ΑΛΕΞΑΝΔΡΕΙΟ ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΘΕΣΣΑΛΟΝΙΚΗΣ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΑΥΤΟΜΑΤΙΣΜΟΥ

Πτυχιακή εργασία

Ανάπτυξη Συστήματος Μηχανικής Όρασης

Σπουδαστής
Τρίγκας Δημήτριος

Επιβλέπων
Επ. Καθ. Τσαγκάρης Απόστολος

Θεσσαλονίκη 2017

Πτυχιακή εργασία

Ανάπτυξη Συστήματος Μηχανικής Όρασης

ΑΛΕΞΑΝΔΡΕΙΟ ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΘΕΣΣΑΛΟΝΙΚΗΣ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΑΥΤΟΜΑΤΙΣΜΟΥ
Τρίγκας Δημήτριος

Περιεχόμενα

Εισαγωγή

Τα τελευταία χρόνια ολοένα και περισσότερο χρησιμοποιείται η εικόνα ως μέσω εισόδου για συσκευές ελέγχου και λήψης αποφάσεων. Σημαντικό ρόλο στην πορεία αυτή έπαιξε η εξέλιξη προγραμμάτων και βιβλιοθηκών που επεξεργάζονται δεδομένα εικόνας. Σημαντικότερη ίσως συμβολή είχε η δημιουργία της βιβλιοθήκης συναρτήσεων επεξεργασίας εικόνας OpenCV που είναι το ακρωνύμιο της Open Computer Vision (όραση υπολογιστή ανοιχτού κώδικα). Η μεγάλη συμβολή της βιβλιοθήκης οφείλεται τόσο στις συναρτήσεις που διαθέτει όσο και στον ανοιχτό κώδικα που είναι χρηστικός σε όλες τις πλατφόρμες.

Σημαντικά παραδείγματα της χρήσης εικόνας ως είσοδο σε συστήματα επεξεργασίας δεδομένων, που συναντώνται τόσο στην έρευνα όσο πλέον και στην καθημερινή ζωή είναι η αναγνώριση προσώπου (face recognition), η αναγνώριση δακτυλικού αποτυπώματος (finger recognition), η αναγνώριση μοτίβου (pattern recognition), η αναγνώριση πινακίδας (plate recognition), η αναγνώριση λωρίδας (lane recognition), η αποφυγή εμποδίων (object avoidance) και πολλά άλλα. Τα παραπάνω συναντώνται σε συστήματα ασφαλείας, σε κινητά τηλέφωνα, στη κρατική ασφάλεια και ασφάλεια αεροδρομίων, σε ιδιωτικές επιχειρήσεις , σε γραμμές παραγωγής, στον έλεγχο ποιότητας και πολλά άλλα. Τα τελευταία χρόνια γίνονται προσπάθειες ενσωμάτωσης της παραπάνω τεχνολογίας στις μεταφορές, τόσο σε θέματα ασφάλειας όσο και στην αυτόνομη οδήγηση οχημάτων.

Abstract

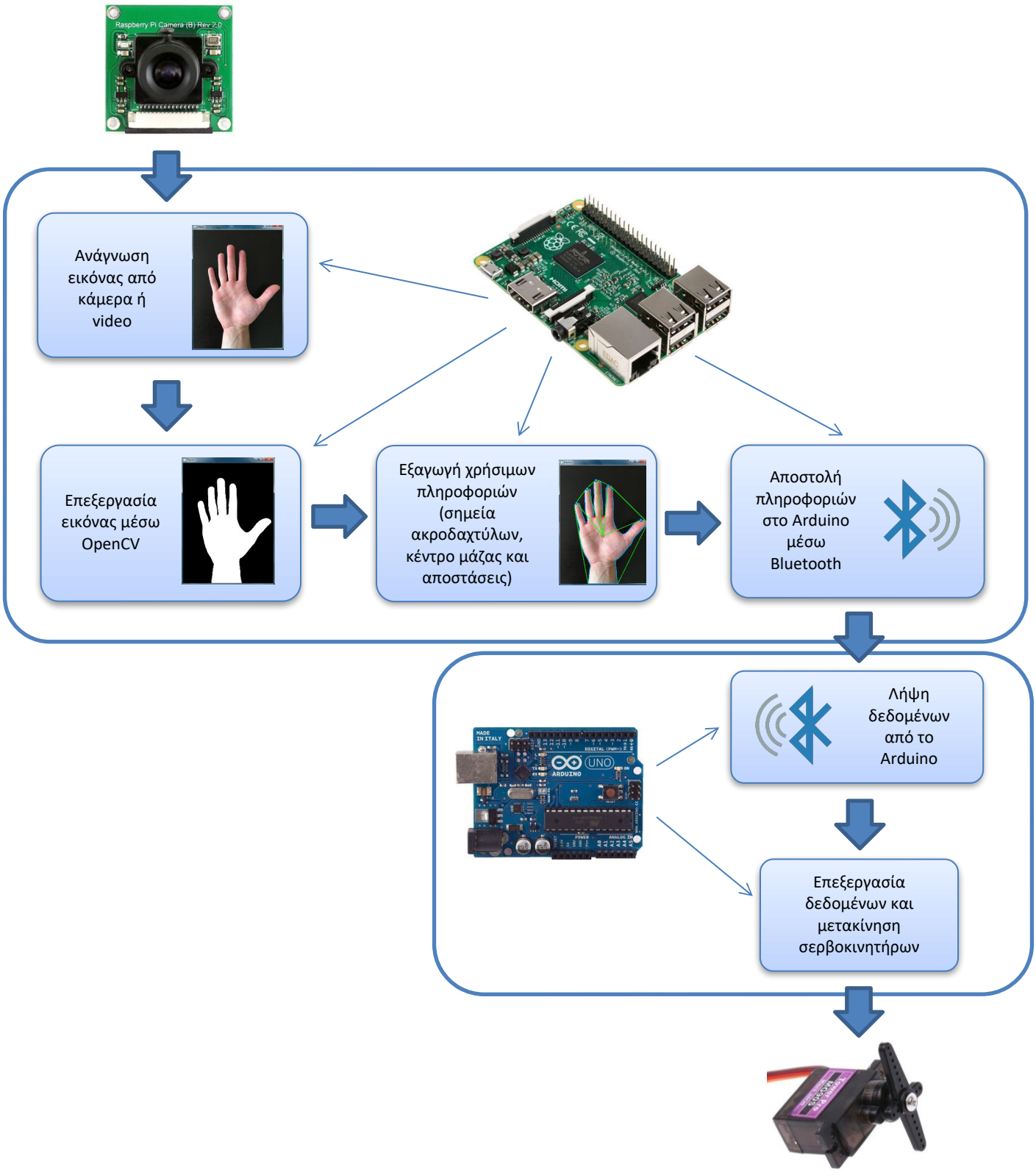
In recent years, the image has increasingly been used as an input for control and decision-making devices. An important role in this process was played by the development of programs and libraries that process image data. The creation of image editing library OpenCV, which is the acronym of Open Computer Vision, was perhaps the most important contribution in the above mentioned technology. The great contribution of the library is due both to its functions and to the cross-platform open source code.

Important examples of image use as input to data processing systems that are found in both research and everyday life are face recognition, finger recognition, pattern recognition, plate recognition, lane recognition, object avoidance, and more. These are found in security systems, mobile phones, state and airport security, businesses, production lines, quality control, and much more. In recent years, efforts have been made to integrate the above technology into transport, both in safety and in the autonomous driving of vehicles.

Περίληψη

Σκοπός αυτής της πτυχιακής εργασίας είναι η ανάπτυξη μιας εφαρμογής που βασίζεται στην οπτική αναγνώριση από μια μηχανή και ο αποτελεσματικός έλεγχος ενός συστήματος. Πιο συγκεκριμένα, γίνεται η οπτική αναγνώριση της παλάμης ενός χεριού και με βάση τη θέση και τις κινήσεις των δακτύλων, γίνεται ο έλεγχος των θέσεων ενός συστήματος πέντε σερβοκινητήρων. Για την αποτύπωση της εικόνας χρησιμοποιήθηκε μια κάμερα που βρίσκεται σε διασύνδεση με ένα μικροϋπολογιστή Raspberry Pi. Η επεξεργασία της εικόνας και η οπτική αναγνώριση πραγματοποιείται στον μικροϋπολογιστή. Για τη μετακίνηση των σερβοκινητήρων χρησιμοποιήθηκε η υπολογιστική πλατφόρμα Arduino, η οποία διαθέτει ενσωματωμένο ένα μικροελεγκτή. Ο χρήστης , με τις κινήσεις των δακτύλων του μπροστά στην κάμερα, έχει τη δυνατότητα να ελέγχει ανεξάρτητα κάθε ένα σερβοκινητήρα, όπου ο καθένας αντιστοιχεί σε κάθε ένα δάχτυλο ξεχωριστά. Ιδιαίτερο χαρακτηριστικό της εφαρμογής, είναι μετάβαση από τα ογκώδη και δύσχρηστα υπολογιστικά συστήματα, που είναι οι υπολογιστές, σε μια συσκευή οπτικής αναγνώρισης σε μέγεθος πιστωτικής κάρτας που την καθιστά φορητή και φορετή. Η φορετότητα είναι και ο λόγος που η επικοινωνία ανάμεσα στη συσκευή αναγνώρισης Raspberry Pi και του Arduino, πραγματοποιείται με την ασύρματη τεχνολογία Bluetooth. Η ανάπτυξη του κώδικα του μικρουπολογιστή Raspberry Pi έγινε σε γλώσσα προγραμματισμού Python, ενώ της υπολογιστικής πλατφόρμας Arduino έγινε στη γλώσσα Wiring, που είναι η C++ με ενσωματωμένες βιβλιοθήκες του Arduino.

Συνολική διεργασία της εφαρμογής



Σχήμα 1. Η συνολική διεργασία συνοπτικά

1. Επισκόπηση

Η μηχανική όραση (MV) είναι η τεχνολογία και οι μέθοδοι που χρησιμοποιούνται για την παροχή αυτοματοποιημένης επιθεώρησης και ανάλυσης με βάση την απεικόνιση. Τα πεδία εφαρμογών είναι συνήθως στη βιομηχανία χωρίς όμως να περιορίζονται αυστηρά σε αυτή. Χρησιμοποιείται για αυτόματη επιθεώρηση, έλεγχο διεργασιών και σε πολλές περιπτώσεις για την καθοδήγηση ρομπότ. Είναι ένας όρος που περιλαμβάνει ένα μεγάλο αριθμό τεχνολογιών, όπως προϊόντα λογισμικού και υλικού, ολοκληρωμένα συστήματα, δράσεις, μεθόδους και εμπειρογνωμοσύνη. Η μηχανική όραση (machine vision) πρέπει να θεωρηθεί ότι διαφέρει από την όραση του υπολογιστή (computer vision), η οποία εντάσσεται στην επιστήμη των υπολογιστών. Προσπαθεί να ενσωματώσει τις υπάρχουσες τεχνολογίες σε νέες μεθόδους και να τις εφαρμόσει για την επίλυση προβλημάτων στον πραγματικό κόσμο.

Οι ορισμοί του όρου "όραση μηχανής" ποικίλλουν, αλλά όλες περιλαμβάνουν την τεχνολογία και τις μεθόδους που χρησιμοποιούνται για την εξαγωγή πληροφοριών από μια εικόνα και την εφαρμογή δράσεων, σε αντίθεση με την επεξεργασία εικόνας, όπου η έξοδος είναι μια άλλη εικόνα. Οι πληροφορίες που εξάγονται μπορεί να είναι ένα απλό σήμα όπως καλό / ελαττωματικό ή ένα πιο σύνθετο σύνολο δεδομένων όπως η ταυτότητα, η θέση τα χαρακτηριστικά και ο προσανατολισμός κάθε αντικειμένου σε μια εικόνα. Οι πληροφορίες μπορούν να χρησιμοποιηθούν για εφαρμογές όπως ο έλεγχος ρομπότ, η καθοδήγηση διαδικασιών στη βιομηχανία, ή ακόμα και για την παρακολούθηση της ασφάλειας.

1.1. Στοιχεία και στάδια εφαρμογής της μηχανικής όρασης.

Τα στοιχεία ενός συστήματος μηχανικής όρασης περιλαμβάνουν συνήθως φωτισμό, κάμερα ή άλλο σύστημα απεικόνισης, επεξεργαστή, λογισμικό και συσκευές εξόδου. Η συσκευή απεικόνισης (π.χ. κάμερα) μπορεί είτε να είναι ξεχωριστή από την κύρια μονάδα επεξεργασίας εικόνας είτε να συνδυαστεί με αυτήν, οπότε ο συνδυασμός αποκαλείται γενικά μια έξυπνη κάμερα ή έξυπνος αισθητήρας.

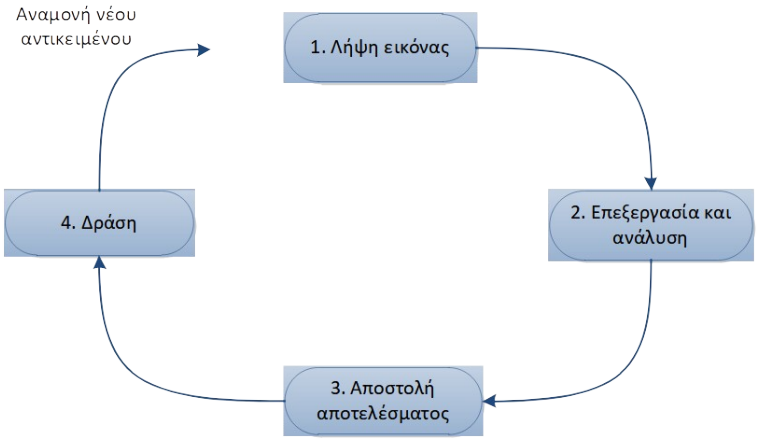
Η εφαρμογή της μηχανικής όρασης σε οποιαδήποτε διαδικασία, αποτελείται από τέσσερα βασικά στάδια

1. Λήψη της εικόνας

2. Επεξεργασία και ανάλυση

3. Αποστολή των δεδομένων ενδιαφέροντος

4. Δράση



Σχήμα 2. Στάδια εφαρμογής Μηχανικής Όρασης

Αρχικό βήμα για την εφαρμογή της μηχανικής όρασης, είναι η λήψη μιας εικόνας του αντικειμένου ή της περιοχής ενδιαφέροντος, από κατάλληλες κάθε φορά κάμερες. Ακολουθεί η εφαρμογή φίλτρων στην εικόνα για την προετοιμασία της, προς επεξεργασία. Αφού γίνει κατάλληλη επεξεργασία, εξάγονται τα δεδομένα ενδιαφέροντος και τελικά γίνεται η εφαρμογή της δράσης σύμφωνα με τα δεδομένα που έχουν ληφθεί.

1.2. Πεδία εφαρμογών

Τα πεδία εφαρμογών της μηχανικής όρασης, ανάλογα με το είδος των εξαγόμενων δεδομένων, μπορούν να διαχωριστούν σε τέσσερις μεγάλες κατηγορίες

1. *Εντοπισμός.*

Ελέγχεται συνήθως η ύπαρξη ή μη του αντικειμένου ενδιαφέροντος στο οπτικό πεδίο της κάμερας, ή σε προκαθορισμένη θέση. Η έξοδος συνήθως είναι εντοπίστηκε ή δεν εντοπίστηκε.

2. *Μέτρηση.*

Γίνεται η μέτρηση χαρακτηριστικών διαστάσεων του αντικειμένου ενδιαφέροντος, καθώς και η θέση του. Έξοδος είναι οι μετρήσεις πάνω στο αντικείμενο, και η θέση του.

3. *Επιθεώρηση.*

Γίνεται ο έλεγχος συμβατότητας ή ταύτισης του αντικειμένου ενδιαφέροντος ως προς ένα πρωτότυπο, ή ως προς προκαθορισμένες απαιτήσεις. Η έξοδος είναι το αποτέλεσμα συμβατότητας ή τα σημεία διαφοροποίησης.

4. *Αναγνώριση.*

Γίνεται η αναγνώριση ενός αντικειμένου, ή χαρακτηριστικών του, μέσα σε μια εικόνα. Η έξοδος εδώ είναι είτε η θετική αναγνώριση ή μη, είτε τα χαρακτηριστικά ενδιαφέροντος.

1.3. Χρήση δεδομένων εξόδου

Η πιο κοινή έξοδος από τα συστήματα αυτόματης επιθεώρησης είναι αποφάσεις της μορφής αποδεκτό/απορριπτέο. Αυτές οι αποφάσεις ενδέχεται να ενεργοποιήσουν μηχανισμούς που απορρίπτουν τα ελαττωματικά στοιχεία ή να σημάνουν συναγερμούς. Άλλες συνηθισμένες έξοδοι περιλαμβάνουν πληροφορίες θέσης και προσανατολισμού αντικειμένων για συστήματα καθοδήγησης ρομπότ. Επιπλέον, οι τύποι εξόδου περιλαμβάνουν αριθμητικά δεδομένα μέτρησης, δεδομένα που διαβάζονται από κωδικούς και χαρακτήρες, μετρήσεις και ταξινόμηση δειγμάτων βιολογικών κυττάρων, απεικονίσεις διεργασιών ή αποτελεσμάτων, αποθηκευμένες εικόνες, συναγερμούς από αυτοματοποιημένα συστήματα παρακολούθησης χώρου MV και σήματα ελέγχου διεργασίας. Περιλαμβάνονται επίσης διεπαφές χρήστη, διεπαφές για την ενσωμάτωση συστημάτων πολλαπλών στοιχείων και αυτοματοποιημένη ανταλλαγή δεδομένων.

1.4. Υλοποίηση

Όλες οι παραπάνω κατηγορίες, μεμονωμένες ή σε συνδυασμό, βρίσκουν πολλές εφαρμογές στην καθημερινή ζωή. Πεδία που εφαρμόζεται η μηχανική όραση είναι

- Αυτοκινητοβιομηχανία
- Βιομηχανία ημιαγωγών
- Βιομηχανία ηλεκτρονικών και υπολογιστών
- Συσκευασία τροφίμων
- Καταναλωτικά προϊόντα
- Ιατρική και φαρμακευτικά

- Μεταφορές
- Γραφικές τέχνες και συσκευασία
- Καλλιέργειες
- Αλιεία
- Ασφάλεια

Δεν πρέπει να παραλειφθεί εδώ, η προσπάθεια που γίνεται τα τελευταία χρόνια από την εταιρία Google και κάποιες αυτοκινητοβιομηχανίες, για την υλοποίηση της αυτόνομης οδήγησης οχήματος. Είναι μια αξιόλογη προσπάθεια, αν λάβει κανείς υπόψη ότι γίνεται η ταυτόχρονη χρήση όλων των κατηγοριών μηχανικής όρασης.

1.5. Αγορά

Η παγκόσμια αγορά Machine Vision αναμένεται να φθάσει τα 15,46 δισ. Δολάρια μέχρι το τέλος του 2022 με 8,18% μέσο ετήσιο ρυθμό ανάπτυξης (CAGR) κατά την περίοδο πρόβλεψης 2017-2022. Οι σημαντικότεροι παράγοντες στην αγορά Machine Vision είναι οι εταιρείες Cognex Corporation, Vitronic GmbH, Perceptron, Inc., Microscan Systems, US, Basler AG, National Instruments, , Optotune AG (Ελβετία), USS Vision (ΗΠΑ), ViDi Systems SA (Ελβετία), Keyence (Ιαπωνία), Omron Corporation (Ιαπωνία) μεταξύ άλλων. Ο τομέας της μηχανικής όρασης αναπτύσσεται με γρήγορους ρυθμούς σε όλες τις περιοχές. Η αύξηση των τομέων εφαρμογής από έτος σε έτος και η πρόοδος στην τεχνολογία, οδηγεί την αγορά της μηχανικής όρασης σε παγκόσμια κλίμακα. Η Ασία κυριαρχεί στην παγκόσμια αγορά με περισσότερο από το 30% του μεριδίου αγοράς ακολουθούμενη από την Ευρώπη, η οποία αποτελεί τη δεύτερη μεγαλύτερη αγορά λόγω της μεγάλης ζήτησης της αυτοκινητοβιομηχανίας και της βιομηχανίας υγειονομικής περίθαλψης. Η Βόρεια Αμερική είναι η τρίτη μεγαλύτερη αγορά. [1]

2. Επισκόπηση Συσκευών

2.1.Raspberry Pi

Το Raspberry Pi [2] είναι μια σειρά μικρών υπολογιστών μιας πλακέτας, που αναπτύχθηκαν στο Ηνωμένο Βασίλειο από το ίδρυμα Raspberry Pi, για να προωθήσουν τη διδασκαλία πληροφορικής στα σχολεία και στις αναπτυσσόμενες χώρες. Το αρχικό μοντέλο έγινε πολύ πιο δημοφιλές από ότι ήταν αναμενόμενο, και έκανε πωλήσεις εκτός της στοχευόμενης αγοράς, για χρήσεις όπως η ρομποτική. Περιφερειακά (συμπεριλαμβανομένων των πληκτρολογίων, και των ποντικιών) δεν περιλαμβάνονται στο Raspberry Pi.

Σύμφωνα με το Ίδρυμα Raspberry Pi, περισσότερα από 5 εκατομμύρια Raspberry Pis έχουν πωληθεί πριν από το Φεβρουάριο του 2015, καθιστώντας το, τον βρετανικό υπολογιστή με τις περισσότερες πωλήσεις. Μέχρι το Νοέμβριο του 2016 είχαν πωληθεί 11 εκατομμύρια μονάδες.

2.1.1. Επισκόπηση

Εικόνα 1. Το Raspberry Pi Zero, το μοντέλο των $5 US που κυκλοφόρησε το 2015

Μέχρι σήμερα έχουν κυκλοφορήσει αρκετές γενιές Raspberry Pi. Η πρώτη γενιά (Raspberry Pi 1 Model B) κυκλοφόρησε τον Φεβρουάριο του 2012. Ακολούθησε το απλούστερο και φθηνότερο Model A. Το 2014, το Ίδρυμα κυκλοφόρησε ένα νέο μοντέλο με βελτιωμένο σχεδιασμό, το Raspberry Pi 1 Model B +. Το μέγεθος του Raspberry Pi είναι περίπου αυτό μιας πιστωτικής κάρτας. Τα βελτιωμένα μοντέλα A + και B + κυκλοφόρησαν ένα χρόνο αργότερα. Μια έκδοση "Compute Module" κυκλοφόρησε τον Απρίλιο του 2014 που προορίζονταν για ενσωματωμένες εφαρμογές. Το Raspberry Pi 2 στο οποίο προστέθηκε περισσότερη μνήμη RAM κυκλοφόρησε τον Φεβρουάριο του 2015.

