Ατομική Διπλωματική Εργασία

ΕΦΑΡΜΟΓΕΣ ΡΟΜΠΟΤΙΚΟΥ ΒΡΑΧΙΟΝΑ

Ιωάννης Παντελή

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2020

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Εφαρμογές Ρομποτικού Βραχίονα

Ιωάννης Παντελή

Επιβλέπων Καθηγητής
Μάριος Δικαιάκος

Η Ατομική Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου

Μάιος 2020

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον επιβλέπων καθηγητή μου, Μάριο Δικαιάκο για την καθοδήγηση που μου προσέφερε κατά την διάρκεια εκπόνησης της διπλωματικής μου εργασίας.

Επίσης, θα ήθελα να ευχαριστήσω το Κέντρο Επιχειρηματικότητας του Πανεπιστημίου Κύπρου, για την συνεργασία που είχαμε αλλά και για την παροχή του απαραίτητου εξοπλισμού.

Περίληψη

Σκοπός της συγκεκριμένης διπλωματικής εργασίας, είναι η ανάπτυξη εφαρμογών για τον ρομποτικό βραχίονα Dobot Magician. Στην εποχή όπου ζούμε σήμερα, με την ραγδαία εξέλιξη της τεχνολογίας παρατηρείται η τάση για περισσότερη αυτοματοποίηση σε όλους τους τομείς της ζωής μας. Πλέον, μπορεί κανείς να συναντήσει ρομπότ παντού γύρω του, στην γεωργία, στις αυτοκινητοβιομηχανίες, σε εργοστάσια ανακύκλωσης, ακόμη και στα σχολεία ή στα ίδια μας τα σπίτια. Στα πλαίσια της παρούσας διπλωματικής εργασίας, έχουν αναπτυχθεί κάποιες εφαρμογές για την επίτευξη μερικών σεναρίων χρήσης ενός ρομποτικού βραχίονα, όπως η μεταφορά αντικειμένων από σταθερό χώρο ή από έναν ιμάντα μεταφοράς αντικειμένων. Για τον σκοπό αυτό έχει χρησιμοποιηθεί ο ρομποτικός βραχίονας Dobot Magician, Conveyor Belt, αισθητήρας υπέρυθρων, αισθητήρας χρώματος και κάμερα JeVois Smart Vision. Η ανάπτυξη των εφαρμογών για τον Dobot Magician έχει γίνει σε γλώσσα προγραμματισμού Python, ενώ τα modules που εκτελούνται στην JeVois Camera, είναι υλοποιημένα σε γλώσσα C++.

Αρχικά, έχει αναπτυχθεί ένα απλό πρόγραμμα για μετακίνηση αντικειμένων από σταθερό σημείο σε ένα καλάθι, και καταγράφηκαν διάφορα προβλήματα που προέκυψαν, όπως η αναγνώριση των αντικειμένων, οι έλεγχοι για την θέση του ρομποτικού βραχίονα, η ανάγκη για συνεχής τροφοδοσία αντικειμένων, αλλά και η μη ανάκτηση του αντικειμένου από τον βραχίονα.

Στην συνέχεια, αναπτύχθηκε πρόγραμμα το οποίο επέτρεπε την διασύνδεση του βραχίονα με τον ιμάντα μετακίνησης και τους δύο αισθητήρες-χρώματος και υπέρυθρων ούτως ώστε να είναι δυνατή η συνεχής τροφοδοσία αντικειμένων στον βραχίονα και η διαλογή των αντικειμένων βάση του χρώματος τους. Επιπρόσθετα, έχει δημιουργηθεί μέσω 3D Printing βάση για την JeVois Camera, ώστε να μπορεί να τοποθετηθεί σε οποιοδήποτε σημείο δίπλα από τον ιμάντα και να είναι συνεχώς σταθερή. Τέλος, πραγματοποιήθηκε η ανάπτυξη δυο προγραμμάτων για επικοινωνία της JeVois camera - Dobot και του Arduino -Dobot , ώστε να είναι εφικτή η χρήση από τον ρομποτικό βραχίονα των δυνατοτήτων της κάμερας και του αισθητήρα απόστασης.

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

1.1 Ο κλάδος της ρομποτικής

Στην σημερινή εποχή, με την ραγδαία εξέλιξη της τεχνολογίας παρατηρείται η τάση για περισσότερη αυτοματοποίηση σε όλους τους τομείς της ζωής μας. Ο κλάδος της ρομποτικής αφορά την κατασκευή και την δημιουργία εφαρμογών για τα ρομπότ, προσφέροντας έτσι λύσεις στα διάφορα προβλήματα που αντιμετωπίζει ο άνθρωπος κατά την διάρκεια εκτέλεσης των διάφορων εργασιών. Η ανάπτυξη της τεχνολογίας και ιδιαίτερα του τεχνολογικού εξοπλισμού (όπως αισθητήρων, αυξημένη υπολογιστική ισχύ, τεχνητή νοημοσύνη κλπ.) έχουν επιτρέψει στα ρομπότ να παρουσιάζουν ιδιαίτερη ευελιξία στην ανάπτυξη τους και συνεπώς στις χρήσεις που μπορούν να επιδείξουν.

Βρισκόμαστε στην εποχή της 4^{ης} βιομηχανικής επανάστασης,[1] όπου στα εργοστάσια παραγωγής γίνεται χρήση ασύρματης συνδεσιμότητας, αισθητήρων, IoT, Artificial Intelligence. Τα ρομπότ που εγκαθιστούνται ελέγχονται συνήθως μέσω ενός κεντρικού υπολογιστή, ενώ χρησιμοποιούν αισθητήρες απόστασης, αφής, όρασης, δύναμης και έχουν την ικανότητα να λαμβάνουν αποφάσεις κατά την διάρκεια εκτέλεσης μια εργασίας.

Παρουσιάζονται διάφορες προκλήσεις όσον αφορά την λειτουργία των ρομπότ. Μια βασική πρόκληση, είναι η απόδοση τόσο ενός ρομπότ σαν μονάδα, όσο και σε συνεργασία με άλλα ρομπότ που σκοπό έχουν να ολοκληρώσουν μια δουλεία. Εξετάζονται συνεπώς τρόποι, ούτως ώστε να βελτιωθεί η απόδοση τους, όσον αφορά την ταχύτητα λειτουργίας τους, την αποτελεσματικότητα, την μείωση του χρόνου επαναφοράς από σφάλματα κλπ.

Η δεύτερη πρόκληση, σχετίζεται με τον τομέα της ασφάλειας και του ελέγχου των ρομπότ στο περιβάλλον που είναι εγκατεστημένο. Είναι μείζονος σημασίας η ασφάλεια τόσο του προσωπικού που βρίσκεται στον χώρο εργασίας των ρομπότ, όπως επίσης και η διασφάλιση των ρομπότ από σφάλματα που μπορούν να προκύψουν κατά την εκτέλεση μιας εργασίας. Για τον σκοπό αυτό, αναπτύσσονται διάφορες τεχνικές ελέγχου των ρομπότ.

Τέλος, η τρίτη πρόκληση αφορά την ανάπτυξη των εφαρμογών όπου επιτυγχάνεται η καλύτερη δυνατή χρήση των εξαρτημάτων που πλαισιώνουν τα ρομπότ.[2]

Παρατηρούμε λοιπόν, ότι ο κλάδος της ρομποτικής σχετίζεται με αρκετούς άλλους κλάδους, όπως η πληροφορική, η ηλεκτρολογία, η μηχανολογία, τα μαθηματικά κοκ.

Ένα ρομπότ χωρίζεται σε τρία υποσυστήματα. Το πρώτο υποσύστημα είναι το μηχανολογικό και αποτελείται από την βάση του ρομπότ, τις αρθρώσεις, συστήματα κίνησης κ.α. Το δεύτερο υποσύστημα είναι το αισθητήριο, το οποίο συλλέγει πληροφορίες μέσω των διαφόρων αισθητήρων και οργάνων από το περιβάλλον. Το τρίτο σύστημα, αφορά την συλλογή και επεξεργασία των πληροφοριών που του δίδονται, ώστε να διοχετευτούν οι κατάλληλες εντολές στο ρομπότ για εκτέλεση τους.

1.2 Κατηγορίες Ρομπότ[3]

Τα σύγχρονα ρομπότ προσαρμόζουν τις δυνατότητες τους ανάλογα με τον εκάστοτε τομέα και εφαρμογή. Συνεπώς, ο πρώτος τρόπος για την κατηγοριοποίηση των ρομπότ είναι βάση της εφαρμογής τους, στις εξής βασικές κατηγορίες:

Καταναλωτικής Χρήσης: Τα συγκεκριμένα ρομπότ μπορούν να χρησιμοποιηθούν για να φέρουν εις πέρας καθημερινές δουλείες στο σπίτι ή ακόμα και για να μας προσφέρουν ψυχαγωγία. Τα τελευταία χρόνια ο συγκεκριμένος κλάδος παρουσιάζει πολύ μεγάλη ανάπτυξη παρουσιάζοντας συνεχώς εξελιγμένα ρομπότ, ικανά να λύνουν όλο και πιο πολύπλοκες δουλειές. Κάποια παραδείγματα τέτοιων ρομπότ είναι οι ρομποτικές σκούπες (Εικόνα 1.1α), παιχνίδια, ψηφιακοί βοηθοί με AI, ρομποτικές κουζίνες (Εικόνα 1.1b), καθαριστές πισίνων κ.α.

Εικόνα 1.1a: Dyson Ρομποτική Σκούπα[4]

Εικόνα 1.1b: Moley Ρομποτική Κουζίνα[5]

Εκπαιδευτικά: Ο τομέας των εκπαιδευτικών ρομπότ αφορά ρομπότ που μπορούν να χρησιμοποιηθούν σε σχολεία ή ακόμη και στα σπίτια μας, προσφέροντας γνώσεις σε διάφορους κλάδους, όπως η ρομποτική και η πληροφορική.

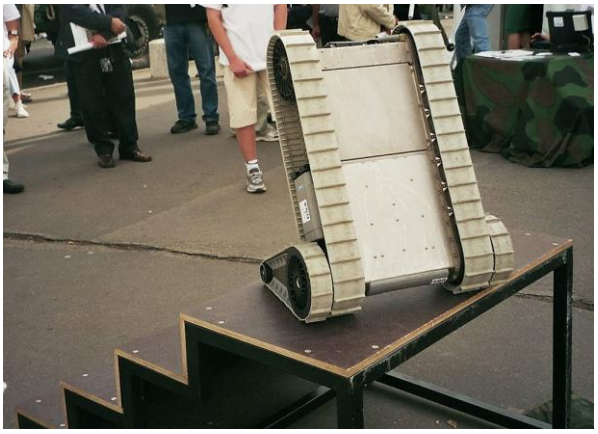
Εικόνα 1.2: Ο ρομποτικός δάσκαλος EMYS[6]

Ιατρικής: Ρομπότ που έχουν χρήση στην ιατρική, όπως ρομπότ που χρησιμοποιούνται σε εγχειρήσεις, ρομποτικοί εξωσκελετοί και βιονικά προσθετικά μέλη.

Εικόνα 1.3a: Da Vinci XI: Surgical System[7]

Εικόνα 1.3b: Εξωσκελετός Guardian Xo[8]

Στρατιωτικά: Τα ρομπότ που χρησιμοποιούνται στην στρατιωτική βιομηχανία. Περιλαμβάνονται ρομπότ που εξουδετερώνουν βόμβες, ρομπότ εξερεύνησης και μη επανδρωμένα αεροσκάφη.



Εικόνα 1.4: PackBot[9]

Υποβρύχια: Τα ρομπότ που είναι κατασκευασμένα για να εκτελούν υποθαλάσσιες λειτουργίες. Τέτοια ρομπότ χρησιμοποιούνται για εξερεύνηση των θαλασσών, χαρτογράφηση του βυθού των ωκεανών και για επιστημονικούς σκοπούς.

Εικόνα 1.5: BPAUV[10]

Βιομηχανικά: Τα ρομπότ που χρησιμοποιούνται σε βιομηχανίες, εκτελώντας συνήθως επαναλαμβανόμενες κινήσεις στην παραγωγική διαδικασία. Κάποιες χρήσεις τέτοιων ρομπότ είναι στην αυτοκινητοβιομηχανία στις συγκολλήσεις που γίνονται στα αυτοκίνητα, στο βάψιμο και στην συναρμολόγηση τους, σε αποθήκες για σκοπό μεταφοράς δεμάτων και σε γραμμές εμφιάλωσης.

Εικόνα 1.6α: Ρομπότ στο εργοστάσια της Tesla[11]

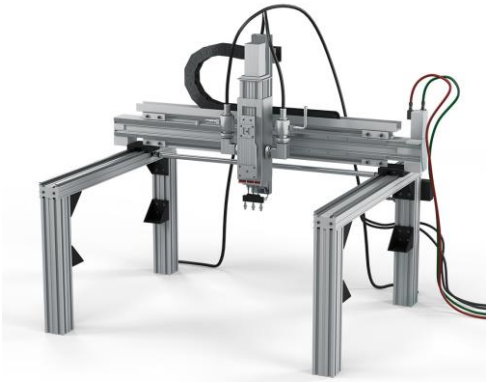
Εικόνα 1.6β: Ρομπότ σε αποθήκη της Amazon[12]

Ο δεύτερος τρόπος να κατηγοριοποιήσουμε τα ρομπότ είναι βάση των δυνατοτήτων κίνησης τους. Διαχωρίζονται λοιπόν στης εξής βασικές κατηγορίες.[13]

1. Σταθερά: Το είδος των ρομπότ αυτών αποτελείται από μια βάση που βρίσκεται σε σταθερή θέση. Τα περισσότερα βιομηχανικά ρομπότ εμπίπτουν στην συγκεκριμένη κατηγορία και συνήθως αποτελούνται από ένα βραχίονα που εκτελεί κινήσεις που είναι απαραίτητες για την ολοκλήρωση της εφαρμογής που σχεδιάστηκαν να εκτελούν. Επιπρόσθετα, σε αυτή την κατηγορία ρομπότ ανήκει

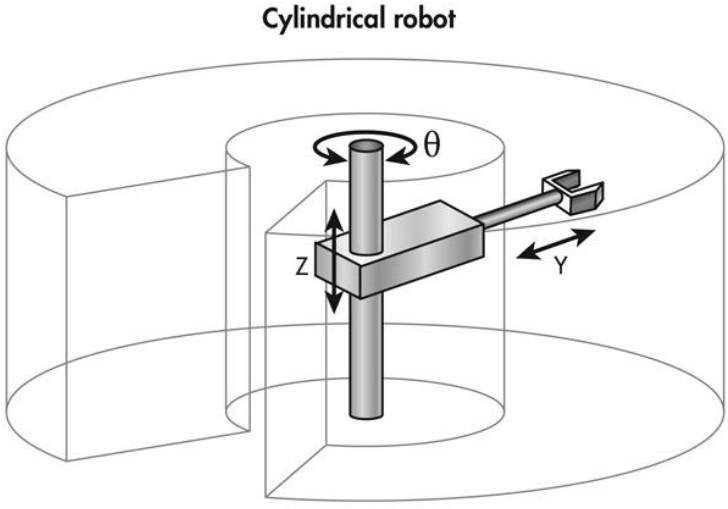
και ο ρομποτικός βραχίονας που θα παρουσιάσουμε σε αυτή την διπλωματική εργασία.

a. Καρτεσιανά: Τα καρτεσιανά ρομπότ έχουν τρείς γραμμικούς άξονες άρθρωσης που χρησιμοποιούν το καρτεσιανό σύστημα συντεταγμένων, εκφράζοντας το κάθε σημείο μετακίνησης του βραχίονα ως τρείς μεταβλητές, X,Y,Z. Έχουν συνήθως την δυνατότητα μεταφοράς μεγάλων φορτίων και χρησιμοποιούνται για μεταφορά αντικειμένων, φόρτωμα και ξεφόρτωμα υλικών κλπ.



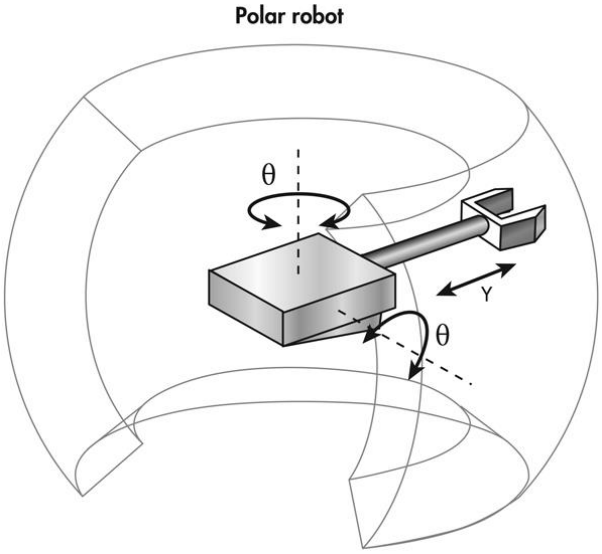
Εικόνα 1.7: Cartesian Robot[31]

b. Κυλινδρικά: Στα κυλινδρικά ρομπότ η άρθρωση είναι στροφική, επιτρέποντας στον βραχίονα να κινείται περιστροφικά γύρω από τον άξονα του. Χρησιμοποιούνται για συναρμολογήσεις, συγκολλήσεις κοκ.



