περιέχει σε μία άλλη, μικρότερη, εικόνα που χρησιμοποιείται για την αντικατάσταση του φόντου της κάμερας. Η νέα, μικρότερη, εικόνα είναι IplImage* αντικείμενο και ονομάζεται gBackgShowImg.*/
extern IplImage* gBackground;
/*Αποτελεί το προσαρμοσμένο background στην ανάλυση της κάμερας. Είναι αυτό που χρησιμοποιείται στην replacement, τόσο για το video, όσο και για την εικόνα (στατικό background).*/
extern IplImage* gBackgShowImg;
/*CvCapture* αντικείμενο για την αποθήκευση του video background που θα επιλεχθεί. Το μέγεθός του παραμένει σταθερό, ένω, κατά τη διάρκεια της αντικατάστασης, παίρνεται το κάθε καρέ του, γίνεται resize στο μέγεθος του IplImage* αντικειμένο gBackgShowImg, το οποίο και αντικαθιστά το τρέχον background της κάμερας.*/
extern CvCapture* gBackCapture;
//Μία global int που κρατάει τον αριθμό που προσθαφαιρείται στα ranges του κάθε χρώματος R,G,B. Καθορίζεται από το slideRange.
extern int gRange;
/*int που χρησιμοποιείται για τον τερματισμό της while που εκτελείται για την αναπαραγωγή του video της web camera. Του δίνεται η τιμή 1 για το ξεκίνημα - while(gI) - ενώ, όταν κλείνει η εφαρμογή του δίνεται η τιμή 0.*/
extern int gI;

//Utils.h
//Εδώ υπάρχουν τα signatures των functions που χρησιμοποιούνται. Περιγραφή του τι κάνουν βρίσκεται στο Utils.cpp

#ifndef ADD_H
#define ADD_H
#include "BackgroundSegmentationDlg.h"

void mouseOnCallback (int event, int x, int y, int flags, void* param);
void playCamera (CvCapture* cam, char* camWindow, int i, bool bIoV, CWnd* mainWindow, int btnCamID, int txtSxID, int txtFxID, int txtSyID, int txtFyID, int comboID);
void mouseOnCallback (int event, int x, int y, int flags, void* param);
```

```
IplImage* replacement (CvPoint sPt, CvPoint fPt, IplImage* camFrame, char* window, CWnd*
mainWindow, int txtSxID, int txtFxID, int txtSyID, int txtFyID);
IplImage* replacement (CvPoint sPt, CvPoint fPt, IplImage* camFrame, IplImage* videoFrame,
char* camWindow, CWnd* mainWindow, int txtSxID, int txtFxID, int txtSyID, int txtFyID);
void colourCalculation (int sX, int sY, int fX, int fY, IplImage* picture);
void appendEditText(CEdit& edit, LPCTSTR pszText);

#endif

/*Utils.cpp
Εδώ υπάρχουν οι βασικές συναρτήσεις που χρησιμοποιούμε για όλες τις απαραίτητες λειτουργίες
(πχ εμφάνιση της εικόνας της κάμερας).
Λίστα των συναρτήσεων και τι κάνει η καθεμία:

playCamera: Παίρνει την εικόνα της κάμερας και την προβάλει σε ένα νέο παράθυρο.
mouseOnCallback: Συνάρτηση που καλείται κάθε φορά που γίνεται κάποιο mouse event.
Χρησιμοποιείται για τη διαχείριση των events που σχετίζονται με την εικόνα του video της web
camera.
replacement: Συνάρτηση που χρησιμοποιείται για την αντικατάσταση των pixels του υποβάθρου
με τα pixels μίας νέας εικόνας/ video.
colourCalculation: Συνάρτηση που υπολογίζει το χρωματικό εύρος του RGB των pixels που
επιλέχθηκαν. Το εύρος αυτό θα καθορίσει τα pixels που θα αντικατασταθούν.
appendEditText: Καλείται κάθε φορά που πρέπει να ενημερωθεί το edit box του κεντρικού
παραθύρου (txtInfo).
*/

#include "stdafx.h"
#include "highgui.h"
#include "cv.h"
#include "Utils.h"
#include "Globals.h"
#include "BackgroundSegmentationDlg.h"

int rangeRed;
//Int μεταβλητές που κρατάνε το range που θα χρησιμοποιηθεί παρακάτω για την εύρεση των
προς αντικατάσταση pixels.
int rangeGreen;
int rangeBlue;
//Χρησιμοποιείται για το κάλεσμα της replacement. Γίνεται true όταν επιλεχθεί κάποιο pixel.
bool bRep = false;
//Δημιουργία ενός CvPoint αντικειμένου που θα περιέχει τα αρχικά (x,y) του τετραγώνου.
CvPoint startPt;
//Δημιουργία ενός CvPoint αντικειμένου που θα περιέχει τα τελικά (x,y) του τετραγώνου.
```

```
CvPoint finishPt;
/*Τα δύο αυτά CvPoint χρησιμοποιούνται από την mouseOnCallback για να καθορίσουν το τετράγωνο των pixels, βάσει των οποίων θα καθοριστεί και το range (για την αντικατάσταση).*/

void playCamera (CvCapture* cam, char* camWindow, int i, bool bImgOrVid, CWnd* mainWindow, int btnCamID, int txtSxID, int txtFxID, int txtSyID, int txtFyID, int comboID)
{
//IplImage* που θα φιλοξενεί το εκάστοτε frame του video. Φορτώνεται το πρώτο frame του video.
	IplImage* frame = cvQueryFrame(cam);
/*Φορτώνεται το επόμενο frame του video, το οποίο ανταποκρίνεται σε αυτό που βλέπει η κάμερα τη στιγμή που πατιέται το "Start Webcamera" button, δεδομένου ότι η κάμερα ανοίγει με το ξεκίνημα της εφαρμογής και το πρώτο της frame αντιστοιχεί στο πρώτο πράγμα που «είδε» η κάμερα.*/
	frame = cvQueryFrame(cam);
//Εμφάνιση του πρώτου frame του video στο ανάλογο παράθυρο.
	cvShowImage(camWindow, frame);
//Βοηθητική int μεταβλητή για τον καθορισμού του ρυθμού ανανέωσης της εικόνας.
	int c;
//Η global gI παίρνει την τιμή 1 (περάστηκε με το κάλεσμα της playCamera, όπου το i πήρε την τιμή 1).
	gI = i;

