



Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης
Πολυτεχνική Σχολή
Τμήμα Ηλεκτρολόγων Μηχανικών &
Μηχανικών Υπολογιστών
Εργαστήριο Επεξεργασίας Πληροφορίας και Υπολογισμών

Διπλωματική Εργασία

Εντοπισμός θέσης drone σε γνωστό 3D χάρτη με
χρήση αισθητήρων απόστασης & πλήρης κάλυψη
του χώρου

Εκπόνηση:
Τσιάκας Κοσμάς
ΑΕΜ: 8255

Επίβλεψη:
Αν. Καθ. Συμεωνίδης
Ανδρέας
Δρ. Τσαρδούλιας
Εμμανουήλ

Θεσσαλονίκη, Ιούνιος 2019

The best way to predict the future is to invent it.
— Alan Kay

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω θερμά τον καθηγητή μου κ. Ανδρέα Συμεωνίδη για την εμπιστοσύνη που μου έδειξε αρχικά κατά την ενασχόληση μου με την ομάδα Robotics 4 All και στη συνέχεια με την ανάθεση και την εκπόνηση αυτής της διπλωματικής εργασίας. Θα ήθελα επίσης να ευχαριστήσω τον Μεταδιδακτορικό Ερευνητή του τμήματος, Δρ. Εμμανουήλ Τσαρδούλια για την καθοδήγηση του, την πολύτιμη βοήθεια και την εξαιρετική συνεργασία των τελευταίων μηνών. Τέλος, θα ήθελα να ευχαριστήσω τον ερευνητή του τμήματος Αλέξανδρο Φιλοθέου, για τις συμβουλές του κατά τη διάρκεια εκπόνησης της εργασίας.

Θα ήθελα επίσης να ευχαριστήσω τους γονείς μου, Χρήστο και Κωνσταντινιά, καθώς και την αδερφή μου Φένια, για τη στήριξη, την αγάπη τους και την παρακίνηση τους για το καλύτερο όλα αυτά τα χρόνια.

Τέλος, θα ήθελα να ευχαριστήσω όλους τους συγγενείς, φίλους και γνωστούς για την όποια συνεισφορά τους τόσο σε επαγγελματικό όσο και σε προσωπικό επίπεδο.

Περίληψη

Τα τελευταία χρόνια έχει παρατηρηθεί ραγδαία εξέλιξη του κλάδου της ρομποτικής και ιδιαίτερα αυτού των Συστήματων μη Επανδρωμένων Αεροσκαφών (ΣυμΕΑ). Η χρήση τους, ενώ αρχικά ήταν κυρίως στρατιωτική, αυξάνεται συνεχώς σε πληθώρα εφαρμογών, όπως η εξερεύνηση δυσπρόσιτων περιοχών, η κινηματογράφηση εκδηλώσεων, η αυτοματοποιημένη απογραφή προϊόντων κ.α.

Η χρήση των μη επανδρωμένων αεροσκαφών σε κλειστούς χώρους απαιτεί πολύ καλή αντίληψη του περιβάλλοντος, άμεση απόκριση σε μεταβολές αυτού και συνεπώς αξιόπιστη εκτίμηση της θέσης τους μέσα στον χώρο. Για την επίτευξη αυτών των στόχων απαιτείται η χρήση όλων των αισθητήρων που διαθέτει το ρομπότ. Η πλοιόγηση του drone με βέλτιστο τρόπο μέσα στο χώρο απαιτεί την ύπαρξη ενός εκ των προτέρων γνωστού πλάνου πτήσης το οποίο θα οδηγήσει στην επίτευξη του στόχου.

Η παρούσα διπλωματική εργασία εστιάζει στην επίλυση του προβλήματος της αυτόνομης και συνεχούς απογραφής προϊόντων σε οποιονδήποτε γνωστό χώρο. Με την χρήση των drones η διαδικασία αυτή απλουστεύεται με επιθυμητό αποτέλεσμα τον προσδιορισμό της θέσης των προϊόντων με ακρίβεια μερικών εκατοστών. Το πρόβλημα αυτό αποτελείται από δύο υπο-προβλήματα: α) αυτό του εντοπισμού θέσης στον κλειστό χώρο, και β) αυτό της πλήρους κάλυψης του χώρου αυτόνομα.

Για την αντιμετώπιση των παραπάνω προβλημάτων χρησιμοποιείται γνωστός τρισδιάστατος χάρτης μορφής OctoMap. Κατά τη διάρκεια της έρευνας, υλοποιήθηκε ένας αλγόριθμος που βασίζεται σε φίλτρο σωματιδίων (Particle Filter) και αξιοποιεί ένα σύνολο αποστάσεων περιμετρικά του drone για τον υπολογισμό της θέσης. Η πλοιόγηση στον χώρο πραγματοποιείται με την χρήση ενός PID ελεγκτή θέσης και εξασφαλίζει την κίνηση με αποφυγή των γνωστών στατικών εμποδίων. Για την πλήρη κάλυψη του χώρου αρχικά πραγματοποιείται μια επιλογή των σημείων που πρέπει να διασχίσει το ρομπότ και στην συνέχεια με την ένωση αυτών δημιουργείται το τελικό μονοπάτι.

Τέλος, πραγματοποιήθηκε μια σειρά πειραμάτων τα οποία αρχικά εξετάζουν την αξιοπιστία του συστήματος εντοπισμού θέσης σε τρία είδη κινήσεων, καθώς και με διαφορετικές ταχύτητες σε κάθε μία από αυτές τις περιπτώσεις. Ταυτόχρονα, εξετάστηκαν διάφοροι τρόποι κίνησης στο χώρο, πραγματοποιώντας κάλυψη του χώρου με χρήση διαφορετικών χαρακτηριστικών του αισθητήρα ανά περίπτωση. Τα πειράματα αυτά πραγματοποιήθηκαν εξ' ολοκλήρου σε περιβάλλον προσομοίωσης.

Κοσμάς Τσιάκας
ktsiakas@ece.auth.gr

Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Γιπολογιστών
Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης
Ιούνιος 2019

Title

Autonomous aerial vehicle localization, path planning and navigation
towards full and optimal 3D coverage of a known environment

Abstract

Over the last years a rapid evolution of the robotics industry, and particularly that of the Unmanned Aerial Vehicles, has been observed. Despite the fact that they were primarily used for military applications, nowadays their use is constantly increasing in a variety of applications, such as the inspection of inaccessible for humans environments, events cinematography, autonomous and real-time inventorying, etc.

The use of unmanned aerial vehicles in indoor environments requires a great perception of the surrounding environment, immediate response to its changes and consequently a robust position estimation. In order to achieve these, the use of multiple available sensors is required by the drone. The navigation in an optimal way inside the environment requires the existence of an already known flight plan that will lead to the desired goal.

The present Diploma Thesis focuses on implementation of algorithms for solving the problem of fast, reliable and low-cost inventorying in the Logistics industry. The usage of drones simplifies this procedure and aims to determine every product's position with a few centimeters accuracy. This problem consists of two subproblems: a) the position estimation in the indoor environment and b) the autonomous full coverage of the area.

In order to successfully tackle the problems described above, a known 3D map in OctoMap format is used. During the research, a Particle Filter based algorithm that uses an array of distance sensors around the drone was implemented, in order to track the pose of the robot against the known map. Navigation is based on a PID position controller that ensures an obstacle free path. As for the full coverage, an extraction of the targets and then their optimal succession is performed.

Finally, a series of experiments were carried out to examine the robustness of the positioning system in three types of motion, as well as different speeds in each of these cases. At the same time, various ways of traversing the environment were examined by using different configurations of the sensor that performs the area coverage. The experiments were entirely performed in a simulated environment.

Kosmas Tsiakas
ktsiakas@ece.auth.gr
Electrical & Computer Engineering Department,
Aristotle University of Thessaloniki, Greece
June 2019

Περιεχόμενα

Ευχαριστίες	iii
Περίληψη	v
Abstract	vii
Ακρωνύμια	xvii
1 Εισαγωγή	1
1.1 Περιγραφή του Προβλήματος	2
1.2 Σκοπός - Συνεισφορά της Διπλωματικής Εργασίας	3
1.3 Διάρθρωση της Αναφοράς	4
2 Επισκόπηση της Ερευνητικής Περιοχής	7
2.1 Εντοπισμός Θέσης	7
2.2 Πλήρης Κάλυψη Χώρου	9
3 Θεωρητικό υπόβαθρο & Εργαλεία	13
3.1 Αρχή λειτουργίας των UAV	13
3.2 Proportional Integral Derivative (PID) ελεγκτής	15
3.3 Robot Operating System (ROS)	16
3.4 Hector Quadrotor	18
3.5 Octomap	19
3.6 Βιβλιοθήκη Particle Filter	19
3.7 The Open Motion Planning Library (OMPL)	20
4 Ύλοποιήσεις	23
4.1 Εντοπισμός θέσης	24
4.2 Πλήρης κάλυψη χώρου	31
5 Πειράματα - Αποτελέσματα	41
5.1 Πειράματα εντοπισμού θέσης	42
5.1.1 Κίνηση σε ευθεία γραμμή	43
5.1.2 Κίνηση σε σπιράλ	48
5.1.3 Κίνηση σε μαίανδρο	53
5.2 Πειράματα κάλυψης χώρου	58
6 Συμπεράσματα	65
6.1 Γενικά Συμπεράσματα	65
6.2 Προβλήματα	66

ΠΕΡΙΕΧΟΜΕΝΑ

7 Μελλοντικές επεκτάσεις	69
Βιβλιογραφία	73

Κατάλογος Σχημάτων

3.1 Δομή ενός Quadcopter	14
3.2 Περιστροφή και ευθεία κίνηση ενός Quadcopter	14
3.3 Ο τρόπος λειτουργίας ενός PID ελεγκτή	15
3.4 Η αρχιτεκτονική του ROS	17
3.5 Το μοντέλο του Hector Quadrotor στον προσομοιωτή Gazebo	18
3.6 Ογκομετρική απεικόνιση σε octree και το αντίστοιχο οκταδικό δέντρο	19
3.7 Σχεδιασμός μονοπατιού με την OMPL	21
4.1 Το μοντέλο του drone που χρησιμοποιήθηκε	23
4.2 Αρχικοποίηση σωματιδίων	25
4.3 Διαδικασία ενός φίλτρου σωματιδίων	27
4.4 Τα στοιχεία του μοντέλου για έναν αισθητήρα μέτρησης απόστασης	28
4.5 Μετάβαση του drone από το ύψος Α στο ύψος Β	32
4.6 Σφάλμα του χάρτη στην αναπαράσταση επιφανειών	34
4.7 Εύρεση σημείων που δεν λαμβάνονται υπόψη στον υπολογισμό του όγκου ενός OctoMap	35
4.8 Οριζόντια και κάθετη ένωση των σημείων για επαναδιάταξη στον τρισδιάστατο χώρο	37
4.9 Τελευταίο στάδιο επεξεργασίας μονοπατιού κάλυψης	37
4.10 Δημιουργία μονοπατιού στο χώρο με αποφυγή εμποδίων	38
4.11 Κάλυψη χώρου από το drone σε πραγματικό χρόνο	39
4.12 Ολοκληρωμένη κάλυψη χώρου από το drone	39
5.1 Κίνηση σε ευθεία γραμμή	43
5.2 Οπτικοποίηση καλύτερης πορείας για κίνηση σε ευθεία γραμμή σε διάδρομο	46
5.3 Οπτικοποίηση καλύτερης πορείας για κίνηση σε ευθεία γραμμή σε αποθήκη	47
5.4 Κίνηση σε σπιράλ	48
5.5 Οπτικοποίηση καλύτερης πορείας για κίνηση σε σπιράλ πορεία σε κενό χώρο	52
5.6 Οπτικοποίηση καλύτερης πορείας για κίνηση σε σπιράλ πορεία σε αποθήκη	52
5.7 Κίνηση σε μαίανδρο	53
5.8 Οπτικοποίηση καλύτερης πορείας για κίνηση σε πορεία μαιάνδρου σε κενό χώρο	56

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

5.9 Οπτικοποίηση καλύτερης πορείας για κίνηση σε πορεία μαιάνδρου σε αποθήκη	57
5.10 Περιβάλλοντα που χρησιμοποιήθηκαν στα πειράματα κάλυψης χώρου	58
5.11 Ποσοστό κάλυψης συναρτήσει του χρόνου για το περιβάλλον του διαδρόμου με στενό τύπο αισθητήρα	59
5.12 Ποσοστό κάλυψης συναρτήσει του χρόνου για το περιβάλλον του διαδρόμου με ευρύ τύπο αισθητήρα	59
5.13 Ποσοστό κάλυψης συναρτήσει του χρόνου για το περιβάλλον της αποθήκης με στενό τύπο αισθητήρα	61
5.14 Ποσοστό κάλυψης συναρτήσει του χρόνου για το περιβάλλον της αποθήκης με ευρύ τύπο αισθητήρα	61

Κατάλογος πινάκων

5.1	Χαρακτηριστικά συστήματος που χρησιμοποιήθηκε για την εκτέλεση των πειραμάτων	41
5.2	Εκδόσεις βιβλιοθηκών και εργαλείων που χρησιμοποιήθηκαν	41
5.3	Τιμές παραμέτρων αλγορίθμου εντοπισμού θέσης	42
5.4	Μορφή δεδομένων για εξαγωγή αποτελεσμάτων του localization	42
5.5	Σφάλμα θέσης (σε μέτρα) για κίνηση σε ευθεία γραμμή με χαμηλή ταχύτητα σε διάδρομο	44
5.6	Σφάλμα θέσης (σε μέτρα) για κίνηση σε ευθεία γραμμή με κανονική ταχύτητα σε διάδρομο	44
5.7	Σφάλμα θέσης (σε μέτρα) για κίνηση σε ευθεία γραμμή με υψηλή ταχύτητα σε διάδρομο	44
5.8	Σφάλμα θέσης (σε μέτρα) για κίνηση σε ευθεία γραμμή με χαμηλή ταχύτητα σε αποθήκη	45
5.9	Σφάλμα θέσης (σε μέτρα) για κίνηση σε ευθεία γραμμή με κανονική ταχύτητα σε αποθήκη	45
5.10	Σφάλμα θέσης (σε μέτρα) για κίνηση σε ευθεία γραμμή με υψηλή ταχύτητα σε αποθήκη	45
5.11	Σφάλμα θέσης (σε μέτρα) για κίνηση σε σπιράλ με χαμηλή ταχύτητα σε κενό χώρο	48
5.12	Σφάλμα θέσης (σε μέτρα) για κίνηση σε σπιράλ με κανονική ταχύτητα σε κενό χώρο	49
5.13	Σφάλμα θέσης (σε μέτρα) για κίνηση σε σπιράλ με υψηλή ταχύτητα σε κενό χώρο	49
5.14	Σφάλμα θέσης (σε μέτρα) για κίνηση σε σπιράλ με χαμηλή ταχύτητα σε αποθήκη	49
5.15	Σφάλμα προσανατολισμού (σε rad) για κίνηση σε σπιράλ με χαμηλή ταχύτητα σε αποθήκη	50
5.16	Σφάλμα θέσης (σε μέτρα) για κίνηση σε σπιράλ με κανονική ταχύτητα σε αποθήκη	50
5.17	Σφάλμα προσανατολισμού (σε rad) για κίνηση σε σπιράλ με κανονική ταχύτητα σε αποθήκη	50
5.18	Σφάλμα θέσης (σε μέτρα) για κίνηση σε σπιράλ με υψηλή ταχύτητα σε αποθήκη	51
5.19	Σφάλμα προσανατολισμού (σε rad) για κίνηση σε σπιράλ με υψηλή ταχύτητα σε αποθήκη	51

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

5.20 Σφάλμα θέσης (σε μέτρα) για κίνηση σε μαίανδρο με χαμηλή ταχύτητα σε κενό χώρο	53
5.21 Σφάλμα προσανατολισμού (σε rad) για κίνηση σε μαίανδρο με χαμηλή ταχύτητα σε κενό χώρο	53
5.22 Σφάλμα θέσης (σε μέτρα) για κίνηση σε μαίανδρο με χανονική ταχύτητα σε κενό χώρο	54
5.23 Σφάλμα προσανατολισμού (σε rad) για κίνηση σε μαίανδρο με χανονική ταχύτητα σε κενό χώρο	54
5.24 Σφάλμα θέσης (σε μέτρα) για κίνηση σε μαίανδρο με υψηλή ταχύτητα σε κενό χώρο	54
5.25 Σφάλμα προσανατολισμού (σε rad) για κίνηση σε μαίανδρο με υψηλή ταχύτητα σε κενό χώρο	54
5.26 Σφάλμα θέσης (σε μέτρα) για κίνηση σε μαίανδρο με χαμηλή ταχύτητα σε αποθήκη	55
5.27 Σφάλμα προσανατολισμού (σε rad) για κίνηση σε μαίανδρο με χαμηλή ταχύτητα σε αποθήκη	55
5.28 Σφάλμα θέσης (σε μέτρα) για κίνηση σε μαίανδρο με χανονική ταχύτητα σε αποθήκη	55
5.29 Σφάλμα προσανατολισμού (σε rad) για κίνηση σε μαίανδρο με χανονική ταχύτητα σε αποθήκη	55
5.30 Σφάλμα θέσης (σε μέτρα) για κίνηση σε μαίανδρο με υψηλή ταχύτητα σε αποθήκη	56
5.31 Σφάλμα προσανατολισμού (σε rad) για κίνηση σε μαίανδρο με υψηλή ταχύτητα σε αποθήκη	56
5.32 Παράμετροι του αισθητήρα RFID	58
5.33 Αξιολόγηση κάλυψης χώρου για το περιβάλλον του διάδρομου	60
5.34 Αξιολόγηση κάλυψης χώρου για το περιβάλλον της αποθήκης	60

Κατάλογος Αλγορίθμων

4.1	Αλγόριθμος υπολογισμού κίνησης και προσανατολισμού	25
4.2	Αλγόριθμος υπολογισμού πιθανότητας μιας σάρωσης απόστασης . .	30
4.3	Αναδειγματοληφία χαμηλής διακύμανσης	30
4.4	Αλγόριθμος υπολογισμού καλύτερης γωνίας θέασης	33
4.5	Υπολογισμός όγκου ενός OctoMap	34
4.6	Αλγόριθμος εύρεσης βέλτιστου μονοπατιού	36

Ακρωνύμια Εγγράφου

Παρακάτω παρατίθενται ορισμένα από τα πιο συχνά χρησιμοποιούμενα ακρωνύμια της παρούσας διπλωματικής εργασίας:

AMCL	→ Adaptive Monte Carlo Localization
FOV	→ Field Of View
IMU	→ Inertial Measurement Unit
LiDAR	→ Light Detection And Ranging
OMPL	→ Open Motion Planning Library
PID	→ Proportional Integral Derivative
RGB	→ Red, Green, Blue
RFID	→ Radio-Frequency Identification
SLAM	→ Simultaneous Localization And Mapping
UAV	→ Unmanned Aerial Vehicle

1

Εισαγωγή

Η Ρομποτική είναι η επιστήμη που ασχολείται με την αντίληψη και το χειρισμό του φυσικού κόσμου μέσω συσκευών που ελέγχονται από υπολογιστές. Τα παραδείγματα πετυχημένων ρομποτικών συστημάτων περιλαμβάνουν κινητές πλατφόρμες για πλανητική εξερεύνηση, βιομηχανικούς ρομποτικούς βραχίονες σε γραμμές συναρμολόγησης, αυτοκίνητα που κινούνται χωρίς οδηγό και βραχίονες που βοηθούν τους χειρουργούς. Τα ρομποτικά συστήματα βρίσκονται στον φυσικό κόσμο, αντιλαμβάνονται πληροφορίες για το περιβάλλον τους μέσω αισθητήρων και εκτελούν χειρισμούς στο περιβάλλον τους μέσω φυσικών δυνάμεων.

Στην επιστήμη της τεχνολογίας υπάρχει παρανόηση ως προς το τι είναι ρομπότ και τι δεν είναι. Ο ορισμός της λέξης ρομπότ εξελίχθηκε με την πάροδο του χρόνου, παράλληλα με τα άλματα της έρευνας και την πρόοδο της τεχνολογίας [1].

Ρομπότ είναι ένα αυτόνομο σύστημα, το οποίο υπάρχει στο φυσικό κόσμο, αισθάνεται το περιβάλλον του και μπορεί να δράσει σε αυτό ώστε να πετύχει κάποιους στόχους.

Ένα αυτόνομο ρομπότ δρα βάσει των δικών του αποφάσεων και δεν ελέγχεται από κάποιον άνθρωπο. Η ικανότητα αίσθησης του περιβάλλοντος υποδηλώνει την ύπαρξη αισθητήρων, δηλαδή μέσων αντίληψης (π.χ., ακοή, αφή, όραση, όσφρηση, κ.λπ) που τροφοδοτούν το ρομπότ με πληροφορίες από τον κόσμο. Μια μηχανή που δεν δρα (δηλαδή δεν κινείται και δεν επηρεάζει τον κόσμο πράττωντας/αλλάζοντας κάτι) δεν θεωρείται ρομπότ.

Το 1950 ο Isaac Asimov διατύπωσε τους τρεις νόμους της Ρομποτικής, οι οποίοι βρίσκονται υπό συνεχή αναθεώρηση μέχρι σήμερα [2] και είναι οι εξής:

- Το ρομπότ δε θα κάνει κακό σε άνθρωπο, ούτε με την αδράνειά του θα επιτρέψει να βλαφτεί ανθρώπινο ον

ΚΕΦΑΛΑΙΟ 1. ΕΙΣΑΓΩΓΗ

- Το ρομπότ πρέπει να υπακούει τις διαταγές που του δίνουν οι άνθρωποι, εκτός αν αυτές οι διαταγές έρχονται σε αντίθεση με τον πρώτο νόμο
- Το ρομπότ οφείλει να προστατεύει την ύπαρξή του, εφόσον αυτό δεν συγκρούεται με τον πρώτο και τον δεύτερο νόμο

1.1 ΠΕΡΙΓΡΑΦΗ ΤΟΥ ΠΡΟΒΛΗΜΑΤΟΣ

Ως γνωστόν, η αγορά και η χρήση των *Mη Επανδρωμένων Αεροσκαφών*, γνωστά και ως drones, έχει αυξηθεί κατακόρυφα τα τελευταία χρόνια. Αρχικά, τα drones είχαν κυρίως στρατιωτική χρήση, κάτι που δημιουργεί αρκετές ηθικές και νομικές προκλήσεις, αλλά και προβληματισμό σχετικά με την παραβίαση της ιδιωτικότητας στην σύγχρονη εποχή. Επίσης, ενώ στην αρχή ήταν απαραίτητη η ύπαρξη κάποιου καταρτισμένου χειριστή, πλέον τα περισσότερα είναι αυτόνομα, μπορούν να ακολουθούν κάποιο προκαθορισμένο πλάνο πτήσης και να προσαρμόζονται στις μεταβολές του περιβάλλοντος. Η επιστημονική κοινότητα θεωρεί τα drones μία από τις τεχνολογίες του μέλλοντος, καθώς οι δυνατότητές τους επιτρέπουν την ανάπτυξη ρομποτικών εφαρμογών που επιλύουν προβλήματα σε διάφορους κλάδους, από την εξερεύνηση δυσπρόσιτων περιοχών μέχρι και την κινηματογράφηση αθλητικών εκδηλώσεων.

Κάποιες χαρακτηριστικές, αλλά όχι οι μόνες, περιπτώσεις, όπου τα μη επανδρωμένα αεροσκάφη μπορούν να χρησιμοποιηθούν είναι οι εξής:

- Η επίβλεψη και συντήρηση μεγάλων εκτάσεων ή κτισμάτων, π.χ. γέφυρες, τούνελ, υπόγεια ορυχεία κ.α.
- Η παροχή ανθρωπιστικής βοήθειας σε δυσπρόσιτες περιοχές μετά από φυσικές καταστροφές
- Η συνεχής απογραφή προϊόντων σε μεγάλες αποθήκες χωρίς την ανθρώπινη παρέμβαση
- Ο κλάδος της γεωργίας και η παρακολούθηση αγροτικών εκτάσεων
- Η μεταφορά φορτίων και δεμάτων μικρού όγκου

Το πρόβλημα στο οποίο αναφέρεται η συγκεκριμένη εργασία είναι αυτό της αυτόνομης συνεχούς απογραφής προϊόντων σε οποιονδήποτε χώρο. Η απογραφή από ανθρώπους είναι μία επίπονη διαδικασία, η οποία σχετίζεται με επιβάρυνση της υγείας τους, λόγω της επαναλαμβανόμενης κίνησης που καλούνται να κάνουν και ενδεχόμενα σφάλματα. Με την χρήση των drones η διαδικασία αυτή απλουστεύεται με επιθυμητό αποτέλεσμα τον προσδιορισμό της θέσης όλων των προϊόντων με ακρίβεια μερικών εκατοστών. Το πρόβλημα αυτό αποτελείται από τα εξής υποπροβλήματα: τον εντοπισμό θέσης του drone στον κλειστό χώρο, τον εντοπισμό θέσης των ετικετών των προϊόντων και της πλήρης κάλυψης του χώρου αυτόνομα.

1.2. ΣΚΟΠΟΣ - ΣΥΝΕΙΣΦΟΡΑ ΤΗΣ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Με τον όρο εντοπισμό θέσης ενός ρομποτικού οχήματος, αναφερόμαστε στον υπολογισμό της θέσης και του προσανατολισμού του ρομπότ ως προς έναν δεδομένο χάρτη. Η χρήση των μη επανδρωμένων αεροσκαφών σε εσωτερικούς χώρους παρουσιάζει μεγάλο ενδιαφέρον, καθώς δεν είναι δυνατή η χρήση του αισθητήρα GPS και ως συνέπεια ο άμεσος εντοπισμός της θέσης του drone στο χώρο. Οποιαδήποτε διαδικασία που θα επιτελεί το drone σε έναν κλειστό χώρο, απαιτεί πολύ καλή αντίληψη του περιβάλλοντος που βρίσκεται, άμεση απόκριση σε μεταβολές του, σταθερή και ασφαλή πλοϊγμηση. Επίσης, είναι γνωστό ότι όλοι οι αισθητήρες περιέχουν θόρυβο, συνεπώς δεν προσφέρουν απόλυτα αξιόπιστα αποτελέσματα από μόνοι τους. Για τους παραπάνω λόγους, απαιτείται η πλήρης εκμετάλλευση των υπόλοιπων διαθέσιμων αισθητήρων.

Στην περίπτωση μας, οι αισθητήρες αποτελούνται από μία κάμερα, από έναν αισθητήρα απόστασης (laser) ο οποίος μας δίνει ορισμένες αποστάσεις περιμετρικά του drone, έναν ακόμη αισθητήρα laser που υπολογίζει το ύψος που βρίσκεται και έναν αισθητήρα αδρανειακής μέτρησης (Inertial Measurement Unit). Το περιβάλλον στο οποίο βρίσκεται το drone θεωρείται γνωστό και ο χάρτης αυτού είναι διαθέσιμος εκ των προτέρων σε μορφή OctoMap [3].

Εκτός από την ορθή εκτίμηση της θέσης του, το μη επανδρωμένο αεροσκάφος πρέπει να ακολουθεί ένα προκαθορισμένο πλάνο πτήσης, το οποίο διαμορφώνεται σύμφωνα με συγκεκριμένα σημεία του χάρτη που παρουσιάζουν κάποιο ενδιαφέρον. Με τον τρόπο αυτό, εξασφαλίζεται η πλήρης τρισδιάστατη κάλυψη του χώρου. Στη συγκεκριμένη περίπτωση μελέτης, το πλάνο πτήσης μπορεί να διαμορφωθεί βάση των προϊόντων που υφίστανται προς απογραφή στο χώρο της αποθήκης. Οι αλγόριθμοι πλήρους κάλυψης ενός χώρου, προϋποθέτουν την ύπαρξη ενός μονοπατιού, το οποίο είναι προσπελάσιμο σε πεπερασμένο χρόνο. Η αξιολόγηση της κάλυψης του χώρου γίνεται με την χρήση μίας RFID κεραίας και αναγνώστη, μέσω του οποίου υπολογίζεται η θέση των αντικειμένων που βρίσκονται στο χώρο και στη συνέχεια υπολογίζεται το ποσοστό των αντικειμένων που εντοπίστηκαν από τον αναγνώστη ως προς τον συνολικό αριθμό αντικειμένων.

1.2 Σκοπος - Συνεισφορα της Διπλωματικης Εργασιας

Σκοπός της παρούσας διπλωματικής εργασίας είναι να παρουσιάσει ένα ολοκληρωμένο σύστημα εντοπισμού θέσης και πλήρους κάλυψης ενός γνωστού χώρου χρησιμοποιώντας ένα μη επανδρωμένο αεροσκάφος. Το σύστημα αυτό θα πρέπει να παρέχει στους αλγορίθμους πλοϊγμησης μια αξιόπιστη εκτίμηση της θέσης του quadcopter, καθώς και τα σημεία τα οποία θα πρέπει να διασχίσει, από τα οποία αποτελείται το πλάνο πτήσης του.

Οι μετρήσεις οι οποίες προέρχονται από τον αισθητήρα απόστασης και το IMU συνδυάζονται, έτσι ώστε να εκτιμηθεί η θέση του αεροσκάφους στο χώρο. Η πληροφορία αυτή, παρέχεται στον αλγόριθμο εντοπισμού θέσης, που βασίζεται σε ένα φίλτρο σωματιδίων.

Για την δημιουργία της πορείας που πρέπει να ακολουθήσει το drone ώστε να φτάσει σε κάποιο στόχο, χρησιμοποιείται ο αλγόριθμος RRT*, χρησιμοποιώντας τον χάρτη σε μορφή OctoMap για την αποφυγή των εμποδίων. Το μονοπάτι που

προκύπτει από αυτόν, αν χρειαστεί, ομαλοποιείται χρησιμοποιώντας συναρτήσεις B-spline. Στη συνέχεια, με τη χρήση ενός PID ελεγκτή θέσης το drone κινείται προς το στόχο αυτό.

