ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΣΥΣΤΗΜΑΤΩΝ ΜΕΤΑΔΟΣΗΣ ΠΛΗΡΟΦΟΡΙΑΣ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΛΙΚΟΥ

Σχεδίαση και ανάπτυξη ενοποιημένης εφαρμογής κινητών συσκευών για τον πολιτισμό με χρήση τεχνικών επαυξημένης πραγματικότητας

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Των

Καραμπλιά Θεοφάνη,

Μπαλάσκα Γρηγορίου

Επιβλέπων: Ιάκωβος Βενιέρης,
Καθηγητής Ε.Μ.Π.

Αθήνα, Φεβρουάριος 2020

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΣΥΣΤΗΜΑΤΩΝ ΜΕΤΑΔΟΣΗΣ ΠΛΗΡΟΦΟΡΙΑΣ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΛΙΚΟΥ

Σχεδίαση και ανάπτυξη ενοποιημένης εφαρμογής κινητών συσκευών για τον πολιτισμό με χρήση τεχνικών επαυξημένης πραγματικότητας

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Των

Καραμπλιά Θεοφάνη,

Μπαλάσκα Γρηγορίου

Επιβλέπων: Ιάκωβος Βενιέρης,
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 10/02/2020

.....	.....	.....
Ιάκωβος Βενιέρης,	Δ.-Θ. Κακλαμάνη,	Γ. Ματσόπουλος,
Καθηγητής Ε.Μ.Π.	Καθηγήτρια Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Φεβρουάριος 2020

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΣΥΣΤΗΜΑΤΩΝ ΜΕΤΑΔΟΣΗΣ ΠΛΗΡΟΦΟΡΙΑΣ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΛΙΚΟΥ

.....

Καραμπλιάς Θεοφάνης

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

.....

Μπαλάσκας Γρηγόριος

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Το αντικείμενο της διπλωματικής εργασίας είναι η σχεδίαση και η ανάπτυξη εφαρμογής για χρήση στον πολιτισμό και στον τουρισμό. Πιο συγκεκριμένα, η εφαρμογή περιέχει μια λίστα από μουσεία, τα οποία μπορεί να επισκεφθεί ο χρήστης, και στο καθένα από αυτά, με τη βοήθεια της κάμερας, να αναγνωρίζει το κάθε έκθεμα και να του εμφανίζει πληροφορίες για αυτό, χωρίς να χρειάζεται να πληκτρολογήσει κάτι. Επιπλέον, για κάθε μουσείο, υπάρχει το ωράριο λειτουργίας, η ακριβής τοποθεσία μέσα από τους χάρτες της Google και μια σύντομη περιγραφή. Δημιουργήθηκαν δύο εφαρμογές για κινητές συσκευές, η πρώτη για το λειτουργικό iOS της Apple με το λογισμικό XCode και το iOS SDK σε γλώσσα Swift και η δεύτερη για το λειτουργικό Android με το λογισμικό Android Studio και το Android SDK σε γλώσσα Java. Η εφαρμογή με την οποία επικοινωνούν και οι δύο αυτές εφαρμογές (Backend) αναπτύχθηκε με την βοήθεια του Spring framework σε γλώσσα Java. Χρησιμοποιήθηκε η MongoDB ως βάση δεδομένων λόγω της ταχύτητας και της ευελιξίας της. Τέλος, χρησιμοποιήθηκε και η πλατφόρμα Firebase Storage για την αποθήκευση των εικόνων της εφαρμογής.

Λέξεις Κλειδιά

Επαυξημένη Πραγματικότητα, Αναγνώριση Εικόνας, Πολιτισμός, Android, Android Studio, Java, iOS, iPhone, Εφαρμογή, MongoDB, Υπηρεσίες Τοποθεσίας, Firebase, AI

Abstract

The subject of this thesis is the design and development of an integrated mobile application for use in culture. Specifically, the mobile app contains a list of museums that the user can visit, identify exhibits and obtain related information with the help of a camera and without any typing. In addition, for each museum there is info abou opening hours, exact location through Google maps and a brief description. Two mobile apps were created, the first for Apple's iOS operating system using Xcode, Apple’s integrated development environment and Swift language SDK and the second for the Android operating system using Android Studio software and Java Android SDK. These applications communicate with a backend developed using Spring framework in Java. MongoDB was used as a database because of its speed and flexibility. Finally, to store application images the Firebase Storage platform was also used.

Key Words

Augmented Reality, Image Recognition, Culture, Android, Android Studio, Java, iOS, iPhone, Application, MongoDB, Location Services, Firebase, AI

Ευχαριστίες

Η παρούσα εργασία είναι το αποτέλεσμα της κοινής, επίμονης προσπάθειας και της επιτυχημένης συνεργασίας δύο συναδέλφων και πολύ καλών φίλων. Σε αυτό το σημείο δεν θα είχαμε οδηγηθεί ποτέ, χωρίς το γνήσιο ενδιαφέρον μας για το αντικείμενο με το οποίο καταπιάνεται η συγκεκριμένη εργασία.

Ολοκληρώνοντας ωστόσο το παρόν πόνημα, ήταν ηθικά επιβεβλημένο να αναγνωρίσουμε και να ευχαριστήσουμε όλους εκείνους, οι οποίοι στάθηκαν δίπλα μας και με τον τρόπο τους συνέβαλαν στο να πετύχουμε τους στόχους μας. Αυτοί φυσικά, δεν είναι άλλοι από την οικογένειά μας και τους καθηγητές, που είχαμε την τύχη να γνωρίσουμε και να μαθητεύσουμε πλάι τους κατά την διάρκεια των σπουδών μας. Η ευγνωμοσύνη που νιώθουμε για αυτούς είναι βαθειά κι ειλικρινής, για διαφορετικούς λόγους.

Τίποτε δεν θα είχε ξεκινήσει, χωρίς τις οικογένειές μας. Η διαρκής υποστήριξή τους, ήταν για εμάς το αναγκαίο συναισθηματικό εφόδιο, προκειμένου να θέτουμε στόχους στην πορεία μας και να αγωνιζόμαστε για την επίτευξή τους. Θα θέλαμε μέσα από αυτές τις λίγες γραμμές να σας ευχαριστήσουμε για τις θυσίες σας, την ανοχή σας και την δύναμη που μας μεταδώσατε με την στάση σας. Ελπίζουμε η παρούσα εργασία μας να σας δικαιώσει, κατά κάποιον τρόπο.

Από την άλλη, θα θέλαμε να ευχαριστήσουμε ολόψυχα τους διδάσκοντες καθηγητές μας για το επίπεδο των σπουδών που μας παρείχαν. Σε μία περίοδο γενικότερης απογοήτευσης, αποτέλεσαν πυρήνα έμπνευσης, καινοτομίας κι ελπιδοφόρας δημιουργικής απασχόλησης. Δεν ήταν μόνο οι επιστημονικές γνώσεις που μας μετέδωσαν. Πολύ περισσότερο, μας οδήγησαν στον κόσμο της έρευνας και της κριτικής σκέψης. Μας διαπαιδαγώγησαν, εξοικειώνοντάς μας με έναν τρόπο ζωής διαρκούς μελέτης, αναζήτησης κι επιτυχίας που έρχεται ως αποτέλεσμά τους. Μακάρι η εργασία αυτή να σταθεί αντάξια των όσων μας μεταδώσατε.

Περιεχόμενα

Ακρωνύμια

ACID - Atomicity, Consistency, Isolation, Durability

AP - Access Point

AR - Augmented Reality

BRIEF - Binary Robust Independent Elementary Features

CRT - Cathode Ray Tube

DGPS - Differential Global Positioning System

DoG - Difference of Gaussian

FAST - Features from Accelerated Segment Test

GPS - Global Positioning System

GSM - Global System for Mobile Communications

HMD - Head Mount Display

IP - Internet Protocol

LDA - Linear Discriminant Analysis

LSH - Locality-Sensitive Hashing

MAC - Media Access Control

ORB - Oriented FAST and Rotated BRIEF

PCA - Principal Component Analysis

PTAM - Parallel Tracking and Mapping

QR - Quick Response

SDK - Software Development Kit

SIFT - Scale-Invariant Feature Transform

SIM - Subscriber Identity Module

SLAM - Simultaneous Localization And Mapping

SSD - Sum of Squared Differences

SSID - Service Set Identifier

SURF - Speeded-Up Robust Features

VR - Virtual Reality

3D - 3 Dimensions

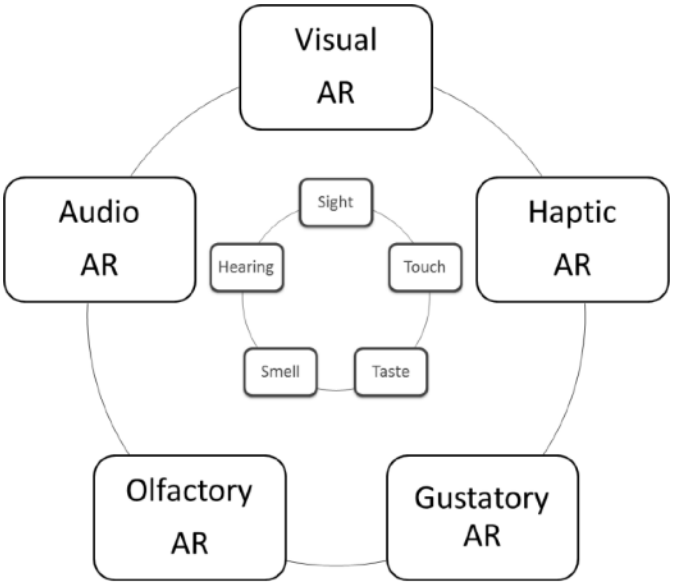
1. Εισαγωγή

1.1 Επαυξημένη Πραγματικότητα

1.1.1 Ορισμός και Κατηγοριοποίηση

> "Σε θεμελιώδεις όρους, η έκφραση επαυξημένη πραγματικότητα, συχνά εν συντομία AR (Augmented Reality) , αναφέρεται σε έναν απλό συνδυασμό του πραγματικού και του εικονικού (παραγόμενου μέσω Η/Υ) κόσμου. Δεδομένου ενός πραγματικού αντικειμένου, που καταγράφεται σε βίντεο ή σε εικόνα, η τεχνολογία επαυξάνει την εικόνα του πραγματικού κόσμου με επιπρόσθετα επίπεδα ψηφιακής πληροφορίας." [1]

Το AR έχει μια πληθώρα εφαρμογών - από την διασκέδαση, εκπαίδευση και ιατρική, τον τουρισμό και τον στρατό, μέχρι την τέχνη, την αρχαιολογία και την πλοήγηση. Σε αντίθεση με την γενική γνώμη, μπορεί να απευθύνεται και σε άλλες αισθήσεις εκτός από την όραση. Όπως φαίνεται στο Σχήμα 1, προς το παρόν έχουμε συστήματα AR που αφορούν την όραση, την ακοή, την αφή, την όσφρηση και την γεύση.

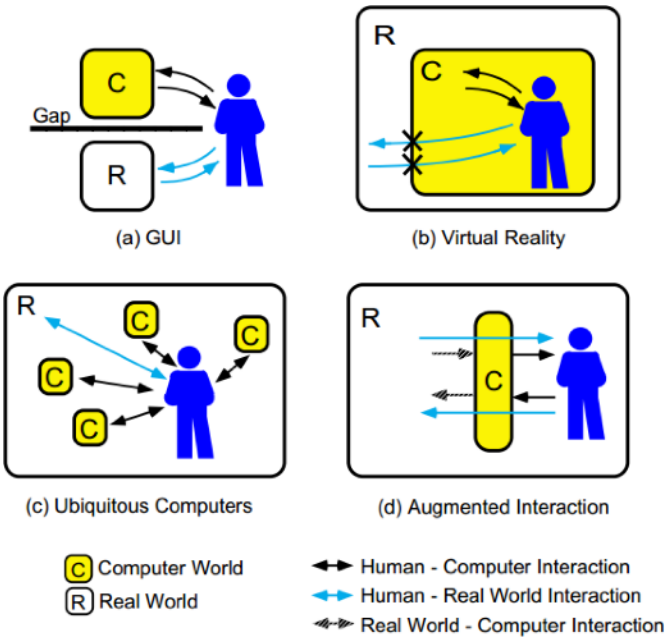


Σχήμα 1: Κατηγοριοποίηση της Επαυξημένη Πραγματικότητας [2]

Ένας από τους πιο κοινά αποδεκτούς ορισμούς του AR δόθηκε από τον ερευνητή Ronald Azuma το 1997 [3]. Η τεχνολογία του AR πρέπει:

1. Να συνδυάζει πραγματικό με ψηφιακό περιεχόμενο
2. Να είναι διαδραστική σε πραγματικό χρόνο
3. Να λαμβάνει χώρο σε τρεις διαστάσεις

Οι Rekimoto και Nagao το 1995 [4] αναφέρουν το AR ως μια πιθανή μορφή αλληλεπίδρασης ανθρώπου - υπολογιστή (HCI). Στο σχήμα 2 φαίνονται οι πιθανές προσεγγίσεις της μετάδοσης δεδομένων από τον υπολογιστή στον χρήστη. Ανάμεσα σε άλλες, αναφέρουν επίσης την εικονική πραγματικότητα (VR). Το AR και το VR (virtual reality - εικονική πραγματικότητα) μοιράζονται πολλές τεχνολογίες, όπως οθόνες που στερεώνονται στο κεφάλι και συστήματα tracking, αλλά το πρώτο σημείο του προαναφερθέντος ορισμού είναι αυτό που τις διαφοροποιεί σαν τεχνολογίες. Το VR αντικαθιστά τον πραγματικό κόσμο με ένα περιβάλλον που δημιουργείται τεχνητά, ενώ το AR επεκτείνει ή αλλάζει τον τρόπο που το υπάρχον περιβάλλον γίνεται αντιληπτό.



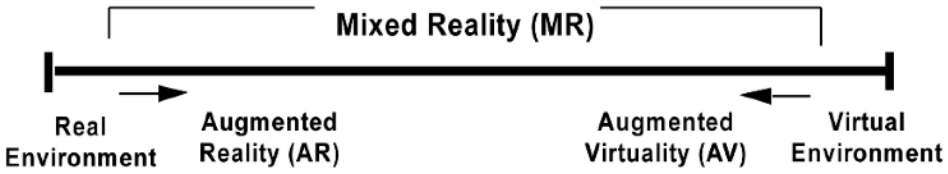
Σχήμα 2: Σύγκριση των ειδών αλληλεπίδρασης ανθρώπου μηχανής [4]

Έχουν επισημανθεί και άλλες διαφορές, όπως το ευρύτερο οπτικό πεδίο, τα ρεαλιστικά γραφικά, το λιγότερο ακριβές tracking και η δημιουργία περιβάλλοντος που αποκόπτει τον χρήστη από το φυσικό περιβάλλον, όσον αφορά το VR, σε

αντίθεση με τα διακριτικά γραφικά και την υψηλή ακρίβεια στο tracking από την πλευρά του AR.

Μια άλλη άποψη των διαφορών μεταξύ AR και VR δίνεται από τους Milgram & Kishino το 1994 [5]. Οι συγγραφείς εισάγουν την έννοια του "συνεχούς της εικονικότητας" - βλέπε σχήμα 3. Στα αριστερά, υπάρχει το "πραγματικό περιβάλλον", δηλαδή ένα περιβάλλον με "οποιοδήποτε αντικείμενο που έχει πραγματική υπόσταση". Το άλλο άκρο είναι το "εικονικό περιβάλλον", το οποίο περιέχει αποκλειστικά αντικείμενα που "υπάρχουν στην ουσία, αλλά όχι στην πραγματικότητα". Το πρώτο περιβάλλον μπορεί να γίνει αντιληπτό απευθείας ή μέσω κάποιας συσκευής απεικόνισης, ενώ το δεύτερο πρέπει να προσομοιωθεί πρώτα. Η "μικτή πραγματικότητα" επομένως είναι τι είδος του περιβάλλοντος όπου, τόσο πραγματικά όσο και εικονικά αντικείμενα "παρουσιάζονται μαζί μέσω της ίδιας συσκευής απεικόνισης". Η μετάβαση από το "πραγματικό" στο "εικονικό" δεν είναι διακριτή, αλλά εξαρτάται από το μέγεθος του περιεχομένου που προστίθεται.

Παρόμοιος τρόπος κατηγοριοποίησης παρουσιάζεται επίσης από τον Mann το 2002 [6], όπου το σύστημα του Milgram εμπλουτίζεται, παραδείγματος χάριν, με την στρέβλωση του βίντεο για να εξισορροπήσει την παραμόρφωση που προκαλείται από τον φακό ή από την αφαίρεση τμημάτων του πλάνου.



Σχήμα 3: Το συνεχές της πραγματικότητας - εικονικότητας [5]

Application	Tracking dof	Aug. type	Time
Archeoguide [33]	6D	VST	$< t_0$
Insitu [14]	6D	VST	t_0
ARMAR [11]	6D	OST	t_0
SAR [2]	6D	SAR	∞
Omnitouch [10]	$2D + \theta$	SAR	t_0
Metro [26]	$2D + \theta$	VST	t_0
Gmaps [9]	2D	VST	t_0
QR codes [5]	0D	VST	t_0
Nintendo [25]	6D	VST	∞
ARToolkit [13]	6D	VST	all

Σχήμα 4: Παραδείγματα κατηγοριοποίησης χρησιμοποιώντας τους 3 από τους 4 άξονες [9]

Παρόλο που ο μεγαλύτερος όγκος της έρευνας για την κατηγοριοποίηση του AR και του VR διεξήχθη την δεκαετία του 1990, υπάρχει πληθώρα από ενδιαφέροντα άρθρα ακόμα και από το πρόσφατο παρελθόν. Άξιες αναφοράς είναι η κατηγοριοποίηση βασισμένη στα υποσυστήματα του AR από τους Braz & Pereira το 2008 [7], κατηγοριοποίηση βασισμένη στον σκοπό λειτουργίας από τον Hugues το 2011 [8] ή τον Normand το 2012 [9], που χρησιμοποιούσε τέσσερις άξονες: τους βαθμούς ελευθερίας του tracking, τον τύπο της επαύξησης, την χρονική βάση και τους μη οπτικούς τρόπους απεικόνισης, για να περιγράψει τις εφαρμογές - βλέπε σχήμα 4.

1.1.2 Ιστορική Αναδρομή

Σε διάφορες πηγές, μπορεί κανείς να βρει αναφορές για "πρώιμα AR" συστήματα που χρονολογούνται στον 16/17ο αιώνα. Άξια αναφοράς είναι η ψευδαίσθηση, γνωστή ως "το φάντασμα του Pepper" ("Pepper's Ghost) [10], όπως ο Τσέχος φιλόσοφος Comenius, ο οποίος οραματίστηκε περί το 1631 κατά έναν αφηρημένο τρόπο, ειδικά γυαλιά που αλλάζουν τον τρόπο που βλέπει τον κόσμο αυτός που τα φοράει.

Παρόλα αυτά, οι περισσότερες πηγές συμφωνούν, ότι το πρώτο λειτουργικό πρωτότυπο AR, δημιουργήθηκε από τον Ivan Sutherland το 1966 [11]. Έφτιαξε ένα σύστημα με μια διάφανη οθόνη καθοδικού σωλήνα (CRT) που στερεωνόταν στο κεφάλι (HMD) και ένα μηχανικό σύστημα tracking, το οποίο αργότερα αντικαταστάθηκε από ένα σύστημα υπερήχου. Το σύστημά του είχε οπτικό πεδίο 40 μοιρών και τζάμια-καθρέπτες, που επέτρεπαν στον χρήστη να βλέπει παραγόμενες πληροφορίες σε συνδυασμό με τα πραγματικά περιβάλλοντα αντικείμενα. Σε επίπεδο οραματισμού, πήγε ακόμα πιο μακριά:

"Η απόλυτη οθόνη, βεβαίως, θα ήταν ένα δωμάτιο μέσα στο οποίο ο υπολογιστής θα μπορούσε να χειρίζεται την ύπαρξη της ύλης. Μια καρέκλα που θα απεικονιζόταν σε αυτό το δωμάτιο θα αρκούσε για να κάτσει κάποιος. Χειροπέδες που θα απεικονίζονταν σε ένα τέτοιο δωμάτιο θα μπορούσαν να περιορίσουν κάποιον, και μια σφαίρα που θα απεικονιζόταν σε ένα τέτοιο δωμάτιο θα ήταν θανάσιμη."

Επίσης, οι ένοπλες δυνάμεις διεξήγαγαν την δική τους έρευνα πάνω στο AR, αναπτύσσοντας το πρόγραμμα "Super Cockpit" για την Αμερικανική Πολεμική

Αεροπορία [12]. Το πρωτότυπο αποτελούνταν από ένα κράνος εξοπλισμένο με HMD, το οποίο πρόβαλλε λεπτομέρειες της πτήσης "μέσα" στο οπτικό πεδίο του πιλότου. Το πρόγραμμα οδήγησε με επιτυχία στο σύστημα στόχευσης κράνους στα ελικόπτερα Apache.

Ένα άλλο ενδιαφέρον πρόγραμμα ήταν το VIEW (Virtual Interface Environment Workstation) της NASA. Όπως περιγράφηκε από τον Fisher το 1986 [13], το σύστημα αποτελούνταν από στερεοσκοπική οθόνη ευρείας γωνίας με tracking 6 βαθμών ελευθερίας, ένα γάντι για απτική είσοδο, αναγνώριση φωνής συνδεδεμένη με τρισδιάστατο ήχο και τεχνολογία σύνθεσης φωνής.

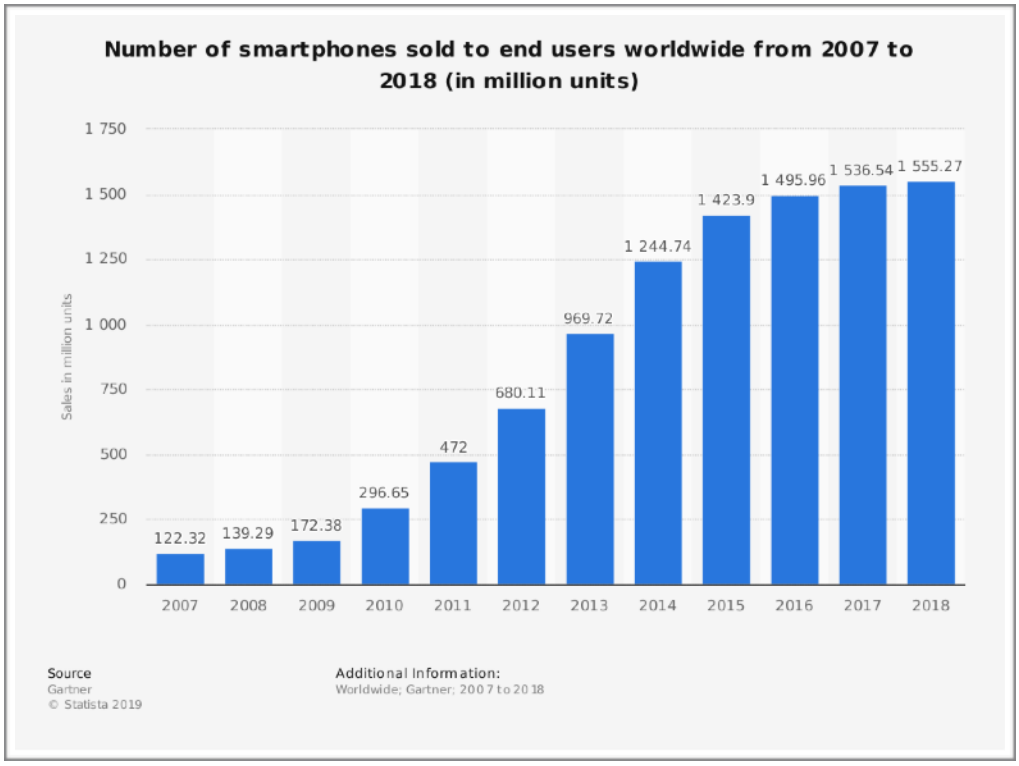
Όσον αφορά την έκφραση "Επαυξημένη πραγματικότητα" ("Augmented Reality"), πιθανότερα εμφανίστηκε το 1992, όταν ο Tom Caudell και ο David Mizell [14], οι οποίοι εργάζονταν για την εταιρία Boeing στην απλοποίηση της συναρμολόγησης των αεροσκαφών, δημοσίευσαν την έρευνά τους.

Στα χρόνια που ακολούθησαν, πληθώρα ερευνητικών ομάδων, τόσο σε ακαδημαϊκό επίπεδο όσο και στον χώρο της βιομηχανίας, εδραιώνονταν, εστιάζοντας την έρευνά τους στην αναγνώριση αντικειμένων, στις τεχνικές tracking, στις οθόνες HMD και στις τεχνικές αλληλεπίδρασης.

Οπότε, παρόλο που το φαινόμενο του AR δεν είναι καινούργιο, δημοφιλές έγινε τα τελευταία χρόνια. Ο Geroimenko το 2012 [2] εντοπίζει μερικούς πιθανούς λόγους:

"Κατ' αρχήν, οφείλεται στο γεγονός ότι η Επαυξημένη Πραγματικότητα είναι ένας φυσικός τρόπος εξερεύνησης 3D (τρισδιάστατων) αντικειμένων και δεδομένων, καθώς φέρνει εικονικά αντικείμενα μέσα στον πραγματικό κόσμο, όπου ζούμε. Δεύτερον, οφείλεται στο γεγονός ότι οι δυνατότητες του AR είναι ατελείωτες…"

Από την έλευση του iPhone το 2007, και της πλατφόρμας του Android το 2008, παρουσιάζεται τεράστια ανάπτυξη στην εμπορική αγορά του AR, καθώς η αναγκαία τεχνολογία για το AR γίνεται διαθέσιμη στην μάζα των καταναλωτών - βλέπε Σχήμα 5. Όπως μπορούμε να δούμε, πάνω από 1.5 δις ανθρώπων έχουν πρόσβαση σε τεχνολογία που παρέχει 6 βαθμούς ελευθερίας tracking, επομένως το AR μπορεί να μεταφερθεί από τα εργαστήρια στον μέσο καταναλωτή.



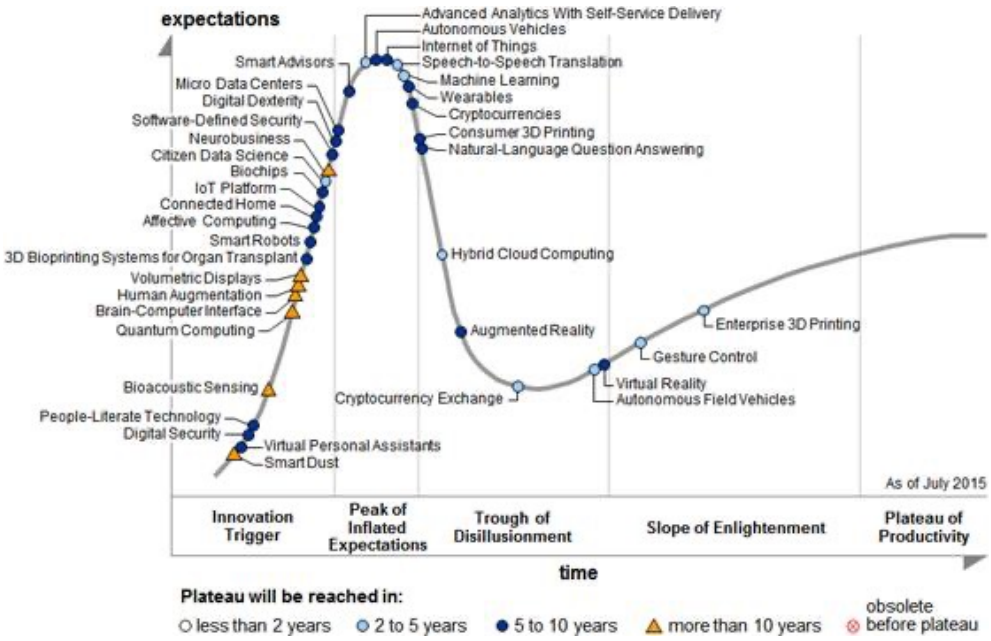
Σχήμα 5: Αριθμός πωλήσεων smartphone στους καταναλωτές παγκοσμίως από το 2007 μέχρι το 2018 (εκατομμύρια μονάδες) [15]

1.1.3 Η έρευνα και το μέλλον

Το AR έχει πλούσια 50ετή ιστορία έρευνας και ανάπτυξης εφαρμογών. Όπως μας δείχνει το σχήμα 6 της Gartner, το AR είναι ήδη από το 2018 στην φάση της "απότομης απογοήτευσης" και αναμένεται να φτάσει στην φάση της σταθεροποίησης στην παραγωγή σε 5-10 χρόνια [16]. Στα χρόνια που θα ακολουθήσουν, σύμφωνα με τον Billinghurst [17], το AR θα εξελιχθεί σε τρεις βασικές τεχνολογίες - στην συσκευή απεικόνισης, στην αλληλεπίδραση και στο tracking.