/*
switch για το διαχωρισμό της λειτουργίας της κάμερας και της replacement που θα χρησιμοποιηθεί, ανάλογα με το αν γίνεται αντικατάσταση με video ή με εικόνα. Σε περίπτωση που η bImgOrVid είναι false, η αντικατάσταση γίνεται με εικόνα, ενώ αν είναι true γίνεται με video. Κατά αντιστοιχία, καλείται η ανάλογη replacement.
*/

	switch (bImgOrVid)
	{
//Αντικατάσταση με εικόνα.
		case false:
			{
//Όσο ο έλεγχος στη while είναι αληθής, θα προβάλλονται νέα frames από την web camera.
				while(gI)
				{
//Ορίζεται η συνάρτηση που καλείται όταν γίνει κάποιο mouse event στο παράθυρο που φιλοξενεί την εικόνα της κάμερας.
					cvSetMouseCallback(camWindow, mouseOnCallback, cam);
//Φορτώνεται το επόμενο frame που λαμβάνει η κάμερα.
```

```
                                        frame = cvQueryFrame(cam);
//Αναμονή τόσα ms για την επανάληψη όλης αυτής της διαδικασίας, όσα ορίζονται από το όρισμα της cvWaitKey
                                        c = cvWaitKey(33);

/*Αν η bRep είναι true (γίνεται true στην mouseOnCallback, δηλαδή όταν γίνει κάποιο click στο παράθυρο που προβάλλεται το video) και ταυτόχρονα έχει επιλεχθεί κάποια εικόνα για να αποτελέσει το νέο υπόβαθρο...*/
                                        if ((bRep) && (gBackgShowImg != NULL))
/*...καλείται η replacement για την αντικατάσταση των pixels του υποβάθρου. Η replacement δέχεται ως όρισμα το τρέχον frame, αλλάζει την πληροφορία του (αλλάζει της τιμές RGB των pixels που πρέπει να αντικατασταθούν με τις τιμές των αντίστοιχων pixels του νέου υποβάθρου) και το επιστρέφει.*/
                                                frame = replacement (startPt, finishPt, frame,
                                                camWindow, mainWindow, txtSxID, txtFxID,
                                                txtSyID, txtFyID);  );

//Προβολή του νέου frame.
                                        cvShowImage (camWindow, frame);
//Αν πατηθεί το escape...
                                        if (c == 27)  {
//Διαγράφεται το gBackgShowImg για την απελευθέρωση της μνήμης και για τη σωστή λειτουργία των παραπάνω ελέγχων.
                                                cvReleaseImage(&gBackgShowImg);
//Ενεργοποιεί το Start webcamera button (btnCam).
                                                mainWindow->GetDlgItem(btnCamID)
                                                ->EnableWindow(true);
//Ενεργοποιεί το comboVidOrImg.
                                                mainWindow->GetDlgItem(comboID)
                                                ->EnableWindow(true);
//Ώστε να σταματήσει το κάλεσμα της replacement (η παραπάνω συνθήκη θα είναι ψευδής).
                                                bRep = false;
                                                break;
                                        }//if (c == 27)
                                }//while(i)
                        }//case false
                        break;

//Αντικατάσταση με video.
                case true:
                        {
//IplImage* που θα φιλοξενεί το εκάστοτε frame του video background που επιλέχθηκε.
                                IplImage* vframe;
```

```
//Μεταβλητή που κρατάει τον αριθμό των frames του video.
                                int numFrames = cvGetCaptureProperty(gBackCapture,
                                CV_CAP_PROP_FRAME_COUNT);
//Απενεργοποιείται το "Open File" button, για να μην επιλεχθεί κάποιο άλλο video, όσο είναι
ανοιχτή η webcamera.
                                mainWindow->GetDlgItem(btnOpenFile)
                                ->EnableWindow(false);

//Όσο ο έλεγχος στη while είναι αληθής, θα προβάλλονται νέα frames από την web camera.
                                while(gI)
                                {
//Αν το τρέχον frame του video είναι ίσο με τον αριθμό των frames...
                                        if (cvGetCaptureProperty(gBackCapture,
                                        CV_CAP_PROP_POS_FRAMES) == numFrames-1)
/*...ορίζεται το επόμενο frame να είναι το πρώτο του video. Ουσιαστικά  επαναλαμβάνεται το
video από την αρχή.*/
                                                cvSetCaptureProperty(gBackCapture,
                                                CV_CAP_PROP_POS_FRAMES, 0);

//Φορτώνεται το επόμενο frame που λαμβάνει η κάμερα.
                                        frame = cvQueryFrame(cam);
//Φορτώνεται το επόμενο frame του video background.
                                        vframe = cvQueryFrame(gBackCapture);

//Αναμονή τόσα ms για την επανάληψη όλης αυτής της διαδικασίας, όσα ορίζονται από το
όρισμα της cvWaitKey.
                                        c = cvWaitKey(33);

/*Αν η bRep είναι true (γίνεται true στην mouseOnCallback, δηλαδή όταν γίνει κάποιο click στο
παράθυρο που προβάλλεται το video) και ταυτόχρονα έχει επιλεχθεί κάποιο video για να
αποτελέσει το νέο υπόβαθρο...*/
                                        if ((bRep) && (gBackCapture != NULL))
/*...καλείται η replacement για την αντικατάσταση των pixels του υποβάθρου. Η replacement
δέχεται ως όρισμα το τρέχον frame της κάμερας και το τρέχον frame του νέου video-υποβάθρου,
αλλάζει την πληροφορία του frame της κάμερας (αλλάζει της τιμές RGB των pixels που πρέπει
να αντικατασταθούν με τις τιμές των αντίστοιχων pixels του νέου υποβάθρου) και το
επιστρέφει.*/
                                        frame = replacement (startPt, finishPt, frame, vframe,
                                        camWindow, mainWindow, txtSxID, txtFxID, txtSyID,
                                        txtFyID);
//Προβολή του νέου frame.
                                        cvShowImage (camWindow, frame);
//Αν πατηθεί το escape...
```

