Διεθνές Πανεπιστήμιο της Ελλάδος
Τμήμα Μηχανικών Πληροφορικής, Υπολογιστών και Τηλεπικοινωνιών

Διπλωματική Εργασία Πρόγραμμα Μεταπτυχιακών Σπουδών στη Ρομποτική

Ανάπτυξη αλγόριθμου χαρτογράφησης με βάση αισθητήρες απόστασης χαμηλού κόστους.

Ζουρνατζίδης Βελισσάριος

Διεθνές Πανεπιστήμιο της Ελλάδος

Σέρρες, 23-09-2022

Εργασία που υποβλήθηκε στο
Πρόγραμμα Μεταπτυχιακών Σπουδών στη Ρομποτική,
του Διεθνούς Πανεπιστημίου της Ελλάδος,
για τη μερική εκπλήρωση υποχρεώσεων για το Δίπλωμα Ειδίκευσης στη Ρομποτική

Επιβλέπων Καθηγητής: Ιωάννης Καλόμοιρος

Βεβαιώνω ότι είμαι συγγραφέας αυτής της Διπλωματικής Εργασίας και πως κάθε βοήθεια που είχα για την προετοιμασία της, είναι πλήρως αναγνωρισμένη και αναφέρεται στη Διπλωματική Εργασία, με κατάλληλη αναφορά. Επίσης, έχω αναφέρει τις πηγές από τις οποίες έκανα χρήση δεδομένων, εικόνων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες και αναλαμβάνω πλήρως την ευθύνη για τη χρήση των πηγών. Τέλος, βεβαιώνω ότι αυτή η πτυχιακή εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά για τις απαιτήσεις του Μεταπτυχιακού Προγράμματος στη Ρομποτική.

Βελισσάριος Ζουρνατζίδης

Περίληψη

Στην παρούσα διπλωματική εργασία μελετάται μια κατασκευή χαμηλού κόστους, για την εκτέλεση χαρτογράφησης με τη βοήθεια πλέγματος κατάληψης. Ο αλγόριθμος πλέγματος κατάληψης διαιρεί τον χώρο σε ένα πλέγμα κελιών και διατηρεί μια πιθανότητα κατάληψης για το κάθε κελί. Με τη βοήθεια ενός αισθητήρα απόστασης μετράται η απόσταση του ρομπότ από αντικείμενα που καταλαμβάνουν το χώρο. Σε κάθε μέτρηση ανανεώνεται η τιμή της αβεβαιότητα για την κατάληψη κάθε κελιού. Η διάταξη μετρήσεων που κατασκευάστηκε είναι εξοπλισμένη με δύο διαφορετικούς αισθητήρες χαμηλού κόστους, για τη μέτρηση της απόστασης των αντικειμένων. Επίσης, περιλαμβάνει έναν σερβοκινητήρα για την περιστροφή των αισθητηρίων, καθώς και έναν μικροεπεξεργαστή Raspberry Pi για τον έλεγχο της διάταξης και την εκτέλεση του αλγόριθμου του πλέγματος κατάληψης. Οι αισθητήρες που χρησιμοποιήθηκαν είναι τύπου ultrasonic και τύπου laser Time-Of-Flight (ToF). Ο αλγόριθμος ανανεώνει σε κάθε βήμα τον διδιάστατο χάρτη του χώρου, γύρω από τη διάταξη μέτρησης. Προκύπτουν συμπεράσματα για την καταλληλόλητα κάθε τύπου αισθητήρα για τον συγκεκριμένο σκοπό.

Abstract

Robotic mapping is the process that a robot performs to create a representation of the environment known as a map. The process of estimating the position of the robot within this environment is called robotic localization and both of these two processes are performed in parallel in robotic application. This thesis presents a low-cost construction for performing occupancy grid mapping. The occupancy grid algorithm divides the area to be imaged into a grid of cells and maintains a probability of occupancy for each cell. An inverse sensor model is then used to incorporate the uncertainty. The construction is equipped with two different low-cost sensors to detect the objects, ultrasonic and laser Time-Of-Flight (ToF). A servo motor is used to rotate the sensors and a Raspberry Pi microprocessor runs the experiment. A suitable software application creates a probabilistic map of the environment while at the same time it shows a two-dimensional map in a graphic representation.

Κεφάλαιο 1.

Διατύπωση του προβλήματος της εργασίας

1.1 Εισαγωγή

Το πρόβλημα με το οποίο ασχοληθήκαμε σε αυτήν την εργασία είναι αυτό της δημιουργίας ενός χάρτη πλέγματος κατάληψης, χρησιμοποιώντας αισθητήρες χαμηλού κόστους. Στόχος μας ήταν να δούμε κατά πόσο είναι εφικτό να δημιουργηθεί ένας αξιόπιστος χάρτης με αισθητήρες δύο διαφορετικών τύπων, καθώς και να διαπιστώσουμε ποιος από τους δύο φέρει καλύτερα αποτελέσματα.

Ο πρώτος αισθητήρας που επιλέξαμε ήταν τύπου ultrasonic, δηλαδή αισθητήρας που ανιχνεύει αντικείμενα στέλνοντας ένα ηχητικό κύμα σε συχνότητα μεγαλύτερη από αυτήν που αντιλαμβάνεται το ανθρώπινο αυτί και έπειτα ο δέκτης λειτουργεί σαν μικρόφωνο λαμβάνοντας πίσω αυτό το κύμα. Η τελική απόσταση από το αντικείμενο υπολογίζεται μέσω υπολογισμού του χρόνου από την εκπομπή του κύματος μέχρι τη λήψη.

Ο δεύτερος αισθητήρας που επιλέξαμε ήταν τύπου laser Time-of-Flight (ToF). Οι αισθητήρες αυτοί εκπέμπουν μια δέσμη laser και αισθητήρες μετρούν τον χρόνο που έχει παρέλθει από την στιγμή εκπομπής ενός παλμού κύματος μέχρι τη στιγμή που επιστρέφει στον αισθητήρα, αφότου έχει ανακλαστεί σε ένα αντικείμενο. Έχουν την δυνατότητα να παράγουν εικόνες τριών διαστάσεων (X, Y, Z), με την ανάλυση του βάθους Z να είναι σχετικά χαμηλή.

Μερικά βασικά πλεονεκτήματα που έχουν οι αισθητήρες υπερήχων σε σχέση με τους αισθητήρες ToF είναι πως αδιαφορούν για το χρώμα των αντικειμένων που ανιχνεύουν και μπορούν να δουλέψουν αποτελεσματικά με αντικείμενα διαφορετικής υφής. Οι αισθητήρες ToF είναι κατάλληλοι για ανίχνευση μακρινής απόστασης, έχουν μεγάλη συχνότητα ανάγνωσης και έχουν την δυνατότητα 3D χαρτογράφησης.

Επιλέξαμε να προσεγγίσουμε το πρόβλημα πειραματικά, παρατηρώντας και εξετάζοντας τη δυνατότητα χαρτογράφησης των δύο αισθητηρίων υπό διαφορετικά σενάρια, με αντικείμενα διαφορετικών διαστάσεων και χρωμάτων, τοποθετημένα σε ευνοϊκές και μη ευνοϊκές θέσεις για τους αισθητήρες.

Τελευταίο βήμα για την εργασία θα ήταν η τοποθέτηση αυτής της κατασκευής πάνω σε ένα όχημα διαφορικής οδήγησης, με σκοπό την εκτέλεση χαρτογράφησης, ενώ το όχημα κινείται στο περιβάλλον. Ωστόσο, πολλά προβλήματα παρουσιαστήκαν κατά την εκτέλεση των αρχικών πειραμάτων μας, τόσο στην εγκατάσταση των επιλεγμένων αισθητήρων μας, όσο και στην σωστή προσαρμογή τους στον αλγόριθμο (mounting, καλωδίωση, εύρεση κατάλληλων βιβλιοθηκών, πρωτόκολλο i2c). Αυτό είχε ως αποτέλεσμα μεγάλο μέρος του διαθέσιμου χρόνου μας να δαπανηθεί στην επίλυση αυτών των προβλημάτων, που αποτελούσαν προτεραιότητα για την πρόοδο της εργασίας και να αναγκαστούμε να αφήσουμε το κομμάτι της τοποθέτησης της κατασκευής πάνω στο όχημα διαφορικής οδήγησης για μελλοντική επέκταση της εργασίας.

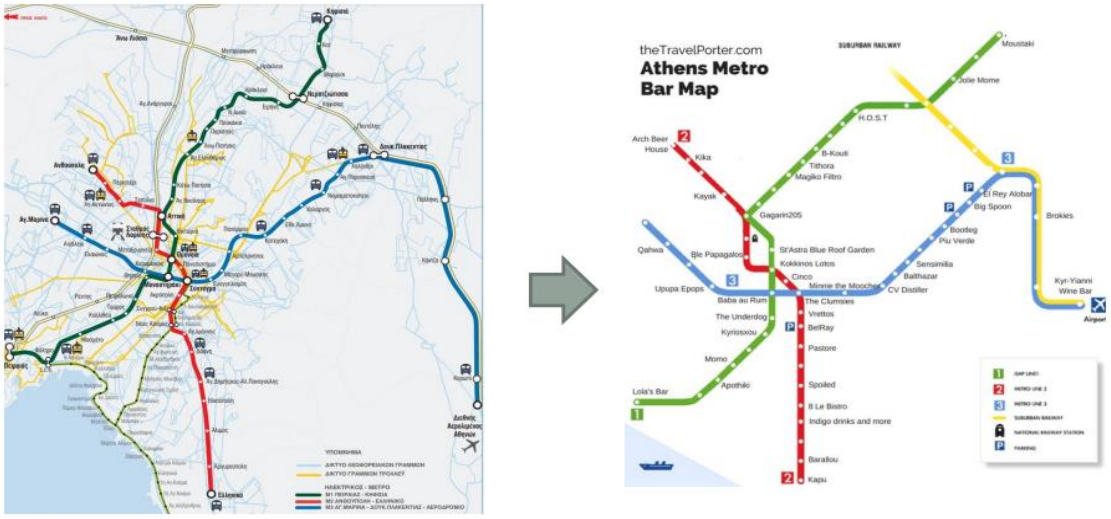
1.2 Χαρτογράφηση

Χαρτογράφηση είναι η διαδικασία κατά την οποία επιλεγμένες πληροφορίες εξάγονται από αισθητήρες και αποθηκεύονται σε μια δομή δεδομένων. Αυτή η δομή δεδομένων αποκαλείται χάρτης. Τέτοιοι χάρτες μπορούν να αναπαραστήσουν το περιβάλλον είτε σε δύο διαστάσεις, είτε σε τρεις. Τα είδη χαρτών που χρησιμοποιούνται στην ρομποτική είναι τρία

- Οι τοπολογικοί χάρτες (topological maps)
- Οι σημασιολογικοί χάρτες (semantic maps)
- Οι μετρικοί χάρτες (metric maps)

1.3 Τοπολογικοί χάρτες

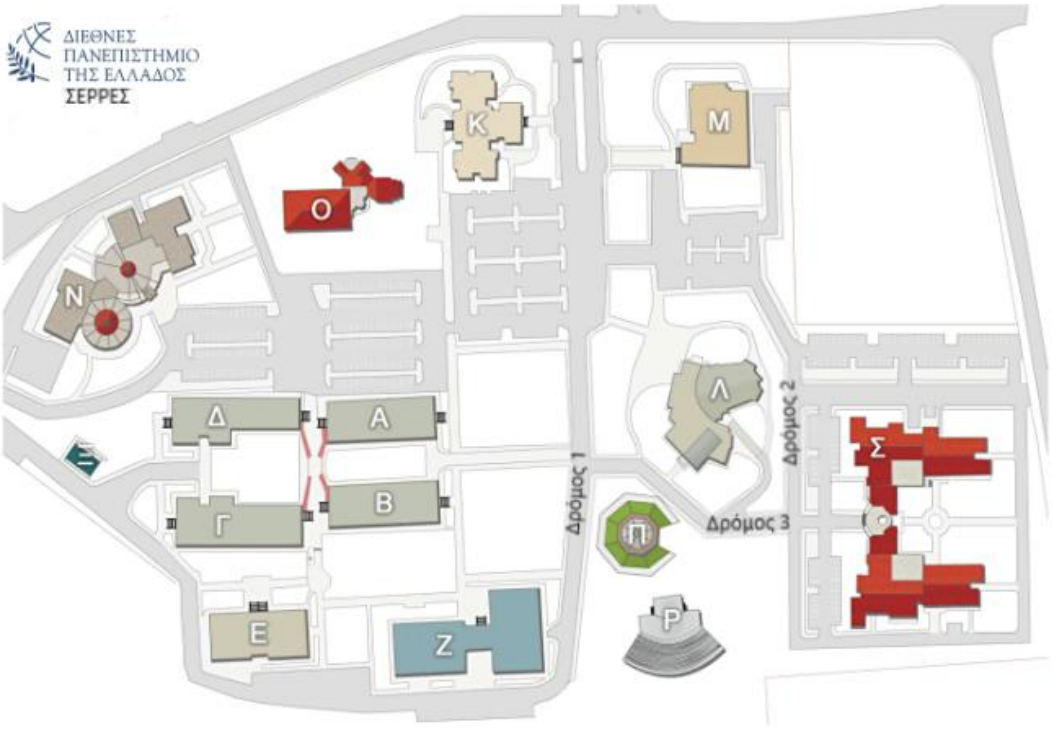
Τοπολογικός χάρτης ονομάζεται ο χάρτης που έχει υποστεί τόσο μεγάλη γενίκευση στα δεδομένα του, ώστε διατηρεί μόνο τα απολύτως απαραίτητα. Στους χάρτες αυτούς, ο προσανατολισμός και η κλίμακα των στοιχείων έχει παραμορφωθεί, αλλά οι σχετικές θέσεις των σημείων διατηρούνται. Τα πιο γνωστά παραδείγματα τοπολογικών χαρτών είναι οι χάρτες που αναπαριστούν τις διαδρομές του μετρό. Το όνομα αυτής της κατηγορίας των χαρτών βασίζεται στην τοπολογία των μαθηματικών.



Εικόνα 1.1 Αντιστοιχία Γεωμετρικού-Τοπολογικού χάρτη

1.3 Σημασιολογικοί χάρτες

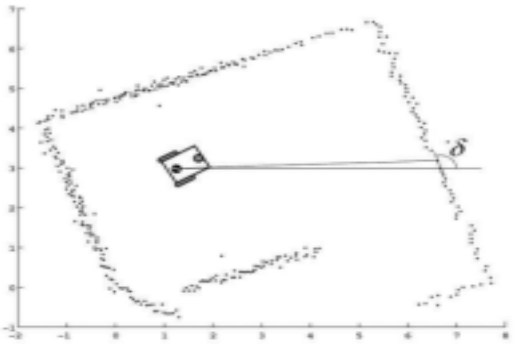
Πρόκειται για χάρτες που προσδιορίζουν αντικείμενα και διαδρομές όπου έχουν δοθεί ετικέτες (labels). Στους χάρτες αυτούς καταγράφεται η σχετική θέση των αντικειμένων και για τον προσδιορισμό των θέσεων χρησιμοποιούνται οι ετικέτες των αναπαριστώμενων στοιχείων, αντί των αριθμητικών συντεταγμένων.



Εικόνα 1.2 Σημασιολογικός χάρτης campus Σερρών.

1.4 Μετρικοί χάρτες

Ένας μετρικός χάρτης, απεικονίζει τις θέσεις των αντικειμένων στο χώρο με βάση μαθηματικές συντεταγμένες. Οι μετρικοί χάρτες βασίζονται στο πλαίσιο απόλυτης αναφοράς και σε αριθμητικές εκτιμήσεις για την θέση των αντικειμένων στον χώρο. Οι χάρτες αυτοί είναι ακριβείς στην αποσύνθεση του περιβάλλοντος. Συνήθως τέτοιοι χάρτες περιέχουν μόνο χαρακτηριστικά που ανακτήθηκαν από αισθητήρες και ενώ είναι πολύ ακριβείς, έχουν πολύ μεγάλες απαιτήσεις σε μνήμη. Σε αυτήν την κατηγορία χαρτών εντάσσονται και οι χάρτες πλέγματος κατάληψης, με τους οποίους θα ασχοληθούμε σε αυτήν την διπλωματική εργασία.



Εικόνα 1.3 Παράδειγμα μετρικού χάρτη παραγόμενου από αισθητήρες του ρομπότ

1.5 Ρομποτική χαρτογράφηση

Η ρομποτική χαρτογράφηση είναι ένας κλάδος που σχετίζεται με τα συστήματα ρομποτικής αίσθησης και την χαρτογραφία. Στόχος για ένα αυτόνομο ρομπότ είναι να μπορεί να κατασκευάσει ή να χρησιμοποιήσει έναν χάρτη για να προσδιορίσει την θέση του και ταυτόχρονα να αποφύγει τυχόν εμπόδια στην διαδρομή του. Οι βασικές πηγές πληροφοριών που χρησιμοποιούν τα ρομπότ για την χαρτογράφηση είναι οι αισθητήρες που φέρουν πάνω τους. Η εσωτερική αναπαράσταση ενός χάρτη στην ρομποτική μπορεί να είναι είτε μετρική, είτε τοπολογική. Στο μετρικό πλαίσιο η αναπαράσταση γίνεται σε έναν δισδιάστατο χώρο στον οποίο τα αντικείμενα τοποθετούνται με ακριβείς συντεταγμένες. Η αναπαράσταση αυτή είναι πολύ ευαίσθητη στον θόρυβο και είναι δύσκολο να υπολογιστούν οι ακριβείς αποστάσεις. Στο τοπολογικό πλαίσιο εξετάζονται μόνο περιοχές και οι σχέσεις μεταξύ τους. Έτσι ο χάρτης αποτελεί ένα γράφημα, στο οποίο οι κόμβοι αντιστοιχούν σε θέσεις και τα τόξα αντιστοιχούν σε μονοπάτια. Για την αντιμετώπιση της αβεβαιότητας, πολλές τεχνικές χρησιμοποιούν πιθανολογικές αναπαραστάσεις του χάρτη.

1.6 Χαρτογράφηση Πλέγματος Κατάληψης

Οι χάρτες πλέγματος κατάληψης δημιουργούν χάρτες από αβέβαια και θορυβώδη δεδομένα μέτρησης, με την υπόθεση πως η πόζα του ρομπότ είναι γνωστή. Ένας τέτοιος χάρτης αναπαριστά το περιβάλλον με την μορφή πίνακα, στον οποίο κάθε κελί αντιστοιχεί σε μια τιμή. Αυτή η τιμή αντιστοιχεί στην πιθανότητα να υπάρχει αντικείμενο ή όχι στην εκάστοτε θέση. Η πιθανότητα αυτή αναπαρίσταται με τιμές από μηδέν μέχρι ένα, με το μηδέν να αναπαριστά την απουσία αντικειμένου, ενώ το ένα την παρουσία.

Στόχος ενός αλγορίθμου χαρτογράφησης πλέγματος είναι να υπολογίσει την ύστερη πιθανότητα (posterior probability) να λάβουμε έναν χάρτη m, με βάση τις μέχρι τώρα θέσεις του ρομπότ και με βάση τις μέχρι τώρα μετρήσεις, όπως φαίνεται στην εξίσωση 1.1.

$$p(m \mid Z_{1:t}, X_{1:t}) \qquad (1.1)$$

Το m αντιπροσωπεύει τον χάρτη. Το $Z_{1:t}$ αντιπροσωπεύει το σύνολο όλων των μετρήσεων και το $X_{1:t}$ το σύνολο κάθε πόζας του ρομπότ μέχρι τον χρόνο t.

Οι χάρτες πλέγματος κατάληψης παριστάνουν τον χάρτη ως ένα πεπερασμένο πλέγμα πάνω από τον συνεχή χώρο στο περιβάλλον. Υποθέτοντας πως το m_i υποδηλώνει το κελί του πλέγματος με δείκτη i.

$$m=\{ m_i \} \text{ ή } m=\sum_i m_i \quad (1.2)$$

Σε κάθε m_i μια δυαδική τιμή πιθανότητας έχει ανατεθεί ως $p(m_i)$. Η τιμή αυτή καθορίζει εάν το κελί είναι κατειλημμένο ή όχι. Ένας λεπτομερής χάρτης πλέγματος κατάληψης μπορεί να αποτελείται από πάρα πολλά μεμονωμένα κελιά. Δεδομένου ότι χρειαζόμαστε δεκάδες χιλιάδες κελιά για να αναπαραστήσουμε τον περιβάλλοντα χώρο μας και λαμβάνοντας υπόψη ότι κάθε κελί έχει ανά πάσα στιγμή μια δυαδική τιμή πιθανότητας να

είναι κατειλημμένο ή όχι (0 ή 1), ο αριθμός των διαφορετικών χαρτών που ορίζονται σε αυτό το διάστημα είναι πολύ μεγάλος.

Μια τυπική προσέγγιση του πλέγματος κατάληψης είναι η ανάλυση του προβλήματος σε μικρότερα προβλήματα, εκτιμώντας την πιθανότητα κατάληψης κάθε μεμονωμένου κελιού

$$p(m_i \mid z_{1:t}, x_{1:t}) \qquad (1.3)$$

Έτσι καταλήγουμε στο ότι αυτά τα προβλήματα εκτίμησης γίνονται σταδιακά δυαδικά προβλήματα. Βέβαια αυτό δημιουργεί νέα προβλήματα, όπως το ότι δεν μπορούμε να εξετάσουμε σχέσεις σε γειτονικά κελιά. Ως εκ τούτου, ο χάρτης προσεγγίζεται ως το γινόμενο των πιθανοτήτων όλων των κελιών του.

$$p(m \mid z_{1:t}, x_{1:t}) = \prod_i p(m_i \mid z_{1:t}, x_{1:t}) \qquad (1.4)$$

Λόγω αυτής της παραγοντοποίησης, η εκτίμηση κάθε κελιού γίνεται πρόβλημα δυαδικής εκτίμησης με στατική κατάσταση. Το φίλτρο Bayes με στατική κατάσταση είναι ο ιδανικός αλγόριθμος για να προσδιορίσουμε αυτές τις εκτιμήσεις. Ο αλγόριθμος στον πίνακα 1.1 εφαρμόζει αυτό το φίλτρο στο πρόβλημα χαρτογράφησης πλέγματος κατάληψης. Μια αξιοσημείωτη ιδιότητα αυτού του αλγόριθμου είναι η χρήση λογαριθμικής αναπαράστασης της κατάληψης.

$$l_{t,i} = \log \frac{p(m_i \mid z_{1:t}, x_{1:t})}{1 - p(m_i \mid z_{1:t}, x_{1:t})} \qquad (1.5)$$

Η χρήση της λογαριθμικής αναπαράστασης ωφελεί τον αλγόριθμο χάρη στα αριθμητικά του πλεονεκτήματα, σε περιπτώσεις μικρών πιθανοτήτων. Οι πιθανότητες μπορούν να ανακτηθούν εύκολα από την λογαριθμική τους αναλογία.

$$p(m_i \mid z_{1:t}, x_{1:t}) = 1 - \frac{1}{e^{l_{t,i}}} \qquad (1.6)$$

Η βασική λειτουργία του αλγορίθμου χαρτογράφησης πλέγματος κατάληψης στον πίνακα 1.1 είναι αρκετά απλή: κάνει κύκλους σε κάθε κελί πλέγματος i και καθορίζει εάν αυτό το κελί ανήκει στο πεδίο αντίληψης. Η τιμή των κελιών που βρίσκονται μέσα στην εμβέλεια κάλυψης του αισθητήρα έχουν την τιμή τους ενημερωμένη, ενώ η τιμή των άλλων κελιών μένει αμετάβλητη.

