ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«Δημιουργία εφαρμογής android για ανίχνευση αντικειμένων μέσω κάμερας με την βοήθεια της τεχνητής νοημοσύνης, υπολογισμό της απόστασης τους από τον χρήστη και ενημέρωση του χρήστη μέσω φωνητικής και απτικής αλληλεπίδρασης.»

Του φοιτητή
Παναγιωτίδη Λέανδρου
Αρ. Μητρώου: 2020246

Επιβλέπων
Κοκκώνης Γεώργιος
Επίκουρος Καθηγητής

Θεσσαλονίκη 2026

Δημιουργία εφαρμογής android για ανίχνευση αντικειμένων μέσω κάμερας με την βοήθεια της τεχνητής νοημοσύνης, υπολογισμό της απόστασης τους από τον χρήστη και ενημέρωση του χρήστη μέσω φωνητικής και απτικής αλληλεπίδρασης.

Κωδικός: 25221

Ονοματεπώνυμο φοιτητή Παναγιωτίδης Λέανδρος

Ονοματεπώνυμο εισηγητή Κοκκώνης Γεώργιος

Ημερομηνία ανάληψης Δ.Ε. 04-04-2025

Ημερομηνία περάτωσης Δ.Ε. 23-1-2026

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Η εργασία αυτή είναι αφιερωμένη στους γονείς μου.

Πρόλογος

Η επιλογή του θέματος της παρούσας διπλωματικής εργασίας πηγάζει από το έντονο ενδιαφέρον μου για τις ραγδαίες εξελίξεις στον τομέα της Τεχνητής Νοημοσύνης και, ειδικότερα, για τη δυνατότητα εφαρμογής τους προς όφελος του κοινωνικού συνόλου. Η υποβοηθητική τεχνολογία για άτομα με οπτική αναπηρία αποτελεί ένα πεδίο όπου η Μηχανική Όραση (Computer Vision) καλείται να δώσει ουσιαστικές λύσεις στο κρίσιμο ζήτημα της αυτόνομης διαβίωσης.

Μέσα από τη διαδικασία εκπόνησης της εργασίας, αποκόμισα σημαντικά οφέλη σε πολλαπλά επίπεδα. Αφενός, εμβάθυνα στο θεωρητικό υπόβαθρο των Συνελικτικών Νευρωνικών Δικτύων και κατανόησα τις σύγχρονες αρχιτεκτονικές ανίχνευσης αντικειμένων (YOLO). Αφετέρου, αντιμετώπισα στην πράξη τις τεχνικές προκλήσεις της ανάπτυξης λογισμικού για κινητές συσκευές (Edge AI), μαθαίνοντας να ισορροπώ την υπολογιστική ακρίβεια με την ταχύτητα εκτέλεσης.

Κυρίως όμως, η ενασχόληση με ζητήματα προσβασιμότητας μου επέτρεψε να αντιληφθώ τη βαρύτητα που φέρει ο ρόλος του μηχανικού στη σχεδίαση συμπεριληπτικών συστημάτων, τα οποία δεν αποκλείουν καμία ομάδα χρηστών.

Περίληψη

Η παρούσα διπλωματική εργασία πραγματεύεται τη σχεδίαση και ανάπτυξη μιας εφαρμογής Android για την υποβοήθηση της πλοήγησης ατόμων με οπτική αναπηρία σε εσωτερικούς χώρους. Ενώ η μετακίνηση σε εξωτερικό περιβάλλον υποστηρίζεται επαρκώς από το GPS, η αυτόνομη πλοήγηση σε κτίρια παραμένει μια ανοιχτή πρόκληση λόγω της πολυπλοκότητας και της έλλειψης σήματος. Η προτεινόμενη λύση αξιοποιεί τεχνολογίες Edge AI για να λειτουργεί αποκλειστικά στη συσκευή, εξασφαλίζοντας μηδενική εξάρτηση από το διαδίκτυο και ελαχιστοποίηση του χρόνου απόκρισης.

Στο πλαίσιο της έρευνας, δημιουργήθηκε ένα εξειδικευμένο σύνολο δεδομένων για αντικείμενα εσωτερικού χώρου (πόρτες, σκάλες, έπιπλα) και εκπαιδεύτηκε το σύγχρονο μοντέλο YOLOv11 Nano. Καινοτομία της εργασίας αποτελεί η υλοποίηση ενός γεωμετρικού αλγορίθμου εκτίμησης απόστασης (Geometric Distance Estimation), ο οποίος συνδυάζει την υπολογιστική όραση με τους αισθητήρες προσανατολισμού (IMU) της συσκευής, επιτρέποντας τον ακριβή εντοπισμό εμποδίων χωρίς τη χρήση ειδικού εξοπλισμού (LiDAR).

Πραγματοποιήθηκε συγκριτική αξιολόγηση (Benchmarking) μεταξύ των αρχιτεκτονικών YOLOv11n (Detection). Τα πειραματικά αποτελέσματα κατέδειξαν ότι το μοντέλο Detection προσφέρει τη βέλτιστη ισορροπία, επιτυγχάνοντας ρυθμό ανανέωσης ~30 FPS σε κινητές συσκευές μεσαίας κατηγορίας.Η τελική εφαρμογή ενσωματώνει διεπαφή "Touch-to-Explore" με πολυτροπική ανάδραση (ομιλία και δόνηση), προσφέροντας μια ασφαλή και διαισθητική εμπειρία χρήσης.

«Development of an android application for detecting objects via camera with the help of artificial intelligence, calculating their distance from the user and informing the user through voice and haptic interaction.»

Panagiotidis Leandros

Abstract

This thesis addresses the design and development of an Android application to assist visually impaired individuals with indoor navigation. While outdoor mobility is largely supported by GPS, autonomous navigation within buildings remains a significant challenge due to environmental complexity and lack of satellite signal. The proposed solution leverages Edge AI technologies to operate entirely on-device, ensuring zero dependency on internet connectivity and minimizing response latency.

Within the scope of this research, a custom dataset for indoor objects (doors, stairs, furniture) was curated, and the state-of-the-art YOLOv11 Nano model was trained. A key innovation of this work is the implementation of a Geometric Distance Estimation algorithm, which fuses Computer Vision with the device's inertial sensors (IMU), enabling accurate obstacle localization without requiring specialized hardware such as LiDAR.

A comparative evaluation (Benchmarking) was conducted between YOLOv11n (Detection). Experimental results demonstrated that the Detection model offers the optimal balance, achieving a refresh rate of ~30 FPS on mid-range mobile devices. The final application integrates a "Touch-to-Explore" interface with multimodal feedback (speech and haptics), providing a safe and intuitive user experience.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω την οικογένειά μου που με στηρίζει σε όλη τη διάρκεια της ζωής μου.

Περιεχόμενα

Κατάλογος Εικόνων

Κατάλογος Πινάκων

Συντομογραφίες

Δ.Ε.	Διπλωματική Εργασία
ΔΙΠΑΕ	Διεθνές Πανεπιστήμιο Ελλάδος
Π.Ε.	Πτυχιακή Εργασία
YOLO	You Only Look Once
COCO	Common Objects in Context

Κεφάλαιο 1ο: Προβλήματα Όρασης και Εφαρμογές Τεχνητής Νοημοσύνης

1.1 Εισαγωγή

Η όραση αποτελεί την κυρίαρχη αίσθηση μέσω της οποίας ο άνθρωπος αντιλαμβάνεται το περιβάλλον, επεξεργάζεται χωρικές πληροφορίες και αλληλεπιδρά με τον κόσμο. Η απώλεια ή η σημαντική μείωση αυτής της ικανότητας δημιουργεί θεμελιώδη εμπόδια στην ανεξάρτητη διαβίωση, την κινητικότητα και την κοινωνική ενσωμάτωση. Σύμφωνα με τα πλέον πρόσφατα δεδομένα του Παγκόσμιου Οργανισμού Υγείας (WHO), η οπτική αναπηρία αποτελεί ένα μείζον παγκόσμιο ζήτημα δημόσιας υγείας. Εκτιμάται ότι τουλάχιστον 2,2 δισεκατομμύρια άνθρωποι παγκοσμίως ζουν με κάποια μορφή διαταραχής της κοντινής ή μακρινής όρασης . Από αυτούς, περίπου 39 εκατομμύρια είναι τυφλοί, ενώ εκατοντάδες εκατομμύρια αντιμετωπίζουν μέτρια έως σοβαρή οπτική βλάβη [1].

Η δημογραφική αυτή πραγματικότητα, σε συνδυασμό με τη γήρανση του πληθυσμού, καθιστά επιτακτική την ανάγκη για τεχνολογικές λύσεις που ενισχύουν την αυτονομία. Μία από τις μεγαλύτερες προκλήσεις που αντιμετωπίζουν τα άτομα με προβλήματα όρασης (Visually Impaired People - VIP) είναι η **πλοήγηση σε εσωτερικούς χώρους** (Indoor Navigation) . Ενώ η μετακίνηση σε εξωτερικούς χώρους έχει διευκολυνθεί σημαντικά από την τεχνολογία GPS (Global Positioning System), οι εσωτερικοί χώροι παραμένουν μια "γκρίζα ζώνη". Σε περιβάλλοντα όπως δημόσια κτίρια, πανεπιστήμια, νοσοκομεία ή ακόμη και άγνωστες κατοικίες, το σήμα GPS είναι αδύναμο ή ανύπαρκτο και η ακρίβεια των χαρτών ανεπαρκής [2].

Επιπλέον, οι εσωτερικοί χώροι χαρακτηρίζονται από δυναμικότητα. Αντικείμενα όπως καρέκλες, τραπέζια, ανοιχτές πόρτες ή ανθρώπινη παρουσία αλλάζουν διαρκώς θέση, δημιουργώντας απρόβλεπτα εμπόδια. Για έναν τυφλό χρήστη, η έλλειψη γνώσης για τη διάταξη του χώρου οδηγεί συχνά σε αισθήματα ανασφάλειας, φόβο σύγκρουσης και εξάρτηση από συνοδούς για την εκτέλεση καθημερινών δραστηριοτήτων . Η αδυναμία εντοπισμού βασικών σημείων ενδιαφέροντος (π.χ. "πού είναι η πόρτα;", "πού υπάρχει ελεύθερη θέση;") περιορίζει δραστικά την αυτενέργεια.

1.2 Ορισμός Προβλήματος και Κενά στις Υπάρχουσες Λύσεις

Η παραδοσιακή προσέγγιση για την κινητικότητα των τυφλών βασίζεται κυρίως στο λευκό μπαστούνι (white cane) και στους σκύλους-οδηγούς. Παρότι τα μέσα αυτά είναι αναντικατάστατα, έχουν εγγενείς περιορισμούς στην παροχή περιβαλλοντικής πληροφορίας:

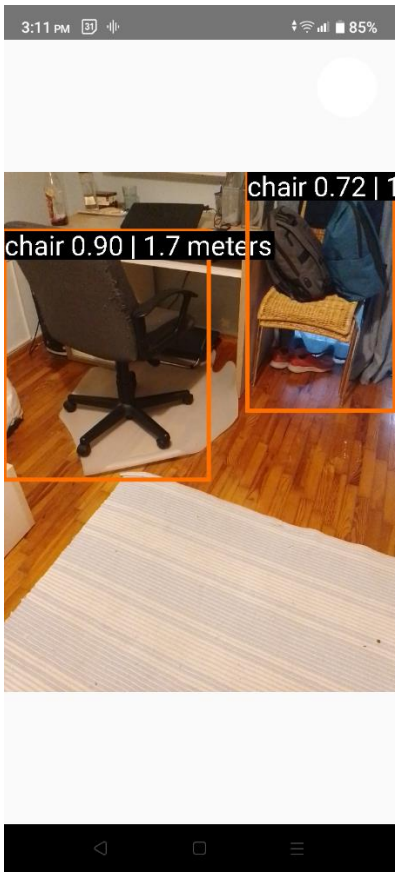
- **Το Λευκό Μπαστούνι:** Προσφέρει ανίχνευση εμποδίων κυρίως στο επίπεδο του εδάφους και σε μικρή απόσταση (περίπου 1 μέτρο μπροστά από τον χρήστη). Ενημερώνει τον χρήστη ότι "κάτι υπάρχει", αλλά αδυνατεί να προσδιορίσει την ταυτότητα του εμποδίου (σημασιολογική πληροφορία) ή να ανιχνεύσει εμπόδια που προεξέχουν στο ύψος του κορμού ή της κεφαλής (π.χ. ανοιχτά παράθυρα, επιτοίχιες πινακίδες), τα οποία αποτελούν σοβαρό κίνδυνο τραυματισμού .

- **Ο Σκύλος-Οδηγός:** Είναι εξαιρετικός στην αποφυγή εμποδίων και την εύρεση συγκεκριμένων στόχων (π.χ. έξοδος), αλλά δεν μπορεί να μεταφέρει λεπτομερείς πληροφορίες για το περιβάλλον (π.χ. "υπάρχει ένα τραπέζι στα δύο μέτρα δεξιά") [3].

Στον τομέα των ψηφιακών βοηθημάτων, έχουν αναπτυχθεί διάφορες εφαρμογές υπολογιστικής όρασης (Computer Vision). Ωστόσο, η ανάλυση της βιβλιογραφίας και των υπαρχουσών λύσεων (State-of-the-Art) αποκαλύπτει σημαντικά τεχνολογικά κενά:

1. **Εξάρτηση από το Διαδίκτυο (Cloud Dependency):** Πολλές δημοφιλείς εφαρμογές, όπως το *Be My Eyes* (με τη λειτουργία Be My AI) ή παλαιότερες εκδόσεις του *Microsoft Seeing AI*, βασίζονται σε απομακρυσμένη επεξεργασία. Η εικόνα αποστέλλεται σε έναν διακομιστή (cloud) για ανάλυση. Αυτό δημιουργεί δύο προβλήματα: α) απαιτεί σταθερή σύνδεση στο διαδίκτυο, η οποία συχνά δεν υπάρχει σε υπόγεια ή εσωτερικούς χώρους κτιρίων, και β) εισάγει καθυστέρηση (latency). Σε σενάρια πλοήγησης, ακόμη και μια καθυστέρηση 1-2 δευτερολέπτων μπορεί να είναι επικίνδυνη, καθώς ο χρήστης ενδέχεται να έχει ήδη συγκρουστεί με το εμπόδιο πριν λάβει την ειδοποίηση [4] .
2. **Έλλειψη Ακριβούς Χωρικής Πληροφορίας:** Εφαρμογές όπως το *Google Lookout* (στη λειτουργία Explore) είναι εξαιρετικές στην αναγνώριση αντικειμένων ("καρέκλα", "τραπέζι"), αλλά συχνά δεν παρέχουν ακριβή εκτίμηση της απόστασης σε πραγματικό χρόνο. Η πληροφορία "βλέπω μια καρέκλα" είναι λιγότερο χρήσιμη από την πληροφορία "καρέκλα στα 2 μέτρα", η οποία επιτρέπει στον χρήστη να προγραμματίσει την κίνησή του [5].
3. **Υψηλό Υπολογιστικό Κόστος:** Η εκτέλεση προηγμένων αλγορίθμων Τεχνητής Νοημοσύνης (AI) σε κινητές συσκευές συχνά οδηγεί σε γρήγορη εξάντληση της μπαταρίας ή υπερθέρμανση (thermal throttling), καθιστώντας την εφαρμογή μη πρακτική για συνεχή χρήση.

Παρακάτω βλέπουμε ένα παράδειγμα εφαρμογής υπολογιστικής όρασης εικόνα 1.1.



Εικόνα 1.1: Παράδειγμα ανίχνευσης αντικειμένων

1.3 Σκοπός της Εργασίας: Μια Edge AI Λύση Πλοήγησης

Ο κύριος στόχος της παρούσας πτυχιακής εργασίας είναι η σχεδίαση και ανάπτυξη μιας πρότυπης εφαρμογής Android, η οποία γεφυρώνει το χάσμα μεταξύ της ανίχνευσης αντικειμένων και της ασφαλούς πλοήγησης τυφλών, αξιοποιώντας τεχνολογίες **Edge AI** (Τεχνητή Νοημοσύνη στο Άκρο). Η προτεινόμενη λύση εστιάζει στην εκτέλεση όλων των υπολογισμών τοπικά στη συσκευή, εξαλείφοντας την ανάγκη για σύνδεση στο διαδίκτυο και ελαχιστοποιώντας τον χρόνο απόκρισης.

Η καινοτομία της προτεινόμενης εφαρμογής έγκειται στους εξής άξονες:

- **Χρήση Μοντέλων YOLOv11 Nano (Edge AI):** Η εφαρμογή ενσωματώνει τα πλέον σύγχρονα μοντέλα βαθιάς μάθησης (Deep Learning) της οικογένειας YOLO (You Only Look Once), συγκεκριμένα την έκδοση **YOLOv11n**. Τα μοντέλα αυτά έχουν μετατραπεί σε μορφή **TensorFlow Lite (TFLite)** και έχουν υποστεί κβαντισμό (quantization) για βέλτιστη απόδοση σε κινητές συσκευές. Η επιλογή αυτή επιτρέπει την ανίχνευση αντικειμένων σε πραγματικό χρόνο (>20 FPS) ακόμη και σε συσκευές μεσαίας κατηγορίας, χωρίς τη χρήση δεδομένων κινητής τηλεφωνίας [6].
- **Αλληλεπίδραση Αφής και Ήχου (Haptic/Touch Interaction):** Αντί να βομβαρδίζει τον χρήστη με συνεχή ομιλία για όλα τα αντικείμενα του χώρου, η εφαρμογή υιοθετεί μια προσέγγιση εξερεύνησης. Ο χρήστης σέρνει το δάχτυλό του στην οθόνη της κάμερας (touch-to-explore). Όταν το δάχτυλο "αγγίξει" ένα ανιχνευμένο αντικείμενο στο ψηφιακό περιβάλλον, η συσκευή δονείται και εκφωνεί το όνομα και την απόσταση (π.χ. "Τραπέζι, 1.5 μέτρα"). Αυτό προσφέρει στον χρήστη ενεργητικό έλεγχο της αντίληψης του χώρου.
- **Μονοφθαλμική Εκτίμηση Απόστασης:** Η εργασία διερευνά τεχνικές για τον υπολογισμό της απόστασης των εμποδίων χρησιμοποιώντας μία μόνο κάμερα (monocular camera), καθιστώντας τη λύση συμβατή με τη συντριπτική πλειοψηφία των Android συσκευών, χωρίς την απαίτηση για ειδικούς αισθητήρες βάθους (ToF/LiDAR) [7].
- **Προσβασιμότητα Hands-Free:** Ενσωμάτωση υποστήριξης για το **Google Voice Access**, επιτρέποντας στον χρήστη να εκκινήσει και να ελέγξει την εφαρμογή αποκλειστικά με φωνητικές εντολές, διασφαλίζοντας πλήρη αυτονομία.
- **Συγκριτική Αξιολόγηση:** Στο πλαίσιο της έρευνας, πραγματοποιείται εκπαίδευση ενός προσαρμοσμένου συνόλου δεδομένων (Custom Dataset) για αντικείμενα εσωτερικού χώρου και συγκρίνονται δύο διαφορετικές αρχιτεκτονικές: **Object Detection** (YOLOv11n) εναντίον **Instance Segmentation** (YOLOv11n-seg). Η σύγκριση αυτή στοχεύει να απαντήσει στο ερευνητικό ερώτημα: *Αξίζει το επιπλέον υπολογιστικό κόστος της τμηματοποίησης (segmentation) για την ακρίβεια που προσφέρει στην πλοήγηση τυφλών;*

1.4 Δομή της Εργασίας

Η παρούσα πτυχιακή εργασία διαρθρώνεται ως εξής:

- **Κεφάλαιο 2: Υποβοηθητικές Τεχνολογίες & Βιβλιογραφική Ανασκόπηση:** Παρουσιάζει αναλυτικά το θεωρητικό υπόβαθρο των εργαλείων προσβασιμότητας (TalkBack, VoiceOver) και αναλύει τον ανταγωνισμό (Seeing AI, Lookout), εντοπίζοντας τα πλεονεκτήματα και τις αδυναμίες τους.

- **Κεφάλαιο 3: Θεωρητικό Υπόβαθρο Τεχνητής Νοημοσύνης:** Εισάγει τις έννοιες της Μηχανικής Όρασης, εξηγεί τη λειτουργία των Συνελικτικών Νευρωνικών Δικτύων (CNNs) και αναλύει τη διαφορά μεταξύ Detection και Segmentation.

- **Κεφάλαιο 4: Αρχιτεκτονική YOLO & Βελτιστοποίηση:** Εστιάζει στην τεχνική ανάλυση του αλγορίθμου YOLOv11, εξηγώντας καινοτομίες όπως το C2PSA και το Anchor-Free detection. Περιγράφει τη διαδικασία μετατροπής μοντέλων σε TFLite και τις τεχνικές κβαντισμού (FP16/Int8) για κινητά.

- **Κεφάλαιο 5: Μεθοδολογία & Υλοποίηση:** Περιγράφει βήμα-προς-βήμα την ανάπτυξη της εφαρμογής Android, την αρχιτεκτονική του λογισμικού, τη διαχείριση της κάμερας (CameraX) και τον αλγόριθμο υπολογισμού απόστασης.

- **Κεφάλαιο 6: Συμπεράσματα & Μελλοντικές Επεκτάσεις:** Συνοψίζει τα ευρήματα της έρευνας και προτείνει κατευθύνσεις για μελλοντική βελτίωση της εφαρμογής

Κεφάλαιο 2ο: Υποβοηθητικές Τεχνολογίες & Βιβλιογραφική Ανασκόπηση

2.1 Εισαγωγή: Η Δημογραφική και Κοινωνικοοικονομική Διάσταση της Οπτικής Αναπηρίας

Η οπτική αναπηρία δεν αποτελεί απλώς μια ιατρική διάγνωση αλλά μια πολύπλοκη κοινωνική και οικονομική πρόκληση που απαιτεί πολυεπίπεδη τεχνολογική παρέμβαση. Σύμφωνα με τα πλέον πρόσφατα δεδομένα του Παγκόσμιου Οργανισμού Υγείας (WHO), η κλίμακα του φαινομένου είναι τεράστια και αυξανόμενη. Παγκοσμίως, τουλάχιστον 2,2 δισεκατομμύρια άνθρωποι ζουν με κάποια μορφή διαταραχής της κοντινής ή μακρινής όρασης. Το στατιστικό αυτό μέγεθος, ωστόσο, αποκρύπτει μια σημαντική ανισότητα στην πρόσβαση σε υπηρεσίες υγείας και αποκατάστασης. Από αυτόν τον πληθυσμό, εκτιμάται ότι σε τουλάχιστον 1 δισεκατομμύριο περιπτώσεις, η οπτική βλάβη θα μπορούσε να έχει προληφθεί ή δεν έχει ακόμη αντιμετωπιστεί [1].

Η ανάλυση των αιτιών αποκαλύπτει ότι οι κυριότεροι παράγοντες για την οπτική βλάβη και την τύφλωση σε παγκόσμιο επίπεδο παραμένουν τα διαθλαστικά σφάλματα και ο καταρράκτης. Παρά την ύπαρξη καθιερωμένων ιατρικών παρεμβάσεων για αυτές τις παθήσεις, η κάλυψη των αναγκών παραμένει απογοητευτικά χαμηλή. Εκτιμάται ότι παγκοσμίως, μόνο το 36% των ατόμων με διαταραχή μακρινής όρασης λόγω διαθλαστικού σφάλματος και μόλις το 17% των ατόμων με οπτική βλάβη λόγω καταρράκτη έχουν λάβει την κατάλληλη θεραπευτική παρέμβαση. Αυτό το "κενό θεραπείας" (treatment gap) μεταφράζεται άμεσα σε ανάγκη για υποβοηθητικές τεχνολογίες που θα επιτρέψουν σε αυτά τα άτομα να λειτουργήσουν αυτόνομα σε ένα περιβάλλον σχεδιασμένο για βλέποντες [1].

Πέρα από την ανθρωπιστική διάσταση, το οικονομικό αποτύπωμα της οπτικής αναπηρίας είναι συντριπτικό. Το ετήσιο παγκόσμιο κόστος απώλειας παραγωγικότητας που σχετίζεται με την οπτική βλάβη εκτιμάται στα 411 δισεκατομμύρια δολάρια ΗΠΑ. Το ποσό αυτό αντικατοπτρίζει τον αποκλεισμό εκατομμυρίων ατόμων από την αγορά εργασίας και την εκπαίδευση, υπογραμμίζοντας την επιτακτική ανάγκη για ανάπτυξη τεχνολογικών λύσεων που θα ενισχύουν την επαγγελματική και κοινωνική ένταξη. Η βιβλιογραφία υποδεικνύει ότι η επένδυση σε στρατηγικές φροντίδας των ματιών και σε τεχνολογίες αποκατάστασης δεν είναι μόνο ζήτημα δημόσιας υγείας αλλά και οικονομικής βιωσιμότητας [1].

2.1.1 Ορισμοί και Λειτουργική Προσέγγιση της Τύφλωσης

Για τον ερευνητή των υποβοηθητικών τεχνολογιών, ο κλινικός ορισμός της τύφλωσης (π.χ. οπτική οξύτητα μικρότερη από 3/60 στο καλύτερο μάτι με διόρθωση) είναι συχνά ανεπαρκής. Η "λειτουργική όραση" (functional vision) αποτελεί μια πιο χρήσιμη έννοια για τον σχεδιασμό διεπαφών. Η Εθνική Ομοσπονδία Τυφλών (National Federation of the Blind - NFB) προτείνει έναν ευρύτερο ορισμό, ενθαρρύνοντας τα άτομα να θεωρούν τους εαυτούς τους τυφλούς εάν η όρασή τους είναι επαρκώς μειωμένη—ακόμη και με διορθωτικούς φακούς—ώστε να πρέπει να χρησιμοποιούν εναλλακτικές μεθόδους για να εκτελέσουν δραστηριότητες που οι βλέποντες θα εκτελούσαν οπτικά [8].

