ΤΜΗΜΑ ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

# ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

## Αναγνώριση ολοκληρωμένων στοιχείων σε πλακέτα με χρήση έξυπνου κινητού

## Using smartphone camera for identifying components on integrated circuits

Μιχαήλου Λάμπρος

ΧΑΝΙΑ 2016

ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ:

Καθηγητής Μιχάλης Ζερβάκης (Επιβλέπων)
Καθηγητής Γιάννης Παπαευσταθίου
Καθηγητής Κοσμήτορας της Σχολής Απόστολος Δόλλας

# Περίληψη

Τα τελευταία χρόνια τα έξυπνα κινητά τηλέφωνα συμβάλουν καθοριστικά στην διεκπεραίωση σύγχρονων καθημερινών εργασιών και λειτουργιών του ανθρώπου. Αυτή η καθημερινά αυξανόμενη χρήση των κινητών οδηγεί στην συνεχόμενη βελτιστοποίηση των τεχνικών χαρακτηριστικών τους από την μεριά των κατασκευαστών (πολυπύρηνοι επεξεργαστές, προηγμένες ενσωματωμένες κάρτες γραφικών, υψηλής ανάλυσης κάμερες) μετατρέποντας τα σε μικρούς υπολογιστές τσέπης, δίνοντας στο χρήστη τεράστιες δυνατότητες.
Ταυτόχρονα οι προγραμματιστές αποκτώντας πρόσβαση σε αυτά τα τεχνικά χαρακτηριστικά δημιουργούν προηγμένες εφαρμογές με εξελιγμένες δυνατότητες. Εφαρμογές όπως πλοήγηση στο διαδίκτυο, αναγνώριση τραγουδιών, κάμερα με λειτουργίες σταθεροποίησης εικόνας, ειδικά φίλτρα εστίασης, ανίχνευσης προσώπου, αντικειμένου κ.α.
Στόχος της παρούσας εργασίας είναι η ανάπτυξη μιας εφαρμογής που αξιοποιεί πλήρως τις δυνατότητες του hardware και των εφαρμογών που προσφέρει ένα έξυπνο κινητό και πιο συγκεκριμένα μια συσκευή Android. Στο τέλος ο χρήστης έχει στη διάθεση του μια πλήρως λειτουργική και εύχρηστη εφαρμογή αναγνώρισης ολοκληρωμένων στοιχείων.
Η εφαρμογή αναπτύχθηκε για την πλατφόρμα Android, κάνοντας χρήση του Android SDK και της μηχανής αναγνώρισης χαρακτήρων Tesseract OCR. Για το κάλεσμα της βιβλιοθήκης Tesseract είναι απαραίτητη η χρήση του Android NDK και του Java Native Interface (JNI).
Παρουσιάζονται τα προβλήματα που προέκυψαν κατά την υλοποίηση της μεθόδου και οι λύσεις που επιλέχθηκαν για την αντιμετώπισή τους. Επίσης, επισημαίνεται η ανάγκη για δημιουργία διεθνούς βάσης δεδομένων για την αναγνώριση ολοκληρωμένων στοιχείων, αφού ελλείψει αυτής δεν μπορεί να γίνει αξιόπιστη σύγκριση των προτεινόμενων λύσεων και επιλογή της βέλτιστης.

# Ευχαριστίες

Στο σημείο αυτό, θα ήθελα να εκφράσω τις θερμές ευχαριστίες μου σε όλους όσους με βοήθησαν και συνέβαλλαν στην ολοκλήρωση της εργασίας αυτής. Πρώτον από όλους ευχαριστώ τον επιβλέποντα καθηγητή μου κ. Ζερβάκη που δεν απέρριψε το θέμα μου, για την πολύ καλή και ευχάριστη συνεργασία που είχαμε και για τις γνώσεις που με τη βοήθειά του αποκόμισα όλο αυτό το διάστημα.

Συνεχίζοντας, θα ήθελα να πω ένα μεγάλο ευχαριστώ στην οικογένειά μου, για την πολύτιμη στήριξη της όλα αυτά τα χρόνια, συμβάλλοντας καθοριστικά στην εξέλιξης μου.

Τέλος, ευχαριστώ όλους τους φίλους και τα κοντινά μου πρόσωπα για την υπομονή, τη βοήθεια και τη στήριξη που μου πρόσφεραν καθ' όλη τη διάρκεια των σπουδών μου.

# Περιεχόμενα

# Εισαγωγή

## 1.1 Στόχος της Εργασίας

Στόχος της παρούσας διπλωματικής εργασίας είναι να εκμεταλλευτεί τις δυνατότητες που προσφέρουν οι έξυπνες κινητές συσκευές και συγκεκριμένα το λειτουργικό σύστημα Android, για την ανάπτυξη μιας εφαρμογής, η οποία δίνει τη δυνατότητα στο χρήστη της να λαμβάνει δεδομένα και πληροφορίες σχετικά με ένα chip.

Η εφαρμογή που υλοποιήθηκε έχει ως σκοπό την ενημέρωση των χρηστών για πληροφορίες σχετικές με τα στοιχεία ενός chip, αλλά και του datasheet του. Οι πληροφορίες αυτές καλύπτουν ένα ευρύ φάσμα των αναγκών των ηλεκτρονικών και μηχανικών υπολογιστών, καθώς ο μέχρι τώρα τρόπος αναγνώρισης ενός chip ήταν με τη χρήση ενός μεγεθυντικού φακού, την καταγραφή των στοιχείων του και της αναζήτησης στο ίντερνετ. Η εφαρμογή διαθέτει ακόμα λειτουργίες που αποσκοπούν στην διάγνωση και διόρθωση φωτογραφιών που δεν έχουν τυχών παρθεί σωστά από τον χρήστη.

Πιο συγκεκριμένα, η εφαρμογή συλλέγει, επεξεργάζεται και εμφανίζει δεδομένα, τα οποία λαμβάνει από μια φωτογραφία με τη βοήθεια της κάμερας. Επιπλέον δίνει την επιλογή στο χρήστη, να απομονώσει το στοιχείο που επιθυμεί για αναγνώριση ή ακόμα και την επιλογή να βελτιώσει την ποιότητα της εικόνας με σκοπό την βελτίωση των αποτελεσμάτων.

## 1.2 Πρότερες Προσπάθειες

Σήμερα η τεχνητή νοημοσύνη στους ηλεκτρονικούς υπολογιστές είναι σχεδιασμένη να προσεγγίζει την ανθρώπινη συμπεριφορά. Η ανάγνωση, η γραφή, η ακρόαση και η ομιλία είναι μερικά από τα πολλά παραδείγματα. Στις αρχές του 1950, οι επιστήμονες προσπάθησαν αρχικά να καταγράψουν εικόνες από χαρακτήρες και τα κείμενα με μηχανικά μέσα, όπως φαίνεται στην εικόνα.

Στη συνέχεια εφευρέθηκαν οπτικά μέσα περιστρεφόμενων δίσκων και φωτοπολλαπλασιαστής, ένα ιπτάμενος σαρωτής σημείου που έκανε χρήση ενός φακού καθοδικού σωλήνα και ακολούθησαν τα φωτοκύτταρα και οι συστοιχίες από αυτούς [5]. Μεταγενέστερες, το τύμπανο και το επίπεδο σαρωτή εφευρέθηκαν, τα οποία επέτρεψαν στις μηχανές OCR την πλήρη ανίχνευση μιας σελίδας.

W. S. Pike, RCA engineer who helped develop electronic reader, moves its eye over text.

Εικόνα 1: Πρόωρο OCR Μηχάνημα Ανάγνωσης. [5]

Με την εφεύρεση του ψηφιακού - ολοκληρωμένου κυκλώματος, η ταχύτητα σάρωσης και ταχύτητα μετατροπής αυξήθηκε ιδιαίτερα. Στις αρχές της δεκαετίας του 1960, διάφορα λάθη στο OCR εμφανίζονταν λόγο της κακής εκτύπωσης, τις μεγάλες

διακυμάνσεις στις γραμματοσειρές και την τραχιά επιφάνεια του χαρτιού. Το μεγάλο άλμα για την ανάπτυξη εφαρμογών οπτικής αναγνώρισης χαρακτήρων (OCR) επετεύχθη την δεκαετία του 1970. Το Αμερικανικό Εθνικό Ινστιτούτο Προτύπων (ANSI) και η Ευρωπαϊκή Ένωση Κατασκευαστών Ηλεκτρονικών Υπολογιστών (ECMA) σχεδίασαν γραμματοσειρές ειδικά προσαρμοσμένες για OCR, όπως την OCRA και την OCRB [5]. Ο Διεθνής Οργανισμός Τυποποίησης (ISO) ενέκρινε σύντομα τις προσαρμοσμένες γραμματοσειρές [5]. Ως εκ τούτου, επιτεύχθηκε μεγαλύτερη ακρίβεια αναγνώρισης. Με όλα αυτά τα επιτεύγματα, το κόστος για υψηλής ταχύτητας και ακρίβειας σάρωσης OCR είχε αισθητά μειωθεί.

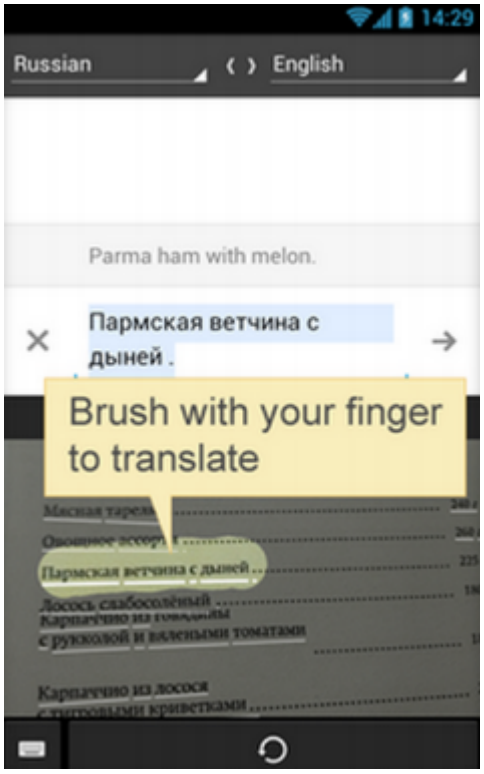
## 1.3 OCR στην καθημερινή ζωή

Το OCR έχει αναπτυχθεί εκτεταμένα για πάνω από 60 χρόνια, και έχει εφαρμοστεί σε διάφορους τομείς [6]. Το OCR βοήθησε τον άνθρωπο να μειώσει ένα τεράστιο ποσό της χειρωνακτικής εργασίας όπως η πληκτρολόγηση.

Τα τελευταία λίγα χρόνια με την άνθηση των υπηρεσιών cloud (νέφους), τις έξυπνες κινητές συσκευές και την ευρεία αναγνώριση των OCR εργαλείων, η οπτική αναγνώριση εικόνων (OCR) δεν παίζει μόνο το ρόλο αποδέσμευσης των ανθρώπων από κόπο και εργατοώρες, αλλά παρέχει καλύτερη εμπειρία στον χρήστη. Τώρα το OCR έχει με τον τρόπο του διεισδύσει στη καθημερινή μας ζωή.

Μερικά παραδείγματα δείχνονται ως ακολούθως.

Google Translate είσοδος (input) από κάμερα. Το 2012, μια λειτουργία OCR προστέθηκε στην υπηρεσία Google Translate, όπως φαίνεται στην εικόνα 2, έτσι ώστε ο χρήστης να μπορεί να μεταφράζει το κείμενο χρησιμοποιώντας μόνο τον φακό της κάμερας του έξυπνου κινητού του.




Εικόνα 2: Google Translate για Android συσκευές πρόσθεσε OCR δυνατότητες [34]

Παρέχει στους χρήστες την ευκολία της γρήγορης εισόδου όταν δεν έχουν ένα προσαρμοστικό πληκτρολόγιο ή στην περίπτωση που το κείμενο είναι πολύ μεγάλο.

### Αυτόματη αναγνώριση πινακίδων κυκλοφορίας (ALPR)

Το OCR βοηθάει τους αστυνομικούς για χρόνια [7].


Εικόνα 3: Ένα παράδειγμα ALPR διεπαφής χρήστη (User Interface UI) [8]

Ο σαρωτής παραπάνω είναι ένα τοποθετημένο σύστημα ALPR σε ένα όχημα, όπως φαίνεται στην εικόνα 3. Βοηθά τους αξιωματικούς ειδοποιώντας τους όταν εντοπίσει όχημα ή οχήματα που αναζητούνται για οποιοδήποτε λόγο στην εκάστοτε περιοχή.

### Χρήση του OCR από τις ταχυδρομικές υπηρεσίες

Η Ταχυδρομική Υπηρεσία των Ηνωμένων Πολιτειών χρησιμοποιεί μηχανές OCR για την ταξινόμηση της αλληλογραφίας από το 1965. Μηχανές με βάση την τεχνολογία που επινοήθηκε κυρίως από την παραγωγικό εφευρέτη Jacob Rabinow.

Η πρώτη χρήση του OCR στην Ευρώπη πραγματοποιήθηκε από το Βρετανικό Γενικό Ταχυδρομείο (GPO).

Το 1965 άρχισε να σχεδιάζεται ένα ολοκληρωμένο τραπεζικό σύστημα, το Εθνικό Giro ( the National Giro), χρησιμοποιώντας την τεχνολογία OCR, μια διαδικασία που έφερε την επανάσταση των συστημάτων πληρωμής λογαριασμών στο Ηνωμένο Βασίλειο.

Η Ταχυδρομική Υπηρεσία του Καναδά χρησιμοποιεί συστήματα OCR από το 1971. Συστήματα OCR που διαβάζουν - αναγνωρίζουν το όνομα και τη διεύθυνση του παραλήπτη, κατά την πρώτη ταξινόμηση στο κέντρο διαλογής, και στην συνέχεια τυπώνουν ένα barcode. Η δρομολόγηση στην συνέχεια του φακέλου γίνεται με βάση τον ταχυδρομικό κώδικα. Έτσι αποφεύγεται η σύγχυση με το πεδίο διεύθυνσης αναγνώσιμη από τον άνθρωπο που μπορεί να βρίσκεται οπουδήποτε στο έγγραφο. Χρησιμοποιείται ειδικό μελάνι (πορτοκαλί στο ορατό φως) το οποίο είναι απόλυτα καθαρά ορατό κάτω από το υπεριώδες φως. Οι φάκελοι μπορούν στη συνέχεια να υποβληθούν σε επεξεργασία με εξοπλισμό που βασίζεται σε απλούς αναγνώστες barcode (barcode readers).

## 1.4 Συνεισφορά της Εργασίας

Η προσπάθεια που έγινε είναι μια πρώτη επιστημονική προσέγγιση πάνω σε ένα τεχνικό ζήτημα που δύναται να χρησιμοποιηθεί από ανθρώπους του κλάδου, δίνοντας τους άμεση και γρήγορη πρόσβαση σε μια ενιαία βάση δεδομένων ολοκληρωμένων στοιχείων με την απλή χρήση του κινητού τους. Η επίλυση των προβλημάτων που αντιμετωπίστηκαν κατά την υλοποίηση αυτής της εργασίας έγινε με βάση ήδη υπαρχόντων αργαλειών, βιβλιοθηκών, και αλγορίθμων και όχι με βάση την ανάπτυξη νέων

## 2.1 Ενσωματωμένα συστήματα

Ένα ενσωματωμένο σύστημα είναι ένα υπολογιστικό σύστημα με μια ειδική λειτουργία μέσα σε ένα μεγαλύτερο μηχανικό ή ηλεκτρικό σύστημα, συχνά προορίζεται για συστήματα που απαιτούν υπολογισμούς πραγματικού χρόνου. Ένα ενσωματωμένο σύστημα είναι μέρος μιας πλήρους συσκευής συμπεριλαμβανομένου του υλικού και του μηχανικού μέρους. Τα ενσωματωμένα συστήματα ελέγχουν πολλές συσκευές που χρησιμοποιούνται ευρέως σήμερα. Ενενήντα οκτώ τοις εκατό όλων των μικροεπεξεργαστών κατασκευάζονται ως κομμάτια των ενσωματωμένων συστημάτων.

Τα σύγχρονα ενσωματωμένα συστήματα βασίζονται συχνά σε μικρο-ελεγκτές (δηλαδή επεξεργαστές με ενσωματωμένη μνήμη ή περιφερειακές διασυνδέσεις),αλλά οι απλοί μικροεπεξεργαστές (χρήση εξωτερικών τσιπ για τη μνήμη και τα κυκλώματα περιφερειακής διασύνδεσης) έχουν επίσης κοινά, ειδικά σε πιο πολύπλοκα συστήματα. Σε κάθε περίπτωση, ο επεξεργαστής που χρησιμοποιείται μπορεί να είναι τύπου γενικής χρήσης ή ειδικά σχεδιασμένος για την εφαρμογή.

Δεδομένου ότι το ενσωματωμένο σύστημα προορίζεται για συγκεκριμένες εργασίες αφού σκοπός είναι να εκτελούν σωστά και αποδοτικά την εφαρμογή για την οποία έχουν σχεδιαστεί, οι μηχανικοί που τα σχεδιάζουν μπορούν να τα βελτιστοποιούν με σκοπό την μείωση του μεγέθους και του κόστους του προϊόντος και παράλληλα την αύξηση της αξιοπιστίας και της απόδοσης του συστήματος.

Σχεδόν κάθε αυτοκίνητο που βγαίνει από τη γραμμή παραγωγής αυτές τις μέρες κάνει χρήση των ενσωματωμένων συστημάτων σε μία μορφή ή σε κάποια άλλη. Και αυτό δεν είναι ένα παράδειγμα, κονσόλες παιχνιδιών(πχ Sony PlayStation, Xbox), οικιακές συσκευές(κουζίνα, ψυγείο, τηλεοράσεις), φορητές συσκευές, όπως ψηφιακά ρολόγια και MP3 players, ψηφιακή φωτογραφική μηχανή, tablets, κινητά τηλέφωνα είναι μερικά από τα πολλά παραδείγματα.

## 2.2 Έξυπνα τηλέφωνα

Το 1973, ο Θεόδωρος Γεώργιος Παρασκευάκος κατοχύρωσε με δίπλωμα ευρεσιτεχνίας τις έννοιες του συνδυασμού νοημοσύνης, της επεξεργασίας δεδομένων και των οθονών οπτικής απεικόνισης με τα τηλέφωνα, περιγράφοντας έτσι τις κοινές πλέον δραστηριότητες των τραπεζικών συναλλαγών και την πληρωμή λογαριασμών κοινής ωφελείας μέσω τηλεφώνου.

Το πρώτο κινητό τηλέφωνο με ενσωματωμένα χαρακτηριστικά PDA ήταν ένα πρωτότυπο IBM που αναπτύχθηκε το 1992 και η επίδειξή του έγινε το ίδιο έτος στην εμπορική έκθεση βιομηχανίας πληροφορικής COMDEX. Μια ανανεωμένη έκδοση του προϊόντος διατέθηκε στο εμπόριο για τους καταναλωτές στις 16 Αυγούστου 1994 από την BellSouth, με την επωνυμία Simon Personal Communicator. Το Simon ήταν η πρώτη συσκευή που μπορεί να αναφέρεται ως smartphone, έστω και αν αυτός ο όρος δεν είχε ακόμη επινοηθεί. Ο όρος smartphone δεν εμφανίστηκε μέχρι το 1997, όταν η Ericsson περιέγραψε το GS 88 "Penelope" ως Smart Phone (Έξυπνο Τηλέφωνο)

Μία από τις πιο σημαντικές διαφορές μεταξύ των smartphones και των απλών κινητών τηλεφώνων είναι ότι οι προηγμένες διεπαφές προγραμματισμού εφαρμογών (APIs) στα smartphones σχετικά με τη λειτουργία τρίτων εφαρμογών μπορούν να επιτρέψουν σε αυτές τις εφαρμογές να έχουν καλύτερη ενσωμάτωση στο λειτουργικό σύστημα και στο hardware του τηλεφώνου απ' ότι συμβαίνει συνήθως στα απλά κινητά τηλέφωνα.

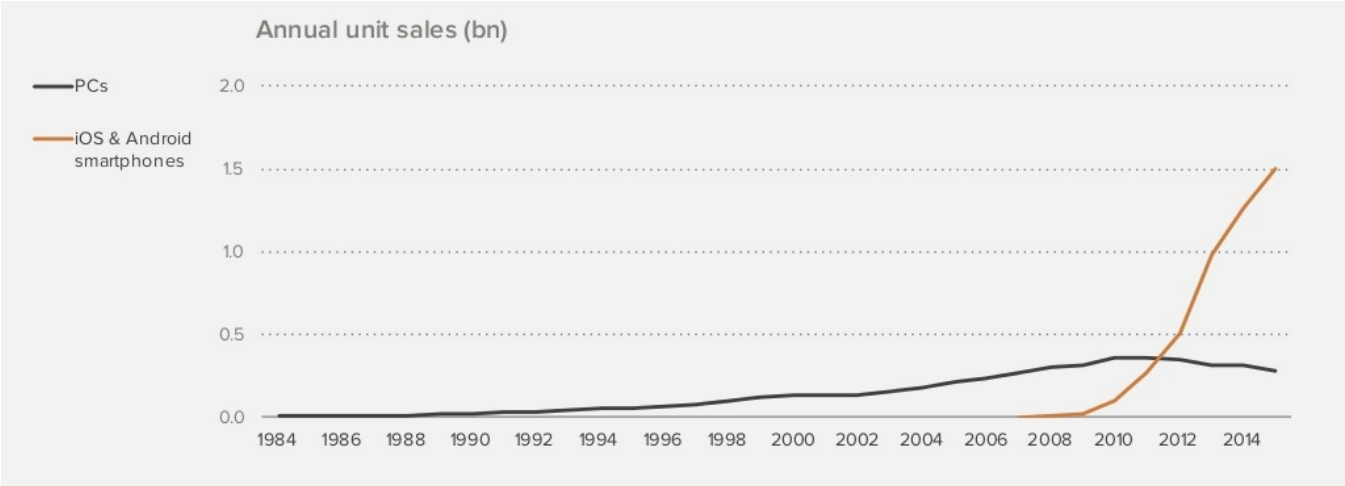
Το 2007 ήταν η χρονιά που θα έφερνε την επανάσταση στα έξυπνα τηλέφωνα καθώς η Apple παρουσίασε το πρώτο iPhone.

Ένα χρόνο αργότερα η Open Handset Alliance ανακοινώνουν την πλατφόρμα Android και το πρώτο κινητό που χρησιμοποίησε αυτή τη πλατφόρμα ήταν το HTC Dream.

Για αρκετά χρόνια, η ζήτηση των έξυπνων τηλεφώνων ξεπέρασε το υπόλοιπο της αγοράς κινητής τηλεφωνίας. Σύμφωνα με μια έρευνα του 2012, περίπου το ήμισυ των καταναλωτών κινητών τηλεφώνων στις ΗΠΑ έχουν στην κατοχή τους smartphones και θα μπορούσαν να αντιπροσωπεύουν περίπου το 70% του συνόλου των κινητών συσκευών των ΗΠΑ από το 2013. Στις αρχές του 2013, οι πωλήσεις smartphone ξεπέρασαν παγκοσμίως εκείνες των κινητών τηλεφώνων παλαιότερων στυλ.

Με την ραγδαία ανάπτυξη του λεγόμενου IoT (Internet of Things), με τα smartphone να γίνονται ολοένα και ισχυρότερα με περισσότερη υπολογιστική ισχύ και RAM να προστίθεται συνεχώς και την ήδη ανάπτυξη λογισμικών όπως Ubuntu Touch ή Windows 10, η δική μου προσωπική εκτίμηση είναι ότι μέχρι το 2022 τα smartphone θα αντικαταστήσουν τον προσωπικό υπολογιστή (PC) όπως τον ξέρουμε τώρα. Στο μέλλον θα έχουμε τον προσωπικό μας υπολογιστή στην τσέπη και θα το συνδέουμε σε

κάποιο monitor ή TV. για να εποφελούμαστε των δυνατοτήτων που μας δίνει μια μεγαλύτερη οθόνη από ότι αυτή του smartphone.




Εικόνα 4: Πωλήσεις έξυπνων κινητών έναντι συμβατικών ηλεκτρονικών υπολογιστών. [35]

## 2.3 Λειτουργικό σύστημα Android

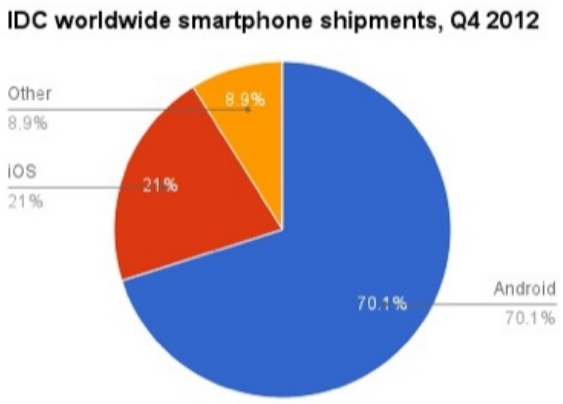
Το Android είναι λειτουργικό σύστημα για συσκευές κινητής τηλεφωνίας το οποίο τρέχει τον πυρήνα του λειτουργικού Linux. Το λογότυπο για το λειτουργικό σύστημα Android είναι ένα ρομπότ σε χρώμα πράσινου μήλου και σχεδιάστηκε από τη γραφίστρια Ιρίνα Μπλόκ.

Αρχικά αναπτύχθηκε από την Google και έπειτα συνεχίστηκε από τον οργανισμό Open Handset Alliance μιας κοινοπραξίας 48 τηλεπικοινωνιακών εταιριών, εταιριών λογισμικού καθώς και κατασκευής hardware. Επιτρέπει στους κατασκευαστές λογισμικού να συνθέτουν κώδικα με την χρήση της γλώσσας προγραμματισμού Java, ελέγχοντας την συσκευή μέσω βιβλιοθηκών λογισμικού ανεπτυγμένων από την Google.

Το Android είναι κατά κύριο λόγο σχεδιασμένο για συσκευές με οθόνη αφής, όπως τα έξυπνα τηλεφωνα και τα τάμπλετ, με διαφορετικό περιβάλλον χρήσης για τηλεοράσεις (Android TV), αυτοκίνητα (Android Auto) και ρολόγια χειρός (Android Wear).

Η πρώτη παρουσίαση της πλατφόρμας Android έγινε στις 5 Νοεμβρίου 2007, στις 23 Σεπτεμβρίου 2008 κυκλοφόρησε η πρώτη έκδοση Android, ενώ στις 19 Δεκεμβρίου 2014 ανακοινώθηκε η τελευταία μέχρι στιγμής έκδοση που είναι η 7.0 με την ονομασία Nougat

Το Android είναι το πιο ευρέως διαδεδομένο λογισμικό στον κόσμο. Οι συσκευές με Android έχουν περισσότερες πωλήσεις από όλες τις συσκευές Windows, iOS και Mac OS X μαζί.



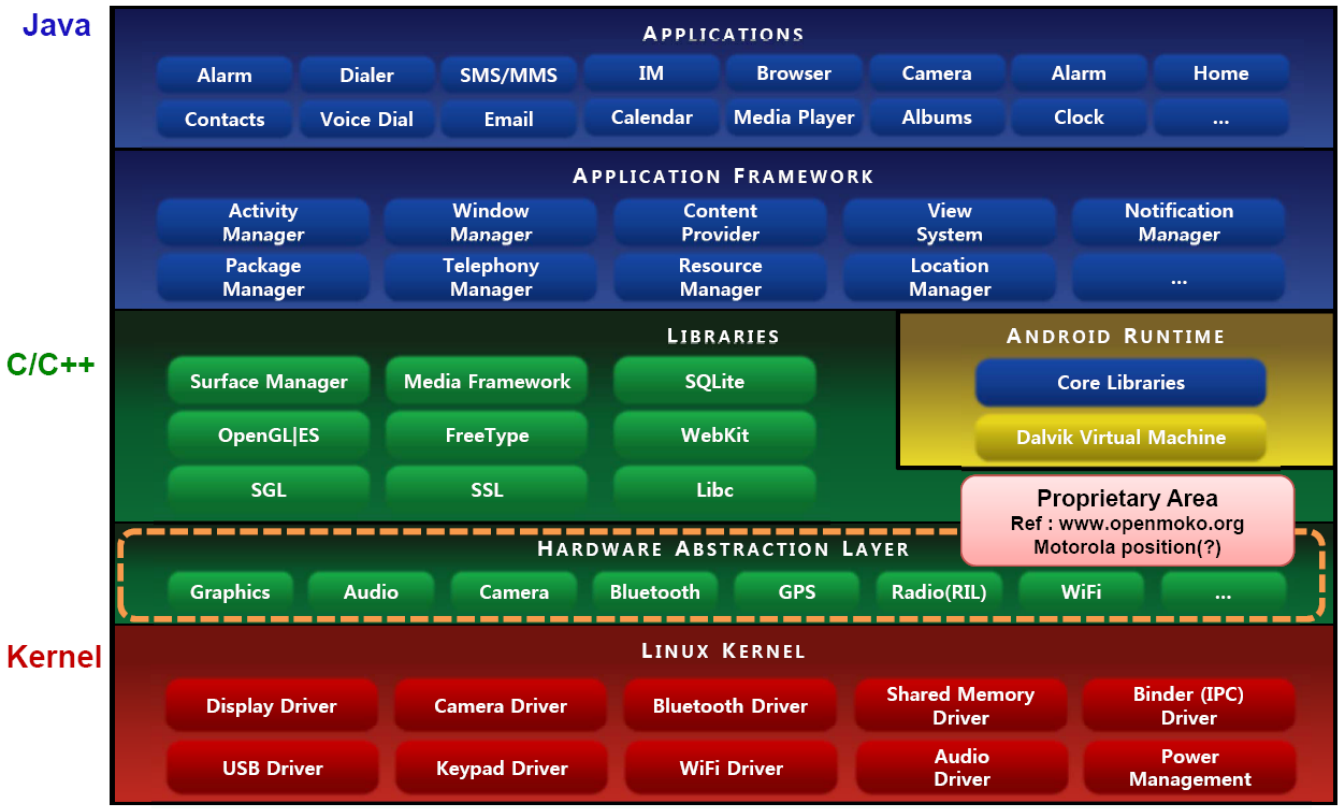

Εικόνα 5: Ποσοστό πωλήσεων Android το 2012 [1]

## 2.3.1 Η Αρχιτεκτονική του Android

Η εικόνα 6 περιγράφει την τρέχουσα (πολυεπίπεδη) αρχιτεκτονική του Android. Ο τροποποιημένος πυρήνας του Linux λειτουργεί ως HAL, και παρέχει τα drivers της συσκευής, τη διαχείριση της μνήμης, την διαχείριση των διαδικασιών, καθώς και τις λειτουργίες που απαιτούνται για την δικτύωση. Το επίπεδο των βιβλιοθήκών διασυνδέεται μέσω Java (η οποία αποκλίνει από την παραδοσιακή σχεδίαση του Linux). Είναι σε αυτό το επίπεδο που η συγκεκριμένη libc Android (Bionic) βρίσκεται. Το πάνω επίπεδο χειρίζεται τα παράθυρα διεπαφής του χρήστη (UI). Το επίπεδο Android runtime περιέχει το Virtual Machine Dalvik (DVM) και τις βασικές βιβλιοθήκες (όπως η Java ή IO). Οι περισσότερες λειτουργίες που είναι διαθέσιμες στο Android παρέχονται μέσω των βασικών βιβλιοθηκών(core libraries). Το επίπεδο εφαρμογών (application framework) περιέχει τη διεπαφή API (API interface). Σε αυτό το επίπεδο , ο “διαχειριστής δραστηριότητας” ρυθμίζει τον κύκλο ζωής της εφαρμογής. Οι πάροχοι περιεχομένου (content providers) επιτρέπουν στις εφαρμογές είτε να έχουν πρόσβαση σε δεδομένα από άλλες εφαρμογές ή να μοιραστούν τα δικά τους δεδομένα. Ο διαχειριστής πόρων (resource manager) παρέχει πρόσβαση σε πόρους μη-κώδικα (όπως γραφικά), ενώ ο διαχειριστής κοινοποίησης (notification manager) επιτρέπει στις εφαρμογές να


Εικόνα 6: Η Αρχιτεκτονική του Android [36]

εμφανίζουν προσαρμοσμένες ειδοποιήσεις. Στην κορυφή του επίπεδο εφαρμογών (application framework) είναι η προσαρμοσμένη διεπαφή, οι ενσωματωμένες εφαρμογές, καθώς και οι εφαρμογές του χρήστη. Πρέπει να επισημανθεί ότι οι εφαρμογές του χρήστη μπορούν να αντικαταστήσουν μια ενσωματωμένη εφαρμογή, και ότι κάθε εφαρμογή Android τρέχει στο δικό της επίπεδο διαδικασιών, μέσα στο δικό της DVM(Dalvik Virtual Machine). Τα περισσότερα από αυτά τα βασικά περιεχόμενα του Android συζητούνται περαιτέρω (με περισσότερες λεπτομέρειες) μέσα στις επόμενες ενότητες της παρούσας εργασίας.