Η κύρια εξέλιξη στις συσκευές απεικόνισης αναμένεται να συμπεριλαμβάνει λεπτές ημιδιάφανες οθόνες με ευρύ οπτικό πεδίο και στερεοσκοπικούς προβολείς κεφαλής (μέσα σε 5 χρόνια) και φακούς επαφής ή οθόνες αμφιβληστροειδούς μέσα σε 5 ή περισσότερα χρόνια.

Η αλληλεπίδραση λογικά θα μετατοπιστεί από τον χώρο της οθόνης στον χώρο του πραγματικού περιβάλλοντος, με αισθητήρες βάθους αλλά και διάφορους άλλους αισθητήρες. Οι φυσικές χειρονομίες αναμένονται να μπουν στους τρόπους αλληλεπίδρασης μέσα στα επόμενα 3 χρόνια, εξελίσσοντας την αλληλεπίδραση σε



Σχήμα 6: Gartner's Hype Cycle (2018) [16]

πολυτροπική (πχ. ομιλία και χειρονομίες) και ευφυή, αναγνωρίζοντας την συμπεριφορά του χρήστη σε βάθος χρόνου.

Τέλος, το tracking, αναμένεται, στα επόμενα 3 χρόνια, να μεταφερθεί από τα τρισδιάστατα μοντέλα και την παρακολούθηση παραμορφώσιμων αντικειμένων, στην παρακολούθηση του περιβάλλοντος χρησιμοποιώντας αισθητήρες βάθους. Στο απώτερο μέλλον, θα εξελιχθεί σε παρακολούθηση ανοιχτού εξωτερικού χώρου και σε απρόσκοπτη παρακολούθηση εσωτερικού χώρου.

Μια άλλη εξέλιξη θα εστιαστεί στις κοινωνικές πτυχές, καθώς λιγότερα από το 8% των δημοσιευμένων άρθρων έχουν επίσημες μελέτες χρηστών [18]. Άλλες πιθανές περιοχές ανάπτυξης είναι η αναζήτηση και μεταφορά δεδομένων, το cloud computing, οι συνεργατικές λειτουργίες και η ασφάλεια.

1.1.4 Συστήματα AR Εξωτερικών Χώρων σε Κινητές Συσκευές

Πολλά προγράμματα εστίαζαν στο AR για εξωτερικούς χώρους. Ένα από τα πρώτα ήταν το "Touring Machine", το οποίο δημιουργήθηκε από τον Höllerer το 1997 [19] και χρησιμοποιούσε GPS και μαγνητόμετρο. Μεταξύ 1998 - 2006 διεξήχθη έρευνα στο Πανεπιστήμιο της Νότιας Αυστραλίας, η οποία είχε σαν αποτέλεσμα το πρόγραμμα Tinmith. Το πρόγραμμα είναι μοναδικό, λόγω της ανάπτυξης υλισμικού

αποκλειστικής χρήσης, όπως ήταν το κράνος, τα γάντια και το σακίδιο μεταφοράς των ηλεκτρονικών εξαρτημάτων. Η πιο δημοφιλής χρήση του είναι ένα από τα πρώτα παιχνίδια AR για φορητές συσκευές - το ARQuake, που παρουσιάστηκε το 2000 από τον Thomas [20].

Ο Βλαχάκης το 2002 [21] έδειξε πώς το AR μπορεί να χρησιμοποιηθεί σαν τμήμα ενός ιστορικού μνημείου. Το σύστημά τους αποκτούσε πληροφορίες τοποθεσίας μέσω DGPS (βλέπε Εντοπισμός GPS) σε συνδυασμό με την σύγκριση live εικόνων με βαθμονομημένες εικόνες από σημεία ενδιαφέροντος. Ενδιαφέρον είναι το γεγονός ότι χρησιμοποίησαν το θεώρημα μετατόπισης Fourier για να υπολογίσουν τον μετασχηματισμό των εικόνων.

Μια άλλη προσέγγιση στο AR σε σχέση με την ιστορική κληρονομιά παρουσιάζεται από τον Zoellner το 2008 [22]. Οι συγγραφείς δείχνουν πώς τα σκίτσα και οι πίνακες ζωγραφικής μπορούν να υπερτίθενται στην ζωντανή εικόνα της κάμερας.

Άλλα ενδιαφέροντα επιμέρους προγράμματα δημιουργήθηκαν επίσης στο πλαίσιο του ερευνητικού προγράμματος του IPCity (2006-2010):

- Time Warp (Herbst, 2008), ένα παιχνίδι reality τοποθετημένο στην πόλη της Κολονίας (Γερμανία), το οποίο συνδυάζει τεχνητά στατικά και κινούμενα αντικείμενα με ήχο.

- MapLens (Morrison, 2008), ένα παιχνίδι βασισμένο στην τοποθεσία που συνδυάζει χάρτες από χαρτί και επικάλυψη της εικόνας με ψηφιακή πληροφορία που περιλαμβάνει φωτογραφίες και σημεία ενδιαφέροντος.

Οι Mata & Claramunt το 2014 [23] δείχνουν πώς τα δεδομένα από κοινωνικά δίκτυα (το Twitter εν προκειμένω) μπορούν να χρησιμοποιηθούν για να δημιουργήσουν έναν τουριστικό οδηγό της πόλης, βασισμένο σε προτιμήσεις του χρήστη και προτάσεις από άλλους ανθρώπους

Το AR μπορεί επίσης να χρησιμοποιηθεί σαν μέρος εγκληματολογικής έρευνας, όπως προτείνει ο Gee το 2013 [24]. Το σύστημά τους παρέχει έναν τρόπο σήμανσης του περιβάλλοντος χώρου και των αντικειμένων, με την χρήση τεχνολογίας GPS και UWB.

Τέλος, άξιοι αναφοράς είναι οι Reitmayr & Drummond το 2006 [25], οι οποίοι χρησιμοποίησαν υβριδικό σύστημα tracking, συνδυάζοντας καινοτόμο μέθοδο εντοπισμού και παρακολούθησης (edge-based) με ανάγλυφα τρισδιάστατα αντικείμενα, γυροσκόπιο, μαγνητόμετρο και αισθητήρα βαρύτητας.

1.2 Παρακολούθηση αντικειμένων και Υπολογισμός στάσης

Η πιο σημαντική λειτουργία κάθε συστήματος AR είναι η ικανότητα να "αγκιστρώνει" το τεχνητό περιεχόμενο στον πραγματικό κόσμο. Ο στόχος είναι η εύρεση και η παρακολούθηση της θέσης και του προσανατολισμού ενός αντικειμένου (αισθητήρα ή κάμερας) σε σχέση με ένα άλλο αντικείμενο (άγκυρα). Η θέση ορίζει τρεις βαθμούς ελευθερίας, όπως και ο προσανατολισμός. Η διαδικασία έχει συνήθως μέχρι δύο φάσεις. Η πρώτη αποκαλείται συνήθως φάση καταχώρησης, και κατά την διάρκειά της λαμβάνεται μία θέση της άγκυρας και του αισθητήρα. Η δεύτερη φάση αποκαλείται φάση παρακολούθησης, κατά την οποία ενημερώνεται η γνωστή στάση. Ανάλογα με την τεχνολογία που χρησιμοποιείται, αυτή η τελευταία φάση μπορεί να μην είναι απαραίτητη και μπορεί να αντικατασταθεί από άλλη καταχώρηση.

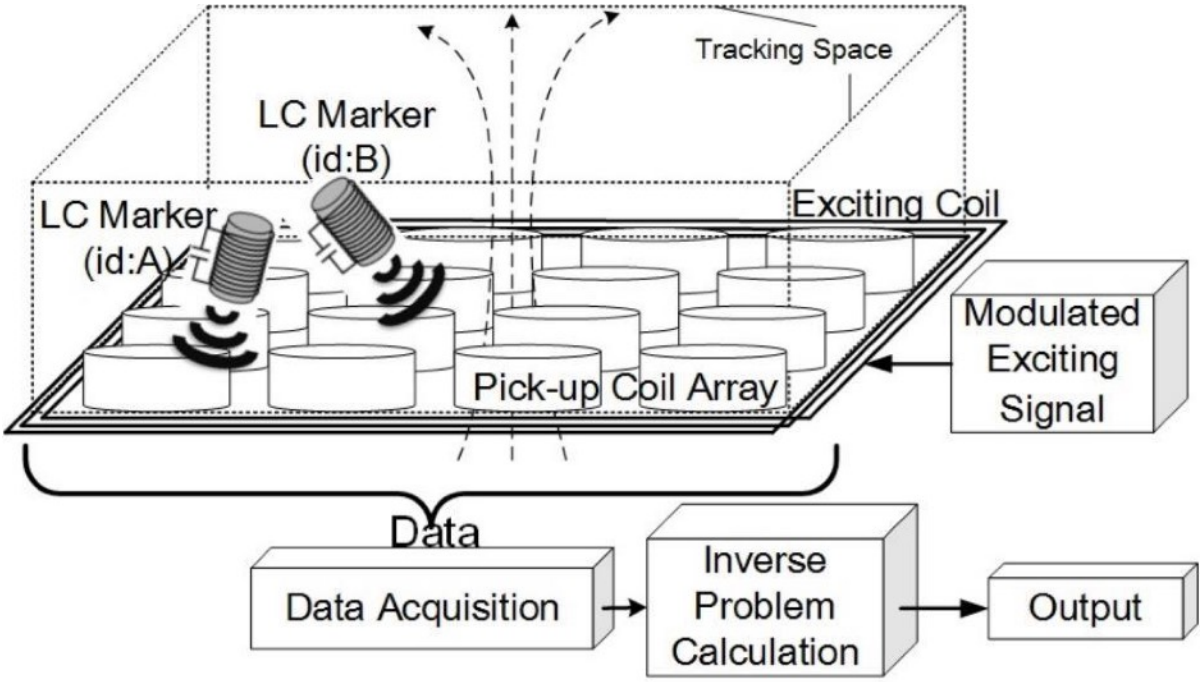
Παρακάτω παρουσιάζονται οι διαθέσιμες τεχνολογίες παρακολούθησης με τη σύντομη περιγραφή, τα πλεονεκτήματα και τα μειονεκτήματά τους.

1.2.1 Μαγνητική Παρακολούθηση

Αυτή η τεχνολογία χρειάζεται μια πηγή δημιουργίας (εναλλασσόμενου ή σταθερού) ηλεκτρομαγνητικού πεδίου και έναν ή περισσότερους αισθητήρες τοποθετημένους εντός του πεδίου (Σχήμα 7). Η θέση και ο προσανατολισμός των αισθητήρων στη συνέχεια υπολογίζονται βάσει των αλλαγών εξασθένησης και προσανατολισμού του πεδίου. Όσο οι αισθητήρες βρίσκονται στο πεδίο, αυτό το σύστημα επιτρέπει σταθερή και ακριβή παρακολούθηση και στους έξι βαθμούς ελευθερίας.

+ Μπορεί να παρακολουθήσει αντικείμενο πίσω από άλλα αντικείμενα, δηλ. οι αισθητήρες και η πηγή δεν πρέπει να έχουν άμεση οπτική επαφή, μικροί και ελαφρείς αισθητήρες

- Χρειάζεται ειδικό εξοπλισμό, περιορισμένη εμβέλεια καθώς η ισχύς του μαγνητικού πεδίου πέφτει ανάλογα με τον κύβο της απόστασης, ευαίσθητη σε παράσιτα



Σχήμα 7: Οι δείκτες που τοποθετούνται σε μαγν. πεδίο είναι διεγερμένοι και δημιουργούν μαγνητική ροή, που συλλέγεται από συστοιχία πηνίων, μετατρέπεται σε σήμα τάσης και επεξεργάζεται περαιτέρω. [26]

1.2.2 Ακουστική Παρακολούθηση

Η ακουστική παρακολούθηση συνήθως χρησιμοποιεί υπερηχητικούς πομπούς και δέκτες σήματος. Ο δέκτης είναι σε θέση να υπολογίσει την απόσταση από τον πομπό μετρώντας το χρόνο που απαιτείται για το ταξίδι του σήματος στον αέρα (Time of Flight). Αυτό απαιτεί τουλάχιστον τρεις πομπούς για να εκτιμήσουν και τους έξι βαθμούς ελευθερίας.

+ Μικρό σε μέγεθος και φτηνό hardware, δουλεύει και κάτω από το νερό

- Μπορεί να επηρεαστεί από διάφορες περιβαλλοντικές συνθήκες, όπως αλλαγές στην πίεση και την θερμοκρασία, χαμηλή ευκρίνεια και διακύμανση της έντασης του σήματος

1.2.3 Οπτική παρακολούθηση

1.2.3.1 Αόρατο Φως

Συνήθως βασισμένο στο υπέρυθρο φάσμα. Ένα αντικείμενο είναι εξοπλισμένο με δείκτες, οι οποίοι είτε αντανακλούν είτε εκπέμπουν φως. Το φως συλλέγεται από μία ή πολλές κάμερες τοποθετημένες σε (όχι απαραίτητα) γνωστή θέση και η θέση του υπό παρακολούθηση αντικειμένου υπολογίζεται στην συνέχεια από συγκερασμό τον δεδομένων κάθε κάμερας. Παρόμοια, στα μαγνητικά και ακουστικά συστήματα, χρειάζεται κανείς τουλάχιστον τρεις δείκτες σε σταθερή διάταξη για τον προσδιορισμό του προσανατολισμού. Είναι αξιοσημείωτο ότι οι δείκτες μπορεί να είναι μέρος του περιβάλλοντος και οι κάμερα/ες μπορεί να είναι προσαρτημένες στα παρακολουθούμενα αντικείμενα ή το αντίστροφο.

+ Ευκολία χρήσης, γρήγορη βαθμονόμηση, υψηλή ακρίβεια

- Τιμή, δεν μπορεί να παρακολουθήσει αντικείμενα κρυμμένα από άλλα αντικείμενα (αντιμετωπίζεται με περισσότερους δείκτες/κάμερες) χρειάζεται ειδικό εξοπλισμό και υποδομή

1.2.3.2 Παρακολούθηση Σταθερού Δείκτη

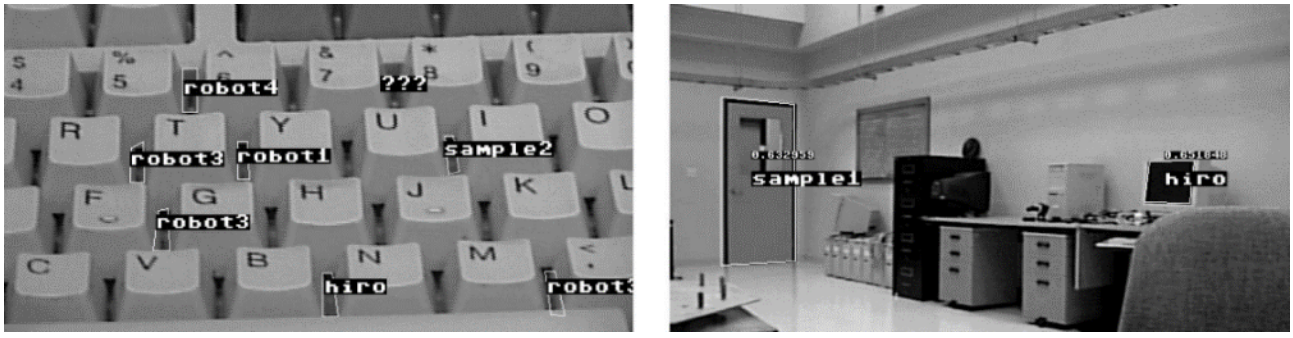
Κάποιες φορές αποκαλείται απλά παρακολούθηση δείκτη (marker tracking), χρησιμοποιεί ειδικά ορόσημα τα οποία προστίθενται στο πραγματικό περιβάλλον. Ο δείκτης έχει λάβει διάφορες μορφές με την πάροδο του χρόνου στις διαφορετικές εφαρμογές AR, ξεκινώντας από (χρωματιστά) κομμάτια χαρτιού και καταλήγοντας σε QR codes - βλέπε Σχήμα 8.

Σχήμα 8: είδη δεικτών AR

Σήμερα, οι περισσότερες από τις εφαρμογές χρησιμοποιούν τετράπλευρους επίπεδους δείκτες, για σχετική ευκολία ανίχνευσης και καλή ακρίβεια. Οι γωνίες του καθορίζουν τα ελάχιστα τέσσερα σημεία που είναι απαραίτητα για την εκτίμηση της θέσης, έτσι ώστε το εσωτερικό του δείκτη να μπορεί να χρησιμοποιηθεί για μοναδική αναγνώριση.

Οι κυκλικοί δείκτες έχουν το πλεονέκτημα ότι προσφέρουν μεγάλη ακρίβεια στον εντοπισμό του κέντρου του κύκλου και ότι το κυκλικό πρότυπο παραμορφώνεται λιγότερο υπό προοπτική.

Ένα πολύ γνωστό εργαλείο που χρησιμοποιεί δείκτες είναι το ARToolKit [27], το οποίο χρησιμοποιεί δυαδικό thresholding και μερική προσαρμογή γραμμής για την ανίχνευση δεικτών. Το μειονέκτημα του ARToolKit ήταν, μεταξύ άλλων, αδυναμία εντοπισμού αντικειμένων πίσω από άλλα αντικείμενα και μερικές φορές εσφαλμένοι εντοπισμοί αντικειμένων - βλ. Σχήμα 9.



Σχήμα 9: παραδείγματα από εσφαλμένες αναγνωρίσεις του ARToolKit

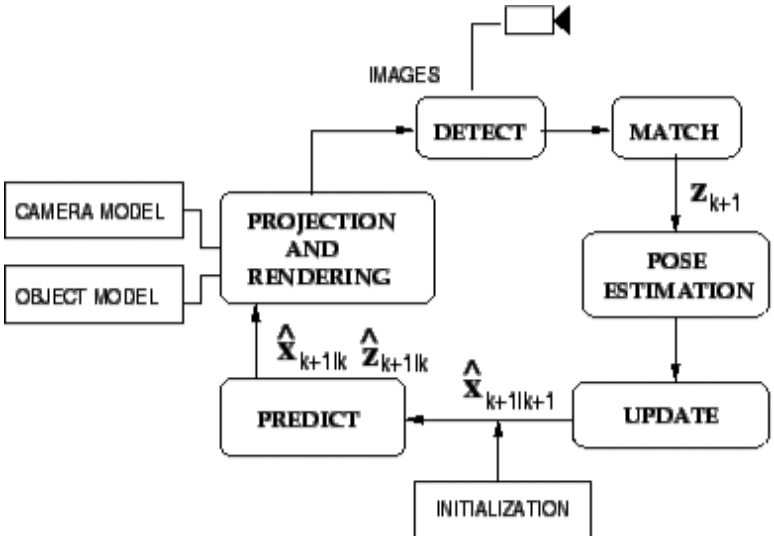
Από την άλλη πλευρά, το σύστημα ήταν πολύ γρήγορο και, αν ήταν επιτυχημένο στον εντοπισμό, ήταν επίσης ακριβές. Λόγω της δημοτικότητάς του, υπάρχει αριθμός εκδόσεων που μεταφέρονται σε διαφορετικές γλώσσες προγραμματισμού (NyARToolkit), φορητές συσκευές (AndAR), βελτιώνοντας τους αλγόριθμους ανίχνευσης και αντιστοίχισης (ARTag) ή προσθέτοντας υποστήριξη πιο περίπλοκων δεικτών (Studierstube ES).

1.2.3.3 Παρακολούθηση Φυσικών Στοιχείων

Παρουσιάζεται εκτενώς στο κεφάλαιο “Τεχνολογίες”.

1.2.3.4 Παρακολούθηση βασισμένη σε Μοντέλα

Μπορούν επίσης να χρησιμοποιηθούν τρισδιάστατα μοντέλα για την εκτίμηση της θέσης. Ορισμένες από τις πρώτες τεχνικές χρησιμοποιούν ανίχνευση άκρων, καθώς είναι σχετικά απλές, γρήγορες και σταθερές στις αλλαγές φωτισμού. Για παράδειγμα, το RAPID [28] ταιριάζει τα σημεία των ακμών του τρισδιάστατου μοντέλου, τα λεγόμενα σημεία ελέγχου, με τις περιοχές με ακμές υψηλής αντίθεσης - βλ. Σχήμα 10.



Σχήμα 10: Τυπικό σύστημα εντοπισμού βασισμένο σε μοντέλα [29]

Άλλη πιθανή προσέγγιση είναι η αντιστοίχιση των αρχέτυπων μοντέλων με αρχέτυπα που εξάγονται από την εικόνα, όπως τμήματα γραμμών ή καμπύλες - βλ. Σχήμα 11.

Σχήμα 11: Ταίριασμα καμπύλων

1.2.3.5 Παρακολούθηση 3D δομών

Η φυσική επέκταση της παρακολούθησης βάσει μοντέλου είναι η παρακολούθηση δομών 3D. Αυτή η τεχνική χρησιμοποιεί εξειδικευμένους αισθητήρες για τη λήψη πληροφοριών βάθους από τη σκηνή. Πιθανώς η πιο γνωστή συσκευή είναι η Microsoft Kinect, εξοπλισμένη με δομημένο αισθητήρα βάθους φωτός. Αυτός ο αισθητήρας εκπέμπει υπέρυθρο μοτίβο στο περιβάλλον. Καθώς το φως επιδρά στα αντικείμενα, παραμορφώνεται και από την προβολή είναι δυνατόν να υπολογιστεί ο χάρτης βάθους του περιβάλλοντος. Αυτή η τεχνική είναι γρήγορη και φθηνή, ωστόσο υποφέρει από το φαινόμενο της ανάκλασης και την περιορισμένη εμβέλεια.

1.2.3.6 SLAM

Ο ταυτόχρονος εντοπισμός και η χαρτογράφηση είναι μια τεχνική κατασκευής χάρτη σε άγνωστο περιβάλλον, ενώ ταυτόχρονα χρησιμοποιείται ο χάρτης που δημιουργείται για την ενημέρωση της τρέχουσας θέσης. Είναι ένα από τα δύσκολα προβλήματα στη ρομποτική με πολλές εφαρμογές - οικιακά ρομπότ (ηλεκτρική σκούπα, χλοοκοπτικά), αυτόνομα οχήματα, αυτόνομοι διαστημικοί περιηγητές κλπ. Το SLAM προτάθηκε αρχικά από τους Smith (κ.α, 1987) και βελτιώθηκε από τους Leonard & Durrant-Whyte το 1991 [30]. Τα βασικά βήματα του αλγόριθμου είναι η εξαγωγή ορόσημων, η συσχέτιση δεδομένων, η εκτίμηση κατάστασης και η ενημέρωση της κατάστασης και των ορόσημων.

1.2.3.7 PTAM

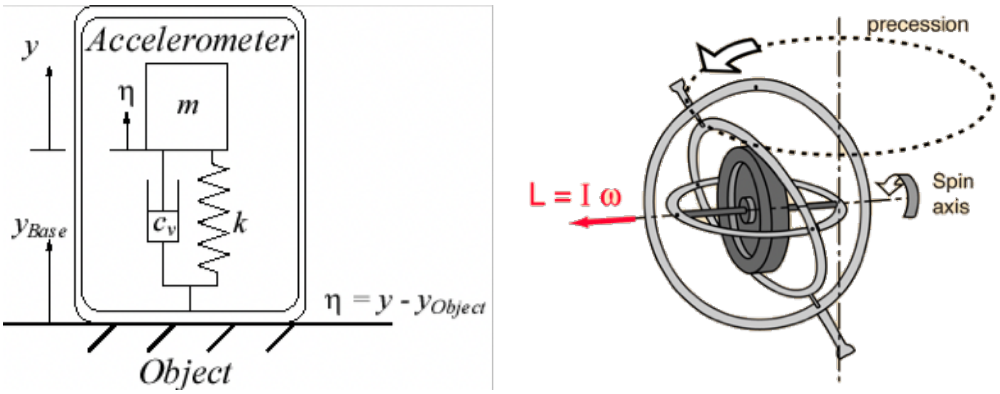
Η παράλληλη παρακολούθηση και χαρτογράφηση προτάθηκε από τους Klein & Murray το 2007 [31]. Σε αντίθεση με το SLAM, που είχε αρχικά σχεδιαστεί για χρήση στη ρομποτική, το PTAM είναι σχεδιασμένο για φορητές κάμερες. Η παρακολούθηση και η χαρτογράφηση χωρίζονται σε δύο φάσεις, προσφέροντας καλύτερη απόδοση και σταθερότητα. Σε σύγκριση με το SLAM, το PTAM χρησιμοποιεί πυκνότερο χάρτη με χαρακτηριστικά χαμηλότερης ποιότητας και απαιτεί (απλή) αρχικοποίηση.

1.2.3.8 Παρακολούθηση Οπτικής Ροής

Η παρακολούθηση μπορεί επίσης να γίνει με τον υπολογισμό της κίνησης των pixel μέσα στην εικόνα. Η ταχύτητα των pixel μπορεί να υπολογιστεί υπό την προϋπόθεση σταθερής έντασης. Η οπτική ροή ορίζεται στη συνέχεια ως διανυσματικό πεδίο της κίνησης των αντίστοιχων pixel μεταξύ των εικόνων. Ο πιο γνωστός αλγόριθμος για την οπτική ροή είναι η μέθοδος Lucas-Kanade (1981) [32]. Αυτή η μέθοδος σπάνια χρησιμοποιείται μόνη της, αλλά παρέχει αρκετά καλά αποτελέσματα ως μέρος υβριδικής παρακολούθησης [33].

1.2.4 Παρακολούθηση Αδράνειας

Η αδρανειακή παρακολούθηση μπορεί να χρησιμοποιηθεί για την εκτίμηση του προσανατολισμού με τη χρήση επιταχυνσιόμετρων, γυροσκοπίων και μαγνητόμετρων (Σχήμα 12). Το μεγαλύτερο πλεονέκτημά τους είναι η μηδαμινή καθυστέρηση και γενικά η καλή αντίσταση στις παρεμβολές. Ωστόσο, παρέχουν μόνο σχετική θέση, με εξαίρεση το μαγνητόμετρο (το οποίο μπορεί να παρέχει προσανατολισμό προς γεωγραφικό ή γεωμαγνητικό Βόρειο Πόλο). Οι αδρανειακοί αισθητήρες είναι επίσης ευαίσθητοι στη μετατόπιση με την πάροδο του χρόνου.



Σχήμα 12: (αριστερά) επιταχυνσιόμετρο, (δεξιά) γυροσκόπιο

1.2.5 Παρακολούθηση GPS

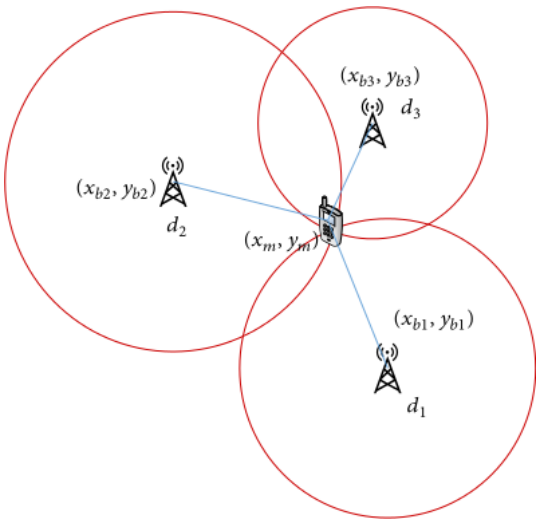
Το Παγκόσμιο Σύστημα Εντοπισμού είναι ένα στρατιωτικό σύστημα των ΗΠΑ που χρησιμοποιεί δορυφόρους για τον προσδιορισμό της θέσης των δεκτών (και της κατεύθυνση, σε περίπτωση κίνησης των δεκτών). Οι δορυφόροι μεταδίδουν τη θέση τους και τον πολύ ακριβή χρόνο. Ο δέκτης μπορεί να τριγωνίσει τη θέση του πάνω

στον πλανήτη με ακρίβεια που κυμαίνεται από μερικά μέτρα έως εκατοστά, σε περίπτωση που έχει δεδομένα από τουλάχιστον τρεις δορυφόρους. Η πραγματική ακρίβεια μπορεί να εξαρτάται από τον δέκτη, τις ατμοσφαιρικές επιδράσεις, τον πόσο καθαρός είναι ο ουρανός και τον τρέχοντα σχηματισμό των δορυφόρων. Η ακρίβεια του GPS μπορεί να ενισχυθεί περαιτέρω χρησιμοποιώντας σταθμούς εδάφους που μεταδίδουν τη θέση τους (Differential GPS), κεραίες κινητής τηλεφωνίας (Assisted GPS), γεωστατικούς δορυφόρους μαζί με DGPS σταθμούς (Wide Area Augmentation System, διαθέσιμο μόνο στις ΗΠΑ), φάση σήματος (Κινηματικό GPS πραγματικού χρόνου) και μετρήσεις σε δεύτερο χρόνο (ακρίβεια χιλιοστού) [34].