```
                                        if (c == 27)
                                        {
//Απελευθερώνεται η μνήμη που καταχωρείται για το gBackCapture cvCapture* αντικείμενο.
                                                cvReleaseCapture(&gBackCapture);
//Διαγράφεται το gBackgShowImg για την απελευθέρωση της μνήμης και για τη σωστή
λειτουργία των παραπάνω ελέγχων.
                                                cvReleaseImage(&gBackgShowImg);
//Ενεργοποιεί το Start webcamera button (btnCam).
                                                mainWindow->GetDlgItem(btnCamID)
                                                ->EnableWindow(true);
//Ενεργοποιεί το comboVidOrImg.
                                                mainWindow->GetDlgItem(comboID)
                                                ->EnableWindow(true);
//Ενεργοποιείται το btnOpenFile.
                                                mainWindow->GetDlgItem(btnOpenFile)
                                                ->EnableWindow(true);
//Ώστε να σταματήσει το κάλεσμα της replacement (η παραπάνω συνθήκη θα είναι ψευδής).
                                                bRep = false;
                                                break;
                                        }//if (c == 27)
                                }//while(i)
                        }//case true
                        break;
        }//switch (bImgOrVid)
//Διαγράφεται το παράθυρο στο οποίο γίνεται η προβολή της web camera.
        cvDestroyWindow(camWindow);
}//playCamera

void mouseOnCallback (int event, int x, int y, int flags, void* param)
{
//Εκτελείται όταν πατηθεί το αριστερό click.
        if (event == CV_EVENT_LBUTTONDOWN)
        {
//Μεταφέρουμε τα αρχικά (x,y) στο startPt.
                startPt.x = x;
                startPt.y = y;
        }//CV_EVENT_LBUTTONDOWN

//Εκτελείται όταν αφαιθεί το αριστερό click.
        if (event == CV_EVENT_LBUTTONUP)
        {
//Περνάτε το τρέχον frame του video σε ένα IplImage* αντικείμενο.
                IplImage* temp = cvQueryFrame((CvCapture*)param);
```

```
//Μεταφέρουμε τα τελικά (x,y) στο finishPt.
                finishPt.x = x;
                finishPt.y = y;

//Αν το ποντίκι βρίσκεται έξω από τα όρια της εικόνας όταν υπολογίζεται το finishPt, η διαδικασία υπολογισμού τερματίζεται (return).
                if ((finishPt.x < 0) || (finishPt.x > temp->width)) return;
                if ((finishPt.y < 0) || (finishPt.y > temp->height)) return;

/*Οι τέσσερις συνθήκες ελέγχου χρησιμοποιούνται για τον καθορισμό του παραλληλογράμμου που ορίζεται από τα τέσσερα σημεία (startPt.x, startPt.y, finishPt.x, finishPt.y). Ανάλογα με τη θέση αυτών των σημείων καθορίζεται και ο τρόπος που θα περαστούν ως ορίσματα στην colourCalculation. Έτσι ο υπολογισμός των range στην παραπάνω συνάρτηση γίνεται με τον ίδιο τρόπο.
*/
                if ((startPt.x < finishPt.x) && (startPt.y < finishPt.y))
                        colourCalculation (startPt.x, startPt.y, (finishPt.x + 1), (finishPt.y + 1),
                        temp);
                else if ((startPt.x < finishPt.x) && (startPt.y > finishPt.y))
                        colourCalculation (startPt.x, finishPt.y, (finishPt.x + 1), (startPt.y + 1),
                        temp);
                else if ((startPt.x > finishPt.x) && (startPt.y < finishPt.y))
                        colourCalculation (finishPt.x, startPt.y, (startPt.x + 1), (finishPt.y + 1),
                        temp);
                else
                        colourCalculation (finishPt.x, finishPt.y, (startPt.x + 1), (startPt.y + 1),
                        temp);

//Η bRep γίνεται true ώστε να αρχίσει το replacement των pixels (συνθήκη ελέγχου μέσα στο while της playCamera).
                bRep = true;

        }//CV_EVENT_LBUTTONUP

}//mouseOnCallback

IplImage* replacement (CvPoint sPt, CvPoint fPt, IplImage* camFrame, char* camWindow,
CWnd* mainWindow, int txtSxID, int txtFxID, int txtSyID, int txtFyID)
{
//Βοηθητικό CvPoint για τον καθορισμό του εκάστοτε pixel.
        CvPoint pt = {0,0};
```

```
//Οι τιμές των τεσσάρων σημείων εμφανίζονται στα ανάλογα edit controls του αρχικού
παραθύρου.
//txtSxID: το ID του txtSxV.
        mainWindow->SetDlgItemInt(txtSxID, sPt.x, 1);
//txtFxID: το ID του txtFxV.
        mainWindow->SetDlgItemInt(txtFxID, fPt.x, 1);
//txtSyID: το ID του txtSyV.
        mainWindow->SetDlgItemInt(txtSyID, sPt.y, 1);
//txtFyID: το ID του txtFyV.
        mainWindow->SetDlgItemInt(txtFyID, fPt.y, 1);

/*Γίνεται resize το gBackground στην ανάλυση του gBackgShowImg. Το gBackgShowImg έχει
την ίδια ανάλυση με την web camera (όταν δημιουργείται έχει την ανάλυση που έχει η web
camera του συστήματος – όπως και το gBackground -, ενώ όταν κινηθεί ο slideRes αλλάζει η
ανάλυσή του ώστε να ταιριάζει πάλι με την ανάλυση της webcamera), οπότε με τη cvResize
περνάται όλη η πληροφορία του gBackground (δηλαδή το νέο υπόβαθρο) στην gBackgShowImg,
η οποία χρησιμοποιείται για την αντικατάσταση..*/
        cvResize(gBackground, gBackgShowImg, 1);

//Αν τα height και width των gBackgShowImg και camFrame (τρέχον frame της web camera)
είναι ίσα...
        if ((gBackgShowImg->height == camFrame->height) && (gBackgShowImg->width ==
        camFrame->width))
        {
//Μία διπλή for για τη σάρωση ολόκληρης της εικόνας.
                for (int i=0; i < camFrame->width; i++)
                        for (int j=0; j < camFrame->height; j++)
                        {
//Ορίζονται τα x και y του αντικειμένου CvPoint να είναι αντίστοιχα με τα i και j της for.
                                pt.x = i;
                                pt.y = j;

