ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΒΙΟΜΗΧΑΝΙΚΗΣ ΣΧΕΔΙΑΣΗΣ & ΠΑΡΑΓΩΓΗΣ
ΤΟΜΕΑΣ ΑΥΤΟΜΑΤΙΣΜΟΥ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΕΦΑΡΜΟΓΕΣ ΜΗΧΑΝΙΚΗΣ ΟΡΑΣΗΣ

Σπουδαστές: Βαμβακούσης Μιχαήλ Α.Μ.: 44895
Καζαντζής Παύλος Α.Μ.: 41702

Επιβλέπων Καθηγητής: Δρ. Νικολάου Γρηγόρης

ΑΙΓΑΛΕΩ ΙΑΝΟΥΑΡΙΟΣ 2019

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ ΓΙΑ ΤΗ ΓΡΑΜΜΑΤΕΙΑ

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος ΒΑΜΒΑΚΟΥΣΗΣ ΜΙΧΑΗΛ, του ΑΓΓΕΛΗ, φοιτητής του Τμήματος ΒΙΟΜΗΧΑΝΙΚΗΣ ΣΧΕΔΙΑΣΗΣ ΚΑΙ ΠΑΡΑΓΩΓΗΣ του Πανεπιστημίου Δυτικής Αττικής, πριν αναλάβω την εκπόνηση της Πτυχιακής Εργασίας μου, δηλώνω ότι ενημερώθηκα για τα παρακάτω:

«Η Πτυχιακή Εργασία (Π.Ε) αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο του συγγραφέα, όσο και του Ιδρύματος και θα πρέπει να έχει μοναδικό χαρακτήρα και πρωτότυπο περιεχόμενο.

Απαγορεύεται αυστηρά οποιοδήποτε κομμάτι κειμένου της να εμφανίζεται αυτούσιο ή μεταφρασμένο από κάποια άλλη δημοσιευμένη πηγή. Κάθε τέτοια πράξη αποτελεί προϊόν λογοκλοπής και εγείρει θέμα Ηθικής Τάξης για τα πνευματικά δικαιώματα του άλλου συγγραφέα. Αποκλειστικός υπεύθυνος είναι ο συγγραφέας της Π.Ε, ο οποίος φέρει και την ευθύνη των συνεπειών, ποινικών και άλλων, αυτής της πράξης.

Πέραν των όποιων ποινικών ευθυνών του συγγραφέα, σε περίπτωση που το Ίδρυμα του έχει απονείμει Πτυχίο, αυτό ανακαλείται με απόφαση της Συνέλευσης του Τμήματος. Η Συνέλευση του Τμήματος με νέα απόφασή της, μετά από αίτηση του ενδιαφερόμενου, του αναθέτει εκ νέου την εκπόνηση Π.Ε με άλλο θέμα και διαφορετικό επιβλέποντα καθηγητή. Η εκπόνηση της εν λόγω Π.Ε πρέπει να ολοκληρώσει εντός τουλάχιστον ενός ημερολογιακού 6μήνου από την ημερομηνία ανάθεσής της. Κατά τα λοιπά εφαρμόζονται τα προβλεπόμενα στο άρθρο 18. παρ.5 του ισχύοντος Εσωτερικού Κανονισμού».

Ο Δηλών

Ημερομηνία

21/01/2019

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος ΚΑΖΑΝΤΖΗΣ ΠΑΥΛΟΣ, του ΘΕΟΔΩΡΟΥ, φοιτητής του Τμήματος ΒΙΟΜΗΧΑΝΙΚΗΣ ΣΧΕΔΙΑΣΗΣ ΚΑΙ ΠΑΡΑΓΩΓΗΣ του Πανεπιστημίου Δυτικής Αττικής, πριν αναλάβω την εκπόνηση της Πτυχιακής Εργασίας μου, δηλώνω ότι ενημερώθηκα για τα παρακάτω:

«Η Πτυχιακή Εργασία (Π.Ε) αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο του συγγραφέα, όσο και του Ιδρύματος και θα πρέπει να έχει μοναδικό χαρακτήρα και πρωτότυπο περιεχόμενο.

