

**ΑΝΑΛΥΣΗ ΚΑΙ ΠΡΟΣΟΜΟΙΩΣΗ ΡΟΜΠΟΤΙΚΟΥ
ΒΡΑΧΙΟΝΑ 6 Β.Ε. ΣΕ ΤΡΟΧΟΦΟΡΟ ΟΧΗΜΑ ΜΕ
ΕΦΑΡΜΟΓΗ ΣΤΗ ΔΙΑΛΟΓΗ ΑΝΤΙΚΕΙΜΕΝΩΝ**

ΓΚΟΥΤΖΑΣ ΝΙΚΟΛΑΟΣ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

*ΕΠΙΒΛΕΠΩΝ: ΒΛΑΧΟΣ ΚΩΝΣΤΑΝΤΙΝΟΣ
ΕΠΙΤΡΟΠΗ: ΛΥΚΑΣ ΑΡΙΣΤΕΙΔΗΣ, ΧΑΝΤΑΣ ΙΩΑΝΝΗΣ*

ΙΩΑΝΝΙΝΑ, ΜΑΡΤΙΟΣ 2024



**ΤΜΗΜΑ ΜΗΧ. Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
UNIVERSITY OF IOANNINA**

ΠΕΡΙΕΧΟΜΕΝΑ

Περίληψη	4
Κεφάλαιο 1 Husky	6
1.1 Διαστάσεις και πληροφορίες	7
1.2 Εξοπλισμός και χρήση	8
1.3 Αισθητήρες	9
1.3.1 D435 κάμερα βάθους	9
1.3.1.1 Περιγραφή κάμερας βάθους	10
1.3.1.2 Τεχνική βάθους	10
1.3.1.3 Διαστάσεις	11
1.3.1.4 Τοποθεσία καμερώ στα ρομπότ	11
1.3.2 Lidar	12
1.3.2.1 Περιγραφή	12
1.3.2.2 Λειτουργία Lidar	13
1.3.2.3 Υπολογισμός μαθηματικού τύπου απόστασης	13
1.3.2.4 Παραδείγματα και topics	14
1.4 Περιστροφή Husky	16
1.4.1 Σχεδιασμός οκταμορίου	16
1.4.2 Quaternions και γωνίες Euler	18
1.4.3 Φαινόμενο “Gimbal lock”	18
1.4.4 Διόρθωση αρνητικής γωνίας Euler	19
1.4.5 Κατεύθυνση περιστροφής	20
1.4.6 Εκτίμηση πολογισμού κατεύθυνσης περιστροφής	21
1.4.7 PID ελεγκτής	24
1.5 Προσανατολισμός Husky	26
1.5.1 Έλεγχος προσανατολισμού	26
1.5.2 Διόρθωση προσανατολισμού	29
1.5.3 Κίνηση	30
1.6 Κάμερα Husky	38
1.6.1 OpenCV	38
1.6.2 Λειτουργία κάμερας μέσω OpenCV	40
1.6.2.1 Χρησιμότητα κάμερας και OpenCV	40
1.6.2.2 Αλγόριθμος κάμερας Husky	41

1.7	Οπισθοχώρηση Husky.....	63
1.8	Επιλογή τραπεζιού.....	65
Κεφάλαιο 2	Ur3.....	66
2.1	Διαστάσεις και πληροφορίες.....	67
2.2	Εξοπλισμός και χρήση βραχίονα.....	69
2.3	Κάμερες.....	70
	2.3.1 Απεικόνιση περιβάλλοντος μέσω καμερών.....	71
2.4	Ευθεία κινηματική.....	72
2.5	Αντίστροφη κινηματική.....	76
2.6	Θέση εκκίνησης βραχίονα.....	84
2.7	Κίνηση βραχίονα.....	85
2.8	Σάρωση αντικειμένου.....	92
2.9	Αλγόριθμος μπροστινής κάμερας Ur3.....	101
2.10	Σύγκριση αλγορίθμων κάμερας.....	105
	2.10.1 Αλγόριθμος IBVS.....	106
	2.10.2 Σύγκριση αλγορίθμου IBVS και κεφαλαίου 2.8.....	109
2.11	Αρπαγή αντικειμένου.....	112
2.12	Σάρωση τραπεζιού-στόχου.....	114
2.13	Αλγόριθμος κάτω κάμρας Ur3.....	127
2.14	Topics Ur3.....	128
Κεφάλαιο 3	Έλεγχοι και συμπεράσματα.....	129
3.1	Επιτυχίες & αποτυχίες αποστολής.....	129
3.2	Menu & έλεγχος των ρομπότ.....	130
Βιβλιογραφία.....		132

ΠΕΡΙΛΗΨΗ

Οι ρομποτικές πλατφόρμες είναι κατασκευές που επιτρέπουν την κίνηση και την εκτέλεση εργασιών σε διάφορα περιβάλλοντα. Συνήθως συνεργάζονται με ρομποτικούς βραχίονες, οι οποίοι είναι μηχανικά συστήματα που αποτελούνται από αρθρώσεις και εργαλεία, επιτρέποντας την εκτέλεση ποικίλων εργασιών. Η συνεργασία μεταξύ αυτών των δύο ειδών ρομπότ μπορεί να εφαρμοστεί σε πολλούς τομείς, όπως η βιομηχανία, η υγεία και η έρευνα, για αποτελεσματική εκτέλεση εργασιών και αυξημένη ευελιξία.

Η συγκεκριμένη διπλωματική εργασία ασχολείται με δύο ρομπότ. Ένα ρομποτικό όχημα με 4 τροχούς και μία πλατφόρμα πάνω στην οποία είναι τοποθετημένος ένας ρομποτικός βραχίονας έξι βαθμών ελευθερίας στο περιβάλλον Gazebo.

Η ονομασία της ρομποτικής πλατφόρμας είναι Husky, ενώ του βραχίονα UR3.

Ο UR3 έχει ως τελικό στοιχείο δράσης μία αρπάγη, καθώς επίσης και δύο κάμερες.

Αντίστοιχα, το Husky έχει μία κάμερα, ένα gps και ένα lidar.

Σκοπός τους, η διαλογή αντικειμένων.

Τα δύο ρομπότ βρίσκονται σε έναν μεγάλο και κλειστό χώρο, όπου υπάρχουν τραπέζια. Κάποια τραπέζια έχουν τοποθετημένα πάνω διάφορα χρωματιστά κουτιά, ενώ στις επιφάνειες άλλων τραπεζιών υπάρχουν δύο υποδοχές με ποκίλα χρώματα, όπου εκεί μπαίνουν τα αντικείμενα που συλλέγονται.

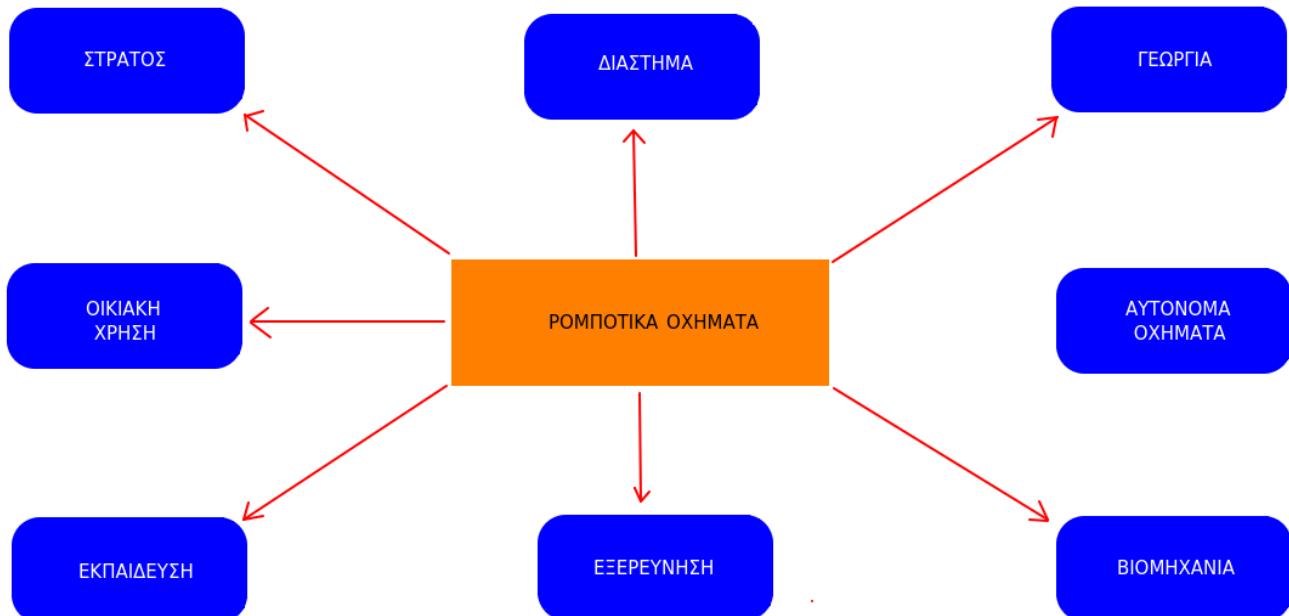
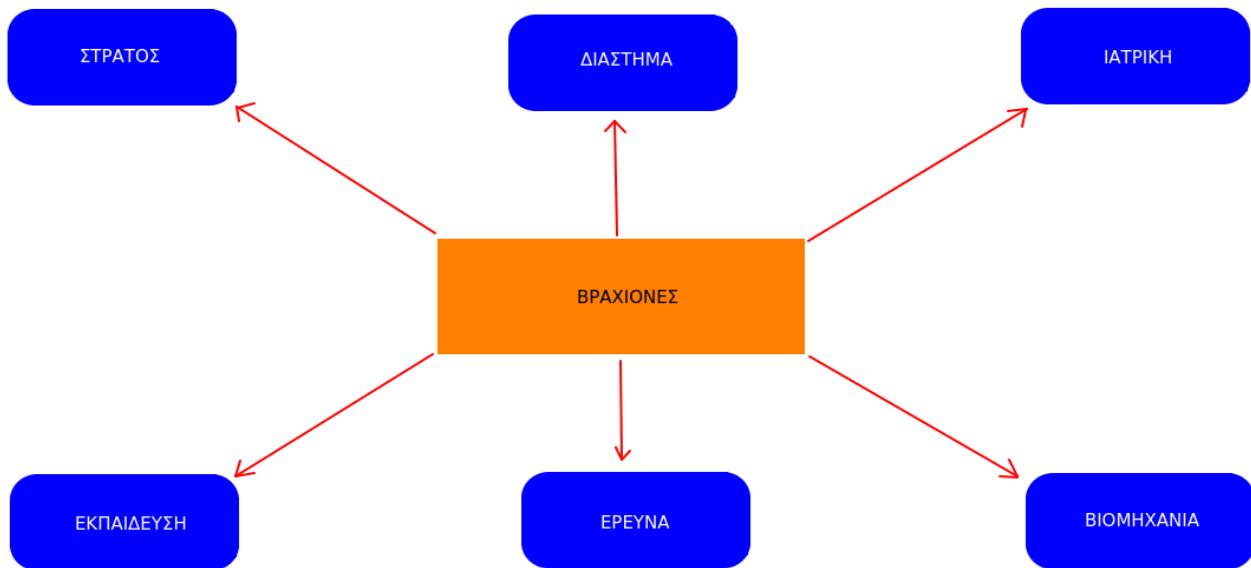
Στόχος του Husky είναι να πάει στα τραπέζια που βρίσκονται τα αντικείμενα και ο UR3 με τη σειρά του να τα πιάσει. Έπειτα, ανάλογα το χρώμα του αντικειμένου, το Husky πηγαίνει στο κατάλληλο τραπέζι με τις υποδοχές, ώστε ο UR3 να αφήσει το αντικείμενο μέσα στην υποδοχή του. Για παράδειγμα, έαν ο UR3 έχει πιάσει ένα κόκκινο αντικείμενο, τότε το Husky πρέπει να πάει σε τραπέζι που η μία από τις υποδοχές του έχει κόκκινο χρώμα και ο UR3 να το αφήσει στη συγκεκριμένη κόκκινη υποδοχή.

Τα δύο μοναδικά δεδομένα που γνωρίζει το Husky είναι οι τοποθεσίες όλων των τραπεζιών και τα χρώματα των τραπεζιών με υποδοχές.

Όλες οι εργασίες είναι πλήρως αυτοματοποιημένες με τη βοήθεια των αισθητήρων και καμερών.

Οταν τοποθετηθούν όλα τα αντικείμενα στις κατάλληλες υποδοχές, τότε η αποστολή έλαβε τέλος με επιτυχία.

ΧΡΗΣΗ ΤΩΝ ΡΟΜΠΟΤ



ΚΕΦΑΛΑΙΟ 1

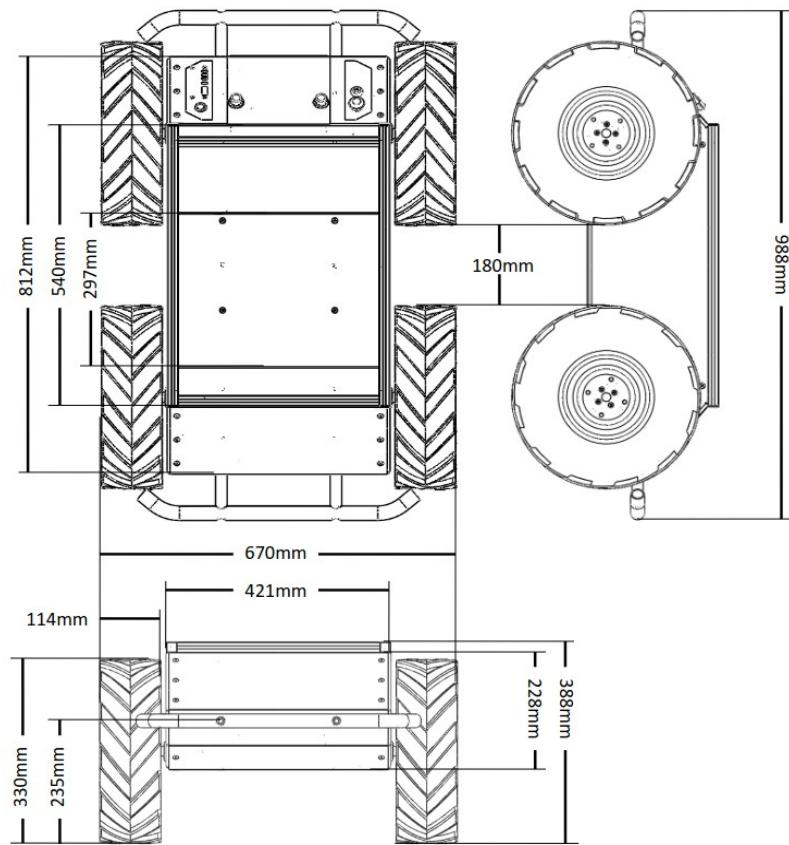
HUSKY



Εικόνα 1.1: Το τροχοφόρο Husky σε εξωτερικό περιβάλλον.

Wiki: [Husky UGV](#)

1.1 ΔΙΑΣΤΑΣΕΙΣ ΚΑΙ ΠΛΗΡΟΦΟΡΙΕΣ



Εικόνα 1.2: Διαστάσεις Husky.

Τύπος	Έτος κυκλοφορίας	Βάρος (Kg)	Φορτίο (Kg)	Μήκος (m)	Πλάτος (m)	Ύψος (m)	Ταχύτητα (m/s)
UGV	2012	50	75	0.988	0.670	0.388	1.0

Πίνακας 1.1: Μερικές πληροφορίες σχετικά με το μοντέλο Husky της Clearpath robotics.

1.2 ΕΞΟΠΛΙΣΜΟΣ ΚΑΙ ΧΡΗΣΗ

Τα συγκεκριμένα ρομπότ μπορούν να εξοπλίζονται με πολλά από τα παρακάτω:

- Βραχίονες
- Αισθητήρες μέτρησης απόστασης
- Αισθητήρες ανίχνευσης και εντοπισμού 3D χώρου
- Κάμερες
- Gps
- Κουτιά αποθήκευσης υλικών
- Γυροσκόπια
- Επιταχυνόμετρα

Τα τροχοφόρα τύπου UGV (Unmanned Ground Vehicle) είναι σχεδιασμένα ώστε να αντέχουν σε ποικίλλες συνθήκες του εξωτερικού περιβάλλοντος, όπως το χιόνι, η λάσπη, το ανώμαλο έδαφος, η κλίση του εδάφους, χάρη των υψηλών ροπών που παρέχουν οι κινητήρες του σε συνδυασμό με το βάρος και τον σχεδιασμό των τροχών του.

Το συγκεκριμένο τροχοφόρο ρομπότ είναι σχεδιασμένο για χρήση σε εφαρμογές έρευνας και ανάπτυξης στον τομέα της ρομποτικής. Επίσης, διαθέτει τέσσερις ηλεκτρικούς κινητήρες, έναν για κάθε τροχό του. Αυτοί οι κινητήρες ελέγχονται από το σύστημα κίνησης του ρομπότ και επιτρέπουν την τετρακίνηση, προσφέροντας βελτιωμένη ευελιξία και μπορεί να μεταφέρει διάφορα-μεγάλα φορτία, ανάλογα με την εφαρμογή.

Στη συγκεκριμένη εφαρμογή το Husky είναι σχεδιασμένο να λειτουργεί είτε τελείως αυτόνομα είτε ύστερα από ανθρώπινη εντολή, όπως η καθοδήγηση του μετά από επιβολή συντεταγμένων και προσανατολισμού προς ένα συγκεκριμένο προορισμό.

1.3 ΑΙΣΘΗΤΗΡΕΣ

Το τροχοφόρο Husky είναι εξοπλισμένο με ένα μικρό πλήθος αισθητήρων, δηλαδή μία κάμερα βάθους d435 στο εμπρόσθιο μέρος του και έναν αισθητήρα μέτρησης απόστασης (lidar), προκειμένου να λαμβάνει πληροφορίες από το περιβάλλον του. Έτσι, μπορεί να περιηγείται αποτελεσματικά σε διάφορα περιβάλλοντα, τα οποία ενδέχεται να περιέχουν εμπόδια και να εκτελεί εργασίες αυτόνομα ή με ελάχιστη ανθρώπινη επιβολή.

1.3.1 D435 ΚΑΜΕΡΑ ΒΑΘΟΥΣ



Εικόνα 1.3: Η κάμερα βάθους D435.

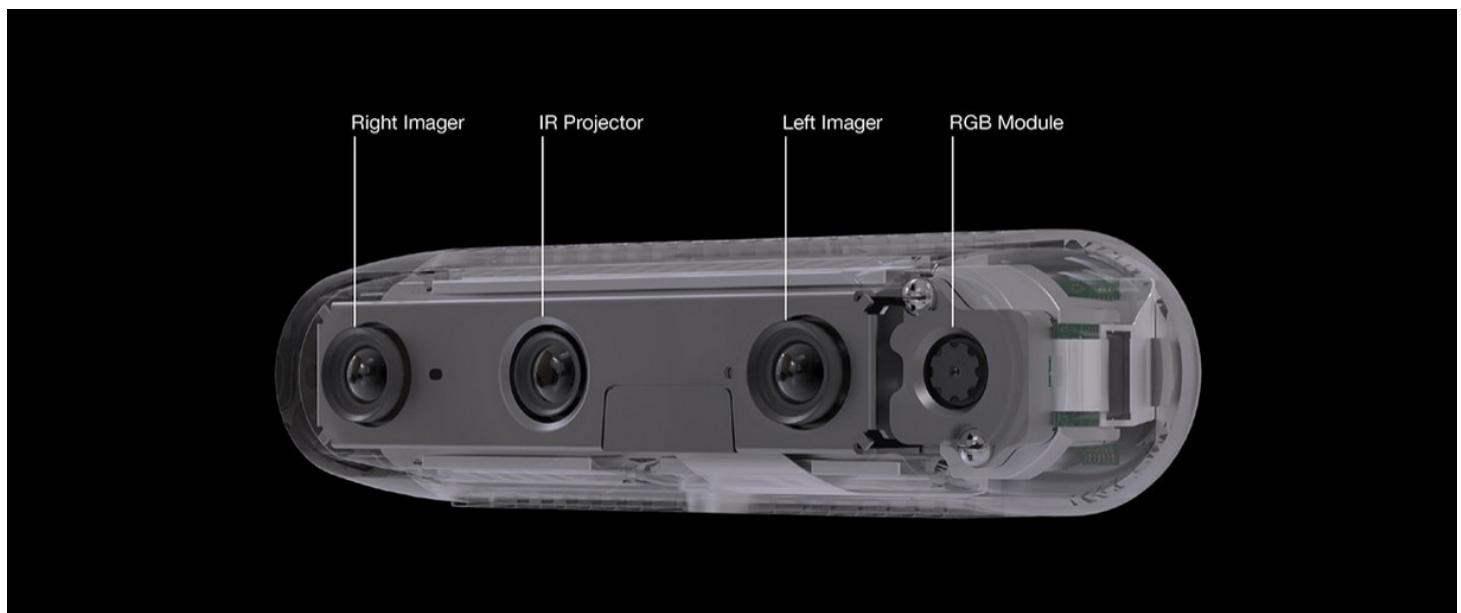
Wiki:[Depth camera D435](#)

1.3.1.1 ΠΕΡΙΓΡΑΦΗ ΚΑΜΕΡΑΣ

Ας ξεκινήσουμε, παρουσιάζοντας τη κάμερα βάθους. Η κάμερα Intel RealSense D435 ανήκει στη σειρά RealSense της Intel, η οποία αναπτύχθηκε για εφαρμογές βάθους, όρασης και ανίχνευσης κίνησης. Παρέχει λειτουργία στερεοσκοπικής απεικόνισης για ανίχνευση βάθους, η οποία καθίσταται κατάλληλη για χρήση σε εφαρμογές όπως ρομποτική, επαυξημένη και εικονική πραγματικότητα.

1.3.1.2 ΤΕΧΝΙΚΗ ΒΑΘΟΥΣ

Η στερεοσκοπική απεικόνιση είναι μια τεχνική που χρησιμοποιεί δύο ή περισσότερες λήψεις ενός σκηνικού αντικειμένου από διαφορετικές γωνίες ή θέσεις για να παράγει μια εικόνα που δίνει την ψευδαίσθηση του βάθους. Κατά κύριο λόγο, αυτή η τεχνική αποσκοπεί στο να αναπαράγει τον τρόπο με τον οποίο βλέπει ο ανθρώπινος εγκέφαλος τον περιβάλλοντα χώρο. Κλασικά, χρησιμοποιούνται δύο εικόνες που λαμβάνονται από τα δύο μάτια (Right Imager και Left Imager, της εικόνας 1.4) και συνδυάζονται για να δημιουργήσουν την εντύπωση του τρισδιάστατου χώρου.



Εικόνα 1.4: Εξαρτήματα κάμερας βάθους D435.

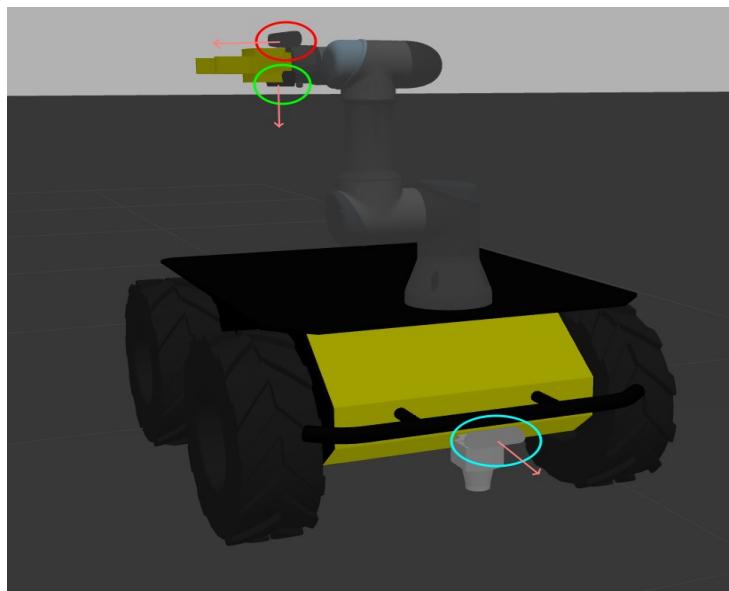
1.3.1.3 ΔΙΑΣΤΑΣΕΙΣ

Διαστάσεις	Πλάτος (m)	0.09
	Ύψος (m)	0.025
	Βάθος (m)	0.02505
Ανάλυση εικόνας	Πλάτος (pixels)	640
	Ύψος (pixels)	480
	Format	R8G8B8
Οπτικό πεδίο	Οριζόντια γωνία πεδίου περίπου 60 βαθμοί	
Βάθος εικόνας	Ελάχιστη απόσταση (m)	0.11
	Μέγιστη απόσταση (m)	10

Πίνακας 1.2: Πληροφορίες για την κάμερα Intel RealSense D435.

1.3.1.4 ΤΟΠΟΘΕΣΙΑ ΚΑΜΕΡΩΝ ΣΤΑ ΡΟΜΠΟΤ

Υπάρχουν τρεις κάμερες βάθους D435 συνολικά που έχουν τοποθετηθεί στα δύο ρομπότ. Συγκεκριμένα, υπάρχει μία κάμερα στο εμπρόσθιο μέρος του Husky. Οι άλλες δύο βρίσκονται στον βραχίονα UR3, η πρώτη στο μπροστινό πάνω μέρος του τελικού στοιχείου δράσης (ΤΣΔ) με προσανατολισμό προς τα μπροστά και η δεύτερη στο μπροστινό κάτω μέρος του ΤΣΔ με προσανατολισμό προς τα κάτω. Οι προσανατολισμοί των τριών καμερών γίνονται αντιληπτοί από τα βελάκια.



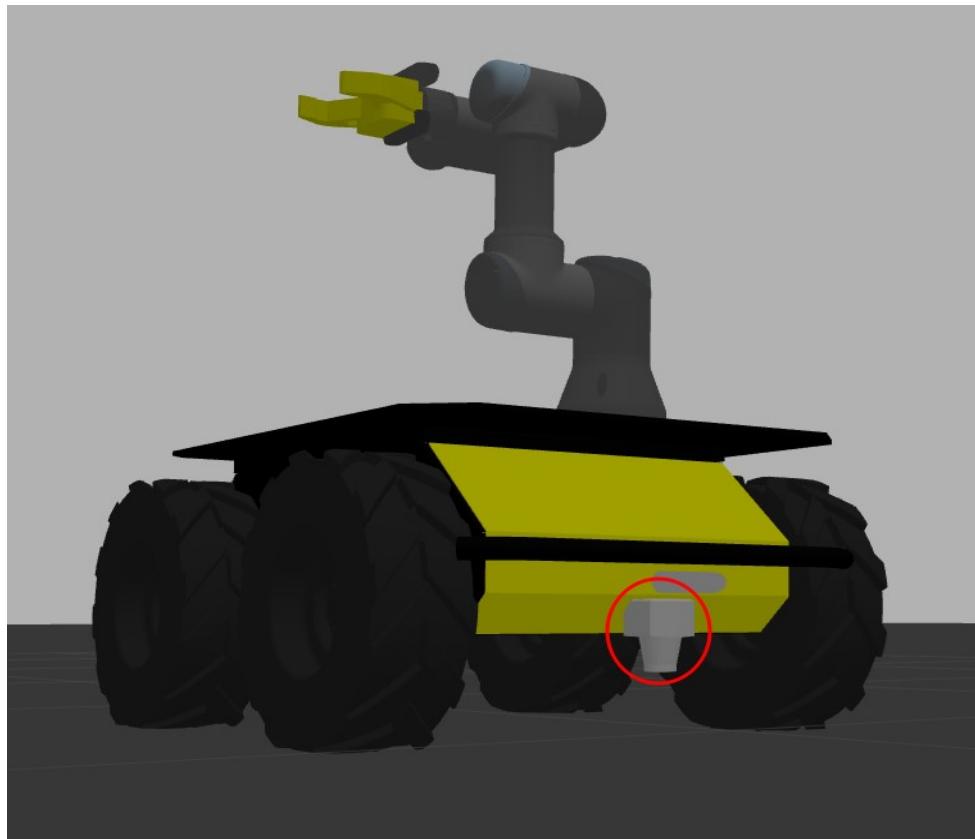
Εικόνα 1.5: Αναπαράσταση των τριών καμέρων.

1.3.2 LIDAR

1.3.2.1 ΠΕΡΙΓΡΑΦΗ

Ο αισθητήρας απόστασης Lidar είναι ένα εξαιρετικά χρήσιμο εργαλείο για το ρομποτικό όχημα Husky. Ουσιαστικά, το Lidar χρησιμοποιεί λέιζερ για να μετρήσει τις αποστάσεις και επίσης έχει τη δυνατότητα να δημιουργήσει έναν χάρτη του περιβάλλοντος γύρω από το ρομπότ.

Αυτό επιτρέπει στο Husky να ανιχνεύει αντικείμενα και εμπόδια, να αποφεύγει εμπόδια κατά την κίνησή του, και να πλοηγείται αποτελεσματικά σε περιβάλλοντα όπου η ακρίβεια της ανίχνευσης αποστάσεων είναι κρίσιμη. Ωστόσο, στη συγκεκριμένη εφαρμογή δεν χρησιμοποιείται χάρτης. Γενικότερα όμως, αυτή η τεχνολογία είναι ιδιαίτερα χρήσιμη σε ρομποτικά όχηματα που χρησιμοποιούνται σε αυτόνομες ή ημιαυτόνομες εφαρμογές, όπως η αυτόνομη κίνηση, η χαρτογράφηση και η αποφυγή εμποδίων. Μπορεί να παρέχει πληροφορίες όχι μόνο για την απόσταση, αλλά και για το σχήμα και το ύψος των αντικειμένων.



Εικόνα 1.6: Αναπαράσταση του αισθητήρα LIDAR στο εμπρόσθιο μέρος του ρομποτικού οχήματος Husky.

1.3.2.2 ΛΕΙΤΟΥΡΓΙΑ LIDAR

Η τεχνολογία LIDAR (Light Detection and Ranging) χρησιμοποιεί το φως για να μετρήσει αποστάσεις με μεγάλη ακρίβεια. Ένα LIDAR συνήθως αποτελείται από έναν πομπό λέιζερ, ένα φακό που επικεντρώνει το φως σε μια στενή δέσμη, έναν καθρέφτη που κατευθύνει το φως προς τον στόχο, και έναν αισθητήρα που μετρά τον χρόνο που χρειάζεται το φως να επιστρέψει από τον στόχο.

Ο πομπός λέιζερ εκπέμπει σύντομες παλμικές ακτίνες φωτός προς το περιβάλλον. Όταν αυτές οι ακτίνες συναντούν ένα αντικείμενο, ανακλώνται πίσω προς το LIDAR. Ο χρόνος που απαιτείται για να επιστρέψει η ακτίνα μετριέται από τον αισθητήρα. Με βάση τον χρόνο πτήσης του φωτός, το LIDAR υπολογίζει την απόσταση μεταξύ του αισθητήρα και του αντικειμένου.

1.3.2.3 ΥΠΟΛΟΓΙΣΜΟΣ ΜΑΘΗΜΑΤΙΚΟΥ ΤΥΠΟΥ ΑΠΟΣΤΑΣΗΣ

Ο υπολογισμός της απόστασης στην τεχνολογία LIDAR βασίζεται στην αρχή του χρόνου πτήσης του φωτός. Ο τύπος υπολογισμού είναι:

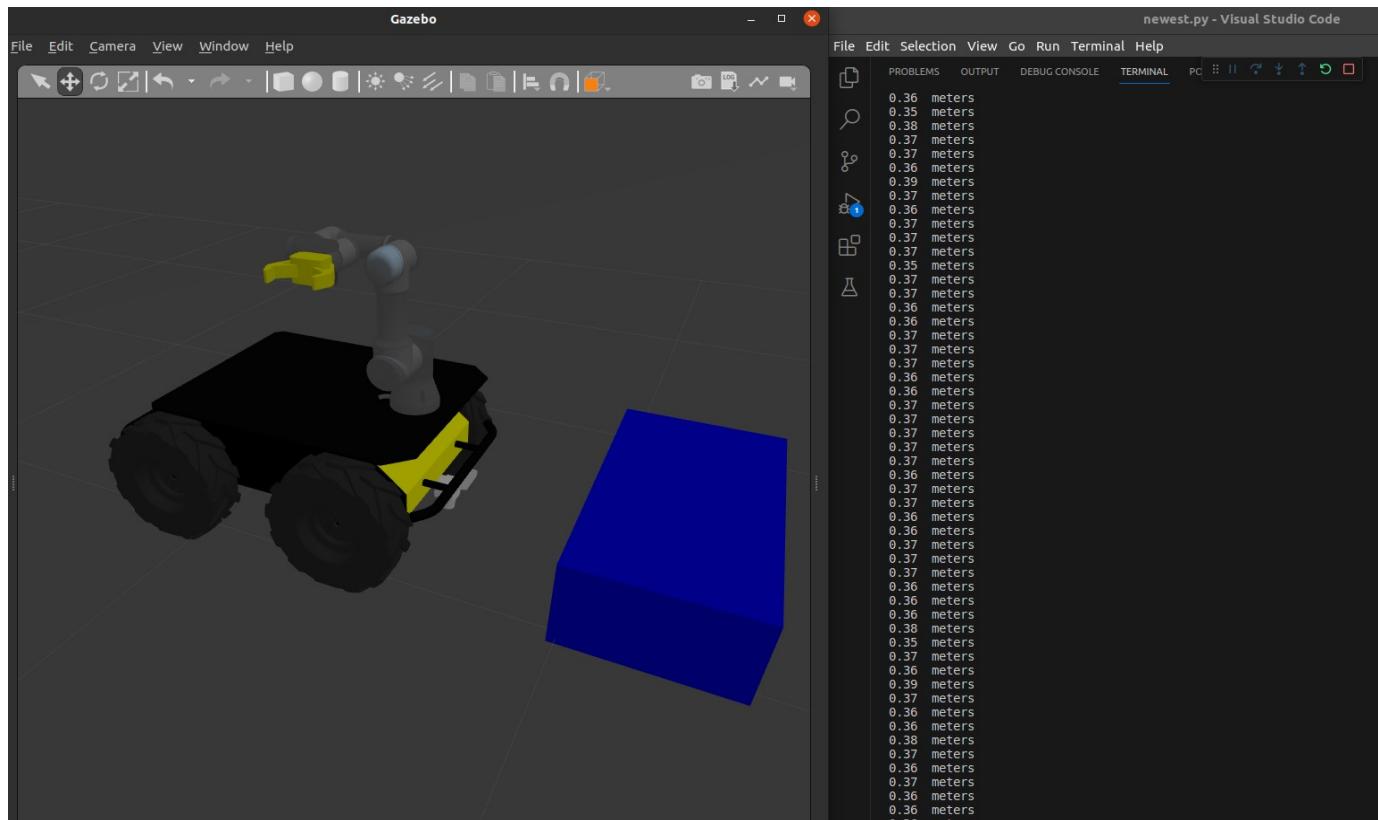
$$\text{Απόσταση} = \frac{\text{Χρόνος πτήσης του φωτός}}{2} \cdot (\text{ταχύτητα του φωτός στο κενό})$$

Η απόσταση υπολογίζεται διαιρώντας τον χρόνο πτήσης του φωτός (που μετριέται από το LIDAR) με το 2, καθώς το λέιζερ κάνει διπλή διαδρομή. Η πρώτη διαδρομή είναι από την έξοδο του αισθητήρα μέχρι τη συνάντηση εμποδίου, ενώ η δεύτερη είναι από την αντανάκλαση του εμποδίου μέχρι την επαφή του και πάλι με τον αισθητήρα.

Στη συνέχεια, πολλαπλασιάζουμε το αποτέλεσμα με την ταχύτητα του φωτός στο κενό (περίπου 3×10^8 μέτρα ανά δευτερόλεπτο) για να υπολογίσουμε την πραγματική απόσταση.

1.3.2.4 ΠΑΡΑΔΕΙΓΜΑΤΑ ΚΑΙ TOPICS

Παρακάτω μπορούμε να δούμε από το γραφικό περιβάλλον Gazebo το ρομποτικό όχημα Husky σε δύο διαφορετικές αποστάσεις από ένα συγκεκριμένο εμπόδιο, τρέχοντας ένα python script τυπώνοντας μας την συγκεκριμένη απόσταση. Το topic “/scan” δίνει πολλές τιμές αποστάσεων, αφού το αισθητήριο όργανο έχει διάφορες γωνίες που εκπέμπει και δέχεται το φως, ώστε να υπολογίσει την απόσταση. Έτσι, λαμβάνουμε τον μέσο όρο αυτών των τιμών για να πάρουμε τη τιμή της απόστασης του ρομπότ από ένα αντικείμενο-εμπόδιο που βρίσκεται μπορτά του σε μία συγκεκριμένη χρονική στιγμή.



Εικόνα 1.7: Απόσταση περίπου 37 εκατοστών του Husky από εμπόδιο.

Στο ROS (Robot Operating System), τα "topics" είναι ένας τρόπος επικοινωνίας μεταξύ διάφορων τμημάτων ενός ρομπότ ή άλλου συστήματος που χρησιμοποιεί το ROS. Κάθε topic είναι σαν ένα κανάλι επικοινωνίας όπου ένα τμήμα λογισμικού μπορεί να δημοσιεύει (publish) δεδομένα και ένα άλλο τμήμα μπορεί να τα λαμβάνει (subscribe). Αυτή η ασύγχρονη επικοινωνία βοηθά στον σχεδιασμό ευέλικτων συστημάτων όπου διάφορα τμήματα μπορούν να λειτουργούν ανεξάρτητα. Παραδείγματα topics μπορεί να περιλαμβάνουν δεδομένα αισθητήρων, θέσης, ή οποιουδήποτε άλλου είδους πληροφορίας που χρειάζεται να μεταφερθεί μεταξύ τμημάτων του ρομπότ. Τα topics τα οποία χρησιμοποιούνται στο όχημα Husky είναι τα εξής:

- **“odometry/filtered”:**

Χρησιμοποιείται για τη μετάδοση δεδομένων οδομετρίας (odometry) που έχουν υποστεί φιλτράρισμα ή επεξεργασία για τη βελτίωση της ακρίβειας και της αξιοπιστίας των πληροφοριών κίνησης του ρομπότ. Η οδομετρία αφορά τη μέτρηση της κίνησης και της θέσης ενός ρομπότ. Δίνει δηλαδή τη τρέχουσα θέση (position) και προσανατολισμό (orientation) του ρομποτικού οχήματος για κάθε άξονα περιστροφής του (X-Y-Z).

- **“cmd/vel”:** Χρησιμοποιείται για τη μετάδοση εντολών κίνησης (velocity commands) προς ένα ρομπότ. Χρήση γραμμικών και γωνιακών ταχυτήτων.

Έχουν πρόσημο προϋποθέτωντας αλλαγή κατεύθυνσης (μπροστά/πίσω ή δεξιά/αριστερά).

- **“h_d435/rgb/h_image_raw”:** Χρησιμοποιείται για τη μετάδοση εικόνας από τη κάμερα τύπου D435 RealSense του Husky.

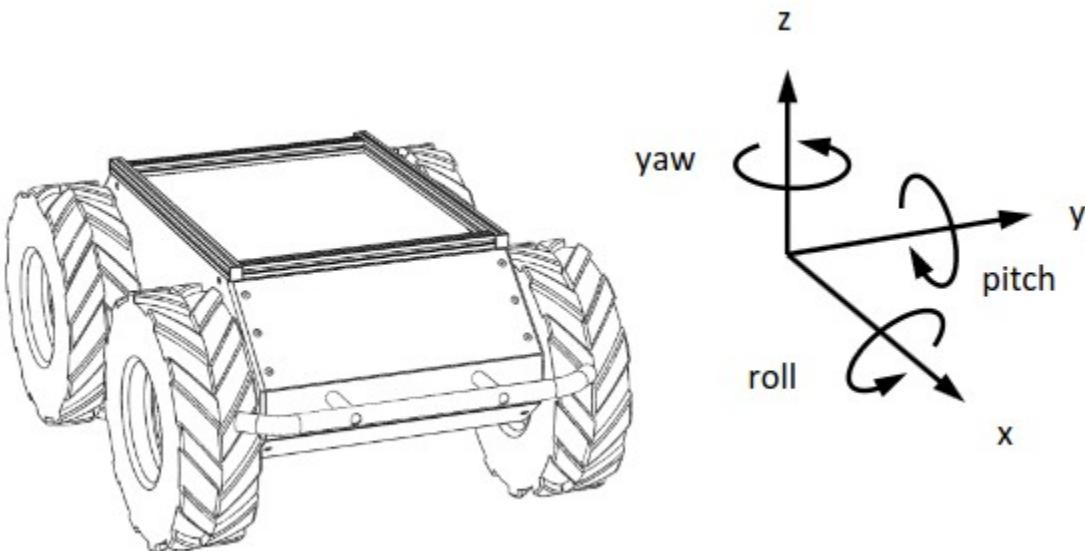
- **“scan”:** Χρησιμοποιείται για τη μετάδοση δεδομένων από τον αισθητήρα LIDAR, παρέχοντας πληροφορίες σάρωσης του περιβάλλοντος.

1.4 Περιστροφή Husky

1.4.1 ΣΧΕΔΙΑΣΜΟΣ ΟΚΤΑΜΟΡΙΟΥ

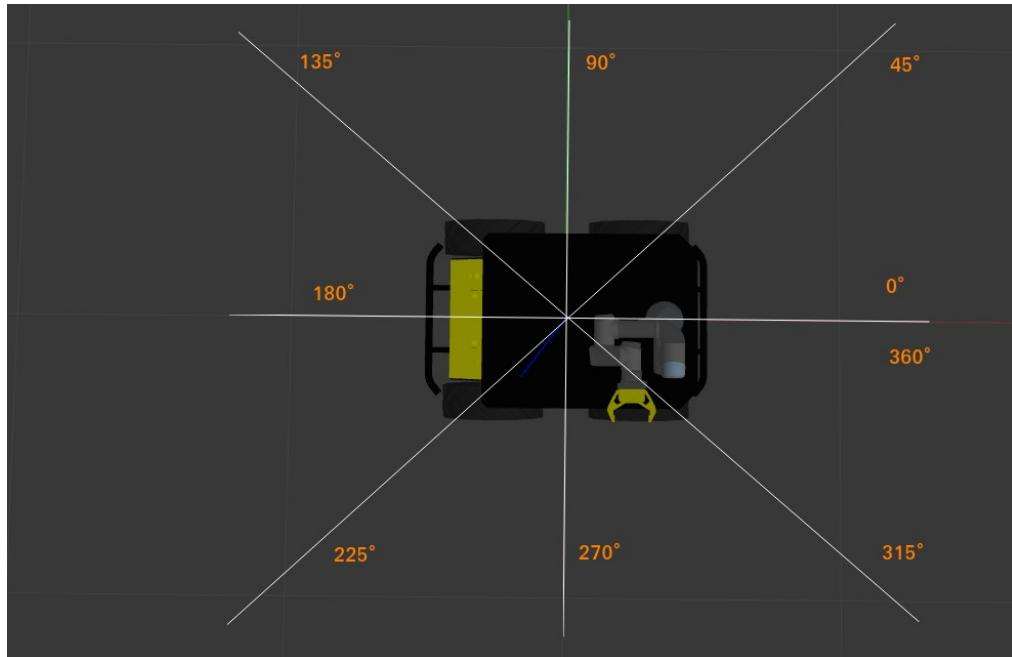
Κύριος και μοναδικός σκοπός του τροχοφόρου Husky είναι να μετακινηθεί στα τραπέζια, τα οποία βρίσκονται σε προκαθορισμένες και γνωστές θέσεις στον τρισδιάστατο κλειστό χώρο όπως προαναφέρθηκε. Για να συμβεί αυτη η μετακίνηση, χρειάζεται να έχει τον σωστό προσανατολισμό (σε σχέση με τα τραπέζια), οπότε η περιστροφή του κατά τον άξονα Z είναι μονόδρομος.

Με αφορμή τα παραπάνω, εκτός από την θέση στην οποία βρίσκεται το Husky κάθε χρονική στιγμή, ο προσανατολισμός αποτελεί πρωταρχικό ρόλο του. Στην ενότητα αυτή θα εξηγήσουμε και θα αναλύσουμε λεπτομερώς την περιστροφή του ρομποτικού οχήματος Husky, ενώ ο προσανατολισμός θα περιγραφεί σε επόμενο κεφάλαιο. Όπως προείπαμε, το Husky περιστρέφεται γύρω από τον άξονα Z, ο οποίος τοποθετείται στο κέντρο του.



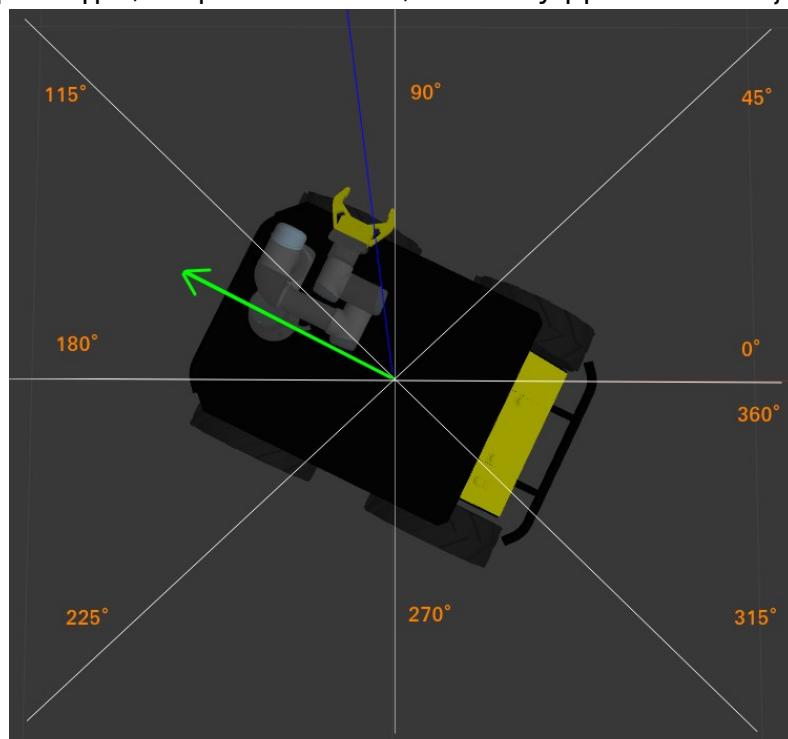
Εικόνα 1.8: Ο μοναδικός άξονας περιστροφής του Husky είναι ο Z (yaw).

Στην εικόνα 1.9 παρακάτω φαίνεται ο σχεδιασμός ενός οκταμορίου (γύρω από το κέντρο του Husky), το οποίο χωρίζεται στις εξής γωνίες: 0° έως 360° ανά 45° .



Εικόνα 1.9: Αναπαράσταση οκταμορίου γύρω από το κέντρο του οχήματος (κάτωψη).

Το Husky έχει στη διάθεση του τον τρέχων προσανατολισμό και έναν προσανατολισμό στόχου. Για παράδειγμα, στην εικόνα 1.10, το Husky βρίσκεται στις 135 μοίρες.



Εικόνα 1.10: Το Husky σε γωνία περιστροφής 135 μοιρών.

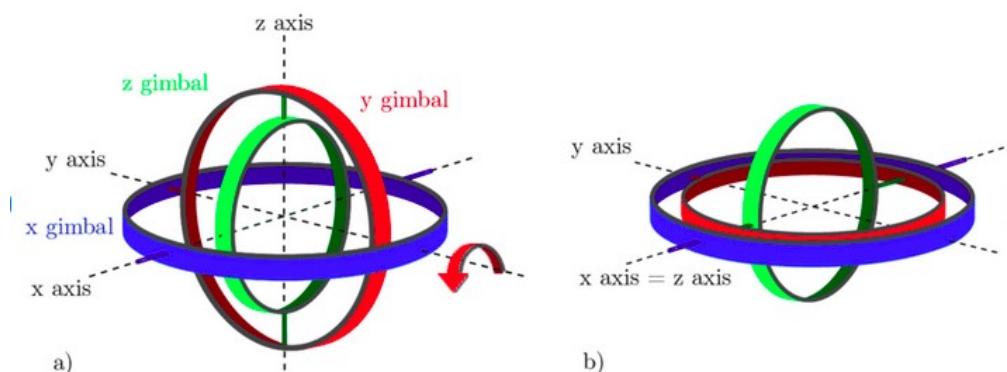
1.4.2 QUATERNIONS KAI ΓΩΝΙΕΣ EULER

Το ρομπότ χρειάζεται δύο διανύσματα. Το πρώτο αντιστοιχεί στη θέση του στους άξονες X και Y στον δισδιάστατο χώρο, ενώ το δεύτερο αντιστοιχεί στο προσανατολισμό, δηλαδή στη γωνία περιστροφής ή αλλιώς σε ένα διάνυσμα περιστροφής στο δισδιάστατο χώρο, το οποίο αφορά τον άξονα Z. Τίθεται το ερώτημα, σε τι μορφή θα αναπαραστήσουμε τη γωνία περιστροφής; Euler ή Quaternions; Εξηγούμε παρακάτω.

Η τελική γωνία περιστροφής είναι σε μορφή euler. Γίνεται δηλαδή μετατροπή των quaternions σε euler από το topic "/odometry/filtered", διότι βρισκόμαστε στο δισδιάστατο χώρο, όσον αφορά τις κινήσεις του Husky. Οπότε, η επιλογή χρήσης euler γωνιών αντί quaternions δικαιολογείται λόγω του ότι είναι πιό εύκολη στην κατανόηση, καθώς αναφέρονται απευθείας σε γωνίες περιστροφής, όπως οριζόντια (roll), κάθετη (pitch) και κατακόρυφη (yaw).

1.4.3 ΦΑΙΝΟΜΕΝΟ “GIMBAL LOCK”

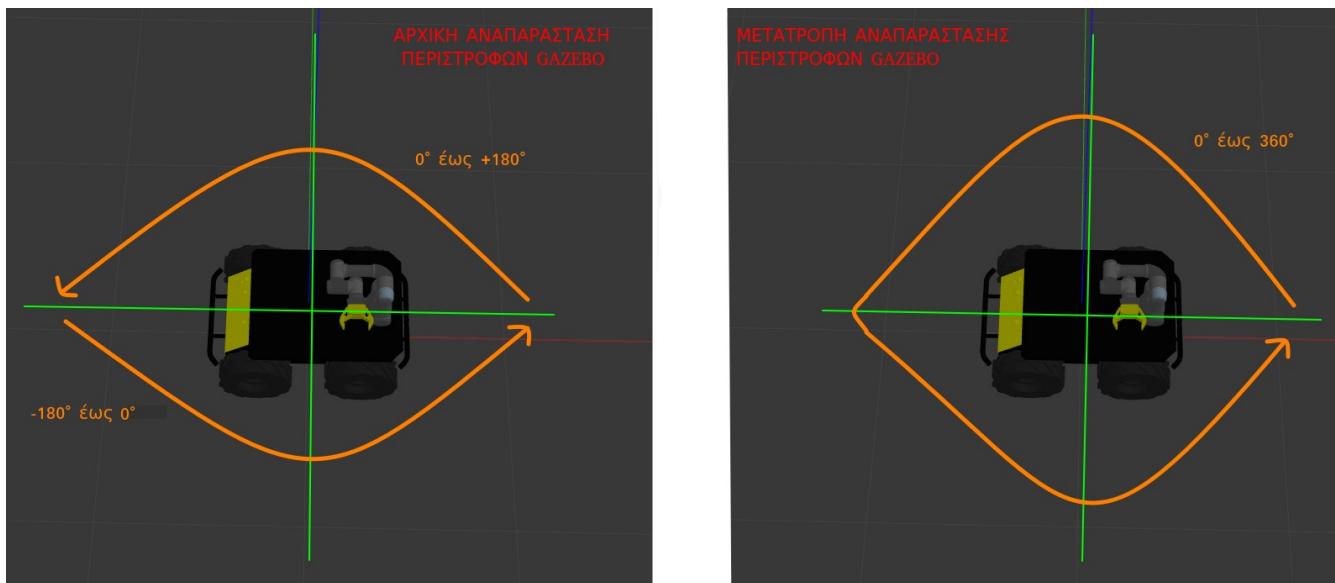
Οι γωνίες Euler υπόκεινται σε φαινόμενα όπως το "gimbal lock" που εμφανίζεται όταν δύο από τις τρεις γωνίες γίνουν ίσες, προκαλώντας απώλεια ελευθερίας περιστροφής. Προφανώς, το φαινόμενο αυτό αποκλείεται να συμβεί σε δισδιάστατο χώρο και αυτός είναι ένας ακόμα λόγος για τον οποίο χρησιμοποιούνται γωνίες euler στη συγκεκριμένη περίπτωση-εφαρμογή. Αντιθέτως, τα quaternions δεν υπόκεινται μεν στο "gimbal lock", παρ'όλα αυτά είναι μία πιό σύνθετη μαθηματική αναπαράσταση.



Εικόνα 1.11: Φαινόμενο “Gimbal lock” (περίπτωση b).

1.4.4 ΔΙΟΡΘΩΣΗ ΑΡΝΗΤΙΚΗΣ ΓΩΝΙΑΣ EULER

Έστερα από την μετατροπή της γωνίας euler από quaternions, γίνεται έλεγχος αν η γωνία euler έχει αρνητική τιμή. Αν συμβεί αυτό, τότε προσθέτουμε τον αριθμό 2π rad στην γωνία euler που ήδη έχουμε. Η προσθήκη αυτή είναι πολύ σημαντική, διότι το Gazebo χρησιμοποιεί γωνίες από 0° έως $+180^\circ$ και από -180° έως 0° για την αναπαράσταση περιστροφών, ενώ μιά καλύτερη αντιμετώπιση θα ήταν η χρήση γωνιών από 0° έως 360° κατευθείαν (εικόνα 1.12). Αν για παράδειγμα είχαμε μία γωνία με τιμή -140° , τότε η προσθήκη του αριθμού 2π rad, δηλαδή προσθήκη 360° (2π rad = 360°), μετατρέπει τη γωνία στις 220° , αφού $-140^\circ + 360^\circ = 220^\circ$. Με αυτόν τον τρόπο, η γωνία έχει μετατραπεί στο αντίστοιχο θετικό εύρος. Από την άλλη, αν η γωνία είναι θετική (0° έως 180°), δεν χρειάζεται να γίνει καμία αλλαγή.

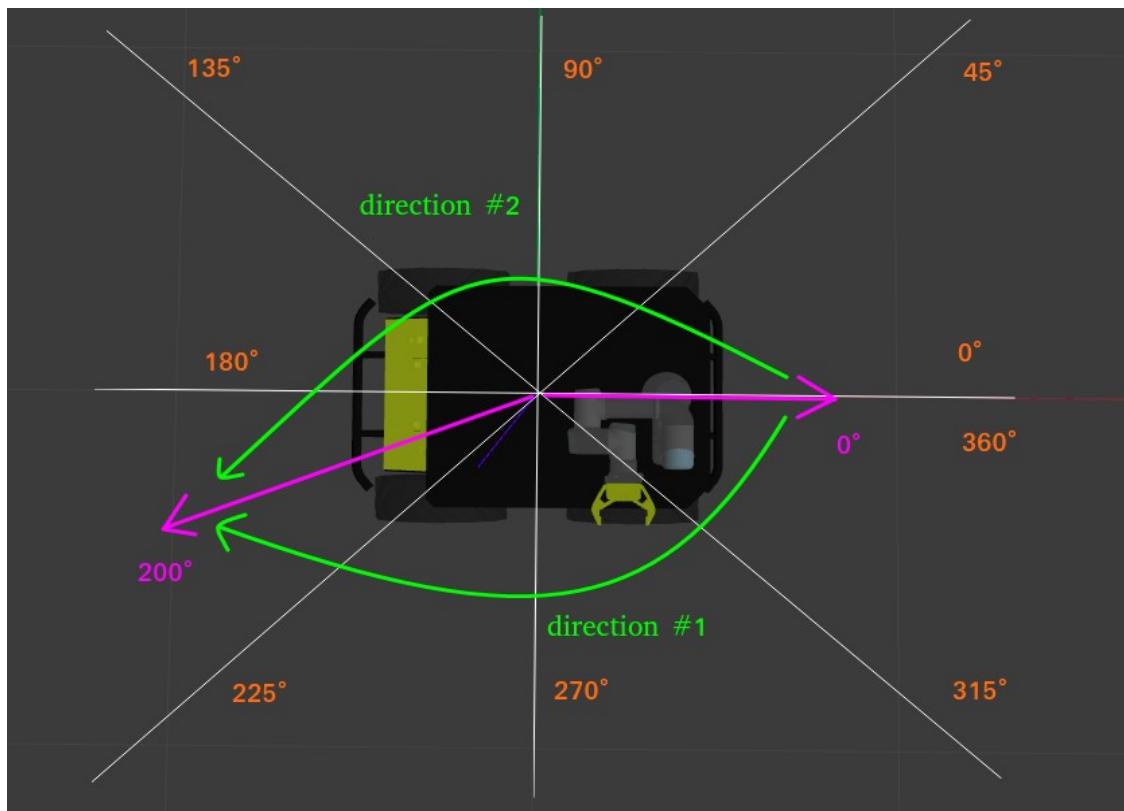


Εικόνα 1.12: Μετατροπή αναπαράστασης γωνιών περιστροφής.

1.4.5 ΚΑΤΕΥΘΥΝΣΗ ΠΕΡΙΣΤΡΟΦΗΣ

Ένα σημαντικό θέμα το οποίο τίθεται στην περιστροφή είναι η επιλογή κατεύθυνσης της (δεξιόστροφα ή αριστερόστροφα) κατά τον άξονα Z προς την επιθυμητή γωνία περιστροφής.