Εικόνα 7: Η Αρχιτεκτονική του Android [37]

## 2.3.2 Η Εικονική μηχανή Dalvik (DVK)

Τα Συστήματα βασισμένα στο Android χρησιμοποιούν τη δική τους εικονική μηχανή (VM), η οποία είναι γνωστή ως Virtual Machine Dalvik (DVM) [19]. Η εικονική μηχανή DVM χρησιμοποιεί ειδικό byte-code, ως εκ τούτου η μητρική Java byte-code δεν μπορεί άμεσα να εκτελεστεί σε συστήματα Android. Η κοινότητα του Android παρέχει ένα εργαλείο (DX) που επιτρέπει τη μετατροπή των Java κλάσεων σε Dalvik εκτελέσιμα (dex). Η υλοποίηση της εικονικής μηχανής DVM είναι ιδιαίτερα

βελτιστοποιημένη προκειμένου να εκτελείται το ίδιο καλά και όσο το δυνατόν αποτελεσματικότερα σε κινητές συσκευές που είναι συνήθως εξοπλισμένες με μέτρια (αυτές τις μέρες χρησιμοποιούνται διπύρηνοι, ή τετραπύρηνοι επεξεργαστές) υποσύστημα CPU, έχουν περιορισμένους πόρους μνήμης, δεν έχουν χώρο για OS swap, και η μπαταρία έχει περιορισμένη χωρητικότητα. The DVM έχει υλοποιηθεί με τέτοιο τρόπο ώστε να επιτρέπει σε μια συσκευή να εκτελεί πολλαπλές εικονικές μηχανές (VM) με ένα μάλλον αποτελεσματικό τρόπο. Πρέπει επίσης, να επισημανθεί ότι η DVM στηρίζεται στον τροποποιημένο πυρήνα του Linux για ενδεχόμενες βελτιώσεις ως προς την διαχείριση μνήμης και τις χαμηλού επιπέδου λειτουργιών. Με το Android 2.2, τέθηκαν σε εφαρμογή κάποιες σημαντικές αλλαγές στην υποδομή του JVM. Μέχρι την έκδοση 2.2, η JVM ήταν πρακτικά ένας διερμηνέας, παρόμοιος με εκείνον που ήταν η αρχική λύση JVM και αναπτύχθηκε με την Java 1.0. Ενώ η Android λύση πάντα εμφανιζόταν ως ένας πολύ αποτελεσματικός διερμηνέας, ήταν απλά ακόμα ένας διερμηνέας και, ως εκ τούτου, δεν δημιουργήθηκε πηγαίος κώδικας. Με την κυκλοφορία του Android 2.2, ένας μεταγλωττιστής just-in-time (JIT) compiler όπως τον είπαν ενσωματώθηκε, ο οποίος μεταφράζει την εικονική μηχανή Dalvik byte-code σε έναν πολύ πιο αποδοτικό κώδικα μηχανής (παρόμοιο με ένα μεταγλωττιστή C). Επί του παρόντος αυτό εφαρμόστηκε μετά την έκδοση Android 4 (Ice Cream Sandwich) και 4.1 / 4.2 (Jelly Bean). Επιπλέον JIT και garbage collections (GC) χαρακτηριστικά έχουν αναπτυχθεί μέχρι την τωρινή έκδοση του Android.

### 2.3.3 Πλατφόρμα προορισμού - Σύνολο οδηγιών

Για χάρη απλοποίησης της συζήτησης, η δήλωση που γίνεται εδώ είναι ότι οι περισσότερες από τις συσκευές που βασίζονται σε Linux 2.6 είναι συστήματα x86, ενώ τα περισσότερα κινητά τηλέφωνα είναι βασισμένα στα συστήματα ARM. Ενώ η ARM αντιπροσωπεύει ένα 32-bit υπολογιστή με μειωμένο σύνολο εντολών (RISC) αρχιτεκτονικής συνόλου εντολών, τα συστήματα x86 βασίζονται κυρίως σε σύνολο περίπλοκων εντολών υπολογιστών (CISC) αρχιτεκτονικής. Σε γενικές γραμμές, μπορεί να ειπωθεί ότι η αρχιτεκτονική ARM (RISC) εκτελεί πιο απλές (αλλά περισσότερες) οδηγίες σε σύγκριση με ένα σύστημα x86 (CISC). Όπως ήδη αναφέρθηκε, η μνήμη είναι περιορισμένη στις κινητές συσκευές λόγω του μεγέθους, του κόστους και τους περιορισμούς ισχύος. Η ARM αρχιτεκτονική αντιμετωπίζει αυτά τα θέματα με την παροχή ενός 2ου σετ εντολών 16-bit που μπορούν να εκτελούνται παράλληλα με τις συνήθεις 32-bit ARM οδηγίες. Αυτό το πρόσθετο σύνολο οδηγιών μπορεί να μειώσει το μέγεθος του κώδικα μέχρι και 30% (σε βάρος κάποιων περιορισμών όσο αναφορά την απόδοση). Επομένως από μια συνολική προοπτική των συστημάτων, η ενσωμάτωση των παραπάνω συνόλων εντολών μπορεί να θεωρηθεί ως μια προσπάθεια συμβιβασμού. Σε σύγκριση με τους επεξεργαστές x86, ο σχεδιασμός του ARM αποκαλύπτει ότι έχουν σχεδιαστεί με ισχυρή έμφαση στην χαμηλή κατανάλωση ενέργειας, γεγονός που τους καθιστά κατάλληλους για φορητές συσκευές [20]. Όπως συμβαίνει με οποιοδήποτε υπολογιστή, ο επεξεργαστής παίζει σημαντικό ρόλο στην

απόδοση του συνολικού συστημάτος. Κατά τις πρώτες ημέρες του Android, οι περισσότερες συσκευές είχαν τον ίδιο επεξεργαστή ARM Qualcomm και, ως εκ τούτου, η απόδοση τους ήταν αρκετά συγκρίσιμη. Με την διανομή του Motorola Droid, η νέα γενιά των chipsets εισήχθη (υποστηρίζοντας βελτιωμένους επεξεργαστές γραφικών). Σήμερα, υπάρχουν και αναπτύσσονται 3 κύρια chipsets για τις συσκευές Android. Εικονογραφώντας μερικά η HTC χρησιμοποιεί τον Qualcomm Snapdragon, η Motorola χρησιμοποιεί τον Texas Instruments OMAP, ενώ η Samsung έχει σχεδιάσει και χρησιμοποιεί το δικό της chipset με κωδικό όνομα Exynos. Πρέπει να τονιστεί ωστόσο ότι και οι 3 προαναφερθέντες επεξεργαστές βασίζονται στην αρχιτεκτονική ARM Cortex-A8 (με τον κάθε πωλητή- εταιρία να εφαρμόζει μικρο-ρυθμίσεις προσφέροντας μοναδικά χαρακτηριστικά).

### 2.3.4 Πυρήνας και διεργασίες κατά την εκκίνηση

Είναι υψίστης σημασίας να τονιστεί ότι ενώ το Android βασίστηκε αρχικά στον Linux 2.6 kernel πλέον Linux 3.4, το Android δεν χρησιμοποιεί ένα πρότυπο πυρήνα του Linux. Ως εκ τούτου, μια συσκευή Android δεν πρέπει να χαρακτηρίζεται ως ένα λειτουργικό σύστημα Linux.

Μερικές από τις ειδικές βελτιώσεις στον πυρήνα (kernel) του Android περιλαμβάνουν:

- alarm driver (παρέχει χρονόμετρα για να ξυπνά- ενεργοποιεί τις συσκευές)
- shared memory driver (παρέχει κοινόχρηστη μνήμη)
- binder (για την επικοινωνία μεταξύ διεργασιών)
- power management (η οποία λαμβάνει μια πιο επιθετική προσέγγιση από τη λύση του Linux PM)
- low memory killer (LMK)
- kernel debugger and logger

Κατά τη διάρκεια της διαδικασίας εκκίνησης του Android, ο πυρήνας Linux του Android απαιτεί πρώτα τη διαδικασία init (σε σύγκριση με το πρότυπο Linux, τίποτα ασυνήθιστο εδώ). Η διαδικασία init πραγματοποιεί πρόσβαση στα αρχεία init.rc και init.device.rc (init.device.rc είναι η συγκεκριμένη συσκευή, η κάθε συγκεκριμένη συσκευή). Έξω από το αρχείο init.rc, μια διαδικασία που αναφέρεται ως zygote έχει ξεκινήσει. Η zygote διεργασία φορτώνει τις βασικές κλάσεις της Java, και εκτελεί τα αρχικά βήματα επεξεργασίας. Αυτές οι κλάσεις Java μπορούν να επαναχρησιμοποιηθούν από τις Android εφαρμογές και, ως εκ τούτου, το βήμα αυτό επισπεύδει τη συνολική διαδικασία εκκίνησης. Μετά την αρχική διεργασία φόρτωσης, το zygote παραμένει άεργο σε ένα socket και περιμένει περαιτέρω αιτήματα. Κάθε εφαρμογή Android τρέχει στο δικό της περιβάλλον διαδικασιών. Ένας ειδικός driver που αναφέρεται ως binder επιτρέπει την (αποτελεσματική) επικοινωνία μεταξύ διεργασιών (IPC). Τα πραγματικά αντικείμενα αποθηκεύονται στην κοινόχρηστη μνήμη. Με τη χρησιμοποίηση κοινόχρηστης μνήμη η

επικοινωνία μεταξύ διεργασιών (IPC) γίνεται βελτιστοποιημένη καθώς μικρότερος αριθμός δεδομένων πρέπει να μεταφερθεί. Σε σύγκριση με τα περισσότερα περιβάλλοντα Linux ή UNIX, το Android δεν μπορεί να παρέχει κανένα χώρο για swap. Ως εκ τούτου, η ποσότητα της εικονικής μνήμης διέπεται από την ποσότητα της φυσικής μνήμης που διατίθεται στην συσκευή [21]

## 2.3.5 Η βιβλιοθήκη Bionic

Σε σύγκριση με το Linux, τα Androids ενσωματώνουν τη δική τους βιβλιοθήκη C (Bionic) [22]. Η βιβλιοθήκη Bionic δεν είναι συμβατή με την Linux glibc. Σε σύγκριση με την glibc, η βιβλιοθήκη Bionic έχει ένα μικρότερο footprint μνήμης. Η Bionic βιβλιοθήκη περιλαμβάνει μια ειδική υλοποίηση των νημάτων (threads) η οποία 1ον, βελτιστοποιεί την κατανάλωση μνήμης του κάθε νήματος(thread) και 2ον, μειώνει το χρόνο εκκίνησης ενός νέου νήματος(thread). Το Android παρέχει πρόσβαση στον πυρήνα του [23]. Ως εκ τούτου, δίνεται στον χρήστη η δυνατότητα να μπορεί να αλλάξει δυναμικά την συμπεριφορά του πυρήνα. Μόνο συγκεκριμένες διεργασίες (processes/threads) ωστόσο διαθέτουν τα κατάλληλα δικαιώματα επιτρέποντας να τροποποιήσουν αυτές τις ρυθμίσεις. Ερωτήματα ως προς την ασφάλεια δημιουργούνται αλλά αυτή διατηρείται εκχωρώντας ένα ζεύγος κλειδιών σε κάθε εφαρμογή. Ένα μοναδικό αναγνωριστικό χρήστη (user ID-UID) και ένα αναγνωριστικό ομάδας (group ID-GID). Καθώς οι φορητές συσκευές συνήθως προορίζονται να χρησιμοποιούνται από μόνο έναν χρήστη (σε αντίθεση με τα περισσότερα συστήματα Linux), οι ρυθμίσεις UNIX / Linux / etc / passwd **και** / etc / έχουν αφαιρεθεί.

Επιπλέον, (για την ενίσχυση της ασφάλειας), τα / etc / services αντικαταστάθηκαν από μια λίστα υπηρεσιών (έτσι συντηρείται μέσα στο ίδιο το εκτελέσιμο).

Συνοψίζοντας, η βιβλιοθήκη C Android είναι ιδιαίτερα κατάλληλη για να λειτουργεί κάτω από τις περιορισμένες δυνατότητες των CPU και την περιορισμένη μνήμη στοιχεία κοινά στις συσκευές Android [23]. Επιπρόσθετα, ειδικές μορφές ασφαλείας έχουν σχεδιαστεί και εφαρμοστεί για να διασφαλιστεί η ακεραιότητα του συστήματος.

## 2.3.6 Μέσα αποθήκευσης και σύστημα αρχείων

Όταν πρόκειται για τη διαμόρφωση και τη δημιουργία κινητών συσκευών, οι παραδοσιακοί σκληροί δίσκοι είναι σε γενικές γραμμές πολύ μεγάλοι (σε μέγεθος), πολύ εύθραυστοι, και καταναλώνουν πάρα πολλή ενέργεια χαρακτηριστικά που τους καθιστούν άχρηστους. Σε αντίθεση, οι συσκευές μνήμης flash συνήθως παρέχουν μια (σχετικά) γρήγορη πρόσβαση γραφής και ανάγνωσης καθώς και καλύτερη (κινητική) αντοχή στην κρούση σε σύγκριση με τους σκληρούς δίσκους. Θεμελιακά, δύο διαφορετικοί τύποι συσκευών μνήμης flash είναι συνηθισμένοι, χαρακτηρίζονται ως

NAND και NOR βασισμένες. Ενώ σε γενικές γραμμές οι NOR βασισμένες παρέχουν χαμηλή πυκνότητα, χαρακτηρίζονται ως (σχετικά) αργές στην γραφή και γρήγορες στην ανάγνωση δεδομένων. Από την άλλη πλευρά, οι μνήμες flash που βασίζονται σε NAND προσφέρονται σε χαμηλό κόστος, με υψηλή πυκνότητα, και επισημαίνονται ως (σχετικά) γρήγορες στην γραφή και αργές στην ανάγνωση δεδομένων.

Ορισμένα ενσωματωμένα συστήματα χρησιμοποιούν μνήμες flash τύπου NAND για την αποθήκευση δεδομένων, και τύπου NOR για το εκτελέσιμο περιβάλλον. Από τη σκοπιά του συστήματος αρχείων, από το Android έκδοση 2.3 και έπειτα, το (γνωστό) σύστημα Linux αρχείων ext4 χρησιμοποιείται. Πριν από το σύστημα αρχείων ext4, το Android χρησιμοποιούσε κυρίως YAFFS (ένα ακόμη σύστημα αρχείων για flash). Το σύστημα αρχείων YAFFS είναι γνωστό ως το πρώτο NAND βελτιστοποιημένο Linux flash σύστημα αρχείων. Ορισμένοι πάροχοι προϊόντων Android (όπως η Archos με ext3 στην έκδοση Android 2.2) αντικατέστησαν το πρότυπο σύστημα αρχείων Archos με ένα άλλο σύστημα αρχείων της επιλογής τους. Όσο για την παρούσα εργασία, το μέγιστο μέγεθος της κάθε εφαρμογής Android ισοδυναμεί με ένα χαμηλό 2-ψήφιο αριθμό MB, το οποίο σε σύγκριση με τα πραγματικά συστήματα που βασίζονται στον πυρήνα Linux πρέπει να θεωρείται πολύ μικρό. Αυτό σημαίνει ότι οι απαιτήσεις της μνήμης και του τρόπου διαχείρισης των αρχείων (από την άποψη του μεγέθους - όχι από τη σκοπιά της ακεραιότητας των δεδομένων) είναι πολύ διαφορετικές για συσκευές που βασίζονται στο λειτουργικό σύστημα Android σε σχέση με τα περισσότερα συστήματα Linux.

### 2.3.7 Διαχείριση Ισχύος

Στον χώρο της μάχης της κινητής τηλεφωνίας, η διαχείριση ενέργειας είναι προφανώς υψίστης σημασίας. Αυτό δεν σημαίνει όμως ότι η διαχείριση ενέργειας θα πρέπει να παραμεληθεί σε οποιοδήποτε άλλο σύστημα. Ως εκ τούτου, η διαχείριση ενέργειας σε οποιοδήποτε πληροφορικό σύστημα, με οποιοδήποτε λειτουργικό σύστημα θεωρείται αναγκαία λόγω της συνεχούς αυξανόμενης ζήτησης ηλεκτρικής ενέργειας από τα σημερινά συστήματα πληροφορικής. Για παράδειγμα, για τη μείωση και τη διαχείριση της κατανάλωσης ενέργειας, τα συστήματα που βασίζονται στο Linux παρέχουν χαρακτηριστικά εξοικονόμησης ενέργειας, όπως το ρολόι gating, η κλιμάκωση της τάσης, η ενεργοποίηση της λειτουργίες ύπνου(sleep mode) ή απενεργοποίηση της προσωρινής μνήμης(memory cache). Κάθε ένα από αυτά τα χαρακτηριστικά μειώνει την κατανάλωση ενέργειας του συστήματος (συνήθως σε βάρος της αυξημένης καθυστέρησης (latency)). Τα περισσότερα συστήματα που βασίζονται στο Linux διαχειρίζονται την κατανάλωση ενέργεια μέσω των ρυθμίσεων παραμέτρων (για προχωρημένους) και το Power Interface (ACPI). Τα συστήματα βασισμένα στο Android παρέχουν τη δική τους υποδομή για διαχείριση της ενέργειας (με την ένδειξη PowerManager) που σχεδιάστηκε με βάση την παραδοχή ότι ο επεξεργαστής δεν θα πρέπει να καταναλώνει ισχύ, αν οι εφαρμογές ή οι υπηρεσίες δεν απαιτούν ενέργεια.

Το Android απαιτεί οι εφαρμογές και οι υπηρεσίες να ζητούν υπολογιστικούς πόρους μέσω “wake” κλειδαριών (wake locks) μέσο του Android application framework και των βασικών βιβλιοθηκών του Linux. Εάν δεν υπάρχουν ενεργές κλειδαριές (wake locks) το Android θα τερματίζει τον επεξεργαστή.

### 2.3.8 Οι Εφαρμογές Android

Οι Android εφαρμογές ομαδοποιούνται σε ένα πακέτο Android (.apk) μέσω του εργαλείου Android Asset Packaging Tool (AAPT). Για την βελτιστοποίηση της διαδικασίας ανάπτυξης, η Google παρέχει τα Android Development Tools (ADT). Η ADT απλοποιεί τη μετατροπή από κλάσεις (classes) σε dex αρχεία, και δημιουργεί το .apk κατά τη διάρκεια της ανάπτυξης. Σε μια πολύ απλοποιημένη μορφή, γενικά οι Android εφαρμογές αποτελούνται από:

- Activities (απαιτούμενες για τη δημιουργία μιας οθόνης για μια εφαρμογή όπου ο χρήστης αλληλεπιδρά - κλάσεις (classes ) με ένα UI).
- Intents (χρησιμοποιείται για τη μεταφορά του ελέγχου από τη μία δραστηριότητα(Activity) στην άλλη).
- Services (classes χωρίς ένα UI, ώστε να μπορούν να εκτελούνται στο παρασκήνιο).
- Content Providers (Επιτρέπει στις εφαρμογές να μοιράζονται πληροφορίες με άλλες εφαρμογές).

## 2.4 Οπτική αναγνώριση χαρακτήρων (OCR)

Οι περισσότεροι άνθρωποι αρχίζουν να μαθαίνουν ανάγνωση και γραφή κατά τα πρώτα χρόνια της εκπαίδευσης. Εφόσον έχουν ολοκληρώσει τη βασική εκπαίδευση, οι άνθρωποι θα πρέπει να έχουν αποκτήσει τις απαραίτητες δεξιότητες γραφής και ανάγνωσης.

Ανεξάρτητα από τις περιπτώσεις φανταχτερού στυλ γραμματοσειράς ή λέξεις με ορθογραφικά λάθη ή περιπτώσεις κατακερματισμένων σημείων και απεικονίσεις, οι περισσότεροι άνθρωποι είναι σε θέση να διαβάσουν και να κατανοήσουν το περιεχόμενο κάνοντας χρήση της εμπειρίας και του περιεχομένου. Σε σχέση με τις ανθρώπινες δεξιότητες ανάγνωσης, υπάρχει ακόμα πολύς δρόμος ώστε οι μηχανές να αναγνωρίζουν κείμενο σε ανταγωνιστικό επίπεδο με τους ανθρώπους.

### 2.4.1 Σημασία του OCR

Τα συστήματα οπτικής αναγνώρισης χαρακτήρων (OCR) είναι εύκολα στη χρήση και μπορούν μακροπρόθεσμα να αντικαταστήσουν την επικοινωνία του ανθρώπου με τη μηχανή (πχ Η\Υ) μέσω πληκτρολογίου και να βοηθήσει στον αυτοματισμό των διαδικασιών στο γραφείο με όφελος την τεράστια εξοικονόμηση χρόνου και ανθρώπινης προσπάθειας.

Τέτοια συστήματα επιτρέπουν την επιθυμητή χειραγώγηση του σαρωμένου κειμένου, καθώς η είσοδος είναι χαρακτήρες σε τυπωμένη μορφή ενώ η έξοδος είναι κωδικοποιημένη σε μορφή ASCII ή κάποια άλλη κωδικοποιημένη μορφή χαρακτήρα που η μηχανή καταλαβαίνει και μπορεί να χειριστεί. Για μια συγκεκριμένη γλώσσα που βασίζεται σε κάποιο αλφάβητο, τεχνικές OCR είτε αποσκοπούν στο τυπωμένο κείμενο ή χειρόγραφο κείμενο. Η παρούσα εργασία έχει ως στόχο τη πρώτη μορφή.

Η Αναγνώριση χαρακτήρων σε τυπωμένη μορφή δεν είναι τόσο απαιτητική όσο η αναγνώριση χειρόγραφου κείμενου. Το δεύτερο είναι μία από τις προκλήσεις της κοινότητα στους τομείς της έρευνας και της αναγνώρισης προτύπων. Σε γενικές γραμμές, τα συστήματα OCR έχουν δυνητικές εφαρμογές στην εξαγωγή δεδομένων από συμπληρωμένα έντυπα, για παράδειγμα έχουν την ικανότητα ερμηνείας χειρόγραφων διευθύνσεων από ταχυδρομικά έγγραφα με σκοπό την αυτόματη δρομολόγηση τους, ή άλλο παράδειγμα η αυτόματη ανάγνωση των τραπεζικών επιταγών κ.λ.π. Η βασική συνιστώσα αυτών των εφαρμογών είναι μια μηχανή OCR, που είναι εξοπλισμένη με τις βασικές λειτουργικές αρχές όπως η εξαγωγή γραμμής, μεμονωμένες λέξεις σε γραμμές, λέξη σε τμηματοποίηση χαρακτήρων, αναγνώρισης χαρακτήρων και την ανάλυση σε επίπεδο λέξεων χρησιμοποιώντας πρότυπα λεξικά.

## 2.4.2 Τύποι OCR

Υπάρχουν δύο τύποι μηχανών οπτικής αναγνώρισης χαρακτήρων μέχρι στιγμής, έντυπα και χειρόγραφα OCR.

**Έντυπα OCR**

Έντυπα OCR αναφέρεται στην αναγνώριση των εκτυπωμένων έγγραφων από την OCR μηχανή. Τα τυπωμένα έγγραφα σαρώνονται και τροφοδοτούνται στη μηχανή OCR τα οποία στη συνέχεια αναγνωρίζονται

**Χειρόγραφα OCR**

Χειρόγραφα OCR αναφέρεται στην αναγνώριση των χειρόγραφων εγγράφων από την OCR μηχανή. Οι σελίδες που περιέχουν χειρόγραφα έγγραφα σαρώνονται σε οποιαδήποτε μορφή (.pdf, .bmp, .tiff κλπ) και στη συνέχεια τροφοδοτούνται στη μηχανή OCR, τα οποία στη συνέχεια αναγνωρίζονται.

## 2.4.3 Υπάρχουσες βιβλιοθήκες OCR

Στην έρευνα που πραγματοποιήθηκε μελετήθηκαν μερικές από τις ήδη υπάρχουσες βιβλιοθήκες OCR και επιλέχθηκε μία από αυτές.

Μελετήθηκαν οι ακόλουθες βιβλιοθήκες:

**ABBYY - OCR**

Η ABBYY είναι μια πολύ γνωστή εταιρεία που παρέχει υπηρεσίες αναγνώρισης εγγράφων, συλλογή δεδομένων και γλωσσολογικών λογισμικών [9]. Παρέχουν επίσης OCR SDK για κινητές πλατφόρμες. Χρησιμοποιώντας το SDK τους, οι προγραμματιστές μπορούν να αναπτύξουν εφαρμογές που βασίζονται στο OCR για κινητά τηλέφωνα.

Παρέχουν τα SDK για ακόλουθες κινητές πλατφόρμες.

- Android
- Windows Mobile
- Mobile Linux (Tizen)
- iPhone
- Symbians

Το ABBY OCR παίρνει μια εικόνα ως είσοδο σε όλους τους γνωστούς τύπους αρχείων εικόνας όπως .png, .gif, .jpg, κ.λπ., και καθιστά το κείμενο της εικόνας επεξεργάσιμο και αναζητήσιμο. Μπορούμε στη συνέχεια να χρησιμοποιήσουμε το εξαγόμενο κείμενο για περαιτέρω επεξεργασία και ανάπτυξη πολλών εφαρμογών.

**Tesseract- OCR**

Tesseract-OCR είναι μια μηχανή ανοιχτού κώδικα. Είναι γραμμένο σε C / C ++ και αναπτύχθηκε στα εργαστήρια της Hewlett-Packard (HP) μεταξύ 1985 και 1996. Αλλά ποτέ δεν το χρησιμοποιούσαν στα προϊόντα τους. Ήταν μία από τις κορυφαία 3 μηχανές OCR στη δοκιμή ακρίβειας στο UNLV 1995 [6]. Το 2005, κυκλοφόρησε ως μηχανή ελεύθερου λογισμικού. Από το 2007, η Google έχει λάβει την εποπτεία της μηχανής αναγνώρισης χαρακτήρων (tesseract-OCR) για την περαιτέρω ανάπτυξη και τη συντήρηση της [6]. Παίρνει γκρι ή έγχρωμη εικόνα ως είσοδο και δίνει έξοδο σε μορφή κειμένου. Στην αρχή, μόνο .tiff τύπος εικόνων υποστηριζόταν αλλά τώρα υποστηρίζει επίσης άλλους τύπους εικόνων, όπως .png, .jpg, κλπ Μπορεί να διαβάσει τα δεδομένα σε οποιαδήποτε γλώσσα από την εικόνα όπως τα αγγλικά, σουηδικά, δανικά, κλπ και προγραμματιστές μπορούν να "εκπαιδεύσουν" τη δική τους γλώσσα, εάν η υποστήριξη για μια συγκεκριμένη γλώσσα δεν είναι διαθέσιμη [6]. Οι προγραμματιστές της Google δοκιμάζουν την λειτουργία του στο λειτουργικό σύστημα Ubuntu και Windows αλλά λειτουργεί επίσης και σε άλλες πλατφόρμες Linux και Mac, κλπ.
Μπορούμε επίσης να χρησιμοποιήσουμε αυτή τη βιβλιοθήκη σε κινητές πλατφόρμες όπως το Android και iPhone κ.λπ.

**Aspire-OCR**

Η Aspire-OCR είναι μια μηχανή οπτικής αναγνώρισης που παρέχει APIs για την γλώσσα προγραμματισμού Java, C, C ++, Delphi, CSharp.Net, Visual Basic.Net και Visual C ++. Net. Περιλαμβάνει επίσης λειτουργικότητα για Barcode ανάγνωση [10]. Για την ανάπτυξη εφαρμογών που εκμεταλλεύονται την μηχανή Aspire-OCR απαιτείται το δικό τους SDK και έχει υποστήριξη για τις ακόλουθες πλατφόρμες.

- Windows
- Mac
- Linux
- Solaris

Παίρνει σχεδόν κάθε μορφή - τύπο εικόνας, όπως .png, .jpg, .bmp, κλπ σαν είσοδο και επιστρέφει ένα κείμενο εικόνας σε μορφή συμβολοσειράς.

**Online (διαδικτυακό) OCR API**

Όπως υποδηλώνει και το όνομα, είναι μια διαδικτυακή υπηρεσία που παρέχει APIs για OCR. Με τη χρήση αυτών των APIs, είναι δυνατόν να αναπτυχθούν διάφορα είδη

εφαρμογών για έξυπνα κινητά τηλέφωνα ή και web-based εφαρμογές. [11]. Υποστηρίζει .png, .jpg, .gif και .tiff τύπους εικόνων και μπορεί να διαβάσει το κείμενο από γνωστές γλώσσες όπως τα αγγλικά, σουηδικά, γερμανικά, δανικά, κ.λπ.

### 2.4.4 Κριτήρια επιλογής

Μία από τις βασικές δυσκολίες στην εργασία μου ήταν να βρεθεί μια αποτελεσματική λύση OCR για να ενσωματωθεί στην εφαρμογή. Υπάρχουν πολλοί πάροχοι μηχανών / υπηρεσίες cloud OCR, όπως είδαμε παραπάνω και εναλλακτικές λύσεις, προσφέροντας εύκολες προσεγγίσεις για την ανάπτυξη της εφαρμογής OCR σε πολλούς διαφορετικούς τομείς. Ωστόσο, οι περισσότερες από τις μηχανές OCR ή πάροχοι υπηρεσιών είναι εμπορικές και κλειστού κώδικα.

Για την παρούσα εργασία επιλέχθηκε η μηχανή αναγνώρισης χαρακτήρων tesseract-OCR. Επειδή είναι ελεύθερο λογισμικό και δίνει επίσης την επιλογή στον προγραμματιστή της εφαρμογής να προσθέσει τη δική του γλώσσα. Επίσης μας βοήθησε στην ανάπτυξη μιας αυτόνομης / offline (εκτός σύνδεσης) εφαρμογής που δεν ήταν δυνατόν για παράδειγμα στην περίπτωση του Online OCR API. ενώ ABBY και ASPIRE είναι μηχανές ιδιοκτησιακού λογισμικού (proprietary software). Αν θέλουμε να χρησιμοποιήσουμε αυτές τις βιβλιοθήκες OCR στην εφαρμογή μας, τότε θα έπρεπε να αγοραστεί η άδεια χρήσης τους.

### 2.4.5 Tesseract

Το Tesseract είναι ένα ελεύθερο λογισμικό, είναι μηχανή οπτικής αναγνώρισης χαρακτήρων για διάφορα λειτουργικά συστήματα και θεωρείται ως μία από τις πιο ακριβής μηχανές OCR ελεύθερου λογισμικού που διατίθεται σήμερα.

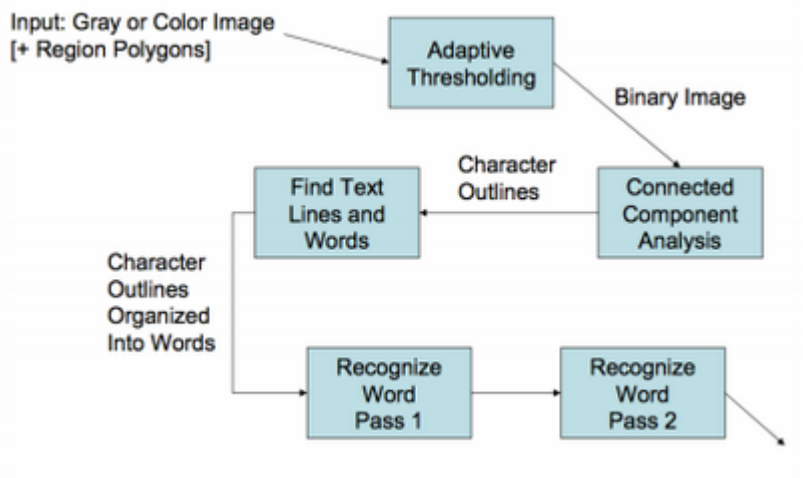
| | | Character | | | Word | | |
|---|---|---|---|---|---|---|---|
| Ver | Set | Errs | %Err | %Chg | Errs | %Err | %Chg |
| HP | bus | 5959 | 1.86 | | 1293 | 4.27 | |
| 2.0 | bus | 6449 | 2.02 | 8.22 | 1295 | 4.28 | 0.15 |
| HP | doe | 36349 | 2.48 | | 7042 | 5.13 | |
| 2.0 | doe | 29921 | 2.04 | -17.68 | 6791 | 4.95 | -3.56 |
| HP | mag | 15043 | 2.26 | | 3379 | 5.01 | |
| 2.0 | mag | 14814 | 2.22 | -1.52 | 3133 | 4.64 | -7.28 |
| HP | news | 6432 | 1.31 | | 1502 | 3.06 | |
| 2.0 | news | 7935 | 1.61 | 23.36 | 1284 | 2.62 | -14.51 |
| 2.0 | total | 59119 | | -7.31 | 12503 | | -5.39 |

Εικόνα 8: Τα αποτελέσματα των σημερινών και των παλαιών Tesseract από Ray [31]

Ο Tesseract OCR engine είναι μια open-source και η πλέον ακριβής μηχανή ελεύθερη προς χρήση OCR engine που δημιουργήθηκε από τα εργαστήρια της HP και εκτενώς βελτιωμένη (όπως φαίνεται από την Εικόνα 8) από την Google προς το παρόν. Η πρόσφατη έκδοση παρουσιάστηκε ως 3.04.01 και η αρχική έκδοση το 1995 εμφανίζεται ως HP. Αυτή τι στιγμή είναι η μόνη ελεύθερη εργαλειοθήκη διαθέσιμη στην αγορά για την πλατφόρμα Android.