Εικόνα 1.8: Σχεδιάγραμμα κυλινδρικού ρομπότ[32]

c. Σφαιρικά: Τα συγκεκριμένα ρομπότ αποτελούνται από δύο περιστροφικές αρθρώσεις και μια γραμμική. Αυτό τους επιτρέπει να είναι πιο ευέλικτα από τις δύο πιο πάνω κατηγορίες, επιτρέποντας τους έτσι να μπορούν να εφαρμοστούν σε περισσότερες περιπτώσεις, όπως σε συγκολλήσεις και injection molding.



Εικόνα 1.9: Σχεδιάγραμμα σφαιρικού ρομπότ[33]

d. SCARA: Αποτελούνται από δύο παράλληλες περιστροφικές αρθρώσεις και μια γραμμική. Συνεπώς, η κίνηση στους X,Y άξονες είναι πιο ευέλικτη, όμως πιο δύσκαμπτη στον Z άξονα. Χρησιμοποιούνται για μεταφορά αντικειμένων σε γραμμές παραγωγής, έλεγχο και συναρμολόγησή αντικειμένων κ.α.

Εικόνα 1.10: Scara ρομπότ από την KUKA[34]

e. Αρθρωτά: Τα αρθρωτά ρομπότ αποτελούνται από πολλούς περιστροφικούς άξονες κίνησης που είναι τοποθετημένα σε μια περιστρεφόμενη βάση. Η διάταξη τους είναι παρόμοια με ανθρώπινο χέρι, εφόσον αποτελούνται από ένα εργαλείο αρπαγής (στην περίπτωση του Dobot Magician Gripper ή Suction Cup) που είναι ανάλογο με παλάμη, την σύνδεση του εργαλείου αρπαγής με τον άνω βραχίονα όπως ο ανθρώπινος αγκώνας, και τέλος την σύνδεση του άνω βραχίονα με την βάση, όπως ο ώμος. Χρησιμοποιούνται σε πολλές περιπτώσεις, όπως συγκολλήσεις, συναρμολογήσεις, χειρισμός υλικών, βαφές κλπ.

Εικόνα 1.11: Αρθρωτό ρομπότ για μεταφορά αντικειμένων.[35]

2. Κινούμενα:
   a. Με τροχούς: Τα ρομπότ που κινούνται στο έδαφος και οφείλουν την κίνηση τους σε τροχούς. Συνήθως είναι απλά στην υλοποίηση τους και μπορούν να αποτελούνται ακόμη και από έναν τροχό. Ένα από τα πιο γνωστά ρομπότ με τροχούς είναι αυτό που η NASA ανάπτυξε για την εξερεύνηση του πλανήτη Άρη, το Rover.

Εικόνα 1.12: Το Curiosity Rover της NASA[14]

b. Με πόδια: Για την κίνηση τους χρησιμοποιούνται μηχανικά πόδια, ενώ παρουσιάζουν το πλεονέκτημα της κίνησης σε δύσκολα εδάφη με εμπόδια. Η υλοποίηση τους είναι όμως πιο πολύπλοκη, ενώ μπορούν να αποτελούνται από ένα μέχρι 8 μηχανικά πόδια. Τα πιο γνωστά ρομπότ με πόδια, είναι τα ανθρωποειδή ρομπότ, όπως η Sophia και διάφορα τετράποδα ρομπότ, όπως το BigDog της Boston Dynamics.

Εικόνα 1.13α:Το Humanoid ρομπότ Sophia[15]

Εικόνα 1.13β: Το τετράποδο ρομπότ BigDog[16]

Έκτος από τις δύο πιο πάνω κύριες κατηγορίες κινούμενων ρομπότ υπάρχουν και άλλες κατηγορίες, όπως ιπτάμενα ρομπότ, υποθαλάσσια ρομπότ και άλλα.

1.3 Σκοπός της εργασίας

Ο σκοπός της παρούσας πτυχιακής εργασίας είναι η ανάπτυξη εφαρμογών που να επιτρέπει την εισαγωγή και επεξεργασία των δεδομένων από τον βραχίονα, ώστε να γίνει εφικτή η χρήση του βραχίονα σε διάφορες εφαρμογές. Πιο συγκεκριμένα, έχει δημιουργηθεί λογισμικό σε γλώσσα Python, το οποίο επιτρέπει τον έλεγχο όλων των δυνατοτήτων του βραχίονα, όπως τη μετακίνηση του σε συγκεκριμένες συντεταγμένες, την ενεργοποίηση/απενεργοποίηση της βεντούζας απορρόφησης (ώστε να μεταφέρει ένα αντικείμενο), τον έλεγχο και εξαγωγή δεδομένων από τους αισθητήρες, και τον έλεγχο του ιμάντα μεταφοράς. Επιπρόσθετα, στο λογισμικό έχει αναπτυχθεί συνάρτηση για τον εισαγωγή δεδομένων από την JeVois Camera, όπως για παράδειγμα την

επεξεργασία μηνύματος σε περίπτωση εντοπισμού ενός αντικειμένου. Τέλος, στο λογισμικό έχει υλοποιηθεί συνάρτηση για την εισαγωγή δεδομένων από Arduino, όπου βρίσκεται συνδεδεμένος ένας αισθητήρας απόστασης για την εύρεση της απόσταση ενός αντικειμένου.

Με την διασύνδεση των εξωτερικών εξαρτημάτων(JeVois και αισθητήρας απόστασης) δίνεται η δυνατότητα επέκτασης των εφαρμογών που μπορούν να αναπτυχθούν στον Dobot Magician. Συγκεκριμένα, με την JeVois Smart Camera υπάρχει η δυνατότητα ταυτοποίησης των αντικειμένων που βρίσκονται στον ιμάντα, συνεπώς μπορούν να αναπτυχθούν εφαρμογές για διαχείριση αντικειμένων, όπως ο διαχωρισμός αντικειμένων για ανακύκλωση.

1.4 Δομή διπλωματικής εργασίας

Στο 2ο κεφάλαιο αναλύονται τα συστήματα που χρησιμοποιήθηκαν στην παρούσα διπλωματική εργασία. Συγκεκριμένα, στο 1ο υποκεφάλαιο παρατίθενται η εισαγωγή στον Ρομποτικό Βραχίονα Dobot Magician, όπου γίνεται αναφορά στις κατηγορίες που ανήκει, βασικές πληροφορίες για τις δυνατότητες του, αναλύεται ο τρόπος προγραμματιστικού ελέγχου και το API του, καθώς και οι εφαρμογές που θα μπορούσαν να αναπτυχθούν. Το 2ο υποκεφάλαιο αφορά την JeVois Camera, όπου παρουσιάζονται οι δυνατότητες τις, ο τρόπος προγραμματιστικού ελέγχου της, αλλά και το πως χρησιμοποιείται σε συνδυασμό με τον βραχίονα. Στο επόμενο υποκεφάλαιο γίνεται γενική αναφορά στο Arduino και τις δυνατότητες του, ενώ παρουσιάζονται οι αισθητήρες που χρησιμεύουν στην επέκταση του συστήματος και ο τρόπος που χρησιμοποιήθηκε με τον Dobot Magician. Το 4ο υποκεφάλαιο σχετίζεται με την τρισδιάστατη εκτύπωση, και πιο συγκεκριμένα αναλύεται ο τρόπος σχεδιασμού του ψηφιακού μοντέλου της βάσης που χρησιμοποιήθηκε με τον κάμερα JeVois και η εκτύπωση της.

Το 3ο κεφάλαιο αφορά την πειραματική διάταξη του συστήματος, δηλαδή την διασύνδεση του ρομποτικού βραχίονα με τον ιμάντα μεταφοράς και τους αισθητήρες του, με την κάμερα JeVois και με το Arduino και τον αισθητήρα απόστασης.

Στο 4ο κεφάλαιο αναλύεται η εκτέλεση των δύο σεναρίων χρήσης, και τα αποτελέσματα-παρατηρήσεις που εξάχθηκαν από τις δύο αυτές εκτελέσεις

Στο 5ο κεφάλαιο παρατίθενται τα συμπεράσματα που έχουν εξαχθεί από την παρούσα διπλωματική εργασία και η μελλοντική εργασία που μπορεί να πραγματοποιηθεί.

Τέλος, παρατίθεται η βιβλιογραφία και ο κώδικας που αναπτύχθηκε.

Κεφάλαιο 2

Περιγραφή Συστημάτων

2.1 Ο ρομποτικός βραχίονας Dobot Magician

2.1.1 Εισαγωγή στον Dobot Magician

Ο Dobot Magician[37] είναι ένας ρομποτικός βραχίονας που επιτρέπει την κίνηση σε 4 άξονες και ανήκει στην κατηγορία των εκπαιδευτικών ρομπότ. Επίσης, βάση των δυνατοτήτων κίνησης του ανήκει και στην κατηγορία των σταθερών ρομπότ. Έχει την δυνατότητα μεταφοράς αντικειμένων βάρους μέχρι 500 g και μπορεί να συνδεθεί μέσω USB, Wi-Fi και Bluetooth με Ηλεκτρονικό Υπολογιστή. Επιπρόσθετα, ο βραχίονας διαθέτει 13 θύρες εισόδου/έξοδού, API και πρωτόκολλα επικοινωνίας. Έτσι, δίνεται η δυνατότητα ανάπτυξης εφαρμογών σε διάφορες γλώσσες προγραμματισμού, PLC Controller, Microcontroller και Arduino. Ακόμη, υπάρχει δυνατότητα διασύνδεσης και

άλλων ρομποτικών βραχιόνων στην ίδια εφαρμογή, κάτι που είναι ιδιαίτερα χρήσιμο σε πολλές εφαρμογές. Το 2018 έχει κατακτήσει δύο βραβεία, Innovation CES Award και iF Design Award.[17]

Εικόνα 2.1.1: Ο ρομποτικός βραχίονας Dobot Magician με το εξάρτημα Gripper[18]

2.1.2 Συνδεσιμότητα του βραχίονα

Ο ρομποτικός βραχίονας είναι συμβατός με διάφορα εξαρτήματα. Όσον αφορά το άκρο του βραχίονα μπορεί να συνδεθεί με τα εξής εξαρτήματα:

1) Pen Holder: Δίνει την δυνατότητα ανάπτυξης εφαρμογών για γράψιμο και σχεδίαση σε χαρτί.
2) Gripper: Επιτρέπει στον βραχίονα να σηκώνει μικρά αντικείμενα, ώστε να τα μεταφέρει σε κάποιο άλλο σημείο.
3) Suction Cup: Επιτρέπει στον βραχίονα να σηκώνει μεγαλύτερα αντικείμενα με χρήση βεντούζας.
4) 3D Printing Kit: Παρέχει την δυνατότητα στον βραχίονα τρισδιάστατης εκτύπωσης με υλικό PLA.
5) Laser Engraving Kit: Παρέχει την δυνατότητα στον Dobot Magician χάραξης με laser σε υλικά όπως ξύλο και δέρμα.

Επιπρόσθετα, ο βραχίονας είναι συμβατός με τα εξής εξωτερικά εξαρτήματα:

1) Linear Rail Kit: Ράγα που δίνει την δυνατότητα στον βραχίονα να μετακινείται γραμμικά σε μια ευθεία.
2) Conveyor Belt Kit: Περιλαμβάνει έναν ιμάντα μεταφοράς αντικειμένων, αισθητήρα χρωμάτων ο οποίος αναγνωρίζει τα τρία βασικά χρώματα (Κόκκινο,

πράσινο, μπλε) και photoelectric αισθητήρα, για εντοπισμού αντικειμένου σε συγκεκριμένο σημείο.

3) Vision Kit: Περιλαμβάνει ειδική βάση για κάμερα, που επιτρέπει μέσω της εξειδικευμένης βιβλιοθήκης και των αλγορίθμων που είναι υλοποιημένοι, την επεξεργασίας του περιβάλλοντος του βραχίονα.
4) Controller: Χειριστήριο για έλεγχο του βραχίονα.
5) Connectivity: Περιλαμβάνει δύο εξαρτήματα (Wi-Fi και Bluetooth) που επιτρέπουν την ασύρματη σύνδεση του βραχίονα.

2.1.3 Έλεγχος του βραχίονα

Υπάρχουν διάφοροι τρόποι για τον προγραμματιστικό έλεγχο του βραχίονα. Ο πρώτος τρόπος είναι το επίσημο λογισμικό που παρέχεται από την Dobot, το Dobot Magician SDK.

Εικόνα 2.1.2[19]

Το Magician SDK περιλαμβάνει διάφορες εφαρμογές. Η πρώτη εφαρμογή, Teaching & Playback δίνει την δυνατότητα στον χρήστη να εισάγει ένα προς ένα τις κινήσεις που θέλει να εκτελέσει ο βραχίονας και στην συνέχεια ο βραχίονας να ξεκινήσει να τις εκτελεί.

Η 2^{η} εφαρμογή, δίνει αφορά την χρήση του Pen Holder, όπου ο χρήστης εισάγει στην οθόνη τι επιθυμεί ο βραχίονας να γράψει στο χαρτί.

Το Blockly είναι από τις πιο ενδιαφέρουσες εφαρμογές, διότι δίνει την δυνατότητα στους αρχάριους χρήστες να αναπτύξουν πολύπλοκες εφαρμογές κάνοντας χρήση ήδη υλοποιημένων blocks, χωρίς να γράψουν καθόλου κώδικα. Στην ουσία κάνουν Drag and Drop το κάθε component, παράγοντας κώδικα σε Python και δημιουργώντας έτσι μια ολοκληρωμένη εφαρμογή.

Η επόμενη εφαρμογή αφορά την δημιουργία, εισαγωγή, εξαγωγή και εκτέλεση Scripts, δηλαδή κώδικα. Η ιστοσελίδα του Dobot παρέχει αρκετά παραδείγματα τέτοιων scripts.
Το LeapMotion αφορά την αντιγραφή από τον βραχίονα των κινήσεων του χεριού του χρήστη, ενώ η εφαρμογή Mouse εκτελεί την κίνηση του Mouse στην οθόνη του υπολογιστή.
Ο επόμενος τρόπος για έλεγχο του βραχίονα είναι εφικτός με χρήση κάποιας γλώσσας προγραμματισμού. Το Dobot Magician API είναι διαθέσιμο σε περισσότερες από 20 γλώσσες, όπως Java, Python, C++ και C#.
Με αυτόν τον τρόπο, δίνεται η δυνατότητα ανάπτυξης ακόμη πιο πολύπλοκων εφαρμογών, εφόσον επεκτείνεται η δυνατότητα διασύνδεσης του βραχίονα και με περισσότερα εξαρτήματα.

2.1.4 Εφαρμογές

Ο συγκεκριμένος ρομποτικός βραχίονας μπορεί να χρησιμοποιηθεί και για εκπαιδευτικούς σκοπούς αλλά και για βιομηχανικούς σκοπούς. Στην εκπαίδευση, μπορεί να χρησιμοποιηθεί για εξοικείωση των μαθητών με τις έννοιες των ρομπότ, την ανάπτυξη διάφορων εφαρμογών, 3D εκτυπώσεις και χάραξης σε διάφορα υλικά. Στην βιομηχανία, το Dobot Magician μπορεί να χρησιμοποιηθεί για την προσομοίωση αρκετών περιπτώσεων όπου χρησιμοποιούνται μεγαλύτερα ρομπότ, όπως σε γραμμές παραγωγής αυτοκινήτων, γραμμές εμφιάλωσης, διαχωρισμού υλικών κλπ.

2.1.5 Dobot Magician API

Η ανάπτυξη του λογισμικού για την χρήση του Dobot Magician και των περιφερειακών του (Conveyor Belt και αισθητήρες) γίνεται είτε μέσω του Dobot Magician API, το οποίο είναι διαθέσιμο μέσω της επίσημης ιστοσελίδας του Dobot είτε μέσω βιβλιοθήκης όπου υπάρχουν υλοποιημένες συναρτήσεις για τις λειτουργίες που προσφέρονται μέσω του API. Επιπρόσθετα, στην ιστοσελίδα υπάρχουν αναρτημένα παραδείγματα χρήσης για την καλύτερη κατανόηση της δομής του κώδικα και χρήσης του API, ενώ επίσης περιλαμβάνει λεπτομερής τεκμηρίωση για τον χρήση της κάθε εντολής και των παραμέτρων που η κάθε συνάρτηση χρειάζεται.
Στον πίνακα 1 γίνεται ανάλυση των βασικών εντολών που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής και σχετίζονται με τον βραχίονα.