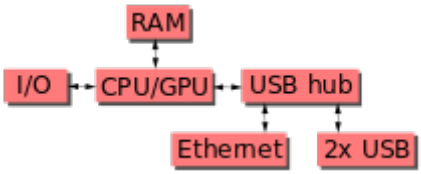
Το Raspberry Pi Zero με μικρότερο μέγεθος αλλά και με λιγότερες δυνατότητες διασύνδεσης, και λιγότερες εισόδους / εξόδους γενικής χρήσης (GPIO), κυκλοφόρησε το Νοέμβριο του 2015 στην τιμή των $ 5 US. Το Raspberry Pi 3 Model B κυκλοφόρησε τον Φεβρουάριο του 2016 και συνοδεύεται από ενσωματωμένες δυνατότητες WiFi, Bluetooth και USB boot. Από τον Ιανουάριο του 2017, το Raspberry Pi 3 Model B είναι το νεότερο μοντέλο Raspberry Pi στην παραγωγή. Οι τιμές των Raspberry Pis κυμαίνονται μεταξύ $ 5-35 US. Στις 28 Φεβρουαρίου 2017 κυκλοφόρησε το Raspberry Pi Zero W, το οποίο είναι πανομοιότυπο με το Raspberry Pi Zero, αλλά έχει ενσωματωμένο Wi-Fi και Bluetooth και κοστίζει $ 10.

Όλα τα μοντέλα διαθέτουν ένα Broadcom ‘System on Chip’ (SoC), το οποίο περιλαμβάνει κεντρική μονάδα επεξεργασίας (CPU) συμβατή με αρχιτεκτονική ARM και μονάδα επεξεργασίας γραφικών (GPU, VideoCore IV). Η ταχύτητα της CPU κυμαίνεται από 700 MHz έως 1,2 GHz και η ενσωματωμένη μνήμη από 256 MB έως 1 GB RAM. Η αποθήκευση του λειτουργικού συστήματος και των υπολοίπων αρχείων, γίνεται σε κάρτες Secure Digital (SD) σε μεγέθη SDHC ή MicroSDHC. Όλα τα μοντέλα διαθέτουν από μια έως τέσσερις υποδοχές USB, έξοδο HDMI και βίντεο και μια υποδοχή 3,5 mm για ήχο. Έξοδοι κατώτερου επιπέδου παρέχονται από έναν αριθμό ακίδων GPIO που υποστηρίζουν κοινά πρωτόκολλα όπως το I²C. Τα μοντέλα B διαθέτουν θύρα Ethernet 8P8C και τα Pi 3 και Pi Zero W διαθέτουν Wi-Fi 802.11n και Bluetooth.

Το Ίδρυμα παρέχει για λήψη το λειτουργικό Raspbian, μια διανομή Linux που βασίζεται στο Debian, καθώς επίσης και διανομές από τρίτους κατασκευαστές όπως το Ubuntu, Windows 10 IOT Core, RISC OS και εξειδικευμένα κέντρα πολυμέσων. Προωθεί την Python και το Scratch ως την κύρια γλώσσα προγραμματισμού, με υποστήριξη όμως και για πολλές άλλες γλώσσες.

2.1.2. Hardware

Το hardware του Raspberry Pi έχει εξελιχθεί στις διάφορες εκδόσεις του, και χαρακτηρίζεται από παραλλαγές στη χωρητικότητα της μνήμης και την υποστήριξη περιφερειακών συσκευών.



Σχήμα 3. Διάγραμμα εσωτερικής αρχιτεκτονικής του Raspberry Pi

Το μπλοκ διάγραμμα απεικονίζει τα μοντέλα A, B, A+ και B+. Τα μοντέλα A, A+ και Pi Zero δεν διαθέτουν διανομέα Ethernet και USB. Ο προσαρμογέας Ethernet είναι εσωτερικά συνδεδεμένος σε μια πρόσθετη θύρα USB. Στα μοντέλα A, A+ και Pi Zero, η θύρα USB συνδέεται απευθείας στο SoC. Στο μοντέλο Pi 1 Model B+ και μεταγενέστερα, το τσιπ USB / Ethernet περιέχει έναν διανομέα USB πέντε σημείων, από τον οποίο διατίθενται τέσσερις θύρες, ενώ το Pi 1 Model B παρέχει μόνο δύο. Στο Pi Zero, η θύρα USB συνδέεται επίσης απευθείας στο SoC, αλλά χρησιμοποιεί μια θύρα micro USB (OTG).

2.1.3. **Επεξεργαστής**

Εικόνα 2. Το Raspberry Pi 2 με επεξεργαστή 32-bit 900 MHz τεσσάρων πυρήνων ARM Cortex-A7.

Το τσιπ Broadcom BCM2835 SoC που χρησιμοποιήθηκε στην πρώτη γενιά Raspberry Pi, είναι περίπου ισοδύναμο με το τσιπ που χρησιμοποιήθηκε στα πρώτα smartphones (ο επεξεργαστής του είναι μια παλαιότερη αρχιτεκτονική ARMv6). Αυτό περιλαμβάνει επεξεργαστή ARM1176JZF-S 700 MHz, μονάδα επεξεργασίας γραφικών VideoCore IV και RAM. Έχει μια προσωρινή μνήμη επιπέδου 1 (L1) 16 KB και μια προσωρινή μνήμη επιπέδου 2 (L2) 128 KB. Η μνήμη επιπέδου 2 χρησιμοποιείται κυρίως από τη GPU.

Το νεότερο μοντέλο Raspberry Pi 2 χρησιμοποιεί ένα Broadcom BCM2836 SoC με επεξεργαστή ARM Cortex-A7 τετραπλού πυρήνα 900 MHz, με κοινή μνήμη L2 256 KB. Το Raspberry Pi 2 V1.2 αναβαθμίστηκε σε Broadcom BCM2837 SoC με επεξεργαστή ARM Cortex-A53 64-bit τετραπλού πυρήνα, και είναι το ίδιο SoC που χρησιμοποιείται στο Raspberry Pi 3.

Το Raspberry Pi 3 χρησιμοποιεί ένα Broadcom BCM2837 SoC με επεξεργαστή ARM Cortex-A53 τετραπλού πυρήνα 1,2 GHz 64bit με 512 KB κοινόχρηστη μνήμη L2.

Το Raspberry Pi έχει κερδίσει αρκετά βραβεία έως σήμερα. Τον Ιούνιο του 2017 η ομάδα του Raspberry Pi, κέρδισε το κορυφαίο βραβείο μηχανικής καινοτομίας του Ηνωμένου Βασιλείου, το βραβείο Royal Academy of Engineering MacRobert. Η παραπομπή για το βραβείο στο Raspberry Pi δήλωνε ότι "για τους φτηνούς της μικροϋπολογιστές μεγέθους πιστωτικής κάρτας, που επαναπροσδιορίζουν τον τρόπο με τον οποίο οι άνθρωποι ασχολούνται με την πληροφορική, εμπνέοντας τους μαθητές να μάθουν προγραμματισμό και την επιστήμη των υπολογιστών και να παρέχουν καινοτόμες λύσεις ελέγχου στη βιομηχανία".

2.1.4. Χρήση στην εκπαίδευση

Από τον Ιανουάριο του 2012,στο Ηνωμένο Βασίλειο, έχουν ζητηθεί πληροφορίες για τη συσκευή από δημόσια και ιδιωτικά σχολεία, με περίπου 5 φορές μεγαλύτερο ενδιαφέρον στα ιδιωτικά. Ελπίζεται ότι οι επιχειρήσεις θα χρηματοδοτήσουν αγορές για λιγότερο ευνοημένα σχολεία. Ο Διευθύνων Σύμβουλος της Premier Farnell δήλωσε ότι η κυβέρνηση μιας χώρας στη Μέση Ανατολή έχει εκφράσει το ενδιαφέρον της για την παροχή ενός Raspberry Pi σε κάθε μαθήτρια, προκειμένου να βελτιώσει τις προοπτικές απασχόλησής της.

Το 2014, το Ίδρυμα Raspberry Pi προσέλαβε διάφορα μέλη της κοινότητάς του, συμπεριλαμβανομένων και πρώην εκπαιδευτικών και προγραμματιστών λογισμικού, για να ξεκινήσει μια σειρά δωρεάν πόρων μάθησης για την ιστοσελίδα του. Οι πόροι διατίθενται δωρεάν υπό την Creative Commons και οι συνεισφορές και οι συνεργασίες διατίθενται στην πλατφόρμα GitHub.

Το Ίδρυμα ξεκίνησε επίσης και μαθήματα επιμόρφωσης εκπαιδευτικών με το όνομα, Picademy με σκοπό να βοηθήσει τους εκπαιδευτικούς να προετοιμαστούν για τη διδασκαλία του νέου αναλυτικού προγράμματος σπουδών χρησιμοποιώντας το Raspberry Pi στην τάξη. Η συνεχής πορεία επαγγελματικής ανάπτυξης παρέχεται δωρεάν για εκπαιδευτικούς και διοικείται από την εκπαιδευτική ομάδα του Ιδρύματος.

Τον Ιανουάριο του 2017, ένα δωρεάν μάθημα MOOC ξεκίνησε στο Kadenze σε συνεργασία με το Πανεπιστήμιο της Νέας Νότιας Ουαλίας στο Σύδνεϋ της Αυστραλίας στο πλαίσιο του Internet of Things, χρησιμοποιώντας το Raspberry Pi Online Rpi Course. Υπάρχουν επίσης καλοί διαδικτυακοί πόροι για το Raspberry Pi στον ιστότοπο Sparkfun καθώς και στην κοινότητα του οργανισμού Raspberry Pi.

2.1.5. Χρήση στον οικιακό αυτοματισμό

Υπάρχουν αρκετοί προγραμματιστές και εφαρμογές που αξιοποιούν το Raspberry Pi για τον οικιακό αυτοματισμό. Αυτοί οι προγραμματιστές καταβάλλουν προσπάθεια να καταστήσουν το Raspberry Pi ως μια οικονομικά προσιτή λύση στην παρακολούθηση και την κατανάλωση ενέργειας. Λόγω του σχετικά χαμηλού κόστους, το Raspberry Pi έχει γίνει μια δημοφιλής και οικονομική λύση και για τις ακριβότερες εμπορικές εναλλακτικές.

2.1.6. Χρήση στον βιομηχανικό αυτοματισμό

Τον Ιούνιο του 2014, η κατασκευάστρια εταιρία βιομηχανικών αυτοματισμών TECHBASE σχεδίασε τον πρώτο βιομηχανικό υπολογιστή παγκοσμίως με βάση το Compute module Raspberry Pi, που ονομάζεται ModBerry. Η συσκευή διαθέτει πολλές διασυνδέσεις, κυρίως σειριακές θύρες RS-485/232, ψηφιακές και αναλογικές εισόδους / εξόδους, CAN και οικονομικούς ζυγούς 1-Wire, τα οποία χρησιμοποιούνται ευρέως στη βιομηχανία αυτοματισμού. Ο κατασκευή επιτρέπει τη χρήση του Compute Module σε σκληρά βιομηχανικά περιβάλλοντα, οδηγώντας στο

συμπέρασμα ότι το Raspberry Pi δεν περιορίζεται πλέον σε οικιακούς αυτοματισμούς και μαθητικές εργασίες, αλλά μπορεί να χρησιμοποιηθεί ευρέως ως λύση Βιομηχανικής Διασύνδεσης και να επιτύχει τους στόχους του Industry 4.0.

2.1.7. Χρήση σε εμπορικά προϊόντα

Το OTTO είναι μια ψηφιακή φωτογραφική μηχανή που δημιουργήθηκε από την Next Thing Co. Περιλαμβάνει ένα Raspberry Pi Compute Module. Χρηματοδοτήθηκε επιτυχώς από το πλήθος στην εκστρατεία Kickstarter τον Μάιο του 2014.

Το Slice είναι ένα ψηφιακό πρόγραμμα αναπαραγωγής πολυμέσων το οποίο χρησιμοποιεί επίσης το Compute Module. Χρηματοδοτήθηκε από πολλούς σε μια εκστρατεία Kickstarter τον Αύγουστο του 2014. Το λογισμικό που τρέχει στο Slice βασίζεται στο Kodi.

Τα εκπαιδευτικά κιτ ηλεκτρονικών υπολογιστών που κατασκευάζονται από την εταιρεία υπολογιστών Kano που εδρεύει στο Λονδίνο, έχουν ως βάση το Raspberry Pi. Ένας από τους στόχους της εταιρίας ήταν να καταστήσει το Raspberry Pi πιο προσιτό στο ευρύτερο κοινό.

2.1.8. Η ιστορία

Εικόνα 3. Πρώιμη πειραματική πλακέτα του Raspberry Pi

Το 2006, η αρχική ιδέα του Raspberry Pi βασίστηκε στον μικροελεγκτή Atmel ATmega644. Η διοίκηση του ιδρύματος Eben Upton συγκέντρωσε μια ομάδα καθηγητών, ακαδημαϊκών και ενθουσιωδών των υπολογιστών για να δημιουργήσουν έναν υπολογιστή που θα εμπνεύσει τα παιδιά. Ο υπολογιστής εμπνέεται από το BBC Micro του Acorn του 1981. Τα ονόματα Model A, Model B και Model B+ είναι

αναφορές στα αρχικά μοντέλα του βρετανικού εκπαιδευτικού υπολογιστή BBC Micro, που αναπτύχθηκε από την Acorn Computers. Η πρώτη ARM έκδοση του υπολογιστή τοποθετήθηκε σε ένα πακέτο με μέγεθος όσο ένα USB memory stick. Είχε μια θύρα USB στο ένα άκρο και μια θύρα HDMI από την άλλη.

Ο στόχος του ιδρύματος ήταν να προσφέρει δύο εκδόσεις αξίας $25 και $35 US. Ξεκίνησαν να δέχονται παραγγελίες για το μοντέλο B υψηλότερης τιμής στις 29 Φεβρουαρίου 2012, το μοντέλο A χαμηλότερου κόστους στις 4 Φεβρουαρίου 2013 και το A+ με ακόμη χαμηλότερο κόστος (US $ 20), στις 10 Νοεμβρίου 2014. Στις 26 Νοεμβρίου 2015, το φθηνότερο Raspberry Pi , το Raspberry Pi Zero, κυκλοφόρησε στα $ 5.

2.2. OpenCV

Η OpenCV [3] (Open Source Computer Vision) είναι μια βιβλιοθήκη συναρτήσεων προγραμματισμού που στοχεύει κυρίως στην ‘όραση υπολογιστή’ σε πραγματικό χρόνο. Αρχικά αναπτύχθηκε από την Intel, υποστηρίχθηκε αργότερα από το Willow Garage και τώρα διατηρείται από την Itseez. Η βιβλιοθήκη υποστηρίζει πολλαπλές πλατφόρμες και είναι δωρεάν για χρήση υπό την άδεια BSD ανοιχτού κώδικα. Η OpenCV υποστηρίζει τα πλαίσια Deep Learning, TensorFlow, Torch / PyTorch και Caffe.

2.2.1. Ιστορία

Το πρόγραμμα OpenCV, το οποίο ξεκίνησε επίσημα το 1999, ήταν αρχικά μια πρωτοβουλία της Intel Research για τις εφαρμογές έντονης επεξεργαστικής ισχύος, μέρος μιας σειράς έργων, όπως η ανίχνευση ακτίνων σε πραγματικό χρόνο και οι τρισδιάστατες οθόνες. Οι κύριοι συντελεστές του έργου συμπεριέλαβαν έναν αριθμό ειδικών βελτιστοποίησης στην Intel Russia, καθώς και την ομάδα Performance

Library Team της Intel. Κατά το ξεκίνημα της OpenCV, οι στόχοι του έργου περιγράφηκαν ως εξής:

• Να προωθήσει την έρευνα στον τομέα της ‘όρασης υπολογιστή’ παρέχοντας όχι μόνο ανοιχτό αλλά και βελτιστοποιημένο κώδικα για βασική υποδομή όρασης. Να μην υπάρχει πλέον η επανεφεύρεση του τροχού.

• Διάδοση της γνώσης της όρασης παρέχοντας μια κοινή υποδομή πάνω στην οποία θα μπορούσαν να βασιστούν οι προγραμματιστές, ώστε ο κώδικας να είναι πιο εύκολα αναγνώσιμος και μεταβιβάσιμος.

• Προηγμένες εμπορικές εφαρμογές βασισμένες σε όραση, καθιστώντας διαθέσιμο δωρεάν κώδικα βελτιστοποιημένης απόδοσης – και χωρίς να απαιτείται κάποια άδεια.

Η πρώτη alpha έκδοση της OpenCV κυκλοφόρησε στο κοινό, στη διάσκεψη IEEE για το Computer Vision and Pattern Recognition το 2000 και πέντε beta εκδόσεις κυκλοφόρησαν μεταξύ του 2001 και του 2005. Η πρώτη 1.0 έκδοση κυκλοφόρησε το 2006, ενώ έκδοση 1.1 κυκλοφόρησε τον Οκτώβριο του 2008.

Η δεύτερη μεγάλη έκδοση της OpenCV ήταν τον Οκτώβριο του 2009. Η OpenCV 2 περιλαμβάνει σημαντικές αλλαγές στη διασύνδεση C++, στοχεύοντας σε ευκολότερα, πιο ασφαλή πρότυπα τύπων, νέες συναρτήσεις και καλύτερες υλοποιήσεις για τις υπάρχουσες από την άποψη επιδόσεων (ειδικότερα σε συστήματα πολλαπλού πυρήνα). Οι επίσημες κυκλοφορίες γίνονται τώρα κάθε έξι μήνες και η ανάπτυξη γίνεται από μια ανεξάρτητη ρωσική ομάδα που υποστηρίζεται από εμπορικές εταιρίες.

Τον Αύγουστο του 2012, η υποστήριξη για την OpenCV αναλήφθηκε από το μη κερδοσκοπικό ίδρυμα *OpenCV.org*, το οποίο διατηρεί έναν ιστότοπο για προγραμματιστές και χρήστες.

2.2.2. Γλώσσα προγραμματισμού

Η OpenCV είναι γραμμένη σε C ++ και η κύρια διεπαφή της είναι στην C ++. Διατηρεί όμως ακόμα μια λιγότερο κατανοητή, αλλά περισσότερο εκτεταμένη, παλαιότερη διασύνδεση με την C. Υπάρχουν διεπαφές με Python, Java και

MATLAB / OCTAVE. Οδηγίες και βοηθήματα για τις γλώσσες αυτές υπάρχουν διαθέσιμες στο διαδίκτυο. Διασυνδέσεις και με άλλες γλώσσες όπως οι C #, Perl, Ch, Haskell και Ruby έχουν αναπτυχθεί για να ενθαρρύνουν την υιοθεσία από ένα ευρύτερο κοινό. Όλες οι νέες βελτιώσεις και αλγόριθμοι στην OpenCV αναπτύσσονται τώρα σε C ++.

2.2.3. Επιτάχυνση υλικού

Εάν η βιβλιοθήκη βρει τα ολοκληρωμένα πρωτότυπα επιδόσεων της Intel στο σύστημα, θα χρησιμοποιήσει αυτές τις βελτιστοποιημένες ρουτίνες για να επιταχυνθεί. Μια διεπαφή GPU βασισμένη σε CUDA βρίσκεται σε εξέλιξη από τον Σεπτέμβριο του 2010. Μια διεπαφή GPU βασισμένη σε OpenCL βρίσκεται σε εξέλιξη από τον Οκτώβριο του 2012. Η τεκμηρίωση για την έκδοση 2.4.13.3 βρίσκεται στο docs.opencv.org.

2.2.4. Υποστήριξη OS

Η OpenCV λειτουργεί σε διάφορες πλατφόρμες.

- Υπολογιστές: Windows, Linux, macOS, FreeBSD, NetBSD, OpenBSD.
- Κινητά: Android, iOS, Maemo, BlackBerry 10.

Ο χρήστης μπορεί να πάρει επίσημες κυκλοφορίες από το SourceForge ή να λάβει τις τελευταίες πηγές από το GitHub. Η OpenCV χρησιμοποιεί την εντολή CMake.

2.3. Arduino

Το **Arduino** [4] είναι μια πλακέτα η οποία περιέχει ένα μικροελεγκτή που διαθέτει εισόδους και εξόδους. Το σχέδιο της πλακέτας είναι ανοιχτού κώδικα και διατίθεται δωρεάν. Ο μικροελεγκτής μπορεί να προγραμματιστεί με τη γλώσσα

Wiring που είναι πρακτικά η γλώσσα προγραμματισμού C++ και ένα σύνολο από βιβλιοθήκες, υλοποιημένες επίσης στην C++. Το Arduino μπορεί να χρησιμοποιηθεί για την ανάπτυξη ανεξάρτητων διαδραστικών εφαρμογών αλλά και να συνδεθεί με υπολογιστή μέσω προγραμμάτων όπως Processing, Max/MSP, Pure Data, SuperCollider. Οι περισσότερες εκδόσεις του Arduino διατίθενται προ-συναρμολογημένες. Το διάγραμμα επίσης και τα υλικά κατασκευής είναι ελεύθερα διαθέσιμα στο διαδίκτυο.Το Arduino έλαβε τιμητική μνεία στην κατηγορία *Digital Communities* στο *Prix Ars Electronica* το 2006.

2.3.1. Ιστορία

Το 2005 ξεκίνησε να σχεδιάζεται μια συσκευή που θα βοηθούσε στις διαδραστικές εφαρμογές μαθητών. Η συσκευή θα έπρεπε να ήταν πιο φθηνή από άλλα πρωτότυπα συστήματα διαθέσιμα εκείνη την περίοδο. Οι ιδρυτές Massimo Banzi και David Cueartielles ονόμασαν το σχέδιο από τον Arduin της Ivrea και ξεκίνησαν να παράγουν πλακέτες σε ένα μικρό εργοστάσιο στην Ιβρέα, μια κωμόπολη της επαρχίας Τορίνο της βορειοδυτικής Ιταλίας. Το σχέδιο Arduino είναι μία διακλάδωση της πλατφόρμας Wiring για λογισμικό ανοικτού κώδικα και προγραμματίζεται χρησιμοποιώντας μια γλώσσα βασισμένη στο Wiring (σύνταξη και βιβλιοθήκες), παρόμοια με την C++ με απλοποιήσεις και αλλαγές, καθώς και ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE).

2.3.2. Υλικό (Hardware)

Μία πλακέτα Arduino αποτελείται από ένα μικροελεγκτή Atmel AVR (ATmega328 και ATmega168 στις νεότερες εκδόσεις, ATmega8 στις παλαιότερες) και συμπληρωματικά εξαρτήματα για την διευκόλυνση του χρήστη στον προγραμματισμό και την ενσωμάτωσή του σε άλλα κυκλώματα. Όλες οι πλακέτες περιλαμβάνουν ένα γραμμικό ρυθμιστή τάσης 5V και έναν κρυσταλλικό ταλαντωτή 16MHz. Ο μικροελεγκτής είναι από κατασκευής προγραμματισμένος με ένα *bootloader*, έτσι ώστε να μην χρειάζεται εξωτερικός προγραμματιστής.