## 2.4.5.1 Tesseract OCR Αρχιτεκτονική

Η διαδικασία OCR του Tesseract ακολουθεί μια σταδιακή μεθοδολογία, όπως φαίνεται την Εικόνα 9.




Εικόνα 9: Tesseract OCR Αρχιτεκτονική από Ray (2007) [32]

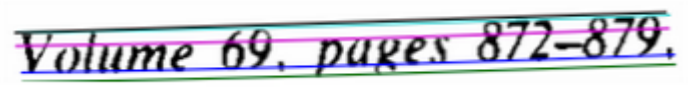
Το Tesseract υποθέτει ότι η εικόνα εισόδου είναι μια πολυγωνική δυαδική περιοχή(polygonal binary area). Τα περιγράμματα του στοιχείου συγκεντρώνονται σε σταγόνες(Blobs) [51]. Οι σταγόνες(Blobs) τοποθετούνται σε γραμμές κειμένου [51]. (Στην οπτική αναγνώριση εικόνας (OCR) οι σταγόνες(Blobs) αναφέρονται ως εκείνες οι περιοχές στην ψηφιακή εικόνα όπου εντοπίζονται να είναι διαφορετικές από τις γύρω περιοχές σε σχέση με το χρώμα ή τη φωτεινότητα.) Το Tesseract ανιχνεύει τις περιοχές κειμένου από τον αλγόριθμο εύρεσης γραμμής.

## 2.4.5.2 Εύρεση Γραμμής και λέξης

Το Tesseract αρχικά  φιλτράρει έξω σταγόνες(out blobs) που είναι μικρότερες από κάποιο κλάσμα του διάμεσου ύψους, το οποίο θα μπορούσε ενδεχομένως να είναι σημεία στίξης ή θόρυβος. Στη συνέχεια το Tesseract επεξεργάζεται τις φιλτραρισμένες

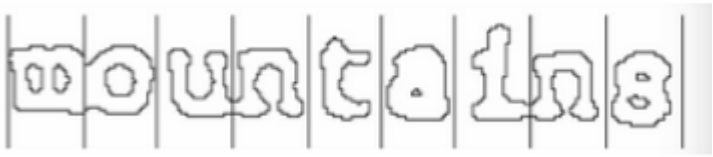
σταγόνες(blobs) σε οριζόντια θέση και τις εκχωρεί σε μια γραμμή κειμένου. Αφού οι σταγόνες έχουν εκχωριθεί, οι φιλτραρισμένες σταγόνες θα πρέπει να τοποθετηθούν και πάλι στην κατάλληλη θέση.

Μόλις οι γραμμές κειμένου έχουν εντοπιστεί, οι βασικές γραμμές τοποθετούνται με μεγαλύτερη ακρίβεια, χρησιμοποιώντας μια τετραγωνική spline(quadratic spline). Οι γραμμές βάσης τοποθετούνται με τον διαχωρισμό των σταγόνων σε ομάδες με λογική συνεχή μετατόπιση για την αρχική ευθεία γραμμή βάσης [51]. Η τετραγωνική spline κρατάει το αποτέλεσμα του υπολογισμού σε ένα σταθερό επίπεδο. Ωστόσο, οι πολλαπλές αυλακώσεις μπορούν να οδηγήσουν σε ασυνέχειες με αυτό να είναι ένα μειονέκτημα. Όπως φαίνεται στην Εικόνα 10, το Tesseract χρησιμοποιεί τη γραμμή βάσης (πράσινη), τη γραμμή κάτω αρχής (μπλε), τη μέση γραμμή (ροζ) και τη γραμμή ανάβασης (κυανό) για να εντοπίσει τη γραμμή του κειμένου.




Εικόνα 10: Ένα παράδειγμα όπου η βασική γραμμή καμπυλώνει από Ray [51]

Αφού η περιοχή κειμένου έχει ανιχνευθεί, το Tesseract πραγματοποιεί τμηματοποίηση της λέξης σε κομμάτια από χαρακτήρες αναλύοντας αν είναι ισοδύναμου πλάτους, όπως φαίνεται στην Εικόνα 11.

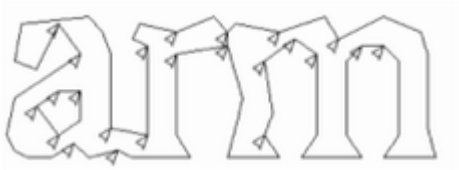


Εικόνα 11: Μια τεμαχισμένη λέξη σταθερού πλάτους από Ray [51]

Όταν το κείμενο με σταθερό πλάτους βρεθεί, η διαδικασία απομόνωσης της λέξης σε γράμματα σταματάει και ξεκινάει η διαδικασία της αναγνώρισης των λέξεων [51].

### 2.4.5.3 Αναγνώριση λέξης

Το Tesseract βελτιώνει το μη ικανοποιητικό αποτέλεσμα τεμαχίζοντας τις σταγόνες(blobs). Προσπαθεί να βρει σημεία υποψήφια κοπής από κοίλες κορυφές μιας πολυγωνικής προσέγγισης του περιγράμματος [51]. Τουλάχιστον τρία ζεύγη σημείων κοπής διασφαλίζουν ότι ένας ξεχωριστός χαρακτήρας είναι από το σύνολο ASCII, [51]. Η Εικόνα 12 δείχνει πώς το Tesseract αναγνωρίζει τον χαρακτήρα όταν ο χαρακτήρας «r» αγγίζει το χαρακτήρα «m».

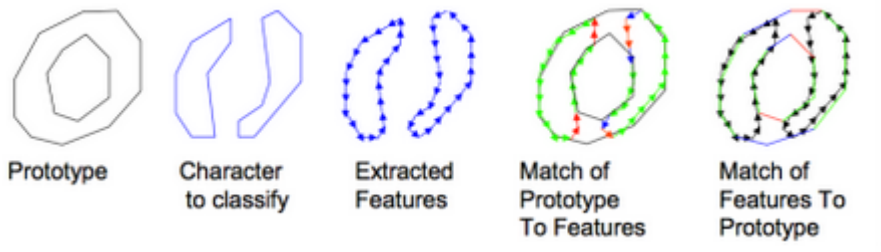


Εικόνα 12: Σημεία υποψήφια κοπής από Ray [51]

Αν τα σημεία κοπής έχουν όλα χρησιμοποιηθεί και το αποτέλεσμα εξακολουθεί να μην είναι ικανοποιητικό, η διαδικασία της ανάλυσης θα πρέπει να αναληφθεί από τον associator. Ο associator πραγματοποιεί την πρώτη καλύτερη αναζήτηση και κάνει το γράφημα σε υποψήφιους χαρακτήρες. Η Εικόνα 13 δείχνει ένα παράδειγμα του “σπασμένου” χαρακτήρα.

Εικόνα 13: Μια “σπασμένη” λέξη έτοιμη για να αναγνωριστεί Ray [51]

Τα χαρακτηριστικά του “σπασμένου” χαρακτήρα εξάγονται από το τεμαχισμό του περιγράμματος από το στατικό ταξινομητή (static classifier).

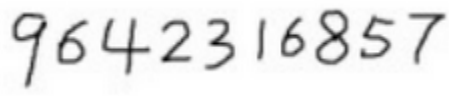



Εικόνα 14: Χαρακτηριστικά γνωρίσματα και αντιστοίχηση ταίριασμα Ray [51]

## 2.4.5.4 Δεδομένα Εκπαίδευσης (Training Data)

Για να βελτιώσουν την ακρίβεια και την επέκταση του περιορισμού της γλώσσας, η μηχανή Tesseract έχει σχεδιαστεί για να είναι πλήρως εκπαιδεύσιμη. Μετά την τμηματοποίηση του χαρακτήρα, μια διαδικασία δύο φάσεων πραγματοποιείται. Στην πρώτη φάση, γίνεται μια προσπάθεια να αναγνωριστεί κάθε μια λέξη τη φορά [51]. Ο ικανοποιητικός χαρακτήρας περνά στον δυναμικό ταξινομητή (dynamic classifier) ως δεδομένα εκπαίδευσης(training data) [51]. Στην δεύτερη φάση, οι λέξεις που δεν είχαν αναγνωριστεί ικανοποιητικά αναγνωρίζονται και πάλι.

Οι χρήστες μπορούν να εκπαιδεύσουν το Tesseract βελτιώνοντας την ακρίβεια της αναγνώρισης για τις δικές τους περιπτώσεις. Η Εικόνα 15 είναι μία ψηφιακή εικόνα σε αναμονή για αναγνώριση.




Εικόνα 15: Δείγμα εικόνας προς αναγνώριση

Για την παρούσα εργασία επειδή οι χαρακτήρες είναι τυπωμένοι και όχι χειρόγραφοι δεν χρειάστηκε να γίνει εκπαίδευση των δεδομένων περαιτέρω από την ήδη υπάρχουσα και πραγματοποιημένη από την Google.

## 2.4.5.5 Χαρακτηριστικά του Tesseract

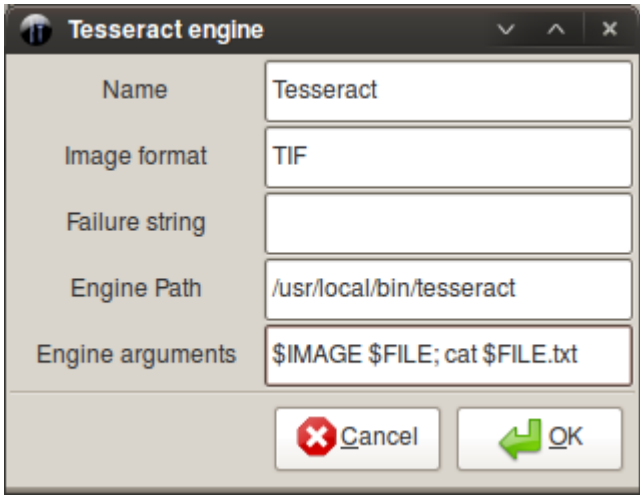
Το Tesseract ήταν στις 3 καλύτερες μηχανές OCR όπως προαναφέραμε όσον αφορά την ακρίβεια χαρακτήρα το 1995. Είναι διαθέσιμο για Linux, Windows και Mac OSX, ωστόσο, λόγω των περιορισμένων πόρων μόνο τα Windows και το Ubuntu ελέγχονται αυστηρά από τους προγραμματιστές.

Το Tesseract μέχρι και την έκδοση 2 μπορούσε να υποστηρίξει ως είσοδο μόνο τις εικόνες τύπου TIFF και μόνο απλό κείμενο μίας στήλης. Αυτές οι πρώιμες εκδοχές δεν περιελάμβαναν ανάλυση της διάταξης και έτσι κατά την εισαγωγή πολλαπλών στηλών κειμένου, εικόνων, ή εξισώσεων το αποτέλεσμα της εξόδου δεν ήταν το επιθυμητό. Από την έκδοση 3.00 το Tesseract υποστηρίζει μορφοποίηση κειμένου κατά την εξόδου, hOCR θέσης πληροφορίες και ανάλυση ως προς την διάταξη της σελίδας.
Επιπλέον υποστήριξη για μια σειρά από νέες μορφές εικόνας προστέθηκε με τη χρήση της βιβλιοθήκης Leptonica.

Οι αρχικές εκδόσεις του Tesseract μπορούσαν να αναγνωρίσουν μόνο αγγλικό κείμενο. Ξεκινώντας με την έκδοση 2 Tesseract ήταν σε θέση να επεξεργαστεί Αγγλικά, Γαλλικά, Ιταλικά, Γερμανικά, Ισπανικά, Πορτογαλικά Βραζιλίας και Ολλανδικά. Ξεκινώντας με την έκδοση 3 μπορούσε να αναγνωρίσει Αραβικά, Αγγλικά, Βουλγαρικά, Καταλανικά, Τσέχικα, Κινέζικα (απλοποιημένα και παραδοσιακά), Δανικά (πρότυπο και το σενάριο Fraktur), Γερμανικά, Ελληνικά, Φινλανδικά, Γαλλικά, Εβραϊκά, Κροατικά, Ουγγρικά, Ινδονησιακά, Ιταλικά, Ιαπωνικά, Κορεατικά, της Λετονίας, της Λιθουανίας, Ολλανδικά, Νορβηγικά, Πολωνικά, Πορτογαλικά, Ρουμανικά, ρωσικά, Σλοβακικά (πρότυπο και το σενάριο Fraktur), Σλοβενικά, Ισπανικά, της Σερβίας, της Σουηδίας, Ταγκαλόγκ, Ταϊλανδικά, Τουρκικά, Ουκρανικά και Βιετναμέζικα. Το Tesseract μπορεί να εκπαιδευτεί για να αναγνωρίζει και άλλες γλώσσες.

Αν το Tesseract χρησιμοποιείται για την επεξεργασία από δεξιά προς τα αριστερά κειμένου όπως τα αραβικά ή τα εβραϊκά τα αποτελέσματα εμφανίζονται σαν να ήταν μια γλώσσα που γράφεται και διαβάζεται από αριστερά προς τα δεξιά. Το Tesseract είναι κατάλληλο για χρήση ως backend, και μπορεί να χρησιμοποιηθεί ακόμη και για πιο πολύπλοκες εργασίες OCR, συμπεριλαμβανομένης της ανάλυσης διάταξης χρησιμοποιώντας ένα frontend όπως το OCRopus.

## 2.4.5.6 Γραφικό περιβάλλον και UI του Tesseract




Εικόνα 16: Γραφικό περιβάλλον και UI του Tesseract [12]

Η παραπάνω εικόνα δείχνει το παράθυρο ρύθμισης παραμέτρων του Tesseract στην εφαρμογή OCRFeeder

Το Tesseract δεν συνοδεύεται από ένα γραφικό περιβάλλον με το οποίο ο χρήστης μπορεί να έχει διεπαφή (GUI). Αντί αυτού, το Tesseract τρέχει από το terminal (διεπαφή γραμμής εντολών).

Υπάρχουν πολλές διαφορετικές εργασίες που παρέχουν ένα γραφικό περιβάλλον για την μηχανή Tesseract:

- FreeOCR– ένα Windows Tesseract GUI
- gImageReaderGTK - GUI διεπαφή για το Tesseract που υποστηρίζει την επιλογή στηλών και τμημάτων του εγγράφου. Μπορεί να ανοίξει πολλαπλά αρχεία PDF ή εικόνες, υποστηρίζοντας όλες τους τύπους εικόνας, μπορούν να μεταβιβαστούν επιλεγμένες περιοχές στο Tesseract για αναγνώριση και ορθογραφικό έλεγχο της εξόδου.
- gscan2pdf– Ένα GUI για την παραγωγή αρχείων PDF ή DjVus από σαρωμένο έγγραφο
- OCRFeeder – Διαθέτει μια πλήρη γραφική διεπαφή για τον χρήστη (GTK) η οποία επιτρέπει στους χρήστες του να διορθώνουν τυχόν μη αναγνωρισμένους χαρακτήρες, να ορίζουν ή να διορθώνουν τις θέσεις οριοθέτησης, να επιλέγουν το στυλ παραγράφου της επιθυμίας τους, να καθαρίζουν τις εικόνες εισόδου, την δυνατότητα ως είσοδο αρχεία τύπου PDF, να αποθηκεύουν και να εισάγουν - φορτώνουν τις εργασίες τους, και τέλος την επιλογή εξαγωγής των πάντων σε πολλούς διάφορους υποστηριζόμενους τύπους, κλπ
- OcrGUI– Ένα Linux GUI, γραμμένο στη γλώσσα C, χρησιμοποιώντας τα πλαίσια (frameworks) GLib και GTK +, υποστηρίζει τόσο το Tesseract όσο και το HOCR. Περιλαμβάνει ορθογραφικό έλεγχος χρησιμοποιώντας Hunspell ένα ελεύθερου λογισμικού εργαλείο ορθογραφικού ελέγχου.
- Qiqqa– Ένα δωρεάν εργαλείο διαχείρισης των αρχείων τύπου PDF που χρησιμοποιεί το Tesseract για την αναγνώριση των λέξεων ενός σαρωμένου αρχείου PDF και την δημιουργία ενός πλήρους Ευρετηρίου με βάση τις αναγνωρισμένες λέξεις.
- Tesseract GUI– Ένα ελεύθερο λογισμικό GUI για τα Mac OS X.
- TextRipper– Ένα Linux Tesseract και / ή Ocrad GUI με υποστήριξη για πολλαπλές σελίδες, -στήλες, και αρχεία.
- VietOCR– Ένα GUI βασισμένο στη Java (Java-based) με υποστήριξη πολλαπλών πλατφορμών, που περιλαμβάνει ένα πακέτο γλώσσας για τη Βιετναμέζικη γλώσσα και ειδικά εργαλεία μετα-επεξεργασία για τη Βιετναμέζικη γλώσσα
- YAGF– Γραφικό περιβάλλον (Qt 4.x) για σφηνοειδή γραφή και tesseract

Βιβλιοθήκες που χρησιμοποιούν την μηχανή Tesseract.

## 2.4.5.7 Ανασκόπηση υπάρχουσας βιβλιογραφίας

Ένας μεγάλος αριθμός τεχνικών αναγνώρισης χειρόγραφων εγγράφων έχει εμφανιστεί στην υπάρχουσα βιβλιογραφία. Ένα paper από το Basu et.al. [13] βασίστηκε στην αναγνώριση των χειρόγραφων δειγμάτων της ρωμαϊκής γραφής χρησιμοποιώντας το Tesseract 2.01 λαμβάνοντας υπόψη τόσο την ελεύθερη ροή γραφής όσο και τα απομονωμένα δείγματα γραφής. Έξι σελίδες χειρόγραφων εγγράφων που αποτελούνταν από 4 απομονωμένα δείγματα γραφής και 2 δείγματα ελεύθερης ροής γραφής συλλέχθηκαν από κάθε χρήστη. Στη φάση της "εκπαίδευσης-training" χρησιμοποιήθηκαν 3 απομονωμένες σελίδες από τα έγγραφα και 1 σελίδα ελεύθερης ροής. Για σκοπούς τιτλοποίησης ο επεξεργαστής που χρησιμοποιήθηκε ήταν ο bbTesseract. Για τους σκοπούς "εκπαίδευσης- training" 8 αρχεία δεδομένων χρησιμοποιήθηκαν από τον υποκατάλογο tessdata.

Η μηχανή αναγνώρισης χαρακτήρων Tesseract είχε εκπαιδευτεί να αναγνωρίζει δείγματα γραφής συγκεκριμένων χρηστών και για τις δύο κατηγορίες των σελίδων του εγγράφου. Σε ένα ενιαίο μοντέλο χρήστη, το σύστημα είχε εκπαιδευτεί με 1844 απομονωμένους χειρόγραφους χαρακτήρες και η απόδοση έχει δοκιμαστεί σε 1133 χαρακτήρες, που λαμβάνονται από το σύνολο δοκιμών. Το σύνολο δοκιμών αποτελούνταν από 3 σελίδες των απομονωμένων και 1 σελίδα της ελεύθερης ροής του χειρόγραφου εγγράφου. Η συνολική ακρίβεια χαρακτήρων επιπέδου του συστήματος παρατηρείται ως 83,5%. Το σύστημα απέτυχε να τμηματοποιήσει 5,56% των χαρακτήρων και έκανε λανθασμένη ταξινόμηση 10,94% χαρακτήρων.

Ένα άλλο έγγραφο του Basu et.al [14] βασίστηκε στην αναγνώριση του χειρόγραφου Ρωμαϊκού σεναρίου χρησιμοποιώντας Tesseract OCR 2.01. Σε αντίθεση με το προηγούμενο έγγραφο [15], το οποίο βασίστηκε στο μοντέλο ενός χρήστη το παρόν paper [14] βασίστηκε στο μοντέλο πολλαπλών χρηστών.

Ο συντάκτης που χρησιμοποιείται για σκοπούς τιτλοποίησης ήταν ο bbTesseract. Για τον σκοπό "εκπαίδευσης- training" 8 φακέλοι δεδομένων των Tesseract χρησιμοποιήθηκαν από τον υποκατάλογο tessdata. Το Tesseract είχε εκπαιδευτεί με δείγματα γραφής συγκεκριμένου χρήστη που αφορούσαν και τις δύο κατηγορίες των σελίδων του εγγράφου ώστε να παράγουν χωριστά μοντέλα χρήστη που αντιπροσωπεύουν μια μοναδική γλώσσα. Κάθε τέτοιο σύνολο γλώσσας αναγνωρίζεται απομονωμένα και ελεύθερης ροής χειρόγραφα δείγματα συλλέγονται από τον ορισθείσα χρήστη. Σε ένα μοντέλο τριών χρηστών, το σύστημα εκπαιδεύτηκε με 1844, 1535 και 1113 χαρακτήρες απομονωμένων χειρόγραφων δειγμάτων που συλλέχθησαν από τους τρεις διαφορετικούς χρήστες και η απόδοση έχει δοκιμαστεί σε 1133, 1186 και 1204 χαρακτήρες δειγμάτων, τα οποία επίσης συλλέχθησαν από τα σύνολα δοκιμής των τριών χρηστών αντίστοιχα. Η ακρίβεια αναγνώρισης των χαρακτήρων ενός συγκεκριμένου χρήστη ελήφθησαν ως 87,92%, 81,53% και 65,71% αντίστοιχα. Ενώ η συνολική ακρίβεια αναγνώρισης των χαρακτήρων του ολικού συστήματος

παρατηρήθηκε ως 78,39%.
Το σύστημα απέτυχε να τμηματοποιήσει 10,96% των χαρακτήρων και έκανε λανθασμένη ταξινόμηση 10,65% χαρακτήρων στο συνολικό σύνολο δεδομένων των τριών διαφορετικών χρηστών.

Η απόδοση του συστήματος δοκιμάστηκε σε δύο διαφορετικά σύνολα δεδομένων. Το ένα που αποτελείται από δείγματα που συλλέγονται από τους “γνωστούς χρήστες” (αυτούς για τους οποίους ετοίμασαν τα trainning data) και το άλλο αποτελείται από χειρόγραφα δείγματα δεδομένων αγνώστων χρηστών. Η δοκιμή 1 (dataset 1) και η δοκιμή 2 (dataset 2) αποτελούνταν από 249 και 349 δείγματα, αντίστοιχα τα οποία συλλέχθηκαν από τους γνωστούς και τους άγνωστους χρήστες αντίστοιχα. Οl συνολικές οι αναγνωρίσεις που πραγματοποιήθηκαν έγιναν με ακρίβεια έως 92,1% και έως 86,59% αντίστοιχα για τα εν λόγω σύνολα δεδομένων δοκιμής. Στο [13] Basu et.al. είχε δοθεί μια ιδέα για την αναγνώριση των χειρόγραφων λατινικών αριθμών χρησιμοποιώντας Tesseract 2.01. Εδώ τα δείγματα εκπαίδευσης (trainning data) συλλέχθηκαν από τρεις διαφορετικούς χρήστες για να παράγουν ένα ενιαίο μοντέλο χρήστη για χειρόγραφους λατινικούς αριθμούς. Το ενιαίο αυτό μοντέλο δοκιμάστηκε σε νέα δείγματα από τους ίδιους χρήστες (οι οποίοι συμμετείχαν στην προετοιμασία των δειγμάτων εκπαίδευσης), καθώς και σε χειρόγραφα δείγματα διαφόρων άλλων χρηστών. Το Tesseract είχε εκπαιδευτεί με δείγματα δεδομένων από διαφορετικά άτομα για να δημιουργήσει ένα μοντέλο γλώσσας ανεξαρτήτου χρήστη, που θα αντιπροσωπεύει το χειρόγραφο ρωμαϊκό ψηφιακό σετ δεδομένων. Τέλος το σύστημα εκπαιδεύτηκε με 1226 ψηφιακά δείγματα όσα και τα ψηφιακά δείγματα που συλλέχθηκαν.

## 2.4.5.8 Περιορισμοί του Tesseract OCR

Μια μηχανή OCR ακόμα και μία omni-font engine [16] δεν μπορεί να διαβάσει - αναγνωρίσει τα πάντα. Συνήθως δυσκολεύονται να διαβάσουν τους χαρακτήρες που είναι συνήθως μικρότεροι από 8 σημεία. Ομοίως δυσκολεύεται στους πολύ μεγάλους χαρακτήρες, πάνω από 24 πόντους, οι οποίοι θα πρέπει να αγνοούνται από την μηχανή OCR. Το μεγαλύτερο εμπόδιο για την καλή αναγνώριση, ωστόσο, είναι η ποιότητα του τύπου. Σε υλικά που δεν εκτυπώνουν εκτυπωτές laser υψηλής ανάλυσης (με πλήρη φυσίγγια τόνερ), η μηχανή OCR δεν θα είναι σε θέση να εκτελέσει μια καλή αναγνώριση. Πολλές εκτυπώσεις γίνονται σε dot matrix εκτυπωτές οι οποίες μπορεί να περιέχουν ξεθωριασμένες λωρίδες. Όταν οι κουκκίδες των χαρακτήρων δεν είναι αρκετά “πατημένες” ή όταν η πληκτρολόγηση είναι πολύ αχνή, τότε η ακρίβεια του OCR μειώνεται.

**Ιστορικό και περιορισμοί του Tesseract OCR**

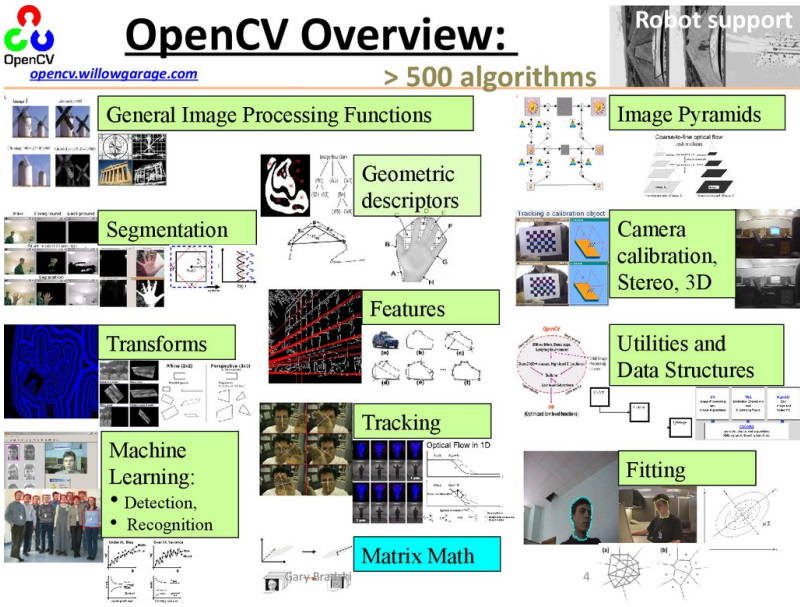
- Το Tesseract σχεδιάστηκε αρχικά για να αναγνωρίζει αγγλικό κείμενο μόνο. Έχουν γίνει προσπάθειες τροποποίησης της μηχανής και του συστήματος ”εκπαίδευσης” ώστε να καταστεί σε θέση να αναγνωρίζει και άλλες γλώσσες και UTF-8 χαρακτήρες. Το Tesseract 3.0 μπορεί να χειριστεί οποιουσδήποτε Unicode χαρακτήρες (κωδικοποιημένους με UTF-8), αλλά υπάρχουν όρια ως προς το εύρος των γλωσσών που είναι σε θέση να αναγνωρίσει, γι 'αυτό παρακαλώ να λάβετε αυτό το τμήμα υπόψη πριν από την κατασκευή της εφαρμογής σας με την ελπίδα ότι θα λειτουργήσει καλά για τη γλώσσα της επιθυμίας σας.
- Το Tesseract σήμερα μπορεί μόνο να χειριστεί γλώσσες που διαβάζονται μόνο από αριστερά προς τα δεξιά. Μπορείτε βεβαίως να πάρετε αποτελέσματα από μια γλώσσα η οποία διαβάζεται από δεξιά προς τα αριστερά αλλά το αρχείο εξόδου θα είναι οι λέξεις όπως θα διαβάζονταν από μια αριστερά προς τα δεξιά γλώσσα. Top-to-bottom γλώσσες δεν υποστηρίζονται από την έκδοση 3.01.
- Το Tesseract είναι απίθανο να είναι σε θέση να χειριστεί γλώσσες όπως τα Αραβικά. Χρειάζονται κάποιοι εξειδικευμένοι αλγόριθμοι και μέχρι στιγμής δεν έχουν γραφτεί.
- Το Tesseract είναι πιο αργό σε μεγάλο σύνολο χαρακτήρων γλωσσών όπως τα κινέζικα.
- Οποιαδήποτε γλώσσα που έχει διαφορετικά σημεία στίξης και αριθμούς πρόκειται να βρεθεί σε μειονεκτική θέση από μερικούς αλγορίθμους που υποθέτουν ASCII στίξης και ψηφία. Αυτό θα φτιαχτεί στην έκδοση 3.0x για x >= 2.

## 2.5 Επεξεργασία εικόνας

Η επεξεργασία εικόνας είναι η τεχνική κατά την οποία διάφορες εργασίες εκτελούνται στην εικόνα για να προσαρμόσουν τα δεδομένα εικόνας σύμφωνα με τις ανάγκες του χρήστη. Βασικό κομμάτι της εργασία είναι η επεξεργασία της εικόνας. Έτσι, η έρευνά που πραγματοποιήθηκε συμπεριλαμβάνει επίσης την εξεύρεση κατάλληλων βιβλιοθηκών για αυτό τον σκοπό. Η Open Source Computer Vision (OpenCV) βιβλιοθήκη είναι μια βιβλιοθήκη η οποία χρησιμοποιείται σε εργασίες που απαιτούν αποτελεσματική και σε πραγματικό χρόνο επεξεργασία εικόνας.

Η OpenCV (Open Source Computer Vision) είναι μια βιβλιοθήκη από συναρτήσεις για υλοποίηση μηχανικής όρασης σε πραγματικό χρόνο. Η OpenCV διατίθεται στο κοινό υπό την άδεια BSD, είναι ελεύθερο λογισμικό τόσο για ακαδημαϊκή όσο και για εμπορική χρήση. Έχει υλοποιηθεί με την χρήση των γλωσσών προγραμματισμού όπως C ++, C, και Python και σύντομα θα αποκτήσει και Java διεπαφή (UI) δίνοντας την δυνατότητα στους χρήστες να την τρέχουν στο αγαπημένο τους λογισμικό, Windows, Linux, Android ή ακόμα και macOS. Η βιβλιοθήκη έχει περισσότερους από 2500 βελτιστοποιημένους αλγορίθμους (βλέπε εικόνα 17). Χρησιμοποιείται ευρέως σε όλο τον κόσμο με πάνω από 2.5 εκατομμύρια κατεβάσματα και περισσότερους από 40 χιλιάδες ανθρώπους και ομάδες χρηστών. Χρησιμοποιείται σήμερα από τις διαδραστικές τέχνες, μέχρι και την προηγμένη ρομποτική. [17]

Η JavaCV παρέχει wrapper classes για την OpenCV [18]. Με τη χρήση της JavaCV μπορούμε να αναπτύξουμε εφαρμογές σε πραγματικό χρόνο σε Java και Android.




Εικόνα 17. Επισκόπηση των λειτουργιών της OpenCV [17]

## 2.6 Ελεύθερο Λογισμικό

Ελεύθερο Λογισμικό είναι το λογισμικό που είναι διαθέσιμο στη μορφή πηγαίου κώδικα δίνοντας έτσι την δυνατότητα στον καθένα να μπορεί ελεύθερα να το χρησιμοποιεί, να το αντιγράφει, να το διανέμει και να το τροποποιεί ανάλογα με τις ανάγκες του. Είναι ένα εναλλακτικό μοντέλο ανάπτυξης και χρήσης λογισμικού το οποίο παρέχει τη δυνατότητα αλλαγών ή βελτιώσεων ώστε να καλύπτονται οι ανάγκες αυτού που το χρησιμοποιεί.