Τέλος, για την πλήρη κάλυψη του χώρου, εξετάζεται ο χάρτης, ώστε να βρεθούν τα σημεία τα οποία προσφέρουν την καλύτερη δυνατή θέση προς τα αντικείμενα. Έχοντας τα σημεία αυτά, χρησιμοποιείται ο αλγόριθμος πλησιέστερου γείτονα (Nearest Neighbor) σε συνδυασμό με τον αλγόριθμο αναρρίχησης λόφων (Hill-Climbing Search) για να προκύψει η τελική σειρά αυτών που δίνει το συνολικό μονοπάτι.

Ο κώδικας που αναπτύχθηκε για την επίλυση του προβλήματος εφαρμόστηκε σε περιβάλλον προσομοίωσης και για αισθητήρες με συγκεκριμένα χαρακτηριστικά. Παρ' όλα αυτά, η μεταφορά σε πραγματικό drone δεν απαιτεί ιδιαίτερη προσαρμογή, καθώς τα κύρια χαρακτηριστικά έχουν παραμετροποιηθεί για εύκολη μεταβολή του συστήματος.

1.3 ΔΙΑΡΘΡΩΣΗ ΤΗΣ ΑΝΑΦΟΡΑΣ

Η διάρθρωση της παρούσας διπλωματικής εργασίας είναι η εξής:

- **Κεφάλαιο 2:** Γίνεται ανασκόπησή της ερευνητικής περιοχής που αφορά την εύρεση θέσης ενός μη επανδρωμένου αεροσκάφους σε γνωστό χώρο και την πλήρη κάλυψη του χώρου από αυτό.
- **Κεφάλαιο 3:** Περιγράφονται τα βασικά θεωρητικά στοιχεία στα οποία βασίστηκαν οι υλοποιήσεις, καθώς και τα εργαλεία που χρησιμοποιήθηκαν. Πιο συγκεκριμένα, περιγράφεται η αρχή λειτουργίας των μη επανδρωμένων αεροσκαφών και ο τρόπος λειτουργίας ενός PID ελεγκτή. Επίσης, περιγράφεται το μεσολειτουργικό σύστημα ROS, πάνω στο οποίο βασίστηκε όλη η υλοποίηση, το σύνολο ROS πακέτων Hector Quadrotor καθώς και οι βιβλιοθήκες OctoMap, Particle Filter και Open Motion Planning Library.
- **Κεφάλαιο 4:** Πλήρης περιγραφή των υλοποιήσεων των αλγορίθμων εύρεσης θέσης, πλοήγησης και πλήρους κάλυψης χώρου.
- **Κεφάλαιο 5:** Παρουσιάζεται αναλυτικά η μεθοδολογία των πειραμάτων και τα αποτελέσματα.
- **Κεφάλαιο 6:** Παρουσιάζονται τα τελικά συμπεράσματα.
- **Κεφάλαιο 7:** Αναφέρονται τα προβλήματα που προέκυψαν και προτείνονται θέματα για μελλοντική μελέτη, αλλαγές και επεκτάσεις.

2

Επισκόπηση της Ερευνητικής Περιοχής

Τα προβλήματα του εντοπισμού θέσης, της αυτόνομης πλοϊγγησης και της πλήρης κάλυψης χώρου από μη επανδρωμένα αεροσκάφη έχουν απασχολήσει έντονα την επιστημονική κοινότητα τα τελευταία χρόνια. Πληθώρα δημοσιεύσεων αναφέρουν διαφορετικές μεθόδους που προσπαθούν να επιλύσουν τα προαναφερθέντα προβλήματα. Οι λύσεις που έχουν παρουσιαστεί διαφέρουν κατά κύριο λόγο στον τύπο των αισθητήρων που χρησιμοποιούνται από τα drones, συνεπώς και την πληροφορία που είναι διαθέσιμη. Αυτό που εισάγει πολλούς περιορισμούς και διαφέρει από αντίστοιχες λύσεις για επίγεια ρομπότ είναι το βάρος το οποίο μπορούν να μεταφέρουν και η κατανάλωση ενέργειας που πρέπει να παραμένει χαμηλή, ώστε να μην μειώνεται σημαντικά η διάρκεια της πτήσης και το διαφορετικό κινηματικό μοντέλο.

Στη συνέχεια, θα εξετάσουμε ξεχωριστά για κάθε πρόβλημα τις λύσεις που έχουν προταθεί από άλλους ερευνητές.

2.1 ΕΝΤΟΠΙΣΜΟΣ ΘΕΣΗΣ

Η εύρεση της θέσης ενός ρομπότ σε εσωτερικό περιβάλλον παρουσιάζει σημαντικές δυσκολίες. Η έλλειψη του αισθητήρα GPS οδηγεί στην ανάγκη για εύρεση άλλων αξιόπιστων μεθόδων υπολογισμού της θέσης και του προσανατολισμού ενός UAV. Αρκετά συχνά συναντάμε στην βιβλιογραφία προσεγγίσεις που βασίζονται κατά κύριο λόγο σε μεθόδους υπολογιστικής όρασης. Αυτό συμβαίνει, καθώς το κόστος, το μικρό βάρος αλλά και η πληροφορία που μπορεί να παρέχει μια RGB-D κάμερα διευκολύνουν την επίλυση του προβλήματος. Το βασικό μειονέκτημα των μεθόδων αυτών είναι η πολυπλοκότητα της επεξεργασίας και οι υψηλές υπολογιστικές απαιτήσεις για επεξεργασία των δεδομένων σε πραγματικό χρόνο. Το πρόβλημα του εντοπισμού θέσης μπορεί να χωριστεί σε δύο διαφορετικές κατηγορίες, ανάλογα με το εάν είναι γνωστή η αρχική θέση του ρομπότ μέσα στο χώρο ή όχι.

ΚΕΦΑΛΑΙΟ 2. ΕΠΙΣΚΟΠΗΣΗ ΤΗΣ ΕΡΕΥΝΗΤΙΚΗΣ ΠΕΡΙΟΧΗΣ

Οι Perez-Grau κ.α. [4] προτείνουν την χρήση του κλασικού αλγορίθμου Monte Carlo Localization [5], ο οποίος χρησιμοποιεί πιθανοτικές μεθόδους και φίλτρο σωματιδίων για την εύρεση της θέσης, σε συνδυασμό με μία κάμερα RGB-D και ορισμένους ραδιοπομπούς τοποθετημένους σε γνωστά σημεία μέσα στο χάρτη. Η απαιτούμενη οδομετρία για τα σωματίδια του AMCL προέρχεται από οπτική οδομετρία (visual odometry), ενώ ο υπολογισμός της απόστασης από τους ραδιοπομπούς βοηθάει στην εξάλειψη του συσσωρευμένου σφάλματος που προκύπτει. Επίσης, κάθε λήψη του αισθητήρα βάθους δημιουργεί μια τρισδιάστατη αναπαράσταση του χώρου. Αυτή συγκρίνεται με τον ήδη υπάρχοντα χάρτη και εάν βρεθούν κοινά σημεία, υπολογίζεται μία εκτίμηση της θέσης του ρομπότ μέσα στο γνωστό περιβάλλον και η πληροφορία αυτή ενισχύει την εκτίμηση του AMCL.

Μία άλλη προσέγγιση είναι η χρήση πολλαπλών RGB καμερών, όπως παρουσιάζεται στο [6]. Παρόλο που δεν αναφέρεται σε UAV, το αποτέλεσμα είναι η εκτίμηση θέσης με 6 βαθμούς ελευθερίας, που σημαίνει ότι μπορεί να χρησιμοποιηθεί και για drones. Συγκεκριμένα, χρησιμοποιείται μια προ-επεξεργασμένη αναπαράσταση του χώρου σε μορφή τρισδιάστατων σημείων (PointCloud) και μια συνάρτηση κόστους, ώστε να υπολογιστεί η θέση με βάση την απόσταση των διαδοχικών λήψεων και την απόκλιση των λήψεων από τον γνωστό χάρτη.

Οι Ok, Greene και Roy [7] χρησιμοποιούν μία RGB-D κάμερα, μόνο για την δημιουργία του χάρτη, ενώ η διαδικασία του εντοπισμού θέσης βασίζεται αποκλειστικά σε RGB εικόνα. Συγκεκριμένα, δημιουργούν εικονικές λήψεις του περιβάλλοντος από διάφορα σημεία κατανεμημένα στο χάρτη και στη συνέχεια κάθε εικόνα που λαμβάνεται από την κάμερα συγκρίνεται με τις αρχικές λήψεις, χρησιμοποιώντας οπτικά χαρακτηριστικά. Είναι σημαντικό να αναφερθεί ότι η υλοποίηση αυτή δεν απαιτεί μεγάλη υπολογιστική ισχύ, λόγω της μείωσης του ρυθμού λήψης εικόνων.

Οι Beul κ.α. [8] αναφέρονται στην αυτόνομη πλοήγηση των UAVs σε μεγάλες αποθήκες. Οι αισθητήρες που χρησιμοποιούνται είναι ένα οριζόντιο και ένα κάθετο lidar, γωνίας 270° το οποίο είναι, τρία ζευγάρια από στερεοσκοπικές κάμερες, ένα IMU και ένας αναγνώστης RFID. Με χρήση της οπτικής οδομετρίας, υπολογίζεται η κίνηση του drone, ενώ οι αποστάσεις που προκύπτουν από τους αισθητήρες laser, μετατρέπονται σε μορφή PointCloud και συγχρίνονται οι επιφάνειες του με τις επιφάνειες του υπάρχοντα χάρτη. Όσον αφορά την πλοήγηση του drone στο χώρο, χρησιμοποιείται ο αλγόριθμος A* για την εύρεση μονοπατιών σε έγαν διακριτοποιημένο κόσμο σε μορφή OctoMap, ενώ παράλληλα χρησιμοποιείται και τοπική αποφυγή εμποδίων για την περίπτωση ύπαρξης δυναμικού περιβάλλοντος.

Μία ακόμη εφαρμογή με UAVs σε χώρο αποθήκης παρουσιάζεται στο [9]. Στην περίπτωση αυτή, χρησιμοποιούνται έναν αισθητήρας απόστασης lidar γωνίας 360° και εύρους 100 μέτρων, δύο κάμερες, ένα IMU και ένας αναγνώστης RFID. Η αποτελεσματικότητα της μεθόδου φαίνεται από το γεγονός ότι το drone είχε τη δυνατότητα να κινείται με σχετικά μεγάλη ταχύτητα (2.1 μέτρα/δευτερόλεπτο) χωρίς αυτό να επηρεάζει την αξιοπιστία της εκτίμησης θέσης και της πλοήγησης. Με την δημιουργία τρισδιάστατων προσωρινών χαρτών στους οποίους προστίθενται συνεχώς τα σημεία που προκύπτουν από τον αισθητήρα απόστασης, διευκολύνεται η εύρεση της θέσης με βάση τον αρχικό χάρτη.

Σε όλες τις προηγούμενες περιπτώσεις, το drone μπορούσε να διατηρεί μία

απόσταση ασφαλείας από τα αντικείμενα του χώρου. Αντίθετα, στο [10] χρησιμοποιείται ένα μη επανδρωμένο αεροσκάφος για τον έλεγχο και την εκτίμηση καταστροφών μέσα σε ένα περιβάλλον πλοίου με χαμηλή ορατότητα. Το περιβάλλον αυτό περιέχει στενά περάσματα, πόρτες και μικρά αντικείμενα, συνθήκες οι οποίες δεν επιτρέπουν την χρήση των μεθόδων που παρουσιάστηκαν στις προηγούμενες βιβλιογραφικές αναφορές. Για την πλοήγηση συνδυάζεται η οπτική οδομετρία που προέρχεται από μια RGB-D κάμερα με τις μετρήσεις ενός IMU, χρησιμοποιώντας ένα φίλτρο Kalman, ώστε να υπολογιστεί η ταχύτητα. Επίσης, για την βελτίωση της εκτίμησης θέσης χρησιμοποιείται ένας αλγόριθμο που χρησιμοποιεί Φίλτρα Σωματιδίων (Particle Filters) και τροφοδοτείται από την οδομετρία. Παράλληλα, χρησιμοποιείται μια υπέρυθρη κάμερα, ώστε να μην επηρεάζεται από την χαμηλή ορατότητα του περιβάλλοντος και να ανιχνεύει τις περιοχές υψηλού κινδύνου λόγω πυρκαγιάς.

2.2 ΠΛΗΡΗΣ ΚΑΛΥΨΗ ΧΩΡΟΥ

Η πλήρης κάλυψη ενός χώρου πρόκειται για το πρόβλημα της δημιουργίας ενός μονοπατιού το οποίο διέρχεται από όλα τα σημεία ενδιαφέροντος ενός περιβάλλοντος, καθώς παράλληλα γίνεται αποφυγή εμποδίων. Σύμφωνα με το [11], οι έξι απαιτήσεις της διαδικασίας πλήρους κάλυψης χώρου είναι οι εξής:

- Το ρομπότ πρέπει να διασχίσει όλα τα σημεία στην περιοχή ενδιαφέροντος, καλύπτοντας την πλήρως
- Το ρομπότ πρέπει να καλύπτει την περιοχή, χωρίς την ύπαρξη αλληλοεπικαλυπτόμενων διαδρομών
- Απαιτούνται συνεχείς και διαδοχικές διεργασίες, χωρίς την επανάληψη καμίας τροχιάς
- Το ρομπότ πρέπει να αποφεύγει κάθε είδους εμπόδιο
- Απλοϊκές τροχιές (π.χ. ευθείες ή κυκλικές κινήσεις) θα έπρεπε να χρησιμοποιούνται, καθώς προσφέρουν απλότητα στην κίνηση
- Ένα βέλτιστο μονοπάτι προτιμάται, εφόσον αυτό είναι εφικτό

Υπάρχουν δύο κατηγορίες προσεγγίσεων για το θέμα αυτό, η απόλυτη και η ευριστική. Κατά την πρώτη, ο χώρος διαχωρίζεται σε τμήματα και μπορεί να εγγυηθεί την πλήρη κάλυψη του χώρου, ενώ κατά την ευριστική μέθοδο το ρομπότ ακολουθεί ένα σύνολο απλών κανόνων που επηρεάζουν την κίνηση του και πιθανόν να μην οδηγήσουν με επιτυχία στην πλήρη κάλυψη.

Στο [12] παρουσιάζεται μία απόλυτη προσέγγιση στο πρόβλημα της πλήρους κάλυψης χώρου για UAVs. Ο χώρος χωρίζεται σε πολυγωνικά τμήματα με μέγεθος που επηρεάζεται από το πεδίο όρασης (Field of View) της κάμερας. Το κέντρο κάθε πολυγώνου θεωρείται ως το σημείο που πρέπει να βρεθεί το drone, ώστε να καλυφθεί η περιοχή γύρω του. Στη συνέχεια, χρησιμοποιείται ένας wavefront propagation αλγόριθμος για να βρει όλα τα δυνατά μονοπάτια που μπορούν να ενώσουν τα σημεία

ΚΕΦΑΛΑΙΟ 2. ΕΠΙΣΚΟΠΗΣΗ ΤΗΣ ΕΡΕΥΝΗΤΙΚΗΣ ΠΕΡΙΟΧΗΣ

αυτά. Για κάθε ένα μονοπάτι υπολογίζεται το συνολικό του κόστος και επιλέγεται αυτό με το μικρότερο δυνατό. Στο τέλος, η πορεία ομαλοποιείται χρησιμοποιώντας πολυωνυμικές συναρτήσεις. Ένα σημαντικό χαρακτηριστικό της προσέγγισης αυτής είναι ότι κατά την δημιουργία των μονοπατιών, λαμβάνεται υπόψη η περιστροφική κίνηση που απαιτείται από το drone για να μεταβεί στο επόμενο σημείο, καθώς αυτή αυξάνει το χρόνο εκτέλεσης και είναι επιθυμητή η ελαχιστοποίηση των στροφών κατά την κίνηση στο χώρο.

Οι Bircher κ.α. [13] παρουσιάζουν έναν αλγόριθμο για την τρισδιάστατη κάλυψη χώρου που στοχεύει στην αυτόνομη διαδικασία της επιθεώρησης χρησιμοποιώντας μη επανδρωμένα αεροσκάφη. Στην προσέγγιση αυτή, για κάθε σημείο του χάρτη υπολογίζουμε τη θέση η οποία επιφέρει την καλύτερη δυνατή θέαση του σημείου αυτού, με βάση τα χαρακτηριστικά των αισθητήρων που διαθέτει το ρομπότ. Στην συνέχεια, εφαρμόζεται μια επαναληπτική διαδικασία πεπερασμένων βιγμάτων, μέσω της οποίας συνδέονται τα σημεία που ελαχιστοποιούν το κόστος, είτε αυτό είναι η απόσταση είτε ο χρόνος εκτέλεσης. Η καλύτερη δυνατή σύνδεση των σημείων επιτυγχάνεται μέσω της επίλυσης του προβλήματος του περιοδεύοντος εμπόρου (Travelling Salesman Problem).

Μια αρκετά διαφορετική προσέγγιση παρουσιάζεται στο [14]. Πιο συγκεκριμένα, χρησιμοποιούνται νευρωνικά δίκτυα για την πλήρη κάλυψη στατικού και μεταβαλλόμενου χώρου σε πραγματικό χρόνο. Η υλοποίηση αυτή εφαρμόζεται για χάρτη δύο διαστάσεων, μπορεί να καλύψει την ύπαρξη ενός ή περισσοτέρων ρομποτικών πρακτόρων και παρουσιάζει αξιόπιστα αποτελέσματα. Ο χώρος διακριτοποιείται με βάση το μέγεθος του ρομπότ και τα σημεία που προκύπτουν ενώνονται μέσω των νευρώνων του δικτύου. Η δυναμική μορφή του δικτύου μπορεί να μεταβάλλει στιγμιαία τα μονοπάτια που δημιουργούνται με βάση την προσθήκη ή την αφαίρεση εμποδίων.

Οι ευριστικές προσεγγίσεις βασίζονται στην κάλυψη του χώρου με διάφορα μοτίβα, όπως για παράδειγμα υλοποιείται στο [15]. Στην περίπτωση αυτή, παρουσιάζεται ένας αλγόριθμος ο οποίος δημιουργεί ένα μονοπάτι που ικανοποιεί, εκτός των άλλων, και τους ενεργειακούς περιορισμούς του ρομπότ. Μετά την δημιουργία μιας πορείας που αποτελείται κυρίως από κινήσεις προς τα μπροστά και πίσω, εξετάζεται η ταχύτητα με την οποία πρέπει να κινείται το drone, ώστε να ελαχιστοποιείται η κατανάλωση ενέργειας.

3

Θεωρητικό υπόβαθρο & Εργαλεία

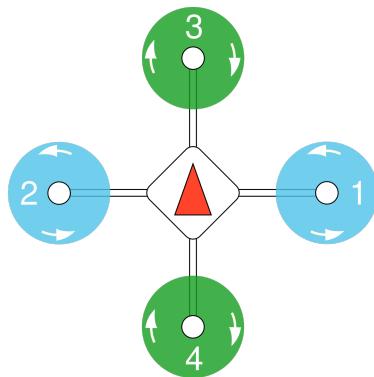
Στο κεφάλαιο αυτό παρουσιάζονται βασικά θεωρητικά στοιχεία που κρίνονται απαραίτητα για την κατανόηση της παρούσας εργασίας. Συγκεκριμένα, παρουσιάζεται η αρχή λειτουργίας των μη επανδρωμένων οχημάτων, καθώς ο τρόπος που λειτουργεί ένας ελεγκτής PID. Επίσης, παρουσιάζονται τα εργαλεία και οι βιβλιοθήκες που χρησιμοποιήθηκαν για την υλοποίηση του συστήματος εντοπισμού θέσης και πλήρους κάλυψης χώρου από ένα μη επανδρωμένο αεροσκάφος. Ειδικότερα, θα αναλυθούν το μεσολειτουργικό σύστημα ROS, πάνω στο οποίο βασίστηκε όλη η υλοποίηση, το σύνολο ROS πακέτων Hector Quadrotor και οι βιβλιοθήκες OctoMap, Particle Filter και OMPL.

3.1 ΑΡΧΗ ΛΕΙΤΟΥΡΓΙΑΣ ΤΩΝ UAV

Τα μη επανδρωμένα οχήματα χωρίζονται σε διαφορετικές κατηγορίες, ανάλογα με τον αριθμό και τη διάταξη των κινητήρων τους. Στην συγκεκριμένη εργασία, θα ασχοληθούμε με το τετρακόπτερο (Quadcopter), του οποίου η μορφή φαίνεται στο παρακάτω σχήμα.

Όπως βλέπουμε, διαθέτει τέσσερις έλικες τοποθετημένους σε ίσες αποστάσεις και συμμετρικά κατανεμημένους από το κέντρο μάζας του drone, οι οποίοι είναι υπεύθυνοι για όλες τις λειτουργίες, όπως την κάθετη απογείωση και την προσγείωση του οχήματος. Διαθέτει έξι βαθμούς ελευθερίας και τέσσερις ελεγχόμενες μεταβλητές, όσες και ο αριθμός των κινητήρων. Μέσω αυτών επιτυγχάνεται ο έλεγχος της θέσης και του ύψους στο οποίο βρίσκεται το Quadcopter.

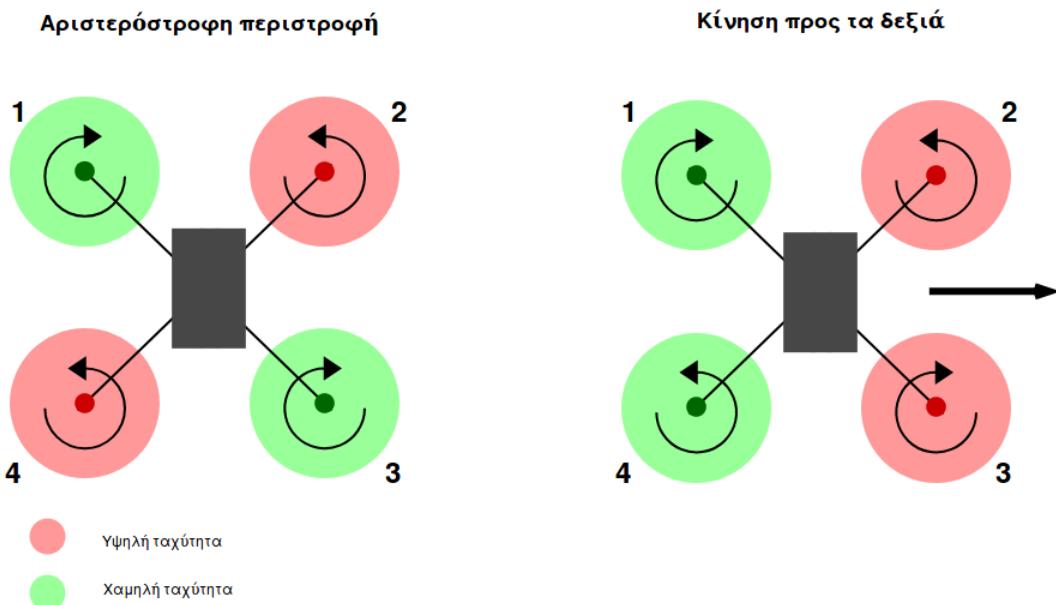
Οι πιο χαρακτηριστικές λειτουργίες είναι η ανύψωση από το έδαφος, η προσγείωση και η σταθερή αιώρηση (hover). Για την ανύψωση από το έδαφος απαιτείται οι έλικες να κινούνται με την ίδια ταχύτητα και με την κατεύθυνση που φαίνεται στο [σχήμα 3.1](#), ενώ η προσγείωση επιτυγχάνεται με την μείωση της ταχύτητας των στροφέων. Η λειτουργία hover, δηλαδή η σταθεροποίηση σε κάποιο συγκεκριμένο



Σχήμα 3.1: Δομή ενός Quadcopter

Πηγή: https://dev.px4.io/en/airframes/airframe_reference.html

ύψος, χρειάζεται την κίνηση των στροφέων με τέτοια ταχύτητα, ώστε η βαρυτική δύναμη να είναι ίση με τη δύναμη που ωθεί το drone προς τα πάνω. Όσον αφορά την περιστροφή του drone, οι έλικες πρέπει να κινούνται με αντίθετη φορά και διαφορετικές ταχύτητες ανά δύο. Συγκεκριμένα, στο σχήμα 3.2 για μία αριστερόστροφη περιστροφή οι έλικες 2 και 4 θα περιστραφούν πιο γρήγορα από τους έλικες 1 και 3. Για την κίνηση προς τα δεξιά, αυξάνουμε την περιστροφική ταχύτητα στους έλικες που βρίσκονται στην πλευρά προς την οποία θέλουμε να κινηθούμε.



Σχήμα 3.2: Περιστροφή και ευθεία κίνηση ενός Quadcopter

Οι κινήσεις οι οποίες μπορεί να πραγματοποιήσει το Quadcopter αναλύονται ως προς τους τρεις άξονες x,y και z με τα roll, pitch και yaw αντίστοιχα. Pitch είναι η κίνηση με την οποία το drone κινείται είτε μπροστά είτε πίσω, roll είναι η κίνηση δεξιά ή αριστερά, ενώ τέλος το yaw ορίζει την κατεύθυνση του drone.

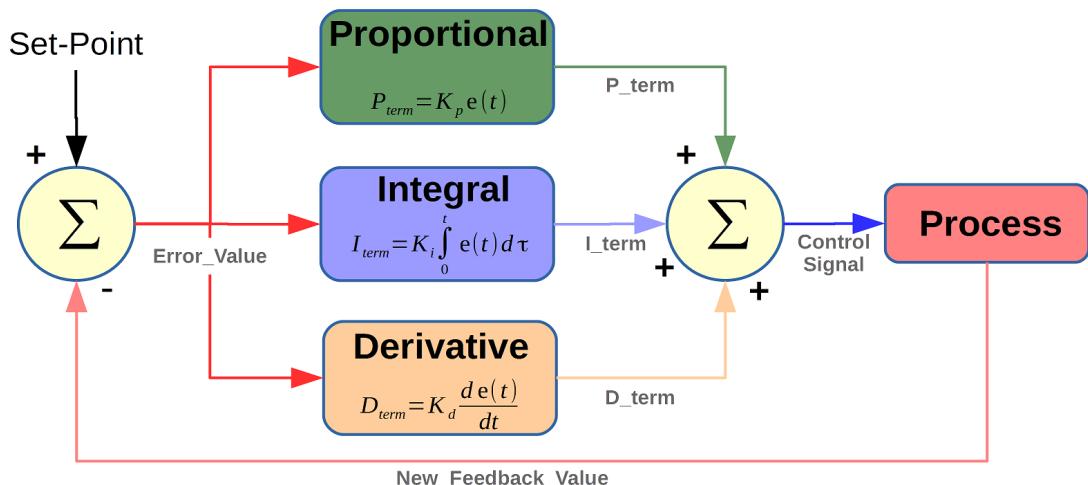
3.2. PROPORTIONAL INTEGRAL DERIVATIVE (PID) ΕΛΕΓΚΤΗΣ

3.2 PROPORTIONAL INTEGRAL DERIVATIVE (PID) ΕΛΕΓΚΤΗΣ

Ο αναλογικός-ολοκληρωτικός-παραγωγικός (PID) ελεγκτής είναι ένας μηχανισμός ανατροφοδότησης βρόχων ελέγχου που χρησιμοποιείται ευρέως σε διάφορα συστήματα ελέγχου, όπως ο έλεγχος της θερμοκρασίας, της πίεσης και της ταχύτητας. Ο ελεγκτής αυτός προσπαθεί να εξαλείψει το σφάλμα ανάμεσα σε ένα επιθυμητό σημείο λειτουργίας και στην τρέχουσα τιμή μιας μεταβλητής. Η έξοδος του είναι μια διορθωτική δράση που ρυθμίζει την διαδικασία αναλόγως.

Η έξοδος του ελεγκτή PID εξαρτάται από τους τρεις όρους που τον αποτελούν, τον αναλογικό (proportional), τον ολοκληρωτικό (integral) και τον παραγωγικό (derivative). Κάθε ένας από αυτούς επηρεάζει διαφορετικά την έξοδο και ρυθμίζονται ανεξάρτητα μεταξύ τους. Πιο συγκεκριμένα, ο αναλογικός όρος αναλαμβάνει τη διόρθωση την εξόδου, αναλογικά με την τιμή της διαφοράς ανάμεσα στην επιθυμητή και την τρέχουσα τιμή. Ο ολοκληρωτικός όρος λαμβάνει υπόψη το άθροισμα των σφαλμάτων και ο παραγωγικός την παράγωγο του σφάλματος. Η επίδραση καθενός από αυτούς τους όρους καθορίζεται από μία σταθερά, η οποία πρέπει να ρυθμίζεται ξεχωριστά για κάθε πρόβλημα που χρησιμοποιείται ο ελεγκτής, καθώς δεν υπάρχει μία τιμή που να καλύπτει όλες τις περιπτώσεις.

Στο [σχήμα 3.3](#) φαίνεται αναλυτικά ο τρόπος λειτουργίας του ελεγκτή.