1. Algorithm occupancy_grid_mapping($\{l_{t-1,i}\}$, χ_t, z_t):
2. For all cells m_i do
3. If m_i in perceptual field of z_t then
4. $l_{t,i} = l_{t-1,i}$ + inverse_sensor_model(m_i, χ_t, z_t)-l_0
5. else
6. $l_{t,i} = l_{t-1,i}$
7. endif
8. endfor
9. Return$\{l_{t,i}\}$

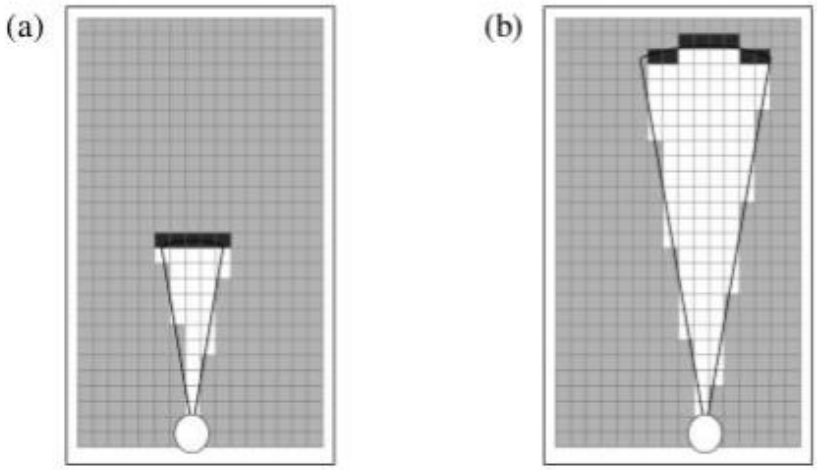
Αλγόριθμος 1.1 Ο αλγόριθμος χαρτογράφησης πλέγματος κατάληψης

Η συνάρτηση inverse_sensor_model υλοποιεί το μοντέλο αντίστροφης μέτρησης της μορφής $p(m_i | z_t, x_t)$.

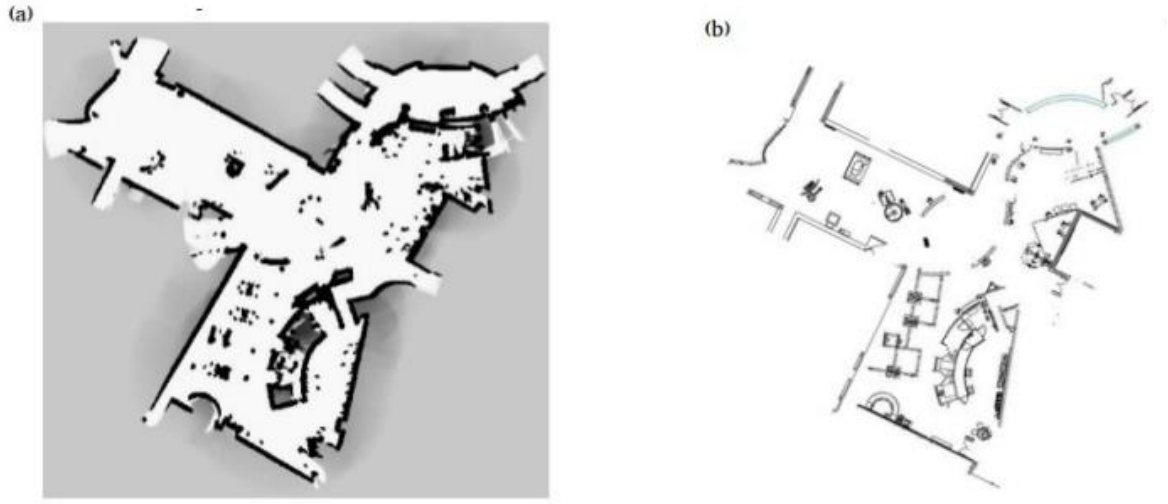
Καθορίζει μια κατανομή στη μεταβλητή δυαδικής κατάστασης mi, που συσχετίζει ένα συγκεκριμένο κελί με τη μέτρηση z_t και την θέση x_t. Αυτό είναι βολικό σε καταστάσεις όπου ο χώρος μέτρησης είναι πολύ πιο περίπλοκος από τον χώρο κατάστασης. Μια βασική λειτουργία για έναν ανιχνευτή εύρους δίνεται στον πίνακα 1.2 και απεικονίζεται στα σχήματα α και β.

1: Algorithm incerse_range_sensor_model(i, χ_t, z_t):
2: Let χ_i, y_i be the center-of-mass of m_i
3: $r = \sqrt{(\chi_i - \chi)^2 + (y_i - y)^2}$
4: $\Phi = \text{atan2}(y_i - y, \chi_i - \chi) - \theta$
5: $k = \text{argmin}_j \mid \Phi - \theta_{j,\,\text{sens}} \mid$
6: if $r > \min(z_{max}, z_t^k + \alpha/2)$ or $\mid \Phi - \theta_{k,\,\text{sens}} \mid > \beta/2$ then
7: return l_0
8: if $z_t^k < z_{max}$ and $\mid r - z_{max} \mid < \alpha/2$
9: return l_{occ}
10: if $r \leq z_t^k$
11: return l_{free}
12: endif

Αλγόριθμος 1.2 Απλό μοντέλο αντίστροφης μέτρησης για ρομπότ εξοπλισμένο με ανιχνευτές απόστασης. Το α είναι το πάχος των εμποδίων και β το πλάτος δέσμης του αισθητήρα. Οι τιμές l_{occ} και l_{free} δηλώνουν τον όγκο των αποδεικτικών στοιχείων που φέρει μια ανάγνωση για τις 2 διαφορετικές περιπτώσεις, που το κελί είναι κατειλημμένο (occ) ή ελεύθερο (free).



Εικόνα 1.4 Δύο διαφορετικά παραδείγματα του μοντέλου αντίστροφης μέτρησης για δύο διαφορετικές εμβέλειες μέτρησης. Η σκοτεινότητα κάθε κελιού δηλώνει την πιθανότητα να είναι κατειλημμένο.



Εικόνα 1.5 (a) Χάρτης πλέγματος κατάληψης και (b) Αρχιτεκτονικός χάρτης ενός μεγάλου χώρου.

Η εικόνα 1.5 απεικονίζει ένα παράδειγμα χάρτη πλέγματος κατάληψης δίπλα στο αρχιτεκτονικό σχέδιο του περιβάλλοντος χώρου. Ο χάρτης κατασκευάστηκε με χρήση μετρήσεων από ρομπότ που λήφθηκαν όσο αυτό εκτελούσε SLAM. Η κλίμακα του γκρι δείχνει την πιθανότητα κατάληψης, με το μαύρο να αντιστοιχεί στην υψηλή πιθανότητα ένα κελί να είναι κατειλημμένο και το λευκό να αντιστοιχεί στην υψηλή πιθανότητα να είναι ελεύθερο. Το γκρι χρώμα αντιπροσωπεύει ενδιάμεση πιθανότητα. Όπως φαίνεται στην εικόνα 1.5, ένας χάρτης πλέγματος κατάληψης δείχνει όλα τα δομικά στοιχεία, καθώς και τα εμπόδια που είναι αντιληπτά από τον αισθητήρα. Αυτά τα χαρακτηριστικά κάνουν τους χάρτες πλέγματος κατάληψης έναν κατάλληλο τρόπο χαρτογράφησης κατά την αντιμετώπιση προβλημάτων ταυτόχρονου εντοπισμούς και χαρτογράφησης (SLAM).

Κεφάλαιο 2.

Κατάσταση της τέχνης

2.1 Εξερεύνηση Και Χαρτογράφηση

Η ρομποτική εξερεύνηση συσχετίζεται άμεσα με τη χαρτογράφηση, μιας και αν οι χάρτες ήταν ήδη διαθέσιμοι, δεν θα ήταν απαραίτητη η εξερεύνηση στο πλαίσιο της χωρικής κατανόησης. Έτσι μπορούμε να ορίσουμε το πρόβλημα της ρομποτικής χαρτογράφησης, ως αυτό της απόκτησης ενός χωρικού μοντέλου του περιβάλλοντος ενός ρομπότ. Αφού η χαρτογράφηση περιλαμβάνει μετρήσεις που αποκτώνται από αισθητήρες, προβλήματα στην χαρτογράφηση προέρχονται από την περιοχή μετρήσεων των αισθητήρων. Όπως συνοψίζεται στο [1], οι βασικές προκλήσεις χαρτογράφησης μπορούν να αποδοθούν σε πέντε πηγές:

1. Θόρυβος μετρήσεων.
2. Μεγάλες διαστάσεις των οντοτήτων που χαρτογραφούνται.
3. Συσχέτιση/Αντιστοιχία δεδομένων.
4. Δυναμικά περιβάλλοντα.
5. Επιλογή ελέγχου/κίνησης του ρομπότ κατά την εξερεύνηση.

Για την αυτόματη πλοήγηση και εξερεύνηση, το όχημα πρέπει να γνωρίζει το περιβάλλον όπου πορεύεται. Επειδή αυτές συνήθως δεν είναι διαθέσιμες εκ των προτέρων, το ρομπότ πρέπει να τις ανιχνεύσει. Εφαρμογές όπως η εξερεύνηση, η αποφυγή εμποδίων ή η πλοήγηση εξαρτώνται από το ίδιο το περιβάλλον [2], άρα πολλές φορές το περιβάλλον χρειάζεται αναπαράσταση. Οι περισσότεροι αλγόριθμοι χαρτογράφησης εργάζονται για τον εντοπισμό του ρομπότ εκτός από την επίλυση του προβλήματος της χαρτογράφησης. Σχεδόν όλοι οι αλγόριθμοι τελευταίας τεχνολογίας για τη ρομποτική χαρτογράφηση είναι μοντελοποιούν με στοχαστικό τρόπο την κίνηση του οχήματος, μετατρέποντας τις πληροφορίες από τους αισθητήρες σε χρήσιμες για τον χάρτη πληροφορίες [1].

Ένα ενδιαφέρον μέρος της χαρτογράφησης είναι να καθοριστεί εάν ο τελικός χάρτης απλώς αντιπροσωπεύει πληροφορίες από τους αισθητήρες ή εάν απεικονίζεται και η σχέση μεταξύ των πληροφοριών στον χάρτη. Ανάλογα με το είδος του χάρτη που χρησιμοποιείται, για να συμπεράνουμε μια σχέση ή να ερμηνεύσουμε τα δεδομένα του χάρτη απαιτείται ακρίβεια στη συσχέτιση δεδομένων. Αυτό συνίσταται στον προσδιορισμό του εάν ένα στοιχείο του περιβάλλοντος που γίνεται αντιληπτό από διαφορετικές θέσεις είναι στην πραγματικότητα το ίδιο [2]. Υποθέτοντας ότι η συσχέτιση δεδομένων έχει επιλυθεί σωστά, το τελευταίο βήμα είναι η συγχώνευση των πληροφοριών που αποκτήθηκαν πρόσφατα, στον χάρτη που δημιουργήθηκε μέχρι στιγμής.

Οι συγγραφείς της εργασίας [2] έχουν μια ενδιαφέρουσα άποψη για τις απαρχές του προβλήματος SLAM και ειδικότερα για το πρόβλημα της χαρτογράφησης, αναφέροντας πως οι χαρτογράφοι του δέκατου έκτου αιώνα είχαν το ίδιο πρόβλημα να λύσουν. Η εξέλιξη των αισθητήρων έχει προχωρήσει πολύ και μαζί της γεννήθηκαν περισσότερες

λύσεις για την αντιμετώπιση του προβλήματος της χαρτογράφησης. Η στοχαστική φύση του προβλήματος δεν έχει αλλάξει ακόμη, ωστόσο οι διαφορετικές μορφές μετρήσεων επιτρέπουν την αντίληψη, την ερμηνεία και την αναπαράσταση του ίδιου περιβάλλοντος με διαφορετικές πληροφορίες. Υπάρχει τεράστια ποικιλία στη βιβλιογραφία που προσεγγίζει τα προβλήματα του εντοπισμού και της χαρτογράφησης, όπως αυτές που χρησιμοποιούν προσεγγίσεις με το φίλτρο Kalman και τις τροποποιήσεις τους [3], [4], [5], [6], εκείνες που επικεντρώνονται στους αλγόριθμους μεγιστοποίησης προσδοκιών του Dempster [7], [8], [9] και αυτές που βασίζονται στην υπολογιστική όραση [9].

Οι χάρτες μπορούν να λάβουν αρκετές μορφές, αλλά οι δύο πιο αντιπροσωπευτικές μορφές που σχετίζονται με την ρομποτική είναι οι μετρικοί χάρτες και οι τοπολογικοί χάρτες. Οι μετρικοί χάρτες βασίζονται στο πλαίσιο απόλυτης αναφοράς και σε αριθμητικές εκτιμήσεις για την θέση των αντικειμένων στον χώρο, ενώ οι τοπολογικοί χάρτες (γνωστοί και ως σχεσιακοί χάρτες) αναπαριστούν μόνο πληροφορίες συνδεσιμότητας και συνήθως σε μορφή γραφήματος. Πολλές πραγματικές αναπαραστάσεις έχουν τόσο μετρικά στοιχεία, όσο και τοπολογικά. Μετρικοί χάρτες που αναπαριστούν την κατάληψη ενός χώρου για παράδειγμα ή γεωμετρικοί χάρτες που αναπαριστούν συγκεκριμένα αντικείμενα με ταμπέλες, συχνά περιλαμβάνουν και πληροφορίες συνδεσιμότητας [33].

Οι τοπολογικοί χάρτες, σε αντίθεση με τους μετρικούς χάρτες, αποτυπώνουν πληροφορίες συσχέτισης, ελαχιστοποιώντας πληροφορίες που είναι άσχετες ή μπερδεύουν. Ως αποτέλεσμα, οι τοπολογικοί χάρτες έχουν μια πολύ ρητή σύνδεση με τη σημασιολογία ενός προβλήματος. Ο χάρτης του μετρό, καθώς και οι οδηγίες πλοήγησης, είναι τυπικά παραδείγματα τοπολογικών χαρτών. Αρκετοί συγγραφείς αναφέρουν ξεκάθαρα πως οι μετρικοί χάρτες δεν είναι κατάλληλοι για την αναπαράσταση χώρου μεγάλης κλίμακας [33].

Προκειμένου να αξιοποιηθούν τα πλεονεκτήματα τόσο των μετρικών όσο και των τοπολογικών αναπαραστάσεων, είναι σκόπιμο να εξεταστεί η κατασκευή μιας αναπαράστασης χρησιμοποιώντας παρατηρήσεις από μια λιγότερο αφηρημένη αναπαράσταση. Επιπλέον, είναι φυσικό να λαμβάνεται υπόψιν η τοπική περιγραφή πριν τις αλληλεπιδράσεις μεγάλης κλίμακας. Αυτό οδηγεί σε μια ιεραρχική στρώση διαδοχικών αναπαραστάσεων των δεδομένων του χάρτη, όπως στα ακόλουθα πέντε επίπεδα [33]

1. Επίπεδο αισθητηρίων: σήματα ακατέργαστων δεδομένων ή μετασχηματισμοί αυτών των σημάτων
2. Γεωμετρικό επίπεδο: δυσδιάστατα ή τρισδιάστατα αντικείμενα που συνάγονται από δεδομένα των αισθητήρων
3. Τοπικό σχεσιακό επίπεδο: Δομικές, σημασιολογικές ή λειτουργικές σχέσεις μεταξύ γεωμετρικών αντικειμένων που βρίσκονται κοντά μεταξύ τους.
4. Τοπολογικό επίπεδο: Οι σχεσιακές συνδέσεις μεγάλης κλίμακας που συνδέουν αντικείμενα και τοποθεσίες στο περιβάλλον.
5. Σημασιολογικό επίπεδο: Λειτουργικές ταμπέλες που σχετίζονται με τα συστατικά του χάρτη

Οι Chatila και Laumond [34] ήταν από τους πρώτους που περιέγραψαν ένα σύνολο χαρτών που κατέληγαν σε μια τοπολογική αναπαράσταση του περιβάλλοντος ξεκινώντας από μια μετρική. Σε αντίθεση, οι Kuipers και Levitt [344] εξέτασαν τα μετρικά δεδομένα

από μια ουσιαστικά τοπολογική αναπαράσταση. Οι μέθοδοι εξερεύνησης τους λάμβαναν υπόψιν την χρήση τοπολογικών οροσήμων χαμηλού επιπέδου, ως την πιο βασική πρωτόγονη παρατήρηση από την οποία μπορεί να προκύψει μετρικό συμπέρασμα [33].

Μια δεύτερη διχοτομία σχετίζεται με τον τύπο των δεδομένων που αναπαρίστανται σε έναν χάρτη. Οι πιο συμβατικοί χάρτες απεικονίζουν την κατάληψη του χώρου σε δύο ή τρεις διαστάσεις. Μια ιδιαίτερης σημασίας διαφορετική κατηγορία χάρτη για τα ρομπότ είναι οι αντιληπτικοί χάρτες που σχετίζονται άμεσα με τις μετρήσεις των αισθητήρων στη χωρική θέση, χωρίς να καταφεύγουν σε κάποια ενδιάμεση περιγραφή, όσον αφορά τα φυσικά αντικείμενα [33].

Η εξερεύνηση των χαρτών αποτελεί απαίτηση για πολλές ρεαλιστικές εφαρμογές κινούμενων ρομπότ. Οι υπάρχοντες χάρτες των περισσότερων περιβαλλόντων είναι ανακριβείς και σχεδόν κάθε περιβάλλον που καταλαμβάνεται από ανθρώπους υφίσταται συνεχή αλλαγή. Ως αποτέλεσμα, τα κινητά ρομποτικά συστήματα πρέπει να είναι ικανά να προσαρμοστούν στις αλλαγές του περιβάλλοντος και αν διατηρούν χάρτες θα πρέπει να μπορούν να τους ανανεώσουν [33].

2.2 Πλέγμα κατάληψης

Το πλέγμα κατάληψης είναι ένα ισχυρό εργαλείο χαρτογράφησης που εισήχθη για πρώτη φορά από τον Elfes [10], [11]. Στην ουσία, το πλέγμα κατάληψης υπολογίζει την πιθανότητα ένα συγκεκριμένο κελί σε μια αναπαράσταση του περιβάλλοντος να είναι κατειλημμένο. Χρησιμοποιεί μοντέλα πιθανοτήτων εξωτερικών αισθητήρων για την καταγραφή της αβεβαιότητας των δεδομένων των αισθητήρων και δεν βασίζεται σε γεωμετρικά μοντέλα του περιβάλλοντος. Επομένως δεν εξαρτάται από την ακρίβεια ενός συγκεκριμένου παγκόσμιου μοντέλου. Έχουν διερευνηθεί αρκετές παραλλαγές του πλέγματος κατάληψης από διάφορους ερευνητές. Στην εργασία [12] παρουσιάζεται μια νέα μέθοδος για τον υπολογισμό του ελεύθερου χώρου, που βασίζεται σε δυναμικό προγραμματισμό. Οι εκτιμήσεις των θέσεων των εμποδίων που δημιουργούνται από στερεοσκοπικές κάμερες με φίλτρο Kalman αποθηκεύονται σε ένα πολικό πλέγμα κατάληψης, το οποίο παρέχει μεγάλη ανάλυση με κόστος κάποιον υπολογιστικό χρόνο. Στο [13] παρουσιάζεται μια νέα λύση παρακολούθησης στο πλέγμα κατάληψης βασισμένη σε σωματίδια για την παρακολούθηση του δυναμικού περιβάλλοντος οδήγησης. Τα σωματίδια αυτά έχουν διπλή φύση. Η μία υποδηλώνει υποθέσεις, όπως στον αλγόριθμο φίλτρου σωματιδίων και η άλλη είναι τα δομικά στοιχεία του μοντελοποιημένου κόσμου. Ο αλγόριθμος παρακολούθησης επικεντρώνεται σε σωματίδια αντί για κελιά. Ωστόσο αυτός ο αλγόριθμος έχει μεγάλο πρόβλημα ταχύτητας λόγω της ποσότητας των σωματιδίων. Για να ξεπεραστεί το πρόβλημα της ανάπτυξης του χάρτη και η σταθερή αύξηση της αποθήκευσης και επεξεργασίας, ένα τοπικό πλέγμα κατάληψης που θα κινείται μαζί με το όχημα περιγράφεται από τον Marlow [14].

Η χαρτογράφηση είναι μια κρίσιμη και απαραίτητη εργασία για ένα ευρύ φάσμα εφαρμογών ρομποτικής. Η ικανότητα κατασκευής ενός χάρτη επιτρέπει σε ένα ρομπότ να εντοπίζει τον εαυτό του και να πλοηγείται στο περιβάλλον αυτόνομα. Για τα αυτόνομα

οχήματα η χαρτογράφηση έχει γίνει αναπόσπαστο μέρος της πλήρους αυτόνομης οδήγησης. Αυτό είναι ιδιαίτερα σημαντικό και για την πλοήγηση εκτός δρόμου σε άγνωστες περιοχές. Επιπλέον τέτοια οχήματα συχνά πρέπει να διασχίζουν ακανόνιστο έδαφος με αντικείμενα που προεξέχουν από το έδαφος, όπως δέντρα, κλαδιά και θάμνοι. Ως εκ τούτου, ένας τρισδιάστατος χάρτης με χωρικές λεπτομέρειες είναι πολύ σημαντικός για πλοήγηση εκτός δρόμου.