Παρόλο που το GPS είναι το πιο γνωστό και χρησιμοποιημένο, υπάρχουν άλλα συστήματα που υπάρχουν ή αναπτύσσονται / σχεδιάζονται, ήτοι το ρωσικό GLONASS, το Ευρωπαϊκό Γαλιλαίο, το Ιαπωνικό QZSS, το Ινδικό IRNSS ή το κινεζικό COMPASS.

1.2.6 Παρακολούθηση πύργων κυψελωτού δικτύου

Οι φορητές συσκευές που είναι εξοπλισμένες με κάρτες SIM (μονάδα GSM) διατηρούν πάντοτε σύνδεση με έναν από τους πλησιέστερους πύργους. Μετρώντας τα επίπεδα ισχύος, μπορεί να τριγωνιστεί η θέση της συσκευής με διαφορετική ακρίβεια (από δεκάδες μέτρα στην πόλη σε χιλιόμετρα στις αγροτικές περιοχές) - βλέπε σχήμα 13.



Σχήμα 13: Τριγωνισμός θέσης

1.2.7 Παρακολούθηση Wi-Fi

Ομοίως με το σήμα GSM, μπορεί κανείς να χρησιμοποιήσει την γνωστή θέση των Wi-Fi hotspot και δρομολογητών (Access Point - AP) για να καθορίσει την τρέχουσα θέση. Κάθε AP εκπέμπει τη διεύθυνση MAC και το SSID (Service Set ID). Με βάση την ισχύ του σήματος που λαμβάνεται από τα διαθέσιμα δίκτυα, είναι δυνατό να τριγωνιστεί η σχετική με τα AP θέση. Η απόλυτη θέση μπορεί στη συνέχεια να αποκτηθεί μέσω εντοπισμού διευθύνσεων IP ή μέσω (δημόσιων) βάσεων δεδομένων τοποθεσίας Wi-Fi, π.χ. Combain Positioning Service, Geomena ή Navizon. Όπως μας δείχνει ο Sapiezynski το 2015 [35], τα δεδομένα αυτά μπορούν να χρησιμοποιηθούν για τον προσδιορισμό των χρόνων άφιξης και αναχώρησης των ανθρώπων, του χρόνου που ξοδεύουμε στη διαμετακόμιση κ.λπ. Επιπλέον, όπως δείχνουν οι Adib & Katabi το 2013 [36] μπορεί επίσης να χρησιμοποιηθεί "ενεργά" για την ανίχνευση κινούμενων αντικειμένων πίσω από εμπόδια.

1.2.8 Υβριδική Παρακολούθηση

Δεδομένου ότι όλες οι προαναφερθείσες τεχνικές έχουν τις αδυναμίες τους στην ακρίβεια ή τις απαιτήσεις, μπορούν να συνδυαστούν για να δημιουργήσουν πιο ισχυρά συστήματα παρακολούθησης. Αυτή η τεχνική μπορεί επίσης να προσθέσει βαθμούς ελευθερίας, π.χ. με τον συνδυασμό δεδομένων παρακολούθησης από GPS (3 βαθμοί ελευθερίας - θέσης) και αδρανειακών αισθητήρων ή κάμερας (3 βαθμοί ελευθερίας - προσανατολισμός), όπως παρουσιάζεται από τον Piekarski το 2003 [37].

2. Τεχνολογίες

2.1 Android OS

Το Android είναι ένα λειτουργικό σύστημα για συσκευές κινητής τηλεφωνίας το οποίο τρέχει τον πυρήνα του λειτουργικού Linux. Αρχικά αναπτύχθηκε από την Google και αργότερα από την Open Handset Alliance. Επιτρέπει στους προγραμματιστές να γράφουν κώδικα με την χρήση της γλώσσας προγραμματισμού Java και της Kotlin, 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 τηλέφωνα και τα τάμπλετ, με διαφορετικό περιβάλλον χρήσης για τηλεοράσεις (Android TV), αυτοκίνητα (Android Auto) και ρολόγια χειρός (Android Wear). Παρόλο που έχει αναπτυχθεί για συσκευές με οθόνη αφής, έχει χρησιμοποιηθεί σε κονσόλες παιχνιδιών, ψηφιακές φωτογραφικές μηχανές, συνηθισμένους Η/Υ (π.χ. το HP Slate 21) και σε άλλες ηλεκτρονικές συσκευές.

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.

Η πρώτη παρουσίαση της πλατφόρμας Android έγινε στις 5 Νοεμβρίου 2007, παράλληλα με την ανακοίνωση της ίδρυσης του οργανισμού Open Handset Alliance, μιας κοινοπραξίας 48 εταιριών τηλεπικοινωνιών, λογισμικού καθώς και κατασκευής hardware, οι οποίες είναι αφιερωμένες στην ανάπτυξη και εξέλιξη ανοιχτών προτύπων στις συσκευές κινητής τηλεφωνίας. Η Google δημοσίευσε το μεγαλύτερο μέρος του κώδικα του Android υπό τους όρους της Apache License, μιας ελεύθερης άδειας λογισμικού. Το λογότυπο για το λειτουργικό σύστημα Android είναι ένα ρομπότ σε χρώμα πράσινου μήλου και σχεδιάστηκε από τη γραφίστρια Ιρίνα Μπλοκ. Η τελευταία έκδοση είναι η Android 10.0 Q, ενώ η εφαρμογή αναπτύχθηκε ώστε να είναι πλήρως συμβατή με την έκδοση Android 9.0 Pie.

2.2 Android Studio

2.2.1 Εισαγωγή

Το Android Studio είναι το επίσημο ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) για το λειτουργικό σύστημα Android, το οποίο βασίζεται στο λογισμικό IntelliJ IDEA της JetBrains. Είναι διαθέσιμο για λήψη στα λειτουργικά συστήματα Windows, macOS και Linux. Αποτελεί την αντικατάσταση του Eclipse Android Development Tools (ADT) το οποίο ήταν το πρωταρχικό IDE για την ανάπτυξη εφαρμογών Android.

Η κυκλοφορία του Android Studio ανακοινώθηκε στις 16 Μαΐου 2013 στο συνέδριο Google I/O. Ήταν σε στάδιο πρώιμης πρόσβασης (early access preview stage) ξεκινώντας από την έκδοση 0.1 τον Μάιο του 2013, στη συνέχεια μπήκε σε beta στάδιο στην έκδοση 0.8 που κυκλοφόρησε τον Ιούνιο του 2014. Η πρώτη σταθερή έκδοση κυκλοφόρησε τον Δεκέμβριο του 2014 (έκδοση 1.0).

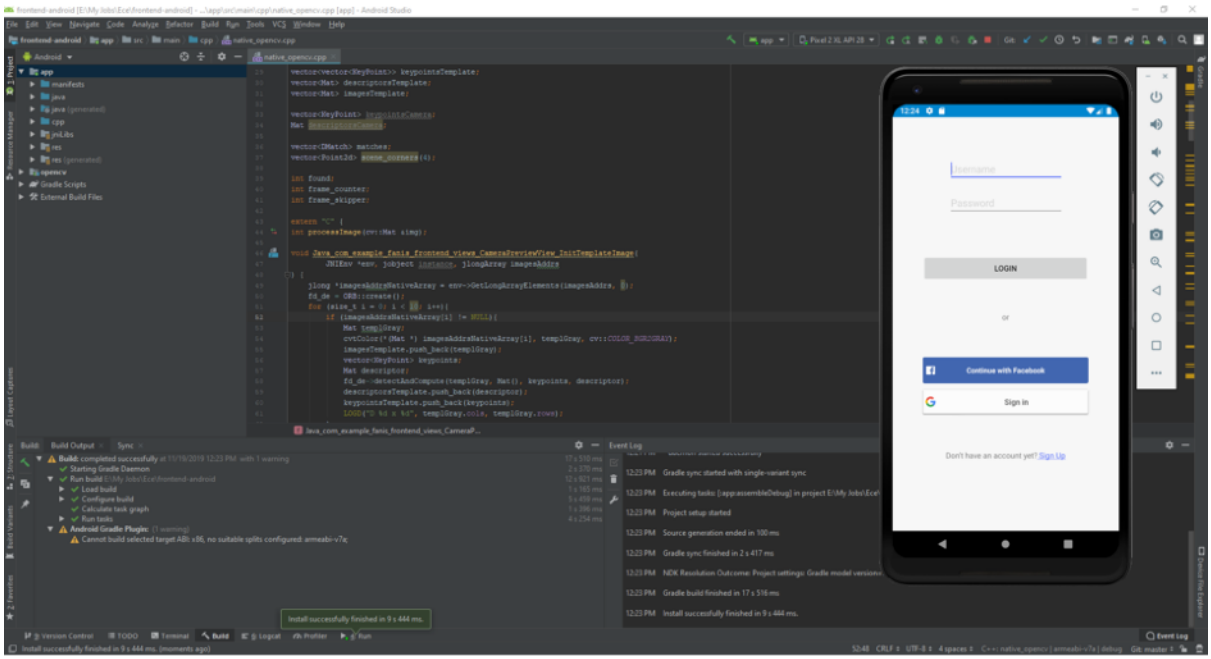
Από τις 7 Μαΐου 2019, η προτιμώμενη γλώσσα προγραμματισμού της Google για την ανάπτυξη εφαρμογών Android είναι η Kotlin. Παρόλα αυτά, το Android Studio υποστηρίζει κι άλλες γλώσσες.

2.2.2 Δυνατότητες

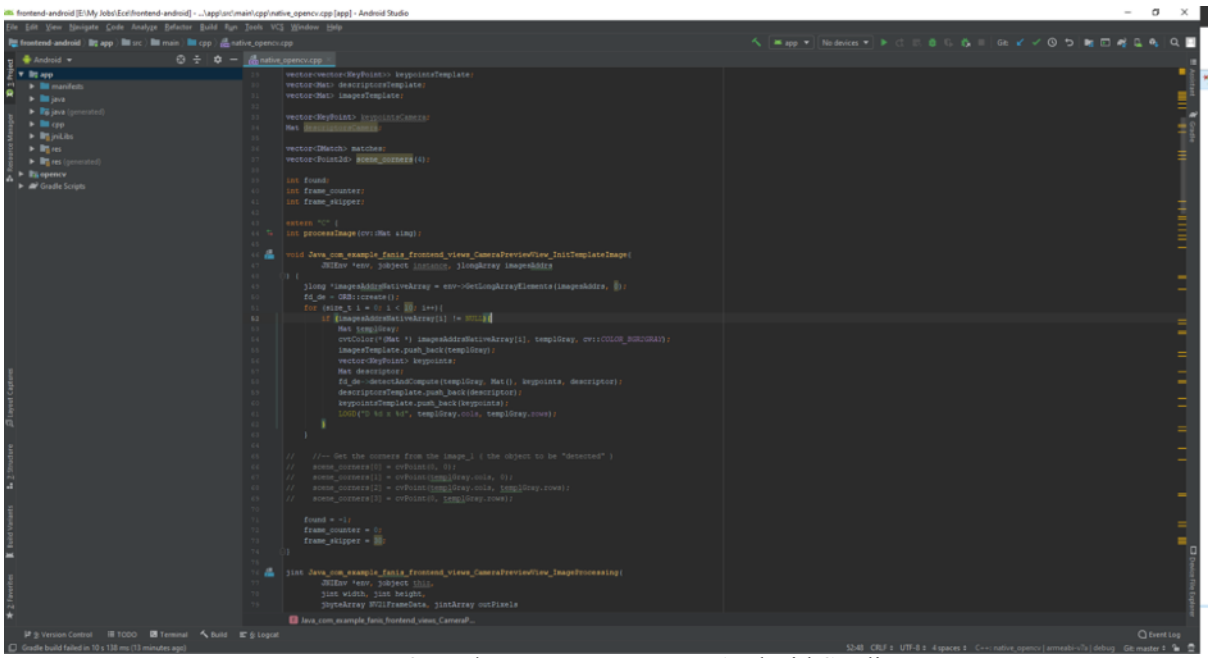
Οι ακόλουθες λειτουργίες παρέχονται στην τρέχουσα σταθερή έκδοση:

- Υποστήριξη αυτοματοποιημένης μεταγλώττισης με βάση τον Gradle
- Επανασχεδιασμός και γρήγορες επιδιορθώσεις κώδικα
- Εργαλεία Lint ώστε να εντοπιστούν προβλήματα χρηστικότητας, απόδοσης και συμβατότητας
- Ενσωμάτωση του ProGuard και δυνατότητες παροχής πιστοποιητικών.
- Οδηγοί με βάση πρότυπα για τη δημιουργία κοινών σχεδίων και στοιχείων

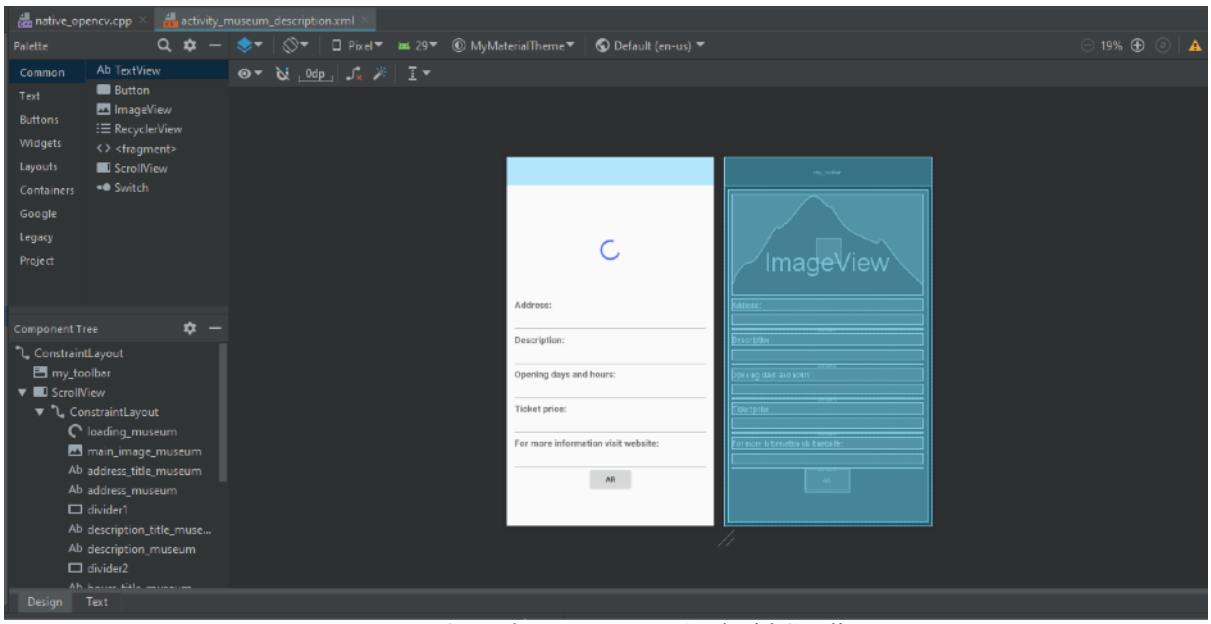
- Επεξεργαστής σχεδίων που επιτρέπει στους χρήστες να επεξεργάζονται το UI με δυνατότητα προεπισκόπησης του σε διαφορετικά μεγέθη οθονών
- Υποστήριξη για την ανάπτυξη εφαρμογών Android Wear.
- Ενσωματωμένη υποστήριξη για την πλατφόρμες της Google Cloud
- Εξομοιωτής (Emulator) για την εκτέλεση των εφαρμογών και τον εντοπισμό σφαλμάτων.



Εικόνα 1 Εξομοιωτής Android



Εικόνα 2 Επεξεργαστής κειμένου Android Studio



Εικόνα 3 Επεξεργαστής UI Android Studio

2.3 iOS

Το iOS (πρώην iPhone OS) είναι ένα λειτουργικό σύστημα για κινητά τηλέφωνα που δημιουργήθηκε και αναπτύχθηκε από την Apple αποκλειστικά για το υλικό της. Το χρησιμοποιούν πολλές από τις κινητές συσκευές της εταιρείας, συμπεριλαμβανομένων του iPhone και του iPod Touch. Ήταν επίσης το λειτουργικό του iPad πριν από την ανάπτυξη του iPadOS το 2019. Είναι το δεύτερο δημοφιλέστερο λειτουργικό σύστημα για κινητές συσκευές παγκοσμίως μετά το Android.

Αρχικά εκδόθηκε το 2007 για το iPhone, αναβαθμίστηκε τον Σεπτέμβριο του 2007 για να υποστηρίξει το iPod Touch και τον Ιανουάριο του 2019 για να υποστηρίξει το iPad. Από τον Μάρτιο του 2018, το App Store της Apple περιέχει πάνω από 2,1 εκατομμύρια εφαρμογές για το λειτουργικό iOS, εκ των οποίων 1 εκατομμύριο είναι συμβατές με το iPad. Αυτές οι εφαρμογές για κινητά έχουν εγκατασταθεί περισσότερες από 130 δισεκατομμύρια φορές.

Η διεπαφή χρήστη (User interface) του iOS βασίζεται στον άμεσο χειρισμό. Η αλληλεπίδραση με το λειτουργικό σύστημα περιλαμβάνει χειρονομίες, όπως το σύρσιμο του ενός δακτύλου στην οθόνη, το στιγμιαίο πάτημα και το σύρσιμο των δύο δαχτύλων είτε από το κέντρο της οθόνης προς τα άκρα της είτε από τα άκρα της προς

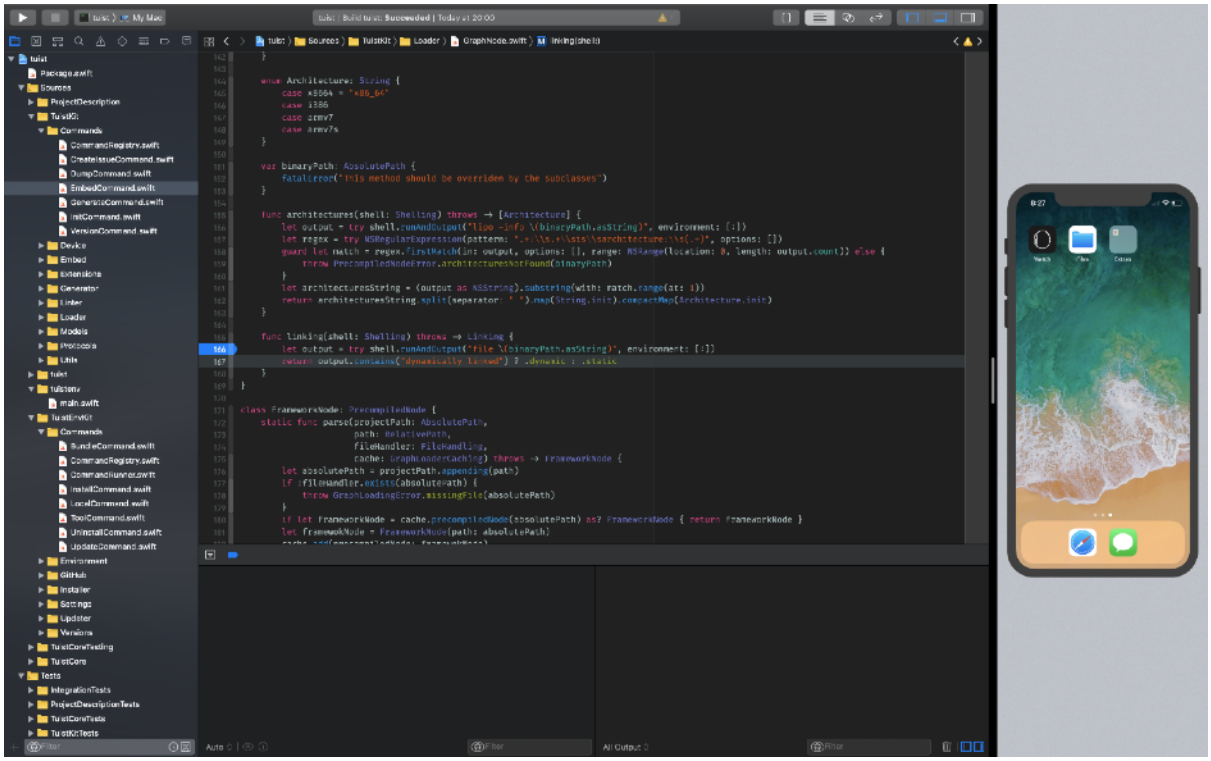
το κέντρο, τα οποία έχουν συγκεκριμένους λειτουργίες στο λειτουργικό σύστημα iOS. Το επιταχυνσιόμετρο χρησιμοποιείται από ορισμένες εφαρμογές για να ανταποκριθούν στην ανακίνηση της συσκευής (ένα κοινό αποτέλεσμα είναι η εντολή αναίρεσης) ή την περιστροφή της σε τρεις διαστάσεις (ένα κοινό αποτέλεσμα είναι η εναλλαγή μεταξύ κατακόρυφου και οριζόντιου τρόπου λειτουργίας). Η Apple έχει αναπτύξει σημαντικά την ενσωμάτωση λεπτομερών λειτουργιών προσβασιμότητας στο iOS, επιτρέποντας στους χρήστες με προβλήματα όρασης και ακοής να χρησιμοποιούν σωστά τα προϊόντα τους.

Νέες εκδόσεις του iOS κυκλοφορούν ετησίως. Σε όλες τις πρόσφατες συσκευές iOS, το iOS ελέγχει συχνά τη διαθεσιμότητα μιας ενημερωμένης έκδοσης και εάν είναι διαθέσιμη, θα ζητήσει από τον χρήστη να επιτρέψει την αυτόματη εγκατάστασή της. Η τρέχουσα έκδοση, το iOS 13, κυκλοφόρησε στο κοινό στις 19 Σεπτεμβρίου 2019, εισάγοντας στο λειτουργικό, αλλαγές στην διεπαφή χρήστη, σκοτεινή λειτουργία, ανασχεδιασμό της εφαρμογής Υπενθυμίσεις, πληκτρολόγηση με σύρσιμο του δακτύλου και βελτίωση της εφαρμογής Φωτογραφίες. Το iOS 13 δεν υποστηρίζει συσκευές με μνήμη RAM μικρότερη από 2 GB, συμπεριλαμβανομένων των iPhone 5, του iPod Touch (6ης γενιάς) και των iPhone 6 και iPhone 6 Plus, τα οποία εξακολουθούν να αποτελούν το 10% όλων των συσκευών iOS.

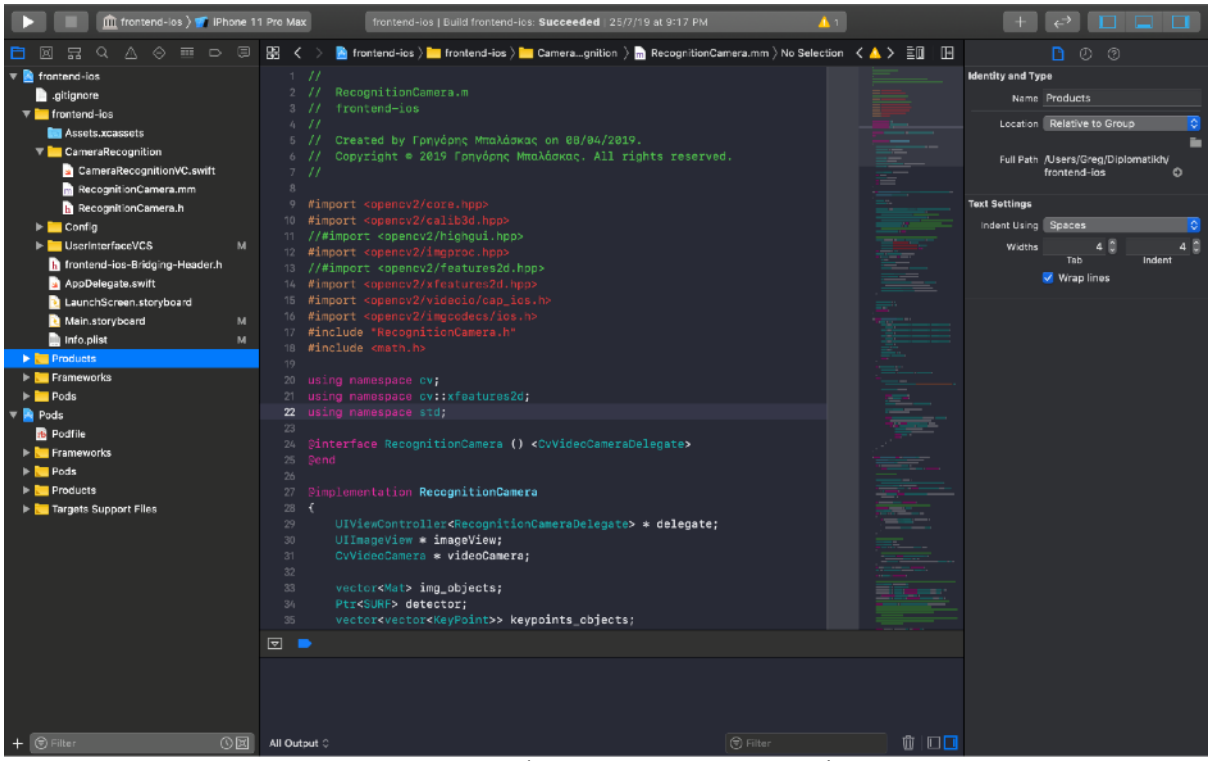
2.4 XCode

Το XCode είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) για macOS που περιέχει μια σειρά από εργαλεία που αναπτύχθηκαν από την Apple για την ανάπτυξη λογισμικού για τα λειτουργικά macOS, iOS, iPadOS, watchOS και tvOS. Κυκλοφόρησε για πρώτη φορά το 2003, η πιο πρόσφατη σταθερή έκδοση είναι η έκδοση 11.0 και διατίθεται δωρεάν μέσω του App Store για τους χρήστες του MacOS. Οι εγγεγραμμένοι προγραμματιστές μπορούν να πραγματοποιήσουν λήψη δοκιμαστικών και προγενέστερων εκδόσεων της σουίτας μέσω της ιστοσελίδας Apple Developer.

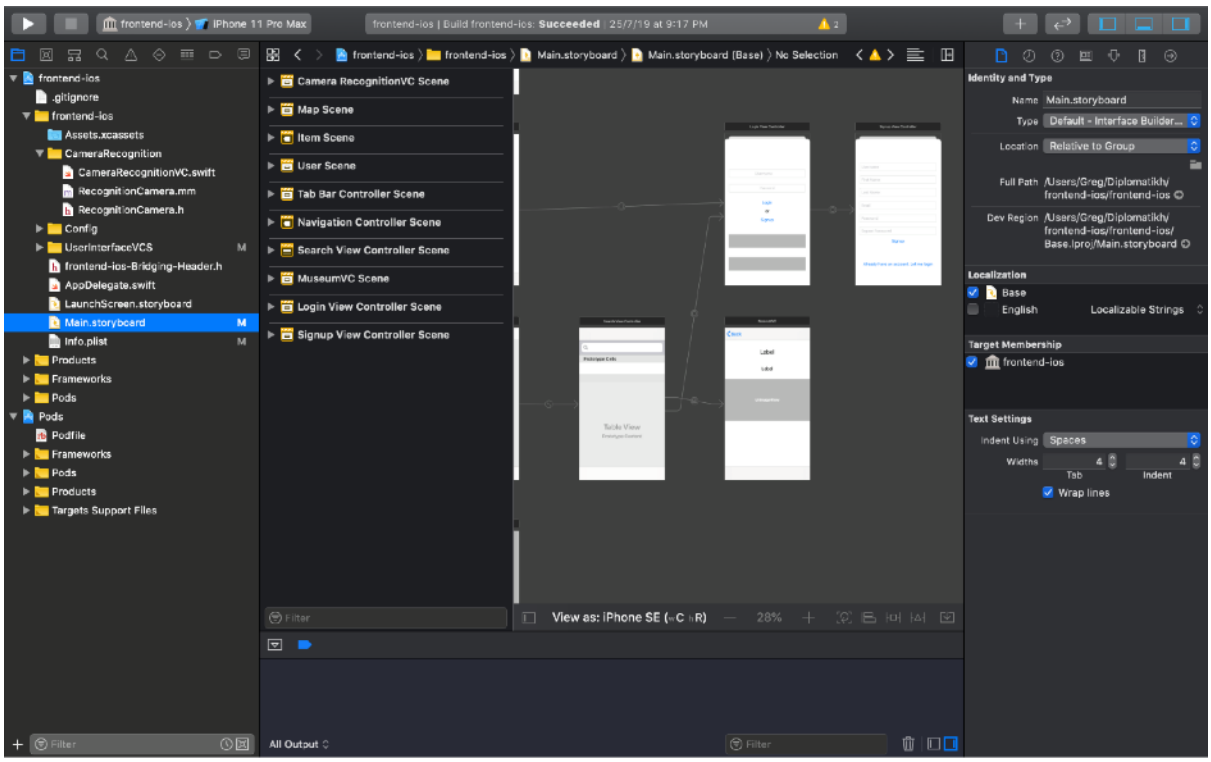
Το XCode υποστηρίζει τις γλώσσες προγραμματισμού C, C ++, Objective-C, Objective-C ++, Java, AppleScript, Python, Ruby, ResEdit και Swift με διάφορα μοντέλα προγραμματισμού.