Για παράδειγμα, βλέποντας την εικόνα 1.13, το Husky έχει αρχική γωνία περιστροφής 0° ή 0 radians. Για να καταλήξει να έχει τελική γωνία περιστροφής ίση με 200° ή 3.49 radians, πρέπει να περιστραφεί προς τα δεξιά, καθώς η γωνία περιστροφής κατά τη δεξιά περιστροφή είναι μικρότερη σε σχέση με εκείνη κατά την αριστερή περιστροφή. Η δεξιά περιστροφή (direction #1) χρειάζεται μόλις 160° έναντι της αριστερής (direction #2) που απαιτεί 200° .



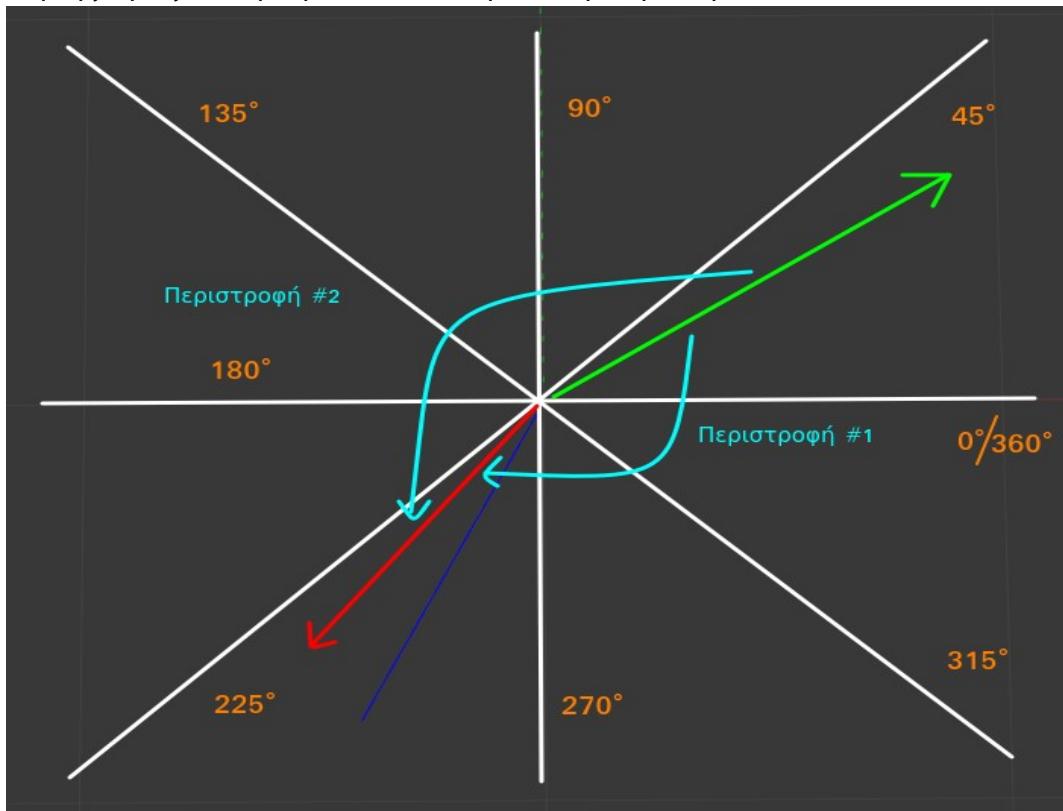
Εικόνα 1.13: Επιλογή κατεύθυνσης από τις 0° προς τις 200° .

Κατά συνέπεια, ο συνολικός χρόνος κατά αυτή τη περιστροφή είναι μικρότερος σε σχέση με το να περιστρεφόταν προς την αντίθετη κατεύθυνση κατά 40° , λαμβάνοντας υπόψιν την ίδια γωνιακή ταχύτητα και στις δύο περιπτώσεις.

1.4.6 ΕΚΤΙΜΗΣΗ ΥΠΟΛΟΓΙΣΜΟΥ ΚΑΤΕΥΘΥΝΣΗΣ ΠΕΡΙΣΤΡΟΦΗΣ

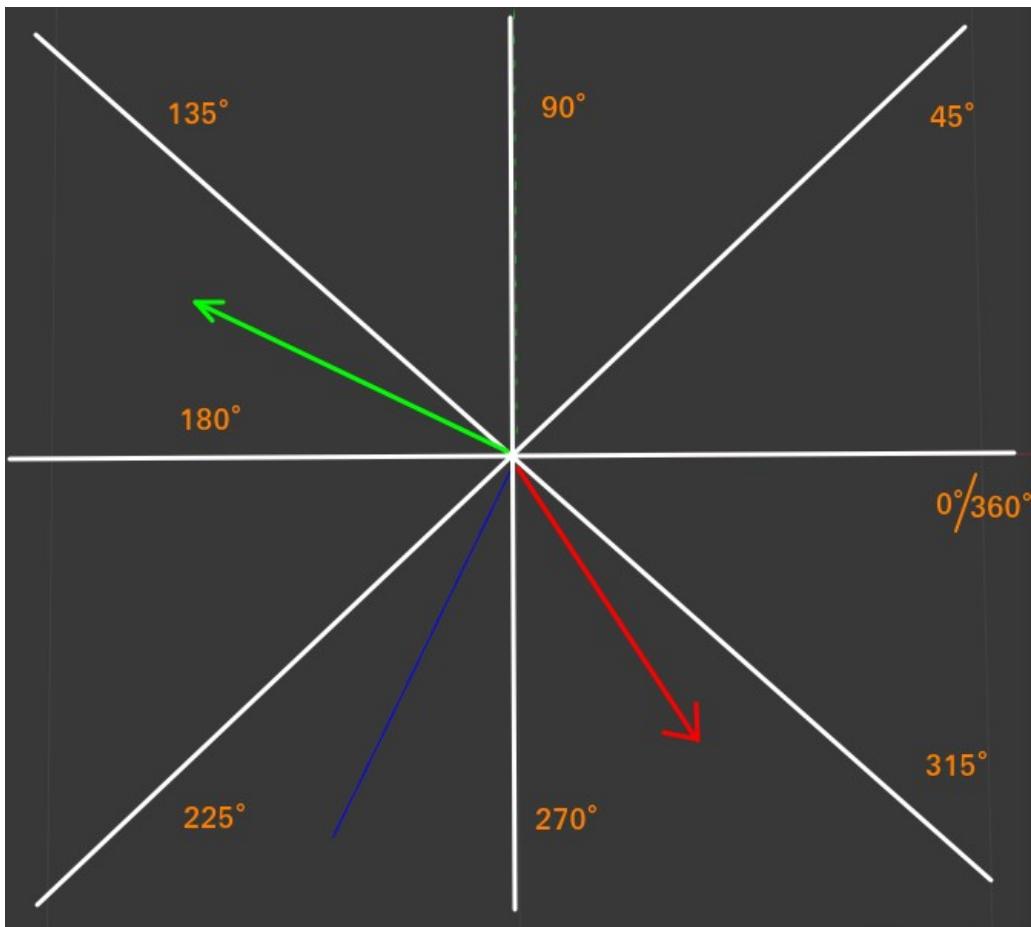
Για να επιτευχθεί το παραπάνω αποτέλεσμα, δηλαδή να εκτιμηθεί η κατεύθυνση περιστροφής του Husky προς την επιθυμητή-τελική γωνία περιστροφής, απαραίτητοι κρίνονται οι υπολογισμοί κάποιων συνθηκών, οι οποίοι θα αναλυθούν παρακάτω. Για το σκοπό αυτόν, χρειάζονται δύο μόνο δεδομένα. Η τρέχουσα και η τελική γωνία περιστροφής του Husky. Ας εξηγήσουμε τώρα τους κανόνες που διέπουν την επιλογή της κατεύθυνσης του ρομπότ προς την επιθυμητή γωνία περιστροφής. Χρειαζόμαστε συνολικά 40 ελέγχους με τους οποίους βλέπουμε σε ποιό οκταμόριο ανήκει τόσο η τρέχουσα όσο και η τελική γωνία περιστροφής και βάση αυτό αποφασίζουμε προς ποιά κατεύθυνση θα περιστραφεί το ρομπότ. Ας πάρουμε ένα παράδειγμα:

Η τρέχουσα γωνία είναι στο 1ο οκταμόριο και η τελική γωνία στο 6o. Σύμφωνα με αυτή τη συνθήκη, το όχημα θα περιστραφεί προς τα δεξιά ως προς τον άξονα Z, διότι το εύρος κίνησης προς αυτή τη κατεύθυνση είναι μικρότερο.



Εικόνα 1.14: Παράδειγμα #1 κατεύθυνσης περιστροφής οχήματος.

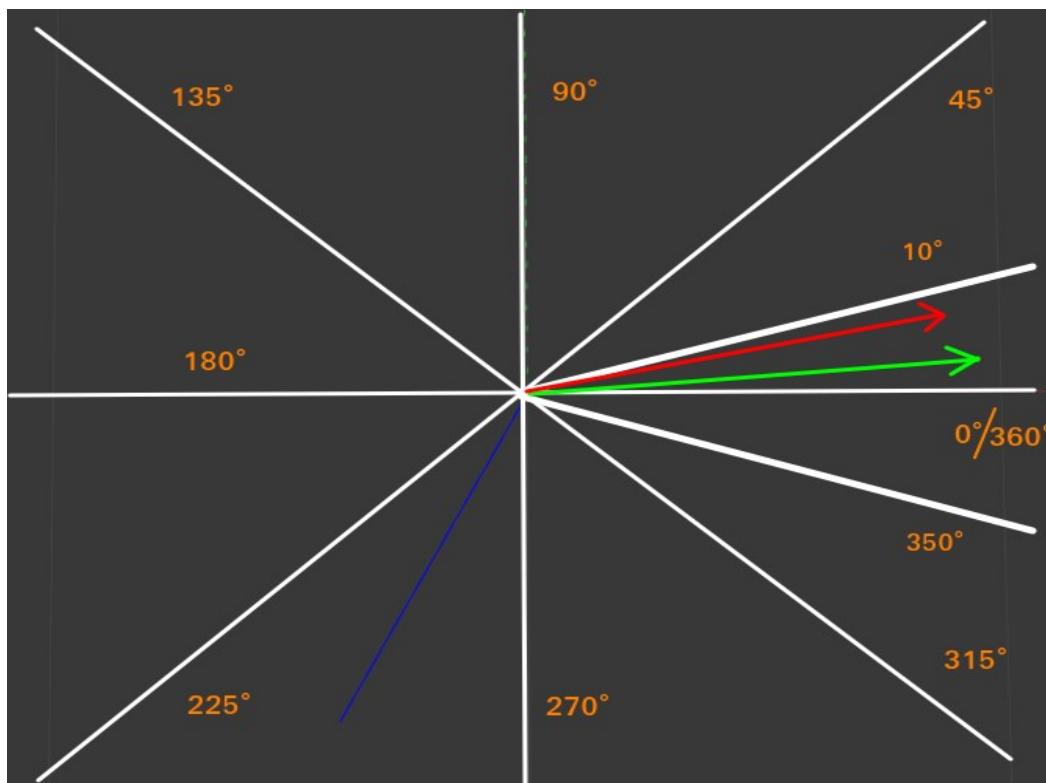
Ένα άλλο παράδειγμα είναι η τρέχουσα γωνία να βρίσκεται στο 4ο οκταμόριο και η τελική γωνία στο 7ο. Σύμφωνα με αυτή τη συνθήκη, το όχημα θα περιστραφεί προς τα αριστερά ως προς τον άξονα Z , διότι το εύρος κίνησης προς αυτή τη κατεύθυνση είναι μικρότερο, άρα και ο συνολικός χρόνος είναι μειωμένος.



Εικόνα 1.15: Παράδειγμα #2 κατεύθυνσης περιστροφής οχήματος.

Υπάρχει όμως ένας μικρός και σημαντικός περιορισμός, ο οποίος δεν σημειώθηκε. Η γωνία περιστροφής 0° συμπίπτει με αυτή των 360° . Οι 0° ανήκουν στο πρώτο τεταρτημόριο, ενώ οι 360° στο τέταρτο τεταρτημόριο. Σύμφωνα με τις παραπάνω συνθήκες-περιπτώσεις, το Husky θα πάθαινε μία είδους ταλάντωση, αφού θα περιστρεφόταν προς τη μία κατεύθυνση και μία προς την αντίθετη ταυτόχρονα. Θα συνέβαινε, δηλαδή, μιά είδους ταλάντωση και θα ήταν αναποφάσιστο να περιστραφεί προς μία συγκεκριμένη κατεύθυνση προς τον αντίστοιχο στόχο. Για να επιλυθεί το πρόβλημα αυτό, έγινε ένας ακόμη συμβιβασμός, ο οποίος έλυσε το πρόβλημα αυτό.

Λήφθηκε ένα όριο μεταξύ 10° και 350° και εντός αυτού του ορίου, πάρθηκαν κάποιες αποφάσεις. Το προαναφερθές διάστημα αναφέρει ότι η τρέχουσα γωνία περιστροφής είναι μικρότερη από 10° ή μεγαλύτερη από 350° . Ουσιαστικά, βρίσκεται σε ένα από τα δύο κλειστά διαστήματα: $[0^\circ, 10^\circ]$ ή $[350^\circ, 360^\circ]$. Τότε, εκτελείται μία κίνηση τύπου zig-zag προκειμένου να διορθωθεί το σφάλμα προσανατολισμού και το όχημα να έρθει στον επιθυμητό προσανατολισμό. Αν η κίνηση διαρκέσει πάνω από 7 δευτερόλεπτα, διακόπτεται ώστε να μην συνεχιστεί επ' αόριστον.



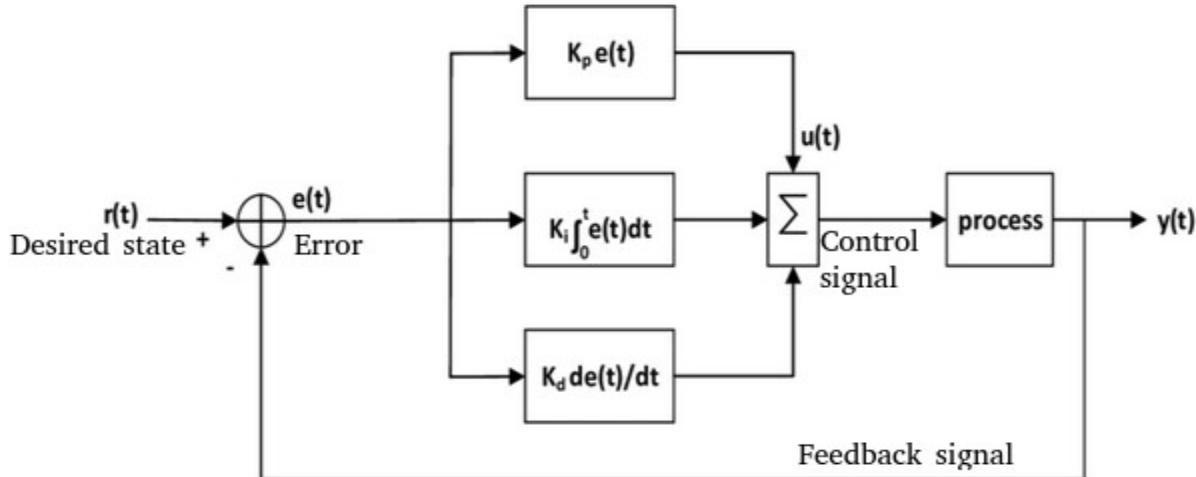
Εικόνα 1.16: Τρέχουσα και τελική γωνία στο διάστημα $[0^\circ, 10^\circ]$.

1.4.7 PID ΕΛΕΓΚΤΗΣ

Ο PID ελεγκτής είναι ένας τύπος ελεγκτή που χρησιμοποιείται για τον έλεγχο συστημάτων, ώστε να διατηρούν μια επιθυμητή κατάσταση. Η λέξη "PID" αντιπροσωπεύει τους τρεις βασικούς όρους του ελεγκτή:

- **Αναλογικός (Proportional - P):** Ο αναλογικός όρος αντιδρά στην τρέχουσα διαφορά μεταξύ της επιθυμητής τιμής και της πραγματικής κατάστασης. Αυτός ο όρος είναι ανάλογος με το μέγεθος του σφάλματος. Με λίγα λόγια, μας λέει το πόσο γρήγορα θέλουμε να πάμε στο στόχο μας (από μία αρχική σε μία τελική κατάσταση).
- **Ολοκληρωτικός (Integral - I):** Ο ολοκληρωτικός όρος λαμβάνει υπόψη το ολοκλήρωμα του σφάλματος με την πάροδο του χρόνου. Αυτός ο όρος διορθώνει τυχόν μόνιμα σφάλματα. Ουσιαστικά, μας λέει το πόσο μικρό θα είναι το σφάλμα αρχικής-τελικής θέσης, οπότε όσο μικρότερο σφάλμα, τόσο πιό κοντά θα φτάσει το ρομπότ στο τελικό του στόχο.
- **Διαφορικός (Derivative - D):** Ο διαφορικός όρος αντιδρά στο ρυθμό μεταβολής του σφάλματος. Βοηθά στην αντιμετώπιση των απότομων αλλαγών στην κατάσταση. Με απλά λόγια, μειώνει αποτελεσματικά τις ταλαντώσεις που δημιουργούνται κατά τη κίνηση του ρομπότ.

Ο PID ελεγκτής χρησιμοποιείται σε ρομποτικά οχήματα, όπως το Husky, για τον έλεγχο της κίνησης και της κατεύθυνσής τους. Επιπρόσθετα, μπορεί να προσαρμόσει την έξοδο του συστήματος (στην περίπτωση του Husky, των κινητήρων και των τροχών) με βάση τη διαφορά, το ολοκληρωτικό σφάλμα και τον ρυθμό μεταβολής, επιτυγχάνοντας έτσι σταθερότητα και ακρίβεια κατά την κίνηση του ρομπότ.



Εικόνα 1.17: Σύστημα ελέγχου κλειστού βρόχου με ελεγκτή PID.

Η παραπάνω εικόνα 1.17 δείχνει την κατάσταση $r(t)$ στην οποία εισάγεται ένα σφάλμα, το οποίο ο ελεγκτης προσπαθεί να διορθώσει μέσω των τριών όρων που αναφέρθηκαν προηγουμένως, δηλαδή τον αναλογικό, τον ολοκληρωτικό και τον διαφορικό όρο. Τελικά, αυτοί οι τρεις όροι δίνουν μία έξοδο στο σύστημα (οι έξοδοι εδώ θεωρούνται οι κινητήρες για κάθε τροχό του Husky). Η έξοδος με τη σειρά της εισάγεται και πάλι στην κατάσταση $r(t)$, κλείνοντας έτσι τον βρόγχο και με αυτόν το τρόπο ακολουθεί η ίδια διαδικασία από την αρχή.

Το Husky έχει στη διάθεση του τέσσερις κινητήρες και τροχούς, οπότε για κάθε κινητήρα χρειάζεται να εισάγουμε τους τρεις όρους που αναλύσαμε προηγουμένως.

Ο κάθε τροχός λοιπόν έχει,

- ◆ για τον αναλογικό όρο τιμή ίση με **20.0**
- ◆ για τον αναλογικό όρο τιμή ίση με **70.0**
- ◆ για τον αναλογικό όρο τιμή ίση με **00.0**

Οι τιμές αυτές βρέθηκαν ύστερα από πάρα πολλές δοκιμές προκειμένου το ρομπότ να ανταποκρίνεται σωστά σε όλες τις κινήσεις του (περιστροφή και μετατόπιση) με όσο το δυνατόν λιγότερα σφάλματα.

1.5 ΠΡΟΣΑΝΑΤΟΛΙΣΜΟΣ HUSKY

1.5.1 ΈΛΕΓΧΟΣ ΠΡΟΣΑΝΑΤΟΛΙΣΜΟΥ

Όπως έχουμε σε προηγούμενο υποκεφάλαιο, στόχος του Husky είναι να πάει κοντά σε τραπέζι, ώστε στη συνέχεια ο βραχίονας UR3 να μπορεί να φτάσει το τραπέζι και να κάνει με ασφάλεια και άνεση τη δουλειά του, αρπάζοντας ή αφήνοντας ενα αντικείμενο εκεί. Υπάρχουν τέσσερις περιπτώσεις με τις οποίες το ρομπότ Husky μπορεί να προσεγγίσει ένα τραπέζι, βλέποντας το σε διάταξη κατόψεως:

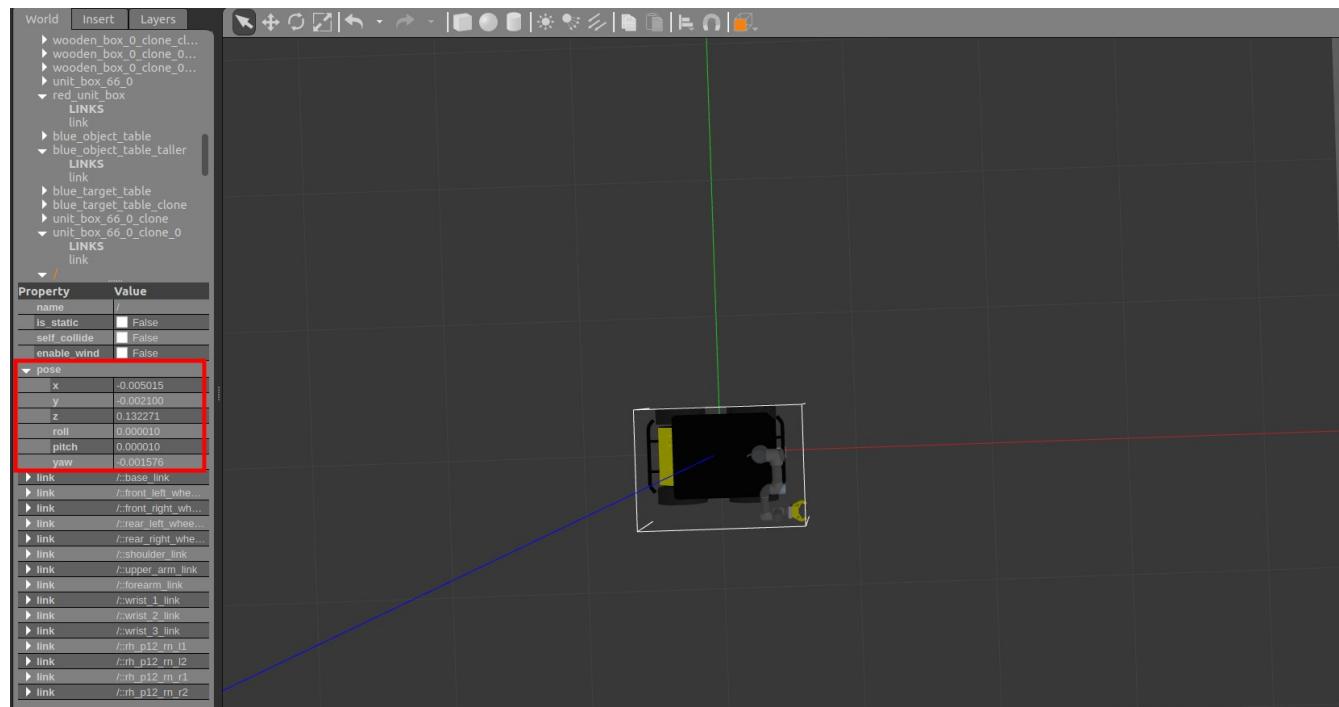
- Από κάτω
- Από αριστερά
- Από πάνω
- Από δεξιά

Το Husky έχει ως δεδομένο αυτούς τους προσανατολισμούς από κάθε τραπέζι στον 3D χώρο του Gazebo, και έτσι μπορεί να προσεγγίσει το καθένα κάθετα προς αυτό, έχοντας το δηλαδή “από κάτω” ή “από αριστερά” ή “από πάνω” ή “από δεξιά”. Η παρακάτω εικόνα δίνει μία καλύτερη οπτική στην παραπάνω περιγραφή.

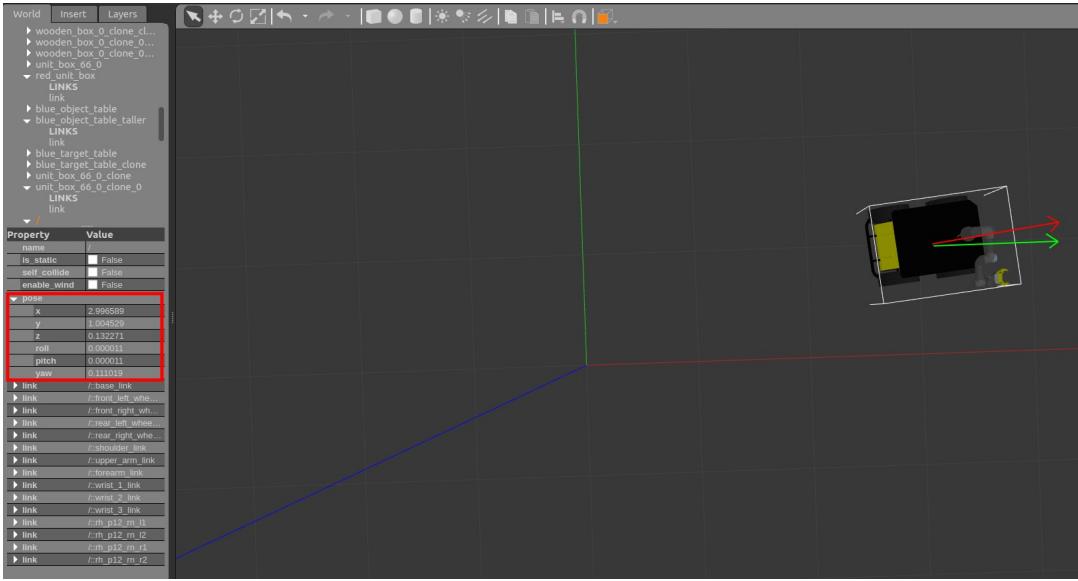


Εικόνα 1.18: Το Husky βρίσκεται στο στόχο του προσεγγίζοντας τον κάθετα “από κάτω”.

Επίσης, να σημειώσουμε πως μετά από κάθε κίνηση του ρομπότ (όπως η μετατόπιση ή η περιστροφή του), το ίδιο θα πρέπει να ευθυγραμμίσει και να διορθώσει τυχόν σφάλματα του προσανατολισμού του. Για παράδειγμα, βλεποντας παρακάτω την εικόνα 1.19 το Husky ξεκινάει από τη θέση (0 , 0) με αρχική γωνία περιστροφής τις 0 μοίρες και παρατηρώντας την εικόνα 1.20 θέλει να καταλήξει στη θέση (3 , 1) με τελική γωνία περιστροφής πάλι τις 0 μοίρες. Μπορεί η τελική γωνία περιστροφής να ήταν διαφορετική, όπως οι 90 μοίρες ή οι 180 μοίρες ή οι 270 μοίρες, δεν έχει σημασία. Σημασία έχει ότι οποιαδήποτε κίνηση προϋποθέτει σφάλματα κατά τη περιστροφή ή μετατόπιση, οπότε θα χρειαστεί διόρθωση.

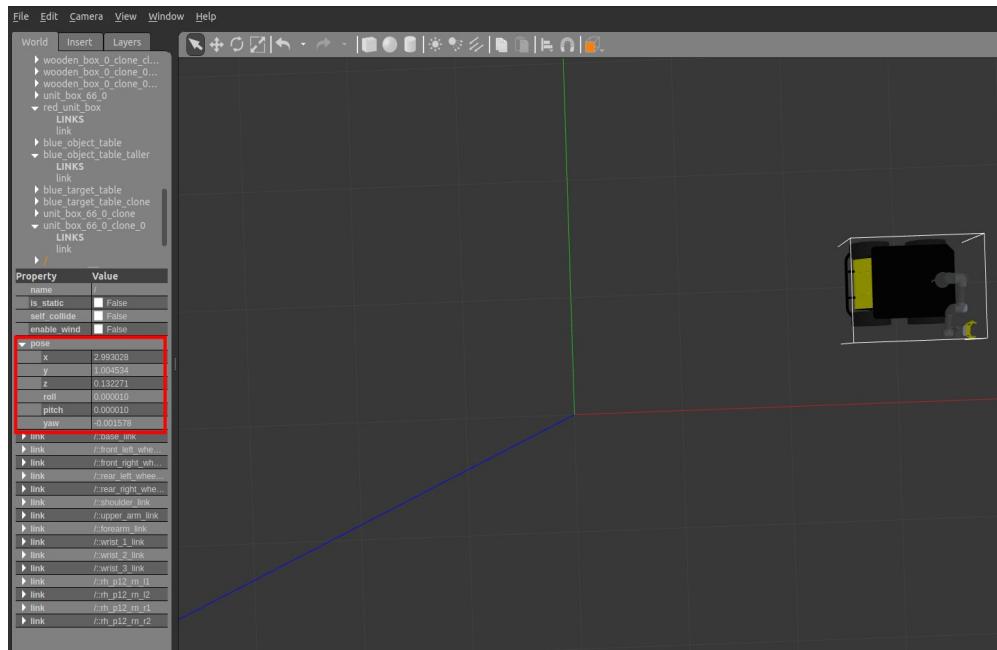


Εικόνα 1.19: Το Husky βρίσκεται στη θέση (0 , 0) με αρχική γωνία περιστροφής 0 μοίρες.



Εικόνα 1.20: Λάθος προσανατολισμός στη θέση (3 , 1) (Yaw: 0.11 rad αντί για 0 rad).

Το κόκκινο βελάκι δείχνει τον εσφαλμένο προσανατολισμό (0.11 rad) που έχει το Husky, ο οποίος δημιουργήθηκε από συνεχή μετάδοση σφαλμάτων κατά τη μετατόπιση και περιστροφή του για να φτάσει εκεί. Αντίθετα, το πράσινο βελάκι δείχνει τον αναμενόμενο και ορθό προσανατολισμό (0 rad) που θα έπρεπε να έχει. Η λύση έρχεται με την εκτέλεση κίνησης zig-zag που αναφέρθηκε παραπάνω.



Εικόνα 1.21: Το Husky βρίσκεται στη θέση (3 , 1)
με ορθό προσανατολισμό (Yaw: 0 rad).

1.5.2 ΔΙΟΡΘΩΣΗ ΠΡΟΣΑΝΑΤΟΛΙΣΜΟΥ

Όπως είδαμε από τις εικόνες 1.20 και 1.21, συγκρίνοντας τες μπορεί να υπάρχει μία μικρή απόκλιση στη τελική γωνία περιστροφής, δηλαδή του προσανατολισμού του Husky, το οποίο μπορεί να επηρεάσει τις μετέπειτα μετακινήσεις ή περιστροφές του ή ακόμα και τις κινήσεις του UR3, εάν αυτό πρόκειται να ενεργοποιηθεί, ώστε να εκτελέσει την εργασία του. Για το σκοπό αυτό, δηλαδή την διόρθωση αυτου του σφάλματος της τιμής του προσανατολισμού του ρομπότ, χρειάστηκε να ληφθούν απαραίτητα μέτρα, τα οποία θα εξηγηθούν αναλυτικά παρακάτω.

Υστερα από κάθε μετακίνηση ή περιστροφή του Husky, τη στιγμή που ακινειτοποιείται από την ενέργεια αυτή, γίνεται έλεγχος του προσανατολισμού του. Όπως έχουμε πει και σε προηγούμενη υποενότητα το ρομπότ έχει ως δεδομένα 2 τιμές, την τρέχουσα και τη τελική γωνία περιστροφής του. Όσο η τρέχουσα γωνία περιστροφής είναι εκτός ορίων της τελικής γωνίας περιστροφής με μία ανεκτή και μικρή απόκλιση της τάξεως των 0.005 rad, τότε περιστρέφουμε το ρομπότ προς τη σωστή κατεύθυνση με τη βοήθεια του topic “cmd_vel” που αναφέρθηκε προηγουμένως, ελέγχοντας συνεχώς για το αν είμαστε εντός ή εκτός ορίων. Την σωστή κατεύθυνση του Husky την ορίσαμε και την διατυπώσαμε αναλυτικά στην υποενότητα 1.4.6 .

Πρέπει δηλαδή να ισχύει το εξής, ώστε να είμαστε εντός ορίων:

$$\text{ΤΕΛΙΚΗ ΓΩΝΙΑ} - 0.005 \leq \text{ΤΡΕΧΟΥΣΑ ΓΩΝΙΑ} \leq \text{ΤΕΛΙΚΗ ΓΩΝΙΑ} + 0.005$$

(Μονάδα μέτρησης: radians)

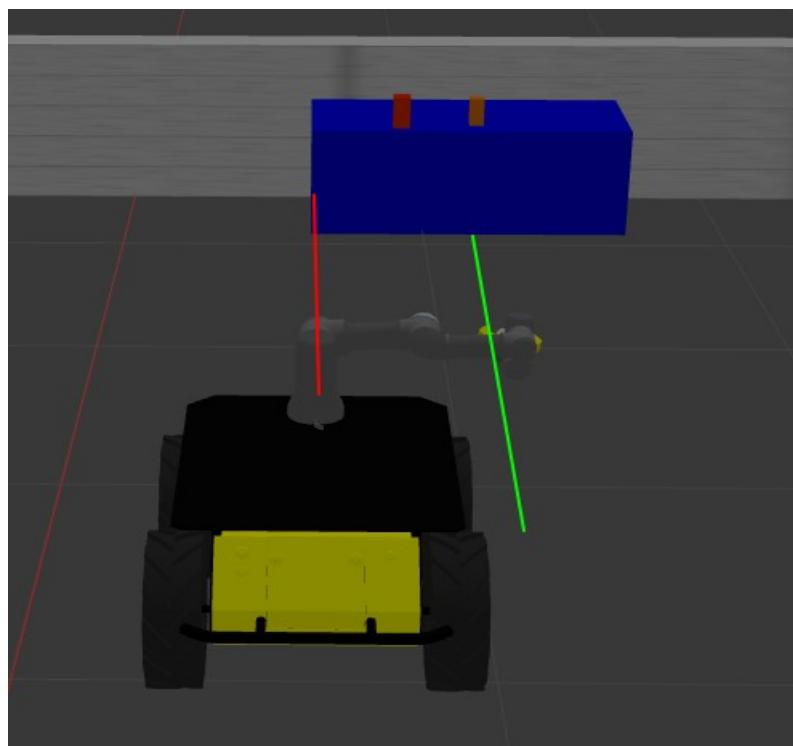
Από την άλλη, εάν η απόλυτη διαφορά μεταξύ τρέχουσας γωνίας περιστροφής και τελική γωνία περιστροφής είναι μικρότερη από τη τιμή 0.3 ή μεγαλύτερη από 6.0, τότε η γωνιακή ταχύτητα περιστροφής θα γίνει 0.02 rad/sec. ή -0.02 rad/sec. (ανάλογα τη φορά περιστροφής), διότι σε εκείνη τη περιοχή οι γωνίες 0 μοίρες και 360 μοίρες συμπίπτουν και δημιουργείται το πρόβλημα που αναπτύξαμε στο υποκεφάλαιο 1.4.5 . Οπότε, το ρομπότ θα περιστρέφεται με χαμηλή γωνιακή ταχύτητα στη περιοχή αυτή, ώστε να

προλάβει να σταματήσει ακριβώς στις 0 μοίρες και να μην υπάρξουν περαιτέρω ταλαντώσεις.

Αντίστοιχα, στις υπόλοιπες περιοχές που δεν υπάρχει θέμα σύμπτωσης σε γωνίες όπως οι 0 και 360 μοίρες, η γωνιακή ταχύτητα είναι σαφώς μεγαλύτερη και έχει τιμή ίση με 0.15 rad/sec. ή -0.15 rad/sec. (ανάλογα τη κατεύθυνση περιστροφής).

1.5.3 ΚΙΝΗΣΗ

Για να πραγματοποιηθεί οποιαδήποτε κίνηση του ρομπότ προς κάποιον στόχο, πρέπει να του δοθούν τα απαραίτητα δεδομένα, τα οποία αφορούν τον προσανατολισμό με τον οποίο πρέπει να προσεγγίσει το στόχο (τραπέζι), καθώς και τις συντεταγμένες του τραπεζιού στο τρισδιάστατο χώρο του Gazebo. Επίσης, όπως είχαμε πει, το ρομπότ Husky κατά τη μετακίνηση ή περιστροφή του στο χώρο εμφανίζει σφάλματα, οπότε αποκλίνει από το στόχο του. Έτσι, μπορεί να βρίσκεται είτε πιό αριστερά είτε πιό δεξιά από ένα τραπέζι-στόχο, όπως φαίνεται στην εικόνα 1.22 .



Εικόνα 1.22: Απόκλιση Husky από τραπέζι-στόχο (βρίσκεται πιό αριστερά).

Θα έπρεπε να βρίσκεται στην πράσινη ευθεία με ίσως μιά ικανοποιητικά μικρή απόκλιση, όπως φαίνεται στην προηγούμενη εικόνα. Παρακάτω φαίνεται η ορθή στάση του Husky απέναντι στο τραπέζι-στόχο. Πρέπει δηλαδή το Husky να βρίσκεται περίπου στο κέντρο του τραπεζιού, ώστε στη συνέχεια ο βραχίονας UR3 να έχει την σχετική άνεση να πίασει ή να αφήσει ένα αντικείμενο στο τραπέζι.



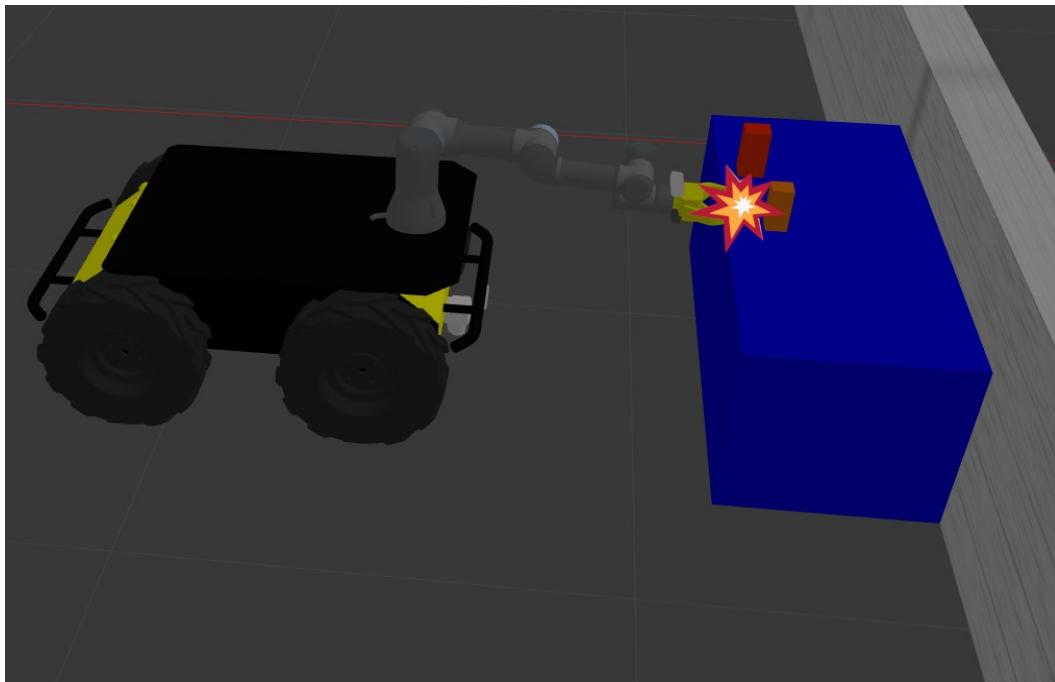
Εικόνα 1.23: Το Husky σε σωστή απόσταση από το κέντρο του τραπεζιού-στόχου.

Εάν λοιπόν προκύψει σφάλμα και υπάρξει απόκλιση από το κέντρο του τραπεζιού, τότε πρέπει να ενεργοποιηθεί η κάμερα του Husky, σκοπός της οποίας είναι να σκανάρει το τραπέζι και με έναν αλγόριθμο να φέρει το όχημα όσο πιο κοντά στο κέντρο μπορεί. Αυτή η σάρωση θα περιγραφεί αναλυτικά σε επόμενο υποκεφάλαιο. Για να πραγματοποιηθεί όμως αυτή η σάρωση, θα πρέπει το όχημα να έχει μία απόσταση ασφαλείας από το

τραπέζι. Αυτή η απόσταση χρειάζεται ώστε το Husky να προλάβει να έρθει στο κέντρο του τραπεζιού προτού πλησιάσει αρκετά σε αυτό, με αποτέλεσμα να μην τα καταφέρει.

Έτσι, το Husky πρέπει να σταματήσει σε απόσταση 3.5 μέτρων ακριβώς πριν το στόχο.

Αυτή η απόσταση ασφαλείας των 3.5 μέτρων βγήκε ύστερα από αρκετές προσπάθειες σάρωσης μέσω της κάμερας του Husky, ώστε να έρθει όσο πιο κοντά γίνεται στο κέντρο του τραπεζιού. Ας συνεχίσουμε όμως με τη διαδικασία κίνησης του ρομποτικού οχήματος Husky και ας αφήσουμε προς το παρόν τη σάρωση για επόμενη υποενότητα. Οποιαδήποτε στιγμή ο ρομποτικός βραχίονας UR3 μπορεί να έχει επεκταθεί προς οποιαδήποτε κατεύθυνση σε οποιοδήποτε σημείο. Οπότε, πριν από κάθε κίνηση του Husky, θα πρέπει ο βραχίονας UR3 να έρθει σε μία θέση, η οποία δεν θα εμποδίζει το Husky να εκτελεί τις εργασίες του. Για παράδειγμα, όταν το όχημα κινείται και ο βραχίονας είναι τελείως εκτεταμένος στο εμπρόσθιο τμήμα του οχήματος Husky, αυτό θα εμπόδιζε το όχημα, διότι θα μπορόυσε ο βραχίονας να χτυπήσει σε κοντινό, μπροστινό εμπόδιο ή τραπέζι. Ας παρατηρήσουμε την παρακάτω εικόνα για να γίνει καλύτερα κατανοητό το νόημα της παραπάνω πρότασης.

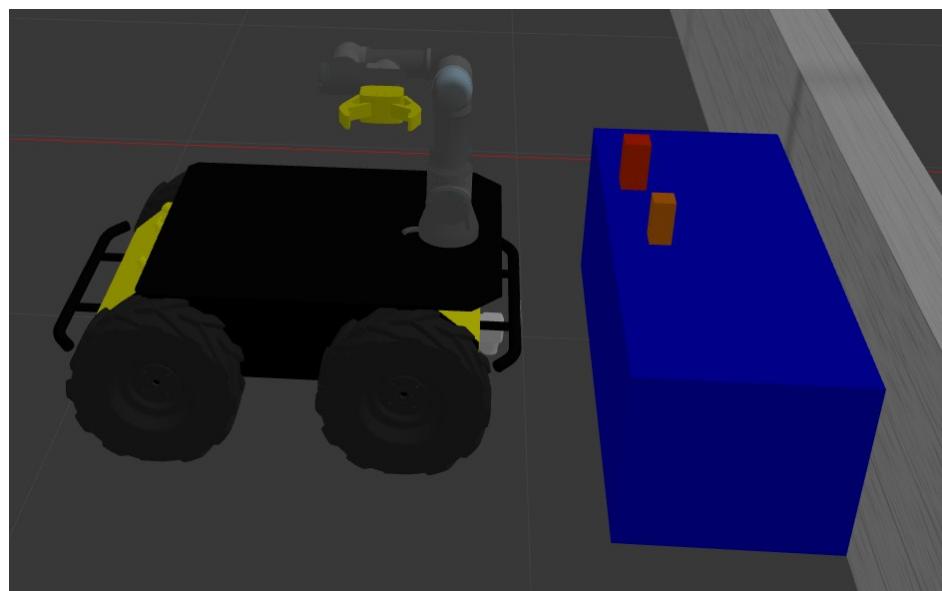


Εικόνα 1.24: Ο UR3 εμποδίζει το Husky να έρθει σε κοντινή απόσταση από το τραπέζι.

Μιά σωστή αντιμετώπιση του προβλήματος αυτού θα ήταν πριν ξεκινήσει το Husky, να φέρνουμε τον βραχίονα σε μία θέση ακριβώς πάνω από το όχημα, όπου δεν θα εμπόδιζε την εργασία του Husky. Την θέση αυτή την ονομάζουμε “Home position”. Πάντα όμως γίνεται έλεγχος αν είναι ήδη στην “Home position”, αλλιώς δεν υπάρχει λόγος να μετακινηθεί ο UR3, εφόσον εκεί δεν εμποδίζει. Παρακάτω στην εικόνα 1.25 παρατηρούμε την “Home position” του βραχίονα UR3.



Εικόνα 1.25: Ο UR3 σε θέση “Home position” που δεν εμποδίζει το όχημα.



Εικόνα 1.26: Ο UR3 σε θέση “Home position” κοντά στο στόχο-τραπέζι.

Έχει η κίνηση του οχήματος στο 3D χώρο και συγκεκριμένα σε διάφορους στόχους. Όπως είπαμε πιό πριν, το Husky χρειάζεται ως δεδομένα τον προσανατολισμό με τον οποίο θα καταλήξει σε έναν στόχο καθώς και τις συντεταγμένες του.

Ας ονομάσουμε ως 'a' τη συντεταγμένη X και ως 'b' τη συντεταγμένη Y του τραπεζιού.

Ο προσανατολισμός του Husky, δηλαδή από ποιά μεριά του τραπεζιού θα προσεγγίσει το ρομπότ το τραπέζι (από δεξιά ή από αριστερά ή από πάνω ή από κάτω, όπως εξηγήθηκε σε προηγούμενο υποκεφάλαιο) δίνεται ως δεδομένο για κάθε τραπέζι ξεχωριστά στο χώρο. Επίσης, το ρομποτικό όχημα έχει και ως επιπλέον δεδομένα, τις συντεταγμένες θέσης του την εκάστοτε χρονική στιγμή, καθώς και την τρέχουσα και τελική γωνία περιστροφής. Η τρέχουσα γωνία περιστροφής δίνεται από το topic "/odometry/filtered" όπως είχαμε πει, ενώ η τελική γωνία περιστροφής υπολογίζεται ως εξής:

$$(1) \quad f = \text{atan2}(m, n), \text{όπου}$$

$$(2) \quad m = a - x + s$$

$$(3) \quad n = b - y + s$$

ΣΥΝΤΕΤΑΓΜΕΝΗ ΘΕΣΗΣ ΣΤΟΧΟΥ ΩΣ ΠΡΟΣ ΑΞΟΝΑ X = 'a'

ΣΥΝΤΕΤΑΓΜΕΝΗ ΘΕΣΗΣ ΣΤΟΧΟΥ ΩΣ ΠΡΟΣ ΑΞΟΝΑ Y = 'b'

ΤΡΕΧΟΥΣΑ ΣΥΝΤΕΤΑΓΜΕΝΗ ΘΕΣΗΣ ΩΣ ΠΡΟΣ ΑΞΟΝΑ Y = 'x'

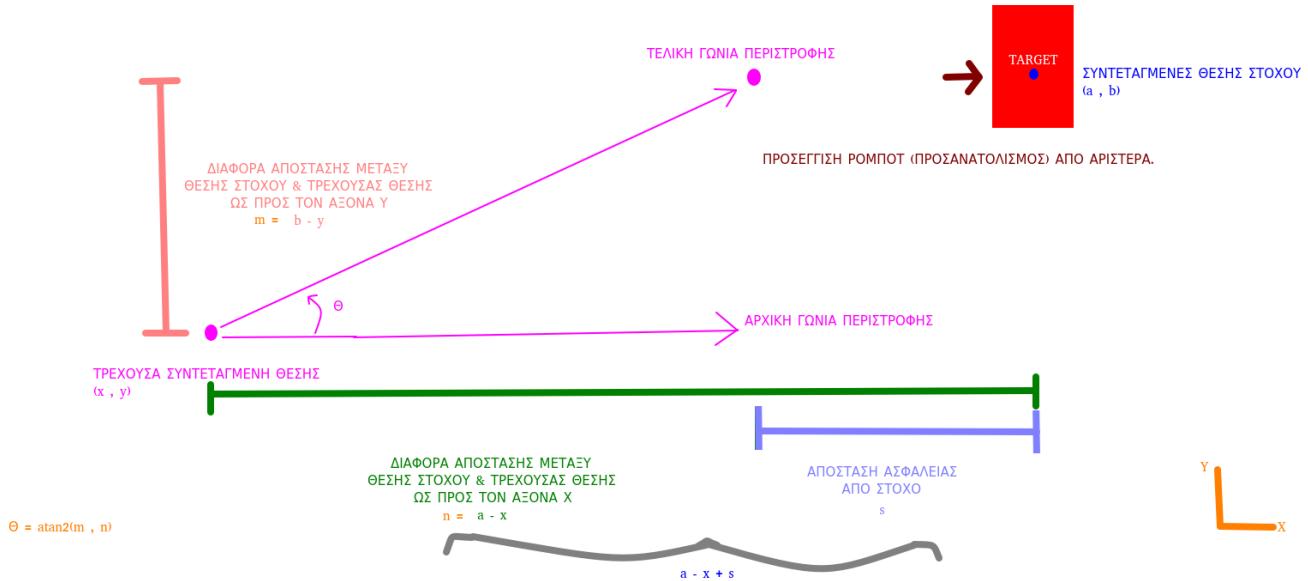
ΤΡΕΧΟΥΣΑ ΣΥΝΤΕΤΑΓΜΕΝΗ ΘΕΣΗΣ ΩΣ ΠΡΟΣ ΑΞΟΝΑ Y = 'y'

ΑΠΟΣΤΑΣΗ ΑΣΦΑΛΕΙΑΣ ΑΠΟ ΤΟΝ ΣΤΟΧΟ = 's'

ΤΡΕΧΟΥΣΑ ΓΩΝΙΑ ΠΕΡΙΣΤΡΟΦΗΣ = 'c'

ΤΕΛΙΚΗ ΓΩΝΙΑ ΠΕΡΙΣΤΡΟΦΗΣ = 'f'

Όπως είχαμε αναφέρει, αν η τελική γωνία περιστροφής έχει αρνητική τιμή, τότε προστίθεται στη τιμή της, η τιμή 2π. Ας εξηγήσουμε και αναπαραστήσουμε πως προκύπτουν οι τύποι **(1)**, **(2)** και **(3)**.

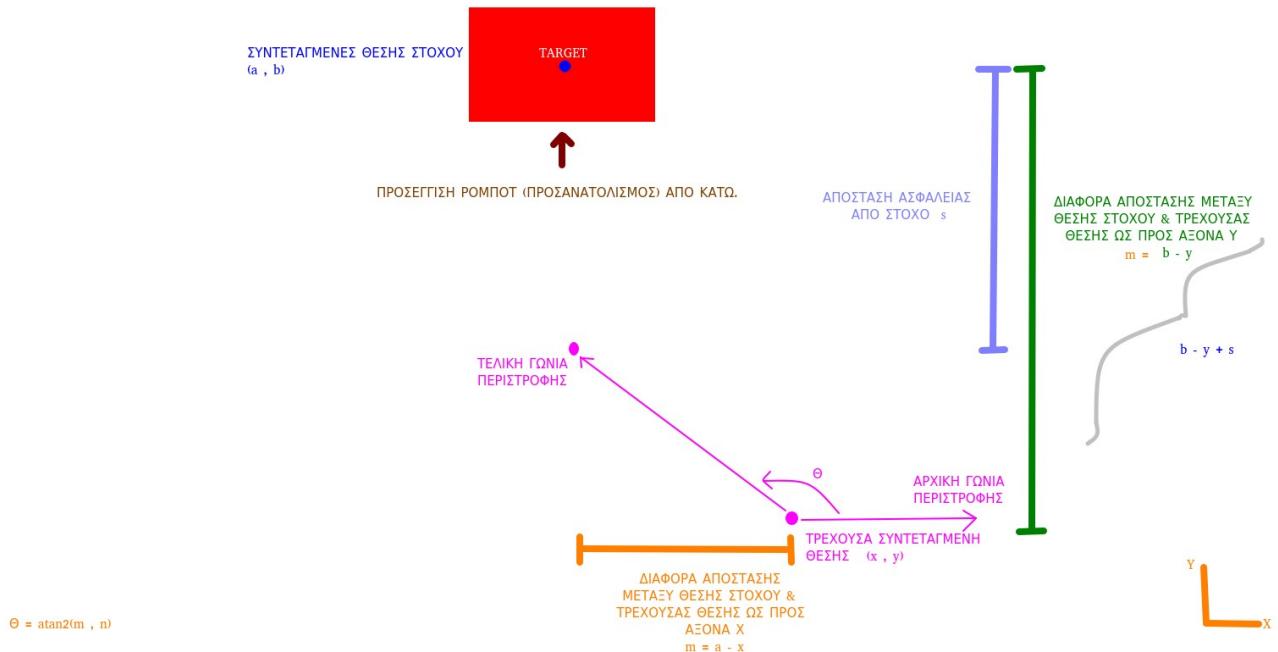


Εικόνα 1.27: Υπολογισμός τύπου περιστροφής Husky με προσεγγιση στόχου (καφέ βελάκι) από αριστερά.

Όπως είναι προφανές, δεν χρειάζεται απόσταση ασφαλείας στον άξονα Y , διότι το ρομπότ προσεγγίζει το στόχο-τραπέζι κάθετα από αριστερά και πρέπει να συμπέσει στο κέντρο του τραπεζιού με όσο το δυνατόν μικρότερη απόκλιση.

Αντίθετα, παρακάτω θα δείξουμε ότι δεν χρειάζεται απόσταση ασφαλείας στον άξονα X όταν το ρομπότ προσεγγίζει το τραπέζι από κατω.

Γενικά, δεν χρειάζεται απόσταση ασφαλείας στον άξονα X όταν το ρομπότ προσεγγίζει κάθετα το τραπέζι-στόχο από κάτω ή από πάνω και στον άξονα Y όταν το ρομπότ προσεγγίζει κάθετα το τραπέζι-στόχο από αριστερά ή από δεξιά.



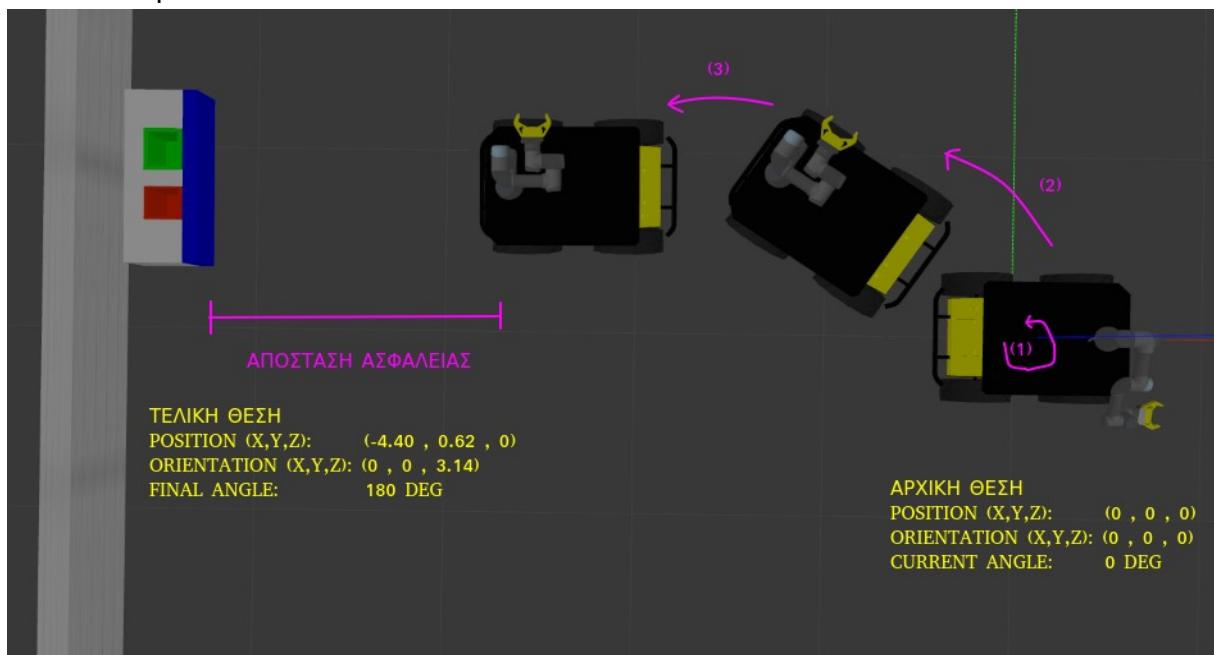
Εικόνα 1.28: Υπολογισμός τύπου περιστροφής Husky με προσεγγιση στόχου (καφέ βελάκι) από κάτω.

Γενικότερα, εάν η απόλυτη διαφορά μεταξύ τρέχουσας και τελικής γωνίας περιστροφής είναι μεγαλύτερη από 10 μοίρες, τότε το ρομπότ δεν μετακινείται αλλά μόνο περιστρέφεται προς την επιθυμητή γωνία στόχου σύμφωνα με τη σωστή κατέυθυνση που έχουμε ορίσει σε προηγούμενη υποενότητα με γωνιακή ταχύτητα 0.12 rad/sec. ή -0.12 rad/sec. . Από την άλλη, εάν η διαφορά αυτή είναι μικρότερη από 10 μοίρες ή εάν η τρέχουσα και τελική γωνία περιστροφής βρίσκονται στα διαστήματα $[0^\circ, 15^\circ]$ ή $[345^\circ, 360^\circ]$, τότε θα υπάρξει μία μικρή περιστροφή προς τη σωστή κατέυθυνση με γωνιακή ταχύτητα 0.08 rad/sec ή -0.08 rad/sec με μία ταυτόχρονη μετακίνηση προς τα εμπρός με γραμμική ταχύτητα 0.4 m/s. .

Αν δεν ισχύει τίποτα από τα παραπάνω, τότε θα πρέπει να υπάρξει μόνο μία μικρή μετακίνηση προς τα εμπρός με γραμμική ταχύτητα 0.4 m/sec. Χωρίς περιστροφή του ρομποτικού οχήματος, διότι ο στόχος βρίσκεται στην ίδια ευθεία με το ρομπότ.