Οι περισσότερες από τις μεθόδους 3D ανακατασκευής χρησιμοποιούν μια μέθοδο αναπαράστασης βασισμένη σε σημεία, αποθηκεύοντας απευθείας τις 3D μετρήσεις απόστασης. Ο καταλαμβανόμενος χώρος στο περιβάλλον μοντελοποιείται με 3D σημεία νέφους που επιστρέφονται από αισθητήρες απόστασης, όπως λέιζερ, στερεοσκοπικούς αισθητήρες ή κάμερες RGB-D. Η προσέγγιση των σημείων νέφους έχει χρησιμοποιηθεί σε πολλά συστήματα τρισδιάστατης χαρτογράφησης όπως αυτά που παρουσιάζονται από τους Geiger [15] και Alcantarilla [16] χρησιμοποιώντας στερεοσκοπικές εικόνες καθώς και SLAM προσεγγίσεις του Nüchter [17]. Τα μειονεκτήματα αυτού του είδους αναπαράστασης είναι ότι ούτε ο ελεύθερος χώρος, ούτε οι άγνωστες περιοχές μοντελοποιούνται και ο θόρυβος από τους αισθητήρες και τα δυναμικά αντικείμενα δεν αντιμετωπίζονται εύκολα. Ως αποτέλεσμα, τα σημεία νέφους είναι κατάλληλα μόνο για αισθητήρες υψηλής ακρίβειας σε στατικά περιβάλλοντα χωρίς την ανάγκη αναπαράστασης άγνωστων περιοχών. Επιπλέον η κατανάλωση μνήμης αυτής της αναπαράστασης αυξάνεται ραγδαία με τον αριθμό μετρήσεων κατά την πάροδο του χρόνου.

Το θέμα του ταυτόχρονου εντοπισμού και χαρτογράφησης, ήταν ένα πολύ ενεργό κομμάτι στην ρομποτική έρευνα από το 1986, όταν οι Peter Cheeseman και Hugh-Durrant Whyte πρωτομίλησαν για αυτό. Η δημιουργία του SLAM είχε ως αποτέλεσμα να αφιερωθούν πάρα πολλές εργασίες στην εύρεση κατάλληλων τεχνικών για να διαχειριστούν ρομπότ που εκτελούν εξερευνήσεις σε άγνωστα περιβάλλοντα. Αρκετές τεχνικές χαρτογράφησης έχουν αναπτυχθεί από τότε, τόσο για εσωτερικούς, όσο και για εξωτερικούς χώρους. Αυτές οι τεχνικές μπορούν χοντρικά να ταξινομηθούν σύμφωνα με την αναπαράσταση τους και την τεχνική εκτίμησης.

Υπάρχουν δύο κύριες μέθοδοι αναπαράστασης ενός χάρτη. Η πιο δημοφιλής είναι το πλέγμα κατάληψης. Οι προσεγγίσεις που βασίζονται στο πλέγμα κατάληψης είναι υπολογιστικά ακριβείς και απαιτούν πολλή μνήμη. Η δεύτερη μέθοδος αναπαράστασης χάρτη είναι με βάση τα χαρακτηριστικά (Feature Based), στην οποία το μοντέλο του χάρτη εκφράζεται με ορόσημα στο περιβάλλον. Αυτή η μέθοδος έγινε δημοφιλής χάρις στο πόσο συμπαγής είναι, πράγμα που αποτελεί πλεονέκτημα όσον αφορά την κατανάλωση μνήμης και την επεξεργαστική ταχύτητα. Από την άλλη πλευρά, τέτοια συστήματα βασίζονται σε προκαθορισμένες γνώσεις για τις δομές του περιβάλλοντος, πράγμα που περιορίζει σαφώς το πεδίο δράσης του ρομπότ.

2.3 Αισθητήρες

Ο Discand [18] παρουσιάζει μια σύντομη μελέτη σχετικά με τους διαθέσιμους αισθητήρες για την ανίχνευση εμποδίων. Από την μία πλευρά, οι παθητικοί αισθητήρες συλλαμβάνουν μόνο ενέργεια από το περιβάλλον και είναι επομένως φθηνοί και δεν προκαλούν παρεμβολές σε άλλους αισθητήρες. Οι πιο κοινοί παθητικοί αισθητήρες για εφαρμογές

αυτοκινήτων είναι οι κάμερες, παρέχοντας εικόνες υψηλής ανάλυσης από το περιβάλλον. Ωστόσο, απαιτείται μια πηγή ενέργειας όπως το ορατό φως ή υπέρυθρη ακτινοβολία για την περίπτωση κανονικών και υπέρυθρων καμερών αντίστοιχα. Πολλοί συγγραφείς έχουν πραγματοποιήσει ανίχνευση εμποδίων με βάση την όραση μέσω κανονικών έγχρωμων εικόνων [19] [20] και υπέρυθρων εικόνων [21]. Παρόλο που επιτυγχάνουν καλές επιδόσεις στην ανίχνευση εμποδίων και στην ταξινόμηση, αυτή η προσέγγιση μπορεί να μην λειτουργήσει καλά τη νύχτα ή κάτω από αντίξοες καιρικές συνθήκες (βροχή, χιόνι, σκόνη). Επιπλέον, οι μονές κάμερες δεν παρέχουν άμεσα πληροφορίες βάθους. Είναι πολύ συνηθισμένο να χρησιμοποιούνται στερεοσκοπικές κάμερες, όπως φαίνεται στην βιβλιογραφία [22], [23], [24]. Αυτή η προσέγγιση βασίζεται πάλι στις κατάλληλες καιρικές συνθήκες, όπως στην περίπτωση των μονών καμερών, για να λειτουργήσουν σωστά.

Από την άλλη πλευρά, είναι επίσης συχνή η χρήση ενεργών αισθητήρων, οι οποίοι εκπέμπουν ένα κύμα και μετράνε τις ληφθείσες αντιδράσεις από πιθανούς στόχους. Σε αυτήν την κατηγορία συμπεριλαμβάνουμε τα radar, τα lidar και τα sonar.

- Οι αισθητήρες τύπου Radar έχουν ενσωματωθεί πρόσφατα στο χώρο της ρομποτικής και των αυτόνομων οχημάτων [25], [26]. Τα ραντάρ που έχουν σχεδιαστεί για εφαρμογές αυτοκινήτων συνήθως λειτουργούν στη ζώνη κυμάτων των χιλιοστών, περίπου στα 76-77 GHz. Το κύριο πλεονέκτημα τους είναι η παντός καιρού ικανότητα τους που αντισταθμίζει την αδυναμία όρασης [27]. Έχουν καλή ανάλυση εύρους και είναι σε θέση να μετρήσουν την ακτινική ταχύτητα κινουμένων αντικειμένων. Το μειονέκτημα τους είναι συχνά μια κακή γωνιακή ανάλυση, η οποία σχετίζεται άμεσα με το μέγεθος της κεραίας. Ωστόσο η χρήση δεκτών συστοιχίας φάσης και οι υπέρ αναλυτικοί αλγόριθμοι επιτρέπουν πολύ καλύτερη ανάλυση, μέχρι ±1^{o} [28].

- Οι Lidar βασίζονται στην εκπομπή ακτίνων λέιζερ σε γωνιακά βήματα συνήθως στη υπέρυθρη ζώνη (900 nm). Παρέχουν εξαιρετική γωνιακή ανάλυση και ακρίβεια εύρους και γίνονται πολύ δημοφιλείς για ανίχνευση εμποδίων σε αυτόνομα οχήματα και SLAM [29], [30]. Ωστόσο, συνήθως υποφέρουν από σκέδαση φωτός κάτω από βαριές συνθήκες βροχής, χιονιού ή σκόνης [31]. Έχουν επίσης δυσκολίες με κατοπτρικές επιφάνειες και διάφανα αντικείμενα όπως το γυαλί. Ένα ακόμη μειονέκτημα, σε σύγκριση με τα ραντάρ, είναι ότι συνήθως δεν μπορούν να μετρήσουν την ακτινική ταχύτητα των στόχων.

- Οι Sonar βασίζονται στην ίδια αρχή με τα ραντάρ, αλλά εκπέμπουν υπερηχητικές κυματομορφές. Είναι φθηνότεροι, αλλά πολύ ευαίσθητοι στις καιρικές συνθήκες (η ταχύτητα του ήχου εξαρτάται σε μεγάλο βαθμό από τη θερμοκρασία). Ως εκ τούτου, δεν είναι πολύ κοινά στο πλαίσιο των οδικών οχημάτων για αυτόνομη οδήγηση, αν και υπάρχουν ορισμένες εφαρμογές στη βιβλιογραφία [32]. Τέλος είναι πιο δημοφιλείς στον χώρο των υποβρύχιων αυτόνομων οχημάτων.

Όπως μπορεί να παρατηρηθεί, κάθε τύπος αισθητήρα έχει τα δικά του πλεονεκτήματα και μειονεκτήματα και δεν υπάρχει ιδανικός για όλες τις περιπτώσεις. Επομένως για να δημιουργηθεί ένα ισχυρό σύστημα ανίχνευσης εμποδίων, είναι απαραίτητο να τα

συνδυάσουμε ώστε να αντισταθμίζουν το ένα τα μειονεκτήματα του άλλου. Αυτή η προσέγγιση είναι ευρέως γνωστή ως σύντηξη αισθητήρων (sensor fusion).

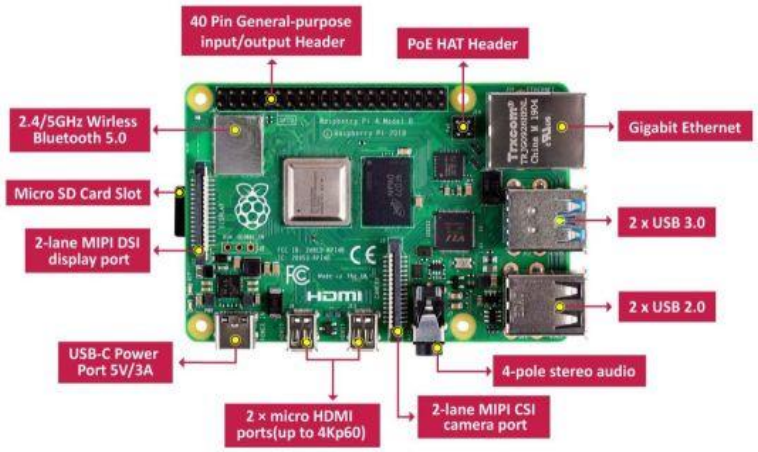
Κεφάλαιο 3.

Μεθοδολογία και εργαλεία

3.1 Raspberry Pi 4 Model B+

Το Raspberry Pi 4 λειτουργεί σαν εγκέφαλος για την κατασκευή μας, μιας και αυτό είναι υπεύθυνο για την επικοινωνία με τους αισθητήρες και τους κινητήρες, καθώς και την εκτέλεση του αλγορίθμου. Θεωρείται υπολογιστής τσέπης και ως μικροελεγκτής έχει κάποια πλεονεκτήματα όπως:

- Χαμηλό κόστος
- Μικρό όγκο
- Εύκολη πρόσβαση στο Internet



Εικόνα 4.1 Πλακέτα Raspberry Pi 4 Model B+

Η πλακέτα Raspberry Pi Model B 4 δημιουργήθηκε το 2018 και αποτελεί το νεότερο προϊόν της σειράς υπολογιστών Raspberry Pi. Τα τεχνικά χαρακτηριστικά της είναι τα εξής:

- Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
- 1GB, 2GB, 4GB or 8GB LPDDR4-3200 SDRAM (depending on model)
- 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE
- Gigabit Ethernet
- 2 USB 3.0 ports; 2 USB 2.0 ports.
- Raspberry Pi standard 40 pin GPIO header (fully backwards compatible with previous boards)
- 2 × micro-HDMI ports (up to 4kp60 supported)
- 2-lane MIPI DSI display port
- 2-lane MIPI CSI camera port
- 4-pole stereo audio and composite video port
- H.265 (4kp60 decode), H264 (1080p60 decode, 1080p30 encode)
- OpenGL ES 3.1, Vulkan 1.0
- Micro-SD card slot for loading operating system and data storage
- 5V DC via USB-C connector (minimum 3A*)
- 5V DC via GPIO header (minimum 3A*)
- Power over Ethernet (PoE) enabled (requires separate PoE HAT)
- Operating temperature: 0 – 50 degrees C ambient

Εικόνα 4.2 Θύρες GPIO για Raspberry Pi 4

3.2 Feetech FS5106B Servo

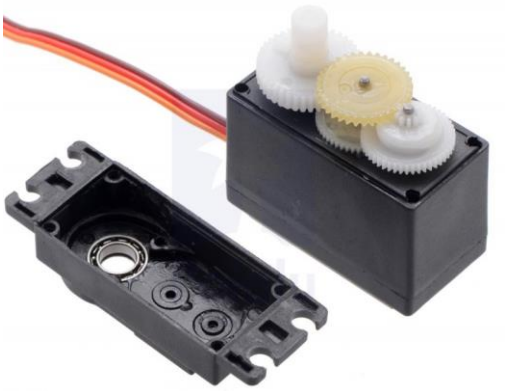
Το FS5106B είναι ένας αναλογικός σερβοκινητήρας τυπικού μεγέθους και γενικής χρήσης από την FEETECH. Ο άξονας εξόδου υποστηρίζεται από δύο ρουλεμάν για μειωμένη τριβή. Το FS5103B λειτουργεί με τυπικούς σερβοπαλμούς RC, παρέχοντας γωνία λειτουργίας περίπου 180° σε εύρος παλμών από 700 μs έως 2300 μs. Στην δική μας κατασκευή χρησιμοποιήθηκε για την περιστροφή των άλλων αισθητήρων, έτσι ώστε για κάθε μετατόπιση του ρομπότ να γίνεται ανίχνευση εύρους 170°.

Εικόνα 4.9 Feetech FS5106B Servo motor

Τα τεχνικά χαρακτηριστικά του Feetech FS5106B είναι τα εξής:

- Κατασκευαστής: Feetech
- Part Number: FS5106B
- Καθαρό Βάρος: 0.051kg
- Χώρα Προέλευσης: Κίνα
- Λειτουργία Servo: Αναλογική
- Μέγεθος Servo: Standard
- Ροπή (6V): 6kg.cm
- Ταχύτητα (6V): 0.16sec/60°
- Τύπος Γραναζιών: Πλαστικά
- Operating Voltage: 4.8V – 6V
- Control System: Analog
- Direction: CCW
- Operation Angle: 120degree

- Required Pulse: 900us-2100us
- Bearing Type: 2BB
- Motor Type: Metal
- Stall Torque:
    - 5kg.cm/69.56oz.in(4.8V)
    - 6kg.cm/83.47oz.in(6V)
- Operation Speed:
    - 0.18sec/60degree (4.8V)
    - 0.16sec/60degree (6V)



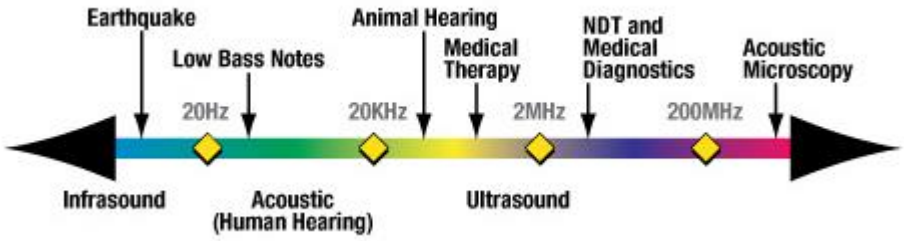
Εικόνα 4.10 Εσωτερικό Feetech FS5106B

3.3 Αισθητήρες

Οι αισθητήρες είναι σημαντικοί στην ρομποτική για αρκετούς λόγους. Για αρχή οι αισθητήρες επιτρέπουν στο ρομπότ να γίνει πιο αυτόνομο μιας και μπορεί να αντιληφθεί το δικό του περιβάλλον και μέσω κατάλληλου προγραμματισμού μπορεί να λαμβάνει αποφάσεις. Οι αισθητήρες είναι επίσης πολύ σημαντικοί για την απομακρυσμένη λειτουργία ενός ρομπότ. Δύο κύριοι αισθητήρες που χρησιμοποιούνται στο έργο μας είναι οι αισθητήρες υπερήχων (Ultrasonic) και οι αισθητήρες time of flight (ToF).

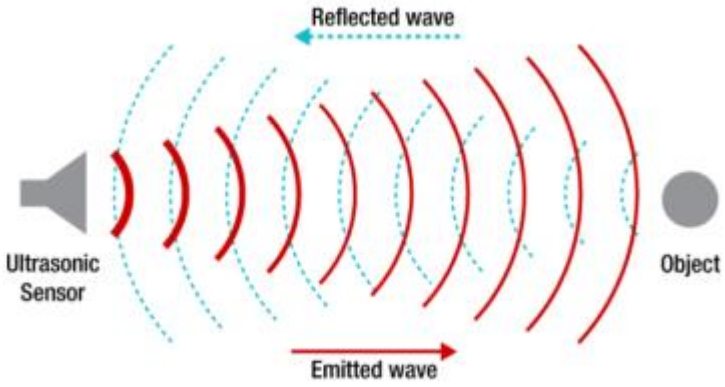
3.31 Ultrasonic Αισθητήρες

Οι αισθητήρες υπερήχων είναι αισθητήρες που χρησιμοποιούνται κυρίως για μέτρηση απόστασης και ανίχνευσης. Ο αισθητήρας λειτουργεί εκπέμποντας ένα υπερηχητικό κύμα, το οποίο αντανακλά οποιοδήποτε αντικείμενο μπροστά του. Το ανακλώμενο αυτό σήμα ανιχνεύεται από τον αισθητήρα και στην συνέχεια χρησιμοποιώντας τον χρόνο μεταξύ εκπομπής και λήψης, μπορούμε να υπολογίσουμε την απόσταση οποιουδήποτε αντικειμένου. Το υπερηχητικό κύμα ή ο υπέρηχος είναι ηχητικά κύματα με συχνότητες πολύ υψηλότερες από το ανώτερο ακουστικό όριο της ανθρώπινης ακοής. Ο άνθρωπος έχει την ικανότητα να ακούει ήχους με συχνότητες που κυμαίνονται σε εύρος από 20 Hz έως 20 KHz, ενώ τα ηχητικά κύματα που εκπέμπονται από τον αισθητήρα έχουν συχνότητα 40 KHz.



Εικόνα 1.3 Διάγραμμα εύρους υπερήχων

Παρά το γεγονός ότι οι αισθητήρες υπερήχων μπορούν να χρησιμοποιηθούν σε πολλές εφαρμογές, έχουν και κάποια μειονεκτήματα. Δεδομένου ότι οι υπέρηχοι λειτουργούν με ηχητικό σήμα, είναι τελείως ακατάλληλοι για εφαρμογές στο κενό, όπου δεν υπάρχει αέρας για να ταξιδέψει ο ήχος. Είναι επίσης ακατάλληλοι για υποβρύχιες εφαρμογές. Ακόμη η ακρίβεια τους μπορεί να μειωθεί από μαλακά υλικά όπως το ύφασμα που απορροφάν τα ηχητικά κύματα, καθώς και από αλλαγές στην θερμοκρασία ύψους 5-10 βαθμών ή και περισσότερων. Τέλος, ένα ακόμα ελάττωμα του αισθητήρα είναι το περιορισμένο εύρος ανίχνευσης.



Εικόνα 1.4 Ηχητικά κύματα αισθητήρα υπερήχων

3.32 Ultrasonic sensor HC-SR04

Το HC-SR04 είναι ένας αισθητήρας εμβέλειας υπερήχων. Κάθε μονάδα περιλαμβάνει έναν πομπό, έναν δέκτη και ένα κύκλωμα ελέγχου. Οι αισθητήρες αυτοί μας εξυπηρετούν για την ανίχνευση αντικειμένων, ώστε να υπολογίσει το ρομπότ μας την απόσταση τους από αυτό καθώς και να τα αποφύγει. Η απόσταση των αντικειμένων υπολογίζεται βάση του χρόνου μεταξύ της εκπομπής και λήψης του ηχητικού σήματος.

Εικόνα 4.3 Μπροστά και πίσω όψη αισθητήρα υπερήχων HC-SR04

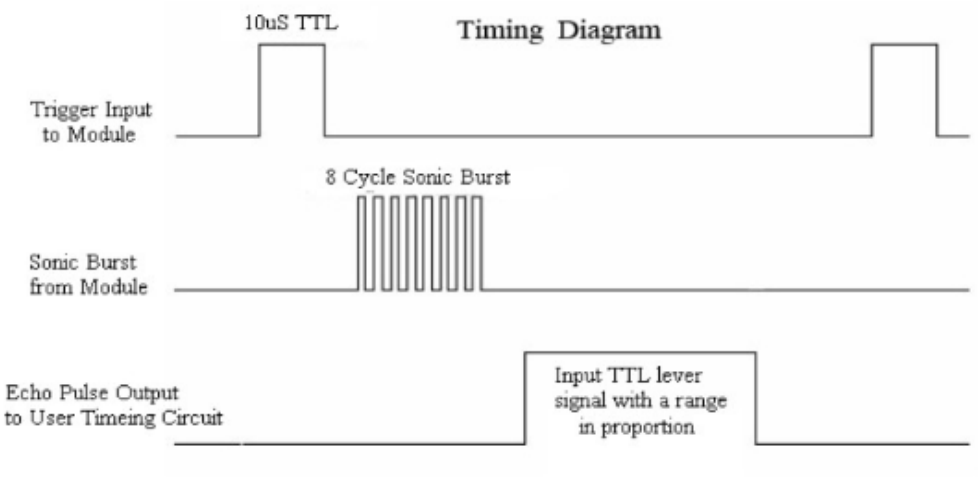
Μερικά από τα πλεονεκτήματα που έχει ένας αισθητήρας HC-SR04 είναι τα εξής:

- Χαμηλό κόστος
- Το πολύ μικρό βάρος του
- Η σχετική ακρίβεια μετρήσεων
- Εύκολη χρήση
- Ικανότητα λειτουργίας σε περιβάλλον με καπνό

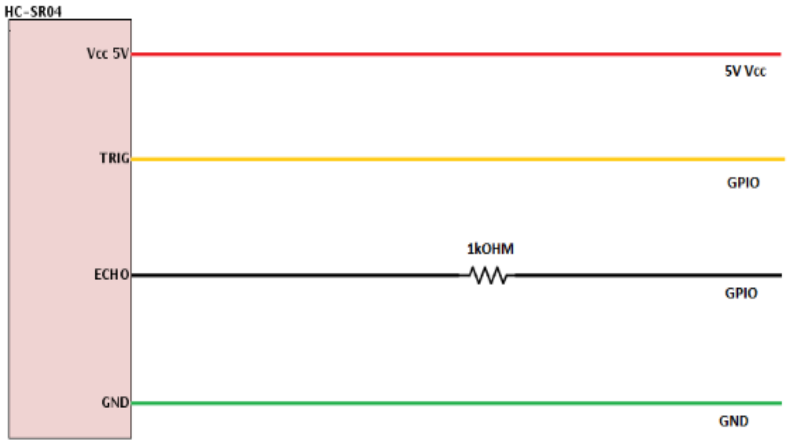
Τα τεχνικά χαρακτηριστικά του αισθητήρα HC-SR04 είναι τα εξής:

- Power Supply: +5V DC
- Quiescent Current: <2mA
- Working Current: <15mA
- Effectual Angle: <15º
- Ranging Distance: 2-400 cm
- Resolution: 0.3cm
- Measuring Angle: 30º
- Trigger Input Pulse Width: 10uS
- Dimension: 45mm x 20mm x 15mm

- Weight: approx.. 10g



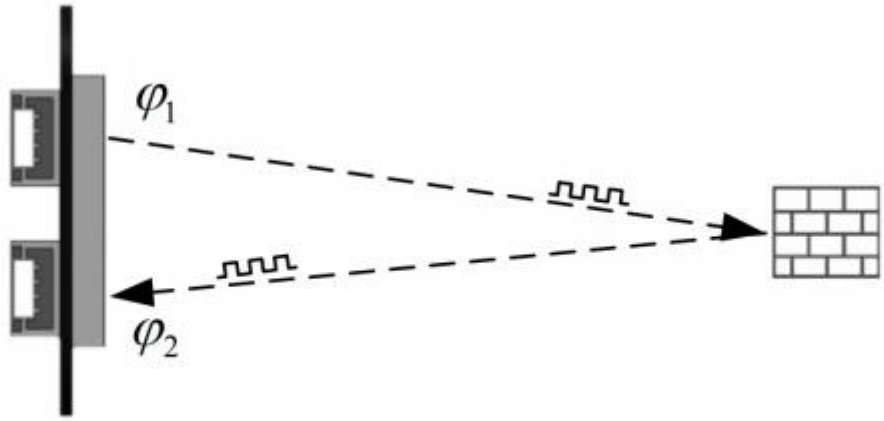
Εικόνα 4.4 Διάγραμμα λειτουργίας εκπομπής/λήψης αισθητήρα υπερήχων HC-SR04



Εικόνα 4.5 Pins αισθητήρα υπερήχων HC-SR04

3.33 TOF Αισθητήρες

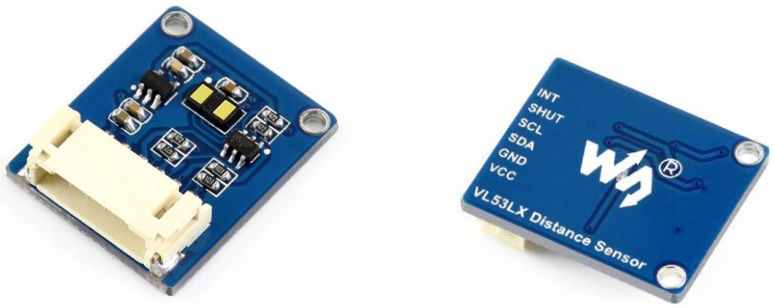
Το TOF (Time of flight) είναι μια τεχνολογία ανίχνευσης απόλυτης απόστασης, δηλαδή ο αισθητήρας εκπέμπει υπέρυθρο φως το οποίο θα ανακλαστεί μόλις συναντήσει αντικείμενο. Ο αισθητήρας υπολογίζει την απόσταση του αντικειμένου μέσω της διαφοράς χρόνου ή της διαφοράς φάσης για την εκπομπή και την ανάκλαση του φωτός και έτσι παράγει την πληροφορία. Οι αισθητήρες TOF χρησιμοποιούνται σε πολλές εφαρμογές λόγω των πλεονεκτημάτων τους όπως την λειτουργία και την ακρίβεια σε μεγάλες αποστάσεις. Ως εκ τούτου, εφαρμόζονται πάντα στην ρομποτική αποφυγή εμποδίων, στην αυτόματη εστίαση των καμερών κ.λπ.