Εικόνα 4 Εξομοιωτής Xcode



Εικόνα 5 Επεξεργαστής κειμένου Xcode



Εικόνα 6 Επεξεργαστής UI Xcode

2.5 Java

2.5.1 Εισαγωγή

Η Java είναι μια γλώσσα προγραμματισμού γενικής χρήσης που βασίζεται σε κλάσεις, είναι αντικειμενοστραφής και έχει σχεδιαστεί για να έχει λίγες εξαρτήσεις από βιβλιοθήκες. Σκοπός της είναι να επιτρέπει στους προγραμματιστές να γράφουν τον κώδικα μία φορά, και να τον τρέχουν οπουδήποτε (WORA, Write once, run anywhere), που σημαίνει ότι ο μεταγλωττισμένος κώδικας Java μπορεί να τρέξει σε όλες τις πλατφόρμες που την υποστηρίζουν χωρίς την ανάγκη να μεταγλωττιστεί ξανά. Οι εφαρμογές Java συνήθως μεταγλωττίζονται σε bytecode (σύνολο εντολών εικονικής μηχανής) που μπορούν να τρέξουν σε οποιαδήποτε εικονική μηχανή Java (JVM), ανεξάρτητα από την αρχιτεκτονική του κάθε υπολογιστή. Η σύνταξη της Java είναι παρόμοια με αυτή της C και της C ++, αλλά έχει λιγότερες λειτουργίες χαμηλού επιπέδου και από τις 2. Το 2019, η Java ήταν μια από τις δημοφιλέστερες γλώσσες προγραμματισμού που χρησιμοποιήθηκαν, ιδιαίτερα για εφαρμογές ιστού από την πλευρά του εξυπηρετητή, με 9 εκατομμύρια προγραμματιστές.

2.5.2 Ιστορική Αναδρομή

Η Java αναπτύχθηκε αρχικά από τον James Gosling στην Sun Microsystems (η οποία από τότε έχει αποκτηθεί από την Oracle) και κυκλοφόρησε το 1995 ως βασικό τμήμα της πλατφόρμας Java της Sun Microsystems. Οι αρχικοί μεταγλωττιστές Java, οι εικονικές μηχανές και οι βιβλιοθήκες υλοποιήθηκαν αρχικά από την Sun υπό άδειες ιδιοκτησίας. Από τον Μάιο του 2007 η Sun έχει ανανεώσει τις άδειες για τις περισσότερες από τις τεχνολογίες Java υπό την Γενική Άδεια Δημόσιας Χρήσης GNU. Παράλληλα, άλλοι προγραμματιστές έχουν αναπτύξει εναλλακτικές εφαρμογές αυτών των τεχνολογιών της Sun, όπως το GNU Compiler για Java (μεταγλωττιστής bytecode), το GNU Classpath (βιβλιοθήκες) και το IcedTea-Web (πρόσθετο προγράμματος περιήγησης).

Οι τελευταίες εκδόσεις είναι η Java 13, η οποία κυκλοφόρησε τον Σεπτέμβριο του 2019 και η Java 11, που υποστηρίζεται επί του παρόντος (LTS), που κυκλοφόρησε στις 25 Σεπτεμβρίου 2018. Η Oracle κυκλοφόρησε για την Java 8 LTS την τελευταία δωρεάν δημόσια ενημέρωση τον Ιανουάριο του 2019 για εμπορική χρήση, ενώ θα εξακολουθεί να υποστηρίζει την Java 8 με δημόσιες ενημερώσεις για προσωπική χρήση τουλάχιστον μέχρι το Δεκέμβριο του 2020.

2.6 Spring Framework

2.6.1 Εισαγωγή

Το Spring Framework είναι ένα λογισμικό ανοιχτού κώδικα για την πλατφόρμα της Java. Τα βασικά χαρακτηριστικά του μπορούν να χρησιμοποιηθούν από οποιαδήποτε εφαρμογή Java, αλλά χρησιμοποιείται κυρίως για υπηρεσίες web.

2.6.2 Ιστορική Αναδρομή

Η πρώτη έκδοση γράφτηκε από τον Rod Johnson, ο οποίος κυκλοφόρησε το λογισμικό με τη δημοσίευση του βιβλίου Expert One-on-One J2EE Design and Development τον Οκτώβριο του 2002. Κυκλοφόρησε για πρώτη φορά τον Ιούνιο του 2003. Η πρώτη πλήρης έκδοση 1.0 κυκλοφόρησε το Μάρτιο του 2004 με επόμενες εκδόσεις τον Σεπτέμβριο του 2004 και τον Μάρτιο του 2005. Η έκδοση 2.0 κυκλοφόρησε τον Οκτώβριο του 2006, η έκδοση 2.5 το Νοέμβριο του 2007, η έκδοση 3.0 το Δεκέμβριο του 2009, η έκδοση 3.1 το Δεκέμβριο του 2011 και η έκδοση 3.2.5 τον Νοέμβριο του 2013. Το Spring Framework 4.0 κυκλοφόρησε τον Δεκέμβριο του 2013. Το Spring Framework 4.2.0 κυκλοφόρησε στις 31 Ιουλίου 2015 και αναβαθμίστηκε αμέσως στην έκδοση 4.2.1, η οποία κυκλοφόρησε στις 01 Σεπτεμβρίου 2015. Είναι συμβατό με Java 6, 7 και 8, με έμφαση στις βελτιώσεις του πυρήνα και τις σύγχρονες δυνατότητες ιστού. Η έκδοση 4.3, η οποία είναι και αυτή που χρησιμοποιήθηκε στην εφαρμογή, κυκλοφόρησε στις 10 Ιουνίου 2016 και θα υποστηριχθεί μέχρι το 2020.

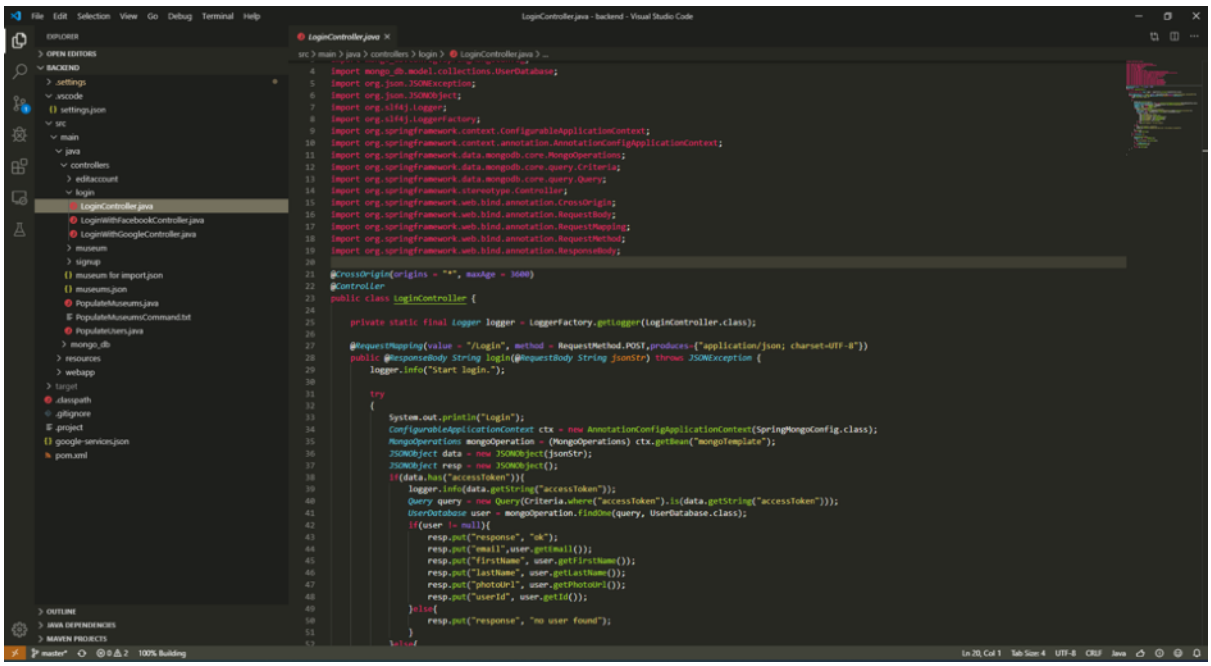
2.7 Visual Studio Code

Για την ανάπτυξη του backend σε Java χρησιμοποιήθηκε το περιβάλλον προγραμματισμού Visual Studio Code. Αναπτύχθηκε από τη Microsoft για Windows, Linux και macOS και περιλαμβάνει υποστήριξη για εντοπισμό σφαλμάτων, ενσωματωμένο έλεγχο Git, επισήμανση σύνταξης, έξυπνη ολοκλήρωση κώδικα και αυτόματη στοίχιση κώδικα. Είναι σε πολύ μεγάλο βαθμό παραμετροποιήσιμος, επιτρέποντας στους χρήστες να αλλάζουν το θέματα, να δημιουργούν συντομεύσεις πληκτρολογίου, προτιμήσεις και να εγκαθιστούν επεκτάσεις που προσθέτουν επιπλέον λειτουργίες. Ο πηγαίος κώδικας είναι ελεύθερος και ανοιχτός και κυκλοφορεί υπό την άδεια MIT. Τα εκτελέσιμο αρχείο είναι δωρεάν για ιδιωτική ή εμπορική χρήση. Το Visual Studio Code βασίζεται στο λογισμικό Electron, το οποίο χρησιμοποιείται για την ανάπτυξη εφαρμογών Node.js. Αν και χρησιμοποιεί το πλαίσιο Electron Σε έρευνα που διεξήχθη (Stack Overflow 2019 Developer Survey),

το Visual Studio Code κατατάχθηκε το πιο δημοφιλές περιβάλλον προγραμματισμού, με το 50,7% των 87.317 ερωτηθέντων να ισχυρίζονται ότι το χρησιμοποιούν.

Το Visual Studio Code μπορεί να χρησιμοποιηθεί με διάφορες γλώσσες προγραμματισμού. Αντί ενός συστήματος έργων επιτρέπει στους χρήστες να ανοίγουν έναν ή περισσότερους φακέλους, οι οποίοι στη συνέχεια μπορούν να αποθηκευτούν για μελλοντική χρήση. Αυτό του επιτρέπει να λειτουργεί ως επεξεργαστής κώδικα για οποιαδήποτε γλώσσα, σε αντίθεση με το Microsoft Visual Studio το οποίο χρησιμοποιεί το αποκλειστικό αρχείο με επέκταση .sln ως αρχείο έργου. Υποστηρίζει πολλές γλώσσες προγραμματισμού και ένα σύνολο χαρακτηριστικών που διαφέρουν ανά γλώσσα. Τα ανεπιθύμητα αρχεία και φάκελοι μπορούν να εξαιρεθούν από το δέντρο του έργου μέσω των ρυθμίσεων. Πολλές από τις λειτουργίες του Visual Studio Code δεν υπάρχουν στο μενού ή στο περιβάλλοντος χρήστη, αλλά μπορούν να χρησιμοποιηθούν μέσω μιας ειδικής παλέτας εντολών.

Επίσης το Visual Studio Code Μπορεί να επεκταθεί μέσω πρόσθετων προγραμμάτων (plugins), που διατίθενται μέσω της ίδιας της εφαρμογής. Οι προσθήκες μπορεί να αφορούν νέες λειτουργίες στον επεξεργαστή κειμένου ή υποστήριξη κάποιας γλώσσας ή κάποιας βιβλιοθήκης. Ακόμη, περιλαμβάνει πολλαπλές επεκτάσεις για FTP, επιτρέποντας στο λογισμικό να χρησιμοποιηθεί ως ελεύθερη εναλλακτική λύση για την ανάπτυξη εφαρμογών ιστού. Ο κώδικας μπορεί να συγχρονιστεί μεταξύ του επεξεργαστή και του διακομιστή, χωρίς να απαιτείται η εγκατάσταση επιπλέον λογισμικού. Τέλος, επιτρέπει στους χρήστες να ορίσουν τη κωδικοποίηση, τον χαρακτήρα νέας γραμμής και τη γλώσσα προγραμματισμού του ενεργού εγγράφου. Αυτό επιτρέπει να χρησιμοποιείται σε οποιαδήποτε πλατφόρμα με οποιαδήποτε γλώσσα προγραμματισμού σε οποιαδήποτε χώρα.



Εικόνα 7 Visual Studio Code

2.8 MongoDB

Η MongoDB είναι ένα μια διαπλατφορμική βάση δεδομένων που βασίζεται σε έγγραφα. Είναι στην κατηγορία των NoSQL βάσεων δεδομένων και χρησιμοποιεί έγγραφα τύπου JSON για την αποθήκευση των δεδομένων. Αναπτύχθηκε από την MongoDB Inc. και παραχωρήθηκε άδεια βάσει της δημόσιας άδειας διακομιστή (SSPL). Άρχισε να αναπτύσσεται από το 2007 από την εταιρία λογισμικού 10gen η οποία το 2013 άλλαξε το όνομά της σε MongoDB Inc. Στις 20 Οκτωβρίου 2017, η MongoDB εισήχθη στο χρηματιστήριο στον NASDAQ ως MDB με τιμή 24$ ανά μετοχή.

2.8.1 Χαρακτηριστικά

2.8.1.1 Ειδικά ερωτήματα (Ad hoc queries)

Η MongoDB υποστηρίζει αναζητήσεις σε πεδίων, εύρη και με την χρήση κανονικών εκφράσεων. Τα ερωτήματα μπορούν να επιστρέψουν συγκεκριμένα πεδία των εγγράφων και επίσης να περιλαμβάνουν συναρτήσεις JavaScript που ορίζονται από το

χρήστη. Τα ερωτήματα μπορούν επίσης να ρυθμιστούν ώστε να επιστρέφουν ένα τυχαίο δείγμα αποτελεσμάτων ενός δεδομένου μεγέθους.

2.8.1.2 C (Indexing)

Τα πεδία σε ένα έγγραφο MongoDB μπορούν να ευρετηριαστούν με πρωτογενείς και δευτερεύοντες δείκτες.

2.8.1.3 Αντιγραφή (Replication)

Η MongoDB μπορεί να παρέχει υψηλή διαθεσιμότητα δημιουργώντας σύνολα αντιγράφων της βάσης. Ένα σύνολο αντιγράφων αποτελείται από δύο ή περισσότερα αντίγραφα των δεδομένων. Κάθε μέλος του συνόλου μπορεί να ενεργεί στο ρόλο του κύριου ή του δευτερογενούς αντιγράφου ανά πάσα στιγμή. Όλες οι εγγραφές και οι αναγνώσεις γίνονται στο κύριο αντίγραφο από προεπιλογή. Τα δευτερεύοντα αντίγραφα διατηρούν ένα αντίγραφο των δεδομένων του κύριου χρησιμοποιώντας τις ενσωματωμένες λειτουργίες της MongoDB. Όταν ένα κύριο αντίγραφο αποτύχει, ένα από τα υπόλοιπα δευτερεύοντα αντίγραφα γίνεται κύριο. Τα δευτερεύοντα τμήματα μπορούν προαιρετικά να παρέχουν λειτουργίες ανάγνωσης.

2.8.1.4 Εξισορρόπηση φορτίου (Load balancing)

Η αρχιτεκτονική της MongoDB βασίζεται στα θραύσματα (shards). Ο χρήστης επιλέγει ένα κλειδί, το οποίο καθορίζει τον τρόπο κατανομής των δεδομένων μιας συλλογής. Τα δεδομένα χωρίζονται σε εύρη τιμών (με βάση το κλειδί shard) και κατανέμονται σε πολλαπλά θραύσματα. Η MongoDB, έτσι, μπορεί να τρέξει σε πολλούς διακομιστές, εξισορροπώντας το φορτίο και να αντιγράψει τα δεδομένα για να διατηρήσει το σύστημα σε λειτουργία και σε περίπτωση αστοχίας υλικού.

2.8.1.5 Αποθήκευση αρχείων (File storage)

Η MongoDB μπορεί να χρησιμοποιηθεί ως σύστημα αρχείων, το οποίο ονομάζεται GridFS, με δυνατότητες εξισορρόπησης φορτίου και αντιγραφής δεδομένων σε πολλαπλές μηχανές αποθήκευσης αρχείων. Αυτή η λειτουργία, που ονομάζεται σύστημα αρχείων πλέγματος, περιλαμβάνεται στα προγράμματα οδήγησης της MongoDB. Η MongoDB παρέχει τις συναρτήσεις επεξεργασίας αρχείων και περιεχομένου σε προγραμματιστές.

2.8.1.6 Συσσωμάτωση (Aggregation)

Η συσσωμάτωση επιτρέπει στους χρήστες να έχουν αποτελέσματα για τα οποία στις SQL βάσεις χρησιμοποιείται η έκφραση GROUP BY. Επίσης, επιτρέπει την ένωση εγγράφων από πολλαπλές συλλογές, καθώς και στατιστικούς υπολογισμούς όπως η τυπική απόκλιση

2.8.1.7 Εκτέλεση JavaScript στην πλευρά του διακομιστή (Server-side JavaScript execution)

Η γλώσσα προγραμματισμού JavaScript μπορεί να χρησιμοποιηθεί σε ερωτήματα, συναρτήσεις συσσωμάτωσης και να αποσταλεί απευθείας στη βάση δεδομένων ώστε να εκτελεστεί.

2.8.1.8 Συλλογές με όριο (Capped collections)

Η MongoDB υποστηρίζει συλλογές σταθερού μεγέθους που ονομάζονται συλλογές με όριο. Αυτός ο τύπος συλλογής διατηρεί την σειρά εισαγωγής και, μόλις επιτευχθεί το καθορισμένο μέγεθος, συμπεριφέρεται σαν μια κυκλική ουρά.

2.8.1.9 Συναλλαγές (Transactions)

Τον Ιούνιο του 2018 προστέθηκε στην MongoDB έκδοση 4.0 υποστήριξη για συναλλαγές ACID πολλαπλών εγγράφων

2.9 Κατανόηση των χαρακτηριστικών (Features)

2.9.1 Ορισμός

Στην όραση υπολογιστών και την επεξεργασία εικόνας, ένα χαρακτηριστικό είναι ένα κομμάτι πληροφορίας, η οποία είναι σημαντική για την επίλυση της υπολογιστικής εργασίας που σχετίζεται με μια συγκεκριμένη εφαρμογή. Χαρακτηριστικά μπορεί να είναι ειδικές δομές στην εικόνα, όπως σημεία, άκρα ή αντικείμενα. Τα χαρακτηριστικά μπορεί επίσης να είναι αποτέλεσμα μιας γενικής λειτουργίας γειτονιάς ή ανίχνευσης χαρακτηριστικών που εφαρμόζεται στην εικόνα. Άλλα παραδείγματα χαρακτηριστικών σχετίζονται με την κίνηση σε ακολουθίες εικόνων, με σχήματα που ορίζονται με όρους καμπύλων ή ορίων μεταξύ διαφορετικών περιοχών εικόνας ή με ιδιότητες μιας τέτοιας περιοχής. Η ιδέα του χαρακτηριστικού είναι πολύ γενική και ο τρόπος επιλογής των χαρακτηριστικών σε ένα συγκεκριμένο πρόβλημα μπορεί να εξαρτάται σε μεγάλο βαθμό από το ίδιο το πρόβλημα.

Δεδομένου ότι τα χαρακτηριστικά χρησιμοποιούνται ως σημείο εκκίνησης πολλών αλγορίθμων υπολογιστικής όρασης (computer vision), ο γενικός αλγόριθμος θα είναι συχνά τόσο καλός όσο ο ανιχνευτής χαρακτηριστικών (feature detector). Συνεπώς, η επιθυμητή ιδιότητα για έναν ανιχνευτή χαρακτηριστικών είναι επαναληψιμότητα, δηλαδή αν το ίδιο χαρακτηριστικό θα ανιχνευθεί ή όχι σε δύο ή περισσότερες διαφορετικές εικόνες της ίδιας σκηνής.

Η ανίχνευση χαρακτηριστικών (feature detection) είναι μια λειτουργία επεξεργασίας εικόνας χαμηλού επιπέδου. Δηλαδή, εκτελείται συνήθως ως η πρώτη ενέργεια σε μια εικόνα και εξετάζεται κάθε εικονοστοιχείο (pixel) ή κάθε μικρή περιοχή της εικόνας για να βρεθούν όλα τα χαρακτηριστικά.

Περιστασιακά, όταν η ανίχνευση χαρακτηριστικών είναι υπολογιστικά δαπανηρή και υπάρχουν χρονικοί περιορισμοί, μπορεί να χρησιμοποιηθεί ένας αλγόριθμος υψηλότερου επιπέδου για την καθοδήγηση της ανίχνευσης χαρακτηριστικών, έτσι ώστε να αναζητούνται μόνο ορισμένα τμήματα της εικόνας.

Υπάρχουν πολλοί αλγόριθμοι ηλεκτρονικής όρασης που χρησιμοποιούν την ανίχνευση χαρακτηριστικών ως αρχικό βήμα, με αποτέλεσμα να έχει αναπτυχθεί ένας πολύ μεγάλος αριθμός ανιχνευτών χαρακτηριστικών. Αυτά ποικίλουν ευρέως στα είδη των ανιχνευθέντων χαρακτηριστικών, στην υπολογιστική πολυπλοκότητα και στην επαναληψιμότητα.

2.9.2 Είδη των χαρακτηριστικών

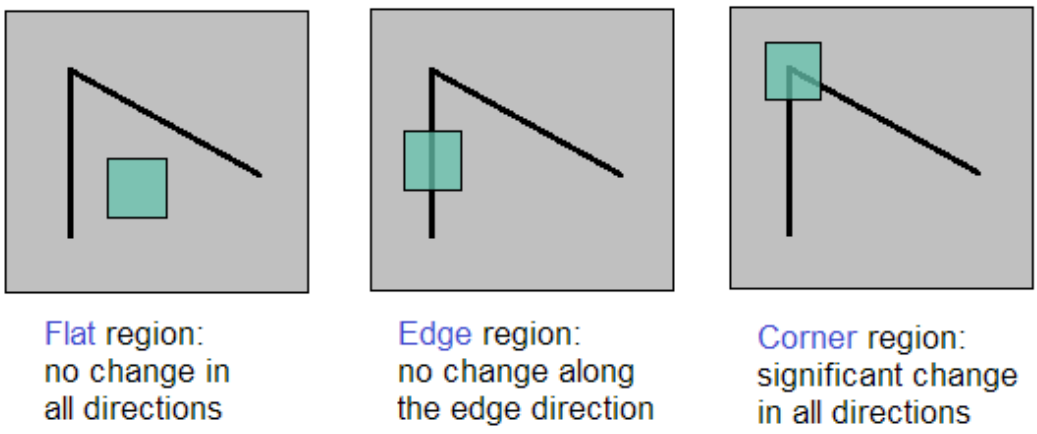
2.9.2.1 Ακμές (Edges)

Οι ακμές είναι σημεία όπου υπάρχει ένα όριο μεταξύ δύο περιοχών της εικόνας. Γενικά, μια ακμή μπορεί να έχει σχεδόν αυθαίρετο σχήμα και μπορεί να περιλαμβάνει κόμβους. Στην πράξη, οι ακμές συνήθως ορίζονται ως σύνολα σημείων στην εικόνα τα οποία έχουν ένα ισχυρό μέγεθος κλίσης (gradient magnitude). Επιπλέον, ορισμένοι συνηθισμένοι αλγόριθμοι θα συνδέσουν μαζί τα σημεία αυτά για να σχηματίσουν μια πληρέστερη περιγραφή μιας ως. Αυτοί οι αλγόριθμοι συνήθως θέτουν ορισμένους περιορισμούς στις ιδιότητες μιας ακμής, όπως το σχήμα, η ομαλότητα και η τιμή της κλίσης (gradient value). Τοπικά, οι ακμές έχουν μονοδιάστατη δομή.

2.9.2.2 Γωνίες / σημεία ενδιαφέροντος (Corners / interest points)

Οι όροι γωνίες και σημεία ενδιαφέροντος χρησιμοποιούνται κάπως αλληλεξαρτώμενοι και αναφέρονται σε χαρακτηριστικά που μοιάζουν με σημεία σε μια εικόνα, τα οποία έχουν μια τοπική δισδιάστατη δομή. Το όνομα «γωνία» προέκυψε από τους αρχικούς αλγόριθμους που πραγματοποιούσαν ανίχνευση άκρων (edge detection) και στη συνέχεια ανέλυαν τις ακμές για να βρουν απότομες αλλαγές

στην κατεύθυνση (γωνίες). Αυτοί οι αλγόριθμοι αναπτύχθηκαν έτσι ώστε να μην απαιτείται πλέον η σαφής ανίχνευση άκρων. Παρατηρήθηκε τότε ότι οι λεγόμενες γωνίες εντοπίζονταν επίσης σε τμήματα της εικόνας που δεν ήταν γωνίες με την παραδοσιακή έννοια (για παράδειγμα μπορεί να ανιχνευόταν ένα μικρό φωτεινό σημείο σε σκούρο φόντο). Αυτά τα σημεία είναι συχνά γνωστά ως σημεία ενδιαφέροντος, αλλά ο όρος "γωνία" χρησιμοποιείται ως σύμβαση.



2.9.2.3 Σφαιρίδια / περιοχές ενδιαφέροντος (Blobs / regions of interest points)

Τα σφαιρίδια παρέχουν μια συμπληρωματική περιγραφή των δομών μιας εικόνας ως περιοχές, σε αντίθεση με τις γωνίες που είναι περισσότερο σαν σημεία. Παρόλα αυτά, οι περιγραφείς κηλίδων (descriptors) μπορεί συχνά να περιέχουν ένα προτιμώμενο σημείο (κάποιο τοπικό μέγιστο ή ένα κέντρο βάρους), που σημαίνει ότι πολλοί ανιχνευτές σφαιριδίων (blob detectors) μπορούν επίσης να θεωρηθούν σημεία ενδιαφέροντος. Οι ανιχνευτές σφαιριδίων μπορούν να ανιχνεύσουν περιοχές μιας εικόνας που είναι πολύ ομαλές ώστε να ανιχνευθούν από έναν ανιχνευτή γωνιών (corner detector). Αν θεωρήσουμε ότι συρρικνώνουμε μια εικόνα και στην συνέχεια ανιχνεύουμε τις γωνίες της, ο ανιχνευτής θα ανταποκρίνεται σε σημεία που είναι αιχμηρά (sharp) στη συρρικνωμένη εικόνα, αλλά μπορεί να είναι ομαλά (smooth) στην αρχική εικόνα. Αυτή είναι και η διαφορά μεταξύ ενός ανιχνευτή γωνιών και ενός ανιχνευτή σφαιριδίων. Ο ανιχνευτής σφαιριδίων θα βρει τα ίδια χαρακτηριστικά ανεξάρτητα με την κλίμακα της εικόνας.

2.9.2.4 Κορυφογραμμές (Ridges)

Για επιμήκη αντικείμενα, η έννοια των κορυφογραμμών είναι ένα φυσικό εργαλείο. Από πρακτική άποψη, μια κορυφογραμμή μπορεί να θεωρηθεί ως μια μονοδιάστατη καμπύλη που αντιπροσωπεύει έναν άξονα συμμετρίας και επιπλέον έχει ένα χαρακτηριστικό τοπικού πλάτους που συνδέεται με κάθε σημείο της κορυφογραμμής. Δυστυχώς, όμως, είναι πιο δύσκολο από έναν αλγόριθμο να εξαχθούν χαρακτηριστικά κορυφογραμμής (ridge features) από γενικές κατηγορίες εικόνων γκρίζου επιπέδου συγκριτικά με τα χαρακτηριστικά ακμής, γωνίας ή σφαιριδίου. Παρ' όλα αυτά, οι περιγραφείς κορυφογραμμών (ridge detectors) χρησιμοποιούνται συχνά για την ανίχνευση δρόμων σε αεροφωτογραφίες και αιμοφόρων αγγείων σε ιατρικές εικόνες.