Όλες οι πλακέτες Arduino προγραμματίζονται με μία RS-232 σειριακή σύνδεση, αλλά ο τρόπος που επιτυγχάνεται αυτό διαφέρει σε κάθε hardware εκδοχή. Οι σειριακές πλάκες Arduino περιέχουν ένα απλό level shifter κύκλωμα για να μετατρέπει τα επίπεδα RS-232 και TTL σημάτων. Ο προγραμματισμός των Arduino

τώρα γίνεται μέσω USB. Η πλακέτα Arduino εκθέτει τα περισσότερα microcontroller I/O pins για χρήση από άλλα κυκλώματα. Τα Diecimila, Duemilanove και το τρέχον Uno για παράδειγμα, παρέχουν 14 ψηφιακά I/O pins, έξι από τα οποία μπορούν να παράγουν pulse-width διαμορφωμένα σήματα, και έξι αναλογικά δεδομένα.

2.3.3. Λογισμικό

Εικόνα 4. Στιγμιότυπο του λογισμικού του Arduino.

Στιγμιότυπο του λογισμικού του Arduino.

Το ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) του Arduino είναι μία εφαρμογή γραμμένη σε Java, που λειτουργεί σε πολλές πλατφόρμες και προέρχεται από το IDE για τη γλώσσα προγραμματισμού Processing και το σχέδιο Wiring. Έχει σχεδιαστεί για να εισαγάγει τον προγραμματισμό στους καλλιτέχνες και τους νέους που δεν είναι εξοικειωμένοι με την ανάπτυξη λογισμικού. Περιλαμβάνει ένα πρόγραμμα επεξεργασίας κώδικα με χαρακτηριστικά όπως είναι η επισήμανση σύνταξης και ο συνδυασμός αγκύλων και είναι επίσης σε θέση να μεταγλωττίζει και να φορτώνει προγράμματα στην πλακέτα με ένα μόνο κλικ. Τα Arduino προγράμματα είναι γραμμένα σε C ή C++. Το Arduino IDE έρχεται με μια βιβλιοθήκη λογισμικού που ονομάζεται "Wiring", από το πρωτότυπο σχέδιο Wiring, γεγονός που καθιστά πολλές κοινές λειτουργίες εισόδου/εξόδου πολύ πιο εύκολες. Οι χρήστες πρέπει μόνο να ορίσουν δύο λειτουργίες για να κάνουν ένα πρόγραμμα κυκλικής εκτέλεσης:

-setup():μία συνάρτηση που τρέχει μία φορά στην αρχή του προγράμματος η οποία αρχικοποιεί τις ρυθμίσεις.

-loop():μία συνάρτηση που καλείται συνέχεια μέχρι η πλακέτα να απενεργοποιηθεί.

Το IDE του Arduino χρησιμοποιεί το GNU toolchain και το AVR Libc για να μεταγλωττίζει προγράμματα και το avrdude για να φορτώνει προγράμματα στην πλακέτα.

Δεδομένου ότι η πλατφόρμα Arduino χρησιμοποιεί Atmel μικροελεγκτές, το περιβάλλον ανάπτυξης της Atmel, το AVR Studio ή το νεότερη έκδοση του Atmel Studio, μπορεί επίσης να χρησιμοποιηθεί για την ανάπτυξη λογισμικού για το Arduino.

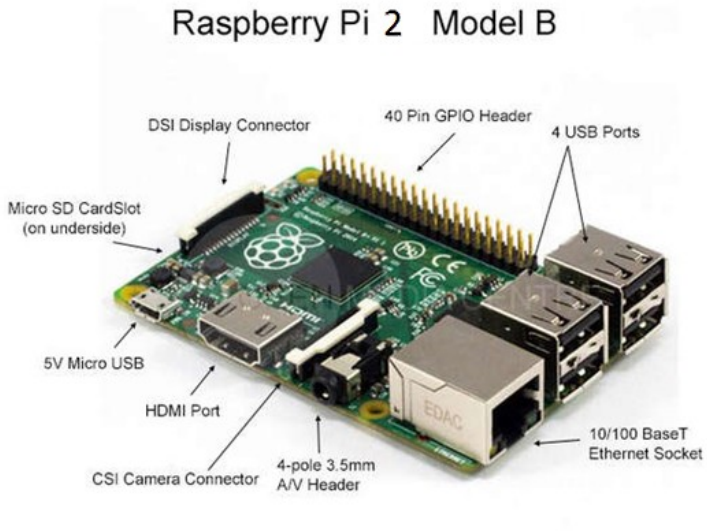
1. Hardware υλικό που χρησιμοποιήθηκε στην εργασία

1.1. Raspberry Pi 2 B+

Εικόνα 5. Το Raspberry Pi 2 B+

Τεχνικά χαρακτηριστικά του Raspberry Pi [5]

Chip	Broadcom BCM2836 SoC
Core architecture	Quad-core ARM Cortex-A7
CPU	900 MHz
GPU	Dual Core VideoCore IV® Multimedia Co-Processor Provides Open GL ES 2.0, hardware-accelerated OpenVG, and 1080p30 H.264 high-profile decode Capable of 1Gpixel/s, 1.5Gtexel/s or 24GFLOPs with texture filtering and DMA infrastructure
Memory	1GB LPDDR2
Operating System	Boots from Micro SD card, running a version of the Linux operating system
Dimensions	85 x 56 x 17mm
Power	Micro USB socket 5V, 2A
Connectors:	
Ethernet	10/100 BaseT Ethernet socket
Video Output	HDMI (rev 1.3 & 1.4)
Audio Output	3.5mm jack, HDMI
USB	4 x USB 2.0 Connector
GPIO Connector	40-pin 2.54 mm (100 mil) expansion header: 2x20 strip Providing 27 GPIO pins as well as +3.3 V, +5 V and GND supply lines
Camera Connector	15-pin MIPI Camera Serial Interface (CSI-2)
Display Connector	Display Serial Interface (DSI) 15 way flat flex cable connector with two data lanes and a clock lane
Memory Card Slot	Micro SDIO



Εικόνα 6. Διάταξη των συσκευών

Το Raspberry Pi που χρησιμοποιήθηκε στην εργασία, είναι η προτελευταία έκδοση της εταιρίας. Η τελευταία έκδοση, που είναι το Raspberry Pi 3 B υπερτερεί αυτής της έκδοσης σε αρκετά σημεία. Παρακάτω φαίνεται ένα συγκριτικό ως προς τα τεχνικά χαρακτηριστικά τριών εκδόσεων του Raspberry Pi.

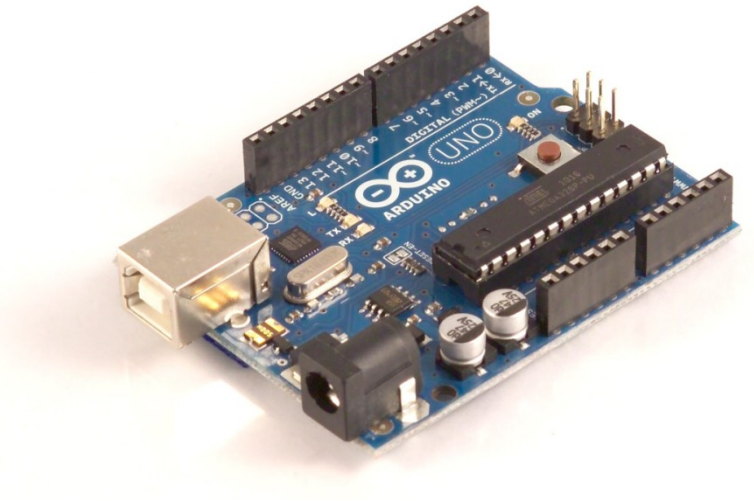
Πίνακας 1. Συγκριτικός πίνακας τριών μοντέλων Raspberry Pi

	Raspberry Pi 3 Model B	**Raspberry Pi 2 Model B**	**Raspberry Pi Model B+**
Processor Chipset	**Broadcom BCM2837 64Bit Quad Core ARM Cortex A53 at 1.2GHz**	Broadcom BCM2836 32Bit Quad Core ARMv7 at 900MHz	Broadcom BCM2835 32Bit ARMv6k at 700MHz
GPU	**Videocore IV @ 400MHz**	Videocore IV @ 250MHz	Videocore IV @ 250MHz
Processor Speed	**QUAD Core @1.2 GHz**	QUAD Core @900 MHz	Single Core @700 MHz
RAM	1GB SDRAM @ 400 MHz	1GB SDRAM @ 400 MHz	512 MB SDRAM @ 400 MHz
Storage	MicroSD	MicroSD	MicroSD
USB 2.0	4x USB Ports	4x USB Ports	4x USB Ports
Max Power Draw/voltage	2.5A @ 5V	1.8A @ 5V	1.8A @ 5V
GPIO	40 pin	40 pin	40 pin
Ethernet Port	Yes	Yes	Yes
WiFi	**Built in**	No	No
Bluetooth LE	**Built in**	No	No
Video Output	HDMI/Composite via RCA Jack	HDMI/Composite via RCA Jack	HDMI/Composite via RCA Jack
Audio Output	3.5mm Jack	3.5mm Jack	3.5mm Jack

Από τον πίνακα γίνεται φανερό ότι η τελευταία έκδοση διαθέτει 30% μεγαλύτερη επεξεργαστική ισχύ, καθώς επίσης και ενσωματωμένο Bluetooth. Στην περίπτωση που χρησιμοποιηθεί το Raspberry Pi 3 για την εφαρμογή, θα υπάρχει

ταχύτερη επεξεργασία δεδομένων τόσο από την πλευρά του κεντρικού επεξεργαστή, όσο και από τον επεξεργαστή των γραφικών. Επίσης δεν είναι απαραίτητη η χρήση του USB Bluetooth Dongle, αφού υπάρχει διαθέσιμο ενσωματωμένο Bluetooth.

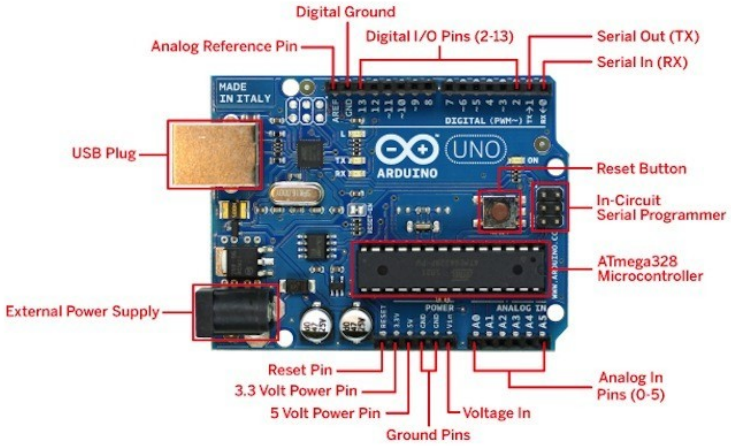
1.2. Arduino Uno



Εικόνα 7. Το Arduino UNO [6]

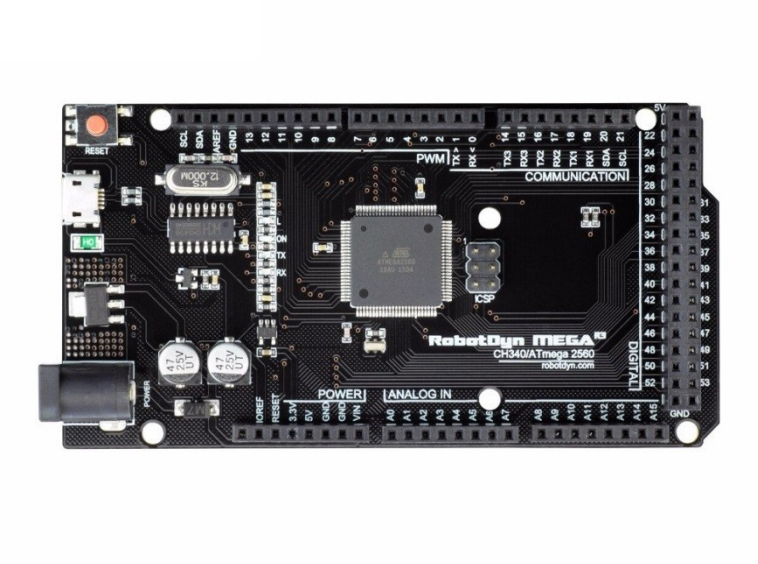
Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13

Length	68.6 mm
Width	53.4 mm
Weight	25 g



Εικόνα 8. Διάταξη συσκευών του Arduino

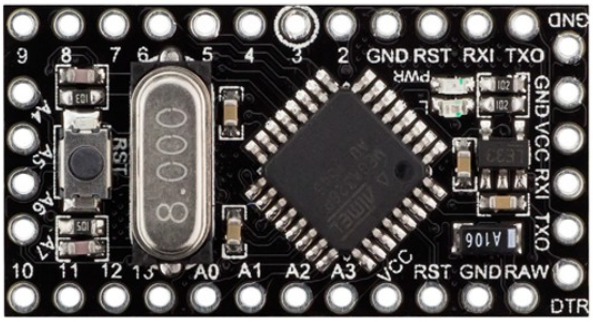
1.3. Arduino Mega



Εικόνα 9. Το Arduino Mega

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
LED_BUILTIN	13
Length	101.52 mm
Width	53.3 mm
Weight	37 g

1.4. Arduino Pro Mini



Εικόνα 10. Το Arduino Pro Mini

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage	7-9 V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	8 (of which 4 are broken out onto pins)

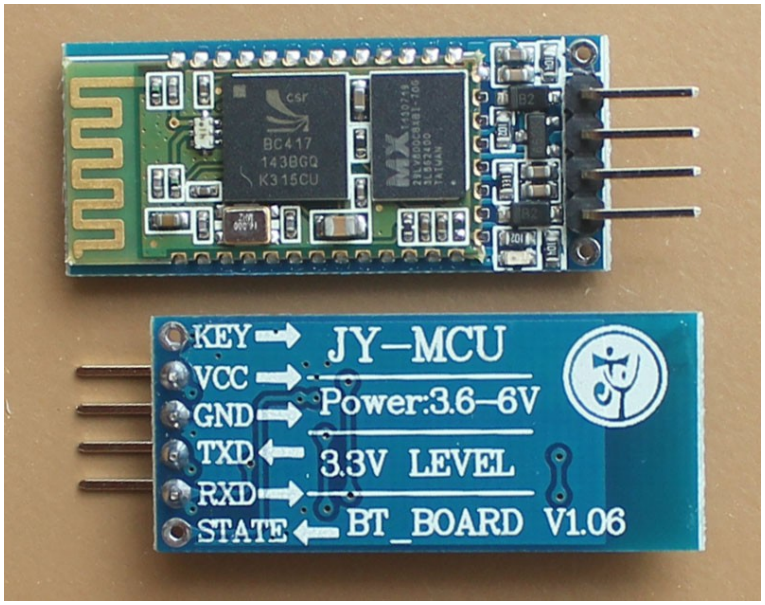
DC Current per I/O Pin	40 mA
Flash Memory	32 KB (of which 2 KB used by bootloader)
SRAM	2 KB
EEPROM	1 KB
Clock Speed	16 MHz
Length	30 mm
Width	18 mm

1.5. Camera

Εικόνα 11. Η camera του Raspberry Pi

- Raspberry Pi Night Vision Camera, supports all revisions of the Pi
- 5 megapixel OV5647 sensor
- Camera specifications
    - CCD size : 1/4inch
    - Aperture (F) : 1.8
    - Focal Length : 3.6MM (adjustable)
    - Diagonal : 75.7 degree
    - Sensor best resolution : 1080p
- 4 screw holes
    - Used for attachment
    - Provides 3.3V power output
    - Supports connecting infrared LED and/or fill flash LED
- Dimension : 25mm x 24mm

1.6. Bluetooth module



Εικόνα 12. Το Bluetooth module HC-06

Bluetooth protocol	Bluetooth Specification v2.0+EDR
Frequency	2.4GHz ISM band
Modulation	GFSK(Gaussian Frequency Shift Keying)
Emission power	≤4dBm, Class 2
Sensitivity	≤-84dBm at 0.1% BER
Speed: Asynchronous	2.1Mbps(Max) / 160 kbps, Synchronous: 1Mbps/1Mbps
Security	Authentication and encryption
Profiles	Bluetooth serial port
Power supply	+3.3VDC 50mA
Working temperature	-20 ~ +75 Centigrade
Dimension	26.9mm x 13mm x 2.2 mm

1.7. Bluetooth Dongle

Εικόνα 13. Το Bluetooth dongle για το Raspberry Pi

Interface (Bus) Type	USB 2.0
Type	Network adapter
Connectivity Technology	Wireless
Data Link Protocol	Bluetooth, Bluetooth 1.2, Bluetooth 2.0 EDR
Compliant Standards	Bluetooth 1.1, Bluetooth 1.2, Bluetooth 2.0
Max Range Indoors	10m
Max Range Open Space	100m

1.8. Ο σερβοκινητήρας

Εικόνα 14. Ο σερβοκινητήρας MG-90S

Weight	13.4 g
Dimension	22.5 x 12 x 35.5mm
Stall torque	1.8 kg·cm (4.8V), 2.2 kg·cm (6 V)
Operating speed	0.1 s/60 degree (4.8V), 0.08s/60 degree (6 V)
Operating voltage	4.8 V – 6.0 V
Dead band width	5 µs

3. Διαδικασία πραγματοποίησης της εφαρμογής.

1.9.Ρύθμιση του Raspberry Pi

1.9.1. Αποθήκευση λογισμικού

Για να λειτουργήσει το Raspberry Pi απαιτείται η αποθήκευση σε μια κάρτα μνήμης microSD του λειτουργικού συστήματος. Το λειτουργικό σύστημα του Raspberry Pi είναι το Raspbian, το οποίο είναι λειτουργικό σύστημα Linux βασισμένο στην έκδοση Debian. Η λειτουργία μπορεί να γίνει είτε μέσω γραφικού περιβάλλοντος, είτε μέσω γραμμής εντολών.

Η διαδικασία που ακολουθεί έγινε μέσω γραμμής εντολών σε περιβάλλον Linux αφού δεν υπάρχει ακόμα διαθέσιμη αυτοματοποιημένη διαδικασία σε γραφικό περιβάλλον.

Απαραίτητη ήταν η εγκατάσταση βιβλιοθηκών, που αποτελούν εξαρτήσεις για τη βιβλιοθήκη OpenCV. Το βήμα αυτό πρέπει να γίνει πριν από την εγκατάσταση της βιβλιοθήκης OpenCV γιατί αλλιώς η τελευταία διακόπτεται, ή δεν ξεκινά καθόλου ζητώντας την εγκατάσταση των εξαρτήσεων.

1.9.2. Αποθήκευση βιβλιοθήκης OpenCV

Το επόμενο βήμα που είναι η εγκατάσταση της βιβλιοθήκης OpenCV είναι ιδιαίτερα απαιτητικό. Η OpenCV είναι ένα αρχείο βιβλιοθηκών που μπορεί να χρησιμοποιηθεί από όλες τις πλατφόρμες. Για το λόγο αυτό δεν υπάρχει αυτοματοποιημένος τρόπος εγκατάστασης στις πλατφόρμες, αλλά πρέπει να γίνει build της βιβλιοθήκης από τον ανοιχτό κώδικα, με την εντολή CMAKE, στην αντίστοιχη πλατφόρμα που θα γίνει η χρήση της. Αυτό αποτελεί μειονέκτημα, γιατί ο χρόνος που απαιτείται για την εκτέλεση αυτής της εντολής ποικίλει στις διάφορες πλατφόρμες και συνήθως είναι αρκετά μεγάλος. Εξαρτάται από τον επεξεργαστή και τη μνήμη του συστήματος. Για το Raspberry Pi 2 B ήταν περίπου 6 ώρες. Για αυτό το λόγο η σειρά εγκατάστασης των προαπαιτούμενων βιβλιοθηκών έπρεπε να γίνει πολύ προσεκτικά. Δυστυχώς στο βήμα αυτό τα προβλήματα δεν αποφεύχθηκαν. Η διαδικασία αυτή επαναλήφθηκε αρκετές φορές. Το πρόβλημα κάθε φορά ήταν η

διακοπή αρχείων στην κάρτα μνήμης πριν ή αφού είχε τελειώσει η εγκατάσταση. Διαπιστώθηκε ότι δύο ήταν οι λόγοι του προβλήματος. Ο πρώτος ήταν η τροφοδοσία της συσκευής, που ανά διαστήματα δεν επαρκούσε για τις απαιτήσεις του συστήματος, και ο δεύτερος ήταν η συμβατότητα της κάρτας μνήμης.

Η μέγιστη απαίτηση του συστήματος σε ρεύμα είναι 2A. Παρόλο που χρησιμοποιήθηκε φορτιστής κινητού με έξοδο 2A, όπως ήταν ενδεδειγμένο, πιθανότατα δεν ικανοποιούνταν η ζήτηση ισχύος του συστήματος. Για αυτό χρησιμοποιήθηκε το ιδιοκατασκευής τροφοδοτικό πάγκου γενικής χρήσης, με δυνατότητα εξόδου ρεύματος 26A στα 5V.

Η αρχική κάρτα μνήμης που χρησιμοποιήθηκε, συμπεριλαμβάνονταν στη λίστα των συμβατών καρτών για το Raspberry Pi όπως αυτή δημιουργήθηκε από χρήστες της συσκευής μετά από ελέγχους. Σημαντική απαίτηση για την κάρτα είναι να είναι κλάσης 10. Στην αρχική κάρτα δεν υπήρχε πρόβλημα λειτουργίας με το λειτουργικό σύστημα, αλλά μόνο όταν γινόταν η εγκατάσταση της βιβλιοθήκης OpenCV.

Τελευταίο βήμα όσων αφορά το Raspberry Pi ήταν η ρύθμιση των παραμέτρων της συσκευής Bluetooth, τόσο στις σειριακές θύρες του λειτουργικού συστήματος, όσο και στον κώδικα της εφαρμογής που καλεί τη συγκεκριμένη σειριακή θύρα.