/*Ελέγχεται αν το εκάστοτε pixel του τρεχόντος frame έχει τιμές R,G,B μέσα στο range που
ορίζουν τα ranges του κάθε χρώματος. Αν ισχύει η συνθήκη, το pixel αυτό αντικαθίσταται με το
αντίστοιχο pixel του νέου background.*/
                                if ( ((((uchar)(camFrame->imageData + camFrame-
                                >widthStep*pt.y)[pt.x*3+2]) < rangeRed+gRange)
                                        && (((uchar)(camFrame->imageData + camFrame
                                        ->widthStep*pt.y)[pt.x*3+2]) > rangeRed-gRange))
                                        && ((((uchar)(camFrame->imageData + camFrame
                                        ->widthStep*pt.y)[pt.x*3+1]) < rangeGreen+gRange)
                                        && (((uchar)(camFrame->imageData + camFrame
                                        ->widthStep*pt.y)[pt.x*3+1]) > rangeGreen-gRange))
```

```
                                    && ((((uchar)(camFrame->imageData + camFrame
                                    ->widthStep*pt.y)[pt.x*3]) < rangeBlue+gRange)
                                    && (((uchar)(camFrame->imageData + camFrame
                                    ->widthStep*pt.y)[pt.x*3]) > rangeBlue-gRange)) )
                                {
//Αν ισχύει η παραπάνω συνθήκη, αντικαθίστανται τις τιμές του RGB του τρέχοντος pixel με τις τιμές του αντίστοιχου pixel του επιλεγμένου background.
                                    ((uchar*)(camFrame->imageData + camFrame
                                    ->widthStep*pt.y))[pt.x*3+2] =
                                    ((uchar*)(gBackgShowImg->imageData +
                                    gBackgShowImg->widthStep*pt.y))[pt.x*3+2];

                                    ((uchar*)(camFrame->imageData + camFrame
                                    ->widthStep*pt.y))[pt.x*3+1] =
                                    ((uchar*)(gBackgShowImg->imageData +
                                    gBackgShowImg->widthStep*pt.y))[pt.x*3+1];

                                    ((uchar*)(camFrame->imageData + camFrame
                                    ->widthStep*pt.y))[pt.x*3] =
                                    ((uchar*)(gBackgShowImg->imageData +
                                    gBackgShowImg->widthStep*pt.y))[pt.x*3];
                                }//if
                            }//for y

//Επιστρέφεται το παραλλαγμένο frame, το οποίο προβάλλεται στην playCamera.
                            return camFrame;
                    }//if ((gBackgShowImg->height == camFrame->height) && (gBackgShowImg->width
                    == camFrame->width))
}//replacement

IplImage* replacement (CvPoint sPt, CvPoint fPt, IplImage* camFrame, IplImage* videoFrame,
char* camWindow, CWnd* mainWindow, int txtSxID, int txtFxID, int txtSyID, int txtFyID)
{
//Βοηθητικό CvPoint για τον καθορισμό του εκάστοτε pixel.
        CvPoint pt = {0,0};

//Οι τιμές των τεσσάρων σημείων εμφανίζονται στα ανάλογα edit controls του αρχικού παραθύρου.
//txtSxID: το ID του txtSxV.
        mainWindow->SetDlgItemInt(txtSxID, sPt.x, 1);
//txtFxID: το ID του txtFxV.
        mainWindow->SetDlgItemInt(txtFxID, fPt.x, 1);
//txtSyID: το ID του txtSyV.
```

```
        mainWindow->SetDlgItemInt(txtSyID, sPt.y, 1);
//txtFyID: το ID του txtFyV.
        mainWindow->SetDlgItemInt(txtFyID, fPt.y, 1);

/*Γίνεται resize του videoFrame στήν ανάλυση του gBackgShowImg. Με τον τρόπο αυτό, η
πληροφορία του videoFrame μεταφέρεται και προσαρμόζεται στις διαστάσεις του
gBackgShowImg, το οποίο έχει πάντα ανάλυση ίδια με την τρέχουσα ανάλυση της web camera.*/
        cvResize(videoFrame, gBackgShowImg, 1);

//Αν τα height και width των gBackgShowImg και camFrame (τρέχον frame της web camera)
είναι ίσα...
        if ((gBackgShowImg->height == camFrame->height) && (gBackgShowImg->width ==
        camFrame->width))
        {
//Μία διπλή for για τη σάρωση ολόκληρης της εικόνας.
                for (int i=0; i < camFrame->width; i++)
                        for (int j=0; j < camFrame->height; j++)
                        {
//Ορίζονται τα x και y του αντικειμένου CvPoint να είναι αντίστοιχα με τα i και j της for.
                                pt.x = i;
                                pt.y = j;

/*Ελέγχεται αν το εκάστοτε pixel του τρεχόντος frame έχει τιμές R,G,B μέσα στο range που
ορίζουν τα ranges του κάθε χρώματος. Αν ισχύει η συνθήκη, το pixel αυτό αντικαθίσταται με το
αντίστοιχο pixel του νέου background.*/
                                if ( ((((uchar)(camFrame->imageData + camFrame
                                        ->widthStep*pt.y)[pt.x*3+2]) < rangeRed+gRange)
                                        && (((uchar)(camFrame->imageData + camFrame
                                        ->widthStep*pt.y)[pt.x*3+2]) > rangeRed-gRange))
                                        && ((((uchar)(camFrame->imageData + camFrame
                                        ->widthStep*pt.y)[pt.x*3+1]) < rangeGreen+gRange)
                                        && (((uchar)(camFrame->imageData + camFrame
                                        ->widthStep*pt.y)[pt.x*3+1]) > rangeGreen-gRange))
                                        && ((((uchar)(camFrame->imageData + camFrame
                                        ->widthStep*pt.y)[pt.x*3]) < rangeBlue+gRange)
                                        && (((uchar)(camFrame->imageData + camFrame
                                        ->widthStep*pt.y)[pt.x*3]) > rangeBlue-gRange)) )
                                {
//Αν ισχύει η παραπάνω συνθήκη, αντικαθιστούμε τις τιμές του RGB του τρέχοντος pixel με τις
τιμές του αντίστοιχου pixel του άλλου background.
                                        ((uchar*)(camFrame->imageData + camFrame
                                        ->widthStep*pt.y))[pt.x*3+2] =
                                        ((uchar*)(gBackgShowImg->imageData +
```