Τέλος, αν απόλυτη διαφορά στις συντεταγμένες X ή Y στόχου και ρομπότ είναι μικρότερη από τη τιμή 0.3, τότε η γραμμική ταχύτητα μειώνεται από το 0.4 m/s. Στο 0.1 m/s. για να έχει την δυνατότητα να σταματήσει εγκαίρως. Για παράδειγμα, αν η τρέχουσα συντεταγμένη X του ρομπότ έχει τη τιμή 1 και η συντεταγμένη του στόχου στον άξονα X έχει τη τιμή 4, τότε το ρομπότ θα προχωράει με γραμμική ταχύτητα 0.4, αφού η απόλυτη διαφορά μεταξύ τρέχουσας συντεταγμένης και συντεταγμένης στόχου στον άξονα X είναι $|1 - 4| = 3 >= 0.3$, αλλιώς θα μετακινείται με ταχύτητα 0.1 .

Η παρακάτω εικόνα δείχνει το ρομπότ να ξεκινάει από τη θέση (0 , 0) με γωνία περιστροφής 0 μοίρες και να θέλει να καταλήξει στο στόχο, δηλαδή στη θέση (-4.40 , 0.62) και γωνία περιστροφής 180 μοίρες. Οπότε, το βήμα (1) είναι η περιστροφή του μέσω του τύπου περιστροφής **(1)** που περιγράψαμε νωρίτερα, το βήμα (2) είναι η μετακίνηση του προς το στόχο με μία απόσταση ασφαλείας και τέλος το βήμα (3) είναι πάλι περιστροφή προς τη τελική γωνία περιστροφής, καθώς και μία διόρθωση προσανατολισμού.



Εικόνα 1.29: Παράδειγμα κίνησης του Husky από μία αρχική θέση σε μία άλλη.

1.6 ΚΑΜΕΡΑ HUSKY

Στο κεφάλαιο αυτό θα μιλήσουμε λεπτομερώς για τη λειτουργία και τη τεράστια σημασία της κάμερας D435 που έχει τοποθετημένο το Husky στο εμπρόσθιο μέρος του, καθώς και για τη βιβλιοθήκη OpenCV.

1.6.1 *OpenCV*

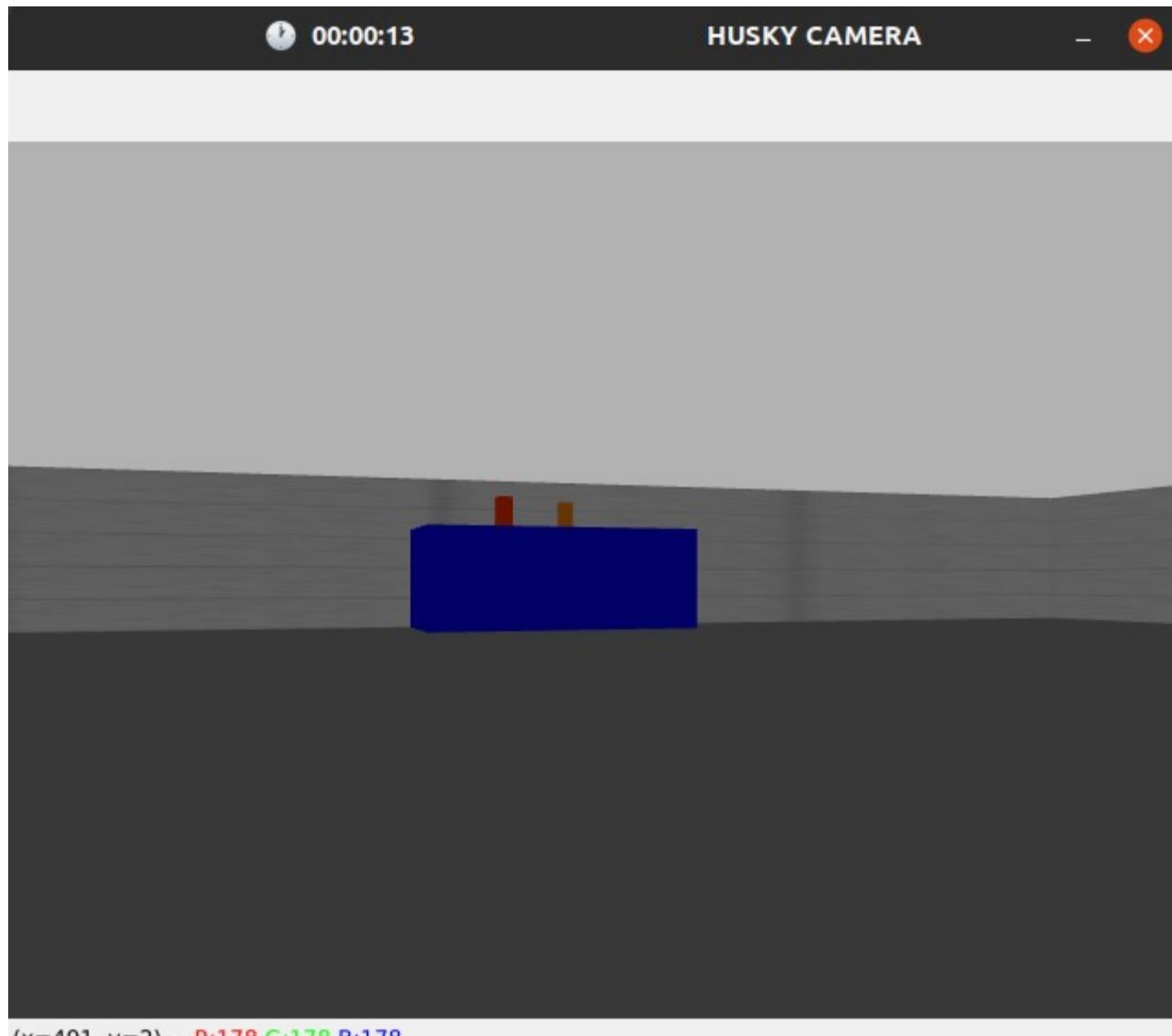


Η OpenCV είναι μία βιβλιοθήκη ανοιχτού κώδικα για υπολογιστική όραση. Σκοπός της είναι να παρέχει εργαλεία και βιβλιοθήκες λογισμικού για την επεξεργασία εικόνων και βίντεο. Οι κύριοι στόχοι της OpenCV περιλαμβάνουν την υλοποίηση αλγορίθμων υπολογιστικής όρασης, την ανάπτυξη εφαρμογών υπολογιστικής όρασης και την διευκόλυνση της έρευνας στον τομέα της υπολογιστικής όρασης.

Ορισμένες βασικές δυνατότητες περιλαμβάνουν η επεξεργασία εικόνων. Η OpenCV παρέχει πολλούς αλγόριθμους και συναρτήσεις για τη βελτίωση, επεξεργασία και φιλτράρισμα εικόνων. Για παράδειγμα, υπάρχει η δυνατότητα όπως αλλαγή μεγέθους, αντίθεσης, φίλτρα, και πολλά ακόμα. Επίσης, η OpenCV έχει την ικανότητα να ανιχνεύει αντικείμενα, παρέχοντας αλγορίθμους ανίχνευσης αντικειμένων που μπορούν να αναγνωρίσουν και να προσδιορίσουν αντικείμενα σε μια εικόνα ή ένα βίντεο. Άλλες δυνατότητες είναι η αναγνώριση προσώπων, ανίχνευση κίνησης, καθώς και στερεοσκοπική όραση, δηλαδή η δυνατότητα εκτίμησης βάθους ενός αντικειμένου ή αλλιώς το πόσο μακριά βρίσκεται το αντικείμενο αυτό, σε μία εικόνα ή βίντεο.

Γενικότερα, η OpenCV είναι ευρέως χρησιμοποιούμενη σε πολλούς τομείς, όπως η ρομποτική, τα αυτόνομα οχήματα, η ιατρική εικονική πραγματικότητα, η αναγνώριση χειρονομιών, καθώς και σε πολλές άλλες εφαρμογές που απαιτούν υπολογιστική όραση.

Παρακάτω φαίνεται ένα απλό παράδειγμα χρήσης της βιβλιοθήκης OpenCV μέσω της οποίας εμφανίζεται ένα παράθυρο γραφικού περιβάλλοντος, το οποίο αναδεικνύει τις εικόνες που λαμβάνει το Husky από το περιβάλλον του (Gazebo) σε πραγματικό χρόνο.



Εικόνα 1.30: Παράθυρο γραφικού περιβάλλοντος με OpenCV της κάμερας Husky.

1.6.2 ΛΕΙΤΟΥΡΓΙΑ ΚΑΜΕΡΑΣ ΜΕΣΩ OPENCV

1.6.2.1 ΧΡΗΣΙΜΟΤΗΤΑ ΚΑΜΕΡΑΣ ΚΑΙ OPENCV

Μόλις έρθει η στιγμή να ξεκινήσει το Husky να κάνει οποιαδήποτε εργασία, όπως περιστροφή ή μετακίνηση, τότε ενεργοποιείται η κάμερα του, σκοπός της οποίας είναι να κατευθύνει το όχημα προς τη σωστή κατεύθυνση και συγκεκριμένα όσο πιό κοντά στο κέντρο του τραπεζιού-στόχου. Είναι ιδιαίτερα σημαντική, διότι όπως έχουμε ξαναπεί με τις μετακινήσεις και περιστροφές του Husky στο χώρο μεταδίδονται σφάλματα, οπότε οι συντεταγμενες θέσης του αλλάζουν δραστικά. Για παράδειγμα, αν ο στόχος του ήταν να κατευθυνθεί στη θέση με συντεταγμένες (3 , 1) , λόγω αυτών των προαναφερθέντων σφαλμάτων, υπάρχει περίπτωση να αποκλίνει από αυτή τη θέση και να καταλήξει στη θέση με συντεταγμένες (3.3 , 1.6). Φυσικά, υπάρχει και ένα ανεκτό όριο στα σφάλματα αυτά, διότι τίποτα πουθενά δεν είναι τέλειο, οπότε μία απόκλιση της τάξεως των 20-30 εκατοστών καθίσταται ανεκτή.

Εδώ λοιπόν, έρχεται η βιβλιοθήκη OpenCV και δίνει τη λύση στα προβλήματα αυτά, επειδή το Husky από μόνο του δεν έχει τη δυνατότητα μέσω του topic "/odometry/filtered" να διορθώσει περαιτέρω τα σφάλματα που γίνονται στις συντεταγμένες.

Η εικόνα 1.22 επιβεβαιώνει την εσφαλμένη θέση του ρομπότ σε σχέση με το τραπέζι-στόχο, καθώς αυτό βρίσκεται μακριά από το κέντρο του.

Αντίθετα, μετά από κάποια επεξεργασία μέσω OpenCV και ενός αλγορίθμου, τα οποία θα εξηγηθούν αναλυτικά πιό κάτω, το ρομπότ θα καταλήξει σε θέση που αντιπροσωπεύει η εικόνα 1.23 .

1.6.2.2 ΑΛΓΟΡΙΘΜΟΣ ΚΑΜΕΡΑΣ HUSKY

Η υποενότητα αυτή αναπτύσσει τον αλγόριθμο, ο οποίος σχεδιάστηκε σε συνδυασμό με την βιβλιοθήκη OpenCV, με σκοπό να διασφαλίσει καλύτερα αποτελέσματα για το Husky, ουσιαστικά να φέρει το ρομποτικό όχημα όσο πιό κοντά στο κέντρο του τραπεζιού-στόχου γίνεται (βλέπουμε συγκριτικά τις εικόνες 1.22 και 1.23).

Καταρχάς, πρέπει να δημιουργήσουμε το παράθυρο του γραφικού περιβάλλοντος με τη βοήθεια της OpenCV. Οι διαστάσεις του είναι (640 , 480), δηλαδή 640 οριζόντια pixels για κάθε γραμμή και 480 κάθετα pixels για κάθε στήλη. Συνολικά $640*480 = 307.200$ pixels για την εικόνα.

Επιπρόσθετα, θα χρειαστεί να μετασχηματιστεί η εικόνα από τον χρωματικό χώρο BGR σε HSV. Ας εξηγήσουμε την διαφορά των δύο και γιατί επιλέγουμε τον χώρο HSV.

Το BGR είναι ένας χρωματικός χώρος που αναπαριστά τα χρώματα ως συνδυασμό τριών συνιστωσών: μπλε (Blue), πράσινο (Green) και κόκκινο (Red). Το HSV αναπαριστά τα χρώματα ως τρεις συνιστώσες: απόχρωση (Hue), κορεσμό (Saturation) και τιμή-φωτεινότητα (Value). Ο HSV χρωματικός χώρος είναι πιο φυσικός στην αντιπροσώπευση της αντίληψης του ανθρώπινου ματιού για τα χρώματα και αυτός είναι ο βασικότερος λόγος που χρησιμοποιείται συχνά από εφαρμογές που απαιτούν πιο εύκολη ρύθμιση των χαρακτηριστικών του χρώματος.

Να συμπληρώσουμε εδώ ότι πάνω αριστερά στο παράθυρο εκτυπώνεται σε πραγματικό χρόνο ο αριθμός των αντικειμένων που έχει τοποθετήσει μέχρι εκείνη τη στιγμή ο βραχίονας στα τραπέζια με τις υποδοχές, ο χρόνος λειτουργίας της εφαρμογής, καθώς και το όνομα της κάμερας που είναι εν λειτουργίᾳ την εκάστοτε στιγμή. Αυτές οι λειτουργίες/μηνύματα εμφανίζονται και για τις τρεις κάμερες των δύο ρομπότ. Ανάλογα ποιά κάμερα ενεργοποείται κάθε φορά, ανανεώνεται και εμφανίζεται στο παράθυρο η σωστή εικόνα/frame της αντίστοιχης κάμερας με τα εκτυπωμένα μηνύματα, όπως ο αριθμός των αντικειμένων, ο χρόνος λειτουργίας της εφαρμογής, καθώς και το όνομα της κάμερας.

Όπως παρατηρούμε από την εικόνα 1.22, όλα τα τραπέζια έχουν ορθογώνιο σχήμα χρώματος μπλε, όπως αναπαριστώνται στο frame της κάμερας σε 2D. Τα τραπέζια έχουν σκόπιμα μπλε χρώμα και ορθογώνιο σχήμα για να μπορέσει να γίνει εύκολα κατανοητό από τον αλγόριθμο ότι πρόκειται για ένα τραπέζι. Τα τραπέζια μπορούν να έχουν διαφορετικά μεγέθη (μήκος-ύψος-πλάτος). Ο αλγόριθμος μέσω της OpenCV μπορεί να αναγνωρίσει το σχήμα χωρίς δυσκολία.

Επόμενο μέλημα είναι πως θα καταλάβουμε το σχήμα και το χρώμα, ώστε να πούμε με σιγουριά ότι αυτό που βλέπουμε πρόκειται για τραπέζι και όχι οτιδήποτε άλλο.

Ας ξεκινήσουμε αρχικά με το χρώμα του τραπεζιού, το μπλε. Ο χρωματικός χώρος ο οποίος χρησιμοποιείται είναι ο HSV. Συγκεκριμένα, χρησιμοποιούνται η απόχρωση (Hue), ο κορεσμός (Saturation) και η τιμή-φωτεινότητα (Value) ως συνιστώσες για τα χρώματα. Ένα χρώμα, όπως το μπλε στη περίπτωση μας, έχει διάφορες αποχρώσεις, κορεσμούς και φωτεινότητες, άρα πρέπει να λάβουμε ένα όριο τιμών, εντός του οποίου θα δεχόμαστε αυτές τις τιμές ως μπλε. Έχει διαπιστωθεί ότι το μπλε χρώμα του τραπεζιού μπορεί να πάρει τις εξής τιμές για τις τρεις παραπάνω συνιστώσες:

- Τιμές εντός του κλειστού διαστήματος [110 , 130] για την απόχρωση.
- Τιμές εντός του κλειστού διαστήματος [100 , 255] για τον κορεσμό.
- Τιμές εντός του κλειστού διαστήματος [100 , 255] για την φωτεινότητα.

Γενικότερα για οποιοδήποτε χρώμα, η ελάχιστη τιμή που μπορεί να πάρει ο κορεσμός και η φωτεινότητα είναι η τιμή 0 και η μέγιστη είναι η τιμή 255, ενώ για την απόχρωση καθίσταται το όριο [0 , 360]. Υπενθυμίζουμε ότι χρησιμοποιείται 8-bit αναπαράσταση για τις τρεις συνιστώσες.

Συνεχίζουμε στην κατανόηση του σχήματος του τραπεζιού παρακάτω.

Χρησιμοποιώντας μία ειδική μέθοδο της OpenCV, δημιουργούμε μία μάσκα που παίρνει τις τιμές 0 ή 1 ανάλογα με το αν το αντίστοιχο pixel βρίσκεται εντός του εύρους τιμών

που ορίστηκε παραπάνω για το μπλε χρώμα. Αυτή η διαδικασία χρησιμοποιείται για τον προσδιορισμό της περιοχής που αντιπροσωπεύει το τραπέζι μπλε χρώματος. Ουσιαστικά, με αυτόν τον τρόπο απομονώνουμε το αντικείμενο επιθυμητού χρώματος που θέλουμε.

Στη συνέχεια, χρησιμοποιείται ο αλγόριθμος Canny για την ανίχνευση ακμών στην εικόνα που έχουμε εφαρμόσει τη μάσκα, η οποία περιλαμβάνει περιοχές της εικόνας που αντιστοιχούν στο επιθυμητό εύρος χρωμάτων, στη περίπτωση μας, το μπλε.

Ο αλγόριθμος Canny είναι ένας αλγόριθμος ανίχνευσης ακμών που είναι ευαίσθητος σε αλλαγές φωτεινότητας σε μία εικόνα.

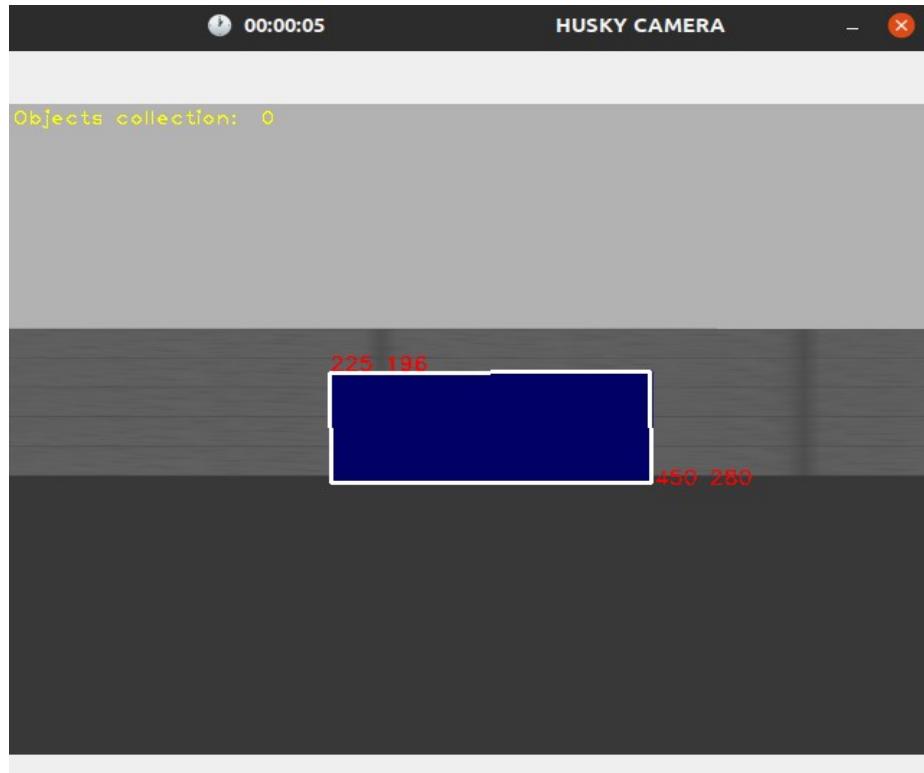
Ακόμα, περνιούνται δύο παράμετροι στη συνάρτηση "Canny". Οι παράμετροι 50 και 150 αφορούν τα κατώφλια για τον αλγόριθμο. Ουσιαστικά, λειτουργούν ως κατώφλια για την ανίχνευση ακμών. Όποιες τιμές είναι χαμηλότερες από το 50 θεωρούνται "αδύναμες" ακμές και ό,τι είναι υψηλότερο από το 150 θεωρείται "ισχυρή" ακμή. Έτσι, γίνεται ο εντοπισμός των ακμών του τραπεζιού.

Επόμενη διαδικασία είναι ο εντοπισμός των συνολικών συνόρων της εικόνας επιστρέφοντας ουσιαστικά τις συντεταγμένες των συνόρων αυτών.

Η "Canny" και η τελευταία διαδικασία χρησιμοποιούνται μαζί για την ανίχνευση αντικειμένων: η "Canny" χρησιμοποιείται για την ανίχνευση ακμών, ενώ η άλλη εφαρμόζεται πάνω στη μάσκα των ακμών για την εύρεση των πραγματικών συνόρων των αντικειμένων της εικόνας.

Στη συνέχεια, μετασχηματίζονται τα σύνορα του τραπεζιού σε πιό απλά πολύγωνα (ουσιαστικά ορθωγώνια), εάν αυτά είναι πολύπλοκα, έτσι ώστε να γίνει πιό κατανοητό το σχήμα του αντικειμένου. Συνοπτικά, απλοποιείται η αναπαράσταση των τραπεζιών στην εικόνα, κάνοντάς τα πιό απλά και ευανάγνωστα.

Έχουμε πλέον τις συντεταγμένες των τεσσάρων ακμών του τραπεζιού. Στην ουσία όμως θα μας χρειαστούν μόνο δύο ακμές. Οι συντεταγμένες (x , y) της πάνω αριστερής και κάτω δεξιάς ακμής, όπως φαίνεται στην εικόνα 1.62 παρακάτω.

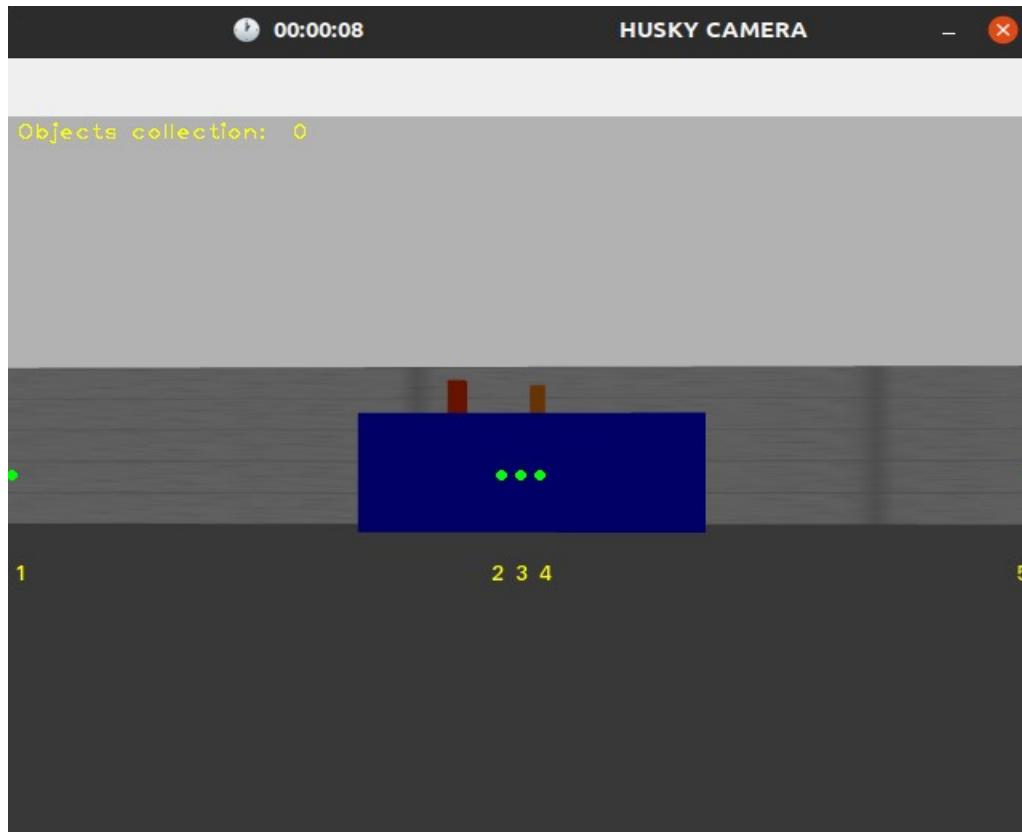


Εικόνα 1.31: Σχηματισμός τετραγώνου γύρω από το μπλε τραπέζι με τις συντεταγμένες στις δύο ακμές.

Μέχρι στιγμής έχουν καθοριστεί βασικές λειτουργίες, με λίγα λόγια, ο εντοπισμός του αντικειμένου. Σκοπός όμως της κάμερας είναι να καθοδηγήσει το όχημα όσο πιό κοντά στο κέντρο γίνεται, εάν αυτό έχει αποκλίνει από το κέντρο του τραπεζιού. Πως θα γίνει αυτό; Θα αναπτύξουμε λεπτομερώς και με απλά λόγια έναν αλγόριθμο ο οποίος λύνει το παραπάνω πρόβλημα με μεγάλη πιθανότητα επιτυχίας, ύστερα από πάρα πολλές δοκιμές.

Θυμίζουμε πως στόχος μας είναι να φέρουμε το όχημα παρατηρώντας το από την εικόνα 1.22 σε μία θέση όπως η εικόνα 1.23 . Η απόσταση ασφαλείας του Husky από το τραπέζι ακριβώς πριν ξεκινήσει ο αλγόριθμος να “καθοδηγεί” το όχημα στη πορεία του προς το κέντρο του τραπεζιού είναι, όπως έχουμε αναφέρει, στα 3.5 μέτρα.

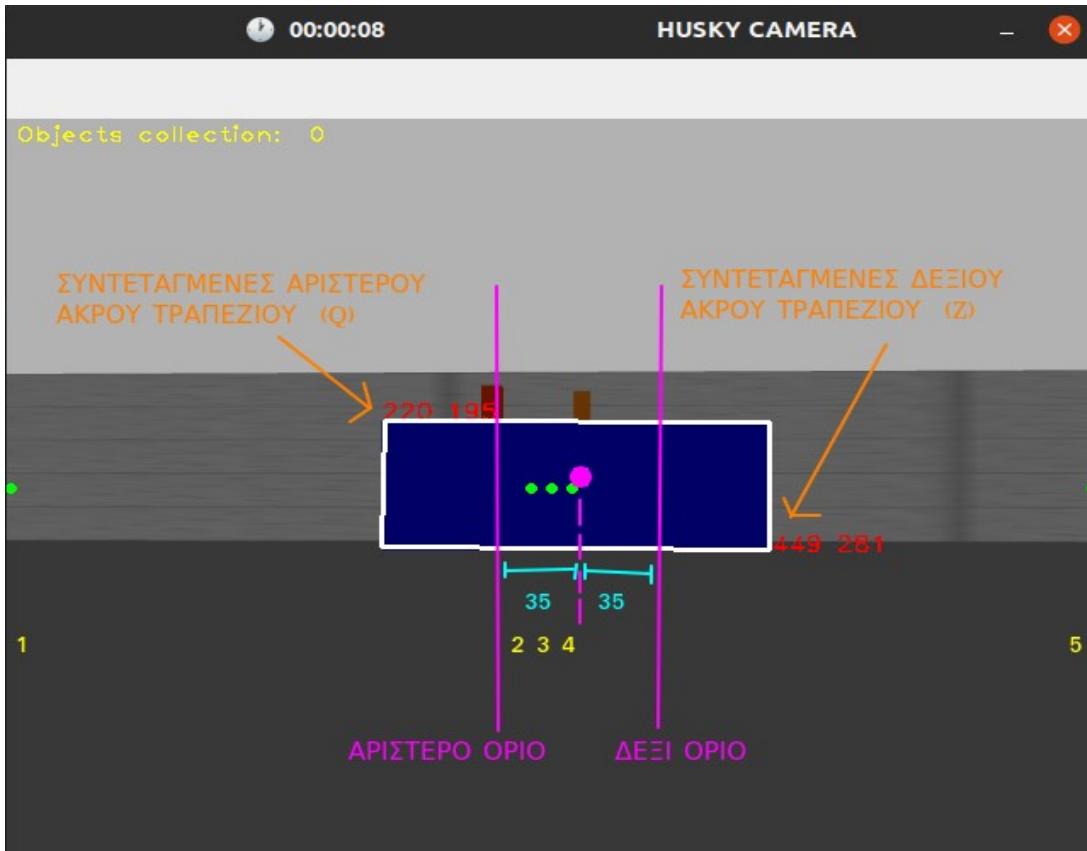
Στο σημείο αυτό πρέπει να αναφέρουμε πως έχουν σχεδιαστεί 5 πράσινα κυκλικά σημεία σε καθορισμένα σημεία του frame μας, τα οποία έχουν ιδιαίτερη σημασία.



Εικόνα 1.32: Τελική αναπαράσταση παραθύρου με τα 5 πράσινα κυκλικά σημεία.

Ο κύριος στόχος του αλγορίθμου είναι να φέρει τη κουκκίδα №3 όσο πιό κοντά στο κέντρο του τραπεζιού γίνεται. Ουσιαστικά, η κουκκίδα αυτή αποτελεί το κεντρικό σημείο του frame της κάμερας, ενώ ταυτόχρονα η κάμερα βρίσκεται στο μπροστινό κεντρικό σημείο του Husky. Με αυτόν τον τρόπο, καταφέρνουμε να φέρουμε το ρομπότ όσο πιό κοντά στο κέντρο του μπλε τραπεζιού.

Κάθε σημείο του frame αποτελείται από συντεταγμένες της μορφής (x , y). Θα χρειαστούμε τις συντεταγμένες των 5 πράσινων κουκκίδων, καθώς και τις συντεταγμένες των δύο άκρων του τραπεζιού. Αυτές οι συντεταγμένες ανανεώνονται συνεχώς, αφού το ρομπότ μετατοπίζεται ή περιστρέφεται στο 3D χώρο. Η κουκκίδα №3 θεωρείται ότι βρίσκεται στο κέντρο του τραπεζιού όταν βρίσκεται μεταξύ του δεξιά (Z) και αριστερού (Q) ορίου του τραπεζιού που παρατίθενται παρακάτω.



Εικόνα 1.33: Τελική αναπαράσταση παραθύρου με τα 5 πράσινα κυκλικά σημεία.

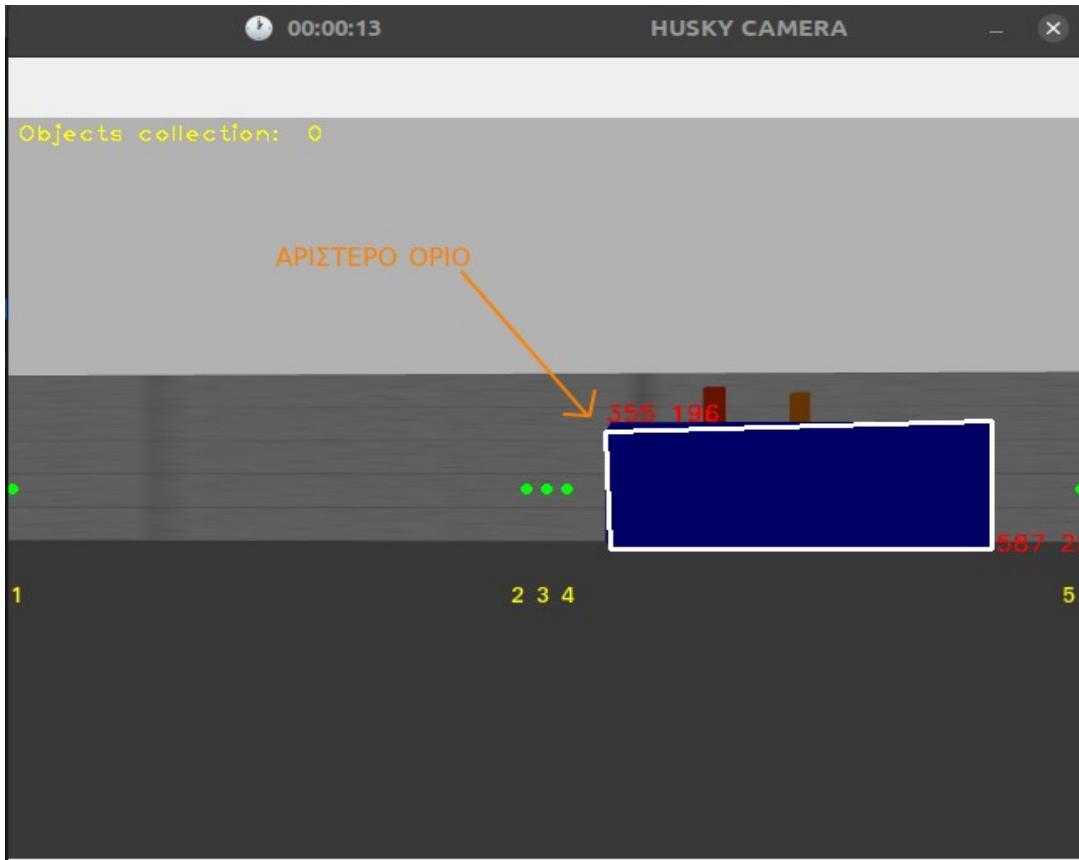
$$\text{ΑΡΙΣΤΕΡΟ ΟΡΙΟ} = \left(\frac{Q+Z}{2} \right) - 35, \quad , \quad \Delta\text{ΕΞΙ ΟΡΙΟ} = \left(\frac{Q+Z}{2} \right) + 35$$

Ουσιαστικά, για κάθε όριο λαμβάνουμε το μέσο όρο των pixels των άκρων του τραπεζιού και προσθέτουμε ή αφαιρούμε αντίστοιχα την σταθερά 35 (σε pixels).

Ο αλγόριθμος χωρίζεται σε τέσσερις κατηγορίες. Ας τις εξηγήσουμε παρακάτω.

● ΠΕΡΙΠΤΩΣΗ 1

Η κουκκίδα No3 βρίσκεται πιό αριστερά του αριστερού ορίου του τραπεζιού.

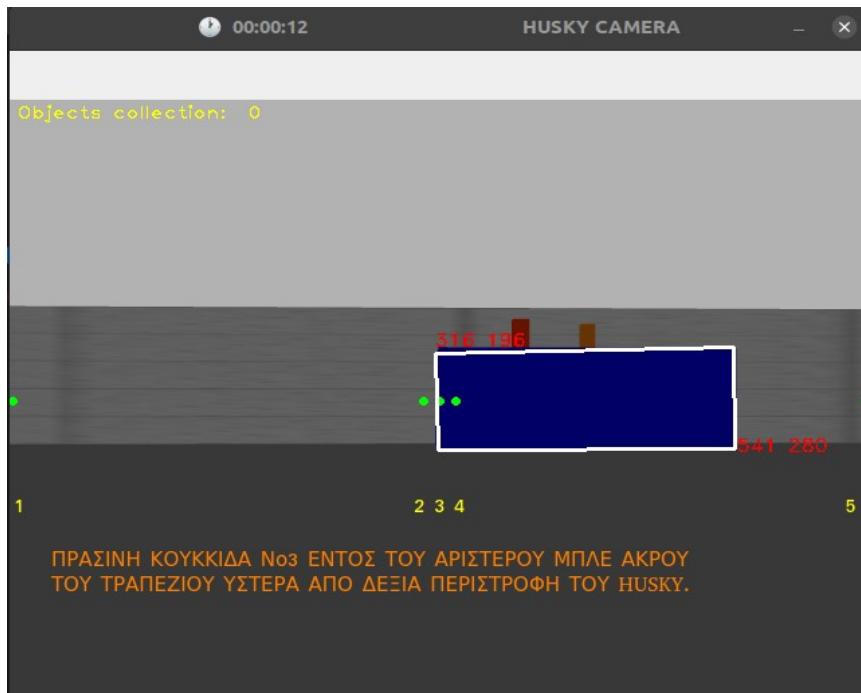


Εικόνα 1.34: Η κουκκίδα No3 έξω από την αριστερή άκρη του μπλε τραπεζιού.

Η παραπάνω εικόνα αποδεικνύει ότι τα σφάλματα που έχουν προκύψει από τις προηγούμενες κινήσεις του Husky έχουν φέρει το κέντρο του ρομπότ σε λάθος θέση και συγκεκριμένα αριστερότερα από την αριστερή άκρη του τραπεζιού. Θυμίζουμε στο σημείο αυτό ότι μέσω εντολής της OpenCV, μπορούμε να λάβουμε το χρώμα του pixel ή σημείου του frame που επιθυμούμε.

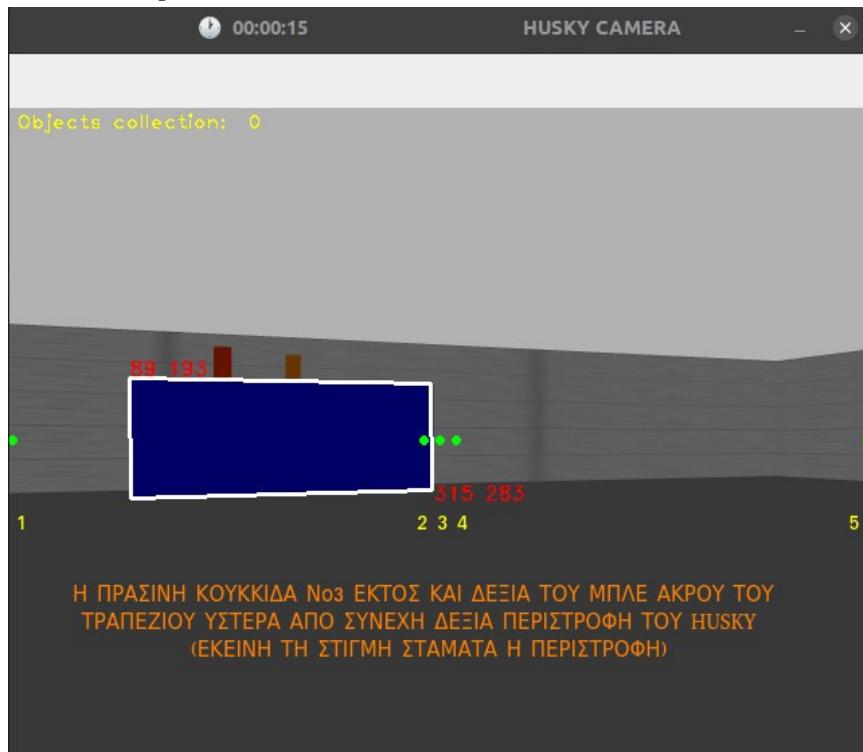
Για να φέρουμε το ρομπότ όσο πιό κοντά στο κέντρο του τραπεζιού, ακολουθούμε την εξής διαδικασία:

Περιστροφή του Husky προς τα δεξιά με ταχύτητα 0.06 rad/sec. μέχρι να βρεί μπλε χρώμα η κουκκίδα No3. Ουσιαστικά, ελέγχουμε αν η απόχρωση (Hue) της συντεταγμένης κουκκίδας No3 έχει μπλε χρώμα (κάθε άλλη χρονική στιγμή έχει διαφορετικό χρώμα, για παράδειγμα γκρι χρώμα, λόγω τοίχου). [εικόνα 1.35]



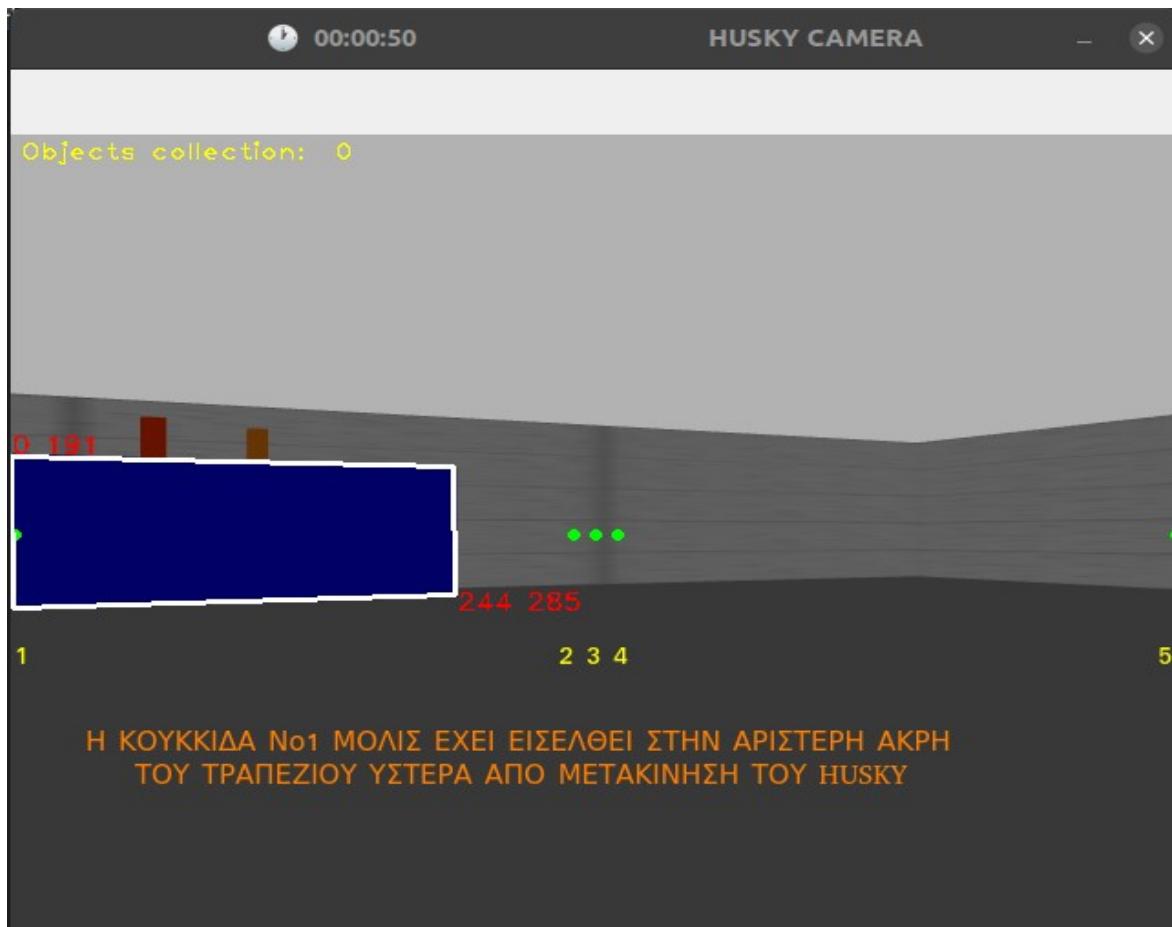
Εικόνα 1.35: Η κουκκίδα Νο3 εντός της αριστερής άκρης του μπλε τραπεζιού.

Έπειτα, συνεχίζεται η δεξιά περιστροφή με ταχύτητα 0.06 rad/sec. μέχρι την δεξιά άκρη του τραπεζιού, δηλαδή γίνεται έλεγχος αν η απόχρωση (Hue) της συντεταγμένης κουκκίδας Νο3 δεν έχει πλέον μπλε χρώμα, αφού έχει πάει στα όρια της δεξιάς άκρης του τραπεζιού. [εικόνα 1.36]



Εικόνα 1.36: Η κουκκίδα Νο3 έξω της δεξιάς άκρης του μπλε τραπεζιού.

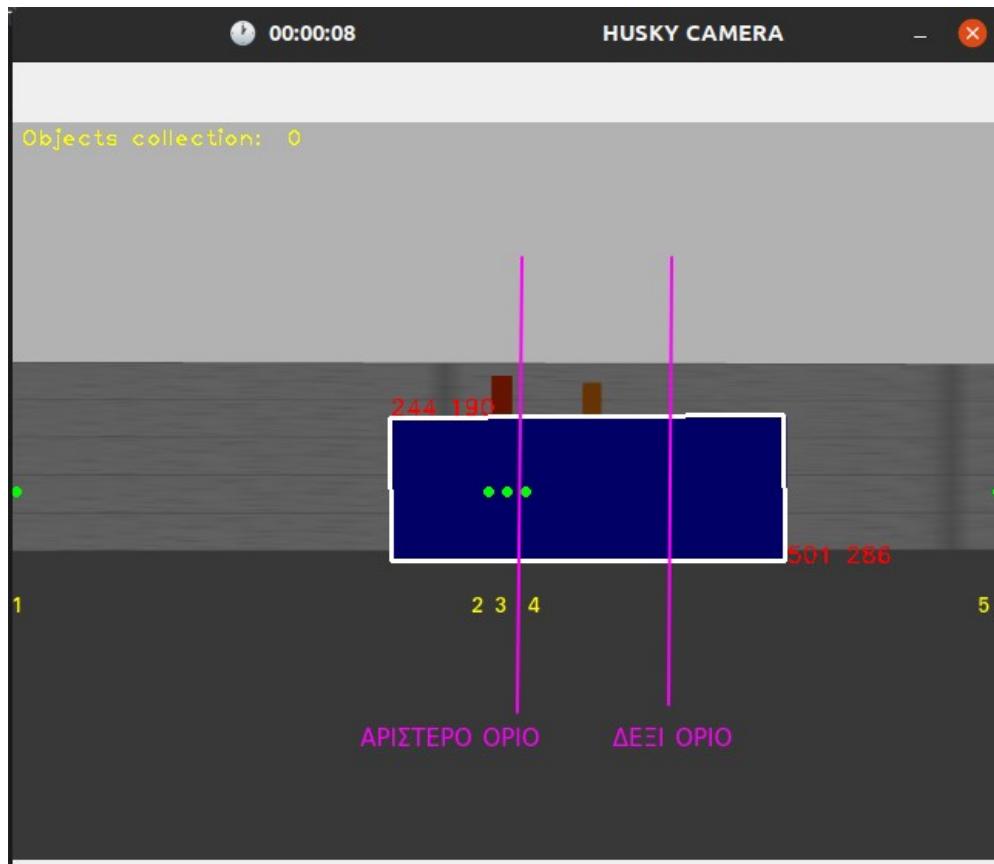
Στη συνέχεια, έχοντας αυτή την εικόνα, το ρομπότ μετακινείται μονάχα ευθεία με ταχύτητα 0.06 m/sec έως ότου η πράσινη κουκκίδα No1 εισέλθει εντός του αριστερού άκρου του τραπεζιού, δηλαδή μέχρι η απόχρωση (Hue) της συντεταγμένης κουκκίδας No1 να έχει μπλε χρώμα. [εικόνα 1.37]



Εικόνα 1.37: Η κουκκίδα No1 εντός-αριστερής άκρης του μπλε τραπεζιού.

Μετά από αυτή τη διαδικασία, σειρά έχει η διόρθωση του προσανατολισμού, δηλαδή φέρνουμε το όχημα σε κάθετη θέση με το τραπέζι, με σκοπό να ελέγξουμε πάλι αν η κουκκίδα No3 βρίσκεται στο κέντρο του τραπεζιού. Οπότε τώρα υπάρχουν δύο περιπτώσεις για τη κουκκίδα No3. Να προσπέσει πάλι εντός του μπλε τραπεζιού ή εκτός αυτού.

- Αναλύουμε την περίπτωση να προσπέσει η κουκκίδα No3 εντός του τραπεζιού.



Εικόνα 1.38: Η κουκκίδα No3 εντός τραπεζιού & αριστερά του κεντρικού ορίου τραπεζιού.

Στη περίπτωση αυτή της εικόνας 1.38, ακολουθούμε την εξής διαδικασία:

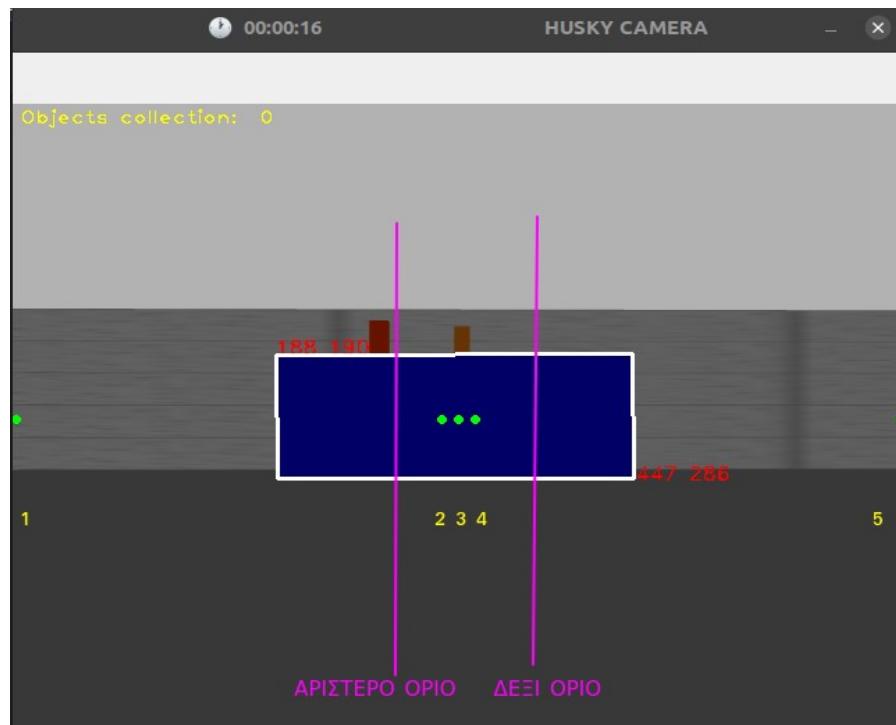
Περιστροφή του Husky προς τα δεξιά με ταχύτητα 0.06 rad/sec. μέχρι να μην βρεί μπλε χρώμα η κουκκίδα No3. Ουσιαστικά, ελέγχουμε αν η απόχρωση (Hue) της συντεταγμένης κουκκίδας No3 δεν έχει πλέον μπλε χρώμα. [εικόνα 1.36]

Στη συνέχεια, το ρομπότ μετακινείται ευθεία με ταχύτητα 0.06 m/sec έως ότου η πράσινη κουκκίδα No2 εξέλθει του δεξιού άκρου του τραπεζιού, δηλαδή μέχρι η απόχρωση (Hue) της συντεταγμένης κουκκίδας No2 να μην έχει πλέον μπλε χρώμα ή μέχρι το όχημα να πλησιάσει αρκετά κοντά στο τραπέζι (θα αναφέρουμε σε επόμενη υποενότητα περισσότερα για την απόσταση αυτή), όπως μπορούμε να δούμε από την εικόνα 1.39 .



Εικόνα 1.39: Η κουκκίδα №2 εκτός του δεξιού άκρου του τραπεζιού.

Έπειτα, διορθώνεται όπως και πριν ο προσανατολισμός του οχήματος, δηλαδή το όχημα έρχεται σε κάθετη θέση με το τραπέζι και έτσι τώρα το ρομπότ βρίσκεται εντός ορίων με μεγάλη πιθανότητα επιτυχίας, όπως φαίνεται στην εικόνα 1.40 .



Εικόνα 1.40: Η κουκκίδα №3 εντός κεντρικού ορίου του τραπεζιού.

- *Αναλύουμε την περίπτωση να προσπέσει η κουκκίδα No3 εκτός του τραπεζιού.*

Όπως στην εικόνα 1.34, ακολουθούμε και και τα εξής βήματα:

Περιστροφή του Husky προς τα δεξιά με ταχύτητα 0.06 rad/sec. μέχρι να βρεί μπλε χρώμα η κουκκίδα No3. Ουσιαστικά, ελέγχουμε αν η απόχρωση (Hue) της συντεταγμένης κουκκίδας No3 έχει μπλε χρώμα (κάθε άλλη χρονική στιγμή έχει διαφορετικό χρώμα, για παράδειγμα γκρι χρώμα, λόγω τοίχου). [εικόνα 1.35]

Έπειτα, συνεχίζεται η δεξιά περιστροφή με ταχύτητα 0.06 rad/sec. μέχρι την δεξιά άκρη του τραπεζιού, δηλαδή γίνεται έλεγχος αν η απόχρωση (Hue) της συντεταγμένης κουκκίδας No3 δεν έχει πλέον μπλε χρώμα, αφού έχει πάει στα όρια της δεξιάς άκρης του τραπεζιού. [εικόνα 1.36]

Μετά, το ρομπότ μετακινείται ευθεία με ταχύτητα 0.06 m/sec έως ότου η πράσινη κουκκίδα No2 εξέλθει του δεξιού άκρου του τραπεζιού, δηλαδή μέχρι η απόχρωση (Hue) της συντεταγμένης κουκκίδας No2 να μην έχει πλέον μπλε χρώμα ή μέχρι το όχημα να πλησιάσει αρκετά κοντά στο τραπέζι. [εικόνα 1.39]

Τέλος, διορθώνεται ο προσανατολισμός του οχήματος, δηλαδή το όχημα έρχεται σε κάθετη θέση με το τραπέζι και έτσι το ρομπότ τώρα βρίσκεται εντός ορίων του τραπεζιού με μεγάλη πιθανότητα επιτυχίας, όπως φαίνεται στην εικόνα 1.40 .

● **ΠΕΡΙΠΤΩΣΗ 2**

Η κουκκίδα No3 βρίσκεται εντός του μπλε τραπεζιού, αλλά αριστερότερα του κεντρικού ορίου του τραπεζιού, όπως φαίνεται από την εικόνα 1.38 .

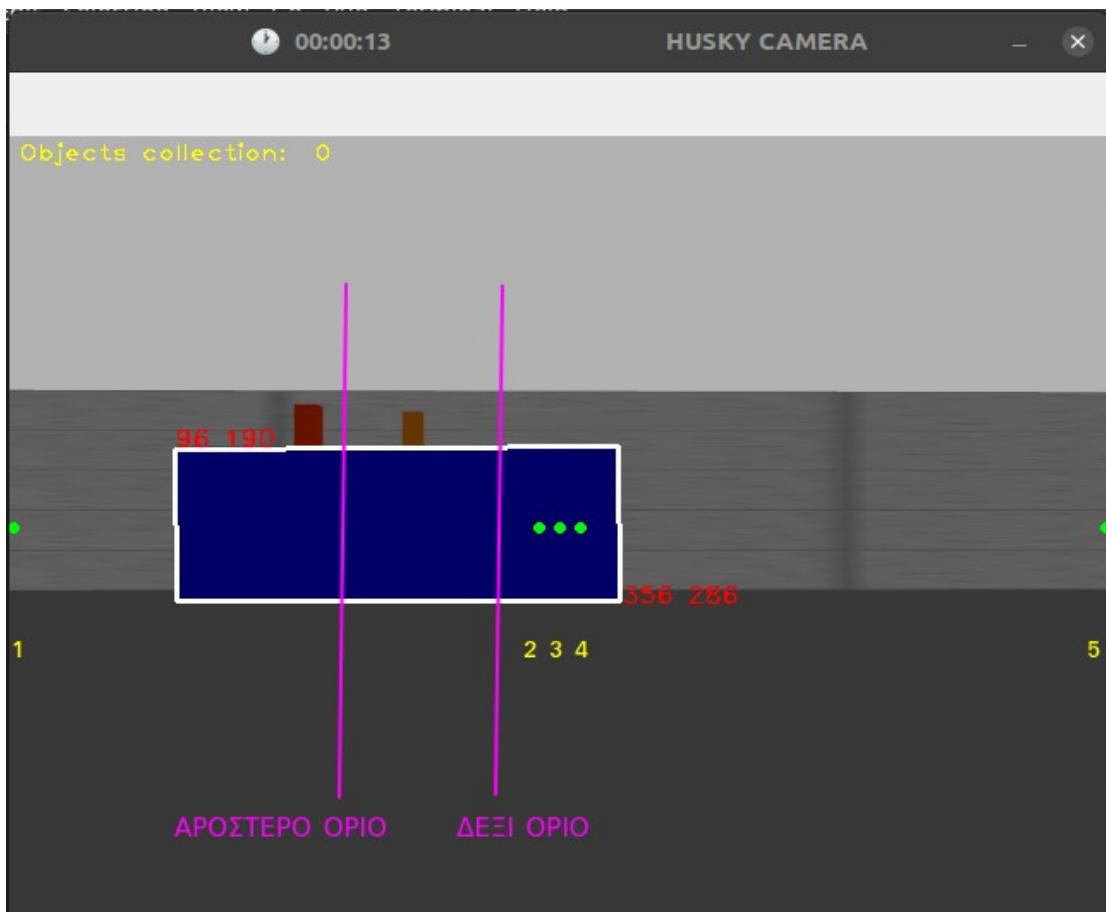
Σε αυτή τη περίπτωση, η κουκκίδα No3 βρίσκεται ήδη εντός του τραπεζιού, δηλαδή η απόχρωση (Hue) της συντεταγμένης κουκκίδας No3 έχει ήδη μπλε χρώμα. Ο αλγόριθμος εδώ περιστρέφει το όχημα προς τα δεξιά με ταχύτητα 0.06 rad/sec. έως ότου η κουκκίδα No3 βγει εκτός του μπλε χρώματος του τραπεζιού, δηλαδή να πάει μέχρι το δεξί άκρο του, όπως φαίνεται στην εικόνα 1.36 προηγουμένως.

Έπειτα, το ρομπότ μετακινείται ευθεία με ταχύτητα 0.06 m/sec έως ότου η πράσινη κουκκίδα No1 εισέλθει εντός του αριστερού άκρου του τραπεζιού, δηλαδή μέχρι η απόχρωση (Hue) της συντεταγμένης κουκκίδας No1 να έχει μπλε χρώμα ή μέχρι το όχημα να πλησιάσει αρκετά κοντά στο τραπέζι. [εικόνα 1.37]

Τέλος, διορθώνεται ο προσανατολισμός του οχήματος, δηλαδή το όχημα έρχεται σε κάθετη θέση με το τραπέζι και έτσι το ρομπότ τώρα βρίσκεται εντός ορίων του τραπεζιού με μεγάλη πιθανότητα επιτυχίας, όπως φαίνεται στην εικόνα 1.40 .

