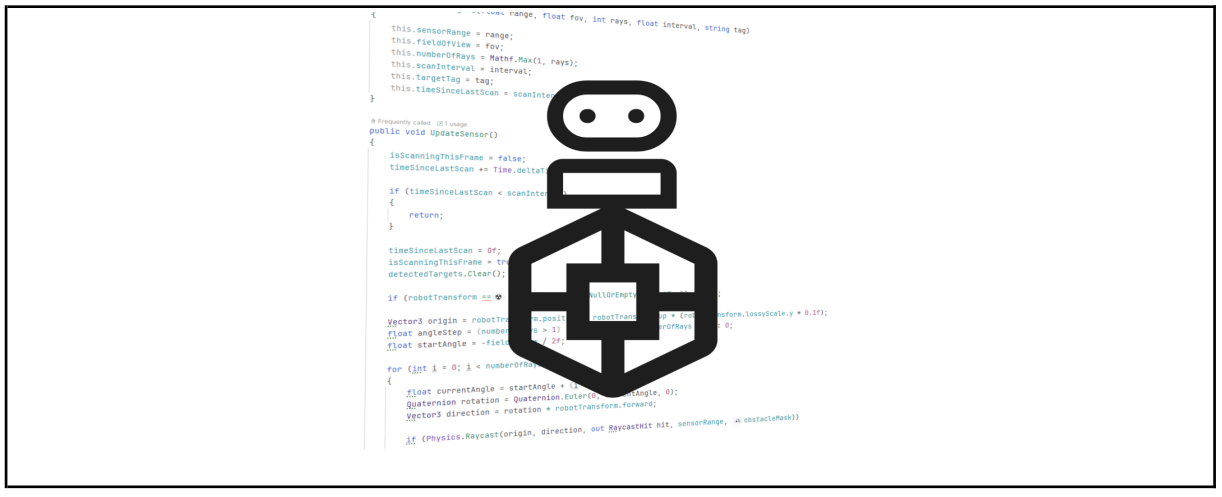
ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

«Χρήση της επαυξημένης πραγματικότητας στην εκπαιδευτική ρομποτική»



Του φοιτητή
Μηδέλια Αθανασίου
Αρ. Μητρώου: 185229

Επιβλέπων
Κεραμόπουλος Ευκλείδης
Καθηγητής

Ημερομηνία 18-01-2026

Τίτλος Δ.Ε. Χρήση της επαυξημένης πραγματικότητας στην εκπαιδευτική ρομποτική
Κωδικός Δ.Ε. 25227
Ονοματεπώνυμο φοιτητή Μηδέλιας Αθανάσιος
Ονοματεπώνυμο εισηγητή Κεραμόπουλος Ευκλείδης
Ημερομηνία ανάληψης Δ.Ε. 30-03-2025
Ημερομηνία περάτωσης Δ.Ε. 23-01-2026

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 έγκριση της διπλωματικής εργασίας από το Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος, δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα, εκ μέρους του Τμήματος.

Πρόλογος

Η επιλογή του συγκεκριμένου θέματος πτυχιακής πήγασε από την επιθυμία μου να εξερευνήσω πώς η Επαυξημένη Πραγματικότητα μπορεί να μετασχηματίσει την εκπαιδευτική ρομποτική, καθιστώντας την πιο προσιτή και διαδραστική.

Μέσα από την εκπόνηση της εργασίας, το όφελος που αποκόμισα ήταν πολυδιάστατο. Σε τεχνικό επίπεδο, εμβάθυνα τις γνώσεις μου στην ανάπτυξη εφαρμογών με τη μηχανή Unity και το AR Foundation, καθώς και στη διασύνδεση υλικού (micro:bit) με λογισμικό μέσω Bluetooth. Παράλληλα, η διαδικασία με βοήθησε να αναπτύξω δεξιότητες επίλυσης σύνθετων προβλημάτων, όπως η διαχείριση αισθητήρων και ο σχεδιασμός αρχιτεκτονικής συστημάτων βασισμένων σε δεδομένα.

Η γεφύρωση της θεωρίας των αλγορίθμων πλοήγησης με την έμπρακτη εφαρμογή τους σε ένα ψηφιακό περιβάλλον υπήρξε μια εξαιρετικά διδακτική εμπειρία. Ευελπιστώ ότι η εργασία αυτή θα αποτελέσει ένα χρήσιμο εργαλείο για την ανάδειξη νέων μεθόδων διδασκαλίας στις επιστήμες STEM, προσφέροντας μια καινοτόμο προσέγγιση στη σύγχρονη εκπαίδευση.

Περίληψη

Αντικείμενο της παρούσας πτυχιακής εργασίας είναι η σχεδίαση και ανάπτυξη μιας καινοτόμου εκπαιδευτικής εφαρμογής που συνδυάζει την Επαυξημένη Πραγματικότητα με τη Ρομποτική. Στόχος της εφαρμογής είναι να προσφέρει ένα προσβάσιμο και διαδραστικό περιβάλλον προσομοίωσης, όπου οι χρήστες μπορούν να πειραματιστούν με βασικές αρχές αυτόνομης πλοήγησης και προγραμματισμού ρομπότ χωρίς την ανάγκη εξειδικευμένου εργαστηριακού εξοπλισμού.

Για την υλοποίηση του συστήματος χρησιμοποιήθηκε η μηχανή ανάπτυξης Unity και το πλαίσιο AR Foundation, επιτρέποντας την οπτικοποίηση ψηφιακών ρομπότ και περιβαλλόντων στον πραγματικό χώρο του χρήστη μέσω κινητών συσκευών Android. Η εφαρμογή υιοθετεί μια αρχιτεκτονική βασισμένη σε δεδομένα (Data-Driven Architecture), η οποία επιτρέπει τη δυναμική παραμετροποίηση της συμπεριφοράς του ρομπότ μέσω ενός γραφικού επεξεργαστή κανόνων (Rule Editor). Το ρομπότ διαθέτει ένα σύνολο προσομοιωμένων αισθητήρων (LiDAR, υπερήχων, υπερύθρων, χρώματος και θερμότητας), οι οποίοι του επιτρέπουν να αντιλαμβάνεται το εικονικό περιβάλλον και να λαμβάνει αποφάσεις πλοήγησης σε πραγματικό χρόνο.

Ένα από τα σημαντικότερα χαρακτηριστικά της εργασίας είναι η υβριδική λειτουργία του συστήματος. Μέσω του πρωτοκόλλου Bluetooth Low Energy (BLE), η εφαρμογή λειτουργεί ως «ψηφιακό δίδυμο», στέλνοντας ασύρματα εντολές σε ένα φυσικό ρομπότ (BBC micro:bit / Yahboom Tiny:bit). Με αυτόν τον τρόπο, οι ενέργειες του εικονικού ρομπότ αναπαράγονται ταυτόχρονα από το φυσικό, ενισχύοντας την κατανόηση της σχέσης μεταξύ προσομοίωσης και πραγματικότητας.

Τα αποτελέσματα των δοκιμών σε επτά διαφορετικά σενάρια προσομοίωσης (όπως αποφυγή εμποδίων, ανίχνευση στόχων και χαρτογράφηση λαβυρίνθου) επιβεβαίωσαν την αξιοπιστία της αρχιτεκτονικής και την εκπαιδευτική αξία της οπτικοποίησης των αισθητήρων. Συμπερασματικά, η εργασία αναδεικνύει την Επαυξημένη Πραγματικότητα ως ένα ισχυρό εργαλείο στη σύγχρονη εκπαίδευση STEM, προσφέροντας μια ολοκληρωμένη πλατφόρμα για την εκμάθηση σύνθετων εννοιών της ρομποτικής με χαμηλό κόστος και υψηλό βαθμό διάδρασης.

Use of Augmented Reality in Educational Robotics

Athanasios Midelias

Abstract

This thesis focuses on the design and development of an innovative educational application that integrates Augmented Reality (AR) with Robotics. The primary objective is to provide an accessible and interactive simulation environment where users can experiment with fundamental principles of autonomous navigation and robot programming without the need for expensive laboratory equipment.

The system was developed using the Unity game engine and the AR Foundation framework, enabling the visualization of digital robots and environments within the user's physical space through Android mobile devices. A key feature of the application is its Data-Driven Architecture, which allows for dynamic parameterization of the robot's behavior through a graphical Rule Editor. The simulated robot is equipped with a comprehensive suite of virtual sensors—including LiDAR, ultrasonic, infrared, color, and thermal sensors—enabling it to perceive its environment and make real-time navigation decisions.

Furthermore, the system supports a hybrid operational mode acting as a "Digital Twin." Utilizing Bluetooth Low Energy (BLE) protocol, the application wirelessly transmits commands to a physical robot (BBC micro:bit / Yahboom Tiny:bit), allowing the virtual robot's actions to be mirrored by the physical hardware. This synchronization enhances the user’s understanding of the relationship between simulation and reality.

Experimental results across seven different scenarios, such as obstacle avoidance, target detection, and maze mapping, validated the reliability of the architecture and the educational value of sensor visualization. In conclusion, this work highlights Augmented Reality as a powerful tool in modern STEM education, offering a comprehensive platform for learning complex robotics concepts with low cost and high interactivity.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά την οικογένεια και τους φίλους μου στάθηκαν δίπλα μου κατά την διάρκεια εκπόνησης της παρούσας εργασίας, αλλά και όλης της διάρκειας των σπουδών μου. Θα ήθελα επίσης να ευχαριστήσω τον καθηγητή μου Κύριο Ευκλείδη Κεραμόπουλο και τον Κύριο Σκαπέτη Γιώργο που μου έδωσαν την ευκαιρία να εκπονήσω αυτήν την εργασία, και να ασχοληθώ με την ένωση διαφορετικών κλάδων της Πληροφορικής για την δημιουργία της εκπαιδευτικής εφαρμογής.

Περιεχόμενα

Κατάλογος Σχημάτων

Συντομογραφίες

Δ.Ε.	Διπλωματική Εργασία
ΔΙΠΑΕ	Διεθνές Πανεπιστήμιο Ελλάδος
Π.Ε.	Πτυχιακή Εργασία
ADC	Analog-to-Digital Converter
AOT	Ahead-Of-Time
API	Application Programming Interface
AR	Augmented Reality
AUV	Autonomous Underwater Vehicle
BLE	Bluetooth Low Energy
CAN	Controller Area Network
DPI	Dots Per Inch
EKF	Extended Kalman Filter
GNSS	Global Navigation Satellite System
GPIO	General-Purpose Input/Output
GPS	Global Positioning System
HMD	Head-Mounted Display
IL	Intermediate Language
IMU	Inertial Measurement Unit
I²C	Inter-Integrated Circuit
JIT	Just-In-Time
LiDAR	Light Detection and Ranging
LINQ	Language Integrated Query
POI	Point of Interest
SDK	Software Development Kit
SLAM	Simultaneous Localization and Mapping
SPI	Serial Peripheral Interface
SRP	Scriptable Render Pipeline
STEM	Science, Technology, Engineering, and Math
TMP	TextMeshPro
UART	Universal Asynchronous Receiver-Transmitter
UI	User Interface
URP	Universal Render Pipeline

VIO	Visual-Inertial Odometry
VR	Virtual Reality

Κεφάλαιο 1ο: Εισαγωγή

1.1 Εισαγωγή

Η ραγδαία εξέλιξη της τεχνολογίας, ειδικά τα τελευταία χρόνια, σε πολλούς τομείς έχει στηριχθεί στα πεδία του προγραμματισμού και της ρομποτικής. Ωστόσο, η κατανόηση και η πρακτική δοκιμή βασικών αρχών της πλοήγησης και του αυτόνομου ελέγχου των ρομπότ συχνά απαιτεί πρόσβαση σε εξειδικευμένο εξοπλισμό και προγραμματιστικά περιβάλλοντα. Η παρούσα πτυχιακή εργασία έχει ως σκοπό την δημιουργία ενός προσβάσιμου, ελκυστικού και πρακτικού εργαλείου που θα γεφυρώνει το χάσμα μεταξύ θεωρίας και πράξης.

Στο πλαίσιο αυτής της πτυχιακής εργασίας, αναπτύχθηκε μία εφαρμογή για κινητές συσκευές Android που αξιοποιεί την τεχνολογία της Επαυξημένης Πραγματικότητας, με χρήση της μηχανής ανάπτυξης παιχνιδιών Unity. Ο κύριος στόχος της εφαρμογής είναι η προσομοίωση της αυτόνομης πλοήγησης ενός ψηφιακού ρομπότ σε ένα εικονικό περιβάλλον, το οποίο με χρήση της κάμερας του κινητού τοποθετείται δυναμικά στον πραγματικό χώρο του χρήστη. Η πλοήγηση του ρομπότ επιτυγχάνεται με τη χρήση προσομοιωμένων αισθητήρων, ενώ η συμπεριφορά του ρομπότ καθορίζεται από ένα ευέλικτο περιβάλλον προγραμματισμού βασισμένο σε ζεύγη κανόνων-ενεργειών.

Ο δευτερεύων στόχος της εφαρμογής είναι η επέκταση αυτής της προσομοίωσης από το ψηφιακό περιβάλλον στον πραγματικό κόσμο. Χρησιμοποιώντας την τεχνολογία Bluetooth Low Energy, η εφαρμογή είναι σχεδιασμένη έτσι ώστε να μπορεί να στέλνει ασύρματα εντολές ανάλογα με την συμπεριφορά του ψηφιακού ρομπότ σε ένα φυσικό ρομπότ. Η αμφίδρομη αυτή λειτουργικότητα προσφέρει στον χρήστη την δυνατότητα να δοκιμάζει και να παρατηρεί τις ίδιες συμπεριφορές τόσο στον εικονικό όσο και στον πραγματικό κόσμο, ενισχύοντας την κατανόηση των αρχών της ρομποτικής.

Η παρούσα έκθεση είναι δομημένη ως εξής:

Κεφάλαιο 2: Επαυξημένη Πραγματικότητα: Αναλύεται το πεδίο της Επαυξημένης Πραγματικότητας (AR), παρουσιάζονται τα βασικά χαρακτηριστικά της (ανίχνευση, καταγραφή, απόδοση) και περιγράφονται οι τεχνολογίες απεικόνισης καθώς και τα εργαλεία ανάπτυξης (SDKs) όπως το ARCore, το ARKit και το AR Foundation.

Κεφάλαιο 3: Ρομποτική: Παρουσιάζονται οι θεμελιώδεις αρχές της αυτόνομης πλοήγησης ρομπότ (εντοπισμός θέσης, χαρτογράφηση, σχεδιασμός τροχιάς) και αναλύεται η χρήση των αισθητήρων (κάμερες, LiDAR, αισθητήρες υπερήχων κ.ά.) για την αντίληψη του περιβάλλοντος.

Κεφάλαιο 4: Simultaneous Localization and Mapping (SLAM): Εξετάζεται το πρόβλημα του Ταυτόχρονου Εντοπισμού και Χαρτογράφησης (SLAM), περιγράφοντας το μαθηματικό υπόβαθρο και τους αλγορίθμους που επιτρέπουν σε ένα ρομπότ να χαρτογραφεί άγνωστα περιβάλλοντα ενώ ταυτόχρονα προσδιορίζει τη θέση του σε αυτά, μία τεχνική που αξιοποιείται και από την επαυξημένη πραγματικότητα.

Κεφάλαιο 5: Τεχνολογίες και Εργαλεία Ανάπτυξης: Περιγράφονται τα εργαλεία υλικού και λογισμικού που χρησιμοποιήθηκαν για την υλοποίηση της εφαρμογής, όπως η μηχανή Unity, το πλαίσιο AR Foundation, ο μικροελεγκτής BBC micro:bit και το πρωτόκολλο επικοινωνίας Bluetooth Low Energy.

Κεφάλαιο 6: Ανάλυση και Σχεδίαση Συστήματος: Αναλύονται οι απαιτήσεις του συστήματος και παρουσιάζεται η αρχιτεκτονική της εφαρμογής, καθορίζοντας τη δομή και τη συμπεριφορά του λογισμικού.

Κεφάλαιο 7: Υλοποίηση και Παρουσίαση Αποτελεσμάτων: Παρουσιάζεται αναλυτικά η διαδικασία υλοποίησης της εφαρμογής, τα σενάρια προσομοίωσης (πλοήγηση με LiDAR, ανίχνευση χρωμάτων, αποφυγή εμποδίων) και τα αποτελέσματα από τη δοκιμή του συστήματος τόσο σε εικονικό όσο και σε πραγματικό περιβάλλον.

Κεφάλαιο 8: Συμπεράσματα και Μελλοντικές Επεκτάσεις: Συνοψίζονται τα συμπεράσματα που προέκυψαν από την εκπόνηση της εργασίας, αναφέρονται οι περιορισμοί του συστήματος και προτείνονται κατευθύνσεις για μελλοντική έρευνα και βελτιώσεις.

Κεφάλαιο 2ο: Επαυξημένη Πραγματικότητα

1.2 Εισαγωγή

Η Επαυξημένη Πραγματικότητα (AR) αποτελεί ένα πεδίο της πληροφορικής που επιδιώκει τον εμπλουτισμό του πραγματικού κόσμου με ψηφιακές πληροφορίες. Σε αντίθεση με την Εικονική Πραγματικότητα (Virtual Reality - VR), η οποία βυθίζει τον χρήστη σε ένα εντελώς εικονικό περιβάλλον, η AR ενσωματώνει ψηφιακά αντικείμενα στον φυσικό χώρο, προσφέροντας μια υβριδική εμπειρία [1].

1.3 Βασικά χαρακτηριστικά της Επαυξημένης Πραγματικότητας

1.3.1 Ανίχνευση και Καταγραφή

Η ακριβής και συνεπής τοποθέτηση των ψηφιακών στοιχείων στο πραγματικό περιβάλλον, γνωστή ως **καταγραφή (registration)**, αποτελεί μία από τις σημαντικότερη προκλήσεις της AR. Για να επιτευχθεί, ένα σύστημα πρέπει να παρακολουθεί συνεχώς τη θέση και τον προσανατολισμό του χρήστη και των αντικειμένων του κόσμου γύρω του. Αυτή η διαδικασία είναι γνωστή ως **ανίχνευση (tracking)**. Η ανίχνευση μπορεί να επιτευχθεί με χρήση αισθητήρων και οπτικών μέσων, έτσι ώστε να παρακολουθείται η θέση στόχων, με βάση τους οποίους γίνεται η τοποθέτηση των ψηφιακών στοιχείων [2].

2.2.2 Απόδοση

Η διαδικασία της απόδοσης αφορά τη δημιουργία των ψηφιακών αντικειμένων και την προβολή τους στην οθόνη του χρήστη κατά τρόπο που να φαίνονται ως αναπόσπαστο μέρος του φυσικού περιβάλλοντος. Για να επιτευχθεί ρεαλισμός, η απόδοση πρέπει να λαμβάνει υπόψη παράγοντες όπως φωτισμός και σκιές, τα εικονικά αντικείμενα πρέπει να φωτίζονται από τις ίδιες πηγές φωτός του πραγματικού περιβάλλοντος και να δημιουργούν ρεαλιστικές σκιές [2]. Επίσης πρέπει να λαμβάνονται υπόψη οι αποκρύψεις (occlusions), δηλαδή ένα ψηφιακό αντικείμενο πρέπει να αποκρύπτεται εν μέρει ή πλήρως όταν βρίσκεται πίσω από ένα φυσικό αντικείμενο [3].

Η σωστή απόδοση των ψηφιακών αντικειμένων είναι καθοριστική για την εμπειρία του χρήστη και την αίσθηση ότι η εικονική προσομοίωση λαμβάνει χώρα στον πραγματικό του χώρο [3].

2.2.3 Τεχνολογίες Απεικόνισης

Η τεχνολογία απεικόνισης αναφέρεται στο μέσο το οποίο χρησιμοποιείται για την προβολή των ψηφιακών αντικειμένων. Στο παρελθόν, η κύρια τεχνολογία απεικόνισης για εφαρμογές επαυξημένης πραγματικότητας ήταν τα Head-Mounted Displays (HMDs), ωστόσο παράγοντες όπως η περιορισμένη ανάλυση, το μικρό οπτικό πεδίο και το μέγεθος και το βάρος των συσκευών, απέτρεψαν την ευρεία υιοθέτηση τους εκτός εργαστηριακών χώρων. [2].

Με την πάροδο του χρόνου, η τεχνολογική εξέλιξη των κινητών συσκευών (όπως smartphones και tablets) και η ευρεία διάδοση τους, τις καθιέρωσε ως τις πιο πολλά υποσχόμενες πλατφόρμες για AR. Η αυξανόμενη υπολογιστική τους ισχύς, σε συνδυασμό με τις βελτιωμένες κάμερες και τις οθόνες υψηλής ανάλυσης, επιτρέπει τη δημιουργία πλούσιων και διαδραστικών εφαρμογών AR, χωρίς την ανάγκη για εξειδικευμένο εξοπλισμό.

2.3 Ανάπτυξη Εφαρμογών Επαυξημένης Πραγματικότητας

Η ανάπτυξη εφαρμογών Επαυξημένης Πραγματικότητας συχνά βασίζεται σε εργαλεία (SDK) που παρέχουν οι κατασκευαστές λειτουργικών συστημάτων, αλλά και εξειδικευμένων συσκευών.

2.3.1 Ανάπτυξη Εφαρμογών AR Για Κινητές Συσκευές

Όσον αφορά την ανάπτυξη εφαρμογών για κινητές συσκευές, υπάρχουν δύο κύρια SDK που παρέχονται από την Google και την Apple, το ARCore και το ARKit αντίστοιχα.

Το ARCore μπορεί να χρησιμοποιηθεί για την ανάπτυξη εφαρμογών για τα συστήματα Android και iOS, και παρέχει λειτουργίες όπως ανίχνευση επίπεδων επιφανειών, ανίχνευση τοποθεσίας της κινητής συσκευής με βάση την τοποθεσία της στον χώρο, ανίχνευση κίνησης, εκτίμηση του φωτισμού του περιβάλλοντος και του βάθους. Πέρα από την ανίχνευση, το ARCore δίνει την δυνατότητα τοποθέτησης ψηφιακών δεικτών (Anchors) για να επιτυγχάνεται η συνεπής τοποθέτηση ψηφιακών αντικειμένων στο χώρο, και να συγκρατείται η ακριβής θέση τους. Για την επίτευξη αυτών των λειτουργιών, γίνεται χρήση της κάμερας της κινητής συσκευής, αλλά και αισθητήρων όπως το γυροσκόπιο και τον αισθητήρα βάθους. [4]

Το ARKit παρέχει αντίστοιχες λειτουργίες, ωστόσο είναι διαθέσιμο μόνο για συσκευές που χρησιμοποιούν το λειτουργικό σύστημα iOS.

Τα SDKs ARCore και ARKit αναπτύχθηκαν ως εξειδικευμένες λύσεις για την διευκόλυνση της δημιουργίας AR εφαρμογών, παρέχοντας σημαντική λειτουργικότητα. Ωστόσο, η ανάπτυξη πλούσιων και διαδραστικών AR εφαρμογών συχνά απαιτεί περαιτέρω λειτουργικότητα που δεν προσφέρεται από τα SDK, όπως απόδοση τρισδιάστατων αντικειμένων με ρεαλιστικό φωτισμό και εργαλεία ανάπτυξης UI. Για την επίλυση αυτών των προβλημάτων, η μηχανή ανάπτυξης ηλεκτρονικών παιχνιδιών Unity προσφέρει το AR Foundation, μία λύση που συνδυάζει τα εργαλεία ανάπτυξης AR με χαρακτηριστικά ανάπτυξης παιχνιδιών όπως 3D Rendering, προσομοιώσεις φυσικής (Physics Engine), UI και αναπαραγωγής ήχων και μουσικής, επιτρέποντας έτσι την δημιουργία πλούσιων εφαρμογών και παιχνιδιών επαυξημένης πραγματικότητας. Μία άλλη σημαντική λειτουργικότητα που προσφέρει το AR Foundation είναι η απλοποίηση της ανάπτυξης των εφαρμογών, καθώς δρα ως ένα αφαιρετικό επίπεδο πάνω από τα AR SDK. Μία εφαρμογή που αναπτύχθηκε για το AR Foundation μπορεί να χρησιμοποιήσει ως υπόβαθρο είτε το ARCore, είτε το ARKit, και μπορεί να εκτελεστεί στις πλατφόρμες Android και iOS.

2.3.2 Ανάπτυξη Εφαρμογών AR Για Εξειδικευμένες Συσκευές

Η ανάπτυξη εφαρμογών AR για εξειδικευμένες συσκευές βασίζεται είτε SDK γενικής χρήσης, είτε σε SDK του κατασκευαστή της συσκευής.

Ένα παράδειγμα SDK γενικής χρήσης αποτελεί το Microsoft Mixed Reality Toolkit, το οποίο αξιοποιεί την μηχανή παιχνιδιών Unity για δυνατότητες προβολής, και υποστηρίζει συσκευές HMD όπως το Microsoft HoloLens, το Oculus Rift και το Meta Quest, και παρέχει έτοιμες λύσεις για προηγμένες λειτουργίες όπως ανίχνευση των ματιών του χρήστη για αλληλεπίδραση, ανίχνευση χειρονομιών για την εκτέλεση εντολών, καθώς και φωνητικές εντολές. Παράλληλα υποστηρίζει ανίχνευση επιφανειών, βάθους και κίνησης.

Ένα παράδειγμα SDK εξειδικευμένης συσκευής είναι το Magic Leap SDK για τις HMD συσκευές Magic Leap, το οποίο επίσης χρησιμοποιεί την μηχανή παιχνιδιών Unity για δυνατότητες προβολής, και παρέχει παρόμοιες δυνατότητες με το Microsoft Mixed Reality Toolkit.

2.4 Πεδία Εφαρμογών της Επαυξημένης Πραγματικότητας

Η Επαυξημένη Πραγματικότητα (AR) έχει εφαρμοστεί σε διάφορους τομείς για την εμπλούτιση εμπειριών του χρήστη. Παρακάτω παρουσιάζονται κάποια από τα πεδία εφαρμογών της επαυξημένης πραγματικότητας.

2.4.1 Εκπαίδευση και Κατάρτιση

Στον εκπαιδευτικό τομέα, η επαυξημένη πραγματικότητα επιτρέπει την ενσωμάτωση ψηφιακών πληροφοριών, όπως κείμενο, εικόνες, βίντεο ή 3D μοντέλα στο πραγματικό περιβάλλον των μαθητών. Αυτό επιτρέπει την ενεργή συμμετοχή της με το υλικό μάθησης, η οποία συμβάλλει στην καλύτερη κατανόηση και την υψηλότερη διατήρηση της γνώσης [5].

Στην τριτοβάθμια εκπαίδευση, ιδίως στις ιατρικές επιστήμες, εφαρμογές AR όπως η AR Magic Mirror ή το HoloHuman, έχουν αποδειχθεί ιδιαίτερα αποτελεσματικές για την εκμάθηση ανατομίας και χειρουργικών δεξιοτήτων, βελτιώνοντας την αντίληψη της τρισδιάστατης δομής του ανθρώπινου σώματος και την επιτυχή απόδοση σε αξιολογήσεις.[6, 7]

2.4.2 Ψυχαγωγία

Στον χώρο της ψυχαγωγίας, η AR προσφέρει διαδραστικές εμπειρίες που μετατρέπουν τον χρήστη από παθητικό θεατή σε ενεργό συμμετέχοντα. Μπορεί να εφαρμοστεί σε παιχνίδια, ζωντανές παραστάσεις, ψηφιακά ερωτηματολόγια σε τοποθεσίες [11], ενώ στις παρουσιάσεις αρχαιολογικών χώρων και μουσείων, η εμπειρία των επισκεπτών μπορεί να εμπλουτιστεί με την χρήση της επαυξημένης πραγματικότητας από εφαρμογές όπως η Cosmote Chronos, η οποία επιτρέπει την ψηφιακή προβολή της ανακατασκευασμένης μορφής κατεστραμμένων ή χαμένων αντικειμένων και χώρων ιστορικής σημασίας στην Ακρόπολη Αθηνών και στο Μουσείο Ακρόπολης [8].

Η εμβύθιση που προσφέρει η AR σε τέτοιες εμπειρίες ενισχύει την οπτικο-κινητική αίσθηση και τη μνήμη, ενώ αυξάνει την ευχαρίστηση και τις θετικές αντιλήψεις των χρηστών [11].

2.4.3 Ιατρική και Υγεία

Στον ιατρικό τομέα, η AR έχει πολλαπλές εφαρμογές, από τη βελτίωση της εκπαίδευσης και της προσομοίωσης χειρουργικών επεμβάσεων έως τη βοήθεια κατά την πραγματική εκτέλεση ιατρικών διαδικασιών. Για παράδειγμα, AR προσομοιώσεις εκπαίδευσης όπως τα ProMIS και ImmersiveTouch επιτρέπουν την εκπαίδευση σε λαπαροσκοπικές τεχνικές με ρεαλιστική ανατροφοδότηση [9].

Στην κλινική πράξη, διατάξεις AR-assisted χειρουργίων επιτρέπουν την προβολή 3D ανατομικών πληροφοριών απευθείας στο πεδίο όρασης του χειρουργού, βελτιώνοντας την ακρίβεια και μειώνοντας τα σφάλματα [10].

2.4.4 Μάρκετινγκ

Η χρήση της Επαυξημένης Πραγματικότητας (AR) στον τομέα του μάρκετινγκ έχει αναδειχθεί ως μια καινοτόμος στρατηγική για την ενίσχυση της εμπειρίας του καταναλωτή και την αύξηση της αλληλεπίδρασης με το κοινό. Πολλές εταιρείες έχουν υιοθετήσει την AR για να δημιουργήσουν διαδραστικές και ελκυστικές καμπάνιες ή εφαρμογές που ενισχύουν την αναγνωρισιμότητα της μάρκας και προάγουν την αφοσίωση των πελατών.

Κάποια παραδείγματα από διαφημιστικές καμπάνιες που έκαναν χρήση της AR αποτελούν το “Unbelievable Bus Shelter” της Pepsi Max, όπου με χρήση κάμερας και οθονών σε στάσεις λεωφορείων, γινόταν προβολή φανταστικών περιστατικών στον πραγματικό χώρο γύρω από την

στάση [11], και η εφαρμογή Beauty Genius της L'Oréal που επιτρέπει στους χρήστες να δοκιμάσουν στο πρόσωπο τους εικονικά προϊόντα μακιγιάζ μέσω της κάμερας του κινητού, προσφέροντας μια προσωποποιημένη εμπειρία αγορών [12].

2.5 Επίλογος

Στο παρόν κεφάλαιο αναλύθηκε το πεδίο της Επαυξημένης Πραγματικότητας (AR), προσδιορίζοντας τα βασικά χαρακτηριστικά που τη διακρίνουν από την Εικονική Πραγματικότητα. Εξετάστηκαν οι θεμελιώδεις τεχνολογίες που επιτρέπουν την υλοποίηση εμπειριών AR, με έμφαση στις διαδικασίες της ανίχνευσης, της καταγραφής και της ρεαλιστικής απόδοσης ψηφιακών αντικειμένων στον φυσικό χώρο.

Ιδιαίτερη βαρύτητα δόθηκε στα εργαλεία ανάπτυξης λογισμικού, παρουσιάζοντας τα ARCore και ARKit, καθώς και το πλαίσιο AR Foundation της Unity. Το τελευταίο αναδείχθηκε ως το κατάλληλο εργαλείο για την παρούσα εργασία, καθώς προσφέρει ένα ενιαίο περιβάλλον ανάπτυξης που συνδυάζει δυνατότητες AR με μηχανισμούς φυσικής και γραφικών.

Κεφάλαιο 3ο: Ρομποτική

3.1 Εισαγωγή

Η ρομποτική αποτελεί έναν από τους πιο δυναμικούς και διεπιστημονικούς τομείς της σύγχρονης τεχνολογίας, συνδυάζοντας μηχανική, πληροφορική, τεχνητή νοημοσύνη και αισθητήρες για την ανάπτυξη αυτόνομων συστημάτων. Η πλοήγηση (navigation) και η χρήση αισθητήρων είναι θεμελιώδη για την ικανότητα των ρομπότ να αντιλαμβάνονται το περιβάλλον τους και να εκτελούν εργασίες με ακρίβεια και ασφάλεια [13].

3.2 Βασικές Αρχές Πλοήγησης Ρομπότ

Η πλοήγηση αποτελεί έναν από τους σημαντικότερους άξονες στη ρομποτική, καθώς καθορίζει την ικανότητα ενός ρομπότ να κινείται αυτόνομα σε ένα περιβάλλον, αποφεύγοντας εμπόδια και φτάνοντας σε προκαθορισμένους στόχους. Η διαδικασία της πλοήγησης περιλαμβάνει τρία κύρια στοιχεία: εντοπισμό θέσης (localization), χαρτογράφηση (mapping) και σχεδιασμό κίνησης (path planning) [14].

3.2.1 Εντοπισμός Θέσης (Localization)

Ο εντοπισμός θέσης (localization) αφορά τη συνεχή εκτίμηση της θέσης και του προσανατολισμού του ρομπότ μέσα στον χώρο. Για την επίτευξη ακριβούς εντοπισμού, έχουν αναπτυχθεί διάφορες τεχνικές. Μία από τις πιο απλές μεθόδους είναι η οδομετρία (odometry), η οποία χρησιμοποιεί δεδομένα από τους τροχούς ή τους κινητήρες του ρομπότ για να υπολογίσει την αλλαγή της θέσης του. Ωστόσο, η μέθοδος αυτή είναι ευαίσθητη σε σφάλματα που συσσωρεύονται με την πάροδο του χρόνου, οδηγώντας σε ανακριβείς εκτιμήσεις [14].

Για την αντιμετώπιση αυτής της πρόκλησης, χρησιμοποιούνται πιο προηγμένες τεχνικές που βασίζονται σε πιθανοτικά φίλτρα (probabilistic filters) [14]. Χαρακτηριστικό παράδειγμα αποτελεί η μέθοδος Monte Carlo Localization (MCL), η οποία χρησιμοποιεί ένα σύνολο σωματιδίων για να εκτιμήσει την πιθανή θέση του ρομπότ, ακόμη και σε αβέβαια περιβάλλοντα [15]. Η ακρίβεια της εκτίμησης μπορεί επίσης να επιτευχθεί με τη χρήση του Extended Kalman Filter (EKF), ενός αλγορίθμου που συγχωνεύει δεδομένα από τους αισθητήρες με την προβλεπόμενη κίνηση του ρομπότ, με στόχο τη μείωση της αβεβαιότητας στον εντοπισμό [16].

3.2.2 Χαρτογράφηση (Mapping)

Η χαρτογράφηση (mapping) είναι η διαδικασία δημιουργίας μιας αναπαράστασης του περιβάλλοντος, η οποία μπορεί να χρησιμοποιηθεί για τον σχεδιασμό της διαδρομής. Η χαρτογράφηση μπορεί να γίνει με τη μέθοδο Simultaneous Localization and Mapping (SLAM), η οποία επιλύει το πρόβλημα του να εντοπίζει ένα ρομπότ τη θέση του και ταυτόχρονα να δημιουργεί έναν χάρτη του άγνωστου περιβάλλοντος [17].

Οι χάρτες που δημιουργούνται μπορούν να έχουν διάφορες μορφές, ένα παράδειγμα είναι οι Μετρικοί Χάρτες (metric maps), οι οποίοι βασίζονται σε γεωμετρικές αναπαραστάσεις του χώρου, όπως τα grid maps, τα οποία χωρίζουν το περιβάλλον σε ένα πλέγμα κελιών που δηλώνουν αν ένας χώρος είναι ελεύθερος ή κατειλημμένος [18].

3.2.3 Σχεδιασμός Κίνησης (Path Planning)

Ο σχεδιασμός κίνησης (path planning) αφορά τον υπολογισμό μιας ασφαλούς και αποδοτικής διαδρομής από την αρχική θέση του ρομπότ προς τον στόχο του, λαμβάνοντας υπόψη τα εμπόδια. Οι αλγόριθμοι που χρησιμοποιούνται για τον σχεδιασμό μπορεί να διαφέρουν ανάλογα με τη γνώση που έχει το ρομπότ για το περιβάλλον [19]. Κλασικοί αλγόριθμοι περιλαμβάνουν τον A-star (A*), μια ευρετική αναζήτηση που εξασφαλίζει τη βέλτιστη διαδρομή σε στατικούς, γνωστούς χάρτες [20]. Για πιο σύνθετα ή δυναμικά περιβάλλοντα, χρησιμοποιούνται πιο προηγμένοι αλγόριθμοι, όπως ο D-star (D*), μια βελτίωση του A* που προσαρμόζεται σε αλλαγές του περιβάλλοντος, ή ο Rapidly-exploring Random Trees (RRT), μια δειγματοληπτική μέθοδος για πολύπλοκα περιβάλλοντα υψηλών διαστάσεων [21], και η παραλλαγή του RRT* η οποία, με όσο το δυνατόν μικρότερες απαιτήσεις μνήμης και επεξεργαστικής ισχύς, βελτιώνει την ποιότητα της διαδρομής συγκλίνοντας προς τη βέλτιστη λύση [22].

Επιπλέον, πιο πρόσφατες προσεγγίσεις βασίζονται σε **μεθόδους βελτιστοποίησης** και **τεχνητής νοημοσύνης**, όπως αλγόριθμοι σμήνους (swarm intelligence), γενετικοί αλγόριθμοι και νευρωνικά δίκτυα, που επιτρέπουν στο ρομπότ να σχεδιάζει διαδρομές σε περίπλοκα και αβέβαια περιβάλλοντα.

3.3 Χρήση Αισθητήρων Στην Πλοήγηση

Η πλοήγηση ενός ρομπότ βασίζεται σε μεγάλο βαθμό στη χρήση αισθητήρων, οι οποίοι παρέχουν κρίσιμες πληροφορίες για την κατανόηση και αλληλεπίδραση με το περιβάλλον. Μέσα από αυτούς, το ρομπότ μπορεί να προσδιορίζει τη θέση του, να ανιχνεύει εμπόδια, να εκτιμά τον προσανατολισμό του και να αντιλαμβάνεται περιβαλλοντικές συνθήκες [23].

3.3.1 Οπτικοί Αισθητήρες (Cameras)

Οι κάμερες αποτελούν θεμελιώδες εργαλείο στη ρομποτική όραση. Ανάλογα με τη διαμόρφωση και τις δυνατότητές τους, διακρίνονται σε μονοκάμερες, στερεοσκοπικές και κάμερες με αισθητήρα βάθους. Επιτρέπουν την ανίχνευση και αναγνώριση αντικειμένων, την ανάλυση χαρακτηριστικών σημείων και τη δημιουργία οπτικών χαρτών, αποτελώντας βασικό στοιχείο για αλγορίθμους SLAM και πλοήγησης [24, 25].

- **Μονοκάμερες (Monocular Cameras)**:
  Οι μονοκάμερες χρησιμοποιούν μία μόνο οπτική πηγή και δεν παρέχουν προηγμένα χαρακτηριστικά όπως άμεσες μετρήσεις βάθους. Ωστόσο, μέσω αλγορίθμων όπως το MonoSLAM και οι μέθοδοι Visual Odometry, μπορούν να εκτιμήσουν την κίνηση του ρομπότ [26, 27] και να ανακατασκευάσουν τη δομή του περιβάλλοντος με τη βοήθεια τεχνικών Structure-from-Motion (SfM) [24]. Χρησιμοποιούνται ευρέως σε drones και μικρά ρομπότ, λόγω του χαμηλού κόστους, του μικρού βάρους και της χαμηλής ενεργειακής κατανάλωσης, χαρακτηριστικά ιδιαίτερα σημαντικά για πλατφόρμες όπου το βάρος και η αυτονομία είναι κρίσιμοι παράγοντες [28]. Εξαρτώνται όμως έντονα από την ποιότητα της ανίχνευσης χαρακτηριστικών και μπορεί να επηρεαστούν σε δυναμικά ή θορυβώδη περιβάλλοντα [29].

- **Στερεοσκοπικές Κάμερες (Stereo Cameras)**
  Οι στερεοσκοπικές κάμερες αποτελούνται από δύο συγχρονισμένους φακούς σε σταθερή απόσταση. Χρησιμοποιώντας την επιπολική γεωμετρία, επιτρέπουν τον υπολογισμό βάθους μέσω της σύγκρισης διαφοράς θέσης μεταξύ δύο εικόνων [30]. Προσφέρουν ακριβείς εκτιμήσεις βάθους σε πραγματικό χρόνο και χρησιμοποιούνται εκτενώς σε αυτόνομα οχήματα

και γενικότερα στην ρομποτική [31]. Ωστόσο, η ακρίβεια μπορεί να μειωθεί σε περιβάλλοντα με χαμηλή υφή ή κακό φωτισμό [32]. Εξελιγμένοι αλγόριθμοι όπως ο Semi-Global Matching (SGM) βελτιώνουν σημαντικά την ακρίβεια των στερεο-χαρτών βάθους [33].