ConnectDobot(const char *portName, uint32_t baudrate, char *fwType, char *version float *time)	Η εντολή για γίνει η σύνδεση με το Dobot. Επιστρέφεται η κατάσταση, δηλαδή εάν υπάρχει σφάλμα, ή εάν επικοινωνεί κάποιο άλλο πρόγραμμα με το Dobot. Δέχεται ως παραμέτρους την θύρα(συνήθως COM3) και την έκδοση του Dobot.
SetPTPJointParams(PTPJointParams *ptpJointParams, bool isQueued, uint64_t *queuedCmdIndex)	Η εντολή όπου γίνεται ανάθεση της ταχύτητας (°/s) και της επιτάχυνσης (°/s2) για το PTP Mode (σημείο σε σημείο). Η εντολή δέχεται ακόμη και το εάν θα εκτελεστεί αμέσως ή αν θα τοποθετηθεί σε Queue.
SetPTPCommonParams(PTPCommonParams *ptpCommonParams,boolisQueued, uint64_t*queuedCmdIndex)	Η εντολή όπου γίνεται ανάθεση του ποσοστού της ταχύτητας και της επιτάχυνσης. Δέχεται ως παραμέτρους το ποσοστό(1-100) για το κάθε ένα καθώς και το κατά πόσο θα εκτελεστεί αμέσως ή αν θα τοποθετηθεί σε Queue.
SetPTPCmd(PTPCmd *ptpCmd, bool isQueued, uint64_t *queuedCmdIndex)	Η εντολή όπου μετακινείται ο βραχίονας σε συγκεκριμένες συντεταγμένες. Δέχεται ως παράμετρο το mode και τις συντεταγμένες x,y,z και r.
SetEndEffectorSuctionCup(bool enableCtrl, bool suck, bool isQueued, uint64_t *queuedCmdIndex)	Εντολή όπου ενεργοποιεί ή απενεργοποιεί το Suction Cup. Η 1η παράμετρος που δέχεται είναι η enable, όπου εάν είναι 0 γίνεται απενεργοποίηση της αντλίας αέρος και εάν είναι 1 ενεργοποίηση. Η επόμενη παράμετρος αφορά το εάν θα εξαγωγή του αέρα(0) ή εισαγωγή(1), ενώ η 3η παράμετρος το κατά πόσο θα εκτελεστεί αμέσως(0) ή αν θα τοποθετηθεί σε Queue(1).
SetEMotorS(EMotorS *eMotorS, bool isQueued, uint64_t *queuedCmdIndex)	Θέτει την ταχύτητα και την απόσταση που θα μετακινηθεί ο ιμάντας. Δέχεται ως παράμετρούς την ένδειξη του Emotor (του μοτέρ στον ιμάντα μετακίνησης), 1 για την ενεργοποίηση του ή 0 για απενεργοποίηση την ταχύτητα και κατά πόσο θα εκτελεστεί αμέσως ή αν θα τοποθετηθεί σε Queue(1 ή 0)
SetInfraredSensor(bool enable,InfraredPort infraredPort, uint8_t version)	Ενεργοποίηση του photoelectric αισθητήρα. Η 1η παράμετρος είναι 1 για την ενεργοποίηση του αισθητήρα ή 0 για την απενεργοποίηση του. Ακόμη, δέχεται

	την θύρα που ο αισθητήρας έχει συνδεθεί και την έκδοση του(0 για Version 1.0, 1 για Version 2)
GetInfraredSensor (InfraredPort infraredPort, uint8_t *value)	Έλεγχος της κατάστασης του photoelectric αισθητήρα. Δέχεται σαν παράμετρο την θύρα που είναι συνδεδεμένος ο αισθητήρας και επιστρέφει 1 εάν έχει εντοπίσει κάποιο αντικείμενο ή 0 εάν όχι.
SetColorSensor(bool enable, ColorPort colorPort, uint8_t version)	Έλεγχος του αισθητήρα χρώματος. Η 1η παράμετρος είναι 1 για την ενεργοποίηση του αισθητήρα ή 0 για την απενεργοποίηση του. Ακόμη, δέχεται την θύρα που ο αισθητήρας έχει συνδεθεί και την έκδοση του(0 για Version 1.0, 1 για Version 2)
GetColorSensor(uint8_t *r, uint8_t *g, uint8_t *b)	Έλεγχος της κατάστασης του αισθητήρα χρώματος. Επιστρέφεται το χρώμα ως r/g/b.
GetPose(Pose *pose)	Επιστρέφει τις καρτεσιανές συντεταγμένες της θέσης του βραχίονα.
DisconnectDobot(void)	Αποσύνδεση με τον βραχίονα.

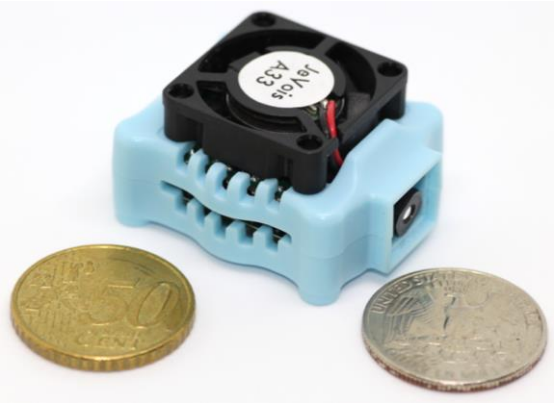
Πίνακας 2.1: Οι κύριες εντολές του Dobot Magician API

2.2 Η κάμερα JeVois Smart Vision

2.2.1 Εισαγωγή

Η JeVois Smart Machine Vision Camera είναι μια κάμερα ανοικτού κώδικα, με 4-πύρηνο επεξεργαστή (ARM Cortex A7) και δυνατότητα σύνδεσης σε Η/Υ, Arduino και Raspberry Pi μέσω USB 2.0. Έχει πολύ μικρό μέγεθος και ζυγίζει μόλις 17 γραμμάρια, καθιστώντας την έτσι ιδιαίτερα ευέλικτη και επιτρέποντας την χρήση της σε πολλές εφαρμογές. Διαθέτει ακόμη Micro SD υποδοχή, σειριακή θύρα και 256 MB Ram. Όσον αφορά τον αισθητήρα της κάμερας, είναι 1.3 Megapixels και μπορεί καταγράφει βίντεο μέχρι και 120 Frames per Second.

Η κάμερα JeVois υποστηρίζει πέραν των 40 λειτουργιών, μέσω βιβλιοθηκών όπως OpenCV και TensorFlow.[20]



Εικόνα 2.2.1: Η JeVois Camera[21]

2.2.2 Δυνατότητες

Πιο συγκεκριμένα, έχει την δυνατότητα αναγνώρισης προσώπων και πέραν των 1000 κατηγοριών αντικειμένων. Ακόμη, έχει την δυνατότητα αναγνώρισης δρόμων, QR Codes και ArUco Tags. Επίσης, υπάρχει η δυνατότητα για εύρεση αντικειμένων μέσω χρώματος αλλά και παρακολούθησης κινούμενων αντικειμένων.

2.2.3 Έλεγχος της JeVois

Ο έλεγχος της JeVois γίνεται κυρίως μέσω του JeVois Inventor, το οποίο περιλαμβάνει υλοποιημένα Modules που παρέχουν τις πιο πάνω δυνατότητες, console για επικοινωνία με την κάμερα και δυνατότητα ρύθμισης της (contrast, brightness κλπ.). Χρησιμοποιούνται δύο γλώσσες προγραμματισμού για την υλοποίηση των Modules, Python και C++. Το JeVois Inventor παρέχει την δυνατότητα ανάπτυξης νέων Modules μόνο σε γλώσσα Python, διότι τα αρχεία Python δεν απαιτούν Compile για να εκτελεστούν. Για την ανάπτυξη ή την επεξεργασία Module σε C++ απαιτείται χρήση JeVois SDK, και είναι εφικτή μόνο μέσω Linux.

Για επικοινωνία της κάμερας με κάποιο άλλο λογισμικό παρέχεται η δυνατότητα εξαγωγής των αποτελεσμάτων από την βιντεοσκόπηση σε δομημένα μηνύματα κειμένου μέσω της θύρας USB.

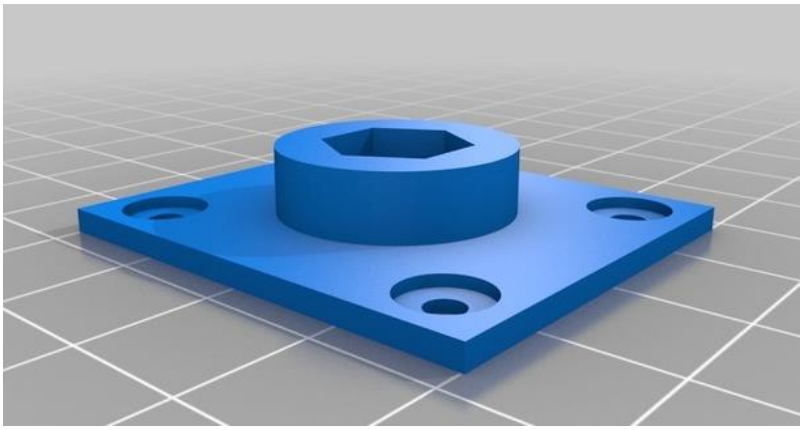
2.2.4 Βάση κάμερας

Κατά την σύνδεση της κάμερας με τον Η/Υ παρατηρήθηκε ότι η κάμερα είναι αρκετά ασταθής κάτι που παρεμπόδιζε την λήψη σταθερού βίντεο, με συνέπεια να μην μπορεί να γίνει αναγνώριση των αντικειμένων. Έτσι θεωρήθηκε αναγκαία η χρήση κατάλληλης βάσης ούτως ώστε η κάμερα να σταματήσει να μετακινείται, να μπορεί να τοποθετείται

σε οποιοδήποτε ύψος αλλά και να παραμένει σε σταθερό σημείο γύρω από τον βραχίονα για να μην χρειάζεται συνεχώς εκπαίδευση της JeVois για το εκάστοτε αντικείμενο.

Μια τέτοια βάση όμως δεν παρέχεται μαζί με την JeVois Camera, ενώ μέσα από την επίσημη ιστοσελίδα[22] παρέχεται ένα συγκεκριμένο τρισδιάστατο σχέδιο για 3D Printing. Για την επισύναψη της κάμερας στην συγκεκριμένη βάση θα έπρεπε να γίνει χρήση βιδών, κάτι που θα δημιουργούσε πρόβλημα στο κάτω μέρος της κάμερας στην περίπτωση όπου θα ήταν αναγκαίο να αφαιρεθεί η κάμερα από την βάση.

Για τον συγκεκριμένο λόγο δημιουργήθηκε μια καινούρια βάση μέσω της τρισδιάστατης εκτύπωσης που αναλύεται στο υποκεφάλαιο 2.4 και δεν προϋποθέτει την χρήση βιδών.



Εικόνα 2.2.2: Η αρχική βάση της JeVois [23]

2.2.5 Διασύνδεση με το περιβάλλον του Dobot Magician

Λόγω την πολλών δυνατοτήτων που διαθέτει η JeVois σχετικά με την επεξεργασία εικόνας θεωρείται ιδιαίτερα χρήσιμη στην παρούσα διπλωματική εργασία. Όπως έχουμε δει προηγουμένως, η JeVois Camera έχει την ικανότητα για αναγνώριση συγκεκριμένων αντικειμένων στο περιβάλλον που καταγράφει κάνοντας χρήση είτε βιβλιοθηκών μηχανικής μάθησης όπως η OpenCV είτε του Module Object Detection όπου εισάγοντας μια φωτογραφία ενός αντικειμένου γίνεται εντοπισμός του συγκεκριμένου αντικειμένου στο περιβάλλον που καταγράφεται.

Η σύνδεση της JeVois με το Dobot Magician γίνεται προγραμματιστικά μέσω της Python. Πιο συγκεκριμένα, γίνεται σύνδεση της JeVois και του Η/Υ μέσω USB όπου όλα τα δεδομένα που δημιουργούνται real-time αποστέλλονται με σειριακά μηνύματα

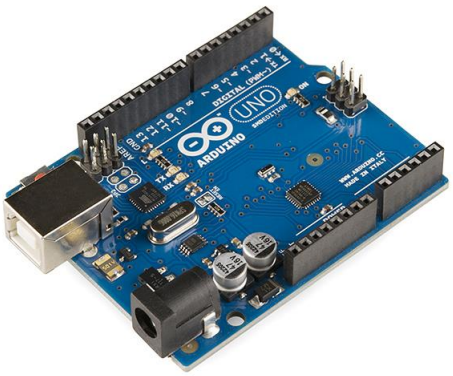
στο λογισμικό που τρέχει στον Η/Υ. Το λογισμικό αποκωδικοποιεί τα μηνύματα της JeVois ελέγχοντας έτσι κατά πόσο το αντικείμενο που ψάχνει έχει εντοπιστεί.
Συνεπώς, τοποθετώντας την κάμερα σε σταθερό σημείο δίπλα από τον ιμάντα μεταφοράς αντικειμένων, μπορούμε να την εκπαιδεύσουμε για να εντοπίζει κάποια αντικείμενα (χρησιμοποιώντας κάποιο από τα modules που βρίσκονται υλοποιημένα στο JeVois Inventor) καθώς αυτά μεταφέρονται στον ιμάντα. Στην συνέχεια ενημερώνεται το πρόγραμμα (βλ. Κεφάλαιο 4.2.1.1) και ο βραχίονας μετακινείται αναλόγως. Ακόμη, υπάρχει η δυνατότητα χρήσης περισσότερων από μιας JeVois κάμερας για την εκτέλεση περισσότερων λειτουργιών. Για παράδειγμα μια δεύτερη κάμερα θα μπορούσε να εντοπίζει αντικείμενα συγκεκριμένου χρώματος ή ακόμη να ελέγχει αν τα αντικείμενα τοποθετήθηκαν ορθά στο τέλος.

2.3 Arduino

2.3.1 Εισαγωγή στο Arduino

Κατά την ανάπτυξη των εφαρμογών για τον ρομποτικό βραχίονα, παρατηρήθηκε ότι δεν υπάρχει τρόπος με χρήση του Conveyor Belt Kit για να εντοπιστεί η ακριβής θέση ενός αντικειμένου στον ιμάντα. Ο photoelectric αισθητήρας που υπήρχε στο kit, έχει σαν έξοδο μονό true ή false, αναλόγως εάν υπάρχει αντικείμενο στην ευθεία του ή όχι. Συνεπώς, στην περίπτωση όπου υπήρχε αντικείμενο, δεν μπορούσε να γνωρίζει ο βραχίονας την ακριβή θέση του συγκεκριμένου αντικειμένου στον ιμάντα, κάτι που οδηγούσε στο να μην μπορεί να μεταφέρει το αντικείμενο. Λόγω του πιο πάνω προβλήματος, επιλέχθηκε να γίνει χρήση ενός Arduino Board σε συνδυασμό με ένα αισθητήρα απόστασης.
Το Arduino είναι ένας ανοικτού κώδικα μικροελεγκτής που διαθέτει εισόδους και εξόδους και μπορεί να προγραμματιστεί με την γλώσσα Wiring μέσω του Arduino IDE. Λόγω του μικρού μεγέθους του και των πολλών δυνατοτήτων που έχει, είναι ιδιαίτερα χρήσιμο στην υλοποίηση διαφόρων κατασκευών, όπως drones και συστήματα που πραγματοποιούν ελέγχους θερμοκρασίας, υγρασίας κλπ. Τροφοδοτείται με ρεύμα 3.3V/5V και διαθέτει USB θύρα για την σύνδεση του στον Η/Υ. Με την σύνδεση άλλων components όπως Wi-Fi και Bluetooth, μπορεί να επεκτείνει τις δυνατότητες συνδεσιμότητας του. Ακόμη, διαθέτει την δυνατότητα να τρέχει αυτόνομα χωρίς ενσύρματη σύνδεση με Η/Υ, μέσω τροφοδοτικού/μπαταρίας για την ικανοποίηση της

τροφοδοσίας και αρχική μεταφόρτωση του λογισμικού που θέλουμε να εκτελέσουμε στο Arduino. [24]



Εικόνα 2.3.1: Arduino Uno[25]

2.3.2 Αισθητήρες

Το Arduino διαθέτει το πλεονέκτημα της επεκτασιμότητας μέσω διαφόρων εξαρτημάτων. Οι αισθητήρες αποτελούν ένα χαρακτηριστικό παράδειγμα hardware που μπορούν να συνδεθούν με πολύ εύκολο τρόπο στο υπόλοιπο κύκλωμα. Υπάρχουν διαθέσιμοι πάρα πολλοί αισθητήρες για κάθε ανάγκη. Μερικά παραδείγματα τέτοιων αισθητήρων είναι ο αισθητήρας υγρασίας για εφαρμογές που σχετίζονται με την μέτρηση της υγρασίας πχ στο έδαφος, αισθητήρας στάθμης νερού που χρησιμοποιείται για την εύρεση εκροών νερού σε συστήματα βρασμού, αισθητήρας θερμοκρασίας, μέτρησης των κτύπων της καρδίας, φλόγας και άλλοι. Ο αισθητήρας που χρησιμοποιήθηκε για την ανίχνευση της απόστασης των αντικειμένων στον ιμάντα μετακίνησης είναι ο αισθητήρας υπερήχων HC-SR04. Χρησιμοποιεί έναν υποδοχέα και έναν πομπό υπερήχων όπου μετρώντας τον χρόνο που μεσολαβεί από την στιγμή που εκπέμπει έναν υπέρηχο ο πομπός μέχρι να τον λάβει ο υποδοχέας και διαιρώντας με την ταχύτητα του ήχου προκύπτει η απόσταση που υπάρχει μέχρι το αντικείμενο. Μπορεί να λάβει μετρήσεις από 2 cm μέχρι 400cm με ακρίβεια μέχρι 3mm.[26]

Εικόνα 2.3.2: Ultrasonic Sensor[27]

2.3.3 Χρήση του Arduino

Όπως έχουμε αναφέρει και πιο πάνω, για τον προγραμματισμό του Arduino παρέχεται το λογισμικό Arduino IDE και γίνεται χρήση της γλώσσας προγραμματισμού Wiring η οποία είναι βασισμένη στην C++. Για αρχικό έλεγχο ορθότητας του συστήματος, δημιουργήθηκε κώδικας στο Arduino IDE ο οποίος εκτύπωνε τις αποστάσεις αντικειμένων που περνούσαν από τον αισθητήρα. Αρχικά συνδέσαμε το Arduino με το αισθητήρα βάση της πιο πάνω συνδεσμολογίας και τον Η/Υ με το Arduino μέσω USB. Στην συνέχεια έγινε φόρτωση του προγράμματος που δημιουργήσαμε στο Arduino και ξεκίνησε η εκτέλεση του, παρουσιάζοντας τα δεδομένα στην οθόνη του υπολογιστή.