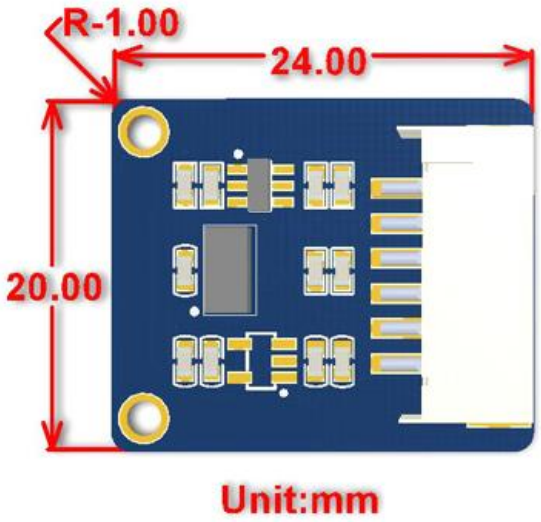
Εικόνα 1.5 Μετάδοση και λήψη αισθητήρα TOF

3.34 Waveshare VL53L1X

Το Waveshare VL53L1X είναι ουσιαστικά ένα μικροσκοπικό αυτόνομα σύστημα lidar που διαθέτει ενσωματωμένο λέιζερ 940nm κατηγορίας 1, το οποίο είναι αόρατο και ασφαλές για τα μάτια. Σε αντίθεση με τους συμβατικούς αισθητήρες υπέρυθρων που χρησιμοποιούν την ένταση του ανακλώμενου φωτός για να εκτιμήσουν την απόσταση από ένα αντικείμενο, το VL53L1X χρησιμοποιεί την τεχνολογία FlightSense της ST για να μετρήσει με ακρίβεια πόσο χρόνο χρειάζονται οι εκπεμπόμενοι παλμοί υπέρυθρου φωτός λέιζερ για να φτάσουν στο πλησιέστερο αντικείμενο και να ανακληθούν πίσω. Στην δική μας κατασκευή θα χρησιμοποιηθεί ακριβώς όπως ο αισθητήρας HC-SR04 για ανίχνευση αντικειμένων, ώστε να γίνει η σύγκριση μεταξύ των δύο.



Εικόνα 4.6 Εμπρός και πίσω όψη αισθητήρα Waveshare VL53L1X



Εικόνα 4.8 Διαστάσεις αισθητήρα Waveshare VL53L1X

Τα τεχνικά χαρακτηριστικά του Waveshare VL53L1X είμαι τα εξής:

- Dimensions: 0.5″ × 0.7″ × 0.085″ (13 mm × 18 mm × 2 mm)
- Weight without header pins: 0.5 g (0.02 oz)
- Operating Voltage: 2.6 V to 5.5 V

- Supply current: ~15 mA (typical average during active ranging at max sampling rate)
    - Varies with configuration, target, and environment; peak current can reach 40 mA
- Fast and accurate ranging with three distance mode options:
    - Short: up to ~130 cm, 50 Hz max sampling rate; this mode is the most immune to interference from ambient light
    - Medium: up to ~300 cm in the dark, 30 Hz max sampling rate
    - Long: up to 400 cm in the dark, 30 Hz max sampling rate
- Minimum range: 4 cm (objects under this range are detected, but measurements are not accurate)
- Emitter: 940 nm invisible Class 1 VCSEL (vertical cavity surface-emitting laser) – eye-safe
- Detector: 16×16 SPAD (single photon avalanche diode) receiving array with integrated lens
    - Typical full field of view (FoV): 27°
    - Programmable region of interest (ROI) size on the receiving array, allowing the sensor FoV to be reduced
    - Programmable ROI position on the receiving array, allowing multizone operation control from the host
- Configurable detection interrupt thresholds for implementing autonomous low-power presence detection:
    - target closer than threshold
    - target farther than threshold
    - target within distance window
    - target outside of distance window
    - no target
- Output format (I²C): 16-bit distance reading (in millimeters)

Λόγω του μειωμένου κόστους παραγωγής, οι αισθητήρες τύπου lidar γίνονται όλο και πιο δημοφιλής για εφαρμογές χαρτογράφησης. Οι lidar αισθητήρες χρησιμοποιούν μια οπτική αρχή μέτρησης στα τριγύρω εμπόδια και μεταδίδουν αυτές της πληροφορίες σε κάτι που είναι γνωστό ως σημεία σύννεφου. Το πλέγμα κατάληψης είναι μια αναπαράσταση του περιβάλλοντος βάσει των ελεύθερων περιοχών και την παρουσία εμποδίων. Αισθητήρες τύπου lidar και ultrasonic θεωρούνται κατάλληλοι για χαρτογράφηση πλέγματος, λόγω της υψηλής γωνιακής ανάλυσης τους, ωστόσο σε καταστάσεις δύσκολων καιρικών συνθηκών οι μετρήσεις τους δεν θεωρούνται ακριβείς και κατ’ επέκταση δεν μπορεί ένα αυτόνομο όχημα να βασιστεί μόνο σε αυτά για μια ακριβή χαρτογράφηση.

Κεφάλαιο 4.

Υλοποίηση του πλέγματος κατάληψης

4.1 Εισαγωγή

Στόχος μας ήταν η ανάπτυξη ενός αλγορίθμου χαρτογράφησης με βάση αισθητήρες απόστασης χαμηλού κόστους, που είναι ικανός να εφαρμοστεί σε οχήματα διαφορικής οδήγησης. Επιλέξαμε να κινηθούμε πειραματικά για την αντιμετώπιση αυτού του ζητήματος. Το πρώτο βήμα ήταν η επιλογή κατάλληλου αλγόριθμου. Ο αλγόριθμος που κατασκευάσαμε είναι βασισμένος στον αλγόριθμο 1.1 και 1.2 του κεφαλαίου 1 και χρησιμοποιεί την τεχνική online χαρτογράφησης πλέγματος κατάληψης που βασίζεται σε ένα μοντέλο αντίστροφου αισθητήρα. Η εφαρμογή λειτουργεί ως εξής: Πρώτα δημιουργεί ένα χάρτη πλέγματος με απόλυτη αβεβαιότητα για τη δομή του. Μετά περιμένει οποιοδήποτε μήνυμα περιέχει δεδομένα για το ρομπότ. Αφού ξεκαθαριστεί η θέση του ρομπότ, αρχίζει την διαδικασία ενημέρωσης του χάρτη. Σε κάθε μετατόπιση του ρομπότ νέες πληροφορίες λαμβάνονται από τους αισθητήρες του για το περιβάλλον και βάση αυτών γίνεται αναθεώρηση και ενημέρωση του χάρτη. Η διαδικασία αυτή συνεχίζεται για όσο χρόνο χρειάζεται το ρομπότ να καλύψει ολόκληρη την περιοχή που του θέσαμε.

Επόμενο βήμα ήταν η επιλογή κατάλληλων αισθητήρων. Επιλέξαμε να πραγματοποιήσουμε δύο διαφορετικά πειράματα χαρτογράφησης πλέγματος κατάληψης χρησιμοποιώντας διαφορετικούς αισθητήρες χαμηλού κόστους με σκοπό να γίνει μια σύγκριση μεταξύ των δύο πειραμάτων ως προς την ακρίβεια και την λειτουργικότητα. Ο πρώτος αισθητήρας που επιλέχθηκε ήταν τύπου ultrasonic, ενώ ο δεύτερος αισθητήρας ήταν τύπου lidar.

Τέλος κατασκευάσαμε ένα τεχνητό περιβάλλον με εμπόδια για να δοκιμάσουμε και τους δύο αισθητήρες κάτω από ακριβώς ίδιες συνθήκες .Η επεξεργασία των δεδομένων για την θέση του ρομπότ, καθώς και για τις πληροφορίες που παράγονται από τους αισθητήρες και στα δύο σενάρια πραγματοποιήθηκε από την ίδια πλακέτα Raspberry Pi και εφαρμόζονται στον ίδιο αλγόριθμο γραμμένο σε γλώσσα προγραμματισμού Python.

Πέρα από το να διαπιστώσουμε ποιος αισθητήρας είναι καταλληλότερος, στόχος μας ήταν και να μάθουμε εάν μια τέτοιου τύπου κατασκευή θα μπορούσε να συμβαδίσει με τα ήδη υπάρχοντα εξειδικευμένα συστήματα χαρτογράφησης και να έχει εφαρμογή σε πραγματικές συνθήκες.

4.1 Software

Για τα πειράματα μας δημιουργήσαμε και δοκιμάσαμε δύο διαφορετικά προγράμματα βασισμένα στον ίδιο αλγόριθμο χαρτογράφησης, με μοναδικές διαφορές τις βιβλιοθήκες και τα κομμάτια κώδικα που χρειάζεται ο κάθε αισθητήρας για να λειτουργήσει και να μεταφράσει τα δεδομένα του σε χρήσιμες για τον αλγόριθμο πληροφορίες.

4.1.1 Software για Lidar sensor VL53L1X

Το πρώτο βήμα ήταν να δηλώσουμε τις κατάλληλες βιβλιοθήκες, όπως αυτές που ενεργοποιούν τα σέρβο, το plot του matlab, τα GPIO και την επικοινωνία με τον αισθητήρα VL53L1X.

```
from gpiozero import Servo
import math
from time import sleep
import RPi.GPIO as GPIO
import time
from gpiozero.pins.pigpio import PiGPIOFactory
import numpy as np
import matplotlib.pyplot as plt
import VL53L1X
import os
from time import sleep
```

Επόμενο βήμα ήταν να κάποιες δηλώσεις, καθώς και αρχικοποιήσεις των μεταβλητών που θα χρησιμοποιήσουμε αργότερα.

```
os.system('i2cdetect -y 1')
sleep(0.1)
os.system('i2cdetect -y 1')

GPIO.setmode(GPIO.BCM)
factory = PiGPIOFactory()
tof = VL53L1X.VL53L1X(i2c_bus=1, i2c_address=0x29)

tof.open()

servo = Servo(12, min_pulse_width=0.5/1000, max_pulse_width=2.5/1000, pin_factory=factory)
servo.value = (-1)
i = 1
meas = []
j = 1
```

Στην συνέχεια φτιάξαμε την λογική του αλγορίθμου. Επιλέξαμε να τα εντάξουμε όλα σε μία κλάση για λόγους καλύτερης οργάνωσης και ανάγνωσης.

```
class OccMap():
    def __init__(self, sizeX, sizeY, sizeGrid):

        self.obj = 1.0 # Paxos antikeimenwn
        self.sensW = 12*np.pi/180.0 # Platos anixneusis aisthitira
        self.sensR = 200.0 # Megisti emvelia aisthitira


        self.sizeX = sizeX+2 # +2 kelia gia ta oria
        self.sizeY = sizeY+2 # +2 kelia cells gia ta oria
        self.sizeGrid = sizeGrid # Klimaka
        self.log_map_prob = np.zeros((self.sizeX, self.sizeY)) # Midenismos pinaka



        # Orismos pithanotitwn gia gemato/adeio keli
        self.l_occ = np.log(0.65/0.35)
        self.l_free = np.log(0.35/0.65)

        # katanomi x,y thesewn gia grid
        self.grid_position_m = np.array([np.tile(np.arange(0, self.sizeX*self.sizeGrid, self.sizeGrid)[:,None], (1, self.sizeY)),
                                         np.tile(np.arange(0, self.sizeY*self.sizeGrid, self.sizeGrid)[:,None].T, (self.sizeX, 1))])



    def map_update(self, pose,z):

        dx = self.grid_position_m.copy()
        dx[0, :, :] -= pose[0] # Pinakas suntetagmenwn x apo keli
        dx[1, :, :] -= pose[1] # Pinakas suntetagmenwn y apo keli
        thetaGrid = np.arctan2(dx[1, :, :], dx[0, :, :]) - pose[2]

        grid_dist = np.linalg.norm(dx, axis=0) # Apostaseis keliwn apo robot

        for i in range(len(z)):
            r = z[i]

            # Ypologismos free/occ keliwn
            free_cell = (np.abs(thetaGrid) <= self.sensW/2.0) & (grid_dist < (r - self.obj/2.0))
            occ_cell = (np.abs(thetaGrid) <= self.sensW/2.0) & (np.abs(grid_dist - r) <= self.obj/2.0)

            # Diamorfosi keliwn
            self.log_map_prob[occ_cell] += self.l_occ
            self.log_map_prob[free_cell] += self.l_free
```

Στην αρχή της κλάσης δίνουμε κάποιες βασικές πληροφορίες όπως την εμβέλεια του αισθητήρα, το πλάτος ανίχνευσης και το πάχος των αντικειμένων. Μετά του δίνουμε της πληροφορίες για να δημιουργήσει έναν μηδενισμένο πίνακα XY δικών μας διαστάσεων. Στην συνέχεια ορίζουμε την πιθανότητα ένα κελί να είναι γεμάτο ή άδειο και κάνουμε μια προ κατανομή όλων των θέσεων x, y για το πλέγμα.

Στην συνάρτηση map_update γίνεται η ενημέρωση του χάρτη που δημιουργούμε σε κάθε μετατόπιση της κατασκευής. Οι πληροφορίες για το αν ένα κελί είναι κατειλημμένο ή όχι λαμβάνονται από μια λίστα που δημιουργούμε μέσα στο κεντρικό κομμάτι του προγράμματος, σύμφωνα με πληροφορίες που λαμβάνουμε από τους αισθητήρες.

```
## MAIN PROGRAM

# Epilogi klimakas (80x120)
sizeGrid = 1.0
rep = 1
sizeY = int(80/sizeGrid)
sizeX = int(120/sizeGrid)
map = OccMap(sizeX, sizeY, sizeGrid)
plt.ion() #Epilogi real-time plotting
plt.figure(1)#Dimiourgia plot

while rep == 1:

    servo.value = (-1)
    k = 3
    posx = input("Enter x value: ")
    posy = input("Enter y value: ")
    state = [[0 for x in range(34)] for y in range(3)] #Dimiourgia pinaka thesewn simulation
    for i in range(34):
        state[0][i] = int(posy)
        state[1][i] = int(posx)
        state[2][i] = k
        k = k - 0.0875
    state = np.array(state)
    t = 0
    for i in (range(len(state.T))):
        meas=[]
        while j <= 10:
            tof.start_ranging(2)
            sleep(0.01)
            distance_in_mm = tof.get_distance()
            dist = distance_in_mm/10
            print(dist)
            j = j +1
            meas.append(dist)
        j = 1
        servo.value = servo.value + 0.0575
        sleep(0.005)

        map.map_update(state[:,i],meas) # Enimerosi xarti
```

Στο βασικό κομμάτι του προγράμματος αφού δώσουμε τις πληροφορίες για τις διαστάσεις του χάρτη μας, δημιουργούμε έναν πίνακα θέσεων. Ο πίνακας αυτός ζητάει από τον χρήστη να του δώσει τις συντεταγμένες της κατασκευής και μετά δημιουργεί τις κατάλληλες τιμές ώστε να εκτελεί μισή περιστροφή γύρω από τον εαυτό της. Στην συνέχεια μια λίστα με μετρήσεις από τον αισθητήρα δημιουργείται για κάθε αλλαγή γωνίας και καλείται η συνάρτηση update_map ώστε να ενημερωθεί ο χάρτης μας ανάλογα με τις νέες αυτές πληροφορίες.

Τελευταίο βήμα είναι η δημιουργία ενός plot με αυτές τις πληροφορίες, ώστε να υπάρξει μια γραφική αναπαράσταση του χάρτη μας. Μετά την εκτέλεση της χαρτογράφησης μια

ερώτηση γίνεται στον χρήστη για το εάν θέλει να συνεχίσει την χαρτογράφηση μετατοπίζοντας την βάση σε νέα θέση.

```
        # Real-Time Plotting
        plt.clf()
        plt.xlim(0, 80)
        plt.ylim(0, 120)
        pose = state[:,i]
        circle = plt.Circle((pose[1], pose[0]), radius=5.0, fc='b')
        plt.gca().add_patch(circle)
        arrow = pose[0:2] + np.array([3.5, 0]).dot(np.array([[np.cos(pose[2]), np.sin(pose[2])], [-np.sin(pose[2]), np.cos(pose[2])]]))
        plt.plot([pose[1], arrow[1]], [pose[0], arrow[0]])
        plt.imshow(1.0 - 1./(1.+np.exp(map.log_map_prob)), 'Greys')
        plt.pause(0.2)

    rep = int(input("Continue on new location? (0-No, 1-Yes) "))
```

4.1.2 Software για Ultrasonic sensor HC-SR04

Όπως και με τον lidar αισθητήρα, η διαδικασία ξεκινάει με τον ίδιο τρόπο και για τον αισθητήρα υπερήχων, δηλαδή δηλώνοντας τις κατάλληλες βιβλιοθήκες.

```
from gpiozero import Servo
import math
from time import sleep
import RPi.GPIO as GPIO
import time
from gpiozero.pins.pigpio import PiGPIOFactory
import numpy as np
import matplotlib.pyplot as plt
```

Έπειτα κάνουμε κάποιες δηλώσεις για τα pins του raspberry pi, καθώς και κάποιες αρχικοποιήσεις μεταβλητών.

```
GPIO.setmode(GPIO.BCM)
factory = PiGPIOFactory()

#set GPIO Pins
GPIO_TRIGGER = 18
GPIO_ECHO = 24

#set GPIO direction (IN / OUT)
GPIO.setup(GPIO_TRIGGER, GPIO.OUT)
GPIO.setup(GPIO_ECHO, GPIO.IN)

servo = Servo(12, min_pulse_width=0.5/1000, max_pulse_width=2.5/1000, pin_factory=factory)
servo.value = (-1)
i = 1
distfull = 0
meas = []
j = 1
```

Η κλάση OccMap( ) μένει σκόπιμα ίδια με αυτήν που χρησιμοποιήσαμε στο προηγούμενο πείραμα για να μπορέσει να γίνει η σύγκριση των αποτελεσμάτων μεταξύ των δύο αισθητήρων.

```
class OccMap():
    def __init__(self, sizeX, sizeY, sizeGrid):

        self.obj = 1.0 # Paxos antikeimenwn
        self.sensW = 12*np.pi/180.0 # Platos anixneusis aisthitira
        self.sensR = 200.0 # Megisti emvelia aisthitira


        self.sizeX = sizeX+2 # +2 kelia gia ta oria
        self.sizeY = sizeY+2 # +2 kelia cells gia ta oria
        self.sizeGrid = sizeGrid # Klimaka
        self.log_map_prob = np.zeros((self.sizeX, self.sizeY)) # Midenismos pinaka



        # Orismos pithanotitwn gia gemato/adeio keli
        self.l_occ = np.log(0.65/0.35)
        self.l_free = np.log(0.35/0.65)

        # katanomi x,y thesewn gia grid
        self.grid_position_m = np.array([np.tile(np.arange(0, self.sizeX*self.sizeGrid, self.sizeGrid)[:,None], (1, self.sizeY)),
                                         np.tile(np.arange(0, self.sizeY*self.sizeGrid, self.sizeGrid)[:,None].T, (self.sizeX, 1))])



    def map_update(self, pose,z):

        dx = self.grid_position_m.copy()
        dx[0, :, :] -= pose[0] # Pinakas suntetagmenwn x apo keli
        dx[1, :, :] -= pose[1] # Pinakas suntetagmenwn y apo keli
        thetaGrid = np.arctan2(dx[1, :, :], dx[0, :, :]) - pose[2]

        grid_dist = np.linalg.norm(dx, axis=0) # Apostaseis keliwn apo robot

        for i in range(len(z)):
            r = z[i]

            # Ypologismos free/occ keliwn
            free_cell = (np.abs(thetaGrid) <= self.sensW/2.0) & (grid_dist < (r - self.obj/2.0))
            occ_cell = (np.abs(thetaGrid) <= self.sensW/2.0) & (np.abs(grid_dist - r) <= self.obj/2.0)

            # Diamorfosi keliwn
            self.log_map_prob[occ_cell] += self.l_occ
            self.log_map_prob[free_cell] += self.l_free
```

Μια νέα συνάρτηση που ήταν απαραίτητο να προστεθεί ήταν η συνάρτηση distance( ). Η συνάρτηση αυτή δηλώνει στο πρόγραμμα μας πως να υπολογίσει την απόσταση, βάση των πληροφοριών που λαμβάνει από τον αισθητήρα υπερήχων. Συγκεκριμένα αποθηκεύει τον χρόνο που ξεκίνησε και τελείωσε η μετάδοση του κύματος, υπολογίζει την διαφορά και μετά την πολλαπλασιάζει επί την ταχύτητα του ήχου και την διαιρεί διά δύο.

```
def distance():
    # set Trigger to HIGH
    GPIO.output(GPIO_TRIGGER, True)

    # set Trigger  to LOW
    time.sleep(0.00001)
    GPIO.output(GPIO_TRIGGER, False)

    StartTime = time.time()
    StopTime = time.time()

    # apothikeusi xronoy ekinisis
    while GPIO.input(GPIO_ECHO) == 0:
        StartTime = time.time()

    # apothikeusi xronou stamatimatos
    while GPIO.input(GPIO_ECHO) == 1:
        StopTime = time.time()

    # diafora xronoy anamesa se ekinisi kai stamatima
    TimeElapsed = StopTime - StartTime
    #ypologismos apostasis
    distance = (TimeElapsed * 34300) / 2

    return distance
```

Στο βασικό μέρος του προγράμματος ακολουθούμε και πάλι ακριβώς τα ίδια βήματα με μοναδική διαφορά πως τώρα η λίστα μετρήσεων δημιουργείται βάση των πληροφοριών που λαμβάνονται από τον αισθητήρα υπερήχων και όχι από τον lidar.

```
## MAIN PROGRAM

# Epilogi klimakas (80x120)
sizeGrid = 1.0
rep = 1
sizeY = int(80/sizeGrid)
sizeX = int(120/sizeGrid)
map = OccMap(sizeX, sizeY, sizeGrid)
plt.ion() #Epilogi real-time plotting
plt.figure(1)#Dimiourgia plot

while rep == 1:

    servo.value = (-1)
    k = 3
    posx = input("Enter x value: ")
    posy = input("Enter y value: ")
    state = [[0 for x in range(34)] for y in range(3)] #Dimiourgia pinaka thesewn simulation
    for i in range(34):
        state[0][i] = int(posy)
        state[1][i] = int(posx)
        state[2][i] = k
        k = k - 0.0875
    state = np.array(state)
    t = 0



    for i in (range(len(state.T))):
        meas=[]
        while j <= 10:
            dist = distance()
            distfull = (distfull + dist)
            time.sleep(0.03)
            j = j +1
            meas.append(dist)
        distavg = distfull / 10
        print(distavg)
        j = 1
        distfull = 0
        servo.value = servo.value + 0.0575
        sleep(0.005)

        map.map_update(state[:,i],meas) # Enimerosi xarti
```

Τέλος και πάλι δημιουργούμε μια γραφική αναπαράσταση του χάρτη, καθώς και πάλι δίνουμε στον χρήστη την δυνατότητα να συνεχίσει την χαρτογράφηση από μία νέα θέση.