- **Κάμερες με Αισθητήρα Βάθους (RGB-D Cameras):**
  Οι RGB-D κάμερες, όπως οι Microsoft Kinect και Intel RealSense, συνδυάζουν έγχρωμες εικόνες (RGB) με μετρήσεις βάθους (D - Depth), χρησιμοποιώντας υπέρυθρη προβολή και ανίχνευση. Επιτρέπουν την απευθείας δημιουργία πυκνών τρισδιάστατων χαρτών του περιβάλλοντος, γεγονός που τις καθιστά εξαιρετικά χρήσιμες για μεθόδους SLAM και 3D ανακατασκευή [34]. Είναι ιδανικές για εφαρμογές σε εσωτερικούς χώρους, όπως ρομποτικούς βοηθούς και αποθήκες, αν και παρουσιάζουν περιορισμένη εμβέλεια (3 έως 5 μέτρα) και είναι ευαίσθητες στο έντονο φυσικό φως [25].

3.3.2 LiDAR (Light Detection and Ranging)

Ο αισθητήρας **LiDAR** βασίζεται στην εκπομπή παλμών λέιζερ και στη μέτρηση του χρόνου που απαιτείται για την επιστροφή τους, ώστε να υπολογιστεί με μεγάλη ακρίβεια η απόσταση των αντικειμένων, καθώς και η μορφή του περιβάλλοντος [14].

Το LiDAR παρέχει άμεσες μετρήσεις απόστασης και χαρτογράφησης με μεγάλη ακρίβεια, κάτι που το καθιστά ιδιαίτερα χρήσιμο σε περιβάλλοντα με υψηλή πολυπλοκότητα ή δυσκολίες στην πρόσβαση, όπως ορυχεία ή σπήλαια [35]. Μία άλλη σημαντική εφαρμογή των μετρήσεων LiDAR γίνεται στην αυτόνομη οδήγηση οχημάτων. Οι μετρήσεις LiDAR μπορούν να αξιοποιηθούν για την ανίχνευση σε πραγματικό χρόνο άλλων οχημάτων, πεζών και οδοσήμανσης, εξασφαλίζοντας την ασφάλεια του οχήματος και του περιβάλλοντος γύρω του ενώ ταυτόχρονα επιτυγχάνεται η πλοήγηση σε ένα μεταβλητό και πολύπλοκο περιβάλλον [36].

Η δημιουργία δυναμικών χαρτών με χρήση LiDAR επιτυγχάνεται με την εφαρμογή αλγορίθμων SLAM πάνω στα δεδομένα που λαμβάνονται, ενώ ταυτόχρονα μπορεί να εκτιμάται η θέση του ρομπότ στον χώρο ανά πάσα στιγμή [14].

Παρά τα πλεονεκτήματά του, το LiDAR παρουσιάζει και προκλήσεις στην υιοθέτηση του. Έντονες ατμοσφαιρικές συνθήκες όπως δυνατή βροχή, ομίχλη ή χιόνι επηρεάζουν τις μετρήσεις καθώς μπορούν να προκαλέσουν εξασθένηση των παλμών λέιζερ, οδηγώντας στην μείωση της απόστασης ανίχνευσης, την απώλεια πληροφορίας και την δημιουργία θορύβου στα δεδομένα [37].

3.3.3 GPS (Global Positioning System)

Η θέση, η ταχύτητα και ο χρόνος ενός δέκτη μπορούν να προσδιοριστούν με τη χρήση του **Global Positioning System (GPS)**, ενός δορυφορικού συστήματος εντοπισμού τύπου GNSS που καλύπτει ολόκληρη την υφήλιο. Για να λειτουργήσει σωστά απαιτούνται τουλάχιστον τέσσερις δορυφόρους να μεταδίδουν σήματα με ακριβή χρονική σήμανση, τα οποία λαμβάνονται από τον δέκτη GPS. Υπολογίζοντας τον χρόνο διάδοσης του σήματος, ο δέκτης εκτιμά την απόστασή του από κάθε δορυφόρο και κατ’ επέκταση την τρισδιάστατη θέση του [38].

Στη ρομποτική και την πλοήγηση, το GPS έχει καθιερωθεί ως θεμελιώδες εργαλείο, ιδιαίτερα για εφαρμογές σε εξωτερικούς χώρους. Παραδείγματα χρήσης του GPS περιλαμβάνουν:

- **Πλοήγηση αυτόνομων οχημάτων**:
  Χρησιμοποιείται για την εκτίμηση της παγκόσμιας θέσης του οχήματος, και μπορεί να συνδυαστεί με άλλους αισθητήρες όπως LiDAR και αισθητήρες κίνησης έτσι ώστε να γίνεται ενημέρωση της τρισδιάστατης τοποθεσίας του οχήματος σε πραγματικό χρόνο, πάνω σε χάρτες υψηλής ακρίβειας [39].

- **Αγροτική ρομποτική**:
  Τα γεωργικά ρομπότ μπορούν να αξιοποιήσουν το GPS για ακριβή εκτέλεση εργασιών σε μεγάλες εκτάσεις, όπως η καταγραφή της ακριβής τοποθεσίας εμφύτευσης κάθε σπόρου στο έδαφος, και η πλοήγηση του ρομπότ εντός εμβέλειας σπαρτών για την ανίχνευση της κατάστασης τους με χρήση άλλων αισθητήρων [40].

Πλεονεκτήματα του GPS αποτελούν η παγκόσμια κάλυψη, καθώς το GPS μπορεί να λειτουργήσει οπουδήποτε στην Γη χωρίς την ανάγκη τοπικών υποδομών, και η συμβατότητα με άλλα GNSS συστήματα όπως το GLONASS και το GALILEO για παροχή μεγαλύτερης αξιοπιστίας και επιδόσεων [38].

Κάποια σημαντικά μειονεκτήματα του GPS είναι:

- **Η εξάρτηση από δορυφορικά σήματα**:
  Σε περιβάλλοντα με χαμηλή ορατότητα προς τον ουρανό ή σε περιπτώσεις που γίνεται ανάκλαση σήματος από κτίρια ή επιφάνειες, η απόδοση του GPS μειώνεται σημαντικά [41].

- **Ευαισθησία σε παρεμβολές:**
  Παρεμβολή του σήματος προκύπτει όταν πιο ισχυρά ραδιοσήματα στην ίδια ή κοντινή συχνότητα εκπομπής με το GPS υπερκαλύπτουν το σήμα των δορυφόρων, με αποτέλεσμα ο δέκτης να αποτυγχάνει να προσδιορίσει τη θέση του. Παρεμβολή μπορεί να προκύψει μη-εσκεμμένα, όπως σε αστικές περιοχές με ισχυρά ηλεκτρομαγνητικά περιβάλλοντα ή εσκεμμένα, όπως σε στρατιωτικά σενάρια όπου γίνονται στοχευμένες παρεμβολές με στόχο την αποτροπή χρήσης συστημάτων GPS [38].

- **Ευαισθησία σε παραποίηση σήματος (spoofing):**
  Στο spoofing, ένας επιτιθέμενος εκπέμπει ψευδή σήματα GPS με σκοπό να ξεγελάσει τον δέκτη, οδηγώντας τον σε λανθασμένη εκτίμηση θέσης ή χρόνου. Αυτό αποτελεί σημαντική απειλή για αυτόνομα οχήματα, καθώς μπορεί να προκαλέσει εκτροπή πορείας ή σφάλματα πλοήγησης [42].

3.3.4 Αισθητήρες Κίνησης και Αδρανειακά Συστήματα Μέτρησης

Τα αδρανειακά συστήματα μέτρησης (Inertial Measurement Units – IMU) αποτελούν έναν συνδυασμό από τους θεμελιώδεις αισθητήρες στην πλοήγηση και τη ρομποτική, καθώς επιτρέπουν την εκτίμηση της κίνησης και του προσανατολισμού ενός φορέα σε πραγματικό χρόνο. Συνήθως συνδυάζουν τρεις κατηγορίες αισθητήρων: επιταχυνσιόμετρα, γυροσκόπια και μαγνητόμετρα τα οποία παρέχουν αμοιβαία συμπληρωματικές πληροφορίες [43], [44].

Τα **επιταχυνσιόμετρα** αποτελούν αισθητήρες που μετρούν τις γραμμικές επιταχύνσεις κατά μήκος των αξόνων τους. Με την ολοκλήρωση των μετρήσεων αυτών μπορεί να υπολογιστεί η ταχύτητα και, σε δεύτερο επίπεδο, η μετατόπιση του συστήματος. Παρά την απλότητα και τη χαμηλή κατανάλωση ενέργειας που τα καθιστούν ιδανικά για φορητές και ρομποτικές εφαρμογές, η αξιοπιστία τους περιορίζεται χρονικά λόγω της συσσώρευσης σφαλμάτων (drift). Τα μικρά σφάλματα μέτρησης, όταν συμβαίνουν επανειλημμένα, οδηγούν σε εκθετική αύξηση του σφάλματος θέσης, καθιστώντας τα επιταχυνσιόμετρα κατάλληλα μόνο για βραχυπρόθεσμες εκτιμήσεις θέσης και κίνησης [43], [45].

Τα **γυροσκόπια**, αντίστοιχα, μετρούν τις γωνιακές ταχύτητες γύρω από τους τρεις άξονες περιστροφής και παρέχουν πληροφορίες σχετικά με τον προσανατολισμό του συστήματος μέσω ολοκλήρωσης των μετρήσεων. Προσφέρουν υψηλή ακρίβεια σε μικρά χρονικά διαστήματα και είναι κρίσιμα για εφαρμογές όπου απαιτείται γρήγορη και ακριβής απόκριση, όπως στην πλοήγηση αυτόνομων οχημάτων και στην αεροναυτική. Ωστόσο, και αυτοί οι αισθητήρες υπόκεινται σε σφάλματα: η σταδιακή απόκλιση λόγω σταθερών σφαλμάτων, οι θερμικές επιδράσεις και ο θόρυβος χαμηλής συχνότητας οδηγούν σε σταδιακή υποβάθμιση της ακρίβειας των εκτιμήσεων προσανατολισμού [45], [46].

Ο συνδυασμός επιταχυνσιόμετρων και γυροσκοπίων στο πλαίσιο ενός αδρανειακού συστήματος μέτρησης (IMU) είναι θεμελιώδης: τα επιταχυνσιόμετρα παρέχουν αναφορά για τη βαρύτητα και επομένως βοηθούν στη σταθεροποίηση του κατακόρυφου προσανατολισμού, ενώ τα γυροσκόπια προσφέρουν βραχυπρόθεσμη ακρίβεια στις γωνιακές κινήσεις. Με την εφαρμογή τεχνικών **sensor fusion** (π.χ. φίλτρο Kalman ή φίλτρα συμπληρωματικού τύπου), οι δύο αυτές τεχνολογίες αλληλοσυμπληρώνονται, με τα επιταχυνσιόμετρα να διορθώνουν την απόκλιση των γυροσκοπίων και τα γυροσκόπια να εξομαλύνουν τον θόρυβο των επιταχυνσιόμετρων [43], [45].

Τα μαγνητόμετρα αποτελούν βασικούς αισθητήρες προσανατολισμού, καθώς μετρούν την ένταση και την κατεύθυνση του γήινου μαγνητικού πεδίου και επιτρέπουν την εκτίμηση της πορείας προς τον μαγνητικό βορρά. Χρησιμοποιούνται ευρέως σε εφαρμογές πλοήγησης, είτε αυτόνομα είτε ως μέρος ολοκληρωμένων αδρανειακών συστημάτων μέτρησης (IMU), σε συνδυασμό με επιταχυνσιόμετρα και γυροσκόπια, ώστε να διορθώνουν τη συσσώρευση σφαλμάτων (drift) που εμφανίζεται στους υπόλοιπους αισθητήρες [47], [48].

Παρά τη χρησιμότητά τους ως αισθητήρες απόλυτου προσανατολισμού, τα μαγνητόμετρα παρουσιάζουν σημαντικά μειονεκτήματα. Είναι ιδιαίτερα ευάλωτα σε τοπικές ηλεκτρομαγνητικές παρεμβολές, μεταλλικές κατασκευές και μαγνητικές ανωμαλίες, γεγονός που μπορεί να επηρεάσει σημαντικά την ακρίβεια και την αξιοπιστία τους, ειδικά σε αστικά ή βιομηχανικά περιβάλλοντα [47], [48]. Επιπλέον, απαιτούν διαδικασίες καλιμπραρίσματος ώστε να αντισταθμίζονται σφάλματα που οφείλονται σε μαγνητική παραμόρφωση του ίδιου του αισθητήρα ή του φορέα στον οποίο είναι τοποθετημένος [47].

Συνολικά, τα μαγνητόμετρα αποτελούν έναν κρίσιμο αισθητήρα για την πλοήγηση, καθώς παρέχουν αναφορά σε απόλυτες κατευθύνσεις, αλλά η αξιοπιστία τους εξαρτάται άμεσα από το περιβάλλον και την ποιότητα της βαθμονόμησης. Έτσι, ως αποτέλεσμα, σπανίως χρησιμοποιούνται μεμονωμένα στη ρομποτική, αλλά ενσωματώνονται σε συνδυαστικά σχήματα αισθητήρων (sensor fusion) με IMU και GPS για πιο ανθεκτικά και αξιόπιστα αποτελέσματα [48], [45].

Ο συνδυασμός επιταχυνσιόμετρων, γυροσκοπίων και μαγνητόμετρων σε ένα IMU παρέχει ολοκληρωμένη πληροφόρηση για την κίνηση και τον προσανατολισμό ενός συστήματος. Ωστόσο, η αποκλειστική χρήση ενός IMU χωρίς εξωτερικές πηγές αναφοράς οδηγεί σε σταδιακή απώλεια ακρίβειας λόγω της συσσώρευσης σφαλμάτων και του drift. Για τον λόγο αυτό, τα IMU συχνά προστίθενται σε συστήματα αισθητηριακής σύντηξης (sensor fusion), όπου οι μετρήσεις τους συνδυάζονται με άλλους αισθητήρες, όπως LiDAR, GPS, ή οπτικούς αισθητήρες. Μέθοδοι όπως το εκτεταμένο φίλτρο Kalman (EKF) ή οι παραγοντικοί γράφοι επιτρέπουν την εκτίμηση της θέσης και του προσανατολισμού με υψηλή ακρίβεια και αξιοπιστία [43], [49].

Η χρήση των IMU προσφέρει υψηλή συχνότητα μετρήσεων, ανεξαρτησία από εξωτερικά σήματα και δυνατότητα λειτουργίας σε περιβάλλοντα όπου άλλες πηγές πληροφορίας, όπως το GPS, είναι περιορισμένες. Παράλληλα, τα μειονεκτήματα τους, κυρίως η συσσώρευση σφαλμάτων και η ευαισθησία σε θερμικές μεταβολές και ηλεκτρομαγνητικές παρεμβολές, καθιστούν αναγκαίο τον συνδυασμό τους με άλλους αισθητήρες για αξιόπιστη και ακριβή πλοήγηση [43], [49], [48].

3.3.5 Αισθητήρες Αφής (Bump Sensors)

Οι αισθητήρες αφής χρησιμοποιούνται στην ανίχνευση φυσικής επαφής με εμπόδια και χρησιμοποιούνται ευρέως σε κινητά ρομπότ για την αποφυγή συγκρούσεων και την πλοήγηση σε περιορισμένα περιβάλλοντα. Αν και απλοί στην κατασκευή, προσφέρουν αξιόπιστη λειτουργία σε συνθήκες όπου άλλοι αισθητήρες ενδέχεται να αποτύχουν, όπως σε περιοχές με περιορισμένη ορατότητα ή σε περιβάλλοντα με έντονο θόρυβο [51], [52].

3.3.6 Αισθητήρες Θερμοκρασίας

Οι αισθητήρες θερμοκρασίας επιτρέπουν την παρακολούθηση θερμικών συνθηκών, με εκτεταμένη χρήση σε πλήθος εφαρμογών, από τον βιομηχανικό αυτοματισμό και τις ιατρικές συσκευές έως τις μετεωρολογικές μετρήσεις και τις συσκευές ευρείας κατανάλωσης. Στη ρομποτική, παρέχουν κρίσιμες πληροφορίες για την κατάσταση του περιβάλλοντος ή για την υγεία των ίδιων των εξαρτημάτων του ρομπότ [52].

Υπάρχει μια μεγάλη ποικιλία αισθητήρων θερμοκρασίας, οι οποίοι λειτουργούν με διαφορετικές αρχές. Οι πιο διαδεδομένοι τύποι περιλαμβάνουν:

- **Θερμίστορ (Thermistors):** Αυτοί οι αισθητήρες είναι αντιστάσεις των οποίων η τιμή αλλάζει δραστικά ανάλογα με τη θερμοκρασία. Διακρίνονται σε δύο κύριες κατηγορίες: τα **NTC (Negative Temperature Coefficient)**, στα οποία η αντίσταση μειώνεται καθώς αυξάνεται η θερμοκρασία, και τα **PTC (Positive Temperature Coefficient)**, στα οποία η αντίσταση αυξάνεται. Τα θερμίστορ προσφέρουν μεγάλη ευαισθησία και γρήγορη απόκριση, αλλά η σχέση τους με τη θερμοκρασία είναι μη-γραμμική, κάτι που μπορεί να περιπλέξει την ανάγνωση των δεδομένων [50].
- **Θερμοηλεκτρικές Διόδους (Thermocouples):** Ένας θερμοζεύγος αποτελείται από δύο διαφορετικά μέταλλα που ενώνονται σε δύο σημεία, δημιουργώντας ένα κύκλωμα. Η διαφορά θερμοκρασίας μεταξύ των δύο σημείων (ζεύξεων) παράγει μια μικρή τάση (γνωστή ως φαινόμενο Seebeck), η οποία είναι ανάλογη της διαφοράς θερμοκρασίας. Τα θερμοστοιχεία είναι ανθεκτικά, λειτουργούν σε πολύ μεγάλα εύρη θερμοκρασίας και είναι σχετικά οικονομικά, ωστόσο έχουν χαμηλότερη ακρίβεια σε σύγκριση με άλλους τύπους αισθητήρων

και απαιτούν μετρήσεις άλλων αισθητήρων (όπως θερμίστορ ή RTD) για να χρησιμοποιήσουν ως αναφορά [50].

- **Ανιχνευτές Θερμοκρασίας Αντίστασης (RTDs - Resistance Temperature Detectors):** Οι ανιχνευτές θερμοκρασίας αντίστασης, όπως οι RTDs, βασίζονται στη μέτρηση της αντίστασης ενός μεταλλικού στοιχείου, συνήθως πλατίνα, η οποία αλλάζει όταν αλλάζει και η θερμοκρασία. Οι RTDs προσφέρουν εξαιρετική ακρίβεια, σταθερότητα και γραμμικότητα, γεγονός που τους καθιστά ιδανικούς για εφαρμογές ακριβείας. Το κύριο μειονέκτημά τους είναι το υψηλότερο κόστος και η πιο αργή απόκριση σε σχέση με τα θερμίστορ [50].

Οι αισθητήρες θερμοκρασίες μπορούν να χρησιμοποιηθούν είτε για την παρακολούθηση της θερμοκρασίας εξαρτημάτων του ίδιου του ρομπότ για την αποτροπή βλάβης σε περίπτωση υπερφόρτωσης, είτε για την παρακολούθηση θερμοκρασίας εξωτερικών εξαρτημάτων ή του περιβάλλοντος σε βιομηχανικές εφαρμογές [52]. Όπως αναφέρθηκε ήδη, το μικρό κόστος, ειδικά συγκεκριμένων τύπων αισθητήρων θερμοκρασίας, και οι σημαντικές εφαρμογές στις οποίες μπορούν να αξιοποιηθούν έχει ως αποτέλεσμα την ευρεία υιοθέτηση τους.

3.3.7 Sonar (Sound Navigation and Ranging)

Οι αισθητήρες υπερήχων, γνωστοί και ως sonar (Sound Navigation and Ranging), βασίζονται στη χρήση υπερηχητικών κυμάτων για την εκτίμηση αποστάσεων και την ανίχνευση αντικειμένων. Η βασική αρχή λειτουργίας τους είναι η εκπομπή παλμών υπερηχητικής συχνότητας (συνήθως 40 kHz) και η μέτρηση του χρόνου που απαιτείται για την αντανάκλαση του ηχητικού κύματος από ένα αντικείμενο και την επιστροφή του στον αισθητήρα (φαινόμενο *time-of-flight*) [52].

Οι αισθητήρες sonar παρουσιάζουν σημαντικά πλεονεκτήματα:

- **Χαμηλό κόστος και ευκολία ενσωμάτωσης:** Οι αισθητήρες υπερήχων είναι φθηνοί, μικρού μεγέθους και με ελάχιστες απαιτήσεις υπολογιστικής ισχύος, γεγονός που τους καθιστά κατάλληλους για ρομποτικές εφαρμογές χαμηλού κόστους [51].

- **Καλή λειτουργία σε σκοτεινά ή σκονισμένα περιβάλλοντα:** Σε αντίθεση με οπτικούς αισθητήρες, οι οποίοι επηρεάζονται από συνθήκες φωτισμού ή σωματίδια στον αέρα, οι αισθητήρες sonar μπορούν να παρέχουν αξιόπιστες μετρήσεις ακόμη και σε χαμηλή ορατότητα [53].

- **Λειτουργία σε υδάτινα περιβάλλοντα:** Ένα από τα βασικά πλεονεκτήματα του sonar είναι η αποτελεσματική διάδοση των ηχητικών κυμάτων στο υδάτινο περιβάλλον. Σε αντίθεση με τα ηλεκτρομαγνητικά κύματα που υφίστανται έντονη απορρόφηση και εξασθένηση στο νερό, οι υπέρηχοι μπορούν να ταξιδεύουν σε μεγάλες αποστάσεις, καθιστώντας το sonar την καταλληλότερη τεχνολογία για υποβρύχιες εφαρμογές, όπως η ναυτιλία, η εξερεύνηση βυθού και η πλοήγηση αυτόνομων υποβρύχιων οχημάτων [54].

Παρά τα πλεονεκτήματα τους όμως, οι αισθητήρες υπερήχων αντιμετωπίζουν αρκετούς περιορισμούς:

- **Περιορισμένη ανάλυση και ακρίβεια:** Σε σύγκριση με αισθητήρες λέιζερ ή οπτικής όρασης, οι μετρήσεις sonar παρουσιάζουν χαμηλότερη ανάλυση και μειωμένη δυνατότητα διάκρισης

λεπτομερειών [53].

- **Φαινόμενα "ηχητικών σκιών" και πολλαπλών ανακλάσεων:** Σε σύνθετα περιβάλλοντα, το σήμα μπορεί να απορροφηθεί, να διαχυθεί ή να ανακλαστεί πολλαπλά, οδηγώντας σε ψευδή δεδομένα ή απώλεια πληροφορίας [55], [56].

- **Περιορισμένο εύρος και ταχύτητα δειγματοληψίας:** Οι αισθητήρες υπερήχων έχουν μικρή αποτελεσματική εμβέλεια, ενώ η μέτρηση εξαρτάται από τον χρόνο διάδοσης του ήχου, περιορίζοντας τη συχνότητα ενημέρωσης [52].

Οι αισθητήρες sonar χρησιμοποιούνται ευρέως σε ρομποτικές πλατφόρμες για:

- **Ρομπότ εσωτερικών χώρων:** Χρησιμοποιούνται ευρέως σε μικρά κινητά ρομπότ και οικιακές ρομποτικές σκούπες για βασική αποφυγή εμποδίων και πλοήγηση [51], [57].

- **Αυτόνομα Υποβρύχια Οχήματα (AUVs):** Το sonar αποτελεί συχνά τη μόνη αξιόπιστη επιλογή αισθητήρα λόγω της απορρόφησης του φωτός στο νερό, με εφαρμογές σε χαρτογράφηση, πλοήγηση και αποφυγή εμποδίων [54], [58].

- **Ιπτάμενα μέσα:** Χρησιμοποιούνται για μέτρηση υψομέτρου και σταθεροποίηση πτήσης σε κοντινή απόσταση από το έδαφος, καθώς και αποφυγή συγκρούσεων στα ιπτάμενα μέσα [59].

- **Βιομηχανικά και ασφαλιστικά συστήματα:** Οι αισθητήρες υπερήχων χρησιμοποιούνται σε απλές και χαμηλού κόστους εφαρμογές, όπως η μέτρηση στάθμης υγρών ή η ανίχνευση παρουσίας αντικειμένων, λόγω της ανθεκτικότητας και της αξιοπιστίας τους σε βιομηχανικά περιβάλλοντα [56]. Παράλληλα, η ίδια τεχνολογία αποτέλεσε θεμέλιο για πιο προηγμένες ρομποτικές εφαρμογές, ιδίως στον τομέα του **SLAM (Simultaneous Localization and Mapping)**. Ερευνητικές εργασίες έδειξαν ότι οι μετρήσεις από sonar μπορούν να ενσωματωθούν σε πιθανολογικά μοντέλα εκτίμησης θέσης και κατασκευής χάρτη, καθιστώντας τους αισθητήρες αυτούς βασικό εργαλείο για την πρώιμη ανάπτυξη τεχνικών αυτόνομης πλοήγησης [56], [53].

Οι αισθητήρες υπερήχων παραμένουν μια από τις πιο διαδεδομένες και οικονομικές λύσεις για την εκτίμηση αποστάσεων σε ρομποτικές εφαρμογές, ιδιαίτερα σε περιβάλλοντα όπου δεν απαιτείται υψηλή ανάλυση. Παρά τους περιορισμούς τους, αποτελούν αναπόσπαστο κομμάτι πολλών συστημάτων, είτε ως βασική τεχνολογία είτε ως συμπληρωματικός αισθητήρας σε πολυαισθητηριακές διατάξεις [53], [55] .

3.3.8 Radar (Radio Detection and Ranging)

Το ραντάρ (Radio Detection and Ranging) βασίζεται στη μετάδοση και λήψη ηλεκτρομαγνητικών κυμάτων για την ανίχνευση αντικειμένων και την εκτίμηση της απόστασης, της σχετικής ταχύτητας και της γωνίας τους. Η τεχνολογία αυτή χρησιμοποιείται εκτενώς σε αεροναυτικές, ναυτικές και επίγειες εφαρμογές λόγω της αξιοπιστίας της σε μεγάλες αποστάσεις και σε δυσμενείς περιβαλλοντικές συνθήκες [60].

Πλεονεκτήματα

- **Αξιοπιστία σε κακές καιρικές συνθήκες**: Σε αντίθεση με τους οπτικούς αισθητήρες ή το LiDAR, το radar μπορεί να λειτουργήσει με υψηλή ακρίβεια σε βροχή, ομίχλη, χιόνι ή σκόνη, γεγονός που το καθιστά ιδανικό για εξωτερικά περιβάλλοντα [61].

- **Ευέλικτες αποστάσεις ανίχνευσης**: Ένας αισθητήρας radar, ανάλογα με το μέγεθος και τα χαρακτηριστικά του, μπορεί να ανιχνεύει στόχους από μικρές αποστάσεις όπως λίγα μέτρα, έως και στόχους που βρίσκονται σε κοντινούς πλανήτες [60].

- **Εκτίμηση ταχύτητας**: Μέσω του φαινομένου Doppler, το οποίο περιγράφει τη μεταβολή στη συχνότητα ενός κύματος όταν η πηγή ή ο παρατηρητής κινούνται., το radar μπορεί να υπολογίσει την ταχύτητα κίνησης των αντικειμένων, κάτι κρίσιμο για δυναμικά περιβάλλοντα [62].

- **Διείσδυση μέσω εμποδίων:** Ορισμένα ραδιοκύματα μπορούν να διεισδύσουν σε μη μεταλλικά εμπόδια (όπως φυλλωσιές, πλαστικά ή ακόμη και λεπτούς τοίχους), επιτρέποντας την ανίχνευση αντικειμένων που θα ήταν αόρατα σε οπτικούς αισθητήρες [60].

Μειονεκτήματα

- **Περιορισμένη ανάλυση**: Σε σύγκριση με LiDAR ή κάμερες, τα ραντάρ έχουν μικρότερη γωνιακή ανάλυση, με αποτέλεσμα χαμηλότερη ακρίβεια στον καθορισμό του σχήματος και της θέσης των αντικειμένων [61].

- **Πολυπλοκότητα επεξεργασίας σήματος**: Η εξαγωγή αξιόπιστων πληροφοριών από τα σήματα του radar είναι μια ιδιαίτερα απαιτητική διαδικασία, καθώς το σήμα περιέχει θόρυβο, παρεμβολές και ανακλάσεις από πολλαπλές επιφάνειες. Για τον λόγο αυτό χρησιμοποιούνται προηγμένοι αλγόριθμοι φιλτραρίσματος και επεξεργασίας σήματος [62]. Ενδεικτικά:

    - **Matched Filtering (Συνελικτικό φίλτρο προσαρμογής):** Χρησιμοποιείται για τη μεγιστοποίηση του λόγου σήματος προς θόρυβο (SNR) και την ακριβή εκτίμηση της απόστασης.

    - **Pulse Compression:** Επιτρέπει στο radar να εκπέμπει μεγαλύτερης διάρκειας παλμούς (για μεγαλύτερη ισχύ) και ταυτόχρονα να διατηρεί υψηλή διακριτική ικανότητα.

    - **FFT-based Doppler Processing:** Χρησιμοποιεί τον **Fast Fourier Transform (FFT)** για να εξάγει τη συχνότητα Doppler, δηλαδή την ταχύτητα στόχων σε πραγματικό χρόνο.

    - **CFAR (Constant False Alarm Rate):** Αλγόριθμος που προσαρμόζει δυναμικά το κατώφλι ανίχνευσης ώστε να μειώνονται τα ψευδώς θετικά από θόρυβο ή περιβαλλοντικές μεταβολές.

    - **Kalman Filters και Particle Filters:** Χρησιμοποιούνται στην παρακολούθηση (tracking) για την εκτίμηση της θέσης και της ταχύτητας κινούμενων στόχων με συνδυασμό πολλαπλών μετρήσεων.

- **Παρεμβολές**: Σε πυκνά αστικά περιβάλλοντα ή με την αυξανόμενη χρήση radar σε αυτόνομα οχήματα, υπάρχει αυξημένος κίνδυνος παρεμβολών μεταξύ συστημάτων, είτε επειδή έγινε ανάκλαση του σήματος από μη-σημαντικό αντικείμενο, είτε επειδή έγινε υπερκάλυψη του σήματος από άλλη πηγή [63].

Εφαρμογές

- **Αυτοκινητοβιομηχανία**: Το radar έχει καταστεί βασικό στοιχείο των συστημάτων Advanced Driver Assistance Systems (ADAS), υποστηρίζοντας λειτουργίες όπως βοήθεια στο πάρκινγκ, adaptive cruise control, αυτόματη πέδηση και ανίχνευση τυφλών σημείων, ενισχύοντας την ασφαλή οδήγηση [61].

- **Αυτόνομα οχήματα**: Χρησιμοποιείται σε συνδυασμό με LiDAR και κάμερες σε πολυαισθητηριακά συστήματα αντίληψης, παρέχοντας αξιόπιστα δεδομένα για την πλοήγηση [61].

- **Αεροναυτικές και ναυτικές εφαρμογές**: Παραμένει θεμελιώδης τεχνολογία για πλοήγηση και αποφυγή συγκρούσεων σε αεροσκάφη και πλοία [60].

Οι αισθητήρες ραντάρ αποτελούν κρίσιμο κομμάτι της σύγχρονης τεχνολογίας αντίληψης, ιδιαίτερα σε εφαρμογές όπου οι καιρικές ή φωτιστικές συνθήκες καθιστούν αναξιόπιστους άλλους αισθητήρες. Παρότι υστερούν σε γωνιακή ανάλυση σε σχέση με LiDAR ή κάμερες, η ικανότητα τους να παρέχουν αξιόπιστες μετρήσεις αποστάσεων και ταχυτήτων σε δύσκολα περιβάλλοντα τους καθιστά αναντικατάστατους σε συστήματα ADAS, αυτόνομα οχήματα και κρίσιμες μεταφορικές εφαρμογές.

3.4 Συνδυασμός Αισθητήρων (Sensor Fusion)

Η αισθητηριακή σύντηξη (sensor fusion) είναι μια θεμελιώδης διαδικασία στη ρομποτική, με στόχο τη βέλτιστη εκτίμηση της κατάστασης ενός ρομπότ (θέση, ταχύτητα, προσανατολισμός) μέσω του συνδυασμού δεδομένων από ετερογενείς αισθητήρες. Η κεντρική ιδέα είναι η στοχαστική μοντελοποίηση της κίνησης και των σφαλμάτων, ώστε να επιτυγχάνεται μια πιο ακριβής και αξιόπιστη εκτίμηση από ό,τι θα ήταν δυνατό με κάθε αισθητήρα ξεχωριστά [53], [64].

3.4.1 Φίλτρα Kalman και Παραλλαγές

Τα φίλτρα Kalman (KF) αποτελούν ένα κλασικό πιθανολογικό πλαίσιο για την αναδρομική εκτίμηση της κατάστασης ενός συστήματος. Η λειτουργία τους βασίζεται σε δύο διακριτά βήματα: την πρόβλεψη (χρησιμοποιώντας ένα μοντέλο κίνησης) και τη διόρθωση (χρησιμοποιώντας τις μετρήσεις των αισθητήρων). Το κλασικό φίλτρο Kalman παρέχει τη βέλτιστη λύση για γραμμικά συστήματα με Γκαουσιανό θόρυβο [65], [64].

Για την αντιμετώπιση μη γραμμικών προβλημάτων, αναπτύχθηκαν παραλλαγές όπως το Extended Kalman Filter (EKF). Αυτό το φίλτρο γραμμικοποιεί το μη γραμμικό μοντέλο γύρω από την τρέχουσα εκτίμηση, χρησιμοποιώντας Ιακωβιανούς πίνακες [53]. Αν και είναι ευρέως διαδεδομένο σε εφαρμογές πραγματικού χρόνου, όπως η ολοκλήρωση GNSS/INS και VIO, μπορεί να είναι ευαίσθητο σε έντονη μη-γραμμικότητα και σφάλματα αρχικοποίησης [43].

Μια πιο προηγμένη εναλλακτική είναι το Unscented Kalman Filter (UKF), το οποίο αντικαθιστά τη γραμμικοποίηση με έναν "unscented transform". Αυτή η προσέγγιση χρησιμοποιεί ένα σύνολο σημείων δειγματοληψίας (sigma points) για να διατηρήσει καλύτερα τις μη γραμμικές ροπές, προσφέροντας συχνά μεγαλύτερη ακρίβεια σε πολύπλοκα προβλήματα, χωρίς να απαιτεί τον υπολογισμό Ιακωβιανών [65], [64]. Επιπρόσθετα, τα συμπληρωματικά φίλτρα (complementary filters), ειδικά οι μη γραμμικές τους παραλλαγές, προσφέρουν μια απλή και αποδοτική λύση για τη συγχώνευση δεδομένων από γυροσκόπια και επιταχυνσιόμετρα με χαμηλό υπολογιστικό κόστος, βελτιώνοντας τη σταθερότητα του προσανατολισμού σε ενσωματωμένες πλατφόρμες [67].

3.4.2 Σωματιδιακά Φίλτρα (Particle Filters)

Τα **σωματιδιακά φίλτρα** (particle filters) προσφέρουν μια ισχυρή λύση για μη γραμμικά και μη Γκαουσιανά προβλήματα, τα οποία δεν μπορούν να επιλυθούν αποτελεσματικά με τα φίλτρα Kalman. Η προσέγγισή τους βασίζεται στη δειγματοληψία της posterior κατανομής μέσω ενός συνόλου από "σωματίδια" με διαφορετικά βάρη. Χρησιμοποιούνται κυρίως για τον καθολικό εντοπισμό (global localization), όπως στη μέθοδο Monte-Carlo Localization (MCL), και για την επίλυση προβλημάτων SLAM σε διακριτούς χάρτες [53]. Βασικό τους πλεονέκτημα είναι η ικανότητά τους να αναπαριστούν πολυτροπικές κατανομές, καθιστώντας τα ανθεκτικά σε ασυνέχειες, ωστόσο το υπολογιστικό τους κόστος είναι υψηλό και απαιτούν αναδειγματοληψία για την αποφυγή του εκφυλισμού [68].

3.4.3 Σχήματα Εξομάλυνσης και Γραφικές Μέθοδοι

Ενώ τα φίλτρα εκτιμούν την κατάσταση μόνο προς το μέλλον, τα σχήματα εξομάλυνσης (smoothers) εκμεταλλεύονται ένα "παράθυρο" δεδομένων από το παρελθόν και το μέλλον για να βελτιστοποιήσουν ολιστικά την εκτίμηση. Η μοντελοποίηση αυτών των προβλημάτων με factor graphs μετατρέπει το πρόβλημα σε μία βελτιστοποίηση ελαχίστων τετραγώνων με αραιή δομή, επιτρέποντας την ενσωμάτωση πολλών αισθητήρων και περιορισμών (π.χ. αναγνώριση κλειστού βρόχου / loop closures). Εμβληματικά παραδείγματα αποτελούν το iSAM2 και η βιβλιοθήκη GTSAM [69], [70]. Τα smoothers προσφέρουν αυξημένη ακρίβεια και συνέπεια, αλλά απαιτούν περισσότερη μνήμη και υπολογιστικούς πόρους σε σχέση με τα φίλτρα.

3.4.4 Εφαρμογές Σύντηξης Ανά Τεχνολογία Αισθητήρων

- **Οπτικο-Αδρανειακή Σύντηξη (VIO):** Συνδυάζει δεδομένα από IMU και κάμερες. Οι φιλτραριστικές προσεγγίσεις (π.χ. MSCKF) εκτιμούν την κατάσταση με χαμηλό κόστος [71], ενώ οι βελτιστοποιητικές/γραφικές μέθοδοι (π.χ. OKVIS, VINS-Mono) προσφέρουν αυξημένη ακρίβεια και δυνατότητες αναγνώρισης βρόχων [72], [73].
- **LiDAR–Inertial Fusion:** Συνδυάζει την υψηλή συχνότητα και τη βραχυπρόθεσμη σταθερότητα του IMU με τις ακριβείς γεωμετρικές μετρήσεις του LiDAR, μειώνοντας την παραμόρφωση της κίνησης και το drift [74].

- **GNSS/INS Ολοκλήρωση:** Ένα κλασικό πεδίο όπου το INS "γεφυρώνει" τα κενά των μετρήσεων του GNSS, ενώ το GNSS περιορίζει το μακροπρόθεσμο drift του INS. Οι προσεγγίσεις είναι είτε loosely coupled (απλή σύζευξη) είτε tightly coupled (σύζευξη των ανεπεξέργαστων δεδομένων για μεγαλύτερη ακρίβεια) [43].
- **Radar–Vision/LiDAR Fusion:** Συνδυάζει την ανθεκτικότητα του Radar σε καιρικές συνθήκες με την υψηλή ανάλυση των καμερών ή του LiDAR, βελτιώνοντας την αντίληψη σε συστήματα αυτόνομης οδήγησης [61].

Η επιλογή της κατάλληλης τεχνικής εξαρτάται από τις απαιτήσεις της εφαρμογής, όπως οι διαθέσιμοι πόροι, το περιβάλλον λειτουργίας και η απαιτούμενη ακρίβεια.

3.5 Μικροελεγκτές και Ενσωματωμένα Συστήματα στη Ρομποτική

Οι μικροελεγκτές αποτελούν την υπολογιστική «καρδιά» των περισσότερων ρομποτικών πλατφορμών, συνδυάζοντας σε ένα ολοκληρωμένο κύκλωμα επεξεργαστή, μνήμη και πλήθος περιφερειακών διεπαφών. Σε αντίθεση με τους μικροεπεξεργαστές γενικού σκοπού, έχουν σχεδιαστεί για εφαρμογές πραγματικού χρόνου, με χαμηλή κατανάλωση και δυνατότητα άμεσης πρόσβασης στις γραμμές εισόδου–εξόδου (GPIO), κάτι που επιτρέπει γρήγορη αλληλεπίδραση με αισθητήρες και ενεργοποιητές [76].