Απαγορεύεται αυστηρά οποιοδήποτε κομμάτι κειμένου της να εμφανίζεται αυτούσιο ή μεταφρασμένο από κάποια άλλη δημοσιευμένη πηγή. Κάθε τέτοια πράξη αποτελεί προϊόν λογοκλοπής και εγείρει θέμα Ηθικής Τάξης για τα πνευματικά δικαιώματα του άλλου συγγραφέα. Αποκλειστικός υπεύθυνος είναι ο συγγραφέας της Π.Ε, ο οποίος φέρει και την ευθύνη των συνεπειών, ποινικών και άλλων, αυτής της πράξης.

Πέραν των όποιων ποινικών ευθυνών του συγγραφέα, σε περίπτωση που το Ίδρυμα του έχει απονείμει Πτυχίο, αυτό ανακαλείται με απόφαση της Συνέλευσης του Τμήματος. Η Συνέλευση του Τμήματος με νέα απόφασή της, μετά από αίτηση του ενδιαφερόμενου, του αναθέτει εκ νέου την εκπόνηση Π.Ε με άλλο θέμα και διαφορετικό επιβλέποντα καθηγητή. Η εκπόνηση της εν λόγω Π.Ε πρέπει να ολοκληρώσει εντός τουλάχιστον ενός ημερολογιακού 6μήνου από την ημερομηνία ανάθεσής της. Κατά τα λοιπά εφαρμόζονται τα προβλεπόμενα στο άρθρο 18. παρ.5 του ισχύοντος Εσωτερικού Κανονισμού».

Ο Δηλών

Ημερομηνία

21/01/2019

Ευχαριστίες

Σε αυτό το σημείο θα θέλαμε να ευχαριστήσουμε τον επιβλέποντα καθηγητή Γρηγόρη Νικολάου για την καθοδήγηση και την πολύτιμη βοήθεια που μας προσέφερε κατά τη διάρκεια της εκπόνησης της πτυχιακής μας εργασίας.

Επίσης θα θέλαμε να ευχαριστήσουμε τις οικογένειές μας και τους φίλους μας για τη συμπαράσταση τους σε όλη την πορεία της πτυχιακής μας -και όχι μόνο- εργασίας.

Περίληψη

Οι εφαρμογές μηχανικής όρασης τα τελευταία χρόνια έχουν εξελιχθεί σε μεγάλο βαθμό με αποτέλεσμα να κατατάσσονται στο επίκεντρο των περισσοτέρων εφαρμογών και σύγχρονων αναγκών του ανθρώπου. Από την εξέλιξη των Συστημάτων Αυτόματου Ελέγχου ο άνθρωπος όρισε τα βασικά χαρακτηριστικά της επιστήμης του αυτοματισμού και άρχισε να τα εισάγει σε διάφορες εφαρμογές.

Εμείς στην παρούσα μελέτη, προσπαθήσαμε να μελετήσουμε διάφορες εφαρμογές με επίκεντρο τη μηχανική όραση και το πώς μπορούμε μέσω διαφόρων εφαρμογών να την εκμεταλλευτούμε για την απόκτηση δεδομένων, τόσο σε λειτουργίες βιομηχανικής παραγωγής και ελέγχου, όσο και σε εφαρμογές αναγνώρισης διαφόρων αντικειμένων και ελέγχου μέσω αισθητηρίων σε DC μοτέρ.

Για την υλοποίηση των εφαρμογών αυτών χρησιμοποιήσαμε τη διαδεδομένη πλατφόρμα ηλεκτρονικών Arduino για τη διασύνδεση των αισθητηρίων και των κυκλωμάτων, καθώς και το πρόγραμμα Simulink της Matlab για τον έλεγχο και τη μοντελοποίηση των διεργασιών μας. Έτσι, συμπεράναμε πώς οι εφαρμογές μέσω της μηχανικής όρασης αποκρίνονται σε διάφορες μορφές ελέγχου και προτείναμε τον πιο επιθυμητό τρόπο ελέγχου με βάση τις αποκρίσεις εξόδου και αποτελεσμάτων των εφαρμογών μας.

Τέλος, στο τελευταίο κεφάλαιο προτείνουμε διάφορες εφαρμογές μηχανικής όρασης μέσω κάμερας για αναγνώριση χρωμάτων, οι οποίες μπορούν να βρουν χρήση σε διάφορους κλάδους της βιομηχανίας πλαστικών, υφασμάτων και μετάλλων.

Abstract

Over the last years, mechanical vision applications have been developed to a great extent and as a result it has a place at the epicenter of most of the applications and modern human needs. From the progress of Automatic Control Systems, the human set forth the basic characteristics of the Science of Automation and started to introduce them in various applications.