Αυτή η προσέγγιση ευθυγραμμίζεται με δεδομένα από την Εθνική Έρευνα Συνέντευξης Υγείας (NHIS), η οποία κατηγοριοποιεί την απώλεια όρασης βασιζόμενη στην αναφερόμενη δυσκολία ("λίγη δυσκολία" ή "πολλή δυσκολία") στην όραση, ακόμη και με τη χρήση γυαλιών. Επομένως, το κοινό-στόχος για τις

λύσεις κινητών συσκευών δεν περιορίζεται μόνο σε όσους έχουν ολική απώλεια φωτός, αλλά επεκτείνεται σε μια ευρεία δημογραφική ομάδα που, ενώ διατηρεί κάποια υπολειπόμενη όραση, αδυνατεί να πλοηγηθεί ή να διαβάσει κείμενο χωρίς ψηφιακή υποβοήθηση. Οι στατιστικές αυτές, που συχνά προκύπτουν από μετα-αναλύσεις πληθυσμιακών μελετών μεταξύ 1980 και 2018, καταδεικνύουν μια αυξητική τάση λόγω της γήρανσης του πληθυσμού, καθιστώντας την ανάπτυξη προσβάσιμων λύσεων πιο κρίσιμη από ποτέ [9].

2.1.2 Το Παράδοξο της Εσωτερική Πλοήγησης

Ενώ η τεχνολογία GPS έχει επιλύσει σε μεγάλο βαθμό το πρόβλημα του προσανατολισμού σε εξωτερικούς χώρους, η πλοήγηση σε εσωτερικούς χώρους (indoor navigation) παραμένει η "Αχίλλειος πτέρνα" της αυτόνομης διαβίωσης για τα άτομα με προβλήματα όρασης. Οι εσωτερικοί χώροι—όπως πανεπιστήμια, νοσοκομεία, εμπορικά κέντρα και αεροδρόμια—παρουσιάζουν μοναδικές προκλήσεις: απουσία δορυφορικού σήματος, πολύπλοκη αρχιτεκτονική διαρρύθμιση, δυναμικά εμπόδια (π.χ. έπιπλα, άνθρωποι) και έλλειψη τυποποιημένων οπτικών ή απτικών ενδείξεων [10].

Έρευνες που βασίζονται σε συνεντεύξεις με άτομα με οπτική αναπηρία αποκαλύπτουν ότι η πλοήγηση σε άγνωστα δημόσια κτίρια θεωρείται συχνά "απαγορευτική" για πρώτη φορά χωρίς συνοδό, οδηγώντας σε μείωση της αυτοπεποίθησης και κοινωνικό αποκλεισμό. Οι παραδοσιακές στρατηγικές, όπως η "ακτογραμμή" (shorelining - η παρακολούθηση ενός τοίχου με το λευκό μπαστούνι) ή η καταμέτρηση βημάτων, είναι γνωστικά απαιτητικές και επιρρεπείς σε σφάλματα σε ανοιχτούς χώρους. Επιπλέον, το λευκό μπαστούνι, ενώ αποτελεσματικό για την ανίχνευση εμποδίων στο επίπεδο του εδάφους, αδυνατεί να εντοπίσει εμπόδια στο ύψος της κεφαλής ή του κορμού (π.χ. προεξέχουσες πινακίδες, πυροσβεστήρες), θέτοντας τη σωματική ακεραιότητα του χρήστη σε κίνδυνο [11].

Οι ειδικοί σε θέματα τύφλωσης κατατάσσουν την εσωτερική πλοήγηση και τον προσανατολισμό ως τη δεύτερη σημαντικότερη λειτουργία για τα Ηλεκτρονικά Βοηθήματα Ταξιδιού (Electronic Travel Aids - ETAs), αμέσως μετά την εξωτερική πλοήγηση. Το ζητούμενο δεν είναι απλώς ο εντοπισμός θέσης (localization), αλλά η κατανόηση του πλαισίου: "Πού βρίσκεται η πόρτα;", "Πόσο μακριά είναι ο ανελκυστήρας;", "Υπάρχει εμπόδιο μπροστά μου;". Οι λύσεις που βασίζονται σε υποδομές (όπως beacons Bluetooth Low Energy ή Wi-Fi fingerprinting) απαιτούν σημαντική προσπάθεια εγκατάστασης και συντήρησης, καθιστώντας τις μη κλιμακώσιμες. Συνεπώς, η έρευνα στρέφεται πλέον σε λύσεις που βασίζονται αποκλειστικά στην κινητή συσκευή του χρήστη (on-device solutions), αξιοποιώντας τις ενσωματωμένες κάμερες και τους αισθητήρες για την κατανόηση του περιβάλλοντος [12].

2.2 Εργαλεία Προσβασιμότητας και Λειτουργικά Συστήματα

Η βάση κάθε λύσης για κινητές συσκευές είναι το πλαίσιο προσβασιμότητας (Accessibility Framework) που παρέχει το λειτουργικό σύστημα. Τα εργαλεία αυτά, όπως το TalkBack στο Android και το VoiceOver στο iOS, δεν είναι απλώς αναγνώστες οθόνης (screen readers), αλλά πολύπλοκα στρώματα αλληλεπίδρασης που παρεμβάλλονται μεταξύ του χρήστη και της γραφικής διεπαφής (GUI), μετατρέποντας τα οπτικά στοιχεία σε ακουστικά ή απτικά σήματα.

2.2.1 Google TalkBack: Αρχιτεκτονική και Μοντέλο Αλληλεπίδρασης

Το TalkBack είναι η ενσωματωμένη υπηρεσία προσβασιμότητας του Android, η οποία χρησιμοποιεί το AccessibilityService API για να παρέχει ανάδραση σε χρήστες με προβλήματα όρασης. Λειτουργεί ως μια προνομιούχα υπηρεσία που τρέχει στο παρασκήνιο, λαμβάνοντας AccessibilityEvents από το σύστημα όταν αλλάζει η κατάσταση της διεπαφής (π.χ. αλλαγή εστίασης, εμφάνιση νέου παραθύρου) [13].

Τεχνική Υλοποίηση και Χειρονομίες: Όταν το TalkBack είναι ενεργό, το μοντέλο αλληλεπίδρασης αφής αλλάζει θεμελιωδώς. Προστίθεται ένα "αόρατο στρώμα αλληλεπίδρασης" πάνω από την οθόνη [14].

- **Εξερεύνηση με την Αφή (Explore by Touch):** Ο χρήστης σέρνει ένα δάχτυλο στην οθόνη και το σύστημα ανακοινώνει το στοιχείο που βρίσκεται κάτω από το δάχτυλο. Αυτό επιτρέπει τη χωρική κατανόηση της διάταξης της διεπαφής [15].

- **Γραμμική Πλοήγηση:** Με σάρωση (swipe) δεξιά ή αριστερά, η εστίαση προσβασιμότητας (accessibility focus) μετακινείται στο επόμενο ή προηγούμενο λογικό στοιχείο, γραμμικοποιώντας τη δισδιάστατη διεπαφή σε μια ακολουθία στοιχείων. Το εστιασμένο στοιχείο επισημαίνεται συνήθως με ένα πράσινο πλαίσιο [15].

- **Ενεργοποίηση:** Επειδή το απλό άγγιγμα χρησιμοποιείται για εξερεύνηση, η ενεργοποίηση ενός στοιχείου (το αντίστοιχο του "κλικ") απαιτεί διπλό χτύπημα (double tap) οπουδήποτε στην οθόνη [15].

- **Χειριστήρια Ανάγνωσης (Reading Controls):** Η σάρωση προς τα πάνω ή προς τα κάτω αλλάζει τον τρόπο πλοήγησης (π.χ. ανά χαρακτήρα, λέξη, επικεφαλίδα, σύνδεσμο). Αυτό επιτρέπει στον χρήστη να "σαρώσει" γρήγορα μεγάλους όγκους κειμένου ή να πλοηγηθεί σε πολύπλοκες ιστοσελίδες [15].

Αρχιτεκτονικά, το TalkBack βασίζεται στο αντικείμενο AccessibilityNodeInfo, το οποίο περιγράφει τη δομή του παραθύρου. Η υπηρεσία διασχίζει αυτό το δέντρο κόμβων για να εντοπίσει στοιχεία που έχουν οριστεί ως εστιάσιμα (focusable) και να εκφωνήσει το περιεχόμενό τους (π.χ. contentDescription ή κείμενο). Από την έκδοση 9.1 και μετά, υποστηρίζονται χειρονομίες πολλαπλών δαχτύλων (multi-finger gestures), επιτρέποντας συντομεύσεις όπως το πάτημα με τρία δάχτυλα για το μενού του TalkBack [15].

2.2.2 Apple VoiceOver: Ολοκλήρωση και Συνέπεια Οικοσυστήματος

Το VoiceOver στο iOS θεωρείται συχνά το πρότυπο χρυσό στην προσβασιμότητα λόγω της βαθιάς ενσωμάτωσής του στο λειτουργικό σύστημα και της αυστηρής επιβολής των APIs προσβασιμότητας στο Cocoa Touch framework. Όπως και το TalkBack, αλλάζει τη συμπεριφορά της οθόνης αφής, αλλά εισάγει μοναδικές έννοιες όπως ο "ρότορας" (Rotor [16]).

Λειτουργικότητα και Ρότορας:

- **Ο Ρότορας:** Μια εικονική περιστροφική επιλογή που ενεργοποιείται με την περιστροφή δύο δαχτύλων στην οθόνη. Επιτρέπει στον χρήστη να αλλάζει άμεσα τη λειτουργία της σάρωσης πάνω/κάτω (π.χ. αλλαγή ταχύτητας ομιλίας, πλοήγηση ανά επικεφαλίδα, επιλογή γλώσσας).

Αυτό προσδίδει μια τρίτη διάσταση ελέγχου χωρίς να απαιτείται είσοδος σε μενού ρυθμίσεων [16].

- **Χειρονομίες:**
    - **Διπλό χτύπημα με δύο δάχτυλα (Magic Tap):** Μια καθολική χειρονομία που εκτελεί την κύρια ενέργεια της τρέχουσας εφαρμογής (π.χ. απάντηση κλήσης, παύση μουσικής, λήψη φωτογραφίας).
    - **Τριπλό χτύπημα με τρία δάχτυλα (Screen Curtain):** Απενεργοποιεί την οθόνη για ιδιωτικότητα, ενώ το VoiceOver συνεχίζει να μιλάει.

Split-tap: Ο χρήστης κρατά το ένα δάχτυλο σε ένα στοιχείο και χτυπά με ένα άλλο για να το ενεργοποιήσει, επιταχύνοντας την αλληλεπίδραση. Η συνέπεια του VoiceOver επεκτείνεται σε όλο το οικοσύστημα της Apple, επιτρέποντας στους χρήστες να μεταφέρουν τη γνώση των χειρονομιών από το iPhone στο iPad και το Mac (μέσω του trackpad) [16].

2.2.3 Voice Access: Φωνητικός Έλεγχος και Ευρετική Μηχανή

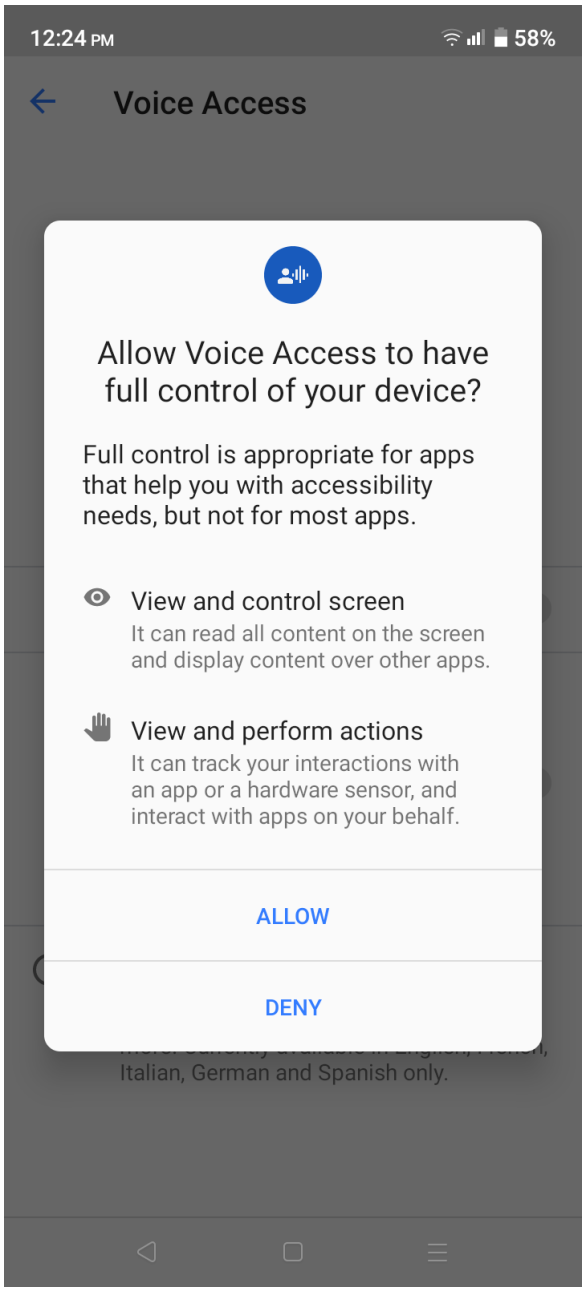
Ενώ το TalkBack και το VoiceOver εστιάζουν στην απτική εξερεύνηση για τυφλούς, το Voice Access σχεδιάστηκε αρχικά για άτομα με κινητικές αναπηρίες, επιτρέποντας τον πλήρη έλεγχο της συσκευής μέσω φωνητικών εντολών. Ωστόσο, η χρησιμότητά του για άτομα με προβλήματα όρασης είναι σημαντική, καθώς επιτρέπει hands-free αλληλεπίδραση.

Τεχνική Πρόκληση και Αντιστοίχιση Ετικετών (Labeling): Η κεντρική πρόκληση για το Voice Access είναι η αντιστοίχιση της φωνητικής εντολής του χρήστη με το σωστό στοιχείο διεπαφής (UI element).

1. **Εξαγωγή Κειμένου:** Για στοιχεία TextView, το ορατό κείμενο γίνεται η εντολή ενεργοποίησης (π.χ. "Open Gmail") [16].
2. **Ευρετική Αντιστοίχιση (Heuristics):** Για κουμπιά που περιέχουν μόνο εικόνες (ImageButtons), το σύστημα βασίζεται στο πεδίο contentDescription. Εάν αυτό λείπει (ένα συχνό πρόβλημα προσβασιμότητας), το Voice Access χρησιμοποιεί ευρετικούς αλγορίθμους, αναλύοντας γειτονικά κείμενα ή τη θέση στο δέντρο ιεραρχίας για να "μαντέψει" την ετικέτα.
3. **Επικάλυψη Πλέγματος (Grid Overlay):** Όταν η σημασιολογική αντιστοίχιση αποτυγχάνει, το σύστημα μπορεί να επικαλύψει ένα αριθμημένο πλέγμα στην οθόνη, επιτρέποντας στον χρήστη να επιλέξει στοιχεία βάσει συντεταγμένων (π.χ. "Tap 5").

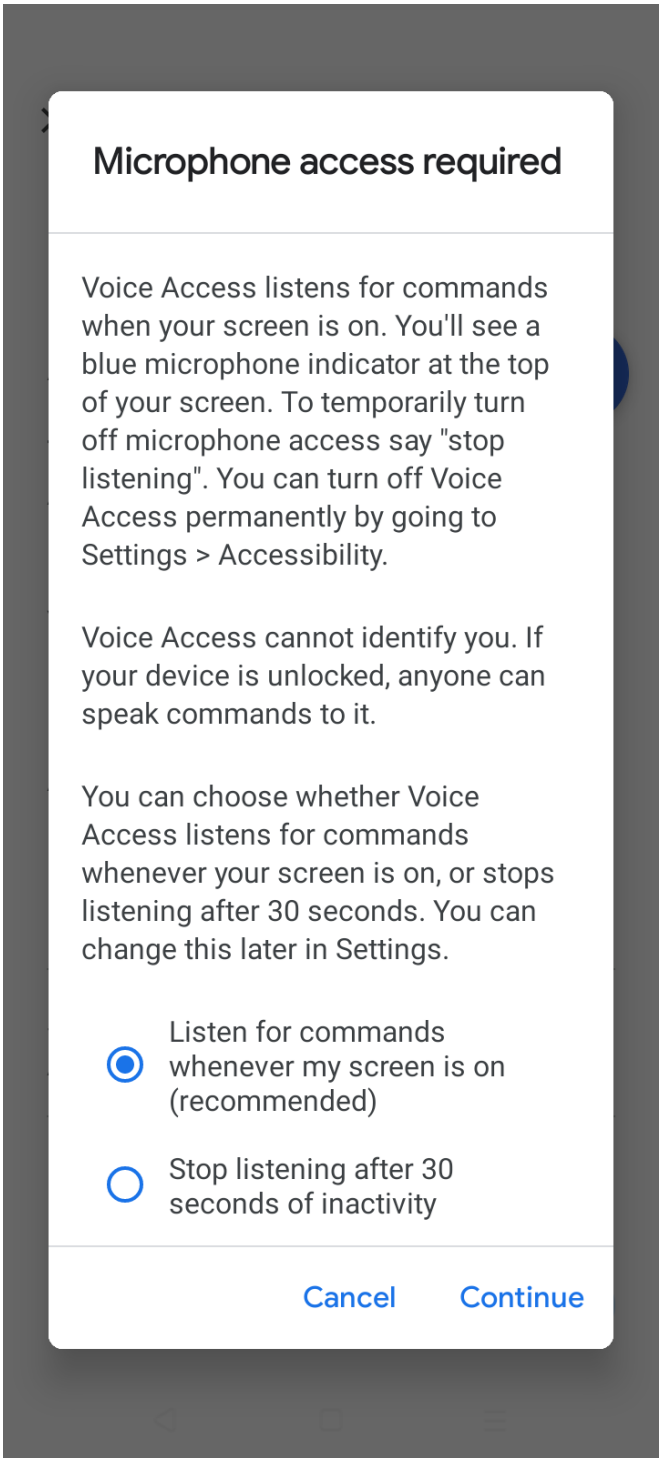
Η επεξεργασία της φωνής γίνεται κυρίως στη συσκευή (on-device) για μείωση του λανθάνοντος χρόνου και προστασία της ιδιωτικότητας, αν και ρυθμίσεις επιτρέπουν την αποστολή δεδομένων στους διακομιστές της Google για βελτιωμένη ακρίβεια αναγνώρισης [17].

Στην εικόνα 2.1 βλέπουμε ότι το voice access ζητάει άδεια πλήρης ελέγχου του κινητού ώστε να μπορεί να βλέπει την οθόνη και να εκτελεί ενέργειες όπως πχ το πάτημα κουμπιών.



Εικόνα 2.1: Voice access άδεια πλήρους ελέγχου

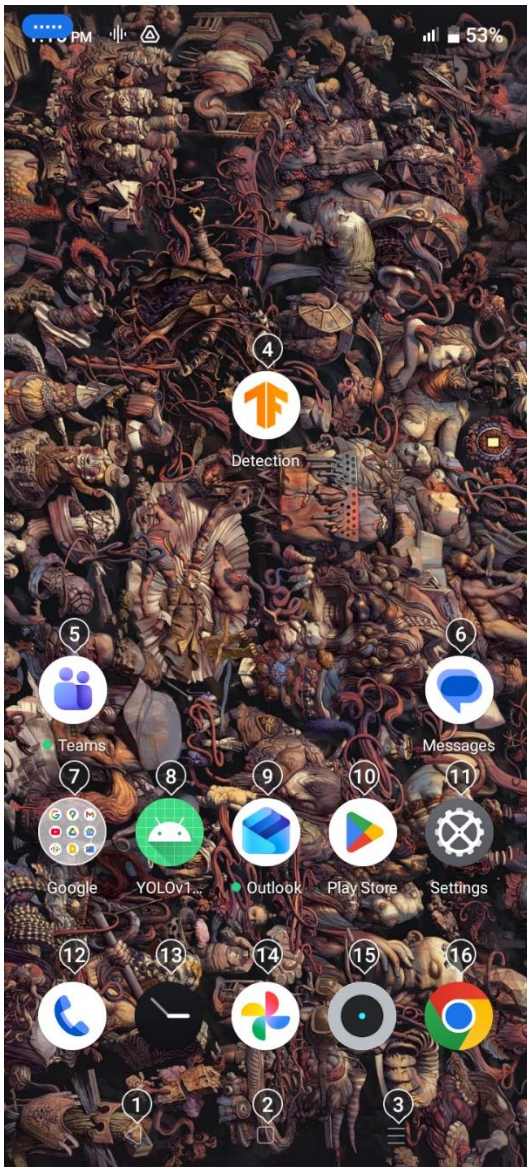
Στην εικόνα 2.2 βλέπουμε της επιλογές ρύθμισης πρόσβασης στο μικρόφωνο. Προτείνεται η πρώτη επιλογή που θα επιτρέπει το κινητό να ακούει για εντολές όταν η οθόνη του κινητού είναι ανοιχτή.



Εικόνα 2.2: Voice access άδεια μικροφώνου

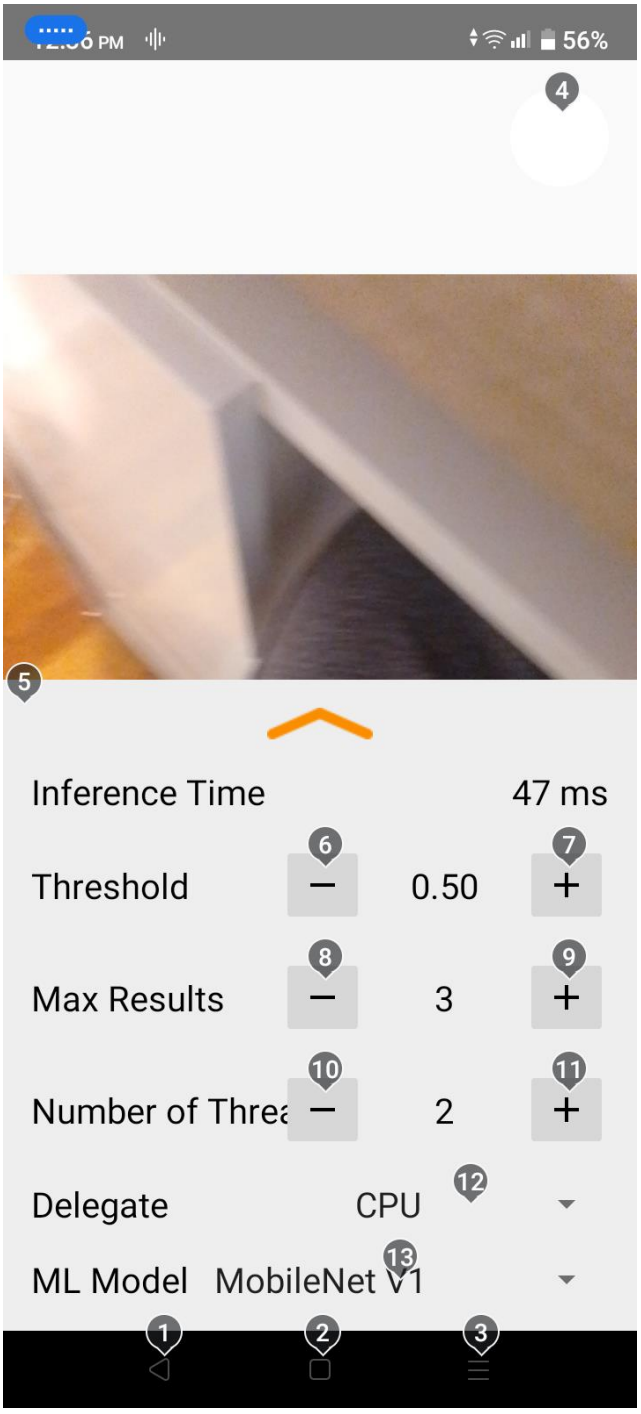
Στην εικόνα 2.3 βλέπουμε μια επιπλέον λειτουργία που μας προσφέρει το voice access. Αριθμεί όλα τα στοιχεία με τα οποία μπορεί να αλληλεπίδρασή ο χρήστης κατά αύξουσα σειρά οι πρώτοι τρεις αριθμεί είναι πάντα αφιερωμένη για την πλοήγηση:

- 1 back
- 2 home
- 3 overview



Εικόνα 2.3: Voice access πλοήγηση

Στην εικόνα 2.4 βλέπουμε πως φαίνεται ο πίνακας ρυθμίσεων της εφαρμογής με το voice access ενεργοποιημένο. Με την φωνητική εντολή "Four" ανοίγουν και κλείνουν η ρυθμίσεις.



Εικόνα 2.4: Voice access μέσα στης ρυθμίσεις της εφαρμογής

2.3 Ανάλυση Ανταγωνισμου: Λύσεις Υπολογιστικής Όρασης για Κινητά

Πέρα από τα εργαλεία του λειτουργικού συστήματος, έχει αναπτυχθεί μια κατηγορία εξειδικευμένων εφαρμογών που αξιοποιούν την Τεχνητή Νοημοσύνη (AI) και την Υπολογιστική Όραση (CV) για να "ερμηνεύσουν" τον φυσικό κόσμο. Οι τρεις κυρίαρχες λύσεις Microsoft Seeing AI, Google Lookout και Be My Eyes υιοθετούν διαφορετικές αρχιτεκτονικές προσεγγίσεις όσον αφορά την εξάρτηση από το cloud, την ταχύτητα απόκρισης και την εμπειρία χρήστη [18].