Η τυπική αρχιτεκτονική ενός μικροελεγκτή περιλαμβάνει πυρήνα CPU (ARM Cortex-M, AVR, RISC-V), μνήμη προγράμματος (Flash) και δεδομένων (SRAM) στο ίδιο chip, καθώς και ενσωματωμένα περιφερειακά, όπως χρονομετρητές, μετατροπείς αναλογικού σε ψηφιακό (ADC) και διαύλους επικοινωνίας. Οι πιο διαδεδομένες διεπαφές είναι:

- **I²C (Inter-Integrated Circuit)**: Δίαυλος δύο γραμμών (SDA, SCL) που υποστηρίζει επικοινωνία master–slave μεταξύ πολλαπλών συσκευών με διευθυνσιοδότηση. Είναι ιδανικός για σύνδεση αισθητήρων χαμηλού ρυθμού μετάδοσης [77].

- **SPI (Serial Peripheral Interface)**: Δίαυλος τεσσάρων γραμμών (MISO, MOSI, SCLK, CS) που προσφέρει υψηλότερους ρυθμούς μετάδοσης σε σχέση με το I²C, κατάλληλος για οθόνες, μνήμες και γρήγορη δειγματοληψία δεδομένων [78].

- **UART (Universal Asynchronous Receiver-Transmitter)**: Σειριακή ασύγχρονη επικοινωνία (TX, RX) χωρίς ρολόι, ευρέως διαδεδομένη για απλή σημείο-προς-σημείο σύνδεση, debugging ή σύνδεση με modules [79].

- **CAN (Controller Area Network)**: Δίαυλος δύο καλωδίων που σχεδιάστηκε για το περιβάλλον της αυτοκινητοβιομηχανίας, προσφέροντας αντοχή σε θόρυβο, ενσωματωμένη ανίχνευση σφαλμάτων και δυνατότητα επικοινωνίας πολλών κόμβων σε πραγματικό χρόνο [76].

Η επιλογή μικροελεγκτή σε ρομποτικά έργα καθορίζεται από απαιτήσεις υπολογιστικής ισχύος, πλήθος I/O, κόστος, κατανάλωση ενέργειας και υποστήριξη πρωτοκόλλων. Οι STM32 (ARM Cortex-M) παρέχουν υψηλή επεξεργαστική ισχύ και πλούσιο οικοσύστημα ανάπτυξης, οι ATmega/Arduino απλότητα και χαμηλό κόστος, οι PIC σταθερότητα σε βιομηχανικές εφαρμογές, ενώ

οι ESP32 ενσωματώνουν Wi-Fi/Bluetooth για κινητές πλατφόρμες [79]. Σε πιο σύνθετα ρομπότ, η αρχιτεκτονική συχνά διαχωρίζει τον έλεγχο χαμηλού επιπέδου (microcontroller) από πιο βαριές διεργασίες (SLAM, όραση υπολογιστή) που εκτελούνται σε single-board computer [77].

Ο προγραμματισμός τους γίνεται κυρίως σε C ή C++, που επιτρέπουν άμεση πρόσβαση στο υλικό, έλεγχο μνήμης και προβλέψιμη εκτέλεση. Οι κατασκευαστές προσφέρουν IDE όπως STM32CubeIDE (ST), MPLAB X (Microchip), Atmel Studio ή Arduino IDE (AVR), Espressif IDF (ESP32). Οι πλατφόρμες τύπου Arduino παρέχουν βιβλιοθήκες υψηλού επιπέδου, διευκολύνοντας ταχεία ανάπτυξη αλλά συχνά με τίμημα τη λεπτομερή βελτιστοποίηση. Σε απαιτητικές εφαρμογές πραγματικού χρόνου, η ανάπτυξη χαμηλού επιπέδου με χειρισμό καταχωρητών και χρήση RTOS (π.χ. FreeRTOS, Zephyr) προσφέρει καλύτερη προβλεψιμότητα και multitasking [76], [78].

Οι γλώσσες υψηλότερου επιπέδου, όπως η MicroPython ή η CircuitPython, έχουν αποκτήσει δημοτικότητα σε εκπαιδευτικά και ταχύ-πρωτοτυπικά έργα, αν και μειώνουν την απόδοση λόγω της ερμηνευόμενης φύσης τους. Η Rust εμφανίζει αυξανόμενο ενδιαφέρον σε embedded εφαρμογές, χάρη στην ασφάλεια μνήμης και την αποδοτικότητα, αλλά το οικοσύστημά της παραμένει σε εξέλιξη .

Η ανάπτυξη λογισμικού σε μικροελεγκτές απαιτεί καλή κατανόηση διακοπών (interrupts), συγχρονισμού, ελέγχου κινητήρων και ταχύτατης δειγματοληψίας αισθητήρων. Η ακρίβεια στη διαχείριση χρόνου και η αξιόπιστη επικοινωνία με άλλα modules είναι κρίσιμες για τη λειτουργία ενός ρομπότ σε πραγματικό χρόνο [76]. Η ενσωμάτωση των μικροελεγκτών στη ρομποτική καθιστά δυνατή την αυτόνομη λειτουργία με χαμηλή καθυστέρηση απόκρισης, ενώ η κλιμακωτή σχεδίαση επιτρέπει επέκταση σε πιο σύνθετες λειτουργίες, όπως συνεργατικά ρομπότ ή βιομηχανική αυτοματοποίηση.

Στο πλαίσιο εκπαιδευτικών και ερευνητικών εφαρμογών, ιδιαίτερα σημαντική είναι η πλατφόρμα BBC micro:bit, η οποία επιλέχθηκε και για την υλοποίηση μέρους του πρακτικού τμήματος της παρούσας πτυχιακής εργασίας. Το BBC micro:bit είναι ένας μικροελεγκτής σχεδιασμένος για εκμάθηση προγραμματισμού και φυσικής υπολογιστών. Το micro:bit ενσωματώνει επεξεργαστή ARM Cortex-M0, επιταχυνσιόμετρο, μαγνητόμετρο, Bluetooth Low Energy (BLE), πλήκτρα, LED matrix 5×5 και πλήθος ακροδεκτών GPIO [80]. Χάρη στον ενσωματωμένο ασύρματο σύνδεσμο BLE, υποστηρίζει άμεση επικοινωνία με φορητές συσκευές και επιτρέπει τη δημιουργία εφαρμογών απομακρυσμένου ελέγχου και συλλογής δεδομένων, κάτι που το καθιστά ιδιαίτερα κατάλληλο για μικρά ρομπότ και έργα STEM.

Ο προγραμματισμός του micro:bit υποστηρίζει πολλαπλά περιβάλλοντα:

- **Microsoft MakeCode** (block-based, JavaScript, MicroPython) για ταχεία ανάπτυξη και εκπαιδευτικά έργα.

- **MicroPython**, που επιτρέπει scripting υψηλού επιπέδου με διαχείριση εισόδων/εξόδων, Bluetooth και αισθητήρων.

- **C/C++ SDK** για πρόσβαση χαμηλού επιπέδου, βελτιστοποίηση επιδόσεων και ενσωμάτωση σε πιο απαιτητικά ρομποτικά έργα.

Η ευκολία χρήσης, η χαμηλή κατανάλωση και τιμή, καθώς και το πλούσιο οικοσύστημα παραδειγμάτων καθιστούν το BBC micro:bit δημοφιλή επιλογή για ταχύ πρωτότυπα ρομποτικών πλατφορμών, ιδιαίτερα όταν το ζητούμενο είναι συνδυασμός **εκπαιδευτικής αξίας** και **πραγματικής λειτουργικότητας**. Επιπλέον, η ενσωματωμένη υποστήριξη I²C, SPI και UART επιτρέπει την επέκταση με εξωτερικούς αισθητήρες ή κινητήρες, ενώ η διεπαφή BLE διευκολύνει τη διασύνδεση με εφαρμογές κινητών συσκευών, κάτι κρίσιμο για πειράματα σε τηλεχειριζόμενα οχήματα ή ρομπότ με δυνατότητες συλλογής δεδομένων [80].

3.6 Εκπαιδευτικά Ρομπότ και ο Ρόλος τους στην Εκμάθηση

Η ραγδαία διάδοση των εκπαιδευτικών ρομπότ έχει μεταβάλει τον τρόπο διδασκαλίας των επιστημών, της τεχνολογίας, της μηχανικής και των μαθηματικών (STEM). Τα ρομποτικά κιτ ενσωματώνουν αισθητήρες, ενεργοποιητές και μικροελεγκτές, προσφέροντας στους μαθητές τη δυνατότητα να αλληλεπιδρούν με φυσικά συστήματα, να κατανοούν τη λογική του ελέγχου και να αναπτύσσουν δεξιότητες προγραμματισμού μέσα από πρακτική εμπειρία [81]. Παράλληλα, καλλιεργούν δημιουργικότητα, ομαδικότητα και κριτική σκέψη, καθώς οι μαθητές καλούνται να επιλύσουν πραγματικά προβλήματα σχεδιάζοντας και βελτιστοποιώντας το δικό τους ρομποτικό έργο.

Η εκπαιδευτική ρομποτική κινείται σε δύο βασικούς άξονες:

1. **Μηχανισμοί και φυσική κατανόηση** – Οι μαθητές αντιλαμβάνονται έννοιες όπως κινηματική, τριβή, ανατροφοδότηση αισθητήρων και αλγοριθμικό έλεγχο.

2. **Προγραμματισμός και αλγοριθμική σκέψη** – Μέσα από γλώσσες κατάλληλες για αρχάριους (block-based) ή προχωρημένους (Python, C++), ο χρήστης μαθαίνει δομές ελέγχου, παραμετροποίηση και λογική σύνδεσης αισθητήρων/ενεργοποιητών.

Τα σύγχρονα κιτ εκπαιδευτικής ρομποτικής βασίζονται συχνά σε ανοιχτά οικοσυστήματα μικροελεγκτών (Arduino, ESP32, micro:bit), επιτρέποντας επεκτάσεις και προσαρμογές. Το BBC micro:bit –λόγω της απλότητας, της υποστήριξης Bluetooth Low Energy, και των ενσωματωμένων αισθητήρων του– έχει καταστεί πρότυπο εκπαιδευτικό εργαλείο διεθνώς [80].

3.6.1 Το Παράδειγμα του Yahboom Tiny:bit Plus

Το **Yahboom Tiny:bit Plus**, που στην παρούσα πτυχιακή εργασία επιλέχθηκε για την πρακτική υλοποίηση ενός πομπού που θα δέχεται εντολές μέσω BLE και θα τις εκτελεί, είναι ένα ρομποτικό κιτ που βασίζεται στο BBC micro:bit, ενσωματώνοντας διπλούς κινητήρες, αισθητήρες υπερύθρων, φωτεινότητας, καθώς και δυνατότητες Bluetooth για τηλεχειρισμό και μεταφορά δεδομένων. Παρέχει μία άμεση «γέφυρα» ανάμεσα στη θεωρία και την πράξη, καθώς οι χρήστες μπορούν:

- να πειραματιστούν με βασικό έλεγχο κίνησης, αποφυγή εμποδίων και γραμμοπαρακολούθηση·

- να δημιουργήσουν δικές τους εφαρμογές αξιοποιώντας MakeCode (block-based ή JavaScript) και MicroPython, προχωρώντας μέχρι χαμηλού επιπέδου έλεγχο μέσω C/C++;

- να ενσωματώσουν επικοινωνία BLE με φορητές συσκευές, δημιουργώντας απλά περιβάλλοντα τηλεχειρισμού ή συλλογής τηλεμετρίας·

- να κατανοήσουν βασικά πρωτόκολλα διασύνδεσης όπως I²C (σειριακή επικοινωνία με αισθητήρες), SPI (ταχύτατη επικοινωνία για modules επέκτασης) και UART (απλή ασύγχρονη σειριακή ανταλλαγή δεδομένων).

Η φιλοσοφία «plug-and-play» του Tiny:bit Plus μειώνει δραστικά τον χρόνο εγκατάστασης, επιτρέποντας στους μαθητές να επικεντρωθούν στον πειραματισμό και στην ανάπτυξη λογικής ελέγχου. Επιπλέον, η συμβατότητα με πληθώρα μαθημάτων και σεναρίων STEM, από απλές εργασίες πλοήγησης έως βασικά project αισθητηριακής ολοκλήρωσης (sensor fusion), το καθιστά ένα ισχυρό εργαλείο για διαθεματική μάθηση [82].

Έρευνες επισημαίνουν ότι η εκπαιδευτική ρομποτική ενισχύει την **εμπλοκή** (engagement) και την **ενεργητική μάθηση**, προσφέροντας αυξημένα κίνητρα συμμετοχής και καλύτερη κατανόηση σύνθετων εννοιών [83]. Με πλατφόρμες όπως το Tiny:bit Plus, οι μαθητές μπορούν να μετατρέψουν αφηρημένες θεωρητικές έννοιες σε πειράματα, επιτυγχάνοντας ουσιαστικότερη κατανόηση.

3.7 Επίλογος

Στο παρόν κεφάλαιο αναλύθηκαν οι θεμελιώδεις αρχές της Ρομποτικής που διέπουν την αυτόνομη πλοήγηση. Παρουσιάστηκαν διεξοδικά οι τρεις πυλώνες της πλοήγησης: ο εντοπισμός θέσης (Localization), η χαρτογράφηση (Mapping) και ο σχεδιασμός κίνησης (Path Planning), καθώς και οι αλγόριθμοι που τους υποστηρίζουν.

Ιδιαίτερη έμφαση δόθηκε στον κρίσιμο ρόλο των αισθητήρων, οι οποίοι αποτελούν τις "αισθήσεις" του ρομπότ. Εξετάστηκε η λειτουργία και η χρησιμότητα πληθώρας αισθητηριακών συστημάτων, από οπτικούς αισθητήρες και LiDAR μέχρι αδρανειακά συστήματα (IMU) και αισθητήρες υπερήχων, αναδεικνύοντας πώς η σύντηξη των δεδομένων τους επιτρέπει στο ρομπότ να αντιλαμβάνεται το περιβάλλον του με ακρίβεια.

Ωστόσο, η μεγαλύτερη πρόκληση για ένα αυτόνομο σύστημα προκύπτει όταν καλείται να λειτουργήσει σε άγνωστα περιβάλλοντα, χωρίς προϋπάρχοντα χάρτη. Αυτό το πρόβλημα αντιμετωπίζεται από τη μεθοδολογία του Ταυτόχρονου Εντοπισμού και Χαρτογράφησης, η οποία αποτελεί τον συνδετικό κρίκο μεταξύ της Ρομποτικής και της Επαυξημένης Πραγματικότητας.

Κεφάλαιο 4ο: Simultaneous Localization and Mapping

4.1 Εισαγωγή

Το Simultaneous Localization and Mapping (SLAM) αποτελεί ένα από τα πιο σημαντικά πεδία έρευνας στη ρομποτική και στις εφαρμογές Επαυξημένης Πραγματικότητας (AR). Αφορά την ικανότητα ενός κινητού πράκτορα, όπως ένα ρομπότ ή μια φορητή συσκευή, να εκτιμά ταυτόχρονα τη δική του θέση και τον χάρτη ενός άγνωστου περιβάλλοντος [84]. Η αυτόνομη πλοήγηση, η αποφυγή εμποδίων, καθώς και η ρεαλιστική προβολή ψηφιακού περιεχομένου, απαιτούν την ακριβή εκτίμηση της θέσης του ρομπότ και τη δημιουργία ενός αξιόπιστου χάρτη του περιβάλλοντος.

4.2 Κλασικές Προσεγγίσεις SLAM

Οι πρώτες επιτυχημένες μέθοδοι SLAM βασίστηκαν σε πιθανολογικά φίλτρα για την εκτίμηση της κατάστασης του ρομπότ και των χαρακτηριστικών του περιβάλλοντος.

EKF-SLAM: Η πρώτη ευρέως διαδεδομένη προσέγγιση SLAM χρησιμοποιούσε το Extended Kalman Filter (EKF) για την από κοινού εκτίμηση του διανύσματος κατάστασης του ρομπότ και των landmarks. Παρά την αποτελεσματικότητά του, ο EKF-SLAM έχει το μειονέκτημα του υπολογιστικού κόστους, το οποίο αυξάνεται τετραγωνικά με τον αριθμό των landmarks, καθιστώντας τον μη επεκτάσιμο σε μεγάλα περιβάλλοντα [85], [53].

Particle Filter SLAM (FastSLAM): Για την αντιμετώπιση του προβλήματος επεκτασιμότητας, αναπτύχθηκε ο αλγόριθμος FastSLAM. Αυτός ο αλγόριθμος διαχωρίζει το πρόβλημα του SLAM, χρησιμοποιώντας ένα σωματιδιακό φίλτρο (particle filter) για την εκτίμηση της πορείας του ρομπότ και ένα ξεχωριστό EKF για την εκτίμηση των landmarks σε κάθε σωματίδιο. Για την επεξεργασία μεγαλύτερων χαρτών, ο αλγόριθμος FastSLAM επιτυγχάνει υψηλή υπολογιστική απόδοση, καθιστώντας τον ως μία κατάλληλη επιλογή [86].

4.3 Γραφικές Μέθοδοι (Graph-SLAM)

Μια εναλλακτική προσέγγιση είναι η διατύπωση του προβλήματος SLAM ως γραφικής βελτιστοποίησης.

Graph-SLAM: Σε αυτή τη μέθοδο, το πρόβλημα μοντελοποιείται ως ένας γράφος, όπου οι κόμβοι (nodes) αναπαριστούν τις καταστάσεις του ρομπότ και των landmarks, ενώ οι ακμές (edges) αναπαριστούν τις μετρήσεις των αισθητήρων και τους περιορισμούς κίνησης. Η λύση του SLAM επιτυγχάνεται μέσω της βελτιστοποίησης του γράφου, ελαχιστοποιώντας τα σφάλματα των μετρήσεων με τη μέθοδο των ελαχίστων τετραγώνων (least-squares) [87].

iSAM και iSAM2: Βασισμένοι στη λογική του Graph-SLAM, οι αλγόριθμοι Incremental Smoothing and Mapping (iSAM) επιτρέπουν την αναδρομική και αποδοτική ενημέρωση των εκτιμήσεων σε πραγματικό χρόνο. Η νεότερη έκδοση, iSAM2, είναι ακόμα πιο αποδοτική, εκμεταλλευόμενη την αραιή δομή του γράφου για να επιτύχει online βελτιστοποίηση, καθιστώντας την κατάλληλη για εφαρμογές σε ενσωματωμένα συστήματα [69].

4.4 SLAM Ανά Τύπο Αισθητήρα

Η επιλογή του αισθητήρα είναι κρίσιμη για την υλοποίηση ενός συστήματος SLAM, καθώς καθορίζει τις δυνατότητες και τους περιορισμούς του.

Visual SLAM (vSLAM): Χρησιμοποιεί κάμερες (μονοοπτικές, στερεοσκοπικές, RGB-D) ως κύριες πηγές δεδομένων. Οι αλγόριθμοι vSLAM μπορούν να χωριστούν σε δύο κατηγορίες:

Feature-based: Χρησιμοποιούν χαρακτηριστικά, όπως οι αλγόριθμοι ORB (ORB-SLAM), για την ανίχνευση και παρακολούθηση σημείων [88].

Direct methods: Χρησιμοποιούν την ένταση των pixels απευθείας, χωρίς την εξαγωγή χαρακτηριστικών, όπως στην περίπτωση του DSO (Direct Sparse Odometry) [89].

LiDAR-SLAM: Αξιοποιεί δεδομένα από αισθητήρες λέιζερ για την κατασκευή ακριβών χαρτών. Τεχνικές όπως το Iterative Closest Point (ICP) και το Normal Distributions Transform (NDT) χρησιμοποιούνται για την ευθυγράμμιση των νεφών σημείων. Ο αλγόριθμος LOAM (Lidar Odometry and Mapping) θεωρείται σημείο αναφοράς για την ικανότητά του να λειτουργεί αποδοτικά και σε πραγματικό χρόνο [90].

Radar-SLAM: Μια νεότερη κατεύθυνση, όπου το radar χρησιμοποιείται για την κατασκευή χαρτών ανθεκτικών σε δύσκολες καιρικές συνθήκες. Η εφαρμογή του είναι ιδιαίτερα σημαντική στην αυτοκινητοβιομηχανία [91].

Sonar-SLAM: Το Sonar-SLAM χρησιμοποιείται κυρίως στην υποβρύχια ρομποτική (AUVs), όπου ο αισθητήρας sonar αποτελεί συχνά τη μοναδική αξιόπιστη επιλογή λόγω της μεγάλης απορρόφησης του φωτός στο νερό [56].

4.5 Σύγχρονες Κατευθύνσεις και Εφαρμογές

Ο τομέας του SLAM συνεχίζει να εξελίσσεται, ενσωματώνοντας πλέον νέες προσεγγίσεις από τον χώρο της μηχανικής μάθησης και της τεχνητής νοημοσύνης.

Semantic SLAM και Deep Learning: Ο συνδυασμός του SLAM με αλγορίθμους βαθιάς μάθησης επιτρέπει την αναγνώριση αντικειμένων και την κατανόηση σκηνών, παρέχοντας εμπλουτισμένους χάρτες με σημασιολογικές πληροφορίες [84].

Multi-Robot SLAM: Η συνεργασία μεταξύ πολλαπλών ρομπότ, τα οποία ανταλλάσσουν δεδομένα με στόχο την κατασκευή ενός κοινού χάρτη, αποτελεί ένα κρίσιμο πεδίο έρευνας για εφαρμογές σε logistics και εξερεύνηση (Multi-Robot SLAM) [92].

Real-time SLAM σε AR/VR: Αλγόριθμοι όπως το PTAM (Parallel Tracking and Mapping) αποτέλεσαν ορόσημο για τον real-time visual SLAM σε AR εφαρμογές [93]. Σήμερα, τα SDKs ARKit και ARCore ενσωματώνουν state-of-the-art SLAM αλγορίθμους για ακριβές tracking και rendering σε φορητές συσκευές.

4.6 Επίλογος

Στο παρόν κεφάλαιο αναλύθηκε η μεθοδολογία του Ταυτόχρονου Εντοπισμού και Χαρτογράφησης (SLAM), η οποία αποτελεί τον πυρήνα της σύγχρονης αυτόνομης ρομποτικής. Εξετάστηκε το πώς ένας πράκτορας μπορεί να χαρτογραφεί ένα άγνωστο περιβάλλον ενώ ταυτόχρονα υπολογίζει τη θέση του εντός αυτού, επιλύοντας ένα πρόβλημα που μοιάζει με τον γρίφο «της κότας και του αυγού».

Ιδιαίτερη αναφορά έγινε στις διάφορες προσεγγίσεις του SLAM, και συγκεκριμένα στο Visual SLAM, το οποίο αξιοποιεί δεδομένα κάμερας. Η κατανόηση αυτών των μηχανισμών είναι θεμελιώδης για την παρούσα εργασία, καθώς τα σύγχρονα συστήματα Επαυξημένης Πραγματικότητας βασίζονται σε εξελιγμένους αλγορίθμους Visual-Inertial Odometry και SLAM για να επιτύχουν τη σταθερή τοποθέτηση των ψηφιακών αντικειμένων στον φυσικό χώρο.

Κεφάλαιο 5ο: Τεχνολογίες και Εργαλεία Ανάπτυξης

5.1 Εισαγωγή

Στο παρόν κεφάλαιο αναλύονται τα λογισμικά εργαλεία, οι γλώσσες προγραμματισμού και το υλικό που επιλέχθηκαν για την υλοποίηση της εφαρμογής. Η ανάλυση εστιάζει στα τεχνικά χαρακτηριστικά της γλώσσας C# και της μηχανής Unity, εξηγώντας πώς συγκεκριμένες προγραμματιστικές δομές και μηχανισμοί καθιστούν εφικτή την προσομοίωση σύνθετων ρομποτικών συστημάτων και λειτουργιών Επαυξημένης Πραγματικότητας.

5.2 Γλώσσα Προγραμματισμού C#

Η C# είναι μια σύγχρονη, αντικειμενοστραφής γλώσσα προγραμματισμού γενικού σκοπού, η οποία αναπτύχθηκε από τη Microsoft και τρέχει στο πλαίσιο .NET. Επιλέχθηκε ως η κύρια γλώσσα ανάπτυξης καθώς αποτελεί το βασικό μέσο συγγραφής σεναρίων (scripting) για τη μηχανή Unity, προσφέροντας ισχυρή τυποποίηση (strong typing) και αυτόματη διαχείριση μνήμης [94]. Για τις ανάγκες της παρούσας εργασίας, αξιοποιήθηκαν προηγμένα χαρακτηριστικά της γλώσσας για τη δημιουργία μιας αρθρωτής αρχιτεκτονικής.

5.2.1 Κλάσεις, Αντικείμενα και Δημιουργία Στιγμιότυπων

Στον πυρήνα της C# βρίσκεται η έννοια της Κλάσης (Class), η οποία λειτουργεί ως το αρχιτεκτονικό σχέδιο για τη δημιουργία τύπων δεδομένων. Μια κλάση ενθυλακώνει δεδομένα (πεδία/ιδιότητες) και συμπεριφορές (μεθόδους) σε μια ενιαία οντότητα. Το Αντικείμενο αποτελεί ένα συγκεκριμένο στιγμιότυπο μιας κλάσης στη μνήμη του υπολογιστή. Η διαδικασία δημιουργίας ενός αντικειμένου ονομάζεται instantiation και πραγματοποιείται συνήθως με τη χρήση της λέξης-κλειδί “new”, η οποία δεσμεύει την απαραίτητη μνήμη και καλεί τον κατασκευαστή (constructor) της κλάσης [95]:

Στο πλαίσιο της Unity, η δημιουργία αντικειμένων διαφοροποιείται ανάλογα με τον τύπο της κλάσης:

- Για κλάσεις που κληρονομούν από τη MonoBehaviour, η δημιουργία και η διαχείριση του κύκλου ζωής γίνεται από τη μηχανή Unity όταν το script επισυνάπτεται σε ένα GameObject.

- Για καθαρές C# κλάσεις, η δημιουργία γίνεται ρητά μέσω του τελεστή new εντός του κώδικα διαχείρισης.

5.2.2 Κληρονομικότητα και Αφαιρετικές (Abstract) Κλάσεις

Η **Κληρονομικότητα (Inheritance)** είναι ο μηχανισμός που επιτρέπει σε μια κλάση (παραγόμενη - derived) να αποκτήσει τις ιδιότητες και τις μεθόδους μιας άλλης κλάσης (βασική - base), προάγοντας την επαναχρησιμοποίηση κώδικα και την ιεραρχική οργάνωση [96].

Ιδιαίτερη σημασία για την παρούσα εργασία έχουν οι **Αφαιρετικές Κλάσεις (Abstract Classes)**. Από μία abstract κλάση δεν επιτρέπεται να δημιουργηθούν αντικείμενα. Χρησιμεύει αποκλειστικά ως βάση για άλλες κλάσεις, ορίζοντας abstract μεθόδους που πρέπει να υλοποιηθούν από τις παραγόμενες κλάσεις, ενώ μπορεί ταυτόχρονα να παρέχει και υλοποιημένη, κοινή λειτουργικότητα υπό την μορφή σαφών (concrete) μεθόδων [97].

5.2.3 Delegates, Actions και Συμβάντα (Events)

Η ανάπτυξη σύνθετων συστημάτων λογισμικού, και ειδικότερα εφαρμογών προσομοίωσης και ρομποτικής, απαιτεί μηχανισμούς για την ασύγχρονη εκτέλεση εργασιών και την αποσύζευξη (decoupling) των κλάσεων. Η C# παρέχει τρεις θεμελιώδεις έννοιες για την επίτευξη αυτών των στόχων: τους Delegates, τα Actions και τα Events.

Ο Delegate είναι ένας τύπος αναφοράς (reference type) που ορίζει μια υπογραφή μεθόδου (method signature). Λειτουργεί ουσιαστικά ως ένας "δείκτης συνάρτησης" (function pointer) με τη σημαντική διαφορά ότι είναι αντικειμενοστραφής και παρέχει ασφάλεια τύπου (type-safe). Ένα από τα ισχυρότερα χαρακτηριστικά των delegates είναι η δυνατότητα Multicasting. Ένα αντικείμενο delegate μπορεί να διατηρεί αναφορές σε περισσότερες από μία μεθόδους ταυτόχρονα. Όταν κληθεί ο delegate, όλες οι συνδεδεμένες μέθοδοι εκτελούνται διαδοχικά. Αυτός ο μηχανισμός επιτρέπει την αντιμετώπιση των μεθόδων ως δεδομένα, καθιστώντας δυνατή την υλοποίηση μηχανισμών "Callback", όπου μια μέθοδος περνιέται ως παράμετρος σε μια άλλη για να εκτελεστεί σε μεταγενέστερο χρόνο [98].

Για την απλούστευση της χρήσης των delegates, το .NET Framework παρέχει τους προκαθορισμένους γενικούς τύπους (generic delegates) Action και Func. Το Action είναι ένας delegate που αντιπροσωπεύει μια μέθοδο η οποία δεν επιστρέφει τιμή (void) και μπορεί να δεχτεί από 0 έως 16 παραμέτρους (π.χ. Action<T>). Η χρήση των Actions εξαλείφει την ανάγκη ρητού ορισμού νέων τύπων delegates για κάθε διαφορετική υπογραφή μεθόδου. Στον προγραμματισμό ρομποτικών συστημάτων, τα Actions χρησιμοποιούνται ευρέως για τον ορισμό Callbacks ολοκλήρωσης. Επιτρέπουν σε μια διεργασία (π.χ. μια χρονοβόρα κίνηση ή μια δικτυακή αίτηση) να ειδοποιήσει το σύστημα ελέγχου μόλις ολοκληρωθεί, χωρίς να μπλοκάρει την κύρια ροή εκτέλεσης (non-blocking execution) [98].

Τα Events αποτελούν μια ειδική υλοποίηση των delegates που εξυπηρετεί το αρχιτεκτονικό πρότυπο Publisher-Subscriber (Εκδότης-Συνδρομητής).

Ενώ ένας απλός delegate (ή Action) μπορεί να κληθεί και να τροποποιηθεί από οποιοδήποτε τμήμα του κώδικα που έχει πρόσβαση σε αυτόν, το Event παρέχει ένα επίπεδο προστασίας και ενθυλάκωσης (encapsulation):

- Περιορισμένη Πρόσβαση: Μόνο η κλάση που δηλώνει το Event (Publisher) έχει το δικαίωμα να το προκαλέσει (invoke). Οι εξωτερικές κλάσεις (Subscribers) μπορούν μόνο να εγγραφούν ή να απεγγραφούν από αυτό.

- Ασφάλεια Συνδρομής: Η εγγραφή γίνεται μέσω των τελεστών “+=” και “-=”, διασφαλίζοντας ότι ένας συνδρομητής δεν μπορεί κατά λάθος να διαγράψει τους υπόλοιπους συνδρομητές (κάτι που θα μπορούσε να συμβεί με την απλή ανάθεση “=” σε έναν delegate).

Μέσω των Events, επιτυγχάνεται υψηλός βαθμός αποσύζευξης, καθώς ο Εκδότης δεν χρειάζεται να γνωρίζει ποιοι ή πόσοι είναι οι Συνδρομητές, ούτε πώς θα αντιδράσουν στο συμβάν [99].

5.2.4 Διεπαφές (Interfaces)

Παράλληλα με τις αφηρημένες κλάσεις, για τον σχεδιασμό της αρχιτεκτονικής χρησιμοποιήθηκαν εκτενώς Διεπαφές (Interfaces). Στην ορολογία της C#, ένα Interface ορίζει ένα "συμβόλαιο συμπεριφοράς", δηλαδή ένα σύνολο υπογραφών μεθόδων, ιδιοτήτων ή συμβάντων που πρέπει

υποχρεωτικά να υλοποιήσει μια κλάση, χωρίς όμως το ίδιο το Interface να παρέχει λειτουργικό κώδικα ή να δεσμεύει μνήμη για δεδομένα [100].

Η χρήση των διεπαφών προσφέρει δύο σημαντικά πλεονεκτήματα σε σχέση με την κλασική κληρονομικότητα:

- Πολλαπλή Υλοποίηση (Multiple Implementation): Σε αντίθεση με την κληρονομικότητα κλάσεων, όπου μια κλάση μπορεί να κληρονομήσει μόνο από μία γονική κλάση (Single Inheritance), η C# επιτρέπει σε μια κλάση να υλοποιεί πολλαπλές διεπαφές ταυτόχρονα. Αυτό επιτρέπει τη δημιουργία σύνθετων αντικειμένων που συνδυάζουν διαφορετικές συμπεριφορές (π.χ. ένα αντικείμενο μπορεί να είναι ταυτόχρονα IDisposable και IComparable).
- Αποσύζευξη (Decoupling) και Εναλλαξιμότητα: Τα Interfaces επιτρέπουν τον προγραμματισμό βάσει δυνατοτήτων και όχι βάσει συγκεκριμένων τύπων. Ένα σύστημα μπορεί να αλληλεπιδρά με αντικείμενα γνωρίζοντας μόνο ότι υλοποιούν μια συγκεκριμένη διεπαφή, αγνοώντας την εσωτερική τους δομή ή την ιεραρχία τους. Αυτό καθιστά το λογισμικό εξαιρετικά ευέλικτο, καθώς επιτρέπει την εύκολη αντικατάσταση ή προσθήκη νέων υλοποιήσεων (π.χ. προσθήκη ενός νέου τύπου αισθητήρα) χωρίς να απαιτείται τροποποίηση του κώδικα που τον διαχειρίζεται (Dependency Inversion Principle).

5.2.5 LINQ (Language Integrated Query)

Το LINQ αποτελεί ένα σύνολο χαρακτηριστικών που εισάγει δυνατότητες υποβολής ερωτημάτων απευθείας στη σύνταξη της γλώσσας C#. Επιτρέπει στον προγραμματιστή να διαχειρίζεται συλλογές δεδομένων (όπως πίνακες και λίστες) χρησιμοποιώντας δηλωτικό προγραμματισμό (declarative programming), εστιάζοντας δηλαδή στο τι πρέπει να επιτευχθεί και όχι στο πώς θα εκτελεστεί βήμα-προς-βήμα η διαδικασία [101].

Η χρήση του LINQ προσφέρει σημαντικά πλεονεκτήματα στην ανάπτυξη πολύπλοκων αλγορίθμων:

- Αναγνωσιμότητα και Συντομία: Αντικαθιστά εκτενείς βρόχους επανάληψης (for, foreach) και εμφωλευμένες συνθήκες ελέγχου (if) με συνοπτικές και ευανάγνωστες εκφράσεις. Αυτό μειώνει την πιθανότητα λογικών σφαλμάτων και καθιστά τον κώδικα πιο εύκολο στη συντήρηση.
- Λειτουργικός Χειρισμός Δεδομένων: Μέσω της χρήσης εκφράσεων λάμδα (lambda expressions) και συναρτήσεων ανώτερης τάξης (όπως .Where(), .Select(), .OrderBy()), επιτρέπει την αλυσιδωτή εκτέλεση πράξεων φιλτραρίσματος, προβολής και ταξινόμησης.

5.2.6 Γενικευμένοι Τύπου (Generics)

Οι Γενικευμένοι Τύποι (Generics) αποτελούν έναν μηχανισμό της C# που επιτρέπει τον ορισμό κλάσεων, διεπαφών και μεθόδων με μία ή περισσότερες παραμέτρους τύπου (type parameters), οι οποίες συνήθως συμβολίζονται με το γράμμα T. Ουσιαστικά, επιτρέπουν τη δημιουργία "προτύπων" κώδικα, όπου ο συγκεκριμένος τύπος δεδομένων καθορίζεται μόνο κατά τη στιγμή της δήλωσης ή της κλήσης και όχι κατά τον ορισμό της κλάσης [102].

Η χρήση των Generics προσφέρει τρία θεμελιώδη πλεονεκτήματα στην ανάπτυξη λογισμικού:

- Επαναχρησιμοποίηση Κώδικα (Reusability): Επιτρέπει τη συγγραφή ενός αλγορίθμου ή μιας δομής δεδομένων μία φορά, η οποία μπορεί να λειτουργήσει με οποιονδήποτε τύπο δεδομένων. Χαρακτηριστικό παράδειγμα είναι οι συλλογές (Collections), όπως η List<T>, η οποία μπορεί να χρησιμοποιηθεί για να αποθηκεύσει ακέραιους (List<int>), συμβολοσειρές ή ακόμα και σύνθετα αντικείμενα (List<GameObject>), χωρίς να απαιτείται η συγγραφή ξεχωριστής κλάσης λίστας για κάθε τύπο.

- Ασφάλεια Τύπου (Type Safety): Σε αντίθεση με παλαιότερες προσεγγίσεις που χρησιμοποιούσαν γενικούς τύπους object, οι Generics επιβάλλουν έλεγχο τύπων κατά τη μεταγλώττιση (compile-time). Αυτό σημαίνει ότι ο μεταγλωττιστής μπορεί να εντοπίσει και να αποτρέψει σφάλματα ασυμβατότητας τύπων πριν την εκτέλεση του προγράμματος, εξαλείφοντας την ανάγκη για ριψοκίνδυνες μετατροπές (casting).
- Βελτιστοποίηση Απόδοσης (Performance): Η χρήση Generics, ειδικά με τύπους τιμής (value types), βελτιώνει σημαντικά την απόδοση της εφαρμογής. Αποφεύγεται η δαπανηρή διαδικασία του "εγκιβωτισμού" και "απεγκιβωτισμού" (boxing/unboxing), καθώς ο κώδικας παράγεται ειδικά για τον συγκεκριμένο τύπο δεδομένων που χρησιμοποιείται [103].

Στο περιβάλλον της Unity, οι Generics είναι πανταχού παρόντες. Μέθοδοι όπως η GetComponent<T>() επιτρέπουν την ασφαλή και άμεση ανάκτηση συγκεκριμένων συστατικών (Components) από ένα αντικείμενο, χωρίς την ανάγκη ελέγχου τύπου ή μετατροπής, συμβάλλοντας στην καθαρότητα και την ταχύτητα του κώδικα.

5.3 Μηχανή Ανάπτυξης Unity

Η Unity αποτελεί μία από τις πιο διαδεδομένες πλατφόρμες ανάπτυξης εφαρμογών πραγματικού χρόνου (Real-Time Development Platform) παγκοσμίως. Αν και αρχικά σχεδιάστηκε ως μηχανή ανάπτυξης ηλεκτρονικών παιχνιδιών, η ευελιξία και η απόδοσή της έχουν οδηγήσει στην ευρεία υιοθέτησή της σε βιομηχανικούς τομείς για εφαρμογές προσομοίωσης, εκπαίδευσης και οπτικοποίησης δεδομένων. Η επιλογή της για την παρούσα εργασία βασίστηκε στην ικανότητά της να συνδυάζει προηγμένη απόδοση φυσικής (Physics), γραφικά υψηλής ποιότητας και υποστήριξη για φορητές συσκευές σε ένα ενιαίο περιβάλλον ανάπτυξης [104].

5.3.1 Αρχιτεκτονική Entity-Component (Σύνθεση)

Σε αντίθεση με την παραδοσιακή κληρονομικότητα κλάσεων που συναντάται σε πολλές εφαρμογές λογισμικού, η Unity βασίζεται στο αρχιτεκτονικό πρότυπο Σύνθεσης (Composition over Inheritance).

Κάθε αντικείμενο στον ψηφιακό κόσμο (το ρομπότ, η πίστα, το φως) είναι ένα GameObject. Από μόνο του, ένα GameObject είναι απλώς ένας περιέκτης (container) που διαθέτει θέση, περιστροφή και κλίμακα (Transform).

Η συμπεριφορά και οι ιδιότητες ενός αντικειμένου καθορίζονται από τα Components που προσαρτώνται σε αυτό. Τα Components είναι C# κλάσεις που κληρονομούν από τη βασική κλάση MonoBehaviour.

Αυτή η προσέγγιση επιτρέπει τη δημιουργία σύνθετων συμπεριφορών μέσω του συνδυασμού μικρών, επαναχρησιμοποιούμενων μονάδων κώδικα, προσφέροντας υψηλή ευελιξία και ευκολία στη συντήρηση του λογισμικού [105].

5.3.2 Ο Κύκλος Ζωής της Εφαρμογής

Οι εφαρμογές προσομοίωσης πραγματικού χρόνου λειτουργούν βάσει ενός ατέρμονου βρόχου, ο οποίος εκτελείται πολλές φορές το δευτερόλεπτο. Η Unity παρέχει συγκεκριμένες μεθόδους-αγκίστρια (hooks) για την εκτέλεση κώδικα σε διάφορα στάδια του βρόχου [106]:

- **Μέθοδοι Start και Awake**: Εκτελούνται μία φορά κατά τη δημιουργία του αντικειμένου, επιτρέποντας την αρχικοποίηση μεταβλητών και τη σύνδεση εξαρτήσεων.