2.9.2.5 Εξαγωγή Χαρακτηριστικών (Feature extraction)

Μόλις ανιχνευτούν τα χαρακτηριστικά, μπορεί να εξαχθεί αναπαράσταση αυτών των χαρακτηριστικών της εικόνας. Αυτή η αναπαράσταση απαιτεί αρκετούς υπολογιστικούς. Το αποτέλεσμα είναι γνωστό ως ένας περιγραφέας χαρακτηριστικών (feature descriptor). ή ένα διάνυσμα χαρακτηριστικών (feature vector).

2.9.3 Βασικοί ανιχνευτές χαρακτηριστικών (Basic feature detectors)

2.9.3.1 Harris Corner Detector

Ο Harris Corner Detector είναι ένας ανιχνευτής γωνιών που χρησιμοποιείται συνήθως σε αλγόριθμους ηλεκτρονικής όρασης. Εμφανίστηκε για πρώτη φορά από τον Chris Harris και τον Mike Stephens το 1988 και αποτελεί βελτίωση του ανιχνευτή γωνίας του Moravec. Σε σύγκριση με τον προηγούμενο, ο Harris Corner Detector λαμβάνει υπόψη τη διαφορά της γωνιακής βαθμολογίας σε σχέση με την κατεύθυνση απευθείας, αντί να χρησιμοποιεί μετατοπίσεις για κάθε γωνία 45 μοιρών. Αποδείχθηκε πιο ακριβής στη διάκριση μεταξύ των άκρων και γωνιών. Από τότε, έχει

βελτιωθεί και υιοθετηθεί σε πολλούς αλγόριθμους για την προ επεξεργασία εικόνων σε μεταγενέστερους αλγορίθμους.

Η ιδέα του αλγορίθμου είναι να εξετάσει ένα μικρό παράθυρο γύρω από κάθε εικονοστοιχείο σε μια εικόνα. Θέλουμε να βρούμε όλα αυτά τα παράθυρα εικονοστοιχείων που είναι μοναδικά. Η μοναδικότητα μπορεί να μετρηθεί με μια μικρή μετατόπιση κάθε παραθύρου σε μια δεδομένη κατεύθυνση και τη μέτρηση της αλλαγής που συμβαίνει στις τιμές των εικονοστοιχείων.

Λαμβάνουμε το άθροισμα της τετραγωνικής διαφοράς (SSD) των τιμών των εικονοστοιχείων πριν και μετά την μετατόπιση και βρίσκουμε τα παράθυρα όπου το SSD είναι μεγάλο για μετατοπίσεις και στις 8 κατευθύνσεις. Αν ορίσουμε τη συνάρτηση αλλαγής E (u, v) ως άθροισμα όλων των SSD, όπου u, v είναι οι συντεταγμένες x, y κάθε εικονοστοιχείο στο παράθυρό μας 3 x 3 και I είναι η τιμή έντασης του εικονοστοιχείο. Τα χαρακτηριστικά της εικόνας είναι όλα τα εικονοστοιχεία που έχουν μεγάλες τιμές του E (u, v), όπως ορίζεται από κάποιο όριο.

$$E(u,v) = \sum_{x,y} w(x,y)[I(x+u, y+v) - I(x,y)]^2$$

Πρέπει να μεγιστοποιήσουμε την συνάρτηση E (u, v) για να ανιχνευθούν οι γωνίες. Αυτό σημαίνει ότι πρέπει να μεγιστοποιήσουμε τον δεύτερο όρο. Εφαρμόζοντας την επέκταση Taylor στην παραπάνω εξίσωση και χρησιμοποιώντας μερικά μαθηματικά βήματα, παίρνουμε την τελική εξίσωση ως:

$$E(u,v) \approx [u \;\; v]\left(\sum \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \right) \begin{bmatrix} u \\ v \end{bmatrix}$$

Μετονομάζουμε το άθροισμα πινάκων ως M:

$$M = \sum w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Και η εξίσωση γίνεται:

$$E(u,v) \approx [u \;\; v] M \begin{bmatrix} u \\ v \end{bmatrix}$$

Θυμηθείτε ότι θέλουμε το SSD να είναι μεγάλο στις μετατοπίσεις και για τις οκτώ κατευθύνσεις, ή αντιστρόφως, το SSD να είναι μικρό για καμία από τις κατευθύνσεις. Επιλύοντας για τα ιδιοδιανύσματα του M, μπορούμε να βρούμε τις κατευθύνσεις τόσο για τις μεγαλύτερες όσο και για τις μικρότερες αυξήσεις του SSD. Οι αντίστοιχες ιδιοτιμές μας δίνουν την πραγματική τιμή αυτών των αυξήσεων. Μια βαθμολογία, R, υπολογίζεται για κάθε παράθυρο:

$$R = detM - k(trace\,M)^2$$
$$detM = \lambda_1\lambda_2$$
$$trace\,M = \lambda_1 + \lambda_2$$

Τα λ1 και λ2 είναι οι ιδιοτιμές του M. Έτσι οι τιμές αυτών των ιδιοτιμών αποφασίζουν εάν μια περιοχή είναι μια γωνία (corner), ακμή (edge) ή επίπεδη (flat).

- Όταν | R | είναι μικρό, κάτι που συμβαίνει όταν τα λ1 και λ2 είναι μικρά, η περιοχή είναι επίπεδη
- Όταν R <0, που συμβαίνει όταν λ1 >> λ2 ή αντίστροφα, η περιοχή είναι ακμή.
- Όταν το R είναι μεγάλο, το οποίο συμβαίνει όταν τα λ1 και λ2 είναι μεγάλα και λ1~λ2, η περιοχή είναι γωνία.

2.9.3.2 Shi-Tomasi Corner Detector

Ο Shi-Tomasi corner detector είναι σχεδόν ίδιος με τον Harris Corner detector. Η διαφορά τους είναι στον τρόπο που υπολογίζεται η βαθμολογία (R). Αυτό στις περισσότερες περιπτώσεις δίνει ένα καλύτερο αποτέλεσμα. Επιπλέον, με αυτή τη μέθοδο, μπορούμε να βρούμε τις καλύτερες N γωνίες, σε περίπτωση που δεν θέλουμε να ανιχνεύσουμε κάθε γωνία της εικόνας.

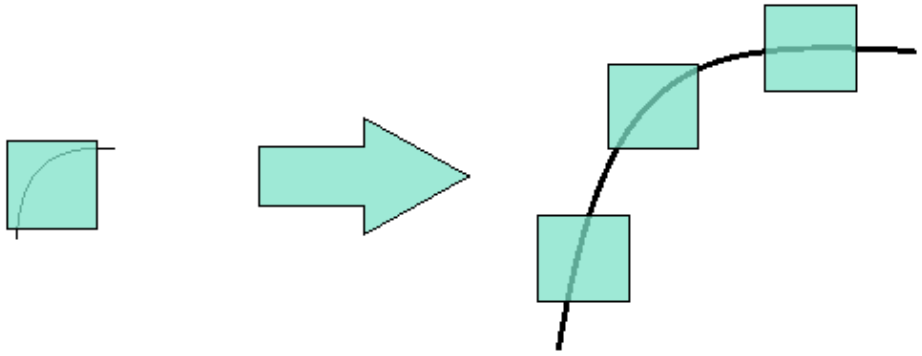
Το R υπολογίζεται ως εξής:

$$R = \min(\lambda_1, \lambda_2)$$

Αν το R είναι μεγαλύτερο από μια τιμή όριο τότε το σημείο προσδιορίζεται ως γωνία.

2.9.3.3 SIFT (Scale-Invariant Feature Transform)

Οι προηγούμενοι δύο ανιχνευτές που εξετάσαμε είναι ανεξάρτητοι της περιστροφής, που σημαίνει ότι ακόμα και αν η εικόνα περιστραφεί, μπορούμε να βρούμε τις ίδιες γωνίες. Αυτό είναι προφανές διότι οι γωνίες δεν αλλάζουν όταν περιστρέψουμε μια εικόνα. Όμως όταν μια εικόνα αλλάζει κλίμακα τότε μια γωνία μπορεί να πάψει να είναι γωνία. Στην παρακάτω εικόνα βλέπουμε ότι μια γωνία σε μια μικρή εικόνα μέσα σε ένα μικρό παράθυρο γίνεται επίπεδη όταν μεγεθύνεται στο ίδιο παράθυρο. Βλέπουμε λοιπόν ότι οι παραπάνω ανιχνευτές δεν είναι ανεξάρτητοι της κλίμακας της εικόνας.



Έτσι, το 2004, ο David Lowe, από το Πανεπιστήμιο British Columbia, εισήγαγε στην εργασία του ένα νέο αλγόριθμο τον SIFT (Scale Invariant Feature Transform).

Για να βρούμε σημεία ενδιαφέροντος ανεξάρτητα της κλίμακας βρίσκουμε το LoG (Laplasian of Gaussian) της εικόνας για διάφορες τιμές σ. Το LoG λειτουργεί σαν ανιχνευτής σφαιριδίων (blob detector) που εντοπίζει σφαιρίδια σε διάφορα μεγέθη λόγω της αλλαγής της τιμής του σ. Με λίγα λόγια το σ λειτουργεί ως παράμετρος της κλίμακας. Για παράδειγμα, στην παραπάνω εικόνα στην μικρή γωνία ταιριάζει χαμηλή τιμή του σ ενώ στην μεγάλη γωνία υψηλή τιμή. Έτσι μπορούμε να βρούμε τοπικά ακρότατα σε κάθε κλίμακα της εικόνας δηλαδή μια λίστα με τιμές της μορφής (x, y, σ) που σημαίνει ότι υπάρχει πιθανό σημείο ενδιαφέροντος (x, y) στην κλίμακα σ. Επειδή όμως το LoG είναι ακριβό υπολογιστικά, ο SIFT χρησιμοποιεί διαφορά

Gauss (DoG, Difference of Gaussian) η οποία ουσιαστικά είναι μια προσέγγιση του LoG.

Μόλις εντοπιστούν τα πιθανά σημεία ενδιαφέροντος πρέπει να επιτευχθούν καλύτερα αποτελέσματα. Με την χρήση των σειράς Taylor παίρνουμε μια ακριβέστερη θέση των ακροτάτων και αν η ένταση σε αυτό το σημείο είναι μικρότερη από μια οριακή τιμή, το σημείο απορρίπτεται.

Το DoG έχει υψηλότερη απόκριση στις ακμές επομένως και αυτές πρέπει να αφαιρεθούν. Για αυτό χρησιμοποιείται, όπως και στον Harris corner detector έναν 2x2 Hessian πίνακας για να υπολογιστεί η κύρια καμπυλότητα. Γνωρίζουμε ήδη, ότι για τις ακμές η μια ιδιοτιμή του πίνακα είναι μεγαλύτερη από την άλλη.

Στην συνέχεια προστίθεται μια κατεύθυνση σε κάθε σημείο κλειδί για να παραμείνουν τα σημεία αμετάβλητα κατά την περιστροφή της εικόνας. Μια γειτονιά λαμβάνεται γύρω από τη θέση του σημείου κλειδιού ανάλογα με την κλίμακα υπολογίζονται σε αυτή την περιοχή το μέγεθος και η κατεύθυνση της κλίσης (gradient). Δημιουργείται ιστόγραμμα με 36 bins που καλύπτουν 360 μοίρες. Ισοσταθμίζεται από το μέγεθος κλίσης (gradient magnitude) και κυκλικό παράθυρο με συντελεστή ισορροπίας Gauss με σ ίσο με 1,5 φορές την κλίμακα του σημείου κλειδιού. Η υψηλότερη κορυφή στο ιστόγραμμα λαμβάνεται και οποιαδήποτε κορυφή πάνω από το 80% της συμπεριλαμβάνεται επίσης στον υπολογισμό του προσανατολισμού. Έτσι δημιουργούνται σημεία κλειδιά με την ίδια θέση και κλίμακα, αλλά με διαφορετικές κατευθύνσεις, συμβάλλοντας στη σταθερότητα της αντιστοίχισης χαρακτηριστικών (feature matching).

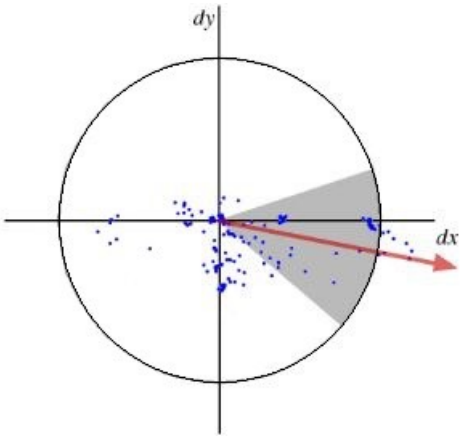
Έπειτα, δημιουργείται ο περιγραφέας σημείου κλειδιού (keypoint descriptor). Λαμβάνεται μια γειτονιά 16x16 γύρω από το σημείο κλειδί. Είναι χωρισμένο σε 16 υποομάδες μεγέθους 4x4. Για κάθε υποομάδα δημιουργείται ιστόγραμμα προσανατολισμού 8 bin. Επομένως, είναι διαθέσιμες συνολικά 128 τιμές bin. Παρουσιάζεται ως διάνυσμα για τον σχηματισμό του περιγραφέα σημείου κλειδιού. Επιπλέον, λαμβάνονται διάφορα μέτρα για την επίτευξη σταθερότητας απέναντι σε αλλαγές φωτισμού, περιστροφή κλπ.

2.9.3.4 SURF (Speeded-Up Robust Features)

Προηγουμένως, είδαμε τον αλγόριθμο SIFT για την ανίχνευση και περιγραφή σημείων κλειδιών. Όμως ο SIFT ήταν σχετικά αργός και δημιουργήθηκε η ανάγκη για μια ταχύτερη έκδοση. Το 2006, τρεις ερευνητές, οι Herbert Bay, Tinne Tuytelaars και Luc Van Gool δημοσίευσαν μια άλλη εργασία με τίτλο, "SURF (Speeded-Up Robust Features)", που παρουσίαζε ένα νέο αλγόριθμο που ονομαζόταν SURF. Όπως υποδηλώνει το όνομα, πρόκειται για μια ταχύτερη έκδοση του SIFT.

Στο SIFT, ο Lowe προσέγγισε το LoG (Laplasian of Gaussian) με το DoG (Difference of Gaussian) για εύρεση της κλίμακας. Το SURF πηγαίνει ένα βήμα ακόμα και προσεγγίζει το LoG με το Φίλτρο Κουτιού (Box Filter). Η παρακάτω εικόνα δείχνει μια τέτοια προσέγγιση. Ένα μεγάλο πλεονέκτημα αυτής της προσέγγισης είναι ότι η συνέλιξη με το φίλτρο κουτιού μπορεί εύκολα να υπολογιστεί με τη βοήθεια ολόκληρων εικόνων και μπορεί να γίνει παράλληλα για διαφορετικές κλίμακες. Επίσης, το SURF βασίζεται σε ένα Hessian πίνακα για την κλίμακα και την τοποθεσία.

Για να βρεθεί ο προσανατολισμός, το SURF χρησιμοποιεί αποκρίσεις μικρών κυμάτων σε οριζόντια και κάθετη κατεύθυνση για μια γειτονιά μεγέθους 6s. Επίσης, εφαρμόζονται βάρη Gauss. Στη συνέχεια, σχεδιάζονται σε ένα χώρο όπως παρουσιάζεται στην παρακάτω εικόνα. Ο κυρίαρχος προσανατολισμός εκτιμάται με τον υπολογισμό του αθροίσματος όλων των αποκρίσεων μέσα σε ένα παράθυρο γωνίας 60 μοιρών. Ενδιαφέρον είναι ότι, η απόκριση μπορεί να βρεθεί χρησιμοποιώντας ολόκληρες εικόνες πολύ εύκολα σε οποιαδήποτε κλίμακα. Για πολλές εφαρμογές, δεν απαιτείται μεταστροφή της περιστροφής, οπότε δεν χρειάζεται να βρεθεί αυτός ο προσανατολισμός, ο οποίος επιταχύνει τη διαδικασία. Το SURF παρέχει μια τέτοια λειτουργία που ονομάζεται Upright-SURF ή U-SURF. Βελτιώνει την ταχύτητα και είναι αποτελεσματική για περιστροφές μέχρι 15 μοίρες.



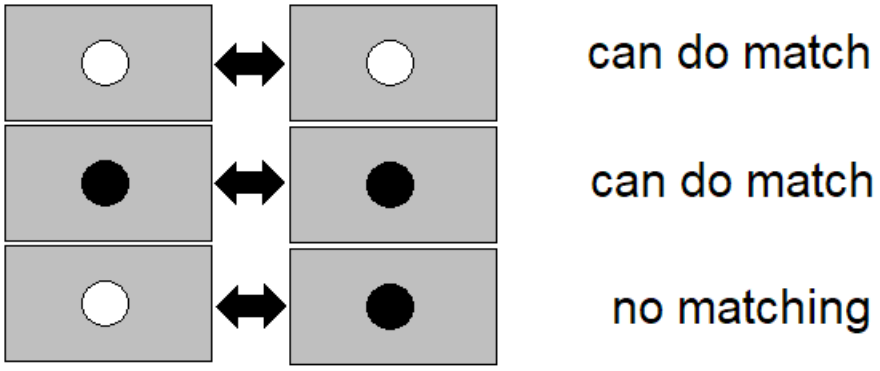
Για την περιγραφή χαρακτηριστικών, το SURF χρησιμοποιεί και πάλι αποκρίσεις μικρών κυμάτων σε οριζόντια και κάθετη. Μια γειτονιά μεγέθους 20sX20s λαμβάνεται γύρω από το σημείο κλειδί. Διαιρείται σε υποπεριοχές 4x4. Για κάθε υποπεριοχή, λαμβάνονται οριζόντιες και κάθετες αποκρίσεις κυμάτων και σχηματίζεται ένα διάνυσμα:

$$\mathrm{v} = \left(\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|, \right)$$

Αυτό όταν αντιπροσωπεύεται ως διάνυσμα δίνει τον περιγραφέα χαρακτηριστικών SURF με συνολικά 64 διαστάσεις. Μειώνοντας τις διαστάσεις, αυξάνεται η ταχύτητα του υπολογισμού και της αντιστοίχισης, αλλά μειώνεται η διακριτότητα των χαρακτηριστικών.

Για περισσότερη διακριτότητα, ο περιγραφέας χαρακτηριστικών SURF έχει μία έκδοση 128 διαστάσεων. Τα ποσά των dx και |dx| υπολογίζονται ξεχωριστά για dy < 0 και dy >= 0. Ομοίως, τα ποσά των dy και |dy| χωρίζονται ανάλογα με το πρόσημο του dx, διπλασιάζοντας έτσι τον αριθμό των χαρακτηριστικών. Δεν προσθέτει πολλή υπολογιστική πολυπλοκότητα.

Μια άλλη σημαντική βελτίωση είναι η χρήση του ίχνους του Hessian πίνακα για το υποκείμενο σημείο ενδιαφέροντος. Δεν προσθέτει κανένα υπολογιστικό κόστος, αφού έχει ήδη υπολογιστεί κατά τη διάρκεια της ανίχνευσης. Το ίχνος διακρίνει τις φωτεινές κηλίδες σε σκοτεινό υπόβαθρο από την αντίστροφη κατάσταση (δηλαδή σκοτεινές κηλίδες σε φωτεινό υπόβαθρο). Στο στάδιο αντιστοίχισης, συγκρίνουμε τα χαρακτηριστικά μόνο εάν έχουν τον ίδιο τύπο αντίθεσης (όπως φαίνεται στην παρακάτω εικόνα). Αυτές οι ελάχιστες πληροφορίες επιτρέπουν την ταχύτερη αντιστοίχιση, χωρίς να μειώνεται η απόδοση του περιγραφέα.



Συνοπτικά, το SURF προσθέτει πολλά χαρακτηριστικά για τη βελτίωση της ταχύτητας σε κάθε βήμα. Η ανάλυση δείχνει ότι είναι 3 φορές ταχύτερη από το SIFT, ενώ η απόδοση είναι συγκρίσιμη με το SIFT. Το SURF είναι καλό στο χειρισμό εικόνων με θόλωση και περιστροφή, αλλά δεν είναι καλό στο χειρισμό της αλλαγής οπτικής γωνίας και φωτισμού.

2.9.3.5 FAST (Features from Accelerated Segment Test)

Μέχρι στιγμής έχουμε εξετάσει αρκετούς ανιχνευτές χαρακτηριστικών και πολλοί από αυτούς είναι πολύ καλοί. Αλλά όταν μιλάμε για εφαρμογή σε πραγματικό χρόνο, δεν είναι αρκετά γρήγοροι. Παραδείγματος χάρη σε εφαρμογές SLAM (Simultaneous Localization and Mapping) σε μια κινητική συσκευή που έχουμε περιορισμένους υπολογιστικούς πόρους.

Ως λύση σε αυτό, ο αλγόριθμος FAST προτάθηκε από τους Edward Rosten και Tom Drummond στην εργασία τους "Machine learning for high-speed corner detection" το 2006 (Αργότερα αναθεωρήθηκε το 2010). Μια βασική περίληψη του αλγορίθμου παρουσιάζεται παρακάτω:

1. Επιλέξτε ένα εικονοστοιχείο p στην εικόνα που πρέπει να αναγνωριστεί ως σημείο ενδιαφέροντος ή όχι. Έστω Ip η έντασή του.
2. Επιλέξτε την κατάλληλη τιμή κατωφλίου t.

3. Υποθέστε έναν κύκλο 16 εικονοστοιχείων γύρω από το υπό δοκιμή εικονοστοιχείο

4. Το εικονοστοιχείο p είναι γωνία αν υπάρχει ένα σύνολο n συνεχόμενων εικονοστοιχείων στον κύκλο των 16 εικονοστοιχείων τα οποία είναι όλα φωτεινότερα από το Ip + t ή όλα πιο σκούρα από το Ip - t (Εμφανίζονται ως λευκές γραμμές παύλας στην παρακάτω εικόνα, το n εδώ επιλέχτηκε να είναι 12).

Έχει προταθεί μια δοκιμή μεγάλης ταχύτητας για τον αποκλεισμό ενός μεγάλου αριθμού μη γωνιών. Αυτή η δοκιμή εξετάζει μόνο τα τέσσερα εικονοστοιχεία, τα 1, 9, 5 και 13 (Πρώτα 1 και 9 ελέγχονται εάν είναι πολύ φωτεινότερα ή πιο σκούρα. Αν ναι, τότε ελέγχουν 5 και 13). Αν p είναι γωνία, τότε τουλάχιστον τρεις από αυτές πρέπει να είναι όλες φωτεινότερες από Ip + t ή πιο σκούρες από Ip - t. Εάν κανένα από αυτά δεν συμβαίνει, τότε το p δεν μπορεί να είναι μια γωνία. Το πλήρες κριτήριο του δοκιμαστικού τμήματος μπορεί στη συνέχεια να εφαρμοστεί στα εικονοστοιχεία που έχουν περάσει την παραπάνω δοκιμή εξετάζοντας όλα τα εικονοστοιχεία στον κύκλο τους.

Αυτός ο ανιχνευτής από μόνος του παρουσιάζει υψηλή απόδοση, αλλά υπάρχουν αρκετές αδυναμίες:

- Δεν απορρίπτει πολλά εικονοστοιχεία για n <12.
- Η επιλογή των εικονοστοιχείων δεν είναι βέλτιστη επειδή η αποτελεσματικότητά της εξαρτάται από την σειρά των ερωτήσεων και τη διανομή των εμφανίσεων των γωνιών.
- Τα αποτελέσματα των δοκιμών μεγάλης ταχύτητας δεν χρησιμοποιούνται κάπου στον αλγόριθμο.
- Πολλαπλά χαρακτηριστικά εντοπίζονται το ένα πολύ κοντά στο άλλο.

Τα πρώτα 3 προβλήματα εξετάζονται με μηχανική μάθηση. Το τελευταίο αντιμετωπίζεται χρησιμοποιώντας μη μέγιστη απόκρυψη.

2.9.3.6 BRIEF (Binary Robust Independent Elementary Features)

Γνωρίζουμε ότι ο SIFT χρησιμοποιεί ένα διάνυσμα 128 διαστάσεων για τους περιγραφείς. Δεδομένου ότι χρησιμοποιεί αριθμούς κινητής υποδιαστολής, απαιτούνται για τον καθένα 512 bytes. Ομοίως, για τον SURF απαιτούνται τουλάχιστον 256 bytes (διάνυσμα 64 διαστάσεων). Η δημιουργία ενός τέτοιου διανύσματος για χιλιάδες χαρακτηριστικά απαιτεί πολλή μνήμη, η οποία δεν είναι εφικτή για εφαρμογές περιορισμένων πόρων, ειδικά για ενσωματωμένα συστήματα. Όσο μεγαλύτερη είναι η απαιτούμενη μνήμη, τόσο περισσότερος χρόνος χρειάζεται για την αντιστοίχιση.

Όμως, όλες αυτές οι διαστάσεις μπορεί να μην χρειάζονται για την αντιστοίχιση. Μπορούμε να το συμπιέζουμε τους περιγραφείς χρησιμοποιώντας διάφορες μεθόδους όπως η PCA, η LDA κλπ. Ακόμη και άλλες μέθοδοι, όπως ο κατακερματισμός (hashing) χρησιμοποιώντας LSH (Locality Sensitive Hashing), χρησιμοποιούνται για να μετατρέψουν τους περιγραφείς SIFT από αριθμούς κινητής υποδιαστολής σε δυαδικές συμβολοσειρές. Αυτές οι δυαδικές συμβολοσειρές χρησιμοποιούνται για αντιστοίχιση χαρακτηριστικών χρησιμοποιώντας τη απόσταση Hamming (Hamming Distance). Αυτό παρέχει πολύ καλύτερη ταχύτητα επειδή η εύρεση της απόστασης hamming είναι απλά εφαρμογή αποκλειστικού ή (XOR) και μέτρησης bit, υπολογισμοί ή οποίοι είναι πολύ γρήγοροι σε σύγχρονους επεξεργαστές με οδηγίες SSE. Όμως, πρέπει να βρούμε πρώτα τους περιγραφείς, ώστε να εφαρμόσουμε κατακερματισμό, κάτι που δεν λύνει το αρχικό μας πρόβλημα στη μνήμη. το σημείο αυτό είναι που εφαρμόζεται ο BRIEF . Παρέχει μια συντόμευση για να βρεθούν οι συμβολοσειρές χωρίς να απαιτείται να βρεθούν πρώτα οι περιγραφείς.

Εν ολίγοις, ο BRIEF είναι μία ταχύτερη μέθοδος υπολογισμού περιγραφέων και αντιστοίχισης εικόνων. Παρέχει, επίσης, πολύ καλή αναγνώριση, εκτός αν υπάρχει μεγάλη περιστροφή της εικόνας στο επίπεδο.

2.9.3.7 ORB (Oriented FAST and Rotated BRIEF)

Ο ORB είναι μια αποτελεσματική εναλλακτική λύση για το SIFT ή το SURF που αναπτύχθηκε το 2011. Ο SIFT και ο SURF είναι κατοχυρωμένοι με δίπλωμα ευρεσιτεχνίας και πρέπει να τα πληρωθούν για τη χρήση τους. Αντιθέτως ο ORB είναι δωρεάν.

Ο ORB είναι βασικά μια ένωση του ανιχνευτή χαρακτηριστικών FAST και του περιγραφέα BRIEF με πολλές τροποποιήσεις που αποσκοπούν στην ενίσχυση της απόδοσης. Αρχικά, χρησιμοποιείται ο FAST για να βρεθούν τα σημεία κλειδιά και, στη συνέχεια, εφαρμόζεται ο Harris Corner για να βρεθούν τα καλύτερα σημεία N μεταξύ τους. Χρησιμοποιεί επίσης πυραμίδα για την παραγωγή χαρακτηριστικών σε πολλαπλές κλίμακες. Όμως, ένα σημαντικό πρόβλημα είναι ότι ο FAST δεν υπολογίζει τον προσανατολισμό. Για την λύση αυτού του προβλήματος οι συγγραφείς παρουσίασαν την ακόλουθη τροποποίηση.

Υπολογίζει το κέντρο βάρους της έντασης μιας επιφάνειας με κέντρο την γωνία. Η κατεύθυνση του διανύσματος από αυτό το γωνιακό σημείο στο κέντρο βάρους δίνει τον προσανατολισμό.