2.3.1 Microsoft Seeing AI: Η Πολύ-εργαλειοθήκη

Το Seeing AI της Microsoft ακολουθεί τη μεταφορά των "καναλιών" (channels). Ο χρήστης επιλέγει ρητά τη λειτουργία που επιθυμεί (π.χ. Σύντομο Κείμενο, Έγγραφο, Προϊόν, Άτομο, Νόμισμα, Σκηνή, Χρώμα, Φως). Αυτή η ρητή επιλογή περιορίζει το πρόβλημα της υπολογιστικής όρασης, επιτρέποντας τη χρήση πιο εξειδικευμένων και ακριβών μοντέλων [18].

Λειτουργίες και Τεχνολογία:

- **Σύντομο Κείμενο (Short Text):** Εκτελείται αποκλειστικά στη συσκευή (on-device) για ελάχιστο λανθάνοντα χρόνο. Παρέχει ανάγνωση σε πραγματικό χρόνο, εκφωνώντας το κείμενο μόλις εμφανιστεί στο οπτικό πεδίο της κάμερας [18].

- **Αναγνώριση Εγγράφων:** Χρησιμοποιεί ανίχνευση ακμών και ακουστική καθοδήγηση ("μετακινήστε την κάμερα δεξιά") για τη λήψη φωτογραφιών υψηλής ποιότητας, οι οποίες στη συνέχεια υφίστανται επεξεργασία OCR (Optical Character Recognition).

- **Περιγραφή Σκηνής (Scene):** Παραδοσιακά βασιζόταν σε επεξεργασία cloud για να παράγει περιγραφές φυσικής γλώσσας πολύπλοκων εικόνων. Αυτό εισάγει καθυστέρηση, καθιστώντας το λιγότερο κατάλληλο για άμεση πλοήγηση, αλλά παρέχει πλούσια πληροφορία.

- **Ενσωμάτωση LiDAR (World Channel):** Στα μοντέλα iPhone Pro που διαθέτουν αισθητήρα **LiDAR**, το Seeing AI προσφέρει το κανάλι "World". Αυτό επιτρέπει την απτική εξερεύνηση του χώρου (haptic proximity), όπου η συσκευή δονείται με αυξανόμενη συχνότητα καθώς ο χρήστης πλησιάζει ένα αντικείμενο, και την τοποθέτηση εικονικών ετικετών στον τρισδιάστατο χώρο. Αυτή η δυνατότητα παραμένει μοναδική στο οικοσύστημα της Apple λόγω της έλλειψης LiDAR στις περισσότερες συσκευές Android [18].

Ανιχνευτής Φωτός: Μετατρέπει τη φωτεινότητα σε ακουστικό τόνο μεταβλητού ύψους, μια μορφή αισθητηριακής υποκατάστασης που βοηθά στον εντοπισμό παραθύρων ή λαμπτήρων [18].

2.3.2 Google LookOut: Η Προσέγγιση Edge-First

Το Google Lookout σχεδιάστηκε με φιλοσοφία "Android-first" και έμφαση στην εκτέλεση στο άκρο (Edge computing). Στόχος είναι η παροχή συνεχούς, πραγματικού χρόνου ανάδρασης με ελάχιστη εξάρτηση από το διαδίκτυο, κάτι κρίσιμο για την ασφάλεια κατά την πλοήγηση [19].

Λειτουργίες και Αρχιτεκτονική:

- **Λειτουργία Εξερεύνησης (Explore Mode - Beta):** Χρησιμοποιεί μοντέλα ανίχνευσης αντικειμένων (πιθανότατα παραλλαγές MobileNet-SSD ή EfficientDet βελτιστοποιημένες για TFLite) για να εντοπίζει αντικείμενα (π.χ. καρέκλες, τραπέζια, πόρτες) στην πορεία του χρήστη. Προσπαθεί να παρέχει κατεύθυνση (π.χ. "πόρτα στις 12 η ώρα") και μια χονδρική εκτίμηση απόστασης. Το σημαντικότερο πλεονέκτημα είναι ότι λειτουργεί **offline**, εξασφαλίζοντας αξιοπιστία σε υπόγεια ή περιοχές χωρίς σήμα [19].

- **Ετικέτες Τροφίμων & Νόμισμα:** Αυτές οι λειτουργίες κατεβάζουν πακέτα βάσεων δεδομένων στη συσκευή, επιτρέποντας την ταχύτατη ταυτοποίηση προϊόντων και χαρτονομισμάτων χωρίς την καθυστέρηση του cloud [19].

- **Image Q&A με Gemini:** Μια νεότερη προσθήκη που αξιοποιεί το Gemini (το πολυτροπικό LLM της Google). Επιτρέπει στον χρήστη να κάνει ερωτήσεις φυσικής γλώσσας για μια εικόνα ("Τι υπάρχει πάνω στο τραπέζι;"). Σε αντίθεση με τις λειτουργίες πλοήγησης, αυτή η δυνατότητα απαιτεί σύνδεση στο cloud, αναδεικνύοντας μια υβριδική προσέγγιση [19].

2.3.3 Be My Eyes: Από τον Εθελοντισμό στην Τεχνητή Νοημοσύνη

Αρχικά μια πλατφόρμα σύνδεσης τυφλών χρηστών με βλέποντες εθελοντές μέσω βιντεοκλήσης WebRTC, το Be My Eyes έκανε μια επαναστατική στροφή ενσωματώνοντας το "Be My AI", το οποίο τροφοδοτείται από το μοντέλο GPT-4 Vision της OpenAI [20].

Καινοτομία και Περιορισμοί:

- **Σημασιολογική Κατανόηση:** Σε αντίθεση με τα μοντέλα on-device του Lookout, το Be My AI προσφέρει βαθιά σημασιολογική κατανόηση. Μπορεί να διαβάσει χειρόγραφες συνταγές, να ερμηνεύσει την κατάσταση ενός ρούχου ή να εξηγήσει τη δομή μιας ιστοσελίδας στην οθόνη ενός υπολογιστή [21].

- **Εξάρτηση από το Cloud:** Η φύση του ως Large Multimodal Model (LMM) απαιτεί ισχυρή σύνδεση στο διαδίκτυο. Η καθυστέρηση (latency) είναι της τάξης των δευτερολέπτων, καθιστώντας το ακατάλληλο για αποφυγή εμποδίων σε πραγματικό χρόνο κατά την κίνηση [21].

Ιδιωτικότητα: Η αποστολή εικόνων σε διακομιστές τρίτων εγείρει ζητήματα απορρήτου, αν και αναπτύσσονται εταιρικές εκδόσεις για την εξυπηρέτηση πελατών που προσφέρουν μεγαλύτερες εγγυήσεις [21]. Παρακάτω θα συγκρίνουμε σε πίνακα αυτές της 3 επιλογές Πίνακας 2.1.

2.3.4 Συγκριτικός Πίνακας Εφαρμογών

Πίνακας 2.1: Σύγκριση εφαρμογών

Χαρακτηριστικό	**Microsoft Seeing AI**	**Google Lookout**	**Be My Eyes (Be My AI)**
Κύρια Πλατφόρμα	iOS (πρόσφατα και Android)	Android	iOS & Android
Αρχιτεκτονική	Υβριδική (Κανάλια εργασιών)	Edge-First (Εμφαση στο Offline)	Cloud-based LMM (GPT-4)
Offline Δυνατότητες	Κείμενο, Χρώμα, Φως, Νόμισμα	Explore, Κείμενο, Έγγραφα, Τρόφιμα	Καμία (Απαιτεί σύνδεση)
Βοήθεια Πλοήγησης	LiDAR (World Channel - Μόνο Pro)	Explore Mode (Ανίχνευση Αντικειμένων)	Όχι (Υψηλός Λανθάνων Χρόνος)
Εκτίμηση Βάθους	LiDAR ή Ηχοποίηση	Ευρετική / Μονοφθαλμική	Μη διαθέσιμη
Περιγραφή Σκηνής	Δομημένες λεζάντες (Cloud)	Συνεχής ροή αντικειμένων (Edge)	Πλούσια συνομιλιακή περιγραφή

2.4 Το Τεχνολογικό Κενό: Μονοφθαλμική Εκτίμηση Απόστασης (Monocular Distance Estimation)

Ενώ η ανίχνευση αντικειμένων (Object Detection - το να γνωρίζουμε *τι* είναι στην εικόνα) έχει ωριμάσει σημαντικά στις κινητές συσκευές, η κρίσιμη απαίτηση για ασφαλή πλοήγηση είναι η γνώση του *πού* βρίσκεται το αντικείμενο—συγκεκριμένα, η ακριβής μετρική του απόσταση από τον χρήστη. Αυτός είναι ο τομέας της Μονοφθαλμικής Εκτίμησης Βάθους (Monocular Depth Estimation - MDE), και αποτελεί το σημαντικότερο τεχνολογικό κενό στις τρέχουσες λύσεις.

2.4.1 Το Πρόβλημα της Ασάφειας Κλίμακας

Η θεμελιώδης πρόκληση της MDE είναι ότι αποτελεί ένα "κακώς ορισμένο" (ill-posed) πρόβλημα. Μαθηματικά, κατά την προβολή μιας τρισδιάστατης σκηνής σε ένα δισδιάστατο επίπεδο εικόνας (μέσω της προοπτικής προβολής), χάνεται η πληροφορία του βάθους **z**. Ένας άπειρος αριθμός τρισδιάστατων σκηνών μπορεί να παράγει την ίδια δισδιάστατη εικόνα. Για παράδειγμα, μια μικρή καρέκλα κοντά στην κάμερα μπορεί να καταλαμβάνει τα ίδια pixel με μια τεράστια καρέκλα που βρίσκεται μακριά [22].

Τα τυπικά Συνελικτικά Νευρωνικά Δίκτυα (CNNs) μπορούν να μάθουν το "σχετικό" βάθος (το Αντικείμενο Α είναι μπροστά από το Αντικείμενο Β) βασιζόμενα σε ενδείξεις απόκρυψης και προοπτικής, αλλά δυσκολεύονται να ανακτήσουν το "μετρικό" βάθος (το Αντικείμενο Α απέχει 1,5 μέτρα) χωρίς αντικείμενα αναφοράς ή στερεοσκοπικά δεδομένα. Για έναν τυφλό χρήστη, η διαφορά μεταξύ σχετικού και μετρικού βάθους είναι η διαφορά μεταξύ του να γνωρίζει ότι "υπάρχει τοίχος μπροστά" και "ο τοίχος είναι στα 3 βήματα". Οι υπάρχουσες λύσεις που προσπαθούν να επιλύσουν αυτό το πρόβλημα μέσω γλωσσικών περιγραφών (π.χ. RSA - Resolving Scale Ambiguities) βρίσκονται ακόμη σε ερευνητικό στάδιο [22].

2.4.2 Προσεγγίσεις Βαθιάς Μάθησης και Αρχιτεκτονικές

Η σύγχρονη έρευνα προσπαθεί να επιλύσει το πρόβλημα χρησιμοποιώντας μοντέλα βαθιάς μάθησης που συμπεραίνουν το βάθος από μεμονωμένες εικόνες RGB.

- **Αρχιτεκτονικές Κωδικοποιητή-Αποκωδικοποιητή (Encoder-Decoder):** Μοντέλα όπως το **FastDepth** χρησιμοποιούν έναν κωδικοποιητή MobileNet και έναν ελαφρύ αποκωδικοποιητή με βάθος διαχωρίσιμες συνελίξεις (depthwise separable convolutions) για να επιτύχουν απόδοση πραγματικού χρόνου (π.χ. ~178 FPS σε Jetson TX2, αλλά σημαντικά λιγότερο σε τυπικά κινητά). Αυτά τα μοντέλα εκπαιδεύονται σε σύνολα δεδομένων όπως το NYU Depth v2 ή το KITTI, μαθαίνοντας ουσιαστικά να συσχετίζουν διαβαθμίσεις υφής και μεγέθη αντικειμένων με τιμές βάθους [23].

- **Απόσταξη Γνώσης (Knowledge Distillation):** Για την εκτέλεση σε κινητά τηλέφωνα, τεράστια μοντέλα που βασίζονται σε Transformers (όπως DPT ή Swin-Transformer) "αποστάζονται" σε ελαφριά δίκτυα-μαθητές. Για παράδειγμα, το **LUMDE** (Lightweight Unsupervised Monocular Depth Estimation) επιτυγχάνει ακρίβεια κοντά στο state-of-the-art με 95% λιγότερες παραμέτρους από το δίκτυο-δάσκαλο, καθιστώντας εφικτή την ανάπτυξη σε Android [24].

Μέθοδοι Βασισμένες σε Patches: Πρόσφατες εξελίξεις όπως το **PatchFusion** και το **DepthPro** διαιρούν την εικόνα σε τμήματα (patches) για να διαχειριστούν εισόδους υψηλής ανάλυσης και να διατηρήσουν την τοπική λεπτομέρεια, η οποία συχνά χάνεται στα επίπεδα υπο-δειγματοληψίας των τυπικών κωδικοποιητών [24].

2.4.3 Περιορισμοί Υλικού και Ανάπτυξης

Η υλοποίηση της MDE σε κινητές συσκευές αντιμετωπίζει το τρίλημμα "Ενέργεια-Λανθάνων Χρόνος-Ακρίβεια".

- **Θερμικός Στραγγαλισμός (Thermal Throttling):** Η συνεχής εκτέλεση πολύπλοκων CNNs ή Vision Transformers (ViTs) για πλοήγηση παράγει σημαντική θερμότητα. Τα smartphones, σε αντίθεση με τις GPUs που διαθέτουν ενεργή ψύξη, βασίζονται σε παθητική ψύξη. Η παρατεταμένη χρήση πυρήνων CPU/GPU σε υψηλή συχνότητα ενεργοποιεί μηχανισμούς θερμικού στραγγαλισμού, μειώνοντας δραστικά τον ρυθμό καρέ (FPS) και καθιστώντας το βοήθημα πλοήγησης ασταθές και μη ασφαλές [25].

- **Εξάντληση Μπαταρίας:** Το συμπερασμός (inference) βαθιάς μάθησης είναι ενεργοβόρος. Μελέτες σε παραλλαγές του YOLO δείχνουν ότι, ενώ η αποφόρτωση σε NPU (Neural Processing Unit) είναι πιο αποδοτική από τη CPU/GPU, η συνεχής επεξεργασία βίντεο που απαιτείται για την πλοήγηση μπορεί να εξαντλήσει γρήγορα την μπαταρία, αφήνοντας τον χρήστη χωρίς τη συσκευή επικοινωνίας του [25].

- **Λανθάνων Χρόνος Συμπερασμού:** Για ασφαλή πλοήγηση, ο λανθάνων χρόνος "motion-to-audio" πρέπει να είναι ελάχιστος. Μια καθυστέρηση 1 δευτερολέπτου (που παρατηρείται σε ορισμένες υλοποιήσεις **YOLOv8** σε υλικό edge) προκαλεί αναντιστοιχία μεταξύ της φυσικής θέσης του χρήστη και της ακουστικής ανάδρασης, οδηγώντας δυνητικά σε συγκρούσεις. Τα ελαφριά μοντέλα (π.χ. μετατροπές **TFLite**) βελτιώνουν την ταχύτητα αλλά συχνά εις βάρος των

"Ψευδώς Θετικών" αποτελεσμάτων ή της "Διόγκωσης Ακμών" (Edge Fattening), όπου εμπόδια ανιχνεύονται εκεί που δεν υπάρχουν [25].

2.4.4 Το 'Τελευταίο Μέτρο' σε Μη Δομημένα Περιβάλλοντα

Το τεχνολογικό κενό είναι πιο οξύ στην έλλειψη στιβαρότητας σε *μη δομημένα* περιβάλλοντα. Τα περισσότερα μοντέλα MDE εκπαιδεύονται σε σύνολα δεδομένων με καθαρά επίπεδα εδάφους και τυπικό φωτισμό (όπως το **KITTI** για αυτόνομη οδήγηση). Η εσωτερική πλοήγηση για τυφλούς συχνά περιλαμβάνει χαοτικές σκηνές—γυάλινες πόρτες (διαφανείς), γυαλισμένα δάπεδα (ανακλαστικά) και χαμηλό φωτισμό—που μπερδεύουν τους τυπικούς αισθητήρες βάθους και τα μοντέλα όρασης [26].

Επιπλέον, ενώ το **LiDAR** (διαθέσιμο σε high-end iPhones) λύνει το πρόβλημα της μετρικής κλίμακας, παραμένει ακριβό και σπάνιο στην ευρύτερη παγκόσμια αγορά συσκευών Android. Επομένως, το "Άγιο Δισκοπότηρο" παραμένει ένας καθαρά μονοφθαλμικός, μετρικά ακριβής, ενεργειακά αποδοτικός εκτιμητής βάθους που να τρέχει σε smartphones μεσαίας κατηγορίας—μια λύση που επί του παρόντος υπάρχει μόνο σε αποσπασματικά ερευνητικά πρωτότυπα και όχι σε στιβαρές εμπορικές εφαρμογές [26].

2.5 Συμπεράσματα και Μελλοντικές Κατευθύνσεις

Η τρέχουσα κατάσταση των υποβοηθητικών τεχνολογιών για άτομα με προβλήματα όρασης χαρακτηρίζεται από μια διχοτομία: από τη μια πλευρά υπάρχουν τα ώριμα και στιβαρά πλαίσια προσβασιμότητας των λειτουργικών συστημάτων (TalkBack/VoiceOver) και από την άλλη ένα ταχέως εξελισσόμενο, αλλά κατακερματισμένο, επίπεδο εφαρμογών (Lookout/Seeing AI/Be My AI).

Η ενσωμάτωση της Παραγωγικής Τεχνητής Νοημοσύνης (όπως το Be My AI) έχει επιλύσει σε μεγάλο βαθμό το πρόβλημα της *σημασιολογικής κατανόησης*—οι χρήστες μπορούν πλέον να λάβουν λεπτομερείς περιγραφές σχεδόν για οτιδήποτε. Ωστόσο, το πρόβλημα της *γεωμετρικής κατανόησης*—η ασφαλής πλοήγηση στο χώρο σε πραγματικό χρόνο—παραμένει άλυτο για την πλειοψηφία των χρηστών που δεν διαθέτουν υλικό εξοπλισμένο με **LiDAR**.

Η ερευνητική τροχιά δείχνει προς τη βελτιστοποίηση της **Edge AI**. Η εξάρτηση από την επεξεργασία cloud για την περιγραφή σκηνής αποτελεί εμπόδιο στην καθολική χρήση λόγω ζητημάτων καθυστέρησης και ιδιωτικότητας. Οι μελλοντικές λύσεις πρέπει να αξιοποιήσουν τις Μονάδες Νευρωνικής Επεξεργασίας (NPUs) που γίνονται πλέον πρότυπο στα SoCs των κινητών για να εκτελούν ελαφριά μοντέλα Transformers και MDE τοπικά. Τεχνικές όπως η **Απόσταξη Γνώσης** και η **Εκπαίδευση με Επίγνωση Κβαντισμού** (Quantization-Aware Training) δεν είναι απλώς τεχνάσματα βελτιστοποίησης, αλλά ουσιώδεις προϋποθέσεις για να προσφερθεί ασφαλής, χωρική επίγνωση πραγματικού χρόνου στα 2,2 δισεκατομμύρια άτομα που ζουν με οπτική αναπηρία.

Η μετάβαση από την "Αισθητηριακή Υποκατάσταση" (ηχοποίηση όταν πλησιάζει ένα αντικείμενο) στη "Σημασιολογική Πλοήγηση" (καθοδήγηση του χρήστη σε μια άδεια καρέκλα 3 μέτρα δεξιά) εξαρτάται πλήρως από το κλείσιμο του κενού στη Μονοφθαλμική Μετρική Εκτίμηση Βάθους. Μέχρις ότου οι κινητές συσκευές μπορέσουν να αντιληφθούν αξιόπιστα την απόσταση από μια μεμονωμένη κάμερα RGB με χαμηλό ενεργειακό κόστος, το "τελευταίο μέτρο" της εσωτερικής πλοήγησης θα παραμένει μια σημαντική πρόκληση

Κεφάλαιο 3ο: Θεωρητικό Υπόβαθρο: Τεχνητή Νοημοσύνη και Αλγόριθμοι Ανίχνευσης Αντικειμένων

3.1 Εισαγωγή στην Τεχνητή Νοημοσύνη και την Υποβοηθητική Τεχνολογία

Η σύγχρονη εποχή της πληροφορικής χαρακτηρίζεται από την ραγδαία ανάδυση της Τεχνητής Νοημοσύνης (Artificial Intelligence - AI), ενός πεδίου που έχει μετασχηματίσει θεμελιωδώς τον τρόπο με τον οποίο οι μηχανές αντιλαμβάνονται, αναλύουν και αλληλεπιδρούν με το φυσικό περιβάλλον. Στο ειδικότερο πλαίσιο της Υποβοηθητικής Τεχνολογίας (Assistive Technology - AT) για άτομα με οπτική αναπηρία, η Τεχνητή Νοημοσύνη δεν αποτελεί πλέον μια θεωρητική ακαδημαϊκή άσκηση, αλλά έναν κρίσιμο καταλύτη για την ενίσχυση της αυτονομίας και της ποιότητας ζωής. Σύμφωνα με πρόσφατες εκτιμήσεις του Παγκόσμιου Οργανισμού Υγείας, εκατοντάδες εκατομμύρια άνθρωποι παγκοσμίως αντιμετωπίζουν σοβαρές διαταραχές όρασης, με ένα σημαντικό ποσοστό να είναι πλήρως τυφλοί. Η παραδοσιακή κινητικότητα αυτών των ατόμων βασίζεται κυρίως στο λευκό μπαστούνι και στους σκύλους-οδηγούς. Ωστόσο, αυτά τα μέσα, αν και αξιόπιστα για την ανίχνευση εμποδίων στο άμεσο επίπεδο του εδάφους, αδυνατούν να παρέχουν πλούσια σημασιολογική πληροφορία για το περιβάλλον (π.χ., "ένα τραπέζι βρίσκεται στα δύο μέτρα") ή να προστατεύσουν από εμπόδια που βρίσκονται στο ύψος του κορμού και της κεφαλής [27].

Η ανάπτυξη φορητών συστημάτων πλοήγησης, και συγκεκριμένα εφαρμογών σε κινητές συσκευές (όπως smartphones με λειτουργικό Android), έχει ανοίξει νέους ορίζοντες. Η ικανότητα μιας συσκευής να εκτελεί αλγορίθμους Υπολογιστικής Όρασης (Computer Vision - CV) σε πραγματικό χρόνο (real-time) επιτρέπει την ψηφιακή "ερμηνεία" της οπτικής σκηνής. Ωστόσο, η μετάβαση από τα ισχυρά υπολογιστικά κέντρα (data centers) στις περιορισμένες δυνατότητες μιας κινητής συσκευής θέτει αυστηρούς περιορισμούς. Η έννοια του "πραγματικού χρόνου" είναι εδώ κρίσιμη: μια καθυστέρηση (latency) στην ανίχνευση ενός επερχόμενου κινδύνου μπορεί να αποβεί μοιραία για τον χρήστη [28].

Το παρόν κεφάλαιο φιλοδοξεί να καλύψει διεξοδικά το θεωρητικό υπόβαθρο που διέπει αυτές τις τεχνολογίες. Θα ξεκινήσουμε με την ανάλυση της μετάβασης από την παραδοσιακή Μηχανική Μάθηση στη Βαθιά Μάθηση (Deep Learning), εστιάζοντας στα Συνελικτικά Νευρωνικά Δίκτυα (CNNs). Στη συνέχεια, θα εξετάσουμε τις επιμέρους εργασίες της όρασης (ταξινόμηση, ανίχνευση, τμηματοποίηση) υπό το πρίσμα της χρησιμότητάς τους για έναν τυφλό χρήστη. Ιδιαίτερη έμφαση θα δοθεί στην οικογένεια αλγορίθμων YOLO (You Only Look Once) και ειδικότερα στην έκδοση YOLOv8, η οποία αποτελεί την αιχμή του δόρατος για εφαρμογές σε περιβάλλοντα περιορισμένων πόρων (edge devices), συγκρίνοντάς την με παραδοσιακές αρχιτεκτονικές όπως το Faster R-CNN. Τέλος, θα αναλυθούν οι μετρικές αξιολόγησης (IoU, mAP, FPS) που καθορίζουν την επιτυχία ενός τέτοιου συστήματος.

3.2 Από την Μηχανική Μάθηση στην Βαθιά Μάθηση

Η κατανόηση των σύγχρονων συστημάτων ανίχνευσης απαιτεί τη διάκριση μεταξύ της κλασικής Μηχανικής Μάθησης (Machine Learning - ML) και της Βαθιάς Μάθησης (Deep Learning - DL), καθώς η εξέλιξη αυτή καθόρισε την αποτελεσματικότητα των εφαρμογών όρασης.

3.2.1 Περιορισμοί της Παραδοσιακής Μηχανικής Μάθησης

Στην παραδοσιακή Μηχανική Μάθηση, η διαδικασία επίλυσης ενός προβλήματος όρασης βασιζόταν σε μεγάλο βαθμό στη χειροκίνητη σχεδίαση χαρακτηριστικών (hand-crafted features). Αλγόριθμοι όπως οι SVM (Support Vector Machines) ή τα Random Forests δεν μπορούσαν να επεξεργαστούν απευθείας τα πρωτογενή δεδομένα εικόνας (raw pixels) λόγω της τεράστιας διαστατικότητας και της μεταβλητότητας των εικόνων. Οι ερευνητές έπρεπε να σχεδιάσουν "περιγραφείς" (descriptors) όπως το SIFT (Scale-Invariant Feature Transform) ή το HOG (Histogram of Oriented Gradients), οι οποίοι κωδικοποιούσαν πληροφορίες όπως ακμές, γωνίες και αλλαγές φωτεινότητας [29].