- **Μέθοδος Update**: Εκτελείται σε κάθε καρέ της οθόνης. Χρησιμοποιείται για την ανάγνωση εισόδων και την ενημέρωση της γραφικής διεπαφής (UI). Ο χρόνος μεταξύ δύο καρέ, γνωστός ως delta time, είναι μεταβλητός και εξαρτάται από τον φόρτο της CPU/GPU. Χρησιμοποιείται για την ανάγνωση εισόδων (Input) και την απόκριση UI.

- **Μέθοδος FixedUpdate**: Εκτελείται σε σταθερά χρονικά διαστήματα. Είναι κρίσιμη για υπολογισμούς που αφορούν τη μηχανή φυσικής, διασφαλίζοντας σταθερότητα και επαναληψιμότητα στην κίνηση ανεξάρτητα από τον ρυθμό καρέ (frame rate). Με αυτόν τον τρόπο η εφαρμογή παρουσιάζει την ίδια συμπεριφορά είτε τρέχει σε μία αδύναμη συσκευή με χαμηλό ρυθμό καρέ, είτε σε μία πιο ικανή συσκευή με υψηλό ρυθμό καρέ.

5.3.3 Μηχανή Φυσικής και Ανίχνευση Συγκρούσεων

Η Unity ενσωματώνει τη μηχανή φυσικής NVIDIA PhysX για 3D προσομοιώσεις. Δύο σημαντικά στοιχεία που προσφέρει η μηχανή φυσικής είναι τα εξής:

- Rigidbodies (Στερεά Σώματα): Προσδίδουν φυσικές ιδιότητες σε ένα αντικείμενο (μάζα, ορμή, τριβή, επίδραση βαρύτητας). Επιτρέπουν στο αντικείμενο να κινείται βάσει δυνάμεων και όχι απλής τηλεμεταφοράς.

- Colliders (Συγκρουστήρες): Καθορίζουν το γεωμετρικό σχήμα του αντικειμένου για τους υπολογισμούς σύγκρουσης.

Μια κρίσιμη λειτουργία βελτιστοποίησης είναι ο Πίνακας Συγκρούσεων Επιπέδων (Layer Collision Matrix). Η Unity επιτρέπει την ομαδοποίηση αντικειμένων σε "Επίπεδα" (Layers). Μέσω του πίνακα, ορίζεται ποια επίπεδα αλληλεπιδρούν μεταξύ τους. Αυτό επιτρέπει ορισμένα αντικείμενα να αντιδρούν σε γεγονότα όπως η σύγκρουση όταν τέμνουν άλλα αντικείμενα του ίδιου επιπέδου, ενώ ταυτόχρονα να αγνοούν τομές με αντικείμενα άλλων επιπέδων [107].

5.3.4 Raycasting

Το Raycasting (Εκπομπή Ακτίνων) είναι η μέθοδος προσομοίωσης αισθητήρων οπτικής επαφής. Η μηχανή φυσικής εκπέμπει μια νοητή ακτίνα και ελέγχει για τομές με Colliders.

Η Unity παρέχει δυνατότητες όπως:

- **RaycastHit:** Δομή δεδομένων που επιστρέφει ακριβείς πληροφορίες για το σημείο τομής, το κάθετο διάνυσμα (normal) της επιφάνειας (χρήσιμο για υπολογισμό γωνίας πρόσπτωσης) και την αναφορά στο αντικείμενο που χτυπήθηκε.
- **SphereCast / BoxCast:** Παραλλαγές που εκπέμπουν όγκους αντί για λεπτές ακτίνες, προσομοιώνοντας παχύτερες δέσμες αισθητήρων (π.χ. σόναρ ευρείας δέσμης).

5.3.5 ScriptableObjects

Τα ScriptableObjects αποτελούν έναν ειδικό τύπο κλάσης της Unity που επιτρέπει την αποθήκευση δεδομένων ως αρχεία (assets), ανεξάρτητα από τα στιγμιότυπα των κλάσεων στη σκηνή [108]. Η χρήση τους είναι θεμελιώδης για την υλοποίηση αρχιτεκτονικών που οδηγούνται από δεδομένα (Data-Driven Architecture). Επιτρέπουν τον διαχωρισμό της λογικής (κώδικας) από τις παραμέτρους

(δεδομένα), καθιστώντας δυνατή τη δημιουργία, αποθήκευση και εναλλαγή διαφορετικών σεναρίων συμπεριφοράς ή ρυθμίσεων συστήματος χωρίς την ανάγκη τροποποίησης ή επανα-μεταγλώττισης του πηγαίου κώδικα.

5.3.6 Σύστημα Prefabs

Το σύστημα των Prefabs επιτρέπει τη δημιουργία προτύπων αντικειμένων που περιλαμβάνουν όλα τα απαραίτητα Components, ρυθμίσεις και θυγατρικά αντικείμενα. Ένα Prefab μπορεί να αναπαραχθεί πολλές φορές κατά τη διάρκεια εκτέλεσης της εφαρμογής. Αυτός ο μηχανισμός είναι απαραίτητος για τη δυναμική διαχείριση περιεχομένου, επιτρέποντας στην εφαρμογή να φορτώνει και να δημιουργεί σύνθετες οντότητες (όπως αντικείμενα ή περιβάλλοντα) σε πραγματικό χρόνο.

5.3.7 Πλαίσιο AR Foundation

Το AR Foundation είναι ένα πακέτο λογισμικού της Unity που παρέχει ένα ενιαίο περιβάλλον ανάπτυξης (API) για εφαρμογές Επαυξημένης Πραγματικότητας. Λειτουργεί ως επίπεδο αφαίρεσης (abstraction layer), γεφυρώνοντας τα εγγενή SDKs των κινητών συσκευών (ARCore για Android και ARKit για iOS) [109]. Μέσω του AR Foundation, η Unity παρέχει πρόσβαση σε υποσυστήματα όπως:

- **Plane Detection**: Η ικανότητα ανάλυσης του νέφους σημείων (point cloud) που λαμβάνει η κάμερα για τον εντοπισμό επίπεδων επιφανειών στο φυσικό χώρο.

- **Trackables & Anchors**: Η διαχείριση χωρικών σημείων αναφοράς που επιτρέπουν την "αγκύρωση" ψηφιακών αντικειμένων σε φυσικές συντεταγμένες, διορθώνοντας αυτόματα τα σφάλματα ολίσθησης (drift) που προκύπτουν από την κίνηση της συσκευής.

Σε συνδυασμό με την μηχανή γραφικών και προσομοιώσεων φυσικής της Unity, μπορεί να γίνει τοποθέτηση ψηφιακών αντικειμένων με βάση τον πραγματικό κόσμο.

5.3.8 Σύστημα Διεπαφής Χρήστη (Unity UI)

Η αλληλεπίδραση του χρήστη με την εφαρμογή, από την επιλογή πίστας μέχρι τη δημιουργία κανόνων λογικής, βασίζεται στο σύστημα Unity UI (γνωστό και ως uGUI). Πρόκειται για ένα σύστημα που βασίζεται σε GameObjects, επιτρέποντας στα στοιχεία της διεπαφής να συνυπάρχουν και να αλληλεπιδρούν με τον τρισδιάστατο κόσμο της εφαρμογής [110].

5.3.8.1 Canvas και Canvas Scaler

Το θεμελιώδες δομικό στοιχείο του UI είναι το Canvas (Καμβάς). Όλα τα γραφικά στοιχεία (κουμπιά, κείμενα, εικόνες) πρέπει να βρίσκονται ιεραρχικά κάτω από ένα Canvas για να αποδοθούν στην οθόνη. Στην παρούσα εφαρμογή, χρησιμοποιείται η λειτουργία Screen Space - Overlay, η οποία διασφαλίζει ότι η διεπαφή θα απεικονίζεται πάντα πάνω από τον τρισδιάστατο κόσμο (AR Camera feed), ανεξάρτητα από τη θέση της κάμερας. Για την αντιμετώπιση του κατακερματισμού (fragmentation) των συσκευών Android, οι οποίες διαθέτουν οθόνες ποικίλων διαστάσεων και πυκνοτήτων pixel (DPI), χρησιμοποιείται το συστατικό Canvas Scaler. Αυτό προσαρμόζει αυτόματα την κλίμακα των στοιχείων UI βάσει μιας ανάλυσης αναφοράς (Reference Resolution), διασφαλίζοντας ότι το μέγεθος των κουμπιών και η αναγνωσιμότητα του κειμένου παραμένουν σταθερά σε διαφορετικές συσκευές [110].

5.3.8.2 RectTransform και Σύστημα Αγκύρωσης (Anchors)

Σε αντίθεση με τα τρισδιάστατα αντικείμενα που χρησιμοποιούν το απλό **Transform**, τα στοιχεία UI χρησιμοποιούν το **RectTransform**. Αυτό το ειδικό component ορίζει ένα ορθογώνιο πλαίσιο αντί για ένα απλό σημείο. Το RectTransform εισάγει την έννοια των **Anchors (Αγκυρώσεων)**. Τα Anchors επιτρέπουν σε ένα στοιχείο UI να ορίζει τη θέση και το μέγεθός του σχετικά με το γονικό του αντικείμενο [110].

5.3.8.3 Σύστημα Συμβάντων (Event System)

Η διαχείριση της αλληλεπίδρασης (αφή, κλικ) γίνεται μέσω του **Event System**. Το σύστημα αυτό λειτουργεί ανεξάρτητα από τη μηχανή φυσικής, χρησιμοποιώντας τη δική του μέθοδο ανίχνευσης που ονομάζεται **Graphic Raycaster**. Το Event System παρακολουθεί τις εισόδους (Input) και αποστέλλει μηνύματα στα στοιχεία UI που βρίσκονται κάτω από τον δείκτη (pointer). Υποστηρίζει μια ιεραρχία γεγονότων (Event Bubbling), επιτρέποντας την υλοποίηση σύνθετων αλληλεπιδράσεων, όπως η κύλιση (scroll) μιας λίστας ή το πάτημα ενός κουμπιού που βρίσκεται μέσα σε ένα κινούμενο παράθυρο [111].

5.3.8.4 Δυναμική Διάταξη (Auto Layout System)

Για τη δημιουργία του "Επεξεργαστή Κανόνων" (Rule Editor), όπου ο χρήστης προσθέτει δυναμικά νέους κανόνες, η εφαρμογή αξιοποιεί το σύστημα Αυτόματης Διάταξης (Layout System) [112]. Χρησιμοποιούνται components όπως:

- **Vertical Layout Group:** Τοποθετεί αυτόματα τα θυγατρικά αντικείμενα (τους κανόνες) σε κατακόρυφη στοίχιση, το ένα κάτω από το άλλο.
- **Content Size Fitter:** Προσαρμόζει το μέγεθος του περιέκτη (container) ανάλογα με το πλήθος και το μέγεθος των περιεχομένων του. Αυτός ο συνδυασμός επιτρέπει τη δημιουργία λιστών μεταβλητού μήκους χωρίς την ανάγκη χειροκίνητου υπολογισμού θέσεων pixel-προς-pixel, καθιστώντας τη διεπαφή ευέλικτη και επεκτάσιμη.

5.3.8.5 TextMeshPro (TMP)

Για την απεικόνιση κειμένου χρησιμοποιήθηκε το πακέτο **TextMeshPro**, το οποίο αποτελεί την σύγχρονη λύση κειμένου της Unity. Σε αντίθεση με το παλαιότερο σύστημα Text που χρησιμοποιούσε bitmap γραμματοσειρές (οι οποίες θόλωναν κατά τη μεγέθυνση), το TMP χρησιμοποιεί την τεχνική **Signed Distance Fields (SDF)**. Αυτό επιτρέπει την απόδοση κειμένου με απόλυτη ευκρίνεια σε οποιαδήποτε ανάλυση και επίπεδο ζουμ, ενώ προσφέρει προηγμένες δυνατότητες μορφοποίησης, απαραίτητες για την ευανάγνωστη παρουσίαση των κανόνων και των δεδομένων των αισθητήρων στην μικρή οθόνη μιας κινητής συσκευής [113].

5.3.9 Σύστημα Απόδοσης Γραφικών (Universal Render Pipeline - URP)

Για την οπτική απεικόνιση της προσομοίωσης, η εφαρμογή δεν χρησιμοποίησε τον παραδοσιακό αγωγό απόδοσης (Built-in Render Pipeline), αλλά το **Universal Render Pipeline (URP)**. Το URP είναι ένας αγωγός απόδοσης που βασίζεται στην τεχνολογία **Scriptable Render Pipeline (SRP)** της Unity. Σε αντίθεση με τις παλαιότερες προσεγγίσεις όπου η διαδικασία απόδοσης (rendering loop) ήταν "σκληρά κωδικοποιημένη" (hard-coded) στη μηχανή, το SRP επιτρέπει τον έλεγχο της

διαδικασίας μέσω κώδικα C#. Το URP είναι ειδικά σχεδιασμένο και βελτιστοποιημένο για να παρέχει τη μέγιστη δυνατή απόδοση σε ευρύ φάσμα συσκευών, από κινητά τηλέφωνα έως συσκευές VR/AR [114].

Η επιλογή του URP για την παρούσα εργασία υπαγορεύτηκε από τρεις κρίσιμους τεχνικούς παράγοντες:

5.3.9.1 SRP Batcher (Βελτιστοποίηση CPU)

Σε εφαρμογές 3D, ένα από τα συχνότερα σημεία συμφόρησης (bottleneck) είναι η επικοινωνία μεταξύ CPU και GPU. Κάθε αντικείμενο που σχεδιάζεται στην οθόνη απαιτεί μια εντολή σχεδίασης (Draw Call) και αλλαγές στην κατάσταση της GPU (Render State changes). Το URP εισάγει τον **SRP Batcher**, έναν μηχανισμό που επιταχύνει δραματικά αυτή τη διαδικασία. Αντί να ρυθμίζει τις παραμέτρους των υλικών (materials) για κάθε αντικείμενο ξεχωριστά, ο SRP Batcher διατηρεί τα δεδομένα των υλικών μόνιμα στη μνήμη της GPU (CBUFFERS). Αυτό επιτρέπει τη σχεδίαση πολλών διαφορετικών αντικειμένων που χρησιμοποιούν την ίδια παραλλαγή Shader (Shader Variant) με ελάχιστο κόστος CPU [115]. Στην προσομοίωση, όπου υπάρχουν πολλά μικρά αντικείμενα (π.χ. τα τμήματα του ρομπότ, τα εμπόδια της πίστας, οι γραμμές του Lidar), ο SRP Batcher μειώνει σημαντικά τον φόρτο του επεξεργαστή, απελευθερώνοντας πόρους για τους υπολογισμούς του AR και της λογικής πλοήγησης.

5.3.9.2 Single-Pass Forward Rendering (Βελτιστοποίηση GPU)

Οι φορητές συσκευές έχουν περιορισμένο ρυθμό πλήρωσης (Fill Rate) και εύρος ζώνης μνήμης. Το URP χρησιμοποιεί μια βελτιστοποιημένη τεχνική απόδοσης φωτισμού γνωστή ως **Single-Pass Forward Rendering**. Στον παραδοσιακό αγωγό, κάθε δυναμικό φως που έπεφτε πάνω σε ένα αντικείμενο απαιτούσε ένα επιπλέον πέρασμα σχεδίασης (additional render pass), αυξάνοντας εκθετικά τον φόρτο της GPU. Το URP υπολογίζει την επίδραση όλων των φώτων σε ένα μόνο πέρασμα (pass) ανά αντικείμενο. Αυτό είναι κρίσιμο για εφαρμογές AR, καθώς επιτρέπει την προσθήκη εικονικού φωτισμού που ταιριάζει με τον φυσικό φωτισμό του χώρου (Light Estimation), χωρίς να προκαλείται πτώση του ρυθμού καρέ (FPS) ή υπερθέρμανση της συσκευής [114].

5.3.9.3 Shader Graph (Οπτική Δημιουργία Shaders)

Το URP υποστηρίζει το **Shader Graph**, ένα εργαλείο οπτικού προγραμματισμού για τη δημιουργία σκιαστών (shaders). Αντί για τη συγγραφή κώδικα σε γλώσσα HLSL/Cg, η δημιουργία των υλικών γίνεται μέσω σύνδεσης κόμβων (nodes). Η δυνατότητα αυτή αξιοποιήθηκε για τη δημιουργία ειδικών εφέ οπτικοποίησης αισθητήρων. Για παράδειγμα, η απεικόνιση της δέσμης του Lidar ή των ζωνών θερμότητας δεν απαιτεί βαριά γεωμετρία, αλλά υλοποιείται μέσω ελαφριών, παραμετρικών shaders που δημιουργήθηκαν στο Shader Graph, επιτρέποντας δυναμικές αλλαγές χρώματος και διαφάνειας με μηδενικό αντίκτυπο στην απόδοση.

5.3.9.4 Μετα-επεξεργασία (Post-Processing)

Το URP διαθέτει ενσωματωμένο και βελτιστοποιημένο σύστημα μετα-επεξεργασίας εικόνας (Post-Processing Stack). Αν και σε εφαρμογές AR η χρήση βαρέων εφέ αποφεύγεται για λόγους ρεαλισμού και απόδοσης, το σύστημα επιτρέπει λεπτές παρεμβάσεις, όπως η διόρθωση χρωμάτων (Color Grading) για την καλύτερη ενσωμάτωση των ψηφιακών αντικειμένων στην παλέτα χρωμάτων

που λαμβάνει η κάμερα, κάνοντας το ρομπότ να φαίνεται πιο "φυσικό" μέσα στο πραγματικό περιβάλλον [116].

5.3.10 Backend Μεταγλώττισης IL2CPP (Intermediate Language to C++)

Για την εξαγωγή της εφαρμογής σε περιβάλλον Android, η Unity προσφέρει δύο επιλογές backend για τη συγγραφή σεναρίων (scripting backends): το παραδοσιακό Mono και το σύγχρονο **IL2CPP**. Για τις ανάγκες της παρούσας εργασίας επιλέχθηκε το IL2CPP, καθώς αποτελεί τη μοναδική λύση που εξασφαλίζει τη βέλτιστη απόδοση και τη συμβατότητα με τις σύγχρονες απαιτήσεις του οικοσυστήματος Android [117].

Το IL2CPP δεν είναι απλώς ένας μεταγλωττιστής, αλλά μια τεχνολογία **AOT (Ahead-Of-Time)** μεταγλώττισης. Η διαδικασία λειτουργίας του αναλύεται στα εξής στάδια:

5.3.10.1 Η Διαδικασία Μετατροπής (Transpilation Pipeline)

Σε αντίθεση με το Mono που χρησιμοποιεί μεταγλώττιση Just-In-Time (JIT) κατά τη διάρκεια εκτέλεσης της εφαρμογής, το IL2CPP μετατρέπει ολόκληρο τον κώδικα πριν την εκτέλεση.

1. **Μεταγλώττιση C# σε IL:** Αρχικά, ο μεταγλωττιστής της C# (Roslyn) μετατρέπει τα scripts σε Διαχειριζόμενα Συναρμολογήματα (Managed Assemblies / .dll) που περιέχουν κώδικα Intermediate Language (IL), όπως σε κάθε τυπική .NET εφαρμογή.
2. **Μετατροπή IL σε C++:** Στη συνέχεια, το εργαλείο AOT του IL2CPP λαμβάνει αυτά τα assemblies και τα μετατρέπει ("transpiles") σε πρότυπο κώδικα C++. Κάθε κλάση και μέθοδος της C# μετατρέπεται σε αντίστοιχες δομές (structs) και συναρτήσεις C++.
3. **Εγγενής Μεταγλώττιση (Native Compilation):** Τέλος, ο παραγόμενος κώδικας C++ μεταγλωττίζεται από τον native compiler της πλατφόρμας (στην περίπτωση του Android, χρησιμοποιείται το Android NDK και ο compiler Clang). Το αποτέλεσμα είναι μια βιβλιοθήκη εγγενούς κώδικα μηχανής (.so file) βελτιστοποιημένη για τον επεξεργαστή της συσκευής [117].

5.3.10.2 Υποστήριξη Αρχιτεκτονικής ARM64

Από τον Αύγουστο του 2019, η Google κατέστησε υποχρεωτική την υποστήριξη της αρχιτεκτονικής 64-bit (ARM64) για όλες τις εφαρμογές που δημοσιεύονται στο Play Store. Το παλαιότερο backend (Mono) στο Android υποστήριζε κυρίως 32-bit αρχιτεκτονικές. Η χρήση του IL2CPP ήταν μονόδρομος για την εργασία, καθώς επιτρέπει την παραγωγή κώδικα ARM64-v8a [118]. Η 64-bit αρχιτεκτονική προσφέρει σημαντικά πλεονεκτήματα στην προσομοίωση:

- **Μεγαλύτερος Χώρος Διευθύνσεων:** Επιτρέπει την προσπέλαση σε περισσότερα από 4GB μνήμης RAM, κάτι απαραίτητο για τη διαχείριση των βαριών textures και των Point Clouds του AR.
- **Βελτιωμένοι Καταχωρητές (Registers):** Οι επεξεργαστές ARM64 διαθέτουν περισσότερους και μεγαλύτερους καταχωρητές, επιτρέποντας ταχύτερους μαθηματικούς υπολογισμούς, οι οποίοι είναι κρίσιμοι για τη μηχανή φυσικής και το Raycasting.

5.3.10.3 Απόδοση και Βελτιστοποίηση Κώδικα

Το IL2CPP βελτιώνει την απόδοση της εφαρμογής μέσω της στατικής ανάλυσης κώδικα:

- **Code Stripping:** Κατά τη μετατροπή σε C++, το IL2CPP αφαιρεί (strips) τον κώδικα που δεν χρησιμοποιείται ποτέ. Αυτό μειώνει δραστικά το μέγεθος του τελικού αρχείου APK.
- **Devirtualization:** Πολλές εικονικές κλήσεις μεθόδων της C# μπορούν να μετατραπούν σε άμεσες κλήσεις στη C++, μειώνοντας το κόστος εκτέλεσης του κώδικα.
- **Συμβατότητα με Native Plugins:** Καθώς το ARCore είναι μια εγγενής βιβλιοθήκη C/C++, το IL2CPP διευκολύνει την επικοινωνία μεταξύ του κώδικα C# της Unity και του ARCore, μειώνοντας την καθυστέρηση κατά την ανταλλαγή δεδομένων θέσης και εικόνας [117].

5.3.10.4 Διαχείριση Μνήμης

Ακόμα και με τη χρήση C++, το IL2CPP διατηρεί τις λειτουργίες διαχειριζόμενης μνήμης της C# . Ενσωματώνει έναν βελτιστοποιημένο **Garbage Collector**, ο οποίος τρέχει σε ξεχωριστό νήμα. Ωστόσο, η μετατροπή σε C++ επιτρέπει καλύτερη διαχείριση των δομών δεδομένων στη στοίβα και στη σωρό, μειώνοντας τον κατακερματισμό της μνήμης, ο οποίος αποτελεί συχνό αίτιο πτώσης της απόδοσης σε κινητές συσκευές [117].

5.3.11 Σύστημα Εισόδου

Για τη διαχείριση των ενεργειών του χρήστη, και συγκεκριμένα για την ανίχνευση αφής στην οθόνη του κινητού, χρησιμοποιήθηκε το νέο πακέτο **Input System** της Unity. Σε αντίθεση με το παλαιότερο σύστημα που βασιζόταν στη συνεχή ερώτηση της κατάστασης της συσκευής σε κάθε καρέ, το νέο σύστημα λειτουργεί βάσει συμβάντων.

Το σύστημα αυτό διαχωρίζει τη λογική του ελέγχου από τη φυσική συσκευή. Στην εφαρμογή αξιοποιήθηκε η λειτουργία **EnhancedTouchSupport**, η οποία επιτρέπει την ακριβή παρακολούθηση πολλαπλών δακτύλων και την ανάκτηση δεδομένων πίεσης και θέσης με χαμηλή καθυστέρηση. Αυτό είναι κρίσιμο για την AR εμπειρία, καθώς η διάδραση με τα ψηφιακά αντικείμενα πρέπει να είναι άμεση και ακριβής [119].

5.3.12 Coroutines και Ασύγχρονη Εκτέλεση

Μια σημαντική πρόκληση στις προσομοιώσεις πραγματικού χρόνου είναι η εκτέλεση εργασιών που πρέπει να διαρκέσουν συγκεκριμένο χρονικό διάστημα (π.χ. “σάρωση ραντάρ κάθε 2 δευτερόλεπτα”) χωρίς να παγώνει η κεντρική ροή του προγράμματος. Η Unity αντιμετωπίζει αυτό το πρόβλημα μέσω των **Coroutines**. Οι Coroutines είναι μέθοδοι που έχουν τη δυνατότητα να διακόψουν την εκτέλεσή τους και να τη συνεχίσουν στο επόμενο καρέ ή μετά από συγκεκριμένο χρονικό διάστημα. Στον κώδικα της εφαρμογής, μηχανισμοί όπως οι χρονομετρητές των αισθητήρων ή η αναμονή για τη σύνδεση Bluetooth υλοποιούνται με αυτή τη λογική, επιτρέποντας στην εφαρμογή να παραμένει διαδραστική ενώ εκτελεί υπολογισμούς στο παρασκήνιο [120].

5.4 Επίλογος

Στο παρόν κεφάλαιο παρουσιάστηκαν αναλυτικά τα τεχνολογικά εργαλεία και το υλικό που επιλέχθηκαν για την υλοποίηση του συστήματος. Η μηχανή γραφικών Unity σε συνδυασμό με το πλαίσιο AR Foundation αποτέλεσαν τη βάση για τη δημιουργία του ψηφιακού κόσμου και την

ενσωμάτωση της Επαυξημένης Πραγματικότητας. Παράλληλα, η πλατφόρμα Android παρείχε το απαραίτητο οικοσύστημα για τη φορητότητα της εφαρμογής.

Κεφάλαιο 6: Ανάλυση και Σχεδίαση Συστήματος

6.1 Εισαγωγή

Το παρόν κεφάλαιο παρέχει μια λεπτομερή ανάλυση της αρχιτεκτονικής σχεδίασης και της υλοποίησης του συστήματος λογισμικού που αναπτύχθηκε. Σκοπός της ανάλυσης είναι η παρουσίαση των δομικών στοιχείων της εφαρμογής, των μεταξύ τους αλληλεπιδράσεων, καθώς και των τεχνικών αποφάσεων που ελήφθησαν για την επίλυση των προβλημάτων της προσομοίωσης και της διεπαφής με τον πραγματικό κόσμο.

Η σχεδίαση του συστήματος δεν περιορίστηκε απλώς στην ανάπτυξη λειτουργικού κώδικα, αλλά βασίστηκε σε θεμελιώδεις αρχές της Μηχανικής Λογισμικού με στόχο τη δημιουργία μιας λύσης που συνδυάζει την αρθρωτότητα, την επεκτασιμότητα και τη συντηρησιμότητα.

Δεδομένης της πολυπλοκότητας που εισάγει η ταυτόχρονη διαχείριση υποσυστημάτων Επαυξημένης Πραγματικότητας (AR), προσομοίωσης Φυσικής (Physics Simulation) και ασύρματης επικοινωνίας μέσω Bluetooth Low Energy, υιοθετήθηκε μια ευέλικτη αρχιτεκτονική προσέγγιση.

6.1.1 Μεθοδολογία Σχεδίασης και Αρχιτεκτονικό Πρότυπο

Η βάση της ανάπτυξης του κώδικα στηρίχθηκε στο παράδειγμα του Αντικειμενοστραφούς Προγραμματισμού, προσαρμοσμένου στο αρχιτεκτονικό μοντέλο Entity-Component-System που επιβάλλει η μηχανή Unity. Ωστόσο, για την αντιμετώπιση ειδικών απαιτήσεων, όπως η δυναμική συμπεριφορά του ρομπότ και η διαχείριση ετερογενών αισθητήρων, το σύστημα επίσης χρησιμοποίησε καθιερωμένα Πρότυπα Σχεδίασης (Design Patterns):

1. **Αρχιτεκτονική Βασισμένη σε Δεδομένα (Data-Driven Architecture):** Για τον καθορισμό της λογικής του ρομπότ, αποφεύχθηκε ο σταθερός ορισμός κανόνων εντός των κλάσεων. Αντ' αυτού, αξιοποιήθηκαν τα ScriptableObjects της Unity, επιτρέποντας τον ορισμό της συμπεριφοράς (Κανόνες, Συνθήκες, Ενέργειες) ως δεδομένα υπό την μορφή αρχείων asset. Αυτό επιτρέπει τη δυναμική τροποποίηση της "νοημοσύνης" του ρομπότ χωρίς την ανάγκη αλλαγής του κώδικα.
2. **Διαχωρισμός Ευθυνών (Separation of Concerns):** Το σύστημα σχεδιάστηκε με σαφή διαχωρισμό μεταξύ των λειτουργικών μονάδων. Η λογική της πλοήγησης (RuleBasedController) είναι πλήρως αποσυνδεδεμένη από την υλοποίηση της κίνησης (RobotController) και την αντίληψη του περιβάλλοντος (SensorManager).
3. **Σχεδίαση βάσει Συμβολαίων (Design by Contract):** Η χρήση Διεπαφών (Interfaces), όπως η IRobotSensor, εξασφαλίζει ότι το σύστημα μπορεί να διαχειριστεί ομοιόμορφα διαφορετικούς τύπους αισθητήρων, καθιστώντας το σύστημα ανοιχτό σε μελλοντικές επεκτάσεις.

6.1.2 Λειτουργικά Υποσυστήματα

Για τους σκοπούς της ανάλυσης, η αρχιτεκτονική του συστήματος διακρίνεται σε τέσσερα κύρια λειτουργικά υποσυστήματα, τα οποία θα αναλυθούν διεξοδικά στις επόμενες ενότητες:

- **Υποσύστημα Αντίληψης:** Περιλαμβάνει το σύνολο των εικονικών αισθητήρων και τον μηχανισμό συλλογής δεδομένων από το περιβάλλον προσομοίωσης.

- **Υποσύστημα Λήψης Αποφάσεων:** Αποτελεί τον εγκέφαλο του ψηφιακού ρομπότ, αξιολογώντας τα δεδομένα των αισθητήρων και επιλέγοντας την κατάλληλη ενέργεια βάσει προκαθορισμένων κανόνων.
- **Υποσύστημα Εκτέλεσης και Κίνησης:** Διαχειρίζεται τη φυσική υπόσταση του ρομπότ, την κινηματική του και την αλληλεπίδραση με τα αντικείμενα της πίστας (συγκρούσεις).
- **Υποσύστημα Διασύνδεσης Φυσικού Κόσμου:** Περιλαμβάνει τη διαχείριση της Επαυξημένης Πραγματικότητας (AR Foundation) για την τοποθέτηση στον χώρο, καθώς και την επικοινωνία Bluetooth Low Energy (BLE) για τον έλεγχο του πραγματικού ρομπότ.

Η ανάλυση που ακολουθεί εστιάζει στον τρόπο με τον οποίο τα παραπάνω υποσυστήματα υλοποιήθηκαν, τις προκλήσεις που αντιμετωπίστηκαν και τις λύσεις που δόθηκαν μέσω του κώδικα C#.

6.2 Υποσύστημα Αντίληψης

Το Υποσύστημα Αντίληψης είναι υπεύθυνο για τη συλλογή πληροφοριών από το εικονικό περιβάλλον. Σε αντίθεση με τα πραγματικά ρομπότ που επεξεργάζονται θορυβώδη αναλογικά σήματα, η προσομοίωση στη Unity παρέχει πρόσβαση σε "τέλεια" γεωμετρικά δεδομένα. Ωστόσο, για να διατηρηθεί ο ρεαλισμός, η υλοποίηση των αισθητήρων σχεδιάστηκε ώστε να μιμείται τους περιορισμούς (εμβέλεια, οπτικό πεδίο, ρυθμός ανανέωσης) των πραγματικών συσκευών που αναλύθηκαν στο Κεφάλαιο 3.

6.2.1 Αρχιτεκτονική Σχεδίαση και Διεπαφές

Η βασική πρόκληση κατά τον σχεδιασμό του υποσυστήματος ήταν η ανομοιογένεια των αισθητήρων. Ένα ρομπότ μπορεί να διαθέτει αισθητήρες που λειτουργούν με εντελώς διαφορετικές αρχές, όπως η εκπομπή ακτίνων (Lidar), η ανίχνευση φυσικής επαφής (Bump Sensor) ή ο υπολογισμός πεδίων (Temperature Sensor). Η απευθείας ενσωμάτωση της λογικής κάθε αισθητήρα στον ελεγκτή του ρομπότ θα οδηγούσε σε έναν κώδικα δύσκαμπτο και δύσκολο στη συντήρηση.

Για την αντιμετώπιση αυτού του προβλήματος, υιοθετήθηκε το πρότυπο σχεδίασης **Strategy Pattern**. Μέσω αυτού, ορίζεται μια οικογένεια αλγορίθμων (οι αισθητήρες), οι οποίοι ενθυλακώνονται σε ξεχωριστές κλάσεις και καθίστανται εναλλάξιμοι μέσω μιας κοινής διεπαφής.

6.2.1.1 Η Διεπαφή IRobotSensor

Η καρδιά της αρχιτεκτονικής είναι η διεπαφή IRobotSensor. Αυτή ορίζει το "συμβόλαιο" που πρέπει να τηρεί κάθε αισθητήρας, ανεξάρτητα από την εσωτερική του λειτουργία. Οι βασικές μέθοδοι που ορίζει είναι:

- Initialize(Transform, LayerMask, float): Η μέθοδος αρχικοποίησης. Ο αισθητήρας λαμβάνει αναφορά στη θέση του ρομπότ, τις μάσκες επιπέδων (Layers) που πρέπει να ανιχνεύει ή να αγνοεί, και τη μέγιστη εμβέλειά του.

- UpdateSensor(): Η μέθοδος που καλείται σε κάθε βήμα της προσομοίωσης. Εδώ εκτελείται ο αλγόριθμος ανίχνευσης (π.χ. Raycasting, έλεγχος Triggers).

- IsObstacleDetected(): Επιστρέφει μια λογική τιμή (true/false) για το αν εντοπίστηκε εμπόδιο, παρέχοντας μια κοινή γλώσσα επικοινωνίας για όλους τους αισθητήρες.

- GetNearestObstacleDistance(): Επιστρέφει την απόσταση από το πλησιέστερο εμπόδιο. Για αισθητήρες επαφής η τιμή είναι μηδενική, ενώ για αισθητήρες απόστασης (Lidar, Sonar) είναι μεταβλητή.

- DrawGizmos(): Μέθοδος αποκλειστικά για χρήση εντός του Unity Editor, η οποία οπτικοποιεί τις ακτίνες και τα σημεία ανίχνευσης, διευκολύνοντας την αποσφαλμάτωση.

6.2.1.2 Ο Διαχειριστής Αισθητήρων (SensorManager)

Η κλάση SensorManager λειτουργεί ως ο κεντρικός κόμβος (Context) του υποσυστήματος. Κληρονομεί από την κλάση MonoBehaviour της Unity, γεγονός που της επιτρέπει να προσαρτηθεί στο GameObject του ρομπότ και να αλληλεπιδρά με τον κύκλο ζωής της μηχανής παιχνιδιού.

Οι αρμοδιότητες του SensorManager περιλαμβάνουν:

1. **Δημιουργία Στιγμιότυπων (Instantiation):** Κατά την εκκίνηση (Start), δημιουργεί τα αντικείμενα των συγκεκριμένων αισθητήρων (LidarSensor, SonarSensor, κ.λπ.) και καλεί τη μέθοδο Initialize για τον καθένα, περνώντας τις παραμέτρους που έχει ορίσει ο χρήστης στον Inspector (π.χ. lidarRange, sonarFieldOfView).

2. **Κύκλος Ενημέρωσης:** Διαθέτει τη μέθοδο UpdateAllSensors(), η οποία καλείται από τον ελεγκτή φυσικής του ρομπότ (FixedUpdate). Η μέθοδος αυτή διατρέχει όλους τους ενεργούς αισθητήρες και εκτελεί τη μέθοδο UpdateSensor() του καθενός, διασφαλίζοντας ότι τα δεδομένα είναι πάντα συγχρονισμένα με την τρέχουσα θέση του ρομπότ.

3. **Αφαίρεση Πολυπλοκότητας (Abstraction):** Εκθέτει τα δεδομένα των αισθητήρων μέσω απλών ιδιοτήτων (Properties), όπως IsForwardBlocked ή LidarDistances. Με αυτόν τον τρόπο, τα υπόλοιπα υποσυστήματα (π.χ. Λήψη Αποφάσεων) δεν χρειάζεται να γνωρίζουν πώς συλλέχθηκαν τα δεδομένα, παρά μόνο την τελική τους τιμή.

Μέσω αυτής της αρχιτεκτονικής, η προσθήκη ενός νέου αισθητήρα (π.χ. Radar) απαιτεί απλώς τη δημιουργία μιας νέας κλάσης που υλοποιεί το IRobotSensor και την προσθήκη μιας αναφοράς στον SensorManager, χωρίς να επηρεάζεται ο υπόλοιπος κώδικας του συστήματος.

6.2.2 Αισθητήρας LiDAR (Light Detection and Ranging)

Ο αισθητήρας LiDAR αποτελεί το κύριο μέσο χαρτογράφησης του χώρου για το ρομπότ. Στον φυσικό κόσμο, ένα LiDAR περιστρέφεται ταχύτατα, λαμβάνοντας μετρήσεις απόστασης σε διαδοχικές χρονικές στιγμές. Στην προσομοίωση, αξιοποιήθηκε η δυνατότητα της μηχανής Unity να εκτελεί υπολογισμούς ακαριαία, επιτρέποντας τη λήψη μιας πλήρους σάρωσης 360 μοιρών σε κάθε καρέ της φυσικής προσομοίωσης μέσω της μεθόδου FixedUpdate() της Unity.

6.2.2.1 Αλγόριθμος Σάρωσης

Η κλάση LidarSensor διατηρεί έναν πίνακα πραγματικών αριθμών (float[] distances) με μέγεθος ίσο με την ανάλυση σάρωσης (scanResolution), η οποία ορίστηκε στις 360 ακτίνες (ανάλυση 1 μοίρας). Κατά την εκτέλεση της μεθόδου UpdateSensor():

1. **Βρόχος Σάρωσης:** Εκτελείται μια επανάληψη για κάθε δείκτη του πίνακα.
2. **Υπολογισμός Κατεύθυνσης:** Για κάθε δείκτη i, υπολογίζεται η γωνία θ σε σχέση με τον τοπικό άξονα του ρομπότ. Το διάνυσμα κατεύθυνσης προκύπτει από την περιστροφή του διανύσματος transform.forward κατά θ μοίρες γύρω από τον κατακόρυφο άξονα (transform.up).
3. **Εκπομπή Ακτίνας (Raycasting):** Εκτελείται η εντολή Physics.Raycast από το κέντρο του ρομπότ προς την υπολογισμένη κατεύθυνση, με μέγιστη εμβέλεια sensorRange.
4. **Αποθήκευση Δεδομένων:**
    - Εάν η ακτίνα προσκρούσει σε αντικείμενο που ανήκει στη μάσκα εμποδίων (obstacleMask), η απόσταση αποθηκεύεται στον πίνακα distances[i].
    - Εάν δεν υπάρξει πρόσκρουση, αποθηκεύεται η τιμή του θετικού άπειρου (float.PositiveInfinity), υποδηλώνοντας ελεύθερο χώρο.

Τα δεδομένα αυτά τροφοδοτούν άμεσα τον αλγόριθμο "Turn To Best Path", ο οποίος αναλύει τα τεταρτημόρια του πίνακα για να αποφασίσει την κατεύθυνση διαφυγής.

6.2.3 Αισθητήρας Υπερήχων (Sonar)

Η προσομοίωση του αισθητήρα υπερήχων παρουσιάζει δύο ιδιαιτερότητες σε σχέση με το LiDAR: την κωνική διάδοση του ηχητικού κύματος και τη μηχανική κίνηση του αισθητήρα (Turret).

6.2.3.1 Προσομοίωση Ηχητικού Κώνου

Μια απλή ακτίνα (Raycast) δεν επαρκεί για την προσομοίωση των υπερήχων, καθώς ο ήχος διαδίδεται σε κώνο. Ένα λεπτό αντικείμενο θα μπορούσε να "ξεφύγει" από μια μοναδική ακτίνα, ενώ στην πραγματικότητα θα επέστρεφε ηχώ. Για να λυθεί αυτό, η κλάση SonarSensor υλοποιεί μια τεχνική "Multicasting":

- Αντί για μία ακτίνα, εκπέμπεται μια δέσμη ακτινών (π.χ. numberOfRays = 9) κατανεμημένων ισομερώς εντός του οπτικού πεδίου (fieldOfView, π.χ. 45 μοίρες).
- Εάν οποιαδήποτε από αυτές τις ακτίνες ανιχνεύσει εμπόδιο, ο αισθητήρας θεωρεί ότι υπάρχει αντικείμενο στον κώνο και επιστρέφει την απόσταση της πλησιέστερης πρόσκρουσης.