2.3.4 Διασύνδεση με το περιβάλλον του Dobot Magician

Το Arduino είναι ιδιαίτερα χρήσιμο στην παρούσα διπλωματική εργασία διότι συνεισφέρει στην επίλυση ενός σημαντικού προβλήματος, της εύρεσης της ακριβούς θέσης των αντικειμένων στον ιμάντα μεταφοράς. Συνεπώς, τοποθετούμε τον αισθητήρα απέναντι από τον photoelectric αισθητήρα που παρέχεται από την Dobot. Με τον συγκεκριμένο τρόπο όταν ένα αντικείμενο φθάσει στον photoelectric αισθητήρα και σταματήσει ο ιμάντας να λειτουργεί, εκτελείται το πρόγραμμα στο Arduino και επιστρέφεται μια τιμή, που αντιστοιχεί στην απόσταση του αντικειμένου από τον αισθητήρα. Όσο μικρότερη η απόσταση, τόσο πιο κοντά στον αισθητήρα βρίσκεται το αντικείμενο, άρα τόσο πιο μακριά θα πρέπει ο βραχίονας να μετακινηθεί για να επιλέξει το αντικείμενο. Αντιθέτως, όσο μεγαλύτερη η απόσταση, τόσο πιο μακριά βρίσκεται το αντικείμενο από τον αισθητήρα, άρα ο βραχίονας θα πρέπει να μετακινηθεί λιγότερο.

Για την σύνδεση του προγράμματος του Arduino έχει δημιουργηθεί μια συνάρτηση σε γλώσσα Python μαζί με το υπόλοιπο λογισμικό που αφορά το Dobot Magician. Η

συνάρτηση αυτή διαβάζει την είσοδο από την θύρα που το Arduino είναι συνδεδεμένο και επιστρέφει την απόσταση του αντικειμένου από τον αισθητήρα, ούτως ώστε να καθοριστούν οι ακριβείς συντεταγμένες που θα πρέπει ο βραχίονας να μετακινηθεί.

2.4 3D Printing

2.4.1 Εισαγωγή στο 3D Printing

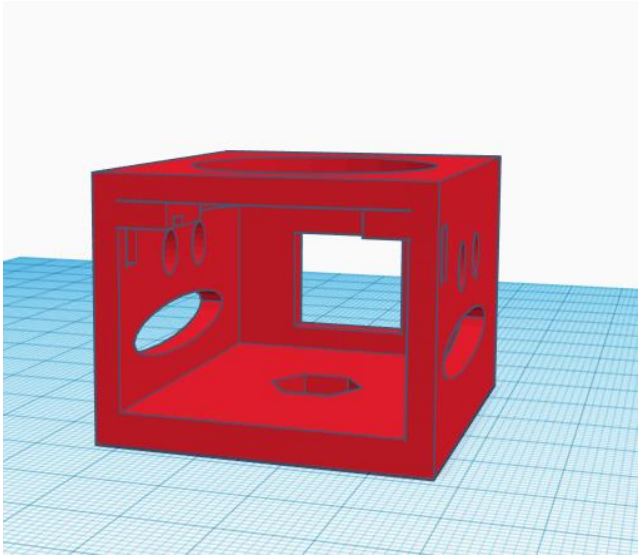
Το 3D Printing είναι μια μέθοδος για την δημιουργία τρισδιάστατων αντικειμένων με χρήση διαδοχικών στρώσεων του υλικού. Η συγκεκριμένη ιδέα προέκυψε την δεκαετία του 1970, ενώ την επόμενη δεκαετία ξεκίνησαν οι πρώτες δημιουργίες εξοπλισμού και υλικών.[28] Τα τελευταία χρόνια η τρισδιάστατη εκτύπωση έχει γίνει ιδιαίτερα δημοφιλής, λόγω των προσιτών τιμών που έχουν πλέον οι τρισδιάστατοι εκτυπωτές και των πολλών περιπτώσεων χρήσης από τους καταναλωτές. Σήμερα οι εκτυπωτές χρησιμοποιούνται ευρέως σε πολλούς τομείς, για την δημιουργία πρωτοτύπων, εξαρτημάτων για αυτοκίνητα, υποδημάτων και πολλά άλλα.

Υπάρχουν διάφορες μέθοδοι για την 3D εκτύπωση, όμως η πιο δημοφιλής και η μέθοδος που χρησιμοποιήθηκε για την εκτύπωση της βάσης είναι η μέθοδος FDM. Στην συγκεκριμένη τεχνική, το υλικό τροφοδοτείται σε μορφή νήματος στην θερμαινόμενη κεφαλή, όπου λιώνει. Η βάση του εκτυπωτή και η κεφαλή μετακινούνται στην κατάλληλη θέση, δημιουργώντας με αυτόν τον τρόπο τις στρώσεις του υλικού. [29]

Ακόμη, υπάρχουν διάφορα υλικά που χρησιμοποιούνται για εκτύπωση με το κάθε υλικό να έχει διαφορετικές ιδιότητες. Συνήθως το υλικό που χρησιμοποιείται για την εκτύπωση των αντικειμένων είναι πολυμερές (PLA, ABS, Nylon κ.α.) όμως υπάρχουν και άλλα υλικά, όπως κεραμικό και μέταλλα που μπορούν να χρησιμοποιηθούν.

2.4.2 Σχεδιασμός Βάσης

Για την εκτύπωση ενός αντικειμένου ακολουθείται μια διαδικασία. Το πρώτο βήμα αποτελεί τον σχεδιασμό του ψηφιακού μοντέλου μέσω μιας CAD εφαρμογής. Για τον σκοπό αυτό, χρησιμοποιήθηκε το λογισμικό TinkerCad της Autodesk. Το πλεονέκτημα του συγκεκριμένου λογισμικού είναι η απλότητα του αλλά και η ευκολία χρήσης από αρχάριους χρήστες. Έτσι δημιουργήθηκε το πιο κάτω 3D object.



Εικόνα 2.4.1: Το ψηφιακό μοντέλο της βάσης

Διαθέτει στο μπροστινό του μέρος θήκη για το μπροστινό μέρος της JeVois, στο κάτω μέρος θήκη βίδας για την σύνδεση του σε τρίποδα και ειδικές οπές για τον καλύτερη ροή του αέρα από τα fans που διαθέτει η κάμερα. Επίσης, διαθέτει στο πίσω μέρος σύστημα για κλείδωμα της κάμερας όταν τοποθετηθεί μέσα στο αντικείμενο.
Τέλος, απομένει η εξαγωγή του ψηφιακού μοντέλου σε μορφή STL/OBJ, ώστε να εισαχθεί στο λογισμικό του 3D Printer.

2.4.3 Εκτύπωση Βάσης

Για την τρισδιάστατη εκτύπωση του αντικειμένου που σχεδιάστηκε ψηφιακά επιλέχθηκε ως εκτυπωτής ο Prusa i3 MK3s που είναι διαθέσιμος στο εργαστήριο του ΚΕΠ, MakerSpace ενώ το υλικό που χρησιμοποιήθηκε είναι PLA. Το δεύτερο βήμα που απαιτείται για την τρισδιάστατη εκτύπωση είναι η εισαγωγή του STL/OBJ αρχείου που προέκυψε από τον ψηφιακό σχεδιασμό του αντικειμένου στο λογισμικό που παρέχεται από την Prusa, PrucaSlicer. Μέσα από το λογισμικό γίνεται η προβολή του αντικειμένου και ο καθορισμός διάφορων παραμέτρων που αφορούν την εκτύπωση, όπως η επιλογή του υλικού, διάμετρος της βελόνας της κεφαλής, το μέγεθος του αντικειμένου κ.α. Επιπρόσθετα, υπάρχει η δυνατότητα προβολής της εξέλιξης της διαδικασίας εκτύπωσης, δηλαδή με ποια σειρά θα εκτυπωθούν οι στρώσεις του αντικειμένου.

Τέλος, γίνεται εξαγωγή του αντικειμένου σε μορφή G-Code, ούτως ώστε να διαβαστεί από τον εκτυπωτή. Το αρχείο που παράγεται αποθηκεύεται στην κάρτα SD στον υπολογιστή και έπειτα εισάγεται στον εκτυπωτή.

Το τελευταίο βήμα που απαιτείται για να ξεκινήσει η διαδικασία εκτύπωσης, είναι η προετοιμασία του Prusa. Για τον σκοπό αυτό, αρχικά καθαρίζουμε την επιφάνεια εκτύπωσης βάση των οδηγιών του κατασκευαστή. Στην συνέχεια μέσα από το μενού του εκτυπωτή επιλέγουμε για εκτύπωση το αρχείο που βρίσκεται στην κάρτα SD και το υλικό εκτύπωσης. Ο εκτυπωτής ξεκινάει την διαδικασία θέρμανσης της κεφαλής και της επιφάνειας, ενώ στην συνέχεια πραγματοποιείται αυτόματα Calibration των τριών αξόνων. Όταν ολοκληρωθεί το calibration, η εκτύπωση ξεκινάει.

Εικόνα 2.4.2: Η βάση της κάμερας

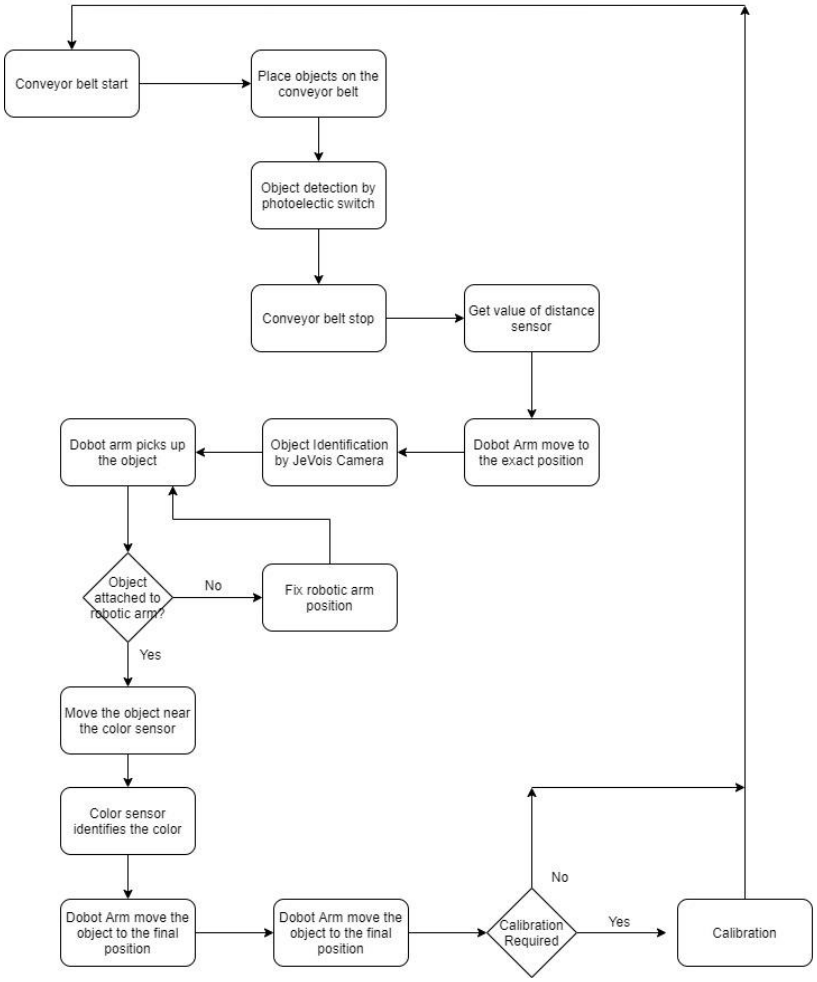
Κεφάλαιο 3

Πειραματική Διάταξη

3.1 Block Diagram

Η διαδικασία που απαιτείται για την αναγνώριση των αντικειμένων και κατηγοριοποίηση τους είναι η εξής. Αρχικά, ξεκινάει να κινείται ο ιμάντας μετακίνησης και τοποθετούνται σε αυτόν αντικείμενα. Όταν ο φωτοηλεκτρικός αισθητήρας εντοπίσει κάποιο αντικείμενο, ο ιμάντας σταματάει την λειτουργία του και ο αισθητήρας απόστασης που είναι συνδεδεμένος με το Arduino μετράει την απόσταση. Στην συνέχεια ο ρομποτικός βραχίονας μετακινείται στις εκάστοτε συντεταγμένες και η κάμερα JeVois εντοπίζει το αντικείμενο. Έπειτα, ο βραχίονας προσπαθεί να σηκώσει το αντικείμενο. Σε περίπτωση που αυτό δεν γίνει κατορθωτό, γίνονται οι απαραίτητες διορθώσεις στις συντεταγμένες μετακίνησης του (επαναλαμβάνεται η μέτρηση του αισθητήρα απόστασης) και ο βραχίονας επαναλαμβάνει την προσπάθεια του. Εάν ο βραχίονας δεν κατορθώσει να επιλέξει το αντικείμενο έπειτα από δύο προσπάθειες, τότε ο ιμάντας ξεκινάει ξανά και η διαδικασία επαναρχίζει για την μεταφορά του επόμενου

αντικειμένου κάνοντας skip το αντικείμενο που δεν μπόρεσε να σηκώσει. Εάν όμως το αντικείμενο επιλεγεί, η διαδικασία προχωράει με την μεταφορά του αντικειμένου πάνω από τον αισθητήρα χρώματος, για να εντοπιστεί το χρώμα του. Στην συνέχεια, το αντικείμενο μεταφέρεται στην τελική του θέση, βάση του χρώματος του ή της κατηγορίας που εντόπισε η κάμερα. Τέλος γίνεται έλεγχος για το κατά πόσον ο Dobot Magician χρειάζεται ρύθμιση σε οποιονδήποτε άξονα του. Στην περίπτωση όπου χρειάζεται, γίνεται το calibration και ο ιμάντας ξεκινάει για να αρχίσει η διαδικασία για την κατηγοριοποίηση του επόμενου αντικειμένου. Σε αντίθετη περίπτωση η διαδικασία αρχίζει και πάλι από την αρχή.



Εικόνα 3.1.1: Block Diagram

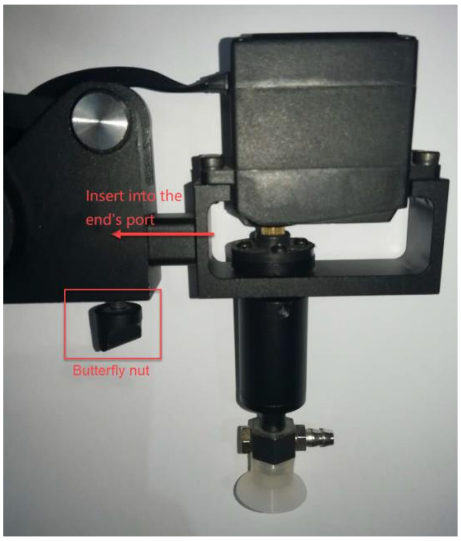
3.2 Συναρμολόγηση του Dobot Magician

Για την υλοποίηση της πειραματικής διάταξης αρχικά απαιτείται η διασύνδεση του βραχίονα με όλα τα εξαρτήματα που χρειάζεται για να λειτουργήσει. Για τον σκοπό αυτό ακολουθήθηκε η διαδικασία που περιγράφεται στον οδηγό χρήσης του Dobot Magician. Το πρώτο βήμα είναι η διασύνδεση του βραχίονα στον υπολογιστή μέσω της

θύρας USB και η τροφοδοσία του βραχίονα με ρεύμα. Στην συνέχεια, απαιτείται η εγκατάσταση του Dobot Studio και του Dobot Magician Driver και γίνεται έλεγχος κατά πόσον υπάρχει διαθέσιμη COM θύρα. Έπειτα τοποθετούμε τον βραχίονα σε κλίση 45 μοιρών από την βάση του και πατάμε το On/Off button που βρίσκεται στην βάση του βραχίονα για να ξεκινήσει την λειτουργία του. Μέσω του Dobot Studio μπορεί να γίνει έλεγχος εάν ο βραχίονας έχει συνδεθεί σωστά, πατώντας το κουμπί Connect που βρίσκεται στο πάνω μέρος του προγράμματος.