```
                                        gBackgShowImg->widthStep*pt.y))[pt.x*3+2];

                                        ((uchar*)(camFrame->imageData + camFrame
                                        ->widthStep*pt.y))[pt.x*3+1] =
                                        ((uchar*)(gBackgShowImg->imageData +
                                        gBackgShowImg->widthStep*pt.y))[pt.x*3+1];

                                        ((uchar*)(camFrame->imageData + camFrame
                                        ->widthStep*pt.y))[pt.x*3] =
                                        ((uchar*)(gBackgShowImg->imageData +
                                        gBackgShowImg->widthStep*pt.y))[pt.x*3];
                                }//if
                        }//for y

//Επιστρέφεται το παραλλαγμένο frame, το οποίο προβάλλεται στην playCamera.
                        return camFrame;
        }//if ((videoFrame->height == camFrame->height) && (videoFrame->width ==
        camFrame->width))
}//replacement

void colourCalculation (int sX, int sY, int fX, int fY, IplImage* picture)
{
//Ενα πλήθος int μεταβλητών που θα χρησιμοποιηθούν για τον υπολογισμό των μεγίστων και των
ελαχίστων των τιμών των τριών χρωμάτων.
        int maxRed = 0;
        int minRed = 256;
        int maxGreen = 0;
        int minGreen = 256;
        int maxBlue = 0;
        int minBlue = 256;

//Δημιουργούμε ένα βοηθητικό CvPoint στο οποίο θα δίνουμε, με κάθε εκτέλεση των for, τα x
και y που αντιστοιχούν στο τετράγωνο των pixels που ορίσαμε.
        CvPoint pt = {0,0};

//Υπολογισμός των μεγίστων και ελαχίστων R,G,B.
        for (int i = sX; i < fX; i++)
                for (int j = sY; j < fY; j++)
                {
                        pt.x = i;
                        pt.y = j;

//red
```

```
                        if ( ((uchar)((picture->imageData + picture->widthStep*pt.y)[pt.x*3+2]))
                        > maxRed )
                                maxRed = (uchar)((picture->imageData + picture
                                ->widthStep*pt.y)[pt.x*3+2]);
                        if ( ((uchar)((picture->imageData + picture->widthStep*pt.y)[pt.x*3+2]))
                        < minRed )
                                minRed = (uchar)((picture->imageData + picture
                                ->widthStep*pt.y)[pt.x*3+2]);

//green
                        if ( ((uchar)((picture->imageData + picture->widthStep*pt.y)[pt.x*3+1]))
                        > maxGreen )
                                maxGreen = (uchar)((picture->imageData + picture
                                ->widthStep*pt.y)[pt.x*3+1]);
                        if ( ((uchar)((picture->imageData + picture->widthStep*pt.y)[pt.x*3+1]))
                        < minGreen )
                                minGreen = (uchar)((picture->imageData + picture
                                ->widthStep*pt.y)[pt.x*3+1]);

//blue
                        if ( ((uchar)((picture->imageData + picture->widthStep*pt.y)[pt.x*3]))
                        > maxBlue )
                                maxBlue = (uchar)((picture->imageData + picture
                                ->widthStep*pt.y)[pt.x*3]);
                        if ( ((uchar)((picture->imageData + picture->widthStep*pt.y)[pt.x*3+1]))
                        < minBlue )
                                minBlue = (uchar)((picture->imageData + picture
                                ->widthStep*pt.y)[pt.x*3]);
                }//for j

//Υπολογισμός των range με βάση των μεγίστων και ελαχίστων του κάθε χρώματος.
        rangeRed = ((minRed-1)+(maxRed+1))/2;
        rangeGreen = ((minGreen-1)+(maxGreen+1))/2;
        rangeBlue = ((minBlue-1)+(maxBlue+1))/2;

}//colourCalculation

void appendEditText(CEdit& edit, LPCTSTR pszText)
{
//Δημιουργία ενός CString αντικειμένου.
        CString strLine;

        strLine.Format(_T("\r\n%s"), pszText);  //
```

```
	int nLength = edit.GetWindowTextLengthA();  //

	edit.SetSel(nLength, nLength);  //

	edit.ReplaceSel(strLine);  //
}//appendEditText

#include "stdafx.h"
#include "BackgroundSegmentation.h"
#include "BackgroundSegmentationDlg.h"
#include "highgui.h"
#include "cv.h"
#include "Utils.h"
#include "Globals.h"

//Global Declarations
//Το νέο image background.
IplImage* gBackground;
//Το image background που αντικαθιστά αυτό της web camera.
IplImage* gBackgShowImg;
//Το νέο video background.
CvCapture* gBackCapture;

int gRange;
int gI;

#ifdef _DEBUG
#define new DEBUG_NEW
#endif

// CAboutDlg dialog που χρησιμοποιείται για το App About
class CAboutDlg : public CDialog
{
public:
	CAboutDlg();

// Dialog Data
	enum { IDD = IDD_ABOUTBOX };

	protected:
	virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV υποστήριξη
```

```
// Υλοποίηση
protected:
	DECLARE_MESSAGE_MAP()
};

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
END_MESSAGE_MAP()


// CBackgroundSegmentationDlg dialog

CBackgroundSegmentationDlg::CBackgroundSegmentationDlg(CWnd* pParent /*=NULL*/)
	: CDialog(CBackgroundSegmentationDlg::IDD, pParent)
{
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}

void CBackgroundSegmentationDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	DDX_Control(pDX, btnCam, c_btnCam);
	DDX_Control(pDX, btnShowFile, c_btnShowFile);
	DDX_Control(pDX, btnOpenFile, c_btnOpenFile);
	DDX_Control(pDX, slideRes, c_slideRes);
	DDX_Control(pDX, slideRange, c_slideRange);
	DDX_Control(pDX, txtRangeV, c_txtRange);
	DDX_Control(pDX, txtInfo, c_txtInfo);
	DDX_Control(pDX, comboVidOrImg, c_comboVidOrImg);
}