Σχήμα 3.3: Ο τρόπος λειτουργίας ενός PID ελεγκτή

Πηγή: <https://se.mathworks.com/matlabcentral/mlc-downloads/downloads/submissions/58257/versions/2/screenshot.png>

Στη συγκεκριμένη εργασία, ο ελεγκτής PID χρησιμοποιείται για τον έλεγχο της θέσης του drone στο χώρο. Λαμβάνει ως είσοδο την θέση που βρίσκεται το drone και την επιθυμητή θέση και υπολογίζει την απόσταση τους. Η έξοδος του ελεγκτή υπολογίζεται με βάση τις τιμές των τριών όρων που αναλύθηκαν προηγουμένως και μετασχηματίζεται στη συνέχεια στην ταχύτητα που πρέπει να δοθεί στο drone, ώστε να φτάσει ομαλά στην επιθυμητή θέση. Χρησιμοποιούνται τέσσερις διαφορετικοί ελεγκτές, οι οποίοι υπολογίζουν ξεχωριστά το σφάλμα σε x , y , z και yaw , καθώς είναι αδύνατο να βρεθεί μια διαμόρφωση που να καλύπτει όλες τις απαι-

ΚΕΦΑΛΑΙΟ 3. ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ & ΕΡΓΑΛΕΙΑ

τήσεις του ελεγκτή. Συνεπώς, ρυθμίζουμε ξεχωριστά την γραμμική ταχύτητα στον x , y και z άξονα και την περιστροφική ταχύτητα γύρω από τον z άξονα, δηλαδή την κίνηση yaw. Για να εξασφαλίσουμε την πιο ομαλή μετάβαση του ρομπότ στην επιθυμητή θέση, εφαρμόζουμε έναν διαδοχικό έλεγχο των τεσσάρων μεταβλητών. Πιο συγκεκριμένα, πρώτα ρυθμίζεται το ύψος και η θέση (x, y) του drone και τέλος η κατεύθυνση του.

Για τον έλεγχο της θέσης, οι τρεις όροι επηρεάζουν με τους παρακάτω τρόπους την έξοδο του ελεγκτή και συνεπώς την θέση του drone:

- **Αναλογικός:** Όσο πιο μεγάλη είναι η απόσταση του drone από την επιθυμητή θέση, τόσο μεγαλύτερη είναι και η επίδραση του όρου αυτού. Αντίθετα, αν πλησιάζει στο στόχο η επίδραση μειώνεται.
- **Ολοκληρωτικός:** Εάν το σφάλμα είναι μικρό για μεγάλο χρονικό διάστημα ή υψηλό για σύντομο χρονικό διάστημα αυξάνεται η επίδραση αυτού του όρου. Αντίθετα, αν το σφάλμα έχει μία μέτρια τιμή, περιορίζεται η επίδραση του. Ο όρος αυτός είναι πολύ σημαντικός για τα UAVs σε περιβάλλοντα με υψηλό αέρα, καθώς διατηρεί σταθερή τη θέση του από τις παρεμβολές. Καθώς η εργασία αυτή πραγματοποιείται εξ' ολοκλήρου σε περιβάλλον προσομοίωσης και δεν υπάρχουν τέτοιου είδους επιδράσεις, ο όρος αυτός αγνοείται.
- **Παραγωγικός:** Όσο αυξάνεται το σφάλμα, αυξάνεται και η επίδραση του όρου, ενώ όσο μειώνεται το σφάλμα παράγει αρνητική έξοδο, ώστε να μειωθεί η περίπτωση υπερύψωσης από τον αναλογικό όρο. Σε περίπτωση που δεν μεταβάλλεται το σφάλμα, ο όρος αυτός δεν επιδρά καθόλου. Μεγάλες τιμές του όρου αυτού μπορούν να οδηγήσουν σε αστάθεια και δονήσεις του drone, λόγω των απότομων μεταβολών.

3.3 Robot OPERATING SYSTEM (ROS)

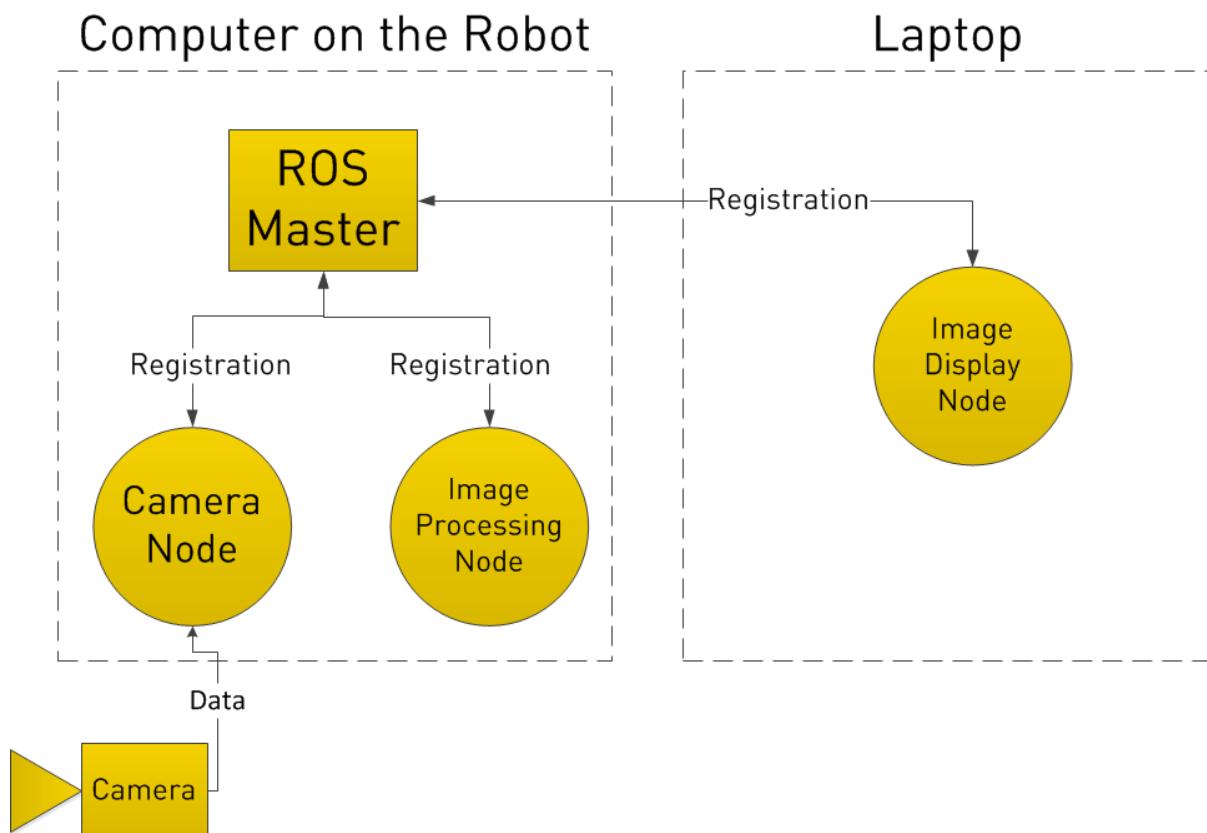
Το Robot Operating System [16] είναι το πιο διαδεδομένο μεσολειτουργικό σύστημα για ρομποτικές εφαρμογές. Αποτελείται από ένα σύνολο βιβλιοθηκών και εργαλείων τα οποία στοχεύουν στην απλοποίηση της δημιουργίας πολύπλοκων και αξιόπιστων εφαρμογών που περιλαμβάνουν τη χρήση ρομποτικών προακτόρων. Το ROS είναι ανοικτού κώδικα και έχει διαμορφωθεί με την συνεισφορά πολλών εθελοντών ανά τον κόσμο. Ο λόγος που δημιουργήθηκε είναι το γεγονός ότι η δημιουργία λογισμικού για ρομποτ είναι αρκετά δύσκολη, ιδιαίτερα όσο αυξάνεται το εύρος των ρομποτικών εφαρμογών. Διαφορετικά ρομπότ διαθέτουν και διαφορετικό υλικό (hardware) και έχουν διαφορετικές λειτουργικές απαιτήσεις ανάλογα με το σκοπό χρήσης τους. Συνεπώς, η δημιουργία ενός κοινού πλαισίου ανάπτυξης λογισμικού για ρομπότ, οδήγησε στην απλοποίηση αυτής της διαδικασίας με την ύπαρξη έτοιμων τμημάτων κώδικα που εκτελούν βασικές λειτουργίες σε κάθε εφαρμογή και την επαναχρησιμοποίησή τους.

Το ROS παρέχει όσα θα περίμενε κανείς και από ένα λειτουργικό σύστημα, δηλαδή αφαίρεση υλικού (hardware), έλεγχο συσκευών σε χαμηλό επίπεδο, ανταλλαγή μηνυμάτων μεταξύ των διεργασιών και διαχείριση πακέτων. Η δομή του βασίζεται

3.3. ROBOT OPERATING SYSTEM (ROS)

στην αρχιτεκτονική peer-to-peer, όπου η κάθε διαδικασία είναι ένας κόμβος (node)¹ στον γράφο. Οι κόμβοι μπορούν να είναι κατανεμημένοι σε διαφορετικά συστήματα, έχουν όμως τη δυνατότητα να επικοινωνούν μέσω της δομής επικοινωνίας του ROS. Υπάρχουν υλοποιημένες διάφορες μορφές επικοινωνίας, όπως η σύγχρονη επικοινωνία μέσω των services², η ασύγχρονη ροή δεδομένων σε topic³ και η αποθήκευση δεδομένων στον Parameter Server⁴. Παρόλο που το ROS δεν είναι ένα σύστημα πραγματικού χρόνου, μπορεί να χρησιμοποιηθεί με κώδικα πραγματικού χρόνου, γεγόνος που το καθιστά αξιόπιστο για λειτουργία σε ρομποτικά συστήματα που απαιτούν άμεση απόκριση.

Ενδεικτικά, η αρχιτεκτονική του ROS φαίνεται στο σχήμα 3.4. Ο κόμβος ROS Master δημιουργεί και συντηρεί τον αρχιτεκτονικό γράφο του συστήματος. Επιτρέπει σε κάθε ξεχωριστό ROS κόμβο να εντοπίσει κάποιον άλλον με τον οποίο θέλει να έρθει σε επικοινωνία.



Σχήμα 3.4: Η αρχιτεκτονική του ROS
Πηγή: <https://robohub.org/ros-101-intro-to-the-robot-operating-system/>

¹<http://wiki.ros.org/Nodes>

²<http://wiki.ros.org/Services>

³<http://wiki.ros.org/Topics>

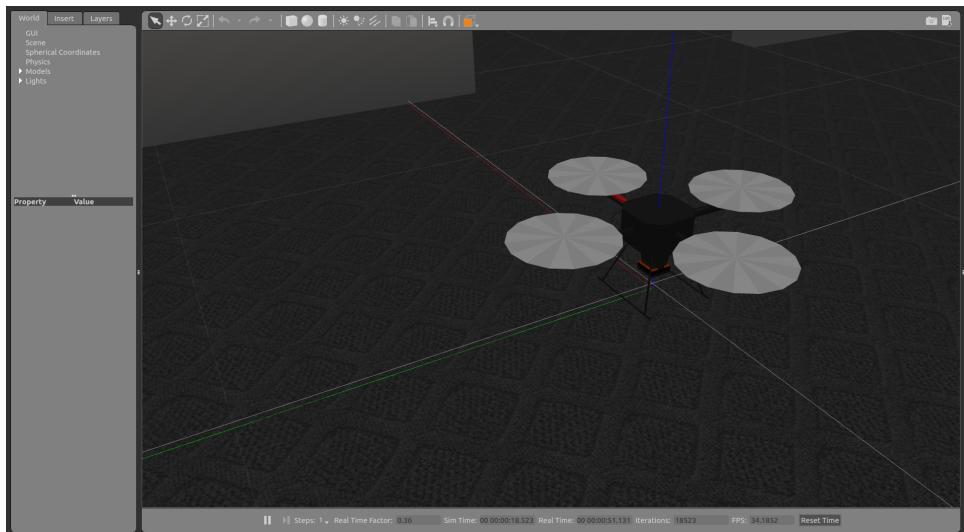
⁴<http://wiki.ros.org/Parameter%20Server>

3.4 HECTOR QUADROTOR

Το Hector Quadrotor⁵ stack [17] είναι ένα σύνολο ROS πακέτων που αφορούν την μοντελοποίηση, τον έλεγχο και την προσομοίωση σε Quadcopter. Η πρώτη του έκδοση δημοσιεύτηκε το 2012 από την ομάδα Team Hector του Πολυτεχνείου του Darmstadt.

Χρησιμοποιείται το περιβάλλον προσομοίωσης Gazebo⁶, καθώς περιλαμβάνει όλους τους φυσικούς νόμους που υπάρχουν σε ένα πραγματικό περιβάλλον και ένα μεγάλο σύνολο ρομπότ, περιβάλλοντων και αισθητήρων που μπορούν να χρησιμοποιηθούν. Για την περιγραφή των χαρακτηριστικών του drone στον προσομοιωτή, χρησιμοποιείται ένα μοντέλο URDF⁷ που περιλαμβάνει σημαντικά χαρακτηριστικά, όπως η μάζα και η αδράνεια των τμημάτων του quadcopter. Επίσης μπορούν να προστεθούν ή και να μεταβληθούν με ευκολία οι αισθητήρες του.

Το συγκεκριμένο μοντέλο διαθέτει IMU, βαρομετρικό αισθητήρα για την προσομοίωση της στατικής πίεσης σε συγκεκριμένα υψομέτρα, αισθητήρα απόστασης sonar για την εκτίμηση του ύψους, αισθητήρα μαγνητικού πεδίου για τον υπολογισμό της διεύθυνσης του drone και τέλος δέκτη GPS. Εκτός αυτών, υπάρχει η επιλογή για την χρήση κάμερας και laser, ανάλογα με τις ανάγκες του περιβάλλοντος και της εφαρμογής.



Σχήμα 3.5: Το μοντέλο του Hector Quadrotor στον προσομοιωτή Gazebo

Το πακέτο αυτό χρησιμοποιήθηκε για την μοντελοποίηση του drone στην συγκεκριμένη διπλωματική εργασία, καθώς περιλαμβάνει όλα τα απαραίτητα κομμάτια κώδικα για την ρεαλιστική προσομοίωση του συστήματος. Συνεπώς, με την χρήση και την τροποποίηση αυτών, δόθηκε έμφαση στην επίλυση του προβλήματος και στη δημιουργία μιας λύσης που θα μπορεί να χρησιμοποιηθεί με οποιοδήποτε μοντέλο drone.

⁵http://wiki.ros.org/hector_quadrotor

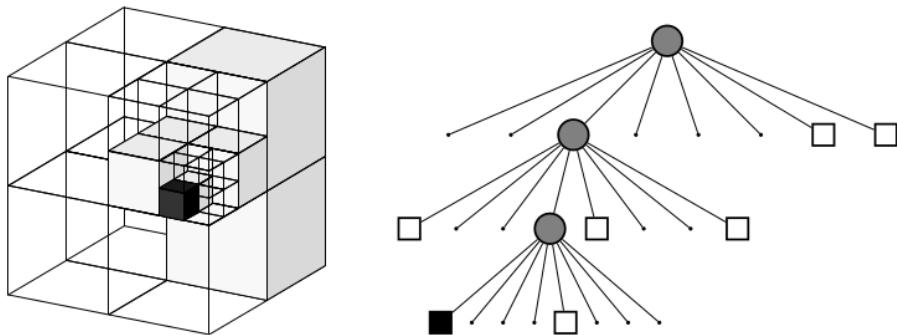
⁶<http://gazebosim.org/>

⁷<http://wiki.ros.org/urdf/XML/model>

Πιο συγκεκριμένα, μεταβλήθηκαν οι αισθητήρες του drone, ώστε να ταιριάζουν με τις απαιτήσεις του προβλήματος, ενώ χρησιμοποιήθηκαν αυτούσιοι οι αεροδυναμικοί παράμετροι και οι ελεγκτές που μεταφράζουν την ταχύτητα που δίνεται στους άξονες σε ομαλή κίνηση του αεροσκάφους.

3.5 Octomap

Η βιβλιοθήκη OctoMap [3] υλοποιεί μία αναπαράσταση του τρισδιάσταστου χώρου, παρέχοντας τις απαραίτητες δομές δεδομένων και αλγορίθμους χαρτογράφησης σε C++, με τρόπο κατάλληλο για χρήση σε ρομποτικές εφαρμογές. Η υλοποίηση του χάρτη είναι σε μορφή οκταδικού δέντρου (octree), το οποίο προσφέρει σημαντικά πλεονεκτήματα. Είναι δυνατή η απεικόνιση τόσο του ελεύθερου όσο και του άγνωστου χώρου, ενώ παράλληλα υποστηρίζεται η προσθήκη νέων πληροφοριών οποιαδήποτε στιγμή στο χάρτη. Στο [σχήμα 3.6](#) φαίνεται η αποθήκευση ελεύθερων και κατειλημμένων κελιών στο οκταδικό δέντρο. Η ανανέωση γίνεται με πιθανοτικό τρόπο, καθώς λαμβάνει υπόψη πιθανό θόρυβο των αισθητήρων και δυναμικές μεταβολές του περιβάλλοντος. Επίσης, το μέγεθος του χάρτη δεν χρειάζεται να είναι γνωστό εκ των προτέρων. Είναι δυνατή η δυναμική επέκταση του και η προβολή του με διαφορετική ανάλυση, ανάλογα με τις ανάγκες του προβλήματος. Τέλος, η αποθήκευση του σε μορφή octree, οδηγεί στην αποδοτική αποθήκευση του τόσο στη μνήμη, όσο και στο δίσκο.



Σχήμα 3.6: Ογκομετρική απεικόνιση σε octree και το αντίστοιχο οκταδικό δέντρο

3.6 ΒΙΒΛΙΟΘΗΚΗ PARTICLE FILTER

Η βιβλιοθήκη που υλοποιεί το φίλτρο σωματιδίων για τον αλγόριθμο εκτίμησης θέσης είναι η libPF⁸. Το φίλτρο σωματιδίων είναι μια συλλογή καταστάσεων (States) τις οποίες παρακολουθεί, τις αξιολογεί με βάση μία τιμή βάρους που προκύπτει από το μοντέλο αξιολόγησης και θεωρεί την κατάσταση με το μεγαλύτερο βάρος ως την πραγματική. Έπειτα, πραγματοποιείται μια δειγματοληψία των σωματιδίων, ανάλογα με την κατάσταση των βαρών και αξιολογούνται τα επόμενα σωματίδια.

⁸<https://github.com/stwirth/libPF>

ΚΕΦΑΛΑΙΟ 3. ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ & ΕΡΓΑΛΕΙΑ

Για την λειτουργία του φίλτρου απαιτείται ο ορισμός την κατάστασης που παρακολουθείται, ενός μοντέλου κίνησης και ενός μοντέλου αξιολόγησης για τις καταστάσεις. Εκτός αυτών, μπορεί να οριστεί και ένα μοντέλο κατανομής καταστάσεων, για την περίπτωση που η αρχική θέση είναι γνωστή και παρέχεται στο φίλτρο.

Όσον αφορά τη διαδικασία που ακολουθεί ένα Particle Filter, αρχικά δημιουργείται το φίλτρο και προσδιορίζονται οι παράμετροι του μη γραμμικού συστήματος. Εάν θέλουμε να λύσουμε το πρόβλημα της καθολικής εύρεσης θέσης (Global Localization) τα σωματίδια αρχικοποιούνται σε τυχαίες θέσεις μέσα στο χώρο, ενώ αν γνωρίζουμε την αρχική θέση τα σωματίδια αρχικοποιούνται με βάση μια κανονική κατανομή γύρω από αυτήν. Το αρχικό βάρος όλων των σωματιδίων είναι ίδιο. Στην συνέχεια, τα σωματίδια κινούνται σύμφωνα με το κινηματικό μοντέλο του ρομποτικού πράκτορα, όπως αυτό λαμβάνεται από την οδομετρία ή άλλους σχετικούς αισθητήρες. Επομένως, τα σωματίδια βρίσκονται σε μια καινούρια κατάσταση και πρέπει να αξιολογηθούν, ώστε να εκτιμηθεί η νέα τιμή του βάρους τους. Η αξιολόγηση γίνεται σύμφωνα με τις μετρήσεις του αισθητήρα απόστασης που βρίσκεται στο drone. Η επαναδειγματοληψία των σωματιδίων γίνεται όταν ο αριθμός των αποτελεσματικών σωματιδίων (N_{eff}) είναι μικρότερος από το μισό του συνολικού αριθμού των σωματιδίων. Η τιμή του N_{eff} δίνεται από τον παρακάτω τύπο:

$$N_{eff} = \frac{1}{\sum_{i=1}^{N_s} (w_k^i)^2}$$

Η βιβλιοθήκη libPF επιτρέπει επίσης την εκτέλεση της επαναδειγματοληψίας είτε σε κάθε βήμα εκτέλεσης του φίλτρου, μια δυνατότητα που επιβαρύνει υπολογιστικά το σύστημα, ή να μην συμβαίνει ποτέ, με αποτέλεσμα να υπάρχει η πιθανότητα κάποια σωματίδια να μηδενίσουν το βάρος τους μετά από κάποια βήματα εκτέλεσης του φίλτρου και να χαθεί η επίδρασή τους.

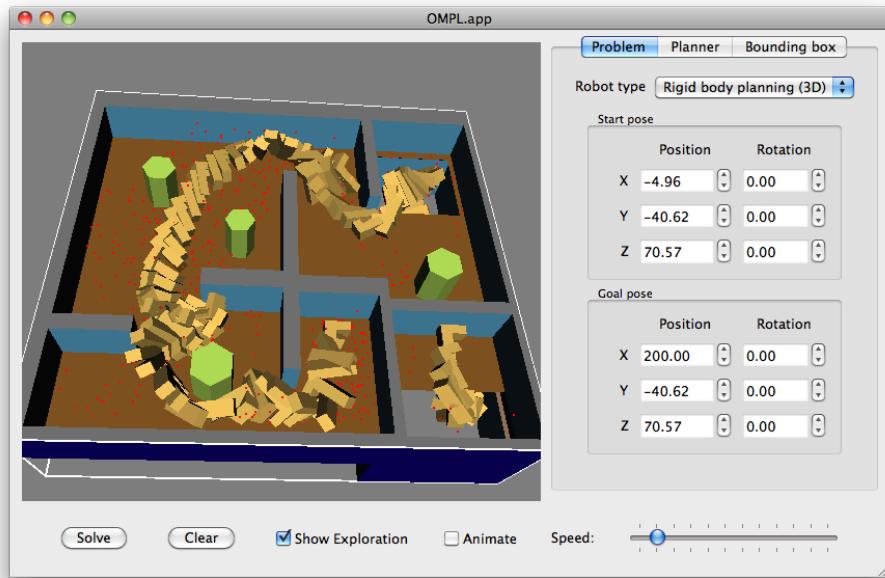
3.7 THE OPEN MOTION PLANNING LIBRARY (OMPL)

Η βιβλιοθήκη OMPL⁹ περιέχει πληθώρα αλγορίθμων που σχετίζονται με το πρόβλημα της πλοήγησης στις ρομποτικές εφαρμογές [18]. Οι αλγόριθμοι που περιέχει όπως τα Rapidly-expanding Random Trees, Probabilistic Roadmap Method και άλλοι, βασίζονται σε δειγματοληπτικές μεθόδους και περιέχουν επιλύσεις για χώρο κάθε διάστασης. Είναι ανοικτού κώδικα, γραμμένη σε C++ και πλήρως συμβατή με το ROS, ενώ έχει χρησιμοποιηθεί από την επιστημονική κοινότητα τόσο για εκπαιδευτικούς όσο και για ερευνητικούς σκοπούς.

Η βασική ιδέα στην οποία βασίζονται οι αλγόριθμοι αυτοί είναι ο διαχωρισμός του χώρου σε σημεία καταστάσεων και η ένωση των σημείων αυτών ως κορυφές σε έναν γράφο. Οι ακμές του γράφου υποδηλώνουν όλα τα εφικτά μονοπάτια μέσα στον χώρο. Η μορφή των σημείων εξαρτάται από το αντικείμενο για το οποίο εφαρμόζεται το path planning και τις δυνατές κινήσεις του, δηλαδή για ένα επίγειο ρομπότ αποτελείται από την κίνηση ως προς τους άξονες x και y , ενώ για ένα drone θα είχε επιπλέον την κίνηση στον άξονα z , δηλαδή το ύψος στο οποίο βρίσκεται. Επίσης, περιέχει έλεγχο των καταστάσεων, ώστε να αποκλειστούν θέσεις

⁹<https://ompl.kavrakilab.org/>

3.7. THE OPEN MOTION PLANNING LIBRARY (OMPL)



Σχήμα 3.7: Σχεδιασμός μονοπατιού με την OMPL

οι οποίες είναι πιθανό να επιφέρουν σύγκρουση ή βρίσκονται εκτός των ορίων του περιβάλλοντος.

Η συγκεκριμένη βιβλιοθήκη προτιμήθηκε καθώς μπορεί να δεχτεί ως είσοδο έναν χάρτη σε μορφή OctoMap και να πραγματοποιήσει σχεδιασμό διαδρομής μέσα σε αυτόν. Επίσης, μετά τον σχεδιασμό των σημείων που απαιτούνται για την επίτευξη του στόχου, πραγματοποιείται μια εξομάλυνση του μονοπατιού, ανάλογα με την μετρική που δείχνει το πόσο ομαλό είναι το μονοπάτι. Με τον τρόπο αυτό, εξασφαλίζουμε την πλοιήγηση με αποφυγή των γνωστών στο χώρο εμποδίων. Στο **σχήμα 3.7** βλέπουμε ενδεικτικά την χρήση της βιβλιοθήκης για την δημιουργία ενός μονοπατιού στο χώρο.

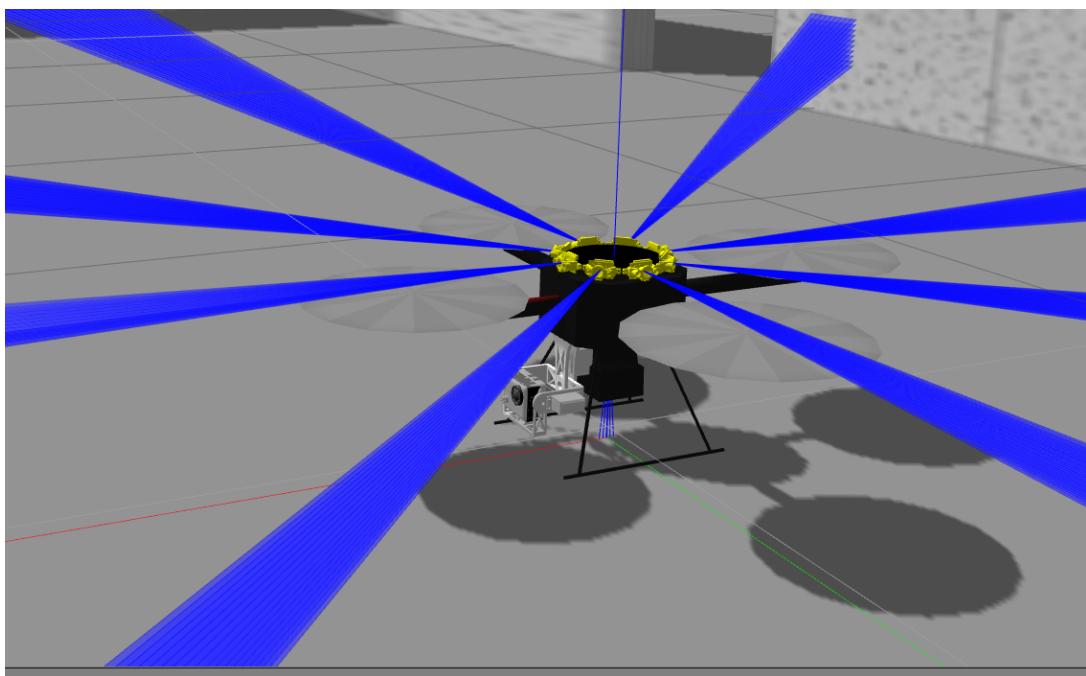
Ο αλγόριθμος που χρησιμοποιήθηκε για την δημιουργία των μονοπατιών είναι ο RRT* (Optimal RRT) [19]. Πρόκειται για μια βελτιωμένη εκδοχή του αλγορίθμου RRT [20], ο οποίος εγγυάται την σύγκλιση σε βέλτιστη λύση, ενώ ο χρόνος εκτέλεσης του είναι σταθερός παράγοντας του χρόνου εκτέλεσης του RRT.

4

Υλοποιήσεις

Ο στόχος της παρούσας διπλωματικής εργασίας είναι η δημιουργία ενός συστήματος που θα είναι ικανό να εντοπίζει την θέση ενός drone σε γνωστό τρισδιάστατο χάρτη και να δημιουργεί κατάλληλα και ασφαλή μονοπάτια ώστε να καλυφθεί κάθε σημείο αυτού. Επομένως, είναι απαραίτητο να έχουμε εκ των προτέρων διαθέσιμο τον εκάστοτε τρισδιάστατο χάρτη του περιβάλλοντος. Προτιμήθηκε η μορφή OctoMap, καθώς περιέχει σημαντικές δυνατότητες που διευκολύνουν την διαδικασία του localization.

Σε όλες τις υλοποιήσεις χρησιμοποιήθηκε το drone που φαίνεται στο [σχήμα 4.1](#).



Σχήμα 4.1: Το μοντέλο του drone που χρησιμοποιήθηκε

To Quadcopter αυτό αποτελείται από τα παρακάτω τμήματα:

- Έναν αισθητήρα TeraRanger Tower¹⁰ που προσφέρει 8 ακτίνες laser περιμετρικά με εμβέλεια 14 μέτρων
- Έναν αισθητήρα laser για τον υπολογισμό του ύψους στο οποίο βρίσκεται το drone
- Έναν αισθητήρα IMU

Στο πρώτο μέρος του συγκεκριμένου κεφαλαίου θα περιγραφεί ο αλγόριθμος Monte Carlo Localization, ο οποίος υλοποιήθηκε για τον εντοπισμό της θέσης του drone σε τρισδιάστατο περιβάλλον μορφής OctoMap. Στο δεύτερο μέρος θα αναλυθεί ο τρόπος υπολογισμού των σημείων που οδηγούν στην πλήρη κάλυψη του χώρου και η ένωση αυτών με ευριστικούς αλγορίθμους.