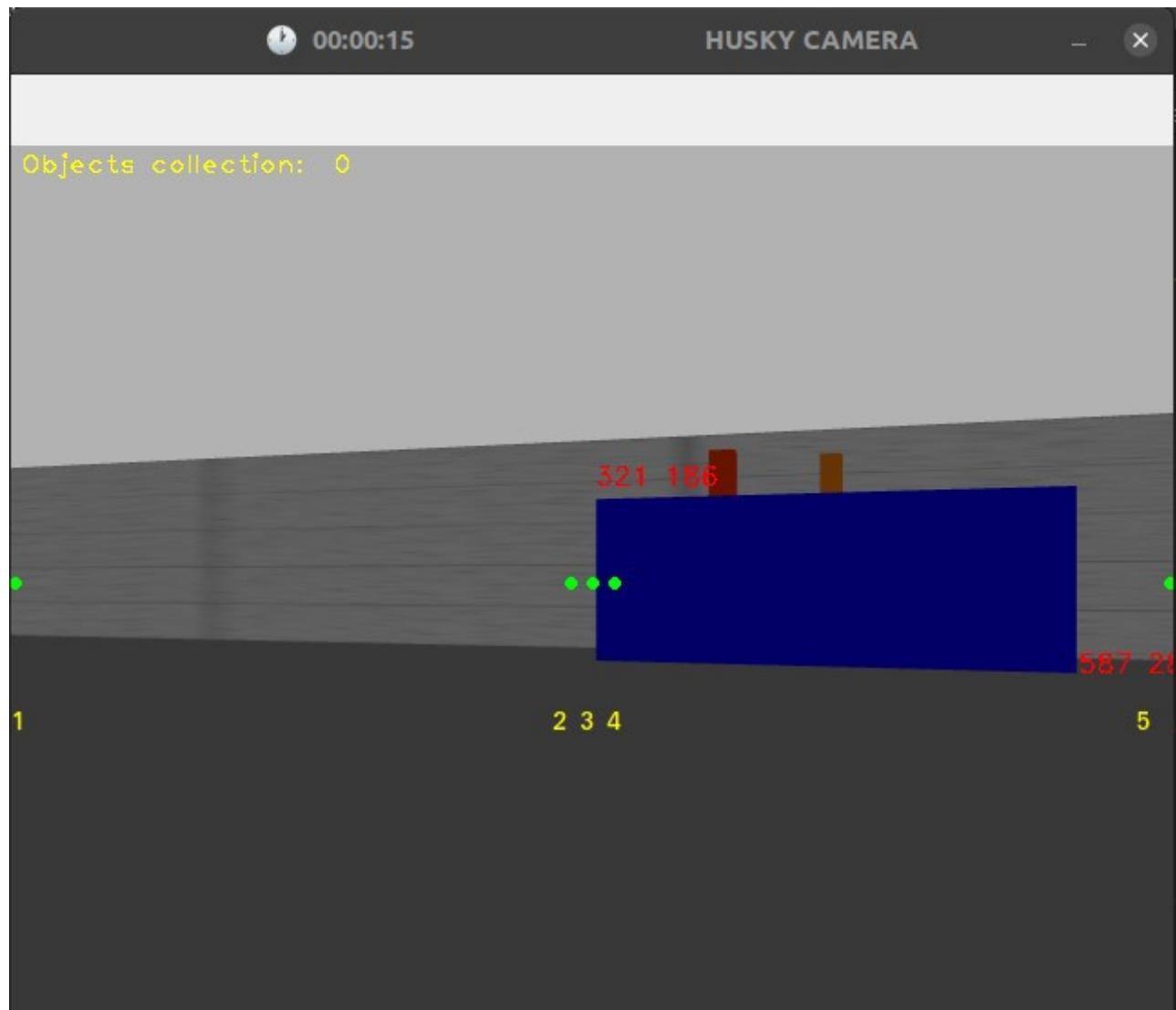
● **ΠΕΡΙΠΤΩΣΗ 3**

Η κουκκίδα No3 βρίσκεται εντός του μπλε τραπεζιού, αλλά δεξιότερα του κεντρικού ορίου του τραπεζιού.



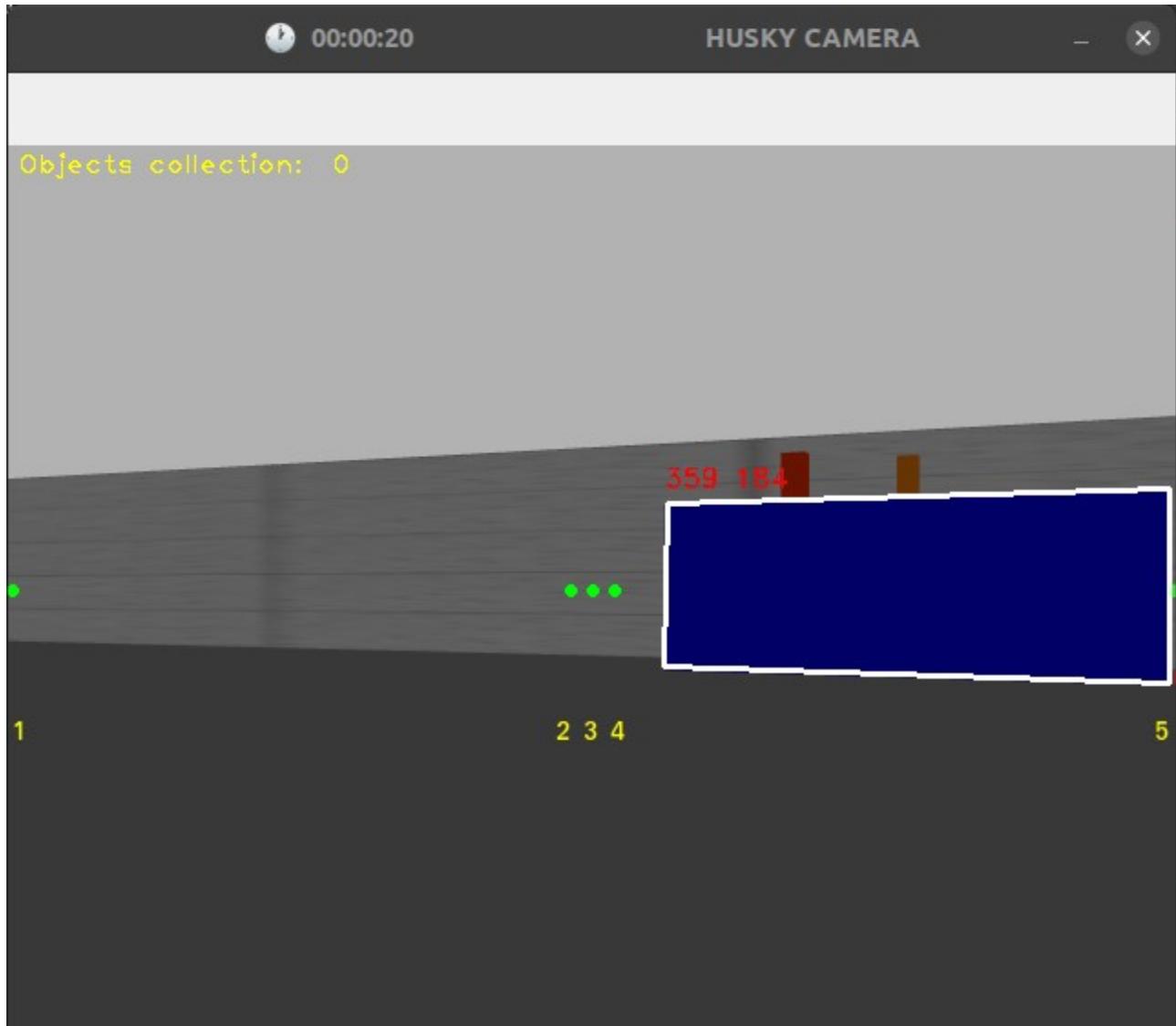
Εικόνα 1.41: Η κουκκίδα No3 εντός τραπεζιού & δεξιά του κεντρικού ορίου τραπεζιού.

Στη περίπτωση αυτή, η κουκκίδα №3 βρίσκεται ήδη εντός του τραπεζιού, δηλαδή η απόχρωση (Hue) της συντεταγμένης κουκκίδας №3 έχει ήδη μπλε χρώμα. Ο αλγόριθμος εδώ περιστρέφει το όχημα προς τα αριστερά με ταχύτητα 0.06 rad/sec. έως ότου η κουκκίδα №3 βγει εκτός του μπλε χρώματος του τραπεζιού, δηλαδή να πάει μέχρι το αριστερό άκρο του, όπως φαίνεται στην εικόνα 1.42 παρακάτω.



Εικόνα 1.42: Η κουκκίδα №3 εκτός του αριστερού άκρου του μπλε τραπεζιού.

Έπειτα, το ρομπότ μετακινείται ευθεία με ταχύτητα 0.06 m/sec έως ότου η πράσινη κουκκίδα No5 εισέλθει εντός του δεξιού άκρου του τραπεζιού, δηλαδή μέχρι η απόχρωση (Hue) της συντεταγμένης κουκκίδας No5 να έχει μπλε χρώμα ή μέχρι το όχημα να πλησιάσει αρκετά κοντά στο τραπέζι. [εικόνα 1.43]

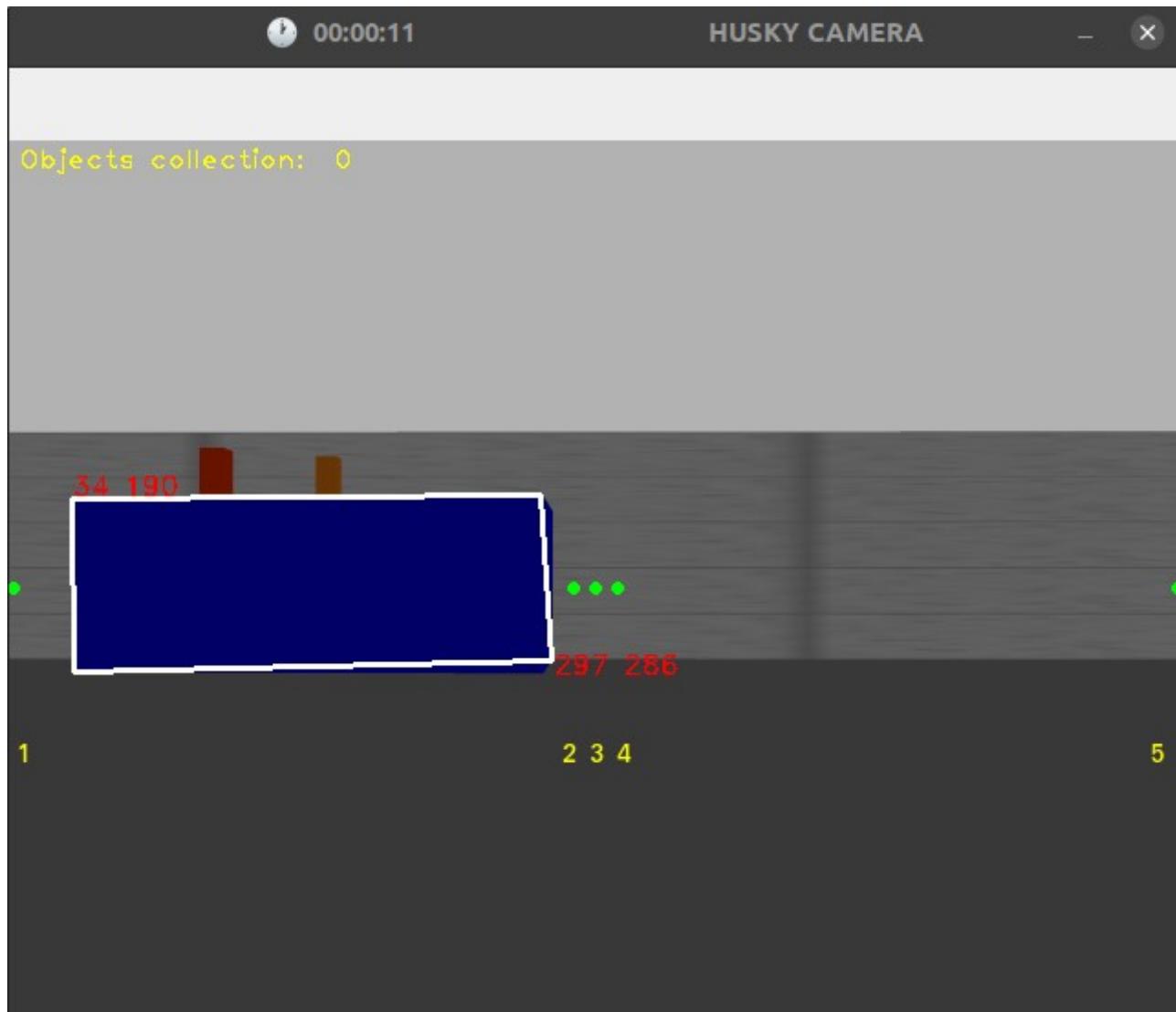


Εικόνα 1.43: Η κουκκίδα No5 εντός του δεξιού άκρου του μπλε τραπεζιού.

Τέλος, διορθώνεται πάλι ο προσανατολισμός του οχήματος, δηλαδή το όχημα έρχεται σε κάθετη θέση με το τραπέζι και έτσι το ρομπότ τώρα βρίσκεται εντός ορίων του τραπεζιού με μεγάλη πιθανότητα επιτυχίας, όπως φαίνεται στην εικόνα 1.40 .

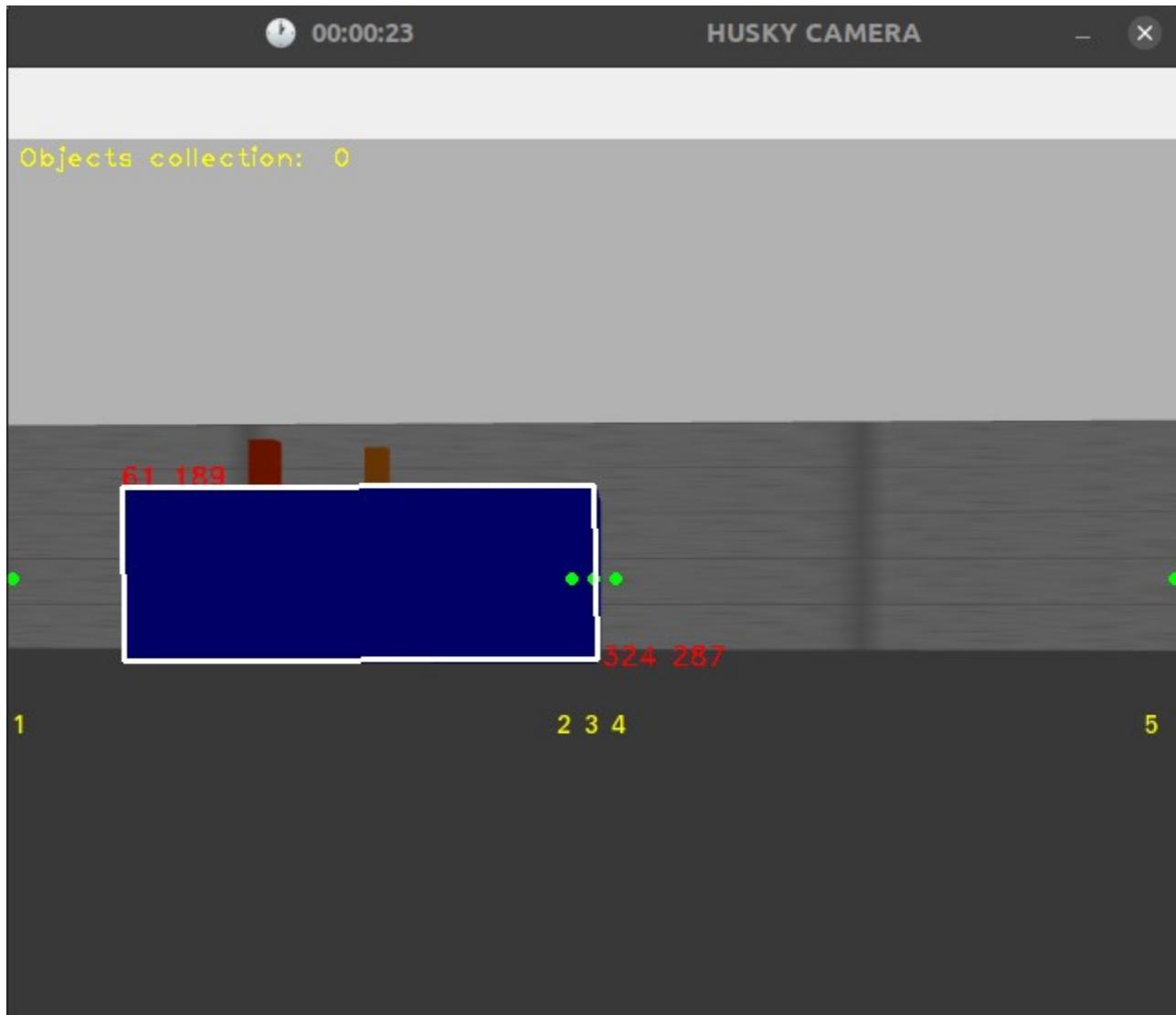
● **ΠΕΡΙΠΤΩΣΗ 4**

Η κουκκίδα Νο3 βρίσκεται πιό δεξιά του δεξιού άκρου του τραπεζιού.



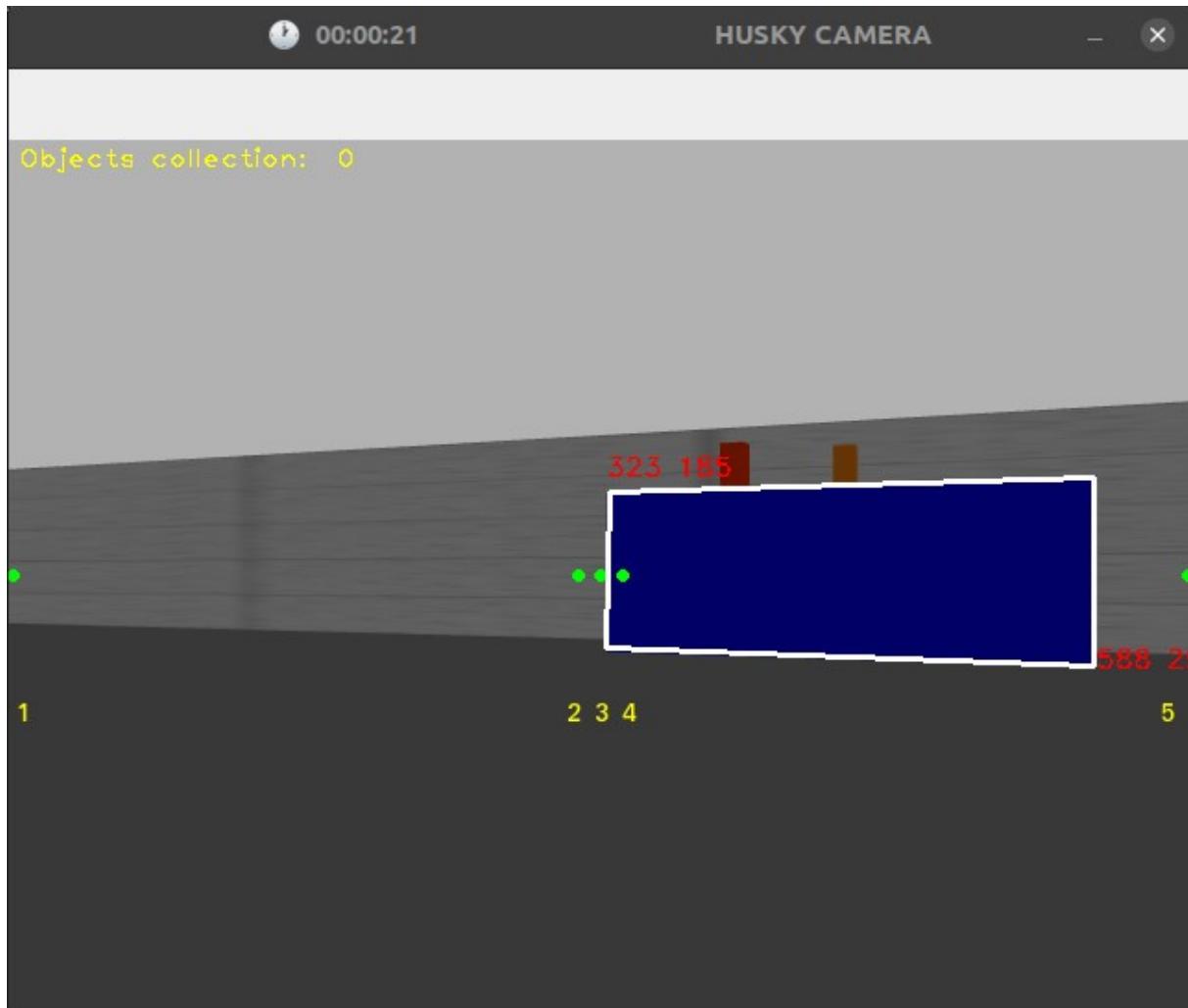
Εικόνα 1.44: Η κουκκίδα Νο3 έξω-δεξιά από την άκρη του μπλε τραπεζιού.

Περιστροφή του Husky προς τα αριστερά με ταχύτητα 0.06 rad/sec. μέχρι να βρεί μπλε χρώμα η κουκκίδα No3. Ουσιαστικά, ελέγχουμε αν η απόχρωση (Hue) της συντεταγμένης κουκκίδας No3 έχει μπλε χρώμα (κάθε άλλη χρονική στιγμή έχει διαφορετικό χρώμα, για παράδειγμα γκρι χρώμα, λόγω τοίχου). [εικόνα 1.45]



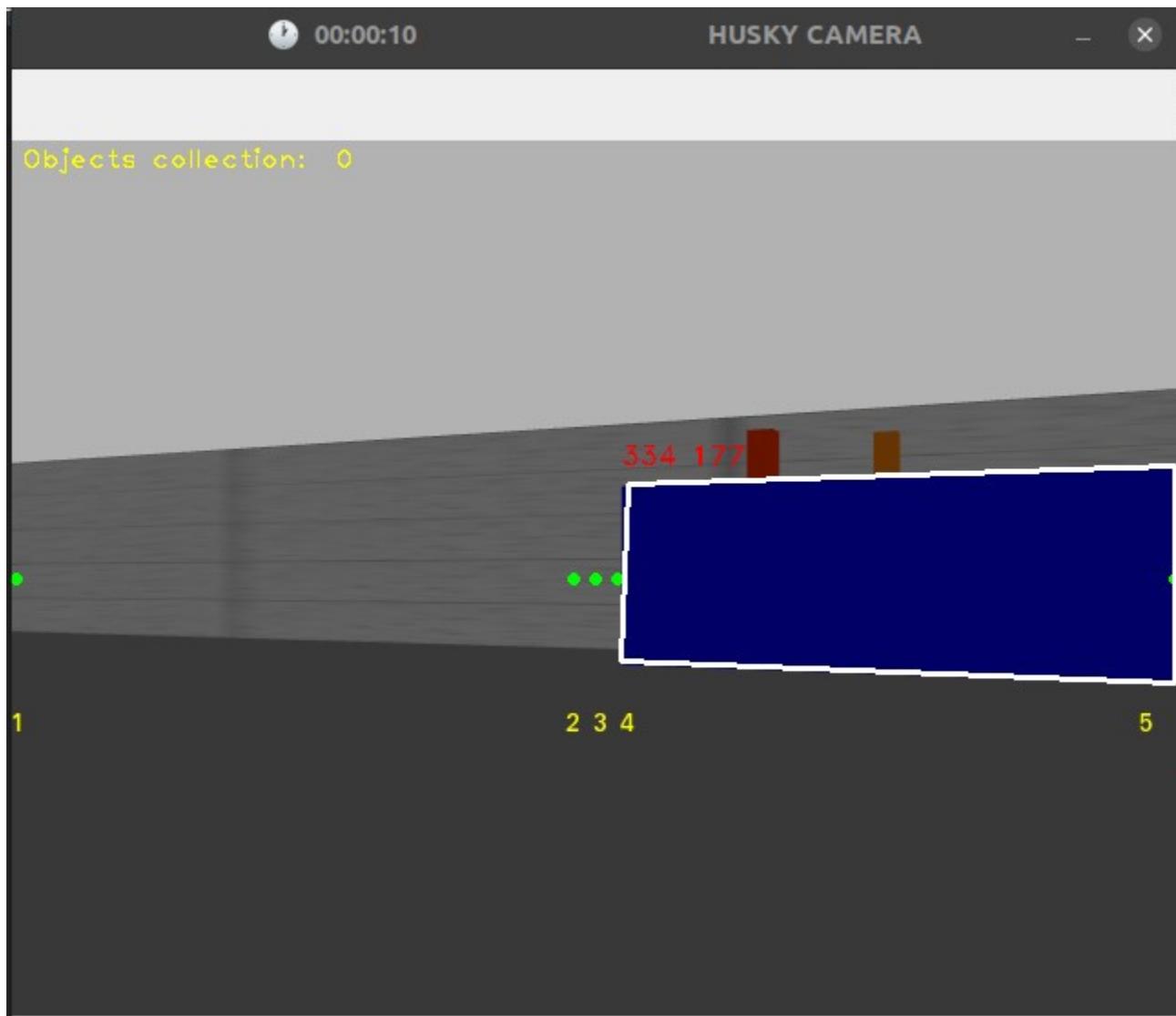
Εικόνα 1.45: Η κουκκίδα No3 εντός της δεξιάς άκρης του μπλε τραπεζιού.

Έπειτα, συνεχίζεται η αριστερή περιστροφή με ταχύτητα 0.06 rad/sec. μέχρι την αριστερή άκρη του τραπεζιού, δηλαδή γίνεται έλεγχος αν η απόχρωση (Hue) της συντεταγμένης κουκκίδας №3 δεν έχει πλέον μπλε χρώμα, αφού έχει πάει στα όρια της αριστερής άκρης του τραπεζιού. [εικόνα 1.46]



Εικόνα 1.46: Η κουκκίδα №3 εκτός της αριστερής άκρης του μπλε τραπεζιού.

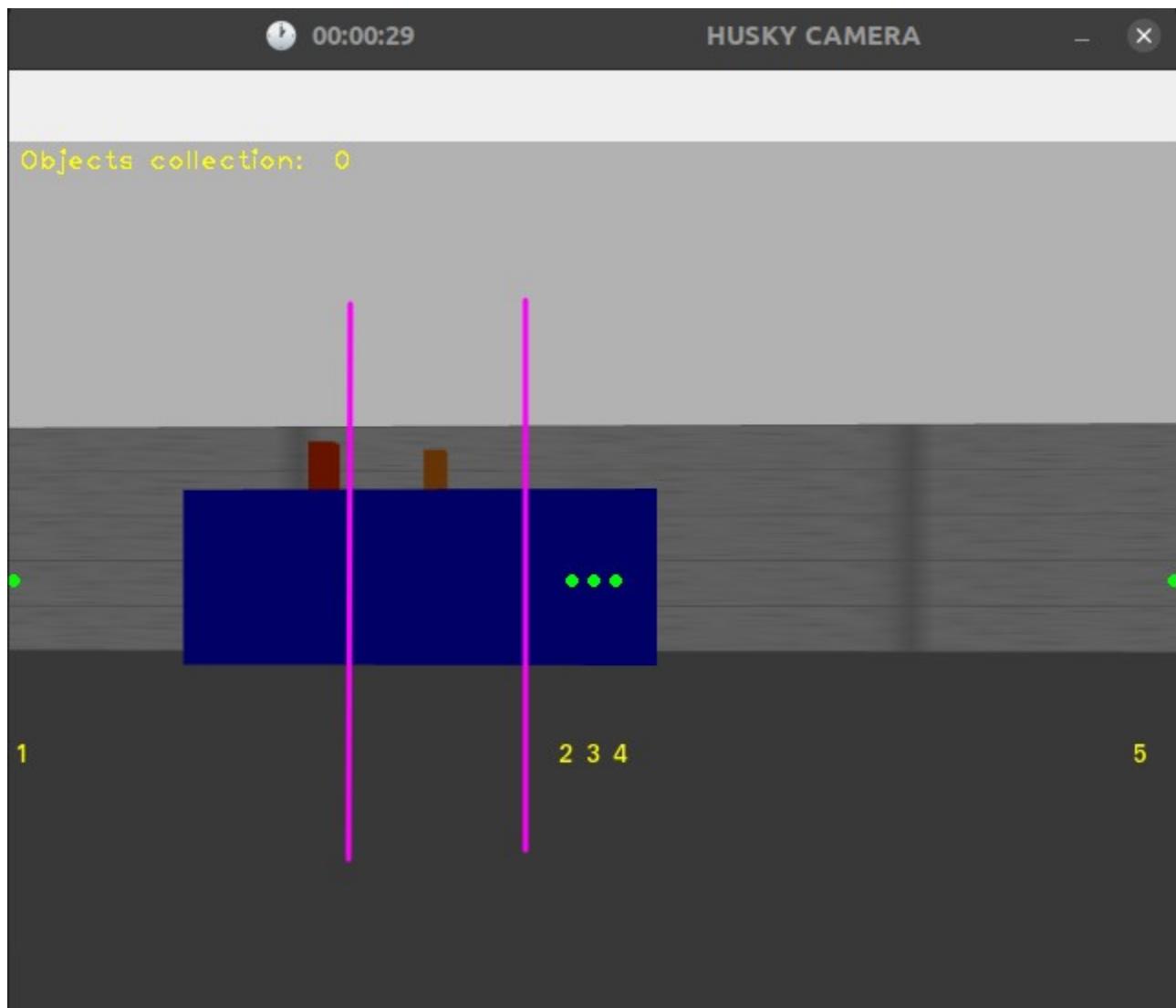
Στη συνέχεια, έχοντας αυτή την εικόνα, το ρομπότ μετακινείται μονάχα ευθεία με ταχύτητα 0.06 m/sec έως ότου η πράσινη κουκκίδα №5 εισέλθει εντός του δεξιού άκρου του τραπεζιού, δηλαδή μέχρι η απόχρωση (Hue) της συντεταγμένης κουκκίδας №5 να έχει μπλε χρώμα. [εικόνα 1.47]



Εικόνα 1.47: Η κουκκίδα №5 εντός της δεξιάς áκρης του μπλε τραπεζιού.

Μετά από αυτή τη διαδικασία, σειρά έχει η διόρθωση του προσανατολισμού, δηλαδή φέρνουμε το όχημα σε κάθετη θέση με το τραπέζι, με σκοπό να ελέγξουμε πάλι αν η κουκκίδα №3 βρίσκεται στο κέντρο του τραπεζιού. Οπότε τώρα υπάρχουν δύο περιπτώσεις για τη κουκκίδα №3, όπως και πριν. Να προσπέσει πάλι εντός του μπλε τραπεζιού ή εκτός αυτού.

- Αναλύουμε την περίπτωση να προσπέσει η κουκκίδα №3 εντός του τραπεζιού.



Εικόνα 1.48: Η κουκκίδα №3 εντός τραπεζιού, αλλά εκτός κεντρικών (μωβ) ορίων.

Στη περίπτωση αυτή της εικόνας 1.48, ακολουθούμε την εξής διαδικασία:

Περιστροφή του Husky προς τα αριστερά με ταχύτητα 0.06 rad/sec. μέχρι να μην έχει πλέον μπλε χρώμα η κουκκίδα №3. Ουσιαστικά, ελέγχουμε αν η απόχρωση (Hue) της συντεταγμένης κουκκίδας №3 δεν έχει μπλε χρώμα.

Στη συνέχεια, το ρομπότ μετακινείται ευθεία με ταχύτητα 0.06 m/sec έως ότου η πράσινη κουκκίδα No4 εξέλθει του αριστερού άκρου του τραπεζιού, δηλαδή μέχρι η απόχρωση (Hue) της συντεταγμένης κουκκίδας No4 να μην έχει πλέον μπλε χρώμα ή μέχρι το όχημα να πλησιάσει αρκετά κοντά στο τραπέζι, όπως μπορούμε να δούμε από την εικόνα 1.49 παρακάτω .

Έπειτα, διορθώνεται όπως και πριν ο προσανατολισμός του οχήματος, δηλαδή το όχημα έρχεται σε κάθετη θέση με το τραπέζι και έτσι τώρα το ρομπότ βρίσκεται εντός ορίων με μεγάλη πιθανότητα επιτυχίας, όπως φαίνεται στην εικόνα 1.40 .

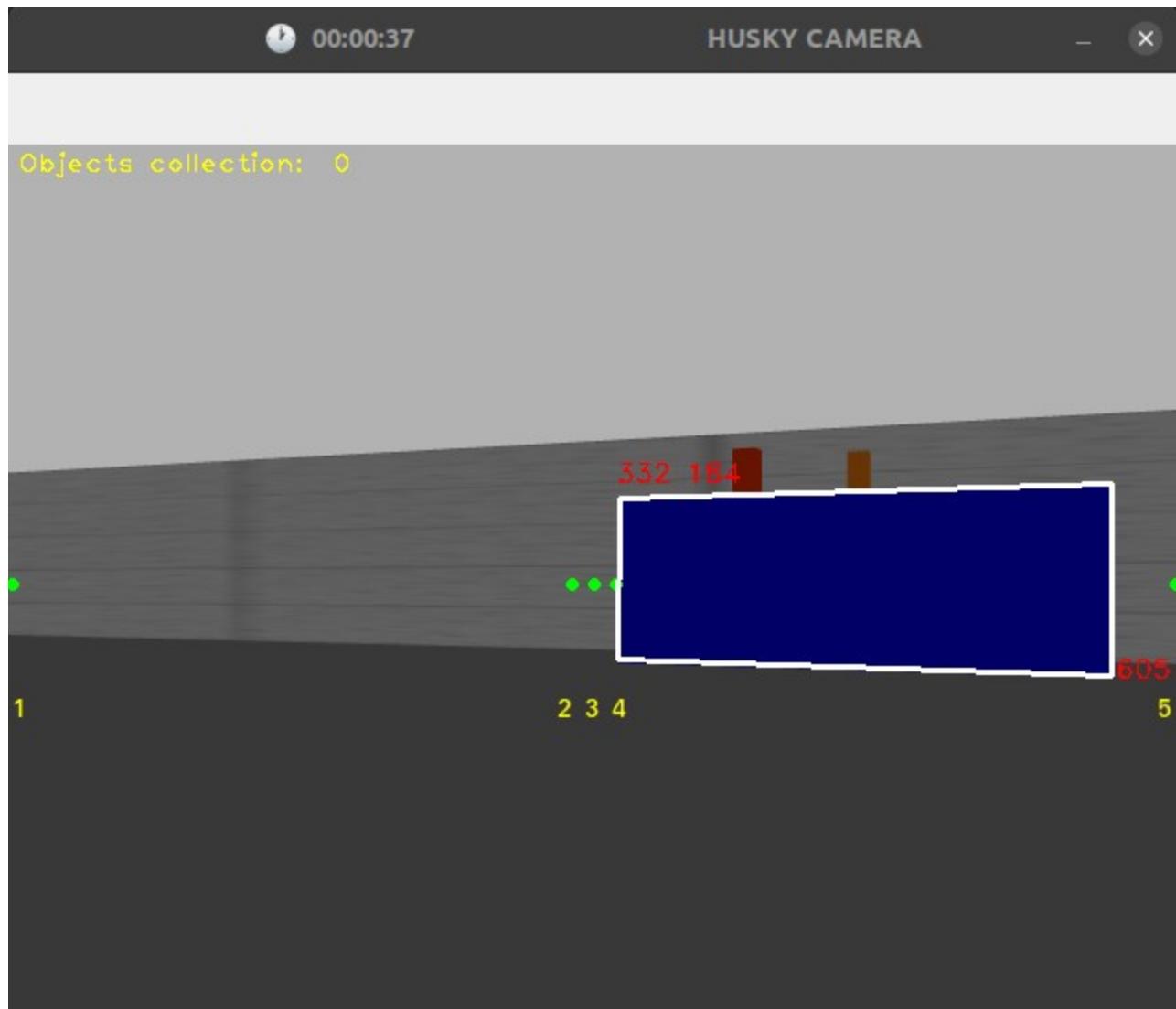
- *Αναλύουμε την περίπτωση να προσπέσει η κουκκίδα No3 εκτός του τραπεζιού.*

Όπως στην εικόνα 1.44, ακολουθούμε και και τα εξής βήματα:

Περιστροφή του Husky προς τα αριστερά με ταχύτητα 0.06 rad/sec. μέχρι να βρεί μπλε χρώμα η κουκκίδα No3. Ουσιαστικά, ελέγχουμε αν η απόχρωση (Hue) της συντεταγμένης κουκκίδας No3 έχει μπλε χρώμα (κάθε άλλη χρονική στιγμή έχει διαφορετικό χρώμα, για παράδειγμα γκρι χρώμα, λόγω τοίχου).

Έπειτα, συνεχίζεται η αριστερή περιστροφή με ταχύτητα 0.06 rad/sec. μέχρι την αριστερή άκρη του τραπεζιού, δηλαδή γίνεται έλεγχος αν η απόχρωση (Hue) της συντεταγμένης κουκκίδας No3 δεν έχει πλέον μπλε χρώμα, αφού έχει πάει στα όρια της αριστερής άκρης του τραπεζιού. [εικόνα 1.46]

Μετά, το ρομπότ μετακινείται ευθεία με ταχύτητα 0.06 m/sec έως ότου η πράσινη κουκκίδα No4 εξέλθει του αριστερού άκρου του τραπεζιού, δηλαδή μέχρι η απόχρωση (Hue) της συντεταγμένης κουκκίδας No4 να μην έχει πλέον μπλε χρώμα ή μέχρι το όχημα να πλησιάσει αρκετά κοντά στο τραπέζι, όπως φαίνεται από την εικόνα 1.49 παρακάτω.



Εικόνα 1.49: Η κουκκίδα №4 εκτός αριστερού άκρου τραπεζιού.

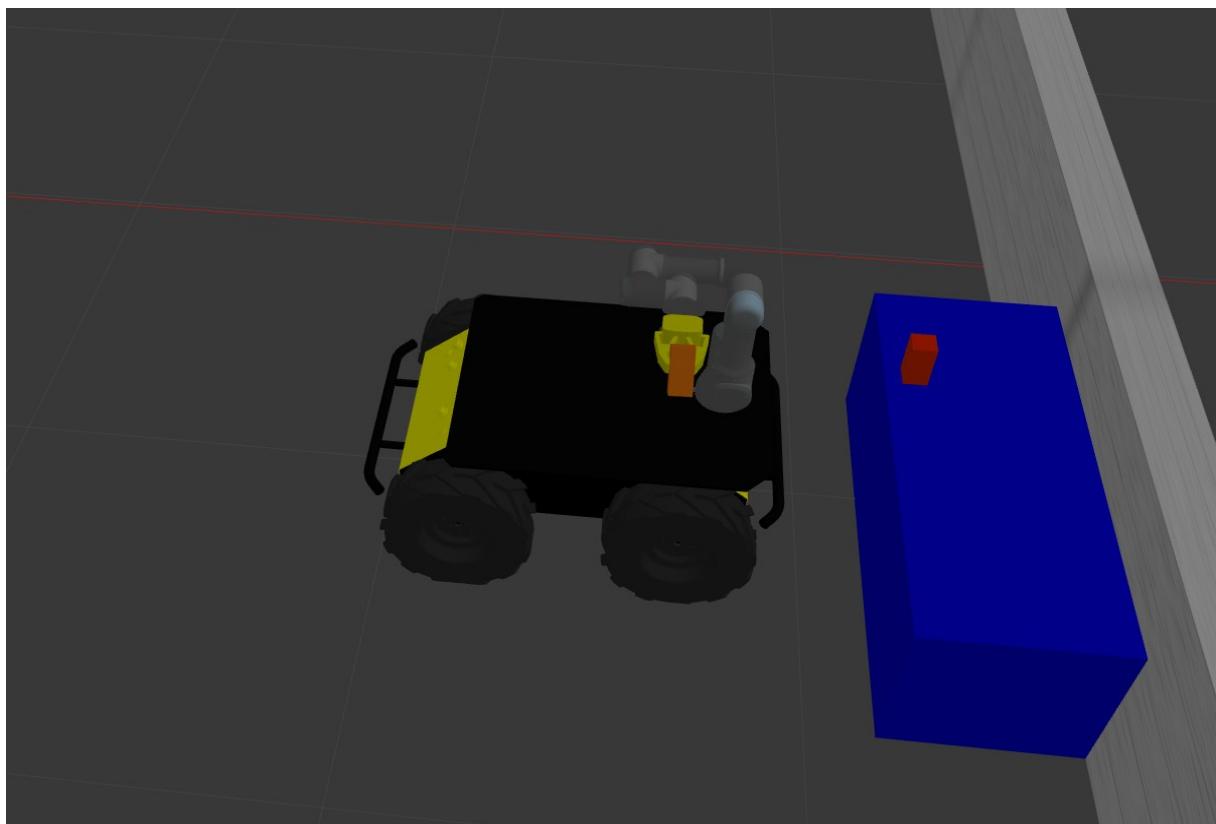
Τέλος, διορθώνεται ο προσανατολισμός του οχήματος, δηλαδή το όχημα έρχεται σε κάθετη θέση με το τραπέζι και έτσι το ρομπότ τώρα βρίσκεται εντός ορίων του τραπεζιού με μεγάλη πιθανότητα επιτυχίας, όπως φαίνεται στην εικόνα 1.40 .

1.7 ΟΠΙΣΘΟΧΩΡΗΣΗ HUSKY

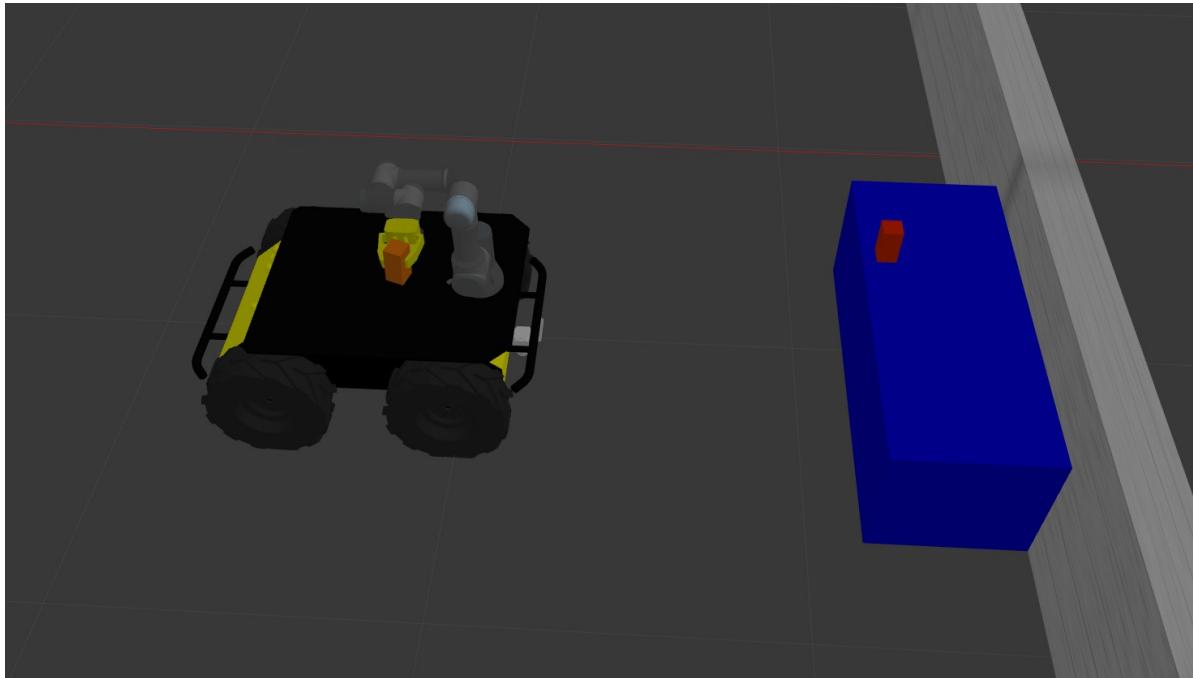
Καταρχάς, όταν ο βραχίονας είναι σε θέση να λάβει ένα αντικείμενο από ένα τραπέζι, το Husky πρέπει να σταματήσει σε απόσταση 0.58 μέτρων από το τραπέζι, ενώ αν ο βραχίονας θέλει να αφήσει ένα αντικείμενο σε ένα τραπέζι, τότε πρέπει το όχημα να σταματήσει 0.2 μέτρα μακριά, ώστε να έχει την δυνατότητα να το αφήσει στην υποδοχή του τραπεζιού.

Τη στιγμή που ο βραχίονας έχει πάρει ή αφήσει ένα αντικείμενο από/στο τραπέζι και έχει τοποθετηθεί πάλι στην αρχική του θέση (“Home position”), σειρά έχει η οπισθοχώρηση του Husky κατά 1 μέτρο από το τραπέζι με ασφάλεια.

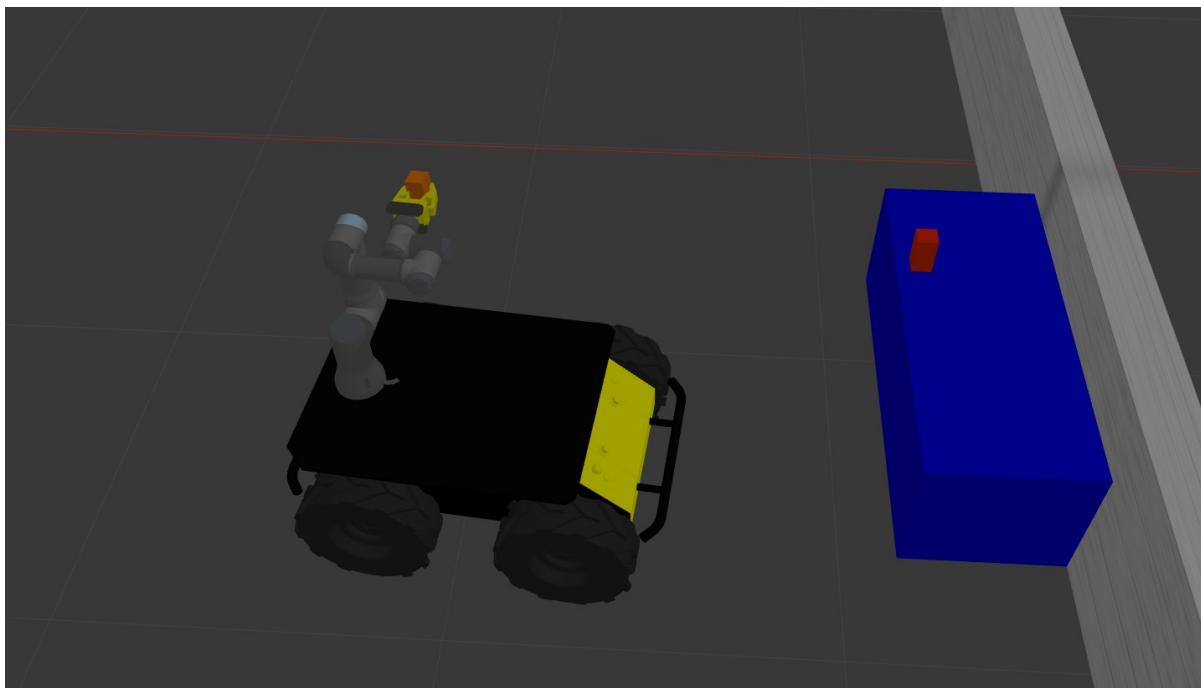
Για να γίνει καλύτερα κατανοητή η αλληλουχία των κινήσεων, παρακάτω παρατίθενται ένας μικρός αριθμός εικόνων με την αντίστοιχη επεξήγησή τους.



Εικόνα 1.50: Ο UR3 μόλις πήρε το αντικείμενο από το τραπέζι και τοποθετήθηκε σε “Home position”.



Εικόνα 1.51: Οπισθοχώρηση Husky κατά 1 μέτρο από το τραπέζι.



Εικόνα 1.52: Περιστροφή Husky κατά 180 μοίρες.

Η παραπάνω αλληλουχία κινήσεων συμβαίνει με σκοπό το Husky να έχει την ασφάλεια και δυνατότητα να περιστραφεί και να κατευθυνθεί προς άλλα τραπέζια για ανάλογη εργασία σε αυτά, όπως το να αφήσει ο βραχίονας το αντικείμενο στην υποδοχή ενός άλλου τραπεζιού ή να πάρει ένα αντικείμενο από ένα άλλο τραπέζι.

1.8 ΕΠΙΛΟΓΗ ΤΡΑΠΕΖΙΟΥ & ΑΝΤΙΚΕΙΜΕΝΟΥ

Όπως έχουμε ξαναπεί σε προηγούμενες υποενότητες, υπάρχουν συγκεκριμένα τέσσερα τραπέζια. Τα δύο τραπέζια έχουν πάνω διάφορα χρωματιστά αντικείμενα και τα υπόλοιπα δύο είναι τραπέζια που έχουν υποδοχές που μπαίνουν τα αντικείμενα αμέσα. Σκοπός των ρομπότ είναι να πάρουν τα αντικείμενα από τα αντίστοιχα τραπέζια και να τα τοποθετήσουν στα αντίστοιχα τραπέζια με τις υποδοχές (ανάλογα το χρώμα τους).

Από ποιό τραπέζι ούμως θα ξεκινήσει να αναζητά για αντικείμενα το Husky. Εξηγούμε παρακάτω.

Προφανώς τα τραπέζια από τα οποία τα ρομπότ παίρνουν τα αντικέιμενα είναι δύο.

Το πρώτο τραπέζι στο οποίο θα κατευθυνθεί το Husky ορίζεται με τυχαίο τρόπο.

Ύστερα, πηγαίνει εναλλάξ στα τραπέζια για να αναζητήσει τα υπόλοιπα αντικείμενα που έμειναν. Για παράδειγμα, ας ονομάσουμε τα δύο τραπέζια για συντομία ως A και B.

Αρχικά, είπαμε ότι η τυχαιότητα καθορίζει την επιλογή του πρώτου τραπεζιού από το όχημα. Έστω ότι αυτό το τραπέζι είναι το B. Μετά από τις εργασίες των δύο ρομπότ, δηλαδή να πάει το όχημα στο κατάλληλο τραπέζι με τις υποδοχές και ο UR3 να τοποθετήσει μέσα το αντικείμενο, σειρά έχει το τραπέζι A, κλπ.

Τέλος, η επιλογή των αντικειμένων γίνεται με τυχαίο τρόπο.

ΚΕΦΑΛΑΙΟ 2

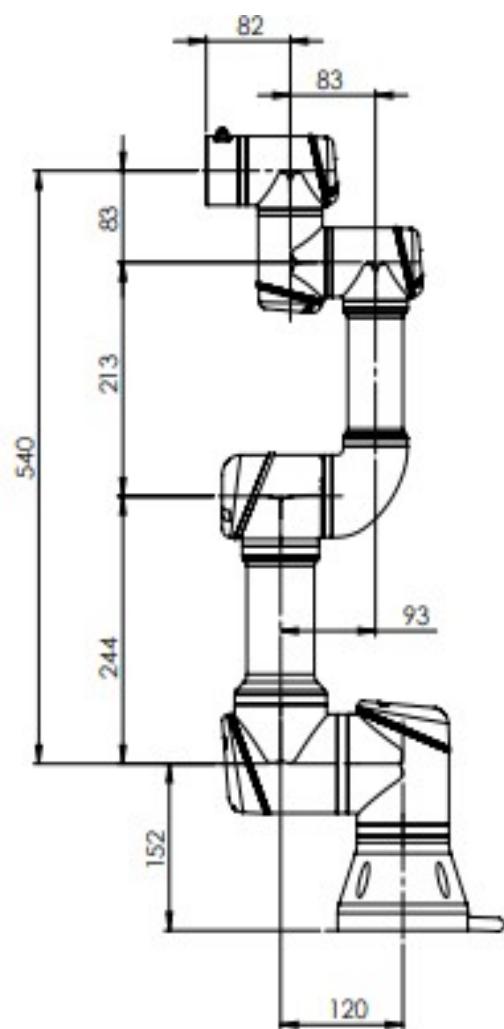
UR3



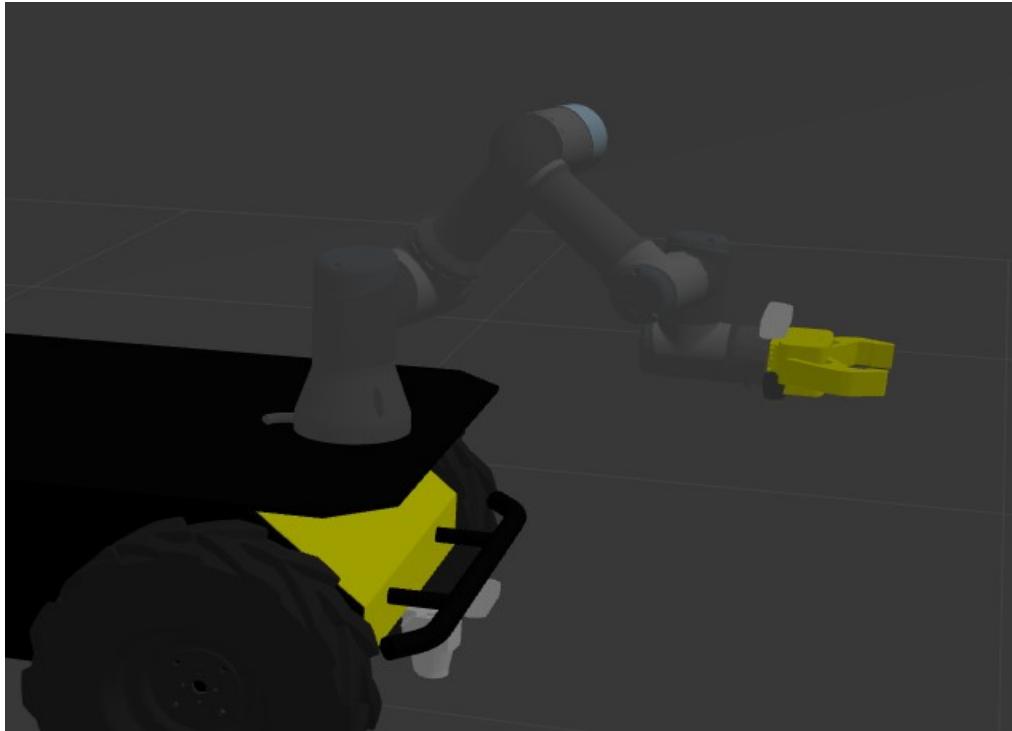
Εικόνα 2.1: Ο ρομποτικός βραχίονας Universal Robot 3.

Wiki: [Universal Robots](#)

2.1 ΔΙΑΣΤΑΣΕΙΣ ΚΑΙ ΠΛΗΡΟΦΟΡΙΕΣ



Εικόνα 2.2: Διαστάσεις UR3.



Εικόνα 2.3: Αναπαράσταση UR3 στο περιβάλλον Gazebo.

<i>Model</i>	<i>Release year</i>	<i>Payload (Kg)</i>	<i>Footprint(m)</i>	<i>Weight(Kg)</i>	<i>Reach(m)</i>
UR3	2015	3	0.19	11.2	0.5

Πίνακας 2.1: Μερικές πληροφορίες σχετικά με το μοντέλο UR3 της Universal robots.

Model: Ο UR3 είναι ένας ρομποτικός βραχίονας 6 βαθμών ελευθερίας.

Release year: Το μοντέλο UR3 κυκλοφόρησε για πρώτη φορά το 2015.

Payload: Ο όρος 'payload' αναφέρεται στο φορτίο ή βάρος το οποίο μπορεί να σηκώσει και να μετακινήσει ένας ρομποτικός βραχίονας. Συγκεκριμένα, ο UR3 μπορεί να σηκώσει φορτίο συνολικού βάρους 3 κιλών, κάνοντάς τον αρκετά ευέλικτο στις εργασίες.

Footprint: Ο όρος 'footprint' σε ένα ρομποτικό βραχίονα αποτελεί το 'αποτύπωμα' που 'αφήνει' στο έδαφος, δηλαδή ο χώρος που καταλαμβάνει.

Weight: Το συνολικό βάρος του ρομποτικού βραχίονα UR3 αντιστοιχεί σε 11 κιλά.

Reach: Ο όρος 'Reach' αποτελεί την εμβέλεια ενός ρομποτικού βραχίονα, δηλαδή τη μέγιστη απόσταση ή μήκος που μπορεί να εκτείνεται ο βραχίονας από τη βάση του.

2.2 ΕΞΟΠΛΙΣΜΟΣ ΚΑΙ ΧΡΗΣΗ ΒΡΑΧΙΟΝΑ

Ένας ρομποτικός βραχίονας έχει πολλές χρήσεις σε διάφορους τομείς, λόγω των διαφορετικών δυνατοτήτων και λειτουργιών που προσφέρει. Οι τομείς αυτοί μπορεί να είναι:

1) Βιομηχανία: Στη βιομηχανία, ρομποτικοί βραχίονες χρησιμοποιούνται για τον αυτοματισμό διαδικασιών παραγωγής, συναρμολόγησης και συσκευασίας με μεγάλη ακρίβεια και ταχύτητα.

2) Ιατρική: Στον τομέα της ιατρικής, ρομποτικοί βραχίονες χρησιμοποιούνται για χειρουργικές επεμβάσεις με ακρίβεια και η ελεγχόμενη κίνηση.

3) Εκπαίδευση: Οι ρομποτικοί βραχίονες χρησιμοποιούνται επίσης σε περιβάλλοντα εκπαίδευσης για την εκμάθηση και την εκπαίδευση σε ρομποτικές εφαρμογές και προγραμματισμό.