### 2.6.1 Σύντομη ιστορία του ελεύθερου λογισμικού

Το 1983 ο Ρίτσαρντ Στάλλμαν ανακοίνωσε το GNU Project. Η ανάπτυξη του λογισμικού για το GNU άρχισε τον Ιανουάριο του 1984, και το Ίδρυμα Ελεύθερου Λογισμικού (FSF) ιδρύθηκε τον Οκτώβριο του 1985. Αυτός ανέπτυξε ένα ορισμό για το ελεύθερο λογισμικό και την έννοια του "copyleft", σχεδιασμένη ειδικά για να διασφαλίσει την ελευθερία του λογισμικού για όλους.

Οφέλη Χρήσης Ελεύθερου Λογισμικού:

- Απόλυτα νόμιμο λογισμικό το οποίο διατίθεται χωρίς κόστος και έχει πολύ μικρότερο κόστος συντήρησης
- Λογισμικό που ενσωματώνει άμεσα τις πιο σύγχρονες τεχνολογικές εξελίξεις
- Συχνές ενημερώσεις με νέες δυνατότητες
- Μεγάλη και φιλική κοινότητα ανάπτυξης και υποστήριξης του λογισμικού
- Πληθώρα πρόσθετων προγραμμάτων με εύκολη και δωρεάν εγκατάσταση
- Σταθερότητα και ασφάλεια
- Σημαντικά ασφαλέστερο και αξιόπιστο σε σχέση με ένα ιδιοκτησιακό λογισμικό που κατεβάζουμε από το διαδίκτυο.
- Δυνατότητα να εξερευνήσουμε και να μάθουμε τον τρόπο λειτουργίας του λογισμικού προσαρμόζοντάς το στις ανάγκες μας

Το Ελεύθερο Λογισμικό ορισμένες φορές αναφέρεται και ως ανοιχτό λογισμικό ή λογισμικό ανοιχτού κώδικα αλλά οι δύο έννοιες δεν είναι ταυτόσημες. Η έννοια του ελεύθερου λογισμικού, δεν είναι κάθε λογισμικό ελεύθερο μόνο και μόνο επειδή είναι ανοιχτού κώδικα.

Η ετικέτα ανοικτού κώδικα βγήκε από ένα συνέδριο στρατηγικής που πραγματοποιήθηκε στις 7 Απριλίου 1998 στο Palo Alto σε αντίδραση με την ανακοίνωση της Netscape τον Ιανουάριο του 1998 για απελευθέρωση του πηγαίου κώδικα ενός πλοηγού στο διαδίκτυο όπως ο Mozilla. Η Netscape κυκλοφόρησε τον πηγαίο κώδικα του Navigator ως “open source”, με ευνοϊκά αποτελέσματα

## 2.6.2 Ευρέως χρησιμοποιούμενα προϊόντα ελεύθερου λογισμικού

Εργασίες(Project) ελεύθερου λογισμικού έχουν κατασκευαστεί και συντηρούνται από ένα δίκτυο εθελοντών προγραμματιστών. Κύρια παραδείγματα των προϊόντων ανοικτού κώδικα είναι ο διακομιστής HTTP Apache, η e-commerce πλατφόρμα osCommerce και το πρόγραμμα περιήγησης στο διαδίκτυο Mozilla Firefox. Ένα από τα πιο επιτυχημένα προϊόντα ελεύθερου λογισμικού είναι το λειτουργικό σύστημα GNU / Linux, ένα Unix-like λειτουργικό σύστημα. Το λειτουργικό σύστημα για φορητές συσκευές Android χρησιμοποιεί το μοντέλο ανάπτυξης του ανοιχτού κώδικα (open source) αλλά αποτυγχάνει να χαρακτηριστεί ελεύθερο λογισμικό γιατί συμπεριλαμβάνει μη--ελεύθερες βιβλιοθήκες. Σε ορισμένους τομείς, το ανοικτό λογισμικό είναι ο κανόνας, όπως και οι εφαρμογές voiceoverIP με Asterix (PBX).

## 2.6.3 Κυριότερα χαρακτηριστικά του ελεύθερου λογισμικού

(i) Το κόστος άδειας χρήσης των εφαρμογών ελεύθερου λογισμικού είναι τις περισσότερες φορές μηδενικό. Δεν αγοράζονται άδειες χρήσης και μπορούμε να έχουμε απεριόριστο αριθμό εγκαταστάσεων. Η χρήση ανοιχτού κώδικα δεν περιορίζει τον οργανισμό ή τον απλό χρήστη σε μια σχέση εξάρτησης από εταιρίες και επειδή η διανομή, η διόρθωση σφαλμάτων και η ανάπτυξη του λογισμικού ΕΛ μπορεί να γίνει από κάθε τεχνικά καταρτισμένη ομάδα, δημιουργείται ένα περιβάλλον έντονου ανταγωνισμού ο οποίος οδηγεί σε χαμηλές τιμές και υψηλές υπηρεσίες υποστήριξης.

(ii) Το Ελεύθερο λογισμικό αποτελείται από μια μεγάλη κοινότητα χρηστών και προγραμματιστών, οι οποίοι συνεργάζονται για τη συνεχή βελτίωση του λογισμικού, παρέχοντας γνώσεις και εργασία. Σήμερα λειτουργεί ένα παγκόσμιο ανοικτό δίκτυο προγραμματιστών, οι οποίοι παράλληλα αναπτύσσουν και διορθώνουν τον κώδικα των προγραμμάτων, κυκλοφορώντας ταχύτατα νέες βελτιωμένες εκδόσεις λογισμικού. Με αυτό τον τρόπο συμβάλλουν καθημερινά στην δημιουργία νέων κοινών αγαθών.

(iii) Η λογική της ανάπτυξης του λογισμικού του ανοιχτού κώδικα είναι τέτοια έτσι ώστε να είναι δοκιμασμένο από πολλούς και αποφεύγονται αρνητικές εκπλήξεις και σφάλματα. Ο κώδικας μελετάται από πλήθος ανθρώπων, άρα τα όποια κενά ασφαλείας εντοπίζονται και διορθώνονται με μεγάλη ταχύτητα. Η υποστήριξη σε περίπτωση εμφάνισης προβλημάτων μπορεί να προέλθει άμεσα.

(iv) Οι χρήστες θα πρέπει να αντιμετωπίζονται ως συν-προγραμματιστές (co-developers)

Οι χρήστες αντιμετωπίζονται σαν συν-προγραμματιστές και έτσι έχουν πρόσβαση στον πηγαίο κώδικα του λογισμικού. Επιπλέον, οι χρήστες ενθαρρύνονται να υποβάλουν τις

προσθήκες τους στο λογισμικό, τις επιδιορθώσεις τους στον κώδικα του λογισμικού, τις αναφορές σφαλμάτων, αλλά και τεκμηρίωση (documentation) κ.λπ. Έχοντας περισσότερους συν-προγραμματιστές αυξάνεται ο ρυθμός με τον οποίο αναπτύσσεται το λογισμικό. Ο νόμος του Linus αναφέρει, "έχοντας αρκετά μάτια όλα τα bugs είναι ρηχά". Αυτό σημαίνει ότι αν πολλοί χρήστες έχουν πρόσβαση, βλέπουν και διαβάζουν τον πηγαίο κώδικα, τότε τελικά θα βρεθούν όλα τα σφάλματα και θα υπάρχουν περισσότερες προτάσεις ως προς το πώς να διορθωθούν. Να σημειωθεί ότι ορισμένοι χρήστες έχουν προηγμένες γνώσεις προγραμματισμού, και επιπλέον, το μηχάνημα κάθε χρήστη είναι ταυτόχρονα ένα νέο--πρόσθετο περιβάλλον δοκιμών. Αυτό το νέο περιβάλλον δοκιμών προσφέρει τη δυνατότητα ώστε να βρεθεί και να διορθωθεί ένα νέο σφάλμα.

(v) Πρόωρες εκδόσεις διατίθενται στο κοινό (Early releases)

Η πρώτη έκδοση του λογισμικού θα πρέπει να κυκλοφορήσει το συντομότερο δυνατόν, έτσι ώστε να αυξήσει τις πιθανότητες ενός ατόμου να βρει συν-προγραμματιστές νωρίς από τα πρώτα στάδια.

(vi) Οι συχνές ενσωματώσεις

Οι αλλαγές στον κώδικα θα πρέπει να ενσωματώνονται (συγχωνεύονται σε έναν κώδικα κοινής βάσης) όσο το δυνατόν συχνότερα, προκειμένου να αποφευχθεί η διόρθωση μεγάλων αριθμών σφαλμάτων στο τέλος του κύκλου ζωής της εργασίας(project). Ορισμένα εργασίες(project) ανοικτού κώδικα έχουν “nightly builds” όπου η ενοποίηση των ενημερώσεων γίνεται αυτόματα σε καθημερινή βάση.

(vii) Διάφορες εκδόσεις

Θα πρέπει να υπάρχουν τουλάχιστον δύο εκδόσεις του λογισμικού. Θα πρέπει να υπάρχει μια “buggier” έκδοση με περισσότερες δυνατότητες- λειτουργίες και μια πιο σταθερή (stable) έκδοση με λιγότερες δυνατότητες- λειτουργίες. Η έκδοση buggy (ονομάζεται επίσης η έκδοση ανάπτυξης(development version)) είναι για τους χρήστες που θέλουν την άμεση χρήση των πιο τελευταίων λειτουργιών, και είναι πρόθυμοι να αποδεχθούν τον κίνδυνο από τη χρήση κώδικα που δεν έχει δοκιμαστεί ακόμη καλά. Οι χρήστες μπορούν στη συνέχεια να δρουν - ενεργούν ως συν-προγραμματιστές (co-developers) της εργασίας(project), συμβάλλοντας στην αναφορά σφαλμάτων και παρέχοντας προτάσεις ικανές για την διόρθωσή των σφαλμάτων.

(viii) Υψηλή τμηματοποίηση της εργασίας(project)
Η γενική δομή του λογισμικού θα πρέπει να είναι τμηματοποιημένη επιτρέποντας την παράλληλη ανάπτυξη σε ανεξάρτητες συνιστώσες.

## 3.1 Εργαλεία Ανάπτυξης

Σε αυτό το κεφάλαιο θα γίνει μια αναφορά για τις τεχνολογίες που χρησιμοποιήσαμε για την εκπόνηση της εφαρμογής της παρούσας διπλωματικής εργασίας.

### 3.1.1 Γλώσσα προγραμματισμού Java

Για την ανάπτυξη της εφαρμογής, χρησιμοποιήθηκε η γλώσσα προγραμματισμού Java γιατί είναι η επίσημη γλώσσα που χρησιμοποιείται για ανάπτυξη android εφαρμογών. Η Java είναι μια αντικειμενοστραφής γλώσσα προγραμματισμού που σχεδιάστηκε από την εταιρία Sun Microsystems.

Στις αρχές του 1991, η Sun αναζητούσε το κατάλληλο εργαλείο για να αποτελέσει την πλατφόρμα ανάπτυξης λογισμικού σε μικρο-συσκευές. Ο "πατέρας" της Java, James Gosling, που εργαζόταν εκείνη την εποχή για την Sun μετά από λίγο καιρό με την βοήθεια και του επιτελείο της εταιρίας κατέληξαν με μια πρόταση, η οποία ήταν η γλώσσα Oak. Το όνομά της το πήρε από το ομώνυμο δένδρο (βελανιδιά) το οποίο ο Gosling είχε έξω από το γραφείο του και έβλεπε κάθε μέρα.

Η Oak ήταν μία γλώσσα που διατηρούσε μεγάλη συγγένεια με την C++. Παρόλα αυτά είχε πολύ πιο έντονο αντικειμενοστραφή (object oriented) χαρακτήρα σε σχέση με την C++ και χαρακτηριζόταν για την απλότητα της. Όμως το όνομα της γλώσσας ήταν ήδη κατοχυρωμένο και τελικά την ονόμασαν Java που εκτός των άλλων ήταν το όνομα της αγαπημένης ποικιλίας καφέ για τους δημιουργούς της.

Ένα από τα βασικά πλεονεκτήματα της Java έναντι των περισσότερων άλλων γλωσσών είναι η ανεξαρτησία του λειτουργικού συστήματος και πλατφόρμας. Τα προγράμματα που είναι γραμμένα σε Java τρέχουνε ακριβώς το ίδιο σε Windows, Linux, Unix και Macintosh (σύντομα θα τρέχουν και σε_Playstation καθώς και σε άλλες κονσόλες παιχνιδιών) χωρίς να χρειαστεί να ξαναγίνει μεταγλώττιση (compiling) ή να αλλάξει ο πηγαίος κώδικας για κάθε διαφορετικό λειτουργικό σύστημα.

### 3.1.2 Android Software development kit (SDK)

Το Android SDK (Software Development Kit) είναι το επίσημο εργαλείο της Google για την ανάπτυξη εφαρμογών στο Android. Περιέχει μια συλλογή από εργαλεία και βιβλιοθήκες απαραίτητα για τους προγραμματιστές που θέλουν να αναπτύξουν μια εφαρμογή Android. Πρώτο βήμα που πρέπει να κάνει ο προγραμματιστής είναι η

εγκατάσταση και ρύθμισή του. Περιλαμβάνει παραδείγματα εφαρμογών με τον πηγαίο τους κώδικα, βοηθήματα, πληροφορίες και εξομοιωτή για την εκτίμηση της προόδου της εργασίας. Ακόμη, αναλαμβάνει την μεταγλώττιση του κώδικα έτσι ώστε να μπορεί να τρέχει στην εικονική μηχανή Dalvik. Η τελευταία έκδοση μέχρι τη στιγμή που γράφεται η παρούσα εργασία είναι η 24.4.1 (Stable release). Τέλος το Android SDK και τα εργαλεία παρέχονται ελεύθερα.

### 3.1.3 Eclipse

Είναι ένα τα προγράμματα που χρησιμοποιήθηκε για την ανάπτυξη της παρούσας εφαρμογής. Το Eclipse είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) που χρησιμοποιείται στο προγραμματισμό υπολογιστών, και είναι το πιο ευρέως χρησιμοποιούμενο Java IDE.

Περιέχει ένα βασικό περιβάλλον εργασίας και ένα σύστημα plug-in για την προσαρμογή του περιβάλλοντος. Το Eclipse είναι γραμμένο ως επί το πλείστον σε Java και κύρια χρήση του είναι η ανάπτυξη εφαρμογών Java, αλλά μπορεί επίσης να χρησιμοποιηθεί και για την ανάπτυξη εφαρμογών σε άλλες γλώσσες προγραμματισμού με τη χρήση των plugins, γλώσσες προγραμματισμού όπως Perl, PHP, Prolog, Python, R, Ruby , C , C ++ κ.α

Το Eclipse μέχρι το τέλος του 2014 υποστηριζόταν επίσημα από την Google, η οποία έχει αναπτύξει το ADT plugin, που συνδέει τις δυνατότητές του με το Android SDK. Επίσης, παρέχει σύνδεση με τον AVD Manager ώστε να διαχειρίζεται τις εικονικές συσκευές για αποσφαλμάτωση. Είναι μια εφαρμογή ελεύθερου λογισμικού που υποστηρίζεται και αναπτύσσεται ραγδαία από την παγκόσμια κοινότητα του ελεύθερου λογισμικού.

### 3.1.4 Android Developer Tools (ADT)

Το Android Developer Tools (ADT) είναι ένα plugin για το Eclipse IDE που έχει σχεδιαστεί για να παρέχει ένα ισχυρό, ολοκληρωμένο περιβάλλον το οποίο δίνει την δυνατότητα δημιουργίας εφαρμογών Android.

Το ADT επεκτείνει τις δυνατότητες του Eclipse δίνοντας την δυνατότητα μέσα από τις ρυθμίσεις ο χρήστης να δημιουργεί γρήγορα νέα Android project, να δημιουργεί μια εφαρμογή με UI (διεπαφή χρήστη), να προσθέσει τα πακέτα με βάση το Android Framework API, να διορθώσει τις εφαρμογές του χρησιμοποιώντας τα Android SDK εργαλεία, ακόμη και την εξαγωγή signed (υπογεγραμμένα) ή unsigned .apk αρχεία προκειμένου να διανείμει την εφαρμογή του.

Η ανάπτυξη σε Eclipse με ADT συνιστάται και είναι ο γρηγορότερος τρόπος ανάπτυξης Android εφαρμογών. [2]

### 3.1.5 Android Studio

Είναι ένα ακόμη πρόγραμμα που χρησιμοποιήθηκε για την ανάπτυξη της παρούσας εφαρμογής μετά το Eclipse.

Το Android Studio είναι ένα ολοκληρωμένο προγραμματιστικό περιβάλλον (IDE) για ανάπτυξη εφαρμογών στην πλατφόρμα Android. Ανακοινώθηκε στις 16 Μαΐου 2013 στο συνέδριο Google I/O.

Η πρώτη σταθερή έκδοση βγήκε το Δεκέμβριο του 2014, με την έκδοση 1.0. Βασισμένο στο λογισμικό της JetBrains' IntelliJ IDEA, το Android Studio σχεδιάστηκε αποκλειστικά για προγραμματισμό Android [1]. Είναι διαθέσιμο για Windows, Mac OS X και Linux, και αντικατέστησε τα Eclipse Android Development Tools (ADT) ως το κύριο IDE της Google για ανάπτυξη εφαρμογών Android.

### 3.1.6 Η Γλώσσα προγραμματισμού C/C++

Η γλώσσα προγραμματισμού C (προφέρεται "σι") είναι μια διαδικαστική γλώσσα προγραμματισμού γενικής χρήσης, η οποία αναπτύχθηκε αρχικά, μεταξύ του 1969 και του 1973, από τον Ντένις Ρίτσι στα εργαστήρια AT&T Bell Labs για να χρησιμοποιηθεί για την ανάπτυξη του λειτουργικού συστήματος UNIX. Η C έχει δυνατότητες δομημένου προγραμματισμού και επιτρέπει τη χρήση αναδρομής, ενώ, ο στατικός ορισμός του τύπου των μεταβλητών που επιβάλλει, προλαμβάνει πολλά σφάλματα κατά την χρήση τους.

Χαρακτηριστικό της ότι ο σχεδιασμός της περιλαμβάνει δομές που μεταφράζονται αποδοτικά σε τυπικές εντολές μηχανής (machine instructions) και εξ αιτίας αυτού χρησιμοποιείται συχνά σε εφαρμογές που παλιότερα γράφονταν σε συμβολική γλώσσα (assembly language).

Αυτό ακριβώς το χαρακτηριστικό της, που έχει σαν συνέπεια και την αυξημένη ταχύτητα εκτέλεσης των εφαρμογών που γράφονται σε αυτή, καθώς και το γεγονός ότι είναι διαθέσιμη στα περισσότερα σημερινά λειτουργικά συστήματα, συνέβαλε κατά πολύ στην καθιέρωση της και την χρήση της για ανάπτυξη λειτουργικών συστημάτων και λοιπών προγραμμάτων συστήματος (system software), αλλά και απλών εφαρμογών.

Η C συγκαταλέγεται πλέον στις πιο ευρέως χρησιμοποιούμενες γλώσσες προγραμματισμού όλων των εποχών και πολλές νεώτερες γλώσσες να έχουν επηρεαστεί άμεσα ή έμμεσα από αυτήν, συμπεριλαμβανομένων των C++.

Η C++ (C Plus Plus, ελληνική προφορά Σι Πλας Πλας) είναι μια γενικού σκοπού γλώσσα προγραμματισμού Η/Υ. Θεωρείται μέσου επιπέδου γλώσσα, καθώς περιλαμβάνει έναν συνδυασμό χαρακτηριστικών από γλώσσες υψηλού και χαμηλού επιπέδου. Η γλώσσα αναπτύχθηκε από τον Μπιάρνε Στρούστρουπ το 1979 στα εργαστήρια Bell της AT&T, ως βελτίωση της ήδη υπάρχουσας γλώσσας προγραμματισμού C, και αρχικά ονομάστηκε "C with Classes", δηλαδή C με Κλάσεις.

Η γλώσσα ορίστηκε παγκοσμίως, το 1998, με το πρότυπο ISO/IEC 14882:1998. Η τρέχουσα έκδοση αυτού του προτύπου είναι αυτή του 2003, η ISO/IEC 14882:2003. Μια καινούρια έκδοση είναι υπό ανάπτυξη, γνωστή ανεπίσημα με την ονομασία C++0x.

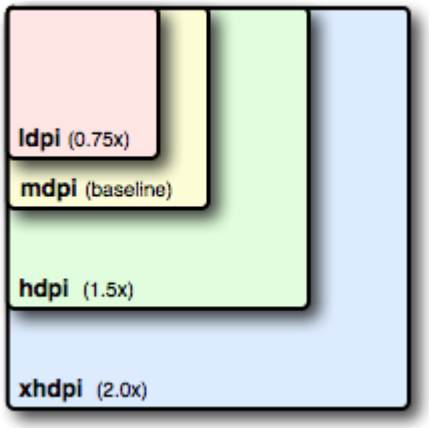
## 3.2 Απαιτήσεις της Εφαρμογής

Το λειτουργικό σύστημα Android τρέχει σε μια ποικιλία συσκευών που προσφέρουν διαφορετικά μεγέθη οθόνης και πυκνότητες, διαφορετικούς επεξεργαστές, κάμερες κτλ. Στόχος μας η υποστήριξη ενός ποσοστού μεγαλύτερου του 90 % των κινητών Android.

### 3.2.1 Υποστήριξη Πολλαπλών Συσκευών

Για εφαρμογές, το σύστημα Android παρέχει ένα συνεπές περιβάλλον ανάπτυξης για όλες τις συσκευές και χειρίζεται το μεγαλύτερο μέρος της διαδικασίας προσαρμογής του UI της κάθε εφαρμογής στην οθόνη στην οποία εμφανίζεται. Ταυτόχρονα, το σύστημα παρέχει API που επιτρέπουν τον έλεγχο του UI της εφαρμογής για συγκεκριμένα μεγέθη οθόνης και πυκνότητες, με στόχο τη βελτιστοποίηση του σχεδιασμού UI για διαφορετικές ρυθμίσεις παραμέτρων της οθόνης. Για παράδειγμα, ένα UI για tablets είναι διαφορετικό από το UI για κινητά τηλέφωνα.

Οι οθόνες της συσκευής μπορούν να κατηγοριοποιηθούν, χρησιμοποιώντας δύο γενικές ιδιότητες: το μέγεθος και την πυκνότητα. Η εφαρμογή θα πρέπει να εγκατασταθεί σε συσκευές με οθόνες που ανεξαρτήτου μεγέθους και πυκνότητας. Ως εκ τούτου, θα πρέπει να περιλαμβάνει ορισμένα εναλλακτικά μέσα που βελτιστοποιούν την εμφάνιση της εφαρμογής, για διαφορετικά μεγέθη οθόνης και πυκνότητες. Υπάρχουν τέσσερις κατηγορίες μεγεθών: μικρό, κανονικό, μεγάλο, XLarge, και τέσσερις κατηγορίες πυκνότητας: χαμηλή (ldpi), μέσο (MDPI), υψηλή (hdpi), επιπλέον υψηλό (xhdpi).



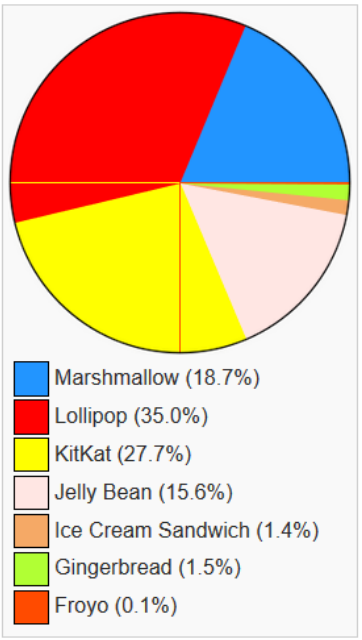

Εικόνα 18: Διαφορετικά μεγέθη οθόνης. [3]

Επίσης όπως γνωρίζουμε υπάρχουν δυο προσανατολισμοί οθονών(οριζόντια ή κατακόρυφη) που πρέπει να υποστηρίζει μια εφαρμογή.

Επιπλέον, οι τελευταίες εκδόσεις του Android παρέχουν συχνά καλύτερα APIs για την εφαρμογή, όμως θα πρέπει να συνεχίσουν να υποστηρίζουν παλαιότερες εκδόσεις του Android.

Παρακάτω παρουσιάζεται μια εικόνα η οποία αφορά το ποσοστό των συσκευών που τρέχουν μια δεδομένη έκδοση του Android.

| Έκδοση | Κωδική ονομασία | Ημερομηνία | API level | Διανομή [16] |
|---|---|---|---|---|
| 7.0 | *Nougat* | 22 Αυγούστου 2016 | 24 | |
| 6.0 - 6.0.1 | *Marshmallow* | 5 Οκτωβρίου 2015 | 23 | 18.7% |
| 5.1 - 5.1.1 | Lollipop | 9 Μαρτίου 2015 | 22 | 21.9% |
| 5.0 - 5.0.2 | | 3 Νοεμβρίου 2014 | 21 | 13.1% |
| 4.4. - 4.4.4 | *KitKat* | 31 Οκτωβρίου 2013 | 19 | 27.7% |
| 4.3 - 4.3.1 | Jelly Bean | 24 Ιουλίου 2013 | 18 | 2.3% |
| 4.2 - 4.2.2 | | 13 Νοεμβρίου 2012 | 17 | 7.7% |
| 4.1 - 4.1.2 | | 9 Ιουλίου 2012 | 16 | 5.6% |
| 4.0 - 4.0.4 | *Ice Cream Sandwich* | 16 Δεκεμβρίου 2011 | 15 | 1.4% |
| 3.2 | *Honeycomb* | 15 Ιουλίου 2011 | 13 | 0% |
| 2.3 - 2.3.7 | *Gingerbread* | 9 Φεβρουαρίου 2011 | 10 | 1.5% |
| 2.2 - 2.2.3 | *Froyo* | 20 Μαΐου 2010 | 8 | 0.1% |
| 2.0 - 2.1 | *Eclair* | 26 Οκτωβρίου 2009 | 7 | ? |
| 1.6 | *Donut* | 15 Σεπτεμβρίου 2009 | 4 | ? |




Εικόνα 19: Ποσοστό των συσκευών που τρέχουν μια δεδομένη έκδοση του Android [4]

Ένας από τους στόχους της συγκεκριμένης διπλωματικής ήταν η δημιουργία μιας εφαρμογής που εμφανίζεται σωστά και παρέχει μια βελτιστοποιημένη εμπειρία στον χρήστη σε όλες τις υποστηριζόμενες διαμορφώσεις οθόνης, χρησιμοποιώντας ένα ενιαίο .apk αρχείο.
Με βάση την παραπάνω εικόνα το 98,4% χρησιμοποιούν τουλάχιστον την έκδοση Android 4.x.x ή νεότερη, επομένως η εφαρμογή που υλοποιήθηκε μπορεί να «τρέξει»

τουλάχιστον σε έκδοση Android 4.x.x ή νεότερη και τέλος να περιλαμβάνει ορισμένα μέσα που να βελτιστοποιούν την εμφάνιση της εφαρμογής σε διαφορετικά μεγέθη οθόνης και πυκνότητας.

### 3.2.2 Λειτουργικές απαιτήσεις

Η συγκεκριμένη εφαρμογή που υλοποιήθηκε, είναι για έξυπνα κινητά τηλέφωνα η tablet, τα οποία να τρέχουν λογισμικό Android και μάλιστα Android 4.0.x η νεότερο, να έχουν ενσωματωμένη κάμερα, και πρόσβαση στο διαδίκτυο. Επίσης για την επικοινωνία μας με τον απομακρυσμένο server χρειαζόμαστε WLAN ή GSM / HSPA / LTE και να έχει τοποθετηθεί κάρτα SIM και να έχει διαθέσιμα MB (δεδομένα).

Αναλυτικότερα, χρειαζόμαστε ένα έξυπνο τηλέφωνο η tablet, το οποίο αρχικά να είναι σε θέση να συνδεθεί στο διαδίκτυο για την πρώτη φορά έναρξης της εφαρμογής και κάμερα υψηλής ανάλυσης Η συγκεκριμένη αναφορά γίνεται κυρίως γιατί οι χαρακτήρες σε ένα μικρο τσιπ είναι πολύ μικροί και θα δούμε σε επόμενα κεφάλαια ότι οι απαιτήσεις τις κάμερας είναι συγκεκριμένες .

Η σύνδεση στο διαδίκτυο είναι υποχρεωτική γιατί έτσι επιτυγχάνεται η επικοινωνία μεταξύ του έξυπνου κινητού τηλεφώνου ή tablet με τον server ώστε να γίνει η εγκατάσταση των training data κατά την έναρξη της εφαρμογής για πρώτη φορά καθώς και τις λήψης του datasheet.

Όσον αφορά το λογισμικό, η εφαρμογή που υλοποιήθηκε πρέπει να τρέχει λογισμικό Android 4.x.x ή νεότερο. Η επιλογή αυτή έγινε γιατί σύμφωνα με έρευνες μέχρι το Μάρτιου του 2015 το 98,4 της εκατό τρέχει λογισμικό Android 4.x.x ή νεότερο.

Τέλος, η εφαρμογή που υλοποιήθηκε δίνει τη δυνατότητα άμεσης πρόσβασης από τον πλοηγητή μας στον απομακρυσμένο server για την λήψη του datasheet

Συνοπτικά οι λειτουργικές απαιτήσεις της εφαρμογής

- Έξυπνο κινητό τηλέφωνο ή tablet
- Ενσωματωμένη κάμερα
- Συνδεσιμότητα με διαδίκτυο
- Ελάχιστη χωρητικότητα 15MB
- Πλοηγητή (Browser)
- Pdf Reader

### 3.2.3 Αποθήκευση δεδομένων σε SQLite

Κατά την υλοποίηση της εφαρμογής, θεωρήθηκε σκόπιμο να αποθηκεύονται κάποιες πληροφορίες οι οποίες είναι χρήσιμες για την γρήγορη ανταπόκριση της εφαρμογής. Τέτοιες πληροφορίες είναι για παράδειγμα η εταιρία κατασκευής του chip, ο μοναδικός σειριακός αριθμός του και τα υπόλοιπα στοιχεία του. Για την αποθήκευση αυτόν τον στοιχείων χρησιμοποιήσαμε την ανοιχτού κώδικα βάση δεδομένων SQLite .

Η SQLite είναι μια βάση δεδομένων ανοικτού κώδικα, υποστηρίζει το πρότυπο σχεσιακής βάσης δεδομένων και ενσωματώνει χαρακτηριστικά, όπως η σύνταξη SQL, τις συναλλαγές και τις έτοιμες καταστάσεις. Η βάση δεδομένων απαιτεί περιορισμένη μνήμη κατά το χρόνο εκτέλεσης (περίπου 250 Kbytes), χρόνος ο οποίος καθίσταται ως καλό υποψήφιο για να ενσωματωθεί σε άλλες χρόνους λειτουργίας.
Η SQLite υποστηρίζει τους τύπους δεδομένων όπως text, integer και άλλους. Όλοι οι άλλοι τύποι θα πρέπει να μετατραπούν σε ένα από αυτά τα πεδία πριν αποθηκευτούν στη βάση δεδομένων.

Χρησιμοποιώντας μια βάση δεδομένων SQLite στο Android δεν απαιτεί μια διαδικασία εγκατάστασης ή διαχείριση της βάσης δεδομένων γιατί είναι ενσωματωμένη σε κάθε συσκευή Android, το μόνο που πρέπει να καθοριστεί είναι οι δηλώσεις SQL για τη δημιουργία και την ενημέρωση της βάσης δεδομένων. Στη συνέχεια, η βάση δεδομένων διαχειρίζεται αυτόματα για σας από την πλατφόρμα Android.

Η πρόσβαση σε μια βάση δεδομένων SQLite περιλαμβάνει την πρόσβαση στο σύστημα αρχείων. Αυτή η διαδικασία μπορεί να είναι αργή. Ως εκ τούτου, συνιστάται η εκτέλεση των λειτουργιών της βάσης δεδομένων να γίνετε ασύγχρονα.

Εάν η αίτηση σας δημιουργεί μια βάση δεδομένων, η βάση αυτή είναι από προεπιλογή αποθηκευμένη στον κατάλογο DATA/data/App_name/databases/FILENAME βέβαια η πρόσβαση στη βάση δεδομένων δεν είναι δυνατή χωρίς ο χρήστης να έχει δικαιώματα διαχειριστή στη συσκευή του (root της συσκευής).