BEGIN_MESSAGE_MAP(CBackgroundSegmentationDlg, CDialog)
	ON_WM_SYSCOMMAND()
	ON_WM_PAINT()
	ON_WM_QUERYDRAGICON()
```

```
	//}}AFX_MSG_MAP
	ON_BN_CLICKED(btnCam, &CBackgroundSegmentationDlg::OnBnClickedbtnCam)
	ON_BN_CLICKED(btnShowFile,
&CBackgroundSegmentationDlg::OnBnClickedbtnshowfile)
	ON_BN_CLICKED(btnOpenFile,
&CBackgroundSegmentationDlg::OnBnClickedbtnopenfile)
	ON_WM_HSCROLL()
	ON_WM_CLOSE()
	ON_CBN_SELCHANGE(comboVidOrImg,
&CBackgroundSegmentationDlg::OnCbnSelchangecombovidorimg)
END_MESSAGE_MAP()


// CBackgroundSegmentationDlg message handlers

BOOL CBackgroundSegmentationDlg::OnInitDialog()
{
	CDialog::OnInitDialog();

	// Πρέπει να προστεθεί "About..." menu στο system menu.

	// IDM_ABOUTBOX πρέπει να είναι στο system command range.
	ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
	ASSERT(IDM_ABOUTBOX < 0xF000);

	CMenu* pSysMenu = GetSystemMenu(FALSE);
	if (pSysMenu != NULL)
	{
		CString strAboutMenu;
		strAboutMenu.LoadString(IDS_ABOUTBOX);
		if (!strAboutMenu.IsEmpty())
		{
			pSysMenu->AppendMenu(MF_SEPARATOR);
			pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX,
			strAboutMenu);
		}
	}

// Τοποθέτηση του icon για αυτό το dialog.  Το framework το κάνει αυτό αυτόματα
// όταν το βασικό παράθυρο της εφαρμογής δεν είναι dialog
// Μεγάλο icon
	SetIcon(m_hIcon, TRUE);
// Μικρό icon
```

```
	SetIcon(m_hIcon, FALSE);
//Δημιουργία του c_toolTip.
	if( !c_toolTip.Create(this))
		TRACE0("Unable to create the ToolTip!");
	else
	{
//Προσθήκη tooltips στα υπάρχοντα controls.
		c_toolTip.AddTool(&c_slideRes, "Changes the resolution of the camera and the
		background.");
		c_toolTip.AddTool(&c_slideRange, "Changes the RGB range of the selected
		pixels.");
		c_toolTip.AddTool(&c_btnOpenFile, "Search for the desired background.");
		c_toolTip.AddTool(&c_btnShowFile, "Shows the selected background.");
//Ενεργοποίηση του c_toolTip.
		c_toolTip.Activate(TRUE);
	}

//Άλλα πράγματα που θα γίνονται όταν φορτώνεται το BackgroundSegmentationDlg.
//Η αρχική τιμή του gRange ορίζεται στα 35.
	gRange = 35;
//To range του slideRes.
	c_slideRes.SetRange(1, 3);
//To range του slideRange.
	c_slideRange.SetRangeMin(0, 100);
//Η θέση του slider αρχικοποιείται στην τιμή του gRange.
	c_slideRange.SetPos(gRange);
//Αρχικοποιούμε το CvCapture* αντικείμενο Video (για την web camera).
	Video = cvCreateCameraCapture(0);
//Αποθηκεύουμε το αρχικό πλάτος του Video.
	videoSWidth = cvGetCaptureProperty(Video, CV_CAP_PROP_FRAME_WIDTH);
//Αποθηκεύουμε το αρχικό ύψος του Video.
	videoSHeight = cvGetCaptureProperty(Video, CV_CAP_PROP_FRAME_HEIGHT);
//Δημιουργούμε το gBackground αντικείμενο (κρατάει την εικόνα που επιλέγεται από το
btnOpenFile).
	gBackground = cvCreateImage(cvSize(videoSWidth,videoSHeight), 8, 3);
//Δημιουργία του CWnd αντικειμένου. Χρησιμοποιείται για να υπάρχει πρόσβαση στα διάφορα
controls του παραθύρου που ορίζεται.
	window = new CWnd();
//Η bImgOrVid αρχικοποιείται σε false (για αντικατάσταση με εικόνα).
	bImgOrVid = false;
//Ορίζεται αρχική επιλογή στο combo box να είναι η πρώτη (το image).
	c_comboVidOrImg.SetCurSel(0);
```

```
        c_txtInfo.SetWindowTextA("To start the web camera, press the 'Start Webcamera'
        button.");
        appendEditText(c_txtInfo, "In order to select a new background for your camera, first
        choose the file type you want from the combo box and then press the 'Open File' button to
        open an image or video file.");

// επιστρέφεται TRUE εκτός και αν τεθεί το focus σε ένα control
        return TRUE;
}

void CBackgroundSegmentationDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
        if ((nID & 0xFFF0) == IDM_ABOUTBOX)
        {
                CAboutDlg dlgAbout;
                dlgAbout.DoModal();
        }
        else
        {
                CDialog::OnSysCommand(nID, lParam);
        }
}

// Αν προστεθεί κουμπί για minimize στο dialog, ο παρακάτω κώδικαν παρακάτω
//  είναι απαραίτητος για το σχεδιασμό του icon.  Για τις MFC εφαρμογές που χρησιμοποιούν
//  αυτό το document/view model , αυτό γίνεται αυτόματα από το framework.

void CBackgroundSegmentationDlg::OnPaint()
{
        if (IsIconic())
        {
// context για painting
                CPaintDC dc(this);
                SendMessage(WM_ICONERASEBKGND,
reinterpret_cast<WPARAM>(dc.GetSafeHdc()), 0);

// Κεντρικό icon για το client rectangle
                int cxIcon = GetSystemMetrics(SM_CXICON);
                int cyIcon = GetSystemMetrics(SM_CYICON);
                CRect rect;
                GetClientRect(&rect);
                int x = (rect.Width() - cxIcon + 1) / 2;
                int y = (rect.Height() - cyIcon + 1) / 2;
```

```
// Σχεδιασμός του icon
                dc.DrawIcon(x, y, m_hIcon);
        }
        else
        {
                CDialog::OnPaint();
        }