4) Εξωτερικές Εφαρμογές: Οι ρομποτικοί βραχίονες μπορούν να χρησιμοποιούνται σε εξωτερικές εφαρμογές, όπως η αντιμετώπιση επειγόντων καταστάσεων, όπως την καταστολή πυρκαγιών ή την εξερεύνηση από απόσταση σε επικίνδυνα περιβάλλοντα.

Συνεπώς, οι ρομποτικοί βραχίονες παρέχουν τη δυνατότητα εκτέλεσης δύσκολων, επικίνδυνων ή/και απλών εργασιών με τεράστια ακρίβεια και αποδοτικότητα, βοηθώντας έτσι στη βελτίωση της ασφάλειας, της παραγωγικότητας και της ανθρώπινης υγείας.

Επιπλέον, ένας ρομποτικός βραχίονας, ανάλογα την ανατιθέμενη εργασία του μπορεί να εξοπλίζεται με πολλά εργαλεία, όπως:

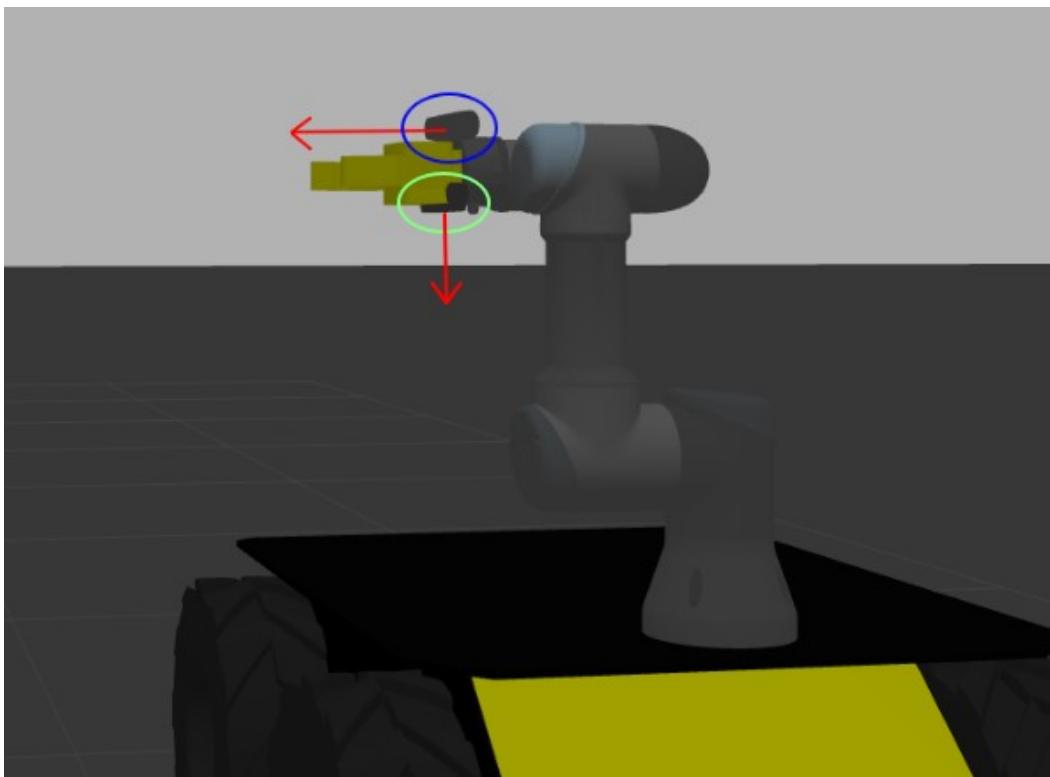
- Λέιζερ
- Αισθητήρες πίεσης
- Γυροσκόπια
- Αισθητήρες θερμοκρασίας/υγρασίας/ταχύτητας/επιτάχυνσης/μέτρησης απόστασης
- Εργαλεία ιατρικής
- Κάμερες
- Εργαλεία βιομηχανικής χρήσης (ηλεκτροκόλληση, τρυπάνι, αρπάγη, βεντούζα, κ.ά)
- Gps

2.3 ΚΑΜΕΡΕΣ

Ο ρομποτικός βραχίονας UR3 είναι εξοπλισμένος με δύο κάμερες βάθους d435.

Όλες οι πληροφορίες για τις κάμερες βρίσκονται στο κεφάλαιο **1.3** και υπάρχει αναλυτική περιγραφή για αυτές.

Η πρώτη, είναι τοποθετημένη στο εμπρόσθιο πάνω μέρος του τελικού στοιχείου δράσης, δηλαδή την αρπάγη, ενώ η δεύτερη στο εμπρόσθιο κάτω μέρο της αρπάγης, με διαφορετικούς προσανατολισμούς η καθεμία, όπως φαίνεται παρακατω.

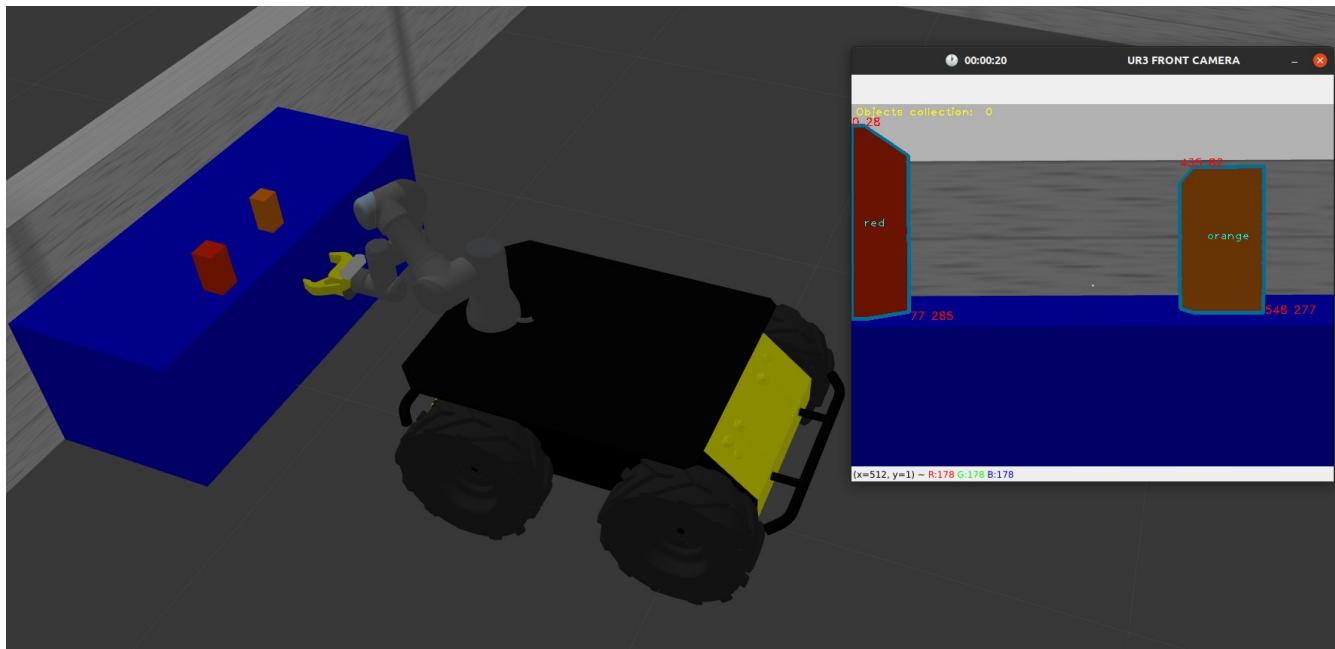


Εικόνα 2.4: Αναπαράσταση πάνω και κάτω κάμερας UR3.

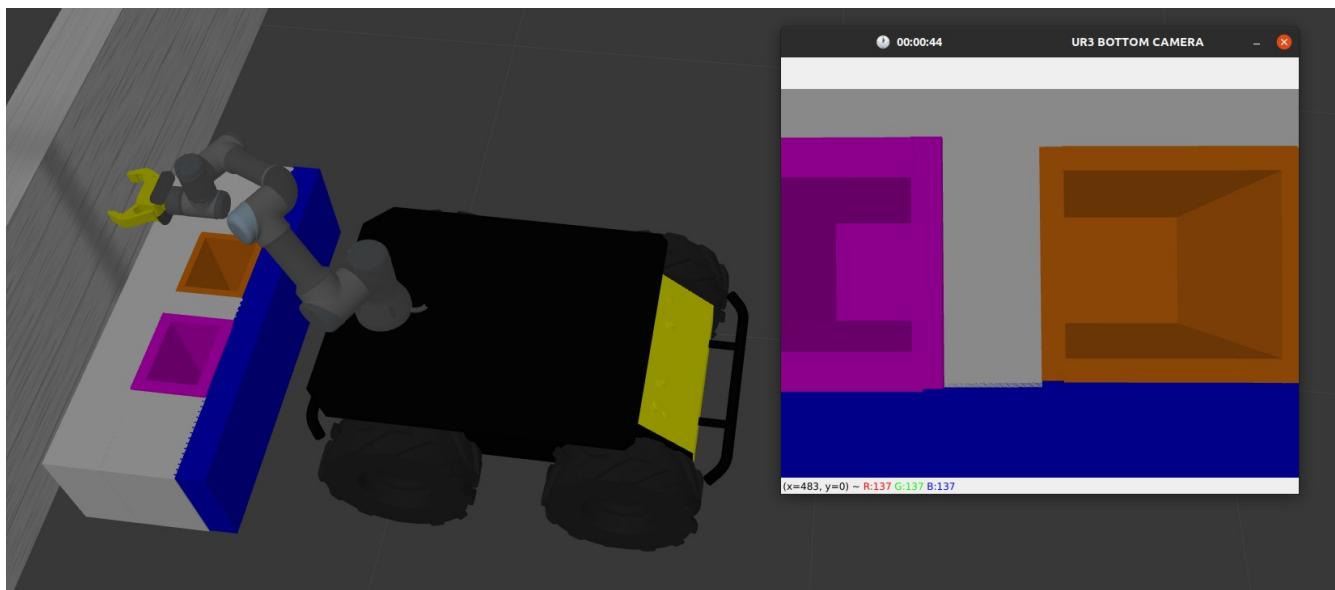
Ο σκοπός αυτών των δύο καμερών βάθους είναι να κατευθύνει την αρπάγη του βραχίονα όσο πιό κοντά στο κέντρο των αντικειμένων ή υποδοχών (ανάλογα το σκοπό εκείνης της στιγμής), ώστε να πιάσει ή ελευθερώσει αντίστοιχα το αντικείμενο.

Οι κάμερες ουσιαστικά λειτουργούν ως τα μάτια του βραχίονα.

2.3.1 ΑΠΕΙΚΟΝΙΣΗ ΠΕΡΙΒΑΛΛΟΝΤΟΣ ΜΕΣΩ ΚΑΜΕΡΩΝ



Εικόνα 2.5: Μπροστινή κάμερα βραχίονα UR3



Εικόνα 2.6: Κάτω κάμερα βραχίονα UR3

2.4 ΕΥΘΕΙΑ ΚΙΝΗΜΑΤΙΚΗ

Η ευθεία κινηματική ενός ρομποτικού βραχίονα ασχολείται με τον υπολογισμό της θέσης και του προσανατολισμού του τελικού στοιχείου δράσης (αρπάγη, βεντούζα, τρυπάνι, κα.) του βραχίονα σε σχέση με το αδρανειακό σύστημα αναφοράς, γνωρίζοντας όμως τις γωνίες όλων των αρθρώσεων.

Ο βασικός τρόπος για τον υπολογισμό της ευθείας κινηματικής είναι μέσω του πίνακα Denavit-Hartenberg (D-H). Ο πίνακας D-H αναπαριστά τις συνδέσεις μεταξύ των αρθρώσεων του βραχίονα και τοποθετεί το σύστημα συντεταγμένων του τελικού στοιχείου δράσης (ΤΣΔ) σε σχέση με το αδρανειακό σύστημα αναφοράς του βραχίονα. Ο πίνακας D-H περιέχει τις παραμέτρους Denavit-Hartenberg θ_i , a_i , d_i και α_i , οι οποίες αναφέρονται παρακάτω (Πίνακας 2.2).

Έτσι, όπως είπαμε, μπορούμε να προσδιορίσουμε τη θέση και τον προσανατολισμό του τελικού στοιχείου δράσης του βραχίονα σε σχέση με το αδρανειακό σύστημα αναφοράς. Αυτό μπορεί να επιτευχθεί με τον πολλαπλασιασμό των ομογενών πινάκων μετασχηματισμών, από τη βάση του βραχίονα έως το τελικό στοιχείο δράσης (ΤΣΔ).

Παρακάτω αναφέρεται ο πίνακας ομογενούς μετασχηματισμού, ο οποίος χρησιμοποιείται για κάθε άρθρωση βάσει των αντίστοιχων παραμέτρων Denavit-Hartenberg της άρθρωσης.

	Theta (rad)	a (m)	d (m)	alpha (rad)	Dynamics	Mass (Kg)
Joint 1	θ_1	0	d_1	$\frac{\pi}{2}$	Link 1	2
Joint 2	θ_2	a_2	0	0	Link 2	3.42
Joint 3	θ_3	a_3	0	0	Link 3	1.26
Joint 4	θ_4	0	d_4	$\frac{\pi}{2}$	Link 4	0.8
Joint 5	θ_5	0	d_5	$-\frac{\pi}{2}$	Link 5	0.8
Joint 6	θ_6	0	d_6	0	Link 6	0.35

Πίνακας 2.2: Πίνακας Denavit-Hartenberg παραμέτρων για τον βραχίονα UR3.

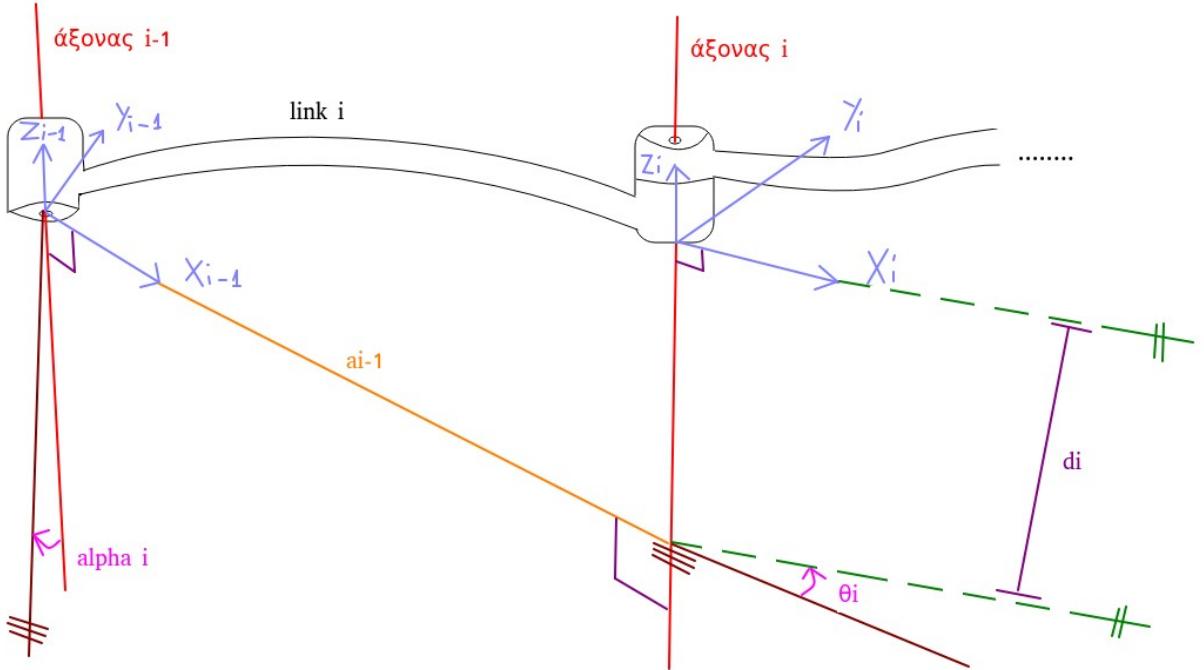
Οι παράμετροι Denavit-Hartenberg (D-H) χρησιμοποιούνται για να περιγράψουν τις γεωμετρικές και κινηματικές σχέσεις μεταξύ των αρθρώσεων ενός ρομποτικού βραχίονα. Οι παράμετροι D-H καθορίζουν τη θέση και τον προσανατολισμό της κάθε άρθρωσης σε σχέση με την προηγούμενη άρθρωση.

d_1	0.15190
a_2	-0.24365
a_3	-0.21235
d_4	0.11235
d_5	0.08535
d_6	0.08190

Πίνακας 2.3: Προσδιορισμός των παραμέτρων Denavit-Hartenberg για τον UR3 .

θ_i	Η γωνία από τον άξονα x_{i-1} προς τον x_i μετρούμενη γύρω από τον άξονα z_i .
a_i	Η απόσταση μεταξύ των z_{i-1} και z_i μετρούμενη κατά μήκος του άξονα x_i
d_i	Η απόσταση μεταξύ των x_{i-1} και x_i μετρούμενη κατά μήκος του άξονα z_i
α_i	Η γωνία μεταξύ των z_{i-1} και z_i μετρούμενη ως προς τον άξονα x_i

Πίνακας 2.4: Ορισμός παραμέτρων θ_i , a_i , d_i και α_i .



Εικόνα 2.7: Αναπαράσταση και σχηματική περιγραφή του πίνακα 2.4.

$${}^{i-1}\mathbf{T}_i = \begin{pmatrix} \cos(\theta_i) & -\sin(\theta_i) \cdot \cos(a_i) & \sin(\theta_i) \cdot \sin(a_i) & a_i \cdot \cos(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \cdot \cos(a_i) & -\cos(\theta_i) \cdot \sin(a_i) & a_i \cdot \sin(\theta_i) \\ 0 & \sin(a_i) & \cos(a_i) & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Ο παραπάνω πίνακας μπορεί να χρησιμοποιηθεί για κάθε άρθρωση του βραχίονα. Συγκεκριμένα, θα υπολογιστούν οι πίνακες ${}^0\mathbf{T}_1$, ${}^1\mathbf{T}_2$, ${}^2\mathbf{T}_3$, ${}^3\mathbf{T}_4$, ${}^4\mathbf{T}_5$, ${}^5\mathbf{T}_6$, ένας για κάθε άρθρωση, εφόσον ο βραχίονας UR3 διαθέτει 6 αρθρώσεις. Εφόσον υπολογιστούν οι 6 παραπάνω πίνακες, μπορούν να πολλαπλασιαστούν, ώστε να δώσουν τον ${}^0\mathbf{T}_6$, ο οποίος αποτελεί τον πίνακα που περιέχει ένα πίνακα προσανατολισμού καθώς και ένα διάνυσμα θέσης του τελικού στοιχείου δράσης αναφορικά με την βάση του ρομπότ.

Ο πολλαπλασιασμός είναι ο εξής: ${}^0\mathbf{T}_6 = {}^0\mathbf{T}_1 \cdot {}^1\mathbf{T}_2 \cdot {}^2\mathbf{T}_3 \cdot {}^3\mathbf{T}_4 \cdot {}^4\mathbf{T}_5 \cdot {}^5\mathbf{T}_6$

Όπως επισημάνθηκε, ο ${}^0\mathbf{T}_6$ αποτελείται από ένα πίνακα προσανατολισμού καθώς και ένα διάνυσμα θέσης και έχει την εξής μορφή:

$${}^0\mathbf{T}_6 = \begin{pmatrix} R & P \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

όπου R ένας πίνακας προσανατολισμού διάστασης $[3 \times 3]$ και P ένα διάνυσμα θέσης διάστασης $[3 \times 1]$.

Επίσης, Ο 0T_6 μπορεί να γραφεί αναλυτικότερα ως εξής:

$${}^0T_6 = \begin{pmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix}, \text{ όπου } R = \begin{pmatrix} n_x & s_x & a_x \\ n_y & s_y & a_y \\ n_z & s_z & a_z \end{pmatrix} \text{ και } P = \begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix}.$$

n_x	$c_6 \cdot (s_1 \cdot s_5 + c_1 \cdot c_{234} \cdot c_5) - s_6 \cdot c_1 \cdot s_{234}$
n_y	$c_6 \cdot (-c_1 \cdot s_5 + s_1 \cdot c_{234} \cdot c_5) - s_6 \cdot s_1 \cdot s_{234}$
n_z	$s_{234} \cdot c_5 \cdot c_6 + c_{234} \cdot s_6$
s_x	$-s_6 \cdot (s_1 \cdot s_5 + c_5 \cdot c_1 \cdot c_{234}) - c_6 \cdot c_1 \cdot s_{234}$
s_y	$-s_6 \cdot (-c_1 \cdot s_5 + c_5 \cdot s_1 \cdot c_{234}) - c_6 \cdot s_1 \cdot s_{234}$
s_z	$-s_{234} \cdot c_5 \cdot s_6 + c_{234} \cdot c_6$
a_x	$-s_{234} \cdot c_5 \cdot s_6 + c_{234} \cdot c_6$
a_y	$-s_5 \cdot s_1 \cdot c_{234} - c_1 \cdot c_5$
a_z	$-s_5 \cdot s_{234}$
p_x	$d_6 \cdot (s_1 \cdot c_5 - c_1 \cdot s_5 \cdot c_{234}) + d_5 \cdot c_1 \cdot s_{234} + d_4 \cdot s_1 + a_3 \cdot c_1 \cdot c_{23} + a_2 \cdot c_1 \cdot c_2$
p_y	$-d_6 \cdot (s_1 \cdot s_5 \cdot c_{234} + c_1 \cdot c_5) + d_5 \cdot s_1 \cdot s_{234} - d_4 \cdot c_1 + a_3 \cdot s_1 \cdot c_{23} + a_2 \cdot s_1 \cdot c_2$
p_z	$-d_6 \cdot s_{234} \cdot s_5 - d_5 \cdot c_{234} + a_3 \cdot s_{23} + a_2 \cdot s_2 + d_1$

Πίνακας 2.5: Προσδιορισμός εξισώσεων για τα στοιχεία του πίνακα R και διανύσματος P.

Σημείωση: Για συντομία στην γραφή των παραπάνω εξισώσεων, θέτουμε τα εξής:

$$\begin{aligned} c_\theta &= \cos(\theta) \\ c_{12} &= \cos(\theta_1 + \theta_2) \end{aligned}$$

$$\begin{aligned} s_\theta &= \sin(\theta) \\ s_{12} &= \sin(\theta_1 + \theta_2) \end{aligned}$$

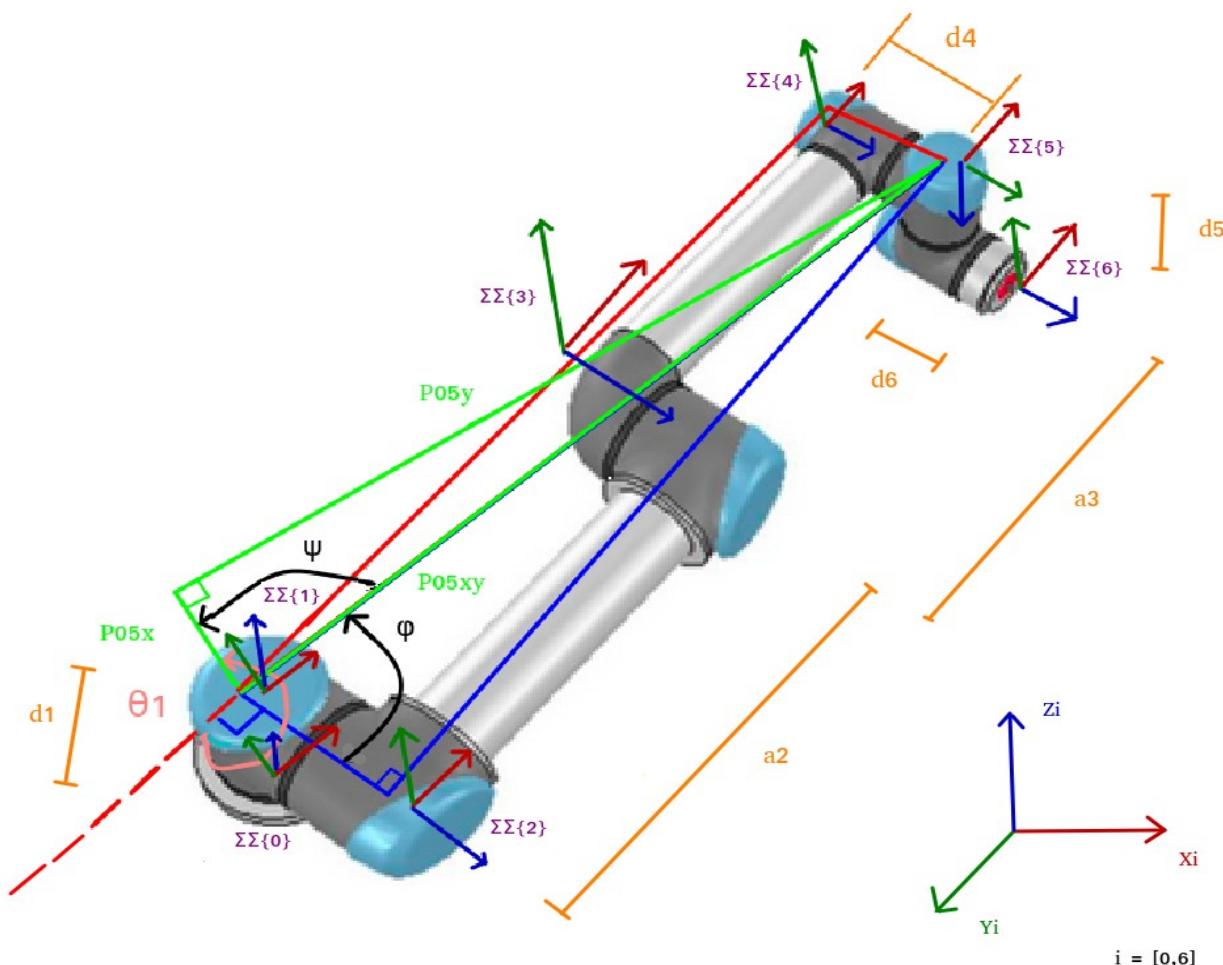
$$\begin{aligned} c_{123} &= \cos(\theta_1 + \theta_2 + \theta_3) \\ s_{123} &= \sin(\theta_1 + \theta_2 + \theta_3) \end{aligned}$$

2.5 ΑΝΤΙΣΤΡΟΦΗ ΚΙΝΗΜΑΤΙΚΗ

Ο στόχος της αντίστροφης κινηματικής σε έναν ρομποτικό βραχίονα είναι να υπολογίσει τις απαιτούμενες γωνίες των αρθρώσεων του, γνωρίζοντας τη θέση και τον προσανατολισμό του τελικού στοιχείου δράσης του βραχίονα. Αυτό επιτρέπει τον ακριβή έλεγχο και την προσαρμογή της θέσης του τερματικού στοιχείου δράσης του βραχίονα.

Με άλλα λόγια, η αντίστροφη κινηματική επιλύει το πρόβλημα του πώς να ρυθμιστούν οι γωνίες των αρθρώσεων ενός βραχίονα, ώστε η τελική θέση του εργαλείου να βρεθεί σε συγκεκριμένες συντεταγμένες χώρου. Αυτό είναι ιδιαίτερα χρήσιμο σε εφαρμογές ρομποτικής, όπου οι γωνίες των αρθρώσεων του ρομπότ πρέπει να υπολογιστούν για να επιτευχθεί μια επιθυμητή θέση και προσανατολισμός του εργαλείου του ρομπότ.

➤ ΥΠΟΛΟΓΙΣΜΟΣ ΤΗΣ ΕΞΙΣΩΣΗΣ ΓΙΑ ΤΗΝ ΓΩΝΙΑ Θ_1 :



Εικόνα 2.8: Γεωμετρική απεικόνιση για την εύρεση εξίσωσης για την γωνία Θ_1 .

Καταρχάς, πρέπει να προσδιοριστεί το διάνυσμα 0P_5 , το οποίο θα χρησιμοποιήσουμε παρακάτω. Ο υπολογισμός είναι ο εξής:

$${}^0P_5 = {}^0P_6 + {}^0T_6 \cdot \begin{bmatrix} 0 \\ 0 \\ -d_6 \\ 1 \end{bmatrix}, \text{ όπου το διάνυσμα } {}^0P_6 \text{ αποτελεί τη θέση του } \Sigma\{6\} \text{ σε σχέση}$$

με το $\Sigma\{0\}$ και επίσης ο πίνακας 0T_6 , ο οποίος δίνεται παραπάνω, αποτελείται από ένα 3×3 πίνακα προσανατολισμού και από ένα 3×1 διάνυσμα, το 0P_6 .

Παρακάτω, υπολογίζεται η γωνία Θ_1 :

$$\psi = \text{atan2}({}^0P_{5y}, {}^0P_{5x})$$

$$\cos(\phi) = \pm \frac{d_4}{\sqrt({}^0P_{5y}^2 + {}^0P_{5x}^2)} \Rightarrow \phi = \pm \arccos\left(\frac{d_4}{\sqrt({}^0P_{5y}^2 + {}^0P_{5x}^2)}\right)$$

$$\theta_{1,1} = \text{atan2}({}^0P_{5y}, {}^0P_{5x}) + \arccos\left(\frac{d_4}{\sqrt({}^0P_{5y}^2 + {}^0P_{5x}^2)}\right) + \frac{\pi}{2}$$

$$\theta_{1,2} = \text{atan2}({}^0P_{5y}, {}^0P_{5x}) - \arccos\left(\frac{d_4}{\sqrt({}^0P_{5y}^2 + {}^0P_{5x}^2)}\right) + \frac{\pi}{2}$$

Τελικά:

$$\Theta_1 = \theta_{1,1} \text{ ή } \theta_{1,2}$$

➤ **ΥΠΟΛΟΓΙΣΜΟΙ ΤΩΝ ΕΞΙΣΩΣΕΩΝ ΓΙΑ ΤΙΣ ΓΩΝΙΕΣ Θ_5 ΚΑΙ Θ_6 :**

Παρακάτω παρουσιάζεται μία τεχνική για την εύρεση εξισώσεων των γωνιών Θ_5 και Θ_6 .

Καταρχάς, βρίσκουμε τον πίνακα περιστροφής 3R_6 ως εξής:

$${}^0R_6 = {}^0R_3 \cdot {}^3R_6 \Rightarrow {}^3R_6 = ({}^0R_3)^{-1} \cdot {}^0R_6$$

Γενικότερα, για οποιονδήποτε πίνακα περιστροφής ισχύει ότι: $({}^{(i-1)}R_i)^{-1} = ({}^{(i-1)}R_i)^T$,

δηλαδή ο αντίστροφος ενός πίνακα περιστροφής ταυτίζεται με τον ανάστροφό του.

Οπότε και στην παραπάνω εξίσωση ισχύει ότι: $({}^0R_3)^{-1} = ({}^0R_3)^T$.

Μπορούμε εύκολα να βρούμε τους πίνακες 0R_6 και 0R_3 από τους αντίστοιχους ομογενείς πίνακες μεταχηματισμού 0T_6 και 0T_3 . Συγκεκριμένα:

$${}^3T_6 = \begin{pmatrix} -s_4 \cdot s_6 + c_4 \cdot c_5 \cdot c_6 & -s_4 \cdot c_6 - s_6 \cdot c_4 \cdot c_5 & -s_5 \cdot c_4 & d_5 \cdot s_4 - d_6 \cdot s_5 \cdot c_4 \\ s_4 \cdot c_5 \cdot c_6 + s_6 \cdot c_4 & -s_4 \cdot s_6 \cdot c_5 + c_4 \cdot c_6 & -s_4 \cdot s_5 & -d_6 \cdot s_4 \cdot s_5 - d_5 \cdot c_4 \\ s_5 \cdot c_6 & -s_5 \cdot s_6 & c_5 & d_6 \cdot c_5 + d_4 \\ 0 & 0 & 0 & 1 \end{pmatrix} \text{ και}$$

$${}^0T_3 =$$

$$\begin{pmatrix} -s_2 \cdot s_3 \cdot c_2 + c_1 \cdot c_2 \cdot c_3 & -s_1 \cdot s_2 \cdot s_3 + s_1 \cdot c_2 \cdot c_3 & s_2 \cdot c_3 + s_3 \cdot c_2 & -a_3 \cdot s_2 \cdot s_3 \cdot c_1 + a_3 \cdot c_1 \cdot c_2 \cdot c_3 + a_2 \cdot c_1 \cdot c_2 \\ -s_2 \cdot c_1 \cdot c_3 - s_3 \cdot c_1 \cdot c_2 & -s_1 \cdot s_2 \cdot c_3 - s_1 \cdot s_3 \cdot c_2 & -s_2 \cdot s_3 + c_2 \cdot c_3 & -a_3 \cdot s_1 \cdot s_2 \cdot s_3 + a_3 \cdot s_1 \cdot c_2 \cdot c_3 + a_2 \cdot s_1 \cdot c_2 \\ s_1 & -c_1 & 0 & a_3 \cdot s_2 \cdot c_3 + a_2 \cdot s_2 + s_3 \cdot c_2 + d_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Ακολουθεί ο υπολογισμός της παραπάνω εξίσωσης: ${}^3R_6 = ({}^0R_3)^{-1} \cdot {}^0R_6 \Rightarrow$

$$\begin{pmatrix} -s_4 \cdot s_6 + c_4 \cdot c_5 \cdot c_6 & -s_4 \cdot c_6 - s_6 \cdot c_4 \cdot c_5 & -s_5 \cdot c_4 \\ s_4 \cdot c_5 \cdot c_6 + s_6 \cdot c_4 & -s_4 \cdot s_6 \cdot c_5 + c_4 \cdot c_6 & -s_4 \cdot s_5 \\ s_5 \cdot c_6 & -s_5 \cdot s_6 & c_5 \end{pmatrix} =$$

$$\begin{pmatrix} -s_2 \cdot s_3 \cdot c_2 + c_1 \cdot c_2 \cdot c_3 & -s_1 \cdot s_2 \cdot s_3 + s_1 \cdot c_2 \cdot c_3 & s_2 \cdot c_3 + s_3 \cdot c_2 \\ -s_2 \cdot c_1 \cdot c_3 - s_3 \cdot c_1 \cdot c_2 & -s_1 \cdot s_2 \cdot c_3 - s_1 \cdot s_3 \cdot c_2 & -s_2 \cdot s_3 + c_2 \cdot c_3 \\ s_1 & -c_1 & 0 \end{pmatrix} \cdot \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}$$

Από την παραπάνω εξίσωση, προκύπτουν νέες εξισώσεις. Εξισώνοντας, λοιπόν, αντίστοιχες γραμμές και στήλες του δεξιού και αριστερού μέρους της παραπάνω εξίσωσης, λαμβάνουμε:

(γραμμή , στήλη): εξίσωση

$$(1 , 3): -s_5 \cdot c_4 = (-s_2 \cdot s_3 \cdot c_1 + c_1 \cdot c_2 \cdot c_3) \cdot r_{13} + (s_1 \cdot c_2 \cdot c_3 - s_1 \cdot s_2 \cdot s_3) \cdot r_{23} + (s_2 \cdot c_3 + s_3 \cdot c_2) \cdot r_{33}$$

$$(2 , 3): -s_4 \cdot s_5 = (-s_2 \cdot c_1 \cdot c_3 - s_3 \cdot c_1 \cdot c_2) \cdot r_{13} - (s_1 \cdot s_2 \cdot c_3 + s_1 \cdot s_3 \cdot c_2) \cdot r_{23} + (c_2 \cdot c_3 - s_2 \cdot s_3) \cdot r_{33}$$

$$(3 , 3): c_5 = s_1 \cdot r_{13} - c_1 \cdot r_{23}$$

$$(3 , 2): -s_5 \cdot s_6 = s_1 \cdot r_{12} - c_1 \cdot r_{22}$$

$$(3 , 1): s_5 \cdot c_6 = s_1 \cdot r_{11} - c_1 \cdot r_{21}$$

Με αυτόν τον τρόπο, προκύπτουν 5 νέες εξισώσεις.

◆ **Εύρεση και υπολογισμός εξίσωσης για την γωνία Θ_5 :**

Για την εύρεση της εξίσωσης για την γωνία Θ_5 , προφανώς θα εκμεταλευτούμε την εξίσωση από την (γραμμή , στήλη) = (3 , 3).

Ισχύει ότι:

$$c_5 = s_1 \cdot r_{13} - c_1 \cdot r_{23} \text{ και} \quad \text{επίσης} \quad \text{ότι}$$

$$s_5^2 + c_5^2 = 1 \Rightarrow s_5 = \pm \sqrt{(1 - c_5^2)} = \pm \sqrt{(1 - (s_1 \cdot r_{13} - c_1 \cdot r_{23})^2)}, \text{ οπότε:}$$

$$\Theta_{5,1} = \text{atan2}(\sqrt{(1 - (s_1 \cdot r_{13} - c_1 \cdot r_{23})^2)}, s_1 \cdot r_{13} - c_1 \cdot r_{23})$$

$$\Theta_{5,2} = \text{atan2}(-\sqrt{(1 - (s_1 \cdot r_{13} - c_1 \cdot r_{23})^2)}, s_1 \cdot r_{13} - c_1 \cdot r_{23})$$

Τελικά:

$$\Theta_5 = \Theta_{5,1} \text{ ή } \Theta_{5,2}$$

Όπως μπορούμε να παρατηρήσουμε η εξίσωση της γωνίας Θ_5 είναι συναρτήσει της Θ_1 και των στοιχείων r_{ij} του πίνακα περιστροφής 0R_6 .

◆ **Εύρεση και υπολογισμός εξίσωσης για την γωνία Θ_6 :**

Αντίστοιχα, για την εύρεση της εξίσωσης για την γωνία Θ_6 , θα βασιστούμε στις 2 τελευταίες εξισώσεις, δηλαδή (3 , 2) και (3 , 1).

$$\text{Συγκεκριμένα, από την (3 , 2): } -s_5 \cdot s_6 = s_1 \cdot r_{12} - c_1 \cdot r_{22} \Rightarrow s_6 = \frac{c_1 \cdot r_{22} - s_1 \cdot r_{12}}{s_5}$$

$$\text{Ενω, από την (3 , 1): } s_5 \cdot c_6 = s_1 \cdot r_{11} - c_1 \cdot r_{21} \Rightarrow c_6 = \frac{s_1 \cdot r_{11} - c_1 \cdot r_{21}}{s_5}$$

Τελικά:

$$\Theta_6 = \text{atan}2\left(\frac{c_1 \cdot r_{22} - s_1 \cdot r_{12}}{s_5}, \frac{s_1 \cdot r_{11} - c_1 \cdot r_{21}}{s_5}\right)$$

Εδώ, βλέπουμε ότι η Θ_6 εξαρτάται άμεσα από τις 2 προηγούμενες γωνίες που βρήκαμε, δηλαδή τις Θ_5 και Θ_1 , καθώς και από στοιχεία r_{ij} του πίνακα περιστροφής 0R_6 .

⚠ ΠΡΟΣΟΧΗ: Ιδιαίτερη σημασία πρέπει να δώσουμε στο γεγονός ότι η Θ_6 δεν μπορεί να οριστεί εάν η γωνία Θ_5 λάβει τις τιμές 0 ή π , καθώς θα μηδενιστούν οι παρανομαστές της εξίσωσης. Έτσι, δεν μπορεί να γίνει κάποια κίνηση για τον βραχίονα, εφόσον η Θ_5 πάρει τις 2 παραπάνω τιμές.

➤ ΥΠΟΛΟΓΙΣΜΟΣ ΤΗΣ ΕΞΙΣΩΣΗΣ ΓΙΑ ΤΗΝ ΓΩΝΙΑ Θ_{234} :

Η συγκεκριμένη γωνία Θ_{234} αποτελείται από το άθροισμα των 3 επιμέρους γωνιών Θ_2 , Θ_3 και Θ_4 (τις οποίες θα βρούμε στη πορεία), δηλαδή:

$$\Theta_{234} = \Theta_2 + \Theta_3 + \Theta_4$$

Από τον πίνακα 2.5 που ορίσαμε παραπάνω, θα λύσουμε τις εξισώσεις των γραμμών (8) και (9). Οπότε, έχουμε:

$$a_y = -s_5 \cdot s_1 \cdot c_{234} - c_1 \cdot c_5 \Rightarrow c_{234} = -\frac{a_y + c_1 \cdot c_5}{s_5 \cdot s_1} \quad \text{και} \quad a_z = -s_5 \cdot s_{234} \Rightarrow s_{234} = -\frac{a_z}{s_5}$$

Τελικά:

$$\Theta_{234} = \text{atan}2\left(-\frac{a_z}{s_5}, -\frac{a_y + c_1 \cdot c_5}{s_5 \cdot s_1}\right)$$

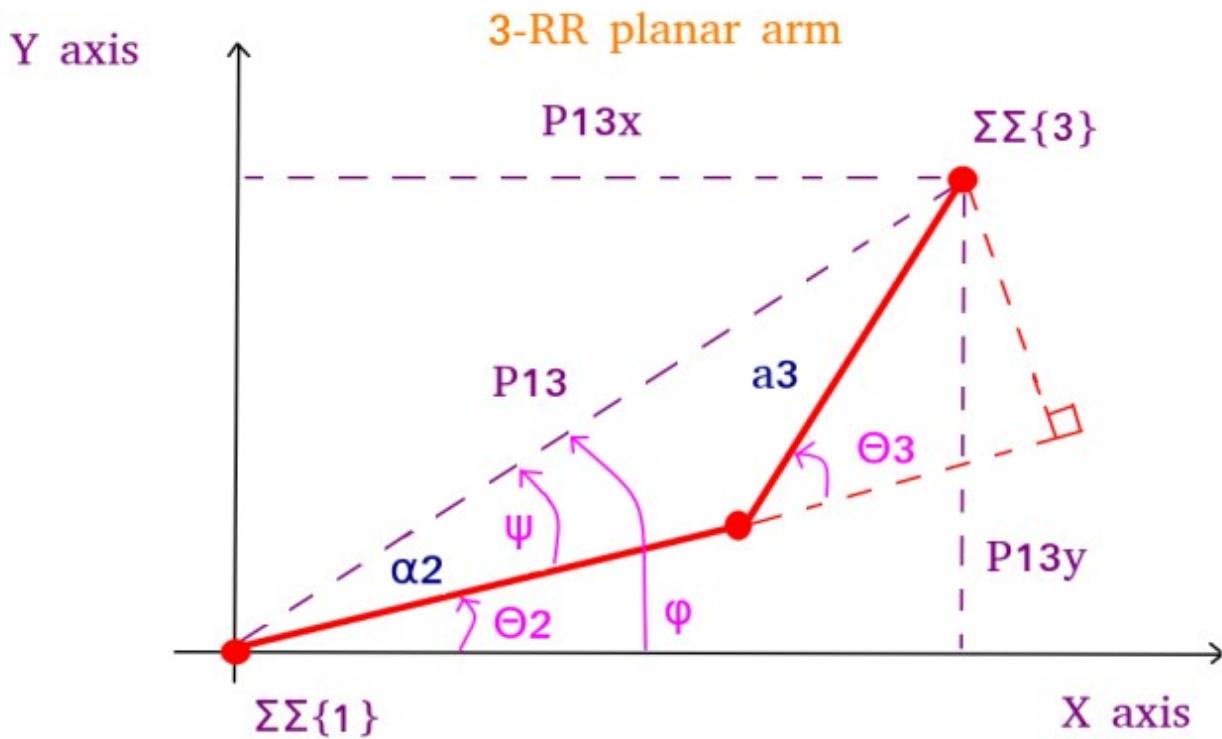
Όπως φαίνεται, η εξίσωση για τη γωνία Θ_{234} εξαρτάται από τις γωνίες Θ_5 και Θ_1 , όπως και από τις παραμέτρους a_z και a_y , δηλαδή στοιχεία του πίνακα περιστροφής 0R_6 .

⚠ ΠΡΟΣΟΧΗ: Όπως με την εξίσωση της γωνίας Θ_6 , το ίδιο ισχύει και εδώ.

➤ ΥΠΟΛΟΓΙΣΜΟΙ ΤΩΝ ΕΞΙΣΩΣΕΩΝ ΓΙΑ ΤΙΣ ΓΩΝΙΕΣ Θ_2 , Θ_3 , Θ_4 :

Οι εναπομείναντες γωνίες Θ_2 , Θ_3 και Θ_4 μπορούν να βρεθούν εύκολα. Θεωρούμε ότι οι αρθρώσεις των γωνιών Θ_2 και Θ_3 λειτουργούν ως ένας επίπεδος βραχίονας 3-RR (Revolute-Revolute-Revolute), δηλαδή βραχίονας όπου και οι 3 αρθρώσεις είναι περιστροφικές και διέρχονται από το ίδιο σημείο. Στην ουσία θα βρούμε τις εξισώσεις αυτών των δύο γωνιών, θεωρώντας έναν βραχίονα με τρεις μόνο αρθρώσεις αδιαφορώντας για τα υπόλοιπα μέρη του πραγματικού βραχιονα. Το μόνο που χρειαζόμαστε είναι να γνωρίζουμε το αρχικό και τελικό στοιχείο δράσης του.

- ◆ Εύρεση και υπολογισμός εξίσωσης για την γωνία Θ_3 :



Εικόνα 2.9: Γεωμετρική αναπαράσταση ενός του 3-RR επίπεδου βραχίονα.

$\Sigma\Sigma\{3\}$: Σύστημα συντεταγμένων 3

$\Sigma\Sigma\{1\}$: Σύστημα συντεταγμένων 1

$\psi, \varphi, \theta_2, \theta_3$: Γωνίες των συνδέσμων 2 και 3

$P_{13}(x, y)$: Θέση άρθρωσης της γωνίας Θ_3 σε σχέση με την άρθρωση της γωνίας Θ_1 .

a_2, a_3 : Μήκη συνδέσμων 2 και 3

Παρακάτω, προσδιορίζεται η προαναφερθείσα θέση 1P_3 :

$${}^1P_3 = {}^1P_4 + {}^1T_4 \cdot \begin{bmatrix} 0 \\ -d_4 \\ 0 \\ 1 \end{bmatrix}$$

Επίσης, πρέπει να ορίσουμε και να βρούμε τον πίνακα 1T_4 , καθώς και το διάνυσμα 1P_4 :

$${}^0T_6 = {}^0T_1 \cdot {}^1T_2 \cdot {}^2T_3 \cdot {}^3T_4 \cdot {}^4T_5 \cdot {}^5T_6 \Rightarrow {}^0T_6 = {}^0T_1 \cdot {}^1T_4 \cdot {}^4T_6 \Rightarrow {}^1T_4 = ({}^0T_1)^{-1} \cdot {}^0T_6 \cdot ({}^4T_6)^{-1}$$

Το διάνυσμα 1P_4 είναι η τελευταία στήλη του πίνακα 1T_4 . Παρακάτω, υπολογίζεται η Θ_3 :

$$\begin{aligned} (a_2 + a_3 \cdot \cos(\theta_3))^2 + (a_3 \cdot \sin(\theta_3))^2 &= ({}^1P_{3x})^2 + ({}^1P_{3y})^2 \Rightarrow \\ \Rightarrow a_2^2 + 2 \cdot a_2 \cdot a_3 \cdot \cos(\theta_3) + a_3^2 \cdot \cos(\theta_3)^2 + a_3^2 \cdot \sin(\theta_3)^2 &= ({}^1P_{3x})^2 + ({}^1P_{3y})^2 \Rightarrow \\ \Rightarrow a_2^2 + a_3^2 + 2 \cdot a_2 \cdot a_3 \cdot \cos(\theta_3)^2 &= ({}^1P_{3x})^2 + ({}^1P_{3y})^2 \Rightarrow \\ \Rightarrow \cos(\theta_3) &= \frac{({}^1P_{3x})^2 + ({}^1P_{3y})^2 - a_2^2 - a_3^2}{2 \cdot a_2 \cdot a_3} \Rightarrow \\ \Rightarrow \theta_3 &= \pm \arccos\left(\frac{({}^1P_{3x})^2 + ({}^1P_{3y})^2 - a_2^2 - a_3^2}{2 \cdot a_2 \cdot a_3}\right) \end{aligned}$$

Δηλαδή:

$$\begin{aligned} \theta_{3,1} &= \arccos\left(\frac{({}^1P_{3x})^2 + ({}^1P_{3y})^2 - a_2^2 - a_3^2}{2 \cdot a_2 \cdot a_3}\right) \\ \theta_{3,2} &= -\arccos\left(\frac{({}^1P_{3x})^2 + ({}^1P_{3y})^2 - a_2^2 - a_3^2}{2 \cdot a_2 \cdot a_3}\right) \end{aligned}$$

Τελικά:

$$\theta_3 = \theta_{3,1} \text{ ή } \theta_{3,2}$$

◆ Εύρεση και υπολογισμός εξίσωσης για την γωνία Θ_2 :

Από το παραπάνω σχήμα, μπορούμε να δούμε ότι:

$$\varphi = \arctan2({}^1P_{3y}, {}^1P_{3x}) \quad \text{και επίσης:}$$

$$\psi_1 = \arctan2(a_3 \cdot \sin(\theta_{3,1}), a_2 + a_3 \cdot \cos(\theta_{3,1})) = \psi \quad \text{και}$$

$$\psi_2 = \arctan2(a_3 \cdot \sin(\theta_{2,2}), a_2 + a_3 \cdot \cos(\theta_{2,2})) = \arctan2(-a_3 \cdot \sin(\theta_{2,1}), a_2 + a_3 \cdot \cos(\theta_{2,1})) = -\psi$$

Δηλαδή:

$$\theta_{2,1} = \varphi - \psi, \quad \theta_{2,2} = \varphi + \psi \quad \text{Τελικά: } \theta_2 = \theta_{2,1} \text{ ή } \theta_{2,2}$$

◆ **Εύρεση και υπολογισμός εξίσωσης για την γωνία Θ₄:**

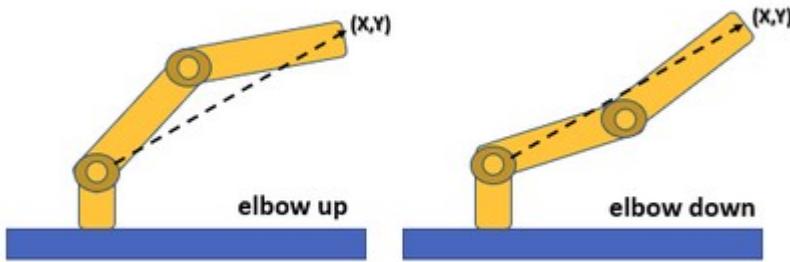
Γνωρίζοντας πλέον τις γωνίες θ₂₃₄, θ₂ και θ₃, μπορούμε να βρούμε την θ₄ ως εξής:

$$\theta_{234} = \theta_2 + \theta_3 + \theta_4 \Rightarrow \theta_4 = \theta_{234} - \theta_2 - \theta_3$$

Τελικά:

$$\theta_4 = \theta_{234} - \theta_2 - \theta_3$$

Ανάλογα τις λύσεις που θα διαλέξουμε για τις αρθρώσεις των γωνιών θ₂ και θ₃, ο βραχίονας μπορεί να δεχτεί δύο διαφορετικές διαμορφώσεις, τις λεγόμενες “elbow up” και “elbow down”. Παρακάτω, βλέπουμε σχηματικά τις δύο διαφορετικές διαμορφώσεις.



Εικόνα 2.10: Σχηματική απεικόνιση των διαμορφώσεων “elbow up” και “elbow down”.

⚠ ΠΡΟΣΟΧΗ: Δεν υπάρχουν λύσεις όταν η απόσταση μεταξύ 2_{ης} και 4_{ης} άρθρωσης υπερβαίνει το άθροισμα $|(a_2+a_3)|$ ή είναι μικρότερη από τη διαφορά $|(a_2-a_3)|$.

Μετά το πέρας της περιγραφής και υπολογισμού της ευθείας και αντίστροφης κινηματικής του βραχίονα, δίνοντας μία θέση στο τρισδιάστατο χώρο (X, Y, Z) και ένα προσανατολισμό, ο UR3 έχοντας υπολογίσει αυτές τις 6 γωνίες για τις αντίστοιχες αρθρώσεις του, τώρα με δεδομένες αυτές είναι έτοιμος να περιστρέψει τους έξι κινητήρες του, έτσι ώστε να κατευθύνει το τελικό στοιχείο δράσης, δηλαδή την αρπάγη στη προκαθορισμένη θέση με τον σωστό προσανατολισμό.

2.6 ΘΕΣΗ ΕΚΚΙΝΗΣΗΣ ΒΡΑΧΙΟΝΑ

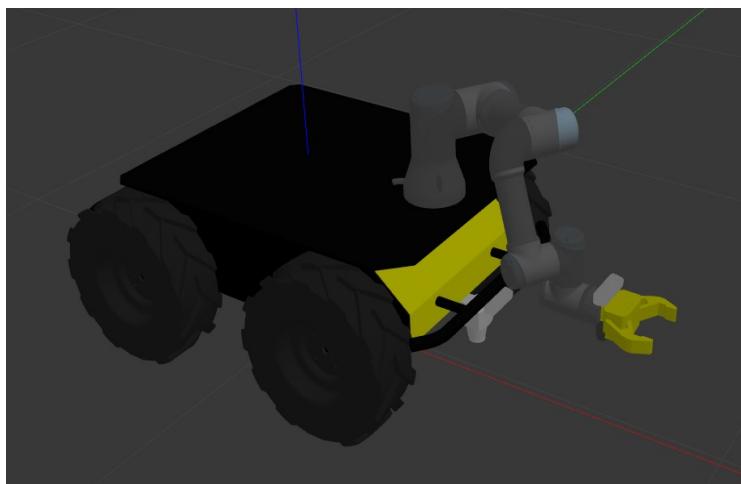
Πριν ξεκινήσει οποιαδήποτε λειτουργία ο βραχίονας ή το τροχοφόρο όχημα, πρέπει ο βραχίονας να λάβει μία αρχική θέση ή θέση εκκίνησης (Home position). Κάθε άρθρωση έχει τη δυνατότητα να στραφεί κατά 180 μοίρες. Η θέση εκκίνησης θέτει τις γωνίες Θ_1 έως Θ_6 των αρθρώσεων 1 έως 6 αντίστοιχα ως εξής:

$$\Theta_1 = -\frac{\pi}{2}, \quad \Theta_2 = -\frac{\pi}{2}, \quad \Theta_3 = \frac{\pi}{2}, \quad \Theta_4 = \frac{\pi}{2}, \quad \Theta_5 = 0, \quad \Theta_6 = -\frac{\pi}{2}$$

Οι αρθρώσεις, λοιπόν, ξεκινούν από τυχαίες γωνίες (γωνίες που βρίσκονται εκείνη τη στιγμή) και η εντολή JointTrajectoryPoint της Python δημιουργεί μία τροχιά με σκοπό να καταλήξει στις προκαθορισμένες γωνίες Θ_1 έως Θ_6 . Η παραπάνω εντολή λαμβάνει μερικά ορίσματα. Το 1ο όρισμα είναι οι τελικές γωνίες Θ_1 έως Θ_6 που δίνεται ως ένας πίνακας ενώ το 2o όρισμα είναι ο χρόνος της συνολικής κίνησης, δηλαδή από την θέση με τις τυχαίες γωνίες μέχρι τις τελικές γωνίες Θ_1 έως Θ_6 .

Στην συγκεκριμένη περίπτωση, δηλαδή η κατάληξη του βραχίονα στη θέση εκκίνησης παίρνει συνολικό χρόνο ίσο με πέντε (5) δευτερόλεπτα.

Η θέση 'Home position' είναι πολλή σημαντική με σκοπό να μην εμποδίζει το ρομποτικό όχημα Husky κατά την εργασία του (μετακίνηση ή περιστροφή), δηλαδή να μην εκτείνεται έξω από αυτό με κίνδυνο να συγκρουστεί με εμπόδιο του περιβάλλοντος χώρου, όπως βλέπουμε στην εικόνα 2.11 και 2.12 παρακάτω.



Εικόνα 2.11: Ο βραχίονας προεξέχει του τροχοφόρου οχήματος με κίνδυνο να συγκρουστεί με άλλο εμπόδιο του δρόμου που τυχόν βρίσκεται μπροστά.



Εικόνα 2.12: Ο βραχίονας σε θέση εκκίνησης χωρίς να εμποδίζει.

2.7 ΚΙΝΗΣΗ ΒΡΑΧΙΟΝΑ

Το υποκέφαλαιο αυτό θα μελετήσει τη κίνηση του βραχίονα UR3 στον χώρο του Gazebo. Όπως είναι γνωστό, το γραφικό περιβάλλον Gazebo αποτελείται από συντεταγμένες, όπου η αρχή των αξόνων είναι το σημείο (0 , 0 , 0). Όταν το Husky κινείται στο χώρο, οι συντεταγμένες αυτές αλλάζουν τόσο για το Husky όσο και για τον βραχίονα.

Όμως, η θέση της τελευταίας άρθρωσης του βραχίονα στο τρισδιάστατο χώρο σχετίζεται με τη βάση του βραχίονα και όχι με την αρχή των αξόνων επιθυμούμε.

Όλες οι θέσεις, όπως του Husky, του UR3, των τραπέζια και αντικειμένων πρέπει να σχετίζονται με την αρχή των αξόνων και τίποτα άλλο.

Μέχρι στιγμής, από την ευθεία και αντίστροφη κινηματική του βραχίονα, η θέση της τελευταίας άρθρωσης του βραχίονα, όπως είπαμε, σχετίζεται με τη βάση του. Όπως και τα υπόλοιπα “αντικείμενα”, πρέπει να σχετίσουμε αυτή την άρθρωση του UR3 με την αρχή των αξόνων του 3D χώρου. Πρέπει, με λίγα λόγια να εφαρμόσουμε ένα μετασχηματισμό σε αυτή την άρθρωση. Παρακάτω, εξηγούμε αναλυτικά πως θα γίνει αυτό.