1.9.3. Ρύθμιση του Arduino

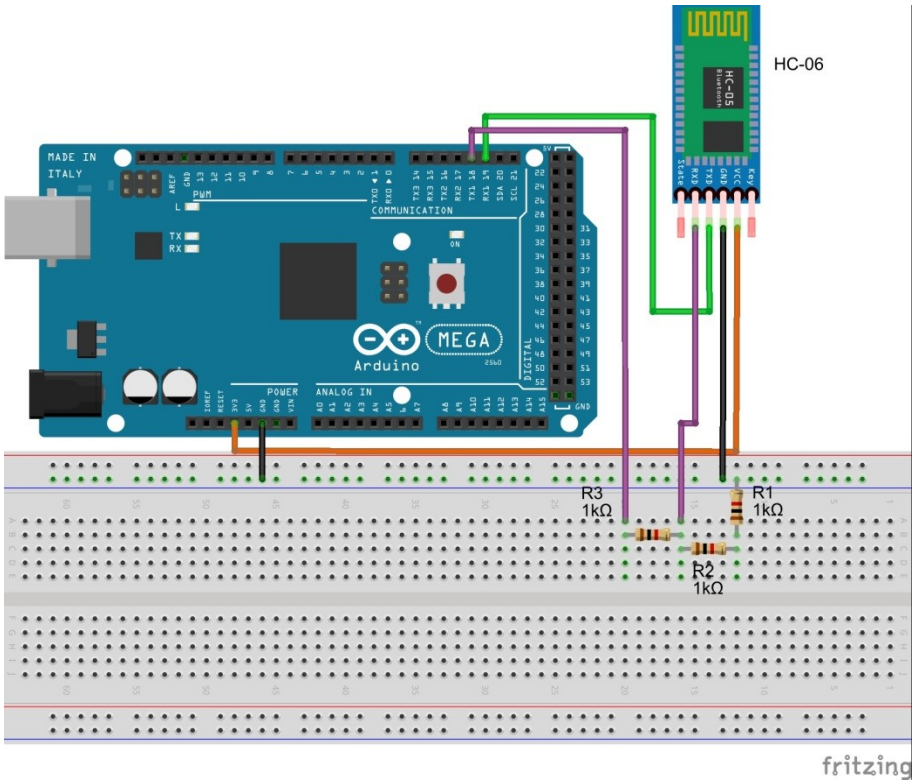
Επόμενο βήμα στην κατασκευή αποτέλεσε η ρύθμιση του Arduino. Η διαδικασία χωρίστηκε σε δυο μέρη. Πρώτο μέρος ήταν ο έλεγχος των σερβοκινητήρων που ήταν μια απλή διαδικασία κάνοντας χρήση της βιβλιοθήκης *servo.h*.

Το δεύτερο μέρος, που ήταν η επικοινωνία με το Bluetooth module, ήταν αρκετά πιο σύνθετο. Η τροφοδοσία και τα σήματα TX-RX του Bluetooth module είναι στα 3.3V, ενώ αυτά του Arduino είναι στα 5V. Για το λόγο αυτό αρχικά χρησιμοποιήθηκε ένας μετατροπέας τάσης σε τάση 5V-3.3V, για να υπάρχει συμβατότητα στα σήματα επικοινωνίας. Ο μετατροπέας όμως δημιούργησε αρκετά προβλήματα στην ποιότητα των δεδομένων που αποστέλλονταν, και για αυτό το λόγο δεν έγινε τελικά η χρήση του. Εναλλακτικά χρησιμοποιήθηκε ένας διαιρέτης τάσης

στην έξοδο TX του Arduino – είσοδο RX του Bluetooth module. Χρησιμοποιήθηκε ένας μόνο διαιρέτης τάσης αφού το Arduino έχει τη δυνατότητα αναγνώρισης σημάτων επιπέδου 3.3V στις εισόδους του ως λογικό 1. Το αποτέλεσμα ήταν πολύ ικανοποιητικό.

Υπάρχουν αναφορές στο διαδίκτυο για την απευθείας σύνδεση των σημάτων επικοινωνίας των δυο συσκευών και την ικανοποιητική λειτουργία τους, χωρίς όμως να εγγυώνται τη διάρκεια ζωής του Bluetooth module.

Παρακάτω φαίνεται η συνδεσμολογία των δυο συσκευών με τον διαιρέτη τάσης. Το σχέδιο δημιουργήθηκε στην εφαρμογή *Fritzing*.



Εικόνα 15. Συνδεσμολογία Arduino - Bluetooth

Επόμενο βήμα ήταν ο συνδυασμός των δυο μερών. Στη πειραματική διάταξη της εφαρμογής χρησιμοποιήθηκε το Arduino Mega. Δυο ήταν οι λόγοι της χρήσης της συγκεκριμένης πλακέτας Arduino. Ο πρώτος ήταν για την αποσφαλμάτωση. Το Arduino Mega διαθέτει τέσσερις θύρες επικοινωνίας σειριακών δεδομένων Serial, Serial1, Serial2, Serial3. Η πρώτη θύρα Serial, όπως και σε όλα τα Arduino,

ταυτίζεται με τη θύρα USB, και χρησιμοποιείται για την επικοινωνία της πλακέτας με τον υπολογιστή, για λόγους προγραμματισμού και αποσφαλμάτωσης. Το Bluetooth module HC-06 απαιτεί και αυτό μια θύρα σειριακής επικοινωνίας. Δεν ήταν δυνατό λοιπόν με τις υπόλοιπες πλακέτες να γίνει ταυτόχρονα αποσφαλμάτωση και επικοινωνία Bluetooth.

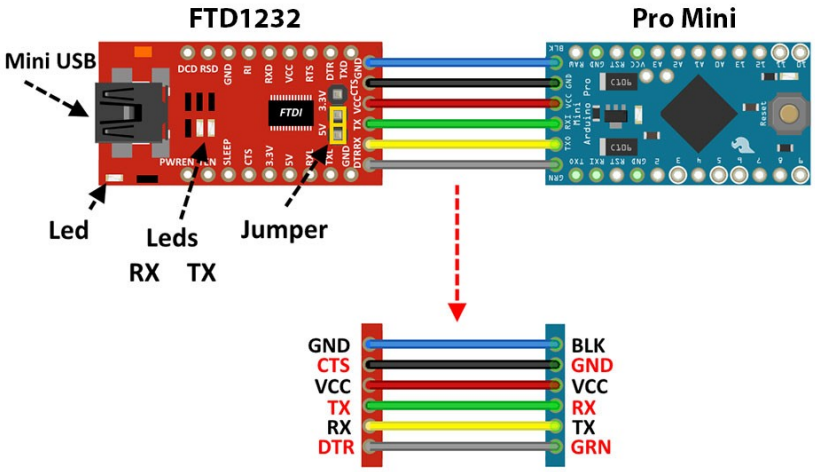
Στο διαδίκτυο υπήρχαν αναφορές για τη χρήση της βιβλιοθήκης *SoftwareSerial.h* του Arduino, η οποία δημιουργεί μια σειριακή θύρα επικοινωνίας με προγραμματιστικό τρόπο. Η συγκεκριμένη βιβλιοθήκη από μόνη της λειτουργεί σωστά. Όταν όμως χρησιμοποιηθεί ταυτόχρονα με τη βιβλιοθήκη *Servo.h*, που είναι απαραίτητη για την οδήγηση των σερβοκινητήρων, τότε δημιουργούνται διενέξεις και το πρόγραμμα δεν λειτουργεί σωστά. Στην περίπτωση που δεν χρειάζεται να γίνει αποσφαλμάτωση, όλες οι πλακέτες Arduino λειτουργούν σωστά, αφού το Bluetooth module μπορεί να συνδεθεί απευθείας στη θύρα επικοινωνίας Serial.

Με τον συγκεκριμένο τρόπο, έχει πραγματοποιηθεί πειραματικός έλεγχος λειτουργίας με τις πλακέτες Arduino UNO, και Arduino Pro Minikαι το πρόγραμμα λειτούργησε χωρίς κανένα πρόβλημα.

Ο δεύτερος λόγος είναι το πλήθος των εξόδων παλμού διαμόρφωσης πλάτους (Pulse Width Modulation, PWM), που είναι απαραίτητες για την οδήγηση των σερβοκινητήρων. Το Arduino Mega διαθέτει 15 εξόδους PWM, ενώ το Arduino UNO και το Arduino Pro Mini διαθέτουν μόνο 6. Αυτές είναι αρκετές για την οδήγηση των πέντε σερβοκινητήρων που αντιστοιχούν στην κατακόρυφη κίνηση των δαχτύλων. Η εφαρμογή όμως υπολογίζει και τη γωνία που σχηματίζουν τα δάχτυλα μεταξύ τους. Έτσι υπάρχει επεκτασιμότητα από τους πέντε βαθμούς ελευθερίας στους εννιά. Το Arduino Mega λοιπόν αποτελεί μια καλή λύση για την εκμετάλλευση των παραπάνω δυνατοτήτων.

Το μειονέκτημα της χρήσης του Arduino Mega είναι μόνο το μέγεθος της πλακέτας, αφού σε σύγκριση με τα άλλα δυο Arduino, είναι αρκετά μεγαλύτερο. Για απλή χρήση της εφαρμογής με πέντε βαθμούς ελευθερίας και χωρίς την απαίτηση αποσφαλμάτωσης , ιδανική λύση αποτελεί το Arduino Pro Mini, αφού οι διαστάσεις του είναι 3cm x 1.8cm μόνο. Και σε αυτό όμως πάλι μειονέκτημα αποτελεί το γεγονός ότι απαιτείται εξωτερικός προγραμματιστής για τον προγραμματισμό του,

πράγμα όμως που συντελεί στο μικρό του μέγεθος. Στην παρακάτω εικόνα φαίνεται ο η συνδεσμολογία του εξωτερικού προγραμματιστή στο Arduino Pro Mini.



Εικόνα 16. Σύνδεση εξωτερικής συσκευής προγραμματισμού. Μετατροπέας USB σε σειριακή μέσω FTDI232

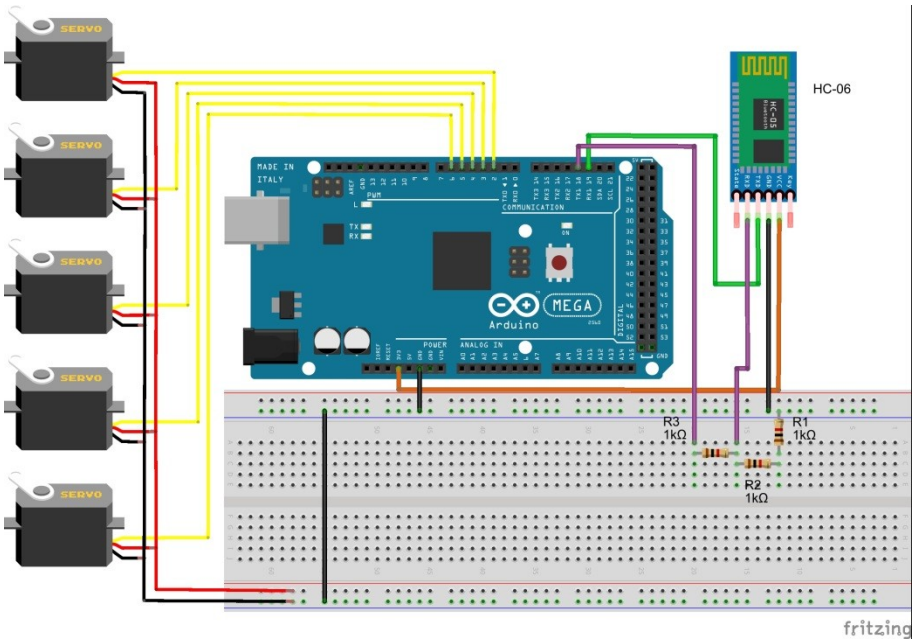
Συνοπτικά παρουσιάζονται τα πλεονεκτήματα και μειονεκτήματα των τριών πλακετών Arduino.

Πίνακας 2. Πλεονεκτήματα – Μειονεκτήματα πλακετών Arduino

Πλακέτα Arduino	Πλεονεκτήματα	Μειονεκτήματα
Arduino Mega	• Απευθείας προγραμματισμός μέσω USB • Τέσσερις σειριακές θύρες επικοινωνίας • 15 PWM έξοδοι	• Μεγάλο μέγεθος 10.2cm x 5.3cm
Arduino UNO	• Απευθείας προγραμματισμός μέσω USB • Μεσαίο μέγεθος 6.8cm x 5.3cm	• Μια σειριακή θύρα επικοινωνίας • 6 PWM έξοδοι
Arduino Pro Mini	• Πολύ μικρό μέγεθος 3cm x 1.8cm	• Μια σειριακή θύρα επικοινωνίας • 6 PWM έξοδοι

		• Εξωτερικός προγραμματιστής

Η τελική συνδεσμολογία συσκευών έχει πραγματοποιηθεί σύμφωνα με το παρακάτω σχέδιο.



Εικόνα 17. Τελική συνδεσμολογία συσκευών (Fritzing)

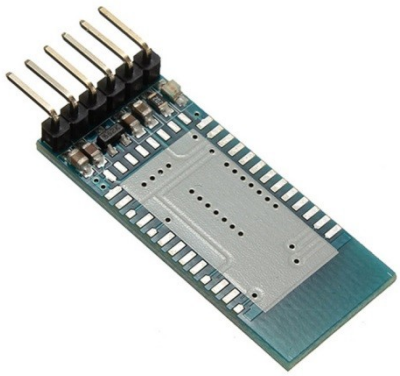
1.10. Συσκευή ασύρματης επικοινωνίας (Bluetooth module HC-06)

Η συσκευή HC-06 είναι μια συσκευή ασύρματης επικοινωνίας βασισμένη στην τεχνολογία Bluetooth [7]. Βασίζεται στο ολοκληρωμένο ασύρματης επικοινωνίας Bluetooth BC417 2.4 GHz της Cambridge Silicon [7] [8] το οποίο χρησιμοποιεί μια εξωτερική μνήμη Flash των 8Mbit. Η διασύνδεση της συσκευής γίνεται με το πρωτόκολλο σειριακής επικοινωνίας (Baud Rate: 9600 bps, Data : 8 bits, Stop Bits: 1 bit, Parity : None, Handshake: None). Η τάση λειτουργίας καθώς και τα σήματα σειριακής επικοινωνίας είναι 3.3V.

Το module, το οποίο ενσωματώνει και την κεραία ασύρματης επικοινωνίας, είναι τοποθετημένο πάνω σε μια πλακέτα διασύνδεσης. Η πλακέτα διαθέτει 4

ακροδέκτες Vcc, GND, TX, RX, από τους οποίους γίνεται η τροφοδοσία της συσκευής και η μεταφορά δεδομένων μέσω της σειριακής επικοινωνίας. Η τροφοδοσία της πλακέτας μπορεί να είναι 5V ή 3.3V αφού διαθέτει voltage regulators. Αντιθέτως όμως η τάση στα pins των δεδομένων TX, RX πρέπει να είναι αυστηρά 3.3V, αλλιώς υπάρχει κίνδυνος καταστροφής της συσκευής.

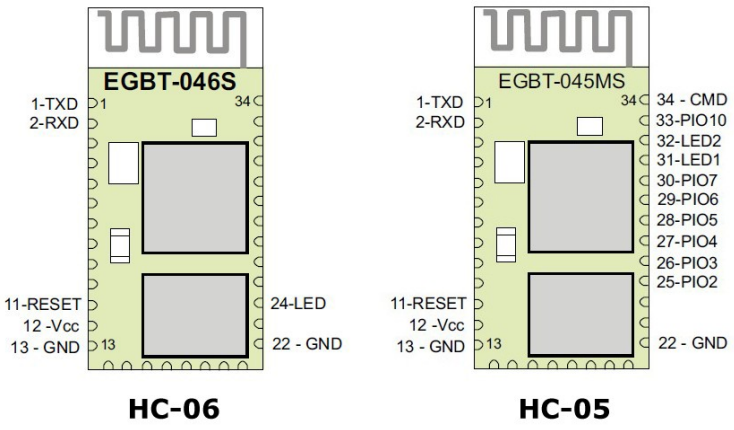
Εικόνα 18. Το Bluetooth module HC-06 χωρίς τη βάση διασύνδεσης



Εικόνα 19. Η βάση διασύνδεσης του HC-06

Παρόμοια συσκευή με την HC-06, είναι η HC-05. Στο εμπόριο είναι διαθέσιμες και οι δυο συσκευές με πολύ χαμηλό κόστος. Οι δυο συσκευές κατασκευαστικά βασίζονται στο ίδιο hardware με βασική μόνο διαφορά στο firmware. Η HC-06 λειτουργεί μόνο σε slave mode ενώ η HC-05 σε master/slave mode με κατάλληλο προγραμματισμό [9]. Και οι δυο συσκευές διαθέτουν εντολές προγραμματισμού (AT), με τις οποίες μπορεί να γίνει αλλαγή βασικών χαρακτηριστικών, όπως το όνομα συσκευής, το Baud Rate, και ο κωδικός

πρόσβασης, καθώς επίσης και άλλων χαρακτηριστικών που αφορούν στη λειτουργία της συσκευής [8] [9].



Εικόνα 20. Οι συσκευές Bluetooth HC-06 και HC-05

Στη παρούσα εργασία μπορούν να χρησιμοποιηθούν και οι δυο συσκευές ως slave Bluetooth modules, αφού το Raspberry Pi είναι το master Bluetooth module. Έχει πραγματοποιηθεί πειραματική λειτουργία και με τα δυο modules, με τα ίδια ακριβώς αποτελέσματα στον τρόπο επικοινωνίας. Η επιλογή του HC-06 έγινε για το λόγο ότι διαθέτει λιγότερα pins διασύνδεσης.

Τα εργοστασιακά στοιχεία επικοινωνίας με το HC-06 είναι:

Όνομα συσκευής:	HC-06
Κωδικός δημιουργίας ζεύγους:	1234
Baud Rate:	9600

1.10.1. Τρόπος επικοινωνίας του HC-06

Το HC-06 είναι στην πράξη μια συσκευή μετάφρασης του πρωτοκόλλου Bluetooth σε πρωτόκολλο σειριακής επικοινωνίας και αντίστροφα. Χρειάζεται ιδιαίτερη προσοχή στον τρόπο σύνδεσης των pins TX (Transmit) και RX (Receive) του module. Το TX αναφέρεται στην σειριακή εκπομπή των δεδομένων τα οποία έχουν ήδη ληφθεί από την επικοινωνία μέσω Bluetooth, και έχουν αποθηκευτεί στο buffer της συσκευής, ενώ το RX αναφέρεται στα σειριακά δεδομένα που λαμβάνει η συσκευή και πρόκειται να σταλούν μέσω Bluetooth.

Ο τρόπος επικοινωνίας του HC-06 περιγράφεται παρακάτω. Τα δεδομένα που λαμβάνονται από τη συσκευή ασύρματα, μεταφράζονται και αποθηκεύονται στο buffer με τη μορφή byte. Το Arduino κάνει έλεγχο για αποθηκευμένα bytes στο buffer του HC-06. Αν υπάρχουν στέλνει αίτημα λήψης. Κάθε φορά λαμβάνει ένα μόνο byte δεδομένων προς επεξεργασία, το οποίο ταυτόχρονα διαγράφεται από τη μνήμη του buffer.

1.10.2. Πρωτόκολλο επικοινωνίας Raspberry Pi και Arduino.

Τα δεδομένα που στέλνει το Raspberry Pi στο Arduino είναι οι 5 γωνίες που θα πρέπει να γυρίσουν οι σερβοκινητήρες, ώστε να αντικατοπτρίσουν τις κινήσεις των δαχτύλων. Επειδή τα δεδομένα που αποστέλλονται είναι σειριακά, και επειδή το Arduino μπορεί να διαβάσει ένα byte κάθε φορά από το buffer του Bluetooth module, έπρεπε να δημιουργηθεί ένα πρωτόκολλο επικοινωνίας μεταξύ των δυο συσκευών.

Το Raspberry Pi στέλνει πακέτα δεδομένων, όπου το κάθε πακέτο αποτελείται από 11 byte στη μορφή <a1,a2,a3,a4,a5> όπου a1,a2,a3,a4,a5 είναι οι γωνίες που αντιστοιχούν στα 5 σερβό. Τα δεδομένα του πακέτου αναλυτικότερα φαίνονται στον παρακάτω πίνακα.

Πίνακας 3. Το πακέτο δεδομένων που αποστέλλεται μέσω Bluetooth

Πακέτο δεδομένων										
<	a1	,	a2	,	a3	,	a4	,	a5	>
byte1	byte2	byte3	byte4	byte5	byte6	byte7	byte8	byte9	byte10	byte11

όπου:

< : χαρακτήρας αρχής δεδομένων

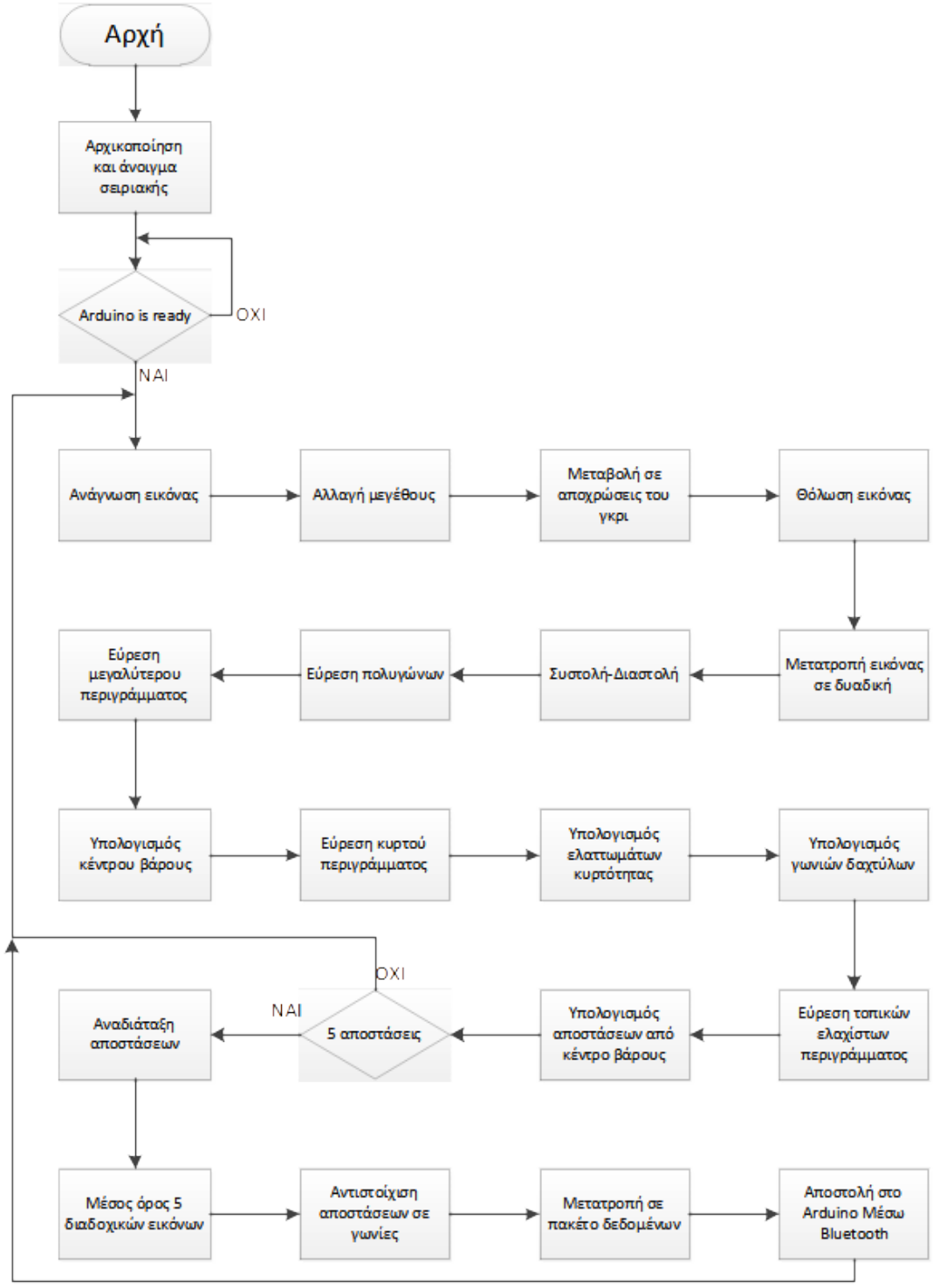
a_i : δεδομένα

, : σύμβολο διαχωρισμού δεδομένων

> : σύμβολο τέλους δεδομένων

4. Περιγραφή λειτουργιών στο Raspberry Pi

1.11. Διάγραμμα ροής κώδικα στο Raspberry Pi



Σχήμα 4. Διάγραμμα ροής του κώδικα στο Raspberry Pi.