Στο επόμενο κεφάλαιο θα περιγράψουμε τον σχεδιασμό της δικής μας βάσης δεδομένων, των πινάκων που δημιουργήσαμε επίσης θα αναφέρουμε τις πληροφορίες που αποθηκεύουμε στη βάση δεδομένων και το λόγο που τις αποθηκεύουμε

## 3.3 Ανάπτυξη OCR

Αναπτύχθηκε η εφαρμογή με ονομασία “ICI” (Integrated Circuit Identifier), για να λήψη και διορθώνει τις εικόνες τροφοδοτώντας στην συνέχεια την μηχανή αναγνώρισης χαρακτήρων Tesseract OCR. Το ICI επιτρέπει στον χρήστη να επιλέξει μια εικόνα που ήδη είναι αποθηκευμένη στη Android συσκευή του ή να χρησιμοποιήσει την κάμερα της συσκευής του για την λήψη μιας νέας εικόνας. Στη συνέχεια, πραγματοποιείται διόρθωση της εικόνας και αποστολή της στην υπηρεσία Tesseract.
Κατά την ολοκλήρωση της διαδικασίας οπτικής αναγνώρισης παράγεται μια συμβολοσειρά κειμένου, η οποία και επιστρέφεται και εμφανίζετε στην αρχική οθόνη της εφαρμογής, όπου ο χρήστης έχει τη δυνατότητα αντιγραφής του κειμένου στην περιοχή αναζήτησης.
Αν ο χρήστης επιλέξει να αναζητήσει ένα ολοκληρωμένο στοιχείο, μια σειρά από αναζητήσεις στην δημιουργημένη από εμάς βάση δεδομένων πραγματοποιείται με σκοπό να εντοπίσει τις πληροφορίες του chip.
Τέλος ο χρήστης επιλέγοντας τον σύνδεσμο ένα νέο παράθυρο διαλόγου ανοίγει αυτή την φορά του πλοηγητή στο διαδίκτυο όπου το data sheet του ολοκληρωμένου στοιχείου παρουσιάζεται με δυνατότητες ανάγνωσης και λήψης του.

## 3.4 Προσαρμογή του Tesseract OCR

Όπως αναφέρεται στις συνήθεις ερωτήσεις στην ιστοσελίδα του tesseract, η αναγνώριση χαρακτήρων εμφανίζει καλά αποτελέσματα μόνο αν το ύψος των μικρών χαρακτήρων είναι τουλάχιστον 20 pixel και το κείμενο δεν είναι περιεστρεμμένο ή με πλάγιους χαρακτήρες.
Επίσης, πρέπει να προηγηθεί απομάκρυνση των ανομοιομορφιών στη φωτεινότητα του
υποβάθρου (θόρυβος χαμηλής συχνότητας), κάτι που μπορεί να επιτευχθεί με χρήση υψιπερατού φίλτρου ή σωστή κατωφλίωση της εικόνας. Τέλος πρέπει να γίνει από πριν απομάκρυνση των γραμμών και των πλαισίων γύρω από το κείμενο, αφού αυτά οδηγούν σε λανθασμένη αναγνώριση, καθώς και η όσο το δυνατόν εκμηδένιση κάθε άλλου θορύβου. Σε αυτό το κεφάλαιο παρουσιάζεται η προσπάθεια που έγινε ώστε να ικανοποιηθούν όλες αυτές οι συνθήκες.

### Αποκοπή και περιστροφή της εικόνας του πλαισίου του chip

Η εικόνα είναι πιθανό στην αρχική της μορφή να περιέχει ένα σύνολο στοιχείων πέρα από το επιθυμητό chip και να εμφανίζεται λίγο ως πολύ περιεστρεμμένη σε σχέση με τον οριζόντιο άξονα, λόγω της πιθανής κίνησης του χεριού. Όμως αυτή η μη απομόνωση του στοιχείου και η περιστροφή μπορεί να οδηγήσει σε μικρότερη πιθανότητα αναγνώρισης των χαρακτήρων. Γι' αυτό πρέπει με κάποιον τρόπο, να απομονωθεί το chip και να βρεθεί η γωνία περιστροφής της εικόνας που περιέχει το chip, έχοντας ως μοναδικό δεδομένο την ίδια την εικόνα.

Μια λύση για την περιστροφής της εικόνας είναι η χρήση του μετασχηματισμού Hough για την εύρεση των κυριότερων γραμμικών συνιστωσών της εικόνας. Όμως ο μετασχηματισμός αυτός είναι αρκετά χρονοβόρος.

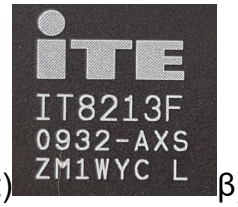
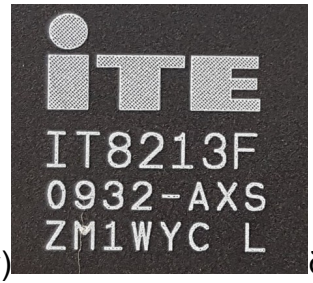
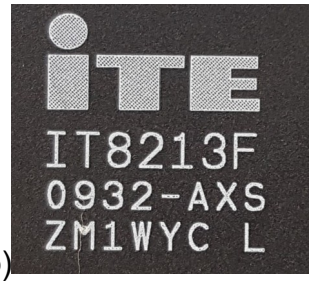
Επειδή όμως στο κομμάτι της απομόνωσης του chip επιλέχθηκε η χρησιμοποίηση των ήδη υπαρχόντων στο smartphone βιβλιοθηκών και στην περίπτωση της περιστροφής της εικόνας ακολουθήσαμε την ίδια μέθοδο.

### Προσαρμογή του μεγέθους της εικόνας

Τώρα που η εικόνα του chip απομονώθηκε και περιστράφηκε (κατά ανάγκη) ώστε να είναι όσο το δυνατόν όρθιοι οι χαρακτήρες, και πριν γίνει η κατωφλίωση, πρέπει να γίνει προσαρμογή του μεγέθους τους.

Εφόσον τα μικρά γράμματα πρέπει να έχουν ύψος τουλάχιστον 20 pixel, πρέπει να εξασφαλιστεί ότι τα κεφαλαία θα είναι μεγαλύτερα από αυτό το μέγεθος. Έτσι, γίνεται προσαρμογή του μεγέθους των φωτογραφιών έτσι ώστε όλες οι φωτογραφίες

να έχουν ύψος τουλάχιστον 32 pixel, μετά την περιστροφή και την αποκοπή του περιθωρίου τους. Το πλάτος τους διαμορφώνεται αναλόγως, ώστε μετά τη μεταβολή του μεγέθους να υπάρχει η ίδια αναλογία ύψους προς πλάτος.
Υπάρχουν πολλοί και διάφοροι μέθοδοι παρεμβολής. (“Nearest Neighbor”, “Bilinear” “Bicubic” , Lanczos3 είναι μερικές από αυτές). Εμείς θέλουμε την μέθοδο η οποία θα αλλοιώσει όσο γίνεται λιγότερο την αρχική μας εικόνα. Οι διαφορές τους παραθέτονται στο. Τα κινητά Android χρησιμοποιούν Nearest-neighbor interpolation από προεπιλογή για αυτό και αλλάζοντας το μέγεθος της εικόνας αυτή αλλοιώνεται(pixelated).

α)  β)  γ)  δ) 


Εικόνα 20: Οι διάφοροι τρόποι παρεμβολής κατά την αλλαγή μεγέθους. (α) αρχική εικόνα έχοντας υποστεί αποκοπή- απομόνωση, (β) δικυβική (bicubic) παρεμβολή παρεμβολή, (γ) παρεμβολή με τη μέθοδο Lanczos, (δ) παρεμβολή με βάση τη μέθοδο linear- bilinear.

Επιλέχθηκε να χρησιμοποιηθεί bicubic interpolation.

**Φιλτράρισμα της εικόνας**

Για τη βελτίωση των αποτελεσμάτων αναγνώρισης με το tesseract, προτείνεται η χρήση φίλτρων εξομάλυνσης του θορύβου χαμηλής συχνότητας. Φίλτρα που όμως δεν αλλοιώνουν την ένταση των ακμών. Αυτό επιτυγχάνεται με το υψιπερατό φίλτρο. Επίσης είναι χρήσιμο να εξαλειφθεί και ο θόρυβος που προέρχεται από μικρές λεπτομέρειες της εικόνας που δεν ανήκουν στους χαρακτήρες, όπως σημάδια από γρατζουνιές πάνω στην πλακέτα, σκόνη, μικρές σκιάσεις κ.α. Τη χρήση τέτοιου φίλτρου θα δούμε στο κεφάλαιο 7.

**Προσαρμογή της βάσης δεδομένων**

Κατά την πρώτη έναρξη του Tesseract OCR γίνεται φόρτωση της αγγλικής γλώσσας καθώς μόνο αγγλικά γράμματα έχουν πάνω τους τα chips και του αντίστοιχου λεξικού, δηλαδή των λέξεων που περιέχει.

Στην παρούσα εργασία όμως στόχος δεν είναι η αναγνώριση λέξεων αλλά μεμονωμένων χαρακτήρων και αριθμών. Επομένως η φόρτωση του λεξικού

απενεργοποιήθηκε. Επίσης έγιναν δοκιμές με μια ενιαία λίστα αλφαριθμητικών (αγγλικά) αλλά και με ξεχωριστές λίστες. Ως ξεχωριστές λίστες ορίζεται ο διαχωρισμός γραμμάτων και αριθμών με απώτερο σκοπό και την βελτίωση της απόδοσης από άποψη ταχύτητας εκτέλεσης της αναγνώρισης. Τελικά επιλέχθηκε η χρήση ξεχωριστών λιστών.

## 3.5 Απαλοιφή Θολώματος

Η μέθοδος απαλοιφής θολώματος που επιλέχθηκε και εστιάσαμε είναι τον Cho και Lee 2009[24]

Αυτό το κεφάλαιο αναλύει τη μέθοδο παρουσιάζοντας το μαθηματικό υπόβαθρο και τον αλγόριθμο που χρησιμοποιείται για την υλοποίηση του. Τέλος, θα παρουσιαστούν σύντομα τα αποτελέσματα του αλγορίθμου, δεδομένου ότι θα αναλυθούν διεξοδικά στο κεφάλαιο 10.

### 3.5.1 Αναφέροντας το πρόβλημα

Προκειμένου να παρασχεθεί μια μέθοδος για την αντιμετώπιση του θολώματος μιας εικόνας είναι πιο σημαντικό να να παρασχεθεί ένα πρακτικό μοντέλο θολώματος.

Πιο συγκεκριμένα η θαμπάδα μπορεί να μοντελοποιηθεί με τον ακόλουθο τρόπο:

$$B = K * L + N \tag{1-1}$$

με

B blurred image
K blur kernel or PSF
L sharp image
N noise
* convolution operation

Η προσέγγιση τυφλής αποσυνέλιξης (blind deconvolution) περιλαμβάνει εναλλασσόμενες βελτιστοποιήσεις του L και του K επαναληπτικά

$$L' = \arg\min_L \{\|B - K * L\| + \rho_L(L)\} \tag{1-2}$$

$$K' = \arg\min_K \{\|B - K * L\| + \rho_K(K)\} \tag{1-3}$$

Στο παραπάνω σύνολο εξισώσεων το pL και pK είναι κανονικοποιημένοι(regularization) όροι και ||B - K * L|| είναι οι όροι των δεδομένων.
Για τον τελευταίο όρο χρησιμοποιείται νόρμα του 2. Οι όροι ρL και ρK είναι κανονικοποιημένοι όροι για την λανθάνουσα εικόνα και πυρήνας αντίστοιχα.

Προκειμένου να επιτευχθεί απαλοιφή θολώματος της εικόνας ο πυρήνας θαμπάδας(blur kernel) ορίζεται επαναληπτικά. Το τελικό αποτέλεσμα προέρχεται από την μη-τυφλή απαλοιφή θολώματος(non-blind deblurring) της αρχικής RGB εικόνας. Οι εικόνες που παράγονται κατά τη διάρκεια της διαδικασίας εκτίμησης του πυρήνα δεν έχουν καμία επίδραση στο τελικό αποτέλεσμα. Προκειμένου να εκτιμηθεί ο πυρήνας θαμπάδας(blur kernel) απαιτείται η εκτίμηση της λανθάνουσα εικόνας.

Αυτό μπορεί να πραγματοποιηθεί με τη χρήση δύο διαφορετικών διαδικασιών:

α) με αποκατάσταση των αιχμηρών ακμών και
β) με εξάλειψη του θορύβου στις ομαλές περιοχές.

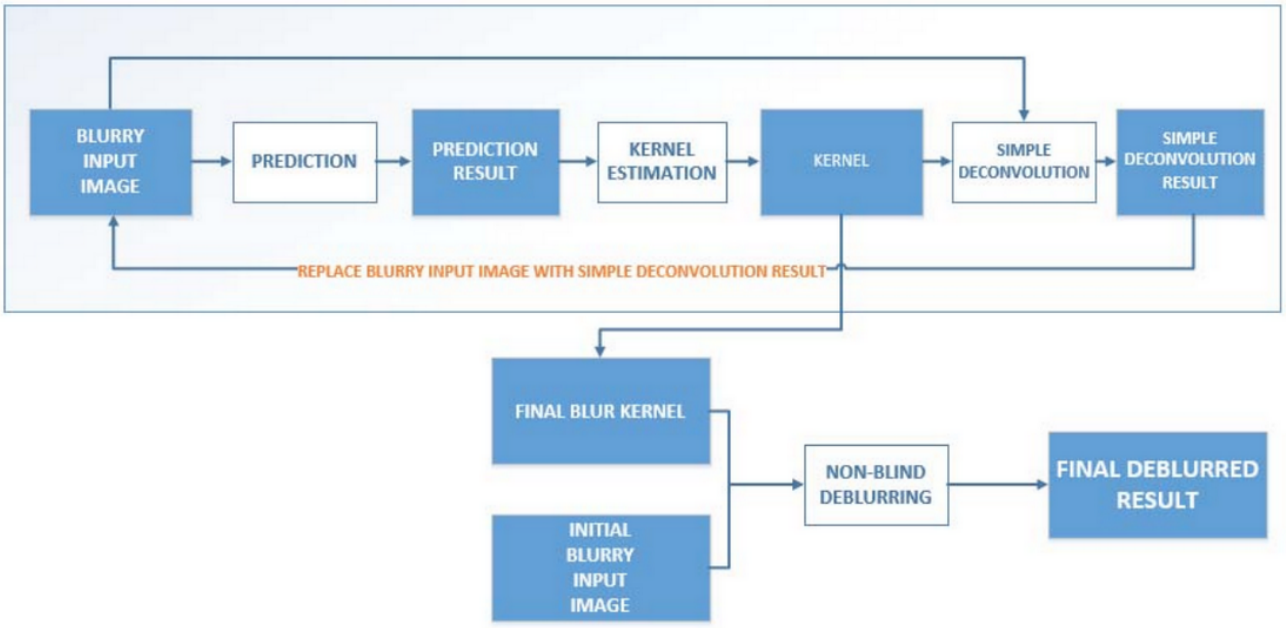
Η εξάλειψη του θορύβου είναι πολύ σημαντική, διότι διαφορετικά ο θόρυβος μπορεί να επηρεάσει τα δεδομένα ||B - K * L||. Μετά την εκτίμηση της λανθάνουσας εικόνας για ένα συγκεκριμένο αριθμό επαναλήψεων ο πυρήνας θαμπάδας(blur kernel) εκτιμάται ελαχιστοποιώντας την (1-3). Ο blur kernel χρησιμοποιείται για γρήγορη μη-τυφλή αποσυνέλιξη(fast non-blind deconvolution) της αρχικής θολής εικόνας. Το αποτέλεσμα, το οποίο εξακολουθεί να είναι μια θολή εικόνα αλλά λιγότερο θολή από την αρχική εικόνα, αντικαθιστά την θολή εικόνα εισόδου και η διαδικασία επαναλαμβάνεται έως ότου η διαφορά μεταξύ του εκτιμώμενου blur kernel της τρέχουσας επανάληψης και της προηγούμενης είναι πολύ μικρή.
Μετά την εκτίμηση του πυρήνα θαμπώματος (blur kernel) υψηλής ποιότητας deblurring εφαρμόζεται στην αρχική εικόνα.
Προκειμένου να πραγματοποιηθεί η διαδικασία πιο γρήγορα, η επαναληπτική διαδικασία εκτελείται με έναν τρόπο πολλαπλής κλίμακας καθώς αρχίζει από μια πολύ μικρή εικόνα που αλλάζει μέγεθος και γίνεται μεγαλύτερη σε κάθε επανάληψη μέχρις ότου να φτάσει στο μέγεθος της αρχικής εικόνας.

## 3.5.2 Επισκόπηση της διαδικασίας

Όπως αναφέρθηκε πριν ο blur kernel γίνεται εξευγενισμένα επαναληπτικά μέσα από μια διαδικασία που περιλαμβάνει τρία βήματα:

α) πρόβλεψη
β) διαδικασία εκτίμησης του πυρήνα (kernel) και
γ) αποσυνέλιξη (deconvolution).

Μετά την ανάκτηση του blur kernel,λαμβάνει χώρα η τελική αποσυνέλιξη (deconvolution) και έτσι αποκαλύπτεται η ευκρινής εικόνα(sharp image). Η διαδικασίες πρόβλεψης και αποσυνέλιξης (deconvolution) αποτελούν μέρος της διαδικασίας εκτίμησης της ευκρινής εικόνας.




Εικόνα 21: Επισκόπηση της διαδικασίας

### 3.5.2.1 Πρόβλεψη

Στη διαδικασία πρόβλεψης τα Px και Py της λανθάνουσας εικόνας L υπολογίζονται. Μόνο ισχυρές άκρες παραμένουν, ενώ οι άλλες περιοχές έχουν μηδενική κλίση-διαβάθμιση(gradient). Το βήμα πρόβλεψης αποτελείται από το bilateral φίλτρο και το shock φίλτρο. Κατά την πρώτη εφαρμογή bilateral φίλτρο εφαρμόζεται στην τρέχουσα εκτίμηση του L για να εξαλείψει την πιθανότητα θορύβου και των μικρών ατελειών. Στη συνέχεια χρησιμοποιείται ένα shock φίλτρο για την αποκατάσταση ισχυρών άκρων του L. Η εικόνα που προκύπτει από το φιλτράρισμα shock περιέχει ισχυρές ακμές και αυξημένο θόρυβο. Ένα shock φίλτρο, είναι ένα αποτελεσματικό φίλτρο για την βελτίωση των χαρακτηριστικών μιας εικόνας. Μπορεί να ανακτήσει αιχμηρές άκρες από βήματα θολώματος [26]. Η εξίσωση εξέλιξης ενός shock φίλτρου μορφοποιείται ως

$$I_{t+1} = I_t - sign\left(\Delta I_t\right)\left\|\nabla I_t\right\| dt$$

1-4

όπου $I_t$ είναι η εικόνα σε χρόνο t, $\Delta I_t$ είναι η Laplacian εικόνα σε χρόνο t και $|\Delta(\text{ανάποδα})I_t|_t$ είναι η διαβάθμιση.

**Εκτίμηση του πυρήνα**

Στο στάδιο της εκτίμησης του πυρήνα (kernel estimation), μόνο οι προέχων άκρες έχουν επίδραση στη διαδικασία εκτίμησης του πυρήνα. Για να εκτιμηθεί ένας blur kernel χρησιμοποιώντας τις {Px, Py}(predicted gradient maps) , η συνάρτηση energy (energy function) ελαχιστοποιείται.

$$f_k(\mathbf{k}) = \|\mathrm{K} * P_* - B_*\|^2 + \beta\|\mathrm{K}\|^2$$

1-5

Κάθε K * $P_*$ - $B_*$ σχηματίζει ένα map I. Με I να μπορεί να θεωρηθεί ως

$$\|I\|^2 = \sum_{(x,y)} I(x,y)^2$$

1-6

όπου (x, y) είναι οι δείκτες ενός pixel σε I. β είναι μια βασική μεταβλητή για Tikhonov κανονικοποίηση (regularization) [27]. Σε αυτή τη μέθοδο μόνο παράγωγα της εικόνας χρησιμοποιούνται χωρίς να συμπεριλαμβάνονται οι τιμές των pixel στη συνάρτηση ελαχιστοποίησης(minimization function). Επιπλέον, η συνάρτηση ελαχιστοποίησης περιλαμβάνει έναν Tikhonov κανονικοποιημένο όρο.

Η εξίσωση για τον υπολογισμό του πυρήνα (kernel estimation ) μπορεί να γραφτεί με τον ακόλουθο τρόπο [24]:

$$fk(\mathbf{k}) = \|\mathbf{Ak} - \mathbf{b}\|^2 + \beta\|\mathbf{k}\|^2 = (\mathbf{Ak} - \mathbf{b})^T(\mathbf{Ak} - \mathbf{b}) + \beta\mathbf{k}^T\mathbf{k}$$

1-7

Όπου:

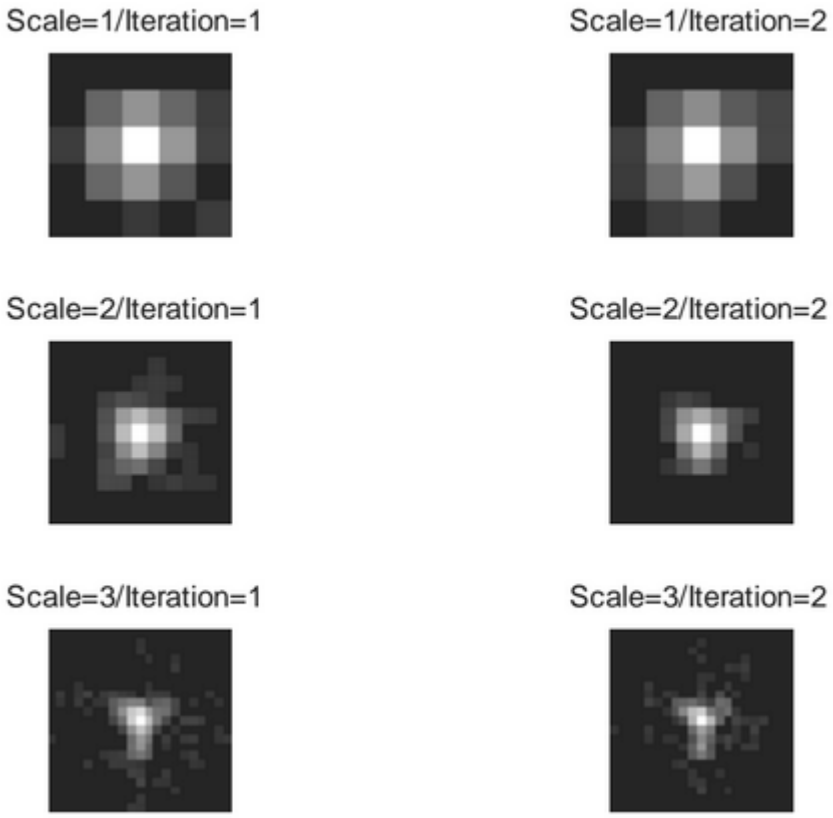
A είναι μια αναπαράσταση μήτρας(matrix) του P
k είναι μια αναπαράσταση μήτρας(matrix) του blur kernel
b είναι μια αναπαράσταση μήτρας(matrix) του B

Για την ελαχιστοποίηση της f(k) μία μέθοδος διαβάθμισης σύζευξης(conjugate gradient method) χρησιμοποιείται. Έτσι, η διαβάθμιση της f (k) στην εξίσωση (1-8)

διαμορφώνεται κατά τον ακόλουθο τρόπο:

$$\frac{\partial f_k(\mathbf{k})}{\partial \mathbf{k}} = 2\mathbf{A}^T\mathbf{A}\mathbf{k} + 2\beta\mathbf{k} - 2\mathbf{A}^T\mathbf{b}$$

1-8




Εικόνα 22: Πυρήνας επαναληπτικής βελτίωσης(refinement) [24]

### 3.5.2.2 Αποσυνέλιξη (Deconvolution)

Στο στάδιο αποσυνέλιξης(deconvolution), εκτιμάτε η λανθάνουσα εικόνα L από έναν δεδομένο- συγκεκριμένο πυρήνα(kernel) K και η θολή εικόνα που έχει δοθεί ως είσοδος B Χρησιμοποιούμε τη συνάρτηση energy

$$f_L(L) = \sum_{\vartheta_*} \omega_* \left\| \mathrm{K} * \partial_* L - \partial_* B \right\|^2 + \alpha \left\| \nabla L \right\|^2$$

1-9

όπου $\vartheta_* \in \{\vartheta_0, \vartheta_x, \vartheta_y, \vartheta_{xx}, \vartheta_{yy}, \vartheta_{xy}\}$ συμβολίζει τη μερική παράγωγο φορέα σε διαφορετικές κατευθύνσεις και εντολές $\omega_* \in \{\omega_0, \omega_1, \omega_2\}$ είναι βασικό για κάθε μερική παράγωγο, και είναι βασικό για την κανονικοποίηση.

Ο πρώτος όρος στη συνάρτησης energy βασίζεται στο μοντέλο blur του Shan το 2008, η οποία χρησιμοποιεί παράγωγα της εικόνα για τη μείωση των σημείων ενδιαφέροντος. Ο όρος συστηματοποίηση ενισχύει το $||\Delta(\text{ανάποδα})L||^2$ με ομαλή διαβάθμιση[28]. Η ανάκτηση της λανθάνουσας εικόνας L εκτελείται με βελτιστοποίηση της εξίσωσης (1-10). Αυτό επιτυγχάνεται με μια πολύ γρήγορη pixel-wise διαίρεση στο πεδίο της συχνότητας, η οποία χρειάζεται μόνο δύο FFTs [24].

Βελτιστοποιώντας την εξίσωση (1-10), μπορεί να μην έχουμε αποτελέσματα υψηλής ποιότητας, σε σύγκριση με εξελιγμένες μεθόδους αποσυνέλιξης [28], [29], [30], και τα αποτελέσματα μπορεί να περιέχουν εξομαλυμένα τα άκρα και τα σημεία ενδιαφέροντος. Ωστόσο, λόγω του σταδίου πρόβλεψης που οξύνει τις άκρες και απορρίπτει μικρές λεπτομέρειες, αυτή η απλή αποσυνέλιξη δημιουργεί ικανοποιητικά αποτελέσματα.

### 3.5.2.3 Τελική αποσυνέλιξη (Deconvolution)

Η τελική αποσυνέλιξη είναι το τελικό βήμα για την εξαγωγή της ευκρινούς εικόνας. Δεν υπάρχει διαφορά μεταξύ της μεθόδου που χρησιμοποιείται για την αποσυνέλιξη κατά τη διάρκεια επαναλήψεων και τη μέθοδο που χρησιμοποιείται για την τελική αποσυνέλιξη. Η τελική αποσυνέλιξη εφαρμόζεται σε κάθε κανάλι(channel) της αρχικής θολής εικόνα χρησιμοποιώντας τον επανεκτιμώμενο πυρήνα που έχει υπολογιστεί επαναληπτικά.

Μετά την τελική αποσυνέλιξη η ευκρινής εικόνα ανακτάται. Η μέθοδος που αναλύονται σε αυτό το κεφάλαιο πρόκειται να υλοποιηθεί σε την χρήση της γλώσσας C ++, προκειμένου να ενσωματωθούν την Android εφαρμογή.

## 3.6 Υλοποίηση

Αυτή η ενότητα αναλύει την κύρια κατηγορία που χρησιμοποιείται για την υλοποίηση του απαλοιφής θολώματος με έναν αντικειμενοστραφή τρόπο. Συγκεκριμένα, η κλάση deblur (deblur class) περιέχει όλες τις παραμέτρους που απαιτούνται για την προετοιμασία της διαδικασίας απαλοιφής θολώματος και όλες τις λειτουργίες για να εκτελέσει την απαλοιφή θολώματος της εισαγόμενης εικόνας. Στις ακόλουθες υπο-ενότητες θα παρουσιαστούν τα λειτουργικά μέλη (function members) και τα μέλη δεδομένων (data members) της κλάσης deblur.

### Μέλη Δεδομένων (Data Members)

Η κλάση deblur περιλαμβάνει τις ακόλουθες κατηγορίες μελών δεδομένων(data members).
Data members που συσχετίζονται με γενικούς σκοπούς, το shock φίλτρο, με το bilateral φίλτρο, με την αποσυνέλιξη (deconvolution), με την εκτίμηση του πυρήνα (kernel estimation), και με την έξοδο. Τα μέλη δεδομένων γενικού σκοπού είναι το όνομα αρχείου της εικόνας RGB, και οι αποχρώσεις του γκρι της μήτρας της εικόνα. Τα μέλη δεδομένων του shock φίλτρου και του bilateral φίλτρου περιλαμβάνουν παραμέτρους και μήτρες που χρησιμοποιούνται για την αποθήκευση του αποτελέσματος των φίλτρων. Τα μέλη των δεδομένων που σχετίζονται με την αποσυνέλιξη (deconvolution) περιλαμβάνουν παραμέτρους της διαδικασίας της αποσυνέλιξης. Τα μέλη των δεδομένων που σχετίζονται με τον πυρήνα περιλαμβάνουν μόνο το αρχικό ακέραιο μέγεθος. Τέλος, τα μέλη των δεδομένων που αφορούν την έξοδο της εικόνας περιλαμβάνουν την ξεθολωμένη εικόνα και τις τελικές μήτρες PSF.

### Λειτουργικά μέλη (Function Members)

Τα λειτουργικά μέλη χωρίζονται σε διάφορες κατηγορίες. Αυτές είναι οι λειτουργίες προετοιμασίας, οι λειτουργίες I / O, οι επαναληπτικές λειτουργίες απαλοιφής θολώματος, το φίλτρο και οι λειτουργίες μετασχηματισμού του Fourier, οι σχετικές λειτουργίες που σχετίζονται με τη λειτουργία της οπτικής μεταφοράς, οι σχετικές λειτουργίες της εκτίμησης του πυρήνα , η λειτουργία αποσυνέλιξης, τέλος η συνάρτηση γκριζαρίσματος (grayscale) αλλά και η συνάρτηση γεμίσματος (padding).

- void analyzeKernel(void)

Η λειτουργία αυτής της συνάρτησης είναι η ανάλυση του πυρήνα με βάση το αρχικό μέγεθος του αριθμού των επιπέδων που χρησιμοποιήθηκαν για την επαναληπτική deblurring συμπεριλαμβανομένου του μεγέθους του πυρήνα για κάθε επίπεδο. Έτσι, αυτή η λειτουργία βασίζεται στην πυραμίδα του πυρήνα.

- void resizeBlurredForScaling (void)

Η συνάρτηση αυτή αλλάζει το μέγεθος της θολής εικόνα στην ελάχιστη κλίμακα, έτσι ώστε να

αρχίσει η διαδικασία deblurring.

- void readImage(void)

Αυτή η συνάρτηση διαβάζει την εικόνα εισόδου και την ομαλοποιεί από mat 0 σε 1.

- void iterativeDeblurring(void)

Η συνάρτηση deblurring είναι η κύρια επαναληπτική συνάρτηση. Για κάθε επανάληψη προβλέπεται η λανθάνουσα εικόνα και εκτιμάται ο πυρήνας. Μετά την ολοκλήρωση της επανάληψης, η έξοδος αλλάζει μέγεθος σύμφωνα με την μεγαλύτερη κλίμακα. Πρέπει να σημειωθεί ότι οι κλίμακες έχουν προκαθοριστεί στην ενότητα initializations

- void getImagePSF(Mat &OriginalBlurredScaled, Mat &OriginalBlurredScale, Mat &PSF, int &iterScaleNum, bool &isFinal, Mat &OriginalBlurred, double &sigmaRange);

Η συνάρτηση αυτή δέχεται ως είσοδο την θολή εικόνα αφού έχει υποστεί κλιμάκωση για την τρέχουσα επανάληψη, ένα άδειο Mat για τον πυρήνα blur που έχει υποστεί αλλαγή μεγέθους για την τρέχουσα κλίμακα επανάληψης, μια μεταβλητή flag καθορίζει εάν η τρέχουσα επανάληψη είναι τελευταία ή όχι και τις παραμέτρους για το bilateral φίλτρο.

- Mat shockfilter(Mat IO, int iteration, double dt, double h);

Η λειτουργία του φίλτρου shock είναι μια εφαρμογή OpenCV μια εργασίας του Guy Gilboa το οποίο βασίζεται στο έργο του Rudin και Osher [4] .

- Mat fft2(Mat input);

Η συνάρτηση fft2 είναι η ισοδύναμη συνάρτηση για την γρήγορο μετασχηματισμό Fourier εικόνας. Υλοποιείται με τη χρήση της συνάρτηση DFT OpenCV.

- Mat ifft2(Mat input);

Η συνάρτηση αυτή χρησιμοποιεί την αντίστροφη dft συνάρτηση όπως είναι στη βιβλιοθήκη OpenCV προκειμένου να εφαρμοστεί η αντίστροφη fft συνάρτηση για τις εικόνες.

- Mat psf2otf(Mat inputPsf, Size finalSize);

Η psf2otf παράγει την συνάρτηση οπτικής μεταφοράς (optical function) από ένα συγκεκριμένο πυρήνα blur.