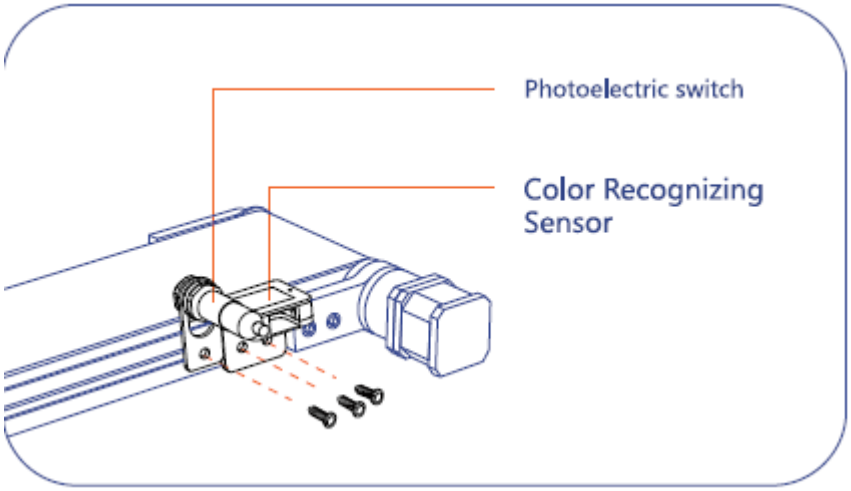
3.2.1 Εγκατάσταση του Suction Cup Kit

Για την εγκατάσταση του Suction Cup, αρχικά συνδέουμε τις δύο θύρες της αντλίας αέρα(GP1 και SW1) με την βάση του βραχίονα και ενώνουμε το εξάρτημα στην άκρη του βραχίονα. Τέλος, συνδέουμε την τροφοδοσία του αέρα στην άκρη του βραχίονα και το GP3 σύρμα του Suction cup με τον βραχίονα.



3.2.2 Εγκατάσταση του Conveyor Belt Kit

Αρχικά συνδέουμε τον ιμάντα και τον βραχίονα μέσω της θύρας Stepper1 που βρίσκεται στην βάση του βραχίονα. Στην συνέχεια, τον φωτοηλεκτρικό αισθητήρα συνδέουμε στην θύρα GP2 που βρίσκεται στην βάση, ενώ τον αισθητήρα αναγνώρισης χρωμάτων στην θύρα GP5 που βρίσκεται στο άκρο του βραχίονα. Τοποθετούμε τους δύο αισθητήρες στον ιμάντα, βιδώνοντας τους στις ειδικές θέσεις που υπάρχουν στις δύο πλευρές του. Απέναντι από τον φωτοηλεκτρικό αισθητήρα πρέπει να μην βρίσκεται κάποιο αντικείμενο, διότι θα επιστρέφει συνεχώς την τιμή 1, με αποτέλεσμα να μην μπορεί να αναγνωρίσει τα αντικείμενα που βρίσκονται στον ιμάντα. Στην συγκεκριμένη πειραματική διάταξη, τοποθετήθηκαν και οι δύο αισθητήρες στο άκρο του ιμάντα όπου είναι τοποθετημένος και ο βραχίονας.



Εικόνα 3.2.1: Διάταξη αισθητήρων

3.2.3 Εγκατάσταση του Wifi Kit

Σε περίπτωση που δεν επιθυμούμε να γίνεται η σύνδεση του ιμάντα με τον Η/Υ ενσύρματα μπορεί να γίνεται μέσω του Wifi Kit. Για την εγκατάσταση του, κλείνουμε τον βραχίονα και συνδέουμε το kit στο UART Interface της βάσης. Στην συνέχεια ανάβουμε τον βραχίονα και μέσω των Wi-Fi ρυθμίσεων στο Dobot Studio συμπληρώνουμε το SSID(Όνομα του Wifi-Πρέπει να είναι ίδιο με τον Wifi που είναι συνδεδεμένος ο υπολογιστής) και τον κωδικό. Κάνουμε κλικ στο OK και έπειτα από 5 δευτερόλεπτα θα ανάψει πράσινο φως στο Wireless Module, δεικνύοντας ότι η σύνδεση είναι επιτυχής. Τέλος, αποσυνδεόμαστε πατώντας το Disconnect, κάνουμε κλικ στις επιλογές σύνδεσης και επιλέγουμε την διεύθυνση IP. Έτσι, ο βραχίονας είναι

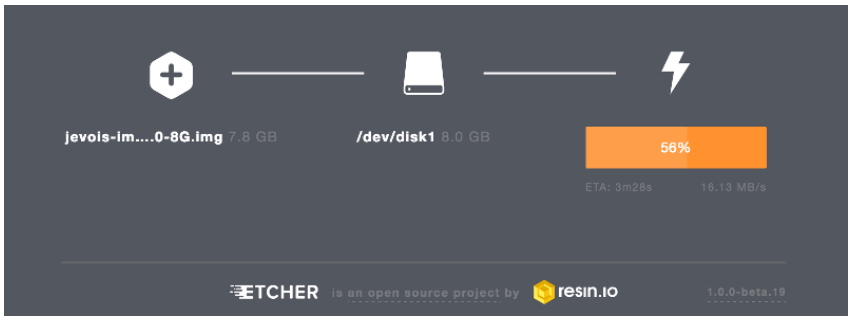
συνδεδεμένος ασύρματα με τον Η/Υ και μπορούμε πλέον να αφαιρέσουμε το USB σύρμα.

Εικόνα 3.2.2: Σύνδεση του Wireless Kit

3.3 JeVois Camera

3.3.1 Flash Image

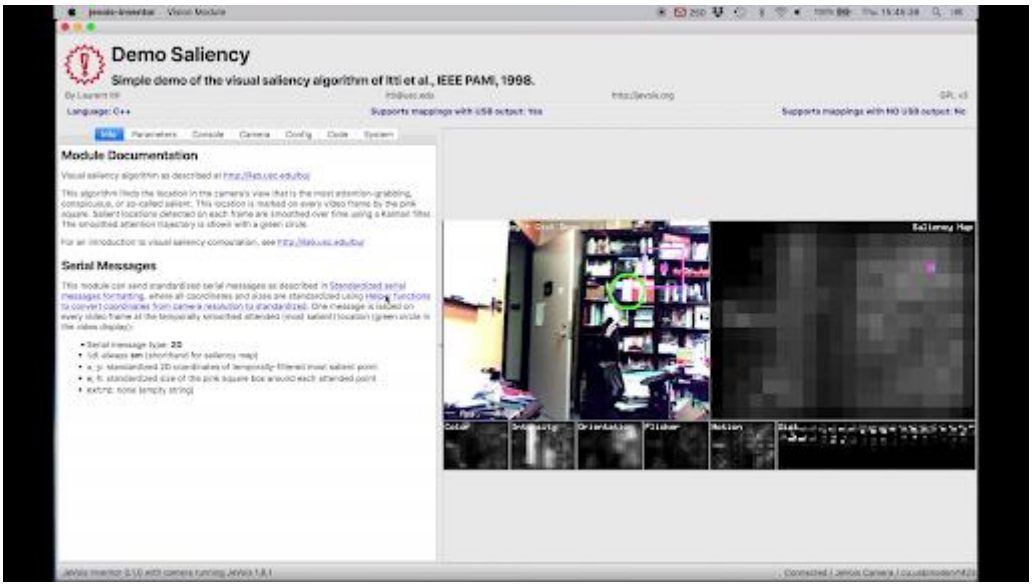
Για την χρήση της JeVois Camera απαιτείται αρχικά να γίνει Flash το λογισμικό της στην κάρτα SD. Για τον σκοπό αυτό χρησιμοποιήθηκε το λογισμικό Etcher. Το πρώτο βήμα είναι να κατεβάσουμε το .zip αρχείο από την ιστοσελίδα της JeVois. Στην συνέχεια, τοποθετούμε την κάρτα SD στον Η/Υ και μέσω του προγράμματος Etcher επιλέγουμε το .zip αρχείο που έχουμε κατεβάσει. Τέλος, επιλέγουμε το drive όπου βρίσκεται η κάρτα SD, κάνουμε κλικ στο κουμπί Flash και περιμένουμε μέχρι να ολοκληρωθεί η διαδικασία.



Εικόνα 3.3.1: Flash image μέσω του λογισμικού Etcher

3.3.2 Εγκατάσταση JeVois Inventor

Εφόσον έχουμε ολοκληρώσει το προηγούμενο βήμα, κάνοντας flash το κατάλληλο λογισμικό στην κάρτα SD (τελευταία έκδοση του Image), εισάγουμε την κάρτα στην κάμερα. Πριν όμως αρχίσουμε να χρησιμοποιούμε τις λειτουργίες της JeVois, απαιτείται η εγκατάσταση της γραφικής διαπροσωπείας της JeVois, το Inventor. Έτσι, κατεβάζουμε το λογισμικό από την ιστοσελίδα της JeVois και το ανοίγουμε. Συνδέουμε την κάμερα με τον υπολογιστή μέσω του καλωδίου USB και πλέον το Inventor είναι έτοιμο να εκτελέσει οποιαδήποτε λειτουργία είναι διαθέσιμη μέσω της κάμερας.



Εικόνα 3.3.2 Το λογισμικό JeVois Inventor

Στην συγκεκριμένη πειραματική διάταξη θα χρησιμοποιήσουμε την λειτουργία εντοπισμού αντικειμένων Object Detect. Για τον σκοπό αυτό, πρέπει είτε να εισάγουμε τα αντικείμενα που επιθυμούμε να εντοπίζονται στην κάρτα SD, στον φάκελο Saved ή να κάνουμε καταγραφή ενός αντικειμένου στην σκηνή, αποθηκεύοντας την εικόνα του αντικειμένου από την οπτική γωνία που θα εντοπίζεται από την JeVois Camera.

Για την εισαγωγή ενός αντικειμένου κατευθείαν στην κάρτα, αποσυνδέουμε την κάμερα από το USB και εξάγουμε την κάρτα SD από την JeVois. Στην συνέχεια, συνδέουμε την κάρτα στον H/Y και μεταβαίνουμε στον φάκελο Saved. Εκεί, αντιγράφουμε την φωτογραφία του αντικειμένου που θέλουμε να εντοπίζει η κάμερα.

Για την καταγραφή αντικειμένου από την σκηνή, αρχικά ενεργοποιούμε το Window μέσω του Inventor, μετακινούμε την κάμερα ούτως ώστε το αντικείμενο να είναι στο μέσα στο Window που έχουμε επιλέξει. Εάν χρειάζεται μπορούμε να αλλάξουμε και τις διαστάσεις του παραθύρου, ώστε να εντοπίζει το αντικείμενο πιο αποδοτικά. Τέλος,

κάνουμε κλικ στο Save, και το αντικείμενο έχει αποθηκευτεί στην μνήμη της JeVois, άρα πλέον θα μπορεί να εντοπίζεται.

Εικόνα 3.3.3: Αποθήκευση αντικειμένου μέσω του Inventor

3.3.3 Διασύνδεση με τον βραχίονα

Για να μπορεί η JeVois να χρησιμοποιείται από το λογισμικό που έχει δημιουργηθεί για τον έλεγχο του βραχίονα, θα πρέπει να μπορεί να αποστέλλει τα δεδομένα που εντοπίζει στην Python. Έτσι, μέσω του JeVois Inventor, επιλέγουμε την εξαγωγή των δεδομένων σε μηνύματα 2D. Τα μηνύματα αυτά έχουν την εξής δομή[36] «N2 id x y w h», όπου

N2	**Το είδος του μηνύματος. N2 = Normal D2 Message**
Id	Συμβολοσειρά που περιέχει το όνομα του αντικειμένου που εντοπίστηκε.
X, Y	Η τοποθεσία σε (x,y) συντεταγμένες του κέντρου του αντικειμένου που εντοπίστηκε.
W, H	Οι διαστάσεις του αντικειμένου που εντοπίστηκε, πλάτος και μήκος.

Με αυτό τον τρόπο, τα δεδομένα που παράγονται από την κάμερα, αποστέλλονται στην θύρα που είναι συνδεδεμένη μέσω USB η κάμερα με την πιο πάνω δομή. Το επόμενο βήμα είναι να διαβάζονται από το λογισμικό του Dobot. Για τον σκοπό αυτό έχει δημιουργηθεί η συνάρτηση ObjectFound (για κώδικα και επεξήγηση βλ. Κεφ. 4.2.1.1)

που όταν καλείται επιστρέφει το όνομα του αντικείμενου που έχει εντοπιστεί από την JeVois στην σκηνή.

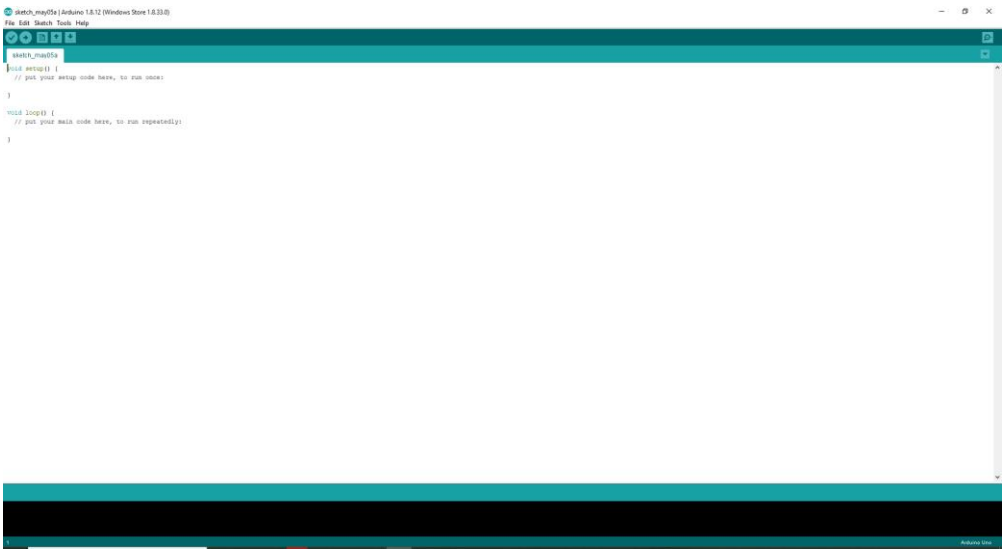
3.3.4 Διάταξη

Για την αποδοτική λειτουργία της κάμερας, πρέπει να τοποθετηθεί σε σταθερό σημείο δίπλα από τον ιμάντα μετακίνησης. Έτσι, αποφασίστηκε ότι η ιδανικότερη θέση που μπορεί να βρίσκεται η κάμερα, είναι απέναντι από τον φωτοηλεκτρικό αισθητήρα, ώστε να αναγνωρίζει το αντικείμενο μόλις σταματήσει την μετακίνηση του ο ιμάντας. Συνεπώς, η διαδικασία που ακολουθείται είναι όταν ο ιμάντας σταματήσει, και εφόσον η κάμερα είναι τοποθετημένη στο κατάλληλο σημείο καταγράφοντας το αντικείμενο καλείται η συνάρτηση ObjectFound. Η συνάρτηση θα επιστρέψει το αντικείμενο που έχει βρεθεί και η διαδικασία θα συνεχιστεί με τον βραχίονα να το επιλέγει και να το ταξινομεί αναλόγως.

3.4 Arduino

3.4.1 Εγκατάσταση Arduino IDE

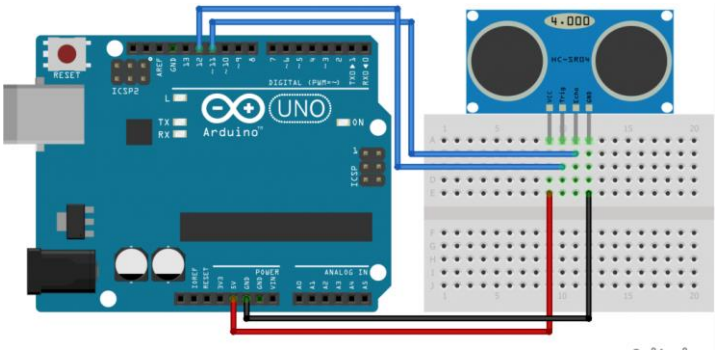
Για την χρήση του Arduino είναι απαραίτητη η εγκατάσταση του λογισμικού Arduino IDE. Μέσω του συγκεκριμένου IDE παρέχεται η δυνατότητα ανάπτυξης λογισμικού σε γλώσσα Wiring για την εκτέλεση του στο Arduino. Αρχικά, κατεβάζουμε το λογισμικό μέσω της επίσημης ιστοσελίδας του Arduino και ακολουθούμε τις οδηγίες εγκατάστασης ανάλογα με τον τρόπο που επιλέξαμε για κατέβασμα. Πλέον, το Arduino είναι έτοιμο για την ανάπτυξη του λογισμικού



Εικόνα 3.4.1: Το interface του Arduino IDE

3.4.2 Σύνδεση αισθητήρα απόστασης

Για την σύνδεση του Ultrasonic Sensor στο Arduino Uno χρησιμοποιήθηκε η εξής συνδεσμολογία σε πλακέτα. Αρχικά συνδέουμε την θύρα Ground(GND) του αισθητήρα με την αντίστοιχη στο Arduino και την θύρα VCC του αισθητήρα με την θύρα 5V στο Arduino. Στην συνέχεια την μια θύρα εισόδου και την μια εξόδου σε δύο διαθέσιμες θύρες(2-13) στο Arduino Uno.