4.1 ΕΝΤΟΠΙΣΜΟΣ ΘΕΣΗΣ

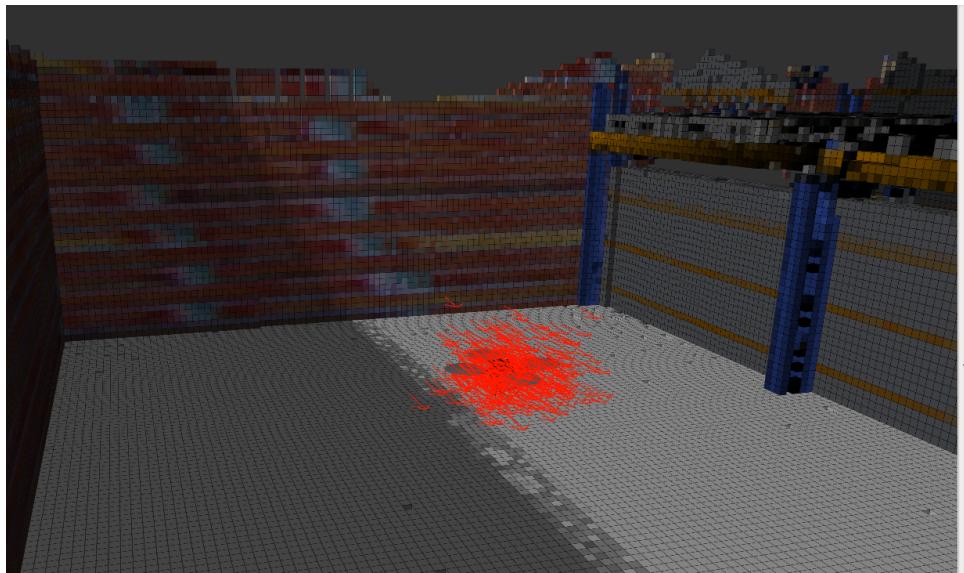
Όπως αναφέρθηκε στο [υποκεφάλαιο 3.6](#), η κύρια μέθοδος που χρησιμοποιήθηκε για τον εντοπισμό της θέσης είναι ο αλγόριθμος Φίλτρου Σωματιδίων (Particle Filter, PF).

Η βασική ιδέα του φίλτρου είναι η αναπαράσταση της εκ των υστέρων πεποίθησης με ένα σύνολο τυχαίων δειγμάτων κατανομής που έχουν ληφθεί από την συγκεκριμένη εκ των υστέρων κατανομή. Κάθε σωματίδιο είναι ένα συνεπτυγμένο στιγμιότυπο της κατάστασης σε κάποια χρονική στιγμή. Ο αλγόριθμος αυτός απαιτεί ως είσοδο το προηγούμενο σύνολο σωματιδίων, ένα μοντέλο μέτρησης και ένα μοντέλο κίνησης. Η κύρια πηγή πληροφορίας για το πρώτο είναι οι αποστάσεις που προκύπτουν από τον αισθητήρα laser, ενώ για το δεύτερο είναι ο υπολογισμός της κίνησης του drone.

Το σύστημα δίνει τη δυνατότητα για την επίλυση του προβλήματος τοπικού και καθολικού εντοπισμού θέσης. Στην πρώτη περίπτωση, τα σωματίδια του φίλτρου αρχικοποιούνται γύρω από την αρχική θέση χρησιμοποιώντας μία κανονική κατανομή με δεδομένη τιμή της τυπικής απόκλισης, η οποία παρέχεται ως παράμετρος. Στην περίπτωση του Global Localization, τα σωματίδια αρχικοποιούνται με μία κανονική κατανομή σε όλο τον χάρτη του περιβάλλοντος. Η αρχική θέση παρέχεται επίσης ως παράμετρος του συστήματος. Η αρχικοποίηση των σωματιδίων φαίνεται στο [σχήμα 4.2](#).

Όσον αφορά το μοντέλο κίνησης του ρομπότ, σε αντίθεση με τα επίγεια οχήματα, τα οποία διαθέτουν αυτήν την πληροφορία μέσω αισθητήρων κίνησης (π.χ. κωδικοποιητές στους τροχούς), σε ένα drone αυτό δεν είναι εφικτό. Μία λύση θα ήταν η χρήση οπτικής οδομετρίας, μέσω της οποίας που διαθέτει. Καθώς όμως η ανάπτυξη του συστήματος πραγματοποιείται σε περιβάλλον προσομοίωσης με ιδανικές επιφάνειες, δεν θα είχαμε τα επιθυμητά αποτελέσματα. Για το λόγο αυτό, συνδυάζουμε την πληροφορία που προκύπτει από το IMU, τον αισθητήρα ύψους

¹⁰<https://www.terabee.com/shop/lidar-tof-multi-directional-arrays/teraranger-tower/>



Σχήμα 4.2: Αρχικοποίηση σωματιδίων

και την ταχύτητα που του παρέχεται, ώστε να προκύψει μια εκτίμηση της κίνησης και της περιστροφής του drone.

Για τον συγχρονισμό των μηνυμάτων χρησιμοποιείται το ROS πακέτο Message Filters¹¹ και πιο συγκεκριμένα το ApproximateTime Policy. Η μέθοδος αυτή χρησιμοποιεί έναν αλγόριθμο που προσαρμόζεται ανάλογα με τις χρονικές σημάνσεις των μηνυμάτων που δέχεται. Για κάθε συνδυασμό μηνυμάτων που λαμβάνονται γίνονται οι υπολογισμοί που φαίνονται στον αλγόριθμο 4.1.

Αλγόριθμος 4.1 Αλγόριθμος υπολογισμού κίνησης και προσανατολισμού

```

1: function CALCULATEODOMETRY(height, imu, velocity)
2:   dt  $\leftarrow$  time.now – time.previous
3:   yaw  $\leftarrow$  imu.yaw
4:   msg  $\leftarrow$  NULL      # Contains position and orientation that will be published
5:   msg.position.x = position.x + ( $\cos(yaw)$  · velocity.x –  $\sin(yaw)$  · velocity.y) * dt
6:   msg.position.y = position.y + ( $\sin(yaw)$  · velocity.x +  $\cos(yaw)$  · velocity.y) * dt
7:   msg.position.z = height
8:   msg.orientation = imu.orientation
9:   publish(msg)
10:  end function
```

Αρχικά, υπολογίζεται το χρονικό διάστημα *dt* που μεσολάβησε ανάμεσα στις διαδοχικές κλήσεις της συνάρτησης. Από το μήνυμα του IMU υπολογίζεται η τιμή του *yaw* στο drone. Δεδομένης αυτής της τιμής, περιστρέφουμε το διάνυσμα της ταχύτητας *velocity*, ώστε να μετασχηματιστεί από το πλαίσιο του ρομπότ στο πλαίσιο

¹¹http://wiki.ros.org/message_filters

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

του κόσμου που βρίσκεται, όπως φαίνεται παρακάτω:

$$\text{linear_velocity}_{\text{transformed}} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \text{linear_velocity}_x \\ \text{linear_velocity}_y \\ \text{linear_velocity}_z \end{bmatrix}$$

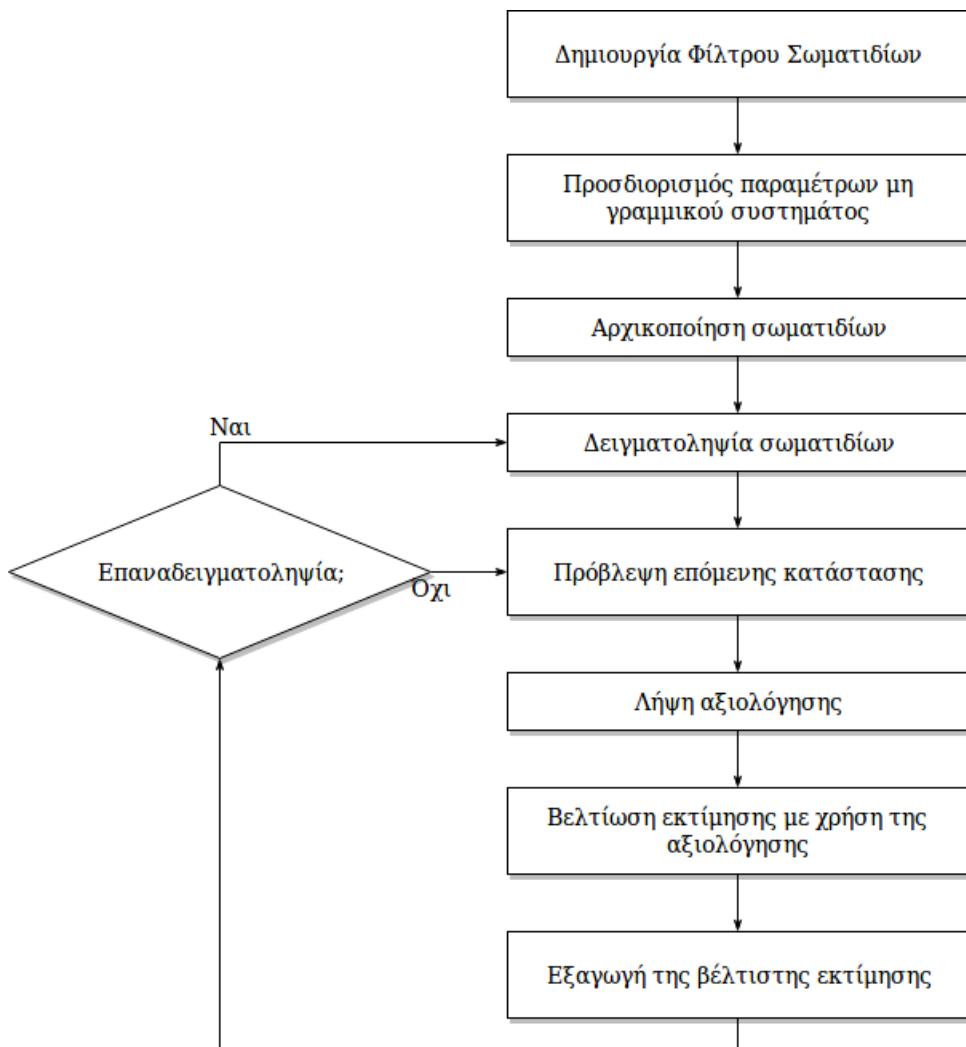
Στη συνέχεια, η ποσότητα αυτή ολοκληρώνεται στο χρόνο που μεσολάβησε και προστίθεται στην προηγούμενη τιμή της θέσης. Το ύψος λαμβάνεται απευθείας από την τιμή *height* του αισθητήρα ύψους. Τέλος, ο προσανατολισμός του drone δίνεται άμεσα από τον αισθητήρα IMU. Η σύνθεση αυτών αποστέλλεται στο κατάλληλο ROS topic, */odom*, και η διαδικασία αυτή επαναλαμβάνεται συνεχώς.

Όσον αφορά το μοντέλο μέτρησης, αυτό χρησιμοποιεί τις μετρήσεις που προέρχονται από το laser, ώστε να εκτιμήσει το νέο βάρος κάθε σωματιδίου, ανάλογα με την θέση του. Αρχικά, οι μετρήσεις του αισθητήρα laser έρχονται σε 8 διαφορετικά ROS topic, τύπου *sensor_msgs/Range*. Με την χρήση μιας τροποποιημένης έκδοσης του πακέτου *teraranger_array_converter*¹² τα μηνύματα αυτά συγχωνεύονται σε ένα topic και στη συνέχεια μετατρέπονται σε μορφή *sensor_msgs/LaserScan*. Η μορφή αυτή αποτελεί τον πιο συνηθισμένο τύπο για χειρισμό δεδομένων από αισθητήρα laser και περιέχει χρήσιμες πληροφορίες σχετικά με την ελάχιστη γωνία κάλυψης, τη γωνία μεταξύ διαδοχικών μετρήσεων και το εύρος του laser. Συνεπώς, ανακατασκευάζουμε τα σημεία στον τρισδιάστατο χώρο και μετατρέπουμε το μήνυμα σε μορφή *PointCloud*, της βιβλιοθήκης PCL¹³. Το βήμα αυτό είναι χρήσιμο για την περίπτωση όπου το laser περιέχει μεγάλο αριθμό ακτινών και η επεξεργασία τους θα οδηγήσει σε αργοπορημένη απόκριση, δημιουργώντας την ανάγκη για υποδειγματοληψία. Στην συγκεκριμένη περίπτωση, δεν πραγματοποιείται καμία επεξεργασία, παρά μόνο παρέχεται μία λύση που καλύπτει κάθε αισθητήρα απόστασης, ανεξαρτήτως χαρακτηριστικών. Το τελευταίο βήμα της προ-επεξεργασίας είναι ο μετασχηματισμός των δεδομένων ως προς το κέντρο του κάθε σωματιδίου. Χρησιμοποιείται ο μετασχηματισμός μεταξύ του ρομπότ και του αισθητήρα, ο οποίος παρέχεται μέσω της κλάσης *tf2_ros::Buffer* της βιβλιοθήκης TF του ROS. Αυτός συνδυάζεται με την θέση του κάθε σωματιδίου και προκύπτει ο τελικός μετασχηματισμός, ώστε τα δεδομένα να βρίσκονται στην σωστή θέση.

Εφόσον η προ-επεξεργασία έχει ολοκληρωθεί το φίλτρο ξεκινάει τη λειτουργία του, η οποία σχηματικά περιγράφεται στο [σχήμα 4.3](#). Ο αριθμός των σωματιδίων ορίζεται ως παράμετρος στο σύστημα και είναι μια τιμή που επιτρέπει την ισορροπία ανάμεσα στην ακρίβεια των υπολογισμών και των υπολογιστικών πόρων που είναι διαθέσιμοι. Αρχικά, εφαρμόζεται δειγματοληψία των σωματιδίων ανάλογα με το βάρος τους. Στη συνέχεια, μετατοπίζουμε την θέση του κάθε σωματιδίου, σύμφωνα με το κινηματικό μοντέλο που έχει δημιουργηθεί. Μέσω της βιβλιοθήκης TF το σύστημα αναζητά την τρέχουσα θέση του ρομπότ στον κόσμο, την οποία μετασχηματίζει σε ένα διαφορετικό σύστημα συντεταγμένων, με χρήση της μεθόδου *transform(·)* της κλάσης *tf2_ros::Buffer*. Δίνοντας ως δόρισμα το μοναδιαίο διάνυσμα του ρομπότ, επιστρέφει τη νέα θέση στο σύστημα συντεταγμένων που μεταβάλλεται με την μεταβολή της οδομετρίας. Έχοντας πλέον τις δύο αυτές θέσεις, μπορούμε να

¹²https://github.com/kosmastsks/teraranger_array_converter

¹³<http://pointclouds.org/>



Σχήμα 4.3: Διαδικασία ενός φίλτρου σωματιδίων

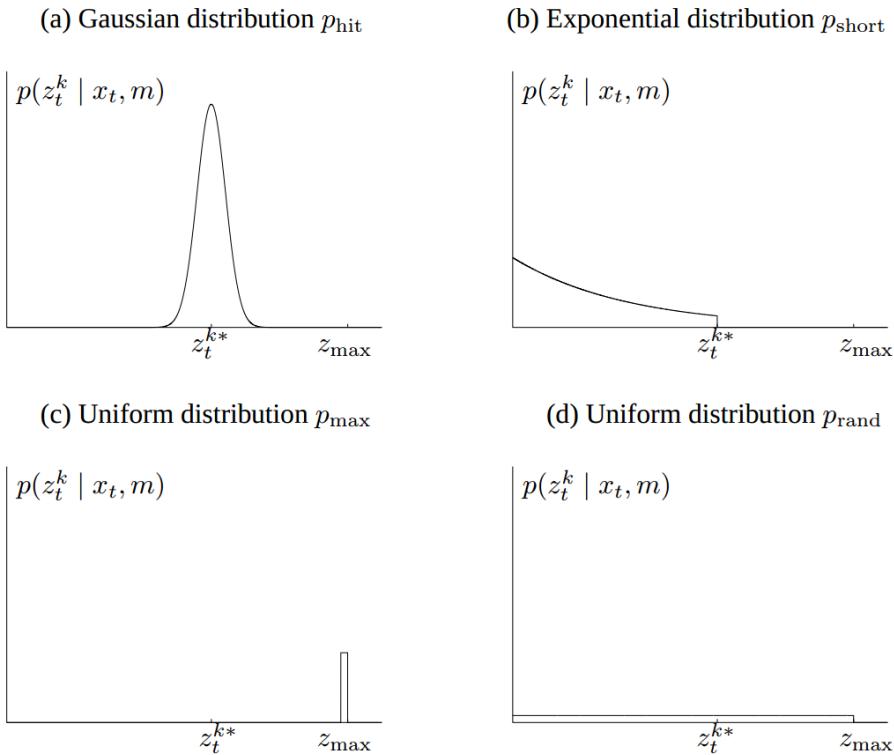
υπολογίζουμε την διαφορά τους και συνεπώς την κίνηση που έχει κάνει το ρομπότ. Ο μετασχηματισμός αυτός εφαρμόζεται σε όλα τα σωματίδια, ώστε να μετατοπιστούν στις νέες τους θέσεις.

Τελικό στάδιο του φίλτρου είναι η αξιολόγηση των μετρήσεων που έχουν ληφθεί από τον αισθητήρα απόστασης με βάση το μοντέλο μέτρησης. Το κάθε σωματίδιο αξιολογείται αναθέτοντας σε αυτό, ανάλογα με το μοντέλο που έχει καθοριστεί ένα βάρος. Το βάρος αυτό προσεγγίζει την πιθανότητα το συγκεκριμένο σωματίδιο να είναι η ορθή κατάσταση, δηλαδή η μέτρηση που έχει ληφθεί να αντιστοιχεί σε αυτήν που θα ανήγεινε αν όντως βρισκόταν σε αυτήν την κατάσταση. Εκτός από μία εκτίμηση θέσης από κάθε σωματίδιο, το σύστημα εξάγει και μια συνολική εκτίμηση για την θέση. Αυτή προκύπτει από ένα ποσοστό των καλύτερων σωματιδίων, το οποίο μπορεί να επιλεγεί από τον χρήστη. Δεν υπάρχει κάποια συγκεκριμένη τιμή του ποσοστού που να εγγυάται καλύτερα αποτελέσματα, καθώς αυτό μπορεί να διαφέρει ανάλογα με τις ιδιαιτερότητες κάθε περίπτωσης.

Ένα βασικό πρόβλημα των αισθητήρων είναι η ύπαρξη θορύβου, κάτι που συχνά δεν λαμβάνεται υπόψη στους υπολογισμούς. Στην συγκεκριμένη περίπτωση

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

όμως, η έλλειψη ακρίβειας που παρουσιάζουν τα μοντέλα των αισθητήρων ενσωματώνεται στην μοντελοποίηση της διαδικασίας μέτρησης ως πυκνότητας μιας υπό συνθήκη πιθανότητας, αντί μιας αιτιοκρατικής συνάρτησης. Το μοντέλο μέτρησης που χρησιμοποιήθηκε περιγράφεται στο [21] και περιλαμβάνει τέσσερις τύπους σφαλμάτων μέτρησης, που είναι όλοι βασικοί για την σωστή λειτουργία του μοντέλου: το λιγοστό θόρυβο μέτρησης, τα σφάλματα που οφείλονται σε μη αναμενόμενα αντικείμενα, τα σφάλματα που οφείλονται σε αποτυχίες εντοπισμού αντικειμένων και τον τυχαίο ανεξήγητο θόρυβο. Η πραγματική απόσταση z_t^k που βρίσκεται η κατάσταση μπορεί να υπολογιστεί εύκολα, μέσω των συναρτήσεων της βιβλιοθήκης OctoMap που παρέχουν αυτήν την δυνατότητα του Raytracing. Σε κάθε διάγραμμα στο [σχήμα 4.4](#) ο οριζόντιος άξονας αντιστοιχεί στη μέτρηση z_t^k και ο κατακόρυφος στην πιθανότητα.



Σχήμα 4.4: Τα στοιχεία του μοντέλου για έναν αισθητήρα μέτρησης απόστασης

Επομένως, το ζητούμενο μοντέλο $p(z_t^k | x_t, m)$ είναι ένα μείγμα των παρακάτω τεσσάρων πυκνοτήτων, καθεμία από τις οποίες αντιστοιχεί σε συγκεκριμένο τύπο σφάλματος.

Σωστή απόσταση με τοπικό θόρυβο μέτρησης: Ακόμα και αν ο αισθητήρας μετράει σωστά την απόσταση από το πλησιέστερο αντικείμενο, η τιμή που επιστρέφει μπορεί να επηρεαστεί από κάποιο σφάλμα. Το τελευταίο οφείλεται στην περιορισμένη ανάλυση των αισθητήρων μέτρησης απόστασης, στις ατμοσφαιρικές επιδράσεις στο σήμα της μέτρησης κ.ο.κ. Αυτός ο θόρυβος μέτρησης μοντελοποιείται συνήθως με μια μικρού πλάτους κατανομή Gauss, όπως φαίνεται στο [σχήμα 4.4\(a\)](#).

Επομένως, η πιθανότητα μέτρησης δίνεται από τον παρακάτω τύπο:

$$p_{hit}(z_t^k | x_t, m) = \begin{cases} \eta N(z_t^k, z_t^{k*}, \sigma_{hit}^2) & \text{if } 0 < z_t^k < z_{max} \\ 0 & \text{else} \end{cases}$$

$$N(z_t^k, z_t^{k*}, \sigma_{hit}^2) = \frac{1}{\sqrt{2\pi\sigma_{hit}^2}} e^{-\frac{1}{2}\frac{(z_t^k - z_t^{k*})^2}{\sigma_{hit}^2}}$$

Μη αναμενόμενα αντικείμενα: Τα περιβάλλοντα των κινητών ρομπότ είναι δυναμικά, ενώ οι χάρτες στατικοί. Αυτό έχει ως αποτέλεσμα τα αντικείμενα που δεν περιέχονται στο χάρτη να αποτελούν την αιτία για την οποία οι αισθητήρες μέτρησης απόστασης παράγουν αναπάντεχα μικρές αποστάσεις. Μια απλή μέθοδος για να αντιμετωπίσουμε τέτοιες καταστάσεις είναι να τις θεωρήσουμε ως θόρυβο. Σε τέτοιες περιπτώσεις η πιθανότητα των μετρήσεων απόστασης περιγράφεται από μια εκθετική κατανομή, όπως φαίνεται στο [σχήμα 4.4\(b\)](#). Η παράμετρος λ_{short} είναι μια εγγενής παράμετρος του μοντέλου μέτρησης. Η πιθανότητα $p_{short}(z_t^k | x_t, m)$ δίνεται από τον παρακάτω τύπο:

$$p_{short}(z_t^k | x_t, m) = \begin{cases} \eta \lambda_{short} e^{-\lambda_{short} z_t^k} & \text{if } 0 < z_t^k < z_{max} \\ 0 & \text{else} \end{cases}$$

Αποτυχίες: Μερικές φορές τα εμπόδια δεν εντοπίζονται. Αυτό μπορεί να συμβεί σε αισθητήρες σόναρ εξαιτίας των κατευθυνόμενων ανακλάσεων ή στους αισθητήρες λέιζερ όταν ανιχνεύονται μαύρα αντικείμενα που απορροφούν το φως. Ένα ακόμη τυπικό παράδειγμα αποτυχίας είναι η μέτρηση μέγιστης απόστασης, ο αισθητήρας επιστρέφει την μέγιστη επιτρεπόμενη τιμή z_{max} . Η περίπτωση αυτή μοντελοποιείται με μια κατανομή σημειακής μάζας που είναι κεντραρισμένη στην τιμή z_{max} , όπως φαίνεται στο [σχήμα 4.4\(c\)](#).

$$p_{max}(z_t^k | x_t, m) = \begin{cases} 1 & \text{if } z = z_{max} \\ 0 & \text{else} \end{cases}$$

Τυχαίες μετρήσεις: Τέλος, οι αισθητήρες απόστασης παράγουν περιστασιακά εντελώς ανεξήγητες μετρήσεις. Για λόγους απλότητας μοντελοποιούμε αυτές τις μετρήσεις χρησιμοποιώντας μια ομοιόμορφη κατανομή που καλύπτει ολόκληρο το εύρος των μετρήσεων, όπως φαίνεται στο [σχήμα 4.4\(d\)](#).

$$p_{rand}(z_t^k | x_t, m) = \begin{cases} \frac{1}{z_{max}} & \text{if } 0 < z_t^k < z_{max} \\ 0 & \text{else} \end{cases}$$

Οι συγκεκριμένες τέσσερις διαφορετικές κατανομές συνδυάζονται με τη χρήση ενός σταθμισμένου μέσου όρου, που ορίζεται από τις παραμέτρους z_{hit} , z_{short} , z_{max} και z_{rand} με το άθροισμα αυτών να ισούται με 1. Αυτό είναι εφικτό, καθώς θεωρούμε ότι οι μεμονωμένες μετρήσεις είναι υπό συνθήκη ανεξάρτητες.

Ο αριθμός των σωματιδίων παραμένει σταθερός σε όλη τη διάρκεια της εκτέλεσης του φίλτρου, με αποτέλεσμα μετά από κάποια βήματα το βάρος ορισμένων

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

Αλγόριθμος 4.2 Αλγόριθμος υπολογισμού πιθανότητας μιας σάρωσης απόστασης

```

1:  $q \leftarrow 1$ 
2: for  $k=1$  to  $K$  do
3:   Ύπολογισμός της  $z_t^{k*}$  για την μέτρηση  $z_t^k$  με ρίψη ακτίνων
4:    $p \leftarrow z_{hit} \cdot p_{hit}(z_t^k | x_t, m) + z_{short} \cdot p_{short}(z_t^k | x_t, m) + z_{max} \cdot p_{max}(z_t^k | x_t, m) + z_{rand} \cdot p_{rand}(z_t^k | x_t, m)$ 
5:    $q \leftarrow q \cdot p$ 
6: end for
7: return  $q$ 

```

σωματιδίων να γίνεται πολύ μικρό και τα σωματίδια αυτά να μην έχουν καμία επίδραση στο φίλτρο. Για το λόγο αυτό, εφαρμόζεται μια διαδικασία, γνωστή ως **Αναδειγματοληφία**, κάθε φορά που η διακύμανση των βαρών των σωματιδίων είναι μικρότερη από το μισό του αριθμού των συνολικών σωματιδίων. Η τιμή αυτή λέγεται N_{eff} και περιγράφει το κατά πόσο όμοια είναι κατανεμημένα τα βάρη. Το βήμα της αναδειγματοληφίας είναι μια πιθανοτική υλοποίηση της ιδέας του Δαρβίνου για την επικράτηση του ισχυρότερου: επανασυγκεντρώνει το σύνολο των σωματιδίων σε περιοχές του χώρου καταστάσεων με μεγάλη εκ των υστέρων πιθανότητα. Με αυτόν τον τρόπο, εστιάζει τους υπολογιστικούς πόρους που χρησιμοποιούνται από τον αλγόριθμο του φίλτρου στις περιοχές του χώρου καταστάσεων που έχουν τη μεγαλύτερη σημασία.

Αλγόριθμος 4.3 Αναδειγματοληφία χαμηλής διακύμανσης

```

1:  $new\_particles \leftarrow 0$ 
2:  $r \leftarrow rand(0; M^{-1})$ 
3:  $c \leftarrow w_t^{[1]}$ 
4:  $i \leftarrow 1$ 
5: for  $m=1$  to  $M$  do
6:    $U \leftarrow r + (m - 1) \cdot M^{-1}$ 
7:   while  $U > c$  do
8:      $i \leftarrow i + 1$ 
9:      $c = c + w_t^{[1]}$ 
10:  end while
11:   $new\_particles \leftarrow new\_particles + w_t^{[i]}$ 
12: end for
13: return  $new\_particles$ 

```

Η στρατηγική που χρησιμοποιείται για την αναδειγματοληφία είναι αυτή της χαμηλής διακύμανσης και φαίνεται στον [αλγόριθμο 4.3](#). Η μέθοδος αυτή, αντί να επιλέγει δείγματα ανεξάρτητα το ένα από το άλλο στη διαδικασία της αναδειγματοληφίας ακολουθεί μια στοχαστική διαδικασία. Ύπολογίζεται ένας τυχαίος αριθμός και τα δείγματα επιλέγονται με βάση αυτόν, με πιθανότητα όμως που παραμένει ανάλογη του συντελεστή στάθμισης του δείγματος. Αυτό επιτυγχάνεται επιλέγοντας τον αριθμό r στο διάστημα $[0, M^{-1}]$, όπου M ο αριθμός των σωματιδίων που πρόκειται να ληφθούν. Έπειτα, επιλέγονται σωματίδια προσθέτοντας συνεχώς την

σταθερή ποσότητα M^{-1} στον αριθμό r και επιλέγοντας το σωματίδιο που αντιστοιχεί στον αριθμό που προκύπτει. Με τον τρόπο αυτό, καλύπτεται ο χώρος με πιο συστηματικό τρόπο, ενώ επιτυγχάνεται δειγματοληψία με πολυπλοκότητα $\mathcal{O}(M)$, κάτι που είναι πολύ σημαντικό για ρομποτικές εφαρμογές που απαιτούν υψηλή απόδοση σε περιορισμένο υλικό.

4.2 ΠΛΗΡΗΣ ΚΑΛΥΨΗ ΧΩΡΟΥ

Το πρόβλημα της κάλυψης χώρου έχει δύο βασικές εκδοχές: με χάρτη και χωρίς χάρτη. Στην εργασία αυτή, ο χάρτης είναι διαθέσιμος σε μορφή OctoMap και σκοπός είναι η πλοήγηση του ρομπότ μέχρι να κάλυψε όλος ο χάρτης. ή τουλάχιστον όλα τα σημεία ενδιαφέροντος σε αυτόν. Υπάρχουν διάφοροι τρόποι για την επίλυση αυτού του προβλήματος, είτε οι αναλυτικοί, αλλά υπολογιστικά αργοί αλγόριθμοι, είτε οι ευριστικοί που με την χρήση ορισμένων πρακτικών κανόνων επιταχύνουν την αναζήτηση.