Υπενθυμίζουμε ότι ο βραχίονας έχει ως τελικό στοιχείο δράσης (ΤΣΔ) μία αρπάγη, η οποία έχει συγκεκριμένο μήκος, πλάτος και ύψος.

→ **ΣΥΝΤΕΤΑΓΜΕΝΗ Z**

Το τελικό ύψος, δηλαδή η συντεταγμένη Z που πρέπει να καταλήξει η αρπάγη σε σχέση με την αρχή των αξόνων (το έδαφος), υπολογίζεται ως εξής:

$$Z = \text{positionZ} - \text{groundToBase} - \text{gripperHeight} - \text{topPlateLinkHeight}$$

- **positionZ:**

Αποτελεί τη θέση του άξονα Z, δηλαδή το ύψος από τη βάση του UR3 έως την τελευταία άρθρωση με γωνία Θ_6 . [Βλέπε εικόνα 2.13]

- **groundToBase:**

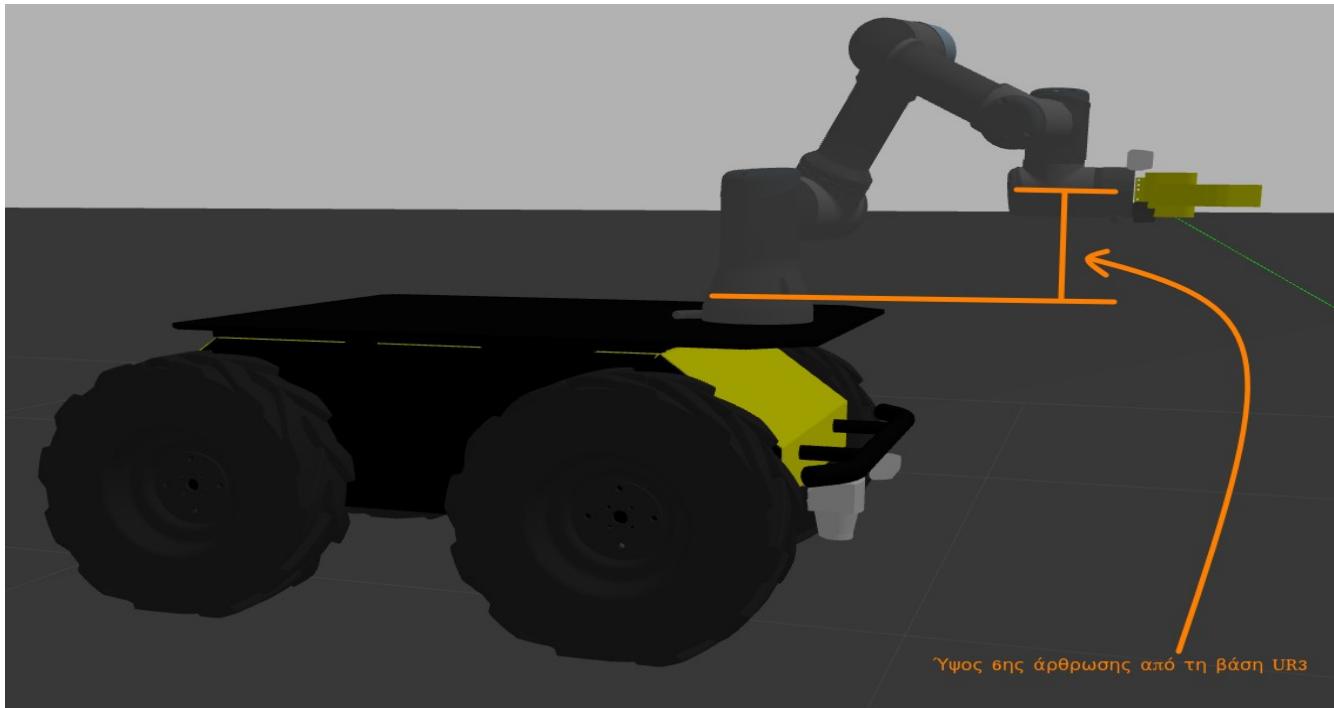
Αποτελεί το ύψος της βάσης του βραχίονα από το έδαφος και έχει σταθερή τιμή (ίση με 0.25 μέτρα (25 εκατοστά), διότι ο βραχίονας είναι προσκολλημένος στο Husky. [Βλέπε εικόνα 2.14]

- **gripperHeight:**

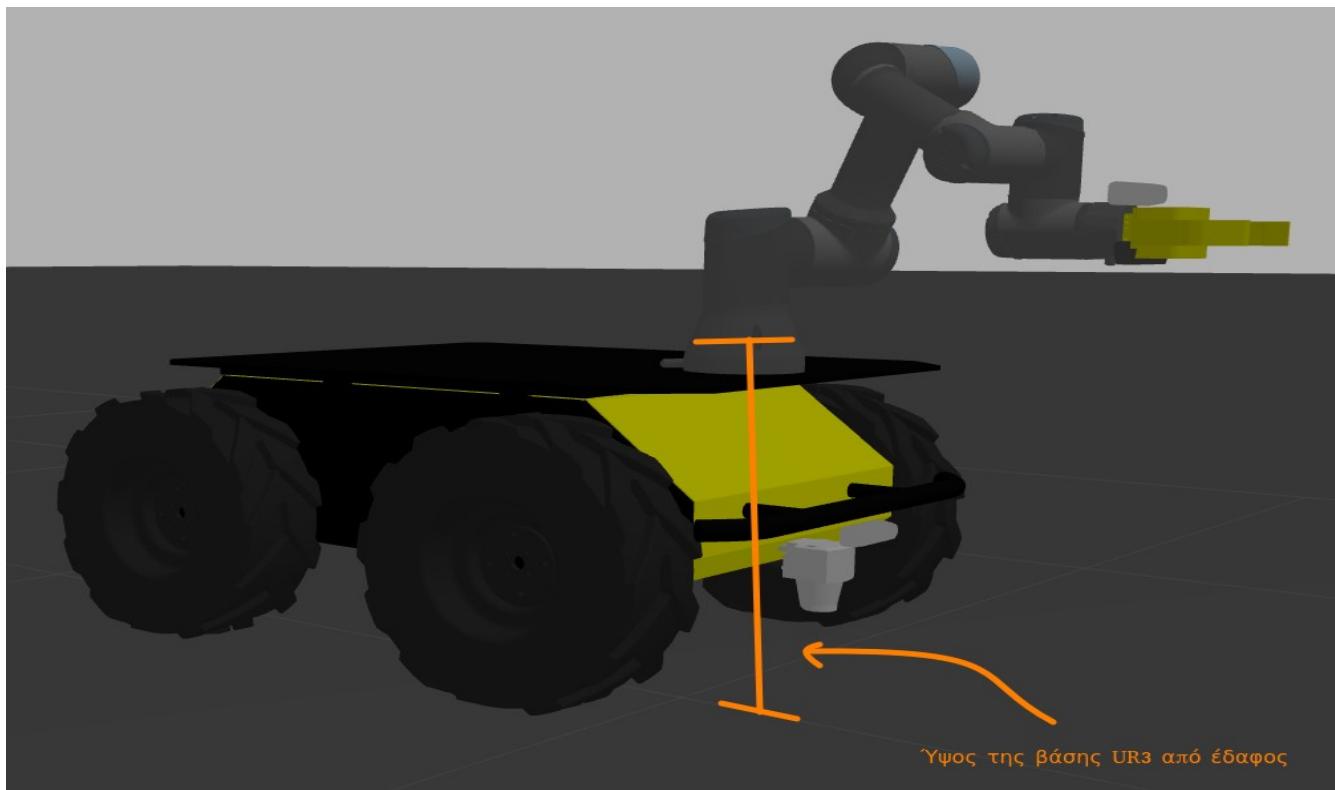
Αποτελεί το ύψος της αρπάγης, όμως η τιμή της είναι ίσης με μηδέν (0) μέτρα, διότι πάντα θα βρίσκεται παράλληλα με το ύψος της τελευταίας άρθρωσης, οπότε δεν χρειάζεται να προσθέσουμε περαιτέρω ύψος σε αυτόν τον παράγοντα. [Βλέπε εικόνα 2.15]

- **topPlateLinkHeight:**

Αποτελεί το ύψος της βάσης του βραχίονα από τη βάση του Husky με σταθερή τιμή (ίση με 0.1225 μέτρα. [Βλέπε εικόνα 2.16]



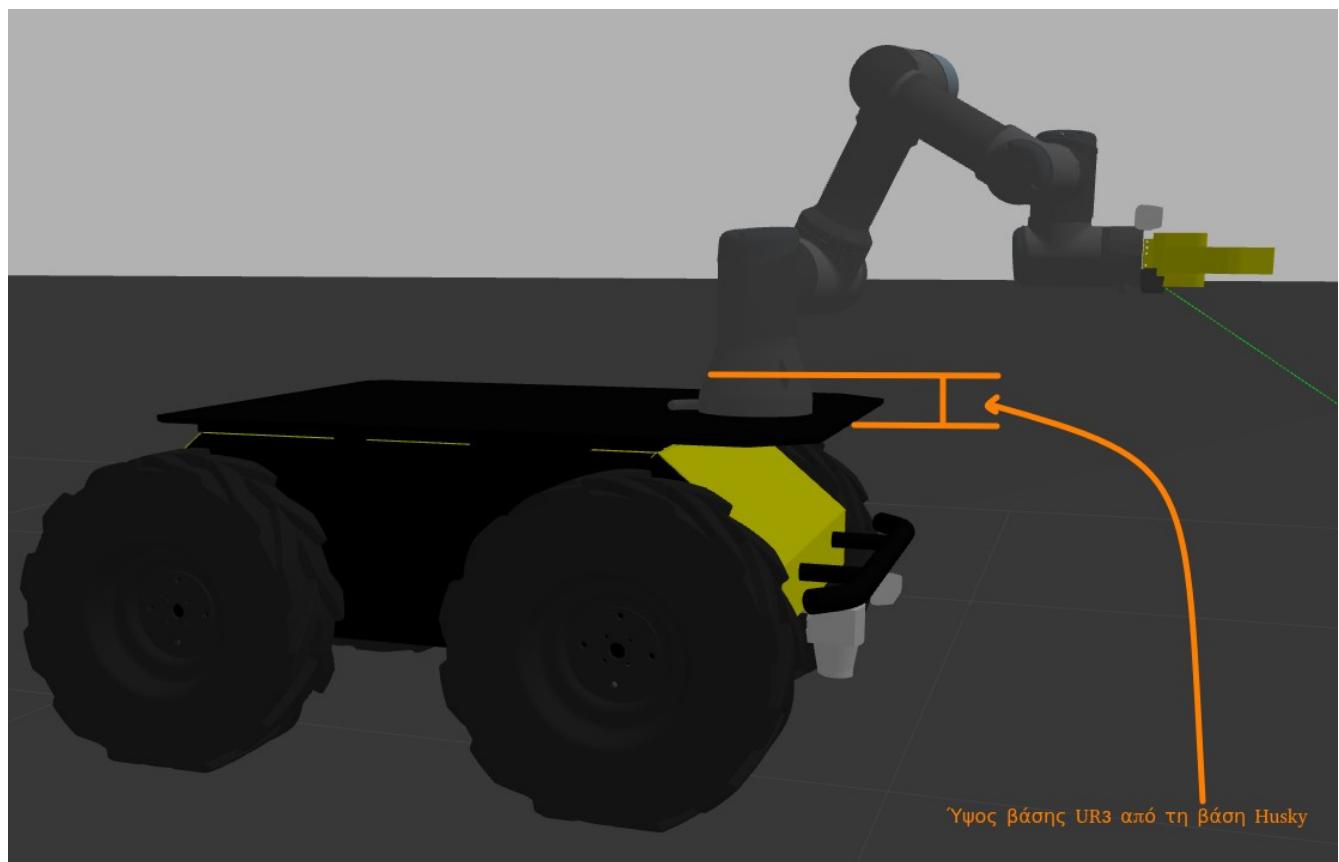
Εικόνα 2.13: Ύψος θης άρθρωσης από βάση του βραχίονα.



Εικόνα 2.14: Ύψος βάσης βραχίονα από έδαφος.



Εικόνα 2.15: Ύψος αρπάγης από 6η άρθρωση βραχίονα (μηδενική διαφορά).



Εικόνα 2.16: Ύψος βάσης UR3 από βάση του οχήματος Husky.

→ **ΣΥΝΤΕΤΑΓΜΕΝΗ Y**

Καταρχάς, όπως έχουμε πει και στο κεφάλαιο 1.5, σημαντικό ρόλο παίζει ο προσανατολισμός του Husky στο χώρο, διότι το γεγονός αυτό επηρεάζει αντίστοιχα τον προσανατολισμό του βραχίονα UR3.

Η συντεταγμένη Y που πρέπει να καταλήξει η αρπάγη σε σχέση με την αρχή των αξόνων, υπολογίζεται ως εξής:

- ◆ Εάν το ρομπότ Husky προσεγγίσει ένα τραπέζι από πάνω ή από κάτω:

$$Y = -\text{positionY} + \text{groundToBaseYcoordinate} + \text{gripperYcoordinate}$$

- ◆ Εάν το ρομπότ Husky προσεγγίσει ένα τραπέζι από δεξιά ή από αριστερά:

$$Y = -\text{positionY} + \text{groundToBaseXcoordinate} + \text{gripperYcoordinate}$$

- **positionY:**

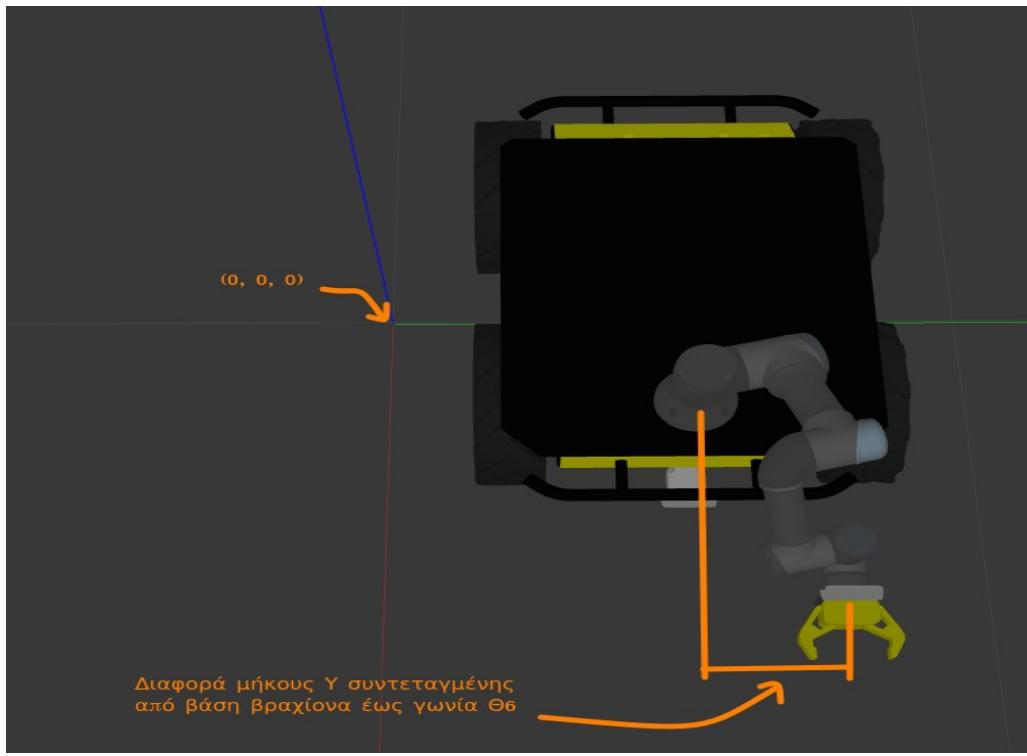
Αποτελεί τη θέση του άξονα Y, δηλαδή τη διαφορά μήκους μεταξύ συντεταγμένης Y από τη βάση του UR3 έως την τελευταία άρθρωση με γωνία Θ_6 . [Βλέπε εικόνα 2.17]

- **groundToBaseYcoordinate/groundToBaseXcoordinate:**

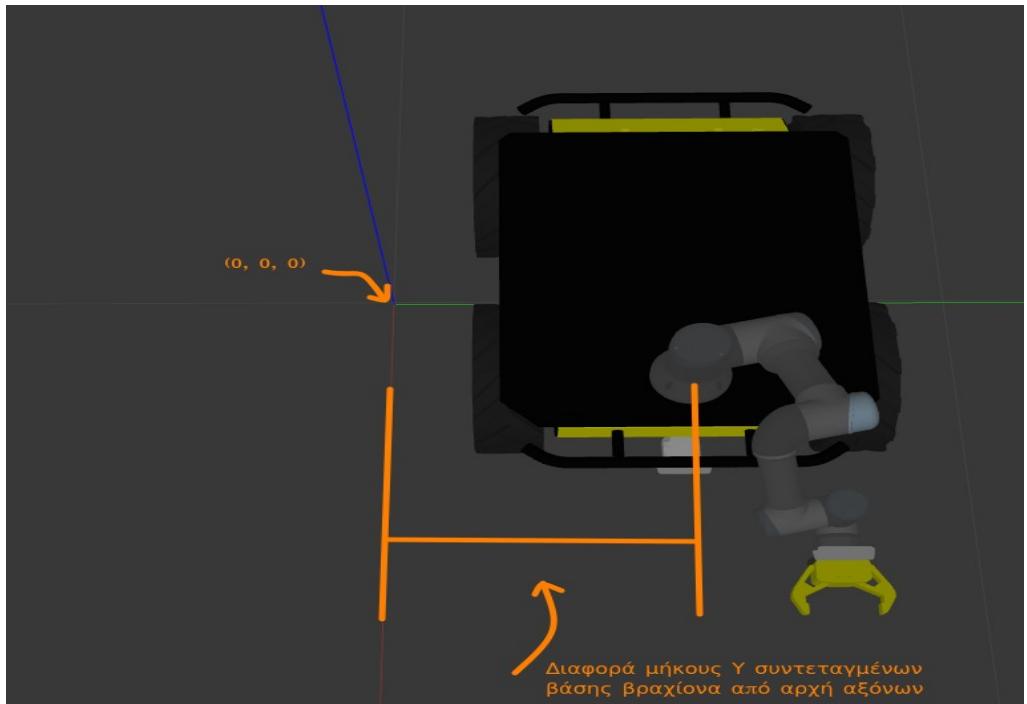
Αποτελεί τη θέση του άξονα Y/X (ανάλογα την παραπάνω περίπτωση), δηλαδή τη διαφορά μήκους μεταξύ συντεταγμένης Y/X από την αρχή των αξόνων έως τη βάση του βραχίονα. [Βλέπε εικόνα 2.18]

- **gripperYcoordinate:**

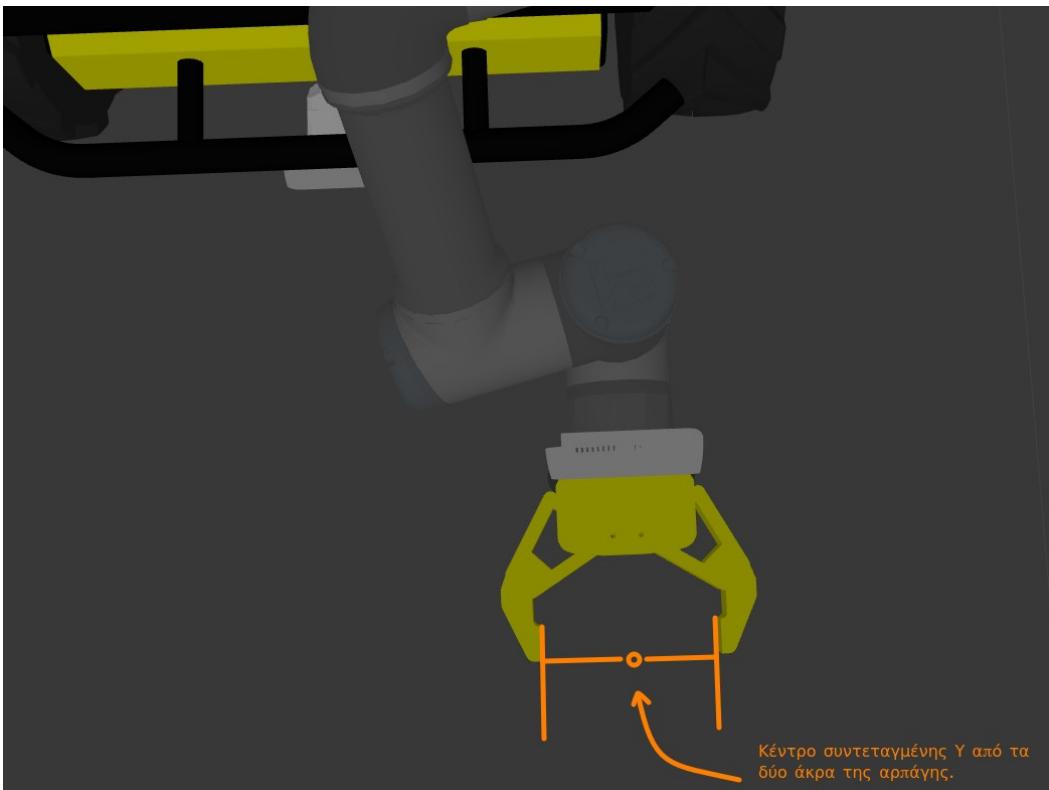
Αποτελεί το κέντρο των συντεταγμένων Y από τα δύο άκρα της αρπάγης και έχει τιμή ίση με 0.13 μέτρα (13 εκατοστά). [Βλέπε εικόνα 2.19]



Εικόνα 2.17: Διαφορά μήκους μεταξύ συντεταγμένης Υ από τη βάση του UR3 έως την τελευταία άρθρωση με γωνία Θ_6 .



Εικόνα 2.18: Διαφορά μήκους μεταξύ συντεταγμένης Υ από την αρχή των αξόνων έως τη βάση του βραχίονα.



Εικόνα 2.19: Κέντρο των συντεταγμένων Y από τα δύο άκρα της αρπάγης.

➔ ΣΥΝΤΕΤΑΓΜΕΝΗ X

Η συντεταγμένη X που πρέπει να καταλήξει η αρπάγη σε σχέση με την αρχή των αξόνων, υπολογίζεται ως εξής:

- ◆ Εάν το ρομπότ Husky προσεγγίσει ένα τραπέζι από πάνω ή από κάτω:

$X = positionX - groundToBaseXcoordinate$

- ◆ Εάν το ρομπότ Husky προσεγγίσει ένα τραπέζι από δεξιά ή από αριστερά:

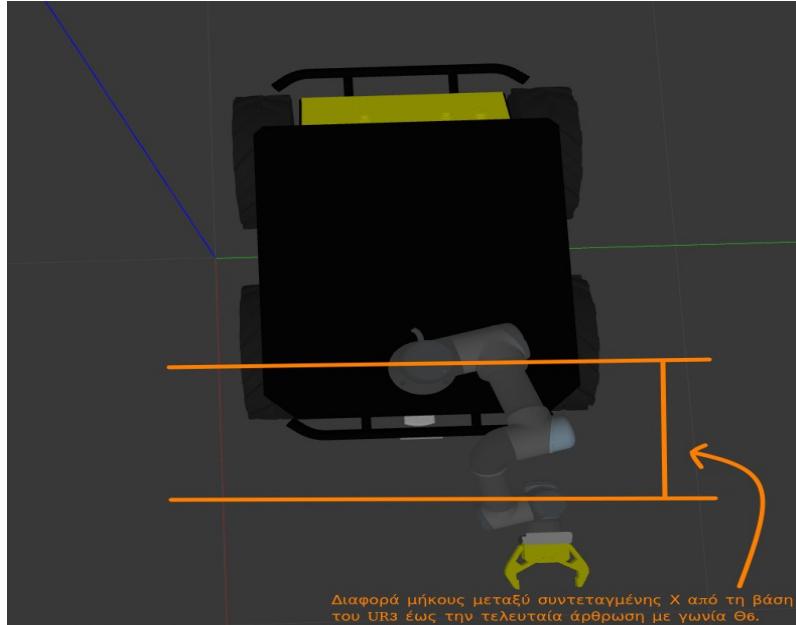
$X = positionX - groundToBaseYcoordinate$

- **positionX:**

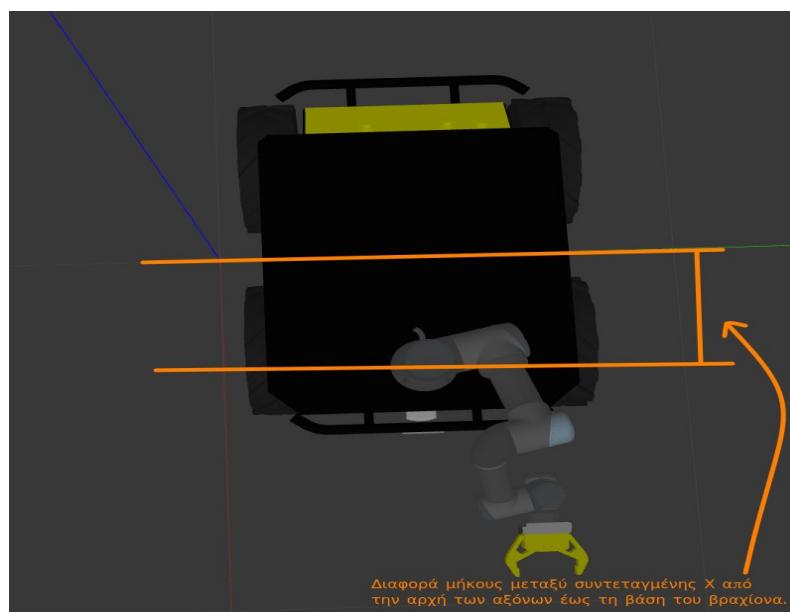
Αποτελεί τη θέση του άξονα X, δηλαδή τη διαφορά μήκους μεταξύ συντεταγμένης X από τη βάση του UR3 έως την τελευταία άρθρωση με γωνία Θ_6 . [Βλέπε εικόνα 2.20]

- **groundToBaseXcoordinate/groundToBaseYcoordinate:**

Αποτελεί τη θέση του áξονα X/Y (ανάλογα την παραπάνω περίπτωση), δηλαδή τη διαφορά μήκους μεταξύ συντεταγμένης X/Y από την αρχή των αξόνων έως τη βάση του βραχίονα. [Βλέπε εικόνα 2.21]



Εικόνα 2.20: Διαφορά μήκους μεταξύ συντεταγμένης X από τη βάση του UR3 έως την τελευταία άρθρωση με γωνία Θ_6 .



Εικόνα 2.21: Διαφορά μήκους μεταξύ συντεταγμένης X από την αρχή των αξόνων έως τη βάση του βραχίονα.

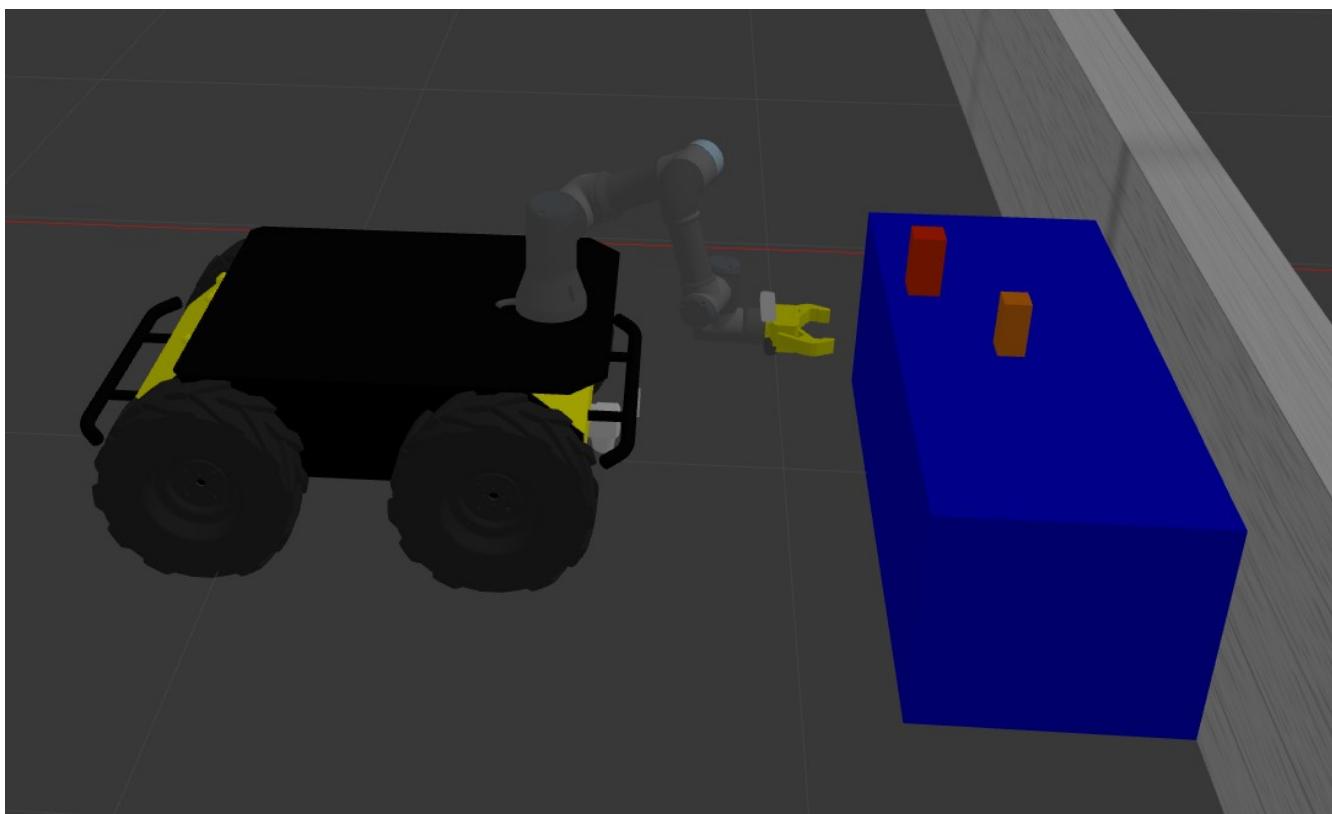
2.8 ΣΑΡΩΣΗ ΑΝΤΙΚΕΙΜΕΝΟΥ

Το υποκεφάλαιο αυτό αποτελεί τη πιό κρίσιμη και σημαντική λειτουργία του βραχίονα, τη σάρωση ενός αντικειμένου. Η λειτουργία των δύο ρομπότ ξεκινάει ως εξής:

Το Husky κινούμενο προς τη κατεύθυνση του τραπεζιού, όπως έχει διατυπωθεί στο κεφάλαιο 1, σταματάει σε ασφαλή απόσταση 0.58 μέτρων από το τραπέζι στο οποίο περιέχονται τα αντικείμενα, τα οποία πρόκειται να ληφθούν από τον βραχίονα.

Ενεργοποιείται ο βραχίονας, προετοιμασμένος για τη σάρωση ενός αντικειμένου, παίρνοντας μία αρχική θέση στο χώρο, η οποία περιγράφεται παρακάτω.

Το ύψος του ΤΣΔ του βραχίονα ορίζεται στα 35 εκατοστά από το έδαφος. Επιπλέον, ο βραχίονας εκτείνεται κατά 85 εκατοστά από το κέντρο του Husky. Η παραπάνω περιγραφή μπορεί να φανεί στην εικόνα 2.22 παρακάτω.



Εικόνα 2.22: Αρχική θέση βραχίονα πριν την εκτέλεση σάρωσης για αντικείμενα.

Να υπενθυμίσουμε στο σημείο αυτό ότι, όπως έχουμε αναφέρει στο κεφάλαιο 1, υπάρχει ένα παράθυρο μέσα από το οποίο μπορούμε να δούμε σε πραγματικό χρόνο τη διαδικασία της σάρωσης και όχι μόνο.

Στη συνέχεια λοιπόν, αν κανένα αντικείμενο δεν είναι ορατό στο παράθυρο (frame), τότε ο βραχίονας πρέπει να εκτελέσει μία αυτόματη σάρωση (zig-zag), έως ότου βρει κάποιο αντικείμενο, αλλιώς αν δεν το βρει, σημαίνει ότι δεν υπάρχουν πλέον αντικείμενα προς αναζήτηση. Ας ονομάσουμε αυτή τη διαδικασία, auto scanning.

Αντιθέτως, αν βρεθεί κάποιο αντικείμενο ή έστω ένα μέρος αυτού του αντικειμένου στο παράθυρο, όσο ο βραχίονας βρίσκεται στην αρχική του θέση, τότε πραγματοποιείται μία κίνηση του βραχίονα προς το κέντρο αυτού του αντικειμένου, με τις πληροφορίες, δηλαδή τις συντεταγμένες που μας δίνει η κάμερα κάθε στιγμή.

Ας ονομάσουμε αυτή τη διαδικασία, manual scanning. Παρακάτω παρατίθενται αυτές οι δύο λειτουργίες αναλυτικότερα.

■ **Έλεγχος λειτουργίας auto scanning:**

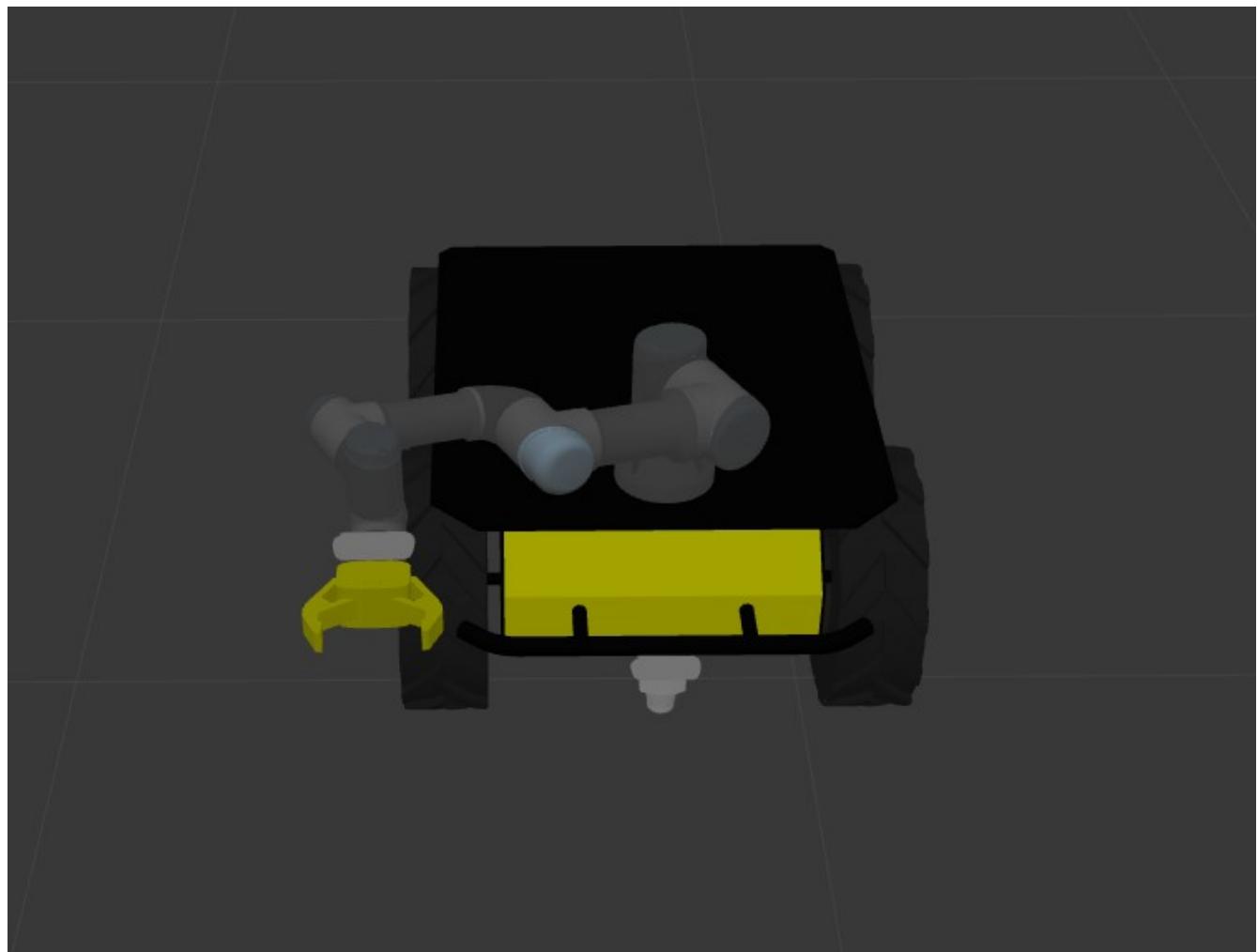
Από την αρχική θέση που βρίσκεται ο βραχίονας, εκτελείται μία κίνηση προς τα δεξιά όσο δεν έχει βρεθεί κάποιο αντικείμενο στο frame, έως ότου εκταθεί πλήρως (βλέπε εικόνα 2.23), δηλαδή η γωνία Θ3 μηδενιστεί ή έως ότου βρεθεί κάποιο αντικείμενο στο παράθυρο, οπότε και σταματά η κίνηση. Αλλιώς αν εκταθεί πλήρως ο βραχίονας, τότε κινείται προς την αντίθετη κατεύθυνση μέχρι να εκταθεί πλήρως ή να βρεθεί κάποιο αντικείμενο στο δρόμο του. Μέχρι στιγμής, η κάμερα τοποθετημένη πάνω στον βραχίονα έχει σαρώσει μία συγκεκριμένη περιοχή, εκτελώντας μία κίνηση τύπου zig-zag.

Αυτή τη χρονική στιγμή, ο βραχίονας έχει εκταθεί πλήρως και προς τις 2 κατευθύνσεις αντίστοιχα, μία προς τα δεξιά και μία προς τα αριστερά (zig-zag).

Στη συνέχεια, υπάρχει περίπτωση το αντικείμενο να βρίσκεται σε ένα υψηλότερο σημείο από το ύψος του ΤΣΔ, οπότε ο βραχίονας θα υψωθεί κατά 0.16 εκατοστά.

Η προηγούμενη διαδικασία zig-zag θα συνεχιστεί (ΔΕΞΙΑ-ΑΡΙΣΤΕΡΑ-ΠΑΝΩ) κατ'επανάληψη άλλη μία φορά, αν το αντικείμενο δεν είναι ακόμα ορατό στο frame της κάμερας.

Αν στο τέλος της διαδικασίας δεν βρεθεί κάποιο αντικείμενο στο frame, τότε γίνεται αντιληπτό ότι δεν υπάρχουν αντικείμενα προς αναζήτηση και τότε τα δύο ρομπότ συνεχίζουν προς εξερεύνηση και πιθανή λήψη των υπόλοιπων αντικειμένων στο χώρο.



Εικόνα 2.23: Πλήρης έκταση βραχίονα προς τα αριστερά (όπως φαίνεται στην εικόνα) πρτος σάρωση αντικείμενου.

Από την άλλη, αν βρεθεί κάποιο αντικείμενο, τότε μπαίνουμε στη διαδικασία που αναφέρθηκε ως manual scanning, η οποία εξηγείται παρακάτω.

■ **Έλεγχος λειτουργίας manual scanning:**

Πλέον ένα μέρος ή ολόκληρο το αντικείμενο φαίνεται στο frame της κάμερας.

Ο βραχίονας βρίσκεται είτε στην αρχική θέση, η οποία ορίστηκε προηγουμένως, είτε συνεχίζει να κινείτε από τη στιγμή που έγινε ορατό το αντικείμενο στο frame από τη διαδικασία auto scanning.

Σκοπός αυτής της λειτουργίας είναι να έρθει το αντικείμενο στο κέντρο του frame (άσπρος κύκλος) της κάμερας ενδεχομένως με μία μικρή αμελητέα απόκλιση.

Θα χρειαστούν οι συντεταγμένες της πάνω αριστερής και κάτω δεξιάς ακμής του περιγράμματος του αντικειμένου, καθώς και οι συντεταγμένες του κέντρου του frame, τα οποία είναι γνωστά. Όλοι οι υπολογισμοί για την διαδικασία να έρθει το αντικείμενο στο κέντρο του frame (άσπρο κυκλάκι) περιγράφονται αναλυτικά παρακάτω, τόσο με λόγια όσο και με την εικόνα 2.24.

Αρχικά, παίρνουμε την απόλυτη τιμή της διαφοράς μεταξύ της οριζόντιας συντεταγμένης X_c και αριστερής ακμής X_1 .

Επιπλέον, παίρνουμε την απόλυτη τιμή της διαφοράς μεταξύ της οριζόντιας συντεταγμένης X_c και δεξιάς ακμής X_2 .

Ύστερα, κάνουμε σύγκριση μεταξύ αυτών των δύο τιμών, δηλαδή $d_{1x} = |X_c - X_1|$ και $d_{2x} = |X_c - X_2|$.

Αν $d_{1x} < d_{2x}$, τότε γίνεται αντιληπτό ότι το κέντρο του αντικειμένου βρίσκεται δεξιότερα του κεντρού του frame, ενώ αν $d_{1x} > d_{2x}$, τότε βρίσκεται αριστερότερα του κεντρού, κατά τον οριζόντιο άξονα X .

Αντίστοιχα, το ίδιο γίνεται και με τις κάθετες συντεταγμένες, δηλαδή $d_{1y} = |Y_c - Y_1|$ και $d_{2y} = |Y_c - Y_2|$.

Αν $d_{1y} < d_{2y}$, τότε γίνεται αντιληπτό ότι το κέντρο του αντικειμένου βρίσκεται χαμηλότερα του κεντρού του frame, ενώ αν $d_{1y} > d_{2y}$, τότε βρίσκεται υψηλότερα του κεντρού, κατά τον κάθετο άξονα Y .

Αυτή είναι η κύρια λογική αυτής της λειτουργίας, ώστε να καταλάβουμε που βρίσκεται το αντικείμενο σε σχέση με το κέντρο του παραθύρου.

Οπότε, με όριο να εκταθεί πλήρως ο βραχίονας προς τα δεξιά ή αριστερά, εκτελείται η εξής διαδικασία:

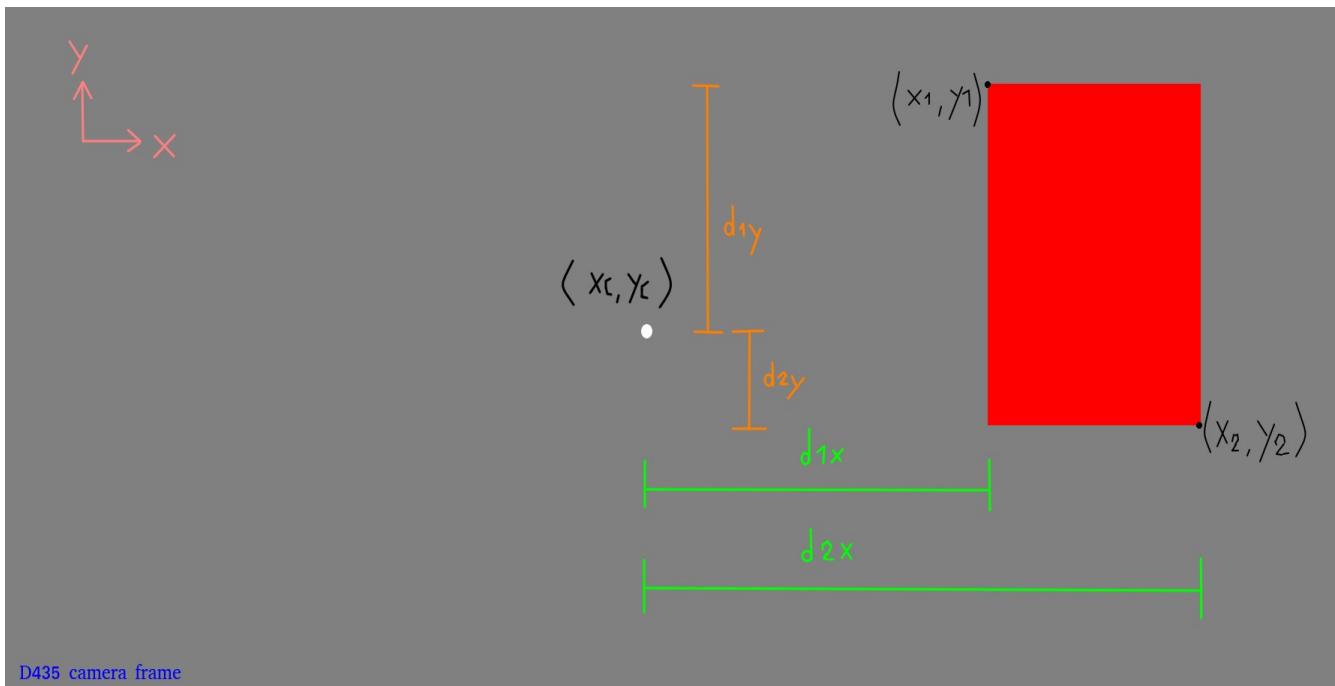
Γίνεται μετακίνηση του ΤΣΔ του βραχίονα κατά 0.005 μέτρα ανά 0.5 δευτερόλεπτα προς την κατεύθυνση του αντικειμένου (δεξιά ή αριστερά), καθώς και 0.005 μέτρα ανά 0.5 δευτερόλεπτα (πάνω ή κάτω).

Όπως αναφέρθηκε προηγουμένως, το αντικείμενο μπορεί να συμπέσει στο κέντρο του παραθύρου με μία μικρή απόκλιση των 10 pixels κατά τους άξονες X και Y της 2D εικόνας, για αυτό και τα βήματα του βραχίονα είναι αρκετά μικρά, δηλαδή 0.005 μέτρα / 0.5 δευτερόλεπτα, ώστε να προλάβει να σταματήσει αρκετά ικανοποιητικά στο στόχο. Η ανοχή αυτή υπάρχει, διότι δεν επηρεάζει το ΤΣΔ, δηλαδη την αρπάγη, να πιάσει το αντικείμενο με επιτυχία. Τη στιγμή αυτή, ο βραχίονας σταματάει οποιαδήποτε κίνηση. Εφόσον το αντικείμενο συμπέσει στο κέντρο του παραθύρου, το μεγαλύτερο ποσοστό της σάρωσης έχει ολοκληρωθεί.

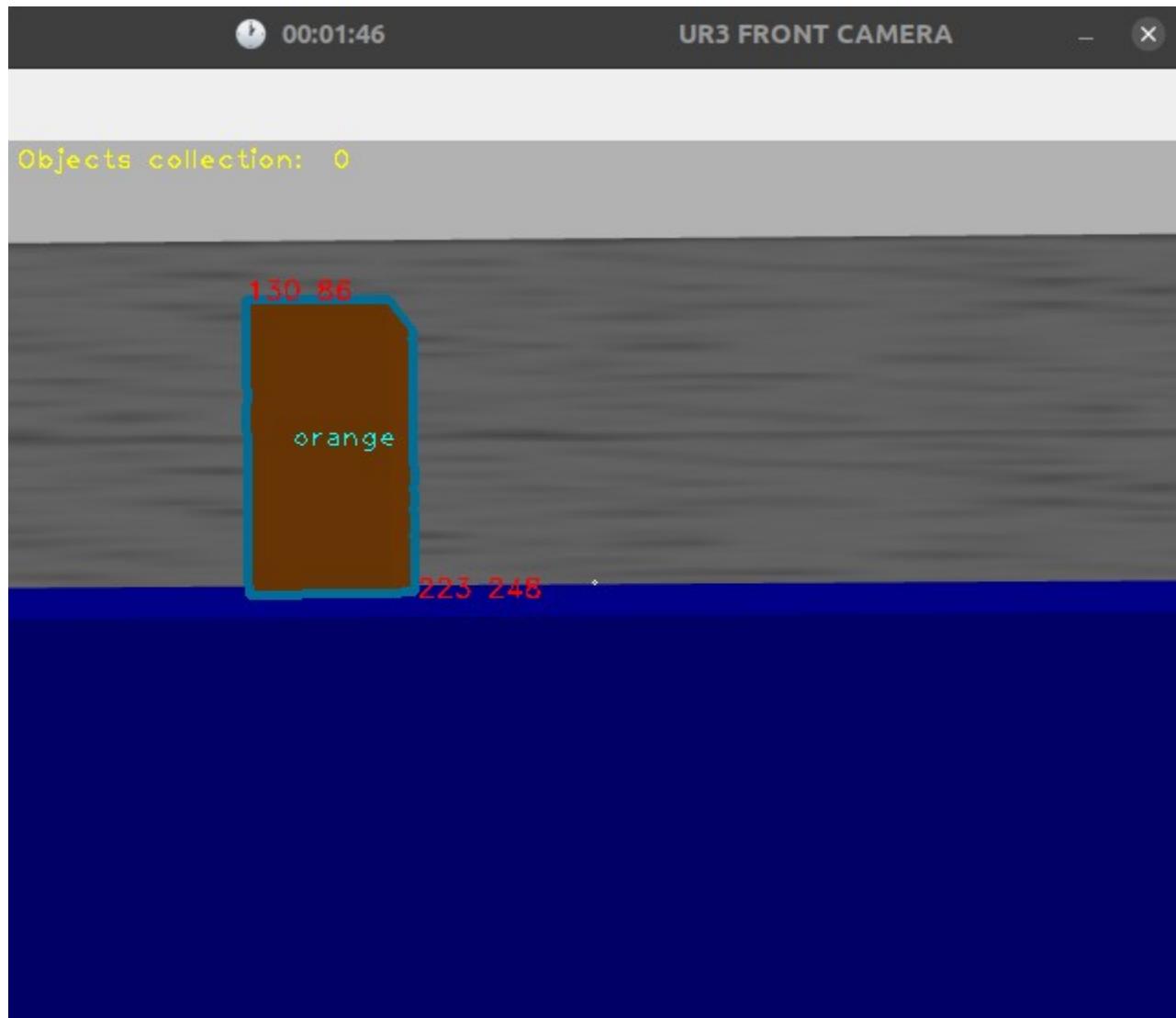
Στη συνέχεια, υψώνουμε τον βραχίονα κατά 0.044 μέτρα.

Η κίνηση αυτή πραγματοποιείται, διότι η κάμερα είναι τοποθετημένη 0.044 μέτρα πάνω από το κέντρο του ΤΣΔ.

Οπότε, με αυτή τη κίνηση το κέντρο του ΤΣΔ θα έρθει ακριβώς στη μέση του ύψους του αντικειμένου. Η εικόνα 2.25 παρακάτω, δείχνει τη πραγματική απεικόνιση στο Gazebo.

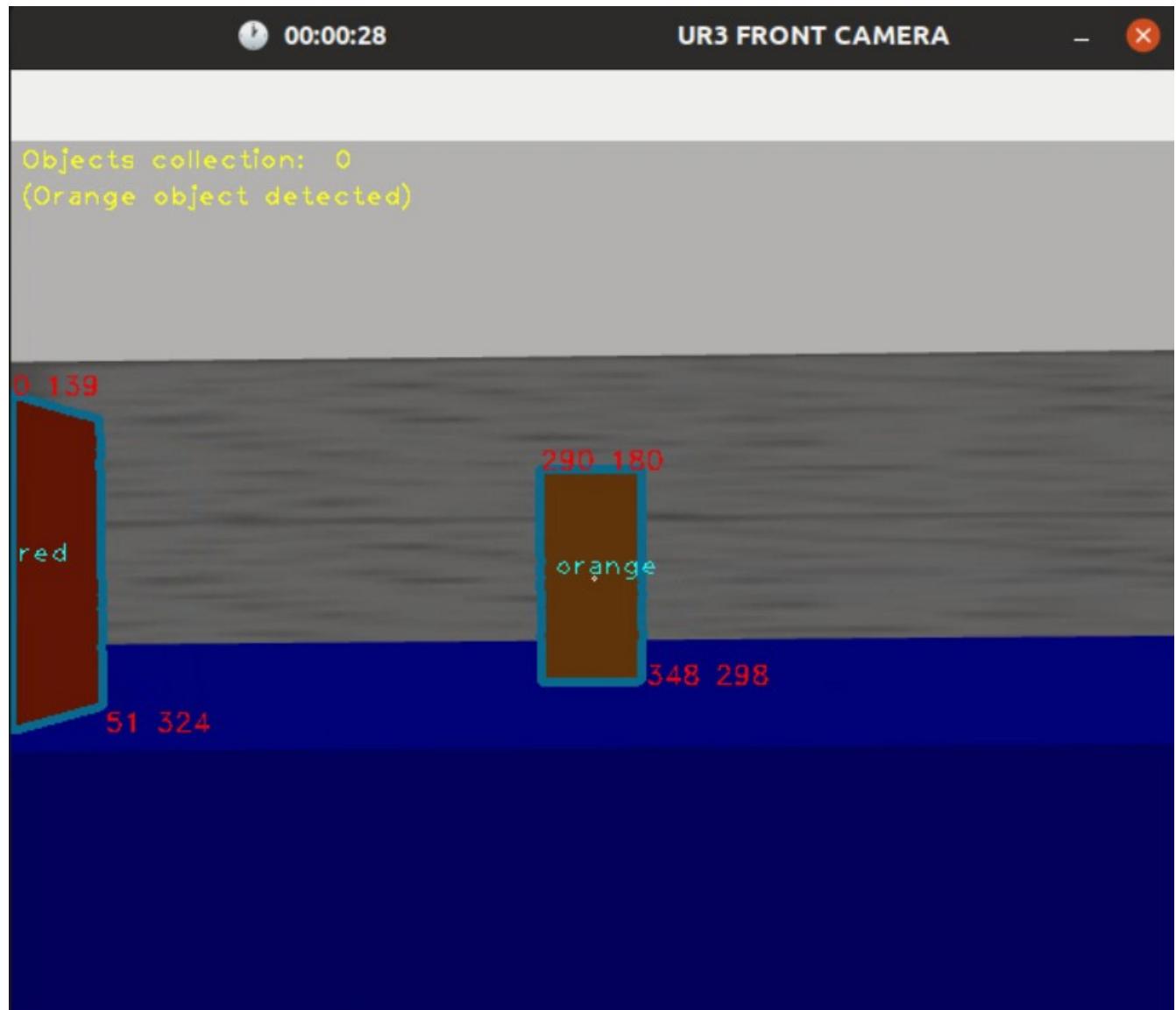


Εικόνα 2.24: Γραφική απεικόνιση frame και υπολογισμοί σάρωσης για ένα τυχαίο αντικείμενο (κόκκινο).



Εικόνα 2.25: Πραγματική γραφική απεικόνιση frame από περιβάλλον Gazebo.

Για παράδειγμα, όπως φαίνεται από την παραπάνω εικόνα, ο βραχίονας πρέπει μέσω των πληροφοριών (συντεταγμένων που έχει από το άσπρο κυκλάκι και των άκρων του αντικειμένου) να κατευθυνθεί προς τα πάνω και αριστερά, με σκοπό να έρθει το αντικείμενο στο κέντρο του frame της κάμερας (βλέπε εικόνα 2.26).



Εικόνα 2.26: Ο βραχίονας βρίσκεται ακριβώς στο κέντρο του αντικειμένου (παρατηρούμε ότι το άσπρο κυκλάκι είναι στο κέντρο του αντικειμένου).

Όλες οι προηγούμενες κινήσεις αντιπροσωπούσαν δύο άξονες του τρισδιάστατου χώρου. Ο τρίτος άξονας σχετίζεται με την απόσταση του ΤΣΔ από το αντικείμενο με σκοπό να το πιάσει. Τη στιγμή αυτή, ο βραχίονας βρίσκεται στο κέντρο του αντικειμένου και με μία απόσταση από αυτό. Με όριο να εκταθεί πλήρως ο βραχίονας με κατεύθυνση προς το κέντρο του αντικειμένου, μετράμε την απόσταση του αντικειμένου από την κάμερα, καθώς ο ίδιος κινείται προς αυτό. Αν η απόσταση της αρπάγης από το αντικείμενο φτάσει τα 12 εκατοστά, τότε ο βραχίονας θα κάνει μία τελευταία κίνηση των 0.02 μέτρων, ώστε να έρθει ακριβώς στο κέντρο του αντικειμένου και τέλος η αρπάγη κλείνει με σκοπό να πιάσει το αντικείμενο.

Σε αντίθετη περίπτωση, εάν η ο βραχίονας εκταθεί πλήρως δίχως να έχει προσεγγίσει τα 12 εκατοστά από το αντικείμενο, τότε ο βραχίονας κάνει μία κίνηση προς τα πίσω κατά 0.1 μέτρα. Στη συνέχεια λαμβάνουμε τα δεδομένα θέσης από τους άξονες X και Y του Husky, καθώς και την απόσταση του από το τραπέζι.