Μηχανισμός Περιστροφής (Turret Mechanism) Ο αισθητήρας δεν είναι στατικός. Η κλάση διαχειρίζεται ένα ξεχωριστό Transform (turretTransform), το οποίο αναπαριστά τον σερβοκινητήρα.

- Σε κάθε κύκλο ενημέρωσης, το turretTransform περιστρέφεται κατά έναν βήμα γωνίας που καθορίζεται από την ταχύτητα σάρωσης (scanSpeed).
- Εφαρμόζεται λογική "Ping-Pong": Όταν η γωνία περιστροφής υπερβεί το όριο σάρωσης (scanArc), η φορά της κίνησης αντιστρέφεται.

- Οι ακτίνες ανίχνευσης εκπέμπονται πάντα σχετικές με την τρέχουσα περιστροφή του turretTransform, επιτρέποντας στο ρομπότ να "σαρώνει" δυναμικά τον χώρο μπροστά του, όπως ακριβώς και το πραγματικό όχημα.

6.2.4 Αισθητήρας Χρώματος (Color Sensor)

Ο αισθητήρας αυτός προσομοιώνει έναν φωτοανιχνευτή στραμμένο προς το έδαφος. Η ανίχνευση χρώματος σε ένα τρισδιάστατο περιβάλλον απαιτεί πρόσβαση στην υφή (Texture) του αντικειμένου και όχι απλώς στη γεωμετρία του. Η κλάση ColorSensor ακολουθεί την εξής διαδικασία:

1. Εκτελεί Raycast κάθετα προς τα κάτω (Vector3.down).

2. Από το σημείο πρόσκρουσης (RaycastHit), ανακτά τις συντεταγμένες υφής (UV coordinates). Αυτές οι συντεταγμένες (τιμές 0.0 έως 1.0) αντιστοιχούν στη θέση του χτυπήματος πάνω στην εικόνα που "ντύνει" το δάπεδο.

3. Χρησιμοποιεί τη μέθοδο Texture2D.GetPixelBilinear(u, v) για να διαβάσει το χρώμα του pixel σε εκείνο το σημείο. Η χρήση διγραμμικής παρεμβολής (bilinear interpolation) εξασφαλίζει ότι το χρώμα θα είναι ομαλό ακόμα και αν η ανάλυση της εικόνας είναι χαμηλή.

6.2.5 Αισθητήρας Υπερύθρων (IR - Infrared Sensor)

Στον πραγματικό κόσμο, οι αισθητήρες IR χρησιμοποιούνται για ανίχνευση εμποδίων μικρής εμβέλειας και συχνά έχουν στενή δέσμη εκπομπής.

Η κλάση InfraredSensor δεν αρκείται σε μία μοναδική ακτίνα, η οποία θα μπορούσε εύκολα να "χάσει" γωνίες αντικειμένων. Αντ' αυτού, υλοποιεί μια "στενή δέσμη" (narrow beam):

- Εκπέμπεται ένα σύνολο 5 έως 10 ακτινών (Raycasts) εντός ενός πολύ μικρού οπτικού πεδίου (περίπου 20°).

- **Διαχωρισμός Αντικειμένων:** Ο αισθητήρας έχει ρυθμιστεί ώστε να ελέγχει δύο διαφορετικές μάσκες συγκρούσεων (LayerMasks):
    1. obstacleMask: Για κοινά εμπόδια (τοίχοι).
    2. POIMask (Point of Interest): Για στόχους που το ρομπότ πρέπει να προσεγγίσει. Σε αντίθεση με το Lidar που επιστρέφει απλώς αποστάσεις, ο αισθητήρας IR επιστρέφει και την αναφορά στο GameObject που χτυπήθηκε (nearestObstacleObject). Αυτό επιτρέπει στο σύστημα λήψης αποφάσεων να αναγνωρίζει αν το αντικείμενο μπροστά είναι εμπόδιο προς αποφυγή ή στόχος προς προσέγγιση.

6.2.6 Αισθητήρας Επαφής (Bump Sensor)

Ο αισθητήρας επαφής είναι ο "μηχανισμός ασφαλείας" του ρομπότ, ενεργοποιούμενος μόνο όταν αποτύχουν οι αισθητήρες απόστασης και υπάρξει φυσική σύγκρουση.

Η προσομοίωση φυσικής επαφής δεν μπορεί να βασιστεί σε Raycasts, καθώς αυτά ανιχνεύουν την απόσταση *πριν* την επαφή. Η υλοποίηση εδώ είναι **γεγονοστραφής (event-driven)** και υβριδική:

- **Physics Component:** Ένα βοηθητικό script (BumpSensorTriggerHandler) είναι προσαρτημένο σε έναν Collider που περιβάλλει το ρομπότ και έχει ρυθμιστεί ως "Trigger". Η μηχανή φυσικής της Unity καλεί αυτόματα τη μέθοδο OnTriggerEnter όταν υπάρξει διείσδυση άλλου αντικειμένου.

- **Logic Component:** Η κλάση BumpSensor συνδέεται με τον BumpSensorTriggerHandler μέσω C# Delegates. Όταν ο Handler ανιχνεύσει σύγκρουση, ενημερώνει τον BumpSensor, ο οποίος αλλάζει την κατάστασή του (isBumped = true).

Αυτή η διαστρωμάτωση επιτρέπει στον SensorManager να αντιμετωπίζει τον αισθητήρα επαφής όπως όλους τους άλλους (IRobotSensor), παρόλο που ο μηχανισμός λειτουργίας του είναι ριζικά διαφορετικός (Physics Events αντί για Raycasting).

6.2.7 Αισθητήρας Ραντάρ

Το ραντάρ χρησιμοποιείται για τον εντοπισμό στόχων σε μεγάλες αποστάσεις. Στην πραγματικότητα, τα ραδιοκύματα έχουν την ιδιότητα να διαπερνούν ορισμένα υλικά ή να ανακλώνται έντονα από μεταλλικούς στόχους.

Η κλάση RadarSensor διαφοροποιείται από το LiDAR σε δύο βασικά σημεία:

1. **Χρονική Υστέρηση (Time Slicing):** Ενώ το LiDAR σαρώνει σε κάθε καρέ, το ραντάρ προσομοιώνει τον χρόνο περιστροφής της κεραίας μέσω της παραμέτρου scanInterval. Η μέθοδος UpdateSensor εκτελεί σάρωση μόνο όταν παρέλθει ο καθορισμένος χρόνος (π.χ. κάθε 1.5 δευτερόλεπτο), εισάγοντας ρεαλιστική καθυστέρηση στην ενημέρωση των δεδομένων.

2. **Σημασιολογικό Φιλτράρισμα (Semantic Filtering):** Για να προσομοιωθεί η ιδιότητα του ραντάρ να "αγνοεί" εμπόδια (όπως ξύλινους τοίχους) και να εντοπίζει συγκεκριμένους στόχους, χρησιμοποιείται το σύστημα ετικετών (Tags) της Unity. Ο αισθητήρας εκπέμπει ακτίνες, αλλά αποθηκεύει στη λίστα detectedTargets μόνο τα αντικείμενα που φέρουν την ετικέτα radarTargetTag (π.χ. "RadarPOI"). Αντικείμενα χωρίς αυτή την ετικέτα αγνοούνται, ακόμη και αν βρίσκονται στην πορεία της ακτίνας.

6.2.8 Αισθητήρας Θερμοκρασίας

Η προσομοίωση θερμότητας αποτελεί ιδιαίτερη περίπτωση, καθώς η θερμότητα δεν είναι γεωμετρικό χαρακτηριστικό που μπορεί να ανιχνευθεί με Raycast.

Η προσέγγιση εδώ είναι καθαρά υπολογιστική/μαθηματική.

1. Ο αισθητήρας (TemperatureSensor) εντοπίζει όλα τα αντικείμενα στη σκηνή που φέρουν το συστατικό HeatSource.
2. Για κάθε πηγή θερμότητας, υπολογίζει την απόσταση d από το ρομπότ.
3. Εφαρμόζει τον νόμο του αντιστρόφου τετραγώνου (Inverse Square Law) για να υπολογίσει τη συνεισφορά της πηγής στη θέση του ρομπότ, βάσει του τύπου:

$$T_{robot} = T_{ambient} + \sum(T_{sourceMax} \bullet (1 - \frac{d}{R})^{P})$$

όπου R είναι η εμβέλεια της πηγής και p ο εκθέτης εξασθένησης (falloffExponent). Αυτό επιτρέπει τη δημιουργία "θερμικών πεδίων" (gradients) στον χώρο, τα οποία το ρομπότ μπορεί να ακολουθήσει (Heat Seeking behavior).

6.2.9 Συμπεράσματα Υποσυστήματος Αντίληψης

Η σχεδίαση του υποσυστήματος αντίληψης πέτυχε την πλήρη αποσύνδεση της λογικής του ρομπότ από την υλοποίηση των αισθητήρων. Μέσω του SensorManager και της διεπαφής IRobotSensor, το σύστημα είναι σε θέση να διαχειριστεί ταυτόχρονα και αποδοτικά επτά διαφορετικούς τύπους αισθητήρων, παρέχοντας πλούσια δεδομένα (αποστάσεις, χρώματα, θερμοκρασίες, επαφές) για τη λήψη αποφάσεων.

6.3 Υποσύστημα Λήψης Αποφάσεων

Ενώ το υποσύστημα αντίληψης παρέχει την πληροφορία για το περιβάλλον, το υποσύστημα λήψης αποφάσεων είναι υπεύθυνο για την αξιολόγηση αυτής της πληροφορίας και την επιλογή της κατάλληλης ενέργειας. Αντί για την υιοθέτηση μιας μονολιθικής προσέγγισης με πολύπλοκες δομές ελέγχου if-then-else ή switch-case εντός του κώδικα, σχεδιάστηκε μια ευέλικτη, αρθρωτή αρχιτεκτονική βασισμένη σε κανόνες.

6.3.1 Ανάλυση Αρχικής Υλοποίησης: Το Πρόβλημα της Μονολιθικής Αρχιτεκτονικής

Αρχικά, η συμπεριφορά του ρομπότ υλοποιήθηκε αποκλειστικά εντός της κλάσης RobotController. Η κλάση αυτή ήταν επιφορτισμένη με αντικρουόμενες αρμοδιότητες:

1. **Διαχείριση Κίνησης:** Έλεγχος του Rigidbody του ρομπότ και της φυσικής.
2. **Λογική Πλοήγησης:** Έλεγχος εμποδίων, αναζήτηση στόχων.
3. **Διαχείριση Κατάστασης:** Μια σύνθετη απαρίθμηση (enum RobotState) με πάνω από 15 καταστάσεις (MovingToRadarTarget, HeatSeeking, PausedAfterPointOfInterestReach, κ.λπ.).
4. **Επικοινωνία:** Απευθείας κλήσεις στον διαχειριστή επικοινωνίας BLE και στο UI.

Αυτή η προσέγγιση οδήγησε σε μια συνάρτηση FixedUpdate εξαιρετικά υψηλής πολυπλοκότητας, γεμάτη εμφωλευμένες εντολές ελέγχου. Κάθε φορά που έπρεπε να προστεθεί μια νέα συμπεριφορά όπως "Αν δεις κίτρινο πάτωμα, σταμάτα", απαιτούνταν η τροποποίηση του κεντρικού βρόχου ελέγχου, αυξάνοντας τον κίνδυνο εμφάνισης σφαλμάτων σε άλλες, άσχετες λειτουργίες.

Κατέστη σαφές κατά την διάρκεια ανάπτυξης της εφαρμογής ότι για να είναι το σύστημα επεκτάσιμο, έπρεπε να διαχωριστεί το **"Τι"** πρέπει να κάνει το ρομπότ (Λήψη Απόφασης) από το **"Πώς"** το κάνει (Εκτέλεση).

6.3.2 Αρχιτεκτονική Βασισμένη σε Δεδομένα

Για την επίλυση των παραπάνω προβλημάτων, σχεδιάστηκε μια νέα αρχιτεκτονική όπου η συμπεριφορά δεν ορίζεται από κώδικα, αλλά από **δεδομένα**. Αξιοποιήθηκαν τα ScriptableObjects της Unity για τη δημιουργία ενός συστήματος όπου οι κανόνες συμπεριφοράς είναι Assets και όχι εντολές C#.

Η νέα αρχιτεκτονική αποτελείται από τρία επίπεδα:

1. **Συνθήκη:** Η ερώτηση προς το περιβάλλον.
2. **Ενέργεια:** Η εντολή προς το ρομπότ.
3. **Κανόνας:** Η σύνδεση "Αν Συνθήκη, τότε Ενέργεια".

Στο επίκεντρο του υποσυστήματος επίλυσης αποφάσεων βρίσκεται ο εγκέφαλος τους συστήματος, η κλάση **RuleBasedController.** Κληρονομεί από το MonoBehaviour και αντικαθιστά την παλιά, μονολιθική λογική του **RobotController**. Πλέον, ο ρόλος του ελεγκτή δεν είναι να αποφασίζει τι θα κάνει με προκαθορισμένες μεθόδους, αλλά να κρατάει μία δυναμική λίστα από ορισμένους κανόνες τους οποίους συνεχώς ελέγχει και εκτελεί.

6.3.2.1 Συνθήκες (Conditions) ως Αφαιρετικές Μονάδες

Ενώ στην αρχική υλοποίηση ο έλεγχος για την συμπεριφορά του ρομπότ γινόταν με περίπλοκες αλυσίδες συνθηκών, η υλοποίηση συνθηκών ως Unity Assets επέτρεψε την ενθυλάκωση της λογικής σε αυτόνομες κλάσεις που λειτουργούν ανεξάρτητα.

Η βάση του συστήματος είναι η αφηρημένη κλάση **Condition**, η οποία κληρονομεί από την **ScriptableObject**. Αυτό καθιστά κάθε συνθήκη ένα αυτόνομο αρχείο στο Unity Project, το οποίο μπορεί να ρυθμιστεί μέσω του Inspector και να επαναχρησιμοποιηθεί σε πολλαπλούς κανόνες.

Η κλάση περιέχει μία αφηρημένη μέθοδο **Evaluate(RobotController robot)**. Η μέθοδος Evaluate λαμβάνει ως παράμετρο το στιγμιότυπο του ρομπότ και επιστρέφει true ή false. Μέσω του μηχανισμού της κληρονομικότητας, δημιουργήθηκε μια εκτενής βιβλιοθήκη εξειδικευμένων συνθηκών που καλύπτουν κάθε πτυχή της αντίληψης του ρομπότ.

Ακολουθεί η κατηγοριοποίηση και ανάλυση των υλοποιημένων συνθηκών:

Α. Συνθήκες Ανίχνευσης Εμποδίων & Ασφάλειας

Αυτές οι συνθήκες σχετίζονται με την ακεραιότητα του ρομπότ και την άμεση αντίδραση σε φυσικά εμπόδια.

- **IsForwardBlockedCondition**: Ελέγχει τον αισθητήρα Υπερύθρων (IR) για την ύπαρξη εμποδίου σε κοντινή απόσταση. Αποτελεί τη θεμελιώδη συνθήκη για την ενεργοποίηση συμπεριφορών αποφυγής. Διαθέτει παράμετρο ignoreTag, ώστε να μπορεί να ρυθμιστεί να αγνοεί συγκεκριμένα αντικείμενα (π.χ. στόχους), εστιάζοντας μόνο σε τοίχους και εμπόδια.
- **IsBumpedCondition**: Ελέγχει τον αισθητήρα επαφής (BumpSensor). Αυτή η συνθήκη επιστρέφει true μόνο όταν το ρομπότ έχει ήδη προσκρούσει φυσικά σε αντικείμενο, λειτουργώντας ως μηχανισμός ασφαλείας όταν αποτύχουν οι αισθητήρες απόστασης.

Β. Συνθήκες Οπτικής Αντίληψης (Χρώμα)

Χρησιμοποιούνται για την πλοήγηση βάσει σημάνσεων στο δάπεδο ή για την αναγνώριση ιδιοτήτων των εμποδίων, με χρήση του αισθητήρα χρώματος.

- **IsFloorColorCondition**: Ελέγχει το χρώμα του δαπέδου ακριβώς κάτω από το ρομπότ. Συγκρίνει το χρώμα που επιστρέφει ο SensorManager με ένα χρώμα-στόχο (targetColor),

χρησιμοποιώντας μια ανοχή 15% για να αντιμετωπίσει τις μικρές αποκλίσεις στις τιμές RGB λόγω φωτισμού. Χρησιμοποιείται για κανόνες όπως "Αν δεις κόκκινο, προχώρα".

- **IsObstacleColorCondition**: Σε περίπτωση που υπάρχει εμπόδιο μπροστά, ελέγχει το χρώμα του. Αυτό επιτρέπει στο ρομπότ να λαμβάνει σημασιολογικές αποφάσεις, όπως το να στρίβει δεξιά σε μπλε εμπόδια και αριστερά σε πράσινα.

Γ. Συνθήκες Εντοπισμού Στόχων

Αυτές οι συνθήκες είναι πιο σύνθετες, καθώς αλληλεπιδρούν με τα συστήματα μνήμης του ρομπότ (TargetMemory) για να διασφαλίσουν ότι δεν εντοπίζονται συνεχώς οι ίδιοι στόχοι.

- **IsTargetReached**: Ελέγχει αν το αντικείμενο μπροστά από τον αισθητήρα υπερύθρων φέρει την ετικέτα "PointOfInterest". Πριν επιστρέψει θετικό αποτέλεσμα, συμβουλεύεται τη μνήμη για να βεβαιωθεί ότι ο συγκεκριμένος στόχος δεν έχει επισκεφτεί πρόσφατα.
- **IsSonarTargetVisibleCondition**: Ελέγχει αν το περιστρεφόμενο Sonar έχει εντοπίσει αντικείμενο με συγκεκριμένο Tag.
- **IsRadarTargetVisibleCondition**: Ελέγχει τη λίστα στόχων του αισθητήρα Ραντάρ. Επιστρέφει true εάν υπάρχει έστω και ένας στόχος στη λίστα που δεν έχει χαρακτηριστεί ως "να αγνοείται" από τη μνήμη.

Δ. Συνθήκες Περιβάλλοντος & Πλοήγησης

Αφορούν την αντίληψη μη-γεωμετρικών μεγεθών ή σύνθετων χωρικών δεδομένων.

- **IsTemperatureRisingCondition** και **IsTemperatureFallingCondition**: Ελέγχουν την τάση της θερμοκρασίας αντλώντας δεδομένα από το **HeatMemory**, μία κλάση που ορίζει πόσο συχνά λαμβάνεται δείγμα θερμοκρασίας και αν η θερμοκρασία ακολουθεί ανοδική ή καθοδική τάση. Είναι απαραίτητες για τον αλγόριθμο αναζήτησης πηγής θερμότητας, επιτρέποντας στο ρομπότ να κατανοεί αν οι κινήσεις του το φέρνουν πιο κοντά ή πιο μακριά από την πηγή.
- **IsOpenPathInDirectionCondition**: Χρησιμοποιεί τα δεδομένα του LiDAR για να ελέγξει αν υπάρχει ελεύθερος χώρος σε συγκεκριμένη κατεύθυνση (Μπροστά, Αριστερά, Δεξιά, Πίσω). Ελέγχει ένα τόξο (checkArc, π.χ. 90°) και επιστρέφει true μόνο αν όλες οι ακτίνες σε αυτό το τόξο δείχνουν απόσταση μεγαλύτερη από το όριο ασφαλείας.

Ε. Βοηθητικές Συνθήκες

- **AlwaysTrueCondition**: Μια ειδική συνθήκη που επιστρέφει πάντα true. Χρησιμοποιείται ως ο τελευταίος κανόνας στη λίστα προτεραιοτήτων (Default Rule), καθορίζοντας την προεπιλεγμένη συμπεριφορά του ρομπότ (π.χ. "Προχώρα Μπροστά") όταν καμία άλλη συνθήκη δεν ικανοποιείται.

6.3.2.2 Σύστημα Ενεργειών

Εφόσον οι συνθήκες καθορίσουν την τρέχουσα κατάσταση του περιβάλλοντος, το επόμενο βήμα στον κύκλο ελέγχου είναι η εκτέλεση μιας ενέργειας. Σε αντίθεση με την παραδοσιακή προσέγγιση όπου οι

ενέργειες είναι απλές μέθοδοι, στη νέα αρχιτεκτονική οι ενέργειες υλοποιήθηκαν ως αυτόνομες κλάσεις.

Η κλάση RobotAction αποτελεί τη βάση για όλες τις ενέργειες. Κληρονομεί επίσης από το ScriptableObject, επιτρέποντας στον χρήστη να δημιουργεί "προκαθορισμένες κινήσεις" (π.χ. "Στροφή 90°", "Στροφή 45°", "Κίνηση Μπροστά") ως Assets μέσα στον Editor της Unity. Περιέχει μία αφηρημένη μέθοδο Execute που δέχεται μία παράμετρο τύπου RobotController, που είναι το στιγμιότυπο του ρομπότ το οποίο διαθέτει τις μεθόδους που θα αξιοποιηθούν για την ουσιαστική εκτέλεση των ενεργειών του ρομπότ, και μία παράμετρο onComplete τύπου Action.

Η σημαντικότερη σχεδιαστική απόφαση εδώ είναι η εισαγωγή της παραμέτρου Action onComplete που χρησιμοποιεί τον μηχανισμό callback που αναλύθηκε στην ενότητα 5.2.3, καθώς επιτρέπει στο σύστημα να καταλαβαίνει πότε ολοκληρώθηκε η εκτέλεση, και να επιστρέφει στην κατάσταση όπου γίνεται αξιολόγηση κανόνων ενώ το ρομπότ βρίσκεται στην βασική κατάσταση ευθείας κίνησης με σκοπό την ανίχνευση εμποδίων ή αντικειμένων ενδιαφέροντος.

Για να επιτευχθεί η ομαλή παρουσίαση της κίνησης του ρομπότ, η εκκίνηση και οι στροφές απαιτούν ένα μικρό χρονικό διάστημα για να ολοκληρωθούν.

- Όταν καλείται η Execute, η ενέργεια αναλαμβάνει τον έλεγχο του ρομπότ.
- Η κλάση RuleBasedController, που θα αναλυθεί στην επόμενη ενότητα, παύει να αξιολογεί νέους κανόνες.
- Μόλις η ενέργεια ολοκληρωθεί (π.χ. το ρομπότ έφτασε στην επιθυμητή γωνία), καλείται η μέθοδος onComplete().
- Ο ελεγκτής ενημερώνεται ότι η κίνηση τελείωσε και ξαναρχίζει τον κύκλο αξιολόγησης κανόνων.

Ακολουθεί η ανάλυση των συγκεκριμένων ενεργειών που υλοποιήθηκαν, αυτές οι ενέργειες αποτελούν τα θεμελιώδη δομικά στοιχεία της συμπεριφοράς του ρομπότ.

Α. Βασικές Ενέργειες Κίνησης: Αυτές οι ενέργειες αποτελούν τα θεμελιώδη δομικά στοιχεία της συμπεριφοράς του ρομπότ.

- **MoveForwardAction:**
  Η κλάση **MoveForwardAction** υλοποιεί την μέθοδο **Execute** με σκοπό την ευθεία κίνηση του ρομπότ, στην πραγματικότητα επιστρέφει το ρομπότ στην βασική κατάσταση του, οποία είναι να κινείται ευθεία με προκαθορισμένη ταχύτητα.

  Η συγκεκριμένη ενέργεια επιλέχθηκε και ως βασική κατάσταση καθώς επιτρέπει στο ρομπότ να προχωράει συνεχώς, επιστρέφοντας στον βρόχο αναζήτησης εμποδίων ή αντικειμένων ενδιαφέροντος καθώς οι αισθητήρες λαμβάνουν νέα δεδομένα.

- **TurnAction:**
  Η κλάση **TurnAction** υλοποιεί την μέθοδο **Execute** με σκοπό την περιστροφή του ρομπότ κατά συγκεκριμένο αριθμό μοιρών, θετικές για δεξιά περιστροφή και αρνητικές μοίρες για αριστερή περιστροφή.

Για εύκολη ρύθμιση της στροφής μέσω του Unity Inspector, περιέχει μία δημόσια μεταβλητή **angle** για την γωνία με βάση την οποία θα εκτελέσει περιστροφή, και ένα δημόσιο enum **direction** που καθορίζει αν θα στρίψει αριστερά, δεξιά ή θα πάρει τυχαία στροφή προς τις δύο κατευθύνσεις με πιθανότητα 50% για κάθε μία.

Υλοποιείται ως τρία Assets, **TurnLeftAction**, **TurnRightAction** και **TurnRandomAction**, για κάθε τιμή του **direction**, έτσι ώστε να μπορεί εύκολα να καθοριστεί είτε από τον χρήστη είτε ως προκαθορισμένη λειτουργία η κατεύθυνση που θα πάρει το ρομπότ όταν αληθεύση η αντίστοιχη συνθήκη.

Οι στροφές δεξιά ή αριστερά χρησιμοποιούνται για σηματοδότηση του ρομπότ ότι πρέπει να εκτελέσει συγκεκριμένη στροφή με βάση κάποια συνθήκη. Η τυχαία στροφή χρησιμοποιείται σε σενάρια όπου το ρομπότ έχει εγκλωβιστεί ή για να εισάγει τον τυχαίο παράγοντα στην περιπλάνηση, ώστε να καλύπτει μεγαλύτερο μέρος της πίστας.

- **PauseAction:**
  Η κλάση **PauseAction** υλοποιεί την μέθοδο **Execute** με σκοπό την πλήρη ακινητοποίηση του ρομπότ, μηδενίζοντας την γραμμική και γωνιακή ταχύτητα του RigidBody component που έχει προστεθεί στο ρομπότ.

Β. Ενέργειες Πλοήγησης & Ευφυΐας (Smart Navigation): Αυτές οι ενέργειες περιέχουν ενσωματωμένη λογική για την επεξεργασία δεδομένων αισθητήρων πριν την κίνηση.

- **TurnToBestPathAction**:
  Η ενέργεια TurnToBestPathAction υλοποιεί έναν αλγόριθμο λήψης αποφάσεων για την επιλογή της βέλτιστης κατεύθυνσης διαφυγής όταν το ρομπότ συναντήσει εμπόδιο. Σε αντίθεση με τις στατικές ενέργειες στροφής, αυτή η ενέργεια επεξεργάζεται δυναμικά το νέφος σημείων του LiDAR για να εντοπίσει τον ελεύθερο χώρο.

  Η λειτουργία της ενέργειας βασίζεται στη χωρική διαίρεση του περιβάλλοντος σε τομείς (Arc Sectors) και την αξιολόγηση της "βατότητας" του καθενός. Η διαδικασία περιλαμβάνει τα εξής βήματα:
  Λήψη Δεδομένων: Η ενέργεια ανακτά τον πίνακα αποστάσεων από τον SensorManager. Ο πίνακας αυτός περιέχει 360 μετρήσεις (μία ανά μοίρα).
  Τμηματοποίηση Χώρου (Segmentation): Ο αλγόριθμος υπολογίζει τη μέση απόσταση εμποδίων σε τέσσερα βασικά τόξα γύρω από το ρομπότ.
  - **Μπροστά (Front):** -45° έως +45° (χρησιμοποιείται για αναφορά).
  - **Δεξιά (Right):** 45° έως 135°.
  - **Πίσω (Back):** 135° έως 225°.
  - **Αριστερά (Left):** 225° έως 315°.

  Η χρήση της μέσης τιμής αντί της μέγιστης εξασφαλίζει ότι ο αλγόριθμος δεν θα ξεγελαστεί από μεμονωμένο κενό σε έναν τοίχο, αλλά θα αναζητήσει ουσιαστικά ανοιχτούς διαδρόμους.
  Εφαρμογή Προτίμησης Εξερεύνησης (Exploration Bias): Ένα κρίσιμο στοιχείο του αλγορίθμου είναι η εισαγωγή μιας ευριστικής παραμέτρου explorationBias για την αποφυγή

οπισθοδρόμησης. Στόχος είναι το ρομπότ να προτιμά νέες διαδρομές (στροφή αριστερά/δεξιά) έναντι της επιστροφής (στροφή 180°, πίσω), ώστε να εξερευνά τον χώρο αποτελεσματικότερα. Η βαθμολογία του πίσω τεταρτημορίου τιμωρείται πολλαπλασιαστικά:

$$Score_{Back} = AvgDistance_{Back} \times explorationBias$$

Όπου το explorationBias λαμβάνει τιμές στο διάστημα (0,1]. Μια τιμή 0.7 σημαίνει ότι ο πίσω δρόμος πρέπει να είναι τουλάχιστον 30% πιο ανοιχτός από τους πλάγιους για να επιλεγεί.

Λήψη Απόφασης & Εκτέλεση: Ο αλγόριθμος συγκρίνει τις βαθμολογίες των κατευθύνσεων **Δεξιά**, **Αριστερά** και **Πίσω** (η κατεύθυνση Μπροστά αγνοείται καθώς είναι ήδη μπλοκαρισμένη, γεγονός που προκάλεσε την κλήση της ενέργειας).

Εάν $AvgDistance_{Right} > AvgDistance_{Left}$ και $AvgDistance_{Right} > Score_{Back}$ τότε θα εκτελεστεί στροφή 90°.

Εάν $AvgDistance_{Left} > AvgDistance_{Right}$ και $AvgDistance_{Left} > Score_{Back}$ τότε θα εκτελεστεί στροφή -90°.

Διαφορετικά (δηλαδή αν τα πλάγια είναι κλειστά ή το πίσω μέρος είναι εξαιρετικά ανοιχτό) τότε θα εκτελεστεί στροφή 180°. Τέλος, καλείται η μέθοδος robot.PerformTurn() με την υπολογισμένη γωνία, και κατά την ολοκλήρωση της στροφής ενεργοποιείται το callback onComplete.

Γ. Ενέργειες Μνήμης & Διαχείρισης Στόχων: Ενέργειες που δεν αφορούν μόνο την κίνηση, αλλά και την ενημέρωση της εσωτερικής κατάστασης του ρομπότ.

Πριν την ανάλυση των ενεργειών που αφορούν την αλληλεπίδραση με τους στόχους, είναι απαραίτητη η περιγραφή του υποσυστήματος που υποστηρίζει αυτές τις αποφάσεις: το σύστημα μνήμης.

Το Σύστημα Μνήμης (TargetMemory): Η κλάση TargetMemory αποτελεί την κεντρική βάση δεδομένων του ρομπότ σχετικά με τους στόχους (Points of Interest) που έχει εντοπίσει ή επισκεφθεί. Υλοποιεί το πρότυπο σχεδίασης Singleton, διασφαλίζοντας ότι υπάρχει μόνο ένα στιγμιότυπο της κλάσης σε όλη τη διάρκεια της εκτέλεσης και ότι είναι προσβάσιμο από οποιαδήποτε Ενέργεια ή Συνθήκη χωρίς την ανάγκη πολύπλοκων συνδέσεων.

Η λειτουργία της στηρίζεται σε δύο βασικούς πυλώνες:

1. **Λίστα Αγνοημένων (Ignore List):** Χρησιμοποιεί μια δομή δεδομένων Dictionary<GameObject, float> για να χαρτογραφήσει αντικείμενα-στόχους με χρονικές στιγμές λήξης. Όταν ένας στόχος "ολοκληρωθεί", προστίθεται σε αυτή τη λίστα με διάρκεια αποκλεισμού. Η μέθοδος Update σαρώνει το λεξικό σε κάθε καρέ και αφαιρεί αυτόματα τους στόχους των οποίων ο χρόνος αποκλεισμού έχει παρέλθει, επιτρέποντας στο ρομπότ να τους ξαναδεί στο μέλλον.
2. **Μηχανισμός Καρφιτσώματος (Target Pinning):** Παρέχει τη δυνατότητα "κλειδώματος" ενός στόχου μέσω των μεταβλητών PinnedTargetPosition και HasPinnedTarget. Αυτό επιτρέπει στο ρομπότ να θυμάται πού βρισκόταν ένας στόχος ακόμα και αν αυτός βγει προσωρινά από το οπτικό πεδίο των αισθητήρων κατά τη διάρκεια μιας στροφής.

Ακολουθούν οι ενέργειες που αξιοποιούν το παραπάνω σύστημα για την υλοποίηση της συμπεριφοράς στόχευσης.

PinVisibleTargetAction (Sonar): Η ενέργεια PinVisibleTargetAction είναι υπεύθυνη για την "αγκίστρωση" (locking) ενός στόχου που εντοπίστηκε από τον αισθητήρα υπερήχων (Sonar). Ενώ ο αισθητήρας Sonar σαρώνει συνεχώς τον χώρο, η συγκεκριμένη ενέργεια μετατρέπει την προσωρινή στιγμιαία ανίχνευση σε ενεργό στόχο πλοήγησης.

Η λειτουργία της περιλαμβάνει τα εξής στάδια:

1. **Ανάκτηση Δεδομένων:** Αντλεί το τελευταίο αντικείμενο που χτύπησε η δέσμη του Sonar (LastSonarHitObject) μέσω του SensorManager.
2. **Έλεγχος Μνήμης (Filtering):** Πριν αποδεχτεί το αντικείμενο ως στόχο, επικοινωνεί με το TargetMemory. Καλεί τη μέθοδο IsIgnored() για να ελέγξει αν το συγκεκριμένο αντικείμενο βρίσκεται σε περίοδο Cooldown. Αυτό είναι κρίσιμο για την αποφυγή εγκλωβισμού σε βρόχο (loop), όπου το ρομπότ θα πήγαινε συνέχεια στον ίδιο στόχο.
3. **Καταχώρηση Στόχου:** Εφόσον ο στόχος είναι έγκυρος και δεν αγνοείται, η ενέργεια ενημερώνει την μεταβλητή currentTargetPOI στον RobotController.
4. **Μετάβαση Κατάστασης:** Η ενέργεια αυτή δεν προκαλεί φυσική κίνηση, αλλά αλλάζει την εσωτερική κατάσταση του ελεγκτή, ώστε στον επόμενο κύκλο FixedUpdate να ενεργοποιηθεί η λογική προσέγγισης (MovingToTarget).

PinRadarTargetAction (Radar): Η ενέργεια PinRadarTargetAction λειτουργεί συμπληρωματικά με τον αισθητήρα Ραντάρ για τον εντοπισμό στόχων σε μεγάλες αποστάσεις ή πίσω από εμπόδια. Σε αντίθεση με το Sonar που εστιάζει σε έναν στόχο τη φορά, το Ραντάρ επιστρέφει λίστα πολλαπλών πιθανών στόχων.

Ο αλγόριθμος της ενέργειας λειτουργεί ως εξής:

1. **Σάρωση Λίστας Στόχων:** Ανακτά τη λίστα RadarDetectedTargets από τον SensorManager.
2. **Επιλογή Βέλτιστου Στόχου:** Διατρέχει τη λίστα των εντοπισμένων στόχων. Για κάθε υποψήφιο στόχο, ελέγχει το TargetMemory. Ο πρώτος στόχος που εντοπίζεται και δεν περιλαμβάνεται στη λίστα αγνοημένων, επιλέγεται ως ο νέος ενεργός στόχος. Αυτή η διαδικασία διασφαλίζει ότι το ρομπότ θα αγνοήσει στόχους που έχει ήδη επισκεφτεί, εστιάζοντας αυτόματα στον επόμενο διαθέσιμο.
3. **Ανάθεση Στόχου:** Ο επιλεγμένος στόχος ανατίθεται στη μεταβλητή currentTargetRadarPOI του RobotController.

TurnToTargetAngleAction: Η ενέργεια TurnToTargetAngleAction αποτελεί μια εξειδικευμένη παραλλαγή στροφής που δεν βασίζεται σε τυχαία επιλογή ή σε δεδομένα εμποδίων, αλλά στη μνήμη χωρικής θέσης του ρομπότ. Σκοπός της είναι να προσανατολίσει το ρομπότ προς έναν στόχο που έχει "κλειδωθεί" από τους αισθητήρες (Σόναρ ή Ραντάρ), ακόμα και αν αυτός ο στόχος δεν είναι πλέον ορατός.

Η λειτουργία της βασίζεται στον υπολογισμό διανυσμάτων και περιλαμβάνει τα εξής βήματα:

1. Έλεγχος Μνήμης: Αρχικά, η ενέργεια επικοινωνεί με το TargetMemory. Ελέγχει αν υπάρχει ενεργός στόχος.

- Περίπτωση Αποτυχίας: Εάν δεν υπάρχει στόχος στη μνήμη, η ενέργεια λειτουργεί ως μηχανισμός ασφαλείας και εκτελεί μια τυχαία στροφή (±90°) για να συνεχιστεί η εξερεύνηση.

2. Υπολογισμός Διανύσματος: Εφόσον υπάρχει στόχος, ανακτάται η θέση του (PinnedTargetPosition) και υπολογίζεται το διάνυσμα κατεύθυνσης από το ρομπότ προς τον στόχο:
   $Direction = TargetPosition - RobotPosition$, όπου TargetPosition και RobotPosition τα τρισδιάστατα διανύσματα θέσης του στόχου και του ρομπότ.
3. Υπολογισμός Γωνίας: Χρησιμοποιείται η συνάρτηση Vector3.SignedAngle για να βρεθεί η γωνιακή διαφορά μεταξύ του διανύσματος "Μπροστά" (transform.forward) του ρομπότ και του διανύσματος του στόχου.
   - Θετική γωνία (>0) υποδηλώνει ότι ο στόχος βρίσκεται στα δεξιά.
   - Αρνητική γωνία (<0) υποδηλώνει ότι ο στόχος βρίσκεται στα αριστερά.
4. Κανονικοποίηση στροφής: Μια σημαντική σχεδιαστική λεπτομέρεια είναι ότι το ρομπότ δεν στρίβει ακριβώς στην γωνία του στόχου (π.χ. 37 μοίρες), αλλά εκτελεί μια στροφή 90 μοιρών προς τη σωστή πλευρά, ανάλογα με το αν η υπολογισμένη γωνία είναι θετική ή αρνητική. Αυτή η προσέγγιση επιλέχθηκε για να διατηρηθεί η κίνηση του ρομπότ συμβατή με τη γεωμετρία του χώρου, διευκολύνοντας την πλοήγηση γύρω από ορθογώνια εμπόδια ενώ ταυτόχρονα μειώνει την απόσταση από τον στόχο.

MarkTargetReachedAction: Η ενέργεια MarkTargetReachedAction αποτελεί το τελικό στάδιο στον κύκλο ζωής της αναζήτησης στόχων. Εκτελείται όταν το ρομπότ φτάσει επιτυχώς σε απόσταση αλληλεπίδρασης από έναν ενεργό στόχο (είτε POI είτε Radar Target).

Η διαδικασία που ακολουθεί ολοκληρώνει τον βρόχο μνήμης:

1. **Ενημέρωση Μνήμης:** Καλεί τη μέθοδο TargetMemory.Instance.AddToIgnoreList(target, duration). Με αυτόν τον τρόπο, ο τρέχων στόχος μπαίνει σε "καραντίνα" για ένα καθορισμένο χρονικό διάστημα (π.χ. 60 δευτερόλεπτα), αποτρέποντας την άμεση επανεπιλογή του.
2. **Εκκαθάριση Αναφορών:** Θέτει τις μεταβλητές currentTargetPOI και currentTargetRadarPOI του ελεγκτή σε null. Αυτό σηματοδοτεί στο υπόλοιπο σύστημα ότι το ρομπότ είναι πλέον "ελεύθερο" και πρέπει να επιστρέψει σε κατάσταση αναζήτησης/περιπολίας.

6.3.4 Η Δομή του Κανόνα (ConditionalRule)

Η κλάση ConditionalRule αποτελεί τη θεμελιώδη λογική μονάδα του συστήματος. Λειτουργεί ως ο συνδετικός κρίκος που γεφυρώνει μια Συνθήκη με μια Ενέργεια, υλοποιώντας τη δομή "Εάν [Συνθήκη], Τότε [Ενέργεια]".

Σε τεχνικό επίπεδο, πρόκειται για μια απλή κλάση C# που είναι Serializable, ώστε να μπορεί να εμφανίζεται και να τροποποιείται μέσα στον Inspector της Unity.