Ο ORB χρησιμοποιεί περιγραφείς BRIEF. Έχουμε, όμως, ήδη δει ότι ο BRIEF έχει πρόβλημα με τις περιστροφές. Έτσι, αυτό που κάνει ο ORB είναι να «κατευθύνει» τον BRIEF σύμφωνα με τον προσανατολισμό των βασικών σημείων.

Η εργασία ισχυρίζεται ότι ο ORB είναι πολύ ταχύτερος από τον SURF και τον SIFT και ο περιγραφέας ORB λειτουργεί καλύτερα από τον SURF. Ο ORB είναι μια καλή επιλογή σε συσκευές χαμηλής κατανάλωσης (κινητά τηλέφωνα), για πανοραμικές ραφές κ.λπ.

2.9.3.8 Πίνακας Σύγκρισης Αλγορίθμων

Στον παρακάτω πίνακα παρουσιάζεται η κατάταξη των τριών αλγορίθμων (SIFT, SURF, ORB) ως προς τέσσερα κριτήρια:

- τον απόλυτο αριθμό των χαρακτηριστικών που εξάγονται
- το ποσοστό των χαρακτηριστικών που παρακολουθούνται εύκολα

- τον χρόνο αναγνώρισης χαρακτηριστικών και
- την μετατόπιση των χαρακτηριστικών σε μετασχηματισμένες εικόνες

Αλγόριθμοι	SIFT	SURF	ORB
Αριθμός εξαγόμενων χαρακτηριστικών	1	2	3
Ποσοστό παρακολουθούμενων χαρακτηριστικών	3	2	1
Χρόνος αναγνώρισης	3	2	1
Μετατόπιση χαρακτηριστικών	2	3	1
Σύνολο	9	9	6

Με βάση αυτήν την κατάταξη, προκύπτει ότι ο πιο αποτελεσματικός αλγόριθμος είναι θεωρητικά ο ORB.

2.9.4 Αντιστοίχιση χαρακτηριστικών

Αλγόριθμος Brute Force Matcher

Ο αλγόριθμος Brute Force Matcher είναι απλός. Παίρνει τον περιγραφέα ενός χαρακτηριστικού στο πρώτο σετ και το συγκρίνει με όλα τα άλλα χαρακτηριστικά του δεύτερου σετ χρησιμοποιώντας έναν υπολογισμό απόστασης. Το πλησιέστερο που ταιριάζει επιστρέφεται.

3. Ανάλυση

3.1 Απαιτήσεις εφαρμογής κινητών συσκευών επαυξημένης πραγματικότητας για τον πολιτισμό

Ο τουρισμός στην Ελλάδα αποτελεί ένα από τα βασικότερα στοιχεία της οικονομικής δραστηριότητας στη χώρα. Η Ελλάδα είναι ένας σημαντικός τουριστικός προορισμός και παγκόσμιος πόλος έλξης, για τον πλούσιο πολιτισμό και την ιστορία της, η οποία αντανακλάται σε μεγάλο βαθμό από τα 18 μνημεία παγκόσμιας κληρονομιάς της UNESCO, που είναι μεταξύ των περισσότερων στην Ευρώπη και τον κόσμο. Είναι, λοιπόν, αναγκαίο σε μια ραγδαία τεχνολογικά εξελισσόμενη κοινωνία να αναπτυχθούν εργαλεία και εφαρμογές που βοηθούν στην ανάπτυξη του πολιτιστικού τουρισμού

Η επαυξημένη πραγματικότητα είναι μια σχετικά παλιά τεχνολογία, που έχει αρχίσει να βρίσκει εφαρμογή τα τελευταία χρόνια μέσω εφαρμογών στα κινητά τηλέφωνα και είναι ιδανική για χρήση στον πολιτισμό καθώς επαυξάνει τον πραγματικό κόσμο με ψηφιακές πληροφορίες. Ο καθένας θα ήθελε να δει την Ακρόπολη όπως ήταν πριν χιλιάδες χρόνια, μέσα από την οθόνη του κινητού του, κάνοντας μια βόλτα στον λόφο. Πολλοί είναι αυτοί που επισκέπτονται τα ελληνικά μουσεία και θα ήθελαν να έχουν έναν ξεναγό για να γνωρίσουν πραγματικά το κάθε έκθεμα. Πλέον, ξεναγός μπορεί να γίνει το κινητό τους τηλέφωνο.

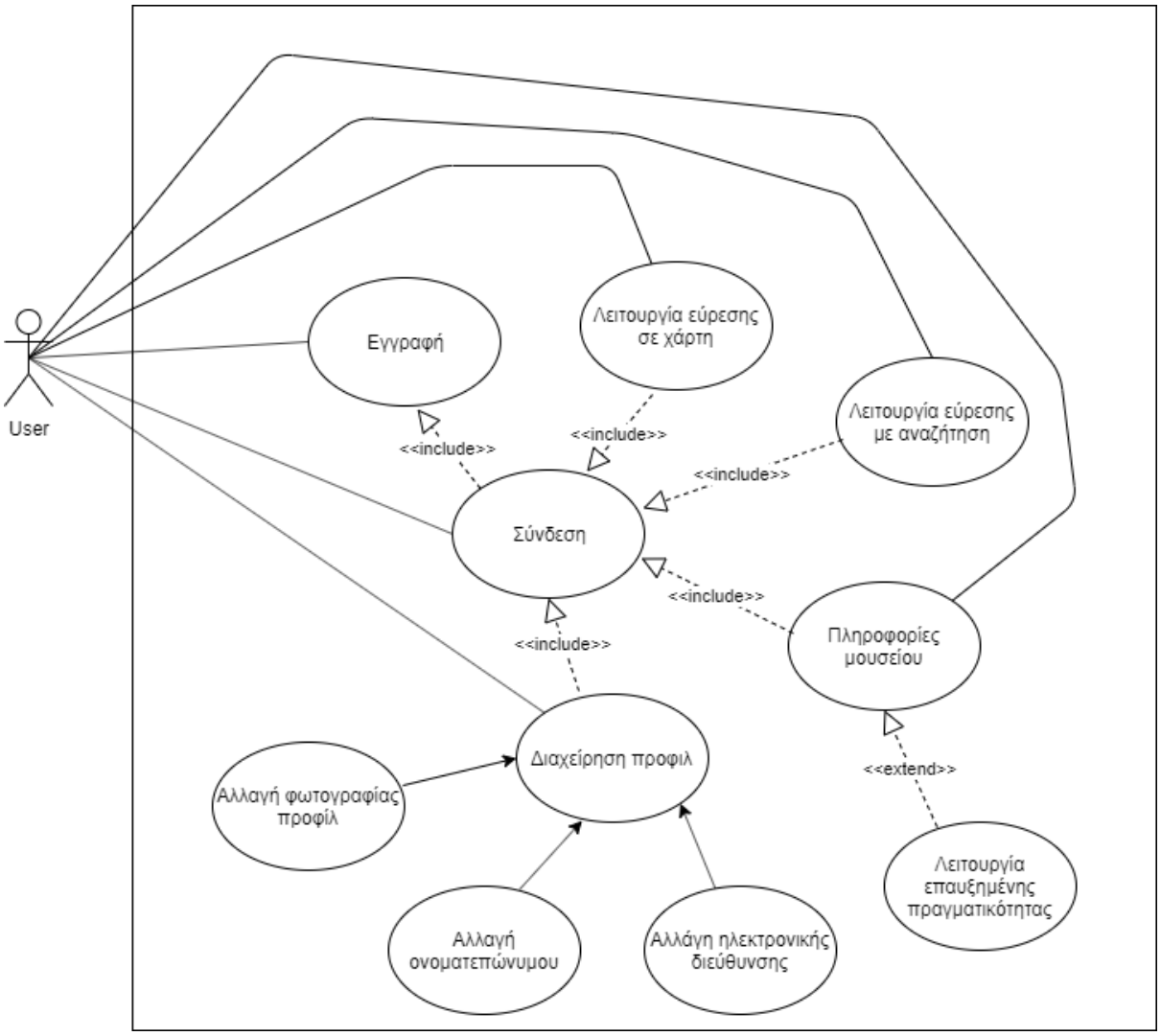
3.2 Η εφαρμογή

Η εφαρμογή που αναπτύχθηκε στην παρούσα διπλωματική εργασία αποτελεί έναν ψηφιακό ξεναγό για τους επισκέπτες των μουσείων. Για την χρήση της απαιτείται δημιουργία λογαριασμού. Παρέχεται, επίσης, η δυνατότητα ο χρήστης να συνδεθεί με τον λογαριασμό Google ή με τον λογαριασμό Facebook του ώστε να μην χρειάζεται να δημιουργήσει νέο λογαριασμό. Αφού συνδεθεί με οποιονδήποτε από τους παραπάνω τρόπους έχει στην διάθεσή του τρεις καρτέλες. Στην πρώτη,

πληκτρολογώντας σε μια μπάρα αναζήτησης μπορεί να βρει σε μια λίστα το μουσείο ή τα μουσεία που τον ενδιαφέρει να επισκεφτεί. Στην δεύτερη, μπορεί να βρει όλα τα διαθέσιμα μουσεία κοντά του πάνω σε έναν χάρτη και στην τρίτη μπορεί να επεξεργαστεί τα στοιχεία του λογαριασμού του, δηλαδή το ονοματεπώνυμο την ηλεκτρονική διεύθυνση και την φωτογραφία προφίλ του.

Αφού επιλέξει μουσείο είτε μέσω της καρτέλας αναζήτησης, είτε μέσω του χάρτη εμφανίζονται στην οθόνη όλες οι διαθέσιμες πληροφορίες του. Αυτές είναι, ο τίτλος του, μια φωτογραφία της πρόσοψης, η διεύθυνση, μια συνοπτική περιγραφή, οι ώρες λειτουργίας, η τιμή του εισιτηρίου και ένας υπερσύνδεσμος στην ιστοσελίδα του μουσείου. Στο τέλος της οθόνης αυτής βρίσκεται ένα κουμπί με τίτλο AR το οποίο ενεργοποιεί την λειτουργία επαυξημένης πραγματικότητας. Ανοίγει η κάμερα και στοχεύοντας την σε οποιοδήποτε έκθεμα εμφανίζονται αναλυτικές πληροφορίες για αυτό, οι οποίες βελτιώνουν σε μεγάλο βαθμό την εμπειρία του χρήστη στο μουσείο.

3.3 Σενάρια Χρήσης



3.3.1 Εγγραφή χρήστη

Παρακάτω παρουσιάζονται όλα τα σενάρια εγγραφής ενός νέου χρήστη

3.3.1.1 Βασική ροή γεγονότων

Χρήστης	Frontend	Backend
1. Εκκίνηση εφαρμογής και επιλογή κουμπιού εγγραφή		
	2. Μετάβαση στην οθόνη εγγραφής της εφαρμογής	

3. Συμπλήρωση ονόματος χρήστη, ονόματος, επωνύμου, διεύθυνσης ηλεκτρονικού ταχυδρομείου, και κωδικού 2 φορές για επιβεβαίωση		
	4. Έλεγχος να μην υπάρχουν κενά πεδία, να είναι σωστή η διεύθυνση ηλεκτρονικού ταχυδρομείου και να ταιριάζουν τα δύο πεδία του κωδικού και αποστολή στο backend	
		5. Έλεγχος μοναδικότητας email και ονόματος χρήστη (username), δημιουργία εγγραφής χρήστη στην βάση δεδομένων, δημιουργία authentication token και αποστολή του στο frontend
	6. Επιτυχής σύνδεση και μετάβαση στην κεντρική οθόνη της εφαρμογής στην καρτέλα της αναζήτησης μουσείων	

3.3.1.2 Ροή διαχείρισης σφάλματος – κενό πεδίο / διαφορετικές τιμές στα πεδία του κωδικού / λανθασμένη μορφή διεύθυνση ηλεκτρονικού ταχυδρομείου

Τα πρώτα 3 βήματα είναι ίδια με τα βήματα της βασικής ροής γεγονότων:

	4. Επισήμανση των πεδίων με προβληματικές τιμές με κόκκινο χρώμα και εμφάνιση μηνύματος σφάλματος κάτω από τα προβληματικά πεδία (κενό πεδίο / διαφορετικές τιμές στα πεδία του κωδικού / λανθασμένη μορφή διεύθυνση ηλεκτρονικού ταχυδρομείου). Στην συνέχεια αναμονή για διόρθωση των πεδίων ώστε να προχωρήσει η διαδικασία της εγγραφής	

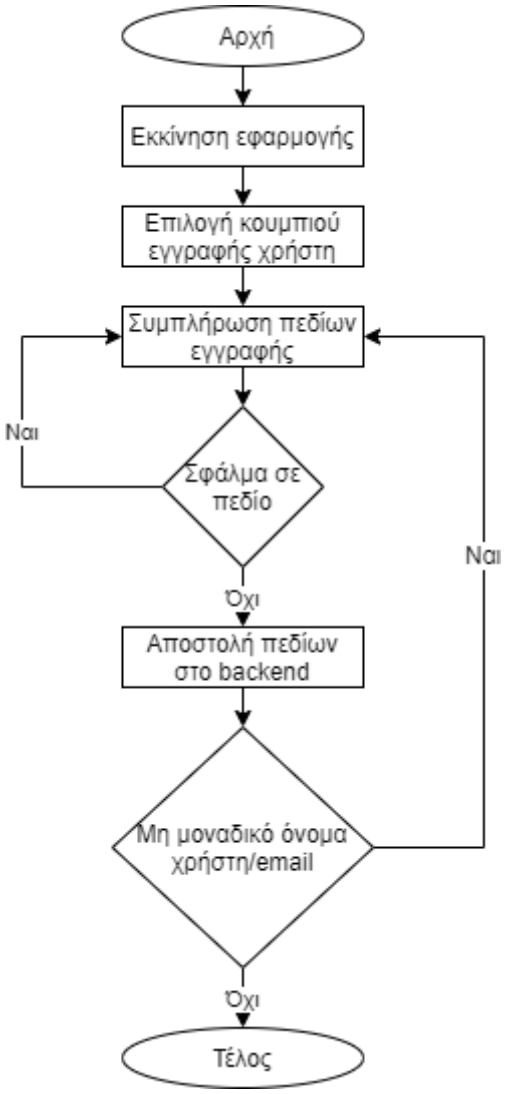
3.3.1.3Ροή διαχείρισης σφάλματος – Μη μοναδικό όνομα χρήστη / διεύθυνση ηλεκτρονικού ταχυδρομείου

Τα πρώτα 4 βήματα είναι ίδια με τα βήματα της βασικής ροής γεγονότων:

Χρήστης	Frontend	Backend
		5. Αποστολή μηνύματος σφάλματος στο frontend (όνομα χρήστη / διεύθυνση ηλεκτρονικού ταχυδρομείου υπάρχει ήδη)

	6. Εμφάνιση αναδυόμενου μηνύματος με το σφάλμα και αναμονή για διόρθωση των πεδίων ώστε να προχωρήσει η διαδικασία της εγγραφής	

3.3.1.4 Διάγραμμα δραστηριοτήτων



Εικόνα 8 Εγγραφή χρήστη

3.3.2 Σύνδεση χρήστη

3.3.2.1 Βασική ροή γεγονότων – Απλή σύνδεση

Χρήστης	Frontend	Backend
1. Εκκίνηση εφαρμογής, εισαγωγή ονόματος χρήστη και κωδικού πρόσβασης και επιλογή του κουμπιού σύνδεση		
	2. Αποστολή διαπιστευτηρίων στο backend	
		3. Επιβεβαίωση διαπιστευτηρίων, δημιουργία authentication token και αποστολή του στο frontend
	4. Επιτυχής σύνδεση και μετάβαση στην κεντρική οθόνη της εφαρμογής στην καρτέλα της αναζήτησης μουσείων	

3.3.2.2 Βασική ροή γεγονότων – Σύνδεση μέσω Google

Χρήστης	Frontend	Backend
1. Εκκίνηση εφαρμογής και επιλογή του κουμπιού σύνδεση με Google		
	2. Ανάκτηση ενός Google Authentication Token και αποστολή στο backend.	

		3. Αν είναι η πρώτη φορά που γίνεται η σύνδεση μέσω Google δημιουργία χρήστη και αποθήκευση στην βάση δεδομένων, αλλιώς αποθήκευση του token στον ήδη υπάρχων χρήστη που έχει δημιουργηθεί στην πρώτη σύνδεση και αποστολή του πάλι πίσω στο frontend
	4. Επιτυχής σύνδεση και μετάβαση στην κεντρική οθόνη της εφαρμογής στην καρτέλα της αναζήτησης μουσείων	

3.3.2.3 Βασική ροή γεγονότων – Σύνδεση μέσω Facebook

Χρήστης	Frontend	Backend
1. Εκκίνηση εφαρμογής και επιλογή του κουμπιού συνέχεια με το Facebook		
	2. Ανάκτηση ενός Facebook Authentication Token και αποστολή στο backend.	

		3. Αν είναι η πρώτη φορά που γίνεται η σύνδεση μέσω Facebook δημιουργία χρήστη και αποθήκευση στην βάση δεδομένων, αλλιώς αποθήκευση του token στον ήδη υπάρχων χρήστη που έχει δημιουργηθεί στην πρώτη σύνδεση και αποστολή του πάλι πίσω στο frontend
	4. Επιτυχής σύνδεση και μετάβαση στην κεντρική οθόνη της εφαρμογής στην καρτέλα της αναζήτησης μουσείων	

3.3.2.4 Ροή διαχείρισης σφάλματος – κενό πεδίο

Χρήστης	Frontend	Backend
1. Εκκίνηση εφαρμογής, εισαγωγή ονόματος χρήστη και κωδικού πρόσβασης και επιλογή του κουμπιού σύνδεση		

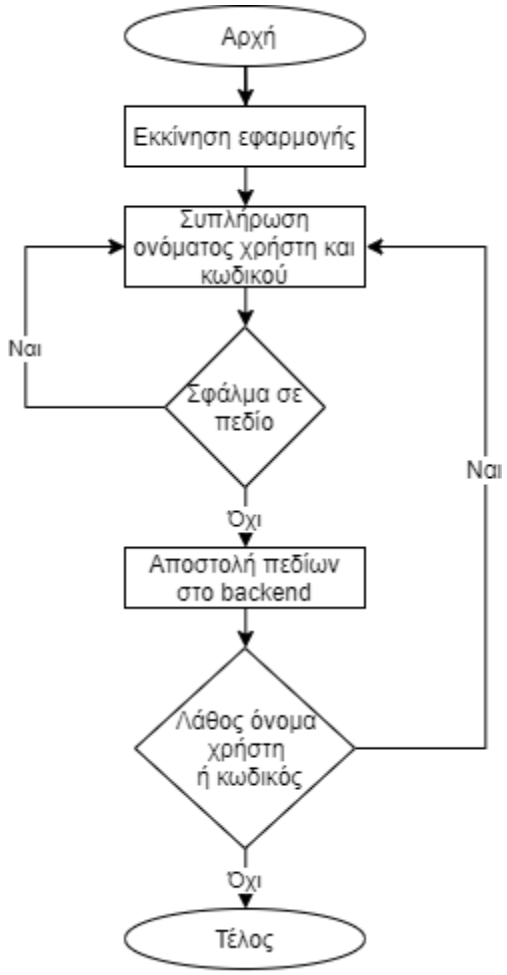
	2. Επισήμανση των πεδίων με προβληματικές τιμές με κόκκινο χρώμα και εμφάνιση μηνύματος σφάλματος κάτω από τα προβληματικά πεδία (κενό πεδίο). Στην συνέχεια αναμονή για διόρθωση των πεδίων ώστε να ξεκινήσει εκ νέου η διαδικασία της σύνδεσης	

3.3.2.5 Ροή διαχείρισης σφάλματος – Μη μοναδικό όνομα χρήστη / διεύθυνση ηλεκτρονικού ταχυδρομείου

Τα πρώτα 2 βήματα είναι ίδια με τα βήματα της βασικής ροής γεγονότων της απλής σύνδεσης:

Χρήστης	Frontend	Backend
		3. Αποστολή μηνύματος σφάλματος στο frontend (λάθος όνομα χρήστη ή κωδικός)
	4. Εμφάνιση αναδυόμενου μηνύματος με το σφάλμα και αναμονή για διόρθωση των πεδίων ώστε να ξεκινήσει εκ νέου η διαδικασία της σύνδεσης	

3.3.2.6 Διάγραμμα δραστηριοτήτων



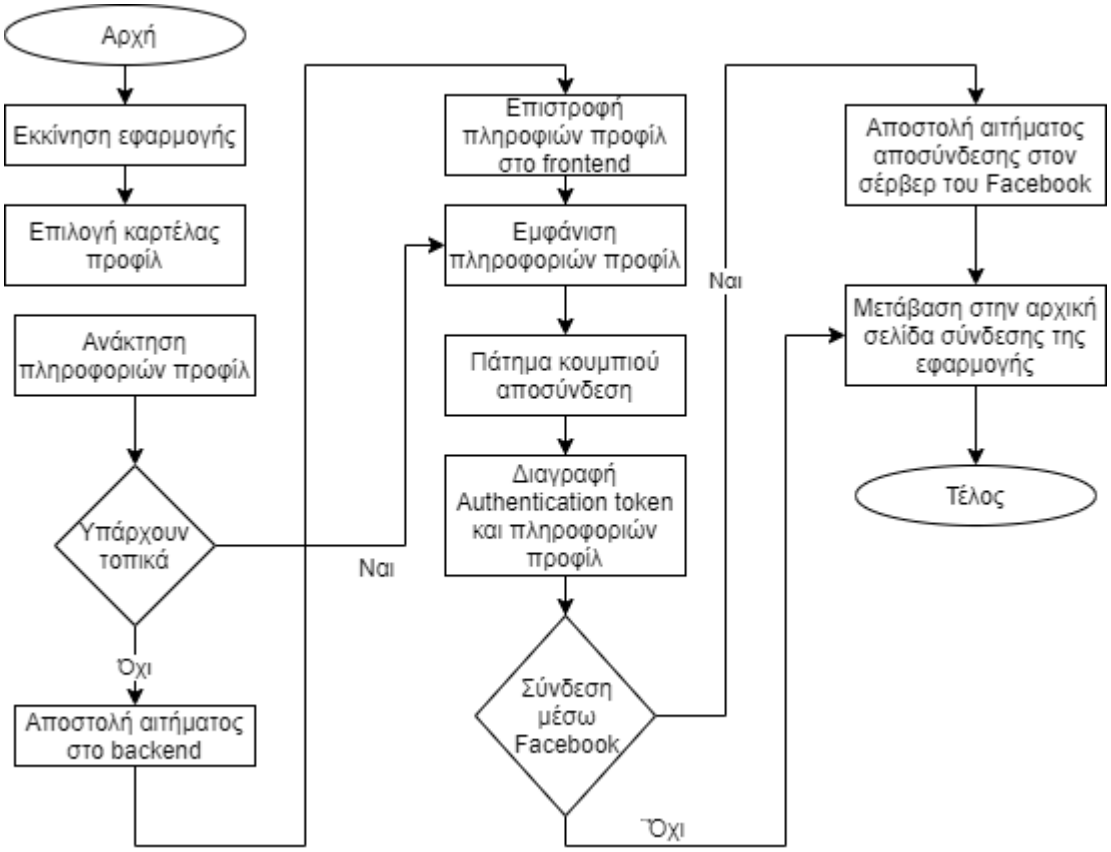
Εικόνα 9 Σύνδεση χρήστη

3.3.3 Αποσύνδεση χρήστη

Χρήστης	Frontend	Backend
1. Επιλογή καρτέλας «Προφίλ»		

	2. Ανάκτηση των πληροφοριών προφίλ (φωτογραφία, όνομα, επώνυμο, διεύθυνση ηλεκτρονικού ταχυδρομείου) από την τοπική μνήμη της συσκευής αν υπάρχουν, αλλιώς αποστολή αιτήματος στο backend για ανάκτηση.	
		3. Σε περίπτωση αιτήματος, εύρεση των πληροφοριών χρήστη στην βάση και αποστολή τους στο frontend. (Η φωτογραφία χρήστη είναι αποθηκευμένη στο firebase και στην βάση αποθηκεύεται το link ώστε να την κατεβάσει το frontend)
	4. Εμφάνιση πληροφοριών προφίλ	
5. Πάτημα κουμπιού «Αποσύνδεση»		
	5. Διαγραφή Authentication token και πληροφοριών προφίλ (Σε περίπτωση σύνδεσης μέσω Facebook αποστολή αιτήματος αποσύνδεσης στον σέρβερ του Facebook) και μετάβαση στην αρχική σελίδα σύνδεσης της εφαρμογής	

3.3.3.1 Διάγραμμα δραστηριοτήτων



Εικόνα 10 Αποσύνδεση χρήστη

3.3.4 Επεξεργασία προφίλ

Παρακάτω παρουσιάζονται οι λειτουργίες που αφορούν στην επεξεργασία των πληροφοριών του προφίλ του χρήστη.