1.12. Ανάλυση της διαδικασίας και του κώδικα

```
while cap.isOpened():
ret, frame1 = cap.read()
```

```
    frame = cv2.resize(frame1, (640, 480))
```

Εδώ αρχίζουν οι επαναλήψεις του προγράμματος. Η διαδικασία που περιγράφεται παρακάτω, επαναλαμβάνεται κάθε φορά που γίνεται η ανάγνωση μιας εικόνας. Όσο η κάμερα είναι ανοιχτή, γίνεται ανάγνωση μιας εικόνας και στη συνέχεια αλλαγή του μεγέθους της. Η αλλαγή του μεγέθους δεν είναι απαραίτητη, αλλά μικρότερη εικόνα σημαίνει και λιγότερα pixels για επεξεργασία, και κατά συνέπεια μεγαλύτερη ταχύτητα και μικρότερη επεξεργαστική ισχύς.

Εικόνα 21. Αρχική εικόνα προς επεξεργασία

```
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

Η εικόνα frame που είναι αποθηκευμένη, μετατρέπεται τώρα από έγχρωμη σε ασπρόμαυρη των 8 bit/pixel .

Εικόνα 22. Μετατροπή εικόνας σε ασπρόμαυρη

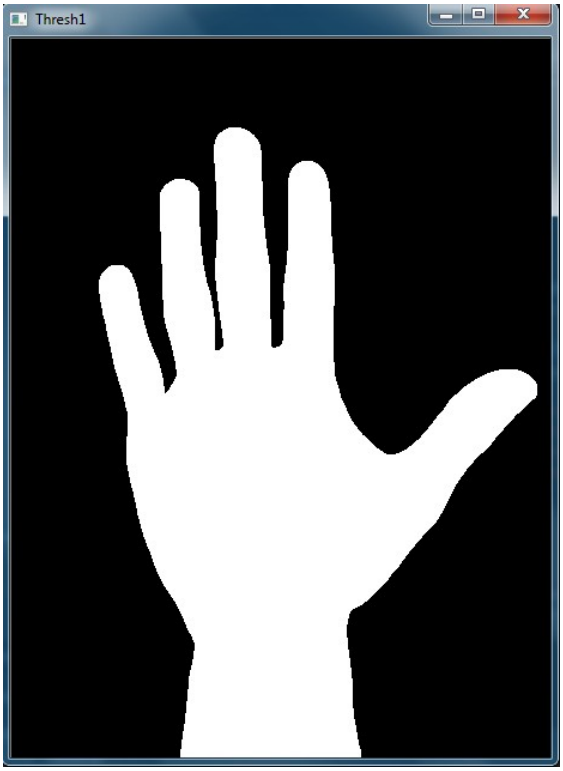
```
blurred = cv2.GaussianBlur(gray, (35, 35), 0)
```

Η ασπρόμαυρη εικόνα θολώνεται για να γίνει εξάλειψη του θορύβου, χρησιμοποιώντας τη μέθοδο *cv2.GaussianBlur*. Με τη μέθοδο αυτή γίνεται συνέλιξη της εικόνας με ένα Gaussian φίλτρο, το οποίο αποτελεί ένα χαμηλοπερατό φίλτρο. Η εφαρμογή του γίνεται σε τετράγωνα 35 x 35 pixel διατηρώντας τυπική απόκλιση 0 στον άξονα των x [10].

Εικόνα 23. Θόλωση εικόνας

```
_, thresh1 = cv2.threshold(blurred, 60, 255, cv2.THRESH_BINARY+cv2.THRESH_OTSU)
```

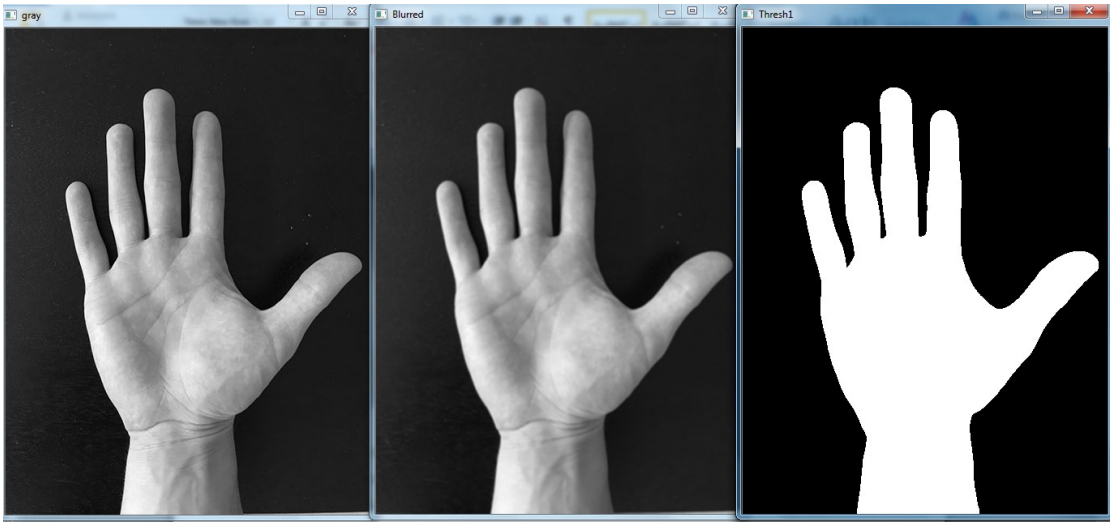
Τελικά με τη μέθοδο *cv2.threshold*, η ασπρόμαυρη εικόνα των 8 bit/pixel (256 αποχρώσεις του γκρι) μετατρέπεται σε μια εικόνα δυαδική 1 bit/pixel (1=άσπρο – 0=μαύρο). Κάθε pixel της εικόνας συγκρίνεται με μια προκαθορισμένη τιμή στη κλίμακα του γκρι, που ορίζεται από τον χρήστη. Αν το pixel έχει μεγαλύτερη τιμή, τότε παίρνει την τιμή 1, ενώ στην αντίθετη περίπτωση παίρνει την τιμή 0. Το τελικό αποτέλεσμα εξαρτάται από τις τιμές του δεύτερου, τρίτου και τέταρτου ορίσματος της μεθόδου *cv2.threshold.*



Εικόνα 24. Μετατροπή εικόνας σε δυαδική

Εδώ αξίζει να σημειωθεί ότι η OpenCV επεξεργάζεται λευκά αντικείμενα σε μαύρο φόντο. Για αυτό το λόγο, θα πρέπει να γίνει κατάλληλη παραμετροποίηση της μεθόδου για το σωστό φόντο που θα χρησιμοποιηθεί.

Στην επόμενη εικόνα φαίνονται συνοπτικά τα τρία βήματα της επεξεργασίας



Εικόνα 25. Στιγμιότυπα διαδοχικής μορφοποίησης εικόνας

Μέχρι αυτή τη στιγμή η επεξεργασία της εικόνας έχει φτάσει στο σημείο όπου η περιοχή ενδιαφέροντος είναι λευκή σε μαύρο φόντο.

```
thresh1 = cv2.erode(thresh1, None, iterations=2)
thresh1 = cv2.dilate(thresh1, None, iterations=2)
```

Με τις μεθόδους *cv2.erode* και *cv2.dilate* γίνεται η 'συστολή' και η 'διαστολή' της εικόνας. Και στις δυο μεθόδους χρησιμοποιείται ένας πυρήνας συνήθως 3x3 pixel (πάντα περιττό πλήθος pixel στις διαστάσεις). Χρησιμοποιώντας ως βάση το κεντρικό σημείο του πυρήνα, η εικόνα σαρώνεται και συγκρίνονται τα σημεία του πυρήνα με το κεντρικό σημείο. Στη περίπτωση της συστολής, το κεντρικό σημείο παίρνει τη μικρότερη τιμή από όλα τα σημεία του πυρήνα, ενώ αντίθετα στη περίπτωση της διαστολής παίρνει τη μεγαλύτερη τιμή. Σε μια εικόνα λοιπόν που όλα τα pixels έχουν τιμή 0 ή 1 (0=μαύρο, 1=λευκό) με τη συστολή μικραίνουν οι λευκές περιοχές, ενώ με τη διαστολή μεγαλώνουν. Η εφαρμογή τους γίνεται συνήθως μαζί ώστε να διατηρηθούν οι αρχικές διαστάσεις της περιοχής ενδιαφέροντος. Παράδειγμα συστολής και διαστολής φαίνεται στις παρακάτω εικόνες.

Εικόνα 26. Εφαρμογή συστολής (erotion)

Εικόνα 27. Εφαρμογή διαστολής (dilation)

Αποτέλεσμα της εφαρμογής των δυο μεθόδων είναι η εξάλειψη μεμονωμένων ή μικρών ομάδων pixel τα οποία αποτελούν θόρυβο και ίσως δημιουργήσουν πρόβλημα στην υπόλοιπη διαδικασία. [11]

```
_, contours, hierarchy = cv2.findContours(thresh1, cv2.RETR_TREE,
   cv2.CHAIN_APPROX_SIMPLE)
# Largest area contour
cnts = contours[ci]
```

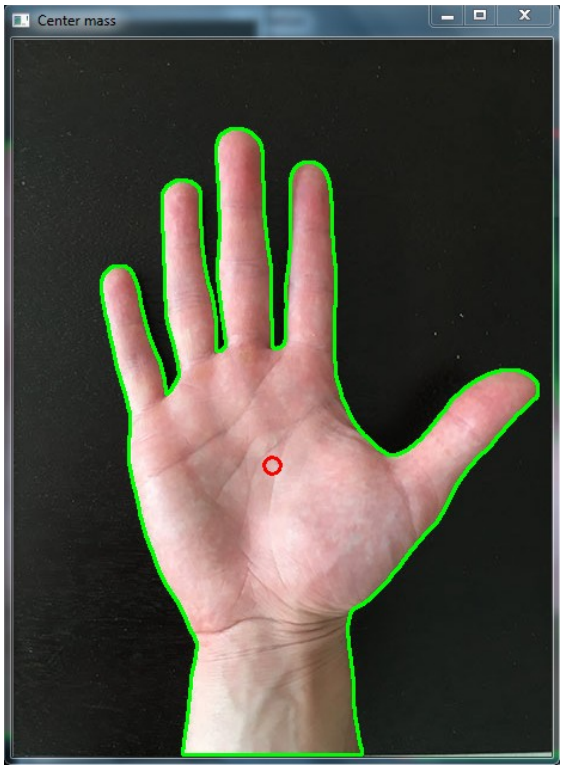
Η μέθοδος *cv2.findContours* είναι από τις σημαντικότερες μεθόδους στην επεξεργασία της εικόνας. Εξάγει όλα τα περιγράμματα που υπάρχουν μέσα στην εικόνα. Στη συνέχεια γίνεται η επιλογή του περιγράμματος με το μεγαλύτερο εμβαδό και αποθηκεύονται τα σημεία των κορυφών του.

Εικόνα 28. Μεγαλύτερο περίγραμμα όμοιων pixels μέσα στην εικόνα

```
        moments = cv2.moments(cnts)
        # Central mass of first order moments
```

Από τα σημεία αυτά με τη μέθοδο *cv2.moments* γίνεται η εξαγωγή των χαρακτηριστικών σημείων του πολυγώνου. Το κέντρο βάρους του πολυγώνου είναι

ένα από αυτά τα σημεία, το οποίο είναι και το πρώτο σημαντικό σημείο για τον έλεγχο των σερβοκινητήρων.



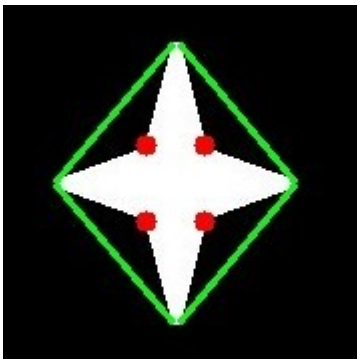
Εικόνα 29. Κέντρο βάρους περιγράμματος

```
        # Find convex hull
        hull = cv2.convexHull(cnts)
        # Find convexity defects
        hull2 = cv2.convexHull(cnts, returnPoints=False)
        defects = cv2.convexityDefects(cnts, hull2)
```

Στη συνέχεια η μέθοδος *cv2.convexHull* δέχεται ως παράμετρο τα σημεία του περιγράμματος που έχουν βρεθεί νωρίτερα και δίνει στην έξοδο τις κορυφές ενός κυρτού πολυγώνου που περικλείει όλα τα σημεία του περιγράμματος. Με αυτό τον τρόπο σχηματίζεται ένα κυρτό πολύγωνο μέσα στο οποίο βρίσκεται ολόκληρο το αντικείμενο προς αναγνώριση.

Με τη βοήθεια των σημείων αυτών και με τη μέθοδο *cv2.convexityDefects*, υπολογίζονται τα ελαττώματα κυρτότητας. Τα ελαττώματα κυρτότητας είναι τα σημεία του περιγράμματος, που βρέθηκε νωρίτερα, τα οποία απέχουν τη μεγαλύτερη απόσταση από δυο διαδοχικές κορυφές του κυρτού πολυγώνου. Στην παρακάτω εικόνα για παράδειγμα, το άσπρο αστέρι είναι το αντικείμενο προς αναγνώριση, το

πράσινο περίγραμμα είναι το κυρτό πολύγωνο που βρίσκεται με τη μέθοδο *cv2.convexHull* και οι κόκκινες τελείες είναι τα ελαττώματα κυρτότητας [12].



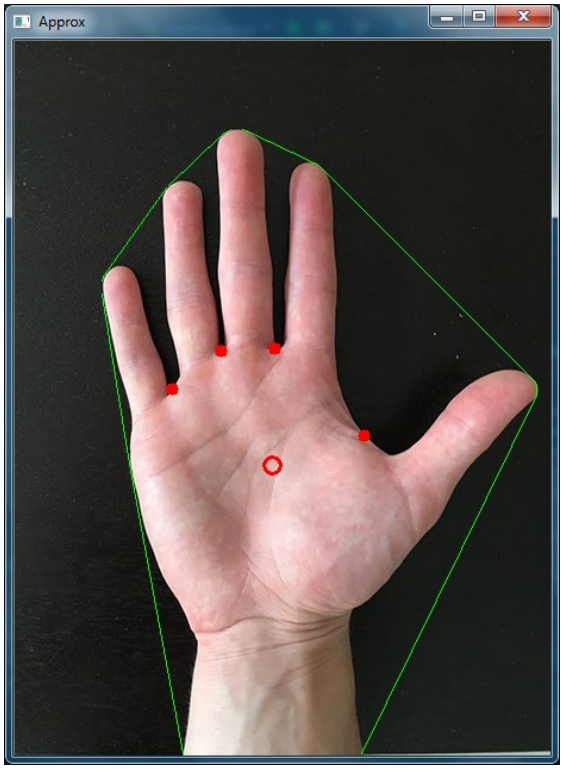
Εικόνα 30. Παράδειγμα ελαττωμάτων κυρτότητας

Στην περίπτωση που το αντικείμενο προς αναγνώριση είναι το χέρι, αφού βρεθούν τα ελαττώματα κυρτότητας, υπολογίζονται οι γωνίες με κορυφές τα σημεία αυτά και άκρα τις αντίστοιχες κορυφές του κυρτού πολυγώνου. Η γωνία που σχηματίζεται μεταξύ τριών σημείων $A(x_1, y_1)$, $B(x_2, y_2)$, $\Gamma(x_3, y_3)$, με κορυφή το σημείο Β, δίνεται από τη σχέση $\phi = \arccos\left(\frac{\overrightarrow{BA} \cdot \overrightarrow{B\Gamma}}{\left|\overrightarrow{BA}\right| \cdot \left|\overrightarrow{B\Gamma}\right|}\right)$ ή πιο αναλυτικά

$$\phi = \arccos\left(\frac{(x_1 - x_2)(x_3 - x_2) + (y_1 - y_2)(y_3 - y_2)}{\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \cdot \sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2}}\right).$$

Αποδεκτά στην περίπτωση του χεριού, είναι τα σημεία που οι αντίστοιχες γωνίες τους είναι μικρότερες των 120^o.

Στην παρακάτω εικόνα φαίνονται τα σημεία αυτά με κόκκινο χρώμα.



Εικόνα 31. Ελαττώματα κυρτότητας του πολυγώνου