Τα βασικά πεδία της κλάσης είναι:

- **ruleName (string):** Ένα περιγραφικό όνομα (π.χ. "Avoid Obstacle"), χρήσιμο για την αποσφαλμάτωση (debugging) και την καταγραφή (logging) της συμπεριφοράς του ρομπότ.
- **condition (Condition):** Αναφορά στο ScriptableObject της συνθήκης που πρέπει να αξιολογηθεί.
- **action (RobotAction):** Αναφορά στο ScriptableObject της ενέργειας που πρέπει να εκτελεστεί εάν η συνθήκη αληθεύει.

Αυτή η σχεδίαση επιτρέπει την επαναχρησιμοποίηση των ίδιων συνθηκών και ενεργειών σε διαφορετικούς κανόνες, απλώς αλλάζοντας τους συνδυασμούς τους μέσα στον Editor.

6.3.5 Ομαδοποίηση Συμπεριφορών (RuleSet)

Για να οργανωθούν οι κανόνες σε ολοκληρωμένα σύνολα συμπεριφοράς, δημιουργήθηκε η κλάση RuleSet. Πρόκειται για ένα ScriptableObject που περιέχει κυρίως μια λίστα από κανόνες:

public List<ConditionalRule> rules;

Η ύπαρξη του RuleSet ως ξεχωριστού Asset προσφέρει σημαντικά πλεονεκτήματα στην αρχιτεκτονική:

1. **Εναλλαγή Συμπεριφοράς (Hot-Swapping):** Το ρομπότ μπορεί να αλλάξει ριζικά τον τρόπο λειτουργίας του απλώς αντικαθιστώντας το ενεργό RuleSet. Για παράδειγμα, μπορεί να υπάρχει ένα RuleSet για "Αυτόνομη Εξερεύνηση" και ένα διαφορετικό για "Ακολούθηση Στόχου". Έτσι επιτυγχάνεται και ο εύκολος προσδιορισμός της συμπεριφοράς από τον χρήστη της εφαρμογής μέσω του UI, που θα αναλυθεί σε επόμενο στάδιο.
2. **Ιεράρχηση (Priority):** Η σειρά των κανόνων μέσα στη λίστα καθορίζει την προτεραιότητά τους. Οι κανόνες που βρίσκονται στην κορυφή της λίστας ελέγχονται πρώτοι. Αυτό επιτρέπει την υλοποίηση μιας αρχιτεκτονικής, όπου οι ζωτικές λειτουργίες (π.χ. αποφυγή σύγκρουσης) υπερισχύουν των λιγότερο σημαντικών (π.χ. κίνηση προς στόχο).

6.3.6 Ο Εγκέφαλος του Συστήματος (RuleBasedController)

Η κλάση RuleBasedController είναι ο ενορχηστρωτής του συστήματος λήψης αποφάσεων. Κληρονομεί από το MonoBehaviour και αντικαθιστά την παλιά, μονολιθική λογική του RobotController. Πλέον, ο ρόλος του ελεγκτή δεν είναι να αποφασίζει μέσω σταθερού κώδικου τι θα κάνει, αλλά να εκτελεί το τρέχον RuleSet.

Κύκλος Εκτέλεσης (Execution Loop) Ο ελεγκτής εκτελείται σε τακτά χρονικά διαστήματα και ακολουθεί την εξής διαδικασία:

1. Έλεγχος Κατάστασης: Αρχικά, ελέγχει αν το ρομπότ εκτελεί ήδη κάποια ενέργεια (isExecutingAction). Εάν ναι, ο κύκλος τερματίζεται, καθώς οι ενέργειες (όπως η στροφή) πρέπει να ολοκληρωθούν πριν ληφθεί νέα απόφαση.
2. Αξιολόγηση Κανόνων: Εάν το ρομπότ είναι αδρανές, ο ελεγκτής διατρέχει τη λίστα κανόνων του ενεργού RuleSet με τη σειρά (από τον 0 έως τον N).
3. Πυροδότηση: Για κάθε κανόνα, καλεί τη μέθοδο Evaluate().
    - Εάν η συνθήκη επιστρέψει false, προχωρά στον επόμενο κανόνα.

- Εάν η συνθήκη επιστρέψει true, η διαδικασία σταματά. Ο κανόνας θεωρείται ο "νικητής".

4. Εκτέλεση: Ο ελεγκτής καλεί τη μέθοδο Execute() της αντίστοιχης ενέργειας, περνώντας μια συνάρτηση Callback OnActionComplete. Ταυτόχρονα, θέτει τη σημαία isExecutingAction σε true για να αποτρέψει την εκτέλεση άλλων κανόνων πριν ολοκληρωθεί ο τρέχον.
5. Ολοκλήρωση: Όταν η ενέργεια τελειώσει, καλεί το Callback, το οποίο επαναφέρει τη σημαία isExecutingAction σε false, επιτρέποντας στον ελεγκτή να ξαναρχίσει τον κύκλο αξιολόγησης στο επόμενο καρέ. Ταυτόχρονα επιστρέφει το ρομπότ στην προκαθορισμένη κατάσταση, η οποία είναι η ευθεία κίνηση, έτσι ώστε να ληφθούν νέα δεδομένα από τους αισθητήρες για να χρησιμοποιηθούν στην αξιολόγηση κανόνων.

Μέσω του RuleBasedController, επιτεύχθηκε ο διαχωρισμός της λογικής από τον κώδικα. Η προσθήκη μιας νέας συμπεριφοράς δεν απαιτεί πλέον την τροποποίηση της κλάσης του ελεγκτή, παρά μόνο τη δημιουργία των κατάλληλων Assets και τη ρύθμισή τους στον Editor.

6.4 Υποσύστημα Εκτέλεσης και Ελέγχου Κίνησης

Μετά τον επανασχεδιασμό της αρχιτεκτονικής, ο ρόλος της κλάσης RobotController επαναπροσδιορίστηκε. Από κεντρικός ελεγκτής λήψης αποφάσεων, μετατράπηκε στο επίπεδο αφαίρεσης υλικού της προσομοίωσης. Η κύρια ευθύνη του είναι η πιστή εκτέλεση των εντολών που λαμβάνει από το υποσύστημα λήψης αποφάσεων και η διαχείριση της φυσικής υπόστασης του ρομπότ.

6.4.1 Διαχείριση Φυσικής (Rigidbody Physics)

Η κίνηση του ρομπότ δεν υλοποιείται μέσω απλής μετατόπισης συντεταγμένων (Transform.Translate), η οποία θα αγνοούσε τις συγκρούσεις, αλλά μέσω της μηχανής φυσικής της Unity (PhysX). Ο RobotController διαχειρίζεται το συστατικό Rigidbody, το οποίο δίνει φυσικά χαρακτηριστικά στο ρομπότ.

Κίνηση (Locomotion): Η μέθοδος MoveForward δεν αλλάζει άμεσα τη θέση, αλλά εφαρμόζει ταχύτητα: rigidBody.MovePosition(rigidBody.position + transform.forward * (moveSpeed * Time.fixedDeltaTime));

Η χρήση της MovePosition στο FixedUpdate εξασφαλίζει ότι η κίνηση σέβεται τους νόμους της φυσικής, επιτρέποντας στο ρομπότ να σταματά όταν προσκρούει σε τοίχους (Collision Detection) αντί να περνάει από μέσα τους.

Περιστροφή (Rotation): Αντίστοιχα, η περιστροφή υλοποιείται μέσω της MoveRotation, η οποία επιτρέπει ομαλή μετάβαση (Interpolation) προς την επιθυμητή γωνία, αποφεύγοντας τις αφύσικες, ακαριαίες αλλαγές προσανατολισμού.

6.4.2 Διαχείριση Καταστάσεων Κίνησης (Internal State Machine)

Παρόλο που η υψηλού επιπέδου λογική αφαιρέθηκε, ο RobotController διατηρεί μια απλοποιημένη Μηχανή Πεπερασμένων Καταστάσεων (enum RobotState) για να διαχειρίζεται τη μικρο-συμπεριφορά

της κίνησης και την επικοινωνία. Οι καταστάσεις περιορίστηκαν στις απολύτως απαραίτητες για τη φυσική λειτουργία:

- MovingForward: Ενεργοποίηση κινητήρων πρόωσης.
- TurningLeft / TurningRight: Ενεργοποίηση διαφορικής στροφής.
- Stopping: Ενεργοποίηση πέδησης (μηδενισμός ταχύτητας).
- Scanning: Ακινητοποίηση για σάρωση με αισθητήρες.

Αυτή η κατάσταση χρησιμοποιείται κυρίως για τον συγχρονισμό με το UI και την αποστολή των κατάλληλων μηνυμάτων BLE.

6.4.3 Επικοινωνία με Εξωτερικά Συστήματα (BLE Interface)

Μια κρίσιμη αρμοδιότητα που παρέμεινε στον RobotController είναι η γεφύρωση του εικονικού με τον πραγματικό κόσμο. Λειτουργεί ως ενδιάμεσος (Proxy) μεταξύ της λογικής του παιχνιδιού και του MicrobitBLEManager.

Κάθε φορά που αλλάζει η φυσική κατάσταση του ρομπότ (π.χ. από Stopping σε MovingForward), ο RobotController καλεί τη μέθοδο bleManager.SendMessageToMicrobit() με τον αντίστοιχο κωδικό εντολής (π.χ. "fwd", "left", "stop").

Αυτή η σχεδίαση απομονώνει πλήρως το σύστημα λήψης αποφάσεων από τις λεπτομέρειες του δικτύου. Οι Κανόνες και οι Ενέργειες δεν γνωρίζουν καν την ύπαρξη του Bluetooth· απλώς καλούν robot.PerformMoveForward(), και ο RobotController αναλαμβάνει αθόρυβα τη μετάδοση της εντολής στο φυσικό ρομπότ BBC micro:bit.

6.5 Διεπαφή Πραγματικού Κόσμου και Υποσύστημα AR

Η τελευταία αλλά κρίσιμη συνιστώσα της αρχιτεκτονικής είναι η γέφυρα μεταξύ της ψηφιακής προσομοίωσης και του φυσικού κόσμου. Αυτή επιτυγχάνεται μέσω δύο μηχανισμών: της Επαυξημένης Πραγματικότητας (AR) για την οπτικοποίηση των σεναρίων στον χώρο του χρήστη και της επικοινωνίας Bluetooth (BLE) για τον έλεγχο του φυσικού ρομπότ.

6.5.1 Διαχείριση Σεναρίων (StageRobotPair Architecture)

Η τοποθέτηση του εικονικού περιβάλλοντος στον πραγματικό χώρο δεν γίνεται αυθαίρετα. Κάθε σενάριο προσομοίωσης απαιτεί έναν συγκεκριμένο συνδυασμό πίστας και ρυθμίσεων ρομπότ. Για παράδειγμα, ένα σενάριο ανίχνευσης θερμότητας απαιτεί μια πίστα με πηγές θερμότητας και ένα ρομπότ με ενεργό θερμικό αισθητήρα.

Για την αντιμετώπιση αυτής της απαίτησης, σχεδιάστηκε η δομή StageRobotPair. Πρόκειται για μια κλάση που ορίζει ένα **Ζεύγος Πίστας-Ρομπότ**, διασφαλίζοντας ότι το υλικό (αισθητήρες) και το λογισμικό (περιβάλλον) είναι πάντα συγχρονισμένα.

Κάθε ζεύγος αποτελείται από:

1. **Το Prefab της Πίστας (Stage):** Σχεδιασμένο ειδικά για τον αλγόριθμο που θα δοκιμαστεί (π.χ. IRStage με στενούς διαδρόμους, LidarStage με πολύπλοκα γεωμετρικά εμπόδια, TemperatureStage με αντικείμενα HeatSource).

2. **Το Prefab του Ρομπότ:** Ένα ρυθμισμένο στιγμιότυπο του ρομπότ, όπου ο SensorManager έχει προεπιλεγμένες τις κατάλληλες τιμές:
    - **Ενεργοποίηση Αισθητήρων:** Π.χ. για το LidarStage, είναι ενεργά τα activateLidarSensor και activateForwardObstacleSensor. Για το ColorStage, είναι ενεργό το activateFloorColorSensor.
    - **Ρύθμιση Μασκών (Masking):** Καθορίζει ποιες κατηγορίες αντικειμένων (Layers) θα "βλέπει" το ρομπότ (π.χ. ObstacleLayer, POILayer).

Μέσω αυτής της αρχιτεκτονικής, ο χρήστης επιλέγει απλώς το σενάριο (π.χ. "Radar Search") και το σύστημα αναλαμβάνει να δημιουργήσει το κατάλληλο ζεύγος, εξαλείφοντας την πιθανότητα σφάλματος παραμετροποίησης (π.χ. να τρέξει αλγόριθμο θερμότητας σε πίστα χωρίς θερμικές πηγές).

6.5.2 Τοποθέτηση στον Χώρο (StagePlacer)

Η κλάση StagePlacer είναι υπεύθυνη για τη διαχείριση της Επαυξημένης Πραγματικότητας μέσω του AR Foundation της Unity. Η λειτουργία της ακολουθεί τα εξής βήματα:

1. **Ανίχνευση Επιπέδων:** Χρησιμοποιεί το υποσύστημα ARPlaneManager για να εντοπίσει οριζόντιες επιφάνειες (δάπεδο, τραπέζι) στον φυσικό χώρο.
2. **Επιλογή Ζεύγους:** Ο χρήστης επιλέγει το επιθυμητό StageRobotPair μέσω του UI της εφαρμογής.
3. **Ενσάρκωση (Instantiation):** Μόλις ο χρήστης επιλέξει σημείο τοποθέτησης (μέσω Raycast στην οθόνη), ο StagePlacer δημιουργεί ταυτόχρονα την πίστα και το ρομπότ του επιλεγμένου ζεύγους, διατηρώντας τη μεταξύ τους σχετική θέση και κλίμακα.

6.5.3 Επικοινωνία Bluetooth (MicrobitBLEManager)

Ο MicrobitBLEManager αποτελεί τον σύνδεσμο εξόδου της εφαρμογής. Λειτουργεί ως Singleton και διαχειρίζεται τον κύκλο ζωής της σύνδεσης Bluetooth Low Energy (BLE) με τον μικροελεγκτή BBC micro:bit.

- **Σάρωση και Σύνδεση:** Αναζητά συσκευές που εκπέμπουν το συγκεκριμένο UUID της υπηρεσίας UART του micro:bit.
- **Μετάδοση Εντολών:** Λαμβάνει αιτήματα από τον RobotController (όπως αναλύθηκε στην ενότητα 6.4) και τα μετατρέπει σε πακέτα χαρακτήρων (Strings) που αποστέλλονται μέσω της χαρακτηριστικής εγγραφής (RX Characteristic).
- **Κωδικοποίηση:** Οι εντολές είναι απλοποιημένες (π.χ. "fwd", "stop", "left") ώστε να ελαχιστοποιείται η καθυστέρηση (latency) και να διευκολύνεται η αποκωδικοποίηση από το firmware του ρομπότ.

6.5.4 Υλοποίηση Firmware στο Φυσικό Ρομπότ

Για την ολοκλήρωση του συστήματος ελέγχου, αναπτύχθηκε το απαραίτητο firmware για τον μικροελεγκτή BBC micro:bit, ο οποίος είναι προσαρτημένος στο σασί Yahboom Tiny:bit, το οποίο προσφέρει δύο μοτέρ, ρόδες, υπέρυθρες και υπέρυχους.

Η ανάπτυξη έγινε στο περιβάλλον Microsoft MakeCode σε γλώσσα MicroPython, αξιοποιώντας την ειδική βιβλιοθήκη που προσφέρει η Yahboom για το Tinybit για τον έλεγχο των μοτέρ κίνησης και τις

επεκτάσεις της Microsoft για την απευθείας χρήση του Bluetooth controller του micro:bit από την MicroPython.

Η λογική του firmware είναι γεγονοστραφής και βασίζεται στην υπηρεσία UART του Bluetooth.

Ανάλυση Κώδικα και Λειτουργίας:

1. **Αρχικοποίηση Υπηρεσίας:** Με την εντολή bluetooth.start_uart_service(), το micro:bit καθίσταται ανιχνεύσιμο και ενεργοποιεί τα χαρακτηριστικά για αποστολή και λήψη δεδομένων.
2. **Διαχείριση Εισερχόμενων Δεδομένων:** Ορίστηκε η συνάρτηση on_uart_data_received, η οποία καλείται αυτόματα κάθε φορά που η συσκευή λαμβάνει δεδομένα που τερματίζουν σε χαρακτήρα νέας γραμμής.
3. **Αποκωδικοποίηση και Εκτέλεση:** Μια δομή ελέγχου if-elif ελέγχει το περιεχόμενο της συμβολοσειράς που ελήφθει και καλεί την αντίστοιχη μέθοδο της βιβλιοθήκης του ρομπότ έτσι ώστε να εκτελέσει ευθεία κίνηση, στροφή προς τα αριστερά ή τα δεξιά, ή να σταματήσει τα μοτέρ με αποτέλεσμα την ακινητοποίηση του ρομπότ.

Το BBC micro:bit επίσης διαθέτει έναν πίνακα από 5x5 LED, τα οποία μπορούν να χρησιμοποιηθούν για την απεικόνιση απλών σχημάτων. Για την παρούσα εργασία χρησιμοποιούνται για να αναδείξουν την εντολή που έλαβε το micro:bit, προβάλλοντας το πρώτο γράμμα της:

- **F** για ευθεία κίνηση, από το **Forward**
- **L** για αριστερή στροφή, από το **Left**
- **R** για δεξιά στροφή, από το **Right**
- **S** για σταματημό του ρομπότ, από το **Stop**

6.6 Επίλογος

Στο Κεφάλαιο 6 αναλύθηκε διεξοδικά η αρχιτεκτονική του συστήματος, ξεκινώντας από την καταγραφή των λειτουργικών απαιτήσεων και προχωρώντας στον σχεδιασμό των επιμέρους υποσυστημάτων. Αποτυπώθηκε η δομή της εφαρμογής και ορίστηκαν οι ρόλοι των κρίσιμων διαχειριστών:

- Ο **SensorManager** για την αντίληψη του περιβάλλοντος.
- Ο **RuleBasedController** για τη λήψη αποφάσεων βάσει κανόνων.
- Ο **RobotController** για την εκτέλεση της κίνησης.
- Ο **MicrobitBLEManager** για τη διασύνδεση με το φυσικό ρομπότ.

Η υιοθέτηση μιας αρθρωτής (modular) και γεγονοστραφούς (event-driven) σχεδίασης εξασφάλισε την ευελιξία και την επεκτασιμότητα του συστήματος.

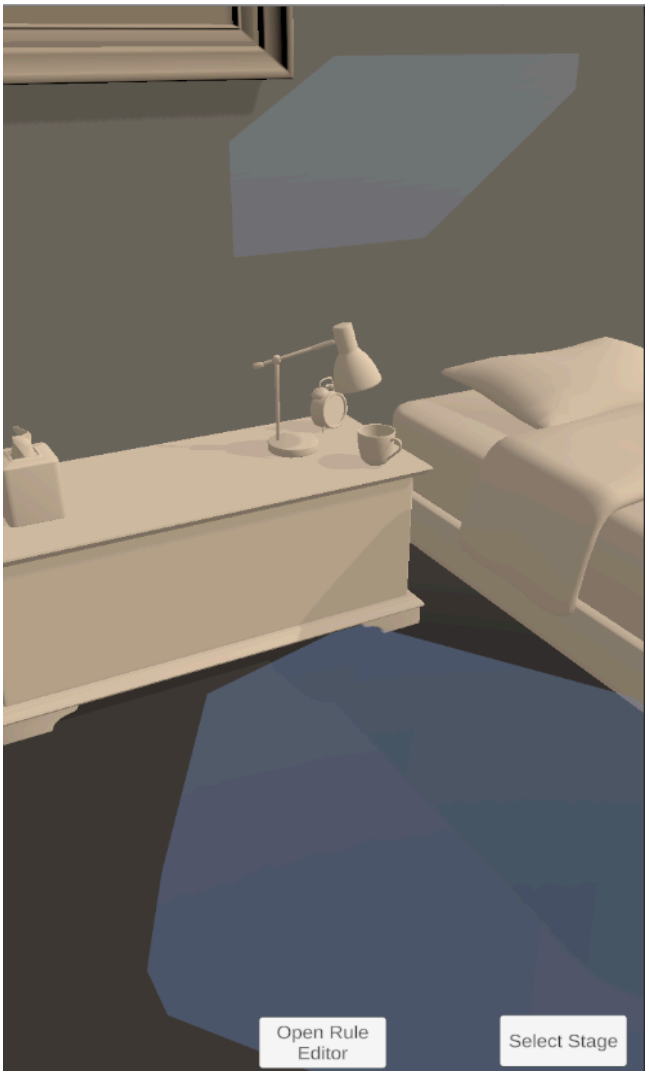
Κεφάλαιο 7: Υλοποίηση και Παρουσίαση Αποτελεσμάτων

7.1 Εισαγωγή

Στο παρόν κεφάλαιο παρουσιάζεται η τελική μορφή της εφαρμογής, καθώς και τα αποτελέσματα των πειραματικών προσομοιώσεων που πραγματοποιήθηκαν. Στόχος είναι η επίδειξη της λειτουργικότητας του συστήματος σε πραγματικές συνθήκες, η αξιολόγηση της χρηστικότητας του περιβάλλοντος διεπαφής και η επιβεβαίωση της ορθής λειτουργίας των αλγορίθμων πλοήγησης μέσω της τεχνολογίας Επαυξημένης Πραγματικότητας (AR).

7.2 Περιβάλλον Διεπαφής Χρήστη και Ροή Εργασίας

Η σχεδίαση της διεπαφής εστίασε στην απλότητα και την αμεσότητα, επιτρέποντας στον χρήστη να στήσει ένα πείραμα ρομποτικής μέσα σε λίγα δευτερόλεπτα, χωρίς πολύπλοκες ρυθμίσεις. Μέσω του XR Environment της Unity, το οποίο καθιστά εύκολη την προσομοίωση μιας AR εφαρμογής σαν να τρέχει σε κινητή συσκευή, θα παρουσιαστεί η εφαρμογή παρακάτω.



Σχήμα 7.1: Η αρχική οθόνη της εφαρμογής, με ανίχνευση χώρου

7.2.1 Εκκίνηση και Ανίχνευση Χώρου (AR Initialization)

Κατά την εκκίνηση της εφαρμογής, ενεργοποιείται αυτόματα το υποσύστημα ARSession. Η κάμερα της συσκευής ξεκινά τη σάρωση του φυσικού χώρου αναζητώντας χαρακτηριστικά σημεία (feature points) για τη δημιουργία ενός χωρικού χάρτη.

Ο χρήστης αντικρίζει την τροφοδοσία της κάμερας σε πραγματικό χρόνο. Μόλις το σύστημα, μέσω του ARPlaneManager, εντοπίσει οριζόντιες επιφάνειες όπως το δάπεδο ή ένα τραπέζι, αυτές οπτικοποιούνται ως **ημιδιάφανα πολύγωνα μπλε-γκρι απόχρωσης**. Η εμφάνιση αυτών των επιπέδων επιβεβαιώνει στον χρήστη ότι το σύστημα έχει κατανοήσει τη γεωμετρία του χώρου και είναι έτοιμο να υποδεχτεί τα εικονικά αντικείμενα. Η εφαρμογή αποτρέπει την τοποθέτηση αντικειμένων εάν δεν έχει ανιχνευθεί τουλάχιστον ένα επίπεδο, διασφαλίζοντας ότι η πίστα και το ρομπότ δεν θα "αιωρούνται" στο κενό.

Μία ακόμα λειτουργία που εκτελεί η εφαρμογή κατά την εκκίνηση είναι η σάρωση για ενεργές συσκευές Bluetooth, αναζητώντας το μοναδικό όνομα του BBC micro:Bit. Μόλις το εντοπίσει θα συνδεθεί αυτόματα, έτσι ώστε το πραγματικό ρομπότ Tiny:Bit να πραγματοποιεί τις ίδιες κινήσεις με το ψηφιακό ρομπότ της προσομοίωσης που θα αναλυθεί παρακάτω.

7.2.2 Κεντρικό Μενού Ελέγχου

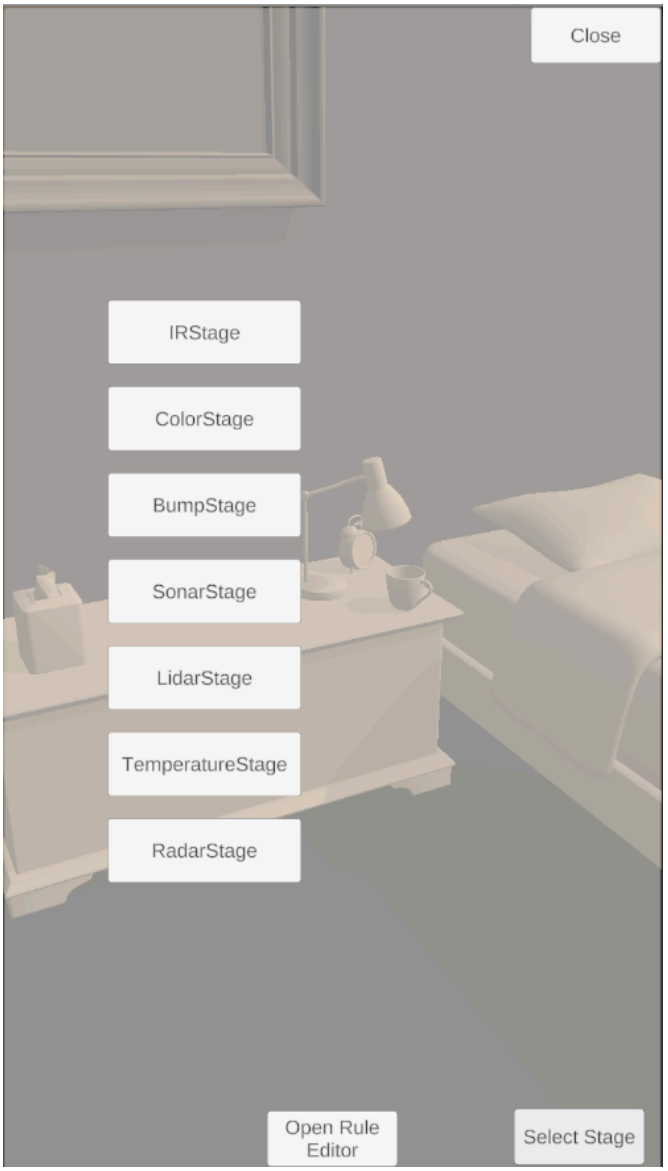
Στο κάτω μέρος της οθόνης, εμφανίζεται το κύριο μενού επιλογών, το οποίο αποτελείται από δύο βασικά λειτουργικά πλήκτρα:

Κουμπί "Select Stage" (Επιλογή Πίστας): Αυτό το κουμπί ανοίγει το μενού επιλογής σεναρίου. Εδώ ο χρήστης καλείται να επιλέξει την επιθυμητή προσομοίωση. Η επιλογή δεν αφορά μόνο τη γεωμετρία της πίστας, αλλά ενεργοποιεί τη λογική του αντίστοιχου ρομπότ που αναλύθηκε στο Κεφάλαιο 6.

Δίνονται επτά επιλογές στον χρήστη:

1. **IRStage**: Η πίστα της προσομοίωσης του συστήματος υπερύθρων. Το ρομπότ με χρήση του αισθητήρα υπερύθρων προσπαθεί να περιηγηθεί στην πίστα, αποφεύγοντας αδιέξοδους και εμπόδια.
2. **ColorStage**: Η πίστα της προσομοίωσης του συστήματος ανίχνευσης χρωμάτων. Με βάση τους αισθητήρες υπερύθρων και χρώματος, το ρομπότ κινείται, στρίβει και σταματάει ανάλογα με χρώματα στο περιβάλλον.
3. **BumpStage**: Η πίστα της προσομοίωσης του συστήματος αφής. Όταν το ρομπότ ακουμπήσει κάποιο εμπόδιο, αλλάζει κατεύθυνση προσπαθώντας να βρει ανοικτό μονοπάτι.
4. **SonarStage:** Η πίστα προσομοίωσης του συστήματος υπερήχων. Εδώ το ρομπότ διαθέτει περιστρεφόμενο αισθητήρα και καλείται να εντοπίσει αντικείμενα ενδιαφέροντος που είναι διασκορπισμένα στον χώρο.
5. **RadarStage:** Η πίστα για τον αισθητήρα Ραντάρ. Σε αυτό το σενάριο, το περιβάλλον είναι πιο ανοιχτό και το ρομπότ καλείται να εντοπίσει απομακρυσμένους στόχους, δοκιμάζοντας την ικανότητα του αισθητήρα να ανιχνεύει αντικείμενα σε μεγάλη εμβέλεια.
6. **LidarStage:** Μια πίστα τύπου λαβυρίνθου με σύνθετη γεωμετρία. Το σενάριο αυτό είναι σχεδιασμένο για την επίδειξη του αισθητήρα LiDAR και του αλγορίθμου έξυπνης διαφυγής, καθώς το ρομπότ πρέπει να χαρτογραφεί τον χώρο 360 μοιρών για να βρει διέξοδο.

7. **TemperatureStage:** Ένα περιβάλλον που περιέχει ενεργές πηγές θερμότητας. Το ρομπότ εξοπλίζεται με τον αισθητήρα θερμοκρασίας και ενεργοποιείται ο αλγόριθμος Heat Seeking για τον εντοπισμό της θερμότερης περιοχής.



Σχήμα 7.2: Το μενού επιλογής πίστας

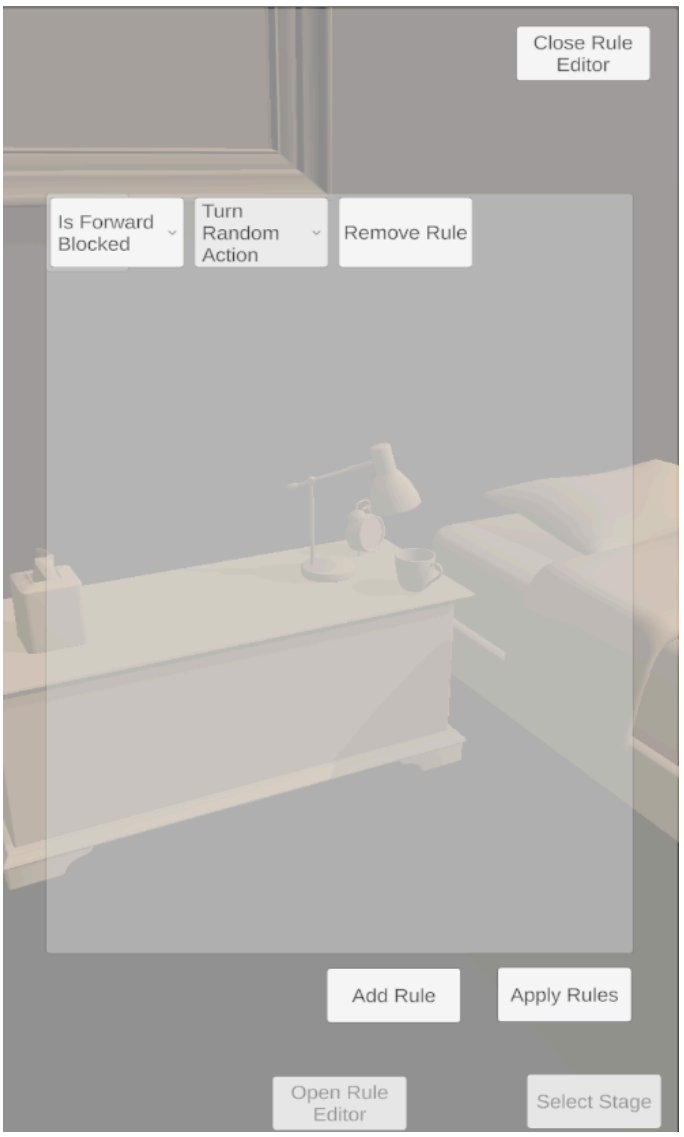
Κουμπί "Rule Editor" (Επεξεργαστής Κανόνων): Το δεύτερο βασικό πλήκτρο οδηγεί τον χρήστη στο υποσύστημα παραμετροποίησης της συμπεριφοράς. Πατώντας το, ανοίγει ένα νέο παράθυρο διεπαφής (Modal Window) το οποίο επιτρέπει την άμεση επισκόπηση και τροποποίηση της λογικής του ρομπότ σε πραγματικό χρόνο.

Η δομή του παραθύρου έχει σχεδιαστεί για να διαχειρίζεται δυναμικά μεταβλητό πλήθος κανόνων:

- **Λίστα Κανόνων:** Το κεντρικό τμήμα του παραθύρου εμφανίζει τους ενεργούς κανόνες του τρέχοντος RuleSet σε σειρά προτεραιότητας (από πάνω προς τα κάτω). Κάθε κανόνας απεικονίζεται ως μια διακριτή γραμμή, η οποία αποτελείται από την συνθήκη του κανόνα, η

οποία βρίσκεται στο drop-down μενού στα αριστερά, την ενέργεια του κανόνα που βρίσκεται στο κεντρικό drop-down μενού, και ένα κουμπί διαγραφής κανόνα “Remove Rule”.

- **Μηχανισμός Κύλισης:** Καθώς η πολυπλοκότητα της συμπεριφοράς αυξάνεται, ο αριθμός των κανόνων ενδέχεται να υπερβεί το ωφέλιμο ύψος της οθόνης. Για την αντιμετώπιση αυτού του ζητήματος, η λίστα έχει ενσωματωθεί εντός ενός ScrollRect container της Unity. Όταν οι κανόνες δεν χωρούν στην οθόνη, επιτρέπεται στον χρήστη να πλοηγηθεί στο σύνολο της λίστας μέσω αφής, διατηρώντας την διεπαφή καθαρή και λειτουργική ανεξαρτήτως του πλήθους των κανόνων.
- **Λειτουργικότητα:** Ανοίγοντας το παράθυρο, εμφανίζεται στον χρήστη η προεπιλεγμένη λίστα κανόνων που αντιστοιχεί στην πίστα που έχει επιλέξει. Μπορεί να προσθέσει νέους κανόνες με το κουμπί “Add Rule”, που θα εμφανίσει μία νέα γραμμή συνθήκης-ενέργειας, και να ενεργοποιήσει την τρέχουσα λίστα κανόνων πατώντας “Apply Rule”. Στην πάνω-δεξιά γωνία μπορεί να κλείσει το παράθυρο κανόνων πατώντας το κουμπί “Close Rule Editor”. Με αυτόν τον τρόπο επιτυγχάνεται και ο δυναμικός προγραμματισμός του ρομπότ από τον χρήστη, αν αυτό είναι επιθυμητό.



Σχήμα 7.3: Το μενού κανόνων

7.2.3 Διαδικασία Τοποθέτησης

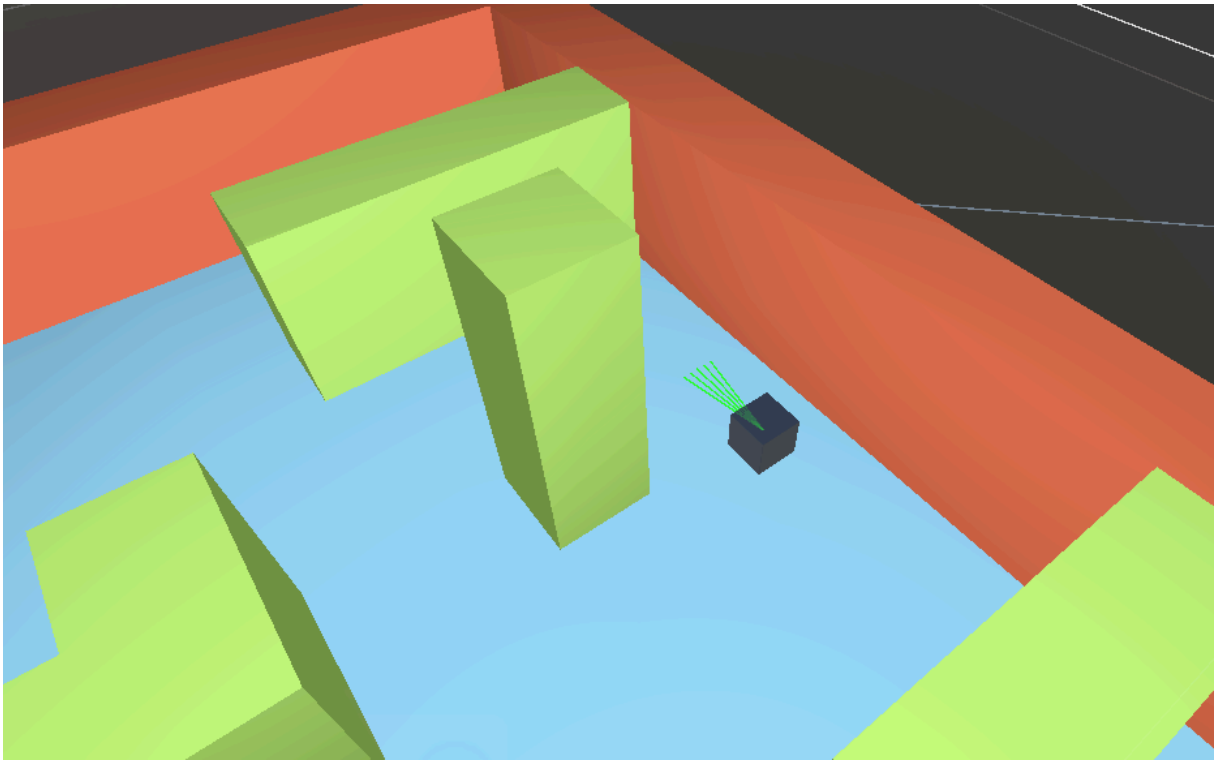
Αφού ο χρήστης επιλέξει την πίστα (μέσω του “Select Stage”), μπορεί να πατήσει σε οποιαδήποτε ανιχνευμένη επιφάνεια έτσι ώστε να τοποθετηθεί η ψηφιακή πίστα. Η πίστα θα αγκυρωθεί στο σημείο που επέλεξε ο χρήστης, επιτρέποντας του να δει την πίστα από πολλαπλές πλευρές κουνώντας την κάμερα της συσκευής.

7.3 Πειραματικά Σενάρια και Αποτελέσματα

Για την αξιολόγηση του συστήματος πραγματοποιήθηκε μια σειρά δοκιμών σε σενάρια αυξανόμενης πολυπλοκότητας. Σκοπός ήταν η επιβεβαίωση της ορθής λειτουργίας των αισθητήρων, της λογικής των κανόνων και της απόκρισης του ρομπότ σε πραγματικό χρόνο.

7.3.1 Σενάριο 1: Πλοήγηση με Αισθητήρα Υπερύθρων

Το πρώτο σενάριο εστιάζει στη θεμελιώδη ικανότητα κάθε αυτόνομου ρομπότ, την αποφυγή εμποδίων. Επιλέγοντας την πίστα IRStage, η οποία αποτελείται από διαδρόμους με τοίχους και γωνίες, ο χρήστης μπορεί να παρατηρήσει την συμπεριφορά του ρομπότ με ενεργό τον αισθητήρα υπερύθρων,



Σχήμα 7.4: Η πίστα IRStage

Η λίστα κανόνων περιέχει έναν κανόνα, που ορίζει την συμπεριφορά “Αν βρεις εμπόδιο, στρίψε προς τυχαία κατεύθυνση”

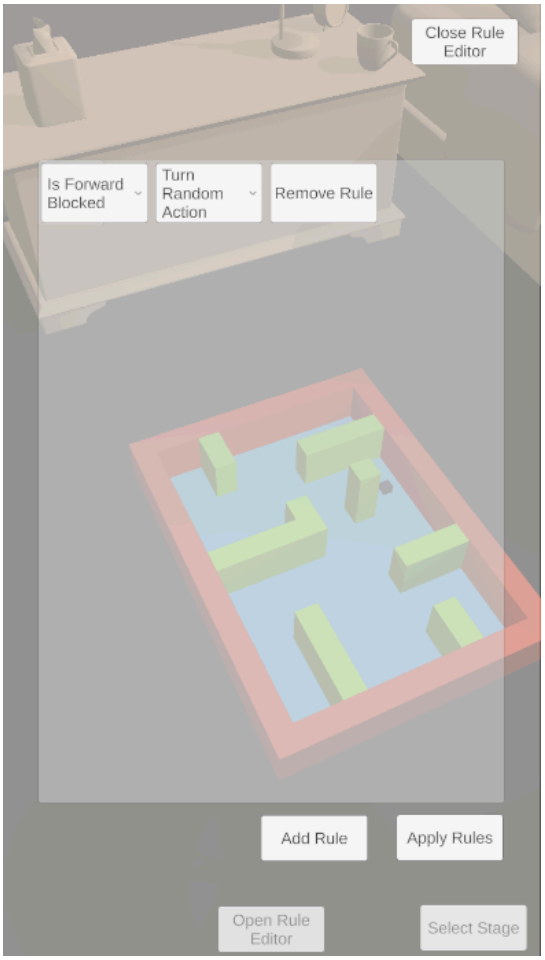
Όπως φαίνεται στην **Εικόνα**, το ρομπότ αναπαριστάται στο χώρο AR ως ένας γκρι κύβος.