Βασική ιδέα είναι όλα τα αντικείμενα του χώρου να βρεθούν κάποια στιγμή στο οπτικό πεδίο ενός αισθητήρα που βρίσκεται πάνω στο ρομπότ. Στη συγκεκριμένη περίπτωση, πρόκειται για έναν αναγνώστη RFID με τα παρακάτω παραμετροποίησιμα χαρακτηριστικά:

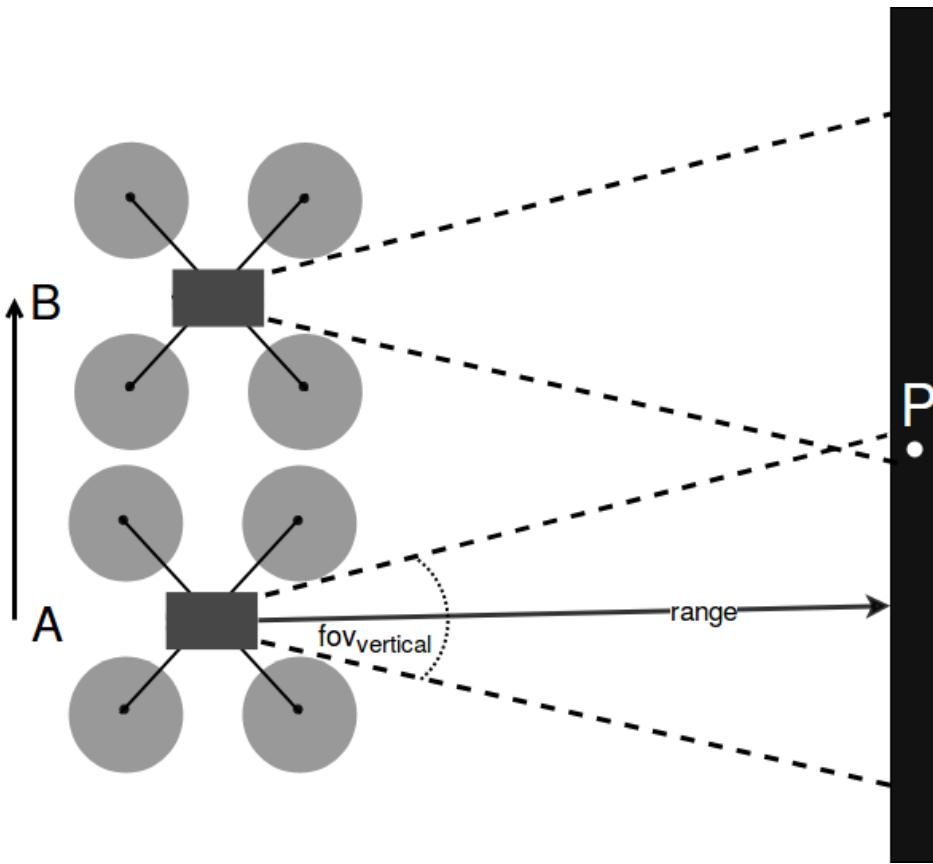
- Εμβέλεια
- Οριζόντιο και κάθετο πεδίο όρασης
- Τη μορφή του πεδίου κάλυψης (օρθογώνιο/κυκλικό)
- Το διάνυσμα που δίνει την κατεύθυνση του αναγνώστη

Για να υπολογίσουμε τα σημεία στα οποία πρέπει να βρεθεί το ρομπότ, ώστε να πετύχει την πλήρη κάλυψη του χώρου, υποδειγματοληπτούμε τον χώρο με κάποιο σταθερό βήμα ως προς x και y και διαφορετικό βήμα ως προς z , φροντίζοντας πάντα να διατηρούνται οι αποστάσεις ασφαλείας του drone από οποιοδήποτε στατικό εμπόδιο στο περιβάλλον. Το βήμα δειγματοληψίας ως προς τον z άξονα υπολογίζεται με βάση τη γωνία του κάθετου οπτικού πεδίου του αισθητήρα, με σκοπό κάθε σημείο του χώρου να βρεθεί εντός του πεδίου οράσεως τουλάχιστον δύο φορές. Συνεπώς, χρησιμοποιώντας τριγωνομετρία, υπολογίζεται με τον παρακάτω τύπο:

$$z_{step} = range \cdot \tan(fov_{vertical}/2) \quad (4.1)$$

Πιο συγκεκριμένα, στο [σχήμα 4.5](#) το σημείο P βρίσκεται εντός του οπτικού πεδίου του drone, όταν αυτό είναι στο ύψος A και όταν είναι στο ύψος $B = A + range \cdot \tan(fov_{vertical}/2)$.

Στη συνέχεια, για κάθε ένα σημείο που προκύπτει από την δειγματοληψία ελέγχουμε εάν η θέση αυτή είναι ασφαλής για το ρομπότ. Οι έλεγχοι που πραγματοποιούνται είναι: αν η θέση αυτή ανήκει σε κατειλημμένο κόμβο του χάρτη μορφής OctoMap και εάν υπάρχουν εμπόδια σε απόσταση μικρότερη από την επιτρεπόμενη απόσταση ασφαλείας που έχει οριστεί. Με χρήση του προβαλλόμενου χάρτη



Σχήμα 4.5: Μετάβαση του drone από το ύφος Α στο ύφος Β

σε δύο διαστάσεις, μπορούμε επιπλέον να ελέγξουμε αν το κελί της θέσης που μας ενδιαιφέρει βρίσκεται σε χώρο που υπάρχει εμπόδιο και δεν έχει αναγνωριστεί με κανέναν από τους προηγούμενους τρόπους.

Εφόσον αυτοί οι έλεγχοι καταλήξουν στο συμπέρασμα ότι η θέση είναι ασφαλής για το drone, υπολογίζουμε τον προσανατολισμό ο οποίος προσδίδει καλύτερη ορατότητα του αντικειμένου. Για το λόγο αυτό, ελέγχουμε διαδοχικά διαφορετικές γωνίες και υπολογίζουμε το ποσοστό κάλυψης από την καθεμία. Ο αλγόριθμος που περιγράφει τον τρόπο υπολογισμού του ποσοστού κάλυψης για μία συγκεκριμένη κατεύθυνση φαίνεται στον [αλγόριθμο 4.4](#). Αρχικά, με χρήση της συνάρτησης `castRay()` της βιβλιοθήκης OctoMap, υπολογίζουμε την θέση του σημείου που βρίσκεται στο κέντρο του οπτικού πεδίου του αισθητήρα. Στη συνέχεια, η θέση αυτή και η κατεύθυνση του αισθητήρα, δίνονται ως ορίσματα στην συνάρτηση υπολογισμού. Εκεί υπολογίζοντας το γινόμενο του κάθετου διανύσματος της επιφάνειας και του διανύσματος της κατεύθυνσης, προκύπτει μία μετρική αξιολόγησης της γωνίας αυτής.

Για τον υπολογισμό του κάθετου διανύσματος της επιφάνειας, χρησιμοποιείται η συνάρτηση `getNormals()` που παρέχει επίσης η βιβλιοθήκη OctoMap. Η συνάρτηση αυτή επιστρέφει τα κάθετα διανύσματα των κορυφών των τριγώνων τα οποία περιέχουν το σημείο το οποίο εξετάζεται. Είναι πιθανό κάποιο σημείο να μην βρίσκεται σε κάποια κορυφή και να βρίσκεται στο εσωτερικό ενός τριγώνου. Στην περίπτωση αυτή, αναζητούμε το κάθετο διάνυσμα γειτονικών κόμβων σε συγκεκρι-

μένη ακτίνα, καθώς θεωρούμε ότι βρίσκονται στην ίδια επιφάνεια και συνεπώς το κάθετο διάνυσμα θα ταυτίζεται. Τέλος, βρίσκουμε τον μέσο όρο των διανυσμάτων αυτών και το αποτέλεσμα θα χρησιμοποιηθεί για τον υπολογισμό της μετρικής.

Αλγόριθμος 4.4 Αλγόριθμος υπολογισμού καλύτερης γωνίας θέασης

```

1: function CALCULATECOVERAGE(point_on_wall, direction)
2:   coverage ← 0
3:   normals ← point_on_wall.getNormals()
4:   while normals.size() = 0 do
5:     normals ← point_on_wall.getNormals()    # Calculate normals in a larger
   bounding box around point_on_wall
6:   end while
7:   mean ← normals.mean()
8:   coverage ← direction · mean
9:   return coverage
10: end function

```

Μετά την εκτέλεση της διαδικασίας αυτής, αποθηκεύεται για κάθε θέση (x, y, z) η τιμή του yaw που προσφέρει την καλύτερη οπτική γωνία. Οι τιμές των roll και pitch θεωρούνται μηδενικές, καθώς είναι επιθυμητοί οι αργοί ελιγμοί, ώστε να επιτευχθεί ακρίβεια.

Στη συνέχεια, εφαρμόζεται μια περαιτέρω επεξεργασία σε αυτά, ώστε να αποκλειστούν σημεία τα οποία δεν είναι ορατά και επομένως δεν είναι προσβάσιμα από κανένα άλλο σημείο του χώρου. Για το λόγο αυτό, σε κάθε ένα από τα σημεία που έχουν προκύψει εφαρμόζεται ο αλγόριθμος πλησιέστερου γείτονα (*Nearest Neighbor Algorithm*), ώστε να ελεγχθούν γειτονικά σημεία του καθενός και να αποκλειστούν αυτά τα οποία δεν είναι προσβάσιμα με άμεσο τρόπο. Πιο συγκεκριμένα, για κάθε σημείο ελέγχουμε να βρούμε γειτονικά σημεία που απέχουν απόσταση μικρότερη από το 0.75 του εύρους του αισθητήρα RFID και δεν έχουν θεωρηθεί ήδη ως προσβάσιμα από κάποιο άλλο σημείο. Ο έλεγχος της ορατότητας για κάθε γείτονα πραγματοποιείται με χρήση της συνάρτησης *computeRayKeys()* της βιβλιοθήκης OctoMap. Η διαδικασία αυτή εφαρμόζεται αναδρομικά για κάθε σημείο, μέχρι να μην υπάρχει άλλο διαθέσιμο σημείο.

Για την οπτικοποίηση της επιφάνειας που καλύπτεται από τον αισθητήρα, δημιουργούμε έναν νέο χάρτη μορφής OctoMap, στον οποίο προστίθενται όλα τα σημεία που έχουν βρεθεί στον οπτικό πεδίο του. Έχοντας πλέον όλες τις θέσεις στις οποίες πρέπει να βρεθεί το ρομπότ με την σωστή κατεύθυνση, μπορούμε να υπολογίσουμε μια εκτίμηση του ποσοστού κάλυψης του χώρου σε m^3 . Η διαδικασία αυτή περιγράφεται στον [αλγόριθμο 4.5](#).

$$\text{coverage}(\%) = 100 * \frac{\text{covered_mapvolume}}{\text{initial_mapvolume}}$$

Όπως φαίνεται στην γραμμή 3 του αλγορίθμου, περιορίζουμε τον χώρο σε ένα ελάχιστο και μέγιστο ύψος. Όσα σημεία βρίσκονται εκτός των ορίων αυτών δεν λαμβάνονται υπόψη στον υπολογισμό του όγκου. Ο λόγος που συμβαίνει αυτό είναι ο αποκλεισμός σημείων του χώρου, όπως το δάπεδο, που δεν παρουσιάζουν

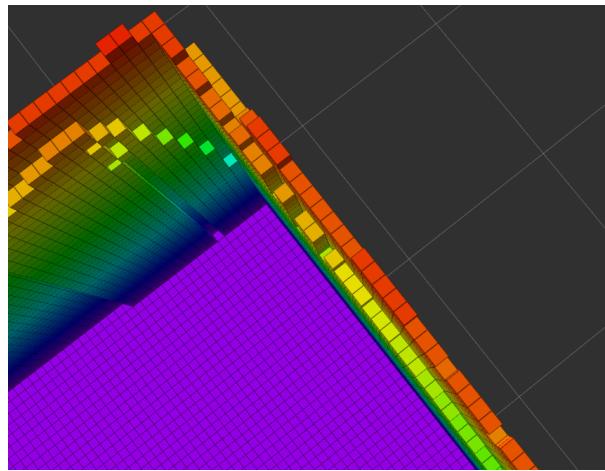
Αλγόριθμος 4.5 Υπολογισμός όγκου ενός OctoMap

```

1: volume  $\leftarrow 0$ 
2: for i = 1 to N do
3:   if node(i).z() > max_height OR node(i).z() < min_height then
4:     continue
5:   end if
6:   if node(i) is Occupied then
7:     volume += node(i).size * node(i).size * node(i).size
8:   end if
9: end for
10: return volume

```

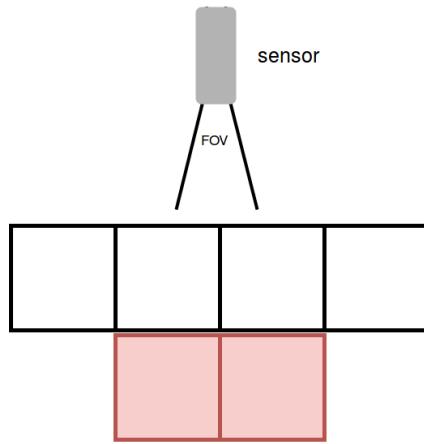
κανένα ενδιαφέρον για κάλυψη, αλλά και η εξάλειψη του θορύβου του χάρτη σε υψηλά σημεία. Επίσης, η ύπαρξη θορύβου στην αναπαράσταση του περιβάλλοντος δημιουργεί προβλήματα στην ακριβή αξιολόγηση της κάλυψης του χώρου. Συγκεκριμένα, υπάρχουν περιπτώσεις όπου δημιουργούνται σημεία τα οποία δεν είναι άμεσα ορατά από το ρομπότ και συνεπώς δεν θα έπρεπε να συνυπολογίζονται στον συνολικό όγκο. Αυτό συμβαίνει λόγω σφάλματος κατά την χαρτογράφηση του χώρου, όπου οι τοίχοι του περιβάλλοντος μπορούν να πάρουν τη μορφή που φαίνεται στο [σχήμα 4.6](#).



Σχήμα 4.6: Σφάλμα του χάρτη στην αναπαράσταση επιφανειών

Δηλαδή κάποια σημεία στις επιφάνειες αποτελούνται εσφαλμένα από δύο διαχοικούς κόμβους και δεν προσφέρουν κάποια χρήσιμη πληροφορία σχετική με το χώρο. Για το λόγο αυτό, εφαρμόζουμε μια επιπλέον επεξεργασία του χάρτη προτύπου υπολογιστεί ο όγκος του. Η διαδικασία αυτή φαίνεται στο [σχήμα 4.7](#), κατά την οποία αποκλείονται τους εξωτερικούς κόμβους από τον υπολογισμό του όγκου, καθώς αρκεί η κάλυψη των άμεσα ορατών από τον αισθητήρα.

Με τα βήματα που περιγράφηκαν έως τώρα γίνεται ένας τυχαίος υπολογισμός των σημείων που προσφέρουν την καλύτερη δυνατή κάλυψη του χώρου. Στη συνέχεια, πρέπει να βρεθεί η σειρά με την οποία το drone πρέπει να προσπελάσει αυτά τα σημεία, ώστε η κάλυψη του χώρου να γίνεται με όσο το δυνατόν πιο βέλτιστο



Σχήμα 4.7: Εύρεση σημείων που δεν λαμβάνονται υπόψη στον υπολογισμό του όγκου ενός OctoMap

τρόπο, είτε όσον αφορά το χρόνο κάλυψης, είτε την απόσταση που θα διανύσει.

Για την επίλυση του προβλήματος αυτού, απαιτείται να το προσεγγίσουμε ως πρόβλημα του περιοδεύοντος πωλητή (*Travelling Salesman Problem*). Εν συντομίᾳ, πρόκειται για την περίπτωση όπου ένας πωλητής πρέπει να επισκεφτεί n διαφορετικές πόλεις και πρέπει να υπολογιστεί η συντομότερη και χαμηλότερου κόστους διαδρομή, ώστε να επισκεφτεί κάθε πόλη μόνο μία φορά. Πρόκειται για ένα NP-hard πρόβλημα, του οποίου όσο αυξάνεται ο αριθμός των πόλεων, τόσο αυξάνεται και η πολυπλοκότητά του.

Αρχικά, επιχειρήθηκε η επίλυση του προβλήματος με τη χρήση του αλγορίθμου *Hill climbing*. Όμως, το μεγάλο πλήθος των σημείων δεν επέτρεπε την εκτέλεση του αλγορίθμου σε λογικό χρόνο και η βελτίωση που υπήρχε ήταν πολύ μικρή. Όπως είναι λογικό, στην περίπτωση κάλυψης χώρου για μία βιομηχανική αποθήκη είναι σημαντική η πιθανότητα ο αριθμός των σημείων να είναι πολύ μεγάλος, με αποτέλεσμα να γίνεται αρκετά πολύπλοκη η επίλυση του προβλήματος με οποιονδήποτε τρόπο.

Για το λόγο αυτό, επεξεργαζόμαστε τα σημεία που έχουν προκύψει, ώστε να μειωθεί ο αριθμός τους και να λυθεί πιο εύκολα το πρόβλημα της ελαχιστοποίησης του κόστους της διαδρομής. Ως κόστος, θεωρούμε την απόσταση που θα διανύσει το drone για να επιτύχει την πλήρη κάλυψη.

Ένας τρόπος για να μειωθεί το πλήθος των σημείων είναι η μείωση των διαστάσεων αυτών. Τα σημεία έως τώρα είναι της μορφής $\kappa = (x, y, z, \theta)$. Το πλήθος αυτών θα μπορούσε να μειωθεί τόσες φορές, όσα είναι και τα διαφορετικά ύψη στα οποία θα πρέπει να βρεθεί το drone, κρατώντας μόνο την τετυμένη και τεταγμένη των σημείων. Ουσιαστικά, πρόκειται για μια προβολή των τρισδιάστατων σημείων στον δισδιάστατο χώρο.

Πλέον, καθώς ο αριθμός των σημείων είναι αρκετά μικρότερος, μπορεί να εφαρμοστεί σε αυτά κάποιος αλγόριθμος για την εύρεση του βέλτιστου μονοπατιού. Δημιουργείται ένας μη κατευθυνόμενος γράφος με όλα τα σημεία (x, y) που έχουν προκύψει, χρησιμοποιώντας την βιβλιοθήκη *Boost Graph Library*¹⁴. Για κάθε σημείο,

¹⁴https://www.boost.org/doc/libs/1_66_0/libs/graph/doc/index.html

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

προστίθεται ως γειτονικός κόμβος κάθε άλλο σημείο που απέχει απόσταση μικρότερη από το 0.75 του εύρους του αισθητήρα κάλυψης. Το βάρος μεταξύ των δύο κόμβων πρόκειται για την απόσταση αυτή.

Για την εύρεση του βέλτιστου μονοπατιού, χρησιμοποιούμε έναν συνδυασμό των αλγορίθμων Nearest Neighbor και Hill Climbing, ώστε να αξιοποιήσουμε γειτονικά σημεία και να προσπαθήσουμε να μειώσουμε τις αποστάσεις μεταξύ σημείων που βρίσκονται αρκετά μακριά με ευριστικό τρόπο. Για τον υπολογισμό της απόστασης μεταξύ των κόμβων στο γράφο, χρησιμοποιείται ο αλγόριθμος A^* . Πιο συγκεκριμένα, για κάθε ένα σημείο (x, y) ελέγχουμε την απόσταση του από το επόμενο και το προηγούμενο του, καθώς ο τρόπος με τον οποίο δημιουργήθηκαν τα σημεία, επιτρέπει την ύπαρξη γειτονικών σημείων διαδοχικά στη δομή που αποθηκεύτηκαν. Επίσης, γνωρίζουμε την ελάχιστη απόσταση που μπορούν να έχουν δύο σημεία, η οποία είναι ίση με το βήμα δειγματοληψίας που χρησιμοποιήθηκε παραπάνω. Εάν κανένα από τα δύο γειτονικά δεν καλύπτουν τις απαιτήσεις απόστασης, επιχειρούμε να βρούμε το καλύτερο επόμενο σημείο επιλέγοντας έναν τυχαίο κόμβο και ελέγχοντας την απόσταση του και αν είναι άμεσα προσβάσιμος με εύκολο τρόπο (π.χ. ενώνονται με ευθεία γραμμή). Η αναζήτηση συνεχίζεται για πεπερασμένο αριθμό επαναλήψεων ή μέχρις ότου βρεθεί σημείο που να βρίσκεται σε ελάχιστη απόσταση.

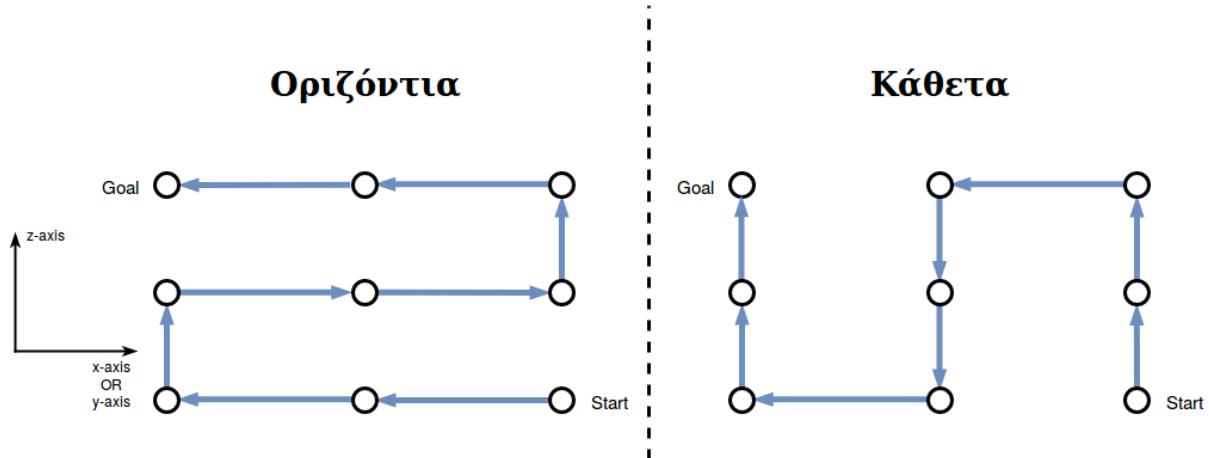
Η παραπάνω διαδικασία εκτελείται για συγκεκριμένο αριθμό επαναλήψεων, ώστε να βελτιωθεί όσο τον δυνατόν περισσότερο η λύση που προκύπτει. Συνολικά, η διαδικασία παρουσιάζεται στον [αλγόριθμο 4.6](#).

Αλγόριθμος 4.6 Αλγόριθμος εύρεσης βέλτιστου μονοπατιού

```
function CALCULATEOPTIMALPATH(points)
    Calculate initial total distance
    for i in restarts do
        Starting from node 0
        while there are nodes left do
            CHECK NEAREST NEIGHBOR
            Check the node after and the node before
            if at least one of them is visible and in a small distance then
                Make this node the current node and skip Hill Climbing below
            end if
            HILL CLIMBING
            do
                Find a random node and calculate its distance from current node
                Keep the node with the minimum distance
            while any distance found is larger than the minimum AND max number
            of iterations is not reached
            end while
            Keep the order of points with the minimum total cost
        end for
    end function
```

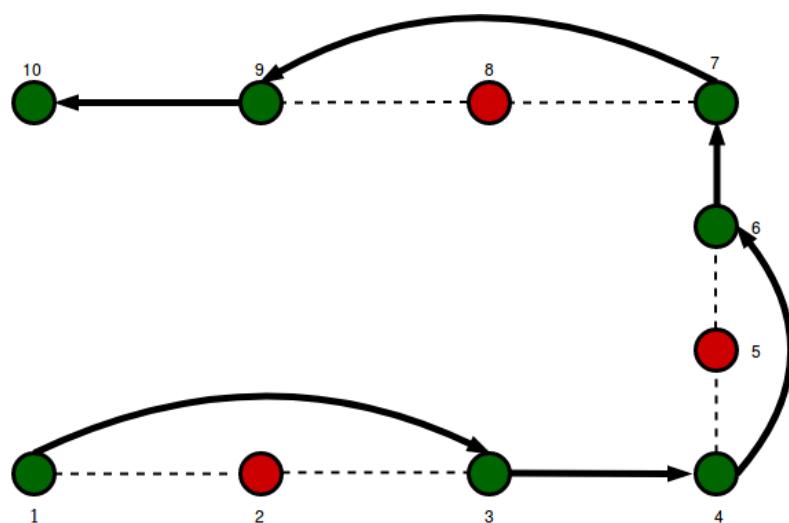
Εφόσον έχει υπολογιστεί ένα μονοπάτι που συνδέει όλα τα σημεία στον δι-

διάστατο χώρο, η λύση μπορεί να μεταφερθεί ξανά στον τρισδιάστατο χώρο. Αυτό είναι δυνατόν να συμβεί με δύο τρόπους, με την οριζόντια ή κάθετη ένωση των σημείων. Στο [σχήμα 4.8](#) φαίνονται με πιο ξεκάθαρο τρόπο οι δύο αυτές μέθοδοι.



Σχήμα 4.8: Οριζόντια και κάθετη ένωση των σημείων για επαναδιάταξη στον τρισδιάστατο χώρο

Τέλος, εφαρμόζεται ένα τελευταίο στάδιο επεξεργασίας του μονοπατιού. Ελέγχονται διαδοχικά όλα τα σημεία και αφαιρούνται αυτά τα οποία βρίσκονται στο ενδιάμεσο άλλων σημείων στους άξονες x , y και z και απέχουν μεταξύ τους την δεδομένη απόσταση δειγματοληψίας. Με τον τρόπο αυτό, επιδιώκουμε να έχουμε όσο το δυνατόν λιγότερα σημεία-στόχους, ώστε να είναι πιο ομαλή και συνεχόμενη η πορεία του ρομπότ. Πιο συγκεκριμένα, όπως φαίνεται στο [σχήμα 4.9](#), τα σημεία 2, 5 και 8 παραλείπονται, καθώς είναι λογικό ότι κατά την μετάβαση του ρομπότ από το 1 στο 3, και αντίστοιχα για τα υπόλοιπα σημεία, θα διασχίσει το ενδιάμεσο αυτών, χωρίς να χρειάζεται να το ορίσουμε.

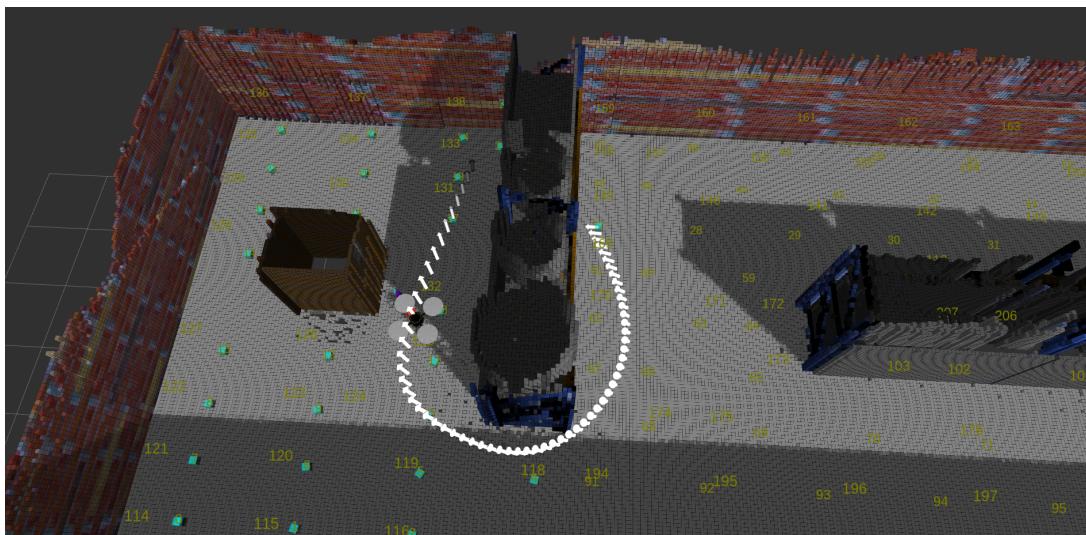


Σχήμα 4.9: Τελευταίο στάδιο επεξεργασίας μονοπατιού κάλυψης

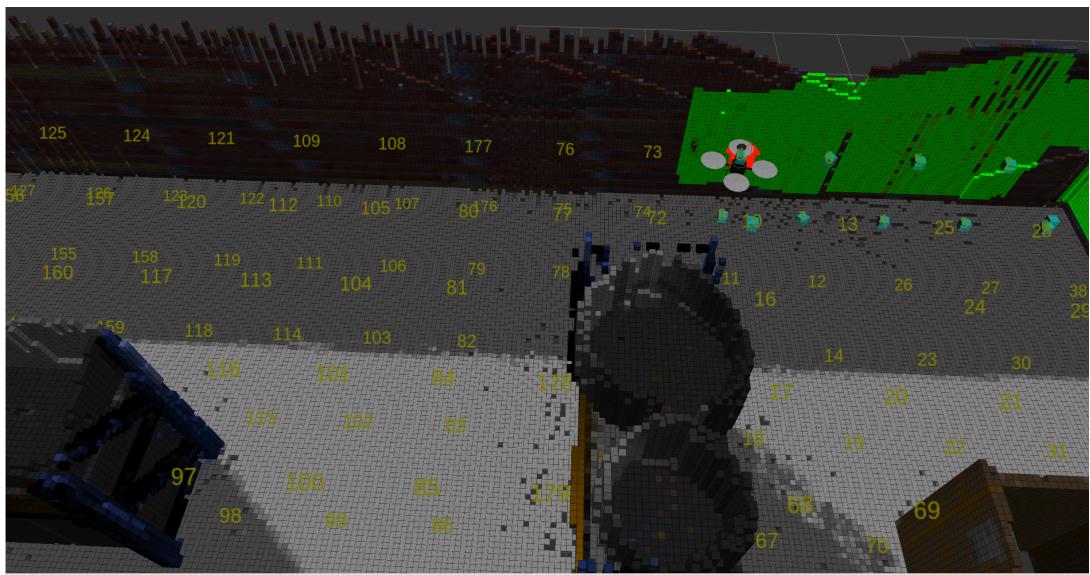
Πλέον, το σύνολο των σημείων που πρέπει να διασχίσει το ρομπότ, ώστε να

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

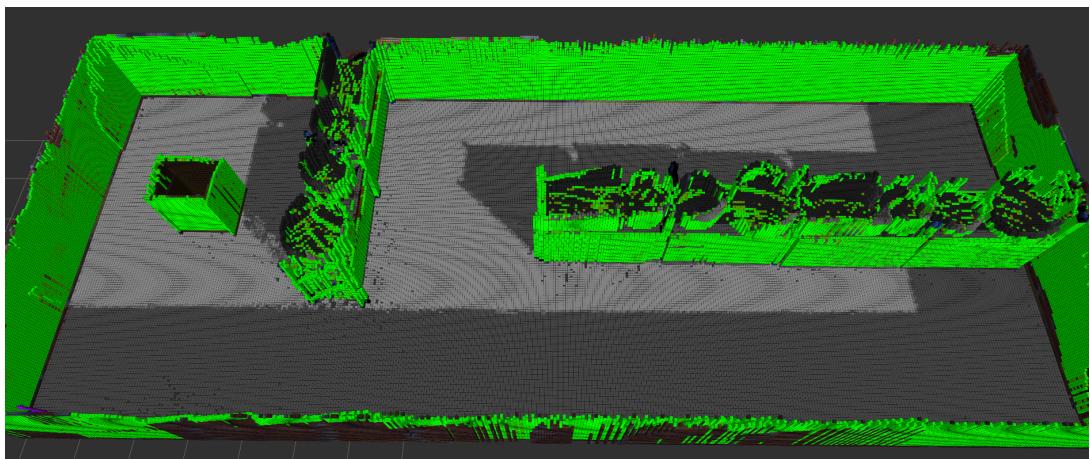
πετύχει την πλήρη κάλυψη του χώρου είναι διαθέσιμο. Για την πλοιήγηση του χρονισμοποιείται η βιβλιοθήκη OMPL, όπως περιγράφεται στο [υποκεφάλαιο 3.7](#) και ο αλγόριθμος RRT^* για την δημιουργία ασφαλών μονοπατιών μεταξύ των σημείων ([σχήμα 4.10](#)). Για τον έλεγχο του drone, χρησιμοποιείται ο PID ελεγκτής θέσης, όπως περιγράφεται στο [υποκεφάλαιο 3.2](#). Για την εξασφάλιση πιο ομαλής κίνησης, δεχόμαστε ότι το drone δεν χρειάζεται να βρεθεί ακριβώς στην θέση που του υποδηλώνουμε, αλλά είναι επιτρεπτή μια σχετικά μικρή απόσταση ώστε να θεωρηθεί ότι ο στόχος επετεύχθη και να αποσταλεί ο επόμενος. Έχει δημιουργηθεί επίσης ένας ROS κόμβος ο οποίος κατά την πλοιήγηση του drone στο χώρο δείχνει σε πραγματικό χρόνο την περιοχή που καλύπτει ο αισθητήρας RFID και το ποσοστό κάλυψης κάθε στιγμή ([σχήμα 4.11](#)). Αφού καλυφθεί όλος ο χώρος, όπως φαίνεται στο [σχήμα 4.12](#), το drone επιστρέφει στην αρχική του θέση.