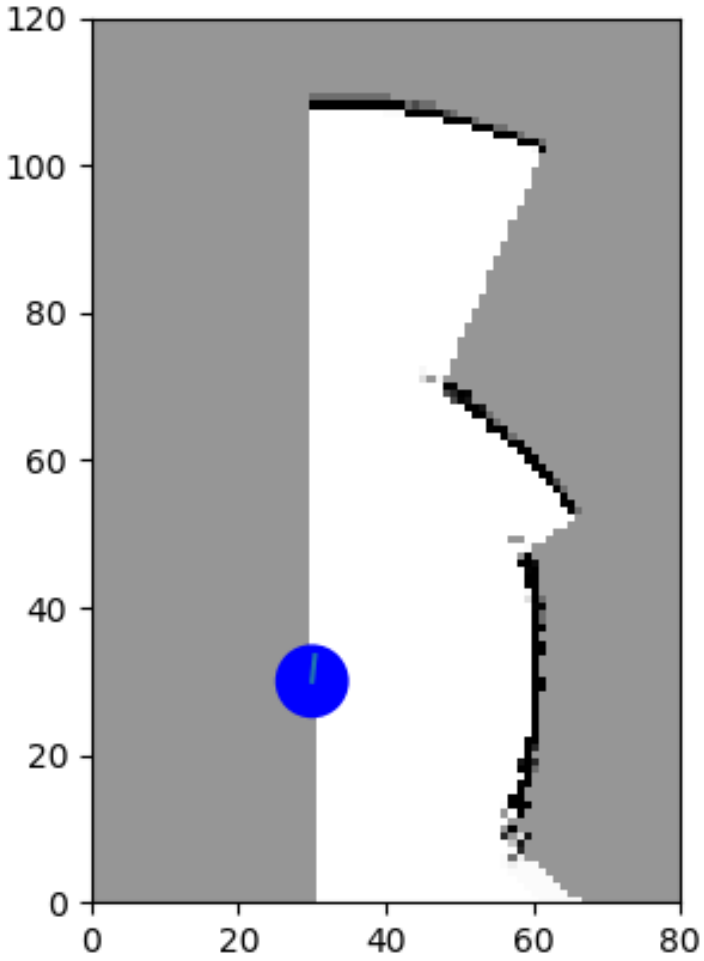
```
        # Real-Time Plotting
        plt.clf()
        plt.xlim(0, 80)
        plt.ylim(0, 120)
        pose = state[:,i]
        circle = plt.Circle((pose[1], pose[0]), radius=5.0, fc='b')
        plt.gca().add_patch(circle)
        arrow = pose[0:2] + np.array([3.5, 0]).dot(np.array([[np.cos(pose[2]), np.sin(pose[2])], [-np.sin(pose[2]), np.cos(pose[2])]]))
        plt.plot([pose[1], arrow[1]], [pose[0], arrow[0]])
        plt.imshow(1.0 - 1./(1.+np.exp(map.log_map_prob)), 'Greys')
        plt.pause(0.2)

    rep = int(input("Continue on new location? (0-No, 1-Yes) "))
```

4.2 Δοκιμές της κατασκευής με τυχαία αντικειμένα

Για τις δοκιμές μας τοποθετήσαμε αρχικά το Ultrasonic sensor πάνω στο σερβοκινητήρα, ώστε να μπορεί να το περιστρέφει και τα στηρίξαμε μαζί σε μια αυτοσχέδια βάση. Στην συνέχεια τοποθετήσαμε διάφορα αντικείμενα σε τυχαίες θέσεις πάνω σε μια ξύλινη επιφάνεια διαστάσεων 80x120. Τέλος τοποθετήσαμε και την βάση πάνω στην επιφάνεια σε μία επίσης τυχαία θέση και ξεκινήσαμε να δοκιμάζουμε τον κώδικα που αναφέραμε παραπάνω.

Εικόνα 4.11 Δοκιμή κατασκευής Ultrasonic sensor στην πρώτη θέση (x=30, y=30)

Το αποτέλεσμα της χαρτογράφησης κατά την πρώτη τοποθέτηση της βάσης στην θέση x=30, y=30 ήταν το εξής:



Εικόνα 4.12 Αποτέλεσμα χαρτογράφησης με Ultrasonic sensor τοποθετημένο στην θέση (x=30, y=30)

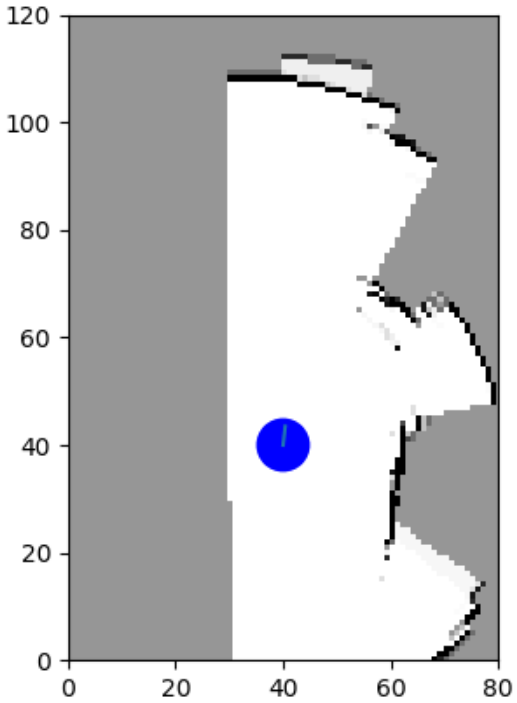
Από την εικόνα μπορούμε να δούμε πως κατά την πρώτη διαδικασία χαρτογράφησης τα αντικείμενα ανιχνεύθηκαν επιτυχώς, αλλά υπάρχουν ατέλειες, όπως το γεγονός πως το κενό μεταξύ των αντικειμένων δεν ανιχνεύθηκε σωστά.

Στην συνέχεια μεταφέραμε την βάση σε μία νέα θέση χωρίς να αγγίξουμε τα υπόλοιπα αντικείμενα και συνεχίσαμε την χαρτογράφηση από αυτό το νέο σημείο.

Εικόνα 4.13 Δοκιμή κατασκευής με Ultrasonic sensor στην δεύτερη θέση (x=40, y=40)

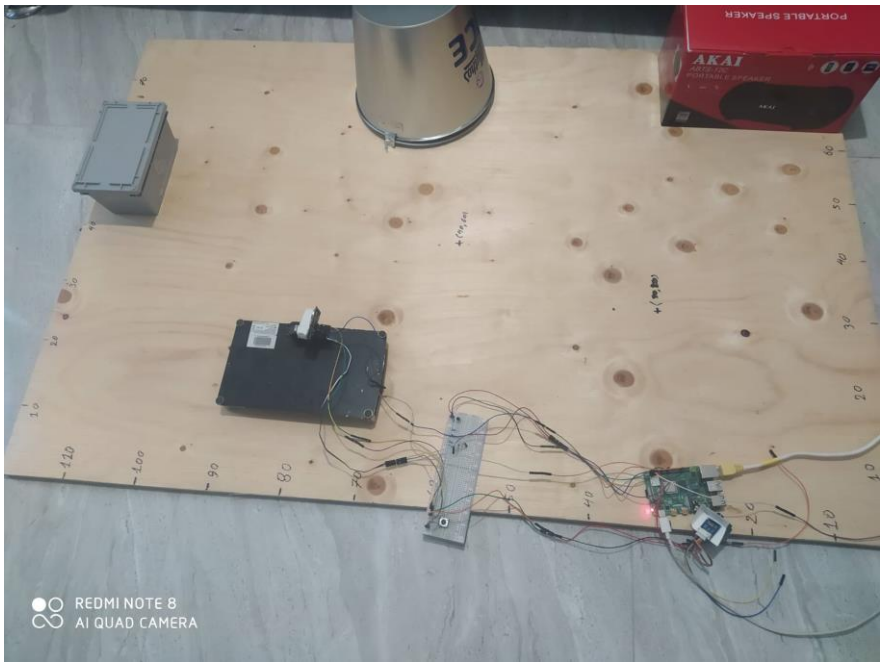
Στόχος μας ήταν να δούμε πως θα συμπεριφερθεί ο αισθητήρας ανιχνεύοντας τα ίδια αντικείμενα από διαφορετική γωνία, αν θα είναι φανερό ότι πρόκειται για τα ίδια και τι αλλαγές θα δημιουργήσει αυτό στον προηγούμενο χάρτη.

Το αποτέλεσμα της χαρτογράφησης στην δεύτερη θέση (x=40, y=40) λοιπών ήταν τα εξής:



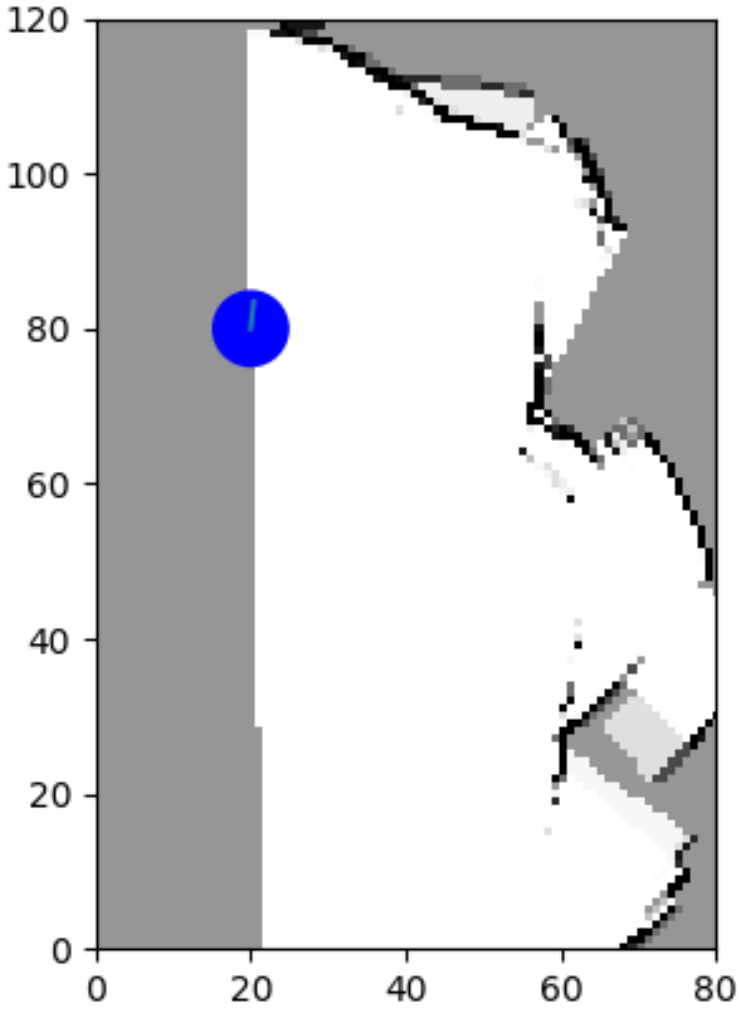
Εικόνα 4.14 Αποτέλεσμα συνέχειας χαρτογράφησης με Ultrasonic sensor τοποθετημένο στην θέση (x=40, y=40)

Επαναλάβαμε την ίδια διαδικασία μία ακόμη φορά τοποθετώντας την βάση σε νέα θέση για να έχουμε μια πιο ολοκληρωμένη εικόνα



Εικόνα 4.15 Δοκιμή κατασκευής με Ultrasonic sensor στην τρίτη θέση (x=20, y=80)

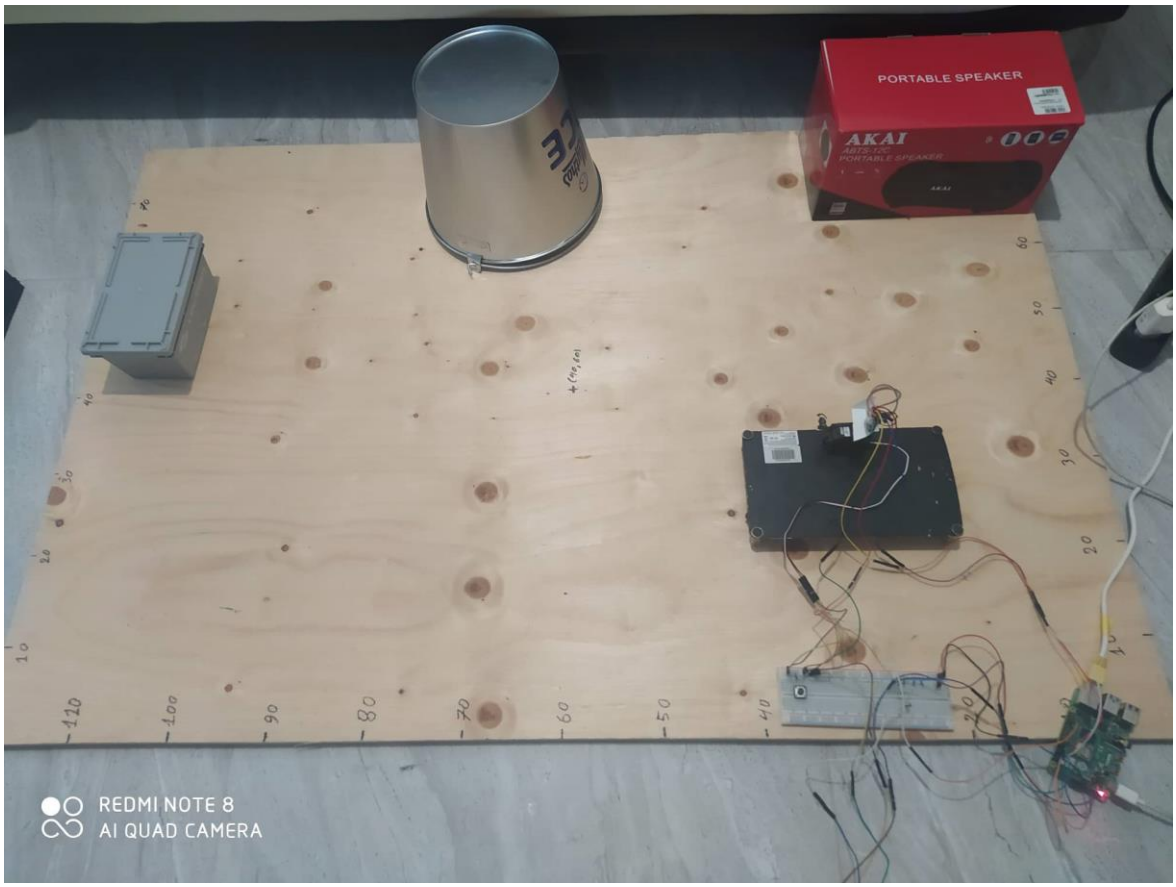
Τα αποτελέσματα της χαρτογράφησης κατά την τοποθέτηση στη τρίτη θέση (x=20, y=80) είναι τα εξής:



Εικόνα 4.16 Αποτέλεσμα συνέχειας χαρτογράφησης με Ultrasonic sensor τοποθετημένο στην θέση (x=20, y=80)

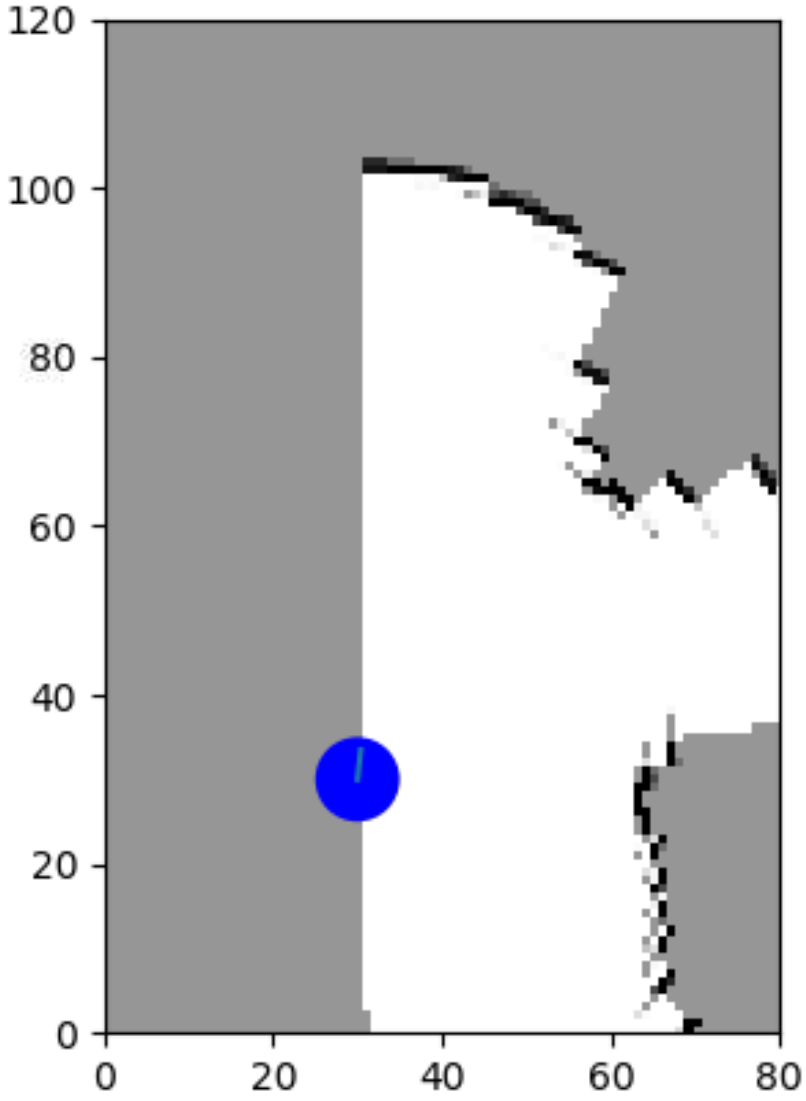
Παρόλο που τα αντικείμενα έχουν ανιχνευθεί, το κενό ανάμεσα στα αντικείμενα ακόμα και μετά από τρεις μετατοπίσεις δεν μπορεί να απεικονιστεί σωστά και υπάρχει αρκετός θόρυβος.