In the present study we tried to research various applications of mechanical vision and how we can take advantage of them through various applications to collect incidence data in industrial production operations and control as well as in applications for object recognition and control by means of sensors in a DC motor.

For the implementation of those applications we used the widely used electronic platform Arduino, for the intervention of sensors and circuits, as well as the program Simulink of Matlab for the control and modeling of our processes. Thus, we got conclusions on how the application respond through the mechanical vision to different forms of control, and we suggested the desired way of control based on the output responses and the results of our applications.

Finally, in the last chapter of our study we suggest various applications with of mechanical vision using a camera for color recognition, which can be utilized by the plastic, material and metal industry.

Πίνακας Περιεχομένων

Κατάλογος Εικόνων

Εισαγωγή

Το θέμα το οποίο θα εξεταστεί στην παρούσα διπλωματική εργασία είναι το πώς μέσω κάποιων εφαρμογών μηχανικής όρασης θα γίνει αναγνώριση θέσης κάποιων αντικειμένων και έλεγχος DC μοτέρ, καθώς και η συμπεριφορά τους με βάση κάποια ανάδραση – αισθητήρα. Με τα αποτελέσματα των εφαρμογών αυτών θα προσδιορίσουμε αν οι εφαρμογές μας αποκρίνονται επιθυμητά και θα συγκρίνουμε τις εξόδους και αποκρίσεις των συστημάτων που θα δημιουργήσουμε για την κάθε εφαρμογή μέσω του εργαλείου Simulink.
Η μελέτη αυτής της εργασίας έγινε προκειμένου μέσω της μηχανικής όρασης να μελετήσουμε εργαλεία (Arduino) και εφαρμογές (Simulink) για να μπορούμε να ελέγχουμε συμπεριφορές συστημάτων καθώς και να παραμετροποιούμε όπως εμείς επιθυμούμε, ως μηχανικοί.
Με την επιλογή της συγκεκριμένης πτυχιακής εργασίας μελετήσαμε τόσο τον οπτικό έλεγχο μέσω κάμερας, που βρίσκεται σε ραγδαία εξέλιξη σε όλες σχεδόν τις εφαρμογές όσο και τον έλεγχο θέσης και ταχύτητας ενός μοτέρ, καθώς και άλλες εφαρμογές της μηχανικής όρασης σε διάφορους τομείς της βιομηχανίας.
Στο τελευταίο κομμάτι της εργασίας μας, με βάση τις τεχνολογίες και την τεχνογνωσία που αποκτήσαμε στα προηγούμενα κεφάλαια, μελετάμε εφαρμογές που κάνουν έλεγχο μέσω κάμερας και είναι υλοποιήσιμες στον τομέα της βιομηχανίας και όχι μόνο.
Η μεθοδολογία, η οποία θα ακολουθήσουμε βασίζεται σε πειράματα που έγιναν βάσει του εξοπλισμού που διαθέτουμε και θα αναφερθεί στο κυρίως μέρος. Με βάση τα περιφερειακά στοιχεία της κάθε εφαρμογής και τα κατάλληλα εργαλεία απόκτησης δεδομένων, καταφέραμε να υπολογίσουμε αριθμητικές τιμές, διαγράμματα και γραφήματα και να τα συγκρίνουμε. Έτσι, καταλήξαμε σε συμπεράσματα και αποφάσεις, τα οποία και διατυπώσαμε όσο πιο κατανοητά μπορούσαμε. Βασικό κομμάτι δυσκολίας της πτυχιακής μας ήταν η συλλογή δεδομένων, καθώς για την απόκτησή τους υπήρξαν πολλές διαταραχές στα περιφερειακά στοιχεία που χρησιμοποιήθηκαν, καθώς και απώλειες. Είχαμε να κάνουμε με ηλεκτρικά και ηλεκτρονικά στοιχεία, με αποτέλεσμα τα συμπεράσματά μας να αποκλίνουν από τις θεωρητικές τιμές. Τελικά, θα μπορούσαμε σε περίπτωση ιδανικών συνθηκών να αποφύγουμε τις διαταραχές αυτές και να έχουμε έναν άρτιο έλεγχο για τις παραπάνω εφαρμογές.