Εικόνα 3.4.2: Σύνδεση του Arduino με τον αισθητήρα απόστασης[30]

3.4.3 Ανάπτυξη και εκτέλεση κώδικα

Μέσω του IDE έχουμε αναπτύξει κώδικα ο οποίος ελέγχει τον αισθητήρα απόστασης και υπολογίζει τις τιμές ανά συγκεκριμένο χρονικό διάστημα (απόσταση) που παράγει ο αισθητήρας. Οι συγκεκριμένες τιμές στην συνέχεια αποστέλλονται μέσω της θύρας USB που είναι συνδεδεμένο το Arduino με τον Η/Υ ώστε να τύχουν περαιτέρω επεξεργασίας από το πρόγραμμα που ελέγχει τον βραχίονα.

```
void loop() {
  long duration, distance;
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = (duration/2) / 29.1;

  if (distance >= 2000 ) {
    Serial.println("Out of range");
  }
  else {
    Serial.print(distance);
    Serial.println(" cm");
  }
  delay(500);
```

}

Η πιο πάνω συνάρτηση αποτελεί το μέρος του προγράμματος που εκτελείται συνεχώς. Αρχικά κάνουμε reset την τιμή της trigPin και στην συνέχεια θέτουμε την trigPin HIGH για 10 μs. Έπειτα αποθηκεύουμε την τιμή της echoPin, η οποία αντιστοιχεί στην χρονική διάρκεια που διαμεσολάβησε μεταξύ της αποστολής ενός ηχητικού κύματος από τον αισθητήρα μέχρι την επιστροφή του. Τέλος, υπολογίζουμε την απόσταση του αντικειμένου που εντοπίστηκε διαιρώντας την διάρκεια διά 2 και στην συνέχεια δια 29.1. Κάνουμε διαίρεση με το δύο, επειδή η διάρκεια σε χρόνο περιλαμβάνει δύο φορές την απόσταση που ψάχνουμε, μια από τον αισθητήρα στο αντικείμενο, και μια από το αντικείμενο στον αισθητήρα. Επίσης, διαιρούμε με 29.1, το οποίο αντιστοιχεί στην ταχύτητα που ταξιδεύει ο ήχος, δηλαδή περίπου 340 m/s. Τέλος, αποστέλλουμε την απόσταση σε cm μέσω της σειριακής θύρας ώστε να διαβαστεί από το πρόγραμμα στην Python.

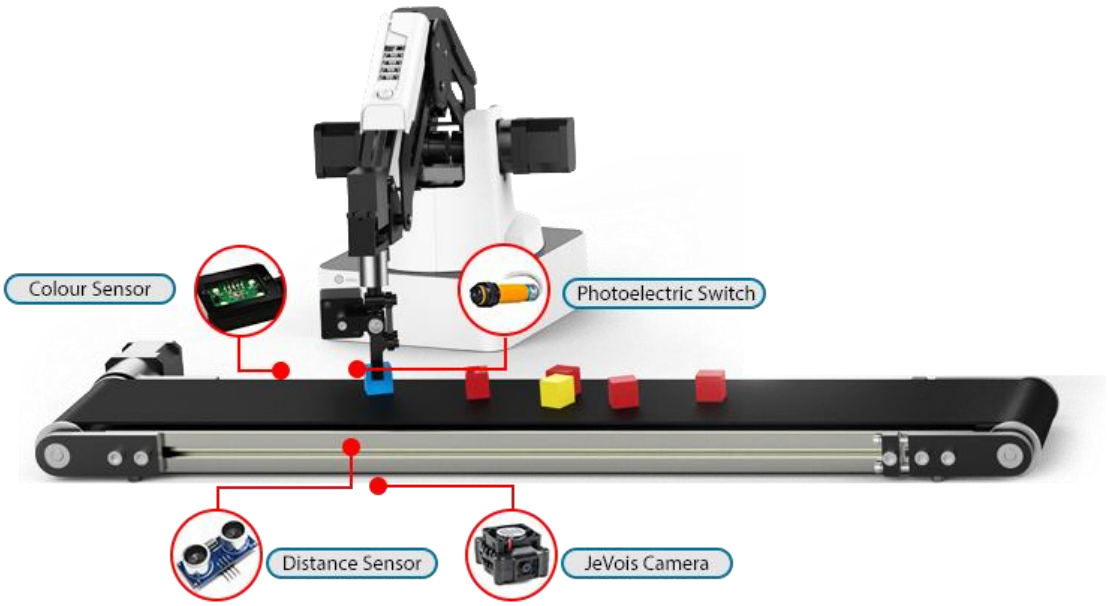
3.4.4 Διασύνδεση με τον βραχίονα

Το Arduino και ο αισθητήρας απόστασης θα αποστέλλουν στον λογισμικό που ελέγχει το βραχίονα την απόσταση από το αντικείμενο που έχει εντοπιστεί ώστε να μετακινείται ανάλογα το άκρο του. Για να γίνει εφικτό αυτό, έχει αναπτυχθεί η συνάρτηση getDistance στην Python όπου τα δεδομένα διαβάζονται από την σειριακή θύρα στην οποία αποστέλλονται τα δεδομένα που προκύπτουν από τον αισθητήρα και μεταβιβάζονται στο υπόλοιπο πρόγραμμα. Στην συνέχεια, ελέγχεται η ορθότητα τους και εάν είναι δυνατό, ο βραχίονας πραγματοποιεί την ανάλογη μετακίνηση.

3.4.5 Διάταξη

Το Arduino και το breadboard μπορούν να τοποθετηθούν οπουδήποτε κοντά στον Η/Υ. Ο αισθητήρας απόστασης μπορεί να τοποθετηθεί σε οποιοδήποτε σημείο δίπλα στον ιμάντα μετακίνησης, στο ύψος των αντικειμένων που περνάνε. Βάση του διαγράμματος και της συγκεκριμένης περίπτωσης χρήσης, έχουμε τοποθετήσει τον αισθητήρα απέναντι από το photoelectric switch. Με τον τρόπο αυτό, όταν το αντικείμενο εντοπίζεται από το switch και ο ιμάντας μετακίνησης θα σταματήσει, διαβάζουμε την τιμή του αισθητήρα, ο οποίος θα έχει ακριβώς μπροστά του το αντικείμενο.

3.5 Τελική Διάταξη



Εικόνα 3.5.1: Σχεδιάγραμμα της τελικής διάταξης

Συνοψίζοντας, όσον αφορά την εφαρμογή μετακίνησης αντικειμένων από τον ιμάντα χρησιμοποιήθηκε η πιο πάνω διάταξη. Τα αντικείμενα τοποθετούνται στα αριστερά του ιμάντα μετακίνησης και όταν ένα αντικείμενο εντοπιστεί στο σημείο που βρίσκεται το Photoelectric switch σταματάει την μετακίνηση του ο ιμάντας. Ο αισθητήρας απόστασης διαβάζει την μεταξύ τους απόσταση και ο βραχίονας μετακινείται στην κατάλληλη θέση. Οι x και z συντεταγμένες είναι σταθερές και άρα συνεχώς οι ίδιες ενώ για να βρεθεί το y, πολλαπλασιάζουμε την απόσταση που επιστρέφει ο αισθητήρας με μια σταθερά, ώστε ο βραχίονας να μετακινηθεί πάνω από το αντικείμενο κατά πλάτος του ιμάντα και έπειτα να το αρπάξει. Τέλος, γίνεται αναγνώριση του αντικείμενου από την κάμερα JeVois και έπειτα ο βραχίονας τοποθετεί το αντικείμενο πάνω από τον αισθητήρα χρώματος και το μεταφέρει στην ανάλογη θέση.

Κεφάλαιο 4

Εκτέλεση πειράματος και αποτελέσματα

4.1 Σενάριο 1: Μεταφορά αντικειμένων από σταθερό σημείο

4.1.1 Κώδικας & Επεξήγηση

Η πρώτη εφαρμογή που αναπτύχθηκε για τον βραχίονα αφορούσε την μετακίνηση αντικειμένων από ένα προεπιλεγμένο σημείο σε κάποιο άλλο. Κατά το σημείο αυτό δεν υπήρχε η δυνατότητα αναγνώρισης των αντικειμένων μέσω των αισθητήρων που χρησιμοποιήθηκαν αργότερα, άρα η εφαρμογή απλά επέλεγε ένα αριθμό αντικειμένων που βρίσκονταν σε προκαθορισμένες θέσεις και τα μετάφερε σε κάποιο άλλο σημείο. Έτσι αναπτύχθηκε ο εξής κώδικας:

```
#Connect Dobot
state = dType.ConnectDobot(api, "", 115200)[0]

if (state == dType.DobotConnect.DobotConnect_NoError):

    # Clean Command Queued
    dType.SetQueuedCmdClear(api)

    # Async Motion Params Setting
    dType.SetHOMEParams(api, 170, 120, 30, 0, isQueued=1)  # home=basket
    dType.SetPTPJointParams(api, 200, 200, 200, 200, 200, 200, 200, 200,
                            isQueued=1)  # the velocity and acceleration of ptp
    dType.SetPTPCommonParams(api, 100, 100, isQueued=1)  # set velocity and
accelaration ratio of ptp mode

    # Async PTP Motion
    offset=30 #insert the offset between each item
```

```
#The program will move 3 items
   for i in range(0, 3):
#move to the specified coordinates
      dType.SetPTPCmd(api, dType.PTPMode.PTPMOVLXYZMode, 105 +
offset, -90, -39, -24, isQueued=1)
#enable the suction cup to grab the item
      dType.SetEndEffectorSuctionCup(api,enableCtrl=1, on=1, isQueued=1)

 # move to the specified coordinates(move up)
dType.SetPTPCmd(api, dType.PTPMode.PTPMOVLXYZMode, 105 + offset, -
90, -10, -24, isQueued=1)
      offset=offset+30
#return to HOME to place the items
      dType.SetHOMECmd(api, temp=0, isQueued=1)
# close suction cup, place the item the its final position
      lastIndex = dType.SetEndEffectorSuctionCup(api, enableCtrl=1, on=0,
isQueued=1)[0]

   # Start to Execute Command Queued
   dType.SetQueuedCmdStartExec(api)

   # Wait for Executing Last Command
   while lastIndex > dType.GetQueuedCmdCurrentIndex(api)[0]:
      dType.dSleep(100)

   # Stop to Execute Command Queued
   dType.SetQueuedCmdStopExec(api)

# Disconnect Dobot
dType.DisconnectDobot(api)
```

Μέσω του πιο πάνω κώδικα πραγματοποιείται αρχικά η διασύνδεση του Η/Υ με το Dobot Magician και γίνεται έλεγχος κατά πόσον αυτή ήταν επιτυχής. Στην συνέχεια θέτονται οι παράμετροι για την εντολή HOME (η τελική θέση που θα τοποθετηθούν τα αντικείμενα), την εντολή PTPJoint, για την ταχύτητα (°/s) και την επιτάχυνση (°/s2) για τον άξονα κίνησης

και την εντολή PTPCommon όπου καθορίζεται το ποσοστό ταχύτητας και επιτάχυνσης για το PTP (Point to Point) . Έπειτα, μέσω του offset καθορίζουμε την απόσταση μεταξύ των αντικειμένων ώστε να μετακινείται ορθά ο βραχίονας σε κάθε επανάληψη και μέσω του βρόγχου for ελέγχουμε τον αριθμό των επαναλήψεων, δηλαδή των αντικειμένων που θα μετακινηθούν από το Dobot. Για να εντοπίσουμε με ακρίβεια τις συντεταγμένες που χρειάζονται για την κάθε εντολή, απαιτείται η μετακίνηση του βραχίονα μέσω της λειτουργίας Operation Panel του Magician Studio και στην συνέχεια

η εισαγωγή τους στις αντίστοιχες εντολές πριν από την εκτέλεση του κώδικα. Αρχίζοντας λοιπόν την εκτέλεση στον βρόγχο, ο βραχίονας μετακινείται στις συντεταγμένες όπου βρίσκεται το αντικείμενο και ενεργοποιείται το Suction Cup ώστε μέσω της εισροής αέρας να αρπάξει το αντικείμενο. Έπειτα ο βραχίονας μετακινείται προς τα πάνω και εκτελείται η εντολή setHomeCmd ούτως ώστε να μετακινηθεί στην θέση που έχουμε θέσει στην αρχή, δηλαδή την τελική θέση των αντικειμένων. Τέλος, όταν ο βραχίονας φτάσει στην θέση αυτή, το suction cup απενεργοποιείται ώστε να αφήσει κάτω το αντικείμενο.

Όταν εκτελεστούν όλες οι επαναλήψεις του κώδικα, γίνεται η αποσύνδεση του Dobot.

4.1.2 Προβλήματα και παρατηρήσεις

Κατά τις εκτελέσεις της πιο πάνω εφαρμογής παρατηρήθηκαν διάφορα προβλήματα. Το πρώτο πρόβλημα που παρατηρήθηκε ήταν αυτό της μετακίνησης του αντικειμένου. Όταν ο βραχίονας μετακινείται πάνω από τον αντικείμενο και ενεργοποιείται το suction cup για να το αρπάξει δεν μπορούμε να γνωρίζουμε εάν η προσπάθεια αυτή ήταν επιτυχής, δηλαδή εάν όντως άρπαξε το αντικείμενο. Αυτό μπορεί να συμβεί για δύο λόγους. Ο πρώτος λόγος αφορά το calibration του βραχίονα, δηλαδή έπειτα από κάποιες επαναλήψεις να έχει χάσει την ακρίβεια του, άρα να μην μετακινείται ακριβώς στις συντεταγμένες που του έχουν δοθεί (δηλαδή πάνω από το αντικείμενο) και να υπάρχει απόκλιση. Ο δεύτερος λόγος αφορά σε λάθος υπολογισμού της παραμέτρου offset, δηλαδή της απόστασης μεταξύ του κάθε αντικειμένου. Σε αυτήν την περίπτωση, στην πρώτη επανάληψη ο βραχίονας θα μετακινόταν ορθά και θα άρπαζε το αντικείμενο, όμως στις υπόλοιπες επαναλήψεις πιθανόν να μην μετακινόταν με ακρίβεια πάνω από το επόμενο αντικείμενο, άρα δεν θα άρπαζε το εκάστοτε αντικείμενο. Το δεύτερο πρόβλημα σχετίζεται με τους ελέγχους που γίνονται στον βραχίονα. Σε περίπτωση όπου δίνονταν συντεταγμένες εκτός του εύρους μετακίνησης του κάθε άξονα ή σε σημείο όπου υπάρχει κάποιο άλλο αντικείμενο, ο βραχίονας θα εκτελούσε την μετακίνηση κάτι που πιθανότατα θα δημιουργούσε πρόβλημα στον βραχίονα.

Το τρίτο πρόβλημα αφορά την αποδοτικότητα του βραχίονα, τόσο σχετικά με τον χρόνο μετακίνησης για κάθε αντικείμενο, όσο και για το ποσοστό επιτυχημένων μετακινήσεων. Κατά την μετακίνηση του κάθε αντικειμένου παρατηρήθηκε ότι υπάρχει μεγάλη καθυστέρηση που οφειλόταν στην εντολή HOMECmd, η οποία χρησιμοποιήθηκε για να μεταφέρει τα αντικείμενα στην τελική τους θέση διότι σε κάθε

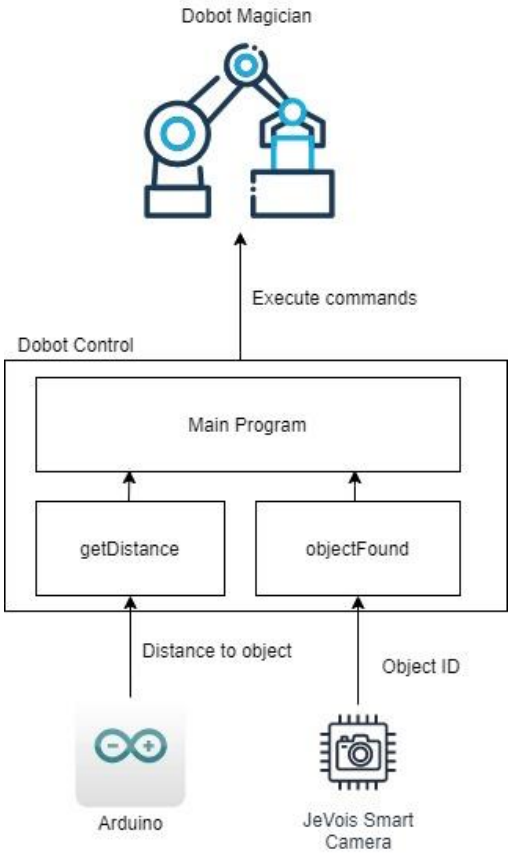
επανάληψη αντί να μετακινείται κατευθείαν στις καθορισμένες συντεταγμένες, μετακινόταν πρώτα στα δύο άκρα του βραχίονα.