Στην συνέχεια αντικαταστήσαμε τον Ultrasonic αισθητήρα με τον Lidar αισθητήρα χωρίς πάλι να μετατοπίσουμε καθόλου τα αντικείμενα και επαναλάβαμε την ίδια διαδικασία, τοποθετώντας την βάση διαδοχικά στις ίδιες θέσεις με τις προηγούμενες δοκιμές



Εικόνα 4.17 Δοκιμή κατασκευής με Lidar sensor στην πρώτη θέση (x=30, y=30)

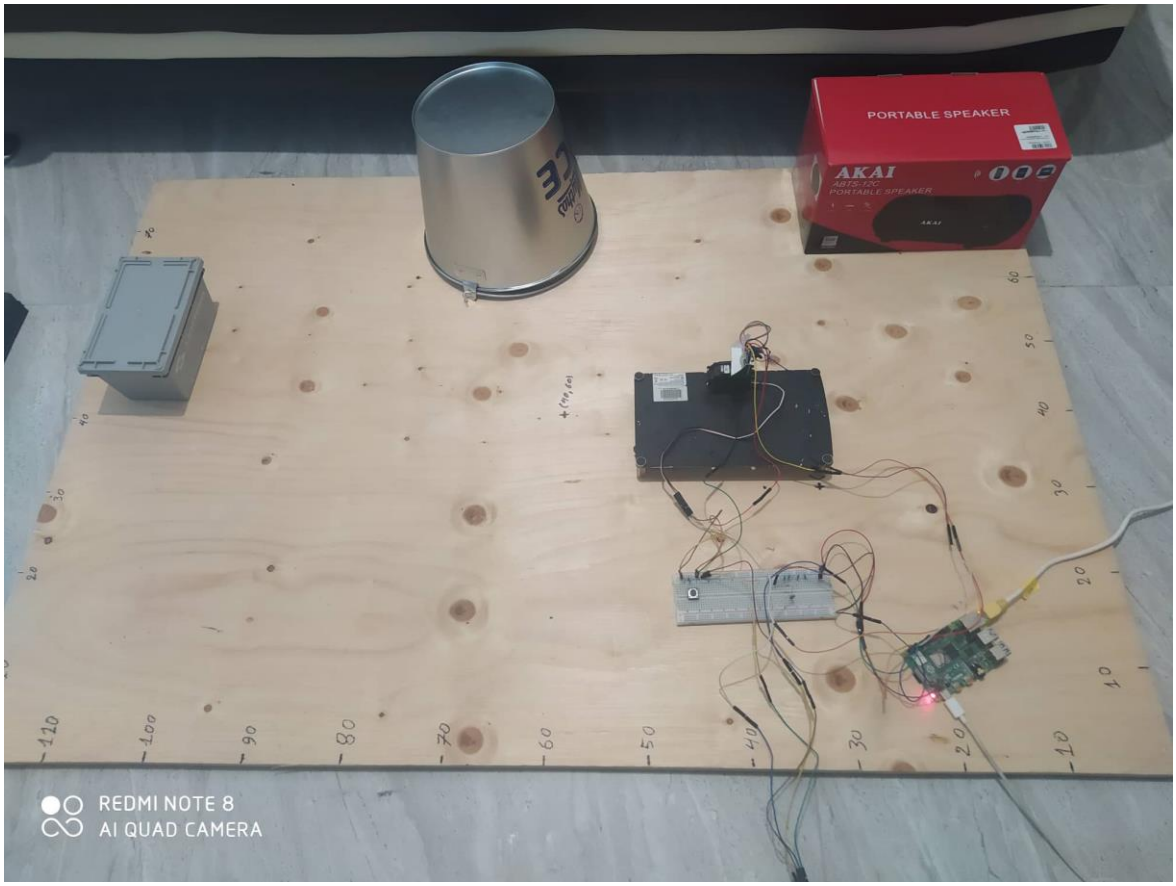
Η χαρτογράφηση που προέκυψε από την τοποθέτηση του αισθητήρα lidar VL53L1X στην πρώτη θέση (x=30, y=30) είναι η εξής:



Εικόνα 4.18 Αποτέλεσμα χαρτογράφησης με Lidar sensor τοποθετημένο στην θέση (x=30, y=30)

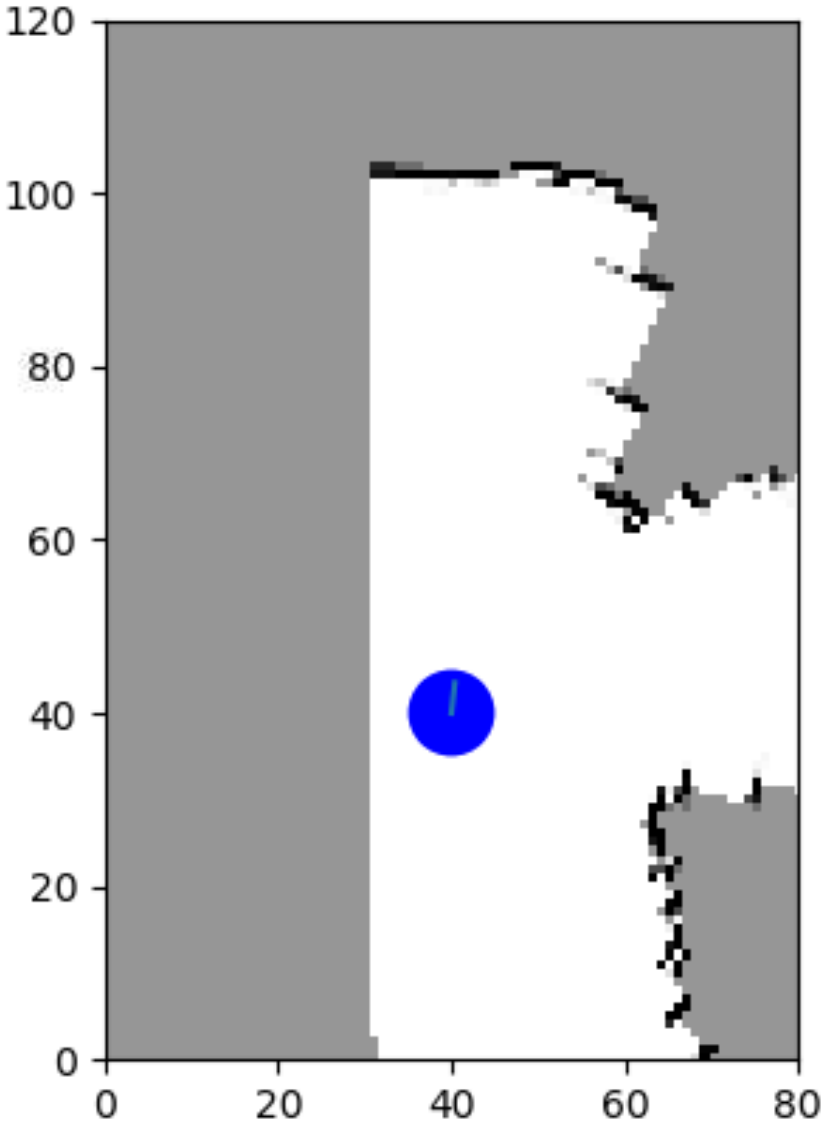
Εδώ μπορούμε ήδη να δούμε πως η πραγματική γεωμετρία των αντικειμένων ανιχνεύθηκε πολύ καλύτερα, καθώς και τα κενά μεταξύ των αντικειμένων σχεδόν σε όλες τις θέσεις.

Στην συνέχεια ακολουθώντας την ίδια διαδικασία που εκτελέσαμε και με το Ultrasonic sensor, τοποθετήσαμε την βάση στην αντίστοιχη δεύτερη θέση και συνεχίσαμε την χαρτογράφηση.



Εικόνα 4.19 Δοκιμή κατασκευής με Lidar sensor στην δεύτερη θέση (x=40, y=40)

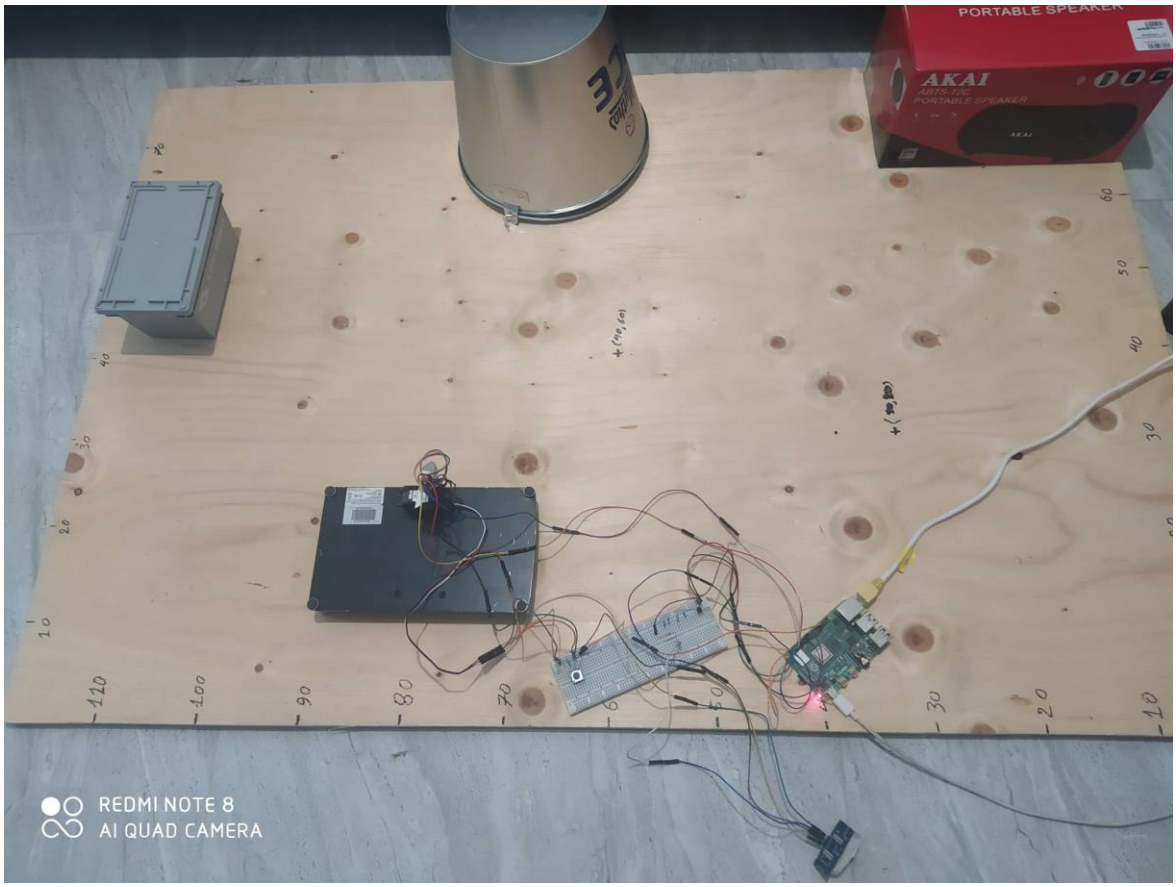
Το αποτέλεσμα της χαρτογράφησης μετά την μετατόπιση της βάσης στην δεύτερη θέση (x=40, y=40) είναι το εξής:



Εικόνα 4.20 Αποτέλεσμα συνέχειας χαρτογράφησης με Lidar sensor τοποθετημένο στην θέση (x=40, y=40)

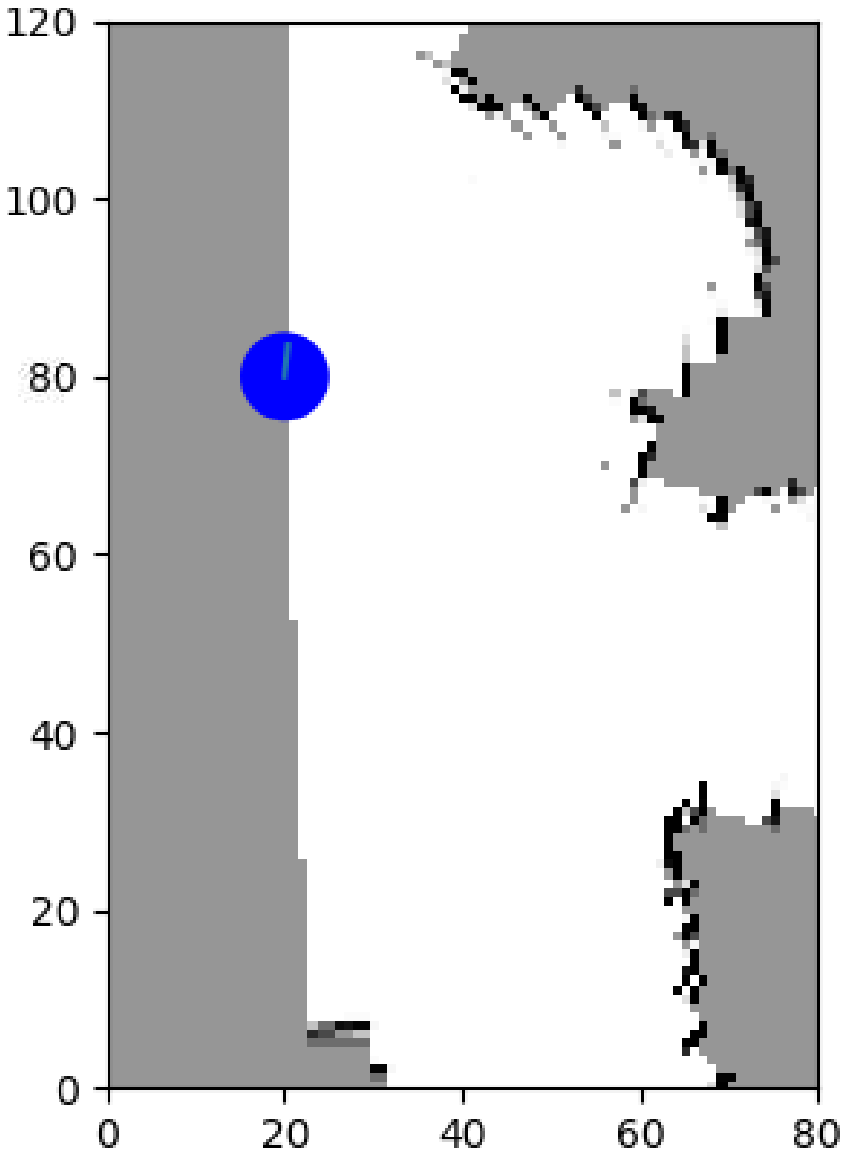
Μετά την ανανέωση του χάρτη μπορούμε να δούμε πως τα αποτελέσματα είναι πολύ κοντινά, πράγμα που σημαίνει πως η κατασκευή μας κατάφερε να αντιληφθεί επιτυχώς, πως οι μετρήσεις που έλαβε αναφέρονταν στα ίδια αντικείμενα.

Έπειτα επαναλάβαμε για ακόμη μία φορά την ίδια διαδικασία, μεταφέροντας τον Lidar αισθητήρα στην τρίτη θέση που είχαμε τοποθετήσει και τον Ultrasonic αισθητήρα και συνεχίσαμε και πάλι την χαρτογράφηση.



Εικόνα 4.21 Δοκιμή κατασκευής με Lidar sensor στην τρίτη θέση (x=20, y=80)

Το τελικό αποτέλεσμα μετά και από την τρίτη εκτέλεση της διαδικασίας χαρτογράφησης στην θέση (x=20, y=80) φαίνεται στην ακόλουθη εικόνα:

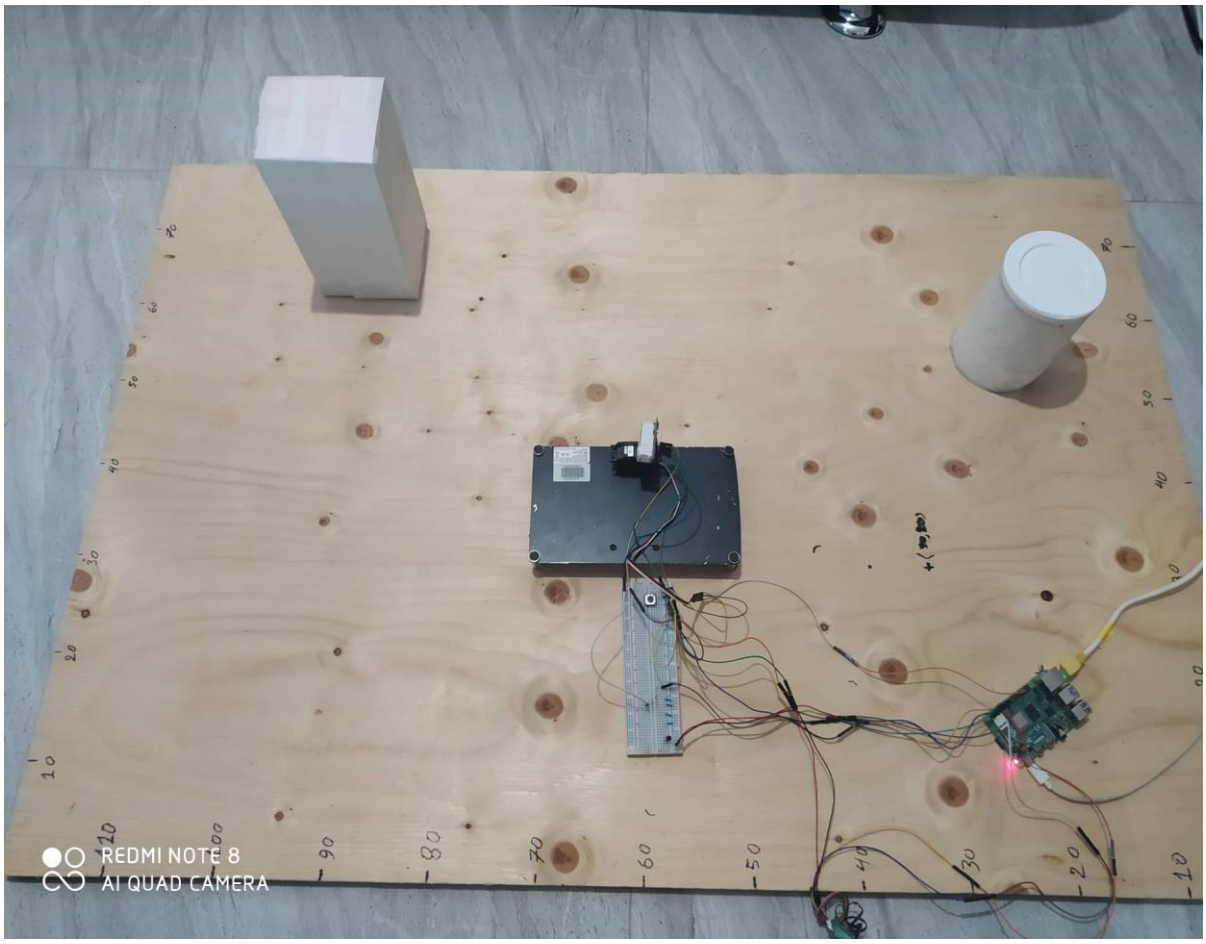


Εικόνα 4.22 Αποτέλεσμα συνέχειας χαρτογράφησης με Lidar sensor τοποθετημένο στην θέση (x=20, y=80)

Μετά και από την τελευταία μετατόπιση της κατασκευής μας μπορούμε να διαπιστώσουμε πως τα αντικείμενα ανιχνεύονται σωστά ως προς την θέση και την γεωμετρία τους, αλλά πάλι υπάρχουν ατέλειες όπως η εσφαλμένη ανίχνευση του κενού μεταξύ του κουβά και του κιβωτίου.

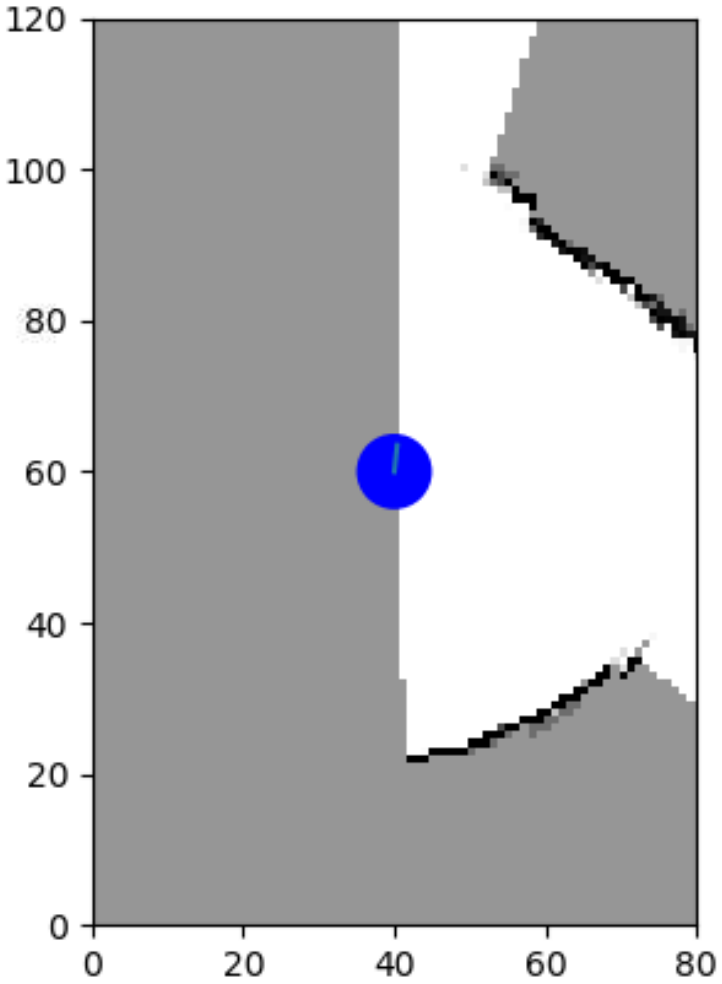
4.3 Δοκιμές της κατασκευής με επιλεγμένα αντικείμενα

Σε αντίθεση με τα παραπάνω πειράματα που εκτελέσαμε, τώρα αποφασίσαμε να βοηθήσουμε την κατασκευή μας επιλέγοντας αντικείμενα που ευνοούν τους αισθητήρες και τοποθετώντας τα σε συγκεκριμένες θέσεις και όχι τυχαίες. Έπειτα επαναλάβαμε ακριβώς τα ίδια βήματα.