Κεφάλαιο 1: Μηχανική Όραση

Η ανάλυση και η ψηφιακή επεξεργασία εικόνας σε αλγορίθμους καλείται μηχανική όραση, υπολογιστική όραση ή τεχνητή όραση. Στο πεδίο της πληροφορικής και συγκεκριμένα της τεχνητής νοημοσύνης, η οποία ασχολείται με τη σχεδίαση και την υλοποίηση υπολογιστικών συστημάτων που μιμούνται στοιχεία της ανθρώπινης συμπεριφοράς, η μηχανική όραση εμπλέκεται στη σχεδίαση και κατασκευή συστημάτων που λαμβάνουν και αναλύουν δεδομένα ψηφιακών εικόνων.

Η ανάλυση των δεδομένων αυτών, μέσω αλγορίθμων, μπορούν να παρουσιαστούν σε τρισδιάστατο σχήμα – εικόνες. Με τη σειρά τους, οι αλγόριθμοι αυτοί δημιουργούν τρισδιάστατα μοντέλα (πίνακες αριθμών), με τη βοήθεια των οποίων μπορούμε να κάνουμε εντοπισμό (Detection) λ.χ. κίνηση μπάλας (παρούσα εφαρμογή).

Ανέκαθεν ο άνθρωπος προσπαθούσε στα πλαίσια βελτίωσης αυτοματοποιημένων συστημάτων να εισαγάγει τη μηχανική όραση και να την εξελίξει στο έπακρο, έτσι ώστε η μηχανική όραση να αποδίδει το ίδιο ή και καλύτερα από την ανθρώπινη όραση. Αυτό έχει επιτευχθεί σε ήδη υπάρχουσες εφαρμογές που έχουν να κάνουν με την ασφάλεια και επίβλεψη κτηρίων, την ιατρική θεραπεία και διάγνωση, την κίνηση σε δρόμους, την επεξεργασία βίντεο, τα διαδραστικά παιχνίδια, αλλά και ρομπότ οικιακής και στρατιωτικής χρήσης.

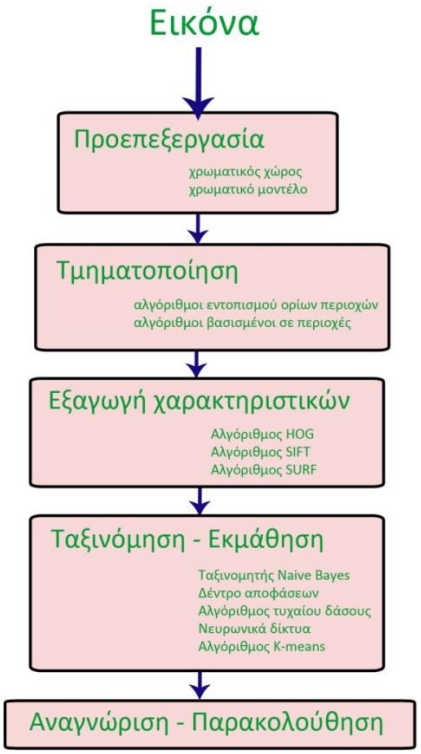
Πριν δούμε αναλυτικά τον τρόπο λειτουργίας της μηχανικής όρασης ας αναλογιστούμε τα ερεθίσματα που δέχεται ο ανθρώπινος εγκέφαλος από τις εικόνες των ματιών. Λέει ο λαός: «μία εικόνα είναι χίλιες λέξεις», κι αυτό σημαίνει ότι από την εικόνα μπορούμε να αντλήσουμε ποικίλες πληροφορίες, είτε από τα χρώματα, είτε από τα γεωμετρικά σχήματα, είτε από αντικείμενα-πρόσωπα που έχει απομνημονεύσει ο εγκέφαλος. Στη συνέχεια, ο εγκέφαλος αυτές τις πληροφορίες τις ερμηνεύει σε καταστάσεις και συναισθήματα.

Με παρόμοιο τρόπο η μηχανική όραση αντλεί πληροφορίες. Η ψηφιακή εικόνα (φωτογραφίες-βίντεο) υφίσταται επεξεργασία μέσα από μια σειρά αλγορίθμων ενός ηλεκτρονικού υπολογιστή και ο υπολογιστής λαμβάνει τις πληροφορίες της εικόνας. Οι αλγόριθμοι επιλέγονται ανάλογα με τις πληροφορίες που θέλουμε να μας δίνει το σύστημα της μηχανικής όρασης. Για παράδειγμα ένα σύστημα που είναι προγραμματισμένο να εντοπίζει μόνο γάτες σε μια εικόνα, δεν μπορεί να εντοπίζει και ανθρώπους.