Το τελευταίο πρόβλημα που παρατηρήθηκε αφορούσε την εισαγωγή των εντολών σε ουρά (Queue) διότι με αυτό τον τρόπο θα περιόριζε την εκτέλεση ελέγχων. Στα παραδείγματα που παρέχονται από την Dobot όλες οι εντολές εισάγονται σε μια ουρά εκτέλεσης (η παράμετρος IsQueued ισούται πάντοτε με 1). Στο τέλος του εκάστοτε παραδείγματος υπάρχει η εντολή `SetQueuedCmdStartExec`, με την οποία οι εντολές τότε αρχίζουν να εκτελούνται σειριακά, κάτι που δημιουργούσε προβλήματα με την σειρά των ελέγχων και την εξαγωγή δεδομένων των άλλων περιφερειακών συστημάτων.

Στην επόμενη εφαρμογή που αναπτύχθηκε ενσωματώθηκε ο ιμάντας μεταφοράς στο υπόλοιπο σύστημα ώστε να γίνεται συνεχώς τροφοδότηση με αντικείμενα στον βραχίονα και έγιναν προσπάθειες για επίλυση των προβλημάτων που αναλύθηκαν πιο πάνω.

4.2 Σενάριο 2: Μεταφορά αντικειμένων από τον ιμάντα μεταφοράς

Η δεύτερη, και πιο βελτιωμένη εφαρμογή που αναπτύχθηκε για τον βραχίονα αφορούσε την μετακίνηση των αντικειμένων από τον ιμάντα μετακίνησης σε κάποιο άλλο σημείο. Πλέον, εκτός από τον ιμάντα μπορούσαν να χρησιμοποιηθούν και οι δύο αισθητήρες που περιλαμβάνονται στο Conveyor Belt Kit, δηλαδή ο αισθητήρας χρώματος και το photoelectric switch, αλλά και η κάμερα JeVois.



Εικόνα 4.1: Διάγραμμα λογισμικού

Συνεπώς, αναπτύχθηκε το πιο κάτω κώδικας:

4.2.1 Κώδικας και Επεξήγηση

4.2.1.1 Η συνάρτηση ObjectFound

```
def objectFound():
    with serial.Serial(serdev2, 6400, timeout=1) as ser:
        while 1:
            # Read a whole line and strip any trailing line ending character:
            line = ser.readline().rstrip()
            print("received: {}".format(line))
            # Split the line into tokens:
            tok = line.split()
            # Skip if timeout or malformed line:
            if len(tok) < 1: continue
            # Skip if not a standardized "Normal 2D" message:
            # See http://jevois.org/doc/UserSerialStyle.html
```

```
if tok[0].decode('utf-8') != 'N2': continue
# From now on, we hence expect: N2 id x y w h
if len(tok) != 6: continue
# Assign some named Python variables to the tokens:
key, id, x, y, w, h = tok
id = (id.decode('utf-8'))
print(id);
return id;
```

Η πιο πάνω συνάρτηση επιστρέφει το όνομα του αντικειμένου που η JeVois εντόπισε στην σκηνή. Αρχικά, γίνεται σύνδεση του προγράμματος με την σειριακή θύρα όπου βρίσκεται συνδεδεμένη η κάμερα μέσω του USB. Στην συνέχεια, γίνεται ανάγνωση όλης της γραμμής που αποστέλλεται από το πρόγραμμα JeVois Inventor και δημιουργείται ένας πίνακας με το περιεχόμενο της γραμμής. Σε περίπτωση όπου ο πίνακας δεν έχει κανένα στοιχεία το πρόγραμμα συνεχίζει από την αρχή, διότι δεν έχουν αποσταλεί δεδομένα. Σε περίπτωση όπου το πρώτο στοιχείο του πίνακα δεν είναι ‘N2’, το πρόγραμμα επίσης συνεχίζει από την αρχή, διότι το μήνυμα που έχει σταλεί δεν έχει την σωστή δομή(Normal 2D). Τέλος, εάν ο πίνακας δεν έχει 6 στοιχεία, το πρόγραμμα θα ξεκινήσει ξανά από την αρχή, διότι δεν περιλαμβάνονται όλα τα στοιχεία στο μήνυμα που έχει σταλεί. Εάν οι πιο πάνω έλεγχοι ήταν επιτυχημένοι, το μήνυμα έχει την σωστή μορφή και επιστρέφουμε το στοιχείο id, που αντιστοιχεί στο όνομα του αντικειμένου που έχει εντοπιστεί.

4.2.1.2 Η συνάρτηση getDistance

```
def getDistance():
    with serial.Serial(serdev, 9600, timeout=1) as ser:
        while 1:
            # Read a whole line of input
            line = ser.readline().rstrip()
            print("received: {}".format(line))
            # Split the line into tokens:
            tok = line.split()
            # Skip if timeout or malformed line:
```

```
if len(tok) < 1: continue
# From now on, we hence expect: Distance cm
if len(tok) != 2: continue
# Assign some named Python variables to the tokens:
distance, unit= tok
#Skip if distance > 0.3m
if (distance>30): continue
print(distance);
return distance;
```

Η πιο πάνω συνάρτηση επιστρέφει την απόσταση μεταξύ του αισθητήρα και του αντικειμένου που στον ιμάντα μετακίνησης. Αρχικά γίνεται σύνδεση του προγράμματος με την θύρα USB όπου βρίσκεται συνδεδεμένο το Arduino. Στην συνέχεια γίνεται έλεγχος εάν τα δεδομένα έχει ληφθεί σωστά, δηλαδή ότι ο πίνακας περιλαμβάνει δύο δεδομένα. Ακόμη, γίνεται έλεγχος κατά πόσον η απόσταση είναι μεγαλύτερη από 30 εκατοστά, κάτι που θα σήμαινε ότι το αντικείμενο που βρίσκεται μπροστά στον αισθητήρα δεν είναι τοποθετημένο στον ιμάντα αλλά κάπου δίπλα από αυτόν. Τέλος, επιστρέφει την τιμή που εντοπίστηκε.

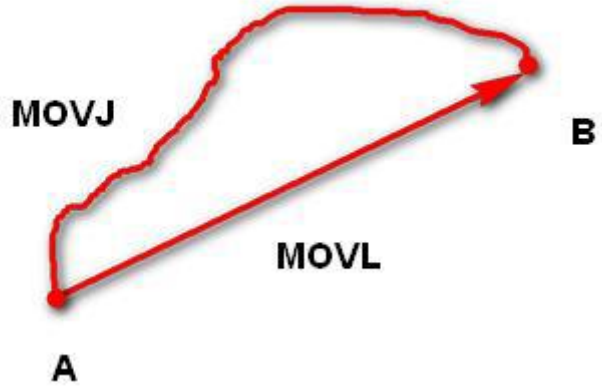
4.2.1.3 Το κυρίως πρόγραμμα

```
connectDobot
if (no_connection_error):
    set PTP_velocity_and_acceleration
    set PTP_velocity_and_accelaration_ratio

    repeat for n items:
        move_conveyor_until_item_arrived
        getDistance()
        moveRoboticArm_toItem
        getObjectId
        moveRoboticArm_AboveItem
        tryToPickUpItem
        if (pickFail<2 times)
            getDistance()
            moveRoboticArm_toItem
            tryToPickUpItem
    getInfraredSensorValue
    if value not 1:
        skip_item
    else:
        moveRoboticArm_AboveColourSensor
        getColor
        moveRoboticArm_toFinalPosition
        checkIfCalibrationRequired
```

Εικόνα 4.2: Ψευδοκώδικας για το κυρίως πρόγραμμα

Η πρώτη επεξεργασία που έγινε, είναι η αλλαγή σε όλες τις εντολές όπου η παράμετρος IsQueued ισούται με 1. Πλέον οι εντολές αυτές δεν εισάγονται στην ουρά εκτέλεσης(IsQueued=0) αλλά εκτελούνται αμέσως , διορθώνοντας έτσι το πρόβλημα που προέκυπτε με τους ελέγχους. Σχετικά με την διαδικασία που ακολουθήθηκε, αρχικά γίνεται φόρτωση της βιβλιοθήκης, πραγματοποιείται η διασύνδεση του Η/Υ με το Dobot Magician και γίνεται έλεγχος κατά πόσον αυτή ήταν επιτυχής. Στην συνέχεια θέτονται οι παράμετροι για την εντολή PTPJoint, για την ταχύτητα (°/s) και την επιτάχυνση (°/s2) για τον άξονα κίνησης και την εντολή PTPCommon όπου καθορίζεται το ποσοστό ταχύτητας και επιτάχυνσης για το PTP.



Εικόνα 4.3: Τρόποι μετακίνησης από σημείο σε σημείο (PTP)

Στην συνέχεια υπολογίζεται η ταχύτητα μετακίνησης του ιμάντα όπου δίνεται ως παράμετρος στην εντολή SetEmotorEx ώστε το conveyor belt να ξεκινήσει την μετακίνηση του. Ο ιμάντας βρίσκεται πλέον ενεργοποιημένος και ελέγχουμε συνεχώς την ένδειξη του Photoelectric switch. Όταν η ένδειξη γίνει 1, τότε απενεργοποιούμε τον ιμάντα, διότι έχει εντοπιστεί κάποιο αντικείμενο από τον αισθητήρα. Σε αυτό το σημείο καλείται η συνάρτηση getDistance που αναλύθηκε πιο πάνω, ώστε να ληφθεί η απόσταση του αντικειμένου που βρίσκεται στον ιμάντα και να μετακινηθεί ορθά ο βραχίονας. Εφόσον ο βραχίονας έχει μετακινηθεί πάνω από το αντικείμενο, ενεργοποιείται το Suction Cup ούτως ώστε να είναι σε θέση να παραλάβει το αντικείμενο ενώ στην συνέχεια γίνεται η αναγνώριση του αντικειμένου από την JeVois Smart Camera. Στο επόμενο βήμα, ο βραχίονας μετακινείται προς τα κάτω, παραλαμβάνει το αντικείμενο και μετακινείται προς τα πάνω. Γίνεται έλεγχος εάν το αντικείμενο έχει ληφθεί επιτυχώς και εάν δεν έχει παραληφθεί, τότε η κίνηση επαναλαμβάνεται υπολογίζοντας ξανά την απόσταση. Εάν έπειτα από δυο προσπάθειες

το αντικείμενο δεν ληφθεί από τον βραχίονα, τότε το απορρίπτει και ξεκινά ξανά την διαδικασία για τον επόμενο αντικείμενο. Στην περίπτωση όπου ο βραχίονας έχει αρπάξει το αντικείμενο, το μεταφέρει πάνω από τον αισθητήρα χρώματος όπου εντοπίζεται το χρώμα του αντικειμένου. Τέλος, το αντικείμενο μετακινείται στην τελική του θέση βάση του χρώματος του ή του είδος του και απενεργοποιούμε το Suction cup και το Photoelectric switch. Γίνεται έλεγχος για το κατά πόσον χρειάζεται ο βραχίονας calibration και εάν δεν χρειάζεται η διαδικασία ξεκινάει ξανά. Όταν ολοκληρωθούν οι επαναλήψεις στον βρόγχο, γίνεται η αποσύνδεση του Dobot.

4.2.2 Προβλήματα και παρατηρήσεις

Αρχικά, έχει διορθωθεί το πρώτο πρόβλημα της προηγούμενης εφαρμογής μέσω του ελέγχου από το photoelectric switch για το κατά πόσο βρίσκεται κάποιο αντικείμενο μπροστά του. Εάν ο βραχίονας δεν παραλάβει το αντικείμενο με την πρώτη προσπάθεια, θα γίνουν ακόμη δύο προσπάθειες. Εάν ούτε αυτές είναι επιτυχείς, τότε θα γίνει παράληψη του συγκεκριμένου αντικειμένου. Σχετικά με το πρόβλημα με τους ελέγχους, πλέον γίνεται έλεγχος πριν γίνει η μετακίνηση του βραχίονα στην ακριβή θέση του αντικειμένου στον ιμάντα, ούτως ώστε σε περίπτωση που η συνάρτηση getDistance επιστρέψει λανθασμένη τιμή να απετράπει η μετακίνηση εκτός ορίου. Ακόμη, έχει αφαιρεθεί η εντολή HomeCmd η οποία προηγουμένως συνέβαλλε αρνητικά στην χρονική απόδοση του βραχίονα και πλέον η μετακίνηση στην τελική θέση γίνεται μέσω της εντολής SetPTPCmd.

4.3 Αποτελέσματα

Μεταφορά αντικειμένων: Μέσω του προγράμματος που υλοποιήθηκε για το δεύτερο σενάριο χρήσης, δηλαδή την μεταφορά αντικειμένων από τον ιμάντα, έγινε δυνατός ο εντοπισμός του κάθε αντικειμένου και στην συνέχεια η παραλαβή και μεταφορά του στο τελικό σημείο από τον βραχίονα. Η διαδικασία αυτή είναι επιτυχής στις περισσότερες περιπτώσεις όμως υπάρχουν περιθώρια βελτίωσης, τόσο σχετικά με την αναγνώριση των αντικειμένων από την JeVois, όσο και για την παραλαβή του κάθε αντικειμένου από τον βραχίονα.

Χρόνος Μεταφοράς: Μέσω της βελτιστοποίησης που έγινε στην $2^{η}$ εφαρμογή σχετικά με την εντολή HomeCmd, ο χρόνος μεταφοράς του κάθε αντικειμένου έχει μειωθεί σε μεγάλο βαθμό και είναι γύρω στα 6 δευτερόλεπτα. Η περισσότερη καθυστέρηση

παρατηρήθηκε στην αναμονή από τον βραχίονα μέχρι να φθάσει νέο αντικείμενο, κάτι που θα μπορεί να διορθωθεί με την συνεχής τροφοδοσία αντικειμένων.

Ακρίβεια Μεταφοράς: Παρατηρήθηκε ότι στον ρομποτικό βραχίονα μειώνεται η ακρίβεια της εκτέλεσης της εντολής μεταφοράς σε κάθε επανάληψη. Βάση παρατήρησης, έχει υπολογιστεί ότι κατά μέσο όρο στις 5 επαναλήψεις του βρόχου ο βραχίονας πρέπει να εκτελεί εντολή calibration (μέσω της εντολής HomeCmd) ούτως ώστε η ακρίβεια μετακίνησης να αυξάνεται για να μειώνονται τα λάθη κατά την παραλαβή των αντικειμένων.

Κεφάλαιο 5

Συμπεράσματα

5.1 Συμπεράσματα

Με την παρούσα διπλωματική εργασία έχει γίνει κατορθωτή η λειτουργία του Dobot Magician για την μεταφορά αντικειμένων μικρών διαστάσεων και βάρους τόσο από σταθερό σημείο όσο και από τον ιμάντα μεταφοράς. Η τελευταία έκδοση της εφαρμογής που αναπτύχθηκε πραγματοποιεί επιτυχώς τις μετακινήσεις και αντιμετωπίζει μέσω των εξαρτημάτων τα προβλήματα και τους περιορισμούς που αρχικά υπήρχαν.

Κατά την πορεία υλοποίησης της διπλωματικής εργασίας διαπιστώθηκαν τόσο τα προβλήματα που συνεχώς προέκυπταν (και σε επίπεδο λογισμικού αλλά και στο περιβάλλον του Dobot), όσο και οι ευκαιρίες που υπάρχουν για περαιτέρω ανάπτυξη παρόμοιων εφαρμογών.

Ο Dobot Magician που χρησιμοποιήθηκε αποτελεί ένα πολύ καλό βραχίονα, στον οποίο μπορούν να αναπτυχθούν περισσότερες εφαρμογές για να καλύψουν και άλλα σενάρια χρήσης ή ακόμη και για προσομοίωση μεγαλύτερης κλίμακας προβλημάτων.

5.2 Μελλοντική Εργασία

Σχετικά με την μελλοντική εργασία, αρχικά θα μπορούσε να γίνει περαιτέρω βελτιστοποίηση του προγράμματος που έχει αναπτυχθεί πιο πάνω, όσον αφορά τους ελέγχους που γίνονται και το calibration του κάθε άξονα. Ακόμη, θα μπορούσε να βρεθεί μια αυτοματοποιημένη λύση για την αποφυγή αντικειμένων που βρίσκονται στην πορεία του βραχίονα καθώς εκτελεί μια μετακίνηση. Σχετικά με την κάμερα JeVois, θα μπορούσε να χρησιμοποιηθεί κάτι ανάλογο που όμως να έχει περισσότερες δυνατότητες, ειδικά σε συνθήκες χαμηλού φωτισμού ή ακόμη και να αναπτυχθούν άλλα modules που να είναι πιο αποτελεσματικά στην συγκεκριμένη περίπτωση χρήσης.