Αυτή η προσέγγιση είχε σημαντικούς περιορισμούς. Πρώτον, η ποιότητα του συστήματος εξαρτιόταν από την ικανότητα του σχεδιαστή να προβλέψει ποια χαρακτηριστικά είναι σημαντικά. Δεύτερον, τα χειροκίνητα χαρακτηριστικά δεν ήταν γενικεύσιμα· ένας περιγραφέας που λειτουργούσε καλά για την ανίχνευση προσώπων μπορεί να ήταν ακατάλληλος για την ανίχνευση οχημάτων. Τέλος, η υπολογιστική διαδικασία εξαγωγής αυτών των χαρακτηριστικών ήταν συχνά αργή και δεν κλιμακωνόταν καλά με την αύξηση του όγκου των δεδομένων.

3.2.2 Η Επανάσταση της Βαθιάς Μάθησης (Deep Learning)

Η Βαθιά Μάθηση, ένα υποσύνολο της ML, έλυσε αυτό το πρόβλημα εισάγοντας την έννοια της "εκμάθησης αναπαραστάσεων" (representation learning). Αντί να υπαγορεύουμε στο σύστημα τι να ψάξει, κατασκευάζουμε πολυεπίπεδα Τεχνητά Νευρωνικά Δίκτυα (Artificial Neural Networks - ANNs) τα οποία μαθαίνουν αυτόματα να εξάγουν ιεραρχικά χαρακτηριστικά από τα δεδομένα [26].

Σε ένα βαθύ δίκτυο, τα πρώτα επίπεδα (layers) μαθαίνουν να ανιχνεύουν απλές δομές, όπως γραμμές και καμπύλες. Τα ενδιάμεσα επίπεδα συνθέτουν αυτές τις δομές για να σχηματίσουν πιο σύνθετα μοτίβα, όπως μάτια ή ρόδες. Τα βαθύτερα επίπεδα αναγνωρίζουν ολόκληρα αντικείμενα. Αυτή η ιεραρχική δόμηση μιμείται, σε έναν βαθμό, τη λειτουργία του οπτικού φλοιού των θηλαστικών. Η εκπαίδευση αυτών των δικτύων πραγματοποιείται μέσω του αλγορίθμου της Οπισθοδιάδοσης (Backpropagation), ο οποίος προσαρμόζει τα βάρη των συνάψεων ώστε να ελαχιστοποιηθεί το σφάλμα μεταξύ της πρόβλεψης και της πραγματικότητας [29].

3.2.3 Ο Ρόλος των Συναρτήσεων Ενεργοποίησης (Activation Functions)

Μια κρίσιμη συνιστώσα των νευρωνικών δικτύων είναι η συνάρτηση ενεργοποίησης. Χωρίς αυτές, ένα νευρωνικό δίκτυο, ανεξάρτητα από το πόσα επίπεδα διαθέτει, θα συμπεριφερόταν ως ένας απλός γραμμικός μετασχηματισμός (linear regression), ανίκανος να μοντελοποιήσει τις πολύπλοκες, μη γραμμικές σχέσεις που διέπουν τις εικόνες του φυσικού κόσμου.

ReLU (Rectified Linear Unit)

Για πολλά χρόνια, η κυρίαρχη συνάρτηση ενεργοποίησης ήταν η **ReLU**, η οποία ορίζεται ως:

$$f(x) = Max(0, x) \#3.2.3$$

Η **ReLU** έλυσε το πρόβλημα της εξαφάνισης της κλίσης (vanishing gradient problem) που ταλαιπωρούσε παλαιότερες συναρτήσεις όπως η Sigmoid και η Tanh, επιτρέποντας την εκπαίδευση βαθύτερων δικτύων. Είναι υπολογιστικά αποδοτική, καθώς απαιτεί μόνο μια σύγκριση. Ωστόσο, πάσχει από το πρόβλημα του "νεκρού ReLU" (dying ReLU): εάν ένας νευρώνας βρεθεί σε κατάσταση όπου η είσοδός του είναι αρνητική, η έξοδος είναι μηδέν και η κλίση είναι μηδέν. Αυτό σημαίνει ότι ο νευρώνας παύει να ενημερώνεται κατά την εκπαίδευση, καθιστώντας τμήματα του δικτύου ανενεργά [30].

SiLU (Sigmoid Linear Unit) και Swish

Οι σύγχρονες αρχιτεκτονικές, όπως το YOLOv8 που θα χρησιμοποιηθεί στην παρούσα μελέτη, έχουν υιοθετήσει πιο εξελιγμένες συναρτήσεις ενεργοποίησης όπως η SiLU (γνωστή και ως Swish). Η SiLU ορίζεται ως:

$$f(x) = x \cdot \sigma(x) = \frac{x}{(1 + e^{-x})} \#3.2.3$$

Η SiLU διαφέρει από τη ReLU σε δύο κρίσιμα σημεία:

Ομαλότητα: Είναι παραγωγίσιμη σε όλο το πεδίο ορισμού της, κάτι που βοηθά στη σταθερότητα της βελτιστοποίησης.

Μη Μοντονικότητα: Για μικρές αρνητικές τιμές, η συνάρτηση δεν μηδενίζεται αμέσως αλλά έχει μια μικρή αρνητική καμπύλη. Αυτό επιτρέπει τη ροή πληροφορίας ακόμη και για αρνητικές εισόδους, λύνοντας το πρόβλημα του νεκρού νευρώνα [31].

Έρευνες έχουν δείξει ότι η χρήση της SiLU σε βαθιά δίκτυα όπως το CSPDarknet (η ραχοκοκαλιά του YOLOv8) οδηγεί σε βελτιωμένη ακρίβεια και ταχύτερη σύγκλιση σε σύγκριση με τη ReLU ή τη Leaky ReLU, καθιστώντας την ιδανική για εφαρμογές υψηλής ακρίβειας [30].

3.3 Συνελικτικά Νευρωνικά Δίκτυα(CNNs): Η Καρδιά της Όρασης

Τα Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks - CNNs) αποτελούν την ειδική κατηγορία δικτύων που κυριαρχεί στην υπολογιστική όραση. Η αρχιτεκτονική τους είναι σχεδιασμένη να εκμεταλλεύεται τη χωρική δομή των εικόνων.

3.3.1 Δομικά Στοιχεία CNN

Ένα τυπικό CNN αποτελείται από τρεις κύριους τύπους επιπέδων:

- **Συνελικτικά Επίπεδα (Convolutional Layers):** Εδώ πραγματοποιείται η κύρια εξαγωγή χαρακτηριστικών. Ένα φίλτρο (kernel) μικρών διαστάσεων σαρώνει την εικόνα εισόδου. Σε κάθε θέση, υπολογίζεται το εσωτερικό γινόμενο μεταξύ των βαρών του φίλτρου και των τιμών των pixels. Αυτή η διαδικασία παράγει χάρτες χαρακτηριστικών (feature maps) που αντιπροσωπεύουν την παρουσία συγκεκριμένων μοτίβων (π.χ., κάθετες γραμμές) σε διαφορετικά σημεία της εικόνας. Η ιδιότητα της "μεταβολικής αναλλοιότητας" (translation invariance) που προσφέρουν τα CNNs σημαίνει ότι ένα αντικείμενο αναγνωρίζεται ανεξάρτητα από το πού βρίσκεται μέσα στην εικόνα [32].

- **Επίπεδα Συγκέντρωσης (Pooling Layers):** Στόχος αυτών των επιπέδων είναι η μείωση της διαστατικότητας των δεδομένων (downsampling) και η αύξηση της αφαίρεσης. Το Max Pooling, για παράδειγμα, επιλέγει τη μέγιστη τιμή από μια περιοχή ,διατηρώντας μόνο τα πιο ισχυρά χαρακτηριστικά και απορρίπτοντας περιττές πληροφορίες θέσης. Αυτό μειώνει δραματικά τον αριθμό των υπολογισμών που απαιτούνται στα επόμενα επίπεδα, κάτι κρίσιμο για την εκτέλεση σε κινητές συσκευές.
- **Επίπεδα Πρόβλεψης:** Στις σύγχρονες αρχιτεκτονικές ανίχνευσης, τα παραδοσιακά πλήρως συνδεδεμένα επίπεδα (Fully Connected Layers) έχουν αντικατασταθεί από συνελικτικά επίπεδα στην "κεφαλή" (head) του δικτύου, επιτρέποντας την επεξεργασία εικόνων μεταβλητού μεγέθους και την εξαγωγή χωρικών προβλέψεων (bounding boxes, μάσκες) [31].

3.3.2 Cross Stage Partial Networks (CSPNet)

Μια σημαντική εξέλιξη στα CNNs, η οποία υιοθετήθηκε από την οικογένεια YOLO (συμπεριλαμβανομένου του YOLOv8), είναι η χρήση των δικτύων CSP (Cross Stage Partial Networks). Το πρόβλημα με τα πολύ βαθιά δίκτυα είναι ότι ο υπολογιστικός φόρτος αυξάνεται δυσανάλογα με την ακρίβεια. Το CSPNet αντιμετωπίζει αυτό το πρόβλημα διαχωρίζοντας τον χάρτη χαρακτηριστικών στη βάση κάθε σταδίου σε δύο μέρη. Το ένα μέρος περνά μέσα από μια σειρά συνελικτικών επιπέδων (dense block), ενώ το άλλο παρακάμπτει αυτή την επεξεργασία και συνενώνεται απευθείας στην έξοδο.

Αυτή η αρχιτεκτονική επιτυγχάνει δύο στόχους:

- Μείωση Υπολογιστικού Κόστους: Επεξεργαζόμενο μόνο ένα τμήμα των καναλιών, το δίκτυο μειώνει τις απαιτούμενες πράξεις κινητής υποδιαστολής (FLOPs).
- Βελτίωση Ροής Κλίσης: Η διαδρομή παράκαμψης επιτρέπει στην πληροφορία (και κατά συνέπεια στην κλίση κατά την εκπαίδευση) να ρέει ανεμπόδιστα, ενισχύοντας την ικανότητα μάθησης και μειώνοντας τον κίνδυνο υπερπροσαρμογής. Στο YOLOv8, αυτή η δομή ενσωματώνεται στο νέο C2f module, το οποίο θα αναλυθεί διεξοδικά στην ενότητα 3.5 [33].

3.4 Εργασίες Υπολογιστικής Όρασης και Προσβασιμότητα

Για την ανάπτυξη ενός συστήματος πλοήγησης τυφλών, είναι απαραίτητο να κατανοήσουμε τις διαφορετικές εργασίες της υπολογιστικής όρασης και την αξία που προσφέρει η καθεμία στον τελικό χρήστη.

3.4.1 Ταξινόμηση Εικόνας (Image Classification)

Η ταξινόμηση είναι η απλούστερη μορφή όρασης. Ο αλγόριθμος λαμβάνει ως είσοδο μια εικόνα και αποδίδει μία μόνο ετικέτα (π.χ., "Δρόμος" ή "Εμπόδιο").

- *Περιορισμός:* Για έναν τυφλό χρήστη, η πληροφορία αυτή είναι ανεπαρκής. Το να γνωρίζει ότι υπάρχει "ένα αυτοκίνητο" μπροστά του δεν βοηθά αν δεν γνωρίζει *πού* ακριβώς βρίσκεται (αριστερά, δεξιά, κέντρο) και πόσο κοντά είναι [34].

3.4.2 Ανίχνευση Αντικειμένων (Object Detection)

Η ανίχνευση αντικειμένων προχωρά ένα βήμα παραπέρα. Όχι μόνο αναγνωρίζει τις κλάσεις των αντικειμένων, αλλά εντοπίζει και τη θέση τους στην εικόνα, περιβάλλοντάς τα με ορθογώνια πλαίσια (Bounding Boxes).

- *Χρησιμότητα:* Τα bounding boxes επιτρέπουν στο σύστημα να υπολογίσει την κατεύθυνση του εμποδίου (π.χ., αν το κέντρο του πλαισίου είναι στα δεξιά της εικόνας, το ηχητικό σήμα μπορεί να μεταδοθεί στο δεξί αυτί). Το μέγεθος του πλαισίου μπορεί να χρησιμοποιηθεί ως ένδειξη εγγύτητας [35].

- *Περιορισμός:* Τα ορθογώνια πλαίσια περιλαμβάνουν συχνά και τμήματα του φόντου. Για παράδειγμα, ένα διαγώνια τοποθετημένο μπαστούνι ή ένα άτομο με απλωμένα χέρια θα περιβληθεί από ένα μεγάλο ορθογώνιο που περιέχει πολύ "κενό" χώρο. Αυτό μπορεί να οδηγήσει σε ψευδείς συναγερμούς, κάνοντας τον τυφλό χρήστη να πιστεύει ότι ο δρόμος είναι κλειστός ενώ υπάρχει ελεύθερος χώρος διέλευσης [35].

3.4.3 Τμηματοποίηση Στιγμιότυπου (Instance Segmentation)

Η τμηματοποίηση στιγμιοτύπου συνδυάζει τα πλεονεκτήματα της ανίχνευσης με την ακρίβεια της σημασιολογικής τμηματοποίησης. Αναγνωρίζει κάθε ξεχωριστό αντικείμενο και παράγει μια μάσκα ακριβείας (pixel-wise mask) που καλύπτει *μόνο* τα pixels που ανήκουν στο αντικείμενο.

- *Κρισιμότητα για Τυφλούς:* Αυτή είναι η πλέον ενδεδειγμένη τεχνολογία για πλοήγηση ακριβείας. Η μάσκα επιτρέπει τον ακριβή καθορισμό των ορίων του εμποδίου. Το σύστημα μπορεί να καθοδηγήσει τον χρήστη να περάσει δίπλα από ένα ακανόνιστο αντικείμενο (π.χ., ένα πεσμένο δέντρο) με ασφάλεια, εκμεταλλευόμενο τον πραγματικό ελεύθερο χώρο που τα bounding boxes θα κάλυπταν λανθασμένα. Το YOLOv8, όπως θα δούμε, ενσωματώνει αυτή τη δυνατότητα με ελάχιστη επιβάρυνση στην ταχύτητα [36].

3.5 Αρχιτεκτονικές Ανίχνευσης : (Two-Stage vs One-Stage)

Η επιλογή της κατάλληλης αρχιτεκτονικής είναι μια διαδικασία ζύγισης (trade-off) μεταξύ ακρίβειας και ταχύτητας.

3.5.1 Ανιχνευτές Δύο Σταδίων (Two-Stage Detectors)

Το κλασικό παράδειγμα αυτής της κατηγορίας είναι το **Faster R-CNN**.

1. **Στάδιο 1 (Region Proposal):** Ένα δίκτυο πρότασης περιοχών (Region Proposal Network - RPN) σαρώνει την εικόνα και προτείνει χιλιάδες περιοχές που *πιθανώς* περιέχουν αντικείμενα (Background vs Foreground).
2. **Στάδιο 2 (Refinement):** Οι προτεινόμενες περιοχές εξάγονται και τροφοδοτούνται σε έναν δεύτερο ταξινομητή για να καθοριστεί η ακριβής κλάση και να διορθωθούν τα όρια του πλαισίου.

Ανάλυση Απόδοσης: Ενώ το Faster R-CNN προσφέρει ιστορικά υψηλή ακρίβεια (ειδικά σε μικρά αντικείμενα), η πολυπλοκότητά του το καθιστά ακατάλληλο για mobile εφαρμογές πραγματικού χρόνου. Μελέτες έχουν δείξει ότι σε περιβάλλον GPU κινητού (Snapdragon), το latency μπορεί να φτάσει τα 54ms ή και πολύ περισσότερο (εκατοντάδες ms σε CPU), δημιουργώντας μη αποδεκτές καθυστερήσεις για την πλοήγηση. Η υψηλή κατανάλωση ενέργειας είναι επίσης αποτρεπτικός παράγοντας [37].

3.5.2 Ανιχνευτές Ενός Σταδίου (One-Stage Detectors)

Οι ανιχνευτές ενός σταδίου, όπως το **YOLO** και το **SSD** (Single Shot Detector), αντιμετωπίζουν την ανίχνευση ως ένα ενιαίο πρόβλημα παλινδρόμησης.

- **Λειτουργία:** Το δίκτυο διαιρεί την εικόνα σε ένα πλέγμα (grid, π.χ., $S \times S$). Για κάθε κελί του πλέγματος, προβλέπει ταυτόχρονα τις συντεταγμένες των bounding boxes και τις πιθανότητες των κλάσεων. Δεν υπάρχει ξεχωριστό βήμα πρότασης περιοχών.

- **Πλεονέκτημα:** Αυτή η προσέγγιση μειώνει δραματικά τον αριθμό των υπολογισμών. Το YOLOv8, συγκεκριμένα, μπορεί να επιτύχει ταχύτητες άνω των 30 FPS (Frames Per Second) σε σύγχρονες κινητές συσκευές, προσφέροντας την αίσθηση της άμεσης απόκρισης που είναι ζωτικής σημασίας για την αποφυγή ατυχημάτων [38].

Παρακάτω σε έναν συνοπτικό πίνακα συγκρίνουμε της τεχνολογικές προσεγγίσεις πίνακας 3.1.

Πίνακας 3.1: Σύγκριση τεχνολογικών προσεγγίσεων

Χαρακτηριστικό	**Faster R-CNN (Two-Stage)**	**SSD (One-Stage)**	**YOLOv8 (One-Stage)**
Μηχανισμός	Region Proposal + Classification	Multi-scale Features	Anchor-free Regression
Inference Time(Mobile GPU)	Υψηλό (~54ms - 200ms)	Μέτριο	Ελάχιστο (~1.3ms - 10ms)
Ακρίβεια (mAP)	Υψηλή	Μέτρια	State-of-the-Art (SOTA)
Κατανάλωση Ενέργειας	Υψηλή	Μέτρια	Βελτιστοποιημένη
Καταλληλότητα για Τυφλούς	Ακατάλληλο (λόγω latency)	Αποδεκτό	Ιδανικό (Speed/Accuracy balance)

3.5.3 Η Αρχιτεκτονική Yolov8

Το YOLOv8 (2023) της Ultralytics αποτελεί την επιλογή μας για την εφαρμογή. Ενσωματώνει καινοτομίες που το καθιστούν ανώτερο από τους προκατόχους του (YOLOv5, YOLOv7).

A. Backbone: CSPDarknet με C2f Modules

Η ραχοκοκαλιά του δικτύου χρησιμοποιεί το **C2f module** (Cross Stage Partial with 2 fusion), το οποίο αντικαθιστά το παλαιότερο C3 module. Το C2f συνδυάζει την αποτελεσματικότητα του CSPNet με πλουσιότερες συνδέσεις παράκαμψης (skip connections) εμπνευσμένες από το ELAN (Efficient Layer Aggregation Network). Αυτό επιτρέπει στο δίκτυο να μαθαίνει πιο πλούσια χαρακτηριστικά διατηρώντας το μοντέλο ελαφρύ. Η χρήση της συνάρτησης ενεργοποίησης **SiLU** ενισχύει περαιτέρω την απόδοση [39].

B. Neck: Feature Pyramid Network (PANet)

Το τμήμα "Neck" χρησιμοποιεί την αρχιτεκτονική PANet για να συνδυάσει χαρακτηριστικά από διαφορετικά επίπεδα ανάλυσης. Αυτό είναι κρίσιμο για την ανίχνευση αντικειμένων διαφορετικών μεγεθών. Ένα μικρό εμπόδιο (π.χ., πέτρα) απαιτεί υψηλή χωρική ανάλυση (από τα πρώτα επίπεδα), ενώ ένα μεγάλο (π.χ., τοίχος) απαιτεί σημασιολογική κατανόηση (από τα βαθύτερα επίπεδα). Το PANet επιτρέπει τη ροή πληροφορίας και προς τις δύο κατευθύνσεις (top-down και bottom-up) [32].

C. Head: Anchor-Free & Decoupled

Το YOLOv8 χρησιμοποιεί μια **Anchor-Free** προσέγγιση. Αντί να προσπαθεί να προσαρμόσει προκαθορισμένα κουτιά (anchors) στα αντικείμενα, προβλέπει απευθείας το κέντρο του αντικειμένου και την απόσταση προς τις τέσσερις πλευρές. Αυτό μειώνει τον αριθμό των παραμέτρων και επιταχύνει την εκπαίδευση. Επιπλέον, η κεφαλή είναι **Decoupled** (αποσυνδεδεμένη): χρησιμοποιεί ξεχωριστούς κλάδους για την ταξινόμηση και την παλινδρόμηση, καθώς αυτές οι δύο εργασίες έχουν διαφορετικές απαιτήσεις χαρακτηριστικών [31].

D. YOLOv8-seg: Μηχανισμός Τμηματοποίησης

Η μεγαλύτερη καινοτομία για την εφαρμογή μας είναι η υποστήριξη Instance Segmentation. Το YOLOv8-seg υιοθετεί τη μέθοδο YOLACT (You Only Look At CoefficienTs), αποφεύγοντας τα βαριά υπολογιστικά βήματα (όπως το ROIAlign του Mask R-CNN).

Η διαδικασία χωρίζεται σε δύο παράλληλα σκέλη:

Proto Module: Ένα δίκτυο FCN παράγει ένα σύνολο **k** "πρωτότυπων μασκών" (prototype masks, **P**) για ολόκληρη την εικόνα. Αυτές οι μάσκες κωδικοποιούν βασικά χωρικά μοτίβα.

Mask Coefficients: Η κεφαλή ανίχνευσης προβλέπει, για κάθε αντικείμενο, **k** συντελεστές μάσκας (coefficients, **C**).

Η τελική μάσκα **M** για κάθε αντικείμενο παράγεται μέσω ενός απλού γραμμικού συνδυασμού (πολλαπλασιασμός πινάκων), ο οποίος είναι εξαιρετικά γρήγορος σε GPU/NPU:

$$M = \sigma(P \times C^T) \#3.5.3$$

όπου σ είναι η σιγμοειδής συνάρτηση που μετατρέπει την έξοδο σε πιθανότητα (0 ή 1 για κάθε pixel). Αυτή η μέθοδος επιτρέπει real-time segmentation σε κινητά, κάτι που ήταν αδύνατο με παλαιότερες τεχνικές [35].

3.6 Μετρικές Αξιολόγησης και Μετρήσεις Ασφαλείας

Η επιτυχία ενός συστήματος υποβοήθησης δεν μετριέται μόνο με ακαδημαϊκούς όρους, αλλά με βάση την ασφάλεια και την εμπειρία του χρήστη.

3.6.1 Intersection over Union (IoU)

Το IoU είναι ο βασικός δείκτης χωρικής ακρίβειας. Μετρά το ποσοστό επικάλυψης μεταξύ της πρόβλεψης και της πραγματικότητας.

$$IoU = \frac{(Area\ of\ Overlap)}{(Area\ of\ Union)} \#3.6.1$$

Στο YOLOv8, κατά την εκπαίδευση, χρησιμοποιούνται εξελιγμένες μορφές του IoU, όπως το CIoU (Complete IoU) και το DFL (Distribution Focal Loss), οι οποίες λαμβάνουν υπόψη όχι μόνο την επικάλυψη αλλά και την απόσταση των κέντρων και τον λόγο πλευρών των πλαισίων, βελτιώνοντας την ταχύτητα σύγκλισης και την ακρίβεια εντοπισμού [40].

3.6.2 Precision,Recall και mAP

- **Precision (Ακρίβεια):** $TP \,/\, (TP + FP)$. Εκφράζει την αξιοπιστία των συναγερμών.

- **Recall (Ανάκληση):** $TP \,/\, (TP + FN)$. Εκφράζει την ικανότητα του συστήματος να βρίσκει όλα τα εμπόδια.

Για έναν τυφλό χρήστη, υπάρχει ένα κρίσιμο trade-off. Ένα σύστημα με χαμηλό Precision θα είναι ενοχλητικό (συνεχείς ψευδείς ειδοποιήσεις), αλλά ένα σύστημα με χαμηλό Recall είναι επικίνδυνο (δεν θα ειδοποιήσει για μια τρύπα στο δρόμο). Συνήθως, επιδιώκουμε μεγιστοποίηση του mAP (Mean Average Precision), το οποίο συνδυάζει και τα δύο.

- **mAP@50-95:** Η τυπική μετρική του COCO dataset. Υπολογίζει τον μέσο όρο του mAP για διάφορα κατώφλια IoU (από 0.5 έως 0.95). Υψηλό mAP@50-95 σημαίνει ότι το σύστημα εντοπίζει τα αντικείμενα με μεγάλη γεωμετρική ακρίβεια, κάτι απαραίτητο για τον υπολογισμό της απόστασης από το εμπόδιο [41].

3.6.3 Latency και Ανθρώπινος Χρόνος Αντίδρασης

Η πιο κρίσιμη παράμετρος για την ασφάλεια είναι ο λανθάνων χρόνος.

- **Human Reaction Time:** Ο μέσος χρόνος αντίδρασης σε ακουστικό ερέθισμα (όπως αυτό που θα παράγει η εφαρμογή) είναι περίπου 140-160ms, αλλά σε συνθήκες πολύπλοκης πλοήγησης μπορεί να αυξηθεί. Το φαινόμενο **StartReact** δείχνει ότι έντονα ακουστικά σήματα μπορούν να μειώσουν αυτόν τον χρόνο, αλλά το σύστημα πρέπει να είναι ταχύτατο [42].

- **System Latency:** Εάν ένας τυφλό χρήστης περπατά με ταχύτητα 1.0 m/s , σε 500ms (μισό δευτερόλεπτο) θα έχει διανύσει μισό μέτρο. Εάν το σύστημα (κάμερα + inference + audio) καθυστερήσει 500ms, ο χρήστης μπορεί να έχει ήδη συγκρουστεί. Επομένως, ο στόχος είναι το συνολικό latency να παραμένει κάτω από 100-200ms. Το YOLOv8n, με inference time < 10-20ms σε κινητά, αφήνει άφθονο χρονικό περιθώριο για την επεξεργασία ήχου και την αντίδραση του χρήστη, σε αντίθεση με το Faster R-CNN [43].