Σχήμα 4.10: Δημιουργία μονοπατιού στο χώρο με αποφυγή εμποδίων



Σχήμα 4.11: Κάλυψη χώρου από το drone σε πραγματικό χρόνο



Σχήμα 4.12: Ολοκληρωμένη κάλυψη χώρου από το drone

5

Πειράματα - Αποτελέσματα

Τα πειράματα χωρίζονται σε δύο φάσεις. Η πρώτη αφορά το σύστημα εντοπισμού θέσης και η δεύτερη το σύστημα πλήρους κάλυψης χώρου από το drone. Για την εκτέλεση των πειραμάτων χρησιμοποιήθηκε υπολογιστής με τα παρακάτω χαρακτηριστικά:

Πίνακας 5.1: Χαρακτηριστικά συστήματος που χρησιμοποιήθηκε για την εκτέλεση των πειραμάτων

Επεξεργαστής	Κάρτα Γραφικών	Μνήμη RAM	Λειτουργικό Σύστημα
Intel® Core™ i7-7500	Intel® HD Graphics 620	8 GB	Ubuntu 16.04 64bit

Οι εκδόσεις των βιβλιοθηκών και εργαλείων που χρησιμοποιήθηκαν είναι:

Πίνακας 5.2: Εκδόσεις βιβλιοθηκών και εργαλείων που χρησιμοποιήθηκαν

ROS	Gazebo	OctoMap	OMPL
Kinetic	7.0.0	1.8.1	1.2.1

Για την εκτέλεση των πειραμάτων, τα δεδομένα που απαιτούνται για την εξαγωγή αποτελεσμάτων αποθηκεύτηκαν σε αρχεία `rosbag`¹⁵. Στη συνέχεια, έγινε εξαγωγή τους σε αρχεία μορφής `.csv` και χρησιμοποιήθηκε κώδικας σε Python για τον υπολογισμό μετρικών και τη δημιουργία διαγραμμάτων.

¹⁵<http://wiki.ros.org/rosbag>

5.1 ΠΕΙΡΑΜΑΤΑ ΕΝΤΟΠΙΣΜΟΥ ΘΕΣΗΣ

Σκοπός των πειραμάτων αυτών είναι ο υπολογισμός της απόκλισης της εκτίμησης θέσης από την πραγματική. Για το λόγο αυτό, χρειαζόμαστε την πραγματική θέση (ground truth) του ρομπότ. Αυτό παρέχεται με την χρήση του plugin GazeboRosP3D¹⁶ στον προσομοιωτή Gazebo, το οποίο εξάγει την πληροφορία αυτή στο ROS topic `/ground_truth/state`. Η εκτίμηση της θέσης από το σύστημα που δημιουργήθηκε αποστέλλεται στο ROS topic `/amcl_pose`.

Ο αλγόριθμος εξετάστηκε σε τρία είδη κινήσεων:

- Ευθεία κίνηση
- Κίνηση σε σπιράλ
- Κίνηση σε μαίανδρο

Για κάθε μία από αυτές τις περιπτώσεις εκτελέστηκαν πέντε διαφορετικές δοκιμές, με τρεις διαφορετικές ταχύτητες κίνησης του drone σε καθεμία. Σε όλα τα πειράματα, χρησιμοποιήθηκε το 100% του αριθμού των σωματιδίων για την εξαγωγή της τελικής θέσης, ενώ οι παράμετροι του αλγορίθμου παρέμειναν σταθεροί και οι τιμές τους παρουσιάζονται στον πίνακα 5.3. Δεν υπάρχει κάποιος συγκεκριμένος τρόπος για τον υπολογισμό των τιμών αυτών και για το λόγο αυτό επιλέχθηκαν μετά από αρκετές δοκιμές. Καθώς όμως γνωρίζουμε κάποια βασικά χαρακτηριστικά των περιβαλλόντων που μπορεί να χρησιμοποιηθεί το σύστημα εντοπισμού θέσης, δόθηκε μεγαλύτερη έμφαση σε μετρήσεις που βρίσκονται εντός του εύρους του αισθητήρα απόστασης σε γνωστά εμπόδια, και λιγότερη σε πολύ κοντινές μετρήσεις και απροσδόκητα εμπόδια.

Πίνακας 5.3: Τιμές παραμέτρων αλγορίθμου εντοπισμού θέσης

Αριθμός σωματιδίων	z_{hit}	z_{short}	z_{rand}	z_{max}	laser σ_{hit}	laser λ_{short}
300	0.6	0.1	0.1	0.2	0.02	0.1

Τα δεδομένα που χρησιμοποιούνται είναι της μορφής:

Πίνακας 5.4: Μορφή δεδομένων για εξαγωγή αποτελεσμάτων του localization

Timestamp	Error in x	Error in y	Error in z	Error in yaw
...

Από τα δεδομένα αυτά, υπολογίστηκε το *Absolute Position Error* και με την χρήση αυτού υπολογίστηκαν οι παρακάτω μετρικές για το Σφάλμα θέσης (σε μέτρα) και προσανατολισμού (σε rad).

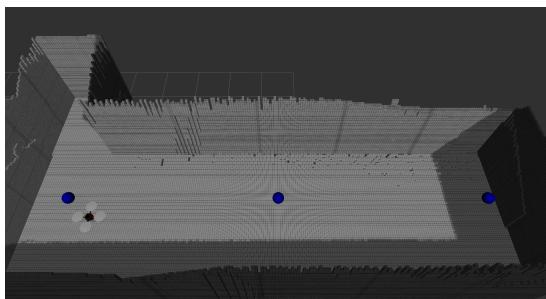
¹⁶http://docs.ros.org/electric/api/gazebo_plugins/html/group__GazeboRosP3D.html

$$\text{Absolute Position Error} = \sqrt{\text{error_x}^2 + \text{error_y}^2 + \text{error_z}^2}$$

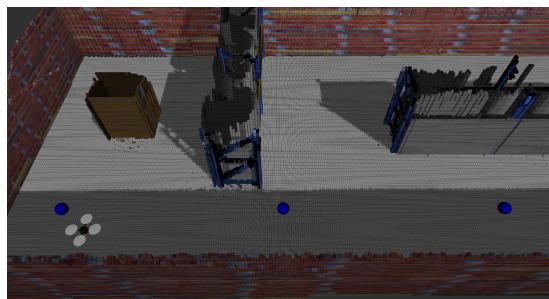
- $\text{Mean} = \frac{1}{n} (\sum_{i=1}^n \text{error}_i) = \frac{\text{error}_1 + \text{error}_2 + \dots + \text{error}_n}{n}$
- $\text{Median} = \frac{\text{error}_{\lfloor (\#n+1)/2 \rfloor} + \text{error}_{\lceil (\#n+1)/2 \rceil}}{2}$
- *Min Error*
- *Max Error*
- $\text{Root Mean Square Error} = \sqrt{\frac{\sum_{i=1}^N (\text{error}_i)^2}{N}}$
- $\text{Sum of Squared Error} = \sum_{i=1}^n \text{error}_i^2$
- $\text{Standard Deviation} = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (\text{error}_i - \bar{\text{error}})^2}$

5.1.1 Κίνηση σε ευθεία γραμμή

Η κίνηση σε ευθεία γραμμή εξετάστηκε σε δύο διαφορετικούς χώρους, έναν διάδρομο και μία αποθήκη. Το drone έπρεπε να φτάσει σε ένα συγκεκριμένο ύψος και στη συνέχεια να διασχίσει τρία σημεία που δημιουργούσαν μία ευθεία στο χώρο. Η κίνηση αυτή στον διάδρομο, φαίνεται στο [σχήμα 5.1α'](#), ενώ στην αποθήκη στο [σχήμα 5.1β'](#).



(α') Διάδρομος



(β') Αποθήκη

Σχήμα 5.1: Κίνηση σε ευθεία γραμμή

Το περιβάλλον του διαδρόμου περιέχει αρκετά συμμετρικά χαρακτηριστικά, κάτι που πιθανόν να οδηγήσει σε μεγαλύτερο σφάλμα. Αντίθετα, η αποθήκη παρουσιάζει μεγαλύτερο ενδιαφέρον, καθώς πρόκειται για ένα πιο ρεαλιστικό περιβάλλον. Όπως προαναφέρθηκε, εξετάστηκαν τρεις διαφορετικές ταχύτητες κίνησης του ρομπότ σε κάθε χώρο. Τα αποτελέσματα των πειραμάτων αυτών, φαίνονται αναλυτικά στους παρακάτω πίνακες.

ΚΕΦΑΛΑΙΟ 5. ΠΕΙΡΑΜΑΤΑ - ΑΠΟΤΕΛΕΣΜΑΤΑ

Πίνακας 5.5: Σφάλμα θέσης (σε μέτρα) για κίνηση σε ευθεία γραμμή με χαμηλή ταχύτητα σε διάδρομο

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.13894	0.14003	0.01624	0.19890	19.18185	0.02638	0.14142
2	0.15154	0.13033	0.01023	0.30975	27.85205	0.06365	0.16436
3	0.14150	0.13463	0.01266	0.34980	23.42868	0.06351	0.15509
4	0.08683	0.07937	0.00766	0.19531	8.85282	0.04161	0.09628
5	0.11425	0.11648	0.00958	0.24263	14.23755	0.03994	0.12102
Avg.	0.12661	0.12017	0.01127	0.25928	18.71059	0.04702	0.13563

Πίνακας 5.6: Σφάλμα θέσης (σε μέτρα) για κίνηση σε ευθεία γραμμή με κανονική ταχύτητα σε διάδρομο

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.16242	0.17036	0.00997	0.32717	15.20109	0.07639	0.17945
2	0.22075	0.26521	0.00718	0.38309	27.18928	0.09102	0.23874
3	0.08396	0.07386	0.00522	0.21196	4.34510	0.04520	0.09534
4	0.11701	0.12045	0.01269	0.33264	10.98280	0.08984	0.14747
5	0.15474	0.15491	0.01836	0.35479	13.49755	0.06558	0.16804
Avg.	0.14778	0.15696	0.01068	0.32193	14.24316	0.07361	0.16581

Πίνακας 5.7: Σφάλμα θέσης (σε μέτρα) για κίνηση σε ευθεία γραμμή με υψηλή ταχύτητα σε διάδρομο

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.24806	0.21986	0.01206	0.57617	32.85699	0.14016	0.28483
2	0.17828	0.11405	0.00874	0.47371	17.97368	0.12724	0.21893
3	0.19176	0.13981	0.01011	0.58964	21.81091	0.12578	0.22925
4	0.33894	0.28538	0.00933	0.73504	72.62943	0.23017	0.40956
5	0.18411	0.12833	0.01144	0.56394	20.48029	0.13074	0.22571
Avg.	0.22823	0.17749	0.01033	0.58770	33.15026	0.15082	0.27366

5.1. ΠΕΙΡΑΜΑΤΑ ΕΝΤΟΠΙΣΜΟΥ ΘΕΣΗΣ

Πίνακας 5.8: Σφάλμα θέσης (σε μέτρα) για κίνηση σε ευθεία γραμμή με χαμηλή ταχύτητα σε αποθήκη

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.114325	0.111956	0.007847	0.330900	16.779918	0.054259	0.126536
2	0.187610	0.215719	0.015321	0.395761	43.740688	0.100231	0.212681
3	0.126385	0.100733	0.015605	0.269795	19.686598	0.065937	0.142536
4	0.082295	0.086740	0.009050	0.156036	7.810667	0.034388	0.089184
5	0.116021	0.111720	0.011625	0.246022	14.842384	0.040495	0.122878
Avg.	0.125327	0.125374	0.011890	0.279703	20.572051	0.059062	0.138763

Πίνακας 5.9: Σφάλμα θέσης (σε μέτρα) για κίνηση σε ευθεία γραμμή με κανονική ταχύτητα σε αποθήκη

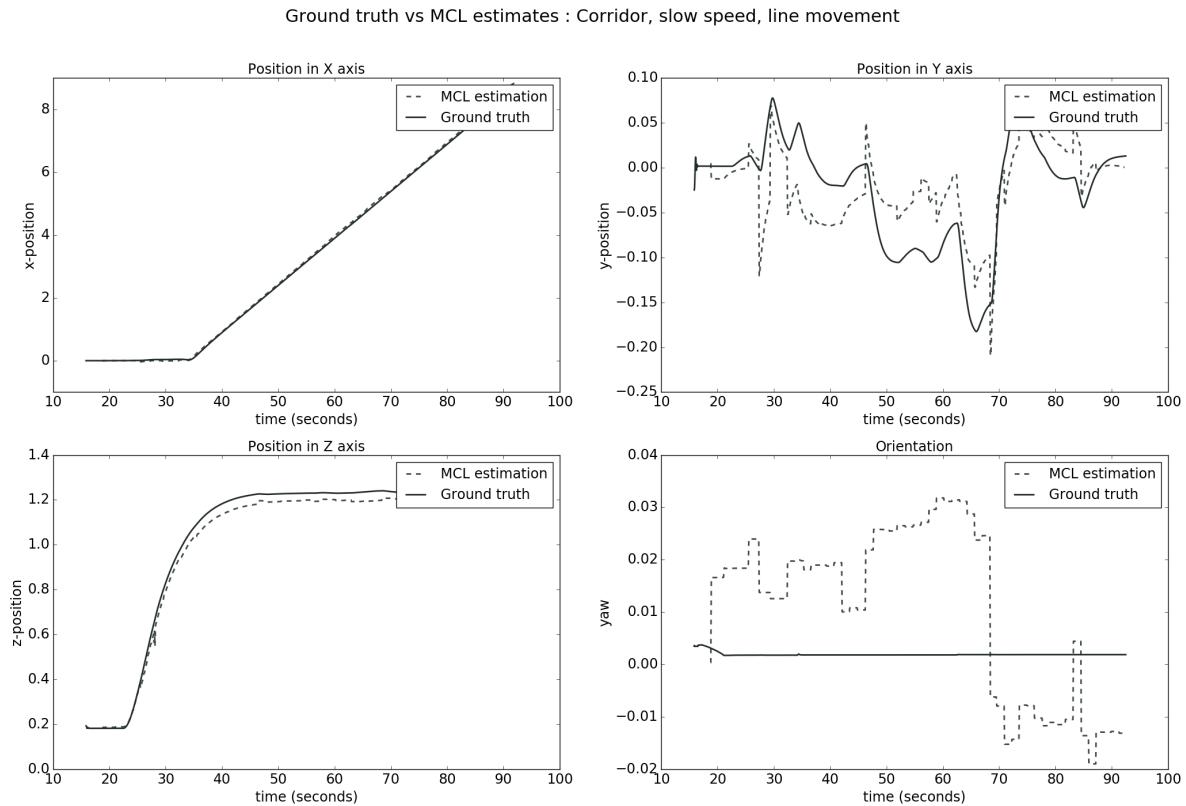
#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.208280	0.199181	0.017934	0.544024	28.269758	0.125675	0.243191
2	0.115554	0.120715	0.004416	0.354717	11.331192	0.075929	0.138233
3	0.092544	0.077934	0.002976	0.295298	5.014900	0.047663	0.104074
4	0.168933	0.153410	0.016503	0.348709	18.885880	0.115212	0.204409
5	0.088185	0.091700	0.009636	0.213946	4.842245	0.049877	0.101287
Avg.	0.134699	0.128588	0.010293	0.351339	13.668795	0.082871	0.158238

Πίνακας 5.10: Σφάλμα θέσης (σε μέτρα) για κίνηση σε ευθεία γραμμή με υψηλή ταχύτητα σε αποθήκη

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.136676	0.074564	0.006855	0.395738	11.975489	0.110164	0.175457
2	0.154984	0.092472	0.010285	0.450830	15.579335	0.121538	0.196862
3	0.223545	0.203554	0.016012	0.461632	23.246522	0.090357	0.241073
4	0.286846	0.314360	0.016945	0.590272	46.801999	0.095643	0.302341
5	0.182972	0.104353	0.005001	0.627469	24.213108	0.169823	0.249488
Avg.	0.197004	0.157861	0.011020	0.505188	24.363290	0.117505	0.233044

Για όλες τις παραπάνω περιπτώσεις έχει υπολογιστεί και το σφάλμα προσανατολισμού του drone. Δεν αναφέρεται όμως αναλυτικά, καθώς σε όλες τις δοκιμές που πραγματοποιήθηκαν, ανεξαρτήτως του περιβάλλοντος και της ταχύτητας, το μέσο σφάλμα είναι συνεχώς μικρότερο του 0.1 rad. Αυτό προφανώς οφείλεται στο γεγονός το ρομπότ δεν πραγματοποιεί καμία περιστροφή κατά την κίνηση του και η κατεύθυνσή του λαμβάνεται απευθείας από τον αισθητήρα IMU που διαθέτει. Στο [σχήμα 5.2](#) και στο [σχήμα 5.3](#), όπου παρέχεται η οπτικοποίηση της πραγματικής και της εκτιμώμενης πορείας στον χώρο, το μεγαλύτερο σφάλμα εμφανίζεται

ΚΕΦΑΛΑΙΟ 5. ΠΕΙΡΑΜΑΤΑ - ΑΠΟΤΕΛΕΣΜΑΤΑ



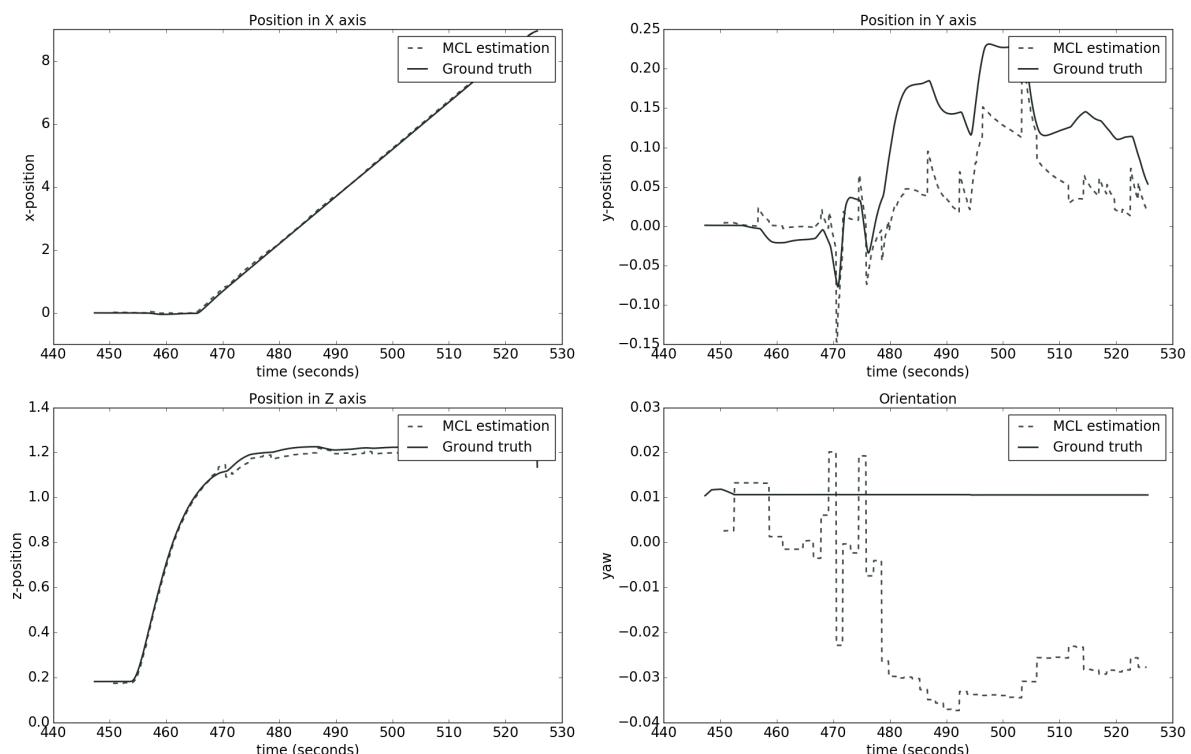
Σχήμα 5.2: Οπτικοποίηση καλύτερης πορείας για κίνηση σε ευθεία γραμμή σε διάδρομο

ως προς τον γάληνο, με αποτέλεσμα να δημιουργείται μια αστάθεια στην ευθεία κίνηση του ρομπότ.

Όπως παρατηρούμε, το μικρότερο σφάλμα παρουσιάζεται κατά την αργή κίνηση του ρομπότ και στις δύο περιπτώσεις. Εκτός της μέσης τιμής, χαμηλή παραμένει και η μέγιστη τιμή που μπορεί να αποκτήσει το σφάλμα, σε αντίθεση με την περίπτωση της υψηλής ταχύτητας. Εκεί όπου το σφάλμα παρουσιάζει και μεγαλύτερη διασπορά, το μέγιστο σφάλμα είναι αρκετά υψηλό για να εγγυηθεί ασφαλή πλοήγηση του drone στο χώρο.

5.1. ΠΕΙΡΑΜΑΤΑ ΕΝΤΟΠΙΣΜΟΥ ΘΕΣΗΣ

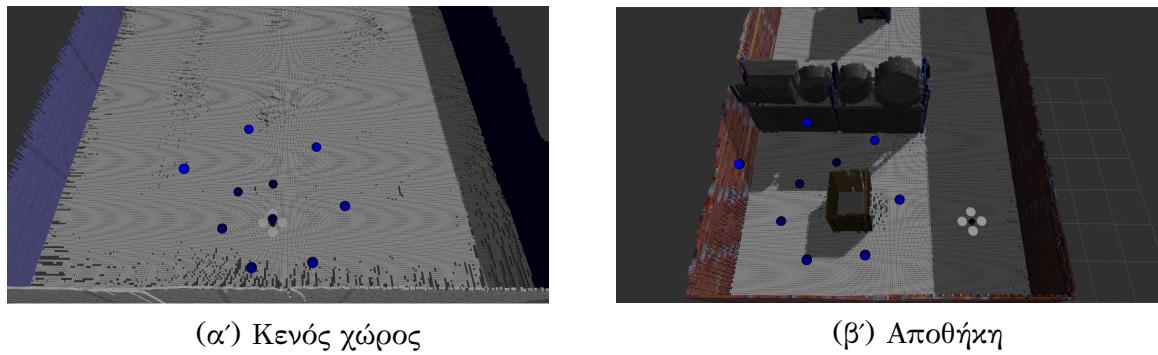
Ground truth vs MCL estimates : Warehouse, slow speed, line movement



Σχήμα 5.3: Οπτικοποίηση καλύτερης πορείας για κίνηση σε ευθεία γραμμή σε αποθήκη

5.1.2 Κίνηση σε σπιράλ

Στη συνέχεια, εξετάστηκε η κίνηση σε πορεία μορφής σπιράλ. Κατά την κίνηση αυτή, το drone καλείται να κινηθεί κυκλικά και προς τα πάνω, καθώς κάθε σημείο βρίσκεται ψηλότερα από το προηγούμενό του. Κατά τη διάρκεια της κίνησης, ο προσανατολισμός του drone παραμένει σταθερός και ίδιος και με τον αρχικό. Επομένως, τα επόμενα πειράματα μελετούν τη συμπεριφορά του συστήματος, καθώς το drone μεταβάλλει τις συντεταγμένες x , y και z ταυτόχρονα. Οι χώροι που χρησιμοποιήθηκαν ήταν ένας κενός χώρος, χωρίς καθόλου εμπόδια παρά μόνο τοίχους και η αποθήκη που αναφέρθηκε και προηγουμένως, όπως φαίνεται στο [σχήμα 5.4](#).



Σχήμα 5.4: Κίνηση σε σπιράλ

Παρακάτω παρουσιάζεται το σφάλμα που υπήρξε σε κάθε διαφορετική περίπτωση. Το σφάλμα του προσανατολισμού δεν παρουσιάζεται για το περιβάλλον του κενού χώρου, καθώς παραμένει μικρότερο του 0.1 rad για κάθε δοκιμή. Αντίθετα, κατά την κίνηση στον χώρο της αποθήκης, το σφάλμα μεταβάλλεται και παρουσιάζεται παρακάτω.