Επιπρόσθετα, μπορεί να γίνει επέκταση του παρόντος συστήματος χρησιμοποιώντας επιπρόσθετο εξοπλισμό, κάτι που θα οδηγούσε στην επέκταση των δυνατοτήτων και των σεναρίων χρήσης του βραχίονα. Αρχικά, θα μπορούσε να προστεθεί ένας δεύτερος βραχίονας στην αρχή του ιμάντα μετακίνησης, ώστε να προσθέτει συνέχεια αντικείμενα στον ιμάντα. Επίσης, θα μπορούσε να γίνει χρήση πιο εξειδικευμένων αισθητήρων είτε κατευθείαν μέσω του Dobot Magician, είτε σε σύνδεση με το Arduino. Τέτοιο αισθητήρες θα ήταν αισθητήρας χρώματος για την αναγνώριση περισσότερων από 3 χρωμάτων και φωτογραφικοί αισθητήρες.

Βιβλιογραφία

[1] dobot.cc, “Industry 4.0 in Education”. [Online]. Available at: https://www.dobot.cc/downloadcenter/dobot-magician.html?sub_cat=149#sub-download

[2] ΠΡΟΚΟΠΗΣ ΚΙΟΥΣΗΣ, “Ρομποτικός Βραχίονας 4 Βαθμών Ελευθερίας”

[3] robots.ieee.org, “Types of Robots”. [Online]. Available at: https://robots.ieee.org/learn/types-of-robots/

[4] dyson.co.uk, “Dyson 360 Heurist”. [Online]. Available at: https://www.dyson.co.uk/robot-vacuums/dyson-360-heurist-overview.html

[5] moley.com, “The world’s first robotic kitchen”. [Online]. Available at: https://www.moley.com

[6] robots.ieee.org/, “EMYS - ROBOTS”. [Online]. Available at: https://robots.ieee.org/robots/emys/

[7] spectrum.ieee.org, “New Da Vinci Xi Surgical Robot”. [Online]. Available at: https://spectrum.ieee.org/automaton/robotics/medical-robots/new-da-vinci-xi-surgical-robot

[8] businesswire.com, “Sarcos Robotics Begins Delivery of Guardian XO Full-Body, Force-Multiplying Industrial Exoskeleton Alpha Units”. [Online]. Available at: https://www.businesswire.com/news/home/20191210005063/en/Sarcos-Robotics-Begins-Delivery-Guardian-XO-Full-Body

[9] wikipedia.org, “PackBot”. [Online]. Available at: https://en.wikipedia.org/wiki/PackBot#/media/File:Robot_501585_fh000026.jpg

[10] naval-technology.com, “Autonomous underwater robots: from Swordfish to the Orca”. [Online]. Available at: https://www.naval-technology.com/features/autonomous-underwater-robots-navy/attachment/bpauv-mp_from_hsv/

[11] teslarati.com, “Tesla reportedly flies in 6 planes’ worth of robots in latest Model 3 push”. [Online]. Available at: https://www.teslarati.com/tesla-model-3-push-six-planes-robots-eu/

[12] spectrum.ieee.org, “Brad Porter, VP of Robotics at Amazon, on Warehouse Automation, Machine Learning, and His First Robot”. [Online]. Available at: https://spectrum.ieee.org/automaton/robotics/industrial-robots/interview-brad-porter-vp-of-robotics-at-amazon

[13] robots.com, “What Are The Main Types Of Robots?”. [Online]. Available at https://www.robots.com/faq/what-are-the-main-types-of-robots

[14] mars.nasa.gov, “Curiosity's Selfie at 'Aberlady' and 'Kilmarie'”. [Online]. Available at: https://mars.nasa.gov/resources/22495/curiositys-selfie-at-aberlady-and-kilmarie/?site=msl

[15] wikipedia.org, “Sophia (robot)”. [Online]. Available at: https://en.wikipedia.org/wiki/Sophia_(robot)

[16] bostondynamics.com, “LEGACY ROBOTS - BigDog”. [Online]. Available at: https://www.bostondynamics.com/legacy

[17] dobot.cc, “DOBOT Magician Specifications”. [Online]. Available at: https://www.dobot.cc/dobot-magician/specification.html

[18][37] dobot.cc, “[OFFICIAL] DOBOT Magician - Lightweight Intelligent Training Robotic Arm ”. [Online]. Available at: https://www.dobot.cc/dobot-magician/product-overview.html

[19] techpowerup.com, “Programming/Using The Dobot Magician”. [Online]. Available at: https://www.techpowerup.com/review/dobot-magician/4.html

[20][21][22] jevois.org, “JeVois Smart Machine Vision Camera”. [Online]. Available at: http://www.jevois.org

[23] jevois.org, “JeVois case mounting guidelines”. [Online]. Available at: http://jevois.org/doc/CaseMounting.html

[24] store.arduino.cc, “ARDUINO UNO”. [Online]. Available at: https://store.arduino.cc/usa/arduino-uno-rev3

[25] wikipedia.org, “Arduino Uno”. [Online]. Available at: https://en.wikipedia.org/wiki/Arduino_Uno

[26] sparkfun.com, “Ultrasonic Distance Sensor - HC-SR04”. [Online]. Available at: https://www.sparkfun.com/products/15569

[27] c4e.org.cy, “Ultrasonic Distance Sensor”. [Online]. Available at: https://www.c4e.org.cy/equipment/eq-viewer?cat=sensors&eq=ultrasonic_sensor

[28][29] wikipedia.org, “3D printing”. [Online]. Available at: https://en.wikipedia.org/wiki/3D_printing

[30] maxphi.com, “Arduino Ultrasonic Sensor?”. [Online]. Available at https://www.maxphi.com/ultrasonic-sensor-interfacing-arduino-tutorial

[31] tss.trelleborg.com, “Application Examples for Cartesian Robots?”. [Online]. Available at https://www.tss.trelleborg.com/en/upcoming/robotics/application-examples/cartesian-robot

[32][33] machinedesign.com, “What’s the Difference Between Industrial Robots??”. [Online]. Available at https://www.machinedesign.com/markets/robotics/article/21835000/whats-the-difference-between-industrial-robots

[34] Wikipedia.com, “SCARA?”. [Online]. Available at https://en.wikipedia.org/wiki/SCARA

[35] Wikipedia.com, “Articulated robot”. [Online]. Available at https://en.wikipedia.org/wiki/Articulated_robot#/media/File:KUKA_robot_for_flat_glas_handling.jpg

[36] jevois.org, “Standardized serial messages formatting”. [Online]. Available at http://jevois.org/doc/UserSerialStyle.html

Παράρτημα Α: Κώδικας Python για την εφαρμογή 1

```python
import threading
import DobotDllType as dType
import serial
import time

serdev = 'COM5:' # serial device of JeVois
def isCubeLoaded():
   with serial.Serial(serdev, 115200, timeout=1) as ser:
      while 1:
      # Read a whole line and strip any trailing line ending character:
         line = ser.readline().rstrip()
         print ("received: {}".format(line))
      # Split the line into tokens:
         tok = line.split()
         # Skip if timeout or malformed line:
         if len(tok) < 1: continue
         # Skip if not a standardized "Normal 2D" message:
         # See http://jevois.org/doc/UserSerialStyle.html
         if tok[0].decode('utf-8') != 'N2': continue
         # From now on, we hence expect: N2 id x y w h
         if len(tok) != 6: continue
         # Assign some named Python variables to the tokens:
         key, id, x, y, w, h = tok
         id=(id.decode('utf-8'))
         print (id);
         if (id=="sdcard.png"):
            return 1;


CON_STR = {
    dType.DobotConnect.DobotConnect_NoError:  "DobotConnect_NoError",
    dType.DobotConnect.DobotConnect_NotFound: "DobotConnect_NotFound",
    dType.DobotConnect.DobotConnect_Occupied: "DobotConnect_Occupied"}

#Load Dll
api = dType.load()

#Connect Dobot
state = dType.ConnectDobot(api, "", 115200)[0]
print("Connect status:",CON_STR[state])

if (state == dType.DobotConnect.DobotConnect_NoError):

    # Clean Command Queued
    dType.SetQueuedCmdClear(api)

    # Async Motion Params Setting
    dType.SetHOMEParams(api, 170, 120, 30, 0, isQueued=1)  # home=basket
    dType.SetPTPJointParams(api, 200, 200, 200, 200, 200, 200, 200, 200,
                            isQueued=1)  # the velocity and acceleration of ptp
    dType.SetPTPCommonParams(api, 100, 100, isQueued=1)  # set velocity and accelaration ratio of ptp mode
```

```
52
53      # Async PTP Motion
54      offset=30
55      for i in range(0, 3):
56          dType.SetPTPCmd(api, dType.PTPMode.PTPMOVLXYZMode, 105 + offset, -90, -39, -24, isQueued=1)
57          dType.SetEndEffectorSuctionCup(api,enableCtrl=1, on=1, isQueued=1)  # enable suction cup
58          dType.SetPTPCmd(api, dType.PTPMode.PTPMOVLXYZMode, 105 + offset, -90, -10, -24, isQueued=1)
59          offset=offset+30
60          dType.SetHOMECmd(api, temp=0, isQueued=1)
61          lastIndex = dType.SetEndEffectorSuctionCup(api, enableCtrl=1, on=0, isQueued=1)[0]  # disable suction cup
62
63      # Start to Execute Command Queued
64      dType.SetQueuedCmdStartExec(api)
65
66      # Wait for Executing Last Command
67      while lastIndex > dType.GetQueuedCmdCurrentIndex(api)[0]:
68          dType.dSleep(100)
69
70      # Stop to Execute Command Queued
71      dType.SetQueuedCmdStopExec(api)
72
73  # Disconnect Dobot
74
75  dType.DisconnectDobot(api)
76
```

Παράρτημα Β: Κώδικας Python για την εφαρμογή 2

```python
import threading
import DobotDllType as dType
import serial
import time

serdev = 'COM5:'  # serial device of JeVois
serdev2 = 'COM6:'  # serial device of Arduino


def objectFound():
    with serial.Serial(serdev2, 6400, timeout=1) as ser:
        while 1:
            # Read a whole line and strip any trailing line ending character:
            line = ser.readline().rstrip()
            print("received: {}".format(line))
            # Split the line into tokens:
            tok = line.split()
            # Skip if timeout or malformed line:
            if len(tok) < 1: continue
            # Skip if not a standardized "Normal 2D" message:
            # See http://jevois.org/doc/UserSerialStyle.html
            if tok[0].decode('utf-8') != 'N2': continue
            # From now on, we hence expect: N2 id x y w h
            if len(tok) != 6: continue
            # Assign some named Python variables to the tokens:
            key, id, x, y, w, h = tok
            id = (id.decode('utf-8'))
            print(id);
            return id;

def getDistance():
    with serial.Serial(serdev, 9600, timeout=1) as ser:
        while 1:
            # Read a whole line of input
            line = ser.readline().rstrip()
            print("received: {}".format(line))
            # Split the line into tokens:
            tok = line.split()
            # Skip if timeout or malformed line:
            if len(tok) < 1: continue
            # From now on, we hence expect: Distance cm
            if len(tok) != 2: continue
            # Assign some named Python variables to the tokens:
            distance, unit= tok
           #Skip if distance > 1m
           if (distance>100): continue
            print(distance);
            return distance;

CON_STR = {
    dType.DobotConnect.DobotConnect_NoError:  "DobotConnect_NoError",
    dType.DobotConnect.DobotConnect_NotFound: "DobotConnect_NotFound",
```

```
    dType.DobotConnect.DobotConnect_Occupied: "DobotConnect_Occupied"}

#Load Dll
api = dType.load()

#Connect Dobot
state = dType.ConnectDobot(api, "", 115200)[0]
print("Connect status:",CON_STR[state])

if (state == dType.DobotConnect.DobotConnect_NoError):

    # Clean Command Queued
    dType.SetQueuedCmdClear(api)

    # Async Motion Params Setting
    dType.SetPTPJointParams(api, 200, 200, 200, 200, 200, 200, 200, 200,
                            isQueued=1)  # the velocity and acceleration of ptp
    dType.SetPTPCommonParams(api, 100, 100, isQueued=1)  # set velocity and accelaration ratio of ptp mode

    STEP_PER_CRICLE = 360.0 / 1.8 * 10.0 * 16.0
    MM_PER_CRICLE = 3.1415926535898 * 36.0
    vel = float(50) * STEP_PER_CRICLE / MM_PER_CRICLE
    i=0;
    for i in range (0,1):
        lastIndex= dType.SetEMotorEx(api,0,1,int(vel),0)
        dType.SetInfraredSensor(api,1,2)
        stopConveyor=0;
        print(stopConveyor);
        while (stopConveyor!=1):
            stopConveyor= dType.GetInfraredSensor(api,2)[0]; #object detection

        lastIndex= dType.SetEMotorEx(api,0,0,int(vel),0);

        #Get value of distance sensor
        distance=getDistance()

        dType.SetEndEffectorSuctionCup(api, enableCtrl=1, on=1, isQueued=0) #enable vacuum pump - suction cup
        dType.SetPTPCmd(api, dType.PTPMode.PTPMOVLXYZMode, 33, -239, 90, 26, isQueued=0)

        #Identify object with JeVois Camera
        objectId=objectFound()

        offset=distance*0.2; #set the offset
        dType.SetPTPCmd(api, dType.PTPMode.PTPMOVLXYZMode, 33, -239+offset, 17, 26, isQueued=0)
        dType.SetPTPCmd(api, dType.PTPMode.PTPMOVLXYZMode, 33, -239+offset, 90, 26, isQueued=0)

        c=0;
        for c in range(0,2):
            #Check if object is picked up
            isPicked=dType.GetInfraredSensor(api,2)[0]
            if (isPicked!=1):
```

```
                    distance=getDistance() #recalculate distance
                    dType.SetPTPCmd(api, dType.PTPMode.PTPMOVLXYZMode, 33, -239+offset, 17, 26, isQueued=0) #try to pick the item again
                    dType.SetPTPCmd(api, dType.PTPMode.PTPMOVLXYZMode, 33, -239+offset, 90, 26, isQueued=0)

        isPicked=dType.GetInfraredSensor(api,2)[0]
        if (isPicked!=1):
            lastIndex= dType.SetEMotorEx(api,0,1,int(vel),0)    #skip the item
            dType.dSleep(500)
            lastIndex= dType.SetEMotorEx(api,0,0,int(vel),0)
        else:
            dType.SetPTPCmd(api, dType.PTPMode.PTPMOVLXYZMode, -25, -186, 35, 26, isQueued=0) #move the object above color sensor
            dType.dSleep(1000)
            dType.SetColorSensor(api,1,1)
            dType.dSleep(1000)
            color = dType.GetColorSensor(api)
            print(color) #identify color
            dType.dSleep(2000)
            dType.SetColorSensor(api,0,1)
            dType.SetPTPCmd(api, dType.PTPMode.PTPMOVLXYZMode, -25, -186, 50, 26, isQueued=0)
            dType.SetPTPCmd(api, dType.PTPMode.PTPMOVLXYZMode, 118, -200, 90, 26, isQueued=0)
            if (color == 'r'):
                dType.SetPTPCmd(api, dType.PTPMode.PTPMOVLXYZMode, 184, -90, 60, 26, isQueued=0)
            elif (objectId =='cube'):
                dType.SetPTPCmd(api, dType.PTPMode.PTPMOVLXYZMode, 200, -90, 60, 26, isQueued=0)
            else:
                dType.SetPTPCmd(api, dType.PTPMode.PTPMOVLXYZMode, 160, -90, 60, 26, isQueued=0)
            dType.dSleep(4000)
            dType.SetEndEffectorSuctionCup(api, enableCtrl=1, on=0, isQueued=0)
            dType.SetInfraredSensor(api,0,2)

            dType.SetPTPCmd(api, dType.PTPMode.PTPMOVLXYZMode, 160, -90, 60, 26, isQueued=0) # robotic arm to the final position
            dType.dSleep(1000)

            #get current x,y,z
            pose=dType.GetPose(api)
            if (pose[0]!= 160 or pose[1] != -90 or pose[2] != 60): #check if calibration required
                dType.SetHOMECmd(api, temp=0, isQueued=0) #calibrate

# Disconnect Dobot

dType.DisconnectDobot(api)

```

Παράρτημα Γ: Κώδικας Arduino

distance_sensor

```
#define trigPin 8
#define echoPin 7

void setup() {
  Serial.begin (9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}

void loop() {
  long duration, distance;
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = (duration/2) / 29.1;

  if (distance >= 2000 ) {
    Serial.println("Out of range");
  }
  else {
    Serial.print(distance);
    Serial.println(" cm");
  }
  delay(500);
}
```