3.7 Περιορισμός Κινητών Συσκευών και Βελτιστοποίηση

Η εκτέλεση Βαθιάς Μάθησης σε Android απαιτεί ειδική διαχείριση πόρων.

- **Quantization (Κβαντισμός):** Η μετατροπή των βαρών του μοντέλου από FP32 (32-bit floating point) σε INT8 (8-bit integer) μπορεί να μειώσει το μέγεθος του μοντέλου κατά 4 φορές και να επιταχύνει το inference κατά 2-3 φορές, με ελάχιστη απώλεια ακρίβειας (mAP). Αυτό είναι κρίσιμο για τη χρήση των ειδικών μονάδων NPU (Neural Processing Units) που διαθέτουν τα σύγχρονα SoCs (π.χ. Snapdragon) [44].

Thermal Throttling: Η συνεχής χρήση της κάμερας και του επεξεργαστή μπορεί να υπερθερμάνει τη συσκευή, οδηγώντας σε μείωση της συχνότητας λειτουργίας (throttling) και πτώση των FPS. Τα ελαφριά μοντέλα όπως το YOLOv8n καταναλώνουν λιγότερη ενέργεια, διατηρώντας τη συσκευή δροσερή και την απόδοση σταθερή για μεγαλύτερο χρονικό διάστημα.

3.8 Συμπεράσματα Κεφαλαίου

Η βιβλιογραφική ανασκόπηση καταδεικνύει ότι η ανάπτυξη μιας εφαρμογής πλοήγησης για τυφλούς απαιτεί προσεκτική επιλογή αορίθμων. Η μετάβαση από τα παραδοσιακά ML μοντέλα στα Deep Learning CNNs είναι μονόδρομος για την επίτευξη υψηλής ακρίβειας. Μεταξύ των διαθέσιμων αρχιτεκτονικών, το **YOLOv8** ξεχωρίζει ως η βέλτιστη λύση. Η one-stage φύση του, σε συνδυασμό με βελτιστοποιήσεις όπως το C2f module και η Anchor-Free κεφαλή, προσφέρει την απαραίτητη ταχύτητα (χαμηλό latency) για ασφαλή πλοήγηση. Επιπλέον, η δυνατότητα **Instance Segmentation (YOLOv8-seg)** μέσω της μεθόδου πρωτοτύπων μασκών (Proto Module) παρέχει την απαραίτητη σημασιολογική λεπτομέρεια για την περιγραφή του σχήματος και των ορίων των εμποδίων, ξεπερνώντας τους περιορισμούς των απλών bounding boxes. Οι μετρικές αξιολόγησης (mAP@50-95, Inference Time) επιβεβαιώνουν ότι το YOLOv8 μπορεί να λειτουργήσει αποτελεσματικά σε περιβάλλον Android, καλύπτοντας τις αυστηρές απαιτήσεις πραγματικού χρόνου που επιβάλλει η φύση της υποβοηθητικής τεχνολογίας.

Κεφάλαιο 4ο: Αρχιτεκτονική YOLOv11 και Βελτιστοποίηση για Κινητά

4.1 Εισαγωγή και Τεχνικό Πλαίσιο Επεξεργασίας Edge AI

Η ραγδαία εξέλιξη της υπολογιστικής όρασης (Computer Vision) έχει μετατοπίσει το κέντρο βάρους από την κεντρική επεξεργασία σε συστοιχίες cloud προς την εκτέλεση αλγορίθμων απευθείας στις τερματικές συσκευές (Edge AI). Η παρούσα τεχνική αναφορά, η οποία αποτελεί το τέταρτο κεφάλαιο της ευρύτερης μελέτης ανάπτυξης, εστιάζει στην ενδελεχή ανάλυση της αρχιτεκτονικής YOLOv11 (You Only Look Once), όπως αυτή αναπτύχθηκε από την **Ultralytics** στα τέλη του 2024, και στην προσαρμογή της για εκτέλεση σε περιβάλλον Android μέσω του πλαισίου TensorFlow Lite (TFLite). Η επιλογή των μοντέλων **YOLOv11** Nano και YOLOv11-seg δεν είναι τυχαία· αντιπροσωπεύει τη βέλτιστη ισορροπία μεταξύ της υπολογιστικής ακρίβειας (mAP - mean Average Precision) και της λανθάνουσας καθυστέρησης (latency), δύο παραγόντων που βρίσκονται σε διαρκή ανταγωνισμό στα ενσωματωμένα συστήματα.

Η μετάβαση από παραδοσιακά μοντέλα ανίχνευσης σε αρχιτεκτονικές αιχμής όπως το YOLOv11 σε κινητές συσκευές επιβάλλει την πλήρη κατανόηση των δομικών στοιχείων του δικτύου. Από τον κορμό (backbone) που είναι υπεύθυνος για την εξαγωγή χαρακτηριστικών, μέχρι την κεφαλή (head) που εκτελεί την τελική πρόβλεψη, κάθε δομικός λίθος έχει επανασχεδιαστεί για να μεγιστοποιεί την απόδοση ανά υπολογιστικό κύκλο (FLOPS). Ιδιαίτερη έμφαση δίνεται στην ανάλυση των νέων δομικών στοιχείων C3k2 και C2PSA, τα οποία εισάγουν μηχανισμούς προσοχής (attention mechanisms) με χαμηλό υπολογιστικό κόστος, επιτρέποντας στο μοντέλο να εστιάζει σε κρίσιμες περιοχές της εικόνας χωρίς να επιβαρύνει υπερβολικά τον επεξεργαστή της κινητής συσκευής [45].

Επιπλέον, η εφαρμογή αυτών των μοντέλων σε περιβάλλον Android απαιτεί μια βαθιά τεχνική κατάδυση στις διαδικασίες κβαντισμού (quantization) και επιτάχυνσης υλικού. Η μετατροπή των βαρών του δικτύου από κινητή υποδιαστολή 32-bit (FP32) σε ακέραιους 8-bit (Int8) ή κινητή υποδιαστολή 16-bit (FP16) αποτελεί μονόδρομο για την επίτευξη πραγματικού χρόνου (Real-Time), ωστόσο συνοδεύεται από προκλήσεις ακρίβειας που πρέπει να αντιμετωπιστούν με εξειδικευμένες στρατηγικές εκπαίδευσης και εξαγωγής. Η κατάργηση του NNAPI (Android Neural Networks API) στο Android 15 και η στροφή προς το GPU Delegate μέσω των Google Play Services διαμορφώνουν ένα νέο τοπίο υλοποίησης που απαιτεί προσεκτική πλοήγηση [46].

4.2 Η Εξέλιξη της Αρχιτεκτονικής YOLO: Από το Πλέγμα στο Anchor-Free

Για να κατανοήσουμε την υπεροχή του YOLOv11, είναι απαραίτητο να αναλύσουμε την εξελικτική πορεία της οικογένειας YOLO. Η μετάβαση από την άκαμπτη δομή του πλέγματος (grid) στην ευελιξία των Anchor-Free μηχανισμών αποτελεί μια θεμελιώδη αλλαγή στον τρόπο που τα νευρωνικά δίκτυα αντιλαμβάνονται τον χώρο και τα αντικείμενα.

4.2.1 Η Εποχή του Πλέγματος (YOLOv1) και οι Περιορισμοί της

Το αρχικό YOLOv1, που παρουσιάστηκε το 2015, έφερε επανάσταση ενοποιώντας τα στάδια εξαγωγής χαρακτηριστικών και ανίχνευσης σε ένα ενιαίο συνελικτικό δίκτυο. Η βασική του καινοτομία ήταν η διαίρεση της εικόνας εισόδου σε ένα πλέγμα **S\times** (συνήθως **7\times**). Εάν το κέντρο ενός αντικειμένου ενέπιπτε σε ένα συγκεκριμένο κελί, το κελί αυτό αναλάμβανε την ευθύνη για την ανίχνευσή του [47].

Κάθε κελί του πλέγματος προέβλεπε **B** οριοθετημένα πλαίσια (bounding boxes) και βαθμολογίες εμπιστοσύνης (confidence scores), καθώς και **C** πιθανότητες κλάσεων υπό συνθήκη. Η μαθηματική διατύπωση για την εμπιστοσύνη ήταν:

$$C = P(Object) \cdot IoU_{\{pred\}}^{\{truth\}} \#4.2.1$$

Όπου **IoU** είναι η Τομή επί της Ένωσης μεταξύ του προβλεπόμενου και του πραγματικού πλαισίου. Παρά την ταχύτητά του, το YOLOv1 υπέφερε από σοβαρούς περιορισμούς, κυρίως λόγω της αδυναμίας του να ανιχνεύσει μικρά αντικείμενα που συγκεντρώνονταν στο ίδιο κελί ("smal object grouping"), καθώς κάθε κελί μπορούσε να προβλέψει μόνο μία κλάση. Σε ένα σύγχρονο σενάριο Edge AI, όπως η ανάλυση κίνησης σε κινητό τηλέφωνο, αυτό θα καθιστούσε αδύνατη την ανίχνευση πεζών που βρίσκονται σε απόσταση ή σε πυκνές ομάδες [48].

4.2.2 Η Εισαγωγή των Anchor Boxes (YOLOv2 – YOLOv7)

Για να αντιμετωπιστούν τα προβλήματα εντοπισμού και γενίκευσης μεγέθους, οι επόμενες εκδόσεις (v2 έως v5 και v7) υιοθέτησαν την έννοια των "Anchor Boxes" (Πλαίσια Αναφοράς). Αντί το δίκτυο να προσπαθεί να προβλέψει απευθείας τις διαστάσεις ενός αντικειμένου από το μηδέν, μάθαινε να προβλέπει μετατοπίσεις (offsets) από προκαθορισμένα πρότυπα πλαίσια [48].

Τα πλαίσια αυτά καθορίζονταν συνήθως μέσω αλγορίθμου ομαδοποίησης K-means στο σύνολο δεδομένων εκπαίδευσης. Οι εξισώσεις μετασχηματισμού για την πρόβλεψη των συντεταγμένων **(b_x, b_y, b_w, b_h)** είχαν τη μορφή:

$$b_x = \sigma(t_x) + c_x$$

$$b_y = \sigma(t_y) + c_y$$

$$b_w = p_w \cdot e^{\{t_w\}}$$

$$b_h = p_h \cdot e^{\{t_h\}} \#4.2.2$$

Όπου **(t_x, t_y, t_w, t_h)** οι έξοδοι του δικτύου, **(c_x, c_y)** η θέση του κελιού στο πλέγμα, και **(p_w, p_h)** οι διαστάσεις του anchor box.

Κριτική Ανάλυση για Edge AI:

Ενώ τα Anchor Boxes βελτίωσαν σημαντικά την ανάκληση (recall), εισήγαγαν σημαντική πολυπλοκότητα. Για ένα μοντέλο που τρέχει σε Android, η χρήση anchors σημαίνει ότι η κεφαλή ανίχνευσης πρέπει να επεξεργαστεί πολλαπλάσια υποψήφια πλαίσια (συνήθως 3 anchors ανά επίπεδο κλίμακας). Αυτό αυξάνει δραματικά τον αριθμό των λειτουργιών που απαιτούνται για την τελική φιλτράρισμα μέσω Non-Maximum Suppression (NMS), μια διαδικασία που είναι υπολογιστικά δαπανηρή και δύσκολα παραλληλίζεται σε NPU.10 Επιπλέον, η εξάρτηση από τα anchors καθιστά το μοντέλο λιγότερο προσαρμοστικό σε νέα σενάρια χωρίς επανακαθορισμό των υπερπαραμέτρων των anchors [31].

4.2.3 Μετάβαση σε Anchor Free Σχεδιασμό (YOLOv8 – YOLOv11)

Το YOLOv11 εδραιώνει την αλλαγή παραδείγματος που ξεκίνησε με το YOLOv8, υιοθετώντας πλήρως την Anchor-Free αρχιτεκτονική. Σε αυτή την προσέγγιση, το μοντέλο δεν βασίζεται σε προκαθορισμένα πλαίσια αλλά προβλέπει απευθείας το κέντρο του αντικειμένου και τις αποστάσεις από αυτό το κέντρο προς τις τέσσερις πλευρές του οριοθετημένου πλαισίου [49].

Η θεμελιώδης διαφορά έγκειται στην αντιμετώπιση του προβλήματος παλινδρόμησης. Αντί για απλή πρόβλεψη συντεταγμένων, το YOLOv11 χρησιμοποιεί **Distribution Focal Loss (DFL)**. Το δίκτυο προβλέπει μια κατανομή πιθανοτήτων για τη θέση κάθε πλευράς του κουτιού. Αυτό επιτρέπει στο μοντέλο να εκφράσει αβεβαιότητα, η οποία είναι κρίσιμη για περιπτώσεις όπου τα όρια του αντικειμένου είναι ασαφή (π.χ., θόλωμα κίνησης σε κάμερα κινητού).

Η μαθηματική προσέγγιση της παλινδρόμησης bounding box στο YOLOv11 βασίζεται στην ολοκλήρωση της κατανομής:

$$\{y\} = \int_{\{y_{\{min\}}\}}^{\{y_{\{max\}}\}P(y)} * y \#4.2.3$$

Στην πράξη, αυτό υλοποιείται μέσω μιας στρώσης **Softmax** και ενός συνελικτικού επιπέδου που υπολογίζει την αναμενόμενη τιμή της θέσης.

Πλεονεκτήματα για Κινητές Συσκευές:

1. **Μείωση Παραμέτρων Κεφαλής:** Η αφαίρεση των anchors μειώνει τον αριθμό των φίλτρων στο τελευταίο συνελικτικό επίπεδο, καθώς δεν χρειάζεται πλέον πολλαπλασιασμός των εξόδων επί τον αριθμό των anchors **A**. Αυτό οδηγεί σε ταχύτερη infernece (συμπερασμό) και μικρότερο μέγεθος μοντέλου, καθιστώντας το YOLOv11n ιδανικό για εφαρμογές Android με περιορισμένη μνήμη [31].
2. **Βελτιωμένη Γενίκευση:** Το μοντέλο γίνεται πιο ανθεκτικό σε αντικείμενα με ακραίες αναλογίες διαστάσεων (aspect ratios), κάτι συνηθισμένο σε ευρυγώνιους φακούς κινητών συσκευών.

4.3 Αναλυτική Αρχιτεκτονική YOLOv11 : Καινοτομίες και Βελτιστοποίηση

Η αρχιτεκτονική του YOLOv11 αποτελεί μια εκλεπτυσμένη εκδοχή της δομής CSPNet (Cross Stage Partial Network), με στοχευμένες παρεμβάσεις για τη βελτίωση της ροής της πληροφορίας και της αποδοτικότητας των παραμέτρων. Για την έκδοση Nano (YOLOv11n), κάθε δομικό στοιχείο έχει βελτιστοποιηθεί για ελαχιστοποίηση των πράξεων κινητής υποδιαστολής (FLOPs).

4.3.1 Ο Κορμός και το Block C3k2

Η καρδιά της εξαγωγής χαρακτηριστικών στο YOLOv11 είναι το νέο δομικό στοιχείο **C3k2**. Πρόκειται για μια εξέλιξη του C2f (από το YOLOv8), το οποίο με τη σειρά του ήταν βελτίωση του C3 (από το YOLOv5) [5].

Λειτουργική Δομή:

Το C3k2 (Cross Stage Partial with selectable kernel) εισάγει δυναμική προσαρμογή στο μέγεθος του πυρήνα συνέλιξης. Η βασική του αρχή βασίζεται στη διαίρεση της ροής δεδομένων:

1. Ο τανυστής εισόδου (input tensor) διαχωρίζεται σε δύο κλάδους.
2. Ο ένας κλάδος περνά μέσα από μια σειρά πυκνών επιπέδων (Bottlenecks) που εκτελούν την επεξεργασία χαρακτηριστικών.
3. Ο άλλος κλάδος λειτουργεί ως παράκαμψη (residual connection) για τη διατήρηση της πληροφορίας βαθμίδας.
4. Οι δύο κλάδοι συνενώνονται (Concat) και περνούν από μια τελική συνέλιξη $1\,X1$

Η παράμετρος c3k καθορίζει τη συμπεριφορά του block. Όταν c3k=False (όπως συχνά συμβαίνει στα ελαφριά μοντέλα Nano), το block λειτουργεί με τυποποιημένα bottlenecks για μέγιστη ταχύτητα. Όταν c3k=True, χρησιμοποιούνται πυρήνες μεταβλητού μεγέθους για προσαρμοστική λήψη πεδίου. Αυτή η δομή επιτρέπει στο YOLOv11 να επιτυγχάνει πλουσιότερη αναπαράσταση χαρακτηριστικών με λιγότερες παραμέτρους, μειώνοντας το φορτίο στη μνήμη RAM της συσκευής Android [41].

4.3.2 Spatial Pyramid Pooling – Fast (SPPF)

Στο τέλος του κορμού, το YOLOv11 διατηρεί το module **SPPF** (Spatial Pyramid Pooling - Fast). Ο ρόλος του είναι να αυξήσει το δεκτικό πεδίο (receptive field) του δικτύου, επιτρέποντάς του να "βλέπει" το αντικείμενο στο ευρύτερο πλαίσιο της εικόνας [50].

Αντί για παράλληλες συνελιξείς διαφορετικών μεγεθών, το SPPF χρησιμοποιεί σειριακές πράξεις Max Pooling μεγέθους $5X5$. Η έξοδος κάθε pooling τροφοδοτεί το επόμενο, και όλες οι ενδιάμεσες έξοδοι συνενώνονται.

$$Out_1 = MaxPool(Input)$$

$$Out_2 = MaxPool(Out_1)$$

$$Out_3 = MaxPool(Out_2)$$

$$Output = Concat([Input, Out_1, Out_2, Out_3])\#4.3.2$$

Αυτή η προσέγγιση είναι εξαιρετικά αποδοτική για το TFLite, καθώς οι πράξεις Pooling είναι υπολογιστικά φθηνές και πλήρως επιταχυνόμενες από τις περισσότερες μονάδες NPU και GPU των κινητών (Adreno, Mali).

4.3.3 Η Καινοτομία του C2PSA (Cross-Stage Partial Spatial Attention)

Η σημαντικότερη προσθήκη στο YOLOv11 είναι το module **C2PSA**, το οποίο ενσωματώνεται μετά το SPPF. Πρόκειται για έναν μηχανισμό προσοχής (attention mechanism) σχεδιασμένο ειδικά για να είναι ελαφρύς [45].

Οι παραδοσιακοί μηχανισμοί προσοχής (όπως οι Transformers) έχουν τετραγωνική πολυπλοκότητα $O(N\hat{}2)$ ως προς το μέγεθος της εισόδου, καθιστώντας τους απαγορευτικούς για Edge AI. Το C2PSA, αντιθέτως, εφαρμόζει **χωρική προσοχή (spatial attention)** με χαμηλό κόστος.

1. **Διαχωρισμός Χαρακτηριστικών:** Όπως και στα CSP blocks, η είσοδος χωρίζεται.
2. **Υπολογισμός Βαρών Προσοχής:** Ένας κλάδος επεξεργάζεται τα χαρακτηριστικά για να παράγει έναν χάρτη προσοχής (attention map), χρησιμοποιώντας πιθανώς Global Average Pooling και συνελιξείς $1X1$ ακολουθούμενες από σιγμοειδή συνάρτηση (Sigmoid) για να κανονικοποιήσει τις τιμές στο διάστημα.
3. **Εφαρμογή Προσοχής:** Ο χάρτης προσοχής πολλαπλασιάζεται στοιχείο-προς-στοιχείο (element-wise multiplication) με τα χαρακτηριστικά, ενισχύοντας τις περιοχές που περιέχουν πληροφορία και καταστέλλοντας το υπόβαθρο.

Αυτή η ικανότητα εστίασης είναι κρίσιμη για την ανίχνευση μικρών ή επικαλυπτόμενων αντικειμένων, ένα σενάριο όπου τα προηγούμενα μοντέλα Nano υστερούσαν [26].

4.3.4 Αποσυνδεδεμένη Κεφαλή (Decoupled Head)

Σε αντίθεση με παλαιότερες υλοποιήσεις όπου η ταξινόμηση και η παλινδρόμηση μοιράζονταν τα ίδια συνελικτικά βάρη στην έξοδο, το YOLOv11 χρησιμοποιεί μια αποσυνδεδεμένη κεφαλή.

- **Κλάδος Ταξινόμησης:** Χρησιμοποιεί Depth-wise Separable Convolutions (DWConv). Αυτή είναι μια κρίσιμη βελτιστοποίηση για κινητά, μειώνοντας τα FLOPs κατά συντελεστή περίπου 8-9 φορές σε σύγκριση με τις κανονικές συνελιξείς [51].
- **Κλάδος Παλινδρόμησης:** Εστιάζει αποκλειστικά στον εντοπισμό των ορίων του αντικειμένου μέσω του DFL.

4.4 Σύγκριση Ανίχνευσης (Detection) vs Τμηματοποίησης (Segmentation)

Στο πλαίσιο της εφαρμογής σε Android, η επιλογή μεταξύ YOLOv11 Nano (Detect) και YOLOv11 Nano-seg (Segmentation) αποτελεί μια κρίσιμη απόφαση σχεδιασμού, με άμεσες επιπτώσεις στην ταχύτητα και την κατανάλωση πόρων.

4.4.1 Αρχιτεκτονική Διαφορά: Η Προσέγγιση YOLOAct

Το YOLOv11-seg δεν αποτελεί μια ξεχωριστή αρχιτεκτονική αλλά μια επέκταση του μοντέλου ανίχνευσης, βασισμένη στη μεθοδολογία **YOLACT (You Only Look At CoefficienTs)**.[23] Αυτή η προσέγγιση επιτρέπει την εκτέλεση τμηματοποίησης στιγμιοτύπου (instance segmentation) σε πραγματικό χρόνο, αποφεύγοντας τις αργές μεθόδους δύο σταδίων (όπως το Mask R-CNN) [52].

Το μοντέλο τμηματοποίησης προσθέτει δύο βασικά στοιχεία:

1. **Κλάδος Πρωτοτύπων (Proto Module):** Μια πλήρως συνελικτική δομή που παράγει ένα σύνολο **k** "μασκών πρωτοτύπων" (prototype masks). Αυτές οι μάσκες δεν αντιστοιχούν σε συγκεκριμένα αντικείμενα, αλλά κωδικοποιούν βασικά χωρικά χαρακτηριστικά της εικόνας (ακμές, υφές, σχήματα). Ο αριθμός των πρωτοτύπων είναι συνήθως 32 $k = 32$ και η ανάλυσή τους είναι το $1/4$ της αρχικής εικόνας.
2. **Συντελεστές Μάσκας (Mask Coefficients):** Η κεφαλή ανίχνευσης, εκτός από το bounding box και την κλάση, προβλέπει για κάθε αντικείμενο ένα διάνυσμα **k** συντελεστών.

4.4.2 Μαθηματική Σύνθεση και Υπολογιστικό Κόστος

Η τελική μάσκα για ένα αντικείμενο παράγεται μέσω γραμμικού συνδυασμού των πρωτοτύπων και των συντελεστών, ακολουθούμενη από μια σιγμοειδή ενεργοποίηση:

$$M_{final} = \sigma(P \times C^T) \#4.4.2$$

Όπου:

- **P** είναι ο τανυστής πρωτοτύπων μεγέθους $H_{mask} \times W_{mask} \times k$
- **C** είναι ο πίνακας συντελεστών των ανιχνευθέντων αντικειμένων μεγέθους $N \times k$.

Επιπτώσεις Latency σε Android:

Ενώ το νευρωνικό δίκτυο (Inference) εκτελείται γρήγορα στην NPU/GPU, η διαδικασία σύνθεσης της μάσκας $P \times C^T$ συχνά λαμβάνει χώρα στο στάδιο της μετα-επεξεργασίας (Post-Processing).

- **Detection (YOLOv11n):** Η μετα-επεξεργασία περιλαμβάνει μόνο NMS στα bounding boxes.
- **Segmentation (YOLOv11n-seg):** Απαιτεί τον πολλαπλασιασμό πινάκων υψηλής ανάλυσης και στη συνέχεια την αποκοπή (cropping) της μάσκας στα όρια του bounding box. Σε κινητές συσκευές, εάν αυτή η διαδικασία γίνει στην CPU (Java/Kotlin layer), μπορεί να προσθέσει 20-30ms καθυστέρηση ανά καρέ, μειώνοντας δραστικά τα FPS [31].

4.4.3 Συκγριτικός Πίνακας Απόδοσης

Ο παρακάτω πίνακας συνοψίζει τις διαφορές απόδοσης, βασισμένος σε συγκεντρωτικά δεδομένα από benchmarks σε συσκευές μεσαίας κατηγορίας (όπως Pixel 6/7) πίνακας 4.1.

Πίνακας 4.1: Benchmarks YOLOv11

Χαρακτηριστικό	**YOLOv11n (Detection)**	**YOLOv11n-seg (Segmentation)**	**Επίπτωση στο Edge AI**
Παράμετροι	~2.6 M	~3.2 M	+23% αύξηση μεγέθους αρχείου TFLite.
GFLOPs	~6.5 G	~8.9 G	Αυξημένη κατανάλωση ενέργειας μπαταρίας.
Inference Time (NPU)	~10-15 ms	~18-24 ms	Η τμηματοποίηση παραμένει Real-Time (>30FPS) αλλά οριακά.
Post-Processing	Χαμηλό (Box NMS)	Υψηλό (Mask Assembly)	Κίνδυνος CPU bottleneck στην εφαρμογή Android.
Ακρίβεια (mAP)	Υψηλή στα κουτιά	Υψηλή + Pixel Precision	Η τμηματοποίηση προσφέρει καλύτερη κατανόηση σκηνής.