Πίνακας 5.11: Σφάλμα θέσης (σε μέτρα) για κίνηση σε σπιράλ με χαμηλή ταχύτητα σε κενό χώρο

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.26975	0.31742	0.01130	0.54886	112.97555	0.15513	0.31114
2	0.11821	0.12610	0.00211	0.35319	23.34022	0.06880	0.13676
3	0.16422	0.17058	0.00408	0.49477	45.33603	0.09678	0.19060
4	0.09561	0.08395	0.01105	0.31162	16.14039	0.06017	0.11296
5	0.08542	0.09533	0.00544	0.26510	12.17983	0.04609	0.09706
Avg.	0.14664	0.15868	0.00680	0.39471	41.99440	0.08539	0.16970

5.1. ΠΕΙΡΑΜΑΤΑ ΕΝΤΟΠΙΣΜΟΥ ΘΕΣΗΣ

Πίνακας 5.12: Σφάλμα θέσης (σε μέτρα) για κίνηση σε σπιράλ με κανονική ταχύτητα σε κενό χώρο

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.10616	0.07093	0.01113	0.41376	14.79245	0.10061	0.14621
2	0.15549	0.13576	0.00313	0.52998	35.06631	0.14126	0.21002
3	0.10490	0.08614	0.00346	0.42966	14.19397	0.08920	0.13766
4	0.12569	0.08250	0.00794	0.39558	17.18100	0.08748	0.15310
5	0.10362	0.08400	0.00438	0.36648	12.58463	0.08410	0.13342
Avg.	0.11917	0.09186	0.00601	0.42709	18.76367	0.10053	0.15608

Πίνακας 5.13: Σφάλμα θέσης (σε μέτρα) για κίνηση σε σπιράλ με υψηλή ταχύτητα σε κενό χώρο

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.13799	0.14837	0.00192	0.55425	22.96128	0.12062	0.18322
2	0.14768	0.10509	0.01162	0.59520	27.27895	0.13518	0.20014
3	0.21043	0.14093	0.00524	0.60197	52.90884	0.17706	0.27493
4	0.11989	0.10549	0.00683	0.39358	16.66327	0.09719	0.15429
5	0.10156	0.06495	0.01010	0.49884	13.70723	0.09663	0.14014
Avg.	0.14351	0.11297	0.00714	0.52877	26.70392	0.12533	0.19054

Πίνακας 5.14: Σφάλμα θέσης (σε μέτρα) για κίνηση σε σπιράλ με χαμηλή ταχύτητα σε αποθήκη

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.25256	0.14320	0.00475	1.28265	273.42973	0.30430	0.39539
2	0.18359	0.19685	0.01235	0.72798	61.80427	0.11271	0.21540
3	0.14626	0.11277	0.00233	1.29595	84.01740	0.17188	0.22565
4	0.16322	0.14987	0.00663	1.23738	73.79179	0.16350	0.23099
5	0.22972	0.19084	0.01103	0.63412	113.64658	0.17994	0.29176
Avg.	0.19507	0.15871	0.00742	1.03562	121.33796	0.18646	0.27184

ΚΕΦΑΛΑΙΟ 5. ΠΕΙΡΑΜΑΤΑ - ΑΠΟΤΕΛΕΣΜΑΤΑ

Πίνακας 5.15: Σφάλμα προσανατολισμού (σε rad) για κίνηση σε σπιράλ με χαμηλή ταχύτητα σε αποθήκη

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.60339	0.04200	0.00003	6.21851	5990.57539	1.75008	1.85071
2	0.06974	0.04814	0.00001	6.20961	196.46472	0.37780	0.38405
3	1.08731	0.03621	0.00009	6.25999	9651.88863	2.16106	2.41860
4	0.05675	0.05289	0.00005	3.96727	29.33249	0.13417	0.14563
5	0.15260	0.10726	0.00024	6.11451	470.37756	0.57384	0.59358
Avg.	0.39396	0.05730	0.00008	5.75398	3267.72776	0.99939	1.07851

Πίνακας 5.16: Σφάλμα θέσης (σε μέτρα) για κίνηση σε σπιράλ με κανονική ταχύτητα σε αποθήκη

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.15150	0.13640	0.00369	0.34122	31.77614	0.10388	0.18366
2	0.16504	0.17102	0.00390	0.44813	34.05847	0.10344	0.19474
3	0.18469	0.17301	0.00221	0.48730	43.33518	0.10551	0.21268
4	0.19868	0.15884	0.01203	0.60410	48.55944	0.14970	0.24871
5	0.10943	0.08234	0.00127	0.47569	19.55174	0.10302	0.15025
Avg.	0.16187	0.14432	0.00462	0.47129	35.45619	0.11311	0.19801

Πίνακας 5.17: Σφάλμα προσανατολισμού (σε rad) για κίνηση σε σπιράλ με κανονική ταχύτητα σε αποθήκη

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.04635	0.01585	0.00010	5.43492	85.80986	0.29839	0.30181
2	0.03806	0.03612	0.00000	0.56901	2.00954	0.02810	0.04730
3	0.13571	0.02801	0.00018	6.22394	578.76217	0.76572	0.77726
4	3.20807	6.17517	0.00252	6.23041	15512.83529	3.07926	4.44539
5	0.02982	0.01975	0.00072	2.42426	6.87179	0.08398	0.08907
Avg.	0.69160	1.25498	0.00070	4.17651	3237.25773	0.85109	1.13217

5.1. ΠΕΙΡΑΜΑΤΑ ΕΝΤΟΠΙΣΜΟΥ ΘΕΣΗΣ

Πίνακας 5.18: Σφάλμα θέσης (σε μέτρα) για κίνηση σε σπιράλ με υψηλή ταχύτητα σε αποθήκη

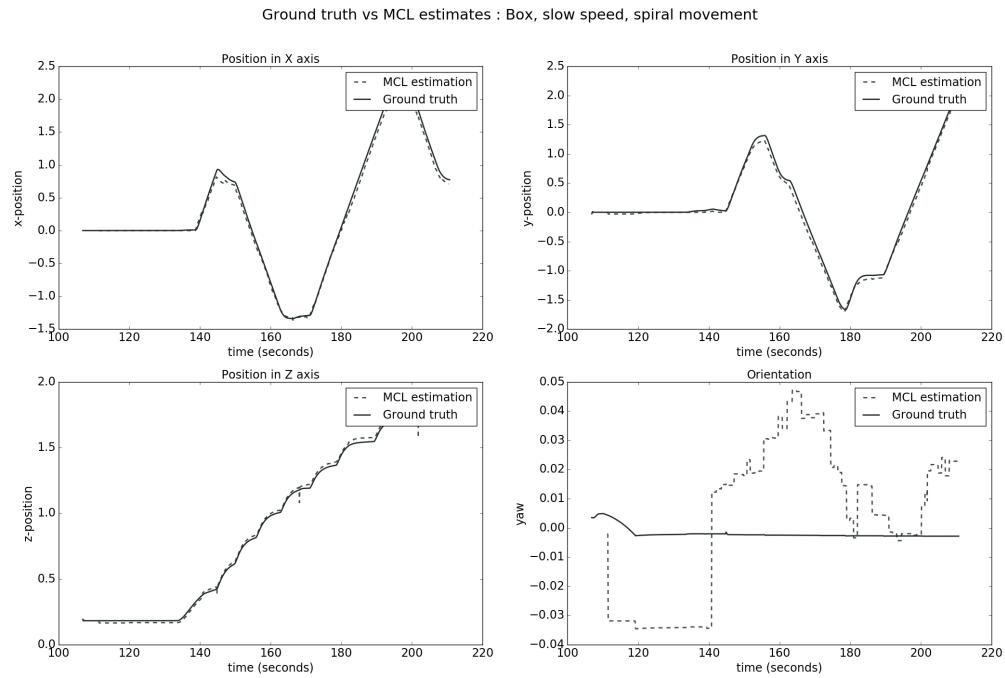
#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.14558	0.14035	0.00234	0.37128	23.15082	0.10030	0.17675
2	0.16669	0.16334	0.00234	0.53938	44.57385	0.13661	0.21547
3	0.31894	0.30145	0.00691	0.76302	99.71369	0.19338	0.37292
4	0.17109	0.16619	0.01282	0.39827	37.36532	0.08499	0.19102
5	0.22178	0.20365	0.01477	0.57958	64.42152	0.12949	0.25678
Avg.	0.20482	0.19500	0.00784	0.53031	53.84504	0.12895	0.24259

Πίνακας 5.19: Σφάλμα προσανατολισμού (σε rad) για κίνηση σε σπιράλ με υψηλή ταχύτητα σε αποθήκη

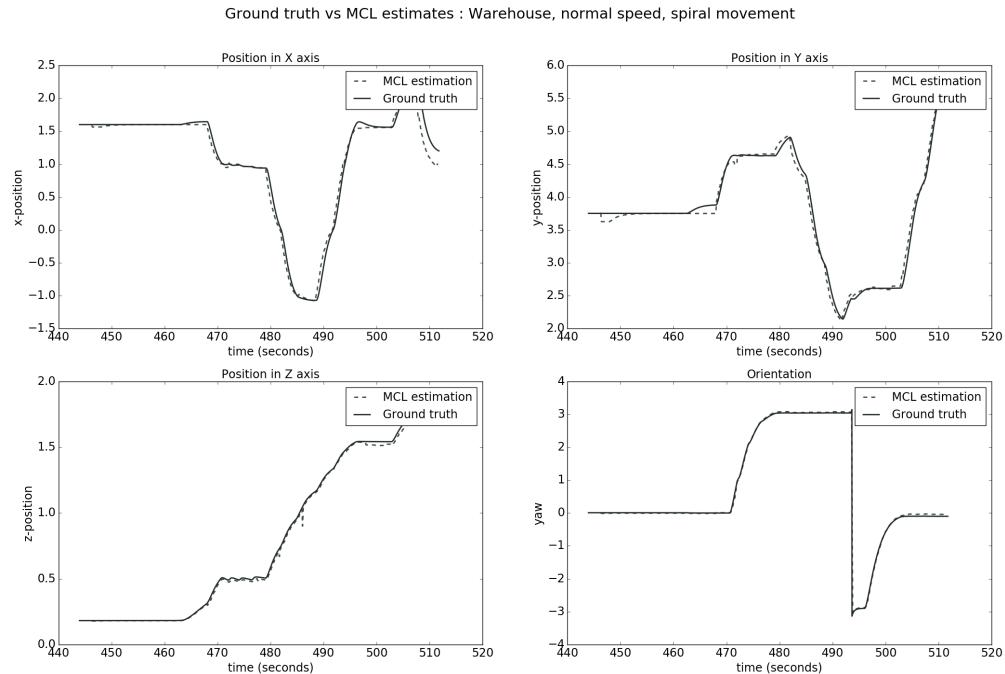
#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.01975	0.01055	0.00012	0.12255	0.63240	0.02153	0.02921
2	1.79168	0.10750	0.00105	6.23073	10366.41819	2.75611	3.28608
3	0.02055	0.01369	0.00013	0.13945	0.49767	0.01649	0.02634
4	0.13325	0.04807	0.00000	6.20004	437.62583	0.64032	0.65373
5	0.05985	0.04065	0.00027	6.20379	106.40434	0.32470	0.33001
Avg.	0.40501	0.04409	0.00031	3.77931	2182.31569	0.75183	0.86507

Όπως φαίνεται στην αναπαράσταση της πορείας που ακολούθησε το ρομπότ, πρόκειται για μία αρκετά σταθερή κίνηση παρόλο που υπάρχει συνεχή μεταβολή της κίνησης σε κάθε άξονα. Πιο συγκεκριμένα, η κίνηση με κανονική ταχύτητα επιφέρει το χαμηλότερο σφάλμα από όλες τις περιπτώσεις. Η κίνηση με αργή ταχύτητα επίσης οδηγεί σε χαμηλό σφάλμα, εκτός από μία μόνο περίπτωση. Επίσης, παρατηρείται ότι το σφάλμα προσανατολισμού λαμβάνει υψηλή μέγιστη τιμή παρά τη σχετικά μικρή μέση τιμή του. Αυτό σημαίνει ότι στιγμιαία μπορεί η εκτίμηση να είναι αρκετά λανθασμένη, όμως το συνολικό σύστημα εντοπισμού θέσης δεν επηρεάζεται.

ΚΕΦΑΛΑΙΟ 5. ΠΕΙΡΑΜΑΤΑ - ΑΠΟΤΕΛΕΣΜΑΤΑ



Σχήμα 5.5: Οπτικοποίηση καλύτερης πορείας για κίνηση σε σπιράλ πορεία σε κενό χώρο

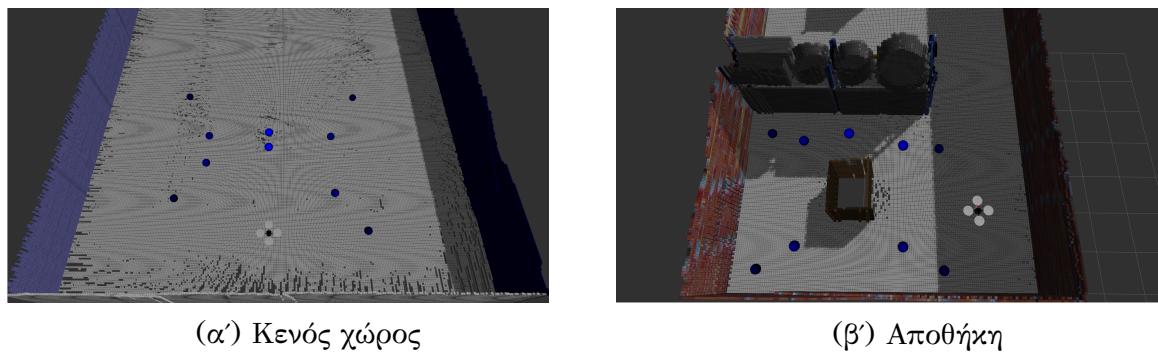


Σχήμα 5.6: Οπτικοποίηση καλύτερης πορείας για κίνηση σε σπιράλ πορεία σε αποθήκη

5.1.3 Κίνηση σε μαίανδρο

Η τελευταία περίπτωση που εξετάστηκε είναι η κίνηση σε πορεία σχήματος μαίανδρου. Το drone καλείται να ακολουθήσει ένα σύνολο σημείων το οποίο φαίνεται στο [σχήμα 5.7](#) για κάθε περιβάλλον. Η σειρά διάσχισης είναι από τα χαμηλά και εξωτερικά σημεία προς αυτά που βρίσκονται σε μεγαλύτερο ύψος και στο εσωτερικό του μονοπατιού. Επίσης, στην κίνηση αυτή ο προσανατολισμούς του ρομπότ αλλάζει ανάλογα με την περίπτωση, στοχεύοντας κάθε φορά σε επόμενο σημείο.

Στη συνέχεια, παρουσιάζονται τα αποτελέσματα της σύγκρισης ανάμεσα στην εκτιμώμενη και στην πραγματική θέση, καθώς και στο σφάλμα προσανατολισμού για κάθε μία ξεχωριστή περίπτωση. Η διαφορά με τις προηγούμενες κινήσεις είναι ότι η κατεύθυνση του drone μεταβάλλεται συχνά.



Σχήμα 5.7: Κίνηση σε μαίανδρο

Πίνακας 5.20: Σφάλμα θέσης (σε μέτρα) για κίνηση σε μαίανδρο με χαμηλή ταχύτητα σε κενό χώρο

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.13941	0.13917	0.01053	0.40395	88.71098	0.07046	0.15620
2	0.17959	0.18069	0.00112	0.31979	120.20220	0.06296	0.19031
3	0.25041	0.20634	0.00875	0.51618	319.34917	0.16220	0.29834
4	0.12573	0.09644	0.01047	0.36000	78.24017	0.08151	0.14984
5	0.15761	0.14391	0.00233	0.33064	115.69076	0.07992	0.17671
Avg.	0.17055	0.15331	0.00664	0.38611	144.43866	0.09141	0.19428

Πίνακας 5.21: Σφάλμα προσανατολισμού (σε rad) για κίνηση σε μαίανδρο με χαμηλή ταχύτητα σε κενό χώρο

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.04816	0.02652	0.00007	6.23700	312.04984	0.28901	0.29295
2	0.03423	0.02254	0.00051	6.20982	123.86660	0.19016	0.19319
3	0.59045	0.02264	0.00002	6.24419	12486.34445	1.76982	1.86548
4	0.03180	0.02515	0.00025	6.25226	120.58698	0.18330	0.18602
5	0.03221	0.01813	0.00002	6.24227	159.83427	0.20522	0.20770
Avg.	0.14737	0.02300	0.00017	6.23711	2640.53643	0.52750	0.54907

ΚΕΦΑΛΑΙΟ 5. ΠΕΙΡΑΜΑΤΑ - ΑΠΟΤΕΛΕΣΜΑΤΑ

Πίνακας 5.22: Σφάλμα θέσης (σε μέτρα) για κίνηση σε μαίανδρο με κανονική ταχύτητα σε κενό χώρο

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.13383	0.13209	0.01411	0.37084	49.13156	0.06705	0.14968
2	0.16226	0.16357	0.00549	0.42715	70.75790	0.09190	0.18647
3	0.17202	0.17970	0.00107	0.40352	76.20777	0.09507	0.19653
4	0.24672	0.22460	0.00168	0.62291	156.15264	0.15479	0.29124
5	0.20597	0.19350	0.00964	0.61865	112.45581	0.11872	0.23772
Avg.	0.18416	0.17869	0.00640	0.48862	92.94114	0.10551	0.21233

Πίνακας 5.23: Σφάλμα προσανατολισμού (σε rad) για κίνηση σε μαίανδρο με κανονική ταχύτητα σε κενό χώρο

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.04689	0.02533	0.00000	6.21171	190.95459	0.29140	0.29508
2	0.03398	0.01843	0.00002	6.22261	115.54487	0.23591	0.23828
3	0.04552	0.02767	0.00002	6.19883	154.07015	0.27578	0.27944
4	0.03950	0.02718	0.00000	6.23749	156.12471	0.28860	0.29121
5	0.03897	0.02222	0.00007	6.23997	101.45287	0.22246	0.22579
Avg.	0.04097	0.02417	0.00002	6.22212	143.62944	0.26283	0.26596

Πίνακας 5.24: Σφάλμα θέσης (σε μέτρα) για κίνηση σε μαίανδρο με υψηλή ταχύτητα σε κενό χώρο

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.41997	0.47342	0.01095	0.97306	378.97266	0.21126	0.47008
2	0.24618	0.26350	0.00640	0.64563	132.31387	0.15444	0.29058
3	0.51469	0.59397	0.00974	1.07424	575.06789	0.25805	0.57572
4	0.21217	0.21110	0.00669	0.58169	100.03961	0.14052	0.25446
5	0.24979	0.22645	0.01368	0.72454	153.89782	0.16209	0.29774
Avg.	0.32856	0.35369	0.00949	0.79983	268.05837	0.18527	0.37772

Πίνακας 5.25: Σφάλμα προσανατολισμού (σε rad) για κίνηση σε μαίανδρο με υψηλή ταχύτητα σε κενό χώρο

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.13294	0.02812	0.00016	6.21965	739.34580	0.64317	0.65659
2	0.52434	0.05197	0.00013	6.26766	4538.56307	1.61959	1.70186
3	0.03920	0.02077	0.00007	6.23783	86.16475	0.21944	0.22285
4	0.37290	0.03904	0.00093	6.25117	3154.05689	1.37972	1.42880
5	0.07434	0.03889	0.00009	6.23467	302.51878	0.41089	0.41745
Avg.	0.22874	0.03576	0.00027	6.24219	1764.12986	0.85456	0.88551

5.1. ΠΕΙΡΑΜΑΤΑ ΕΝΤΟΠΙΣΜΟΥ ΘΕΣΗΣ

Πίνακας 5.26: Σφάλμα θέσης (σε μέτρα) για κίνηση σε μαίανδρο με χαμηλή ταχύτητα σε αποθήκη

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.16461	0.15817	0.00502	0.30393	98.66991	0.05284	0.17289
2	0.12103	0.12106	0.00141	0.25082	55.44583	0.04189	0.12807
3	0.07707	0.07404	0.00402	0.19366	25.75932	0.04161	0.08758
4	0.20393	0.17800	0.00716	0.46433	162.21094	0.08732	0.22184
5	0.24893	0.25953	0.00073	0.55382	277.37072	0.14133	0.28625
Avg.	0.16312	0.15816	0.00367	0.35331	123.89134	0.07300	0.17933

Πίνακας 5.27: Σφάλμα προσανατολισμού (σε rad) για κίνηση σε μαίανδρο με χαμηλή ταχύτητα σε αποθήκη

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.03500	0.02523	0.00001	4.66208	62.37786	0.13295	0.13746
2	0.04165	0.02371	0.00002	4.72697	108.26666	0.17408	0.17897
3	0.02163	0.01688	0.00004	4.28138	23.19017	0.08024	0.08310
4	0.03641	0.02727	0.00003	3.62527	19.71341	0.06823	0.07733
5	0.04105	0.04252	0.00001	3.66151	20.75406	0.06668	0.07830
Avg.	0.03515	0.02712	0.00002	4.19144	46.86043	0.10444	0.11103

Πίνακας 5.28: Σφάλμα θέσης (σε μέτρα) για κίνηση σε μαίανδρο με κανονική ταχύτητα σε αποθήκη

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	1.07495	0.82528	0.00200	2.27038	4097.95301	0.67114	1.26719
2	0.50567	0.46140	0.01199	1.67076	742.84486	0.24718	0.56283
3	0.29637	0.25446	0.00681	1.53723	255.81494	0.20987	0.36313
4	0.09909	0.09806	0.00625	0.20110	17.51989	0.04845	0.11030
5	0.32056	0.29371	0.00209	1.35330	281.32884	0.19527	0.37533
Avg.	0.27797	0.25026	0.00762	1.05732	270.44227	0.16022	0.32147

Πίνακας 5.29: Σφάλμα προσανατολισμού (σε rad) για κίνηση σε μαίανδρο με κανονική ταχύτητα σε αποθήκη

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.17421	0.04715	0.00027	6.21773	814.83564	0.73556	0.75565
2	0.05018	0.03114	0.00007	6.22709	154.26887	0.25158	0.25649
3	0.04360	0.02087	0.00001	6.22664	141.97265	0.26705	0.27052
4	0.31922	0.01610	0.00002	6.23409	2608.81174	1.30804	1.34598
5	0.46441	0.08796	0.00041	6.20350	4753.61433	1.47166	1.54285
Avg.	0.21032	0.04064	0.00016	6.22181	1694.70065	0.80678	0.83430

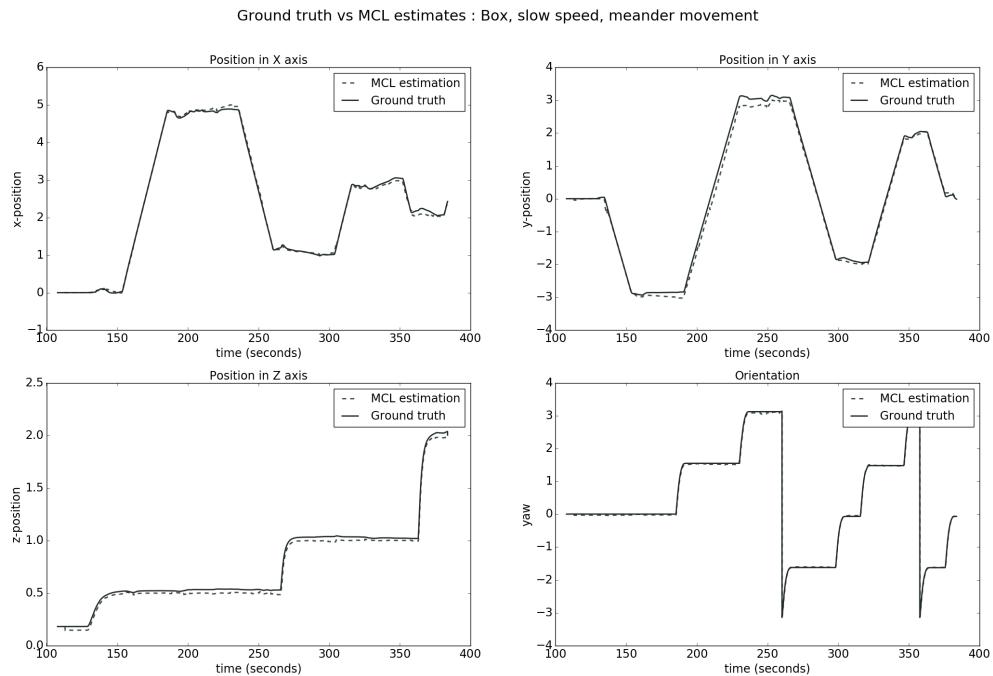
ΚΕΦΑΛΑΙΟ 5. ΠΕΙΡΑΜΑΤΑ - ΑΠΟΤΕΛΕΣΜΑΤΑ

Πίνακας 5.30: Σφάλμα θέσης (σε μέτρα) για κίνηση σε μαίανδρο με υψηλή ταχύτητα σε αποθήκη

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.31894	0.28612	0.01336	1.40791	205.45110	0.15796	0.35590
2	0.38288	0.28412	0.00728	1.47429	383.08797	0.28306	0.47610
3	0.25022	0.20078	0.01576	1.45133	166.62539	0.19783	0.31894
4	0.42412	0.34258	0.00868	1.40271	428.05702	0.26686	0.50105
5	0.36361	0.29345	0.00946	1.76602	336.18949	0.27287	0.45456
Avg.	0.34795	0.28141	0.01091	1.50045	303.88219	0.23572	0.42131

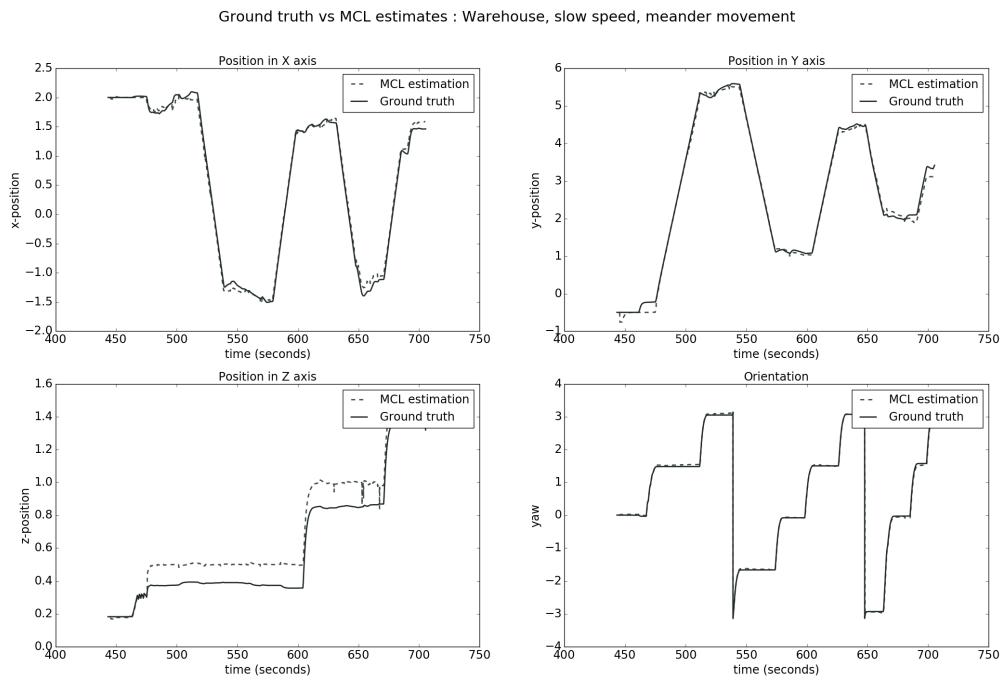
Πίνακας 5.31: Σφάλμα προσανατολισμού (σε rad) για κίνηση σε μαίανδρο με υψηλή ταχύτητα σε αποθήκη

#	Mean	Median	Min	Max	SSE	STD	RMSE
1	0.58158	0.07151	0.00175	6.23028	5223.65858	1.69825	1.79458
2	0.03930	0.02466	0.00002	6.23386	76.62195	0.20933	0.21293
3	0.03897	0.02281	0.00007	6.23772	78.94221	0.21611	0.21953
4	0.10915	0.09875	0.00012	6.23123	137.97221	0.26277	0.28447
5	0.03319	0.02453	0.00002	5.71171	44.32420	0.16173	0.16505
Avg.	0.16044	0.04845	0.00040	6.12896	1112.30383	0.50964	0.53531



Σχήμα 5.8: Οπτικοποίηση καλύτερης πορείας για κίνηση σε πορεία μαιάνδρου σε κενό χώρο

5.1. ΠΕΙΡΑΜΑΤΑ ΕΝΤΟΠΙΣΜΟΥ ΘΕΣΗΣ

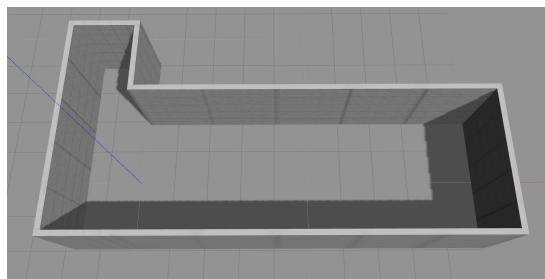


Σχήμα 5.9: Οπτικοποίηση καλύτερης πορείας για κίνηση σε πορεία μαιάνδρου σε αποθήκη

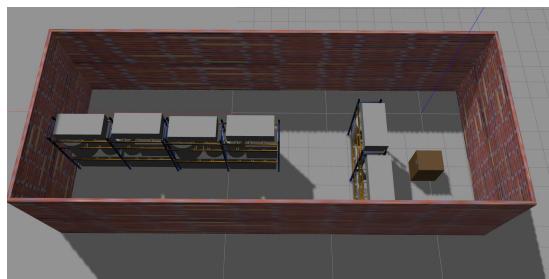
Η περίπτωση αυτή παρουσιάζει το μεγαλύτερο σφάλμα από όλες τις προηγούμενες κινήσεις που εξετάστηκαν. Και πάλι, όμως, η χαμηλή ταχύτητα οδηγεί στο χαμηλότερο σφάλμα θέσης και προσανατολισμού. Αξίζει να σημειωθεί η μεγάλη διασπορά του σφάλματος κατά την κίνηση σε υψηλή ταχύτητα, το οποίο σημαίνει ότι πρόκειται για μια ασταθή κίνηση.

5.2 ΠΕΙΡΑΜΑΤΑ ΚΑΛΥΨΗΣ ΧΩΡΟΥ

Σκοπός των πειραμάτων κάλυψης χώρου είναι ο υπολογισμός του ποσοστού κάλυψης του χώρου που πραγματοποιείται από το ρομπότ, καθώς και ο χρόνος που απαιτείται για τη διαδικασία αυτή. Θα χρησιμοποιηθούν δύο διαφορετικά περιβάλλοντα, τα οποία έχουν ήδη αναφερθεί προηγουμένως, ο διάδρομος και η αποθήκη, όπου φαίνονται στο [σχήμα 5.10](#) μέσα από τον προσομοιωτή Gazebo. Οι διαστάσεις του πρώτου χώρου είναι $11.65m \times 2.8m \times 2.35m$, ενώ του δεύτερου είναι $19.75m \times 7.85m \times 2.4m$.



(α') Διάδρομος



(β') Αποθήκη

Σχήμα 5.10: Περιβάλλοντα που χρησιμοποιήθηκαν στα πειράματα κάλυψης χώρου

Χρησιμοποιήθηκαν δύο διαφορετικές παραμετροποιήσεις του αισθητήρα RFID, μέσω του οποίου πρέπει να εντοπιστούν όσο το δυνατόν περισσότερα αντικείμενα στον χώρο. Οι τιμές για το ευρύ και στενό οπτικό πεδίο φαίνονται στον [πίνακα 5.32](#).