Εικόνα 4.23 Δοκιμή κατασκευής Ultrasonic sensor στην πρώτη θέση (x=40, y=60)

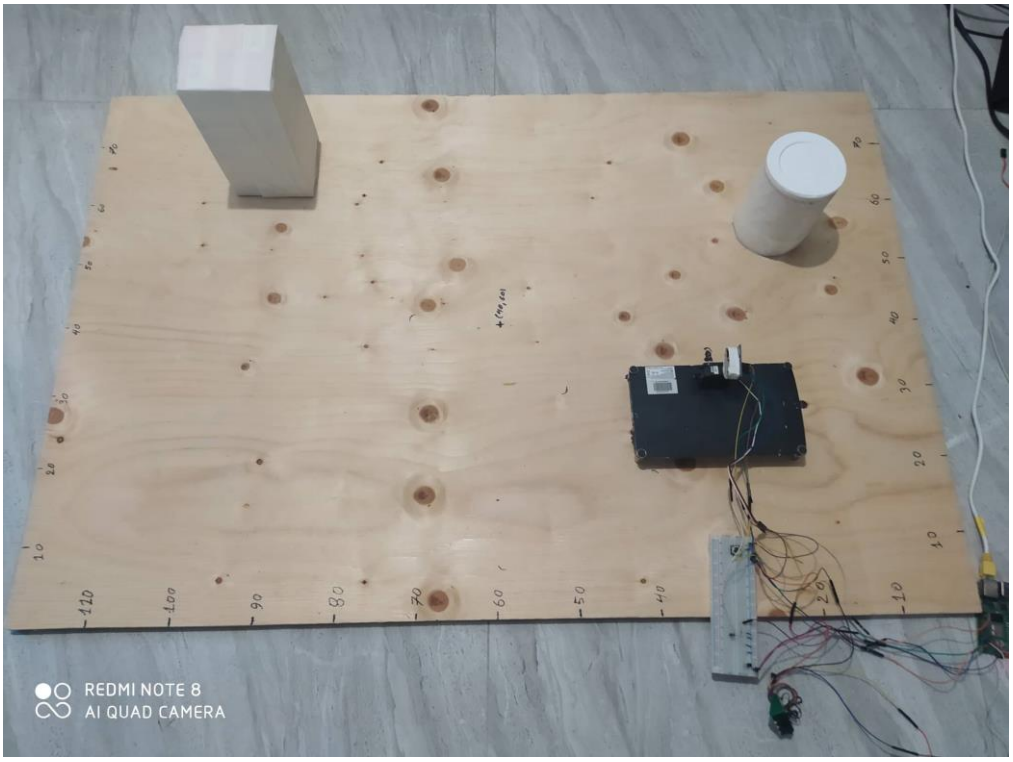
Το αποτέλεσμα της χαρτογράφησης κατά την πρώτη τοποθέτηση της βάσης στην θέση (x=40, y=60) ήταν το εξής:



Εικόνα 4.24 Αποτέλεσμα χαρτογράφησης με Ultrasonic sensor τοποθετημένο στην θέση (x=40, y=60)

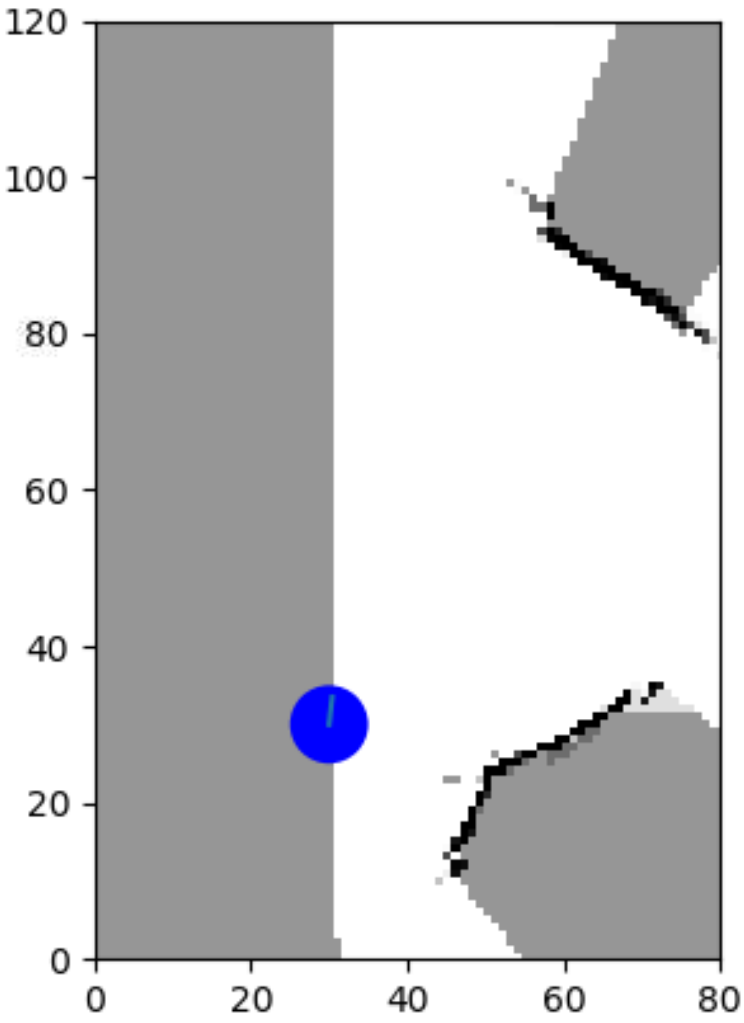
Παρατηρώντας την παραπάνω εικόνα μπορούμε να δούμε πως τα αντικείμενα ανιχνεύονται επιτυχώς, όμως υπάρχουν ατέλειες ως προς το πραγματικό τους μέγεθος και την γεωμετρία τους.

Στην συνέχεια, ακριβώς όπως την πρώτη φορά, μεταφέραμε την βάση σε μία νέα θέση χωρίς να αγγίξουμε τα υπόλοιπα αντικείμενα και συνεχίσαμε την χαρτογράφηση από αυτό το νέο σημείο.



Εικόνα 4.25 Δοκιμή κατασκευής με Ultrasonic sensor στην δεύτερη θέση (x=30, y=30)

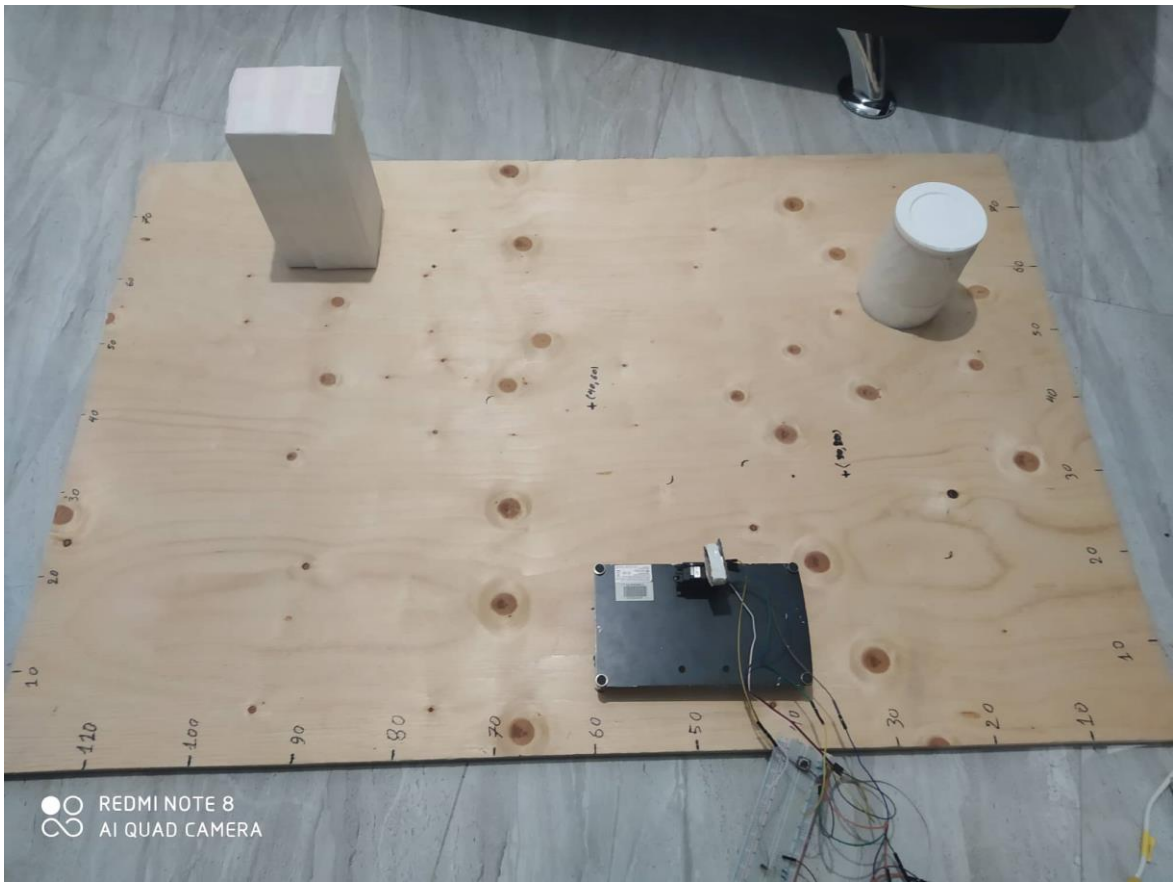
Το αποτέλεσμα της χαρτογράφησης, κατά την τοποθέτηση της κατασκευής στην δεύτερη θέση (x=30, y=30) ήταν τα εξής:



Εικόνα 4.26 Αποτέλεσμα συνέχειας χαρτογράφησης με Ultrasonic sensor τοποθετημένο στην θέση (x=30, y=30)

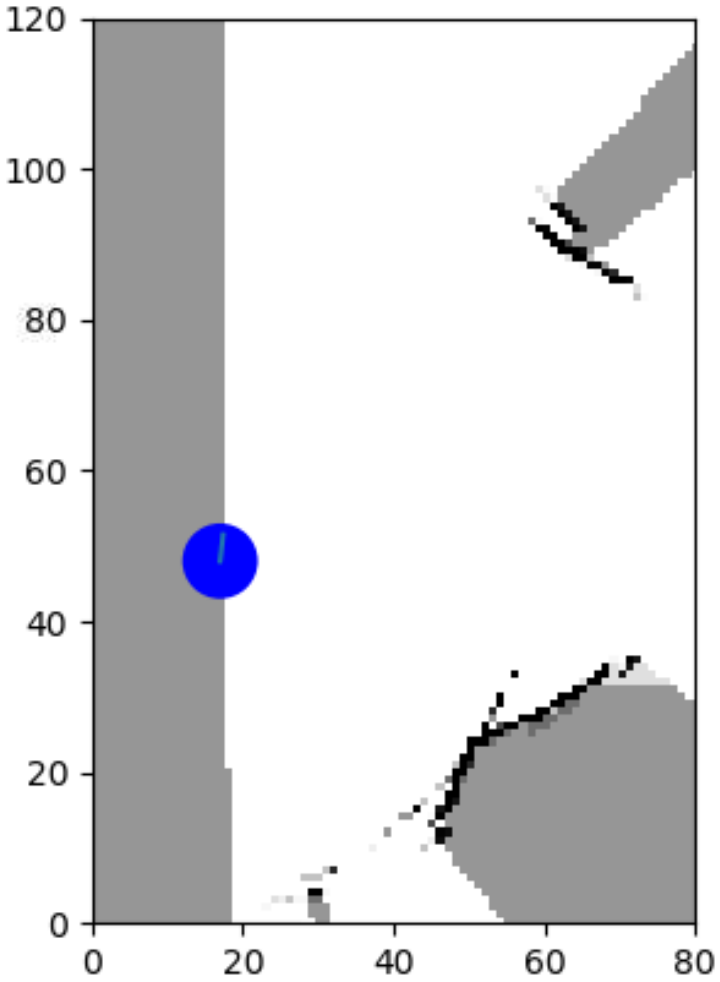
Μετά την πρώτη ανανέωση του χάρτη, στην παραπάνω εικόνα διακρίνουμε μια βελτίωση ως προς την απεικόνιση του μεγέθους των αντικειμένων, καθώς και κάποιων κενών που έπρεπε να υπάρχουν.

Τοποθετήσαμε για μια ακόμη φορά την κατασκευή μας σε νέα θέση και επαναλάβαμε την διαδικασία ελπίζοντας να έχουμε καλύτερα τελικά αποτελέσματα από την πρώτη δοκιμή με τον ultrasonic sensor.



Εικόνα 4.27 Δοκιμή κατασκευής με Ultrasonic sensor στην τρίτη θέση (x=17, y=48)

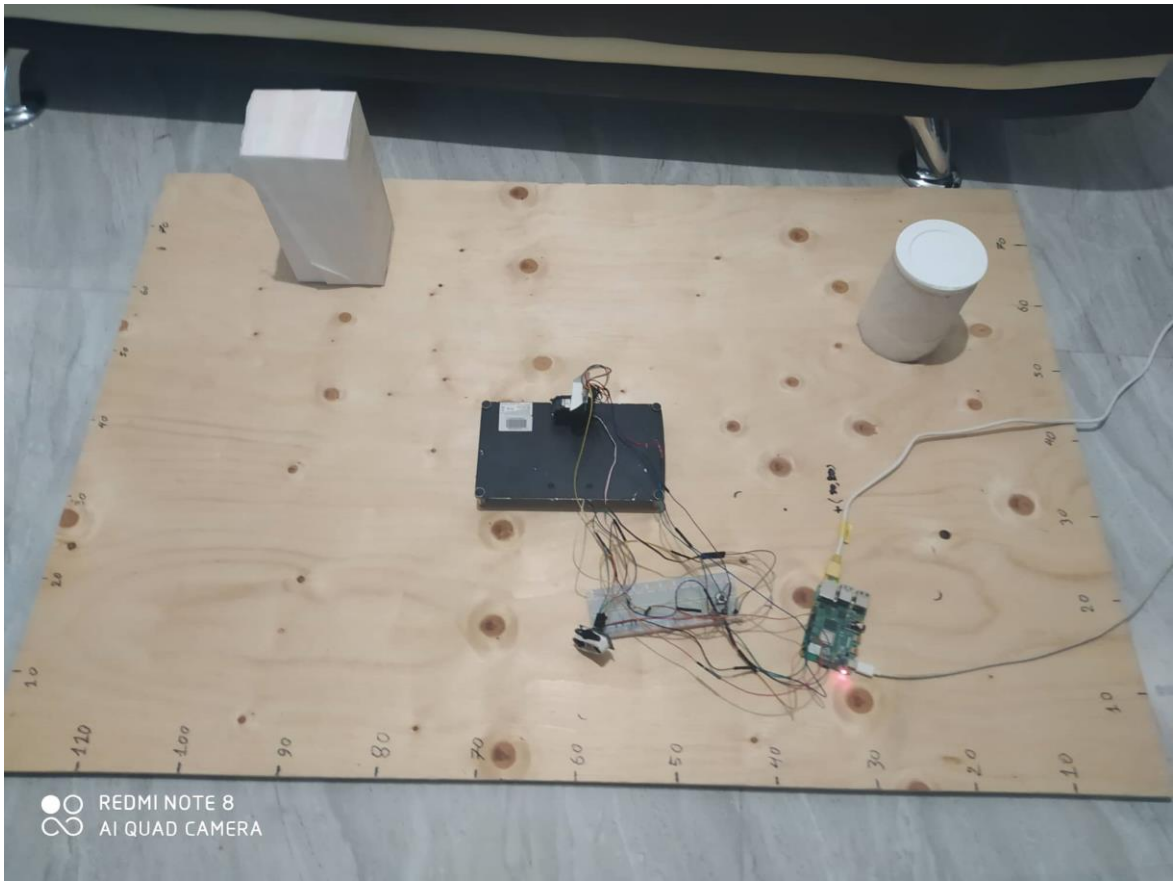
Τα αποτελέσματα της τελικής χαρτογράφησης κατά την τοποθέτηση στη τρίτη θέση (x=17, y=48) είναι τα εξής:



Εικόνα 4.28 Τελικό αποτέλεσμα χαρτογράφησης με Ultrasonic sensor τοποθετημένο στην θέση (x=17, y=48)

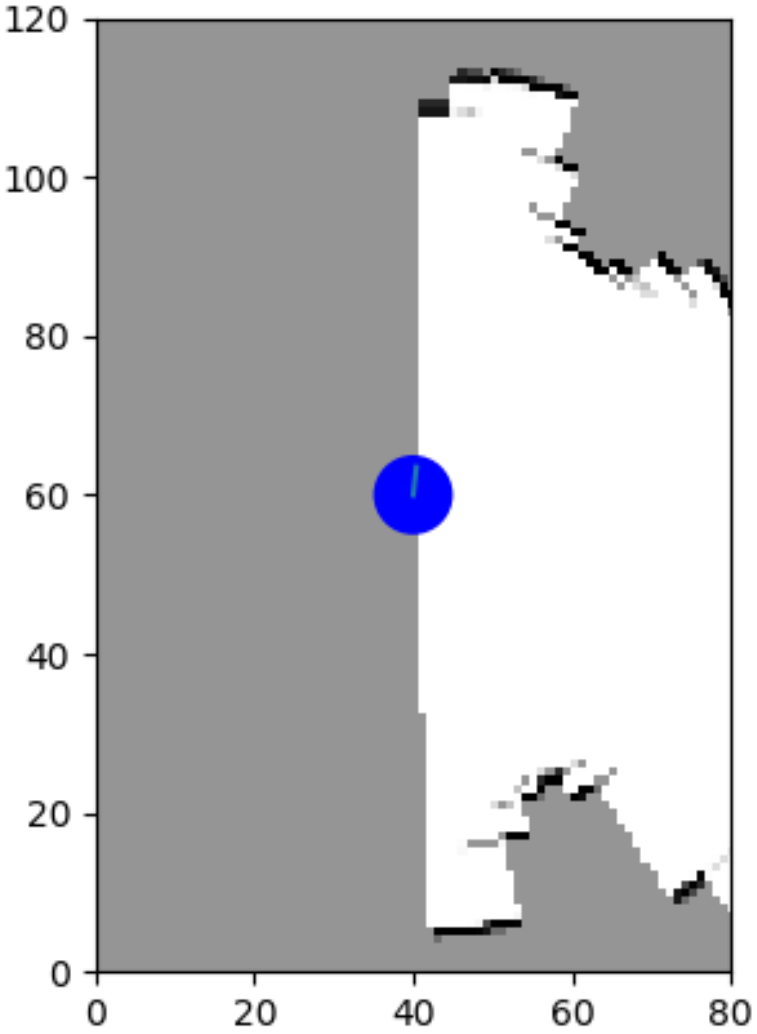
Και πάλι μπορούμε να διακρίνουμε μια βελτίωση στην αναπαράσταση των μεγεθών των αντικειμένων, αν και μετά την τελική ανανέωση του χάρτη παρατηρούνται και αρκετά σημεία θορύβου.

Τέλος ξεκινήσαμε την διαδικασία και πάλι τοποθετώντας τον lidar αισθητήρα στην κατασκευή με τα επιλεγμένα αντικείμενα.



Εικόνα 4.29 Δοκιμή κατασκευής με Lidar sensor στην πρώτη θέση (x=40, y=60)

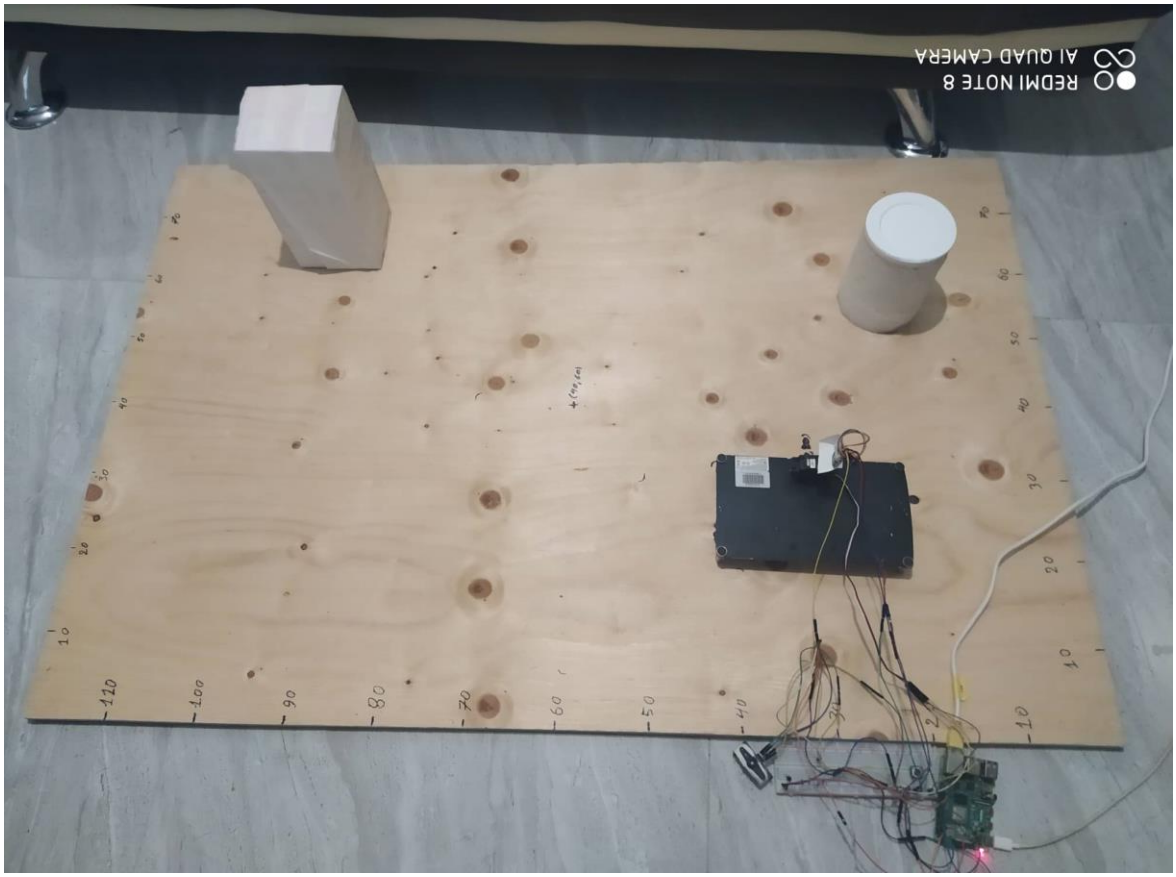
Το αποτέλεσμα της χαρτογράφησης σε αυτήν την θέση (x=40, y=60) ήταν:



Εικόνα 4.30 Αποτέλεσμα χαρτογράφησης με Lidar sensor τοποθετημένο στην θέση (x=40, y=60)

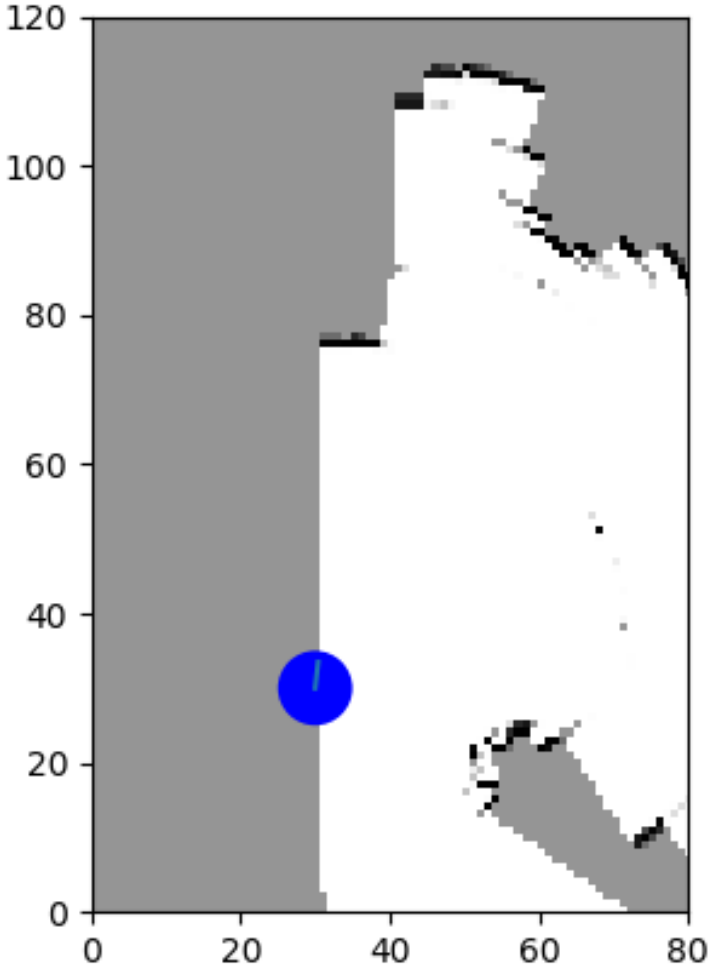
Μετά την αντικατάσταση του αισθητήρα υπερήχων με τον lidar, παρατηρούμε πως το πραγματικό μέγεθος των αντικειμένων απεικονίζεται αρκετά καλά και υπάρχει μια βελτίωση ως προς την αναπαράσταση της γεωμετρίας των αντικειμένων.