```
        epsilon = eps*cv2.arcLength(cnts, True)
        approx      =      cv2.approxPolyDP(cnts,      epsilon,      closed=True)
#counterclockwise
```

Η μέθοδος *cv2.approxPolyDP* προσεγγίζει το σχήμα ενός περιγράμματος σε ένα άλλο σχήμα με μικρότερο αριθμό κορυφών, το οποίο εξαρτάται από την ακρίβεια που ορίζουμε εμείς μέσω της παραμέτρου epsilon. Το epsilon ορίζει τη μέγιστη απόσταση που μπορεί να έχει το αρχικό περίγραμμα από το προσεγγισμένο περίγραμμα. Είναι μια παράμετρος ακρίβειας και η επιλογή της πρέπει να γίνει με σωστό τρόπο. Η μέθοδος είναι μια εφαρμογή του αλγορίθμου Douglas-Peucker [13].

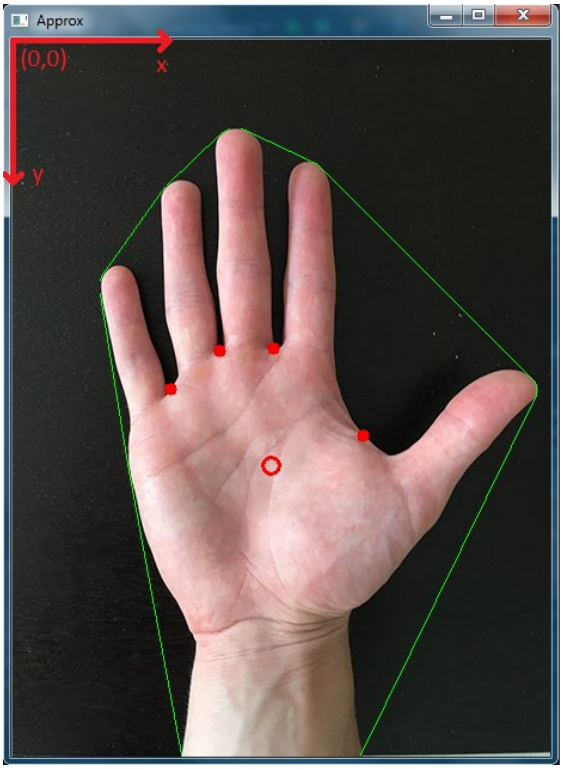
Στην παρακάτω εικόνα φαίνεται η εφαρμογή της μεθόδου για epsilon ίσο με 10% και 1% του μήκους της συνολικής καμπύλης [14].

Εικόνα 32. Παράδειγμα εφαρμογής της μεθόδου cv2.approxPolyDP. 10% ακρίβεια , 1% ακρίβεια

Εδώ αξίζει να γίνει μια σύγκριση της μεθόδου *cv2.approxPolyDP* με τη μέθοδο *cv2.convexHull* που χρησιμοποιήθηκε νωρίτερα. Οι δύο μέθοδοι με μια πρώτη ματιά φαίνονται ίδιες αλλά στην πραγματικότητα έχουν μια μεγάλη διαφορά. Ενώ η *cv2.approxPolyDP* προσεγγίζει ένα περίγραμμα με το ποσοστό ακρίβειας που ορίζουμε εμείς, η *cv2.convexHull* ελέγχει το περίγραμμα για ελαττώματα κυρτότητας και τα διορθώνει. Πιο απλά η πρώτη μέθοδος μπορεί να δώσει ένα πολύγωνο που περιέχει ελαττώματα κυρτότητας ενώ η δεύτερη δίνει πάντα ένα κυρτό πολύγωνο. Σε πολλές περιπτώσεις όμως μπορεί να δώσουν περιγράμματα που πλησιάζουν πολύ μεταξύ τους.

Το τελευταίο όρισμα της μεθόδου *cv2.approxPolyDP* επιτρέπει να ορισθεί αν το προσεγγισμένο πολύγωνο που θα εξαχθεί, θα είναι κλειστό ή ανοιχτό.

Τα σημεία του πολυγώνου που δίνει η *cv2.approxPolyDP* αποθηκεύονται τελικά στη λίστα *aprrox.* Επόμενο βήμα της διαδικασίας είναι η εύρεση των τοπικών ακροτάτων του πολυγώνου. Πριν γίνει όμως η περιγραφή της εύρεσης των ακροτάτων, πρέπει να αναφερθούν τα χαρακτηριστικά του επιπέδου στην Opencv. Όπως και σε πολλές άλλες εφαρμογές επεξεργασίας εικόνας, η Opencv δουλεύει με το Καρτεσιανό σύστημα συντεταγμένων. Ορίζει ως αρχή των συντεταγμένων (0,0), την επάνω αριστερή γωνία της εικόνας, με θετικό ημιάξονα των τετμημένων x, τον οριζόντιο άξονα και θετικό ημιάξονα των τεταγμένων y, τον κατακόρυφο ημιάξονα (εικόνα ?). Η μονάδα μέτρησης επάνω στους άξονες είναι το pixel. Είναι προφανές ότι δεν χρησιμοποιούνται οι αρνητικοί ημιάξονες των αξόνων. Η αποθήκευση των σημείων και σε ολόκληρη την προηγούμενη διαδικασία, γίνεται με αυτή την παραδοχή.



Εικόνα 33. Σύστημα συντεταγμένων της εικόνας στην OpenCV

Στα μαθηματικά, η εύρεση των ακροτάτων μιας συνάρτησης συνεχούς ή μη, γίνεται με τη βοήθεια της παραγώγου. Στη περίπτωση που η συνάρτηση είναι διακριτή, χρησιμοποιείται ο κανόνας της κλίσης ευθείας που διέρχεται από δυο σημεία. Η κλίση της ευθείας που διέρχεται από τα σημεία $A(x_1, y_1)$ και $B(x_2, y_2)$ δίνεται από τον τύπο $\lambda_{AB} = \frac{y_2 - y_1}{x_2 - x_1}$ με τον περιορισμό ότι $x_1 \neq x_2$. Αν λοιπόν για τρία διαδοχικά σημεία $A(x_1, y_1)$, $B(x_2, y_2)$, $\Gamma(x_3, y_3)$, η κλίση λ_{AB} της ευθείας που διέρχεται από τα σημεία Α και Β, έχει διαφορετικό πρόσημο από την κλίση $\lambda_{B\Gamma}$ της ευθείας που διέρχεται από τα σημεία Β και Γ, τότε το σημείο Β αποτελεί ακρότατο για τα σημεία Α, Β, Γ. Αποτελεί μέγιστο αν $\lambda_{AB} > 0$ και $\lambda_{B\Gamma} < 0$, ενώ ελάχιστο αν $\lambda_{AB} < 0$ και $\lambda_{B\Gamma} > 0$.

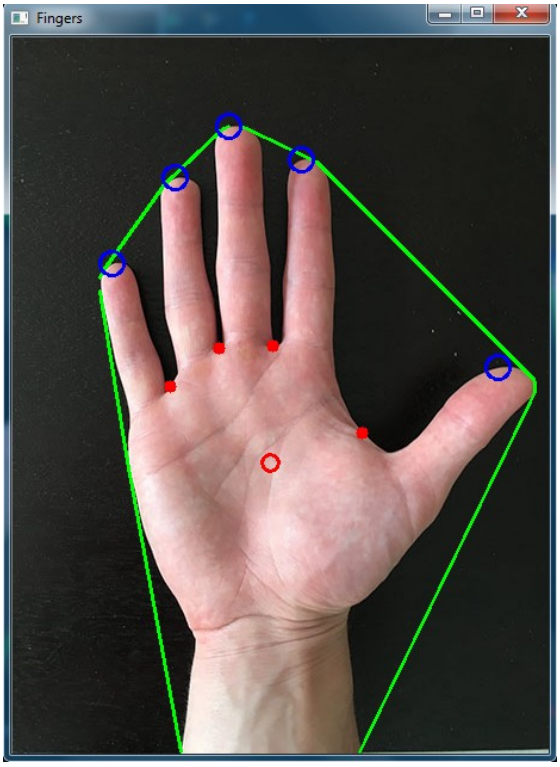
Επειδή στο σύστημα συντεταγμένων που χρησιμοποιεί η Opencv ο θετικός ημιάξονας y έχει φορά προς τα κάτω, οι άκρες των δακτύλων αποτελούν τοπικά ελάχιστα. Πρέπει να σημειωθεί εδώ ότι τα σημεία του πολυγώνου δεν αποτελούν διακριτά σημεία κάποιας συνάρτησης. Παρόλα αυτά η μέθοδος των κλίσεων εφαρμόζεται για τον εξής λόγο. Τα σημεία του πολυγώνου αποθηκεύονται στη λίστα

aprrox όχι με βάση τις τετμημένες τους αλλά ως διαδοχικά σημεία πολυγώνου κατά την φορά των δεικτών του ρολογιού ή αντίθετα. Στη συγκεκριμένη περίπτωση αποθηκεύονται κατά την αντίθετη φορά, πράγμα που μπορεί να αλλάξει, αλλάζοντας ένα όρισμα στη μέθοδο *cv2.approxPolyDP* .

Στην παρούσα εργασία αρχικά χρησιμοποιήθηκε αυτή η μέθοδος με αρκετά ικανοποιητικά αποτελέσματα. Χρησιμοποιήθηκε όμως ένας ακόμα περιορισμός. Για να αποτελέσει ένα σημείο τοπικό ελάχιστο θα πρέπει η απόλυτη τιμή του μέσου όρου των δυο κλίσεων αριστερά και δεξιά του σημείου, να είναι μικρότερη από έναν αριθμό (0,6) για να είναι κοντά σε οριζόντια εφαπτομένη. Επειδή όμως το πολύγωνο που προσεγγίζει το σχήμα αποτελείται από τεθλασμένες γραμμές που αλλάζουν συχνά κυρτότητα, η μέθοδος αυτή εμφάνιζε τοπικά ελάχιστα και σε σημεία που δεν είναι αποδεκτά για τον σκοπό της εργασίας. Με κατάλληλη διεργασία τα συγκεκριμένα σημεία μπορούσαν να απαλειφτούν από τη λίστα των τοπικών ακροτάτων. Διαπιστώθηκε λοιπόν ότι η μέθοδος αυτή ήταν αρκετά απαιτητική σε επεξεργαστική ισχύ.

Ο αλγόριθμος που τελικά χρησιμοποιήθηκε περιγράφεται παρακάτω. Από τη λίστα των σημείων πολυγώνου, γίνεται σύγκριση κάθε φορά τριών διαδοχικών σημείων. Αν υποθέσουμε πέντε διαδοχικά σημεία Α, Β, Γ, Δ, Ε, συγκρίνεται η τεταγμένη του σημείου Γ με τις τεταγμένες των σημείων Β και Δ. Αν είναι μικρότερη τότε συγκρίνεται με τις τεταγμένες των σημείων Α και Ε. Αν είναι μικρότερη και από αυτές, τότε το σημείο Γ αποτελεί τοπικό ακρότατο. Η δεύτερη σύγκριση γίνεται για την αποφυγή των πολλαπλών αλλαγών κυρτότητας που αναφέρθηκε προηγουμένως.

Για τον καθορισμό τελικά των πέντε ακροδαχτύλων, γίνεται ένας τελευταίος έλεγχος. Ελέγχεται η απόσταση δυο διαδοχικών ακροτάτων και αν είναι μικρότερη από ένα κατώφλι (11 pixels) τότε κρατάμε αυτό με τη μικρότερη τεταγμένη. Σε σύγκριση με την προηγούμενη μέθοδο, αυτή δίνει καλύτερα αποτελέσματα και με μικρότερη επεξεργαστική ισχύ. Στην παρακάτω εικόνα φαίνεται το αποτέλεσμα της εύρεσης των ακροδαχτύλων.



Εικόνα 34. Τοπικά ελάχιστα του περιγράμματος του χεριού

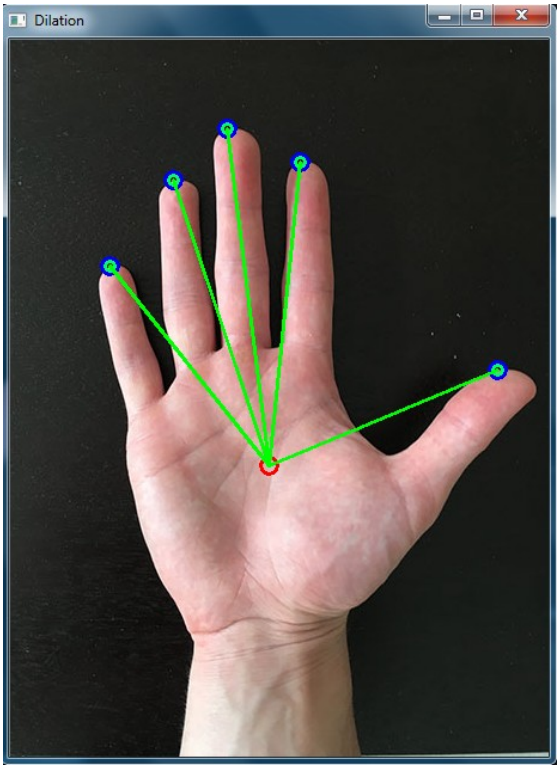
Στο σημείο αυτό έχει γίνει η εξαγωγή των τοπικών ελαχίστων και κατά συνέπεια των ακροδαχτύλων του χεριού. Στο επόμενο βήμα γίνεται η χρήση των ακροτάτων για την εύρεση της απόστασης του καθενός από αυτά, με το κέντρο βάρους του χεριού που υπολογίστηκε νωρίτερα.

```
        fingersx = sorted(maximum, key=lambda xp: xp[0])  # ,reverse=True)
                fingerDistance = []
                for i in range(0, len(fingersx)):
                    cordfing = tuple(maximum[i])
                    cv2.circle(frame, cordfing, 4, [100, 255, 0], 1)
                    cv2.line(frame, cordfing, centerMass, [0, 255, 0], 1)
                            distance1  =  int(np.sqrt(np.power(fingersx[i][0]-
centerMass[0], 2)+np.power(fingersx[i][1]-centerMass[1], 2)))
                    fingerDistance.append(distance1)
```

Από την Ευκλείδεια γεωμετρία, είναι γνωστό ότι η σχέση που δίνει την απόσταση δυο σημείων $A(x_1, y_1)$ και $B(x_2, y_2)$ του επιπέδου είναι $d(A,B) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

Η εύρεση των αποστάσεων πραγματοποιείται αφού τοποθετηθούν τα τοπικά ακρότατα σε μια λίστα με αύξουσα σειρά τετμημένων. Ο λόγος για τον οποίο γίνεται

αυτό, είναι για να έχουμε στη τελική λίστα τις αποστάσεις των δαχτύλων με τη σωστή σειρά. Στην παρακάτω εικόνα φαίνονται οι αποστάσεις που υπολογίζονται.



Εικόνα 35. Αποστάσεις ακροδαχτύλων από το κέντρο βάρους

Υπάρχει περίπτωση κάποιες φορές να μην αναγνωριστούν πέντε ακροδάχτυλα. Αυτό μπορεί να συμβεί λόγω της διαδικασίας του προγράμματος ως αποτέλεσμα ανάγνωσης μιας μη σαφούς εικόνας, είτε λόγω της θέσης των δακτύλων από την πλευρά του χρήστη. Ο παρακάτω κώδικας ελέγχει αν στη λίστα υπάρχουν πέντε αποστάσεις. Αν για κάποιο λόγο δεν είναι πέντε τότε προσπερνάει το υπόλοιπο πρόγραμμα και διαβάζει μια νέα εικόνα, επαναλαμβάνοντας όλη τη διαδικασία.

Για να αποφευχθούν επίσης λανθασμένες πληροφορίες για τον τρόπο κίνησης των ακροδαχτύλων, χρησιμοποιείται ο μέσος όρος των αποστάσεων για ένα πλήθος εικόνων, που ορίζεται από τον χρήστη.

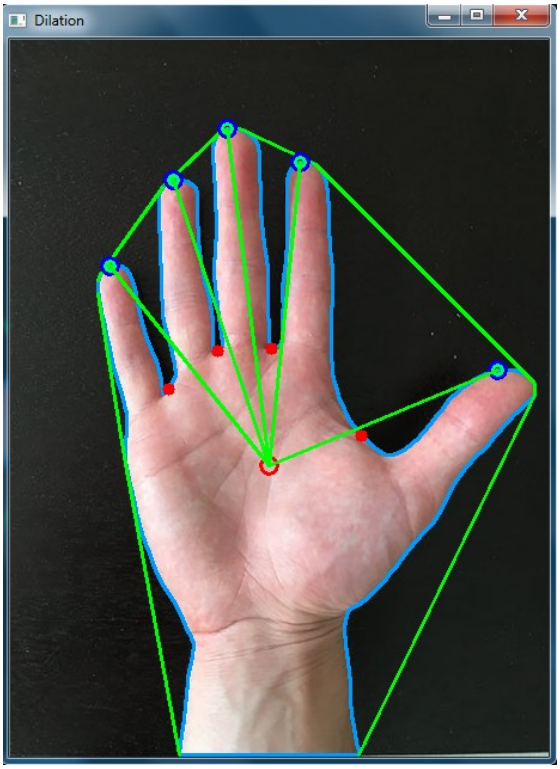
```
        if len(fingerDistance) == 5:
                for i in range(0, len(fingerDistance)):
                    datalist[i] = datalist[i] + fingerDistance[i]
                counter += 1
            #print counter
```

```
            if counter == bypass:
                f1 = datalist[0]/bypass
                f2 = datalist[1]/bypass
                f3 = datalist[2]/bypass
                f4 = datalist[3]/bypass
                f5 = datalist[4]/bypass


                s1 = mymap(f1)
                s2 = mymap(f2)
                s3 = mymap(f3)
                s4 = mymap(f4)
                s5 = mymap(f5)
        print datalist
            datalist = [0, 0, 0, 0, 0]
            counter = 0
```

Αφού οριστικοποιηθεί η λίστα των πέντε αποστάσεων, γίνεται αντιστοίχιση καθεμιάς από αυτές σε μια γωνία στο διάστημα $0^{\circ} - 180^{\circ}$, που είναι και η πληροφορία που θα αποσταλεί για την μετακίνηση των σερβοκινητήρων. Το τελευταίο βήμα πριν την έναρξη της διαδικασίας αποστολής των δεδομένων, είναι η μετατροπή των αριθμών της λίστας σε συμβολοσειρές και η δημιουργία της μορφής των δεδομένων που απαιτείται από το πρωτόκολλο επικοινωνίας που αναφέρθηκε σε προηγούμενη ενότητα.

```
                                                                    data                =
'<'+str(s1)+','+str(s2)+','+str(s3)+','+str(s4)+','+str(s5)+'>'
```

Συνοπτικά η επεξεργασία της εικόνας και τα χρήσιμα στοιχεία που υπολογίζονται, φαίνονται παρακάτω.



Εικόνα 36. Συνολικά τα στοιχεία που υπολογίζονται από το πρόγραμμα

1.13. Κώδικας για την αποστολή των δεδομένων

Το παρακάτω τμήμα του κώδικα τρέχει μια φορά μόνο, κατά την έναρξη της λειτουργίας του προγράμματος.

```
        serPort = 'COM6'   # "/dev/ttyS80"#'/dev/rfcomm0'
        baudRate = 9600
        bluetooth = serial.Serial(serPort, baudRate)
        bluetooth.flushInput()
        if bluetooth.isOpen():
            print(bluetooth.name + " is open...")
              print "Serial port " + serPort + " opened at Baudrate " +
str(baudRate)

        startMarker = 60
        endMarker = 62

        # Wait for arduino to start the program
        waitForArduino()
```

Αρχικά ορίζεται η σειριακή θύρα από την οποία θα γίνει η μετάδοση των δεδομένων, και ο ρυθμός μετάδοσης στα 9600 bps (bits per second). Η τιμή αυτή είναι η προεπιλεγμένη του Bluetooth module, και πρέπει αυτές οι δυο να συμφωνούν. Όπως αναφέρθηκε σε προηγούμενη ενότητα, υπάρχει η δυνατότητα να γίνει αλλαγή της τιμής αυτής στο Bluetooth module, αλλά για την παρούσα εργασία, η τιμή αυτή είναι πολύ ικανοποιητική για τον όγκο των δεδομένων που πρέπει να αποσταλούν.

Στη συνέχεια ορίζεται μια αναφορά στη σειριακή επικοινωνία με το όνομα bluetooth,και γίνεται διαγραφή των δεδομένων που μπορεί να υπάρχουν στο buffer εισόδου.

Τελικά το πρόγραμμα φτάνει στην υπορουτίνα *waitForArduino()*. Στην υπορουτίνα αυτή γίνεται αναμονή έως ότου το Arduino στείλει ένα σήμα ότι είναι έτοιμο για λειτουργία. Πιο συγκεκριμένα, το πρόγραμμα θα συνεχίσει τη λειτουργία του, όταν λάβει από το Arduino τη φράση <Arduino is ready>. Η υπορουτίνα αυτή πρακτικά βοηθάει στο να γίνει ο συγχρονισμός των δυο συσκευών.

Το τελευταίο σημείο του κώδικα, είναι η κλήση της υπορουτίνας *runTest()* με όρισμα τη συμβολοσειρά *data* που περιέχει τις γωνίες που θα πρέπει να κινηθούν οι σερβοκινητήρες.

```
        runTest(data)
```

1.14. Υπορουτίνες που χρησιμοποιούνται στον κώδικα Python.

- ***mymap(xin)*:** κάνει αντιστοίχιση μιας απόστασης σε μια γωνία στο διάστημα $0^{\circ} - 180^{\circ}$

```
        def mymap(xin):


            # maps finger distances from center mass to angles
             return int((xin-in_min) * (out_max-out_min) / (in_max-in_min) +
out_min)
```

- ***sendToArduino(sendstr)*:** Γράφει στη σειριακή θύρα τα δεδομένα που πρόκειται να αποσταλούν. Στη συνέχεια αυτά αποστέλλονται στο Arduino μέσω Bluetooth.

```
        def sendToArduino(sendstr):


            # sends data to Arduino
            bluetooth.write(sendstr)
```

- ***recvFromArduino()*:** Λαμβάνει δεδομένα από το Arduino μέσω Bluetooth και τα αποκωδικοποιεί. Τα πακέτα δεδομένων που στέλνει το Arduino έχουν

τη μορφή που προαναφέρθηκε σε προηγούμενη ενότητα, με αρχή πακέτου το σύμβολο ‘<’ και τέλος πακέτου το σύμβολο ’>’.

```
def recvFromArduino():

    # receives data from Arduino and returns it to Python
    global startMarker, endMarker


    ck = ""
    xk = "z"  # any value that is not an end- or startMarker
    bytecount = -1  # to begin saving from byte 0


    # wait for the start character
    while ord(xk) != startMarker:
        xk = bluetooth.read()


    # save data until the end marker is found
    while ord(xk) != endMarker:
        if ord(xk) != startMarker:
            ck = ck + xk
            bytecount += 1
        xk = bluetooth.read()
    return ck
```

- ***waitForArduino()*:** Περιμένει μέχρι να λάβει από το Arduino το μήνυμα ‘Arduino is ready’. Η ρουτίνα χρησιμοποιείται μόνο στην εκκίνηση του προγράμματος, και ουσιαστικά κάνει παύση σε αυτό μέχρι το Arduino να είναι έτοιμο προς λειτουργία.

```
def waitForArduino():


    # wait until Arduino sends 'Arduino Ready'


    global startMarker, endMarker


    msg = ""
    while msg.find("Arduino is ready") == -1:
        while bluetooth.inWaiting() == 0:
            pass
        msg = recvFromArduino()
        print msg
```

```
            print
```

- ***runTest(td):*** καλεί την υπορουτίνα *sendToArduino(td)* για αποστολή δεδομένων και κάνει μια παύση 100ms.

```
    def runTest(td):

        # sends data to Arduino and pauses for 100 ms

        sendToArduino(td)
        #print "Sent from RPi -- DATA " + td

        time.sleep(0.1)
```

- ***angle(v1,v2):*** υπολογίζει τη γωνία μεταξύ δυο διανυσμάτων χρησιμοποιώντας τον τύπο του εσωτερικού γινομένου διανυσμάτων.

```
    def angle(v1, v2):

        # returns the angle between 2 arrays
        dot = np.dot(v1, v2)
        x_modulus = np.sqrt(v1[0]*v1[0]+v1[1]*v1[1])  # ((v1*v1).sum())
        y_modulus = np.sqrt(v2[0]*v2[0]+v2[1]*v2[1])  # ((v2*v2).sum())
        cos_angle = dot / x_modulus / y_modulus
        anglee = np.degrees(np.arccos(cos_angle))
        return anglee
```

- ***distance(a1,a2):*** υπολογίζει την απόσταση μεταξύ δυο σημείων.

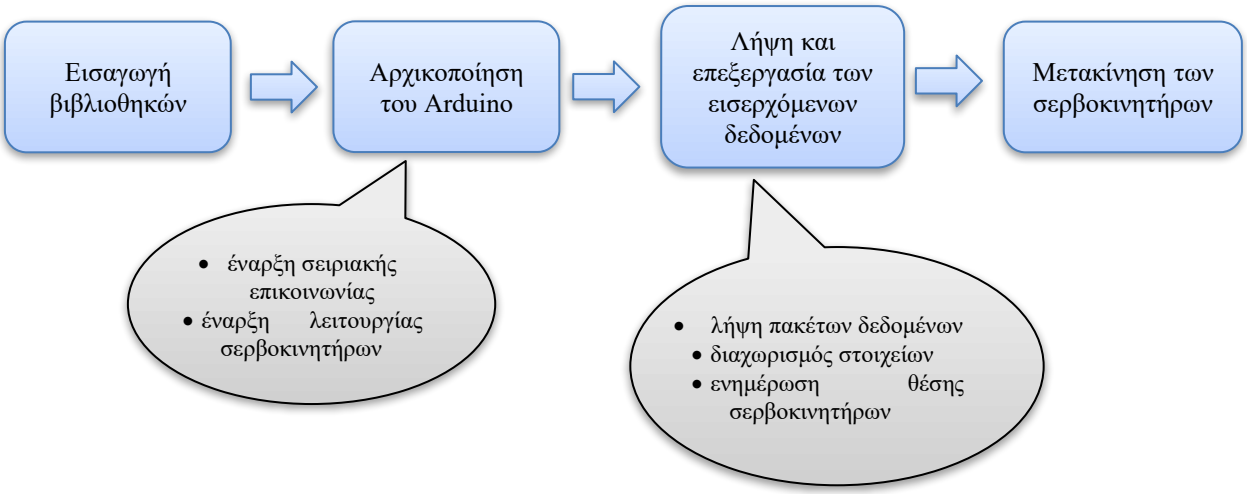
```
    def distance(a1, a2):