- x Mat otf2psf(Mat inputOtf, Size outSize);

Η otf2psf συνάρτηση δημιουργεί το psf λαμβάνοντας ως είσοδο μια συνάρτηση οπτικής μεταφοράς (optical function).

- Mat estimateKernel(Mat Prediction, Mat OriginalBlurredScaledSingleChannel, Mat PSF, int numberOfIterations);

Η συνάρτηση estimateKernel υπολογίζει τον πυρήνα θαμπάδας (blur kernel) ως είσοδο παίρνει την προβλεπόμενη λανθάνουσα εικόνα και τη θολή εικόνα. Η λειτουργία γίνεται επαναληπτικά. Για κάθε επανάληψη της συνάρτηση η get K συνάρτηση εκτελείται δίνοντας τον πυρήνα

- Mat getKMulSpectrumSupport(Mat PredictionPadded, Mat BlurredPadded, Mat PSF, double belta);

Η συνάρτηση get kernel επιστρέφει τον πυρήνα θαμπάδας (blur kernel) και ως είσοδο παίρνει την προβλεπόμενη λανθάνουσα εικόνα και τη θολή εικόνα (blurry image).

- Mat deltaKernel(int s);

Η συνάρτηση deltaKernel δημιουργεί ένα πυρήνα που αποτελείται από μηδενικά με μόνο μία κορυφή στο κέντρο του πυρήνα ίση με την μέγιστη τιμή που είναι 1 στην προκειμένη περίπτωση.

- Mat paddArray(Mat input, Size padding, std::string method, std::string direction);

Η συνάρτηση paddArray εφαρμόζει padding σε μια ορισμένη μήτρα. Η συνάρτηση αυτή δέχεται ως είσοδο τον πίνακα εισόδου, το μέγεθος padding και το είδος του padding. Η τρέχουσα λειτουργία υλοποιεί δύο τύπους padding: Zero padding και replicate padding ίσο με τις τιμές των pixel που είναι οι οριακές τιμές της εικόνας.

- Size getPadSize(Mat f);

Αυτή η συνάρτηση επιστρέφει το εκτιμώμενο απαιτούμενο μέγεθος padding για μια εικόνα αποθηκευμένη στην μήτρα f`

- Mat r2g(Mat input);

Αυτή η συνάρτηση μετατρέπει μια εικόνα σε grayscale, αν η εικόνα βρίσκεται στο χρωματικό χώρο RGB.

- Mat deconv(Mat &Blurred, Mat &PSF, double &w0alpha);

Η συνάρτηση deconv αποσυνελίσσει (deconvolves) τη θολή εικόνα με τον επιλεγμένο πυρήνα. Αν η εικόνα είναι “έντονη εικόνα ”, τότε ο αποσυνέλιξη εκτελείται μόνο για ένα κανάλι (channel). Αν η εικόνα είναι μια εικόνα RGB η αποσυνέλιξη εκτελείται λαμβάνοντας υπόψη την εικόνα πολλαπλών καναλιών(channel). Για να εκτελεστεί η συνάρτηση αποσυνέλιξης deconvFn καλείται.

- Mat deconvFn(Mat &Blurred, Mat &PSF, double &w0alpha);

Αυτή η συνάρτηση είναι η καρδιά της διαδικασίας της αποσυνέλιξης. Λαμβάνει ως είσοδο την θολή εικόνα και το PSF και εκτελεί την αποσυνέλιξη. Η παράμετρος alpha είναι ένας παράγοντας που καθορίζει πόσο έντονη η αποσυνέλιξης θα είναι. Για ένα μεγάλο alpha η τελική εικόνα μπορεί να καταστραφεί. Για ένα μικρό άλφα η εικόνα θα παραμείνει ακόμα θολή. Κατά τη διάρκεια των πειραμάτων αποφασίσαμε ότι η βέλτιστη alpha παράμετρος πρέπει να κυμαίνεται μεταξύ 0,5 και 1.

### Συναρτήσεις γενικού σκοπού

Οι συναρτήσεις γενικού σκοπού είναι πιο γενικές στον τρόπο που επεξεργάζονται τα δεδομένα εισόδου. Αυτές παρουσιάζονται στην παρακάτω λίστα. Στην ενότητα αυτή περιγράφεται η λειτουργική περιγραφή της κάθε συνάρτησης.

- Mat equalizeIntensity(const Mat& inputImage);

Αυτή η συνάρτηση εξισώνει την ένταση του ιστογράμματος μιας εικόνας με float ή double format που κυμαίνονται μεταξύ του 0 και του 1.

- Mat getChannel(Mat input, int theChannel);

Αυτή η συνάρτηση επιστρέφει ένα ενιαίο πίνακα καναλιού ίσο με το επιλεγμένο κανάλι ενός πλέγματος πολλαπλών καναλιών.

- Mat conjMat(Mat src);

Αυτή η συνάρτηση επιστρέφει το συζυγή ενός σύνθετου OpenCV πλέγματος-πίνακα-μήτρα.

- void computeDft(Mat& image, Mat& dest);

Αυτή η συνάρτηση υπολογίζει τον αντίστροφο DFT ενός πίνακα εισόδου.

- void computeIDft(Mat& complex, Mat& dest);

Αυτή η συνάρτηση υπολογίζει το DFT ενός πίνακα εισόδου.

- Mat divideComplexMats(Mat InputA, Mat InputB);

Αυτή η συνάρτηση εκτελεί μια διαίρεση ανά στοιχείο από μονά κανάλια με δεδομένο ότι κάθε μήτρα περιέχει μιγαδικούς αριθμούς. Χρησιμοποιεί OpenCV object Mat ως είσοδο και έξοδο

- void divSpectrums(InputArray _srcA, InputArray _srcB, OutputArray _dst, int flags, bool conjB);

Αυτή η συνάρτηση εκτελεί μια διαίρεση ανά στοιχείο από μονά κανάλια με δεδομένο ότι κάθε μήτρα περιέχει μιγαδικούς αριθμούς. Χρησιμοποιεί OpenCV object inputArray ως είσοδο και έξοδο.

- void shift(const cv::Mat& src, cv::Mat& dst, cv::Point2f delta, int fill = cv::BORDER_CONSTANT, cv::Scalar value = cv::Scalar(0, 0, 0, 0));

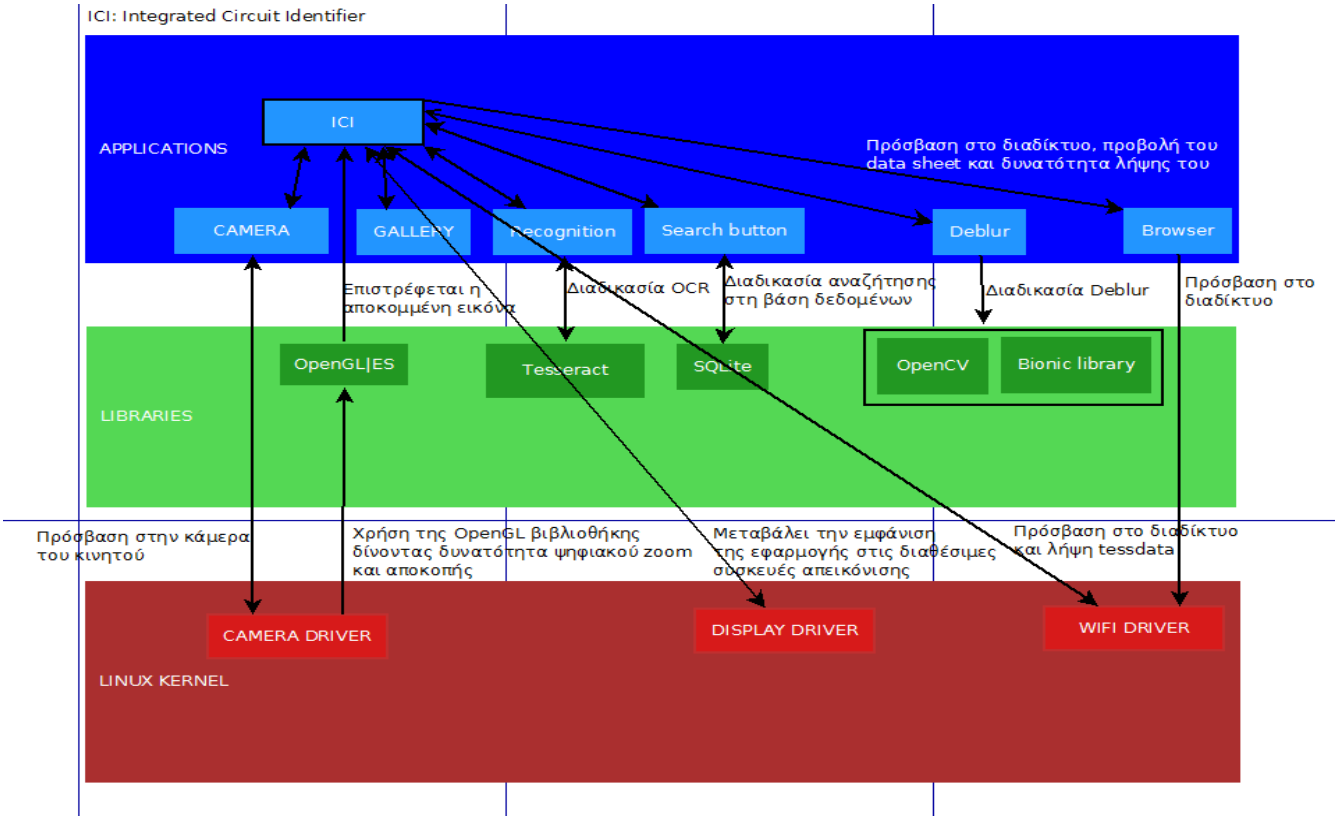
Αυτή η συνάρτηση αλλάζει κυκλικά τα pixels μιας εικόνας ή της μήτρας προς την επιλεγμένη κατεύθυνση.

- void bilateralFilter(InputArray src, OutputArray dst, double sigmaColor, double sigmaSpace);

Αυτή η συνάρτηση εφαρμόζει ένα bilateral φίλτρο στην επιλεγμένη εικόνα.

## 4.1 Ανάπτυξη για την πλατφόρμα Android

### Διάγραμμα της εφαρμογής




Εικόνα 23: Διάγραμμα της εφαρμογής σε πλήρη αντιστοιχία με την αρχιτεκτονική Android

## 4.1.2 Υλοποίηση των ενοτήτων της εφαρμογής

Η αρχική πρόθεση της δημιουργίας μιας εφαρμογής αναγνώρισης chip ήταν η βελτιωθεί της εμπειρίας του χρήστη, όταν αυτός επέλεγε να αναγνωρίσει ένα chip με όχι τον συμβατικό τρόπο δηλαδή μεγεθυντικό φακό, καταγραφή των αναγνωριστικών και έπειτα αναζήτηση στο διαδίκτυο.

## 4.1.2.1 Android Υλοποίηση OCR

Δεδομένου ότι το Tesseract είναι γραμμένο στη γλώσσα προγραμματισμού C++ δεν είναι εύκολο έργο η χρήση του στο Android, λειτουργικό βασισμένο στη Java.

Ο κώδικας C++ πρέπει να ενθυλακωθεί σε μια κλάση Java και η εκτέλεση του να γίνεται εγγενώς μέσω του Native Interface Java (JNI).
Αν και απαιτείται κάποια προσπάθεια, ένα μεγάλο όφελος της εκτέλεσης του Tesseract εγγενώς είναι οτι η C++ είναι αισθητά πιο γρήγορη από την Java.

Για την ενσωμάτωση της βιβλιοθήκη Tesseract OCR στο Android, χρησιμοποιήθηκαν τα εργαλεία tesseract-android-tool που παρέχονται από την Google.

Στη συνέχεια ανοίγοντας το README, και ακολουθήθηκαν τα παρακάτω βήματα

```
cd <project-directory>
curl -O https://tesseract-ocr.googlecode.com/files/tesseract-ocr-3.02.02.tar.gz
curl -O http://leptonica.googlecode.com/files/leptonica-1.69.tar.gz
tar -zxvf tesseract-ocr-3.02.02.tar.gz
tar -zxvf leptonica-1.69.tar.gz
rm -f tesseract-ocr-3.02.02.tar.gz
rm -f leptonica-1.69.tar.gz
mv tesseract-3.02.02 jni/com_googlecode_tesseract_android/src
mv leptonica-1.69 jni/com_googlecode_leptonica_android/src
ndk-build -j8
android update project --target 1 --path .
ant debug (release)
```

Σημειώνετε ότι η χρήση NDK r9 οδηγεί σε μήνυμα λάθους και αποτυχία. Η λύση βρίσκεται στο αρχείο **Application.mk** όπου με την προσθήκη της ακόλουθης γραμμής κώδικα το σφάλμα διορθώνετε.

```
APP_CFLAGS += -Wno-error=format-security
```

Ενσωματώνοντας επιτυχώς την βιβλιοθήκη OCR, στο φάκελο libs έχει εισαχθεί το αρχείο class.jar.

Πρώτο βήμα της υλοποίησης είναι η δημιουργία ενός Android project και η εισαγωγή των σχετικών βιβλιοθηκών όπως φαίνεται παρακάτω.

```
public class TessOCR {
   private TessBaseAPI mTess;

   public TessOCR() {
      // TODO Auto-generated constructor stub
      mTess = new TessBaseAPI();
      String datapath = Environment.getExternalStorageDirectory() + "/tesseract/";
```

```
        String language = "eng";
        File dir = new File(datapath + "tessdata/");
        if (!dir.exists())
            dir.mkdirs();
        mTess.init(datapath, language);
    }
}
```

Η διεύθυνση των tessdata στον constructor δεν πρέπει να είναι λανθασμένη. Σε κάθε άλλη περίπτωση μια εξαίρεση (exception) θα εμφανιστεί στην init().

```
 public boolean init(String datapath, String language) {
        if (datapath == null) {
            throw new IllegalArgumentException("Data path must not be null!");
        }
        if (!datapath.endsWith(File.separator)) {
            datapath += File.separator;
        }

        File tessdata = new File(datapath + "tessdata");
        if (!tessdata.exists() || !tessdata.isDirectory()) {
            throw new IllegalArgumentException("Data path must contain subfolder
tessdata!");
        }

        return nativeInit(datapath, language);
    }
```

Φυσικά στη συνέχεια το πρώτο πράγμα που χρειάστηκε για ένα OCR app σαν το δικό μας είναι ένας μηχανισμός οποίος φορτώνει μια εικόνα για επεξεργασία. Ο εξυπνότερος τρόπος να γίνει αυτό ήταν η εντολή για χρήση των είδη προ εγκατεστημένων σε κάθε κινητό εφαρμογών κάμερας ή συλλογής φωτογραφιών.
Αν η εφαρμογή εκτελεστεί από τον προσομοιωτή και δεν υπάρχει διαθέσιμη καμία φυσική μηχανή-κάμερα η επιλογή "Λήψη φωτογραφίας" δεν λειτουργεί.

Σε κάθε μια από τις παραπάνω δύο περιπτώσεις αυτό που απαιτείται είναι η αποκωδικοποίηση της διεύθυνσης αποθήκευσης(URI) της εικόνας όπως φαίνεται παρακάτω.

```
if (Intent.ACTION_SEND.equals(intent.getAction())) {
    Uri uri = (Uri) intent.getParcelableExtra(Intent.EXTRA_STREAM);
    uriOCR(uri);
```

```
}
private void uriOCR(Uri uri) {
        if (uri != null) {
            InputStream is = null;
            try {
                is = getContentResolver().openInputStream(uri);
                Bitmap bitmap = BitmapFactory.decodeStream(is);
                mImage.setImageBitmap(bitmap);
                doOCR(bitmap);
            } catch (FileNotFoundException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            } finally {
                if (is != null) {
                    try {
                        is.close();
                    } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                    }
                }
            }
        }
}
```

Έχοντας την δυνατότητα πλέον πρόσβασης στην διεύθυνση της εικόνας αρκεί για την φόρτωση της η λήψη της ή η επιλογή της από την συλλογή.

Για την λήψη εικόνας επιλέχθηκε η επιβολή της εφαρμογής να χρησιμοποιήσει μια από τις είδη εγκατεστημένες εφαρμογές κάμερας(δίνεται στον χρήστη να επιλέξει την αγαπημένη του). Υλοποιώντας το κατά αυτό τον τρόπο εξασφαλίζουμε ότι η εικόνα που θα παρθεί θα είναι όσο το δυνατόν κρυστάλλινη αξιοποιώντας πλήρως  τα προηγμένα τεχνικά χαρακτηριστικά που παρέχει μια εταιρία κινητής τηλεφωνίας στον χρήστη.

**Λήψη εικόνας**

```
Intent takePicIntent = new Intent("android.media.action.IMAGE_CAPTURE");
startActivityForResult(takePicIntent, CAMERA_REQUEST);
```

**Επιλογή της από την συλλογή**

```
Intent = new Intent
(Intent.ACTION_PICK,android.provider.MediaStore.Images.Media.EXTERNAL_CONT
ENT_URI);
startActivityForResult(i, RESULT_LOAD_IMAGE);
```

Για την βελτίωση της αναγνώρισης του ολοκληρωμένου κρίθηκε σκόπιμη η απομόνωση του. Αφαιρώντας το περιεχόμενο στο πίσω μέρος της εικόνας μειώνετε και η πιθανότητα εσφαλμένης αναγνώρισης.

```
i.setType("image/*");
i.putExtra("crop", "true");
i.putExtra("outputX", 200);
i.putExtra("outputY", 200);
i.putExtra("aspectX", 1);
i.putExtra("aspectY", 1);
i.putExtra("scale", true);
i.putExtra(MediaStore.EXTRA_OUTPUT, image_uri);
i.putExtra("outputFormat",
```

Αν μια εικόνα είναι πολύ μεγάλη ή πολύ μικρή, το Tesseract μπορεί να επιστρέψει κακά αποτελέσματα ή ακόμα και να τερματίσει απροσδόκητα εμφανίζοντας μήνυμα σφάλματος EXC_BAD_ACCESS.
Για τη λύση αυτού του προβλήματος όπως είδαμε στο κεφάλαιο 3 εφαρμόστηκε μια μέθοδος αλλαγής μεγέθους της εικόνας χωρίς όμως να επηρεάζει τις αναλογίες της, έτσι η εικόνα στρεβλώνεται όσο το δυνατόν λιγότερο.

Προς αποφυγή της αλληλεπίδρασης του χρήστη με την εφαρμογή ενώ το Tesseract πραγματοποιεί διεργασίες η αλληλεπίδραση απενεργοποιήθηκε και ως αντάλλαγμα εμφανίζετε ένα πλαίσιο διαλόγου ζητώντας από τον χρήστη να περιμένει.

## 4.1.2.2 Android Υλοποίηση Deblur

Στην ενότητα αυτή παρουσιάζουμε την εφαρμογή του αλγορίθμου που παρουσιάζεται στο Κεφάλαιο 3 για τη πλατφόρμα Android. Η υλοποίηση αποτελείται από δύο ενότητες. Τις ενότητες openCVDeblur και openCVLibrary300. Η openCVLibrary έχει ληφθεί από την επίσημη ιστοσελίδα της OpenCV ως “prebuilt” και στη συνέχεια εισήχθη στο Android project. Το openCVDeblur project περιέχει τον πηγαίο κώδικα της Java για το περιβάλλον εργασίας του χρήστη (user interface-UI) και τον πηγαίο κώδικας της C ++ που εκτελεί-πραγματοποιεί τη διαδικασία απαλοιφής θολώματος. Συγκεκριμένα ο φάκελος περιέχει το Java DeblurActivity.java και το UserPicture.java.

Για την C / C ++, ο κώδικας βρίσκεται στο φάκελο jni. Το αρχείο jni_part.cpp περιέχει τις λειτουργίες που παρέχει τη διεπαφή(interface) μαζί με τον κώδικα Java, την deblur.cpp και την deblur.h αρχεία τα οποία περιέχουν τον πηγαίο κώδικα, κώδικας όπως περιγράφτηκε και αναλύθηκε στο κεφάλαιο 3.

**Java-Native Interface (JNI)**

Η JNI λειτουργεί ως γέφυρα μεταξύ του κώδικα Java και τον κώδικα γραμμένο σε C + +. Το jni_part.cpp αρχείο περιέχει τη λειτουργία που αναγνωρίζεται από το αρχείο main activity του του προγράμματος(DeblurActivity.java)

**Native C++**

Η C / C ++ υλοποίηση παρουσιάζεται στον φάκελο jni στα deblur.cpp και deblur.h αρχεία. Η jni_part.cpp καλεί τη συνάρτηση moutput = deblur (minput) που αρχικοποιεί τις παραμέτρους για deblurring και καλεί τη συνάρτηση blinddeblurmap.

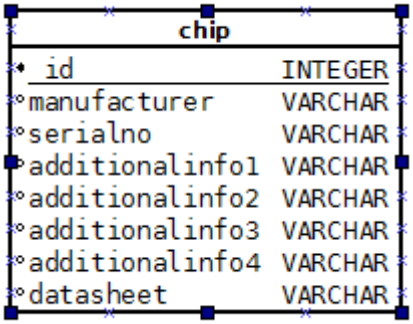
Η πηγαία υλοποίηση της λειτουργίας απαλοιφής θολώματος καλείται μέσα από τη java-native-interface functionality.

## 4.1.2.3 SQLite Βάση Δεδομένων

Κατά την υλοποίηση της εφαρμογής, θεωρήθηκε σκόπιμο να αποθηκεύονται κάποιες πληροφορίες οι οποίες είναι χρήσιμες για τον χρήστη της εφαρμογής. Τέτοιες πληροφορίες είναι για παράδειγμα η εταιρεία κατασκευής και ο συριακός αριθμός του chip. Για την αποθήκευση αυτόν τον στοιχείων χρησιμοποιήσαμε η βάση δεδομένων SQLite .
Η SQLite είναι μια πανίσχυρη και συνάμα πολύ ελαφριά βάση δεδομένων ανοικτού κώδικα, υποστηρίζει το πρότυπο σχεσιακής βάσης δεδομένων και ενσωματώνει χαρακτηριστικά, όπως η σύνταξη SQL, τις συναλλαγές και τις έτοιμες καταστάσεις. Η βάση δεδομένων απαιτεί περιορισμένη μνήμη κατά το χρόνο εκτέλεσης (περίπου 250 Kbytes), το οποίο το καθιστά ένα καλό υποψήφιο από το να ενσωματωθεί σε άλλες χρόνους λειτουργίας.
Η SQLite υποστηρίζει τους τύπους δεδομένων όπως text,integer και άλλους που μας αρκούν για τις ανάγκες της εφαρμογής μας. Όλοι οι άλλοι τύποι θα πρέπει να μετατραπούν σε ένα από αυτά τα πεδία πριν αποθηκεύουν στη βάση δεδομένων.
Χρησιμοποιώντας μια βάση δεδομένων SQLite στο Android δεν απαιτείτε καμία διαδικασία εγκατάστασης ή διαχείριση της βάσης δεδομένων γιατί είναι ενσωματωμένη σε κάθε συσκευή Android, το μόνο που πρέπει να καθοριστεί είναι οι δηλώσεις SQL για τη δημιουργία και την ενημέρωση της βάσης δεδομένων. Στη συνέχεια, η βάση δεδομένων διαχειρίζεται αυτόματα από την πλατφόρμα Android.

Παρακάτω παρατίθενται οι βιβλιοθήκες που προστέθηκαν στην εφαρμογή.
android.database.Cursor;
android.database.SQLException;
android.database.sqlite.SQLiteDatabase;
android.database.sqlite.SQLiteOpenHelper;

Το σχήμα της τοπικής βάσης δεδομένων SQLite που υποστηρίζεται στην τρέχουσα υλοποίηση, είναι:




Εικόνα 24: Σχήμα της τοπικής βάσης δεδομένων SQLite

Τονίζεται ότι τα δεδομένα δεν έχουν ομογενοποιηθεί λόγω σχεδιαστικής απόφασης, προκειμένου να παρουσιαστεί ένας τρόπος υποστήριξης δεδομένων ετερογενούς προέλευσης.

## 4.1.2.4 Διεπαφή με τον χρήστη (User Interface UI)

Το user interface είναι “χτισμένο” στα αρχεία xml μέσα στο φάκελο layout και στον φάκελο menu και κάθε μέρος της διεπαφής είναι συνδεδεμένη με μια λειτουργικότητα στο αρχείο MainActivity.java ή οποιοδήποτε άλλο αρχείο πηγαίου κώδικα java.

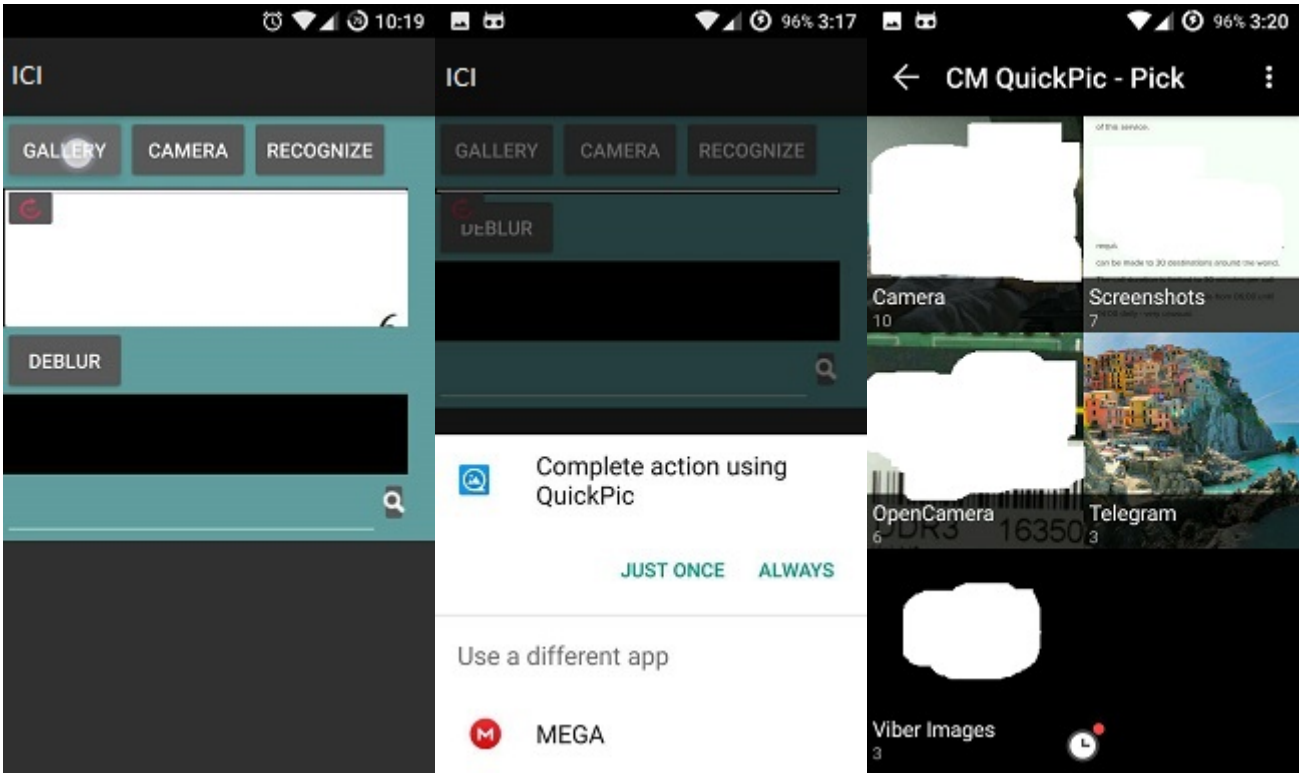
Στο περιβάλλον εργασίας του χρήστη η αλληλεπίδραση με την εφαρμογή γίνεται με τέσσερις βασικές επιλογές. GALLERY, CAMERA, RECOGNIZE,DEBLUR και ένα εικονίδιο που υποδηλώνει αναζήτηση.

Όταν η εφαρμογή ξεκινάει για πρώτη φορά κάνει λήψη των trainning data από την επίσημη ιστοσελίδα της Google για το tesseract.

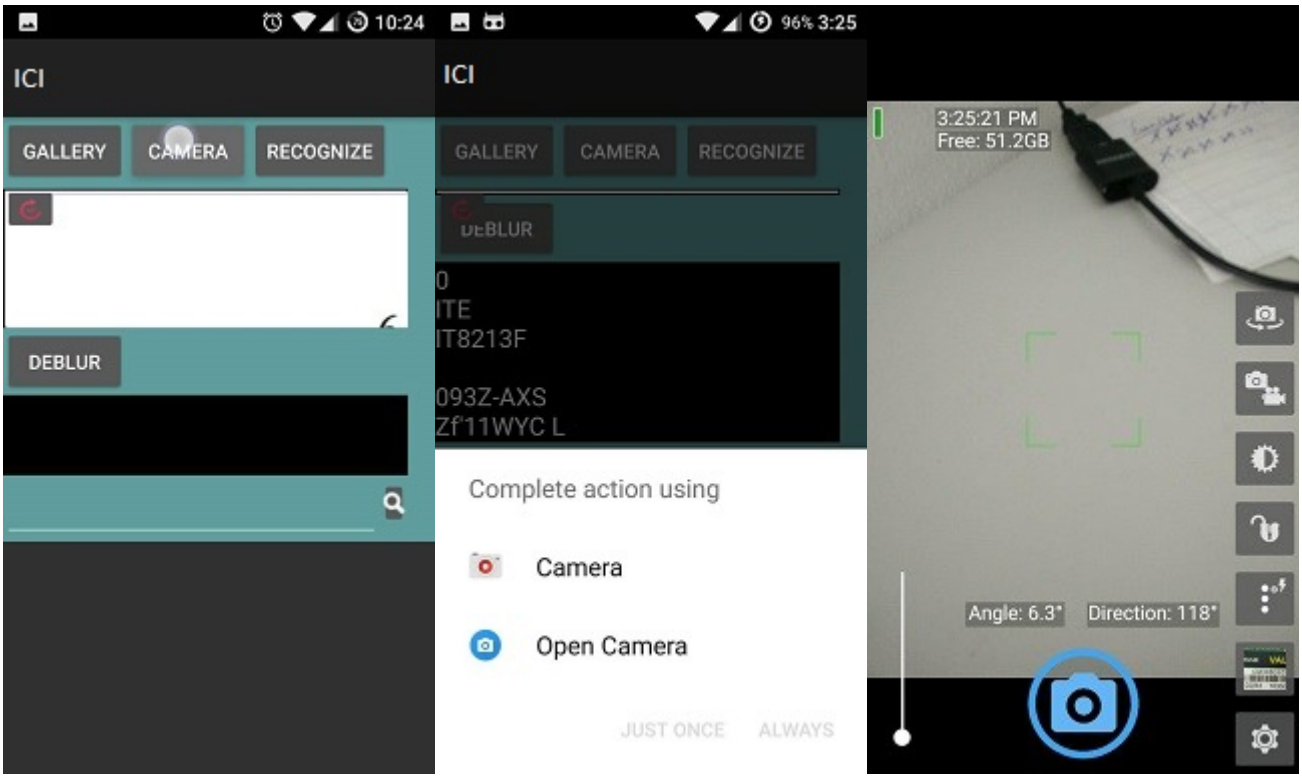
Στην συνέχεια εμφανίζεται η αρχική εικόνα διεπαφής της εφαρμογής όπως φαίνεται παρακάτω.


Εικόνα 25: Αρχική εικόνα διεπαφής με τις επιλογές GALLERY, CAMERA, RECOGNIZE, DEBLUR,Ένδειξη περιστροφής, Ένδειξη αναζήτησης

Έπειτα δίνεται στον χρήστη η δυνατότητα να πραγματοποιήσει αναγνώριση ενός chip είτε από φωτογραφία που έχει είδη βγάλει σε προηγούμενη χρονική στιγμή από το κινητό του είτε βγάζοντας μια καινούρια μέσω της εφαρμογής.