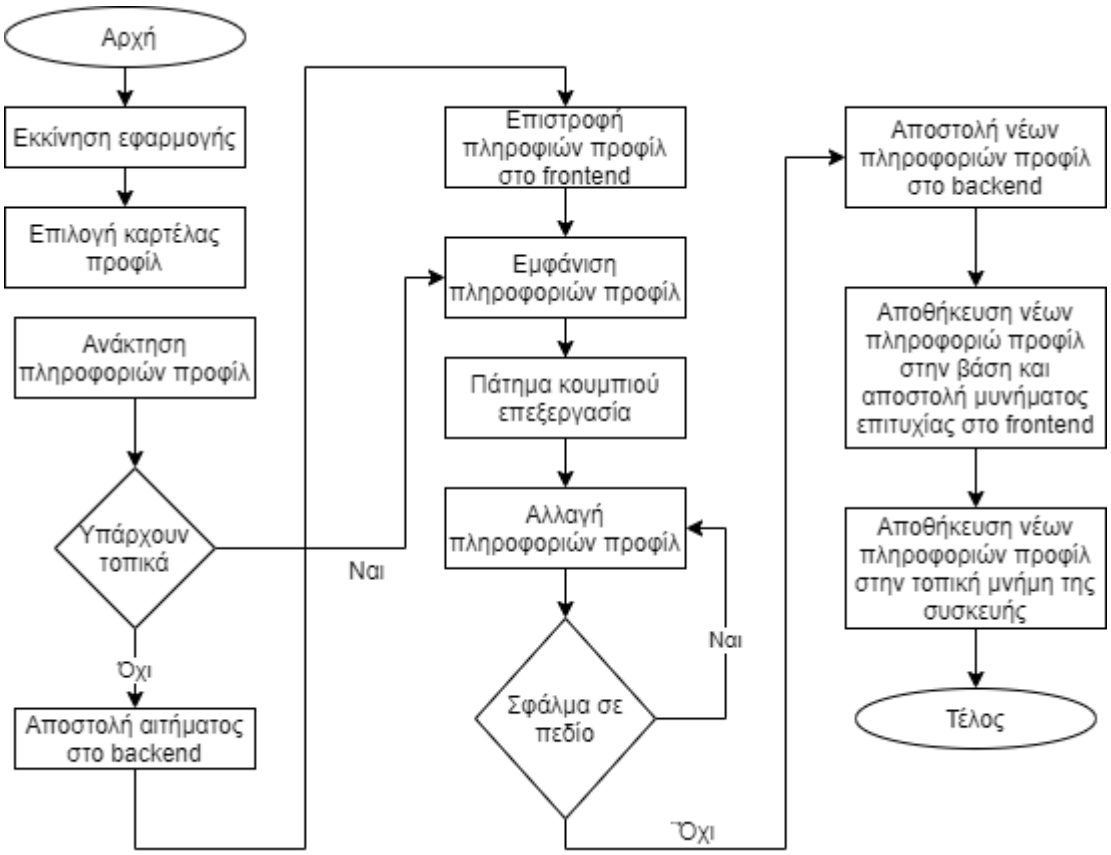
3.3.4.1 Επεξεργασία ονόματος – επωνύμου – ηλεκτρονικής διεύθυνσης

Τα πρώτα 4 βήματα είναι ίδια με την ροή αποσύνδεσης χρήστη

Χρήστης	Frontend	Backend

5. Πάτημα κουμπιού «Επεξεργασία», αλλαγή οποιουδήποτε πεδίου (όνομα, επώνυμο ή διεύθυνση ηλεκτρονικού ταχυδρομείου) και πάτημα του κουμπιού αποθήκευση		
	6. Έλεγχος πεδίων και σε περίπτωση που δεν υπάρχουν κενά πεδία αποστολή των νέων πληροφοριών προφίλ στο backend. Σε περίπτωση που υπάρχουν αναμονή για εισαγωγή μη κενών πεδίων	
		7. Αποθήκευση των νέων πληροφοριών προφίλ στην βάση και αποστολή μηνύματος επιτυχίας στο frontend
	8. Αποθήκευση των νέων πληροφοριών στην τοπική μνήμη της συσκευής	

Διάγραμμα δραστηριοτήτων



Εικόνα 11 Επεξεργασία προφίλ

3.3.4.2 Επεξεργασία φωτογραφίας προφίλ – Χρήση κάμερας

Τα πρώτα 4 βήματα είναι ίδια με την ροή αποσύνδεσης χρήστη

Χρήστης	Frontend	Backend
5. Πάτημα κουμπιού «Φωτογραφία»		

	6. Εμφάνιση αναδυόμενης λίστας με τις επιλογές χρήση κάμερας, επιλογή από γκαλερί, αφαίρεση της εικόνας προφίλ. (Σε περίπτωση σύνδεσης Google – Facebook υπάρχει και η επιλογή κατέβασμα από Google - Facebook)	
7. Επιλογή «Χρήση κάμερας»		
	8. Άνοιγμα προεπιλεγμένης εφαρμογής κάμερας της συσκευής	
9. Λήψη φωτογραφίας		
	10. Επιστροφή στην εφαρμογή, αποθήκευση της νέας φωτογραφίας στο firebase και λήψη του υπερσυνδέσμου της. Αποστολή του υπερσυνδέσμου στο backend και κλείσιμο της αναδυόμενης λίστας	
		11. Αποθήκευση του υπερσυνδέσμου στην βάση και αποστολή μηνύματος επιτυχίας στο frontend
	12. Αποθήκευση της φωτογραφίας στην τοπική μνήμη της συσκευής	

3.3.4.3 Επεξεργασία φωτογραφίας προφίλ – Επιλογή από γκαλερί

Τα πρώτα 6 βήματα είναι ίδια με την ροή Επεξεργασία φωτογραφίας προφίλ – Χρήση κάμερας

Χρήστης	Frontend	Backend
7. Επιλογή «Επιλογή από γκαλερί»		
	8. Άνοιγμα προεπιλεγμένης εφαρμογής προεπισκόπησης φωτογραφιών της συσκευής	
9. Επιλογή φωτογραφίας από την γκαλερί του κινητού		
	10. Επιστροφή στην εφαρμογή, αποθήκευση της φωτογραφίας που επιλέχθηκε στο firebase και λήψη του υπερσυνδέσμου της. Αποστολή του υπερσυνδέσμου στο backend και κλείσιμο της αναδυόμενης λίστας	
		11. Αποθήκευση του υπερσυνδέσμου στην βάση και αποστολή μηνύματος επιτυχίας στο frontend
	12. Αποθήκευση της φωτογραφίας στην τοπική μνήμη της συσκευής	

3.3.4.4 Επεξεργασία φωτογραφίας προφίλ – Κατέβασμα από Facebook ή Google

Τα πρώτα 6 βήματα είναι ίδια με την ροή Επεξεργασία φωτογραφίας προφίλ – Χρήση κάμερας

Χρήστης	Frontend	Backend
7. Επιλογή «Κατέβασμα από Facebook - Google»		
	8. Λήψης της φωτογραφίας από τον σέρβερ της Facebook – Google αποθήκευση στο firebase και λήψη του υπερσυνδέσμου της. Αποστολή του υπερσυνδέσμου στο backend και κλείσιμο της αναδυόμενης λίστας	
		9. Αποθήκευση του υπερσυνδέσμου στην βάση και αποστολή μηνύματος επιτυχίας στο frontend
	10. Αποθήκευση της φωτογραφίας στην τοπική μνήμη της συσκευής	

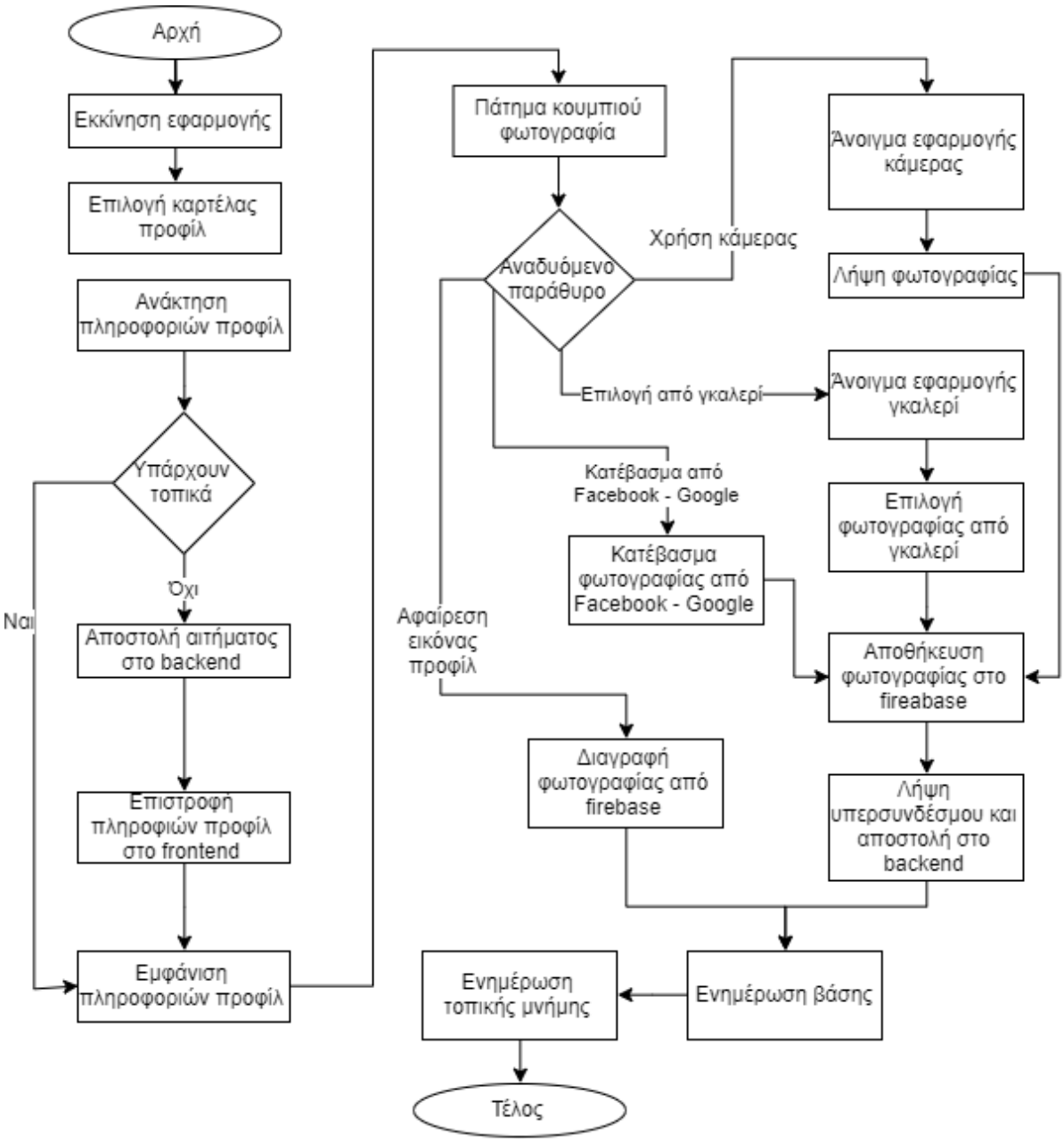
3.3.4.5 Επεξεργασία φωτογραφίας προφίλ – Αφαίρεση της εικόνας προφίλ

Τα πρώτα 6 βήματα είναι ίδια με την ροή Επεξεργασία φωτογραφίας προφίλ – Χρήση κάμερας

Χρήστης	Frontend	Backend
7. Επιλογή «Αφαίρεση της εικόνας προφίλ»		

	8. Διαγραφή της φωτογραφίας από το firebase και αποστολή αιτήματος διαγραφής του υπερσυνδέσμου στο backend	
		9. Διαγραφή του υπερσυνδέσμου από την βάση και αποστολή μηνύματος επιτυχίας στο frontend
	10. Διαγραφή της φωτογραφίας από την τοπική μνήμη της συσκευής	

Διάγραμμα δραστηριοτήτων



3.3.5 Χρήση AR λειτουργίας της εφαρμογής

Παρακάτω παρουσιάζονται οι λειτουργίες που αφορούν στην χρήση της AR λειτουργίας της εφαρμογής.

3.3.5.1 Προσπέλαση της λειτουργίας AR μέσω της αναζήτησης

Χρήστης	Frontend	Backend

1. Επιλογή καρτέλας «Αναζήτηση» και πληκτρολόγηση του μουσείου		
	2. Αποστολή της τιμής του πεδίου της αναζήτησης στο backend κάθε φορά που αλλάζει	
		3. Εύρεση της λίστας των μουσείων που εμπεριέχουν τμήμα ή ολόκληρη την τιμή του πεδίου της αναζήτησης στον τίτλο τους και αποστολή στο frontend
	4. Εμφάνιση της λίστας των μουσείων στον χρήστη	
5. Επιλογή μουσείου		
	6. Μετάβαση στην σελίδα αναλυτικών πληροφοριών του μουσείου με την ένδειξη φόρτωση και αποστολή αιτήματος στο backend για λήψη αναλυτικών πληροφοριών για το επιλεγμένο μουσείο	
		7. Εύρεση των αναλυτικών πληροφοριών του επιλεγμένου μουσείου και των υπερσυνδέσμων για τις φωτογραφίες στις οποίες θα γίνει η αναγνώριση στην βάση και αποστολή στο frontend
	8. Εμφάνιση αναλυτικών πληροφοριών του μουσείου και λήψη από το firebase των φωτογραφιών του μουσείου στις οποίες θα γίνει η αναγνώριση	

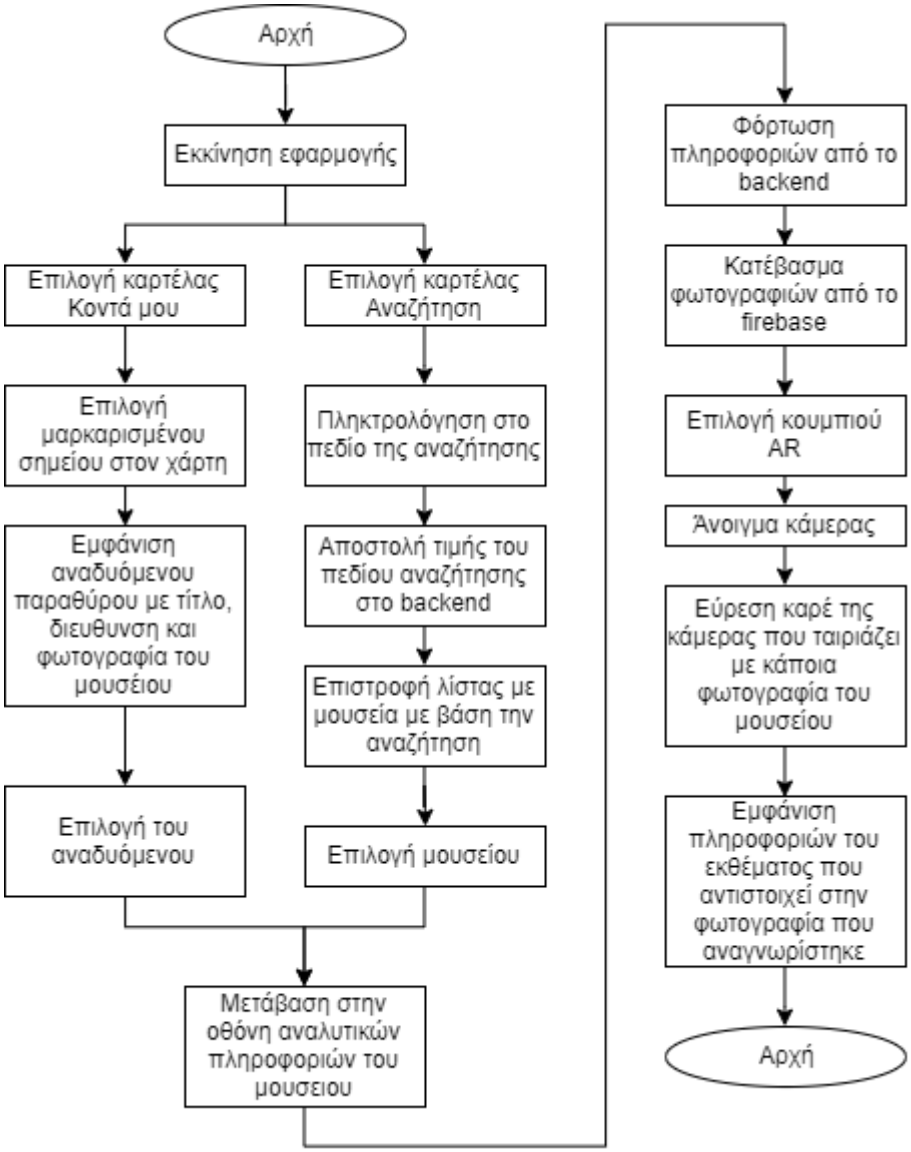
9. Πάτημα κουμπιού «AR»		
	10. Άνοιγμα κάμερας συσκευής και συνεχής σύγκριση σε πραγματικό χρόνο της εικόνας που προέρχεται από την κάμερα με τις φωτογραφίες που έχουν ληφθεί για το συγκεκριμένο μουσείο. Όταν βρεθεί φωτογραφία που ταιριάζει με κάποιο καρέ της κάμερας μετάβαση στην σελίδα πληροφοριών εκθέματος με πληροφορίες για το έκθεμα της φωτογραφίας που αναγνωρίστηκε	

3.3.5.2 Προσπέλαση της λειτουργίας AR μέσω του χάρτη

Χρήστης	Frontend	Backend
1. Επιλογή καρτέλας «Κοντά μου» και πάτημα ενός μαρκαρισμένου σημείου στον χάρτη.		
	2. Εμφάνιση τίτλου, διεύθυνσης και φωτογραφίας του μουσείου για το μαρκαρισμένο σημείο που επιλέχθηκε σε αναδυόμενο παράθυρο	
3. Επιλογή του μουσείου στο αναδυόμενο παράθυρο		

Το υπόλοιπα βήματα είναι ίδια με τα βήματα 6 έως 10 της προσπέλασης της λειτουργίας AR μέσω της αναζήτησης.

3.3.5.3 Διάγραμμα δραστηριοτήτων



Εικόνα 12 Λειτουργία AR

4. Ανάπτυξη Συστήματος

4.1 Κλάσεις

4.1.1 Main Activity / Tab View Controller

Η κλάση αυτή είναι το σημείο από το οποίο εκκινεί η εφαρμογή. Μέσω αυτής, ο χρήστης μπορεί να επιλέξει ανάμεσα σε 3 καρτέλες: «Αναζήτηση», «Κοντά μου» και «Προφίλ» οι οποίες αποτελούν άλλες κλάσεις που θα αναλυθούν παρακάτω. Σε περίπτωση που κατά την εκκίνηση της εφαρμογής δεν έχει συνδεθεί ξανά κάποιος χρήστης ή έχει αποσυνδεθεί, ο χρήστης μεταφέρεται στην κλάση Login Activity.

4.1.2 Search Fragment / View Controller

Η κλάση αυτή αποτελεί την πρώτη από τις 3 καρτέλες της Main Activity. Παρέχει την δυνατότητα αναζήτησης ενός ή περισσότερων μουσείων και παραθέτει σε μια λίστα τα μουσεία που ταιριάζουν στην αναζήτηση του χρήστη. Με την επιλογή οποιουδήποτε μουσείου από αυτά που εμφανίστηκαν στην λίστα ο χρήστης μεταβαίνει στην κλάση Museum Description Activity η οποία θα αναλυθεί παρακάτω.

4.1.3 MyMap Fragment / View Controller

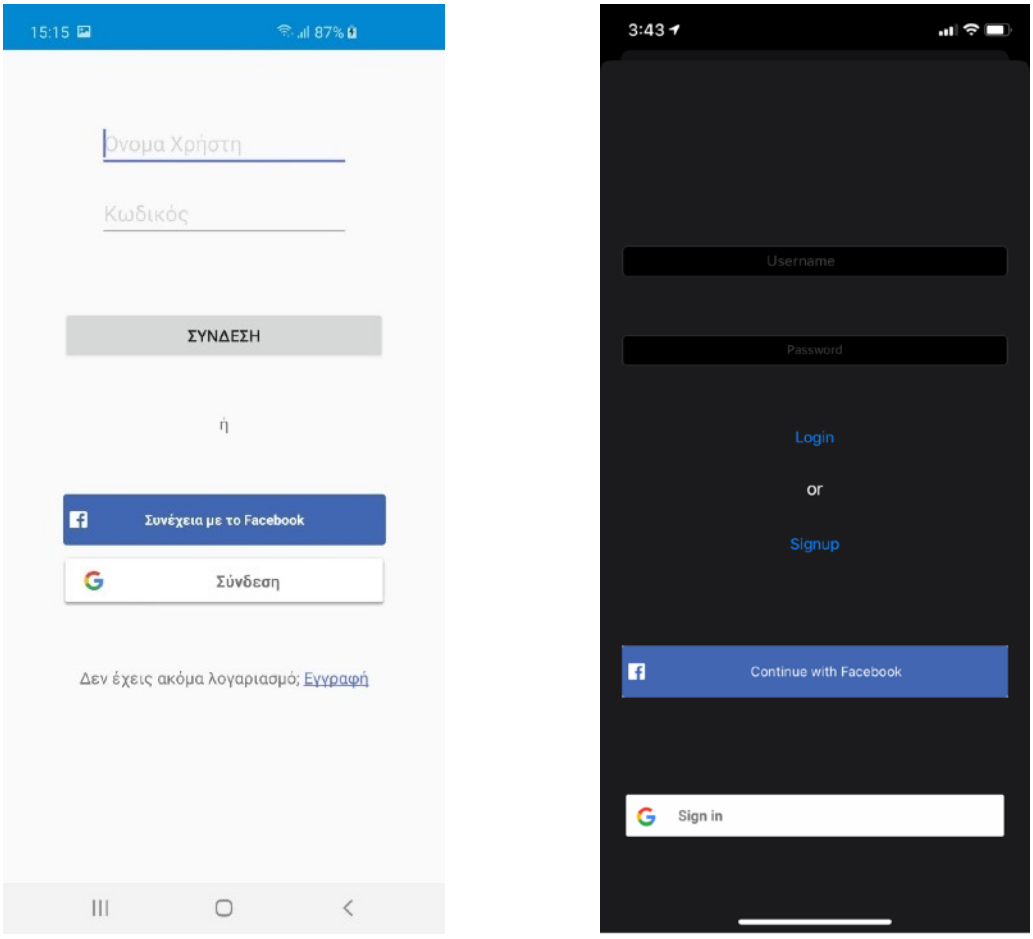
Αυτή η κλάση δημιουργεί έναν χάρτη με σημεία πάνω του όλα τα μουσεία τα οποία είναι εγγεγραμμένα στην εφαρμογή. Μέσα από τον χάρτη αυτό ο χρήστης μπορεί να επιλέξει οποιοδήποτε μουσείο και να μεταβεί στην κλάση Museum Description Activity.

4.1.4 Profile Fragment / View Controller

Εδώ δίνεται η δυνατότητα της επεξεργασίας των στοιχείων του προφίλ του χρήστη καθώς και της αποσύνδεσής του από την εφαρμογή. Στην οθόνη αυτής της κλάσης ο χρήστης μπορεί να αλλάξει εικόνα προφίλ, ονοματεπώνυμο και email. Κατά την αποσύνδεση γίνεται μεταφορά στην κλάση Login Activity.

4.1.5 Login Activity / View Controller

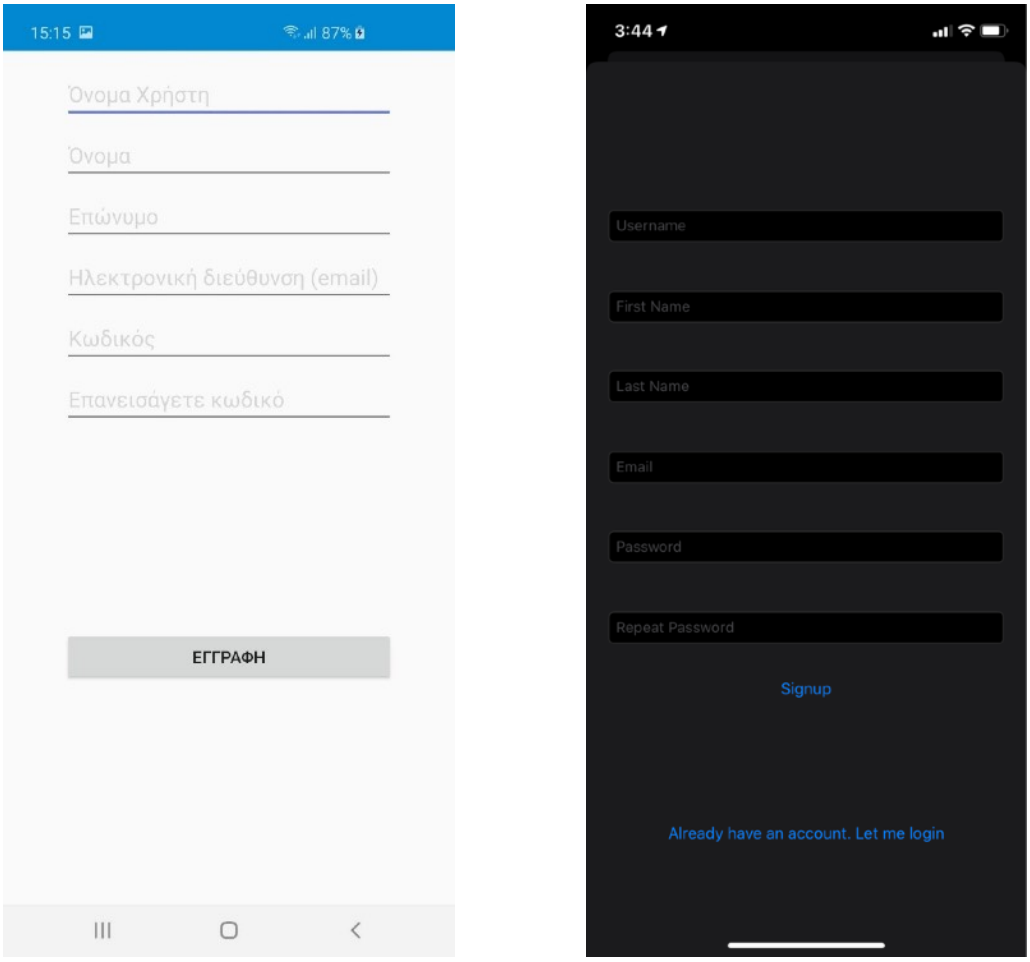
Η κλάση Login Activity ελέγχει τις συνδέσεις και τις εγγραφές των χρηστών. Υπάρχει η δυνατότητα απλής σύνδεσης του χρήστη, καθώς και σύνδεσης μέσω Google ή Facebook. Επίσης, στην οθόνη αυτής της κλάσης υπάρχει υπερσύνδεσμος για μετάβαση στην κλάση Signup Activity.



Εικόνα 13 Login Activity layout Android

4.1.6 Signup Activity / View Controller

Η κλάση αυτή επιτρέπει την δημιουργία νέου χρήστη. Για την εγγραφή του χρήστη απαιτούνται τα πεδία: Όνομα χρήστη, Όνομα, Επώνυμο, Ηλεκτρονική διεύθυνση (email) και κωδικός. Με το όνομα χρήστη και τον κωδικό που θα δημιουργηθούν θα γίνεται η σύνδεση όταν απαιτείται. Αμέσως μετά την επιτυχημένη εγγραφή η εφαρμογή μεταβαίνει στην οθόνη της Main Activity.



Εικόνα 14 Signup Activity Layout Android

4.1.7 Museum Description Activity / View Controller

Αυτή η κλάση είναι υπεύθυνη για τις αναλυτικές πληροφορίες του κάθε μουσείου που παρουσιάζονται στον χρήστη. Στην οθόνη της υπάρχει μια φωτογραφία του μουσείου,

η διεύθυνση του, μια σύντομη περιγραφή, οι ώρες λειτουργίας, η τιμή του εισιτηρίου και ένας υπερσύνδεσμος για την ιστοσελίδα του. Παράλληλα κατά την έναρξη αυτής της κλάσης γίνεται λήψη των φωτογραφιών του μουσείου για την μετέπειτα αναγνώριση. Στο τέλος της οθόνης υπάρχει ένα κουμπί με τίτλο AR το οποίο μεταφέρει τον χρήστη στην κλάση Ar Activity.