      # returns the distance between two points
       z  =  np.sqrt(np.power(a2[1][0]-a1[1][0],  2)+np.power(a2[0][0]-a1[0]
[0], 2))
      return z
```

Ο κώδικας του Raspberry Pi καθώς και οι υπορουτίνες του δίνονται στο παράρτημα Α

5. Περιγραφή λειτουργιών του Arduino.

1.15. Κώδικας που χρησιμοποιείται στο Arduino.

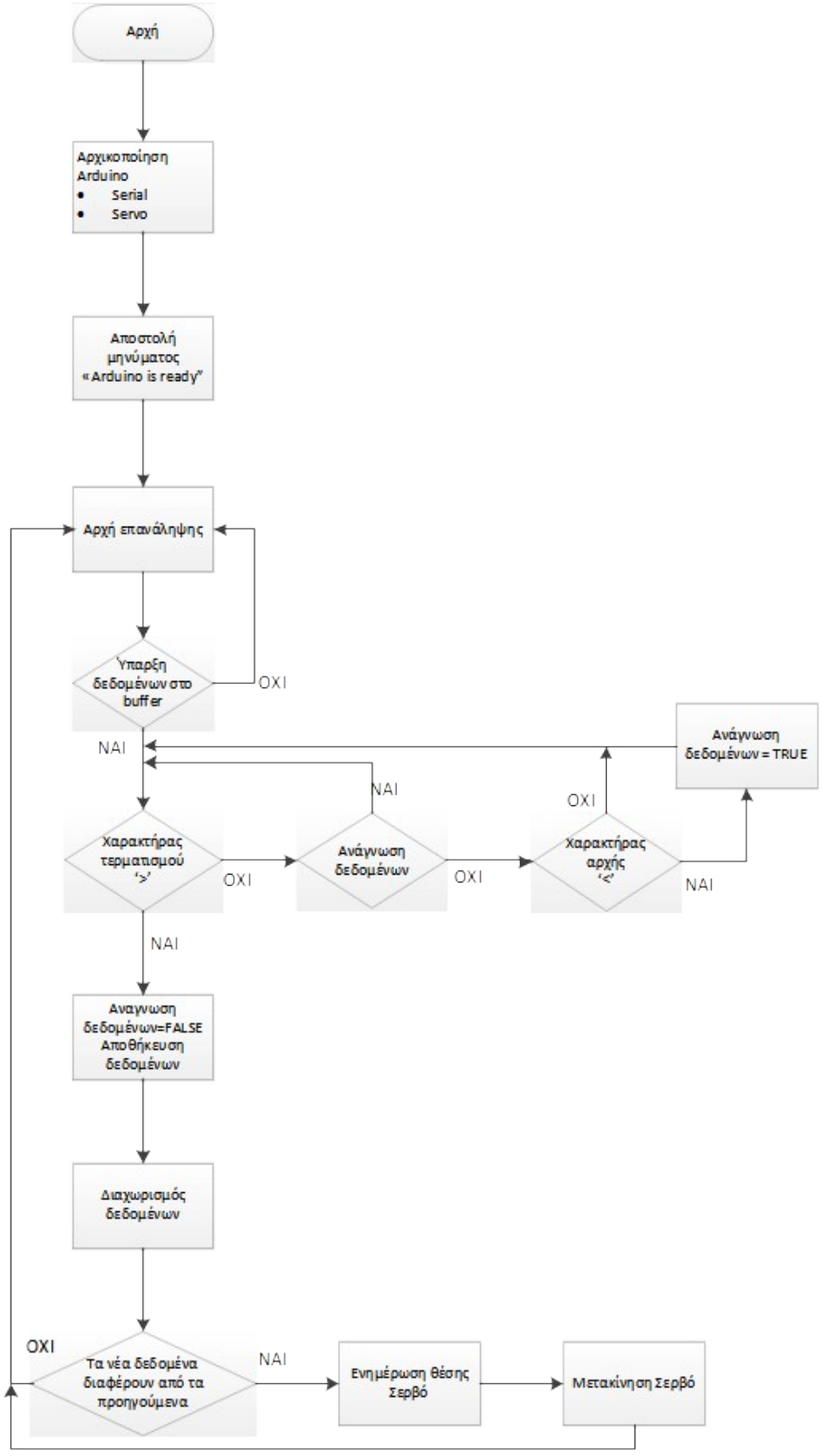
Οι λειτουργίες που εκτελεί το Arduino είναι λιγότερες και αρκετά πιο απλές από αυτές που εκτελούνται στο Raspberry Pi. Συνοπτικά περιγράφονται στο παρακάτω διάγραμμα.



Σχήμα 5. Διάγραμμα λειτουργιών του Arduino

Κάποιες από τις λειτουργίες του δεύτερου, όπως για παράδειγμα ο υπολογισμός των αποστάσεων των ακροδαχτύλων από το κέντρο βάρους και η μετατροπή τους σε γωνίες, θα μπορούσαν να ανατεθούν στο Arduino. Επειδή όμως η ταχύτητα επεξεργασίας δεδομένων και η εσωτερική μνήμη στο Raspberry Pi είναι αρκετά μεγαλύτερες από ότι αυτών του Arduino, προτιμήθηκε οι λειτουργίες αυτές να ανατεθούν στο πρώτο. Ένας ακόμη λόγος για τον οποίο επιλέχθηκε η ανάθεση αυτή, είναι επειδή το Arduino είναι προτιμότερο να παραμείνει ένας απλός ελεγκτής των σερβοκινητήρων, ή όποιων συσκευών κατάλληλων μπορούν να συνδεθούν στις εξόδους του, χωρίς να γίνεται σύνθετη η δομή του προγραμματισμού και των παραμετροποιήσεων. Στο Raspberry Pi αντίθετα, που είναι και η φορητή συσκευή, μπορούν να γίνουν πιο εύκολα οι όποιες παραμετροποιήσεις, οι οποίες μελλοντικά θα μπορούσαν να γίνουν και μέσω γραφικού περιβάλλοντος, αφού διαθέτει εξόδους για οθόνη και πληκτρολόγιο.

1.16. Διάγραμμα ροής του κώδικα στο Arduino.



Σχήμα 6. Διάγραμμα ροής του κώδικα στο Arduino

1.17. Ανάλυση του κώδικα

Το Arduino χρησιμοποιεί μια μόνο εξωτερική βιβλιοθήκη, την *Servo.h,* που χρησιμοποιείται για τον έλεγχο των σερβοκινητήρων.

```
// Arduino Mega 2560


#include <Servo.h>                          //Servo
```

Το πρόγραμμα ξεκινάει εισάγοντας αυτή την βιβλιοθήκη.

Στη συνέχεια γίνεται η δήλωση όλων των μεταβλητών και των παραμέτρων προγράμματος.

```
// 5 Servo objects
Servo myServo[5];

// 5 servo pin attachments to Arduino
byte servoPin[5] = {2, 3, 4, 5, 6};


// Max-Min servo position to protect them
byte servoMin = 10;
byte servoMax = 170;


// initial servo positions
byte  newServoPos[5]  =  {servoMin,  servoMin,  servoMin,  servoMin,
servoMin};
byte servoPos[5] = {2, 2, 2, 2, 2};


const byte buffSize = 40;
char inputBuffer[buffSize];
const char startMarker = '<';
const char endMarker = '>';
byte bytesRecvd = 0;
boolean readInProgress = false;
boolean newDataFromRPi = false;
```

Ακολουθεί η υπορουτίνα *setup()*. Σε αυτή γίνεται η αρχικοποίηση του Arduino και τρέχει μια φορά μόνο στην αρχή του προγράμματος. Αρχικά ενεργοποιείται η σειριακή επικοινωνία του Arduino στα 9600 bps για να συμφωνεί με το Baudrate του Bluetooth module. Στη συνέχεια δηλώνονται τα pins του Arduino και

η χρήση τους ως PWM έξοδοι για την οδήγηση σερβοκινητήρων. Αυτοί μετακινούνται στην αρχική τους θέση. Τελικά αφού γίνει η αρχικοποίηση όλων των υποσυστημάτων του Arduino, αποστέλλεται στο Raspberry Pi μέσω Bluetooth το μήνυμα *'Arduino is ready'* για να ξεκινήσει η διαδικασία ανάγνωσης και επεξεργασίας εικόνων.

```
void setup() {


  //initialize bluetooth
  Serial1.begin(9600);
  //initialize serial to PC
  Serial.begin(9600);

  // initialize the servo
  for(byte i = 0; i < 5; i++){                    //  Servo
    myServo[i].attach(servoPin[i]);               //  Servo
  }

  // move servo to initial angle
  moveServo();                                     //  Servo

  // tell Raspberry Pi we are ready
  Serial1.println("<Arduino is ready>");
   // tell the PC we are ready
  Serial.println("<Arduino is ready>");
}
```

Στον παραπάνω κώδικα γίνεται αρχικοποίηση της σειριακής επικοινωνίας με τον υπολογιστή και η αποστολή του ίδιου μηνύματος *'Arduino is ready'*. Αυτό δεν είναι μέρος της λειτουργίας του συστήματος και έχει γίνει καθαρά για λόγους αποσφαλμάτωσης.

Το επόμενο βήμα είναι η υπορουτίνα *loop()* . Είναι το κυρίως πρόγραμμα που τρέχει το Arduino και επαναλαμβάνεται συνεχώς. Αποτελείται μόνο από τέσσερις υπορουτίνες [15].

```
void loop() {


  getDataFromRPi();
  updateServoPos();
```

```
        moveServo();
        //reply();
        //replyToRPi();
        delay(10);
      }
```

- *getDataFromRPi()*. Λαμβάνει δεδομένα από το buffer του Bluetooth

Γίνεται έλεγχος στο buffer και αν υπάρχουν bytes δεδομένων τότε γίνεται η λήψη τους. Τα πακέτα δεδομένων, όπως προαναφέρθηκε, στέλνονται στη μορφή <a1,a2,a3,a4,a5>. Ελέγχεται αν το byte που λαμβάνει, είναι το σύμβολο αρχής '<' ή τέλους '>' πακέτου δεδομένων και αποθηκεύονται μόνο όλα τα ενδιάμεσα bytes.

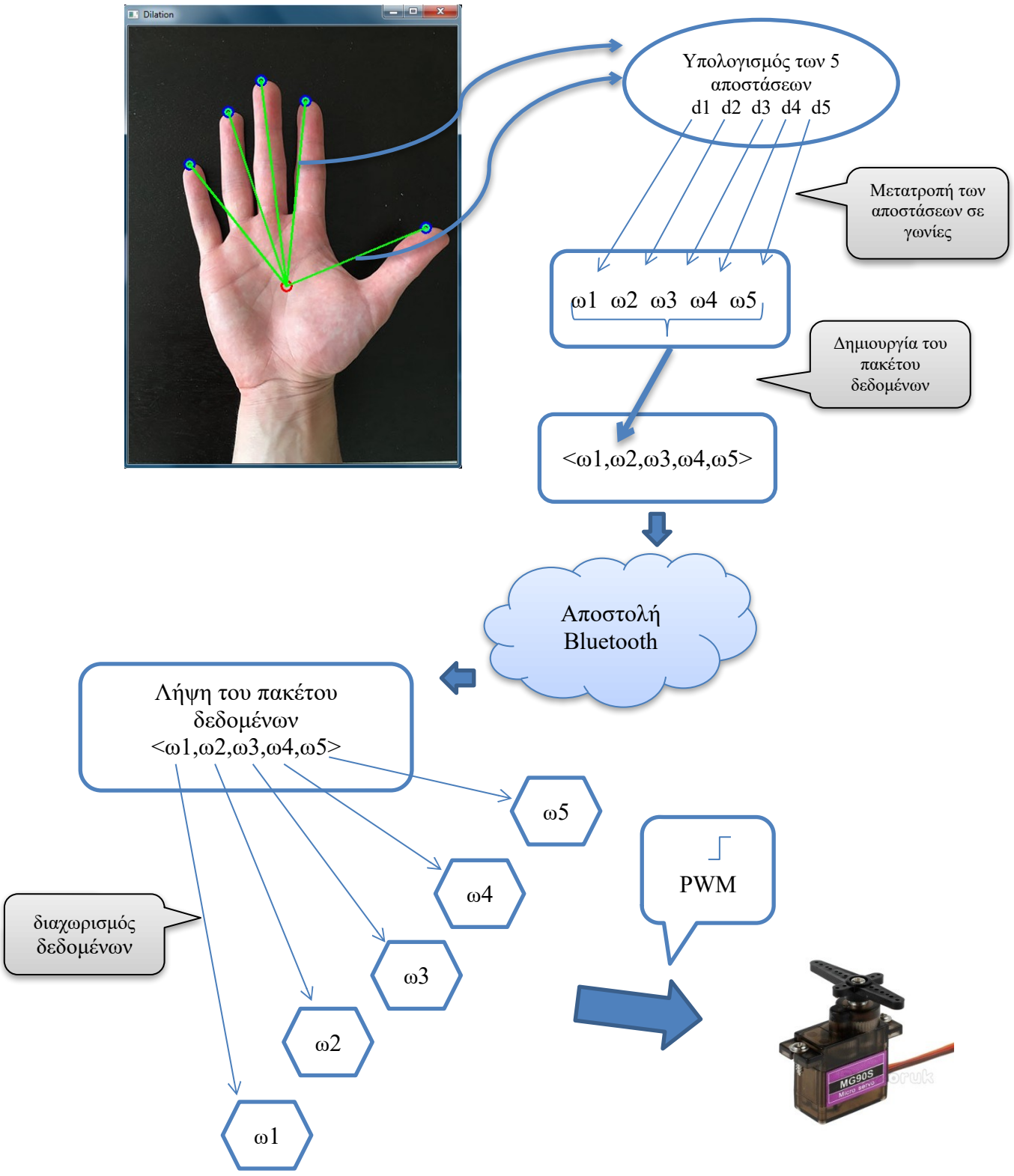
Η υπορουτίνα *parseData()* διαχωρίζει τα αποθηκευμένα δεδομένα με κριτήριο το ',' και μετατρέπει τις πέντε συμβολοσειρές που προκύπτουν σε ακέραιους αριθμούς. Αυτές είναι οι πέντε γωνίες που θα πρέπει να μετακινηθούν οι σερβοκινητήρες.

- *updateServoPos()*. Ελέγχει αν οι νέες θέσεις διαφέρουν από τις προηγούμενες και τις ανανεώνει.
- *moveServo()*. Μετακινεί τους σερβοκινητήρες στις ανανεωμένες θέσεις
- *delay(10)*. Κάνει παύση 10ms

Οι ρουτίνες `reply()`, `replyToRPi()` που βρίσκονται σε σχόλιο, αποστέλλουν τα δεδομένα στον υπολογιστή και στο Raspberry Pi, και χρησιμοποιούνται μόνο για την αποσφαλμάτωση.

Ο κώδικας του Arduino καθώς και οι υπορουτίνες του δίνονται στο παράρτημα Α

Στο παρακάτω σχήμα φαίνεται η διαδικασία δημιουργίας, αποστολής και αποκωδικοποίησης των πακέτων δεδομένων.



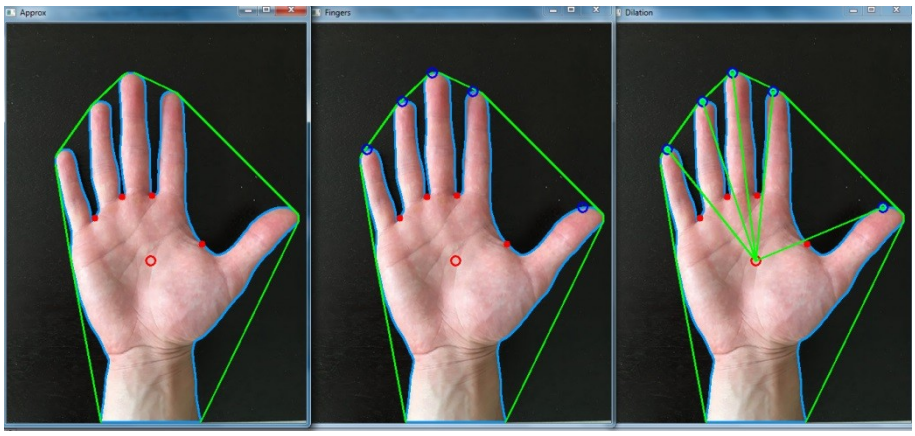
Σχήμα 7. Δημιουργία, αποστολή και αποκωδικοποίηση των πακέτων δεδομένων

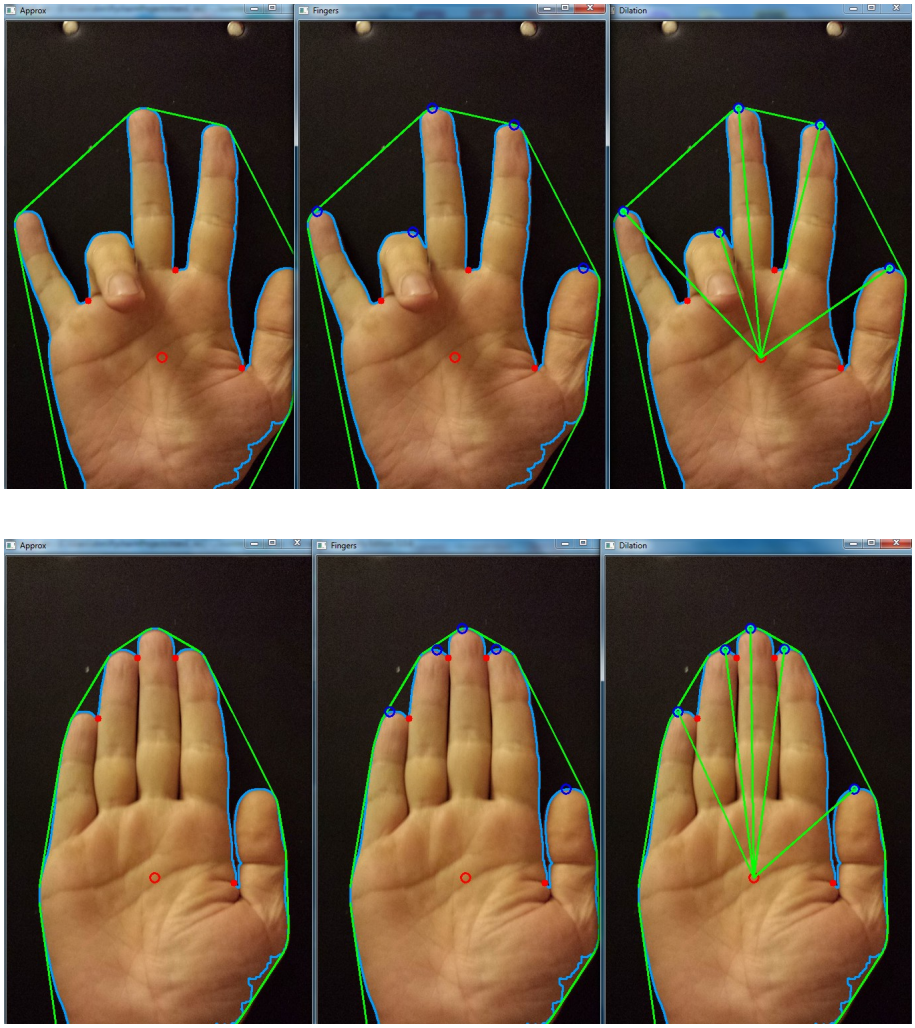
6. Αποτελέσματα και συμπεράσματα

Είναι γενικά δύσκολο να γίνει η αποτύπωση της πειραματικής λειτουργίας του συστήματος σε γραπτό κείμενο. Ακόμα και οι εικόνες αποτυπώνουν μια στιγμή της λειτουργίας ενός συστήματος που δουλεύει σε πραγματικό χρόνο. Θα γίνει λοιπόν μια σύντομη περιγραφή των αποτελεσμάτων της λειτουργίας.

Το πρώτο σκέλος της εφαρμογής που έχει να κάνει με την αναγνώριση των χαρακτηριστικών της εικόνας και την εξαγωγή των δεδομένων από το Raspberry Pi, λειτουργεί πολύ ικανοποιητικά, ως προς τις απαιτήσεις και τις παραδοχές που έχουν οριστεί. Όπως έχει προαναφερθεί, η πειραματική λειτουργία γίνεται σε ένα μονόχρωμο background με πολύ καλά αποτελέσματα. Μετά από δοκιμή λειτουργίας σε ένα πιο σύνθετο background (που δεν είναι στο πλαίσιο της παρούσας εργασίας και αφήνεται για μελλοντική έρευνα) διαπιστώθηκε ότι μπορεί να υπάρχουν αποκλίσεις στην αναγνώριση των χαρακτηριστικών ενδιαφέροντος.

Παρακάτω φαίνονται στιγμιότυπα της αναγνώρισης του χεριού σε διάφορες κινήσεις



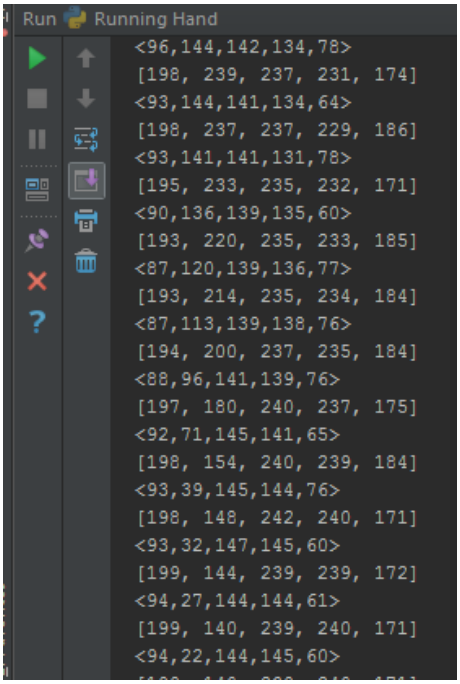


Εικόνα 37. Στιγμιότυπα αναγνώρισης

Παρατηρήθηκε επίσης η σχετικά μικρή καθυστέρηση της επεξεργασίας εικόνας από το Raspberry Pi. Η χρήση του Raspberry Pi 3, που όπως προαναφέρθηκε διαθέτει ταχύτερο επεξεργαστή, και ίσως μικρές βελτιώσεις του κώδικα Python, μπορούν να αποτελέσουν λύσεις για το γεγονός αυτό.

Το δεύτερο σκέλος που είναι η επικοινωνία μέσω Bluetooth του Raspberry Pi με το Arduino, έδωσε άριστα αποτελέσματα. Η ζεύξη των δυο συσκευών γίνεται χωρίς κανένα πρόβλημα και η ταχύτητα της επικοινωνίας είναι μεγάλη. Επίσης το πρωτόκολλο επικοινωνίας που δημιουργήθηκε για την αποστολή των πακέτων δεδομένων, λειτούργησε με απόλυτη ακρίβεια χωρίς να υπάρχει απώλεια πακέτων ή λανθασμένα δεδομένα.

Στην παρακάτω εικόνα φαίνεται ένα στιγμιότυπο των αποστάσεων που υπολογίστηκαν και των πακέτων γωνιών που αποστέλλονται μέσω Bluetooth.



Εικόνα 38. Αποστάσεις που υπολογίζονται και πακέτα γωνιών που αποστέλλονται

Τα αποτελέσματα λειτουργίας του τρίτου σκέλους κρίνονται επίσης άριστα. Στο σκέλος αυτό περιλαμβάνεται η αποκωδικοποίηση των δεδομένων από το Arduino και η οδήγηση μέσω PWM των σερβοκινητήρων. Παρατηρήθηκε μια σχετικά συνεχόμενη μικρή κίνηση των σερβοκινητήρων σε περιπτώσεις “ακινησίας” των δαχτύλων. Αυτό διαπιστώθηκε ότι οφείλεται στη μεγάλη ευαισθησία του συστήματος. Παρόλο που η παλάμη και τα δάχτυλα φαινομενικά μπορεί να είναι ακίνητα μπροστά στην κάμερα, στην πραγματικότητα υπάρχουν μικρές μετατοπίσεις που γίνονται αντιληπτές από το σύστημα και μεταφράζονται σε κινήσεις των σερβοκινητήρων. Θα μπορούσε να πει κανείς λοιπόν ότι ίσως να χρειάζεται μείωση αυτής της μεγάλης ευαισθησίας.

Σε γενικές γραμμές μπορούμε να πούμε ότι το σύστημα λειτουργεί πολύ αποτελεσματικά. Πραγματοποιήθηκαν αρκετοί έλεγχοι ως προς την απόδοση και την ταύτιση της εισόδου (εικόνας) και εξόδου (γωνίες σερβοκινητήρων). Τέλος οι κινήσεις των σερβομηχανισμών αντικατοπτρίζουν σε πραγματικό χρόνο στις κινήσεις των δαχτύλων.

7. Μελλοντική έρευνα και βελτιώσεις.

Η παρούσα εργασία βασίστηκε στην κύρια δομή της μηχανικής όρασης. Στην αναγνώριση ενός αντικειμένου από μια εικόνα και επιλεγμένων χαρακτηριστικών του, και στην χρήση των δεδομένων εξόδου για τον έλεγχο μιας διεργασίας. Για την πραγματοποίησή της εφαρμόστηκαν κάποιες παραδοχές. Η σημαντικότερη ήταν η χρήση του μονόχρωμου background (λευκό ή μαύρο) της εικόνας προς αναγνώριση. Υπάρχουν μέθοδοι και λύσεις για την εξαγωγή χρήσιμων δεδομένων από εικόνα με οποιοδήποτε background, όπως η μέθοδος background extraction με την οποία γίνεται απαλοιφή του background και δέσμευση μόνο του αντικειμένου ενδιαφέροντος, ή η μέθοδος skin detection, στην οποία δεσμεύεται από την εικόνα μόνο το χρώμα του δέρματος και γίνεται απαλοιφή των υπολοίπων. Παρόλο που έγινε ένας απλός πειραματισμός με τις μεθόδους αυτές, τελικά δεν χρησιμοποιήθηκαν γιατί οδηγούσαν περισσότερο σε αλγοριθμική βελτιστοποίηση της επεξεργασίας εικόνας μέσω της OpenCV, και ξέφευγαν από τον κύριο στόχο της εφαρμογής. Δεν παύουν να αποτελούν όμως το επόμενο βήμα για την εξέλιξη και βελτιστοποίηση της παρούσας εργασίας.

Ένα άλλο θέμα το οποίο αποτελεί αντικείμενο προς περαιτέρω ανάπτυξη, είναι η δημιουργία βιβλιοθηκών εντολών. Υπάρχει η δυνατότητα μέσω προγραμματισμού, ή εκμάθησης μέσω νευρωνικών δικτύων, της δημιουργίας κωδικοποιημένων εντολών, οι οποίες μπορούν να αποτελέσουν έξοδο για την οδήγηση μιας πληθώρας άλλων εφαρμογών, όπως ένας ρομποτικός βραχίονας, ένα drone ή ενός τηλεκατευθυνόμενου αυτοκινήτου, που χρειάζονται βασικές εντολές οδήγησης ή πλοήγησης.

Τέλος οι βαθμοί ελευθερίας του συστήματος που υλοποιήθηκε στην παραπάνω εργασία, είναι συνολικά εννιά. Από αυτούς έγινε η χρήση μόνο των πέντε, αφήνοντας πολλά περιθώρια για την υλοποίηση σε συστήματα με περισσότερους βαθμούς ελευθερίας.

Βιβλιογραφία

[1] [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Machine_vision. [Πρόσβαση 3 10 2017].

[2] [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Raspberry_Pi. [Πρόσβαση 29 9 2017].

[3] [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/OpenCV. [Πρόσβαση 29 9 2017].

[4] [Ηλεκτρονικό]. Available: https://el.wikipedia.org/wiki/Arduino. [Πρόσβαση 29 9 2017].

[5] [Ηλεκτρονικό]. Available: http://www.geeetech.com/wiki/index.php/Raspberry_Pi_Model_B%2B. [Πρόσβαση 24 9 2017].

[6] [Ηλεκτρονικό]. Available: http://datasheet.octopart.com/A000066-Arduino-datasheet-38879526.pdf. [Πρόσβαση 24 9 2017].

[7] [Ηλεκτρονικό]. Available: https://arduino-info.wikispaces.com/BlueTooth-HC05-HC06-Modules-How-To. [Πρόσβαση 17 9 2017].

[8] [Ηλεκτρονικό]. Available: http://yourduino.com/docs/CSR-BC417-datasheet.pdf. [Πρόσβαση 17 9 2017].

[9] [Ηλεκτρονικό]. Available: http://www.fecegypt.com/uploads/dataSheet/1480849570_hc06.pdf. [Πρόσβαση 17 9 2017].

[10] [Ηλεκτρονικό]. Available: https://www.tutorialspoint.com/opencv/opencv_gaussian_blur.htm. [Πρόσβαση 20 9 2017].

[11] [Ηλεκτρονικό]. Available: http://docs.opencv.org/2.4/doc/tutorials/imgproc/erosion_dilatation/erosion_dilatation.html. [Πρόσβαση 20 9 2017].

[12] [Ηλεκτρονικό]. Available: http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_imgproc/py_contours/py_contours_more_functions/py_contours_more_functions.html. [Πρόσβαση 22 9 2017].

[13] [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Ramer%E2%80%93Douglas%E2%80%93Peucker_algorithm. [Πρόσβαση 22 9 2017].

[14] [Ηλεκτρονικό]. Available: http://docs.opencv.org/3.1.0/dd/d49/tutorial_py_contour_features.html. [Πρόσβαση 22 9 2017].

[15] [Ηλεκτρονικό]. Available: http://forum.arduino.cc/index.php?topic=225329.msg1810764#msg1810764. [Πρόσβαση 23 9 2017].

ΠΑΡΑΡΤΗΜΑ Α

Κώδικας του Raspberry Pi