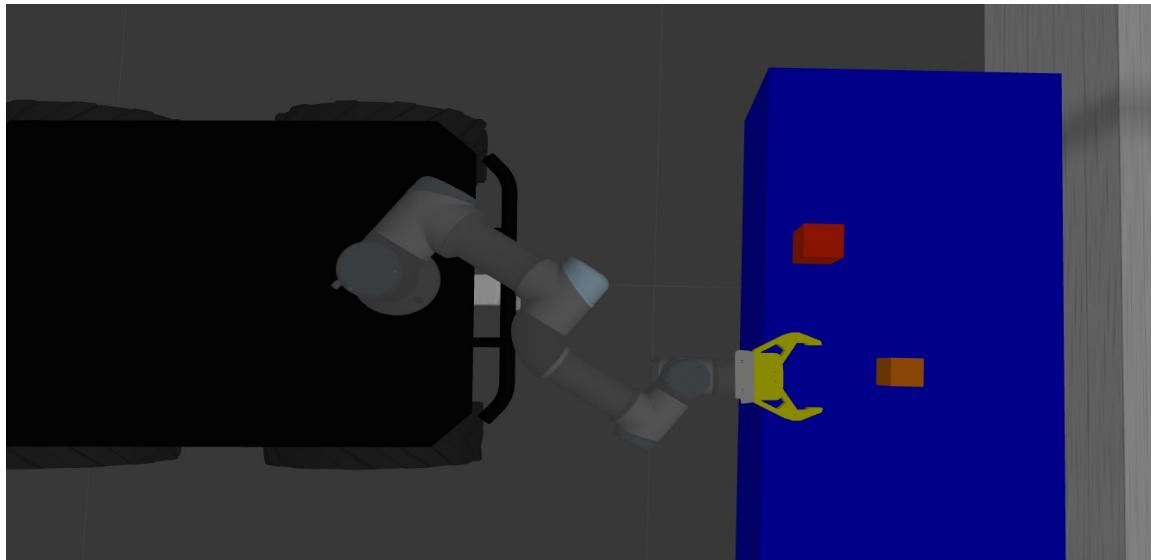
Ταυτόχρονα, υπάρχει ένα όριο απόστασης του Husky από το τραπέζι, η οποία είναι μία μεταβλητή με αρχική τιμή ίση με 0.55 μέτρα. Ας ονομάσουμε αυτή τη μεταβλητή A, για λόγους ευκολίας στη συνέχεια της περιγραφής μας. Επίσης, το ελάχιστο όριο απόστασης από το τραπέζι που μπορεί να έχει το Husky είναι τα 0.2 μέτρα.

Αν η απόσταση του Husky από το τραπέζι είναι μεγαλύτερη από 0.2 μέτρα, τότε μειώνουμε την μεταβλητή A κατά 0.1 μέτρα και όσο η απόσταση του Husky από το τραπέζι είναι μεγαλύτερη από την μεταβλητή A, τότε μετακινούμε το Husky προς τα εμπρός. Έπειτα, λαμβάνουμε πάλι τα δεδομένα θέσης από τους άξονες X και Y του Husky και παίρνουμε την απόλυτη διαφορά της τωρινής θέσης από την προηγούμενη (πριν γίνει η μετακίνηση). Ας ονομάσουμε αυτην την απόλυτη διαφορά με το όνομα K. Έπειτα, μετακινούμε τον βραχίονα προς τα εμπρός κατά K μέτρα και αυτή η διαδικασία εκτελείται κατ' επανάληψη έως ότου η απόσταση του ΤΣΔ προσεγγίσει τα 0.12 μέτρα από το αντικείμενο. Προηγουμένως, αναφέραμε μία οπισθοδρομική κίνηση του βραχίονα κατά 0.1 μέτρα.

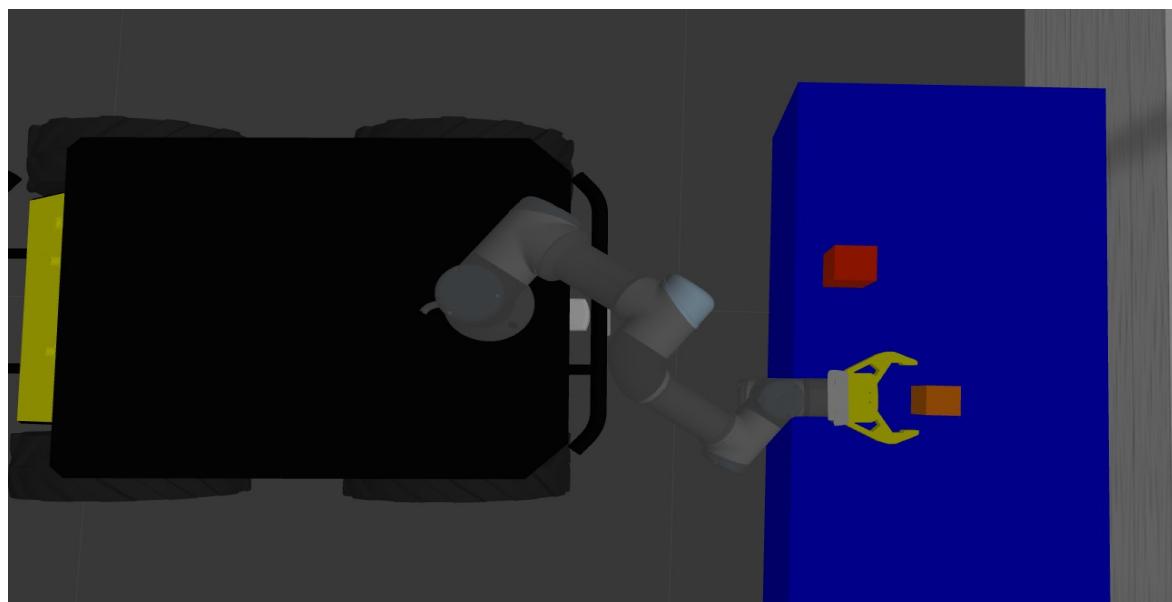
Θα αναρωτιόταν κανείς, γιατί γίνεται αυτή η κίνηση και δεν προχωράει κατευθείαν μπροστά το Husky. Λοιπόν, αυτή η κίνηση προς τα πίσω συμβαίνει, διότι αν η απόσταση του ΤΣΔ από το αντικείμενο ήταν λίγο μεγαλύτερη από τα 0.12 μέτρα, τότε αν

προχωρούσε μπροστά το Husky, ο βραχίονας πιθανόν να παρέσερνε το αντικείμενο που θα ήταν μπροστά του.

Τέλος, αν το Husky δεν έχει áλλο περιθώριο να προχωρήσει μπροστά και ο βραχίονας έχει εκταθεί πλήρως με αποτέλεσμα να μη μπορεί να πιάσει το αντικείμενο, τότε καταλαβαίνουμε ότι, παρ' óλη αυτή την συνεργασία μεταξύ των δύο ρομπότ να πίασουν το αντικείμενο, αυτό καθίσταται αδύνατο. Όλη η παραπάνω διαδικασία φαίνεται και από τις εικόνες 2.27 και 2.28 παρακάτω.



Εικόνα 2.27: Κίνηση-έκταση βραχίονα προς αντικείμενο (πριν).



Εικόνα 2.28: Κίνηση-έκταση βραχίονα προς αντικείμενο (μετά).

2.9 ΑΛΓΟΡΙΘΜΟΣ ΜΠΡΟΣΤΙΝΗΣ ΚΑΜΕΡΑΣ UR3

Η υποενότητα αυτή περιγράφει αναλυτικά τη διαδικασία με την οποία η κάμερα αναγνωρίζει αντικείμενα, καθώς και τα χρώματα αυτών, που βρίσκονται πάνω στα τραπέζια με σκοπό ο βραχίονας να κατευθυνθεί προς αυτά βάσει σχήματος και χρώματος (η διαδικασία κατεύθυνσης διατυπώθηκε στο υποκεφάλαιο 2.8).

Ουσιαστικά, θα περιγράψουμε τον αλγόριθμο αναγνώρισης χρωμάτων και σχήματος.

Επίσης, η αναγνώριση του χρώματος ενός αντικειμένου καθίσταται ένα γεγονός εξαιρετικά σημαντικό για τη μετέπειτα διαδικασία της αποστολής, η οποία θα εξηγηθεί σε επόμενο υποκεφάλαιο.

Όπως έχουμε εξηγήσει και σε προηγούμενα υποκεφάλαια, με την έναρξη της αποστολής αναδύεται ένα παράθυρο, το οποίο αναπαριστά σε πραγματικό χρόνο το περιβάλλον μίας από τις τρεις (3) διαφορετικές κάμερες των δύο (2) ρομπότ. Ουσιαστικά, αυτό το παράθυρο είναι τα “μάτια” των καμερών. Συγκεκριμένα, ο αλγόριθμος αυτός χρησιμοποιεί τη πάνω μπροστινή κάμερα του βραχίονα UR3, διότι αυτή “βοηθάει” τον βραχίονα να κατευθυνθεί προς ένα συγκεκριμένο αντικείμενο. Παρακάτω, λοιπόν, εξηγούμε αναλυτικά όλη τη διαδικασία.

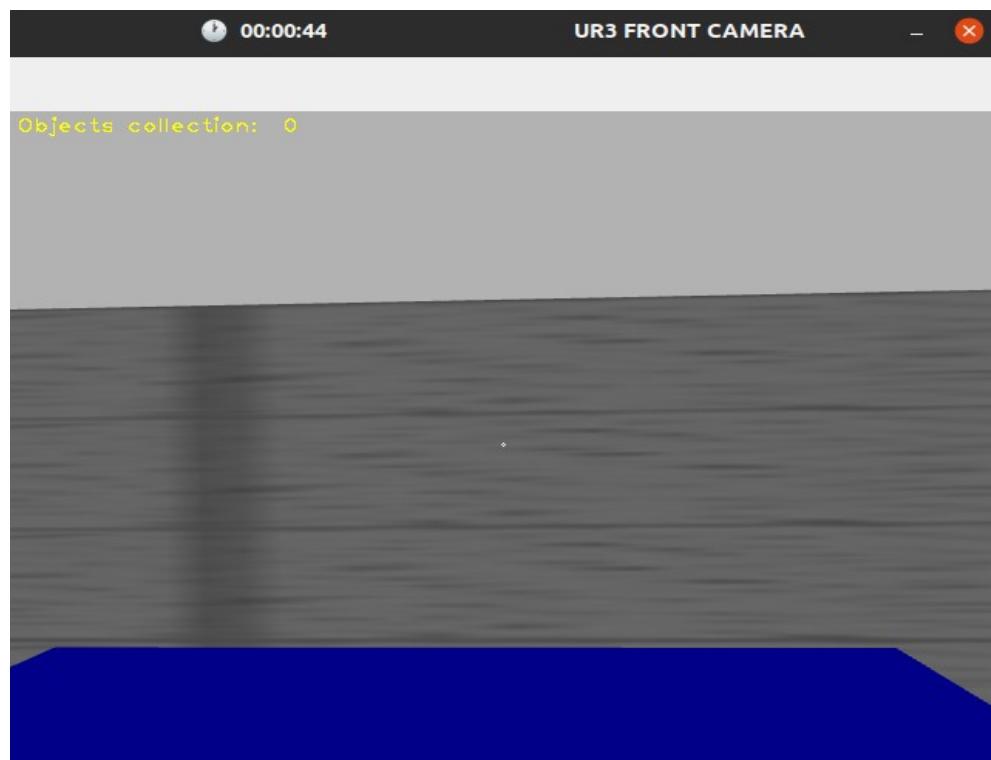
Καταρχάς, δημιουργούμε το παράθυρο του γραφικού περιβάλλοντος με τη βοήθεια της OpenCV. Οι διαστάσεις του είναι (640 , 480), δηλαδή 640 οριζόντια pixels για κάθε γραμμή και 480 κάθετα pixels για κάθε στήλη. Συνολικά, έχουμε $640 \times 480 = 307.200$ pixels για την εικόνα. Επίσης, μετασχηματίζουμε την εικόνα από τον χρωματικό χώρο BGR σε HSV. Το πλεονέκτημα έχει αναφερθεί στο υποκεφάλαιο 1.6.1.2, στο οποίο αναπτύσσεται ο αλγόριθμος κάμερας του Husky.

Έπειτα, δημιουργούμε έναν μικρό άσπρο κύκλο στο κέντρο του frame (βλέπε εικόνα 2.25), όπου σε αυτό το σημείο αναμένουμε να συμπέσει το κέντρο του αντικειμένου (η διαδικασία έχει εξηγηθεί στο προηγούμενο υποκεφάλαιο).

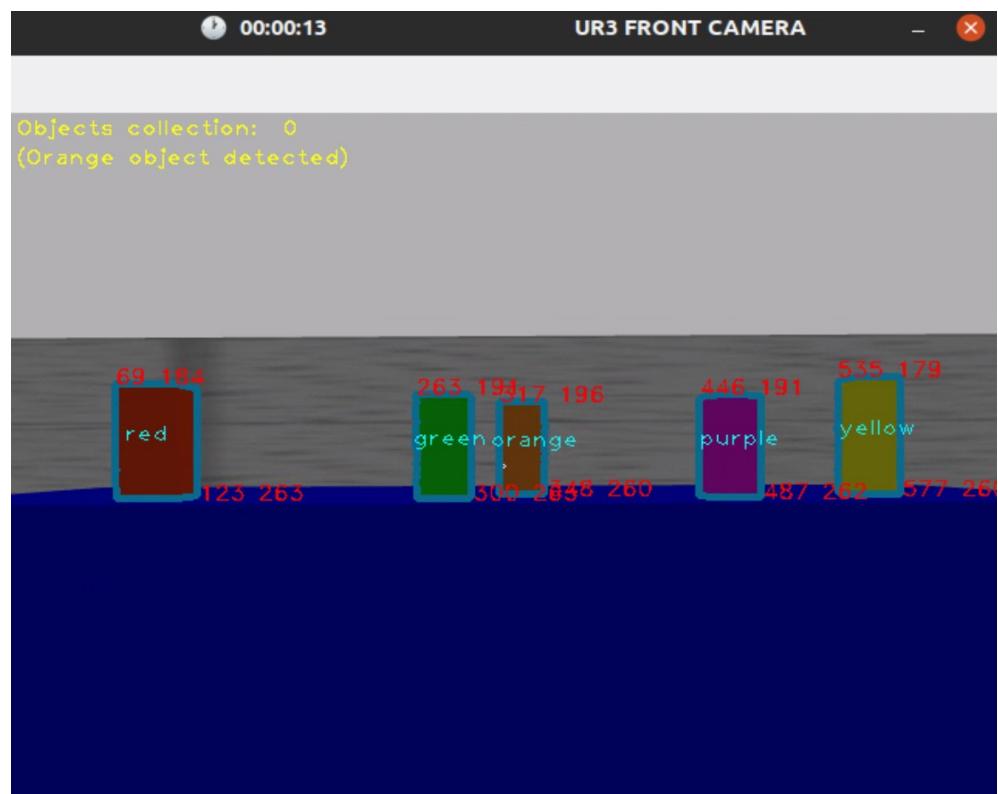
Ακόμα, αρχικοποιούμε τις τιμές του χρωματικού χώρου HSV για τα 5 διαφορετικά χρώματα που θα χρησιμοποιήσουμε για τα αντικείμενα μας στο χώρο του Gazebo.

- **Κόκκινο:**
 - Τιμές εντός του κλειστού διαστήματος [1 , 10] για την απόχρωση.
 - Τιμές εντός του κλειστού διαστήματος [100 , 255] για τον κορεσμό.
 - Τιμές εντός του κλειστού διαστήματος [100 , 255] για την φωτεινότητα.
- **Πορτοκαλί:**
 - Τιμές εντός του κλειστού διαστήματος [11 , 20] για την απόχρωση.
 - Τιμές εντός του κλειστού διαστήματος [100 , 255] για τον κορεσμό.
 - Τιμές εντός του κλειστού διαστήματος [100 , 255] για την φωτεινότητα.
- **Κίτρινο:**
 - Τιμές εντός του κλειστού διαστήματος [21 , 49] για την απόχρωση.
 - Τιμές εντός του κλειστού διαστήματος [100 , 255] για τον κορεσμό.
 - Τιμές εντός του κλειστού διαστήματος [100 , 255] για την φωτεινότητα.
- **Πράσινο:**
 - Τιμές εντός του κλειστού διαστήματος [50 , 70] για την απόχρωση.
 - Τιμές εντός του κλειστού διαστήματος [100 , 255] για τον κορεσμό.
 - Τιμές εντός του κλειστού διαστήματος [100 , 255] για την φωτεινότητα.
- **Μωβ:**
 - Τιμές εντός του κλειστού διαστήματος [140 , 160] για την απόχρωση.
 - Τιμές εντός του κλειστού διαστήματος [100 , 255] για τον κορεσμό.
 - Τιμές εντός του κλειστού διαστήματος [100 , 255] για την φωτεινότητα.

Όλα τα παραπάνω αντικείμενα διαφορετικού χρώματος φαίνονται στην εικόνα 2.30.



Εικόνα 2.29: Αναπαράσταση άσπρου κύκλου στο κέντρο του frame.



Εικόνα 2.30: Αναπαράσταση όλων των αντικειμένων στο frame.

Στη συνέχεια, χρησιμοποιούμε περίπου την ίδια τεχνική που αναφέρθηκε στην υποενότητα 1.6.2.2.

Αρχικά, κάθε χρονική στιγμή κρατάμε τα κάτω και άνω χρωματικά όρια για τα χρώματα που ορίσαμε παραπάνω και έπειτα χρησιμοποιούμε μία ειδική μέθοδο της OpenCV για τη δημιουργία μιας μάσκας, η οποία κρατάει τα pixels που βρίσκονται μεταξύ των παραπάνω ορίων στον χρωματικό χώρο HSV. Η μάσκα περιέχει τιμές 0 και 1, όπου το 1 αντιστοιχεί στα pixels που είναι εντός των ορίων χρώματος και το 0 σε αυτά που βρίσκονται εκτός. Συνολικά, αυτή η διαδικασία εκτελεί ανίχνευση χρώματος σε μία εικόνα χρησιμοποιώντας τον χρωματικό χώρο HSV και δημιουργεί μιά μάσκα που κρατάει μόνο τα pixels που έχουν χρώμα εντός συγκεκριμένων ορίων.

Μετά, εκμεταλλευόμαστε τη μάσκα που δημιουργήθηκε στο προηγούμενο βήμα για την ανίχνευση των συνεχόμενων περιοχών με χρώμα που πληρούν τα καθορισμένα χρωματικά όρια. Χρησιμοποιείται μία ειδική μέθοδος της OpenCV για την εύρεση των συνεχόμενων περιοχών στη μάσκα. Τέλος, χρησιμοποιούμε μία ακόμη μέθοδο για να σχεδιάσουμε τα όρια των αντικειμένων πάνω στο αρχικό frame, όπως ακριβώς φαίνεται από την εικόνα 2.30. Ακόμα, θέλουμε εκτυπώσουμε τη λέξη του χρώματος (πχ. κίτρινο, μπλε, κόκκινο, πράσινο, πορτοκαλί ή μωβ) για το κάθε αντικείμενο στο frame στο κέντρο του αντικειμένου, όπως μπορούμε να παρατηρήσουμε από την εικόνα 2.30. Κάνουμε χρήση μίας συνάρτησης, η οποία υπολογίζει το κέντρο βάρους των ορίων όλων των αντικειμένων της εικόνας. Ουσιαστικά, η μέθοδος αυτή παίρνει το μέσο όρο των συντεταγμένων του οριζόντιου άξονα και αντίστοιχα το μέσο όρο των συντεταγμένων του κάθετου άξονα, οπότε προκύπτει ότι το κέντρο βάρους βρίσκεται στο κέντρο του αντικειμένου και σε εκείνο το σημείο τυπώνεται η λέξη του χρώματος για το αντίστοιχο αντικείμενο.

Τέλος, γίνεται έλεγχος αν η συνιστώσα χρώματος (Hue) από το άσπρο κυκλάκι (pixel) στο κέντρο του frame συμπέσει σε κάποιο από τα παραπάνω χρωματικά όρια που διατυπώσαμε. Τότε, μπορούμε να καταλάβουμε το χρώμα του αντικειμένου που βρίσκεται στο κέντρο του παραθύρου μας, το οποίο εκμεταλλευόμαστε αργότερα. Για παράδειγμα, η εικόνα 2.30 μας δείχνει το μήνυμα: “Orange object detected”.

2.10 ΣΥΓΚΡΙΣΗ ΑΛΓΟΡΙΘΜΩΝ ΚΑΜΕΡΑΣ

Στο κεφάλαιο 2.8 εξηγήθηκε λεπτομερώς ο αλγόριθμος, ο οποίος χρησιμοποιήθηκε από τα “μάτια” της πάνω μπροστινής κάμερας του βραχίονα για να κατευθύνει τον ίδιο στο κέντρο του αντικειμένου (βλέπε εικόνα 2.26).

Από την άλλη, στο ίδιο κεφάλαιο, θα εξεταστεί πλέον ο δεύτερος αλγόριθμος, ο οποίος ονομάζεται IBVS (Image-Based Visual Servoing) και χρησιμοποιείται για τον ίδιο σκοπό, δηλαδή την ανάλυση της εικόνας για τον έλεγχο κίνησης ενός βραχίονα. Σε αυτήν την ενότητα, θα παρουσιαστούν λεπτομερώς τα χαρακτηριστικά του αλγορίθμου και η λειτουργία του. Έπειτα, θα προχωρήσουμε σε μια σύγκριση με τον προηγούμενο αλγόριθμο που περιγράφηκε στην παράγραφο 2.8, αναδεικνύοντας τις ομοιότητες και τις διαφορές τους. Τέλος, θα αξιολογήσουμε την απόδοση του δεύτερου αλγορίθμου σε σχέση με τα κριτήρια που έχουν οριστεί, προσφέροντας έτσι μια ολοκληρωμένη κατανόηση των δυνατοτήτων και των περιορισμών του.

Αρχικά, ας πούμε μερικά λόγια για τον ακλγόριθμο IBVS.

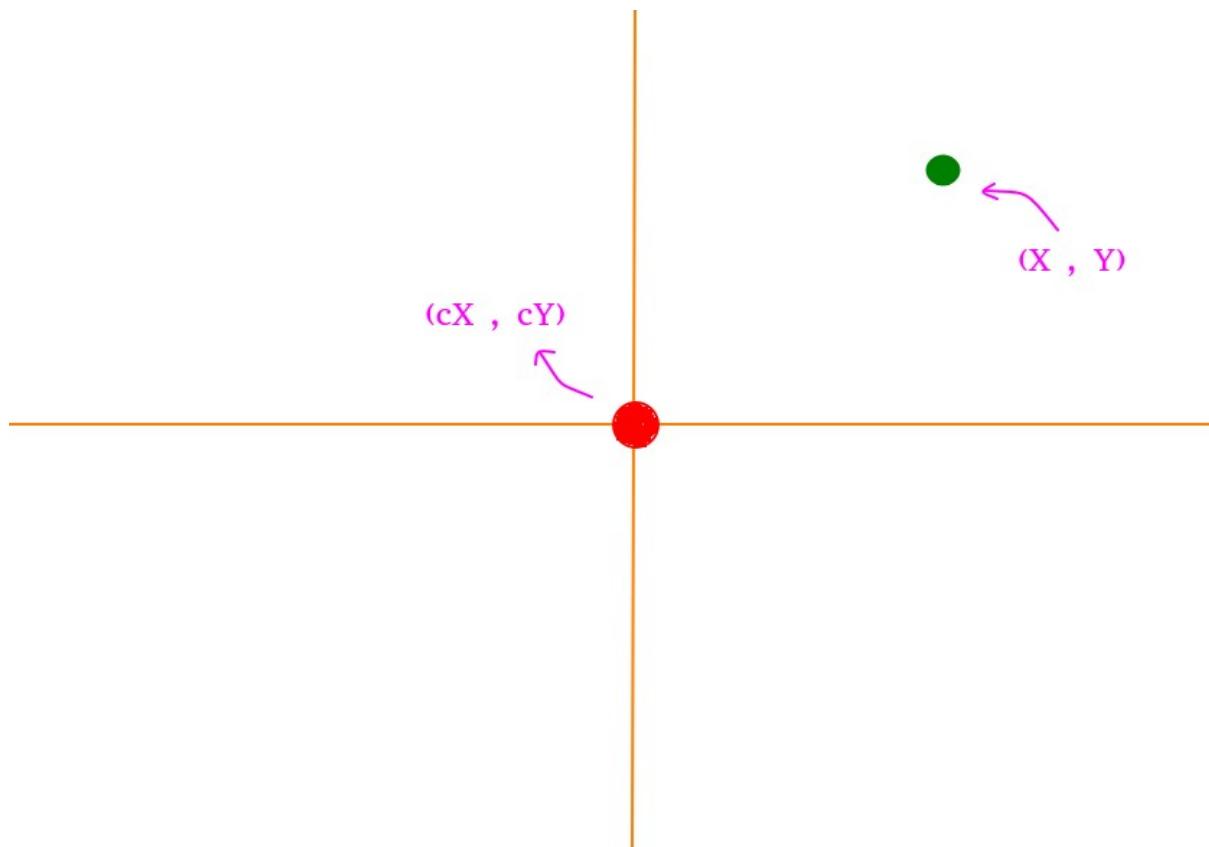
Ο IBVS επικεντρώνεται στη χρήση πληροφοριών που προέρχονται από μία εικόνα για τον έλεγχο της κίνησης ενός ρομπότ, στη περίπτωση μας, ένα 6-DOF βραχίονα. Ας δούμε τα βασικά στάδια του αλγορίθμου όταν αναφερόμαστε σε μία εικόνα:

Η πρώτη φάση είναι η εξαγωγή χαρακτηριστικών από την εικόνα, όπως ένα σημείο ενδιαφέροντος (συντεταγμένες της 2D εικόνας). Τα χαρακτηριστικά αυτά πρέπει να είναι επιλεγμένα με προσοχή και να είναι ανθεκτικά σε αλλαγές στο περιβάλλον, διότι η κάμερα μέσω του βραχίονα μετακινείται στο 3D χώρο, οπότε η αλλαγή αυτή επηρεάζει τις συντεταγμένες της 2D εικόνας. Έπειτα, ο χρήστης καθορίζει τον επιθυμητό στόχο για το ρομπότ (πχ το κέντρο του frame). Στη συνέχεια, υπολογίζεται το σφάλμα εικόνας, που αντιπροσωπεύει τη διαφορά μεταξύ των παρατηρούμενων χαρακτηριστικών στην τρέχουσα εικόνα και των επιθυμητών χαρακτηριστικών που ανιχνεύονται από την κάμερα. Ο ελεγκτής IBVS χρησιμοποιεί το σφάλμα εικόνας για να υπολογίσει τις απαραίτητες εντολές ελέγχου που πρέπει να εφαρμοστούν στο βραχίονα, προκειμένου να μετακινηθεί προς τον επιθυμητό στόχο. Οι αλγόριθμοι IBVS είναι χρήσιμοι σε ποικίλους τομείς, όπως η βιομηχανία, η ιατρική και η ρομποτική όπου η ακρίβεια στις μεταβαλλόμενες συνθήκες είναι σημαντικές.

2.10.1 ΑΛΓΟΡΙΘΜΟΣ IBVS

Περιγραφή & τύποι υπολογισμού:

Παρακάτω στην εικόνα 2.31, φαίνεται το frame, όπου στις συντεταγμένες (X , Y) βρίσκεται το κέντρο του αντικειμένου και οι συντεταγμένες (cX , cY) αναπαριστούν το κέντρο του frame/παραθύρου, όπου σε αυτό το σημείο επιθυμούμε να φέρουμε το αντικείμενο μας. Ο τρόπος υπολογισμού εξηγείται αναλυτικά με τους παρακάτω μαθηματικούς τύπους.



Εικόνα 2.31: Αναπαράσταση συντεταγμένων αντικειμένου (πράσινο)
και κέντρου frame (κόκκινο).

Στόχος είναι να βρούμε, ανά πάσα χρονική στιγμή και μέχρι να έρθει το αντικείμενο στο κέντρο του παραθύρου μας, τις θέσεις του τελικού στοιχείου δράσης (ΤΣΔ) του βραχίονα.

Τύπος υπολογισμού: $V = -K \cdot L_e^{-1} \cdot e$, όπου

$V \rightarrow$ Ταχύτητα ΤΣΔ

$L_e^{-1} \rightarrow$ Αντίστροφος πίνακας αλληλεπίδρασης ή Ιακωβιανή εικόνας (προσέγγιση)

$e \rightarrow$ σφάλμα εικόνας

Ύστερα, επειδή θέλουμε τη θέση του ΤΣΔ, από τον τύπο $x = v \cdot t$, λαμβάνουμε τη θέση, όπου $x \rightarrow$ θέση ΤΣΔ και $t \rightarrow$ χρόνος που θα διανύσει το ΤΣΔ του βραχίονα από μία τυχαία αρχική θέση προς τον στόχο, δηλαδή το κέντρο του αντικειμένου μας.

Παρακάτω δίνεται η εξήγηση των παραπάνω όρων.

◆ e

Γίνεται προσπάθεια ελαχιστοποίησης του σφάλματος: $e(t) = s(m(t), a) - s^*$

- Διάνυσμα $m(t)$: Συντεταγμένες σημείων ενδιφέροντος/κέντρου ενός αντικειμένου.
- Διάνυσμα $s(m(t), a)$: Οπτικά χαρακτηριστικά, τα οποία υπολογίζονται από τις μετρήσεις $m(t)$ και τις παραμέτρους a .
- Διάνυσμα a : Παράμετροι κάμερας [f, a_x, a_y, c_x, c_y].
- Εστιακή απόσταση κάμερας f : 0.008 meters
- Μέγεθος εικονοστοιχείου (pixel) [a_x, a_y] = [640 // 0.008, 480, 0.008] px/m
- Συντεταγμένες κέντου frame (c_x, c_y) = (320, 240) px
- Μέγεθος εικόνας: 640 x 480 px
- Διάνυσμα s^* : Συντεταγμένες στόχου (x^*, y^*).

Το ε είναι ένας πίνακας 2x1:

$$e = \begin{bmatrix} init_x - target_x \\ init_y - target_y \end{bmatrix}, \text{ όπου:}$$

$$init_x = \frac{(X - c_x)}{(f \cdot a_x)}, \quad target_x = \frac{(X - c_x)}{(f \cdot a_x)},$$

$$init_y = \frac{(Y - c_y)}{(f \cdot a_y)}, \quad target_y = \frac{(Y - c_y)}{(f \cdot a_y)},$$

X,Y: Συντεταγμένες κέντρου αντικειμένου (βλέπε εικόνα 2.31).

◆ K

Το K είναι ένα θετικό κέρδος, μία σταθερά με τιμή (εδώ) ίση με 0.02.

◆ L_e

Είναι ο πίνακας αλληλεπίδρασης ή αλλιώς Ιακωβιανή εικόνας 6x2. Γενικά, είναι δύσκολο να βρεθεί ο L_e σε πραγματικά συστήματα, οπότε για αυτόν τον λόγο γίνεται ένας μικρός συμβιβασμός παίρνοντας μία προσέγγιση του.

Παρακάτω δίνεται ο πίνακας αλληλεπίδρασης σε μητρωϊκή μορφή:

$$L_e = \begin{pmatrix} -\frac{1}{Z} & 0 & \frac{x}{Z} & x \cdot y & -(1+x^2) & y \\ 0 & \frac{-1}{Z} & \frac{y}{Z} & 1+y^2 & -x \cdot y & -x \end{pmatrix}$$

1) $Z \rightarrow$ απόσταση κάμερας από αντικείμενο, με τιμή ίση με 0.5 μέτρα.

$$2) x \rightarrow \frac{(X - c_x)}{(f \cdot a_x)}$$

$$3) y \rightarrow \frac{(Y - c_y)}{(f \cdot a_y)}$$

4) $X \rightarrow$ Οριζόντια συντεταγμένη κέντρου αντικειμένου (βλέπε εικόνα 2.31).

5) $Y \rightarrow$ Κάθετη συντεταγμένη κέντρου αντικειμένου (βλέπε εικόνα 2.31).

Στη συγκεκριμένη περίπτωση, κάνουμε 150 επαναλήψεις και από τον τύπο $V = -K \cdot L_e^{-1} \cdot e$ και $x = v \cdot t$, λαμβάνουμε 150 διαφορετικές τιμές για τη θέση $x = [X, Y, Z]^T$ αντίστοιχα, δηλαδή τις συντεταγμένες του βραχίονα στο τρισδιάστατο χώρο του περιβάλλοντος Gazebo. Έπειτα, προσθέτουμε αυτές 3 τιμές αντίστοιχα στις ήδη υπάρχουσες του βραχίονα και με αυτόν τον τρόπο ο UR3 μέσω της κάμερας και του αλγορίθμου IBVS που αναπτύξαμε, κατευθύνεται προς το κέντρο του αντικειμένου μας.

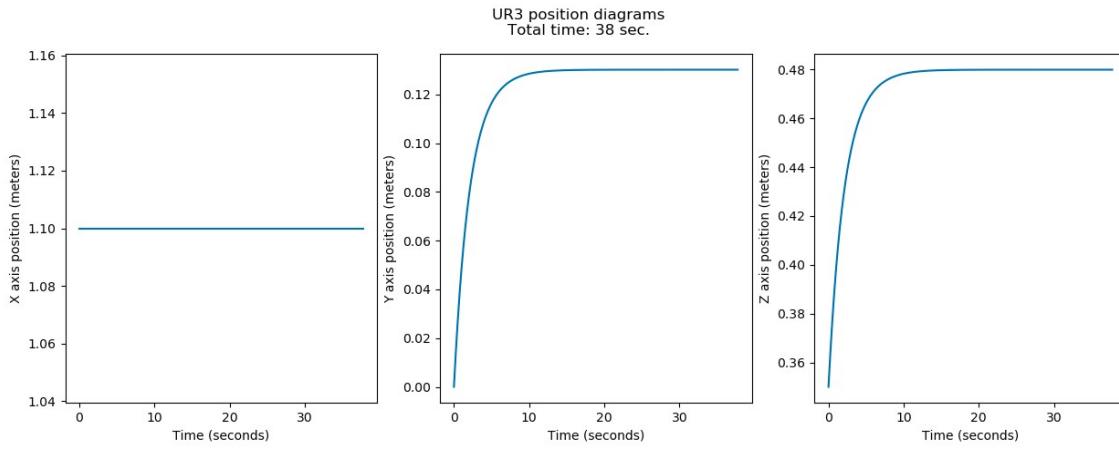
2.10.2 ΣΥΓΚΡΙΣΗ ΑΛΓΟΡΙΘΜΟΥ IBVS ΚΑΙ ΚΕΦΑΛΑΙΟΥ 2.8

Και οι 2 αλγόριθμοι έτρεξαν από την ίδια αρχική θέση. Επίσης, το αντικείμενο ήταν στην ίδια θέση, ώστε η σύγκριση να είναι δίκαιη και στις δύο περιπτώσεις. Όπως βλέπουμε παρακάτω στις εικόνες 2.32 και 2.33, ο βραχίονας ξεκινάει και στις δύο περιπτώσεις από αρχική θέση $X = 1.10$ μέτρα από την αρχή των αξόνων του Gazebo, $Y = 0$ μέτρα από την αρχή των αξόνων και $Z = 0.35$ μέτρα από το έδαφος. Τέλος, καταλήγουν και οι δύο αλγόριθμοι στην ίδια θέση, δηλαδή στο κέντρο του αντικειμένου με θέση $X = 1.10$, εφόσον δεν υπάρχει κίνηση βραχίονα προις τα εμπρος, παρά μόνο δισδιάσταση κίνηση (δεξιά, αριστερά, πάνω και κάτω) στο χώρο, $Y = 0.175$ μέτρα από την αρχή των αξόνων και $Z = 0.48$ μέτρα από το έδαφος. Οι χρόνοι εκπλήρωσης φαίνονται παρακάτω.

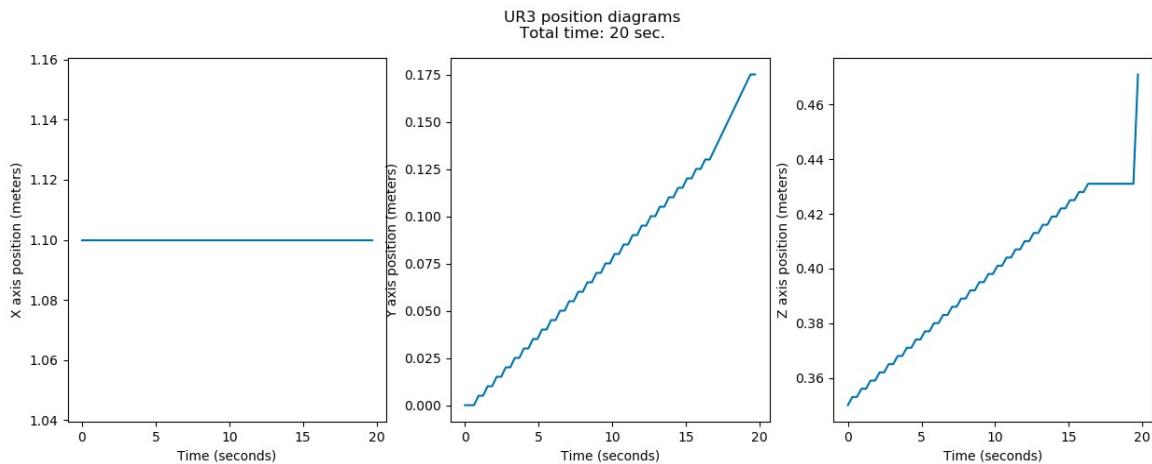
$$T_{ibvs} = 38 \text{ seconds}$$

$$T_{2.8} = 20 \text{ seconds}$$

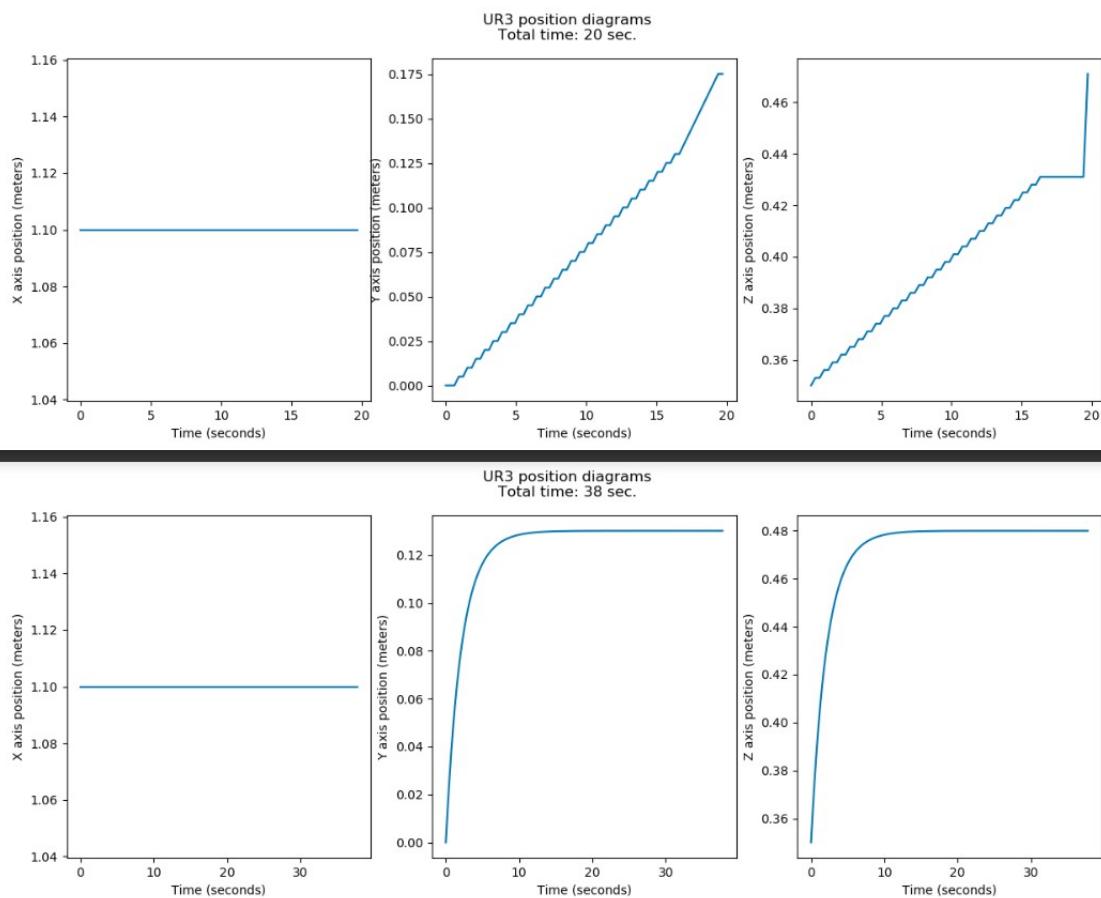
Συμπέρασμα: Υπάρχει μεγαλύτερη ομάλοτητα στη κίνηση-τροχιά του βραχίονα μέσω του αλγορίθμου IBVS, αλλά η ταχύτητα του αλγορίθμου που αναπτύχθηκε στην παράγραφο 2.8 παραπάνω είναι σαφώς μεγαλύτερη και κοστίζει πολύ λιγότερο χρόνο να εκπληρωθεί τελικά η κίνηση.



Εικόνα 2.32: Αλγόριθμος IBVS.



Εικόνα 2.33: Αλγόριθμος παραγράφου 2.8.



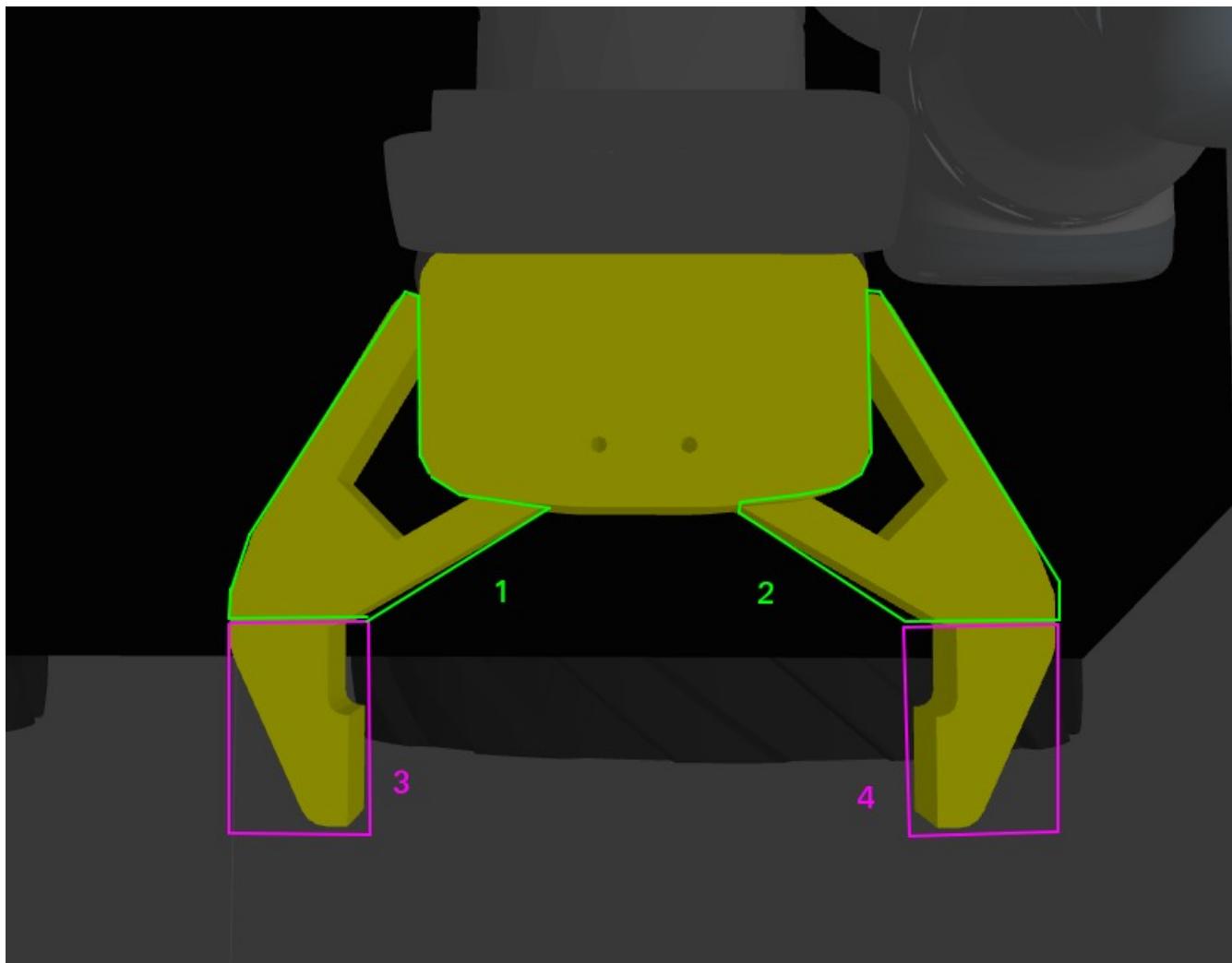
Εικόνα 2.34: Σύγκριση αλγορίθμων IBVS και παραγράφου 2.8 (μαζί).

2.11 ΑΡΠΑΓΗ ΑΝΤΙΚΕΙΜΕΝΟΥ

Το Husky έχει ολοκληρώσει τις κινήσεις και ενέργεις του με σκοπό να έρθει όσο πιο κοντά στο κέντρο του τραπεζιού. Με τη σειρά του ο βραχίονας έχει ενεργοποιηθεί, έχει σαρώσει τον χώρο του τραπεζιού για τυχόν αντικείμενα, έχει βρει ένα αντικείμενο με συγκεκριμένο χρώμα και τελικά έχει κατευθυνθεί προς αυτό σε πολύ κοντινή απόσταση. Αυτό που απομένει τώρα είναι να το αρπάξει.

Στο υποκεφάλαιο αυτό, περιγράφουμε τη λειτουργία της αρπάγης του βραχίονα UR3.

Καταρχάς, η αρπάγη αποτελείται από τέσσερα (4) μέρη-δαγκάνες, οι οποίες έχουν την δυνατότητα να ανοιγοκλείνουν ανεξάρτητα από τις υπόλοιπες. Παρακάτω, στην εικόνα 2.35 μπορούμε να δούμε μία αναπαράσταση της αρπάγης με τις 4 δαγκάνες της.



Εικόνα 2.35: Αναπαράσταση μέρη δαγκάνων από αρπάγη βραχίονα.

Παρ'όλα αυτά, η χρήση του βραχίονα στην εφαρμογή είναι να κλείνουν οι δαγκάνες 1-3 μαζί και ταυτόχρονα με τις 2-4, όπως μπορούμε να δούμε στην εικόνα 2.36 .



Εικόνα 2.36: Κλείσιμο δαγκάνων 1-3 και 2-4 ταυτόχρονα (κλείσιμο αρπάγης).

Γενικά, η αρπάγη έχει μία δύναμη λαβής, όπου όταν κυμαίνεται από τις τιμές 0 έως 1.05. Συγκεκριμένα, στη τιμή 0 θεωρείται ότι η αρπάγη είναι τελείως ανοιχτή, όπως στην εικόνα 2.35 και στη τιμή 1.05 θεωρείται τελείως κλειστή. Στην εικόνα 2.36 η δύναμη λαβής της αρπάγης έχει τιμή ίση με 0.8 (ελαφρώς κλειστή δύναμη λαβής).

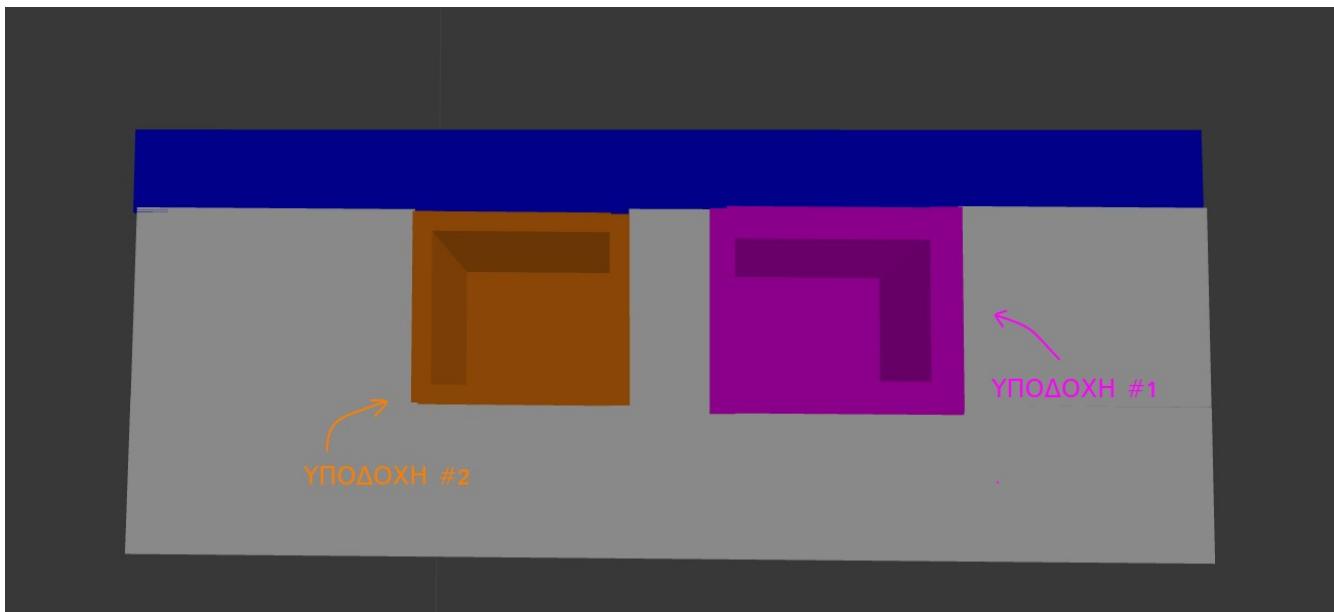
Οπότε, τη στιγμή που η αρπάγη βρίσκεται κοντά στο αντικείμενο, όπως φαίνεται από την εικόνα 2.28, η αρπάγη κλείνει με τιμή δύναμης λαβής ίση με 0.73 σε όλα τα αντικείμενα, εφόσον αυτά έχουν το ίδιο μέγεθος και σχήμα για ευκολία διαχείρισης.

Όταν κλείσει η αρπάγη, ύστερα από 5 δευτερόλεπτα, ο βραχίονας υψώνεται κατά 30 εκατοστά και τελικά κατευθύνεται και πάλι στην αρχική του θέση, δηλαδή τη θέση εν ονόματι “Home position”, ώστε να συνεχίστει η αποστολή.

2.12 ΣΑΡΩΣΗ ΤΡΑΠΕΖΙΟΥ-ΣΤΟΧΟΥ

Το υποκεφάλαιο περιγράφει αναλυτικά την εισαγωγή ενός χρωματιστού αντικειμένου στη κατάλληλη υποδοχή του κατάλληλου τραπεζιού από τα δύο ρομπότ.

Καταρχάς, ας δούμε παρακάτω στην εικόνα 2.45 πως μοιάζουν τα τραπέζια-στόχοι.



Εικόνα 2.37: Αναπαράσταση τραπεζιού-στόχου με χρωματιστές υποδοχές.

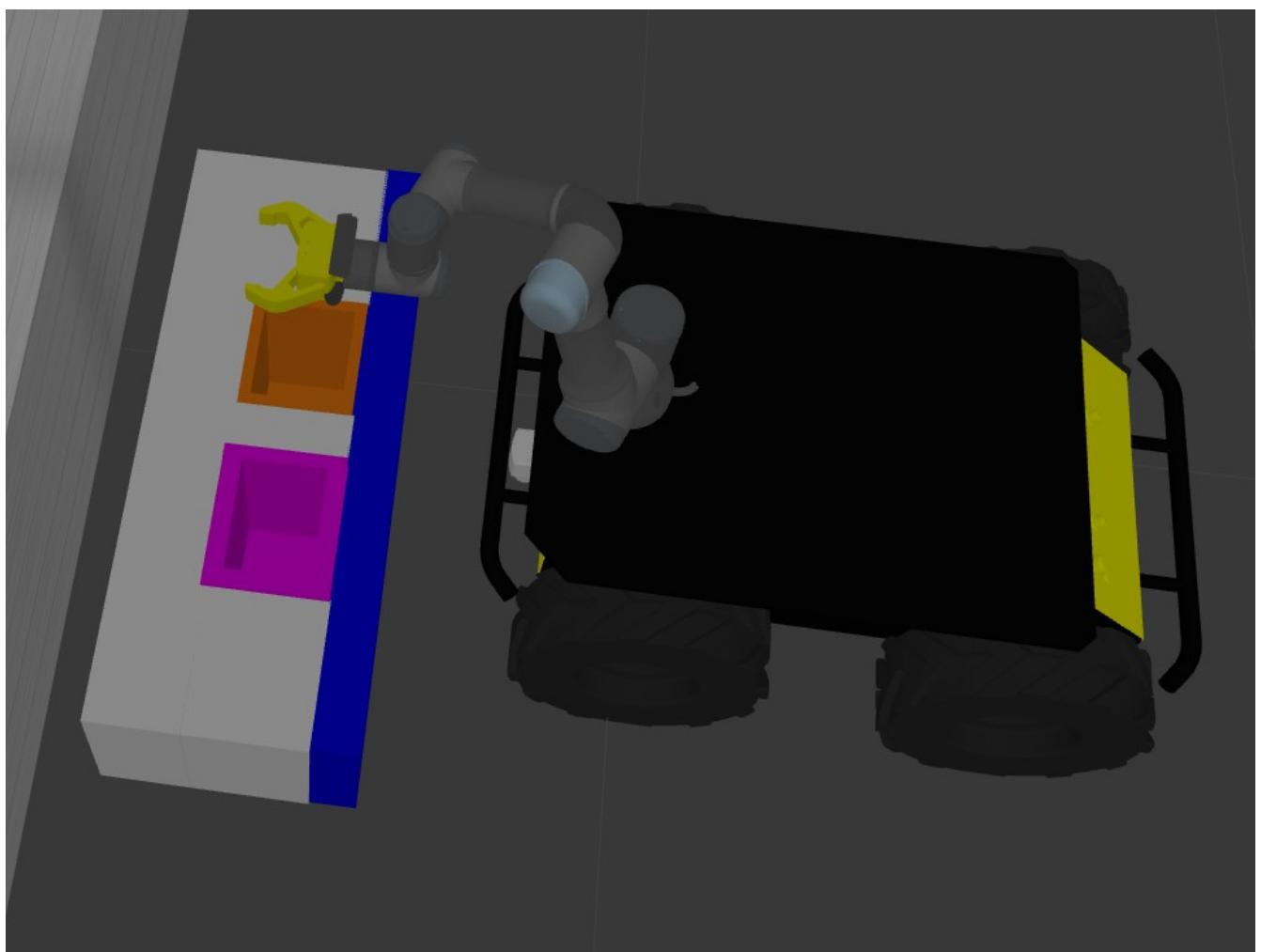
Υπάρχουν δύο τραπέζια με δύο υποδοχές διαφορετικού χρώματος στο χώρο. Στόχος είναι το Husky να κατευθυνθεί στο τραπέζι, το οποίο περιέχει την υποδοχή με χρώμα ίδιο με το αντικείμενο που κρατάει ο UR3 και στη συνέχεια όταν πλησιάσει αρκετά κοντά στο τραπέζι, ο βραχίονας να επιλέξει και να αφήσει το χρωματιστό αντικείμενο στη κατάλληλη υποδοχή, δηλαδή την υποδοχή ίδιου χρώματος με το αντικείμενο που έχει.

Συγκεκριμένα, το Husky πρέπει να πλησιάσει αρκετά κοντά στο τραπέζι και σε απόσταση μικρότερη από τα υπόλοιπα τραπέζια (τα οποία δεν είναι τραπέζια-στόχοι), ώστε ο βραχίονας να έχει τη δυνατότητα και άνεση να αφήσει το αντικείμενο στην υποδοχή.

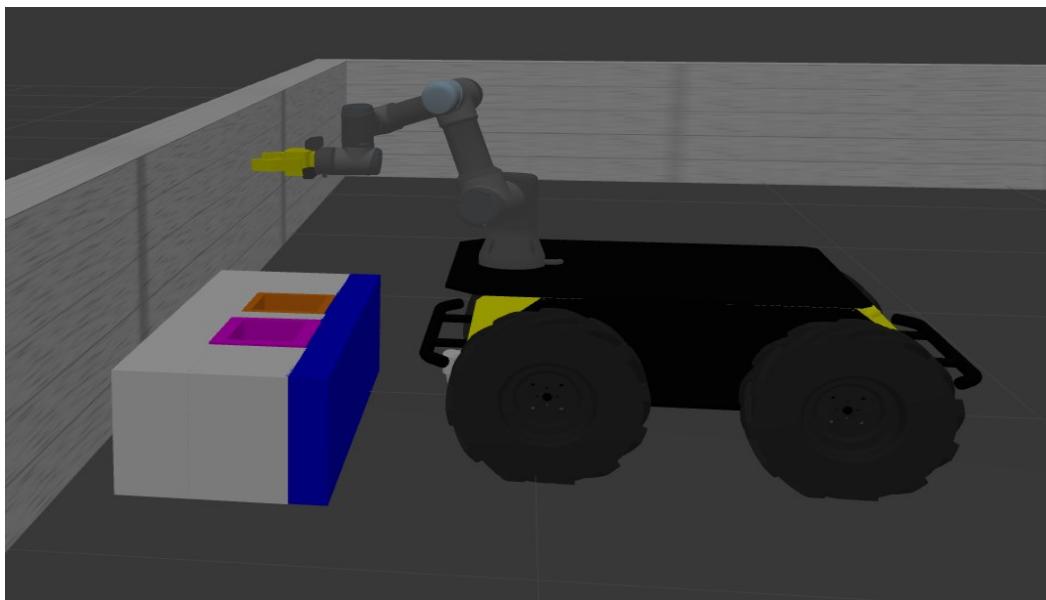
Η απόσταση αυτή είναι ίση με 20 εκατοστά. Τότε, ενεργοποείται η τρίτη κάμερα των δύο ρομπότ, δηλαδή η κάτω μπροστινή κάμερα του βραχίονα UR3 (βλέπε εικόνα 2.4). Τώρα, ο βραχίονας πρέπει να λάβει τη κατάλληλη αρχική θέση, ώστε να είναι έτοιμος στη συνέχεια να σαρώσει και να βρει την αντίστοιχη υποδοχή.

Ο βραχίονας, λοιπόν, από την θέση "Home position" που βρίσκεται, υψώνεται στα 60 εκατοστά από το έδαφος, ενώ παράλληλα εκτείνεται κατά 85 εκατοστά προς το μέρος του τραπεζιού και σε ίση απόσταση μεταξύ των μπροστινών τροχών του Husky.