Συνεχίζοντας την χαρτογράφηση τοποθετήσαμε την κατασκευή στην δεύτερη θέση.



Εικόνα 4.31 Δοκιμή κατασκευής με Lidar sensor στην δεύτερη θέση (x=30, y=30)

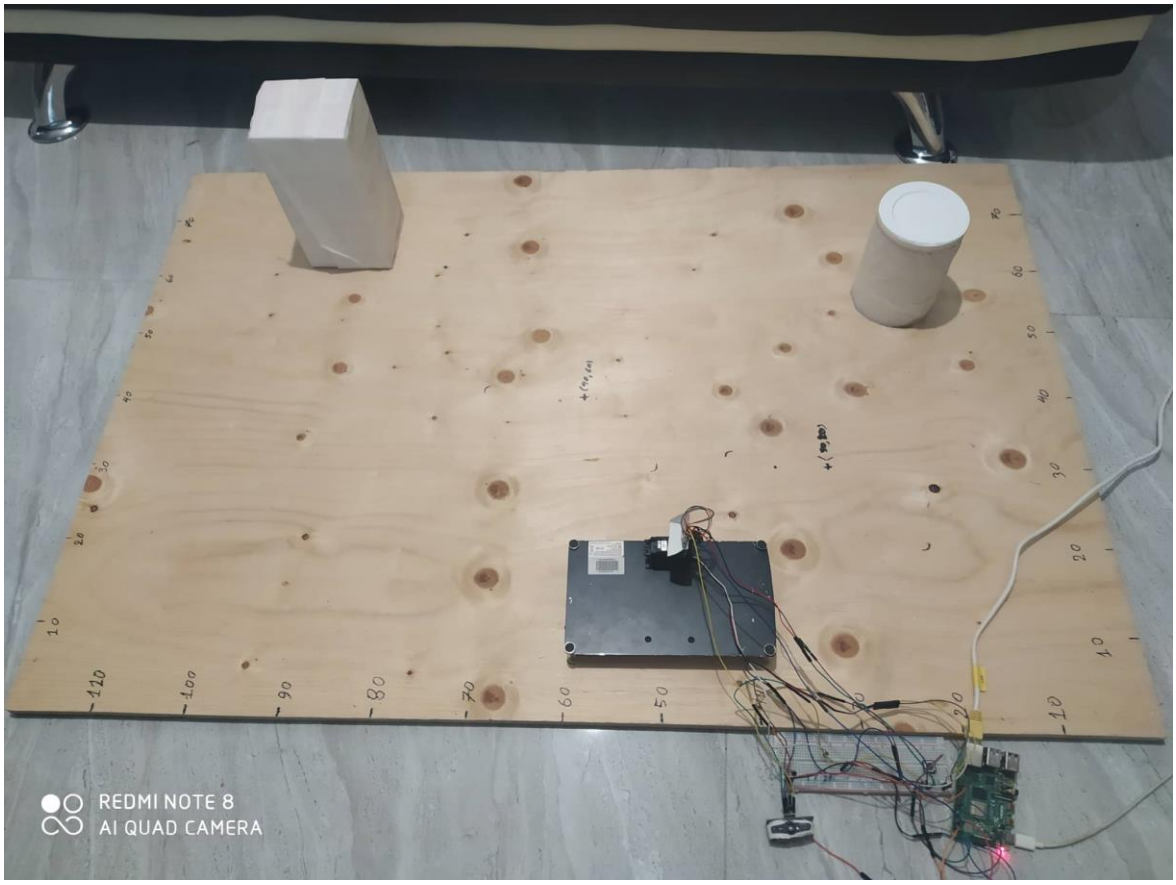
Το αποτέλεσμα της συνέχειας της χαρτογράφησης ήταν το εξής:



Εικόνα 4.32 Αποτέλεσμα συνέχειας χαρτογράφησης με Lidar sensor τοποθετημένο στην θέση (x=30, y=30)

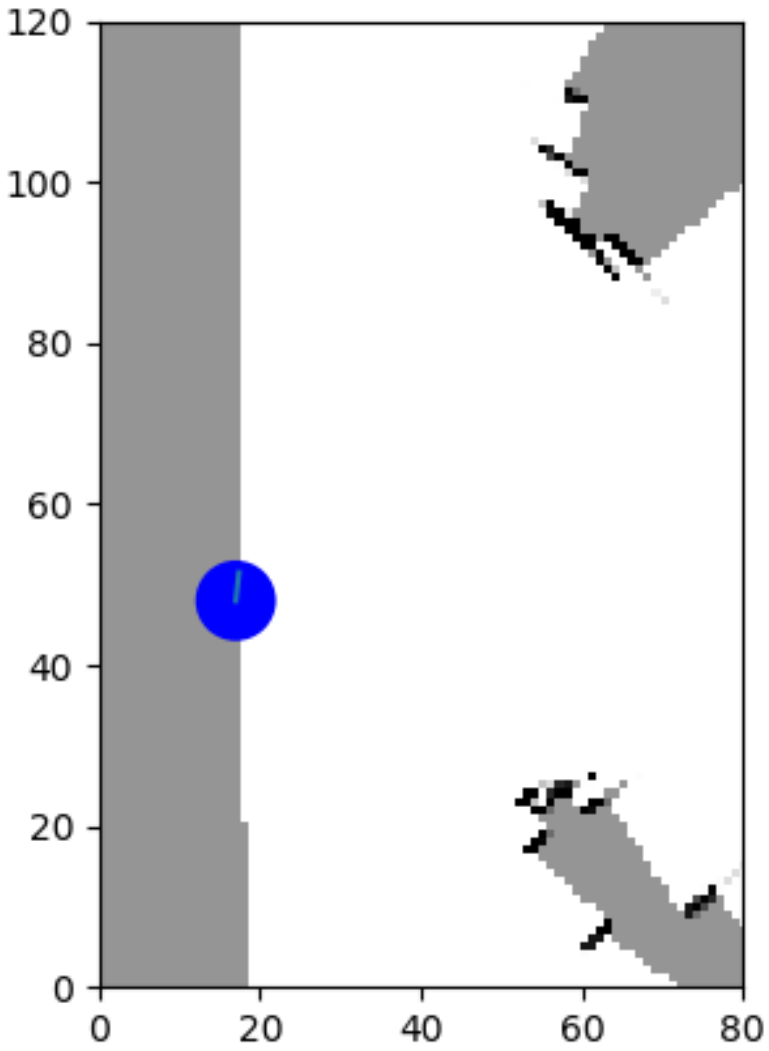
Μετά την μετατόπιση της κατασκευής στην νέα θέση και την ανανέωση του χάρτη μπορούμε να δούμε πως ενώ υπήρξε βελτίωση στην απεικόνιση των αντικειμένων, παρουσιάστηκαν κάποια σημεία θορύβου.

Ολοκληρώνοντας την διαδικασία, τοποθετήσαμε την κατασκευή μας και στην τελευταία θέση (x=17, y=48)



Εικόνα 4.33 Δοκιμή κατασκευής με Lidar sensor στην τρίτη θέση (x=17, y=48)

Το αποτέλεσμα της τελικής χαρτογράφησης φαίνεται στην παρακάτω εικόνα:



Εικόνα 4.34 Αποτέλεσμα τελικής χαρτογράφησης με Lidar sensor τοποθετημένο στην θέση (x=17, y=48)

Στο τελικό μας αποτέλεσμα παρατηρούμε πως τα σημεία θορύβου εξαλείφθηκαν, οι θέσεις των αντικειμένων είναι σωστές και το μέγεθος τους ανιχνεύθηκε αρκετά σωστά. Μειονέκτημα αποτελεί η αδυναμία της κατασκευής μας να αναπαραστήσει σωστά την γεωμετρία των αντικειμένων.

Κεφάλαιο 5.

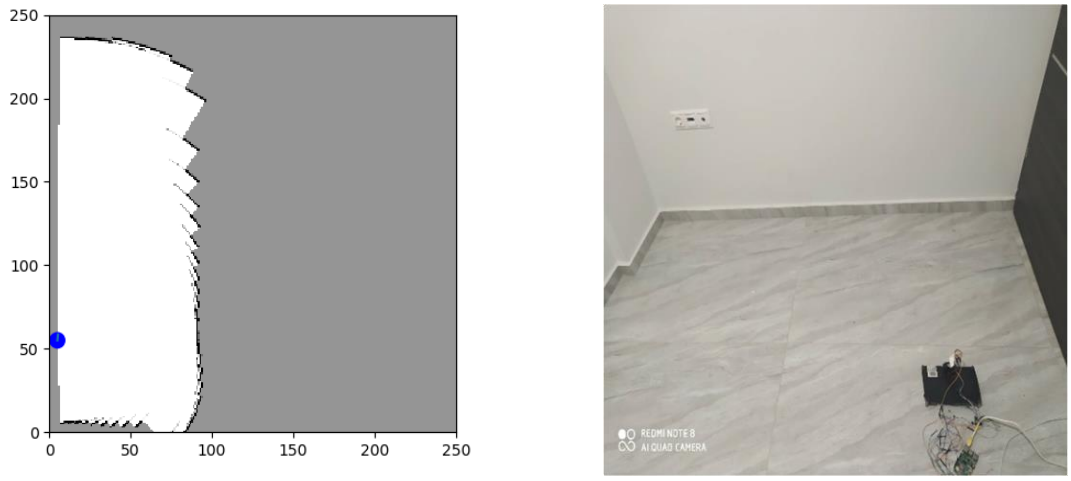
Αξιολόγηση των αποτελεσμάτων

Μετά από ένα μεγάλο σύνολο δοκιμών και πειραμάτων που εκτελέσθηκαν συνολικά στην παρούσα διπλωματική εργασία μπορούμε να πούμε πως κατασκευάσαμε έναν αλγόριθμο απολύτως ικανό να υλοποιήσει χαρτογράφηση πλέγματος κατάληψης λαμβάνοντας κατάλληλες μετρήσεις. Επίσης στα πλαίσια των πειραμάτων διαπιστώθηκαν αρκετά πράγματα για την καταλληλόλητα και την αποτελεσματικότητα τόσο των αισθητηρίων όσο και του αλγορίθμου.

Αρχικά παρατηρήσαμε πως όσον αφορά τον αλγόριθμο, ένας μεγάλος αριθμός μετρήσεων πρέπει να εφαρμοσθεί σε κάθε μετατόπιση των αισθητηρίων για να λάβουμε ακριβείς ενδείξεις. Συγκεκριμένα εμείς καταλήξαμε πως τουλάχιστον δέκα μετρήσεις πρέπει να λαμβάνονται σε κάθε βήμα του σερβοκινητήρα, ώστε μια θέση να θεωρηθεί ορθά κατειλημμένη ή όχι.

Σχετικά με τα αντικείμενα, καταλήξαμε στο ότι το μέγεθος των αντικειμένων που καλείται η κατασκευή μας να ανιχνεύσει παίζει καθοριστικό ρόλο. Μεγαλύτερα αντικείμενα ανιχνεύονται με πολύ μεγαλύτερη ακρίβεια σε σχέση με μικρότερα. Επίσης παρατηρήθηκε πως αντικείμενα σύνθετης γεωμετρίας αποτυπώνονται πολύ δυσκολότερα απ’ ότι τα απλά (τετράγωνα, ορθογώνια). Ένα τελευταίο πράγμα που μάθαμε για την ανίχνευση των αντικειμένων είναι πως αντικείμενα με ανακλαστικές επιφάνειές και χρώματα επηρεάζουν αρνητικά τα αποτελέσματα μας.

Κατά την διάρκεια των πειραμάτων μας με τον αισθητήρα υπερήχων παρατηρήσαμε πως το μεγάλο εύρος του λοβού της υπερηχητικής δέσμης καθιστούσε ιδιαίτερα δύσκολη την ανίχνευση γωνιών, καθώς και την ανίχνευση του κενού μεταξύ αντικειμένων που βρισκόντουσαν πολύ κοντά το ένα στο άλλο. Αισθητά καλύτερα ήταν τα αποτελέσματα των μετρήσεων που λαμβάνονταν από μεγαλύτερες αποστάσεις, μιας και το εύρος του λοβού επηρέαζε λιγότερο τις μετρήσεις. Επίσης προβλήματα λόγο του εύρους του λοβού παρουσιάστηκαν και σε αντικείμενα που ο αισθητήρας έπρεπε να τα αναγνωρίσει υπό γωνία. Παρόμοια προβλήματα αντιμετωπίσαμε και με τον ToF αισθητήρα, αλλά σε μικρότερο βαθμό, λόγω του σημαντικά μικρότερου εύρους λοβού ανίχνευσης που παρέχει. Ένα ακόμη κοινό πρόβλημα που αντιμετώπισαν οι αισθητήρες μας λόγω του εύρους ανίχνευσης ήταν η αναγνώριση των τοίχων στο περιβάλλον όπως μπορούμε να δούμε στην ακόλουθη εικόνα.



Εικόνα 5.1 Αριστερά το αποτέλεσμα της χαρτογράφησης. Δεξιά η φωτογραφία του πραγματικού περιβάλλοντος.

Στόχος της εργασίας μας ήταν επίσης να διαπιστώσουμε ποιος από τους δύο αισθητήρες είναι ικανός να δημιουργήσει ένα πλέγμα κατάληψης με μεγαλύτερη ακρίβεια και πλέων έχοντας εκτελέσει έναν μεγάλο αριθμό πειραμάτων μπορούμε να πούμε με σιγουριά πως ο αισθητήρας ToF είναι καταλληλότερος του ultrasonic για την συγκεκριμένη εφαρμογή. Βασικό λόγο αποτελεί το μικρότερο εύρος λοβού που μας επιτρέπει πιο ακριβή ανίχνευση κατά τις μετατοπίσεις του αισθητήρα, καθώς και η μεγαλύτερη ακρίβεια μετρήσεων που παρέχει ο συγκεκριμένος αισθητήρας.

Κεφάλαιο 6.

Συμπεράσματα και προοπτικές μελλοντικής έρευνας.

Συνοψίζοντας, μπορούμε να πούμε πως η κατασκευή μας είναι σε ικανή να εκτελέσει χαρτογράφηση πλέγματος κατάληψης και με τους δύο αισθητήρες, αλλα ενώ και οι αισθητήρες είναι κατάλληλοι για να επιτρέψουν σε ένα ρομπότ να αποφεύγει εμπόδια και να κινείται στο περιβάλλον, κανένας από τους δύο δεν επαρκεί για την δημιουργία ενώς λεπτομερούς χάρτη απεικόνησης του περιβάλλοντος.

Μερικά από τα πράγματα που θα μπορούσαμε να δοκιμάσουμε είτε για να βελτιώσουμε την κατασκευή μας, είτε για να επεκτείνουμε την έρευνα μας είναι:

- Η τοποθέτηση της κατασκευής πάνω σε ένα όχημα διαφορικής οδήγησης.

- Η δοκιμή πολλαπλών αισθητήρων ταυτόχρονα. Για παράδειγμα δοκιμές με δύο αισθητήρες υπερήχων τοποθετημένους πάνω στο ρομπότ ή έναν αισθητήρα υπερήχων και έναν ToF. Εναλλακτικά θα μπορούσαμε να δοκιμάσουμε την αντικατάσταση των συγκεκριμένων αισθητήρων με κάποιους μεγαλύτερης ακριβείας ή αισθητήρες lidar και radar που παρέχουν ανίχνευση 360 μοιρών.

- Η αντικατάσταση του συγκεκριμένου σερβοκινητήρα με κάποιον ικανό να εκτελέσει περιστροφή 360 μοιρών, μιας και ο τωρινός δεν είχε την δυνατότητα να εκτελέσει πλήρη περιστροφή.

- Η τοποθέτηση δυναμικών αντικειμένων στο περιβάλλον που χαρτογραφείται.

- Η εφαρμογή φίλτρων στο τελικό αποτέλεσμα της χαρτογράφησης για την απαλοιφή των θορύβων.

- Η χρήση Arduino boards για την οδήγηση του ρομπότ και των αισθητήρων.

Βιβλιογραφία

[1] S. Thrun. “Robotic mapping: A survey.” In G. Lakemeyer and B. Nebel, editors, “Exploring Artificial Intelligence in the New Millenium,” Morgan Kaufmann. To appear (2002).

[2] M. E. Yeap, Wai K; Jefferies. Robotics and cognitive approaches to spatial mapping, volume 38 of Springer tracts in advanced robotics. Springer (2008).

[3] Mobile robot localization and map building: a multisensor fusion approach. Kluwer Academic Publishers (1999).

[4] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba. “A solution to the simultaneous localization and map building (slam) problem.” Robotics and Automation, IEEE Transactions on 17(3), 229–241. ISSN 1042-296X (2001).

[5] J. Guivant and E. Nebot. “Optimization of the simultaneous localization and map-building algorithm for real-time implementation.” Robotics and Automation, IEEE Transactions on 17(3), 242–257. ISSN 1042-296X (2001).

[6] J. J. Leonard, H. F. Durrant-Whyte, and I. J. Cox. “Dynamic map building for an autonomous mobile robot.” The International Journal of Robotics Research 11(4), 286–298 (1992).

[7] H. Shatkay. “Learning models for robot navigation.” (1998). 134

[8] H. Shatkay and L. P. Kaelbling. “Learning topological maps with weak local odometric information.” In “IN PROCEEDINGS OF IJCAI-97. IJCAI, INC,” pages 920–929 (1997).

[9] M. C. Newman. “Fab-map: Probabilistic localization and mapping in the space of appearance.” SECS 175. (2008).

[10] Elfes, A. (1987) “Sonar-based real-world mapping and navigation,” Robotics and Automation, IEEE Journal of, 3(3), pp. 249–265.

[11] (2013) “Occupancy grids: A stochastic spatial representation for active robot perception,” arXiv preprint arXiv:1304.1098.

[12] Badino, H., U. Franke, and R. Mester (2007) “Free space computation using stochastic occupancy grids and dynamic programming,” in Workshop on Dynamical Vision, ICCV, vol. 20, Rio de Janeiro, Brazil.

[13] Danescu, R., F. Oniga, and S. Nedevschi (2011) “Modeling and tracking the driving environment with a particle-based occupancy grid,” Intelligent Transportation Systems, IEEE Transactions on, 12(4), pp. 1331–1342.

[14] Marlow, S. Q. and J. W. Langelaan (2011) “Local terrain mapping for obstacle avoidance using monocular vision,” Journal of the American Helicopter Society, 56(2), pp. 22007–22007.

[15] Andreas Geiger, Julius Ziegler, and Christoph Stiller. Stereoscan: Dense 3D Reconstruction in Real-Time. In Intelligent Vehicles Symposium (IV), 2011 IEEE, pages 963–968. IEEE, 2011. 2.4, 5.1

[16] Pablo F Alcantarilla, Chris Beall, and Frank Dellaert. Large-Scale Dense 3D Reconstruction from Stereo Imagery. Georgia Institute of Technology, 2013. 2.4, 5.1

[17] Andreas Nüchter, Kai Lingemann, Joachim Hertzberg, and Hartmut Surmann. 6D SLAM3D Mapping Outdoor Environments. Journal of Field Robotics, 24(8-9):699–722, 2007. 2.4

[18] A. Discant et al. “Sensors for Obstacle Detection - A Survey”. In: Electronics Technology, 30th International Spring Seminar on. 2007, pp. 100–105.

[19] B. ElHalawany et al. “Vision-based obstacles detection for a mobile robot”. In: Informatics and Systems (INFOS), 2012 8th International Conference on. 2012, pp. MM–93–MM–99.

[20] J. Li and M. Chen. “On-Road Multiple Obstacles Detection in Dynamical Background”. In: Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2014 Sixth International Conference on. Vol. 1. 2014, pp. 102–105.

[21] U. Meis, W. Ritter, and H. Neumann. “Detection and classication of obstacles in night vision trac scenes based on infrared imagery”. In: Intelligent Transportation Systems, 2003. Proceedings. 2003 IEEE. Vol. 2. 2003, 1140–1144 vol.2.

[22] N. Bernini et al. “Real-time obstacle detection using stereo vision for autonomous ground vehicles: A survey”. In: Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on. 2014, pp. 873–878.

[23] Z. Khalid, E.-A. Mohamed, and M. Abdenbi. “Stereo vision-based road obstacles detection”. In: Intelligent Systems: Theories and Applications (SITA), 2013 8th International Conference on. 2013, pp. 1–6.

[24] Z. Zhang et al. “Real-time obstacle detection based on stereo vision for automotive applications”. In: Education and Research Conference (EDERC), 2012 5th European DSP. 2012, pp. 281–285. 108 REFERENCES

[25] K. Kaliyaperumal, S. Lakshmanan, and K. Kluge. “An algorithm for detecting roads and obstacles in radar images”. In: Vehicular Technology, IEEE Transactions on 50.1 (2001), pp. 170–182.

[26] G. Brooker, M. Bishop, and S. Scheding. “Millimetre waves for robotics”. In: Australian Conference for Robotics and Automation (2001).

[27] J. Ryde and N. Hillier. “Performance of laser and radar ranging devices in adverse environmental conditions”. In: Journal of Field Robotics 26.9 (2009), pp. 712–727.

[28] U. Nickel. “Applications of superresolution for radar: Examples, problems and solutions”. In: Signal Processing Conference (EUSIPCO), 2013 Proceedings of the 21st European. 2013, pp. 1–5.

[29] J. Han et al. “Enhanced Road Boundary and Obstacle Detection Using a DownwardLooking LIDAR Sensor”. In: Vehicular Technology, IEEE Transactions on 61.3 (2012), pp. 971–985.

[30] O. Yalcin et al. “Detection of road boundaries and obstacles using LIDAR”. In: Computer Science and Electronic Engineering Conference (CEEC), 2014 6th. 2014, pp. 6–10.

[31] R. H. Rasshofer, M. Spies, and H. Spies. “In-uences of weather phenomena on automotive laser radar systems”. In: Advances in Radio Science 9 (2011), pp. 49–60.

[32] C. Huihai, L. Shuqiang, and Z. Yingsheng. “An obstacle detection algorithm used sequential sonar data for Autonomous Land Vehicle”. In: Electronic Measurement Instruments (ICEMI), 2011 10th International Conference on. Vol. 4. 2011, pp. 255–259.

[33] G. Dudek, M. Jenkin. “Computational Principles of Mobile Robotics”. In: Cambridge University Press, 2010, pp 226-227.

[34] R. Chatila and J. Laumond. “Position referencing and consistent world modelling for mobile robots”. In: Proc. IEEE International Conference on Robotics and Automation 1985, pp. 138-170.