4.1.8 Ar Activity / View Controller

Εδώ βρίσκεται όλος ο πυρήνας της αναγνώρισης εικόνας. Με την έναρξη της κλάσης αυτής καλείται μια άλλη κλάση σε c++ με τίτλο native_opencv.cpp η οποία συγκρίνει κάθε καρέ με καθεμία από τις εικόνες του μουσείου που μεταφορτώθηκαν στην κλάση Museum Description Activity. Όταν ο αλγόριθμος αυτός ταιριάξει ένα καρέ με κάποια από τις εικόνες, μεταβαίνουμε στην κλάση Exhibit Info Activity / View.

```
/*
*  ImageProcessing.cpp
*/
#include <jni.h>
#include <android/log.h>
#include <opencv2/core/types_c.h>
#include <opencv2/core/core.hpp>
#include <opencv2/imgproc/imgproc.hpp>
#include <opencv2/features2d.hpp>
#include <opencv2/xfeatures2d.hpp>
#include <opencv2/calib3d/calib3d.hpp>

using namespace std;
using namespace cv;

#define  LOG_TAG    "native_opencv"
#define  LOGE(...)  __android_log_print(ANDROID_LOG_ERROR,LOG_TAG,__VA_ARGS__)
#define  LOGW(...)  __android_log_print(ANDROID_LOG_WARN,LOG_TAG,__VA_ARGS__)
#define  LOGD(...)  __android_log_print(ANDROID_LOG_DEBUG,LOG_TAG,__VA_ARGS__)
#define  LOGI(...)  __android_log_print(ANDROID_LOG_INFO,LOG_TAG,__VA_ARGS__)

Mat *mCanny = NULL;

const Scalar RGBA_DRAWING_COLOR = Scalar(0, 255, 0, 255);

Ptr<ORB> fd_de;
// cv::BFMatcher dm(NORM_HAMMING);

vector<vector<KeyPoint>> keypointsTemplate;
vector<Mat> descriptorsTemplate;
vector<Mat> imagesTemplate;

vector<KeyPoint> keypointsCamera;
Mat descriptorsCamera;

vector<DMatch> matches;
vector<Point2d> scene_corners(4);

int found;
int frame_counter;
int frame_skipper;
```

```
extern "C" {
int processImage(cv::Mat &img);

void Java_com_example_fanis_frontend_views_CameraPreviewView_InitTemplateImage(
        JNIEnv *env, jobject instance, jlongArray imagesAddrs
) {
    jlong *imagesAddrsNativeArray = env->GetLongArrayElements(imagesAddrs, 0);
    fd_de = ORB::create();
    for (size_t i = 0; i < 10; i++){
        if (imagesAddrsNativeArray[i] != NULL){
            Mat templGray;
            cvtColor(*(Mat *) imagesAddrsNativeArray[i], templGray, cv::COLOR_BGR2GRAY);
            imagesTemplate.push_back(templGray);
            vector<KeyPoint> keypoints;
            Mat descriptor;
            fd_de->detectAndCompute(templGray, Mat(), keypoints, descriptor);
            descriptorsTemplate.push_back(descriptor);
            keypointsTemplate.push_back(keypoints);
            LOGD("D %d x %d", templGray.cols, templGray.rows);
        }
    }

    found = -1;
    frame_counter = 0;
    frame_skipper = 30;
}

jint Java_com_example_fanis_frontend_views_CameraPreviewView_ImageProcessing(
        JNIEnv *env, jobject thiz,
        jint width, jint height,
        jbyteArray NV21FrameData, jintArray outPixels
) {
    jbyte *pNV21FrameData = env->GetByteArrayElements(NV21FrameData, 0);
    jint *poutPixels = env->GetIntArrayElements(outPixels, 0);


    if (mCanny == NULL) {
        LOGD("created mCanny");
        mCanny = new Mat(height, width,
                         CV_8UC1);   // single channel array with 8 bit unsigned integers
    }

    Mat mGray(height, width, CV_8UC1,
              (unsigned char *) pNV21FrameData); // single channel array with 8 bit
unsigned integers
    Mat yuvImg(height + height / 2, width, CV_8UC1, (unsigned char *) pNV21FrameData);
    Mat rgbaImg;
    cvtColor(yuvImg, rgbaImg, cv::COLOR_YUV2RGBA_NV21);

    Mat mResult(height, width, CV_8UC4,
                (unsigned char *) poutPixels);   // 4 channel array with 8 bit unsigned
integers
    int idxFound = processImage(mGray);

    cvtColor(rgbaImg, mResult, cv::COLOR_RGBA2BGRA);

    env->ReleaseByteArrayElements(NV21FrameData, pNV21FrameData, 0);
    env->ReleaseIntArrayElements(outPixels, poutPixels, 0);

    return idxFound;
}

// Homography checks

bool isInside(double minX, double minY, double maxX, double maxY, vector<Point2d> coors)
{
    Point2d x;
    for (vector<Point2d>::iterator c = coors.begin(); c != coors.end(); ++c) {
        if ((c->x < minX) || (c->y < minY) || (c->x > maxX) || (c->y > maxY))
            return false;
    }
```

```
    return true;
}
bool isLeft(Point2d a, Point2d b, Point2d c) {
    return ((b.x - a.x) * (c.y - a.y) - (b.y - a.y) * (c.x - a.x)) > 0;
}
bool isConvex(vector<Point2d> points) {
    if (points.size() > 2) {
        bool left = isLeft(points[0], points[1], points[2]);
        points.push_back(points.front());
        for (size_t i = 3; i < points.size(); i++)
            if (isLeft(points[i - 2], points[i - 1], points[i]) != left)
                return false;
        return true;
    } else
        return false;
}

double radianToDegree(double radian) {
    double degree = radian * ((double) 80 / (double) M_PI);
    if (degree < (double) 0)
        degree = (double) 360 + degree;

    return degree;
}
double angleInDegrees(Point2d v1, Point2d v2) {
    return fmod((radianToDegree(atan2(v1.y - v2.y, v1.x - v2.x))), (double) 360);
}

bool minAngleCheck(vector<Point2d> points, double angle_InDegrees) {
    //20d * Math.PI / 180d
    if (points.size() > 2) {
        points.push_back(points.front());
        for (size_t i = 2; i < points.size(); i++) {
            double a1 = angleInDegrees(points[i - 2], points[i - 1]);
            double a2 = angleInDegrees(points[i], points[i - 1]);
            double d = fmod(fabs(a1 - a2), (double) 180);

            if ((d < angle_InDegrees) || (((double) 180 - d) < angle_InDegrees))
                return false;
        }
        return true;
    } else
        return false;
}

vector<Point2d> MyPerspectiveTransform3(vector<Point2f> yourData, Mat
transformationMatrix) {
    vector<Point2f> transformed;
    vector<Point2d> ret;
    perspectiveTransform(yourData, transformed, transformationMatrix);

    for (size_t i = 0; i < transformed.size(); i++) {
        ret.push_back(Point2d((double) transformed[i].x, (double) transformed[i].y));
//        ret.push_back((Point2d)transformed[i]);
    }
    return ret;
}
double square(double x) {
    return x * x;
}
double points_distance(Point2d a, Point2d b) {
    double ret = sqrt(square(a.x - b.x) + square(a.y - b.y));
    return ret;
}
bool matchedImage(int idx, Mat img_scene, vector<KeyPoint> keypoints_scene, Mat
descriptors_scene) {
    if (descriptors_scene.empty() || descriptorsTemplate[idx].empty()) return false;

    //    match images

    Ptr<DescriptorMatcher> matcher =
DescriptorMatcher::create(DescriptorMatcher::BRUTEFORCE_HAMMING);
```

```
    vector<vector<DMatch> > knn_matches;
    if (keypoints_scene.size() < 2 && keypointsTemplate[idx].size() < 2) return false;
    matcher->knnMatch(descriptorsTemplate[idx], descriptors_scene, knn_matches, 2);
    if (knn_matches.empty()) return false;
    //-- Filter matches using the Lowe's ratio test
    const float ratio_thresh = 0.75f;
    vector<DMatch> good_matches;
    for (size_t i = 0; i < knn_matches.size(); i++) {
        if (knn_matches[i][0].distance < ratio_thresh * knn_matches[i][1].distance) {
            good_matches.push_back(knn_matches[i][0]);
        }
    }

//-- Localize the object
    vector<Point2f> obj;
    vector<Point2f> scene;
    for (size_t i = 0; i < good_matches.size(); i++) {
//-- Get the keypoints from the good matches
        obj.push_back(keypointsTemplate[idx][good_matches[i].queryIdx].pt);
        scene.push_back(keypoints_scene[good_matches[i].trainIdx].pt);
    }
    if (obj.empty() || scene.empty()) return false;
    Mat H = findHomography(obj, scene, RANSAC, 10);

    if (H.empty()) return false;

//-- Get the corners from the image_1 ( the object to be "detected" )
    vector<Point2f> obj_corners(4);
    obj_corners[0] = Point2f(0, 0);
    obj_corners[1] = Point2f((float) imagesTemplate[idx].cols, 0);
    obj_corners[2] = Point2f((float) imagesTemplate[idx].cols, (float)
imagesTemplate[idx].rows);
    obj_corners[3] = Point2f(0, (float) imagesTemplate[idx].rows);
//        perspectiveTransform( obj_corners, scene_corners, H);
    scene_corners = MyPerspectiveTransform3(obj_corners, H);

    double marginH = img_scene.cols * (double) 0.1;
    double marginV = img_scene.rows * (double) 0.1;
    bool homographyOK = isInside(-marginH, -marginV, img_scene.cols + marginH,
                                 img_scene.rows + marginV, scene_corners);
    if (homographyOK) {
        for (size_t i = 1; i < scene_corners.size(); i++) {
            if (points_distance(scene_corners[i - 1], scene_corners[i]) < 1) {
                homographyOK = false;
                break;
            }
        }
    }
    if (homographyOK)
        homographyOK = isConvex(scene_corners);
    if (homographyOK)
        homographyOK = minAngleCheck(scene_corners, (double) 20);

    if (homographyOK) return true;
    else return false;
}

int processImage(cv::Mat &img) {
    if (found >= 0) {
        line(img, scene_corners[0], scene_corners[1], Scalar(0, 255, 0, 255), 4);
        line(img, scene_corners[1], scene_corners[2], Scalar(0, 255, 0, 255), 4);
        line(img, scene_corners[2], scene_corners[3], Scalar(0, 255, 0, 255), 4);
        line(img, scene_corners[3], scene_corners[0], Scalar(0, 255, 0, 255), 4);
        int idxFound = found;
        found = -1;
        frame_counter = 0;
        return idxFound;
    }

    if (frame_counter != 1) {
        frame_counter++;
        frame_counter = frame_counter % frame_skipper;
```

```
    } else {
        frame_counter++;
        if (img.empty()) {
            return -1;
        }
        // Mat img_scene;
        // cvtColor(img, img_scene, cv::COLOR_BGR2GRAY);
        vector<KeyPoint> keypoints_scene;
        Mat descriptors_scene;
        fd_de->detectAndCompute(img, noArray(), keypoints_scene, descriptors_scene);
        for (int i = 0; i < imagesTemplate.size(); i++) {
            if (matchedImage(i, img, keypoints_scene, descriptors_scene)) {
                found = i;
                break;
            }
        }

    }
    return -1;
}
}
```

native_opencv.cpp / RecognitionCamera.mm

4.1.9 Exhibit Info Activity / View

Σε αυτήν την κλάση εμφανίζεται η εικόνα του εκθέματος που αναγνωρίστηκε το όνομα του και οι πληροφορίες που το αφορούν. Με το κουμπί πίσω ο χρήστης μπορεί να επιστρέψει στην κλάση Ar Activity για την αναγνώριση κάποιου άλλου εκθέματος.

4.2 Backend Controllers

4.2.1 MuseumListController

Ο ελεγκτής αυτός είναι υπεύθυνος να επιστρέφει μια λίστα με μουσεία με βάση την αναζήτηση του χρήστη στην κλάση Search Fragment.

4.2.2 MuseumLongInfoController

Ο ελεγκτής αυτός είναι υπεύθυνος να επιστρέψει τις αναλυτικές πληροφορίες του μουσείου στην κλάση Museum Description Activity

4.2.3 MuseumMapController

Ο ελεγκτής αυτός είναι υπεύθυνος να επιστρέφει την λίστα των μουσείων στον χάρτη της κλάσης MyMap Fragment

4.2.4 MuseumShortInfoController

Ο ελεγκτής αυτός επιστρέφει τον τίτλο, την διεύθυνση και μια μικρή φωτογραφία ενός μουσείου όταν επιλεγεί στον χάρτη από τον χρήστη.

4.2.5 Signup Controller

Ο ελεγκτής αυτός είναι υπεύθυνος για τις εγγραφές νέων χρηστών μέσω της κλάσης Signup Activity

4.2.6 LoginController - LoginWithFacebookController – LoginWithGoogleController

Οι ελεγκτές αυτοί είναι υπεύθυνοι για τις συνδέσεις των χρηστών (Απλή σύνδεση – μέσω Facebook – μέσω Google)

4.2.7 ChangeInfoController - ChangePhotoController

Ο ελεγκτής αυτός είναι υπεύθυνος για τις αλλαγές στις πληροφορίες του προφίλ (όνομα, επώνυμο, email και φωτογραφία) μέσω της κλάσης Profile Fragment.

4.2.8 GetSocialPhotoController

Τέλος ο ελεγκτής αυτός είναι υπεύθυνος για την λήψη και αποθήκευση στην βάση των φωτογραφιών των λογαριασμών Facebook και Google.

4.3 Βάση δεδομένων

Οντότητες

4.3.1 Users

Id: String	Ο αύξων αριθμός κλειδί
firstName: String	Το όνομα του χρήστη
lastName: String	Το επώνυμο του χρήστη
username: String	Το username του χρήστη
accessToken: String	Δημιουργείται για την επαλήθευση των στοιχείων του χρήστη στο backend κατά την σύνδεση
photoUrl: String	Ο υπερσύνδεσμος για την φωτογραφία προφίλ του χρήστη που είναι αποθηκευμένη στο firebase
socialPhotoUrl: String	Ο υπερσύνδεσμος για την φωτογραφία προφίλ του χρήστη στο Facebook ή στο Google
lastLoginDate: Date	Ημερομηνία τελευταίας σύνδεσης
firstLoginDate: Date	Ημερομηνία πρώτης σύνδεσης
password: String	Ο κωδικός του χρήστη

4.3.2 Museum

Id: String	Ο αύξων αριθμός κλειδί
name: MLString	Το όνομα του μουσείου
smallImageFileName: String	Το όνομα του αρχείου της μικρής εικόνας του μουσείου ώστε να βρεθεί στο firebase
largeImageFileName: String	Το όνομα του αρχείου της μεγάλης εικόνας του μουσείου ώστε να βρεθεί στο firebase
location: Location	Η τοποθεσία του μουσείου
description: MLString	Η περιγραφή του μουσείου

openingDaysAndHours: MLString	Οι ώρες και μέρες λειτουργίας του μουσείου
coordinates: Coordinates	Οι συντεταγμένες του μουσείου
ticketPrice: MLString	Οι τιμές του εισιτηρίου του μουσείου
websiteLink: String	Η ιστοσελίδα του μουσείου
imageInfo: List<ImageInfo>	Λίστα με τις πληροφορίες των εικόνων προς αναγνώριση

4.3.3 MLString

english: String	Αγγλική μετάφραση
greek: String	Ελληνική μετάφραση

4.3.4 Location

address: MLString	Διεύθυνση
addressNumber: int	Αριθμός διεύθυνσης
city: MLString	Πόλη
postalCode: String	Ταχυδρομικός κώδικας

4.3.5 Coordinates

latitude: float	Γεωγραφικό πλάτος
longitude: float	Γεωγραφικό μήκος

4.3.6 ImageInfo

imageName: String	Όνομα αρχείου εικόνας εκθέματος
imageTitleML: MLString	Τίτλος εκθέματος (Δύο γλώσσες)
imageTitleL: String	Τίτλος εκθέματος

imageTextML: MLString	Πληροφορίες εκθέματος (Δύο γλώσσες)
imageTextL: String	Πληροφορίες εκθέματος

5. Αποτίμηση

Μέρος αυτής της εργασίας είναι επίσης η αξιολόγηση των τεχνικών ιδιοτήτων της εφαρμογής.

5.1 Εξοπλισμός που χρησιμοποιήθηκε

Κατά την εξέταση της απόδοσης, χρησιμοποιήθηκαν δύο συσκευές. Παρακάτω παρατίθενται οι πιο σημαντικές προδιαγραφές.

iPhone 11

- Apple A13 Bionic (7 nm+) Hexa-core (2x2.65 GHz Lightning + 4x1.8 GHz Thunder)
- 12 MP, f/1.8, 26mm camera
- RAM 4GB
- iOS 13.2
- GSM, Wi-Fi, GPS, Accelerometer, Gyroscope, Geomagnetic Sensor, Rotation Vector Sensor

Samsung Galaxy S10

- Exynos 9820 (8 nm) Octa-core (2x2.73 GHz Mongoose M4 & 2x2.31 GHz Cortex-A75 & 4x1.95 GHz Cortex-A55)
- 12 MP, f/1.5-2.4, 26mm
- RAM 8GB
- Android 9.0 (Pie)
- GSM, Wi-Fi, GPS, Accelerometer, Gyroscope, Geomagnetic Sensor, Rotation Vector Sensor

5.2 Αξιολόγηση Επιδόσεων

Δεδομένου ότι μία από τις σημαντικότερες απαιτήσεις ήταν η απόδοση σε πραγματικό χρόνο, η ταχύτητα της ανεπτυγμένης λύσης έχει δοκιμαστεί τόσο σε τεχνητά δεδομένα όσο και σε πραγματικές εισροές δεδομένων. Η απόδοση μετρήθηκε καταγράφοντας τις μετρήσεις στην μνήμη της συσκευής, κάτι που έχει ελάχιστη επίδραση στα αποτελέσματα.

5.2.1 Αναγνώριση Αντικειμένων με όραση Η/Υ

Για την αποτίμηση του αλγορίθμου αναγνώρισης εικόνας, διενεργήθηκαν δύο ειδών δοκιμές.

5.2.1.1 Δοκιμή Ταχύτητας

Κατά την πρώτη δοκιμή, μετρήθηκε η διάρκεια σε δευτερόλεπτα, με ακρίβεια ενός μSecond, της εκτέλεσης των υπολογισμών της αναγνώρισης εικόνας. Το δείγμα των εικόνων ήταν 12 τυχαίοι πίνακες ζωγραφικής. Στο δείγμα αυτό έγιναν μετρήσεις σε 2 περιβάλλοντα φωτισμού, σε συνθήκες φωτός δωματίου, και σε σκοτεινό περιβάλλον. Σε κάθε περιβάλλον έγιναν 50 μετρήσεις.

Μια ένδειξη της ταχύτητας του αλγορίθμου αναγνώρισης είναι ο μέσος όρος των μετρήσεων αυτών, ο οποίος ήταν:

Περιβάλλον	Φως	Σκοτάδι
Μέσος χρόνος Αναγνώρισης (sec)	0.773035	0.798199

Η μέση διάρκεια των υπολογισμών, όπως προκύπτει από τις μετρήσεις, καθιστά την εκτέλεσή τους εφικτή σε πραγματικό χρόνο, εφαρμόζοντας τους υπολογισμούς σε κάθε δέκατο καρέ.

5.2.1.2 Δοκιμή Ευστοχίας

Κατά την δοκιμή αυτή, δοκιμάστηκε ο αλγόριθμος σε καρέ εικόνας προερχόμενα από την κάμερα της κινητής συσκευής, ώστε να διαπιστωθεί το ποσοστό αστοχίας (false positive) του αλγορίθμου. Τα καρέ που εξετάστηκαν προέρχονταν κατά 50% από φωτεινά περιβάλλοντα και κατά 50% από σκοτεινά περιβάλλοντα και το πλήθος τους ήταν 600 καρέ.

Στο δείγμα αυτό ο συνολικός αριθμός των false positive ήταν 8 καρέ, 6 από τα οποία προήλθαν από σκοτεινά καρέ και τα 2 από φωτεινά. Ποσοστιαία αυτό μεταφράζεται σε 1.33% false positive, 0.67% στο φως και 2% στο σκοτάδι, ποσοστά τα οποία είναι αποδεκτά στα πλαίσια αυτής της εργασίας. Η αστοχία αυτή οφείλεται στον θόρυβο που εισέρχεται στα καρέ της κάμερας σε συνθήκες χαμηλού φωτισμού, ο οποίος εισάγει θόρυβο και στους πίνακες των περιγραφέων χαρακτηριστικών.

6. Επίλογος

6.1 Μελλοντική Εργασία

Παρόλο που έχει ήδη γίνει πολλή δουλειά σε αυτήν την εφαρμογή, υπάρχουν ακόμα περιοχές που μπορούν / πρέπει να αλλάξουν ή να ενισχυθούν.

Μιλώντας για την συνολική εμπειρία του AR, θα μπορούσε να ενισχυθεί εισάγοντας την παρακολούθηση θέσης των αντικειμένων μετά την αναγνώρισή τους με την χρήση των κατάλληλων αλγορίθμων. Στην συνέχεια θα μπορούσε η πληροφορία που αφορά το αναγνωρισμένο αντικείμενο να υπερτίθεται στην ροή της εικόνας της κάμερας. Για την βελτίωση της ακρίβειας καθώς και της ταχύτητας της αναγνώρισης των αντικειμένων, στο μέλλον θα έπρεπε ακόμα να γίνει χρήση ταξινομητή (classifier) όπως ο Haar, ο οποίος κάνει χρήση νευρωνικού δικτύου, στο οποίο θα γινόταν εκμάθηση με το κατάλληλο σύνολο εικόνων. Σε αυτήν την περίπτωση θα έπρεπε να επανεξεταστούν οι διάφοροι αλγόριθμοι υπό διάφορες συνθήκες, για να βρεθεί η βέλτιστη λύση.

Η εφαρμογή που αναπτύξαμε έχει φτιαχτεί με τέτοιο τρόπο ώστε να μπορεί να ενσωματώσει με πολύ εύκολο τρόπο αυτήν την λειτουργία καθώς ο κώδικας που υλοποιεί τον αλγόριθμο αναγνώρισης εικόνας είναι πλήρως διαχωρισμένος από την υπόλοιπη εφαρμογή (modular) και ενεργοποιείται με μια κλήση συνάρτησης από το κύριο μέρος της, επιστρέφοντας σε αυτήν το ID της εικόνας που βρέθηκε.

Μια άλλη λειτουργία επίσης θα μπορούσε να είναι η αναγνώριση αντικειμένων από αποθηκευμένη φωτογραφία, ώστε να γίνεται αναγνώριση σε δεύτερο χρόνο, σε περίπτωση που δεν υπάρχει πρόσβαση στο διαδίκτυο.

Στο επίπεδο της αρχιτεκτονικής της εφαρμογής, θα μπορούσε να γίνει δουλειά μελλοντικά για την βελτίωση της επικοινωνίας του Backend Layer με το Database Layer, όπως και της επικοινωνίας των Native Mobile εφαρμογών με το Backend Layer. Πιο συγκεκριμένα, το Backend θα μπορούσε να διατηρεί ανοιχτό ένα σύνολο συνδέσεων με την βάση δεδομένων και να τα επαναχρησιμοποιεί για κάθε HTTP request που δέχεται. Επίσης, τα native mobile apps θα μπορούσαν να επικοινωνούν

με το Backend Layer με την χρήση WebSocket, αντί να ανοίγουν ξεχωριστό HTTP connection κάθε φορά. Και οι δύο αυτές βελτιώσεις θα είχαν ως αποτέλεσμα την ταχύτερη εξυπηρέτηση του χρήστη κατά την διάρκεια της χρήσης της εφαρμογής, δημιουργώντας μικρότερο δικτυακό φορτίο, καθώς επίσης και την δυνατότητα εξυπηρέτησης μεγαλύτερου αριθμού χρηστών την ίδια χρονική στιγμή (user concurrency - scalability). Μια ακόμα ιδέα βελτίωσης είναι, σε περίπτωση που η αναγνώριση εικόνας γίνει αρκετά πιο περίπλοκος υπολογισμός από ό,τι τώρα, να μεταφερθεί στο Backend Layer και με έναν μηχανισμό που θα κρατάει τα δεδομένα στην προσωρινή μνήμη (cache), να συνδυάζεται με την παρακολούθηση που θα γίνεται στον client (mobile natives).

Τελευταίο αλλά εξίσου σημαντικό είναι η πρακτική χρήση της εφαρμογής. Διάφορες επιλογές εξετάζονται:

- Περισσότερα μουσεία και αντικείμενα να εισαχθούν στην βάση δεδομένων της εφαρμογής
- Να δημοσιευθεί η εφαρμογή στο Google Play Store (Android) και στο App Store (iOS)
- Να προσφερθεί η εφαρμογή για χρήση σε μουσεία της Ελλάδας.

6.2 Συμπεράσματα

Σκοπός αυτής της εργασίας ήταν η δημιουργία μιας εφαρμογής AR για iOS και Android, που μπορεί να χρησιμοποιηθεί στον πολιτισμικό τουρισμό.

Μέσα από την εργασία παρουσιάστηκαν ο Ορισμός και η Ταξινόμηση, η σύντομη ιστορία και το πιθανό μέλλον του AR καθώς και μερικά υφιστάμενα έργα και λύσεις. Μετά από την επισκόπηση της τεχνολογίας του εντοπισμού και παρακολούθησης και την παρουσίαση τεχνολογιών, παρουσιάστηκε το λογισμικό που αναπτύχθηκε και αποτιμήθηκε.

Κατά τη διάρκεια της ανάπτυξης, εμφανίστηκαν αρκετά προβλήματα που μπορεί να κατηγοριοποιηθούν σε δύο ομάδες. Τα προβλήματα που σχετίζονται με την ανάπτυξη εφαρμογών σε iOS και Android, τα οποία προκλήθηκαν από την έλλειψη εμπειρίας μας σχετικά με τις πλατφόρμες στα αρχικά στάδια της ανάπτυξης (π.χ. διαρροή μνήμης των εφαρμογών, χειρισμός συνηθισμένων αποτυχιών του κώδικα). Η δεύτερη ομάδα περιέχει προβλήματα με βιβλιοθήκες ανοιχτού κώδικα, και κυρίως το OpenCV. Παρόλο που συνέβαλαν στην επιτάχυνση της ανάπτυξης, προκάλεσαν επίσης και προβλήματα και εισήγαγαν περιορισμούς στον κώδικα και την λειτουργικότητα.

Ωστόσο, η τελική εφαρμογή είναι σταθερή, με σχετικά καλή απόδοση στην αναγνώριση εικόνας.

Αναφορές

[1] Maxwell, K., 2010. Macmillan Dictionary: BuzzWord.

[2] Geroimenko, V., 2012. Augmented Reality Technology and Art: The Analysis and Visualization of Evolving Conceptual Models. Montpellier, France, IEEE Computer Society

[3] Azuma, R. T., 1997. A survey of augmented reality. Presence

[4] Rekimoto, J. & Nagao, K., 1995. The world through the computer: Computer augmented interaction with real world environments. Pittsburgh, Pennsylvania, USA, ACM

[5] Milgram, P. & Kishino, F., 1994. A taxonomy of mixed reality visual displays. IEICE TRANSACTIONS on Information and Systems

[6] Mann, S., 2002. Mediated reality with implementations for everyday life.

[7] Braz, J. M. & Pereira, J. M., 2008. TARCAST: Taxonomy for augmented reality CASTing with web support. The International Journal of Virtual Reality

[8] Hugues, O., Fuchs, P. & Nannipieri, O., 2011. New augmented reality taxonomy: Technologies and features of augmented environment. In: Handbook of augmented reality

[9] Normand, J.-M., Servières, M. & Moreau, G., 2012. A new typology of augmented reality applications. Megève, France

[10] Brooker, J., 2007. The polytechnic ghost. Early Popular Visual Culture

[11] Sutherland, I. E., 1965. The Ultimate Display

[12] Furness, T., 1969. The application of head-mounted displays to airborne reconnaissance and weapon delivery. Ohio, Wright-Patterson Air Force Base.

[13] Fisher, S. S., McGreevy, M., Humphries, J. & Robinett, W., 1986. Virtual Environment Display System. Chapel Hill, North Carolina, USA

[14] Caudell, T. P. & Mizell, D. W., 1992. Augmented reality: an application of heads-up display technology to manual manufacturing processes. Kauai, HI, USA, IEEE

[15] Statista, 2019. http://www.statista.com

[16] Gartner Inc. Gartner's Hype Cycle for Emerging Technologies Identifies the Computing Innovations That Organizations Should Monitor

[17] Billinghurst, M., 2015. Augmented Reality: The Next 20 Years

[18] Edward, S. & Gabbard, J., 2005. Survey of User-Based Experimentation in Augmented Reality. Las Vegas, NV, USA, IEEE

[19] Höllerer, S. F. a. B. M. a. T., Feiner, S., MacIntyre, B. & Höllerer, T., 1997. A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring the Urban Environment. Cambridge, MA, USA, IEEE

[20] Thomas, B. et al., 2000. ARQuake: An Outdoor/Indoor Augmented Reality First Person Application. Atlanta, GA, USA, IEEE Computer Society

[21] Vlahakis, V. et al., 2002. Archeoguide: an augmented reality guide for archaeological sites. Computer Graphics and Applications

[22] Zoellner, M. et al., 2008. Reality Filtering: A Visual Time Machine in Augmented Reality. Aire-la-Ville, Switzerland, Eurographics Association

[23] Mata, F. & Claramunt, C., 2014. A social navigation guide using augmented reality. Dallas, Texas, USA

[24] Gee, A. P. et al., 2013. Augmented crime scenes: virtual annotation of physical environments for forensic investigation. Firenze, Italy

[25] Reitmayr, G. & Drummond, T. W., 2006. Going out: robust model-based tracking for outdoor augmented reality. Santa Barbara, CA, USA, IEEE

[26] Huang, J., Takashima, K., Hashi, S. & Kitamura, Y., 2014. IM3D: magnetic motion tracking system for dexterous 3D interactions. Vancouver, Canada

[27] Billinghurst, M. & Kato, H., 1999. Marker Tracking and HMD Calibration for a Video-based Augmented Reality Conferencing System. San Francisco, CA, USA, IEEE

[28] Harris, C. & Stennett, C., 1990. RAPID - A Video Rate Object Tracker. Oxford, UK, Proc. British Machine Vision Conf.

[29] Kragic, D. & Christensen, H. I., 2002. Model Based Techniques for Robotic Servoing and Grasping. EPFL, Switzerland, IEEE

[30] Leonard, J. J. & Durrant-Whyte, H. F., 1991. Simultaneous Map Building and Localization for an Autonomous Mobile Robot. Osaka, Japan, IEEE

[31] Klein, G. & Murray, D., 2007. Parallel Tracking and Mapping for Small AR Workspaces. Nara, Japan, IEEE Computer Society

[32] Lucas, B. D. & Kanade, T., 1981. An Iterative Image Registration Technique with an Application to Stereo Vision. San Francisco, CA, USA, Morgan Kaufmann Publishers Inc

[33] Pressigout, M., Marchand, É. & Mémin, É., 2008. Hybrid Tracking Approach Using Optical Flow and Pose Estimation. San Diego, CA, USA, IEEE

[34] William J. Hughes Technical Center , 2014. Global Positioning System (GPS), Standard Positioning Service (SPS), Performance Analysis Report #86, Atlantic City: NSTB/WAAS T&E Team

[35] Sapiezynski, P., Stopczynski, A., Gatej, R. & Lehmann, S., 2015. Tracking Human Mobility using WiFi signals. PLoS ONE

[36] Adib, F. & Katabi, D., 2013. See Through Walls with Wi-Fi!. Hong Kong, China, ACM

[37] Piekarski, W., Avery, B., Thomas, B. H. & Malbezin, P., 2003. Hybrid Indoor and Outdoor Tracking for Mobile 3D Mixed Reality. Washington, DC, USA, IEEE

Βιβλιογραφία

Introduction to Harris Corner Detector, https://medium.com/@deepanshut041/introduction-to-harris-corner-detector-32a88850b3f6?

Harris Corner Detection and Shi-Tomasi Corner Detection, https://medium.com/pixel-wise/detect-those-corners-aba0f034078b

Introduction to SIFT (Scale-Invariant Feature Transform), https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_sift_intro/py_sift_intro.html

Introduction to SURF (Speeded-Up Robust Features), https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_surf_intro/py_surf_intro.html#surf

FAST Algorithm for Corner Detection, https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_fast/py_fast.html#fast

Edward Rosten and Tom Drummond: Machine learning for high-speed corner detection

BRIEF (Binary Robust Independent Elementary Features), https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_brief/py_brief.html#brief

ORB (Oriented FAST and Rotated BRIEF), https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_orb/py_orb.html#orb

Feature Matching, https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_matcher/py_matcher.html#matcher

Wikipedia, https://www.wikipedia.org/