//Εκτελέσεις που θα γίνουν με το που σχεδιαστεί το βασικό dialog της εφαρμογής.
//Τοποθετούμε στο txtRangeV το range που ορίζεται από τον slideRange.
        window->GetActiveWindow()->SetDlgItemInt(txtRangeV, c_slideRange.GetPos(), 1);
//Τοποθέτηση στο txtResXV το Width του Video.
        window->GetActiveWindow()->SetDlgItemInt(txtResXV, cvGetCaptureProperty(Video,
CV_CAP_PROP_FRAME_WIDTH));
//Τοποθέτηση στο txtResYV το Height του Video.
        window->GetActiveWindow()->SetDlgItemInt(txtResYV, cvGetCaptureProperty(Video,
CV_CAP_PROP_FRAME_HEIGHT));
}

// Το σύστημα καλεί αυτή τη function για να λάβει το cursor που θα προβάλεταιόσο ο χρήστης
//  σείρει το minimized window.
HCURSOR CBackgroundSegmentationDlg::OnQueryDragIcon()
{
        return static_cast<HCURSOR>(m_hIcon);
}

//Event handler για το Start Webcamera button (btnCam).
void CBackgroundSegmentationDlg::OnBnClickedbtnCam()
{
//Απενεργοποιεί το btnStart.
        c_btnCam.EnableWindow(false);
//Απενεργοποιεί το comboVidOrImg.
        c_comboVidOrImg.EnableWindow(false);
        appendEditText(c_txtInfo, "Camera started.");
        appendEditText(c_txtInfo, "After selecting a new background, choose the pixel(s)
        according to which the Background Segmentation will take place.");

//Καλεί την playCamera.
        playCamera(Video, "Web Camera", 1, bImgOrVid, window->GetActiveWindow(),
        btnCam, txtSxV, txtFxV, txtSyV, txtFyV, comboVidOrImg);

        appendEditText(c_txtInfo, "Camera stopped.");
```

```
}//OnBnClickedbtnCam()

//Event handler για το Open Image button (btnOpenImg).
void CBackgroundSegmentationDlg::OnBnClickedbtnopenfile()
{
	switch (bImgOrVid)
	{
//Αν η bImgOrVid είναι false, το btnOpenFile ανοίγει εικόνα.
		case false:
			{
//Δημιουργείται ένα παράθυρο για την εύρεση και το άνοιγμα μίας εικόνας.
				CFileDialog dlgOpenImage (true, ("*.bmp"), NULL,
				OFN_FILEMUSTEXIST|OFN_HIDEREADONLY|OFN_PATH
				MUSTEXIST, ("image files (*.bmp; *.jpg) |*.bmp;*.jpg|All
				Files (*.*)|*.*||"), NULL);

//Ορίζεται ο τίτλος του παραθύρου που δημιουργήθηκε.
				dlgOpenImage.m_ofn.lpstrTitle = "Open image";

//Αν πατηθεί το OK button, τότε αποθηκεύεται σε μία global IplImage* (gBackground) η εικόνα
που επιλέχθηκε.
				if (dlgOpenImage.DoModal() == IDOK)
				{
//Απελευθερώνεται η προηγούμενη εικόνα που φορτώθηκε στο gBackground ώστε να
απελευθερωθεί μνήμη.
					cvReleaseImage(&gBackground);
					cvReleaseImage(&gBackgShowImg);
//Φορτώνει την εικόνα που επιλέχθηκε στο gBackground.
					gBackground = cvLoadImage
					(dlgOpenImage.GetPathName());
//Δημιουργείται το gBackgShowImg. Η ανάλυσή του ορίζεται να είναι ίδια με την ανάλυση του
video την τρέχουσα στιγμή.
					gBackgShowImg =
					cvCreateImage(cvSize(cvGetCaptureProperty(Video,
					CV_CAP_PROP_FRAME_WIDTH),
					cvGetCaptureProperty(Video,
					CV_CAP_PROP_FRAME_HEIGHT)),
					gBackground->depth, gBackground->nChannels);

//Γίνεται resize το gBackground στο gBackgShowImg, ώστε να χωρέσει η πληροφορία του
πρώτου στην ανάλυση του δεύτερου.
					cvResize(gBackground, gBackgShowImg, 1);
```

```
                                        appendEditText(c_txtInfo, "A new image background
has been selected.");
                                }//if (dlgOpenImage.DoModal() == IDOK)
                        }//case false
                        break;

//Αν η bImgOrVid είναι true, το btnOpenFile ανοίγει video.
                case true:
                        {
//Δημιουργείται ένα παράθυρο για την εύρεση και το άνοιγμα μίας εικόνας.
                                CFileDialog dlgOpenVideo (true, ("*.avi"), NULL,
                                OFN_FILEMUSTEXIST|OFN_HIDEREADONLY|OFN_PATH
                                MUSTEXIST, ("video files (*.avi; *.mpeg; *.wmv) |*.avi;
                                *.mpeg; *.wmv|All Files (*.*)|*.*||"), NULL);
//Ορίζεται ο τίτλος του παραθύρου που δημιουργήθηκε.
                                dlgOpenVideo.m_ofn.lpstrTitle = "Open video";

//Αν πατηθεί το OK button, τότε αποθηκεύεται σε μία global CvCapture* (gBackCapture) το
video που επιλέχθηκε.
                                if (dlgOpenVideo.DoModal() == IDOK)
                                {
//Απελευθερώνεται το προηγούμενο video που φορτώθηκε στο gBackCapture ώστε να
απελευθερωθεί μνήμη.
                                        cvReleaseCapture(&gBackCapture);
//Φορτώνει το video που επιλέχθηκε στο gBackCapture.
                                        gBackCapture = cvCreateFileCapture
                                        (dlgOpenVideo.GetPathName());
//Δημιουργείται το gBackgShowImg. Η ανάλυσή του ορίζεται να είναι ίδια με την ανάλυση του
video την τρέχουσα στιγμή.
                                        gBackgShowImg =
                                        cvCreateImage(cvSize(cvGetCaptureProperty(Video,
                                        CV_CAP_PROP_FRAME_WIDTH),
                                        cvGetCaptureProperty(Video,
                                        CV_CAP_PROP_FRAME_HEIGHT)),
                                        gBackground->depth, gBackground->nChannels);