Το χαρακτηριστικότερο στοιχείο της εικόνας είναι οι πράσινες γραμμές που εκτείνονται από το μπροστινό μέρος του ρομπότ.

- Αυτές οι γραμμές αποτελούν την οπτικοποίηση των ακτινών (raycasts) του αισθητήρα υπερύθρων.
- Το πράσινο χρώμα υποδηλώνει ότι η δέσμη είναι ελεύθερη και δεν έχει προσκρούσει σε κάποιο αντικείμενο εντός της εμβέλειας ανίχνευσης.
- Το στενό εύρος της δέσμης (περίπου 20 μοίρες) προσομοιώνει ρεαλιστικά την περιορισμένη γωνία θέασης των απλών αισθητήρων υπερύθρων που χρησιμοποιούνται σε εκπαιδευτικά ρομπότ.

Κατά την εκκίνηση του πειράματος, το ρομπότ κινείται ευθεία, βρίσκεται δηλαδή στην προκαθορισμένη κατάσταση. Καθώς πλησιάζει τον τοίχο της πίστας:

1. Οι πράσινες γραμμές τέμνουν το Collider του τοίχου και αλλάζουν χρώμα σε κόκκινο.
2. Ο SensorManager ανιχνεύει την πρόσκρουση και θέτει τη μεταβλητή IsForwardBlocked σε αληθής.
3. Ο RuleBasedController αξιολογεί τον κανόνα αποφυγής ως αληθή.
4. Το ρομπότ διακόπτει την κίνηση και εκτελεί στροφή προς τυχαία κατεύθυνση, αποφεύγοντας τη σύγκρουση, και επιστρέφει στην προκαθορισμένη κατάσταση ευθείας κίνησης.



Σχήμα 7.5: Η λίστα κανόνων της προσομοίωσης υπερύθρων

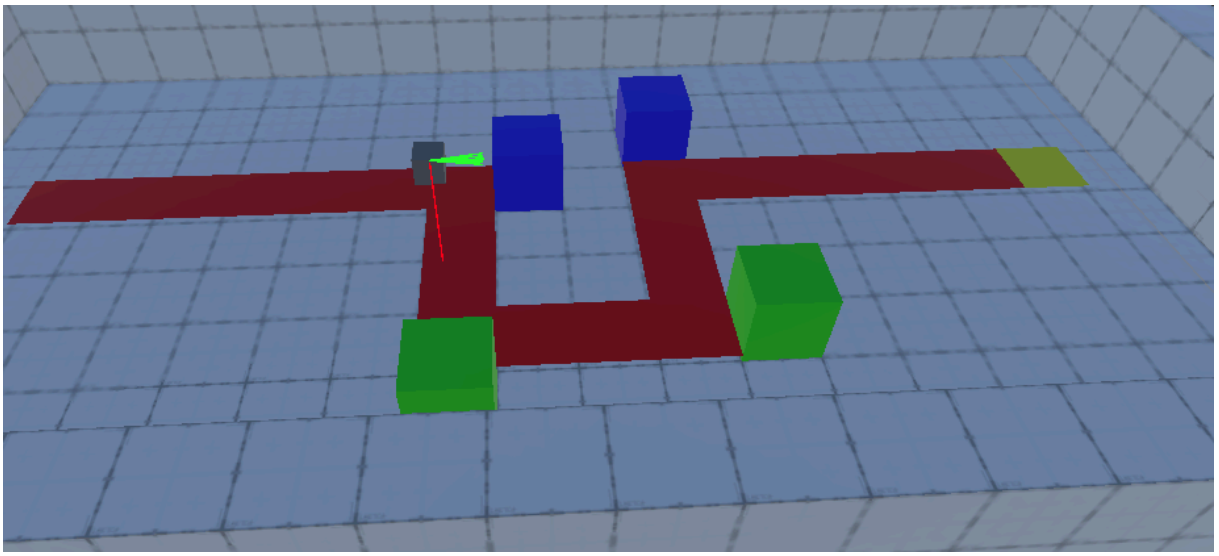
Η προσομοίωση επιβεβαίωσε ότι η ανίχνευση εμποδίων μέσω Raycasting λειτουργεί με ακρίβεια στο περιβάλλον AR, και ότι το ρομπότ αντιλαμβάνεται τα όρια της εικονικής πίστας σαν να ήταν φυσικά εμπόδια.

7.3.2 Σενάριο 2: Πλοήγηση και Λήψη Αποφάσεων βάσει Χρώματος (Color Navigation)

Το δεύτερο σενάριο αυξάνει την πολυπλοκότητα της πλοήγησης, εισάγοντας "σημασιολογική" πληροφορία. Το ρομπότ δεν καλείται απλώς να αποφύγει εμπόδια, αλλά να λάβει συγκεκριμένες αποφάσεις κατεύθυνσης ανάλογα με το χρώμα τους, καθώς και να αναγνωρίσει τον τερματισμό της διαδρομής βάσει της σήμανσης του δαπέδου.

Στην πίστα “ColorStage”, υπάρχει ένας διάδρομος όπου έχουν τοποθετηθεί χρωματιστοί κύβοι ως εμπόδια-σημάνσεις, και μια κίτρινη περιοχή στο τέλος που ορίζει τον τερματισμό.

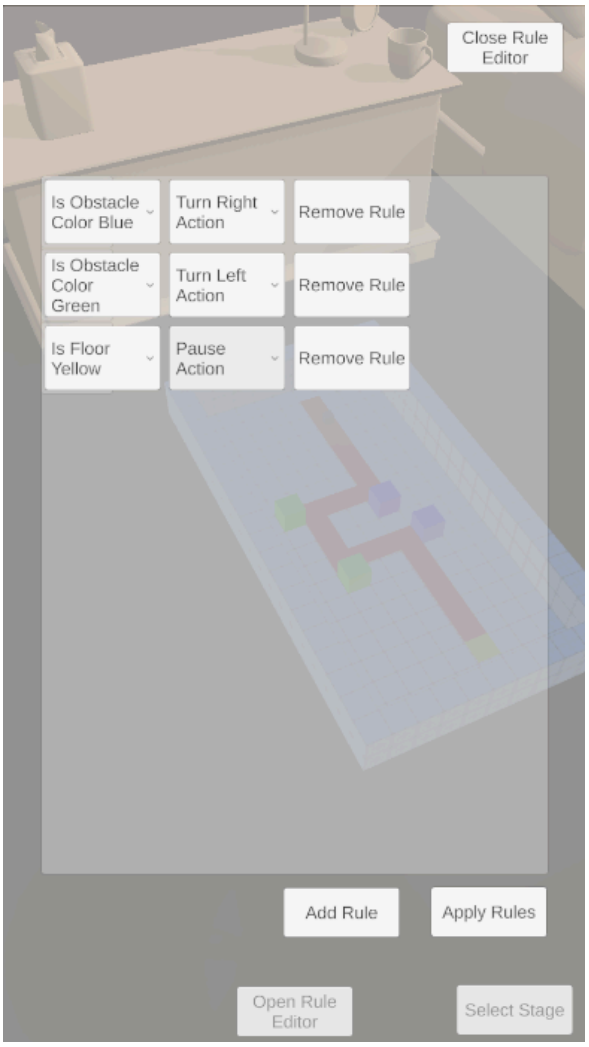
Το ρομπότ συνδυάζει την ανίχνευση εμποδίων του αισθητήρα υπερύθρων με την άντληση πληροφορίας για αντικείμενα του περιβάλλοντος από τους αισθητήρες χρωμάτων. Γίνεται ανίχνευση χρώματος του αντικειμένου ακριβώς κάτω από το ρομπότ με την κόκκινη γραμμή raycast στην εικόνα .., και ακριβώς μπροστά από το ρομπότ, όπου τα raycast του αισθητήρα υπερύθρων χρησιμοποιούνται και για ανάγνωση χρώματος του αντικειμένου.



Σχήμα 7.6: Η πίστα ColorStage

Κατά την εκτέλεση του σεναρίου, παρατηρήθηκε η εξής ακολουθία ενεργειών:

1. Το ρομπότ κινήθηκε ευθεία μέχρι να εντοπίσει τον πρώτο **Μπλε κύβο**. Ο RuleBasedController αναγνώρισε το χρώμα, παρέκαμψε τον προεπιλεγμένο κανόνα κίνησης και εκτέλεσε στροφή 90 μοιρών δεξιά.
2. Στη συνέχεια, συναντώντας τον **Πράσινο κύβο**, το σύστημα αντέδρασε σωστά εκτελώντας στροφή 90 μοιρών αριστερά.
3. Τέλος, μόλις ο κάθετος αισθητήρας (που οπτικοποιείται με μια κόκκινη ακτίνα προς το έδαφος) ανίχνευσε την τιμή χρώματος του **Κίτρινου δαπέδου**, ενεργοποιήθηκε ο κανόνας παύσης και το ρομπότ ακινητοποιήθηκε επιτυχώς.



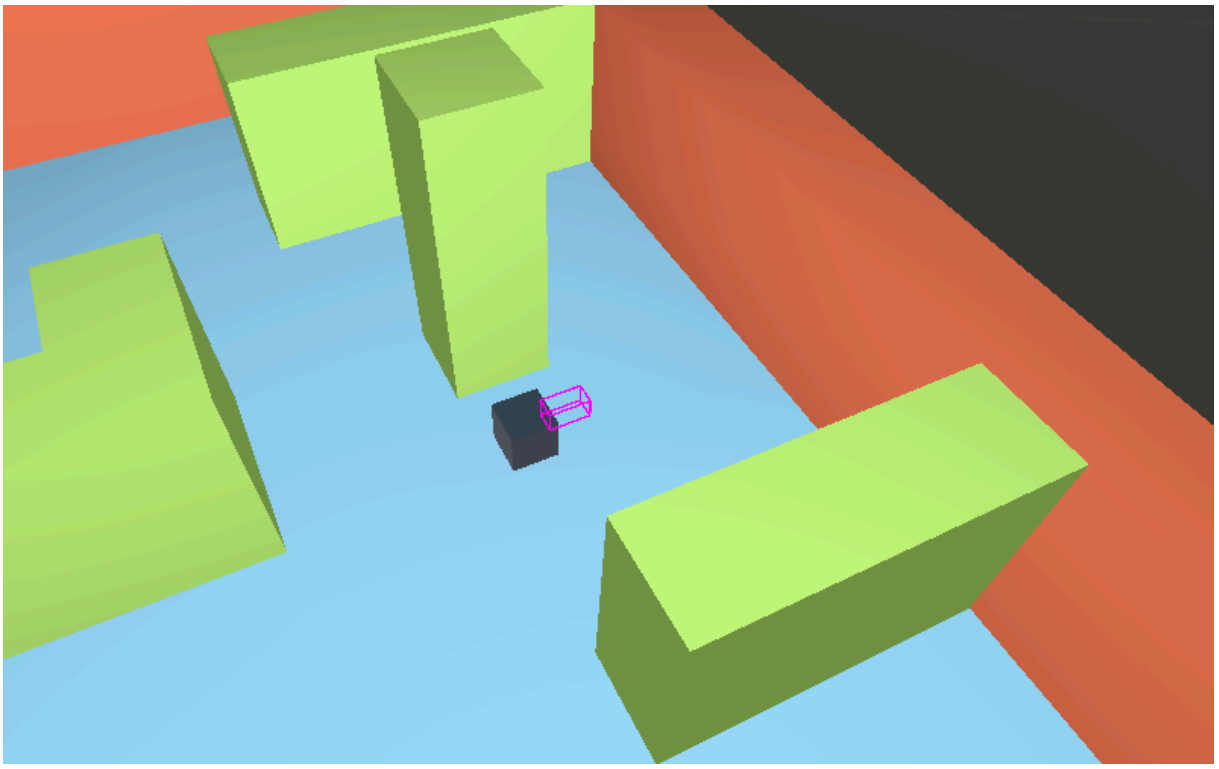
Σχήμα 7.7: Η λίστα κανόνων της προσομοίωσης του αισθητήρα χρώματος

Η προσομοίωση απέδειξε την ικανότητα του συστήματος να διαχειρίζεται πολυτροπική πληροφορία (απόσταση και χρώμα) και να εκτελεί σύνθετες εντολές πλοήγησης βασισμένες σε οπτικές ενδείξεις.

7.3.3 Σενάριο 3: Πλοήγηση με Αισθητήρα Αφής (Bump Sensor)

Το τρίτο σενάριο εξετάζει τον μηχανισμό ασφαλείας του ρομπότ, ο οποίος ενεργοποιείται όταν αποτύχουν τα συστήματα απομακρυσμένης ανίχνευσης. Σε αντίθεση με τους αισθητήρες IR και Lidar που προλαμβάνουν τις συγκρούσεις, ο αισθητήρας επαφής αντιδρά μετά το φυσικό γεγονός της πρόσκρουσης.

Επιλέγοντας την πίστα BumpStage, η οποία περιέχει εμπόδια που δεν είναι απαραίτητα ορατά από άλλους αισθητήρες (ή προσομοιώνοντας τυφλή πλοήγηση), ο χρήστης παρατηρεί τη συμπεριφορά του ρομπότ βάσει του κανόνα: «Αν ανιχνευθεί επαφή (IsBumped), εκτέλεσε ενέργεια απομάκρυνσης».



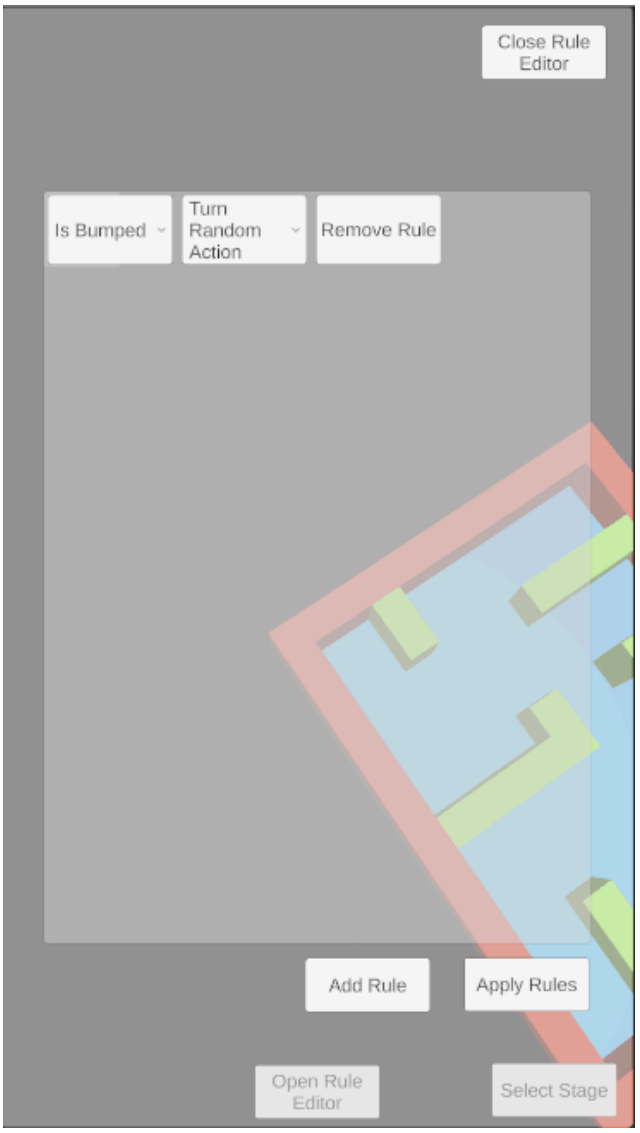
Σχήμα 7.8: Η πίστα BumpStage

Σε αυτό το σενάριο, η οπτική πληροφορία διαφοροποιείται σημαντικά από τα Raycasts.

- Αντί για γραμμές, γύρω από το σασί του ρομπότ εμφανίζεται ένα πλαίσιο που αναπαριστά τον όγκο του αισθητήρα επαφής (Collider).
- Όσο το ρομπότ κινείται ελεύθερα, το πλαίσιο έχει χρώμα μωβ, υποδεικνύοντας ότι η λίστα επαφών είναι κενή.
- Δεν υπάρχουν ακτίνες πρόβλεψης, καθώς ο αισθητήρας δεν «βλέπει» απόσταση, αλλά αντιλαμβάνεται μόνο την άμεση επαφή.

Κατά την εκκίνηση, το ρομπότ κινείται ευθεία. Η διαδικασία αντίδρασης περιγράφεται στα εξής βήματα:

1. **Φυσική Επαφή:** Το ρομπότ προσκρούει σε ένα εμπόδιο. Το σύστημα φυσικής της Unity εντοπίζει τη διείσδυση του εμποδίου στον χώρο του Trigger Collider.
2. **Ενεργοποίηση Συμβάντος:** Το βοηθητικό script BumpSensorTriggerHandler ενεργοποιείται (OnTriggerEnter) και ειδοποιεί μέσω Delegate τον κυρίως αισθητήρα.
3. **Αλλαγή Κατάστασης:** Το χρώμα του πλαισίου (Gizmo) αλλάζει ακαριαία σε κίτρινο, και η μεταβλητή IsObstacleDetected γίνεται αληθής.
4. **Αντίδραση:** Ο RuleBasedController ενεργοποιεί τον κανόνα σύγκρουσης. Στην συγκεκριμένη περίπτωση η ενέργεια που ακολουθεί είναι πιο σύνθετη από μια απλή στροφή: το ρομπότ εκτελεί πρώτα μια μικρή οπισθοχώρηση για να αποκολληθεί από το εμπόδιο και στη συνέχεια στρίβει για να αλλάξει πορεία.



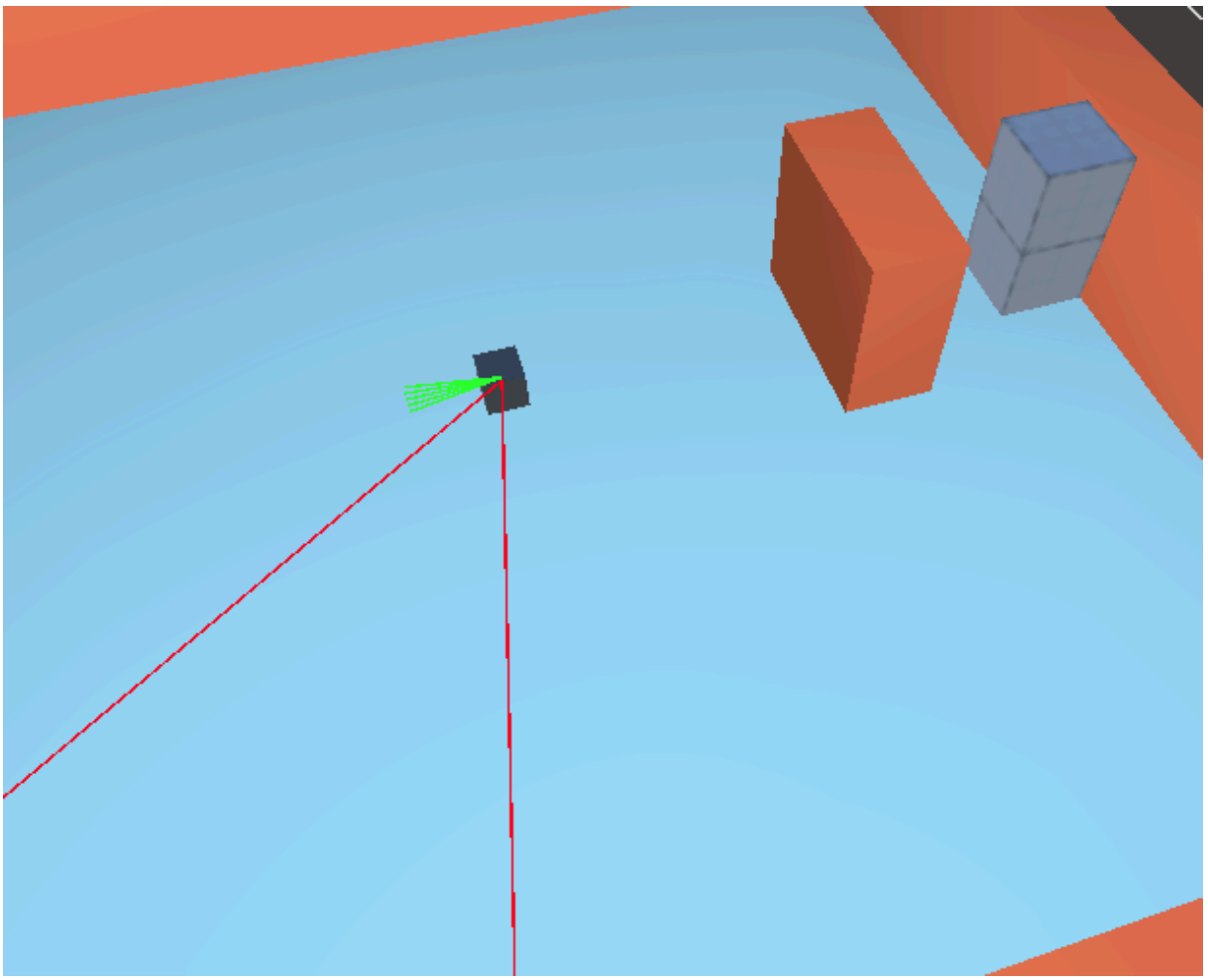
Σχήμα 7.9: Η λίστα κανόνων της προσομοίωσης αφής

Η προσομοίωση επιβεβαίωσε την ορθή λειτουργία της γεγονοστραφούς αρχιτεκτονικής. Παρόλο που ο αισθητήρας δεν τρέχει υπολογισμούς σε κάθε καρέ όπως γίνεται με το raycasting, η άμεση επικοινωνία μεταξύ του TriggerHandler και του SensorManager εξασφάλισε μηδενική καθυστέρηση στην αντίδραση του ρομπότ μόλις συνέβη η σύγκρουση. Αξίζει να αναφερθεί ότι υπάρχουν περιθώρια βελτίωσης στην προσομοίωση, καθώς θα μπορούσε να προστεθεί μία ενέργεια οπισθοχώρησης και μία συνθήκη για το τέλος της οπισθοχώρησης έτσι ώστε να γενικευτεί το σύστημα και να αφαιρεθούν αυτές οι στατικές λειτουργίες από την μοναδική ενέργεια που εκτελεί το ρομπότ, και να γίνουν ξεχωριστοί κανόνες.

7.3.4 Σενάριο 4: Εντοπισμός Στόχου με Υπέρηχο

Το τέταρτο σενάριο σηματοδοτεί τη μετάβαση από την απλή πλοήγηση στην εκτέλεση αποστολών. Εδώ, το ρομπότ καλείται να εντοπίσει συγκεκριμένα αντικείμενα ενδιαφέροντος (Points of Interest - POIs) στον χώρο, χρησιμοποιώντας έναν προσομοιωμένο αισθητήρα υπερήχων (Sonar) τοποθετημένο σε περιστρεφόμενη βάση.

Η πίστα SonarStage αποτελεί το κατάλληλο περιβάλλον για την δοκιμή του αισθητήρα υπερήχων, καθώς αποτελείται από έναν ανοιχτό χώρο, περιτριγυρισμένο από τοίχους.



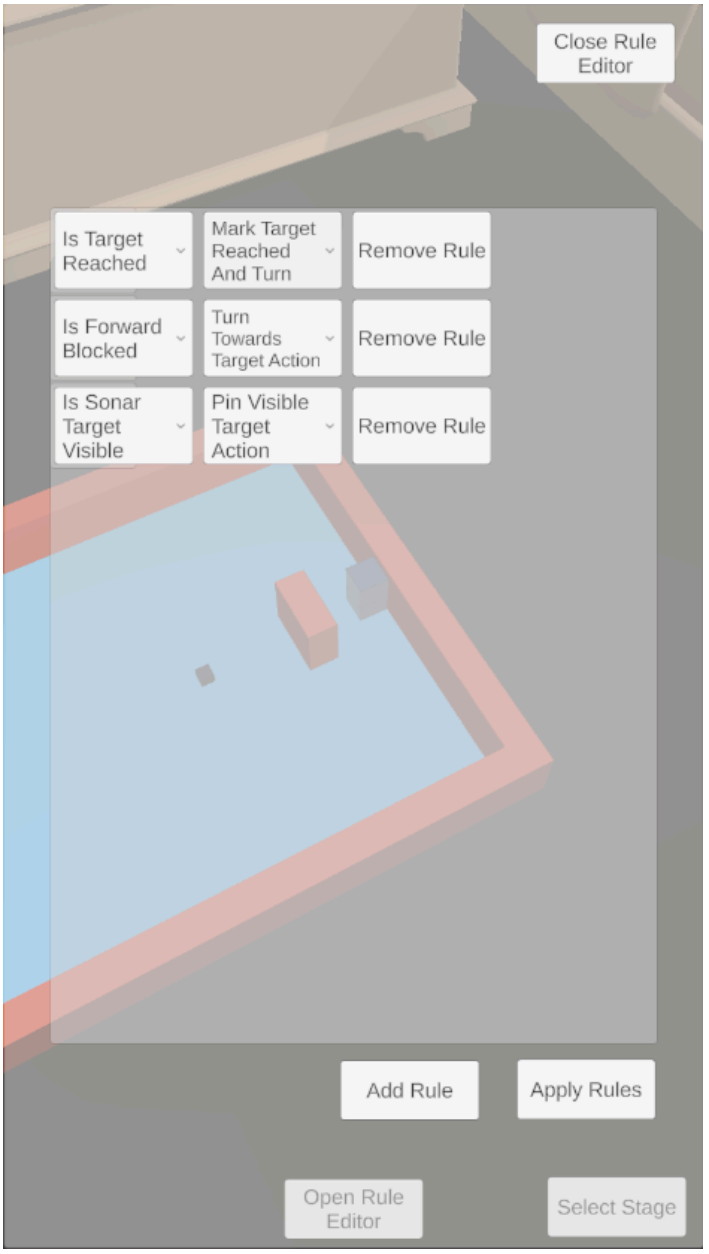
Σχήμα 7.10: Η πίστα SonarStage

Στην εικόνα του πειράματος ξεχωρίζουν τα εξής στοιχεία:

1. **Ο Κόκκινος Κώνος:** Μπροστά από το ρομπότ εκτείνεται ένας ημιδιάφανος κόκκινος κώνος.
    - Αυτό το σχήμα αναπαριστά το ηχητικό πεδίο του αισθητήρα υπερήχων. Η κωνική μορφή επιλέχθηκε για να δείξει την αύξηση της αβεβαιότητας όσο μεγαλώνει η απόσταση, προσομοιώνοντας ρεαλιστικά τη διασπορά των ηχητικών κυμάτων.
    - Κατά την εκτέλεση, ο κώνος αυτός περιστρέφεται ανεξάρτητα από το σώμα του ρομπότ, σαρώνοντας τον χώρο.
2. **Γκρίζος Στύλος:** Αποτελεί τον στόχο, δηλαδή το αντικείμενο ενδιαφέροντος.
3. **Πράσινες Γραμμές (Υπέρηθρες):** Παραμένουν ενεργές, διασφαλίζοντας ότι ενώ το ρομπότ ψάχνει για στόχους, δεν θα προσκρούσει σε τοίχους ενώ ταυτόχρονα μπορούν να εντοπίσουν αντικείμενα ενδιαφέροντος σε κοντινή απόσταση.

Η συμπεριφορά του ρομπότ έχει τρεις φάσεις:

1. **Σάρωση:** Όσο το ρομπότ κινείται, γίνεται συνεχής σάρωση του χώρου. Στην προκειμένη φάση αν βρεθεί εμπόδιο μπροστά στο ρομπότ θα εκτελέσει στροφή προς τυχαία κατεύθυνση δεξιά ή αριστερά.
2. **Κλείδωμα:** Μόλις ο κώνος τέμνει το αντικείμενο ενδιαφέροντος, ενεργοποιείται ο κανόνας PinVisibleTarget, αποθηκεύοντας την θέση του αντικειμένου στην μνήμη του ρομπότ. Από την στιγμή που υπάρχει συγκεκριμένος στόχος, το ρομπότ όταν συναντά εμπόδιο θα παίρνει στροφή προς την κατεύθυνση που θεωρεί ότι θα το φέρει πιο κοντά στον στόχο (ενέργεια TurnToTargetAction).
3. **Άφιξη και Ενημέρωση Μνήμης:** Όταν φτάσει σε κοντινή απόσταση, ενεργοποιείται η ενέργεια MarkTargetReached. Ο στόχος καταγράφεται στη μνήμη ως "ολοκληρωμένος" και το ρομπότ παίρνει στροφή για να απομακρυνθεί, προς αναζήτηση επόμενου στόχου, αγνοώντας πλέον το συγκεκριμένο γκρίζο αντικείμενο για ένα χρονικό διάστημα.

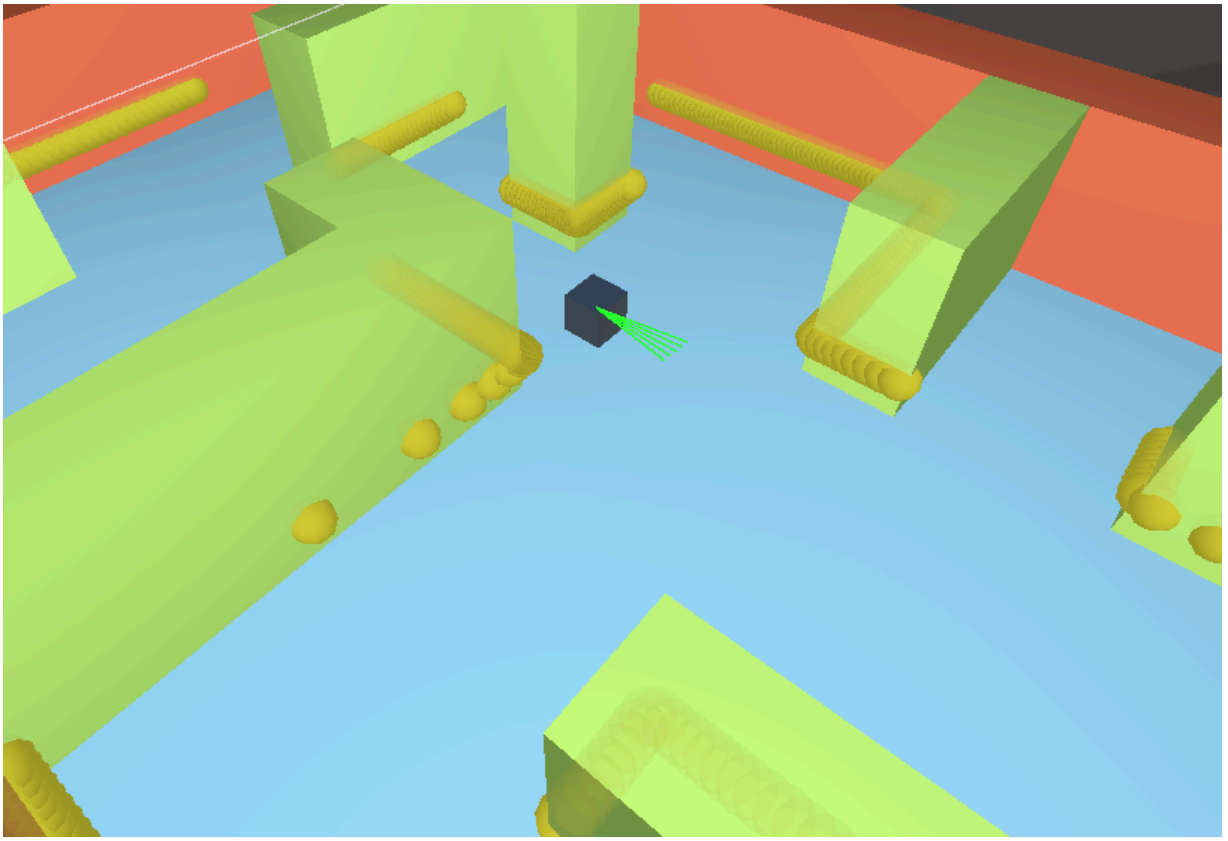


Σχήμα 7.11: Η λίστα κανόνων της προσομοίωσης σόναρ

Το σενάριο αυτό απέδειξε την επιτυχή εφαρμογή της σύντηξης αισθητήρων (Sensor Fusion), όπου το σύστημα IR διασφαλίζει την ακεραιότητα του ρομπότ (χαμηλό επίπεδο ασφάλειας) ενώ το Sonar διαχειρίζεται την αποστολή (υψηλό επίπεδο στόχευσης). Η προσομοίωση ανέδειξε την ικανότητα του συστήματος για στοχευμένη πλοήγηση σε ανοιχτούς χώρους. Σε αντίθεση με τα προηγούμενα σενάρια λαβυρίνθου, εδώ το ρομπότ έδειξε σκοπιμότητα μέσω της έξυπνης στροφής (TurnToTargetAction). Η αξιοποίηση της χωρικής μνήμης επέτρεψε στο ρομπότ να διατηρεί τον προσανατολισμό του προς τον στόχο, ακόμη και όταν αναγκαζόταν να τον αποφύγει προσωρινά λόγω εμποδίων, βελτιστοποιώντας έτσι τη διαδρομή του.

7.3.5 Σενάριο 5: Πλοήγηση σε Λαβύρινθο με LiDAR

Η πέμπτη προσομοίωση αφορά την προηγμένη πλοήγηση σε κλειστούς χώρους. Εδώ, το ρομπότ καλείται να πλοηγηθεί στην ίδια πίστα που χρησιμοποιήθηκε για τα σενάρια 1 και 3. Σε αντίθεση με τους απλούς αισθητήρες IR που παρέχουν δυαδική πληροφορία (εμπόδιο/όχι εμπόδιο), το LiDAR προσφέρει μια πλήρη χαρτογράφηση του περιβάλλοντος χώρου, επιτρέποντας την στοχευμένη ανίχνευση μονοπατιού.

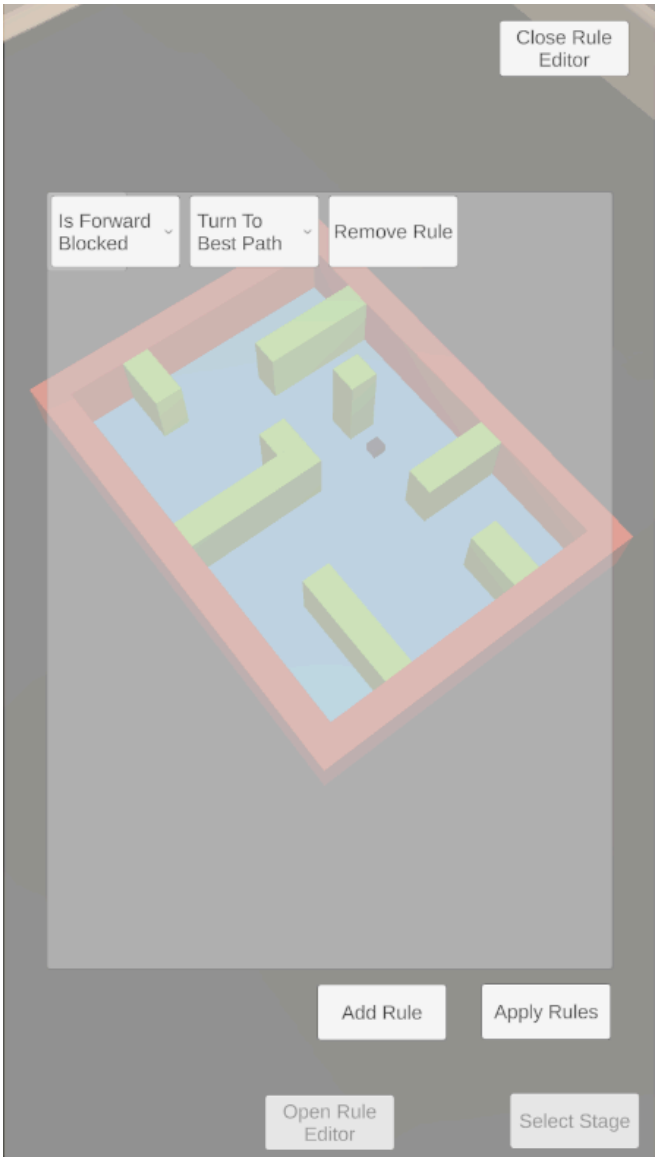


Σχήμα 7.12: Η πίστα LiDARStage

Στην εικόνα της προσομοίωσης, η λειτουργία του LiDAR οπτικοποιείται ως σημεία σάρωσης. Ένας πλήρης κύκλος από κίτρινες σφαίρες εκπέμπεται από το κέντρο του ρομπότ προς όλα τα ορατά αντικείμενα σε κοντινή απόσταση. Κάθε σφαίρα αντιπροσωπεύει μια μέτρηση απόστασης από το ρομπότ προς το αντικείμενο. Ταυτόχρονα είναι ορατές και οι πράσινες γραμμές του αισθητήρα υπερύθρων.

Κατά την διάρκεια εκτέλεσης της προσομοίωσης ακολουθείται η εξής συμπεριφορά:

1. **Ανίχνευση Εγκλωβισμού:** Αν το ρομπότ βρει εμπόδιο μπροστά του σε κοντινή απόσταση, θα ενεργοποιηθεί η συνθήκη “Is Forward Blocked”.
2. **Ανάλυση Χώρου:** Αντί να στρίψει τυχαία, όπως στο Σενάριο 1, το ρομπότ ξεκινά την εκτέλεση την ενέργεια TurnToBestPath χρησιμοποιώντας τα δεδομένα του αισθητήρα LiDAR. Ο αλγόριθμος υπολογίζει τη μέση απόσταση ελεύθερου χώρου στα τέσσερα τεταρτημόρια (Μπροστά, Πίσω, Αριστερά, Δεξιά).
3. **Λήψη Βέλτιστης Απόφασης:** Το σύστημα εντοπίζει την κατεύθυνση που οδηγεί στον πιο ανοιχτό χώρο.
4. **Εκτέλεση:** Το ρομπότ εκτελεί στροφή προς την βέλτιστη κατεύθυνση, και επιστρέφει στην προκαθορισμένη λειτουργία ευθείας κίνησης.



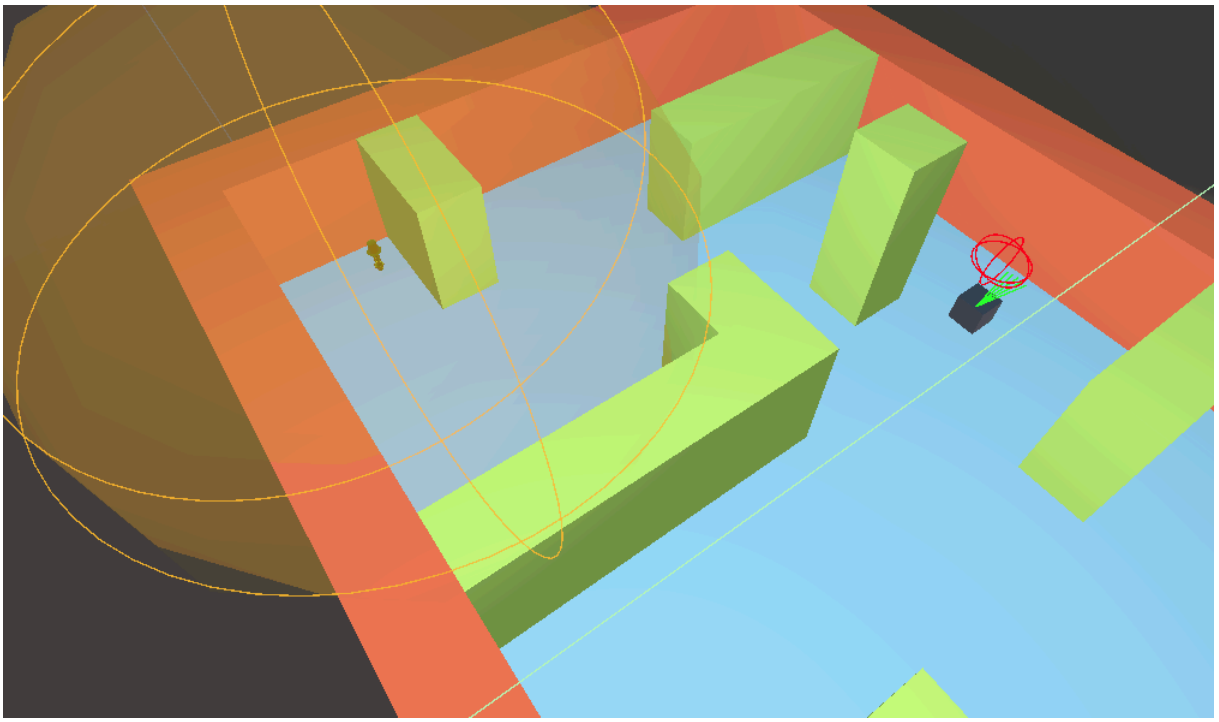
Σχήμα 7.13: Η λίστα κανόνων της προσομοίωσης LiDAR

Συμπέρασμα: Η συγκεκριμένη προσομοίωση αναδεικνύει την υπεροχή της χαρτογράφησης έναντι της απλής ανίχνευσης. Το ρομπότ δεν πλοηγείται τυφλά, αλλά δείχνει κατανόηση της τοπολογίας του χώρου, επιλέγοντας ενεργά το βέλτιστο μονοπάτι.