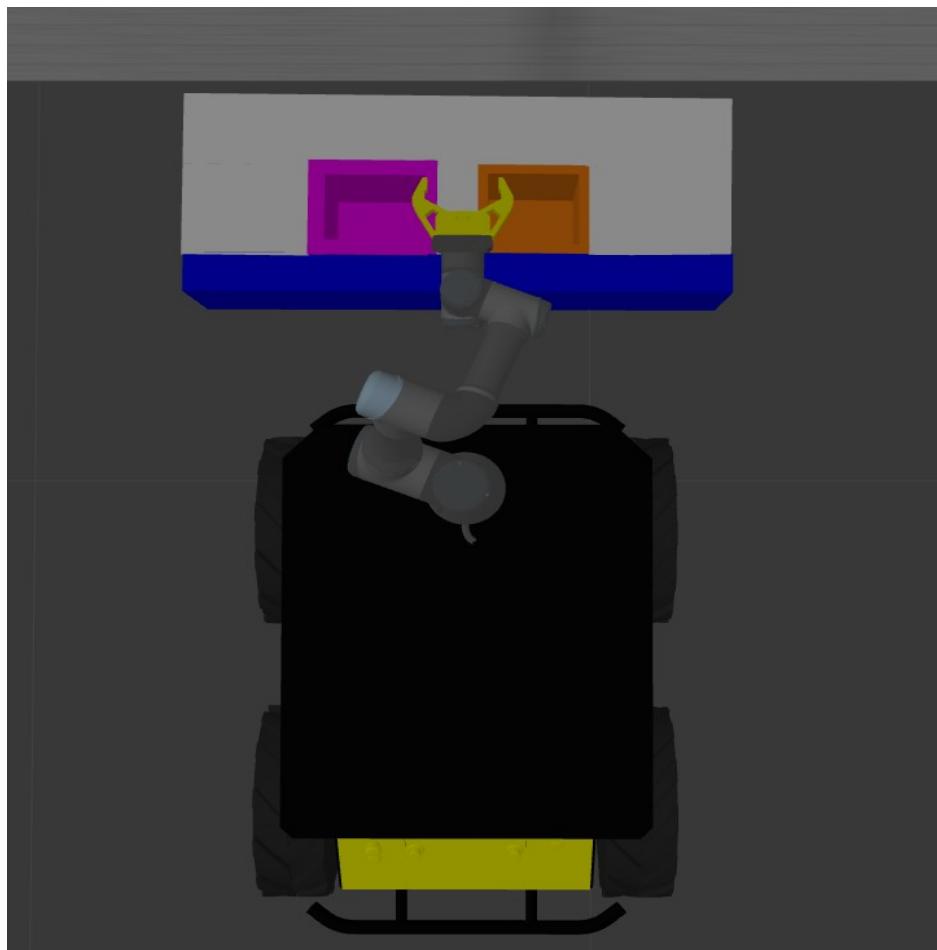
Οι εικόνες 2.38, 2.39 και 2.40 μαρτυρούν την παραπάνω περιγραφή.



Εικόνα 2.38: Θέση βραχίονα στο τραπέζι-στόχο πριν τη διαδικασία σάρωσης (όψη #1).



Εικόνα 2.39: Θέση βραχίονα στο τραπέζι-στόχο πριν τη διαδικασία σάρωσης (όψη #2).



Εικόνα 2.40: Θέση βραχίονα στο τραπέζι-στόχο πριν τη διαδικασία σάρωσης (όψη #3)

Τρία (3) δευτερόλεπτα μετά την λήψη αρχικής θέσης του βραχίονα, το Huksy διορθώνει τον προσανατολισμό του με τη γνωστή διαδικασία zig-zag, προκειμένου να φέρει το όχημα όσο το δυνατόν πιο κάθετα γίνεται στο τραπέζι.

Η στιγμή της σάρωσης έφτασε.

Αυτή τη χρονική στιγμή, η κάμερα δίνει την πληροφορία που φαίνεται από την εικόνα 2.41 παρακάτω.



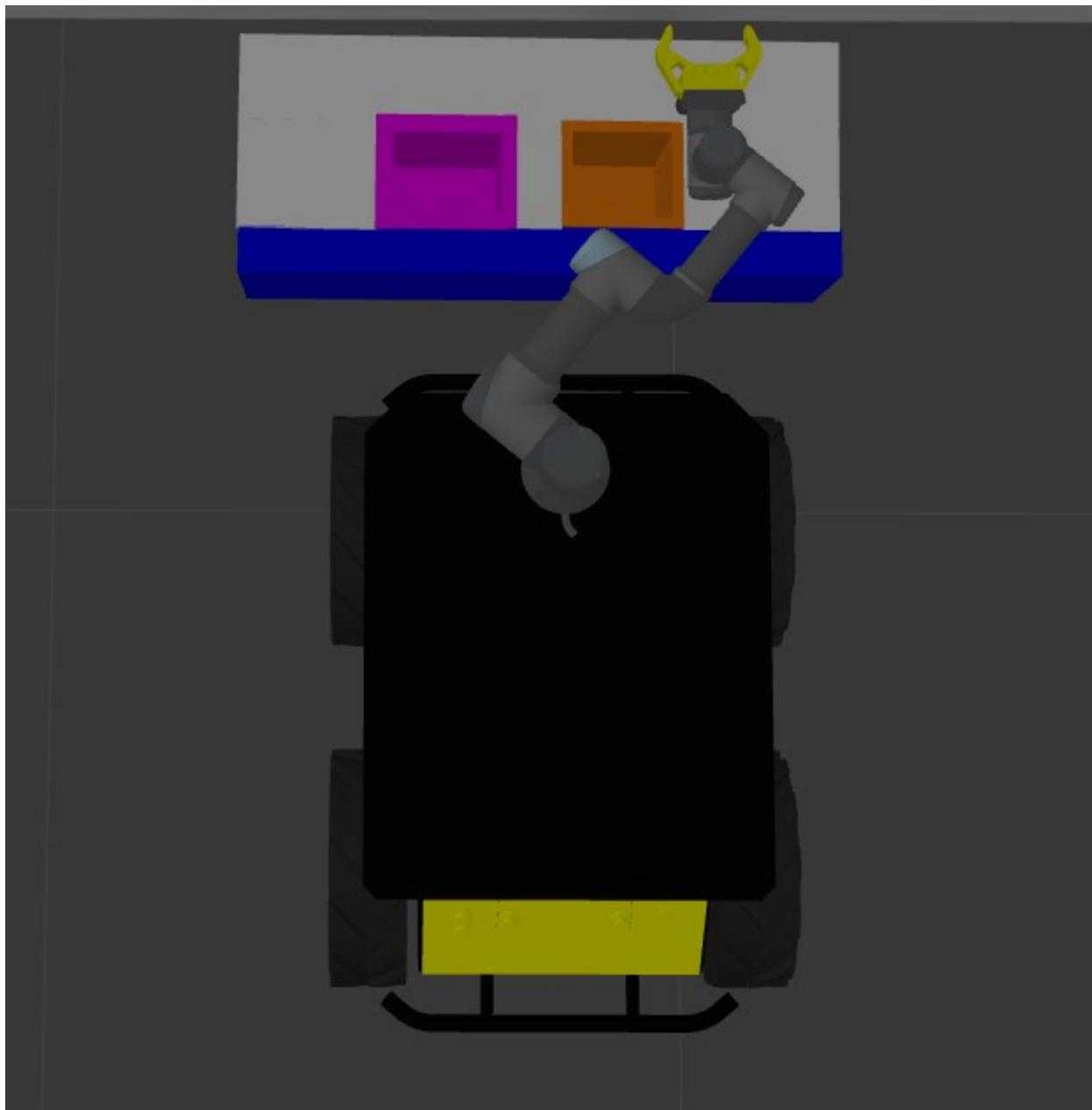
Εικόνα 2.41: Αναπαράσταση εικόνας από την κάτω μπροστινή καμερα του UR3.

Αρχικά, πραγματοποιείται μία αυτόματη σάρωση προς τα δεξιά έως ότου εκταθεί πλήρως ο βραχίονας ή μέχρι να βρεί την υποδοχή του κατάλληλου χρώματος.

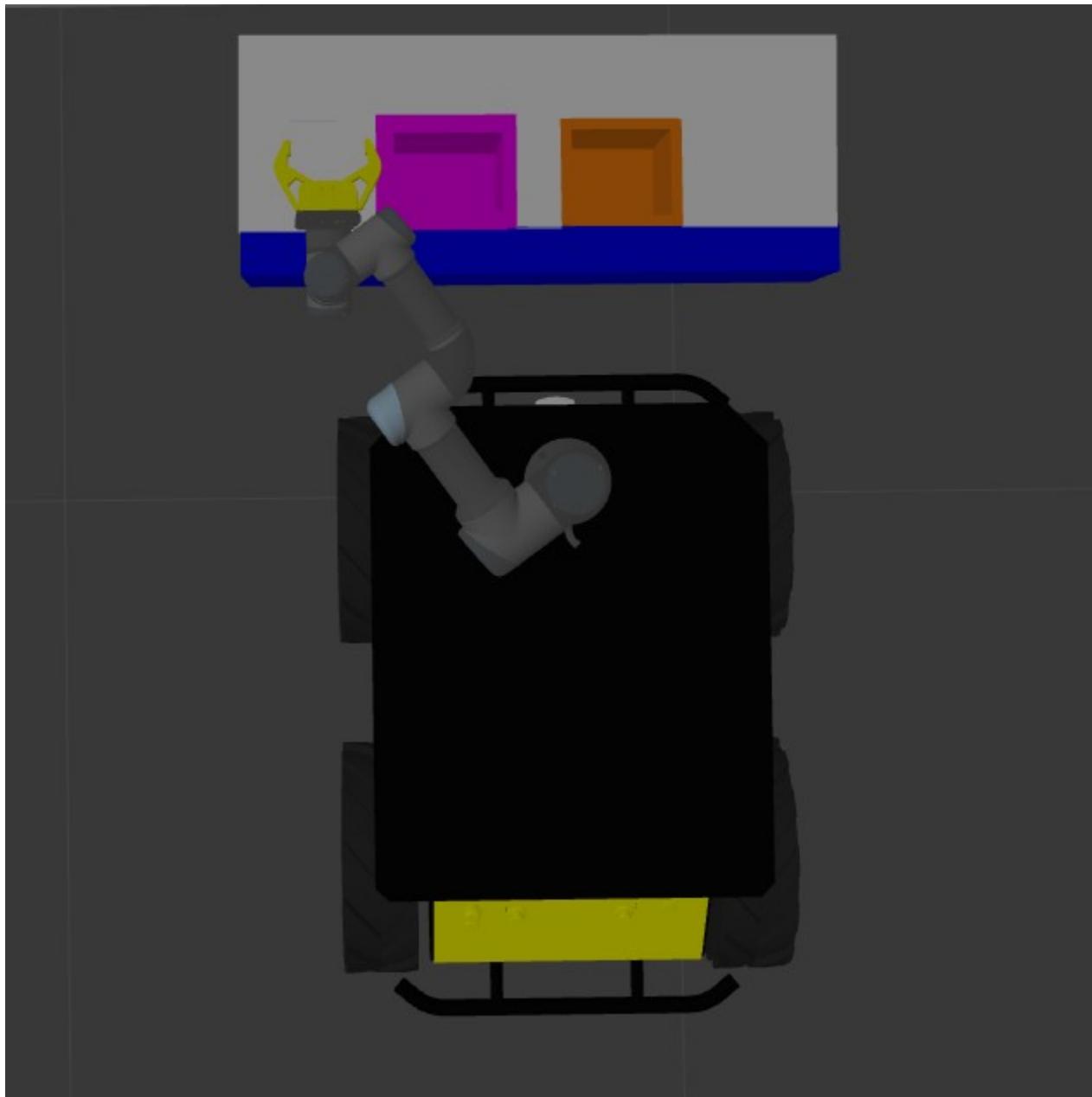
Αν εκταθεί πλήρως και η σωστή υποδοχή δεν έχει βρεθεί ακόμα, τότε συνεχίζει τη σάρωση προς την αντίθετη κατεύθυνση, δηλαδή προς τα αριστερά έως ότου εκταθεί πλήρως ή μέχρι να βρει την σωστή υποδοχή. Οι κινήσεις του ΤΣΔ είναι αργές, δηλαδή πραγματολποιεί βήματα 0.005 μέτρων κάθε 0.5 δευτερόλεπτο, ώστε να προλάβει να βρει τη σωστή υποδοχή.

Αν ολοκληρώσει και τις 2 κινήσεις σάρωσης (δεξιά και αριστερά) και δεν έχει βρεθεί η υποδοχή με χρώμα ίδιο με το χρώμα του αντικειμένου που κρατάει UR3, τότε αποτυγχάνει η αποστολή και η εφαρμογή τερματίζεται με το μήνυμα: "Mission failed ! Target not found."

Παρακάτω, στις εικόνες 2.42 και 2.43 φαίνεται η πλήρης εκτάση του UR3 κατά τη διαδικασία της αυτόματης σάρωσης (μία προς τα δεξιά και μία προς τα αριστερά).



Εικόνα 2.42: Πλήρης έκταση βραχίονα προς τα δεξιά κατά τη διαδικασία αυτόματης σάρωσης μέσω της κάτω μπροστινής κάμερας του UR3.



Εικόνα 2.43: Πλήρης έκταση βραχίονα προς τα αριστερά κατά τη διαδικασία αυτόματης σάρωσης μέσω της κάτω μπροστινής κάμερας του UR3.

Όσο πραγματοποιείται η παραπάνω αυτόματη σάρωση, αν κάποια στιγμή ο βραχίονας αντιληφθεί μέσω της κάμερας την υποδοχή με το σωστό χρώμα, τότε κατευθύνεται προς το κέντρο της υποδοχής αυτής. Ο αλγόριθμος κατεύθυνσης προς το κέντρο της υποδοχής μοιάζει αρκετά με αυτόν που χρησιμοποιήθηκε για να κατευθυνθεί το ΤΣΔ προς το κέντρο του αντικειμένου με σκοπό την αρπαγή του, όπως περιγράφηκε αναλυτικά από το υποκεφάλαιο 2.8 και συγκεκριμένα την παράγραφο με τίτλο:

“Έλεγχος λειτουργίας manual scanning”.

Το μοναδικό που αλλάζει σε αυτόν τον αλγόριθμο είναι το γεγονός ότι χρησιμοποιείται άλλος άξονας κίνησης και σάρωσης.

Ο αλγόριθμος της παραγράφου 2.8 ασχολούταν με σάρωση αντικειμένου που βρισκόταν μπροστά του, άρα ο UR3 χρησιμοποιεί τους άξονες Y-Z, όπου ο Y είναι για κινήσεις δεξιά και αριστερά και ο Z για κινήσεις πάνω και κάτω., έως ότου το κέντρο του αντικειμένου συμπίπτει με το κέντρο του frame.

Αντίστοιχα, στον αλγόριθμο αυτής της παραγράφου η υποδοχή βρίσκεται κάτω από τη κάμερα (όχι μπροστά, όπως πριν) , οπότε ο UR3 χρησιμοποιεί τους άξονες X-Z όπου ο X είναι για κινήσεις μπροστά και πίσω και ο Y για κινήσεις δεξιά και αριστερά, έως ότου το κέντρο του αντικειμένου συμπίπτει με το κέντρο του frame.

Ας περιγράψουμε τον αλγόριθμο και ας δούμε παρακάτω μερικές εικόνες που μαρτυρούν την περιγραφή για τον αλγόριθμο που ακολουθεί.

Μόλις ολοκληρωθεί η διαδικασία της αυτόματης σάρωσης, πλέον μόνο ένα μέρος ή ολόκληρη η υποδοχή φαίνεται στο frame της κάμερας.

Ο βραχίονας συνεχίζει να κινείτε από τη στιγμή που έγινε ορατή η υποδοχή στο frame από τη διαδικασία αυτόματης σάρωσης που προηγήθηκε. Όπως είπαμε πρέπει να φέρουμε το ΤΣΔ του βραχίονα σε τέτοια θέση, ώστε το κέντρο της υποδοχής να συμπίπτει με το άσπρο κυκλάκι, το οποίο είναι τοποθετημένο στο κέντρο του frame της κάμερας ,οπως φαίνεται από την εικόνα 2.44 .

Για το σκοπό αυτόν, θα χρειαστούν οι συντεταγμένες της πάνω αριστερής και κάτω δεξιάς ακμής του περιγράμματος της υποδοχής, καθώς και οι συντεταγμένες του κέντρου του frame, τα οποία είναι γνωστά. Όλοι οι υπολογισμοί για την διαδικασία να έρθει η υποδοχή στο κέντρο του frame (άσπρο κυκλάκι) περιγράφονται αναλυτικά παρακάτω, τόσο με λόγια όσο και με την εικόνα 2.25.

Αρχικά, παίρνουμε την απόλυτη τιμή της διαφοράς μεταξύ της οριζόντιας συντεταγμένης X_c και αριστερής ακμής X_1 .

Επιπλέον, παίρνουμε την απόλυτη τιμή της διαφοράς μεταξύ της οριζόντιας συντεταγμένης X_c και δεξιάς ακμής X_2 .

Υστερα, κάνουμε σύγκριση μεταξύ αυτών των δύο τιμών, δηλαδή $d_{lx} = |X_c - X_1|$

και $d2x = |Xc - X2|$.

Αν $d1x < d2x$, τότε γίνεται αντιληπτό ότι το κέντρο της υποδοχής βρίσκεται δεξιότερα του κεντρου του frame, ενώ αν $d1x > d2x$, τότε βρίσκεται αριστερότερα του κέντρου, κατά τον οριζόντιο άξονα X.

Αντίστοιχα, το ίδιο γίνεται και με τις κάθετες συντεταγμένες, δηλαδή $d1y = |Yc - Y1|$ και $d2y = |Yc - Y2|$.

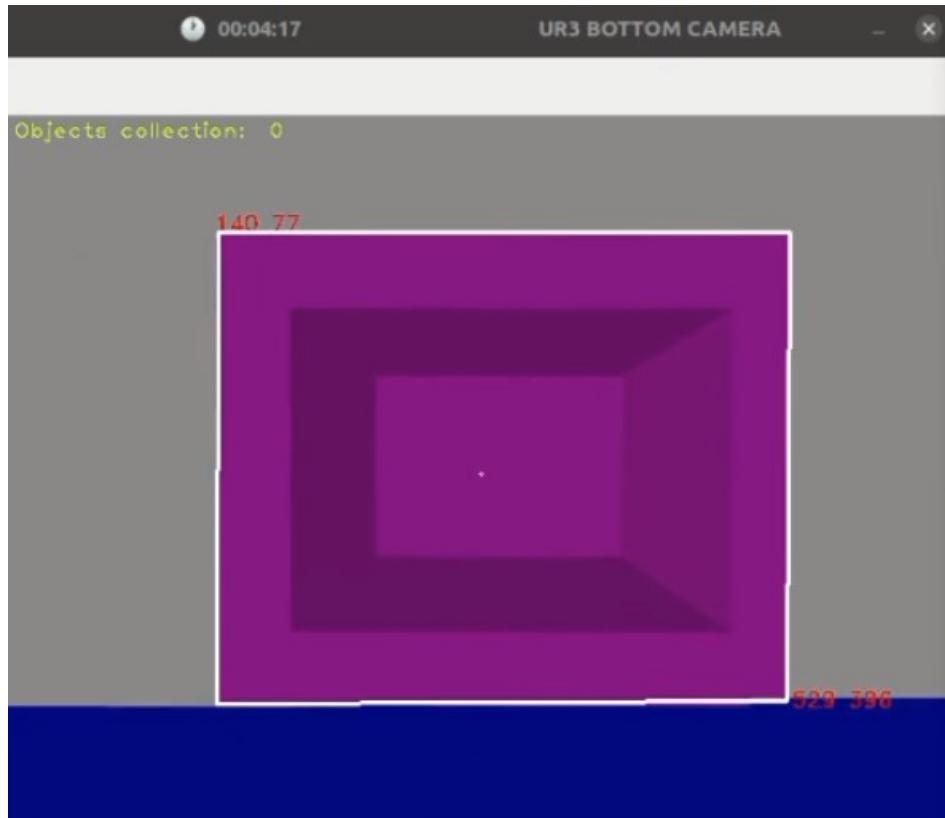
Αν $d1y < d2y$, τότε γίνεται αντιληπτό ότι το κέντρο της υποδοχής βρίσκεται χαμηλότερα του κεντρου του frame, ενώ αν $d1y > d2y$, τότε βρίσκεται υψηλότερα του κέντρου, κατά τον κάθετο άξονα Y.

Αυτή είναι η κύρια λογική αυτής της λειτουργίας, ώστε να καταλάβουμε που βρίσκεται η υηποδοχή σε σχέση με το κέντρο του παραθύρου.

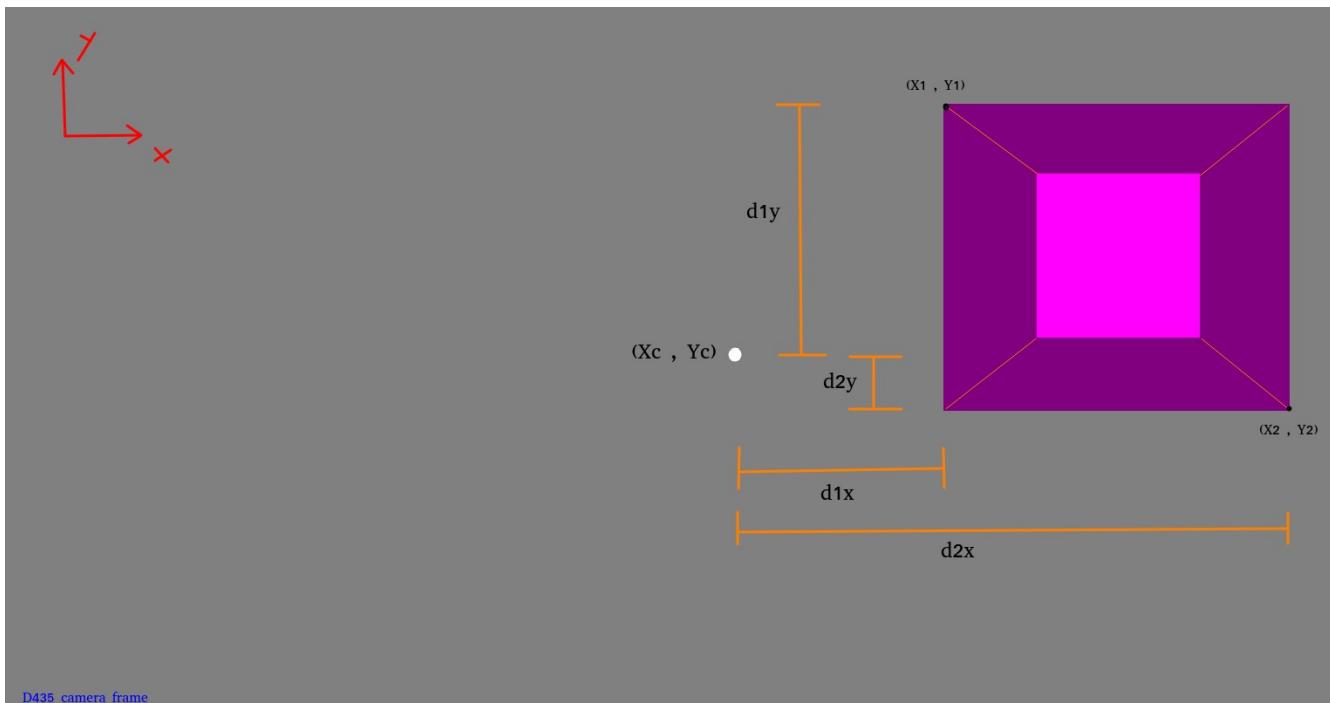
Οπότε, με όριο να εκταθεί πλήρως ο βραχίονας προς τα δεξιά ή αριστερά, εκτελείται η εξής διαδικασία:

Γίνεται μετακίνηση του ΤΣΔ του βραχίονα κατά 0.005 μέτρα ανά 0.5 δευτερόλεπτα προς την κατεύθυνση της υποδοχής (δεξιά ή αριστερά), καθώς και 0.005 μέτρα ανα 0.5 δευτερόλεπτα (μπροστά ή πίσω) στον 3D χώρο.

Όπως αναφέρθηκε προηγουμένως, η υποδοχή μπορεί να συμπέσει στο κέντρο του παραθύρου με μία μικρή απόκλιση των 10 pixels κατά τους άξονες X και Y της 2D εικόνας, για αυτό και τα βήματα του βραχίονα είναι αρκετά μικρά, δηλαδή 0.005 μέτρα / 0.5 δευτερόλεπτα, ώστε να προλάβει να σταματήσει αρκετά ικανοποιητικά στο στόχο. Η ανοχή αυτή υπάρχει, διότι δεν επηρεάζει το ΤΣΔ, δηλαδη την αρπάγη, να αφήσει το αντικείμενο εντός της υποδοχής με επιτυχία. Τη στιγμή αυτή, ο βραχίονας σταματάει οποιαδήποτε κίνηση. Εφόσον η υποδοχή συμπέσει στο κέντρο του παραθύρου, το μεγαλύτερο ποσοστό της σάρωσης έχει ολοκληρωθεί.

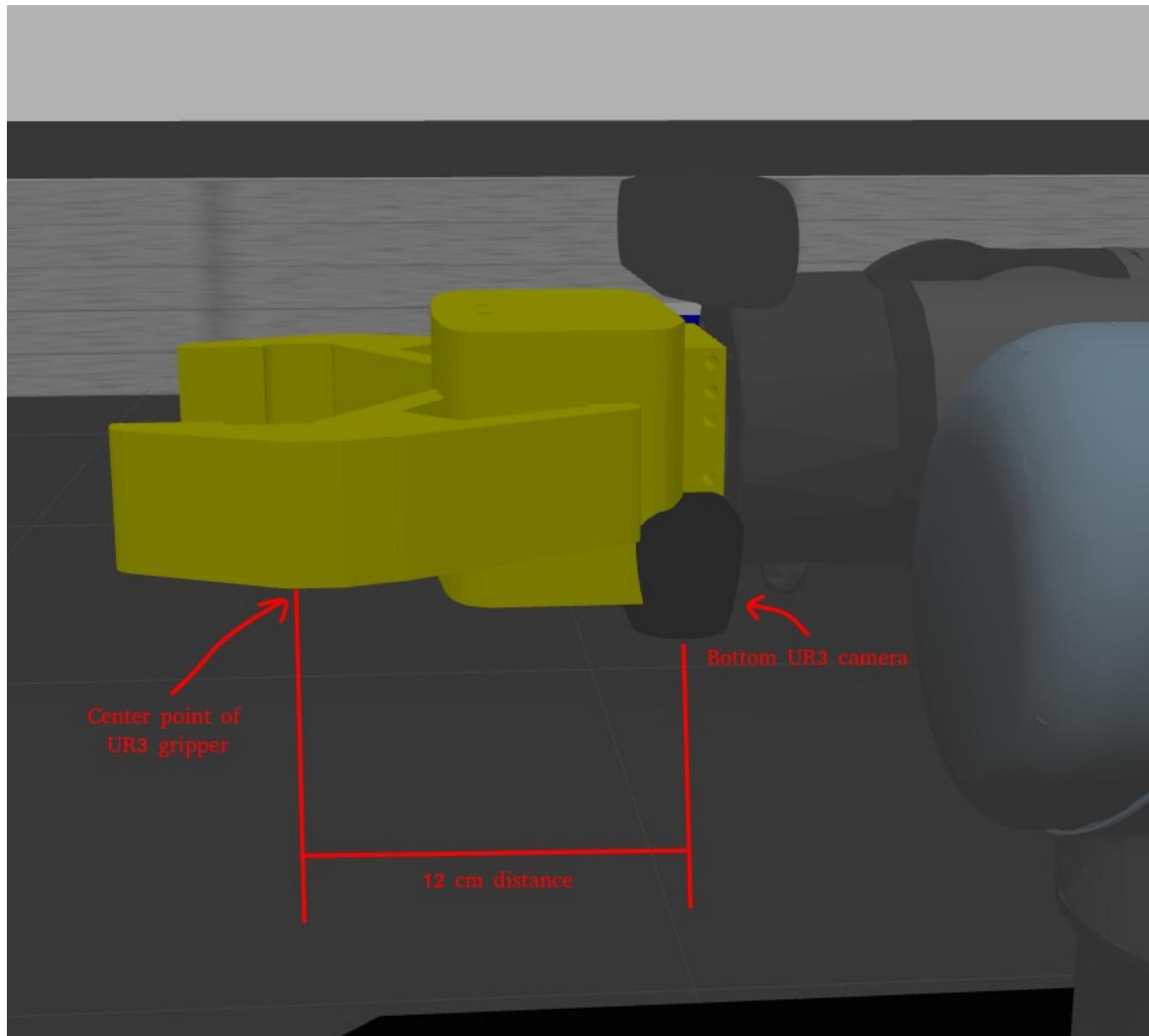


Εικόνα 2.44: Αναπαράσταση κέντρου υποδοχής στο κέντρο του frame (άσπρο κυκλάκι).



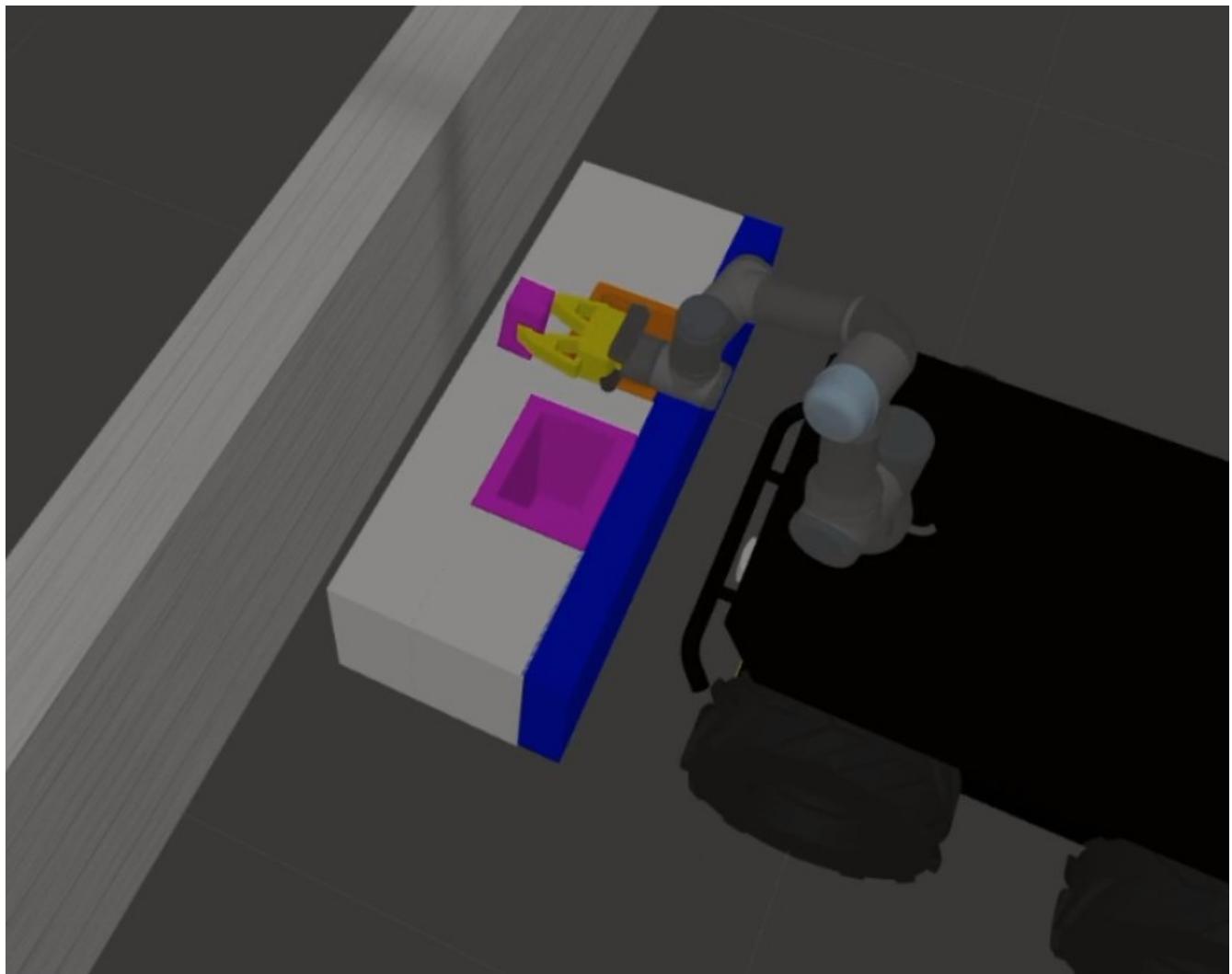
Εικόνα 2.45: Γραφική απεικόνιση frame και υπολογισμοί σάρωσης για την υποδοχή.

Τη στιγμή αυτή, η υποδοχή βρίσκεται στο κέντρο του παραθύρου. Έπειτα, ο βραχίονας πραγματοποιεί μία κίνηση προς τα πίσω κατά 0.12 μέτρα (12 εκτοστά), διότι υπάρχει μία απόσταση 12 εκατοστών μεταξύ κάμερας και κέντρο ΤΣΔ βραχίονα, όπως φαίνεται από την εικόνα 2.46 παρακάτω.

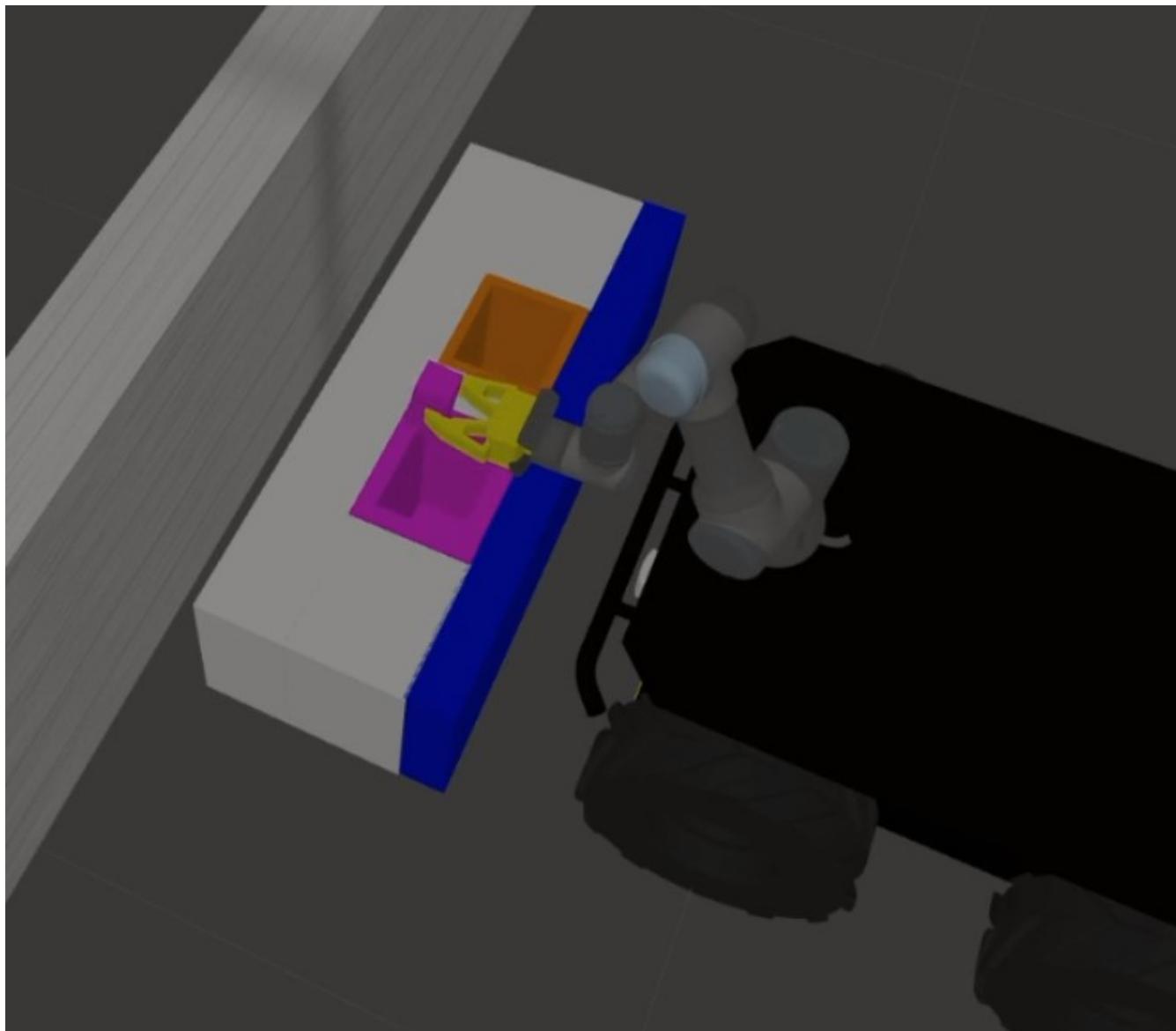


Εικόνα 2.46: Γραφική αναπάρασταση απόστασης μεταξύ κέντρου ΤΣΔ και κάτω μπροστινής κάμερας βραχίονα.

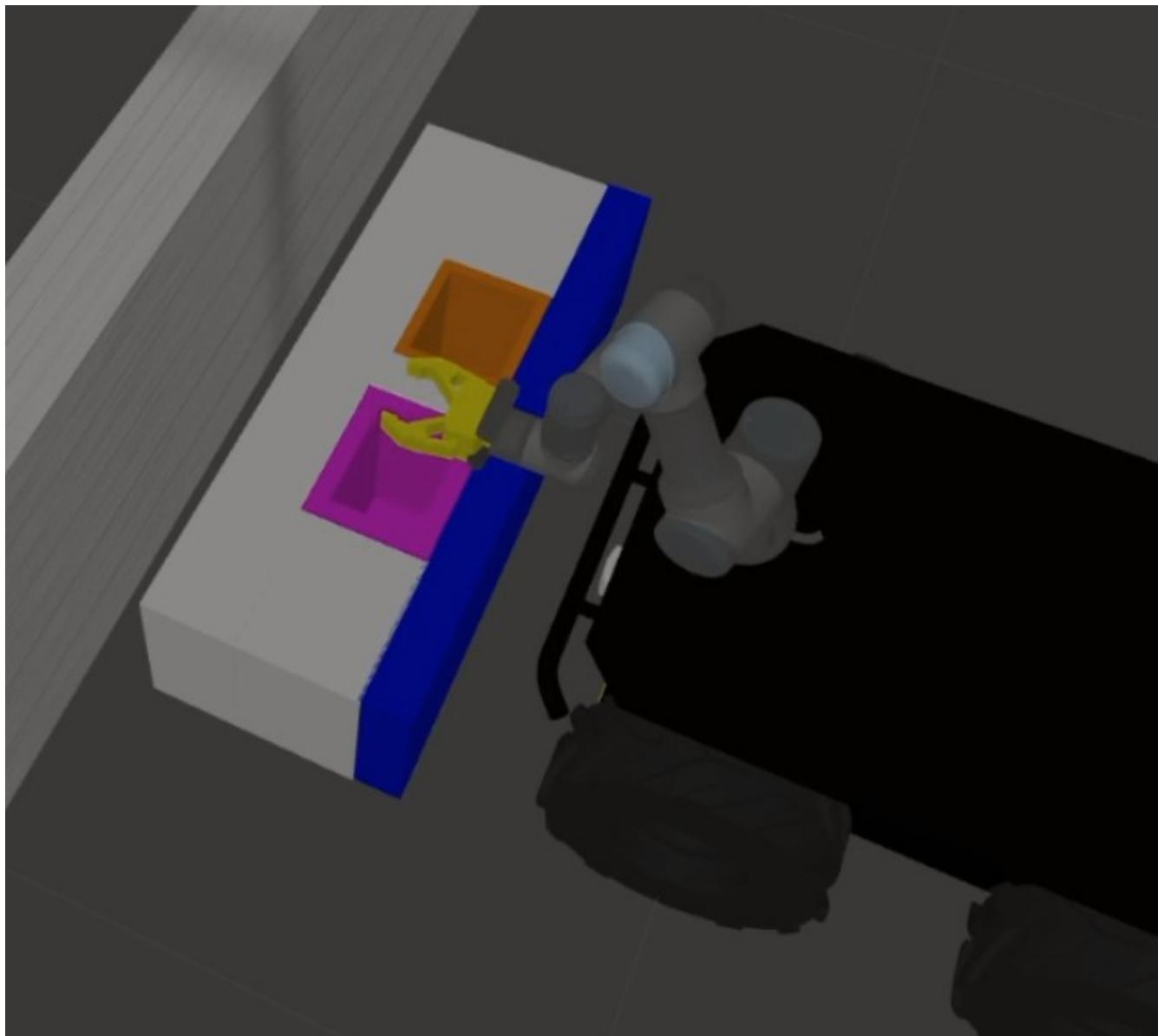
Στη συνέχεια της περιγραφής, ο UR3 κατεβαίνει σε ύψος 0.45 μέτρων (45 εκατοστών). Έπειτα, ανοίγει τελείως η αρπάγη με τιμή δύναμης λαβής ίση με 0 και με αυτό το τρόπο τοποθετείται το αντικείμενο στην αντίστοιχη υποδοχή. Τέλος, το ρομποτικό όχημα Husky κάνει οπισθοδρόπμηση για να συνεχίσει την αποστολή του στο χώρο. Οι παρακάτω εικόνες αποδεικνύουν την παραπάνω διατύπωση.



Εικόνα 2.47: Η αρπάγη σε ύψος 60 εκατοστών από το έδαφος πάνω από το κέντρο της μωβ υποδοχής.



Εικόνα 2.48: Η αρπάγη σε ύψος 45 εκατοστών από το έδαφος πάνω από το κέντρο της μωβ υποδοχής.



Εικόνα 2.49: Η αρπάγη αφήνει το αντικείμενο από ύψος 45 εκατοστών από το έδαφος να εισέλθει στην αντίστοιχη υποδοχή.

Μέχρι στιγμής, έχει ολοκληρωθεί μία πλήρης διαδικασία της αποστολής, δηλαδή την συνεργασία των Husky και Ur3 με απώτερο σκοπό την διαλογή αντικειμένων. Η αποστολή συνεχίζεται έως ότου διαλεχθούν όλα τα αντικείμενα που βρίσκονται στο χώρο.

2.13 ΑΛΓΟΡΙΘΜΟΣ ΚΑΤΩ ΚΑΜΕΡΑΣ UR3

Η υποενότητα αυτή περιγράφει αναλυτικά τη διαδικασία με την οποία η κάτω κάμερα του βραχίονα αναγνωρίζει υποδοχές, καθώς και τα χρώματα αυτών, με σκοπό ο βραχίονας να κατευθυνθεί προς αυτά βάσει σχήματος και χρώματος (η διαδικασία κατεύθυνσης διατυπώθηκε στο υποκεφάλαιο 2.12).

Θα περιγράψουμε τον αλγόριθμο αναγνώρισης χρωμάτων και σχήματος.

Στο υποκεφάλαιο αυτό, μπορούμε να κατανοήσουμε πλέον τη σημαντικότητα αναγνώρισης χρωμάτος ενός αντικειμένου από τα προηγούμενα βήματα, προκειμένου αυτό να εισέλθει στην αντίστοιχη υποδοχή.

Όπως έχουμε εξηγήσει και σε προηγούμενα υποκεφάλαια, με την έναρξη της αποστολής αναδύεται ένα παράθυρο, το οποίο αναπαριστά σε πραγματικό χρόνο το περιβάλλον μίας από τις τρεις (3) διαφορετικές κάμερες των δύο (2) ρομπότ. Ουσιαστικά, αυτό το παράθυρο είναι τα “μάτια” των καμερών. Συγκεκριμένα, ο αλγόριθμος αυτός χρησιμοποιεί τη κάτω μπροστινή κάμερα του βραχίονα UR3, διότι αυτή “βοηθάει” τον βραχίονα να κατευθυνθεί προς μία συγκεκριμένη υποδοχή, “κοιτώντας” προς τα κάτω (έδαφος). Παρακάτω, λοιπόν, εξηγούμε αναλυτικά όλη τη διαδικασία.

Καταρχάς, δημιουργούμε το παράθυρο του γραφικού περιβάλλοντος με τη βοήθεια της OpenCV. Οι διαστάσεις του είναι (640 , 480), δηλαδή 640 οριζόντια pixels για κάθε γραμμή και 480 κάθετα pixels για κάθε στήλη. Συνολικά, έχουμε $640 \times 480 = 307.200$ pixels για την εικόνα. Επίσης, μετασχηματίζουμε την εικόνα από τον χρωματικό χώρο BGR σε HSV. Το πλεονέκτημα έχει αναφερθεί στο υποκεφάλαιο 1.6.1.2, στο οποίο αναπτύσσεται ο αλγόριθμος κάμερας του Husky.

Έπειτα, δημιουργούμε έναν μικρό άσπρο κύκλο στο κέντρο του frame (βλέπε εικόνα 2.25), όπου σε αυτό το σημείο αναμένουμε να συμπέσει το κέντρο της υποδοχής.

Αρχικά, χρησιμοποιούμε μία ειδική μέθοδο της OpenCV για τη δημιουργία μιας μάσκας, η οποία κρατάει τα pixels που βρίσκονται μεταξύ των χρωματικών ορίων στο χώρο HSV. Η μάσκα περιέχει τιμές 0 και 1, όπου το 1 αντιστοιχεί στα pixels που είναι εντός των ορίων χρώματος και το 0 σε αυτά που βρίσκονται εκτός. Συνολικά, αυτή η διαδικασία εκτελεί ανίχνευση χρώματος σε μία εικόνα χρησιμοποιώντας τον χρωματικό χώρο HSV και

δημιουργεί μιά μάσκα που κρατάει μόνο τα pixels που έχουν χρώμα εντός συγκεκριμένων ορίων.

Μετά, εκμεταλλευόμαστε τη μάσκα που δημιουργήθηκε στο προηγούμενο βήμα για την ανίχνευση των συνεχόμενων περιοχών με χρώμα που πληρούν τα καθορισμένα χρωματικά όρια. Χρησιμοποιείται μία ειδική μέθοδος της OpenCV για την εύρεση των συνεχόμενων περιοχών στη μάσκα. Ουσιαστικά, χρησιμοποιούμε μία ειδική μέθοδο για να προσεγγίσουμε το περίγραμμα με ένα πολύγωνο και στη συνέχεια γίνεται έλεγχος εάν το πολύγωνο που προσεγγίστηκε έχει τέσσερις πλευρές. Αν ναι, θεωρείται ότι βρέθηκε ένας στόχος (ένα ορθογώνιο). Στη συνέχεια, σχηματίζουμε το ορθογώνιο πάνω στο frame χρησιμοποιώντας μία μέθοδο της OpenCV και το αποτέλεσμα που προκύπτει αναπαρίσταται στην εικόνα 2.44 παραπάνω.

2.14 TOPICS UR3

Ο βραχίονας UR3 χρησιμοποιεί τα εξής topics:

"/g_d435/rgb/g_image_raw" για άντληση πληροφοριών από τη πάνω κάμερα του.

"/g_d435_down/rgb/g_image_raw_down" για άντληση πληροφοριών από τη κάτω κάμερα του.

"/g_d435/depth/g_image_raw" για πληροφορίες σχετικά με το βάθος της πάνω κάμερας.

"arm_controller/command" για αποστολή εντολών για τον έλεγχο του βραχίονα.

"/rh_p12_rn_position/command" για αποστολή εντολών για τον έλεγχο της αρπάγης.

ΚΕΦΑΛΑΙΟ 3

Έλεγχοι και συμπεράσματα

3.1 ΕΠΙΤΥΧΙΕΣ & ΑΠΟΤΥΧΙΕΣ ΑΠΟΣΤΟΛΗΣ

Στη συγκεκριμένη αποστολή των δύο ρομπότ μπορεί να υπάρξει είτε επιτυχία είτε αποτυχία της διανομής αντικειμένων στο στόχο τους.

● ΕΠΙΤΥΧΙΑ

Η αποστολή έχει τερματιστεί με επιτυχία όταν το ρομποτικό όχημα Husky έχει κατευθυνθεί σε όλα τραπέζια που υπήρχαν αντικείμενα και ο βραχίονας UR3 δεν αντικρίζει πλέον αντικείμενα μέσω της σάρωσης του. Τότε καταλαβαίνουμε ότι δεν υπάρχουν πλέον αντικείμενα προς λήψη, δηλαδή έχουν τοποθετηθεί όλα στις κατάλληλες υποδοχές τους και έτσι η αποστολή εξετελέσθη με επιτυχία.

● ΑΠΟΤΥΧΙΑ

Η αποστολή αποτυγχάνει συνήθως όταν ο βραχίονας δεν μπορεί να αφήσει ένα αντικείμενο μεσα στην υποδοχή ενός τραπεζιού, με αποτέλεσμα το αντικείμενο να πέφτει στο δάπεδο. Αυτό συμβαίνει λόγω των σφαλμάτων που γίνονται και διογκώνονται από τις μετακινήσεις ή περιστροφές του Husky. Ύστερα, τα ρομπότ συνεχίζουν την εργασία/πορεία τους κανονικά, αλλά το αντικείμενο έχει ήδη πέσει, οπότε η διανομή έγινε με αποτυχία.

Άλλη πιθανότητα αποτυχίας εμφανίζεται όταν ο βραχίονας δεν μπορεί να πιάσει ένα αντικείμενο από το τραπέζι, λόγω μικρών σφαλμάτων κατά τη σάρωση, αλλά αυτή η πιθανότητα είναι πολύ μικρότερη από τη προηγούμενη που αναπτύξαμε στη προηγούμενη παράγραφο.

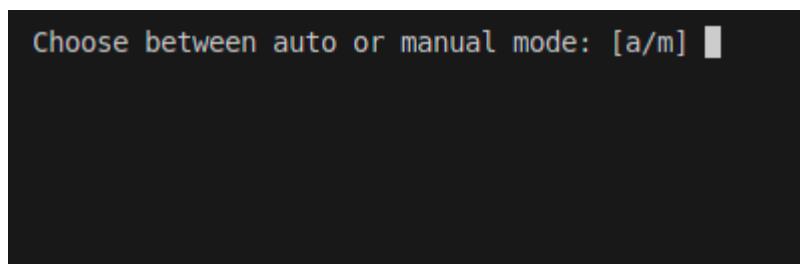
3.2 MENU & ΕΛΕΓΧΟΣ ΤΩΝ ΡΟΜΠΟΤ

Η εφαρμογή στηρίζεται τόσο σε χειροκίνητη όσο και σε αυτόματη λειτουργία.

Το άτομο που χειρίζεται την εφαρμογή έχει τη δυνατότητα να επιλέξει ανάμεσα σε δύο βασικές λειτουργίες πριν ξεκινήσει το πρόγραμμα.

Αν θέλει να πραγματοποιήσει μία ικανοποιητική συνεργασία και μία εντελώς αυτόματη πλοϊγηση των ρομπότ στο χώρο, δηλαδή να εκτελέσουν τις εργασίες τους μόνα τους, χωρίς ανθρώπινο παράγοντα για βοήθεια, τότε προτείνεται να επιλέξει τη λειτουργία **AUTO MODE**. Η λειτουργία αυτή έχει ως στόχο την αυτόματη συνεργασία μεταξύ των δύο ρομπότ με σκοπό να φέρουν εις πέρας μία αποστολή, τη διανομή διάφορων αντικειμένων.

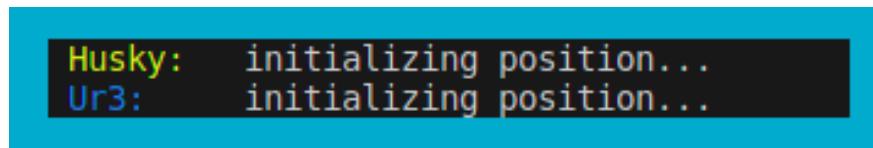
Από την άλλη πλευρά, ο χρήστης της εφαρμογής μπορεί να ενεργοποιήσει το **MANUAL MODE**, μέσω του οποίου μπορεί να μετακινήσει το τροχοφόρο Husky, το βραχίονα UR3, την αρπάγη του UR3, ανεξάρτητα, ακόμα και να σκανάρει με την εμπρόσθια ή την κάτω κάμερα του βραχίονα για αντικείμενα. Με αυτόν τον τρόπο, μπορεί να δοκιμάσει ως αρχάριος χρήστης όλες τις λειτουργίες τις οποίες παρέχει η εφαρμογή και να δει πως ανταποκρίνονται τα δύο ρομπότ στις εντολές του. Η εφαρμογή ξεκινάει, δίνοντας στον χρήστη τη δυνατότητα να επιλέξει ανάμεσα στις δύο λειτουργίες που αναφέρθηκαν προηγουμένως. Αμέσως μετά την επιλογή λειτουργίας, το πρόγραμμα ξεκινάει ανάλογα.



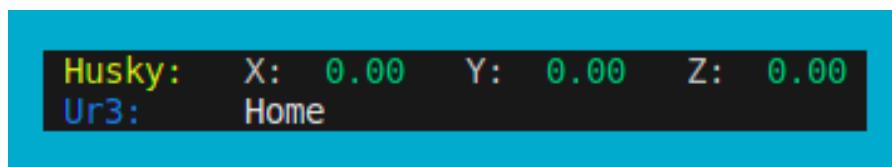
Εικόνα 3.1: Το αρχικό μήνυμα της εφαρμογής κατά την εκκίνηση.

Κατά την εκκίνηση, υπολογίζεται από τα topics του Husky η θέση του στο 3D χώρο του Gazebo. Ταυτόχρονα, ο βραχίονας UR3 παίρνει τη θέση εκκίνησης (Home position).

Όσο γίνονται αυτές οι ενέργειες, οι οποίες διαρκούν περίπου 5-6 δευτερόλεπτα, εκτυπώνονται στην οθόνη της εφαρμογής τα αντίστοιχα μηνύματα, όπως φαίνεται στην παρακάτω εικόνα.



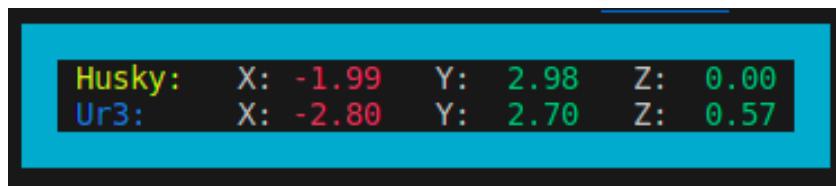
Εικόνα 3.2: Μηνύματα κατά τη διάρκεια αρχικοποίησης.



Εικόνα 3.3: Μηνύματα μετά το πέρας της αρχικοποίησης.

Όσο το Husky και οπ Ur3 μετακινούνται/περιστρέφονται στο χώρο, το μενού ανανεώνεται με τις αντίστοιχες τιμές (X,Y,Z) των 2 ρομπότ.

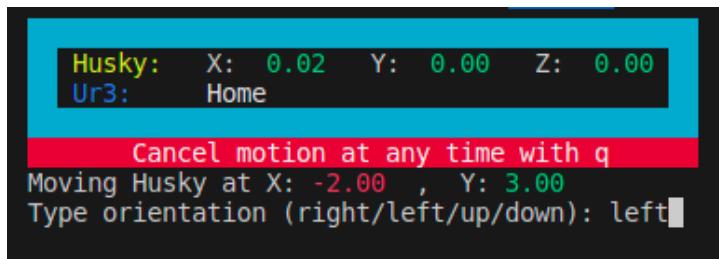
Ένα παράδειγμα θέσεων των 2 ρομπότ στο χώρο φαίνεται στις παρακάτω εικόνες.



Εικόνα 3.4: Θέσεις των 2 ρομπότ στο 3D χώρο (Μενού).

Οι είσοδοι δίνονται από το TERMINAL όπως φαίνεται στη παρακάτω εικόνα και ισχύει μόνο όταν ο ίδιος έχει ενεργοποιήσει το MANUAL MODE.

Αντίθετα, η ενεργοποίηση του AUTO MODE αυτοματοποιεί τη διαδικασία.



Εικόνα 3.5: Διαδικασία εισόδου τιμών (στο Husky).

ΒΙΒΛΙΟΓΡΑΦΙΑ

- [1] https://github.com/QualiaT/husky_ur3_simulator
- [2] https://www.youtube.com/watch?v=wDus2EKLg3s&ab_channel=bioMechatronicsLab
- [3] https://www.youtube.com/watch?v=t6EGMvL6lE0&t=1698s&ab_channel=NPTEL-NOCIITM
- [4] <https://repository.gatech.edu/server/api/core/bitstreams/e56759bc-92c8-43df-aa62-0dc47581459d/content>
- [5] <https://journals.sagepub.com/doi/10.1177/1461348419874925>
- [6] <https://www.mdpi.com/2218-6581/11/6/137>
- [7] https://www.youtube.com/watch?v=zc8b2Jo7mno&ab_channel=GuerrillaCG
- [8] https://www.youtube.com/watch?v=t71sQ6WY7L4&ab_channel=Pysource
- [9] <https://www.geeksforgeeks.org/reading-image-opencv-using-python/>
- [10] <https://www.geeksforgeeks.org/find-and-draw-contours-using-opencv-python/>
- [11] https://www.youtube.com/watch?v=fv6dLTEvl74&t=422s&ab_channel=RealPars
- [12] https://www.universal-robots.com/media/240787/ur3_us.pdf
- [13] <https://clearpathrobotics.com/husky-unmanned-ground-vehicle-robot/>
- [14] <https://github.com/JenniferBuehler/gazebo-pkgs/issues/9>
- [15] https://roboticscasual.com/ros-tutorial-control-the-ur5-robot-with-ros_control-tuning-a-pid-controller/
- [16] <http://wiki.ros.org/tf/Tutorials/Writing%20a%20tf%20listener%20%28Python%29>