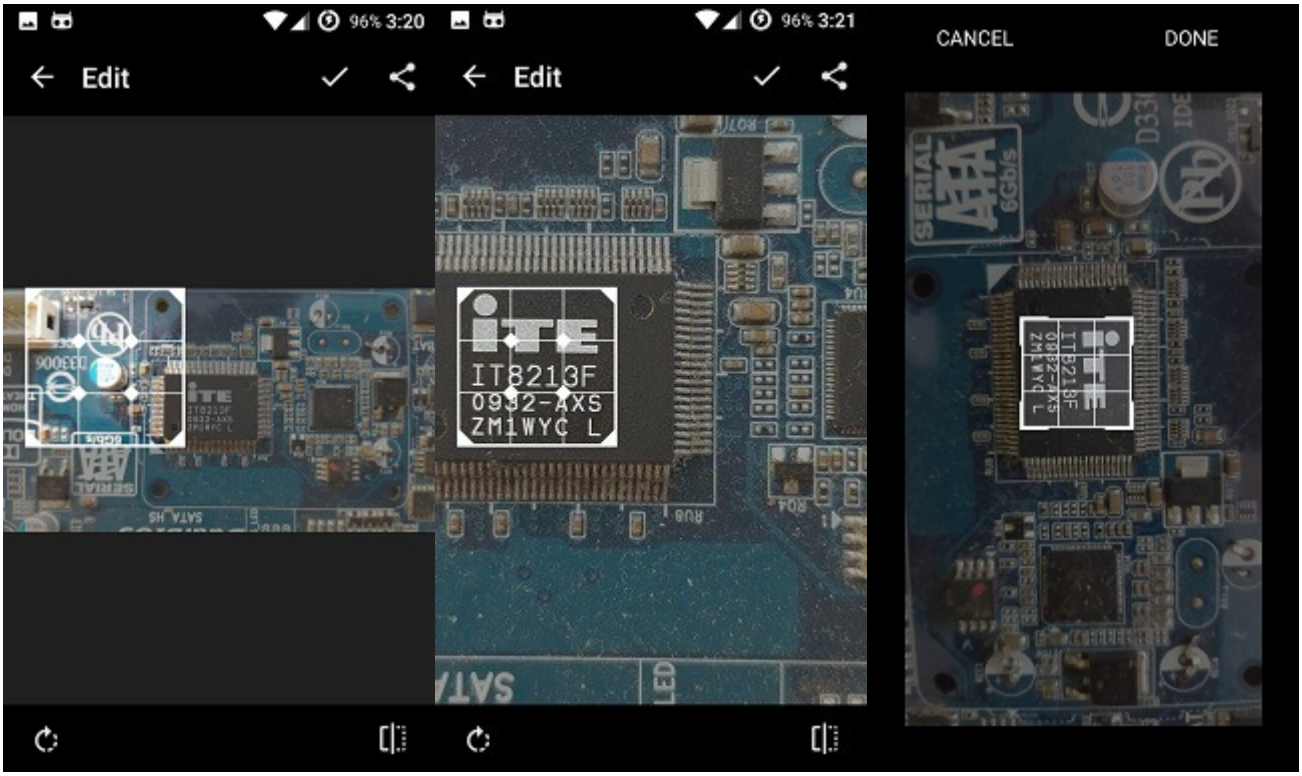



Εικόνα 26: Επιλογή εικόνας από την συλλογή



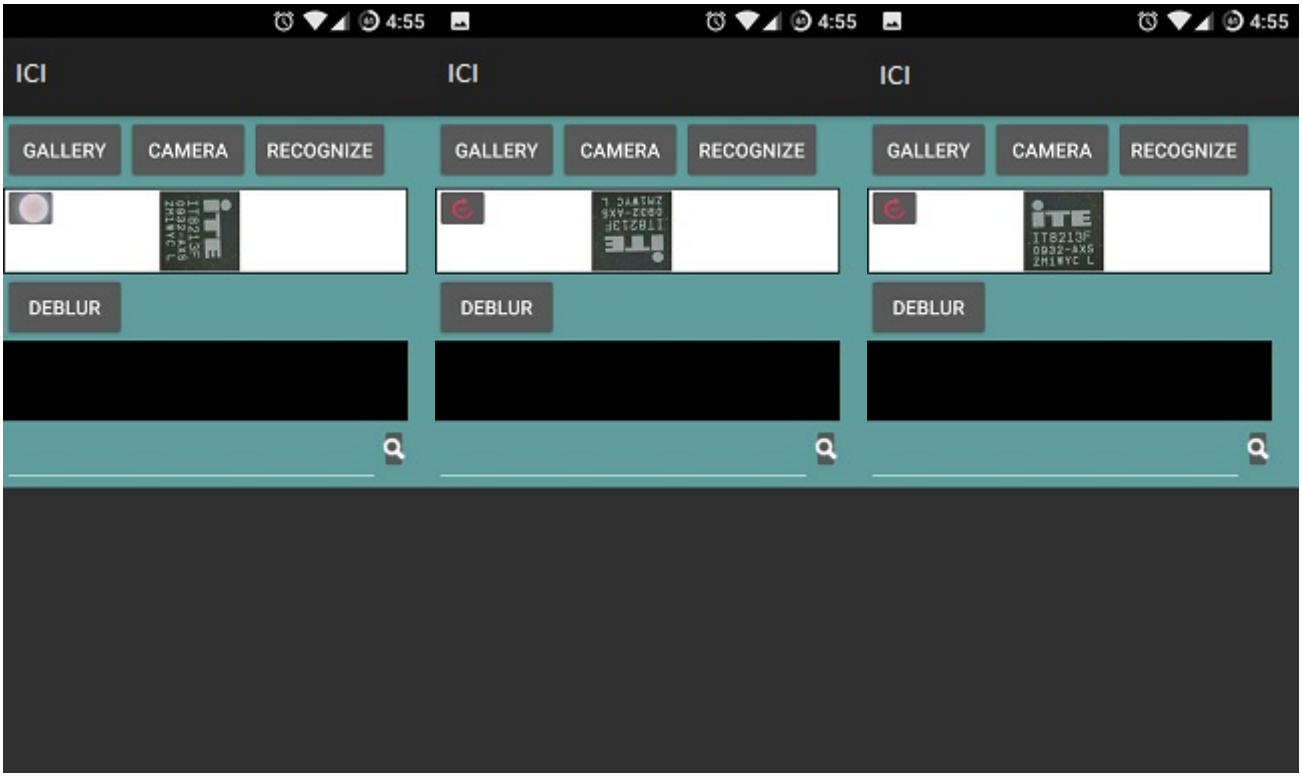

Εικόνα 27: Λήψη εικόνας από την κάμερα

Στην συνέχεια στον χρήστη δίνεται η επιλογή να απομονώσει το chip το οποίο επιθυμεί να αναγνωρίσει, περικόπτοντας το επιθυμητό τμήμα της εικόνας. Η δυνατότητα αυτή δίνεται στον χρήστη ανεξαρτήτου επιλογής εισαγωγής εικόνας (Gallery ή Camera).



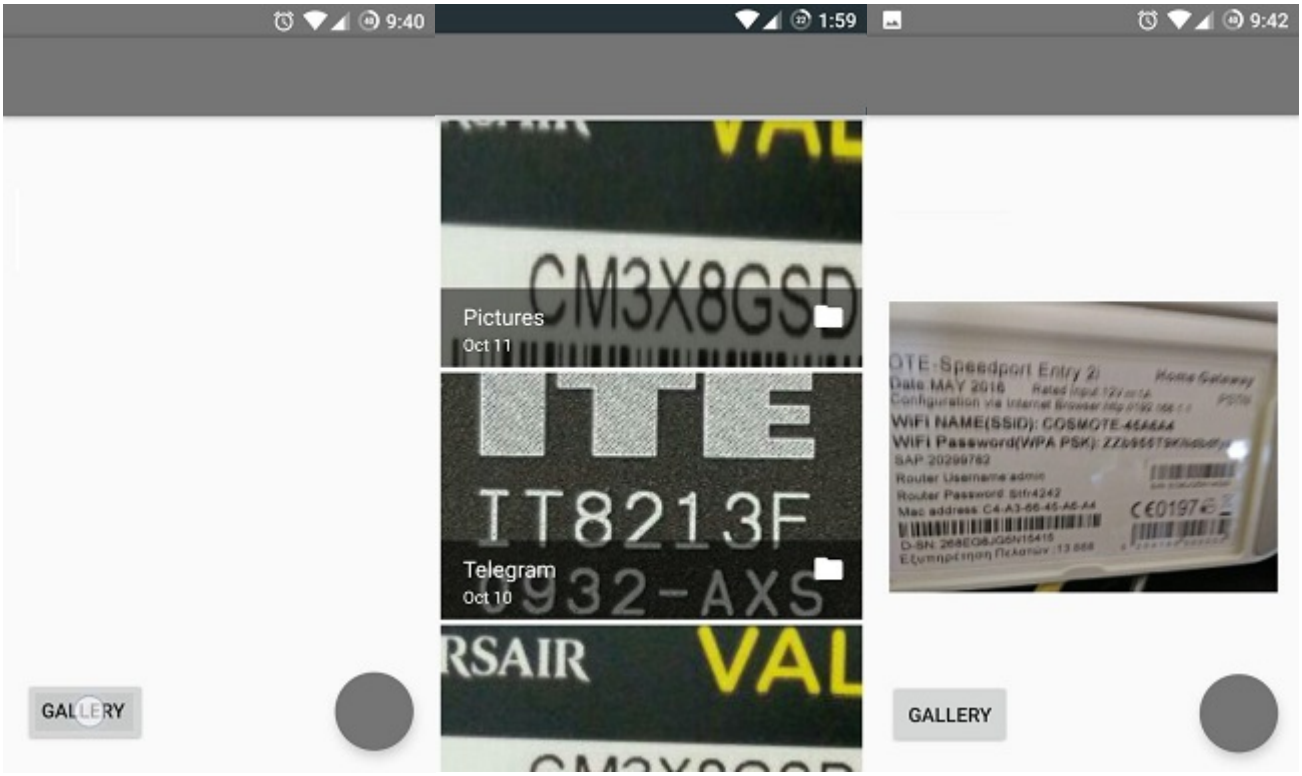

Εικόνα 28: Περικοπή και απομόνωση του chip

Υπάρχει πιθανότητα η εικόνα να έχει περιστραφεί. Σε αυτή την περίπτωση ο χρήστης έχει τη δυνατότητα να περιστρέψει την εικόνα κατά $90^0$ με το πάτημα της επιλογής ένδειξης της περιστροφής.

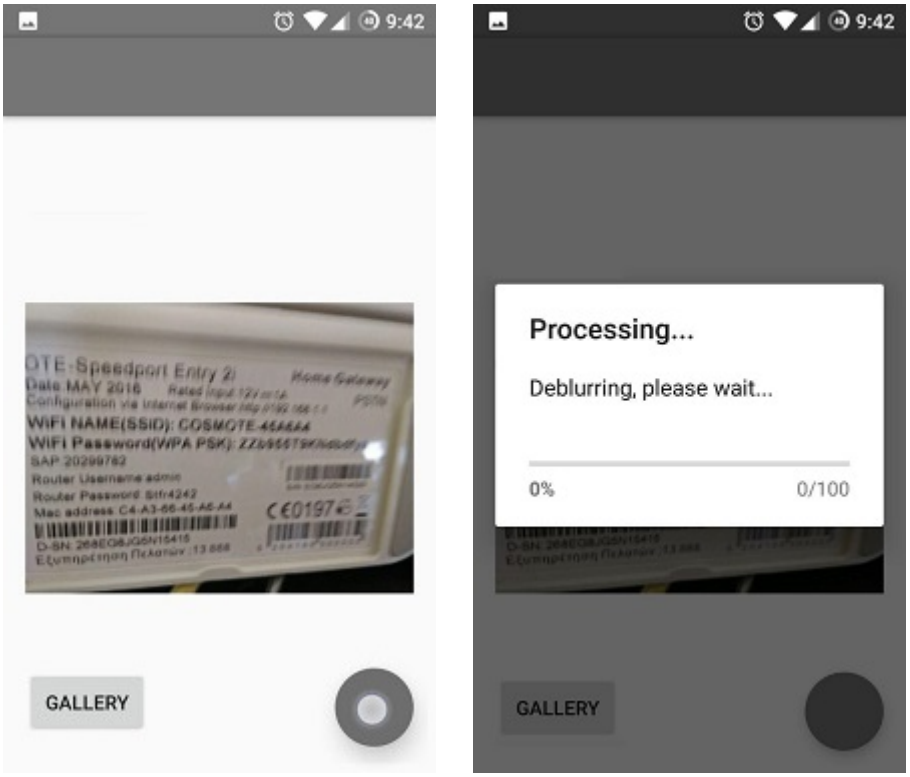



Εικόνα 29: Περιστροφή εικόνας

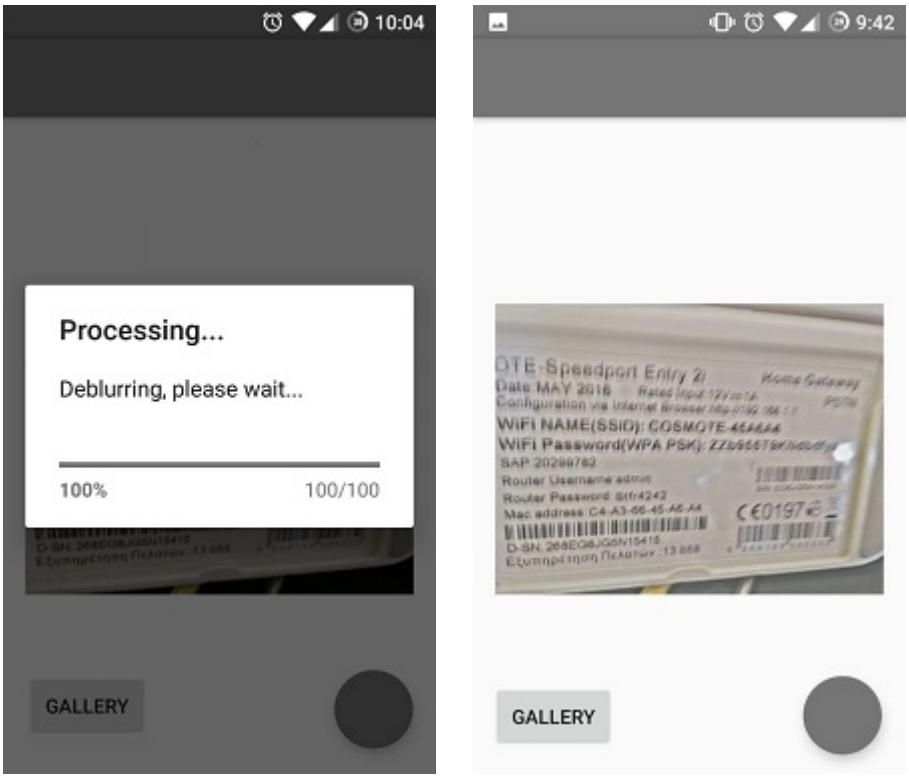
Με το πάτημα του κουμπιού “gallery”, ο χρήστης μπορεί να επιλέξει μια εικόνα από τη συλλογή εικόνων για επεξεργασία. Στη συνέχεια, ο χρήστης μπορεί να πατήσει το γκρι κουμπί, προκειμένου να αρχίσει η διαδικασία deblurring. Ένα νέο παράθυρο διαλόγου εμφανίζεται προτείνοντας στον χρήστη να περιμένει. Μετά την ολοκλήρωση της διαδικασίας εμφανίζεται η καινούρια (deblurred) εικόνα.




Εικόνα 30: Βελτίωση της ποιότητας της εικόνας επιλογή από συλλογή

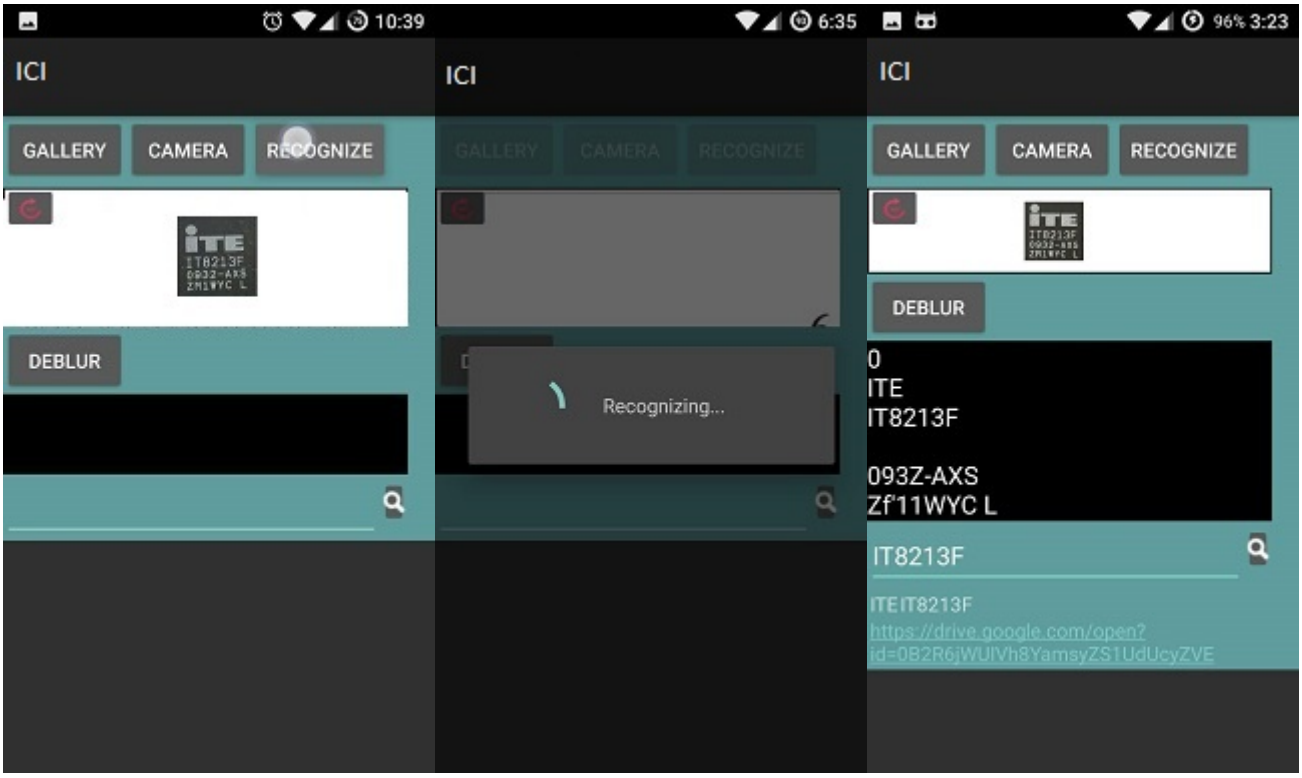


Εικόνα 31: Η επιλεγμένη εικόνα εμφανίζεται στο περιβάλλον εργασίας. Ο χρήστης μπορεί να πατήσει το γκρι κουμπί, προκειμένου να αρχίσει η διαδικασία deblurring. Το πλαίσιο διαλόγου στα δεξιά ενημερώνει τον χρήστη ότι η επεξεργασία είναι σε εξέλιξη.



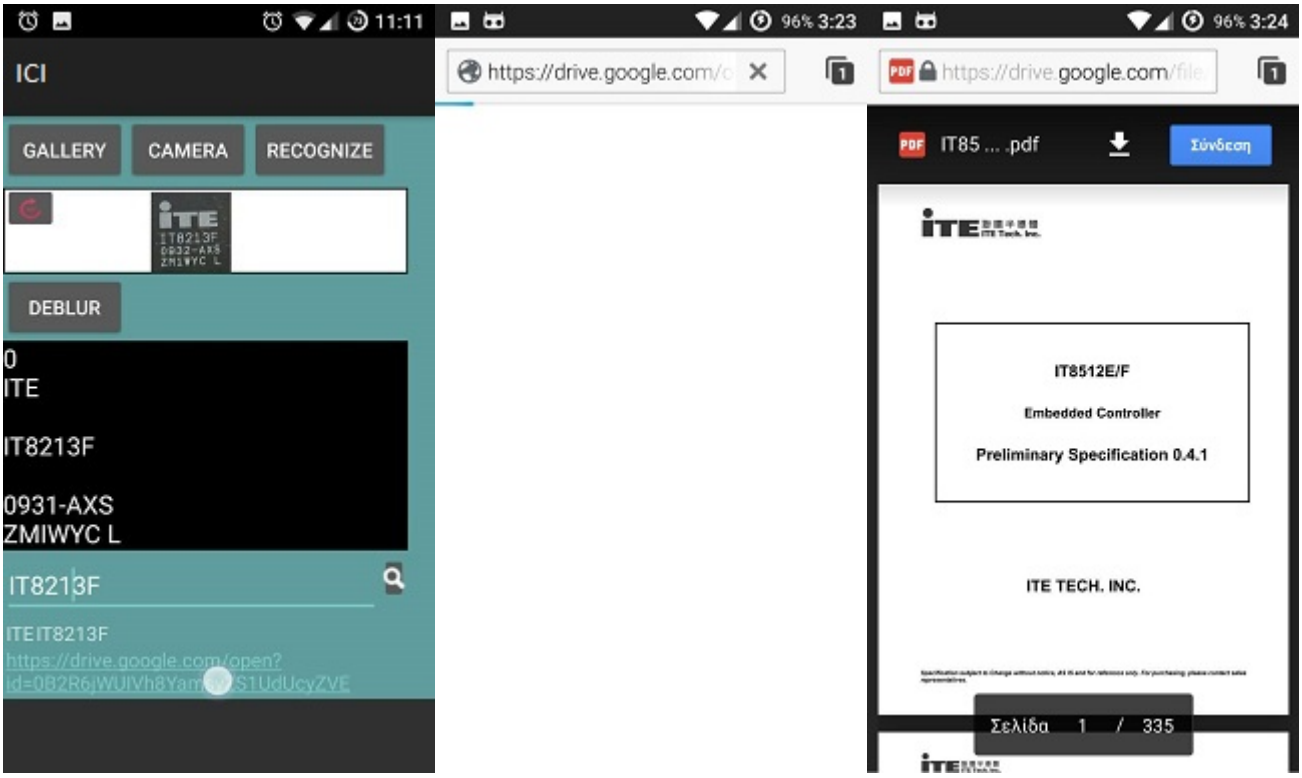

Εικόνα 32: Η επιλεγμένη εικόνα εμφανίζεται στο περιβάλλον εργασίας χωρίς θόλωση (deblured)

Ακολουθεί η επιλογή RECOGNIZE με την οποία γίνεται έναρξη της αναγνώρισης χαρακτήρων του chip. Το αποτέλεσμα εμφανίζεται στο μαύρο πλαίσιο.




Εικόνα 33: Αναγνώριση εικόνας και εμφάνιση αποτελεσμάτων

Τέλος ο χρήστης επιλέγοντας την ένδειξη της αναζήτησης και πατώντας στον εμφανιζόμενο σύνδεσμο έχει πρόσβαση στο datasheet του chip όπως φαίνεται παρακάτω. Έχοντας την δυνατότητα και της λήψης του.

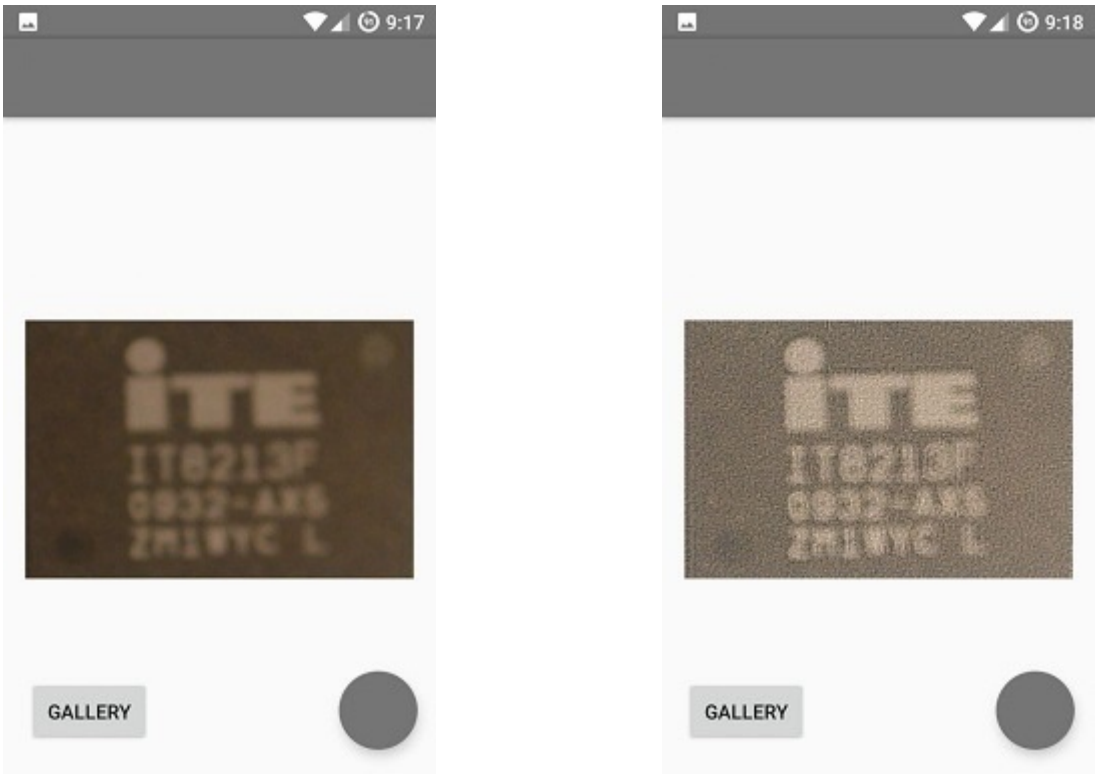



Εικόνα 34: Πρόσβαση στο datasheet του chip

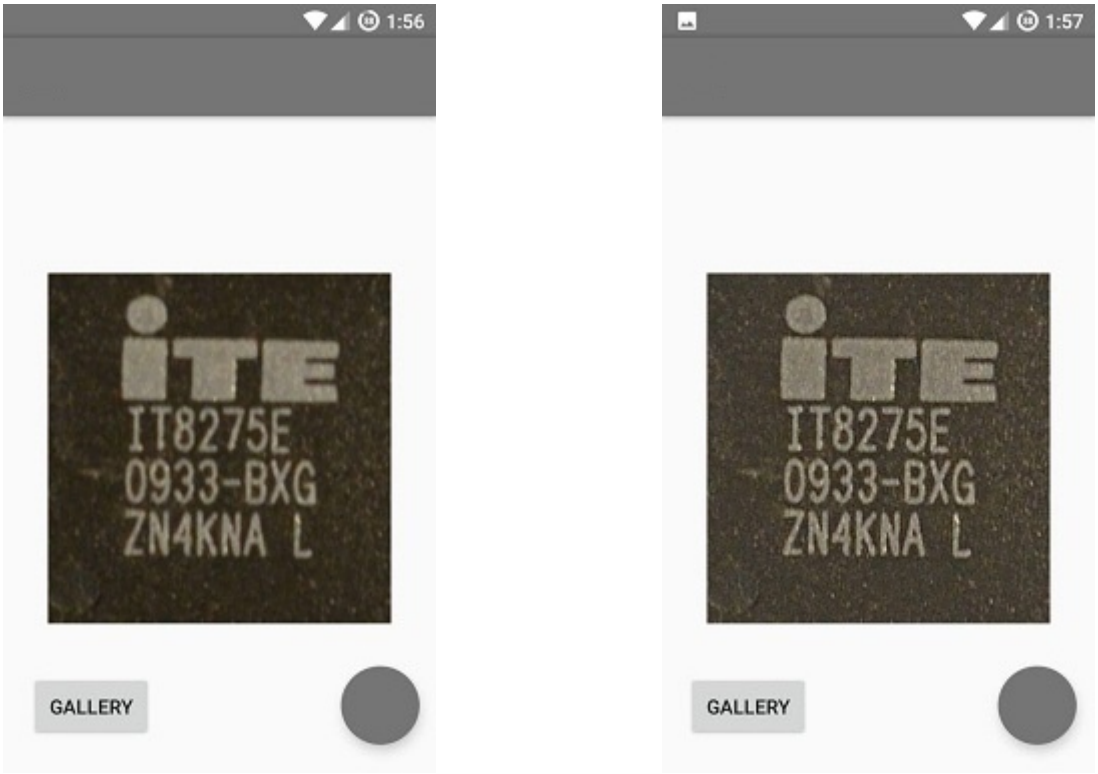
## 4.2 Δοκιμή της εφαρμογής

### Αποτελέσματα της διαδικασίας απαλοιφής θολώματος

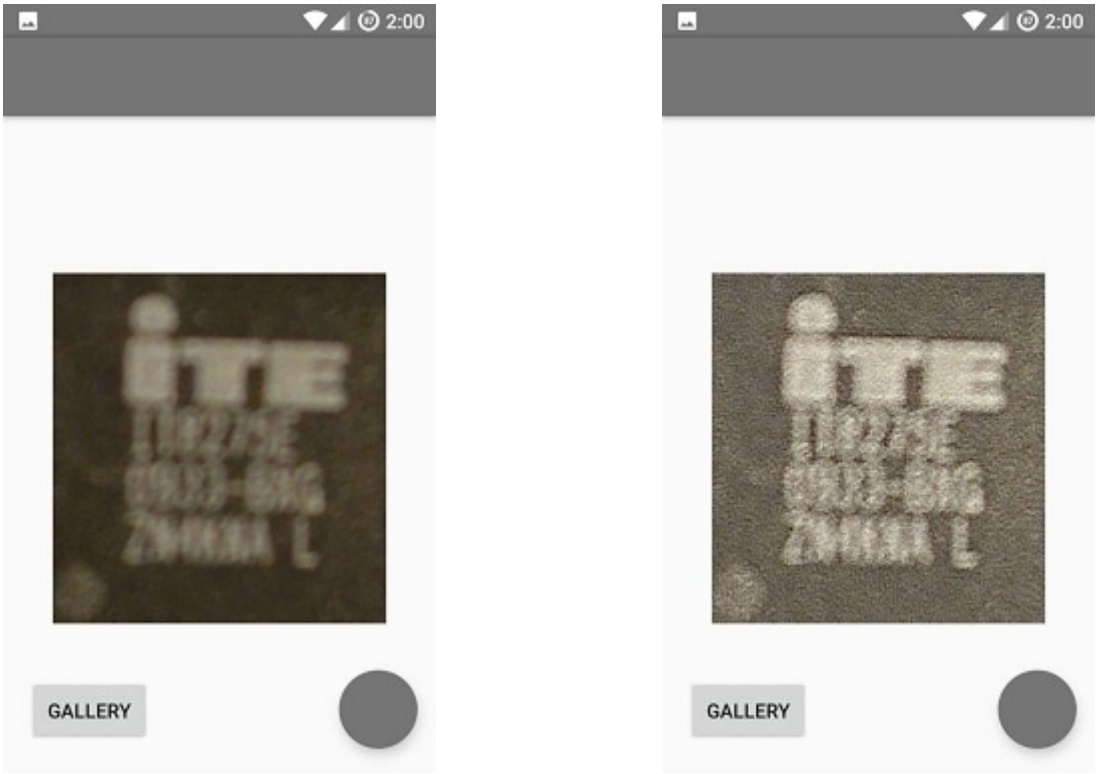
Η ενότητα αυτή παρουσιάζει τα αποτελέσματα της διαδικασίας deblurring όπως εμφανίζονται στην εφαρμογή Android. Τα ακόλουθα στοιχεία περιλαμβάνουν το αποτέλεσμα για κάθε μία από τις εικόνες δοκιμής. Στην αριστερή στήλη, παρουσιάζουμε την αρχική θολή εικόνα όπως είχε επιλεγεί από το χρήστη από τη συλλογή του τηλεφώνου ή του tablet του. Αφού επιλέξει την εικόνα ο χρήστης πατάει το κουμπί για να ξεκινήσει η διαδικασία απαλοιφής θολώματος (deblurring). Το αποτέλεσμα παρουσιάζονται στην δεξιά στήλη.



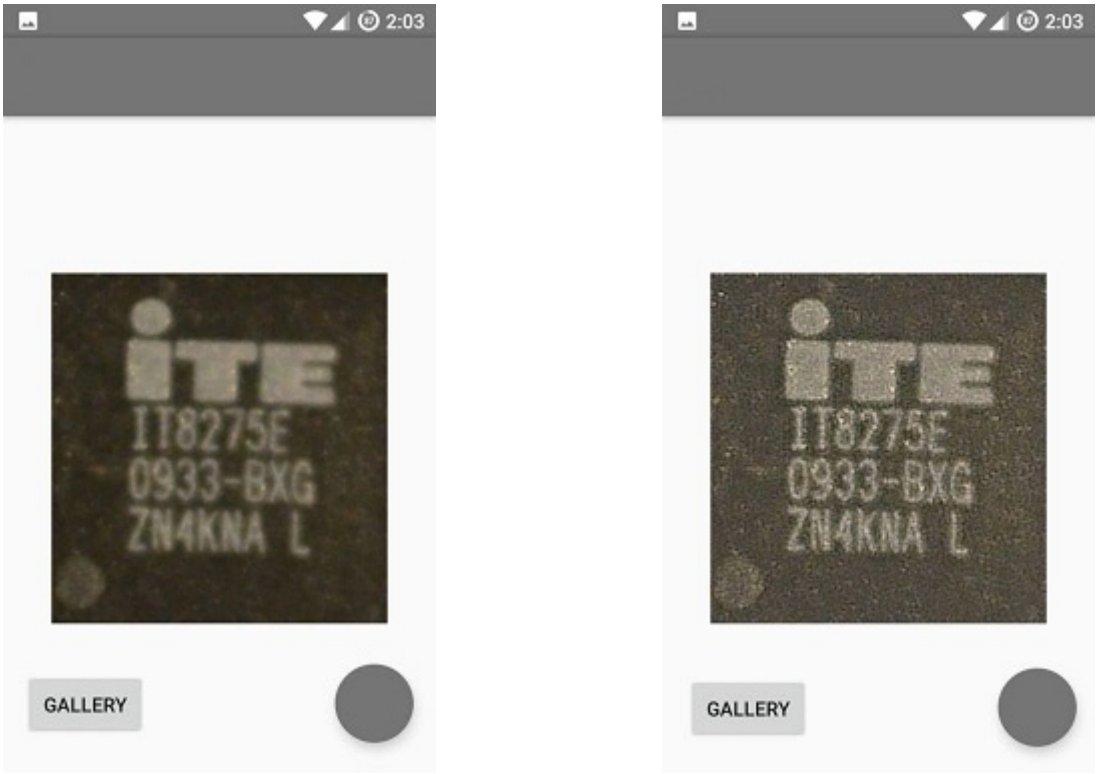

Εικόνα 35: Chip 2x1.5 εκατοστών παράδειγμα δοκιμαστικής εικόνας. Η εικόνα που παράγεται από την Android εφαρμογή φαίνεται δεξιά όπως αφού ο χρήστης προηγουμένως την έχει επιλέξει από την Android βιβλιοθήκη της συσκευής και έχει πατήσει το γκρι κουμπί.