                                        appendEditText(c_txtInfo, "A new video background
                                        has been selected.");
                                }//if (dlgOpenVideo.DoModal() == IDOK)
                        }//case true

                        break;
        }//switch (bImgOrVid)
```

```
}//OnBnClickedbtnopenfile()

//Event handler για το Show Image button (btnShowFile).
void CBackgroundSegmentationDlg::OnBnClickedbtnshowfile()
{
	switch (bImgOrVid)
	{
//Αν η bImgOrVid είναι false, το btnShowFile εμφανίζει την εικόνα που επιλέχθηκε.
		case false:
			if (gBackgShowImg != NULL)
				cvShowImage("New Background", gBackgShowImg);
//Εμφανίζει την εικόνα που επιλέξαμε για νέο background σε ένα νέο παράθυρο.
			break;

//Αν η bImgOrVid είναι true, το btnShowFile εμφανίζει το video που επιλέχθηκε.
		case true:
			{
				if (gBackCapture != NULL)
				{
//Κώδικας για την αναπαραγωγή του video file που επιλέχθηκε.
					IplImage* frame;
					char c;

					while(1)
					{
						frame = cvQueryFrame(gBackCapture);
						cvShowImage("New Video Background",
						frame);
						c = cvWaitKey(33);
						if (c == 27)
						{
							cvDestroyWindow("New Video
							Background");
							break;
						}//if (c == 27)
					}//while(1)
				}//if (gBackShowCapture != NULL)
			}//case true

			break;
	}//switch (bImgOrVid)
}//OnBnClickedbtnshowfile()
```

```
//Event Handler για τη μετακίνηση ενός οριζόντιου slider.
void CBackgroundSegmentationDlg::OnHScroll(UINT nSBCode, UINT nPos, CScrollBar*
pScrollBar)
{
//Αν κινηθεί ο slideRes, εκτελείται το παρακάτω.
	if (pScrollBar->GetDlgCtrlID() == slideRes)
	{
//Το Width του video γίνεται ίσο με το αρχικό Width / τη θέση του slideRes.
		cvSetCaptureProperty(Video, CV_CAP_PROP_FRAME_WIDTH,
		videoSWidth/c_slideRes.GetPos());
//Το Height του video γίνεται ίσο με το αρχικό Height / τη θέση του slideRes.
		cvSetCaptureProperty(Video, CV_CAP_PROP_FRAME_HEIGHT,
		videoSHeight/c_slideRes.GetPos());
//Τοποθέτηση στο txtResXV το Width του Video.
		window->GetActiveWindow()->SetDlgItemInt(txtResXV,
		cvGetCaptureProperty(Video, CV_CAP_PROP_FRAME_WIDTH));
//Τοποθέτηση στο txtResYV το Height του Video.
		window->GetActiveWindow()->SetDlgItemInt(txtResYV,
		cvGetCaptureProperty(Video, CV_CAP_PROP_FRAME_HEIGHT));

		if (gBackgShowImg != NULL)
		{
			cvReleaseImage(&gBackgShowImg);
//Δημιουργούμε και καταχωρούμε στο gBackgShowImg μία νέα εικόνα, με διαστάσεις τις νέες
διαστάσεις του video της web camera.
			gBackgShowImg = cvCreateImage(cvSize(cvGetCaptureProperty(Video,
			CV_CAP_PROP_FRAME_WIDTH), cvGetCaptureProperty(Video,
			CV_CAP_PROP_FRAME_HEIGHT)),
			gBackground->depth, gBackground->nChannels);
//cvResize(gBackground, gBackgShowImg, 1);  //Γίνεται resize του gBackground στήν ανάλυση
του gBackgShowImg. Με τον τρόπο αυτό, η πληροφορία του gBackground μεταφέρεται και
		}
	}//if (pScrollBar->GetDlgCtrlID() == slideRes)

//Αν κινηθεί ο slideRange, εκτελείται το παρακάτω.
//Αν ο slider που κινήθηκε είναι ο slideRange...
	if (pScrollBar->GetDlgCtrlID() == slideRange)
	{
//Η gRange παίρνει την τιμή της θέσης του slideRange.
		gRange = c_slideRange.GetPos();

//Τοποθετούμε στο txtRangeV το range που ορίζεται από τον slideRange.
```

```
                window->GetActiveWindow()->SetDlgItemInt(txtRangeV,
                c_slideRange.GetPos(), 1);
        }//if (pScrollBar->GetDlgCtrlID() == slideRange)

        CDialog::OnHScroll(nSBCode, nPos, pScrollBar);
}//OnHScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar)

//Event handler για την αλλαγή της επιλογής στο comboVidOrImg.
void CBackgroundSegmentationDlg::OnCbnSelchangecombovidorimg()
{
/*Ανάλογα με την επιλογή που γίνεται στο combo box, το bImgOrVideo γίνεται true ή false. Με
τον τρόπο αυτό αλλάζει η λειτουργία των btnOpenFile, btnShowFile και, κατ' επέκταση, η
διαδικασία της αντικατάστασης.
*/
        if (c_comboVidOrImg.GetCurSel() == 0)
                bImgOrVid = false;
        else
                bImgOrVid = true;
}//OnCbnSelchangecombovidorimg()


//Όταν κλείσει η εφαρμογή...
void CBackgroundSegmentationDlg::OnClose()
{
        gI = 0;
        cvDestroyWindow("Web Camera");
//Απελευθερώνεται το capture της web camera.
        cvReleaseCapture(&Video);
//Διαγράφεται το CWnd αντικείμενο.
        delete window;
//Κλείνει η εφαρμογή.
        CDialog::OnCancel();
}//OnClose()

void CBackgroundSegmentationDlg::OnCancel()
{
}//OnCancel()

BOOL CBackgroundSegmentationDlg::PreTranslateMessage(MSG* pMsg)
{
//Πέρασμα ενός mouse message στο tooltip control (c_toolTip) για επεξεργασία.
        c_toolTip.RelayEvent(pMsg);
```

```
        return CDialog::PreTranslateMessage(pMsg);
}//PreTranslateMessage(MSG* pMsg)
```