7.3.6 Σενάριο 6: Πλοήγηση βάσει Θερμικής Κλίσης

Η προσομοίωση εξετάζει μία διαφορετική προσέγγιση αναζήτησης στόχων. Σε αντίθεση με τους αισθητήρες LiDAR ή Sonar που παρέχουν πληροφορία κατεύθυνσης, ο αισθητήρας θερμοκρασίας παρέχει μόνο μία βαθμωτή τιμή για το σημείο όπου βρίσκεται το ρομπότ. Συνεπώς, το ρομπότ πρέπει να συγκρίνει μετρήσεις στον χρόνο για να καταλάβει αν πλησιάζει ή απομακρύνεται σε μία πηγή θερμότητας.

Η πίστα έχει την ίδια μορφή με τις προσομοιώσεις LiDAR και υπερύθρων, αποτελεί δηλαδή έναν κλειστό χώρο με τοίχους και εμπόδια. Στην πίστα υπάρχει ένα αντικείμενο το οποίο αποτελεί την πηγή θερμότητας, και ο στόχος του ρομπότ είναι να βρει αυτήν την πηγή, μαζί με χρήση του αισθητήρα υπερύθρων για απλή αποφυγή εμποδίων.



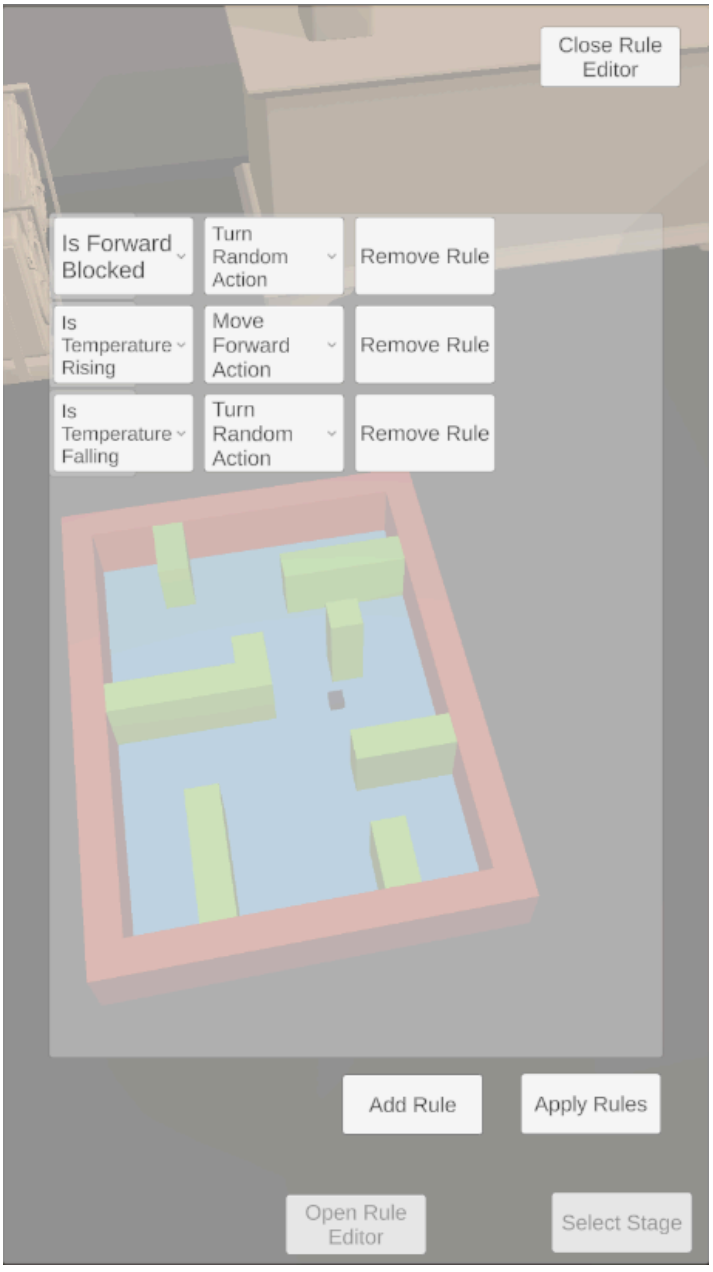
Σχήμα 7.13: Η πίστα TemperatureStage

Στην εικόνα της προσομοίωσης απεικονίζεται ένα αντικείμενο περιτριγυρισμένο από μία πορτοκαλί σφαίρα. Αυτή η σφαίρα λειτουργεί ως οπτικός δείκτης για τον χρήστη, υποδεικνύοντας το "θερμό σημείο" (Hotspot) της πίστας. Το ρομπότ δεν "βλέπει" τη σφαίρα (δεν χρησιμοποιεί raycasts προς αυτήν), αλλά αντιλαμβάνεται την επιρροή της μέσω του εικονικού θερμικού πεδίου που την περιβάλλει. Η θερμοκρασία στην πίστα είναι μέγιστη στο κέντρο της σφαίρας και μειώνεται γραμμικά όσο αυξάνεται η απόσταση από αυτή.

Το ρομπότ προβάλλει πράσινες ακτίνες που είναι η ένδειξη του αισθητήρα υπερύθρων, και μία κόκκινη σφαίρα από πάνω του για τον αισθητήρα θερμότητας.

Η κίνηση του ρομπότ χαρακτηρίζεται από μια διαδικασία δοκιμής και λάθους:

1. Το ρομπότ ξεκινάει σε ευθεία. Αν βρει εμπόδιο, θα στρίψει τυχαία.
2. Όσο πλησιάζει την πορτοκαλί σφαίρα η θερμοκρασία ανεβαίνει, και η κίνηση είναι σταθερή και ευθύγραμμη.
3. Αν προσπεράσει την πηγή ή παρεκκλίνει, δηλαδή η θερμοκρασία πέφτει, το ρομπότ αρχίζει να εκτελεί τυχαίες στροφές μέχρι να βρει την κατεύθυνση προς την οποία ανεβαίνει η θερμοκρασία.
4. Τελικά, το ρομπότ καταλήγει να περιφέρεται γύρω από την πορτοκαλί σφαίρα, καθώς εκεί εντοπίζεται το τοπικό μέγιστο της συνάρτησης θερμότητας.

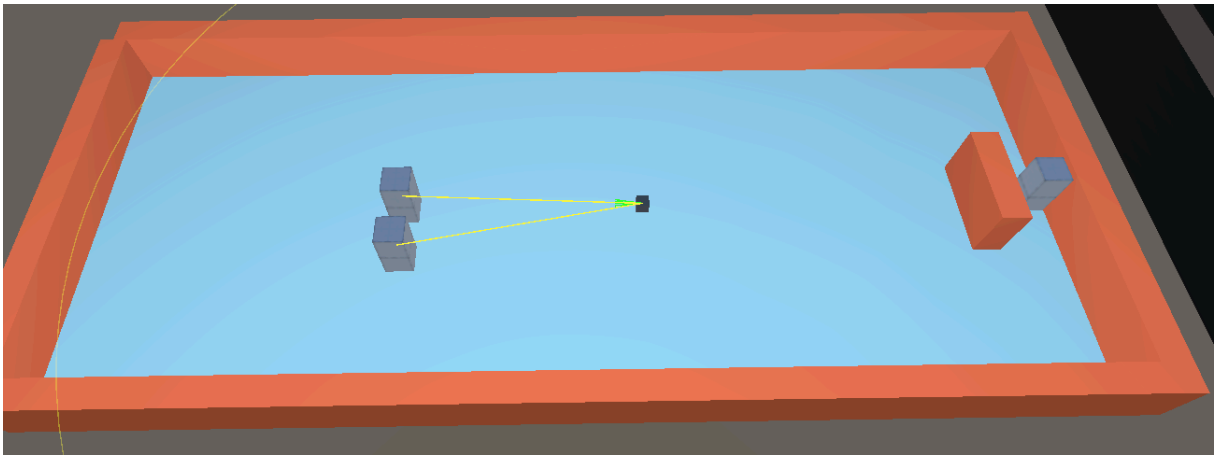


Σχήμα 7.14: Η λίστα κανόνων προσομοίωσης του αισθητήρα θερμοκρασίας

Η προσομοίωση αυτή δείχνει τη δυνατότητα του συστήματος να υποστηρίξει αλγορίθμους που βασίζονται σε χρονοσειρές δεδομένων (HeatMemory) και όχι μόνο σε στιγμιαία χωρική αντίληψη, ανοίγοντας τον δρόμο για σύνθετες συμπεριφορές αναζήτησης.

7.3.7 Σενάριο 6: Εντοπισμός Πολλαπλών Στόχων με Ραντάρ

Το τελευταίο σενάριο προσομοίωσης εξετάζει τη λειτουργία ενός αισθητήρα μακράς εμβέλειας για την ανίχνευση στόχων σε ανοιχτά πεδία. Σε αντίθεση με την Σόναρ που εστιάζει σε έναν στόχο τη φορά (στενός κώνος), το Ραντάρ σαρώνει έναν ευρύ τομέα και έχει τη δυνατότητα να εντοπίζει και να διατηρεί στη μνήμη του πολλαπλούς στόχους ταυτόχρονα.



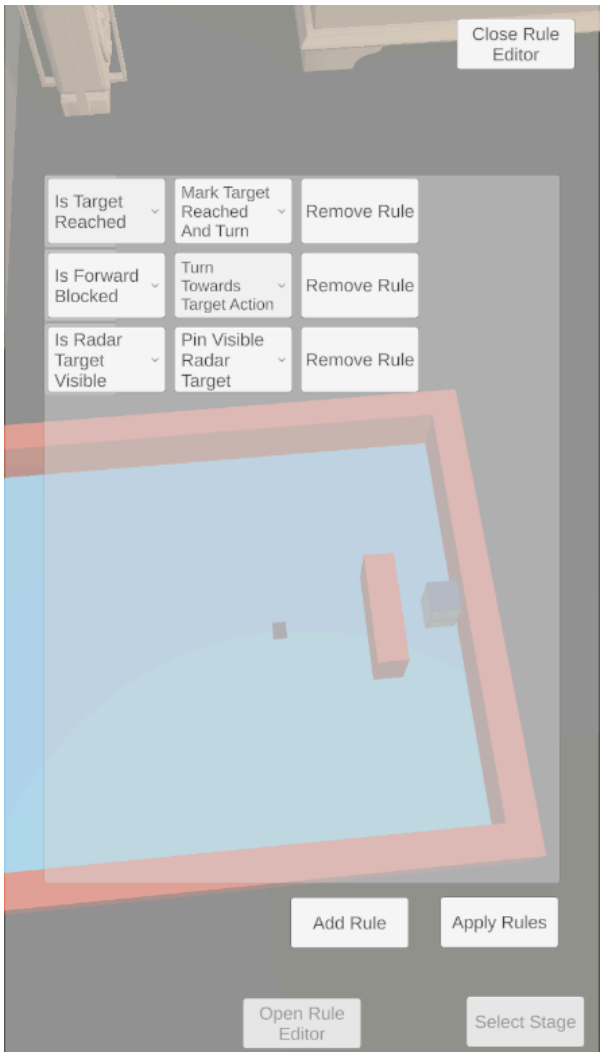
Σχήμα 7.15: Η πίστα RadarStage

Στην παραπάνω εικόνα φαίνεται η πίστα RadarStage, ένας ανοιχτός χώρος με πολλαπλούς στόχους. Το ραντάρ είναι ρυθμισμένο με γωνία θέασης 120°, και εκπέμπει έναν παλμό σάρωσης ανά τακτά χρονικά διαστήματα για τον εντοπισμό στόχων.

Η οπτική αναπαράσταση του ραντάρ αποτελείται από ένα κίτρινο τόξο το οποίο δείχνει το πεδίο κάλυψης του ραντάρ και κίτρινες γραμμές εντοπισμού προς τους στόχους που βρίσκονται εντός εμβέλειας.

Η συμπεριφορά του ρομπότ περιγράφεται παρακάτω:

1. **Σάρωση:** Το ρομπότ κινείται στο χώρο. Κάθε φορά που το χρονόμετρο σάρωσης ξεπεράσει το όριο, εκτελείται η μέθοδος UpdateSensor για να προσομοιωθεί η εκπομπή παλμού η οποία γεμίζει η λίστα με στόχους που βρέθηκαν.
2. **Επιλογή Στόχου:** Εφόσον η λίστα στόχων δεν είναι κενή, ενεργοποιείται ο κανόνας IsRadarTargetVisible. Η ενέργεια PinRadarTargetAction αναλαμβάνει να φιλτράρει τους στόχους, αφαιρώντας αυτούς που βρίσκονται στη λίστα αγνοημένων της μνήμης, στόχους που ήδη βρέθηκαν και προσεγγίστηκαν, και επιλέγει τον πλησιέστερο διαθέσιμο.
3. **Πλοήγηση:** Το ρομπότ στρέφεται προς τον επιλεγμένο στόχο και κινείται προς αυτόν.
4. **Ολοκλήρωση:** Μόλις φτάσει, ο στόχος μπαίνει στην λίστα αγνοημένων στόχων και το ραντάρ, στον επόμενο παλμό του, αγνοεί αυτόν τον στόχο και κλειδώνει στον επόμενο διαθέσιμο, συνεχίζοντας την αποστολή του.



Σχήμα 7.16: Η λίστα κανόνων προσομοίωσης ραντάρ

Η προσομοίωση του Ραντάρ ανέδειξε τη δυνατότητα διαχείρισης πολλαπλών δεδομένων. Το σύστημα δείχνει ότι μπορεί να φιλτράρει πληροφορία, να ιεραρχεί στόχους και να λειτουργεί αποδοτικά ακόμα και με χαμηλή ροή δεδομένων, κάτι που είναι σύνηθες σε πραγματικά συστήματα μακράς εμβέλειας.

7.4 Επίλογος

Στο παρόν κεφάλαιο παρουσιάστηκε η πρακτική εφαρμογή και η λειτουργική επαλήθευση του συστήματος που σχεδιάστηκε. Μέσα από μια σειρά έξι διακριτών σεναρίων προσομοίωσης, καλύφθηκε ένα ευρύ φάσμα ρομποτικών συμπεριφορών, από την απλή αντίδραση σε εμπόδια (IR, Bump) μέχρι τη σύνθετη χαρτογράφηση χώρου (LiDAR) και τη στοχευμένη πλοήγηση (Sonar, Radar, Heat).

Τα αποτελέσματα κατέδειξαν ότι η προτεινόμενη αρχιτεκτονική επιτυγχάνει τους στόχους της:

1. **Ορθότητα Αλγορίθμων:** Οι κανόνες συμπεριφοράς λειτούργησαν προβλέψιμα και αξιόπιστα σε όλα τα σενάρια, επιβεβαιώνοντας τη χρησιμότητα του RuleBasedController.
2. **Εποπτεία:** Η οπτικοποίηση των αισθητήρων παρείχε άμεση και κατανοητή ανατροφοδότηση για το "τι βλέπει" το ρομπότ, καθιστώντας την εφαρμογή ένα ισχυρό εκπαιδευτικό εργαλείο.

3. **Ευελιξία:** Η δυνατότητα δυναμικής εναλλαγής κανόνων (Rule Editor) και η άμεση ανταπόκριση του συστήματος στις αλλαγές του περιβάλλοντος απέδειξαν ότι η πλατφόρμα μπορεί να υποστηρίξει τον πειραματισμό χωρίς χρονοβόρες διαδικασίες επαναπρογραμματισμού.

Συνοψίζοντας, η υλοποίηση απέδειξε ότι η τεχνολογία της Επαυξημένης Πραγματικότητας, συνδυασμένη με μια αρθρωτή αρχιτεκτονική λογισμικού, μπορεί να προσομοιώσει πειστικά πολύπλοκα συστήματα ρομποτικής, γεφυρώνοντας το χάσμα μεταξύ της θεωρητικής σχεδίασης αλγορίθμων και της φυσικής τους υλοποίησης.

Κεφάλαιο 8: Συμπεράσματα και Μελλοντικές Επεκτάσεις

8.1 Σύνοψη και Συμπεράσματα

Η παρούσα διπλωματική εργασία ασχολήθηκε με τη σχεδίαση και την ανάπτυξη ενός ολοκληρωμένου συστήματος προσομοίωσης ρομποτικής, το οποίο αξιοποιεί την τεχνολογία της Επαυξημένης Πραγματικότητας (AR) για να γεφυρώσει το χάσμα μεταξύ της θεωρητικής αλγοριθμικής σχεδίασης και της φυσικής υλοποίησης.

Τα βασικά συμπεράσματα που προκύπτουν από την υλοποίηση και τις δοκιμές είναι τα εξής:

1. **Η Αξία της Αρθρωτής Αρχιτεκτονικής:** Η μετάβαση από μια μονολιθική σχεδίαση σε μια αρχιτεκτονική βασισμένη σε Κανόνες (Rule-Based Architecture) και Δεδομένα (ScriptableObjects) αποδείχθηκε καθοριστική. Επέτρεψε τη δημιουργία σύνθετων συμπεριφορών (π.χ. διαφυγή από λαβύρινθο, αναζήτηση στόχων) συνδυάζοντας απλές δομικές μονάδες, χωρίς την ανάγκη συγγραφής νέου κώδικα για κάθε σενάριο.
2. **Ρεαλισμός μέσω AR:** Η χρήση του AR Foundation δεν προσέφερε απλώς οπτικό εντυπωσιασμό, αλλά ουσιαστική εκπαιδευτική αξία. Η δυνατότητα του χρήστη να τοποθετεί εικονικές πίστες στον φυσικό του χώρο και να παρατηρεί τις ακτίνες των αισθητήρων (Gizmos) να αλληλεπιδρούν με τα εμπόδια, καθιστά κατανοητές αφηρημένες έννοιες όπως η "δέσμη LiDAR" ή ο "κώνος υπερήχων".
3. **Υβριδική Λειτουργία:** Η επιτυχής διασύνδεση της μηχανής Unity με το φυσικό ρομπότ (BBC micro:bit) μέσω Bluetooth Low Energy απέδειξε ότι η προσομοίωση μπορεί να λειτουργήσει ως "ψηφιακό δίδυμο" (Digital Twin). Ο χρήστης μπορεί να επαληθεύσει τη λογική του στο κινητό και να δει το φυσικό ρομπότ να εκτελεί τις ίδιες κινήσεις, παρά τις αναπόφευκτες καθυστερήσεις του δικτύου.
4. **Επεκτασιμότητα:** Το σύστημα σχεδιάστηκε με γνώμονα την επέκταση. Η προσθήκη ενός νέου αισθητήρα ή μιας νέας συμπεριφοράς απαιτεί απλώς την υλοποίηση μιας νέας κλάσης που κληρονομεί από τις βασικές διεπαφές, χωρίς να επηρεάζει τον πυρήνα της εφαρμογής. Ένα ακόμα προτέρημα είναι ότι η εφαρμογή μπορεί να συνδεθεί με οποιοδήποτε ρομπότ και μικροελεγκτή μέσω BLE, αρκεί ο δέκτης εντολών να μπορεί να ερμηνεύσει τις απλές εντολές που εκπέμπει η εφαρμογή.

8.2 Περιορισμοί του Συστήματος

Παρά την επιτυχή λειτουργία, εντοπίστηκαν ορισμένοι περιορισμοί που οφείλονται κυρίως στο υλικό και τις τρέχουσες τεχνολογικές δυνατότητες:

- **Αστάθεια AR Tracking:** Η σταθερότητα των εικονικών αντικειμένων εξαρτάται σε μεγάλο βαθμό από τον φωτισμό του περιβάλλοντος και την υφή των επιφανειών. Σε μονόχρωμα δάπεδα ή συνθήκες χαμηλού φωτισμού, παρατηρήθηκε ολίσθηση (drift) της πίστας.
- **Latency Επικοινωνίας:** Η επικοινωνία BLE, αν και ενεργειακά αποδοτική, εισάγει μία μικρή καθυστέρηση ανάμεσα στην προσομοίωση και την εκτέλεση ενεργειών από το πραγματικό ρομπότ. Αυτό καθιστά το σύστημα ακατάλληλο για εφαρμογές που απαιτούν αντανακλαστικά χιλιοστών του δευτερολέπτου.

8.3 Μελλοντικές Επεκτάσεις

Η παρούσα εφαρμογή αποτελεί μια σταθερή βάση πάνω στην οποία μπορούν να αναπτυχθούν νέες λειτουργίες. Προτείνονται οι ακόλουθες επεκτάσεις για μελλοντική έρευνα:

1. **Ενσωμάτωση Μηχανικής Μάθησης (Machine Learning):** Αντικατάσταση ή ενίσχυση του RuleBasedController με πράκτορες Ενισχυτικής Μάθησης (Reinforcement Learning) μέσω του Unity ML-Agents. Το ρομπότ θα μπορούσε να "μάθει" μόνο του πώς να βγαίνει από τον λαβύρινθο μετά από χιλιάδες επαναλήψεις, αντί να προγραμματιστεί με κανόνες.
2. **Συστήματα Πολλαπλών Ρομπότ (Swarm Robotics):** Επέκταση της αρχιτεκτονικής για την υποστήριξη πολλών ρομπότ ταυτόχρονα στην ίδια πίστα AR. Αυτό θα επέτρεπε τη μελέτη συνεργατικών συμπεριφορών (π.χ. ομαδική εξερεύνηση, σχηματισμοί σμήνους) και την επικοινωνία μεταξύ των ρομπότ.
3. **Υποστήριξη IoT και Cloud:** Δημιουργία μιας βάσης δεδομένων στο cloud όπου θα αποθηκεύονται στατιστικά στοιχεία από τις προσομοιώσεις, όπως χρόνος επίλυσης λαβυρίνθου ή αριθμός συγκρούσεων. Αυτό θα επέτρεπε σε κάποιον ερευνητή να παρακολουθεί τις αποδόσεις της προσομοίωσης εξ αποστάσεως.
4. **Προηγμένοι Αισθητήρες Όρασης:** Προσθήκη προσομοίωσης κάμερας με δυνατότητες Υπολογιστικής Όρασης (Computer Vision). Το ρομπότ θα μπορούσε να επεξεργάζεται εικόνα (εικονική κάμερα Unity) για να αναγνωρίζει αντικείμενα (π.χ. σήματα τροχαίας, QR Codes) χρησιμοποιώντας ελαφριά νευρωνικά δίκτυα.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] R. T. Azuma, "A Survey of Augmented Reality," *Presence: Teleoperators and Virtual Environments*, vol. 6, no. 4, pp. 355–385, Aug. 1997.

[2] O. Bimber and R. Raskar, *Spatial Augmented Reality: Merging Real and Virtual Worlds*. CRC Press, 2005.

[3] Y. Weng et al., "Occlusion Culling for Augmented Reality on Mobile Devices,", 2019 [Online]. Available: https://arxiv.org/abs/1907.03125.

[4] Google Developers, "Fundamental concepts of ARCore," [Online]. Available: https://developers.google.com/ar/develop/fundamentals.

[5] S. Tzima, G. Styliaras, and A. Bassounas, "Augmented Reality Applications in Education: Teachers Point of View," *Education Sciences*, vol. 9, no. 2, p. 99, 2019. [Online]. Available: https://doi.org/10.3390/educsci9020099.

[6] P. Dhar, T. Rocks, R. M. Samarasinghe, G. Stephenson, and C. Smith, "Augmented reality in medical education: students' experiences and learning outcomes," *Medical Education Online*, vol. 26, no. 1, p. 1953953, Dec. 2021. [Online]. Available: https://doi.org/10.1080/10872981.2021.1953953.

[7] D. Kamińska et al., "Augmented Reality: Current and New Trends in Education," *Electronics*, vol. 12, no. 16, p. 3531, 2023. [Online]. Available: https://doi.org/10.3390/electronics12163531.

[8] Cosmote, "Chronos Akropoli Virtual Tour," [Online]. Available: https://www.cosmote.gr/static/cosmote/en/cosmote-chronos-akropoli-virtual-tour.

[9] K. S. Tang, D. L. Cheng, E. Mi, and P. B. Greenberg, "Augmented reality in medical education: a systematic review," *Canadian Medical Education Journal*, vol. 11, no. 1, pp. e81–e96, 2020. [Online]. Available: https://doi.org/10.36834/cmej.61705.

[10] Euphoria XR, "Augmented Reality in Healthcare," [Online]. Available: https://euphoriaxr.com/augmented-reality-in-healthcare.

[11] Pepsi Max, "Pepsi Max Bus Shelter AR Experience," Grand Visual. [Online]. Available: https://grandvisual.com/work/pepsi-max-bus-shelter/.

[12] L'Oréal Paris, "Beauty Genius AI Virtual Assistant," [Online]. Available: https://www.lorealparisusa.com/beauty-genius-ai-beauty-virtual-assistant.

[13] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*, 2nd ed. Springer, 2016.

[14] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2005.

[15] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte Carlo localization for mobile robots," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 1999.

[16] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in *Autonomous Robot Vehicles*, Springer, 1990, pp. 167–193.

[17] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robotics & Automation Magazine*, vol. 13, no. 2, pp. 99–110, Jun. 2006.

[18] S. Thrun, "Learning metric-topological maps for indoor mobile robot navigation," *Artificial Intelligence*, vol. 99, no. 1, pp. 21–71, 1998.

[19] L. Liu et al., "Path planning techniques for mobile robots: Review and prospect," *Expert Systems with Applications*, vol. 227, 120254, 2023.

[20] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[21] A. Stentz, "Optimal and efficient path planning for partially-known environments," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 1994.

[22] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[23] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots*. MIT Press, 2011.

[24] R. Szeliski, *Computer Vision: Algorithms and Applications*, 2nd ed. Springer, 2022.

[25] C. Cadena et al., "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016.

[26] A. J. Davison et al., "MonoSLAM: Real-time single camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[27] D. Scaramuzza and F. Fraundorfer, "Visual odometry [Tutorial]," *IEEE Robotics & Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[28] L. von Stumberg et al., "From monocular SLAM to autonomous drone exploration," in *Proc. 2017 European Conference on Mobile Robots (ECMR)*, Paris, France, 2017, pp. 1–8.

[29] P. Chen et al., "A Review of Research on SLAM Technology Based on the Fusion of LiDAR and Vision," *Sensors*, vol. 22, 2022.

[30] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2004.

[31] N. Kemsaram, A. Das, and G. Dubbelman, "A Stereo Perception Framework for Autonomous Vehicles," in *Proc. IEEE Intelligent Vehicles Symposium*, 2020.

[32] Y. Zhang et al., "UnrealStereo: Controlling Hazardous Factors to Analyze Stereo Vision," in *Proc. 2018 International Conference on 3D Vision (3DV)*, 2018.

[33] H. Hirschmüller, "Stereo processing by semiglobal matching and mutual information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 328–341, 2008.

[34] P. Henry et al., "RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments," *The International Journal of Robotics Research*, vol. 31, no. 5, pp. 647–663, 2012.

[35] R. Zlot and M. Bosse, "Efficient Large-scale Three-dimensional Mobile Mapping for Underground Mines," *Journal of Field Robotics*, vol. 31, pp. 758–779, 2014.

[36] J. Levinson et al., "Towards fully autonomous driving: Systems and algorithms," in *Proc. 2011 IEEE Intelligent Vehicles Symposium*, 2011, pp. 163–168.

[37] J. Kim et al., "Empirical Analysis of Autonomous Vehicle's LiDAR Detection Performance Degradation for Actual Road Driving in Rain and Fog," *Sensors*, vol. 23, no. 2972, 2023.

[38] E. D. Kaplan and C. J. Hegarty, *Understanding GPS/GNSS: Principles and Applications*. Artech House, 2017.

[39] Z. Wang, Y. Wu, and Q. Niu, "Multi-Sensor Fusion in Automated Driving: A Survey," *IEEE Access*, vol. 8, pp. 2847–2868, 2020.

[40] S. Blackmore et al., "Robotic agriculture - the future of agricultural mechanisation?," in *Proc. 5th European Conference on Precision Agriculture*, 2005.

[41] P. D. Groves, *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*. Artech House, 2013.

[42] N. O. Tippenhauer et al., "On the requirements for successful GPS spoofing attacks," in *Proc. 18th ACM Conference on Computer and Communications Security (CCS '11)*, 2011.

[43] P. D. Groves, *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*, 2nd ed. Boston, MA: Artech House, 2013.

[44] A. M. Sabatini, "Quaternion-based strap-down integration method for applications of inertial sensing to gait analysis," *Medical & Biological Engineering & Computing*, vol. 44, no. 11, pp. 941–950, Nov. 2006.

[45] O. J. Woodman, "An introduction to inertial navigation," University of Cambridge, Computer Laboratory, Tech. Rep. UCAM-CL-TR-696, Aug. 2007.

[46] J. A. Farrell and M. Barth, *The Global Positioning System and Inertial Navigation*. New York: McGraw-Hill, 1999.

[47] D. Gebre-Egziabher, G. H. Elkaim, J. D. Powell, and B. W. Parkinson, "Calibration of Strapdown Magnetometers in Magnetic Field Domain," *Journal of Aerospace Engineering*, vol. 19, no. 2, pp. 87–102, Apr. 2006.

[48] J. Borenstein, L. Ojeda, and S. Kwanmuang, "Heuristic Reduction of Gyro Drift in IMU-based Personnel Tracking Systems," in *Proc. SPIE 7306, Unmanned Systems Technology XI*, Orlando, FL, 2009, Art. no. 730612.

[49] D. H. Titterton and J. L. Weston, *Strapdown Inertial Navigation Technology*, 2nd ed. Stevenage, U.K.: IET, 2004.

[50] J. Fraden, *Handbook of Modern Sensors: Physics, Designs, and Applications*, 4th ed. New York: Springer, 2010.

[51] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots*, 2nd ed. Cambridge, MA: MIT Press, 2011.

[52] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*, 2nd ed. Berlin, Germany: Springer-Verlag, 2016.

[53] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA: MIT Press, 2005.

[54] R. J. Urick, *Principles of Underwater Sound*, 3rd ed. New York: McGraw-Hill, 1983.

[55] J. Borenstein and Y. Koren, "Histogramic in-motion mapping for mobile robot obstacle avoidance," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 4, pp. 535–539, Aug. 1991.

[56] J. J. Leonard and H. F. Durrant-Whyte, *Directed Sonar Sensing for Mobile Robot Navigation*. Boston, MA: Kluwer Academic Publishers, 1992.

[57] E. Prassler, A. Ritter, C. Schaeffer, and P. Fiorini, "A short history of cleaning robots," *Autonomous Robots*, vol. 9, no. 3, pp. 211–226, 2000.

[58] J. C. Kinsey, R. M. Eustice, and L. L. Whitcomb, "A survey of underwater vehicle navigation: Recent advances and new challenges," in *IFAC Conference of Manoeuvring and Control of Marine Craft*, Lisbon, Portugal, 2006, pp. 1–12.

[59] G. M. Hoffmann, H. Huang, S. L. Waslander, and C. J. Tomlin, "Quadrotor helicopter flight dynamics and control: Theory and experiment," in *Proc. AIAA Guidance, Navigation and Control Conference and Exhibit*, Hilton Head, SC, 2007, Art. no. 6461.

[60] M. I. Skolnik, Ed., *Radar Handbook*, 3rd ed. New York: McGraw-Hill, 2008.

[61] S. M. Patole, M. Torlak, D. Wang, and M. Ali, "Automotive radars: A review of signal processing techniques," *IEEE Signal Processing Magazine*, vol. 34, no. 2, pp. 22–35, Mar. 2017.

[62] M. A. Richards, *Fundamentals of Radar Signal Processing*, 2nd ed. New York: McGraw-Hill Education, 2014.

[63] A. Roth, "Challenges and trends in automotive radar," *Microwave Journal*, vol. 63, no. 3, pp. 26–36, 2020.

[64] S. Särkkä, *Bayesian Filtering and Smoothing*. Cambridge, U.K.: Cambridge University Press, 2013.

[65] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. New York: Wiley, 2001.

[66] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, Mar. 2004.

[67] R. Mahony, T. Hamel, and J.-M. Pflimlin, "Nonlinear complementary filters on the special orthogonal group," *IEEE Transactions on Automatic Control*, vol. 53, no. 5, pp. 1203–1218, Jun. 2008.

[68] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, Feb. 2002.

[69] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the Bayes tree," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, Feb. 2012.

[70] F. Dellaert, "Factor graphs and GTSAM: A hands-on introduction," Georgia Institute of Technology, Tech. Rep. GT-RIM-CP&R-2012-002, 2012.

[71] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proc. IEEE International Conference on Robotics and Automation*, Rome, Italy, 2007, pp. 3565–3572.

[72] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, Mar. 2015.

[73] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018.

[74] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled Lidar inertial odometry via smoothing and mapping," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Las Vegas, NV, 2020, pp. 5135–5142.

[76] M. Barr and A. Massa, *Programming Embedded Systems: With C and GNU Development Tools*, 2nd ed. Sebastopol, CA: O'Reilly Media, 2006.

[77] M. A. Mazidi, S. Naimi, and S. Naimi, *The AVR Microcontroller and Embedded Systems: Using Assembly and C*. Pearson, 2018.

[78] M. J. Pont, *Embedded C*. Addison-Wesley, 2001.

[79] A. Kurniawan, *Internet of Things Projects with ESP32: Build exciting and powerful IoT projects using the all-new Espressif ESP32* . Packt Publishing, 2019.

[80] C. Smith, *Micro:bit IoT In C*. Leanpub, 2017.

[81] A. Eguchi, "RoboCupJunior for promoting STEM education, 21st century skills, and technological literacy," *Robotics and Autonomous Systems*, vol. 75, Part B, pp. 692–699, Jan. 2016.

[82] Yahboom, "Tiny:bit smart car for micro:bit," [Online]. Available: https://www.yahboom.net/study/Tiny_bit

[83] T. A. Mikropoulos and I. Bellou, "Educational robotics as mindtools," *Themes in Science and Technology Education*, vol. 6, no. 1, pp. 5–14, 2013.

[84] C. Cadena et al., "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception-age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016.

[85] R. C. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," *The International Journal of Robotics Research*, vol. 5, no. 4, pp. 56–68, 1986 (Published 1988 in some

collections, commonly cited as the EKF-SLAM foundation). *Σημείωση: Στο κείμενο γράφει Smith et al., 1988. Αυτή είναι η κλασική αναφορά.*

[86] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *Proc. AAAI National Conference on Artificial Intelligence*, Edmonton, AB, Canada, 2002, pp. 593–598.

[87] S. Thrun and M. Montemerlo, "The GraphSLAM algorithm with applications to large-scale mapping of urban structures," *The International Journal of Robotics Research*, vol. 25, no. 5-6, pp. 403–429, 2006.

[88] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, Oct. 2015.

[89] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 611–625, Mar. 2017.

[90] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time," in *Robotics: Science and Systems*, Berkeley, CA, 2014.

[91] G. Kim, Y. S. Park, Y. Cho, J. Jeong, and A. Kim, "MulRan: Multimodal range dataset for urban place recognition," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, Paris, France, 2020, pp. 6246–6253.

[92] T. Cieslewski, S. Choudhary, and D. Scaramuzza, "Data-efficient decentralized visual SLAM," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, Montreal, QC, Canada, 2018.

[93] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, Nara, Japan, 2007, pp. 225–234.

[94] Microsoft, "C# Documentation - Get Started with C#," Microsoft Learn. [Online]. Available: https://learn.microsoft.com/en-us/dotnet/csharp/.

[95] Microsoft, "Classes (C# Programming Guide)," Microsoft Learn. [Online]. Available: https://learn.microsoft.com/en-us/dotnet/csharp/fundamentals/types/classes.

[96] Microsoft, "Inheritance in C# and .NET," Microsoft Learn. [Online]. Available: https://learn.microsoft.com/en-us/dotnet/csharp/fundamentals/object-oriented/inheritance.

[97] Microsoft, "Abstract and Sealed Classes and Class Members (C# Programming Guide)," Microsoft Learn. [Online]. Available: https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/abstract-and-sealed-classes-and-class-members.

[98] Microsoft, "Delegates (C# Programming Guide)," Microsoft Learn. [Online]. Available: https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/delegates/.

[99] Microsoft, "Events (C# Programming Guide)," Microsoft Learn. [Online]. Available: https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/events/.

[100] Microsoft, "Interfaces - Define behavior for multiple types," Microsoft Learn. [Online]. Available: https://learn.microsoft.com/en-us/dotnet/csharp/fundamentals/types/interfaces.

[101] Microsoft, "Language Integrated Query (LINQ) (C#)," Microsoft Learn. [Online]. Available: https://learn.microsoft.com/en-us/dotnet/csharp/linq/.

[102] Microsoft, "Generics (C# Programming Guide)," Microsoft Learn. [Online]. Available: https://learn.microsoft.com/en-us/dotnet/csharp/fundamentals/types/generics.

[103] Microsoft, "Performance tips for .NET," Microsoft Learn. [Online]. Available: https://learn.microsoft.com/en-us/dotnet/core/performance/performance-best-practices.

[104] Unity Technologies, "Unity User Manual," Unity Manual. [Online]. Available: https://docs.unity3d.com/Manual/index.html.

[105] Unity Technologies, "GameObjects and Components," Unity Manual. [Online]. Available: https://docs.unity3d.com/Manual/GameObjects.html.

[106] Unity Technologies, "Order of execution for event functions," Unity Manual. [Online]. Available: https://docs.unity3d.com/Manual/ExecutionOrder.html.

[107] Unity Technologies, "Physics," Unity Manual. [Online]. Available: https://docs.unity3d.com/Manual/PhysicsSection.html.

[108] Unity Technologies, "ScriptableObjects," Unity Manual. [Online]. Available: https://docs.unity3d.com/Manual/class-ScriptableObject.html.

[109] Unity Technologies, "AR Foundation," Unity Manual. [Online]. Available: https://docs.unity3d.com/Packages/com.unity.xr.arfoundation@latest.

[110] Unity Technologies, "UI System (uGUI)," Unity Manual. [Online]. Available: https://docs.unity3d.com/Manual/UIToolkits.html.

[111] Unity Technologies, "Event System," Unity Manual. [Online]. Available: https://docs.unity3d.com/Packages/com.unity.ugui@latest/index.html?subfolder=/manual/EventSystem.html.

[112] Unity Technologies, "Auto Layout," Unity Manual. [Online]. Available: https://docs.unity3d.com/Packages/com.unity.ugui@latest/index.html?subfolder=/manual/UIAutoLayout.html.

[113] Unity Technologies, "TextMeshPro," Unity Manual. [Online]. Available: https://docs.unity3d.com/Manual/com.unity.textmeshpro.html.

[114] Unity Technologies, "Universal Render Pipeline overview," Unity Manual. [Online]. Available: https://docs.unity3d.com/Packages/com.unity.render-pipelines.universal@latest.

[115] Unity Technologies, "SRP Batcher," Unity Manual. [Online]. Available: https://docs.unity3d.com/Manual/SRPBatcher.html.

[116] Unity Technologies, “Introduction to post-processing”, Unity Manual. [Online]. Available: https://docs.unity3d.com/Manual/PostProcessingOverview.html

[117] Unity Technologies, "IL2CPP," Unity Manual. [Online]. Available: https://docs.unity3d.com/Manual/IL2CPP.html.

[118] Android Developers, "Support 64-bit architectures," Android Developers Guide. [Online]. Available: https://developer.android.com/distribute/best-practices/develop/64-bit.

[119] Unity Technologies, "Input System," Unity Manual. [Online]. Available: https://docs.unity3d.com/Packages/com.unity.inputsystem@latest.

[120] Unity Technologies, "Coroutines," Unity Manual. [Online]. Available: https://docs.unity3d.com/Manual/Coroutines.html.