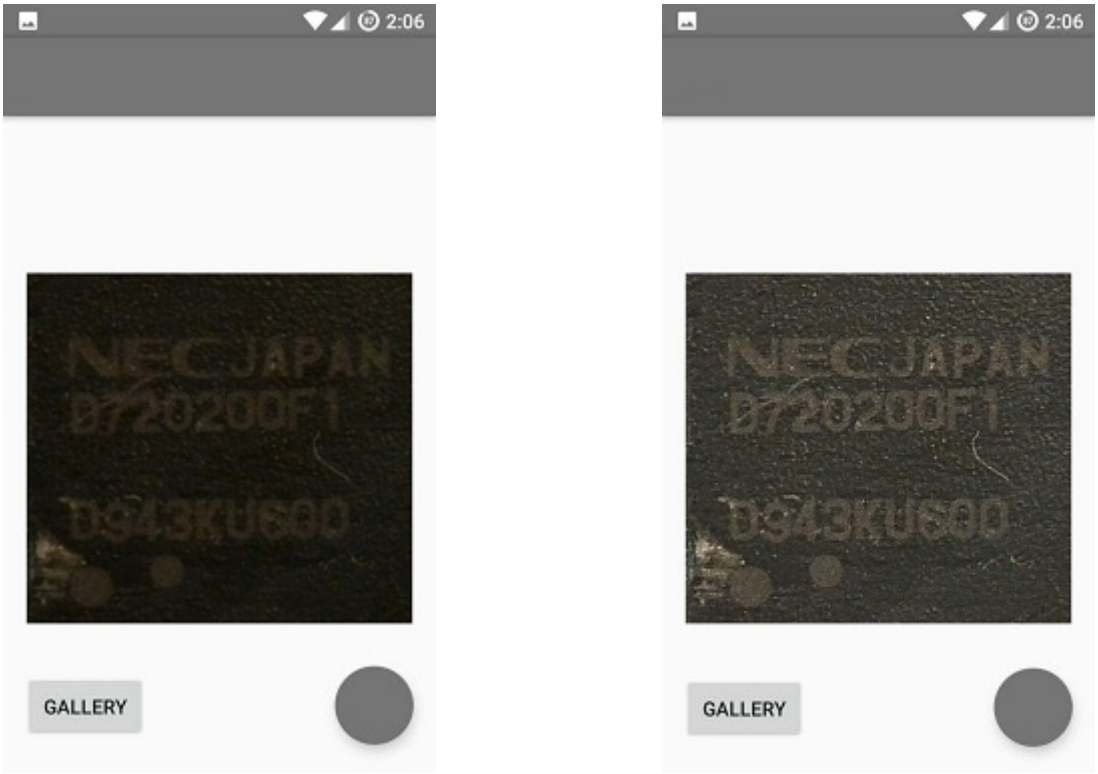

Εικόνα 36: Chip 0.5x0.5 εκατοστών παράδειγμα δοκιμαστικής εικόνας. Η εικόνα που παράγεται από την Android εφαρμογή φαίνεται δεξιά όπως αφού ο χρήστης προηγουμένως την έχει επιλέξει από την Android βιβλιοθήκη της συσκευής και έχει πατήσει το γκρι κουμπί.




Εικόνα 37: Chip 0.5x0.5 εκατοστών παράδειγμα δοκιμαστικής εικόνας. Η εικόνα που παράγεται από την Android εφαρμογή φαίνεται δεξιά όπως αφού ο χρήστης προηγουμένως την έχει επιλέξει από την Android βιβλιοθήκη της συσκευής και έχει πατήσει το γκρι κουμπί.




Εικόνα 38: Chip 0.5x0.5 εκατοστών παράδειγμα δοκιμαστικής εικόνας. Η εικόνα που παράγεται από την Android εφαρμογή φαίνεται δεξιά όπως αφού ο χρήστης προηγουμένως την έχει επιλέξει από την Android βιβλιοθήκη της συσκευής και έχει πατήσει το γκρι κουμπί.

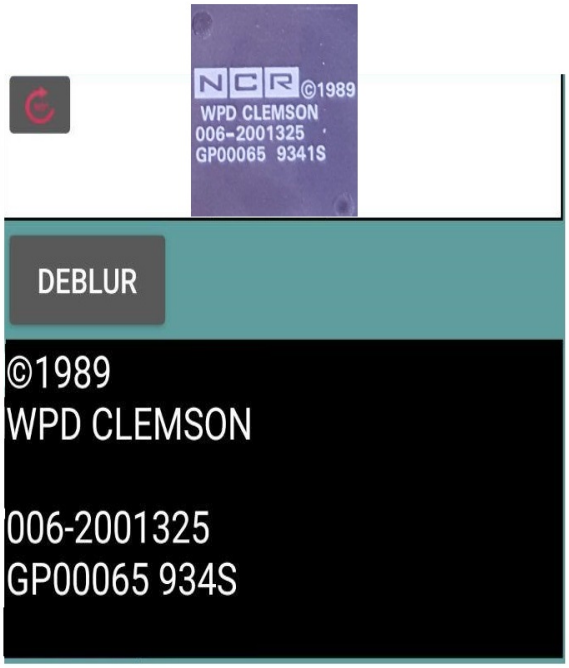



Εικόνα 39: Chip 1x1 εκατοστών παράδειγμα δοκιμαστικής εικόνας. Η εικόνα που παράγεται από την Android εφαρμογή φαίνεται δεξιά όπως αφού ο χρήστης προηγουμένως την έχει επιλέξει από την Android βιβλιοθήκη της συσκευής και έχει πατήσει το γκρι κουμπί.

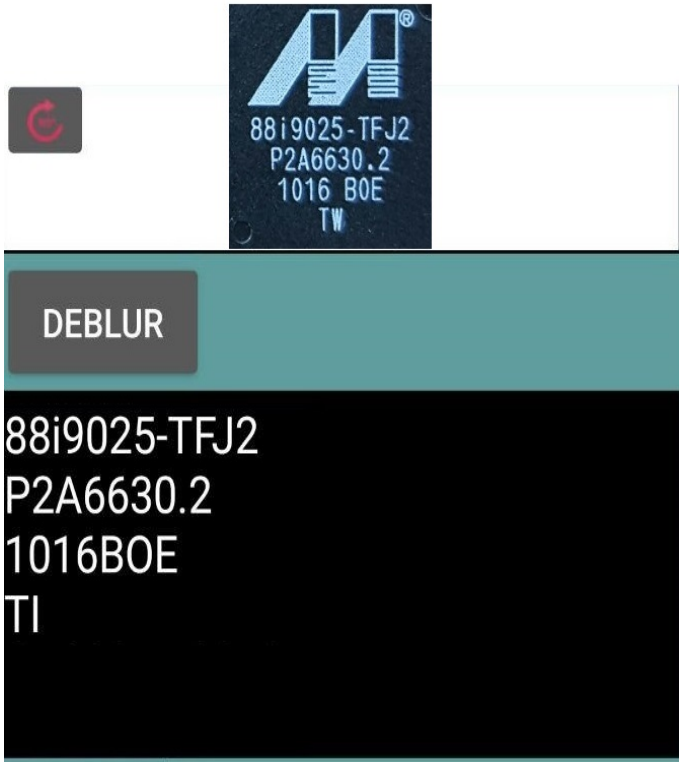
Όπως έγινε φανερό από τις παραγόμενες εικόνες είναι πράγματι πιο έντονες και πιο σαφείς από την αρχική εικόνα εισόδου. Ωστόσο, ακόμα δεν είναι τέλεια. Αυτό είναι το αποτέλεσμα του συμβιβασμού, μεταξύ της απόδοσης και της ποιότητας που έχει ακόμα πιο ειδικούς περιορισμούς όταν γίνεται αναφορά σε κινητές συσκευές.

### Αποτελέσματα της διαδικασίας οπτικής αναγνώρισης

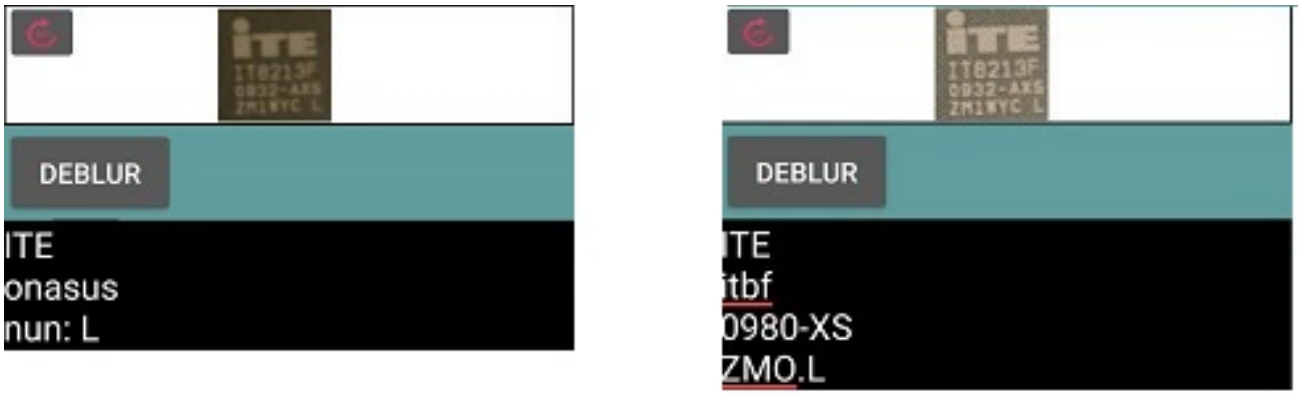
Η ενότητα αυτή παρουσιάζει τα αποτελέσματα της διαδικασίας οπτικής αναγνώρισης της εικόνας όπως εμφανίζονται στην εφαρμογή Android. Τα ακόλουθα στοιχεία περιλαμβάνουν το αποτέλεσμα για κάθε μία από τις εικόνες δοκιμής. Στην περίπτωση όπου έχει προηγηθεί απαλοιφή θολώματος στην αριστερή στήλη παρουσιάζεται η αρχική θολή εικόνα όπως είχε επιλεγεί από το χρήστη από τη συλλογή του τηλεφώνου ή του tablet του. Ενώ στην δεξιά στήλη παρουσιάζονται τα αποτέλεσμα μετά την απαλοιφή θολώματος.




Εικόνα 41: Chip 1x1 παράδειγμα δοκιμαστικής εικόνας. Το αποτέλεσμα της αναγνώρισης που παράγεται από την Android εφαρμογή φαίνεται στο μαύρο πλαίσιο αφού ο χρήστης προηγουμένως την έχει επιλέξει από την Android βιβλιοθήκη της συσκευής και έχει πατήσει το recognize κουμπί.

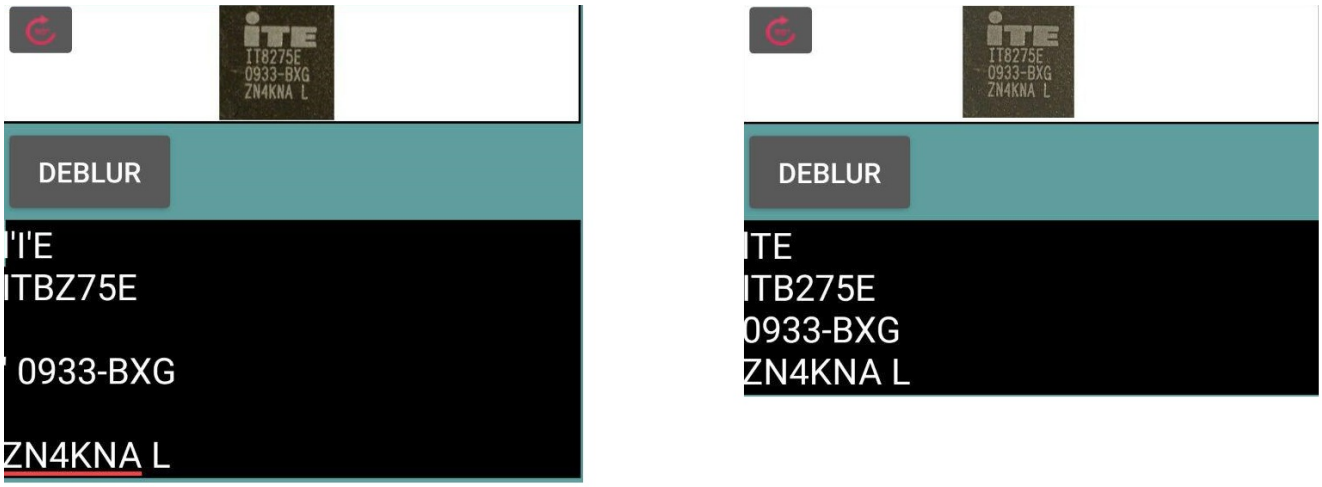



Εικόνα 40: Chip 1x1 παράδειγμα δοκιμαστικής εικόνας. Το αποτέλεσμα της αναγνώρισης που παράγεται από την Android εφαρμογή φαίνεται στο μαύρο πλαίσιο αφού ο χρήστης προηγουμένως την έχει επιλέξει από την Android βιβλιοθήκη της συσκευής και έχει πατήσει το recognize κουμπί.



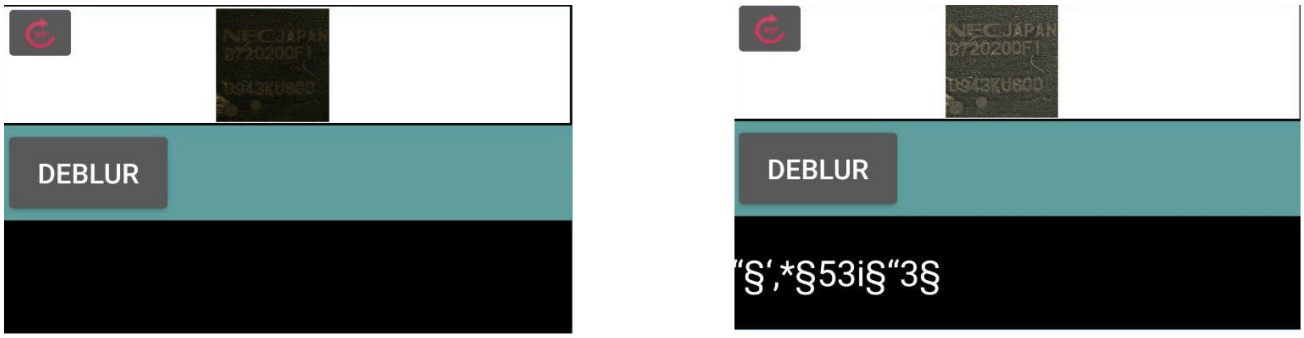

Εικόνα 41: Chip 2x1.5 εκατοστών παράδειγμα δοκιμαστικής εικόνας. Το αποτέλεσμα της αναγνώρισης χωρίς αποκατάστασης της εικόνας εμφανίζεται στο αριστερό μέρος ενώ στο δεξί μέρος παρουσιάζεται το αποτέλεσμα της αναγνώρισης μετά την αποκατάσταση της εικόνας.


Εικόνα 42: Chip 2x1.5 εκατοστών παράδειγμα δοκιμαστικής εικόνας. Το αποτέλεσμα της αναγνώρισης χωρίς αποκατάστασης της εικόνας εμφανίζεται στο αριστερό μέρος ενώ στο δεξί μέρος παρουσιάζεται το αποτέλεσμα της αναγνώρισης μετά την αποκατάσταση της εικόνας.




Εικόνα 43: Chip 0.5x0.5 εκατοστών παράδειγμα δοκιμαστικής εικόνας. Το αποτέλεσμα της αναγνώρισης χωρίς αποκατάστασης της εικόνας εμφανίζεται στο αριστερό μέρος ενώ στο δεξί μέρος παρουσιάζεται το αποτέλεσμα της αναγνώρισης μετά την αποκατάσταση της εικόνας.


Εικόνα 44: Chip 0.5x0.5 εκατοστών παράδειγμα δοκιμαστικής εικόνας. Το αποτέλεσμα της αναγνώρισης χωρίς αποκατάστασης της εικόνας εμφανίζεται στο αριστερό μέρος ενώ στο δεξί μέρος παρουσιάζεται το αποτέλεσμα της αναγνώρισης μετά την αποκατάσταση της εικόνας.




Εικόνα 45: Chip 1x1 εκατοστών παράδειγμα δοκιμαστικής εικόνας. Το αποτέλεσμα της αναγνώρισης χωρίς αποκατάστασης της εικόνας εμφανίζεται στο αριστερό μέρος ενώ στο δεξί μέρος παρουσιάζεται το αποτέλεσμα της αναγνώρισης μετά την αποκατάσταση της εικόνας.

## 4.2.2 Αξιοποίηση Πόρων

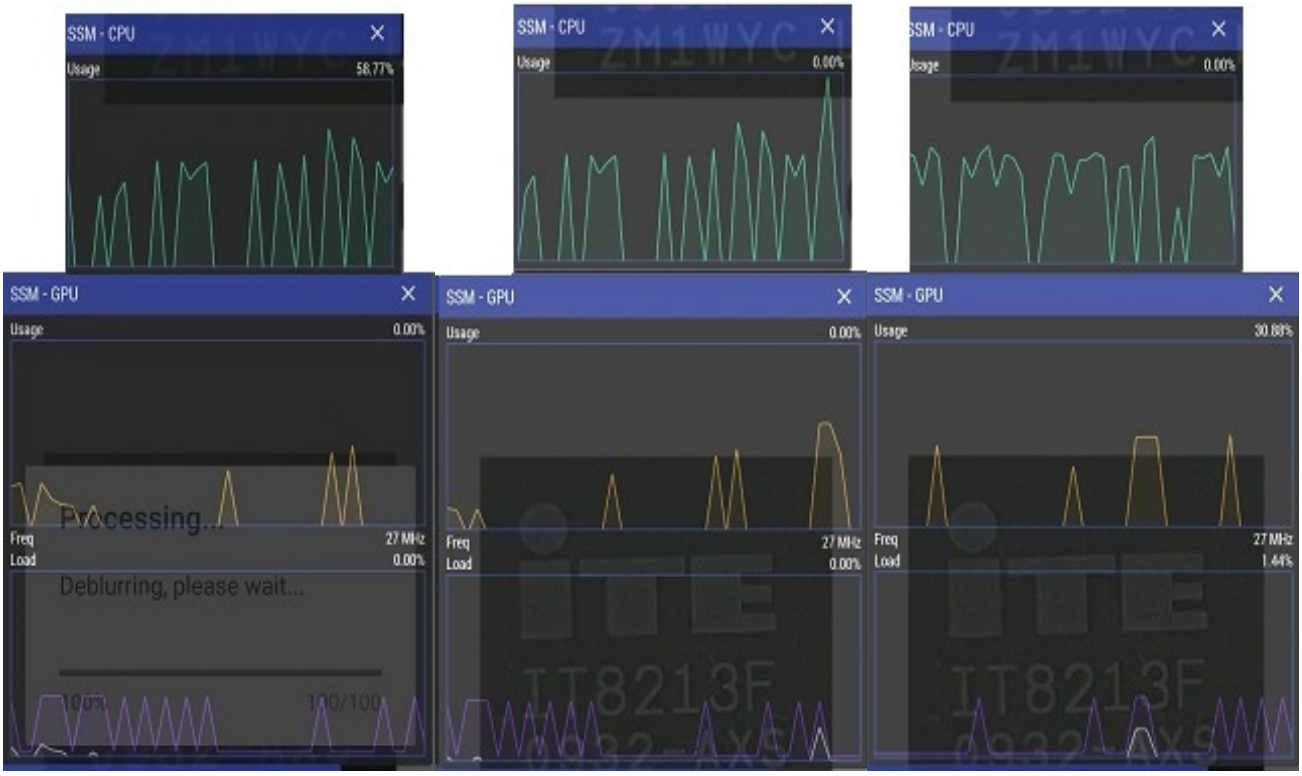
Για να εξαλειφθεί η ανάγκη ενσωμάτωσης plugin στο Eclipse καθώς ούτε στο Android Studio τέτοιο εργαλείο δεν υπάρχει τουλάχιστον σε σταθερή έκδοση του λογισμικού. Για να παρουσιαστούν τα αποτελέσματα απόδοση και χρήσης των υπολογιστικών πόρων της εφαρμογής χρησιμοποιήθηκε η εφαρμογή Simple System Monitor. Αυτή η εφαρμογή επιτρέπει τη συλλογή «σε πραγματικό χρόνο» δεδομένων χρήσης της CPU, GPU, RAM κ.α.

Τα γραφήματα παρακάτω αντιπροσωπεύουν την χρήση της CPU και GPU αντίστοιχα:


Εικόνα 46: Χρήση CPU και GPU κατά την διαδικασία OCR

Λεπτομερώς:
Το σύστημα που χρησιμοποιήθηκε είναι ένα OnePlus One

- Το σύστημα σε πλήρη ηρεμία σε σχέση με την εφαρμογή μας καταναλώνει 54% έως 61% της CPU και περιοδικά 30% έως 40% της GPU όπως φαίνεται στην πρώτη στα αριστερά εικόνα
- Η πρώτη κορύφωση σε σχέση με την εφαρμογή μας και το κομμάτι της αναγνώρισης ανιχνεύεται κατά την έναρξη της αναγνώρισης όπου τα επίπεδα της CPU κυμαίνονται μεταξύ 70% και 90% και 40% και 60% για την GPU.
- Στη συνέχεια, ακολουθεί μια πτώση της απόδοσης της CPU στο 24% σε αντίθεση με τα επίπεδα απόδοσης της GPU τα οποία παραμένουν σταθερά στο 40%.
- Στη συνέχεια και μόνο για λίγο, ακολουθεί μια πτώση της απόδοσης της GPU στο 0% σε αντίθεση με τα επίπεδα απόδοσης της CPU τα οποία αγγίζουν το 50%.
- Ακολουθεί πτώση της απόδοσης της CPU στο 35% και η άνοδος της απόδοσης της GPU στο 40% για τελευταία φορά αφού η διαδικασία τελειώνει με την χρήση της να πέφτει στο 0% και την χρήση της CPU αγγίζει για τελευταία φορά στο 35% για να πέσει και αυτή αμέσως μετά στο 0%.
- Για την αναγνώριση της εικόνας το σύστημα χρειάζεται περίπου 1s του χρόνου.




Εικόνα 47: Χρήση CPU και GPU κατά την διαδικασία deblur

Λεπτομερώς:
Το σύστημα που χρησιμοποιήθηκε είναι ένα OnePlus One

- Το σύστημα σε πλήρη ηρεμία σε σχέση με την εφαρμογή μας καταναλώνει 54% έως 68% της CPU και περιοδικά 35% έως 45% της GPU όπως φαίνεται στην πρώτη στα αριστερά εικόνα
- Η πρώτη κορύφωση σε σχέση με την εφαρμογή μας και το κομμάτι της απαλοιφής θολώματος ανιχνεύεται κατά την έναρξη της διαδικασίας απαλοιφής θολώματος όπου τα επίπεδα της CPU κορυφώνονται στο 97% και μεταξύ 60% και 40% για την GPU.
- Στη συνέχεια, ακολουθεί ηρεμία σε σχέση με την εφαρμογή μας καταναλώνοντας 54% έως 68% της CPU και ανά διαστήματα 35% έως 50% της GPU όπως φαίνεται στην τρίτη ξεκινώντας από τα αριστερά εικόνα
- Για την απαλοιφή θολώματος της εικόνας το σύστημα χρειάζεται περίπου 9s του χρόνου.

## 5.1 Συμπεράσματα

Όπως έχει γίνει πλέον φανερό το θέμα της αναγνώρισης ολοκληρωμένων στοιχείων είναι ένα έργο δύσκολο. Αφενός γιατί δεν υπάρχει μια ενιαία βάση δεδομένων ολοκληρωμένων στοιχείων και αφετέρου γιατί η αναγνώριση κειμένου από πραγματικές φωτογραφίες είναι ένα δύσκολο πρόβλημα που οφείλεται στους διαφορετικούς περιβαλλοντικούς παράγοντες.
Ακόμη και η καλύτερη διαθέσιμη μηχανή ανοιχτού κώδικα OCR έχει περιορισμούς και η βελτίωση της είναι απαραίτητη για την αποτελεσματική λειτουργία της εφαρμογής.
Μέρος αυτής της διπλωματικής ήταν η διερεύνηση μεθόδων για την ανάκτηση της λανθάνουσας εικόνας όπως εκείνη θα είχε παραχθεί αν η φωτογραφική μηχανή ήταν σε τέλεια ακινησία. Κριτήρια επιλογής ήταν η μέθοδος αυτή να είναι γρήγορη, να χρησιμοποιεί λίγους υπολογιστικούς πόρους και να έχει ικανοποιητικά αποτελέσματα. Υπήρχαν βέβαια και ορισμένα μειονεκτήματα. Η ανάκτηση της λανθάνουσας εικόνας δεν ήταν πάντα τέλεια. Θα μπορούσαν να χρησιμοποιηθούν άλλες πιο υπολογιστικά ακριβείς μέθοδοι με συνέπεια τα αποτέλεσμα να είναι καλύτερα. Τέτοιες μέθοδοι αποτελούν πρόκληση όσο αναφορά την υλοποίηση τους καθώς απαιτούν μεγάλο αριθμό υπολογιστικών πόρων, πόροι που πιθανόν έναν ενσωματωμένο σύστημα να μην έχει.
Αξιοποιώντας πλήρως τα προηγμένα τεχνικά χαρακτηριστικά των έξυπνων κινητών τηλεφώνων το αποτέλεσμα είναι μια εξαιρετικά λειτουργική εφαρμογή χρήσιμη για έναν ηλεκτρονικό.

## 5.2 Μελλοντικές Εργασίες

Ενδιαφέρον πολύ θα ήταν αν κάποιος ερευνούσε το “συμβιβασμό” μεταξύ της ταχύτητας και της ποιότητας, δεδομένου ότι κάθε μέθοδος παράγει διαφορετικά αποτελέσματα για διαφορετικούς χρόνους επεξεργασίας.
Όσον αφορά το επίπεδο εφαρμογής μια άλλη πολύ ενδιαφέρουσα εφαρμογή θα ήταν να επιτευχθεί μια διαδικασία αναγνώρισης και deblurring σε πραγματικό χρόνο πράγμα που σημαίνει ότι ο χρόνος της διαδικασίας θα πρέπει να είναι κάτω του ενός δευτερολέπτου.
Τέλος, είναι σημαντικό να συναχθεί το συμπέρασμα ότι η ενσωμάτωση μεθόδων και τεχνικών οπτικής αναγνώρισης και επεξεργασίας εικόνας σε ενσωματωμένα συστήματα για την δημιουργία εφαρμογών που αλληλεπιδρούν με την πραγματική ζωή είναι ένα δύσκολο αλλά ελπιδοφόρο έργο.

# Αναφορές


[1] Smartphones Market Share

http://www.iphonehellas.gr/44296/ios-vs-android-by-the-numbers/

[2] Android Developer Tools (ADT)

http://marketplace.eclipse.org/content/android-development-tools-eclipse

[3] Σχετική μεγέθη για δυνατότητα σχεδίασης bitmap που υποστηρίζουν κάθε πυκνότητα.

https://developer.android.com/images/screens_support/screens-densities.png

[4] Ποσοστό των συσκευών που τρέχουν μια δεδομένη έκδοση του Android

https://en.wikipedia.org/wiki/Android_version_history

[5] Cheriet M. Character Recognition System: A Guide for Students and

Practitioners. New Jersey, United States of American: John Wiley & Sons, Inc 2007.

[6] Tesseract OCR [online]. Google Developers; March 2013.

https://code.google.com/p/tesseract-ocr/.

[7] Roberts D. Automated License Plate Recognition Systems: Policy and Operational

guidance for Law Enforcement [online]; 2012.

https://www.ncjrs.gov/pdffiles1/nij/grants/239604.pdf

[8] ALPR

http://www.iacp.org/ALPR-About

[9] ABBYY SOLUTIONS FOR BUSINESS PROCESSES

https://www.abbyy.com/en-apac/?DN=691656f9-d628-4f90-b330-da6e42d3d4fe

[10] ASPRISE OCR

http://asprise.com/home/

[11] Online OCR Services

http://ocrapiservice.com/

[12] Tesseract UI

https://upload.wikimedia.org/wikipedia/commons/3/3c/Tesseract_on_ocrfeeder.png

[13] Sandip Rakshit, Amitava Kundu, Mrinmoy Maity, Subhajit Mandal, Satwika Sarkar, Subhadip Basu, Recognition of handwritten Roman Numerals using Tesseract open source OCR engine.

[14] Sandip Rakshit and Subhadip Basu, Recognition of Handwritten Roman Script Using Tesseract Open source OCR engine

[15] Sandip Rakshit and Subhadip Basu, Development of a multi-user handwriting recognition system using Tesseract open source OCR engine

[16] Omnifont engine

http://www.webopedia.com/TERM/O/omnifont_recognition.html

[17] Επισκόπηση των λειτουργιών της OpenCV

http://ppt-online.org/48830

[18] javaCV

https://code.google.com/archive/p/javacv/

[19] Dalvik vm internals

D. Bornstein, “Dalvik vm internals,” in Google I/O Developer Conference, 2008, σελ. 17–30.

[20] F. Maker and Y. Chan, “A survey on android vs. linux,” Univ. Calif., σελ.. 1–10, 2009.

[21] Dominique A. Heger, Quantifying IT Stability, 2nd Edition : Grid, Cloud, Cluster, and SMP Systems.2011.

[22] P. Brady, “Android anatomy and physiology,” in Google IO developer conference, 2008, σελ. 119.

[23] Liang, “System Integration for the Android Operating System.” National Taipei University, 2010.

[24] S. Cho and S. Lee, “Fast motion deblurring,” ACM Trans. Graph. σελ.1–145, 2009.

[26] S. Osher and L. I. Rudin, “Feature-Oriented Image Enhancement Using Shock Filters,” SIAM J. Numer. Anal., σελ. 919–940, Αυγ 1990.

[27] A. N. Ti, “Solution of incorrectly formulated problems and the regularization method,” Sov. Math. Dokl, σελ. 1035–1038, Nov. 1963.

[28] A. Levin and W. T. Freeman, “Deconvolution using natural image priors,” Artif. Intell., σελ. 0–2, 2007.

[29] Q. Shan, J. Jia, and A. Agarwala, “High-quality motion deblurring from a single image,” ACM Trans. Graph., σελ. 1, 2008.

[30] J. Chen, L. Yuan, C. K. Tang, and L. Quan, “Robust dual motion deblurring,” in 26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2008, σελ. 1–8.

[31] Smith R. An Overview of the Tesseract OCR Engine [online]

http://static.googleusercontent.com/media/research.google.com/zhCN//pubs/archive/33418.pdf

[32] Smith R. Tesseract OCR Engine 2007.

http://tesseract-ocr.googlecode.com/files/TesseractOSCON.pdf

[33] Smith R. An Overview of the Tesseract OCR Engine

http://static.googleusercontent.com/media/research.google.com/zh-N//pubs/archive/33418.pdf

[34] https://languagetime.org/2013/05/09/google-translate-for-android-can-now-interpret-16-additional-languages-by-camera-adds-phrasebook-support/

[35] http://tfmainsights.com/mobile-eating-world-overview-mobile-trends-2013/

[36] https://www.oneclickroot.com/android-updates/android-5-0-will-switch-from-art-to-dalvik/

[37] http://edureka.in/blog/wp-content/uploads/2013/01/Android-Stack.jpg