```
from picamera.array import PiRGBArray
from picamera import PiCamera

import cv2
import numpy as np
import math
import serial
import time



# ************************************************************
# VIDEO  ************************************************************
#cap = cv2.VideoCapture('video3.mp4')
#while not cap.isOpened():
#    cap = cv2.VideoCapture("video3.mp4")
#    cv2.waitKey(1000)
#    print "Wait for the header"

# Decrease frame size
#cap.set(cv2.CAP_PROP_FRAME_WIDTH, 480)
#cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 360)



# CAMERA  ************************************************************
# cap = cv2.VideoCapture(0)
eps = 0.0005
disx = 5
disy = 10
loops = 0
counter = 0
bypass = 1
datalist =[0,0,0,0,0]
in_min = 130
in_max = 260
out_min = 10
out_max = 170
s1 = 0
s2 = 0
s3 = 0
s4 = 0
s5 = 0
# image = "hand2.jpg"

# FUNCTION DEFINITIONS
```

```
#*****************************************************************************

def mymap(xin):
   return int((xin-in_min) * (out_max-out_min) / (in_max-in_min) + out_min)



def sendToArduino(sendstr):
   bluetooth.write(sendstr)



def recvFromArduino():
   global startMarker, endMarker

   ck = ""
   xk = "z"  # any value that is not an end- or startMarker
   bytecount = -1  # to allow for the fact that the last increment will be one too many

   # wait for the start character
   while ord(xk) != startMarker:
      xk = bluetooth.read()

   # save data until the end marker is found
   while ord(xk) != endMarker:
      if ord(xk) != startMarker:
         ck = ck + xk
         bytecount += 1
      xk = bluetooth.read()
   return ck



def waitForArduino():

   # wait until the Arduino sends 'Arduino Ready' - allows time for Arduino reset
   # it also ensures that any bytes left over from a previous message are discarded

   global startMarker, endMarker

   msg = ""
   while msg.find("Arduino is ready") == -1:
      while bluetooth.inWaiting() == 0:
         pass
      msg = recvFromArduino()
      print msg
      print



def runTest(td):
```

```
    #n = 0
    sendToArduino(td)
    #print "Sent from RPi -- DATA " + td
    #n += 1
    time.sleep(0.01)


def angle(v1, v2):
    dot = np.dot(v1, v2)
    x_modulus = np.sqrt(v1[0]*v1[0]+v1[1]*v1[1])  # ((v1*v1).sum())
    y_modulus = np.sqrt(v2[0]*v2[0]+v2[1]*v2[1])  # ((v2*v2).sum())
    cos_angle = dot / x_modulus / y_modulus
    anglee = np.degrees(np.arccos(cos_angle))
    return anglee


def distance(a1, a2):
    z = np.sqrt(np.power(a2[1][0]-a1[1][0], 2)+np.power(a2[0][0]-a1[0][0], 2))
    return z

# ----------------------------------------------------------------------------------------------------------

# SERIAL PORT COMMUNICATION
#*******************************************************************************

# NOTE the user must ensure that the serial port and baudrate are correct
serPort = '/dev/rfcomm0' #'COM6'
baudRate = 9600
bluetooth = serial.Serial(serPort, baudRate)
bluetooth.flushInput()
if bluetooth.isOpen():
    print(bluetooth.name + " is open...")
    print "Serial port " + serPort + " opened  Baudrate " + str(baudRate)


startMarker = 60
endMarker = 62

# Wait for arduino to start the programm
waitForArduino()
# ----------------------------------------------------------------------------------------------------------



#*********************************************************
#  PiCamera
```

```
# initialize the camera and grab a reference to the raw camera capture
camera = PiCamera()
camera.resolution = (640, 480)
camera.framerate = 32
rawCapture = PiRGBArray(camera, size=(640, 480))

# allow the camera to warmup
time.sleep(0.1)

# capture frames from the camera
for frame1 in camera.capture_continuous(rawCapture, format="bgr", use_video_port=True):
        # grab the raw NumPy array representing the image, then initialize the timestamp
        # and occupied/unoccupied text
   frame = frame1.array




# Open Camera or Video and start reading Images
#*********************************************************************************

#while cap.isOpened():
#    ret, frame1 = cap.read()
#    frame = cv2.resize(frame1, (480, 360))
   # frame = cv2.imread(image)
   # height, width, depth = frame.shape
   # print height, width, depth

   gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
   blurred = cv2.GaussianBlur(gray, (35, 35), 0)

      _, thresh1 = cv2.threshold(blurred, 60, 255, cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)   #
BINARY_INV Has problems
   # BINARY_INV -> White shape into black backround. This is what OpenCV recognizes.



   #  gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
   # blurred = cv2.GaussianBlur(gray, (5, 5), 0)
   #
    # _, thresh1 = cv2.threshold(blurred, 60, 255, cv2.THRESH_BINARY)  # BINARY_INV Has problems
+cv2.THRESH_OTSU
   # # BINARY_INV -> White shape into black backround. This is what OpenCV recognizes.



   thresh1 = cv2.erode(thresh1, None, iterations=2)
   thresh1 = cv2.erode(thresh1, None, iterations=2)
   thresh1 = cv2.dilate(thresh1, None, iterations=2)
   thresh1 = cv2.dilate(thresh1, None, iterations=2)
```

```
    # cv2.imshow('Blurred', blurred)
    #cv2.imshow('Thresh', thresh1)
    # cv2.imshow('CORNERS', gray)

    # _, contours, hierarchy = cv2.findContours(thresh1, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
    _, contours, hierarchy = cv2.findContours(thresh1, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

    if len(contours) != 0:
        # print contours
        max_area = -1
        ci = 0
        for i in range(len(contours)):
            cnt = contours[i]
            area = cv2.contourArea(cnt)
            if area > max_area:
                max_area = area
                ci = i
                # Largest area contour
        cnts = contours[ci]

        moments = cv2.moments(cnts)

        # Central mass of first order moments
        cx = 1
        cy = 1
        if moments['m00'] != 0:
            cx = int(moments['m10']/moments['m00'])  # cx = M10/M00
            cy = int(moments['m01']/moments['m00'])  # cy = M01/M00
        centerMass = (cx, cy)

        # Draw center mass
        cv2.circle(frame, centerMass, 7, [0, 0, 255], 2)
#
*********************************************************************************

        # Find convex hull
        hull = cv2.convexHull(cnts)

        # Find convex defects
        hull2 = cv2.convexHull(cnts, returnPoints=False)
        defects = cv2.convexityDefects(cnts, hull2)

        # Get defect points and draw them in the original image
        FarDefect = []
        count_defects = 0
        for i in range(defects.shape[0]):
            s, e, f, d = defects[i, 0]
            start = tuple(cnts[s][0])
```

```
        end = tuple(cnts[e][0])
        far = tuple(cnts[f][0])
        FarDefect.append(far)
        cv2.line(frame, start, end, [0, 255, 0], 1)
        cv2.circle(frame, far, 7, [100, 255, 255], 2)

        a = math.sqrt((end[0] - start[0])**2 + (end[1] - start[1])**2)
        b = math.sqrt((far[0] - start[0])**2 + (far[1] - start[1])**2)
        c = math.sqrt((end[0] - far[0])**2 + (end[1] - far[1])**2)
        angle1 = math.acos((b**2 + c**2 - a**2)/(2*b*c)) * 57
        if angle1 <= 120:
            count_defects += 1
            cv2.circle(frame, far, 5, [0, 0, 255], -1)

#*************************************************************************************
    epsilon = eps*cv2.arcLength(cnts, True)
    approx = cnts #cv2.approxPolyDP(cnts, epsilon, closed=True)         # counterclockwise
#*************************************************************************************
    cv2.drawContours(frame, [approx], -1, (0, 255, 0), 1)
    # cv2.drawContours(frame, cnts, -1, (0,255 , 255), 1)
    # print epsilon
    # print len(approx)
    # print len(cnts)



    # LOCAL MAXIMUMS
     #*************************************************************************************
****
    maximum = []
    if approx[0][0][1] <= approx[len(approx)-1][0][1] and approx[0][0][1] < approx[1][0][1]:
        if approx[0][0][1] <= approx[len(approx)-2][0][1] and approx[0][0][1] < approx[2][0][1]:
            maximum.append(approx[0][0])
    if approx[1][0][1] <= approx[0][0][1] and approx[1][0][1] < approx[2][0][1]:
        if approx[1][0][1] <= approx[len(approx)-1][0][1] and approx[0][0][1] < approx[3][0][1]:
            maximum.append(approx[1][0])


    for i in range(2, len(approx)-3):
        if approx[i][0][1] <= approx[i-1][0][1] and approx[i][0][1] < approx[i+1][0][1]:
            if approx[i][0][1] <= approx[i-2][0][1] and approx[i][0][1] < approx[i+2][0][1]:
                maximum.append(approx[i][0])


    if approx[len(approx)-2][0][1] <= approx[len(approx)-1][0][1] and\
                approx[len(approx)-2][0][1] < approx[len(approx)-3][0][1]:
        if approx[len(approx)-2][0][1] <= approx[0][0][1] and\
                    approx[len(approx)-2][0][1] < approx[len(approx)-4][0][1]:
            maximum.append(approx[len(approx)-2][0])
    # if approx[len(approx)-1][0][1] <= approx[0][0][1] and\
    #               approx[len(approx)-1][0][1]< approx[len(approx)-2][0][1]:
    #     if approx[len(approx)-1][0][1] <= approx[1][0][1] and\
```

```
    #                 approx[len(approx)-1][0][1]< approx[len(approx)-3][0][1]:
    #       maximum.append(approx[len(approx)-1][0])

    # Ignore very close local maximums
    maximumfin = []
    # print maximum
    for i in range(0, len(maximum)-1):
            if (np.absolute(maximum[i][0]-maximum[i+1][0]) < 10) and (np.absolute(maximum[i][1]-
maximum[i+1][1]) < 10):
          if maximum[i][1] < maximum[i+1][1]:
            maximumfin.append(maximum[i])
          else:
            maximumfin.append(maximum[i+1])
        else:
          maximumfin.append(maximum[i])

    if (np.absolute(maximum[len(maximum)-1][0]-maximum[0][0]) < disx) and\
          (np.absolute(maximum[len(maximum)-1][1]-maximum[0][1]) < disy):
        if maximum[len(maximum)-1][1] < maximum[0][1]:
          maximumfin.append(maximum[len(maximum)-1])
        else:
          maximumfin.append(maximum[0])
    else:
        maximumfin.append(maximum[len(maximum)-1])

    maximum = maximumfin

    for i in range(0, len(maximum)):
        cv2.circle(frame, tuple(maximum[i]), 10, [50, 150, 30], 2)
    # --------------------------------------------------------------------------------------------------

    # Distance of each finger tip to the center mass
                                                                                                    #
**************************************************************************************
**********************
    fingersx = sorted(maximum, key=lambda xp: xp[0])  # ,reverse=True)
    fingerDistance = []
    for i in range(0, len(fingersx)):
        cordfing = tuple(maximum[i])
        cv2.circle(frame, cordfing, 4, [100, 255, 0], 1)              # Laxani
        cv2.line(frame, cordfing, centerMass, [0, 255, 0], 1)
            distance1 = int(np.sqrt(np.power(fingersx[i][0]-centerMass[0], 2)+np.power(fingersx[i][1]-
centerMass[1], 2)))
        fingerDistance.append(distance1)
    # print 'finger distance'
    # print fingerDistance
    # --------------------------------------------------------------------------------------------------

    #  Finger Angles
```

```
                                                                                    #
***********************************************************************************
************************
    # angles = []
    # for i in range(0, len(fingersx)-1):
    #     x11 = fingersx[i][0]-centerMass
    #     # print x11
    #     x12 = fingersx[i+1][0]-centerMass
    #     # print x12
    #     angles.append(angle(x11, x12))
    # # print 'finger angles'
    # # print angles
    # -----------------------------------------------------------------------------------------------------------

    # MIN Enclosing Circle
                                                                                    #
***********************************************************************************
*************************
    center, radius = cv2.minEnclosingCircle(cnts)
    xi, yi = center
    x = int(xi)
    y = int(yi)
    rad = int(radius)
    cv2.circle(frame, (x, y), rad, [0, 0, 255], 2)
    # -----------------------------------------------------------------------------------------------------------

    cv2.imshow('Dilation', frame)

  if len(fingerDistance) == 5:
    for i in range(0, len(fingerDistance)):
      datalist[i] = datalist[i] + fingerDistance[i]
    counter += 1
  #print counter
  if counter == bypass:
    f1 = datalist[0]/bypass
    f2 = datalist[1]/bypass
    f3 = datalist[2]/bypass
    f4 = datalist[3]/bypass
    f5 = datalist[4]/bypass

    s1 = mymap(f1)
    s2 = mymap(f2)
    s3 = mymap(f3)
    s4 = mymap(f4)
    s5 = mymap(f5)
      # for x in datalist:
      #     data.append(x/bypass)
    print datalist
    datalist = [0, 0, 0, 0, 0]
    counter = 0
```

```
        data = '<'+str(s1)+','+str(s2)+','+str(s3)+','+str(s4)+','+str(s5)+'>'
        runTest(data)

        print data



    # if loops > 800:
    #     break
     loops += 1
     k = cv2.waitKey(1) & 0xFF
# clear the stream in preparation for the next frame
     rawCapture.truncate(0)
     if k == ord("q"):#27:
        break
     #if cap.get(cv2.CAP_PROP_POS_FRAMES) == cap.get(cv2.CAP_PROP_FRAME_COUNT)-60:
        # If the number of captured frames is equal to the total number of frames,
        # we stop
     #    break
# ka1= cap.get(cv2.CAP_PROP_FRAME_COUNT)
# ka2 =cap.get(cv2.CAP_PROP_POS_FRAMES)

print loops
bluetooth.close
#cap.release()
# cv2.destroyAllWindows()
```

ΠΑΡΑΡΤΗΜΑ Β

Κώδικας του Arduino

```
// Arduino Mega 2560

#include <Servo.h>                  //Servo

// 5 Servo objects
Servo myServo[5];                  //  Servo

// 5 servo pin attachments
byte servoPin[5] = {2, 3, 4, 5, 6};

byte servoMin = 10;
byte servoMax = 170;
```

```
// initial servo positions
byte newServoPos[5] = {servoMin, servoMin, servoMin, servoMin, servoMin};
byte servoPos[5] = {2, 2, 2, 2, 2};

const byte buffSize = 40;
char inputBuffer[buffSize];
const char startMarker = '<';
const char endMarker = '>';
byte bytesRecvd = 0;
boolean readInProgress = false;
boolean newDataFromRPi = false;



//============================================================================
void setup() {


 //initialize bluetooth
 Serial1.begin(9600);
 //initialize serial to PC
 Serial.begin(9600);

 // initialize the servo
 for(byte i = 0; i < 5; i++){                    //  Servo
   myServo[i].attach(servoPin[i]);           //  Servo
 }

 // move servo to initial angle
 moveServo();                                  //  Servo

 // tell Raspberry Pi we are ready
 Serial1.println("<Arduino is ready>");
  // tell the PC we are ready
 Serial.println("<Arduino is ready>");
}
//============================================================================

void loop() {

 getDataFromRPi();
 updateServoPos();
 moveServo();
 //reply();
 delay(10);
```

```
  //replyToPC();


}

//================================================================================

void getDataFromRPi() {
   // receive data from BT and save it into inputBuffer


  while (Serial1.available()>0) {
   char x = Serial1.read();

  //bt.write(x);                          // Uncomment to send reply to python
  // newDataFromRPi = true;

     // the order of these IF clauses is significant

   if (x == endMarker ) {
    readInProgress = false;
    newDataFromRPi = true;
    inputBuffer[bytesRecvd] = 0;
    parseData();

   }

   if(readInProgress) {
    inputBuffer[bytesRecvd] = x;
    bytesRecvd ++;
    if (bytesRecvd == buffSize) {
      bytesRecvd = buffSize - 1;
    }
   }

   if (x == startMarker ) {
    bytesRecvd = 0;
    readInProgress = true;
   }
  }
//}
delay(10);
}

//================================================================================

void parseData() {
   // split the data into its parts
 char * strtokIndx;                          // this is used by strtok() as an index
```

```
  strtokIndx = strtok(inputBuffer,",");      // get the first part - Servo1Pos
  newServoPos[0] =  atoi(strtokIndx);     // convert this part to an integer

  strtokIndx = strtok(NULL, ",");             // this continues where the previous call left off
  newServoPos[1] = atoi(strtokIndx);     // convert this part to an integer

  strtokIndx = strtok(NULL, ",");             // this continues where the previous call left off
  newServoPos[2] = atoi(strtokIndx);     // convert this part to an integer

  strtokIndx = strtok(NULL, ",");            // this continues where the previous call left off
  newServoPos[3] = atoi(strtokIndx);     // convert this part to an integer

  strtokIndx = strtok(NULL, ",");            // this continues where the previous call left off
  newServoPos[4] = atoi(strtokIndx);     // convert this part to an integer

}



//============================================================================

void reply() {

  if (newDataFromRPi) {
    newDataFromRPi = false;

    Serial.print("Servo1 ");
    Serial.println(newServoPos[0]);
    Serial.print("Servo2 ");
    Serial.println(newServoPos[1]);
    Serial.print("Servo3 ");
    Serial.println(newServoPos[2]);
    Serial.print("Servo4 ");
    Serial.println(newServoPos[3]);
    Serial.print("Servo5 ");
    Serial.println(newServoPos[4]);
    Serial.println("===========");
    Serial1.write("OK");
  }
}
//============================================================================

void replyToPC() {

  if (newDataFromRPi) {
    newDataFromRPi = false;

    Serial1.print("<Servo1 >");
    Serial1.println(newServoPos[0]);
```

```
   Serial1.print("<Servo2 >");
   Serial1.println(newServoPos[1]);
   Serial1.print("<Srvo3 >");
   Serial1.println(newServoPos[2]);
   Serial1.print("< Servo4 >");
   Serial1.println(newServoPos[3]);
   Serial1.print("< Servo5 >");
   Serial1.println(newServoPos[4]);
   Serial1.print("<END>");
  }
}

//================================================================================

void updateServoPos() {

  for(byte i = 0; i < 5; i++){
    if(newServoPos[i] < servoMin){
      newServoPos[i] = servoMin;
    }
    if(newServoPos[i] > servoMax){
      newServoPos[i] = servoMax;
    }
  }
 }

//================================================================================

void moveServo() {
  for(byte i = 0; i < 5; i++){
    if (servoPos[i] != newServoPos[i]) {
      servoPos[i] = newServoPos[i];
      myServo[i].write(servoPos[i]);        //  Servo
    }

  }


}
```