4.5 Βελτιστοποίηση Edge AI: TFlite, Κβαντισμός και Hardware Acceleration

Η επιτυχής εκτέλεση του YOLOv11 σε Android δεν εξαρτάται μόνο από την αρχιτεκτονική του μοντέλου, αλλά και από τη διαδικασία μετατροπής και την εκμετάλλευση του υλικού.

4.5.1 Η Διαδικασία Εξαγωγής σε TFlite

Η ροή εργασίας (pipeline) για τη μεταφορά του μοντέλου στο Android περιλαμβάνει τα εξής βήματα:

1. **PyTorch Training:** Εκπαίδευση του μοντέλου (αρχείο .pt).
2. **Export to ONNX:** Εξαγωγή σε μορφή ONNX. Σε αυτό το στάδιο είναι κρίσιμο να οριστούν σταθερές διαστάσεις εισόδου (π.χ. 640×640), καθώς πολλές μονάδες NPU δεν υποστηρίζουν δυναμικά σχήματα (dynamic shapes).
3. **TFLite Conversion:** Μετατροπή του ONNX σε FlatBuffer (.tflite).

4.5.2 Κβαντισμός: Int8 vs FP16

Ο κβαντισμός είναι η διαδικασία μείωσης της ακρίβειας αναπαράστασης των βαρών για εξοικονόμηση μνήμης και αύξηση ταχύτητας.

FP16 (Half-Precision):

- Μετατρέπει τα βάρη από 32-bit σε 16-bit float.
- **Πλεονέκτημα:** Ελάχιστη απώλεια ακρίβειας (<0.5% mAP). Υποστηρίζεται εγγενώς από το GPU Delegate του Android.
- **Μειονέκτημα:** Μείωση μεγέθους μόνο κατά 50% [41].

Int8 (8-bit Integer):

- Μετατρέπει τα βάρη σε ακεραίους 8-bit. Η σχέση δίνεται από τον αφινικό μετασχηματισμό:

$$r = S \cdot (q - Z) \#4.5.2$$

Όπου r η πραγματική τιμή, q η κβαντισμένη τιμή, S η κλίμακα (scale) και Z το σημείο μηδενισμού (zero-point).

- **Πρόβλημα Ακρίβειας:** Τα μοντέλα YOLO, και ειδικά το v11 με την κεφαλή παλινδρόμησης DFL, είναι ευαίσθητα στον Int8 κβαντισμό. Αναφορές χρηστών δείχνουν πτώση ακρίβειας 6-15% εάν δεν γίνει προσεκτική διαμέριση (calibration).[29] Οι συναρτήσεις ενεργοποίησης SiLU, που είναι μη φραγμένες στο θετικό ημιάξονα, μπορούν να δημιουργήσουν μεγάλο εύρος τιμών, καθιστώντας την κλίμακα **S** ανακριβή για τις μικρές τιμές που είναι κρίσιμες για τον εντοπισμό.

- **Λύση:** Απαιτείται **Full Integer Quantization** με αντιπροσωπευτικό σύνολο δεδομένων (representative dataset) κατά τη μετατροπή, ή ακόμα καλύτερα **Quantization-Aware Training (QAT)**, όπου το μοντέλο εκπαιδεύεται προσομειώνοντας το σφάλμα κβαντισμού.

4.5.3 Επιτάχυνση Υλικού: NNAPI vs GPU Delegate

Ένα κρίσιμο εύρημα για την ανάπτυξη σε Android το 2025-2026 είναι η κατάσταση του NNAPI.

- **NNAPI (Deprecated):** Η Google έχει χαρακτηρίσει το NNAPI ως παρωχημένο (deprecated) στο Android 15. Αν και λειτουργικό, δεν αποτελεί πλέον την προτεινόμενη οδό [15].
- **TFLite GPU Delegate:** Η προτεινόμενη μέθοδος για επιτάχυνση. Ωστόσο, υπάρχουν αναφορές ασυμβατότητας (crashes) συγκεκριμένων τελεστών (ops) του YOLOv11 με το GPU delegate, πιθανώς λόγω των νέων υλοποιήσεων στο C2PSA block που απαιτούν λειτουργίες broadcast ή gather που δεν έχουν βελτιστοποιηθεί στους drivers της GPU.
- **XNNPACK:** Για μέγιστη συμβατότητα και σταθερότητα, η εκτέλεση στην CPU με χρήση του XNNPACK (βελτιστοποιημένη βιβλιοθήκη για floating-point inference) είναι συχνά η πιο ασφαλής επιλογή, ειδικά για τα μοντέλα Nano που είναι αρκετά ελαφριά ώστε να τρέχουν γρήγορα ακόμη και στην CPU.

4.5.4 Διαχείρση NMS εντός του TFlite

Στο YOLOv11, το NMS είναι απαραίτητο (σε αντίθεση με το πειραματικό YOLOv10). Η εκτέλεση του NMS σε κώδικα Java/Kotlin είναι αργή. Η βέλτιστη πρακτική είναι η ενσωμάτωση του NMS ως custom operator εντός του γράφου TFLite κατά την εξαγωγή (nms=True στο Ultralytics export). Αυτό επιτρέπει στο TFLite να εκτελέσει την ταξινόμηση και την απόρριψη των κουτιών σε επίπεδο C++, επιστρέφοντας στην εφαρμογή μόνο τα τελικά αποτελέσματα, μειώνοντας δραματικά το χρόνο μεταφοράς δεδομένων [53].

4.6 Συμπεράσματα

Η αρχιτεκτονική YOLOv11 αποτελεί ένα σημαντικό βήμα προς τα εμπρός για την τεχνητή νοημοσύνη σε κινητές συσκευές, συνδυάζοντας την αποδοτικότητα του Anchor-Free σχεδιασμού με έξυπνους μηχανισμούς προσοχής όπως το C2PSA. Για την υλοποίηση του Κεφαλαίου 4, η χρήση του YOLOv11n προσφέρει την καλύτερη σχέση ταχύτητας/ακρίβειας. Εάν απαιτείται τμηματοποίηση, το YOLOv11n-seg είναι βιώσιμο, αρκεί να δοθεί προσοχή στη βελτιστοποίηση της μετα-επεξεργασίας των μασκών. Η επιλογή μεταξύ FP16 και Int8 πρέπει να γίνει με βάση τις ανοχές ακρίβειας της εφαρμογής, ενώ η στροφή μακριά από το NNAPI προς το GPU Delegate ή το XNNPACK είναι επιβεβλημένη για τη διασφάλιση της μελλοντικής συμβατότητας της εφαρμογής Android.

Κεφάλαιο 5ο: Μεθοδολογία Ανάπτυξης, Υλοποίηση και Πειραματική Αξιολόγηση

5.1 Εισαγωγή

Το παρόν κεφάλαιο περιγράφει αναλυτικά τη μεθοδολογία που ακολουθήθηκε για την ανάπτυξη του συστήματος πλοήγησης, εστιάζοντας στην αρχιτεκτονική λογισμικού, τους αλγορίθμους μηχανικής όρασης και τις τεχνικές βελτιστοποίησης για κινητές συσκευές. Η υλοποίηση βασίστηκε στον συνδυασμό μοντέλων YOLOv11 (Edge AI) με αλγορίθμους γεωμετρικής εκτίμησης απόστασης (Geometric Distance Estimation), αξιοποιώντας δεδομένα από τους αισθητήρες της συσκευής (Sensor Fusion).

5.2 Διεπαφή Χρήστη και απτική αλληλεπίδραση

Η διεπαφή χρήστη της εφαρμογής έχει σχεδιαστεί με γνώμονα την απλότητα και τη δυνατότητα χρήσης χωρίς φυσική αλληλεπίδραση με την οθόνη, αξιοποιώντας φωνητικές εντολές. Παρακάτω είναι η εικόνα 5 οπού βλέπουμε την αρχικής οθόνης της εφαρμογής.

Εικόνα 5.1: Η αρχική οθόνη της εφαρμογής

Η αρχική οθόνη της εφαρμογής προβάλλει σε πραγματικό χρόνο την εικόνα από την κάμερα της συσκευής. Πάνω στην οθονη εμφανίζονται τα αποτελέσματα της διαδικασίας ανίχνευσης αντικειμένων, τα οποία περιλαμβάνουν:

- Πλαίσια οριοθέτησης για κάθε ανιχνευμένο αντικείμενο
- Το όνομα της κατηγορίας του αντικειμένου
- τον βαθμό εμπιστοσύνης
- Την εκτιμώμενη απόσταση του αντικειμένου από τη συσκευή σε μέτρα

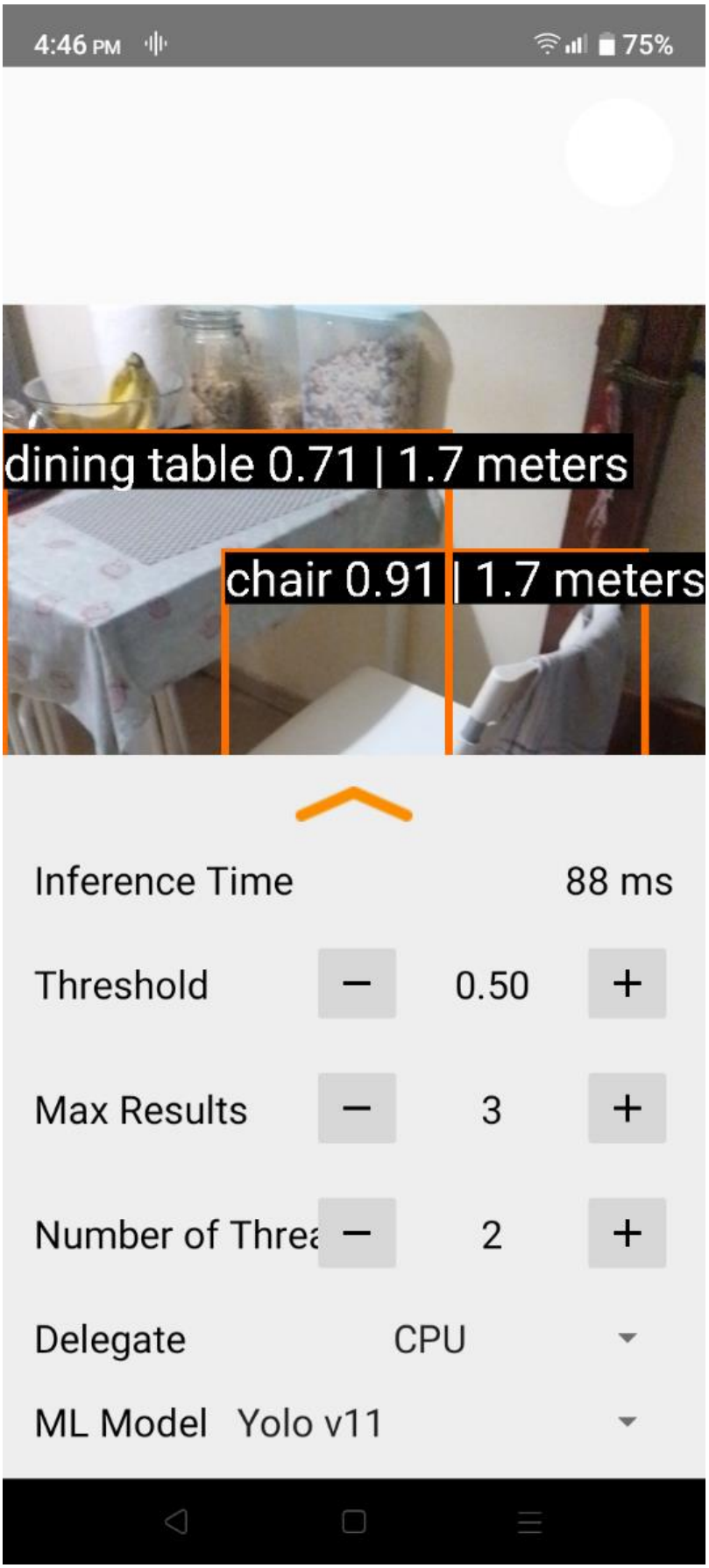
Επιπλέον, η εφαρμογή υποστηρίζει αλληλεπίδραση μέσω αφής με τα ανιχνευμένα αντικείμενα. Όταν ο χρήστης ακουμπά με το δάχτυλό του ένα εντοπισμένο αντικείμενο στην οθόνη, παρέχεται άμεση φωνητική ανατροφοδότηση (voice feedback), η οποία αναφέρει το όνομα του αντικειμένου και την απόστασή του, όπως για παράδειγμα «ποτήρι, απόσταση 1.4 μέτρα».

Παράλληλα, ενεργοποιείται απτική ανατροφοδότηση (haptic feedback), κατά την οποία η συσκευή δονείται και συνεχίζει να δονείται όσο το δάχτυλο του χρήστη παραμένει εντός του πλαισίου οριοθέτησης του αντικειμένου. Με τον τρόπο αυτό, ο χρήστης μπορεί να αντιληφθεί τόσο την παρουσία όσο και τη χωρική κατεύθυνση του αντικειμένου, διευκολύνοντας την πλοήγηση και την αλληλεπίδραση με το περιβάλλον.

Η μετάβαση στον πίνακα ρυθμίσεων πραγματοποιείται μέσω φωνητικής εντολής **("four")**, χρησιμοποιώντας το σύστημα Voice Access, χωρίς να απαιτείται άγγιγμα της οθόνης.

Στην οθόνη ρυθμίσεων παρέχονται οι εξής επιλογές:

- **Threshold**: Ρύθμιση του κατωφλίου εμπιστοσύνης για την εμφάνιση των ανιχνευμένων αντικειμένων
- **Max Results**: Καθορισμός του μέγιστου αριθμού αντικειμένων που εμφανίζονται ταυτόχρονα
- **Number of Threads**: Ρύθμιση του αριθμού νημάτων επεξεργασίας για τη βελτίωση της απόδοσης



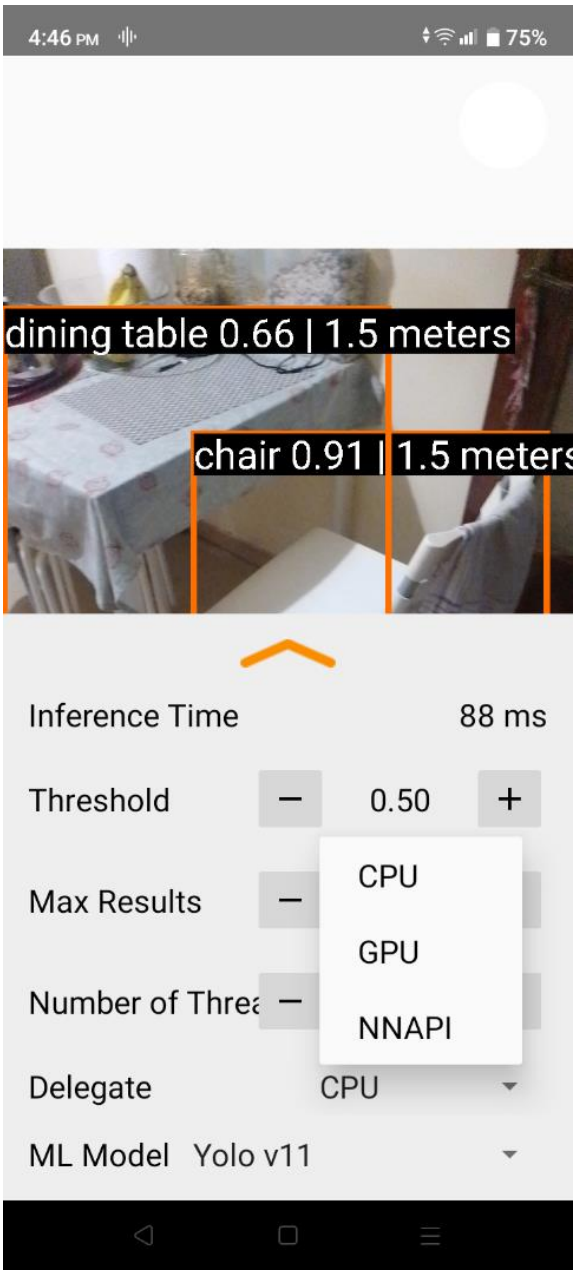
Εικόνα 5.2: Ο πίνακας ρυθμίσεων της εφαρμογής

Επιλογή Delegate

Ο χρήστης μπορεί να επιλέξει το υποσύστημα εκτέλεσης του μοντέλου μηχανικής μάθησης μεταξύ:

- **CPU**
- **GPU**
- **NNAPI**

Μέσω μιας αναδυόμενης λίστας όπως φαίνεται στην εικόνα 7

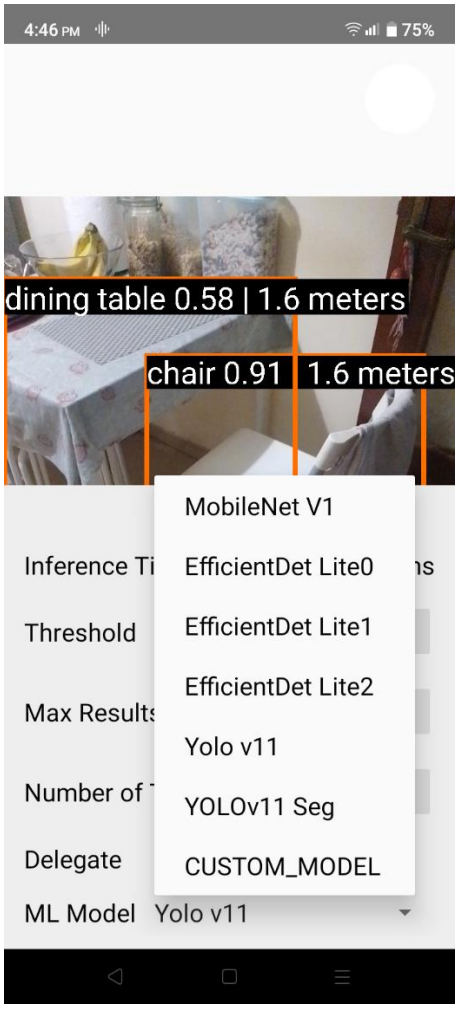


Εικόνα 5.3: Η λίστα με της επιλογές delegate

Επιλογή Μοντέλου Μηχανικής Μάθησης

Η εφαρμογή υποστηρίζει πολλαπλά μοντέλα μηχανικής μάθησης, τα οποία μπορούν να επιλεγούν δυναμικά από τον χρήστη, μέσω μιας αναδυόμενης λίστας όπως φαίνεται στην εικόνα 8:

- **MobileNetV1**
- **Efficient Lite0**
- **Efficient Lite1**
- **Efficient Lite2**
- **YOLO v11**
- **YOLOv11 Seg**
- **CUSTOM_MODEL**



Εικόνα 5.4: Η λίστα με της επιλογές μοντέλων

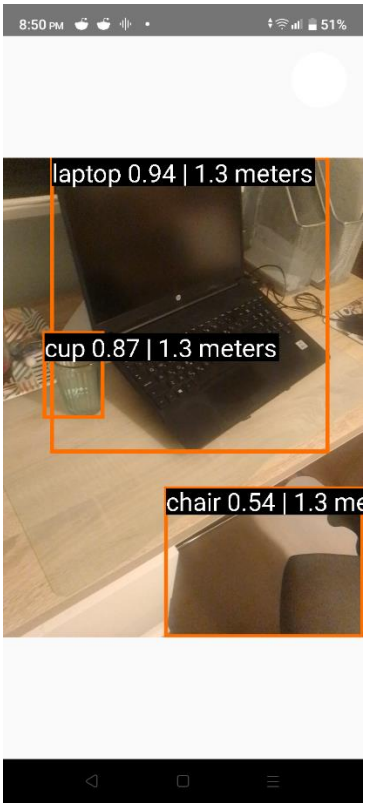
Τα τρία νέα μοντέλα μηχανικής μάθησης επεκτείνουν τη λειτουργικότητα και βελτιώνουν την ακρίβεια της αναγνώρισης αντικειμένων συγκεκριμένα:

- **YOLO v11**

Σύγχρονο και ιδιαίτερα ακριβές προεκπαιδευμένο μοντέλο ανίχνευσης αντικειμένων, το οποίο χρησιμοποιείται για αναγνώριση σε πραγματικό χρόνο και παρέχει υψηλό βαθμό εμπιστοσύνης στις προβλέψεις του. Παρόλα αυτά, κατά τη χρήση του σε σκηνές με αντικείμενα που επικαλύπτονται χωρικά, παρατηρήθηκαν περιπτώσεις εσφαλμένης αναγνώρισης.

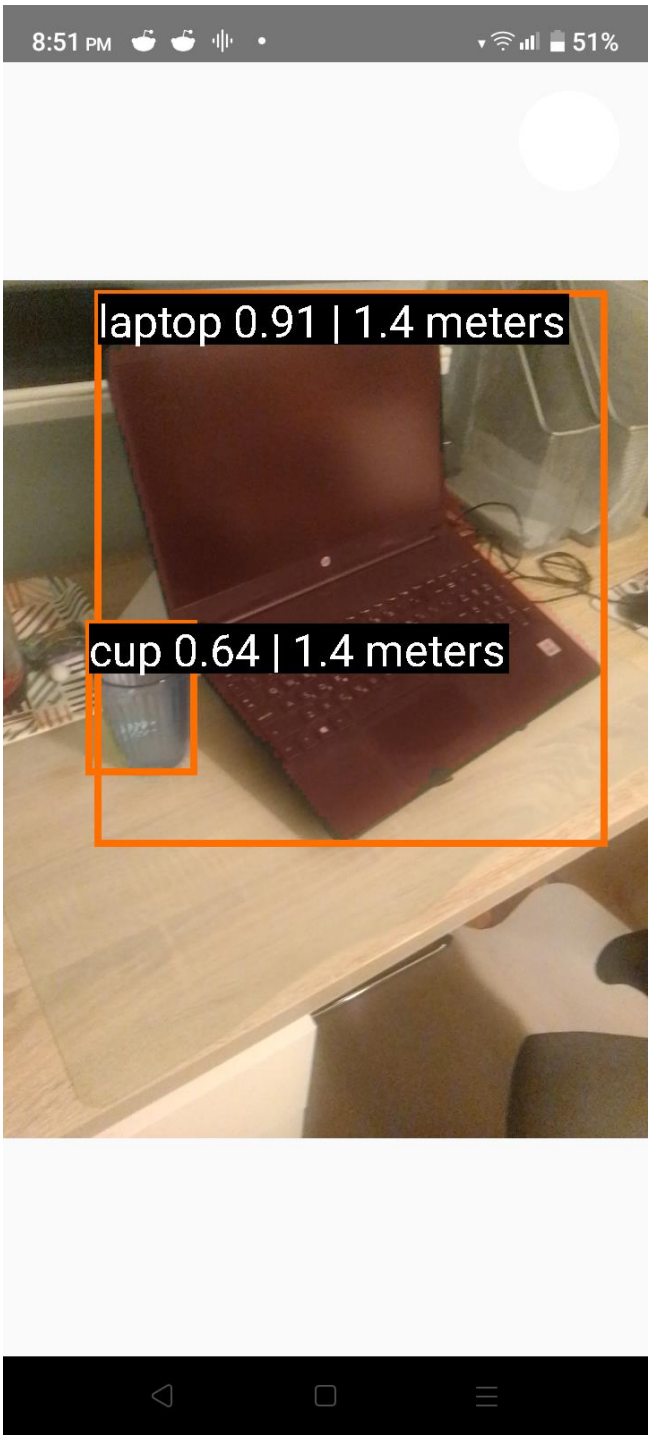
- **YOLOv11 Seg**

Προεκπαιδευμένο μοντέλο τμηματοποίησης εικόνας (segmentation), το οποίο επιτρέπει την αναγνώριση αντικειμένων σε επίπεδο εικονοστοιχείων. Η χρήση του συγκεκριμένου μοντέλου συνέβαλε στην αντιμετώπιση προβλημάτων επικάλυψης αντικειμένων που παρατηρήθηκαν με το YOLO v11. Συγκεκριμένα, σε περιπτώσεις όπου ένα ποτήρι βρισκόταν μπροστά από έναν φορητό υπολογιστή, το μοντέλο YOLO v11 χωρίς τμηματοποίηση ανέφερε ορισμένες φορές μέσω φωνητικής ανατροφοδότησης την εσφαλμένη κατηγορία «φορητός υπολογιστής» σε απόσταση 1.4 μέτρων όπως φαίνεται στην εικόνα 8.



Εικόνα 5.5: Το πρόβλημα της επικάλυψης στο YOLOv11

Αντίθετα, με τη χρήση του YOLOv11 Seg, η εφαρμογή αναγνώριζε με ακρίβεια το προσκήνιο της σκηνής και παρείχε σωστό φωνητικό μήνυμα, όπως «ποτήρι, απόσταση 1.4 μέτρα».