Πίνακας 5.32: Παράμετροι του αισθητήρα RFID

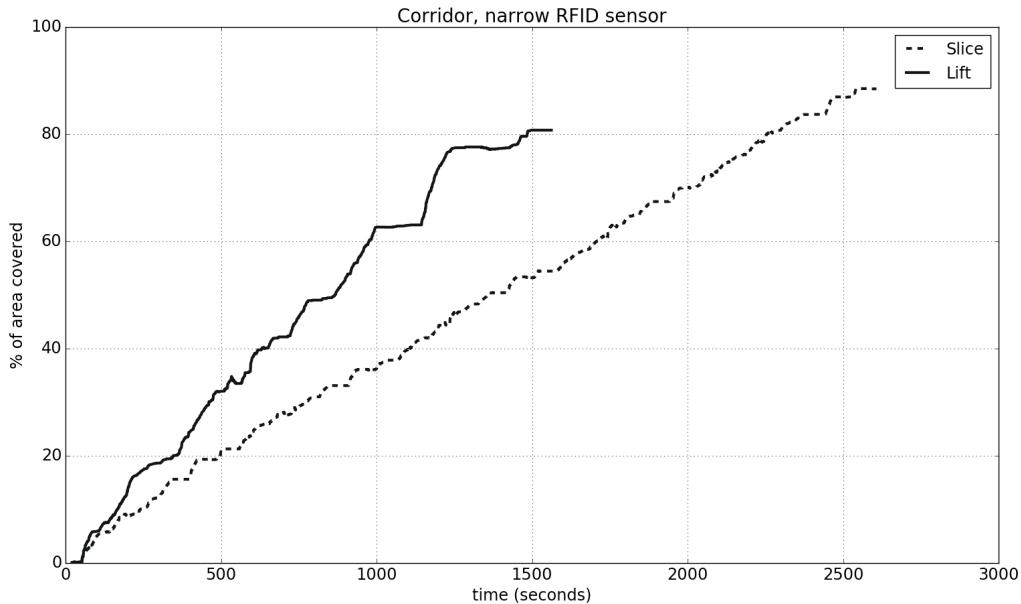
Τύπος Στενός	Οριζόντιο Πεδίο Όρασης 45°	Κάθετο Πεδίο Όρασης 25°	Εύρος
Ευρύς	80°	40°	2.5 m

Για την εύρεση των σημείων τα οποία πρέπει να διασχίσει το ρομπότ για την κάλυψη του χώρου, χρησιμοποιήθηκαν διαφορετικά βήματα δειγματοληψίας, ανάλογα με τον χώρο αλλά και τον αισθητήρα που χρησιμοποιείται σε κάθε περίπτωση. Επίσης, εξετάστηκαν τα διαφορετικά αποτελέσματα ανάλογα με το εάν τα σημεία στον τρισδιάστατο χώρο ενώνονται οριζόντια (slice) ή κάθετα (lift), σύμφωνα με τις μεθόδους που περιγράφονται στο [υποκεφάλαιο 4.2](#). Επίσης, συγχρίνεται η εκτίμηση της κάλυψης του χώρου κατά τη διάρκεια της δημιουργίας του μονοπατιού με την πραγματική κάλυψη που επιτυγχάνεται κατά την κίνηση του ρομπότ στον χώρο. Είναι σημαντικό να επισημάνουμε ότι το σύστημα εντοπισμού θέσης θεωρείται αναγκαίο για να πραγματοποιηθεί μιας ασφαλής και ορθή πλοιήγηση του ρομπότ στο χώρο. Η ταχύτητα με την οποία κινήθηκε το drone σε όλες τις περιπτώσεις είναι η χαμηλή, καθώς όπως φαίνεται στο [υποκεφάλαιο 5.1](#) οδηγεί στο μικρότερο σφάλμα σε όλες τις κινήσεις.

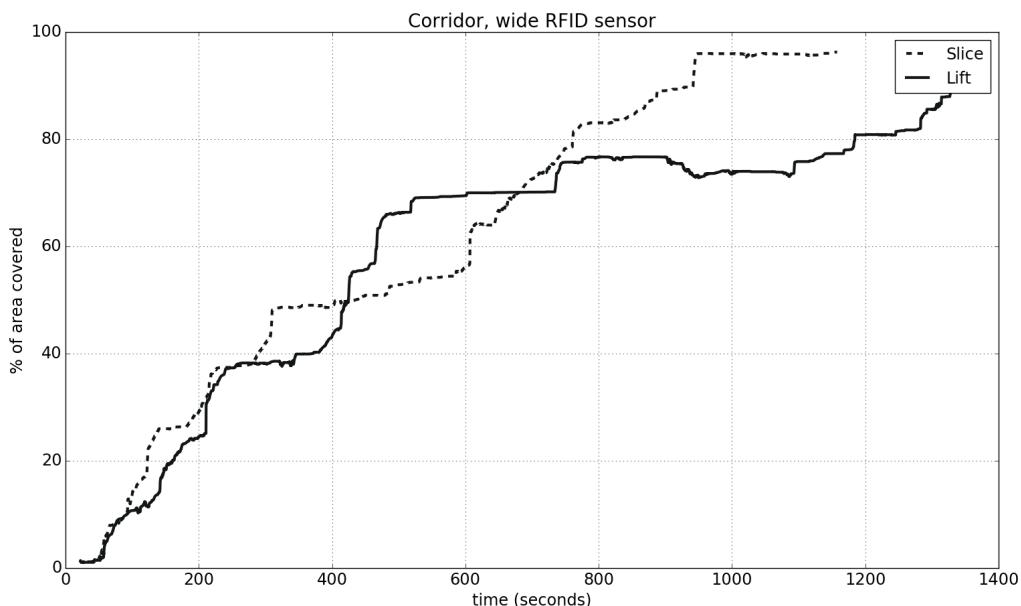
Αρχικά, θα εξεταστούν όλες οι περιπτώσεις για το περιβάλλον του διαδρόμου. Στα διαγράμματα φαίνεται το ποσοστό κάλυψης του χώρου συγκριτικά ως προς

5.2. ΠΕΙΡΑΜΑΤΑ ΚΑΛΥΨΗΣ ΧΩΡΟΥ

τον τρόπο ένωσης των σημείων για ευρύ και στενό οπτικό πεδίο, ενώ στον παρακάτω πίνακα εμφανίζονται συγκεντρωτικά τα αποτελέσματα για κάθε ξεχωριστή περίπτωση.



Σχήμα 5.11: Ποσοστό κάλυψης συναρτήσει του χρόνου για το περιβάλλον του διαδρόμου με στενό τύπο αισθητήρα



Σχήμα 5.12: Ποσοστό κάλυψης συναρτήσει του χρόνου για το περιβάλλον του διαδρόμου με ευρύ τύπο αισθητήρα

ΚΕΦΑΛΑΙΟ 5. ΠΕΙΡΑΜΑΤΑ - ΑΠΟΤΕΛΕΣΜΑΤΑ

Πίνακας 5.33: Αξιολόγηση κάλυψης χώρου για το περιβάλλον του διάδρομου

Τύπος	Ένωση σημείων	Βήμα δειγματοληφίας	Χρόνος κάλυψης	Εκτίμηση κάλυψης	Πραγματική κάλυψη
Στενός	Κάθετη	0.6 m	25.69 min	72.27%	80.76%
	Οριζόντια		43.06 min		88.46%
Ευρύς	Κάθετη	0.6 m	21.70 min	93.30%	88.69%
	Οριζόντια		18.93 min		96.26%

Παρατηρούμε ότι η χρήση αισθητήρα με ευρύ πεδίο όρασης αποδίδει πολύ καλύτερα συνολικά αποτελέσματα σε συντομότερο χρόνο, όπως ήταν αναμενόμενο. Όταν διαθέτουμε στενό οπτικό πεδίο, το ποσοστό κάλυψης αυξάνεται σχεδόν γραμμικά με το χρόνο, ενώ στην αντίθετη περίπτωση η καμπύλη κάλυψης θυμίζει την λογαριθμική.

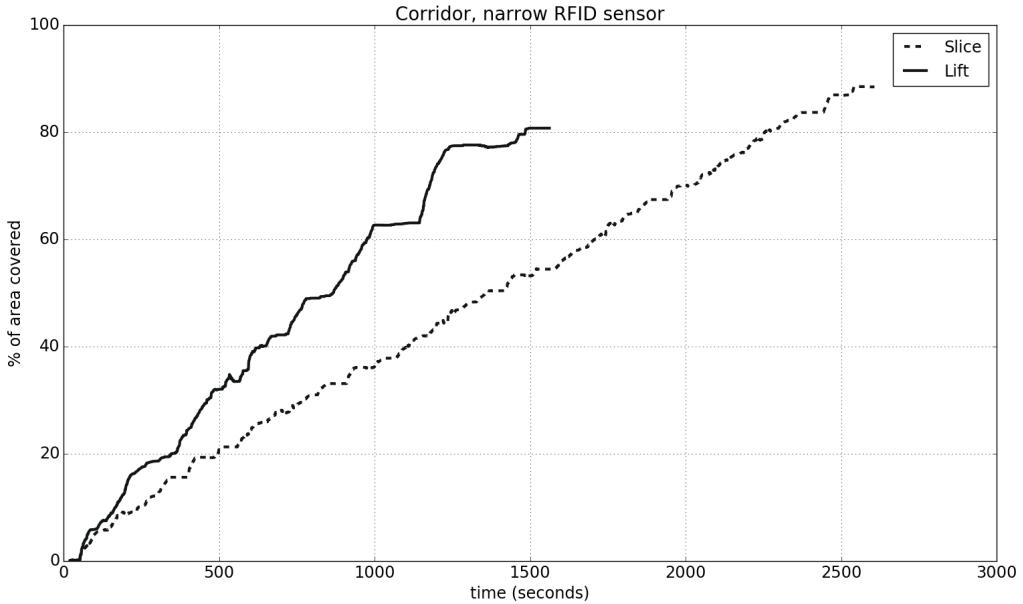
Αυτό όμως που αξίζει να σημειωθεί είναι η διαφορά που υπάρχει στον τρόπο ένωσης των σημείων ανάλογα με τον τύπο του αισθητήρα. Πιο συγκεκριμένα, για αισθητήρα με στενό FOV η κάθετη ένωση των σημείων οδηγεί σε μικρότερο ποσοστό κάλυψης, χρειάζεται όμως λιγότερο χρόνο για να ολοκληρωθεί. Αντίθετα, στην περίπτωση του ευρυγώνιου αισθητήρα, η οριζόντια ένωση των σημείων οδηγεί σε μεγαλύτερη κάλυψη αλλά και πιο γρήγορη επίτευξη του στόχου. Η διαφορά που παρατηρείται ανάμεσα στην εκτίμηση της κάλυψης και την πραγματική, οφείλεται στην κίνηση του drone μέσα στον χώρο που οδηγεί είτε σε σημεία τα οποία δεν είχαν υπολογιστεί εκ των προτέρων, είτε σε αστοχία ορισμένων αρχικών στόχων.

Παρακάτω παρουσιάζονται τα αντίστοιχα αποτελέσματα για το περιβάλλον της αποθήκης.

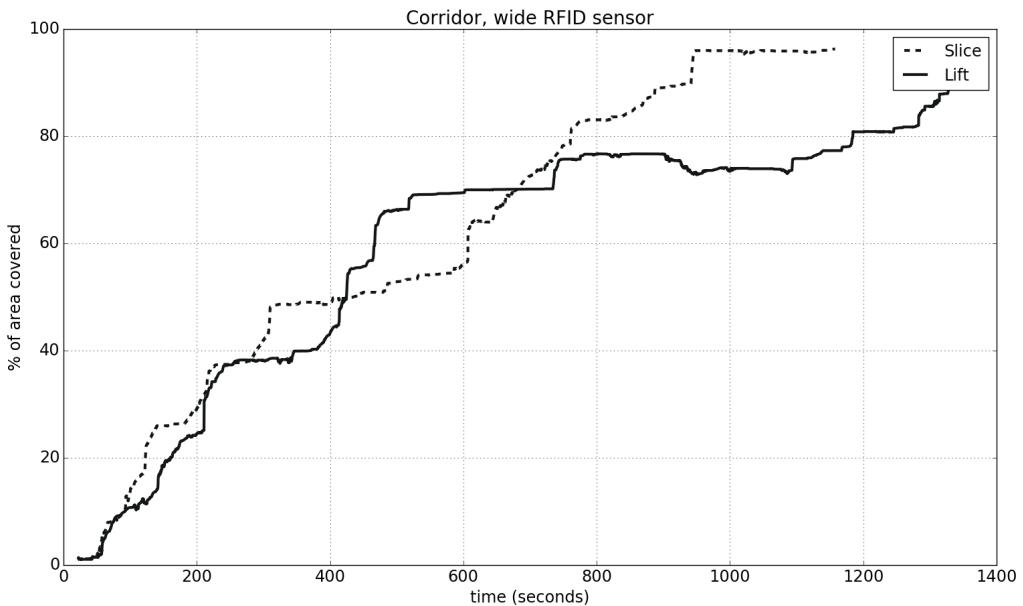
Πίνακας 5.34: Αξιολόγηση κάλυψης χώρου για το περιβάλλον της αποθήκης

Τύπος	Ένωση σημείων	Βήμα δειγματοληφίας	Χρόνος κάλυψης	Εκτίμηση Κάλυψης	Πραγματική Κάλυψη
Στενός	Κάθετη	0.8 m	40.40 min	50.18%	65.43%
	Οριζόντια		76.52 min		72.49%
Ευρύς	Κάθετη	0.9 m	37.18 min	62.33%	84.12%
	Οριζόντια		46.59 min		84.88%

Στην περίπτωση αυτή χρησιμοποιήθηκε μεγαλύτερο βήμα δειγματοληφίας, καθώς πρόκειται για έναν μεγαλύτερο σε έκταση χώρο. Παρατηρούμε πάλι την ίδια συμπεριφορά με πριν στις καμπύλες κάλυψης και των δύο τύπων αισθητήρα. Ανεξαρτήτως στενού ή ευρύ οπτικού πεδίου, η ένωση των σημείων με κάθετο τρόπο οδηγεί σε συντομότερη ολοκλήρωση της διαδικασίας κάλυψης. Όμως, το τελικό ποσοστό είναι μεγαλύτερο όταν χρησιμοποιείται η οριζόντια ένωση των σημείων. Πιο αναλυτικά, όταν διαθέτουμε στενό FOV η χρονική διαφορά είναι μεγάλη και αυτό οφείλεται σε όλα τα διαφορετικά ύψη τα οποία πρέπει να βρεθεί το ρομπότ, ώστε να καλύψει ολόκληρο το χώρο. Σε όλες τις περιπτώσεις, είναι χαρακτηριστική η



Σχήμα 5.13: Ποσοστό κάλυψης συναρτήσει του χρόνου για το περιβάλλον της αποθήκης με στενό τύπο αισθητήρα



Σχήμα 5.14: Ποσοστό κάλυψης συναρτήσει του χρόνου για το περιβάλλον της αποθήκης με ευρύ τύπο αισθητήρα

διαφορά της κάλυψης από την εκτίμηση κάλυψης, κάτι που δημιουργείται λόγω της μεγαλύτερης σε διάρκεια κίνησης του ρομπότ στον χώρο.

Στην αποθήκη, καθώς πρόκειται για ένα πιο περίπλοκο περιβάλλον, το συνολικό ποσοστό κάλυψης είναι μικρότερο από ότι θα ήταν το επιθυμητό για την πλήρη

ΚΕΦΑΛΑΙΟ 5. ΠΕΙΡΑΜΑΤΑ - ΑΠΟΤΕΛΕΣΜΑΤΑ

κάλυψη. Αυτό όμως οφείλεται και στις ιδιαιτερότητες του χάρτη, με την ύπαρξη σημείων που είναι αδύνατον να ανιχνευτούν από το ρομπότ και δεν αφαιρέθηκαν με την επεξεργασία των σημείων κατά τον υπολογισμό του όγκου που περιγράφηκε προηγουμένως.

6

Συμπεράσματα

Στο κεφάλαιο αυτό παρουσιάζονται τα συμπεράσματα που προέκυψαν από την έκβαση των πειραμάτων τόσο του συστήματος εντοπισμού θέσης, όσο και της κάλυψης χώρου. Για την αξιολόγηση των αποτελεσμάτων χρησιμοποιούμε τις τιμές του σφάλματος στην εκτίμηση θέσης και τον χρόνο και το ποσοστό κάλυψης για την πλήρη κάλυψη. Στη συνέχεια, αναφέρονται προβλήματα που παρουσιάστηκαν κατά τη διάρκεια των υλοποιήσεων και των πειραμάτων.

6.1 ΓΕΝΙΚΑ ΣΥΜΠΕΡΑΣΜΑΤΑ

Ο εντοπισμός της θέσης ενός drone σε κλειστό χώρο είναι ένα από τα βασικά προβλήματα που απασχολούν τις εφαρμογές των μη επανδρωμένων οχημάτων. Γνωρίζοντας την θέση του, το ρομπότ μπορεί να πραγματοποιήσει πληθώρα εφαρμογών που απαιτούν την ασφαλή και αξιόπιστη πλοήγηση στο χώρο. Στην παρούσα διπλωματική εργασία αναπτύχθηκε ένα σύστημα εντοπισμού θέσης που βασίζεται σε φίλτρο σωματιδίων και έχει ως είσοδο αποκλειστικά τις μετρήσεις του αισθητήρα απόστασης και ενός IMU. Επιπλέον, επιλύθηκε το πρόβλημα της πλήρους κάλυψης χώρου σε γνωστό τρισδιάστατο χάρτη μορφής OctoMap με την επιλογή των σημείων κάλυψης, αλλά και την ένωση τους με όσο το δυνατόν πιο βέλτιστο τρόπο.

Με βάση τα αποτελέσματα των πειραμάτων που αγαλύθηκαν στο [κεφάλαιο 5](#) και αφορούν το σφάλμα της εκτίμησης θέσης από την πραγματική, παρατηρήθηκαν τα εξής:

- Το σύστημα εντοπισμού θέσης εξαρτάται άμεσα από τον τρόπο κίνησης του drone, από την ταχύτητα του, καθώς και από την πορεία που ακολουθεί.
- Το μικρότερο σφάλμα στην εκτίμηση θέσης εμφανίζεται όταν το drone κινείται με χαμηλή ταχύτητα. Όσο χαμηλότερη, τόσο πιο σταθερή και ομαλή είναι η

ΚΕΦΑΛΑΙΟ 6. ΣΥΜΠΕΡΑΣΜΑΤΑ

κίνησή του. Παρόλα αυτά, ακόμη και για υψηλότερες ταχύτητες το σφάλμα παραμένει σε σχετικά αποδεκτές τιμές.

- Στην περίπτωση όπου το drone διατηρεί σταθερό τον προσανατολισμό του, όπως συνέβαινε στην περίπτωση της κίνησης σε σπιράλ, επιτρέπεται η κίνηση σε υψηλότερες ταχύτητες, καθώς το σφάλμα παραμένει στα ίδια επίπεδα.
- Στην πιο ρεαλιστική περίπτωση, όπου το drone περιστρέφεται κατά την κίνηση του, η κίνηση πρέπει να είναι αργή και διατηρώντας όσο τον δυνατόν μεγαλύτερη απόσταση ασφαλείας από τα εμπόδια στο χώρο. Κάποιο στιγμιαίο σφάλμα στην εκτίμηση της θέσης μπορεί να οδηγήσει σε λανθασμένη πλοήγηση και συνεπώς σε σύγκρουση.
- Κατά την πλήρη κάλυψη χώρου, προτιμάται η χρήση ενός αισθητήρα με ευρύ οριζόντιο και κάθετο πεδίο όρασης, καθώς οδηγεί σε πιο γρήγορη και αποτελεσματικότερη κάλυψη του χώρου.
- Στην περίπτωση όπου ο χρόνος κάλυψης ενός χώρου έχει μεγαλύτερη σημασία είναι προτιμότερη η κάθετη ένωση των σημείων στο χώρο.
- Στην περίπτωση όπου το ποσοστό κάλυψης του χώρου έχει μεγαλύτερη σημασία, ανεξαρτήτως διάρκειας, η οριζόντια ένωση οδηγεί σε καλύτερα αποτελέσματα.

6.2 ΠΡΟΒΛΗΜΑΤΑ

Ένα βασικό πρόβλημα που παρουσιάστηκε κατά την υλοποίηση της διπλωματικής εργασίας ήταν η ορθή αναπαράσταση του χώρου σε μορφή OctoMap. Ο χάρτης λήφθηκε με χρήση του πακέτου SLAM RTAB-Map [22] και ενός TurtleBot στο Gazebo¹⁷. Η ορατότητα αυτού του επίγειου ρομπότ φτάνει μέχρι ένα συγκεκριμένο ύψος. Σημεία του χώρου τα οποία βρίσκονται σε μεγαλύτερο ύψος και είναι απαραίτητα για την πλοήγηση του drone, δεν είναι δυνατόν να συμπεριληφθούν στον χάρτη. Επίσης, η ύπαρξη θορύβου στην αναπαράσταση του χώρου αποτέλεσε πρόβλημα στην σωστή εκτίμηση της όγκου του χώρου, γεγονός που δυσκόλευε την διαδικασία αξιολόγησης της πλήρους κάλυψης. Μία λύση για το πρώτο πρόβλημα θα ήταν η χρήση ενός πακέτου που πραγματοποιεί SLAM με την χρήση ενός drone και θα παράγει εξίσου καλά αποτελέσματα. Για το δεύτερο, μια πρόταση είναι η χρήση μεθόδων για εξαγωγή χαρτών από τρισδιάστατα μοντέλα.

Ένα ακόμη πρόβλημα είναι η ταχύτητα επεξεργασίας της πληροφορίας που δέχεται το drone. Παρόλο που η υλοποίηση δεν έχει υψηλές υπολογιστικές απαιτήσεις, η γρήγορη λήψη πληροφοριών οδηγεί στην αδυναμία χειρισμού της. Το γεγονός αυτό περιορίζει σημαντικά την ταχύτητα που μπορεί να κινηθεί το ρομπότ στο χώρο και να διατηρεί την ορθή εκτίμηση της θέσης του.

¹⁷https://wiki.ros.org/turtlebot_gazebo

7

Μελλοντικές επεκτάσεις

Μια σημαντική βελτίωση της παρούσας εργασίας θα ήταν η υλοποίηση του Adaptive Particle Filter, μιας μεθόδου η οποία προσαρμόζει τον αριθμό των σωματιδίων που χρησιμοποιούνται ανάλογα με το βάρος αυτών. Με τον τρόπο αυτό μπορεί να μειωθεί η υπολογιστική ισχύς που απαιτείται, καθώς όταν υπάρχει μεγάλη βεβαίότητα για την εκτίμηση της θέσης θα υπάρχουν λιγότερα σωματιδια για επεξεργασία. Επιπλέον, θα μπορούσε να αντιμετωπιστεί και η περίπτωση του καθολικού εντοπισμού θέσης, με την αρχικοποίηση περισσότερων σωματιδίων σε ολόκληρο το χώρο και τη μείωση τους αργότερα. Επίσης, για την χρήση του συστήματος εντοπισμού θέσης εκτός περιβάλλοντος προσομοίωσης είναι σημαντικό να χρησιμοποιηθεί οπτική οδομετρία, μέσω της οποίας που ήδη διαθέτει το ρομπότ. Παρόλο που θα αυξηθούν οι υπολογιστικές απαιτήσεις, πιθανόν θα επιφέρει καλύτερα αποτελέσματα. Επίσης, μπορεί να μελετηθεί η χρήση της κάμερας που δεν διαθέτει καμία πληροφορία βάθους, για την υποστήριξη του συστήματος εντοπισμού θέσης. Η αντιστοίχιση των λήψεων με κάποιον τρόπο με τμήματα του χάρτη, μπορεί να προσδόσει σημαντική πληροφορία.

Όσον αφορά το σύστημα πλήρους κάλυψης χώρου, θα μπορούσε να επεκταθεί με την υλοποίηση μιας μεθόδου που θα αποφασίζει σε πραγματικό χρόνο τα σημεία τα οποία θα επισκεφτεί. Με βάση τις περιοχές που έχουν ήδη καλυφθεί και με τους περιορισμούς χρόνου που πιθανόν υπάρχουν, μπορεί να επιλέγει τα σημεία τα οποία θα επιφέρουν καλύτερο συνολικό αποτέλεσμα και όχι απλά να ακολουθεί το αρχικό πλάνο. Επίσης, παρουσιάζει μεγάλο ενδιαφέρον η μελέτη της ύπαρξης μοτίβων ανάλογα με την κάθε περίπτωση και η σημασιολογικός διαχωρισμός του χώρου.

Μια επίσης ενδιαφέρουσα υλοποίηση, θα ήταν η κάλυψη του χώρου από πολλαπλά drone. Ο συγχρονισμός αυτών στο χώρο και η εύρεση ενός τρόπου κάλυψης, ο οποίος θα χρησιμοποιούσε μη επικαλυπτόμενα μονοπάτια μπορεί να οδηγήσει στην πλήρη κάλυψη του χώρου σε μειωμένο χρόνο και πιθανόν με μεγαλύτερη ακρίβεια.

Επιπλέον, είναι σημαντικό να μελετηθούν οι ενεργειακές ανάγκες του drone.

ΚΕΦΑΛΑΙΟ 7. ΜΕΛΛΟΝΤΙΚΕΣ ΕΠΕΚΤΑΣΕΙΣ

Στην συγκεκριμένη εργασία, οι υλοποίησεις δεν λαμβάνουν υπόψη τους τη διάρκεια πτήσης του drone, ούτε την επιρροή της ταχύτητας και του φορτίου στην κατανάλωση ενέργειας. Σε μία ρεαλιστική κατάσταση όμως, ο χρόνος πτήσης είναι περιορισμένος και θα πρέπει να συμπεριλαμβάνεται στον σχεδιασμό μονοπατιού κάλυψης του χώρου.

Βιβλιογραφία

- [1] Maja J Mataric. “*The Robotics Primer*“. MIT Press, 2007. ISBN 9780262633543.
- [2] Lee McCauley. “*AI Armageddon and the Three Laws of Robotics*“. Ethics and Information Technology, 9:153–164, 07 2007.
- [3] Armin Hornung, Kai M. Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. “*OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees*“. Autonomous Robots, 2013. Software available at <http://octomap.github.com>.
- [4] Francisco J Perez-Grau, Fernando Caballero, Antidio Viguria, and Anibal Ollero. “*Multi-sensor three-dimensional Monte Carlo localization for long-term aerial robot navigation*“. International Journal of Advanced Robotic Systems, 14(5):1729881417732757, 2017.
- [5] Dieter Fox, Wolfram Burgard, Frank Dellaert, and Sebastian Thrun. “*Monte Carlo Localization: Efficient Position Estimation for Mobile Robots*“. pages 343–349, 01 1999.
- [6] A. D. Stewart and P. Newman. “*LAPS - localisation using appearance of prior structure: 6-DoF monocular camera localisation using prior pointclouds*“. In “*2012 IEEE International Conference on Robotics and Automation*“, pages 2625–2632, May 2012.
- [7] K. Ok, W. N. Greene, and N. Roy. “*Simultaneous tracking and rendering: Real-time monocular localization for MAVs*“. In “*2016 IEEE International Conference on Robotics and Automation (ICRA)*“, pages 4522–4529, May 2016.
- [8] Marius Beul, Nicola Krombach, Matthias Nieuwenhuisen, David Droeschel, and Sven Behnke. “*Autonomous Navigation in a Warehouse with a Cognitive Micro Aerial Vehicle*“, pages 487–524. Springer International Publishing, Cham, 2017. ISBN 978-3-319-54927-9.
- [9] M. Beul, D. Droeschel, M. Nieuwenhuisen, J. Quenzel, S. Houben, and S. Behnke. “*Fast Autonomous Flight in Warehouses for Inventory Applications*“. IEEE Robotics and Automation Letters, 3(4):3121–3128, Oct 2018. ISSN 2377-3766.
- [10] Zheng Fang, Shichao Yang, Sezal Jain, Geetesh Dubey, Stephan Roth, Silvio Mano Maeta, Stephen T. Nuske, Yuzhang Wu, and Sebastian Scherer. “*Robust Autonomous Flight in Constrained and Visually Degraded Shipboard Environments*“. Journal of Field Robotics, 34(1):25–52, January 2017.

ΒΙΒΛΙΟΓΡΑΦΙΑ

- [11] Enric Galceran and Marc Carreras. “*A survey on coverage path planning for robotics*“. *Robotics and Autonomous Systems*, 61:1258–1276, 12 2013.
- [12] L. H. Nam, L. Huang, X. J. Li, and J. F. Xu. “*An approach for coverage path planning for UAVs*“. In “*2016 IEEE 14th International Workshop on Advanced Motion Control (AMC)*“, pages 411–416, April 2016.
- [13] Andreas Bircher, Mina Samir Kamel, Kostas Alexis, Michael Burri, Philipp Oettershagen, Sammy Omari, Thomas Mantel, and Roland Siegwart. “*Three-dimensional coverage path planning via viewpoint resampling and tour optimization for aerial robots*“. *Autonomous Robots*, 40, 11 2015.
- [14] S. X. Yang and C. Luo. “*A neural network approach to complete coverage path planning*“. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 34(1):718–724, Feb 2004. ISSN 1083-4419.
- [15] Carmelo Di Franco and Giorgio Buttazzo. “*Coverage Path Planning for UAVs Photogrammetry with Energy and Resolution Constraints*“. *Journal of Intelligent & Robotic Systems*, 83(3):445–462, Sep 2016. ISSN 1573-0409.
- [16] Morgan Quigley, Ken Conley, Brian P Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. “*ROS: an open-source Robot Operating System*“. volume 3, 01 2009.
- [17] Johannes Meyer, Alexander Sendobry, Stefan Kohlbrecher, Uwe Klingauf, and Oskar von Stryk. “*Comprehensive Simulation of Quadrotor UAVs using ROS and Gazebo*“. In “*3rd Int. Conf. on Simulation, Modeling and Programming for Autonomous Robots (SIMPAR)*“, page to appear, 2012.
- [18] I. A. Sucan, M. Moll, and L. E. Kavraki. “*The Open Motion Planning Library*“. *IEEE Robotics Automation Magazine*, 19(4):72–82, Dec 2012. ISSN 1070-9932.
- [19] Sertac Karaman and Emilio Frazzoli. “*Sampling-based Algorithms for Optimal Motion Planning*“. CoRR, abs/1105.1186, 2011.
- [20] Steven M. Lavalle. “*Rapidly-Exploring Random Trees: A New Tool for Path Planning*“. 05 1999.
- [21] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. “*Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*“. The MIT Press, 2005. ISBN 0262201623.
- [22] M. Labbé and F. Michaud. “*Appearance-Based Loop Closure Detection for Online Large-Scale and Long-Term Operation*“. *IEEE Transactions on Robotics*, 29(3):734–745, June 2013. ISSN 1552-3098.