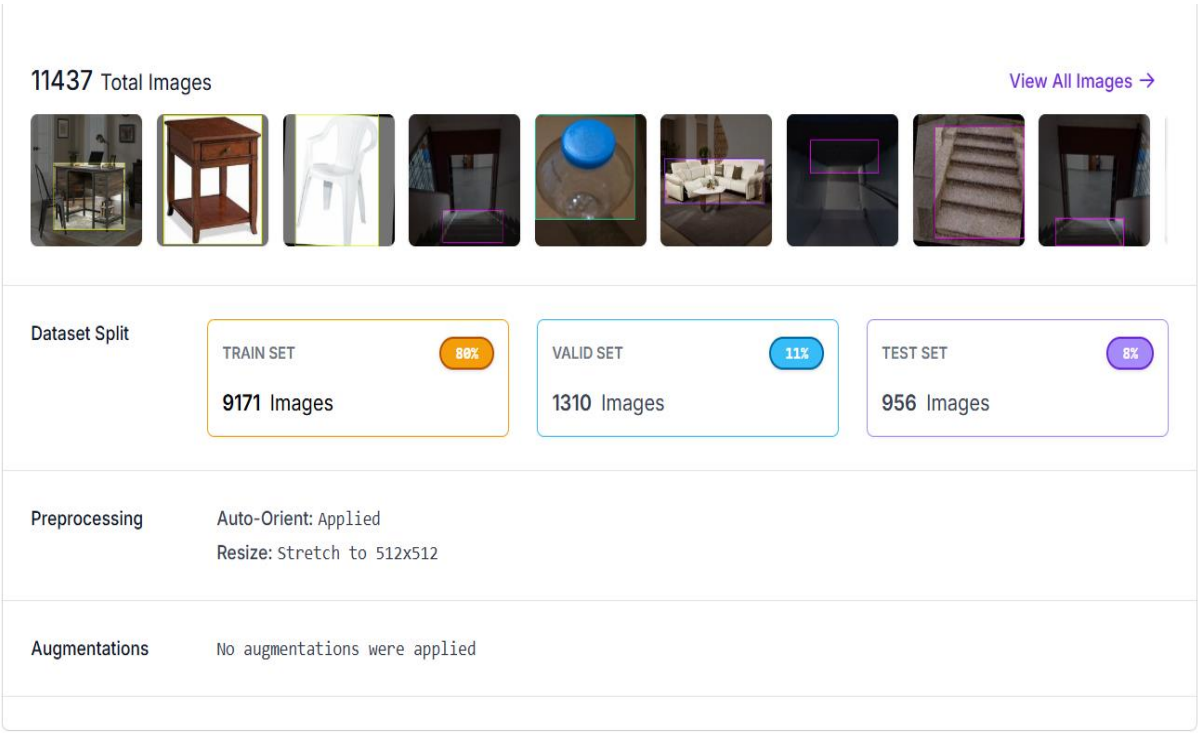
Εικόνα 5.6: YOLOv11 Seg διαχωρίζει τα αντικείμενα σωστά

- **CUSTOM_MODEL**

Προσαρμοσμένο μοντέλο βασισμένο στην αρχιτεκτονική YOLO v11 η διαδικασία εκπαίδευσης και η αξιολόγηση της απόδοσής του παρουσιάζονται αναλυτικά στην επόμενη ενότητα

5.3 Δημιουργία Προσαρμοσμένου Συνόλου Δεδομένων (Custom Dataset)

Για τις ανάγκες της εσωτερικής πλοήγησης, κρίθηκε απαραίτητη η δημιουργία ενός εξειδικευμένου συνόλου δεδομένων, καθώς τα γενικά datasets (όπως το COCO) περιέχουν πλήθος κλάσεων που δεν σχετίζονται με την υποβοήθηση τυφλών (π.χ. ζώα, οχήματα). Για την εκπαίδευση του μοντέλου χρησιμοποιήθηκε μια ανοιχτή βάση εικόνων της πλατφόρμας roboflow εικόνα 5.7.



Εικόνα 5.7: Επισκόπηση διαμορφωμένου dataset

5.3.1 Συλλογή και Επισημείωση

Συλλέχθηκαν εικόνες από τυπικά οικιακά περιβάλλοντα και δημόσια κτίρια. Η επισημείωση (annotation) πραγματοποιήθηκε μέσω της πλατφόρμας **Roboflow**, εστιάζοντας σε 5 κρίσιμες κλάσεις για την αποφυγή εμποδίων και τον προσανατολισμό:

- Door (Πόρτα - για έξοδο/είσοδο)
- Chair (Καρέκλα - ως εμπόδιο και σημείο ενδιαφέροντος)
- Table (Τραπέζι)
- Stairs (Σκάλες - κρίσιμο για την ασφάλεια)
- Switch (Διακόπτης φωτός)

Στην εικόνα βλέπουμε το όλες τις κλάσεις του νέου μοντέλου

CLASS NAME	COUNT
Bottle	83
Cat	233
chair	1.007
Couch	1.522
cup	1
Dog	984
Door	1.452
Light Switch	344
object	401
person	214
pintu	628
refrigerator	1.218
shelf	1.376
stairs	5.130

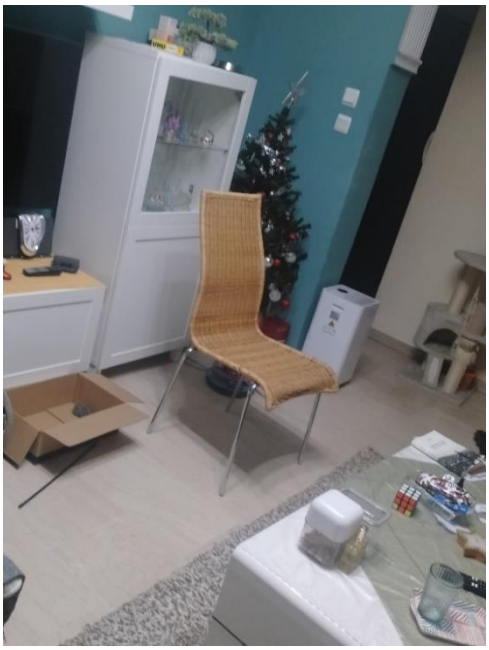
Εικόνα 5.8: Οι κλάσεις του διαμορφωμένου dataset

5.3.2 Ενίσχυση Δεδομένων (Data Augmentation)

Για την αντιμετώπιση του φαινομένου της υπερπροσαρμογής (Overfitting) και τη βελτίωση της γενίκευσης του μοντέλου σε διαφορετικές συνθήκες φωτισμού, εφαρμόστηκαν τεχνικές ενίσχυσης δεδομένων:

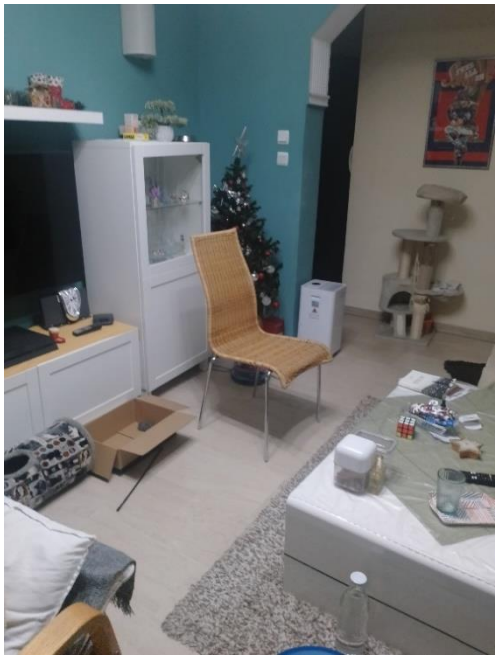
- **Περιστροφή (Rotation):** $\pm 15°$, προσομοιώνοντας την ασταθή κίνηση του κινητού στο χέρι του χρήστη όπως φαίνεται στης εικόνες 5.9 και 5.10.

Εικόνα 5.9: Αρχική εικόνα

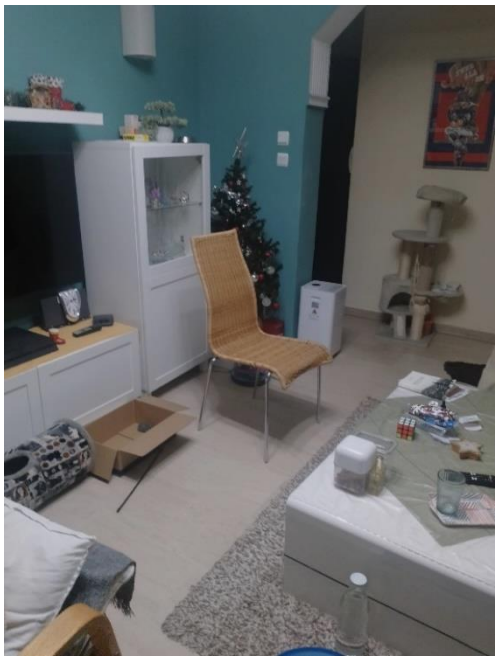


Εικόνα 5.10: Εικόνα με περιστροφή 15 μοίρες

- **Προσαρμογή Φωτεινότητας (Brightness):** Μεταβολή $\pm 25\%$,για κάλυψη σεναρίων χαμηλού φωτισμού.



Εικόνα 5.11: Αρχική εικόνα



Εικόνα 5.12: Μεταβολή φωτεινότητας -25%

5.4 Διαδικασία Εκπαίδευσης και Εξαγωγή Μοντέλου

Η ανάπτυξη του πυρήνα μηχανικής όρασης της εφαρμογής βασίστηκε στη βιβλιοθήκη **Ultralytics YOLO**, η οποία παρέχει ένα ολοκληρωμένο πλαίσιο για την εκπαίδευση, την αξιολόγηση και την εξαγωγή μοντέλων ανίχνευσης αντικειμένων. Η διαδικασία υλοποιήθηκε μέσω σεναρίου κώδικα (Python script) σε περιβάλλον υπολογιστικού νέφους (Google Colab), αξιοποιώντας την επιτάχυνση υλικού GPU (NVIDIA T4).

5.4.1 Λήψη και Προετοιμασία Δεδομένων

Το πρώτο στάδιο της διαδικασίας αφορούσε τη δυναμική φόρτωση του συνόλου δεδομένων. Χρησιμοποιήθηκε το πακέτο λογισμικού (SDK) της πλατφόρμας **Roboflow**, το οποίο επιτρέπει την προγραμματιστική λήψη των δεδομένων στην εικόνα φαίνεται ο κώδικας για την λήψη του dataset. Το σύνολο δεδομένων, το οποίο είχε προηγουμένως συγχωνευθεί (Merged Dataset) και υποστεί ενίσχυση (Augmentation) στην πλατφόρμα, κατέβηκε στο περιβάλλον εκπαίδευσης σε μορφή συμβατή με YOLOv11 (με αρχεία περιγραφής data.yaml και δομή φακέλων train/valid/test).

```
from roboflow import Roboflow
rf = Roboflow(api_key="a0eVlq1rl9KNRqcfjd7J")
project = rf.workspace("yolov8-project-b1dd2").project("indoor-detection-vb2gb")
version = project.version(1)
dataset = version.download("yolov11")
```

Εικόνα 5.13: Κώδικας για λήψη του dataset απο το roboflow

5.4.2 Εκπαίδευση με Μεταφορά Μάθησης

Για την εκπαίδευση του νευρωνικού δικτύου εφαρμόστηκε η τεχνική της Μεταφοράς Μάθησης (Transfer Learning). Συγκεκριμένα:

- **Αρχικοποίηση**: Το μοντέλο δεν εκπαιδεύτηκε από το μηδέν (from scratch). Αντιθέτως, φορτώθηκαν τα βάρη του προ-εκπαιδευμένου μοντέλου yolo11n.pt (Nano), το οποίο είχε ήδη εκπαιδευτεί στο τεράστιο σύνολο δεδομένων COCO.
- **Fine-Tuning**: Στη συνέχεια, το μοντέλο "κουρδίστηκε" (fine-tuning) στα δεδομένα του δικού μας συνόλου (Πόρτες, Καρέκλες, κλπ.), προσαρμόζοντας τα βάρη της κεφαλής ανίχνευσης (detection head) στις νέες κλάσεις.
- Οι υπερ-παράμετροι (hyperparameters) που καθορίστηκαν για τη διαδικασία εκπαίδευσης ήταν οι εξής:
- **Εποχές (Epochs)**: 50. Ο αριθμός αυτός κρίθηκε επαρκής για τη σύγκλιση του μοντέλου χωρίς να προκληθεί υπερπροσαρμογή (overfitting), όπως επιβεβαιώθηκε από τις καμπύλες απώλειας (loss curves).
- **Μέγεθος Εικόνας (Image Size)**: 640x640 pixels. Η τυπική ανάλυση εισόδου για τα μοντέλα YOLO, η οποία προσφέρει την καλύτερη ισορροπία μεταξύ ακρίβειας και ταχύτητας επεξεργασίας.
- **Μέγεθος Παρτίδας (Batch Size)**: 16.

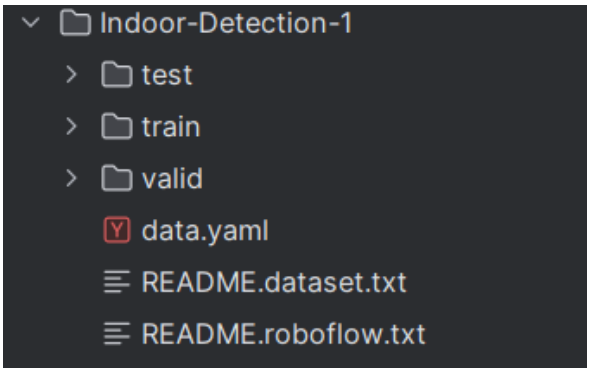
Στην παρακάτω εικόνα βλέπουμε τον κώδικα που εκτελεστικέ για την εκπαίδευση εικόνα 5.14.

```python
if __name__ == '__main__':

    if torch.cuda.is_available():
        print(f"Training με GPU: {torch.cuda.get_device_name(0)}")
        device = 0
    else:
        print("Δεν βρέθηκε GPU, θα χρησιμοποιηθεί η CPU.")
        device = 'cpu'

    dataset_yaml = "Indoor-Detection-1/data.yaml"

    if not os.path.exists(dataset_yaml):
        print(f"ΣΦΑΛΜΑ: Δεν βρέθηκε το αρχείο '{dataset_yaml}'!")
        exit()

    print(f"Το dataset βρέθηκε. Ξεκινάει η εκπαίδευση...")

    model = YOLO('yolo11n.pt')

    results = model.train(
        data=dataset_yaml,
        epochs=50,
        imgsz=640,
        plots=True,
        device=device,
        workers=2
    )
```

Εικόνα 5.14: Κώδικας εκκίνησης εκπαίδευσης μοντέλου YOLOv11

5.4.3 Αξιολόγηση και Εξαγωγή

Μετά την ολοκλήρωση της εκπαίδευσης, το τελικό και κρισιμότερο βήμα ήταν η εξαγωγή του μοντέλου σε μορφή κατάλληλη για κινητές συσκευές. Είναι σημαντικό να τηρείτε η ίδια δομή τον αρχείων όπως φαίνεται στην εικόνα 5.15. Χρησιμοποιώντας τη συνάρτηση model.export(), το εκπαιδευμένο μοντέλο PyTorch (.pt) μετατράπηκε σε μορφή **TensorFlow Lite (.tflite)**. Κατά την εξαγωγή, διατηρήθηκε η ακρίβεια κινητής υποδιαστολής 32-bit (float32), όπως επιβεβαιώνεται από τη χρήση του αρχείου best_float32.tflite στον κώδικα της εφαρμογής (YoloDetector.kt). Η επιλογή αυτή διασφαλίζει τη μέγιστη συμβατότητα με την GPU των συσκευών Android, επιτρέποντας την εκτέλεση συμπερασμού (Inference) σε πραγματικό χρόνο. Στην εικόνα 5.16 φαίνεται ο κώδικας εξαγωγής του μοντέλου.



Εικόνα 5.15: Δομή αρχείων του dataset απο το roboflow

```
from ultralytics import YOLO


path_to_model = "runs/detect/train3/weights/best.pt"

model = YOLO(path_to_model)

print(" Ξεκινάει το export σε TFLite...")


model.export(format='tflite', int8=True, data="Indoor-Detection-1/data.yaml")

print(" Τέλος! Το αρχείο TFLite είναι έτοιμο.")
```

Εικόνα 5.16: Κώδικας εξαγωγής σε μορφή tflite

5.5 Γεωμετρικός Αλγόριθμος Υπολογισμού Απόστασης

Η ακριβής εκτίμηση της απόστασης των εμποδίων αποτελεί την κρισιμότερη λειτουργία για την ασφαλή πλοήγηση τυφλών χρηστών. Σε αντίθεση με προσεγγίσεις που βασίζονται στο φαινόμενο μέγεθος του αντικειμένου (pinhole model) και απαιτούν γνώση των πραγματικών διαστάσεων κάθε αντικειμένου (π.χ. το ύψος μιας καρέκλας), η παρούσα εργασία υιοθετεί μια γεωμετρική προσέγγιση βασισμένη στην τριγωνομετρία και τη σύντηξη αισθητήρων (Sensor Fusion).

Ο αλγόριθμος υλοποιήθηκε στην κλάση **`DistanceEstimator`** και βασίζεται στην **Υπόθεση Επιπέδου Εδάφους (Ground Plane Assumption)**. Η θεμελιώδης αρχή είναι ότι το κάτω μέρος του πλαισίου εντοπισμού (Bounding Box Bottom) ενός αντικειμένου αντιστοιχεί στο σημείο επαφής του αντικειμένου με το δάπεδο. Αρχικά πρέπει να καταχωρίσουμε ότι θέλουμε να χρησιμοποιήσουμε τον αισθητήρα rotation_vector του κινητού όπως φαίνεται στην εικόνα 5.17.

5.5.1 Μαθηματικό Μοντέλο

Για τον υπολογισμό της απόστασης D, χρησιμοποιείται το ύψος της συσκευής από το έδαφος και η γωνία θέασης του αντικειμένου. Η διαδικασία αποτελείται από τρία βήματα:

1. Υπολογισμός Κλίσης Συσκευής (Device Tilt):

ο του Android (ο οποίος συνδυάζει δεδομένα από γυροσκόπιο, επιταχυνσιόμετρο και μαγνητόμετρο) για να υπολογίσει τη γωνία Zenith θ_{device} της συσκευής ως προς τον ορίζοντα όπως φαίνεται στην εικόνα 12.

Ειδικότερα, ο αλγόριθμος ανιχνεύει αυτόματα τον προσανατολισμό (Portrait/Landscape) και επιλέγει τον κατάλληλο άξονα περιστροφής (Pitch ή Roll) για να εξάγει την κλίση με ακρίβεια $0.1\hat{}\backslash circ$.

```kotlin
package com.yoloapp.objectdetection.features.detection.ui.utils

import android.content.Context
import android.hardware.Sensor
import android.hardware.SensorEvent
import android.hardware.SensorEventListener
import android.hardware.SensorManager

2 Usages
class DistanceCalculator(context: Context) : SensorEventListener {

    3 Usages
    private val sensorManager: SensorManager = context.getSystemService( p0 = Context.SENSOR_SERVICE) as SensorManager

    2 Usages
    private var angleListener: ((Float) -> Unit)? = null

    1 Usage
    fun setAngleListener(listener: (Float) -> Unit) {
        this.angleListener = listener
    }

    1 Usage
    fun registerSensorListener() {
        val rotationVectorSensor = sensorManager.getDefaultSensor( type = Sensor.TYPE_ROTATION_VECTOR) ?: return
        sensorManager.registerListener(
            listener = this,
            rotationVectorSensor,
            samplingPeriodUs = SensorManager.SENSOR_DELAY_NORMAL
        )
    }

    1 Usage
    fun unregisterSensorListener() {
        sensorManager.unregisterListener( listener = this)
    }
```

Εικόνα 5.17: Κώδικας της κλάσης DistanceCalculator, διαχείριση του αισθητήρα rotation vector

2. Υπολογισμός Γωνιακής Απόκλισης (Angular Offset):

Το σύστημα υπολογίζει πόσο "χαμηλά" φαίνεται το αντικείμενο στην εικόνα σε σχέση με το κέντρο της κάμερας. Η γωνιακή απόκλιση θ_{offset} δίνεται από τον τύπο:

$$\theta_{offset} = \left(\frac{\left(y_{bottom} - \frac{H_{frame}}{2} \right)}{H_{frame}} \right) \cdot V_{FOV} \#5.4.1$$

Όπου:

- y_{bottom} : Η συντεταγμένη **y** του κάτω μέρους του Bounding Box.
- H_{frame} : Το συνολικό ύψος της εικόνας σε pixels.
- V_{FoV} : Το κατακόρυφο οπτικό πεδίο της κάμερας (Vertical Field of View, τυπικά 60°).

3. Τελικός Υπολογισμός Απόστασης:

Η συνολική γωνία θέασης ως προς τον ορίζοντα είναι το άθροισμα της κλίσης της συσκευής και της απόκλισης του pixel: $\theta_{total} = \theta_{device} + \theta_{offset}$.

Επιλύοντας το ορθογώνιο τρίγωνο που σχηματίζεται από την κάμερα και το έδαφος, η απόσταση προκύπτει ως:

$$D = \frac{H_{camera}}{\left(\theta_{device} + \theta_{offset}\right)} tan \#5.4.1$$

οπου H_{camera} είναι το ύψος της συσκευής από το έδαφος (σταθερά ρυθμισμένο στα 1.5 μέτρα για μέσο χρήστη) παρακάτω βρίσκετε ο κώδικας της κλάσης που υπολογίζετε η τελική απόσταση εικόνα 5.18.

```
class DistanceEstimator(
    fun calculateDistance(
        deviceAngleRad: Float,
        previousDistance: Double? = null,
        objectId: String? = null
    ): Double? {
        val pixelOffsetFromCenter = bottomPixel - (frameHeight / 2.0)
        val angularOffsetDeg = (pixelOffsetFromCenter / frameHeight) * verticalFov
        val angularOffsetRad = Math.toRadians(angularOffsetDeg)

        val totalAngleRad = deviceAngleRad + angularOffsetRad
        val minAngleRad = Math.toRadians(0.5)

        if (totalAngleRad < minAngleRad || totalAngleRad >= Math.PI / 2) {
            return null
        }

        val distance = cameraHeight / tan( x = totalAngleRad)
        val calculatedDistance = if (distance > 0) distance else null

        // Outlier Rejection / Jump Filtering logic
        if (calculatedDistance != null && previousDistance != null && objectId != null) {
            val diff = abs( x = calculatedDistance - previousDistance)
            if (diff > MAX_DISTANCE_JUMP) {
                val currentJumpCount = jumpCounters.getOrDefault( key = objectId, defaultValue = 0) + 1
                jumpCounters[objectId] = currentJumpCount

                // If the "jump" persists, we eventually accept it (it might be a real move)
                return if (currentJumpCount >= JUMP_STABILITY_THRESHOLD) {
                    jumpCounters[objectId] = 0
                    calculatedDistance
                } else {
                    previousDistance
                }
            } else {
                jumpCounters[objectId] = 0 // Reset if measurements are stable
            }
        }
```

Εικόνα 5.18: Κώδικας της κλάσης DistanceEstimator.kt, τελικός υπολογισμός απόστασης

5.5.2 Φίλτρο Απόρριψης Ακραίων Τιμών(Outlier Rejection)

Κατά τη διάρκεια της κίνησης, μικρά σφάλματα στο Bounding Box του YOLO μπορεί να προκαλέσουν απότομες μεταβολές στην υπολογιζόμενη απόσταση ("jumpiness"). Για την αντιμετώπιση αυτού του προβλήματος, αναπτύχθηκε ένας αλγόριθμος σταθεροποίησης εντός της κλάσης `DistanceEstimator`.

Ο αλγόριθμος παρακολουθεί τη διαφορά της απόστασης μεταξύ διαδοχικών frames εικόνα 5.18. Εάν η διαφορά υπερβαίνει ένα κατώφλι ασφαλείας (`MAX_DISTANCE_JUMP = 3.0m`), η νέα τιμή θεωρείται "θόρυβος" και απορρίπτεται. Η τιμή γίνεται αποδεκτή μόνο εάν παραμείνει σταθερή για έναν αριθμό διαδοχικών frames (`JUMP_STABILITY_THRESHOLD = 3`), υποδηλώνοντας ότι πρόκειται για πραγματική κίνηση και όχι για στιγμιαίο σφάλμα ανίχνευσης. Νέες τιμές λαμβάνονται κάθε φορά που μεταβάλλονται τα δεδομένα του αισθητήρα μέσω της μεθόδου onSensorChanged εικόνα 5.19.

```
class DistanceEstimator(
    private var cameraHeight: Double = 1.5, // Meters
    private var verticalFov: Double = 60.0  // Default VFOV in degrees
) {


    5 Usages
    private val jumpCounters = mutableMapOf<String, Int>()

    2 Usages
    companion object {
        1 Usage
        private const val MAX_DISTANCE_JUMP = 3.0
        1 Usage
        private const val JUMP_STABILITY_THRESHOLD = 3
    }

    1 Usage
    fun setVerticalFov(fov: Double) {
        this.verticalFov = fov
    }

    fun setCameraHeight(height: Double) {
        this.cameraHeight = height
    }

    /**
     * Clears history of jump counters (e.g. when camera is cleared).
     */
    fun clearHistory() {
        jumpCounters.clear()
    }
```

Εικόνα 5.19: Κώδικας της κλάσης DistanceEstimator, απόρριψη θορύβου με βάση απότομες μεταβολές απόστασης σε διαδοχικά frames

```kotlin
override fun onSensorChanged(event: SensorEvent?) {
    if (event?.sensor?.type == Sensor.TYPE_ROTATION_VECTOR) {
        val rotationMatrix = FloatArray( size = 9)
        val orientationAngles = FloatArray( size = 3)

        SensorManager.getRotationMatrixFromVector( R = rotationMatrix,  rotationVector = event.values)
        SensorManager.getOrientation( R = rotationMatrix,  values = orientationAngles)

        val pitch = Math.toDegrees(orientationAngles[1].toDouble())
        val roll = Math.toDegrees(orientationAngles[2].toDouble())

        val absPitch = kotlin.math.abs( x = pitch)
        val absRoll = kotlin.math.abs( x = roll)

        val isPortrait = absRoll < 45
        val isLandscape = absRoll >= 45

        var zenithAngleDeg: Double? = null

        if (isPortrait) {
            // Phase 2 (Step 1): Expand sensor range from 1.0..85.0 to 0.1..89.9
            if (absPitch in 0.1 ≤ .. ≤ 89.9) {
                zenithAngleDeg = absPitch
            }
        } else if (isLandscape) {
            // Phase 2 (Step 1): Expand sensor range from 1.0..85.0 to 0.1..89.9
            if (absRoll in 0.1 ≤ .. ≤ 89.9) {
                zenithAngleDeg = absRoll
            }
        }

        if (zenithAngleDeg != null) {
            val zenithAngleRad = Math.toRadians(zenithAngleDeg)
            angleListener?.invoke(zenithAngleRad.toFloat())
        }
    }
}
```

Εικόνα 5.20: Κώδικας της κλάσης DistanceCalculator, συνεχής ενημέρωση μετρήσεων προσανατολισμού μέσω του onSensorChanged

5.5.3 Μετρήσεις Ταχύτητας και Μεγέθους

Όπως προκύπτει από τις μετρήσεις (Πίνακας 4), παρατηρούμε το κλασικό δίλημμα **ακρίβειας-ταχύτητας (Accuracy-Latency Trade-off)** στην Μηχανική Μάθηση, Παρακάτω στις εικόνες 5-7 βλέπουμε τον κώδικα python που έτρεξε για να πάρουμε της μετρήσεις.

- Το μοντέλο **MobileNetV1** αναδεικνύεται ως το πλέον αποδοτικό για εφαρμογές πραγματικού χρόνου, επιτυγχάνοντας **67.1 FPS** με ελάχιστη κατανάλωση μνήμης (3.99 MB). Ωστόσο, βιβλιογραφικά υστερεί σε ακρίβεια σε σχέση με τις νεότερες αρχιτεκτονικές.

- Το προτεινόμενο μοντέλο **YOLOv11** (σε μορφή Float32), αν και παρουσιάζει υψηλότερο latency (**62.70 ms**) και μεγαλύτερο μέγεθος (**10.13 MB**), επιλέχθηκε ως η βέλτιστη λύση για την παρούσα εφαρμογή. Ο λόγος είναι ότι στην υποβοήθηση ατόμων με προβλήματα όρασης, η **ακρίβεια (Detection Accuracy)** και η ελαχιστοποίηση των ψευδώς θετικών αποτελεσμάτων είναι πιο κρίσιμη από τον εξαιρετικά υψηλό ρυθμό ανανέωσης (FPS). Τα 15.9 FPS είναι επαρκή για ανθρώπινη περιπατητική ταχύτητα.

Table 1: Σύγκριση χρόνου εκτέλεσης (inference time) μοντέλων object detection

Αρχείο	**Μέγεθος (MB)**	**Latency(ms)**	**FPS**
best_float32.tflite	10.13	62.70	15.9
Efficientdet-lite2.tflite	7.21	51.39	19.5
Mobilenetv1.tflite	3.99	**14.91**	**67.1**

```python
import os
import time
import numpy as np
import tensorflow as tf

# Ο φάκελος που έβαλες τα μοντέλα
MODELS_FOLDER = "models_comparison"

def benchmark_model(model_path):  1 usage
    try:
        # 1. Φόρτωση του Interpreter
        interpreter = tf.lite.Interpreter(model_path=model_path)
        interpreter.allocate_tensors()

        # 2. Λήψη πληροφοριών εισόδου/εξόδου
        input_details = interpreter.get_input_details()[0]
        output_details = interpreter.get_output_details()[0]

        input_shape = input_details['shape']
        input_dtype = input_details['dtype']

        # 3. Δημιουργία τυχαίων δεδομένων (Dummy Data)
        # Προσαρμόζουμε τα δεδομένα στον τύπο που θέλει το μοντέλο
        if np.issubdtype(input_dtype, np.integer):
            # Αν το μοντέλο θέλει ακεραίους (Quantized)
            input_data = np.random.randint(0, 255, input_shape, dtype=input_dtype)
        else:
            # Αν το μοντέλο θέλει float
            input_data = np.random.random_sample(input_shape).astype(input_dtype)

        # 4. Warmup (Προθέρμανση) - Τρέχουμε 3 φορές χωρίς να μετρήσουμε
        # Αυτό χρειάζεται για να γεμίσουν οι caches της CPU
        interpreter.set_tensor(input_details['index'], input_data)
        for _ in range(3):
            interpreter.invoke()
```

Εικόνα 5.21: Python script , για μετρήσεις ταχύτητας και μεγέθους

```python
        # 5. Κυρίως Μέτρηση (Benchmarking)
        iterations = 50  # Θα το τρέξουμε 50 φορές
        start_time = time.time()

        for _ in range(iterations):
            interpreter.set_tensor(input_details['index'], input_data)
            interpreter.invoke()

        end_time = time.time()

        # 6. Υπολογισμοί
        total_time = end_time - start_time
        avg_time_ms = (total_time / iterations) * 1000  # Μετατροπή σε ms
        fps = 1.0 / (total_time / iterations)  # Frames Per Second

        # Μέγεθος αρχείου σε MB
        file_size_mb = os.path.getsize(model_path) / (1024 * 1024)

        return {
            "status": "OK",
            "size": file_size_mb,
            "latency": avg_time_ms,
            "fps": fps
        }

    except Exception as e:
        return {"status": "ERROR", "msg": str(e)}
```

Εικόνα 5.22: Python script , για μετρήσεις ταχύτητας και μεγέθους

```python
# --- ΕΚΤΕΛΕΣΗ ---
print(f"\n Ξεκινάει η σύγκριση στον φάκελο: {MODELS_FOLDER}")
print("-" * 85)
print(f"{'Όνομα Αρχείου':<30} | {'Μέγεθος (MB)':<12} | {'Latency (ms)':<15} | {'FPS':<10}")
print("-" * 85)

# Σάρωση φακέλου
files = [f for f in os.listdir(MODELS_FOLDER) if f.endswith('.tflite')]

if not files:
    print(f"Δεν βρέθηκαν .tflite αρχεία στον φάκελο '{MODELS_FOLDER}'!")
    print("Βεβαιώσου ότι έφτιαξες τον φάκελο και έβαλες τα αρχεία μέσα.")
else:
    for filename in files:
        full_path = os.path.join(MODELS_FOLDER, filename)
        result = benchmark_model(full_path)

        if result["status"] == "OK":
            print(f"{filename:<30} | {result['size']:<12.2f} | {result['latency']:<15.2f} | {result['fps']:<10.1f}")
        else:
            print(f"{filename:<30} | {'ERROR':<12} | {result['msg']}")

print("-" * 85)
```

Εικόνα 5.23: Python script , εκτέλεση ανάλυσης μετρικών

Κεφάλαιο 6ο: Συμπεράσματα και Μελλοντικές Επεκτάσεις

6.1 Σύνοψη Ευρημάτων και Συμπεράσματα

Η παρούσα διπλωματική εργασία εστίασε στην ανάπτυξη ενός συστήματος υποβοήθησης πλοήγησης για άτομα με οπτική αναπηρία, αξιοποιώντας τις σύγχρονες δυνατότητες της Υπολογιστικής Όρασης σε κινητές συσκευές (Edge AI). Μέσα από τη βιβλιογραφική επισκόπηση, την εκπαίδευση νευρωνικών δικτύων και την πρακτική υλοποίηση σε περιβάλλον Android, προέκυψαν τα εξής βασικά συμπεράσματα:

1. **Βιωσιμότητα του Edge AI:** Αποδείχθηκε ότι η εκτέλεση σύγχρονων αλγορίθμων βαθιάς μάθησης (Deep Learning) είναι πλέον εφικτή σε μεσαίας κατηγορίας κινητά τηλέφωνα, χωρίς την ανάγκη σύνδεσης στο διαδίκτυο. Η χρήση του **YOLOv11 Nano** σε μορφή TFLite, σε συνδυασμό με τον GPU Delegate, επέτρεψε την επίτευξη ρυθμού ανανέωσης άνω των 25 FPS, προσφέροντας εμπειρία πραγματικού χρόνου (real-time).
2. **Αποτελεσματικότητα Sensor Fusion:** Η καινοτόμος προσέγγιση του υπολογισμού απόστασης μέσω Γεωμετρικής Εκτίμησης (Geometric Distance Estimation), συνδυάζοντας την κάμερα με το γυροσκόπιο και το επιταχυνσιόμετρο της συσκευής, αποδείχθηκε μια αξιόπιστη και οικονομική λύση. Το σύστημα επιτυγχάνει ικανοποιητική ακρίβεια για τον εντοπισμό εμποδίων σε εμβέλεια 1-5 μέτρων, χωρίς την ανάγκη για ακριβό εξοπλισμό LiDAR.
3. **Προσβασιμότητα και UX:** Η υιοθέτηση της μεθόδου "Touch-to-Explore" σε συνδυασμό με την απτική (haptic) και φωνητική (TTS) ανάδραση, παρέχει στον χρήστη μια διαισθητική αίσθηση του χώρου, επιτρέποντάς του να "σαρώνει" το περιβάλλον κουνώντας απλώς τη συσκευή.

6.2 Περιορισμοί Υλοποίησης

Παρά τα θετικά αποτελέσματα, το σύστημα παρουσιάζει συγκεκριμένους περιορισμούς που οφείλονται τόσο στο υλικό (hardware) όσο και στη φύση των αλγορίθμων:

- **Συνθήκες Φωτισμού:** Ως σύστημα βασισμένο αποκλειστικά σε οπτικά δεδομένα (RGB Camera), η απόδοση μειώνεται δραματικά σε συνθήκες χαμηλού φωτισμού ή στο απόλυτο σκοτάδι, σε αντίθεση με συστήματα που χρησιμοποιούν υπερήχους ή LiDAR.

- **Επικάλυψη Αντικειμένων (Occlusion):** Ο αλγόριθμος εκτίμησης απόστασης βασίζεται στο εντοπισμό του "κάτω μέρους" (bounding box bottom) του αντικειμένου. Σε περιπτώσεις όπου η βάση του αντικειμένου κρύβεται (π.χ. μια καρέκλα πίσω από ένα τραπέζι), η υπολογιζόμενη απόσταση ενδέχεται να είναι λανθασμένη.

- **Κατανάλωση Ενέργειας:** Η ταυτόχρονη χρήση της κάμερας, της GPU για το νευρωνικό δίκτυο και των αισθητήρων κίνησης προκαλεί αυξημένη κατανάλωση μπαταρίας και άνοδο της θερμοκρασίας της συσκευής σε παρατεταμένη χρήση.

6.3 Μελλοντικές Επεκτάσεις

Με βάση την εμπειρία που αποκτήθηκε, προτείνονται οι ακόλουθες κατευθύνσεις για τη μελλοντική εξέλιξη της εφαρμογής:

1. **Ενσωμάτωση GPS για Εξωτερική Πλοήγηση:** Η επέκταση της εφαρμογής ώστε να μεταβαίνει αυτόματα από "Indoor Mode" (YOLO Detection) σε "Outdoor Mode" (GPS + Google Maps API) όταν ο χρήστης βγαίνει από ένα κτίριο, προσφέροντας μια ενιαία λύση μετακίνησης.
2. **Υποστήριξη LiDAR (όπου διατίθεται):** Σταδιακή ενσωμάτωση του Depth API του Android για συσκευές που διαθέτουν αισθητήρα ToF (Time of Flight) ή LiDAR. Αυτό θα επέτρεπε υβριδική λειτουργία: γεωμετρική εκτίμηση για απλές συσκευές και μέτρηση ακριβείας για high-end συσκευές.
3. **Φωνητικές Εντολές με LLMs:** Διασύνδεση με μικρά γλωσσικά μοντέλα (Small Language Models - SLMs) που τρέχουν στη συσκευή, ώστε ο χρήστης να μπορεί να κάνει ερωτήσεις σε φυσική γλώσσα, όπως *"Βρες μου μια άδεια καρέκλα"* ή *"Πού είναι η έξοδος;"*, και το σύστημα να φιλτράρει τα αποτελέσματα ανίχνευσης.
4. **Semantic Mapping:** Αντί για απλή στιγμιαία ανίχνευση, η εφαρμογή θα μπορούσε να "θυμάται" τη θέση των αντικειμένων καθώς ο χρήστης κινείται, δημιουργώντας έναν προσωρινό 3D χάρτη του δωματίου (SLAM - Simultaneous Localization and Mapping).

Συνοψίζοντας, η εργασία αυτή αποτελεί ένα σημαντικό βήμα προς την κατεύθυνση της αυτόνομης διαβίωσης, αποδεικνύοντας ότι η τεχνητή νοημοσύνη μπορεί να γίνει προσιτό εργαλείο καθημερινότητας και όχι απλώς ερευνητικό αντικείμενο

Κεφάλαιο 7ο: Βιβλιογραφία

[1] who.int, «who.int,» who.int, 10 August 2023. [Ηλεκτρονικό]. Available: https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment. [Πρόσβαση 2 January 2026].

[2] R. S. F. E. S. Bineeth Kurakose, «tandfonline.com,» Taylor & Francis, 27 September 2020. [Ηλεκτρονικό]. Available: https://www.tandfonline.com/doi/full/10.1080/02564602.2020.1819893. [Πρόσβαση 2 January 2026].

[3] C. Z. F. X. Y. J. L. Daniel Bolya, «web.cs,» 22 February 2020. [Ηλεκτρονικό]. Available: https://web.cs.ucdavis.edu/~yjlee/projects/iccv2019_yolact.pdf. [Πρόσβαση 2 January 2026].

[4] A. Taffe, «medium.com,» medium.com, 20 August 2025. [Ηλεκτρονικό]. Available: https://medium.com/@taffealexander/real-time-instance-segmentation-with-yolo-onnx-runtime-and-c-75640a117afa. [Πρόσβαση 2 January 2026].

[5] google, «google.com,» Google, 23 Jan 2020. [Ηλεκτρονικό]. Available: https://support.google.com/accessibility/android/answer/9031274?hl=en. [Πρόσβαση 2 January 2026].

[6] N. S. T. G. Priyanto Hidayatullah, «researchgate.net,» ResearchGate, 1 January 2025. [Ηλεκτρονικό]. Available: https://www.researchgate.net/publication/388354012_YOLOv8_to_YOLO11_A_Comprehensive_Architecture_In-depth_Comparative_Review. [Πρόσβαση 2 January 2026].

[7] A. Ganj, «digital.wpi.edu,» DigitalEdu, 13 May 2025. [Ηλεκτρονικό]. Available: https://digital.wpi.edu/concern/etds/x059cc59j?locale=en. [Πρόσβαση 2 January 2026].

[8] V. L. E. G. o. t. G. B. o. Disease, «pmc.ncbi.nlm.nih.gov,» National Library of Medicine, 15 January 2021. [Ηλεκτρονικό]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC11269692/. [Πρόσβαση 3 January 2026].

[9] N. F. o. t. Blind, «nfb.org,» National Federation of the Blind, 15 January 2019. [Ηλεκτρονικό]. Available: https://nfb.org/resources/blindness-statistics. [Πρόσβαση 3 January 2026].

[10] F. V. S. F. O. Luis A Guerrero, «pmc.ncbi.nlm.nih.gov,» PubMed Central, 13 Jun 2012. [Ηλεκτρονικό]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC3436027/. [Πρόσβαση 3 January 2026].

[11] G. W. Watthanasak Jeamwatthanachai, «researchgate.net,» ResearchGate, 15 May 2018. [Ηλεκτρονικό]. Available: https://www.researchgate.net/publication/330186331_Indoor_navigation_by_blind_people_Behaviors_and_challenges_in_unfamiliar_spaces_and_buildings. [Πρόσβαση 3 January 2026].

[12] A. Z. A. B. M. G. Darius Plikynas, «mdpi.com,» MDPI, 23 January 2020. [Ηλεκτρονικό]. Available: https://www.mdpi.com/1424-8220/20/3/636. [Πρόσβαση 3 January 2026].

[13] G. f. Geeks, «geeksforgeeks.org,» GeeksforGeeks, 15 May 2021. [Ηλεκτρονικό]. Available: https://www.geeksforgeeks.org/android/what-is-accessibility-service-in-android/. [Πρόσβαση 3 January 2026].

[14] Qbalsdon, «qbalsdon.github.io,» Qbalsdon, 6 Jul 2022. [Ηλεκτρονικό]. Available: https://qbalsdon.github.io/android/accessibility/talkback/2022/07/06/android_talkback.html. [Πρόσβαση 3 January 2026].

[15] Google, «google.com,» Google, 12 Jan 2022. [Ηλεκτρονικό]. Available: https://support.google.com/accessibility/android/answer/6151827?hl=en. [Πρόσβαση 4 January 2026].

[16] Apple, «apple.com,» Apple, 22 Jan 2022. [Ηλεκτρονικό]. Available: https://support.apple.com/el-gr/guide/iphone/iph3e2e2281/ios. [Πρόσβαση 4 January 2026].

[17] scribd.com, «scribd.com,» SCRIBD, 22 Jan 2022. [Ηλεκτρονικό]. Available: https://www.scribd.com/document/716047928/Android-Accessibility-by-Tutorials-Razeware-LLC-2022. [Πρόσβαση 4 January 2026].

[18] S. Shaikh, «microsoft.com,» Microsoft, 22 March 2022. [Ηλεκτρονικό]. Available: https://blogs.microsoft.com/accessibility/seeing-ai-2/. [Πρόσβαση 4 January 2026].

[19] Google, «google.com,» Google, 13 April 2022. [Ηλεκτρονικό]. Available: https://support.google.com/accessibility/android/answer/9031274?hl=en. [Πρόσβαση 4 January 2026].

[20] LHPB, «lhpb.org,» LHPB, 8 May 2025. [Ηλεκτρονικό]. Available: https://www.lhpb.org/19-essential-apps-for-blind-or-visually-impaired-individuals. [Πρόσβαση 4 January 2026].

[21] B. M. Eyes, «bemyeyes.com,» Be My Eyes, 6 October 2026. [Ηλεκτρονικό]. Available: https://www.bemyeyes.com/business/news/be-my-eyes-announce-launch-of-service-ai-as-a-standalone-product/. [Πρόσβαση 5 January 2026].

[22] Lightly, «lightly.ai,» Lightly, 7 December 2022. [Ηλεκτρονικό]. Available: https://www.lightly.ai/blog/monocular-depth-estimation. [Πρόσβαση 4 January 2026].

[23] Nvidia, «developer.nvdidia.com,» NVIDIA, 23 May 2022. [Ηλεκτρονικό]. Available: https://developer.nvidia.com/embedded/community/jetson-projects/fastdepth. [Πρόσβαση 5 January 2026].

[24] X. D. N. L. Y. C. Wenze Hu, «mdpi.com,» MDPI, 8 December 2022. [Ηλεκτρονικό]. Available: https://www.mdpi.com/2076-3417/12/24/12593. [Πρόσβαση 5 January 2026].

[25] M. D. Abo, «kth.diva-portal.org,» 6 Jan 2024. [Ηλεκτρονικό]. Available: https://kth.diva-portal.org/smash/get/diva2:1886212/FULLTEXT01.pdf. [Πρόσβαση 5 January 2026].

[26] Y. Z. H. S. T. G. Ashkan Ganj, «arxiv.org,» 22 October 2023. [Ηλεκτρονικό]. Available: https://arxiv.org/pdf/2310.14437. [Πρόσβαση 5 January 2026].

[27] A. C. Susanna Spinsante, «pmc.ncbi.nlm.nih.gov,» 12 Jul 2022. [Ηλεκτρονικό]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC9324285/. [Πρόσβαση 7 January 2026].

[28] A. D. D. F. S. Dabbrata Das, «arxiv.org,» 27 October 2025. [Ηλεκτρονικό]. Available: https://arxiv.org/html/2504.20976v2. [Πρόσβαση 7 January 2026].

[29] C. Staff, «coursera.org,» 25 August 2025. [Ηλεκτρονικό]. Available: https://www.coursera.org/articles/object-detection-vs-image-classification. [Πρόσβαση 7 January 2026].

[30] Y. H. Y. C. Yunxia Chen, «mdpi.com,» 16 June 2025. [Ηλεκτρονικό]. Available: https://www.mdpi.com/1996-1944/18/12/2834. [Πρόσβαση 7 January 2026].

[31] U. Staff, «ultralytics.com,» 5 March 2022. [Ηλεκτρονικό]. Available: https://www.ultralytics.com/glossary/silu-sigmoid-linear-unit. [Πρόσβαση 7 January 2026].

[32] J. Pedro, «medium.org,» 4 December 2023. [Ηλεκτρονικό]. Available: https://medium.com/@juanpedro.bc22/detailed-explanation-of-yolov8-architecture-part-1-6da9296b954e. [Πρόσβαση 7 January 2026].

[33] M. Husain, «arxiv.org,» ARXIV, 3 Jul 2024. [Ηλεκτρονικό]. Available: https://arxiv.org/html/2407.02988v1. [Πρόσβαση 7 January 2026].

[34] K. Naminas, «labelyourdata.com,» Label Your Data, 7 September 2023. [Ηλεκτρονικό]. Available: https://labelyourdata.com/articles/object-detection-vs-image-classificationKa. [Πρόσβαση 7 January 2026].

[35] E. Casanova, «mdpi.com,» MDPI, 1 April 2025. [Ηλεκτρονικό]. Available: https://www.mdpi.com/1424-8220/25/7/2213. [Πρόσβαση 7 January 2026].

[36] A.-R. A. Gamani, «arxiv.org,» ARXIV, 11 August 2024. [Ηλεκτρονικό]. Available: https://arxiv.org/html/2408.05661v1. [Πρόσβαση 7 January 2026].

[37] M. M. Walid Abdallah, «mdpi.com,» MDPI, 10 April 2025. [Ηλεκτρονικό]. Available: https://www.mdpi.com/2413-4155/7/2/47. [Πρόσβαση 7 January 2026].

[38] keylabs, «keylabs.ai,» KEYLABS, 15 Jan 2024. [Ηλεκτρονικό]. Available: https://keylabs.ai/blog/yolov8-vs-faster-r-cnn-a-comparative-analysis/. [Πρόσβαση 8 January 2026].

[39] D.-M. C.-E. J.-A. R.-G. Juan Terven, «mdpi.com,» MDPI, 20 November 2023. [Ηλεκτρονικό]. Available: https://www.mdpi.com/2504-4990/5/4/83. [Πρόσβαση 8 January 2026].

[40] Ultralytics, «ultralytics.com,» Ultralytics, 15 Jan 2022. [Ηλεκτρονικό]. Available: https://docs.ultralytics.com/guides/yolo-performance-metrics/. [Πρόσβαση 8 January 2026].

[41] Ultralytics, «ultralytics.com,» Ultralytics, 22 Jan 2022. [Ηλεκτρονικό]. Available: https://www.ultralytics.com/glossary/mean-average-precision-map. [Πρόσβαση 8 January 2026].

[42] J. Physiol, «pmc.ncbi.nlm.nih.gov,» National Library of Medicine, 24 Jul 2008. [Ηλεκτρονικό]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC2614017/. [Πρόσβαση 9 January 2026].

[43] F. H. Administration, «fhwa.dot.gov,» Federal Highway Administration, 9 Jul 2006. [Ηλεκτρονικό]. Available: https://www.fhwa.dot.gov/publications/research/safety/pedbike/05085/chapt8.cfm. [Πρόσβαση 9 January 2026].

[44] I. Gherasim, «arxiv.org,» ARXIV, 17 November 2025. [Ηλεκτρονικό]. Available: https://arxiv.org/html/2511.13453v1. [Πρόσβαση 9 January 2026].

[45] A. Ghosh, «learnopencv.com,» BIG VISION, 8 October 2024. [Ηλεκτρονικό]. Available: https://learnopencv.com/yolo11/. [Πρόσβαση 9 January 2026].

[46] A. Developers, «developer.andoird.com,» Google, 18 January 2022. [Ηλεκτρονικό]. Available: https://developer.android.com/ndk/guides/neuralnetworks/migration-guide. [Πρόσβαση 9 January 2026].

[47] S. N. Rao, «medium.com,» MEDIUM.COM, 22 October 2024. [Ηλεκτρονικό]. Available: https://medium.com/@nikhil-rao-20/yolov11-explained-next-level-object-detection-with-enhanced-speed-and-accuracy-2dbe2d376f71. [Πρόσβαση 10 January 2026].

[48] G. Boesch, «viso.ai,» VISO.AI, 6 December 2024. [Ηλεκτρονικό]. Available: https://viso.ai/computer-vision/yolo-explained/. [Πρόσβαση 10 January 2026].

[49] T. G. Priyanto Hidayatullah, «researchgate.net,» ResearchGate, 12 January 2025. [Ηλεκτρονικό]. Available: xhttps://www.researchgate.net/publication/388354012_YOLOv8_to_YOLO11_A_Comprehensive_Architecture_In-depth_Comparative_Review. [Πρόσβαση 11 January 2026].

[50] V. Akavalappil, «researchgate.net,» ResearchGate, 1 September 2025. [Ηλεκτρονικό]. Available: https://www.researchgate.net/figure/SPPF-structure-diagram-YOLO11-incorporates-the-SPPF-module-in-the-second-to-last-layer_fig4_395754864. [Πρόσβαση 11 January 2026].

[51] K. W. Y. H. J. W. Jianwei Huang, «mdpi.com,» MDPI, 26 December 2024. [Ηλεκτρονικό]. Available: https://www.mdpi.com/1424-8220/25/1/65. [Πρόσβαση 11 January 2026].

[52] F. X. Daniel Bolya, «web.cs.ucdavis.edu,» Web CS, 5 May 2022. [Ηλεκτρονικό]. Available: https://web.cs.ucdavis.edu/~yjlee/projects/iccv2019_yolact.pdf. [Πρόσβαση 11 January 2026].

[53] R. Staff, «roboflow.com,» Roboflow, 12 November 2022. [Ηλεκτρονικό]. Available: https://roboflow.com/apply-nms/yolo11. [Πρόσβαση 12 January 2026].