Η άντληση αυτών των πληροφοριών γίνεται με την ανίχνευση κάποιων προτύπων (Detection). Αν το πρότυπο είναι αντικείμενο τότε η διαδικασία ονομάζεται **Object**

detection. Από εκεί και ύστερα ανάλογα με τον τύπο του αντικειμένου έχουμε Face detection, Color detection, κ.λπ.

Για να φτάσουμε να κάνουμε Object detection από μία εικόνα θα πρέπει να ακολουθήσουμε μια σειρά από βήματα-διαδικασίες, όπως παρουσιάζονται στο παρακάτω διάγραμμα:



Εικόνα 1.1 Διάγραμμα Μηχανικής Όρασης

1.1 Προεπεξεργασία εικόνας (Image Preprocessing)

Για να ξεκινήσουμε να αναλύουμε μία εικόνα, θα πρέπει αρχικά να της δώσουμε μια μαθηματική υπόσταση. Δηλαδή να χτίσουμε μια βάση μαθηματικής ανάλυσης πάνω στην οποία θα πατήσουν τα υπόλοιπα βήματα. Αυτό το βήμα πραγματοποιείται όταν οριστεί ο χρωματικός χώρος και το χρωματικό μοντέλο στο οποίο ανήκει η εικόνα. Με πολύ απλά λόγια, καθορίζονται μαθηματικά τα χρώματα της εικόνας. Η διαδικασία αυτή είναι απαραίτητη για τα επόμενα βήματα.

1.1.1 Χρωματικοί χώροι - Χρωματικά μοντέλα

Στον κόσμο των υπολογιστών, για να μπορέσουμε να περιγράψουμε το κάθε χρώμα με ακρίβεια είναι απαραίτητο να συνδεθεί το κάθε χρώμα με μεταβλητές ή με αριθμούς. Αυτό έχουν σκοπό τα χρωματικά μοντέλα. Ομάδες διαφορετικών χρωματικών μοντέλων είναι οι χρωματικοί χώροι. [1]

Γνωστά χρωματικά μοντέλα είναι το RGB, το HIS, το HSV, το CMY(K) κλπ. Ο διαχωρισμός μεταξύ των χρωματικών μοντέλων γίνεται με βάση τα παρακάτω χαρακτηριστικά:

- λαμπρότητα (brightless)
- απόχρωση (Hue)
- χρωματική καθαρότητα ή χρωματικός κορεσμός (Saturation)
- ένταση φωτός (Intensity)

1.1.1.1 Μοντέλο RGB

Στο χρωματικό μοντέλο RGB συμπεριλαμβάνονται χρώματα που αποτελούνται από τα τρία πρωτεύοντα ή βασικά χρώματα: το κόκκινο (R-red), το πράσινο (G-green) και το μπλε (B-blue). Η δημιουργία κάθε χρώματος είναι αποτέλεσμα των τριών χρωμάτων διαφορετικών εντάσεων. Ο συνδυασμός και των τριών βασικών χρωμάτων σε πλήρη ισχύ μάς δίνει το λευκό χρώμα ενώ η απουσία τους το μαύρο. [1]

1.1.1.2 Χώρος YCbCr

Χρησιμεύει σε εφαρμογές συμπίεσης εικόνας, όμως τα χρώματα δεν απεικονίζονται σύμφωνα με τον τρόπο αντίληψης των ανθρώπων. Η συνιστώσα Y υποδηλώνει τη φωτεινότητα και οι Cb, Cr τη χρωματική διαφορά της Y με το μπλε και το κόκκινο, αντίστοιχα.
Οι λόγοι που χρησιμοποιείται αυτός ο μετασχηματισμός είναι:
α) Η όραση του ανθρώπου είναι πιο ευαίσθητη στη φωτεινότητα παρά στο χρώμα.
β) Η ταχύτητα μετάδοσης της εικόνας βελτιστοποιείται μετασχηματίζοντας το σήμα RGB σε σήμα φωτεινότητας και δύο σήματα μεταφοράς χρωματικής πληροφορίας.
Τα δύο σήματα χρώματος χρειάζονται μικρότερη ακρίβεια για να αναπαρασταθούν και να μεταδοθούν από την ακρίβεια ενός σήματος φωτεινότητας. [1]

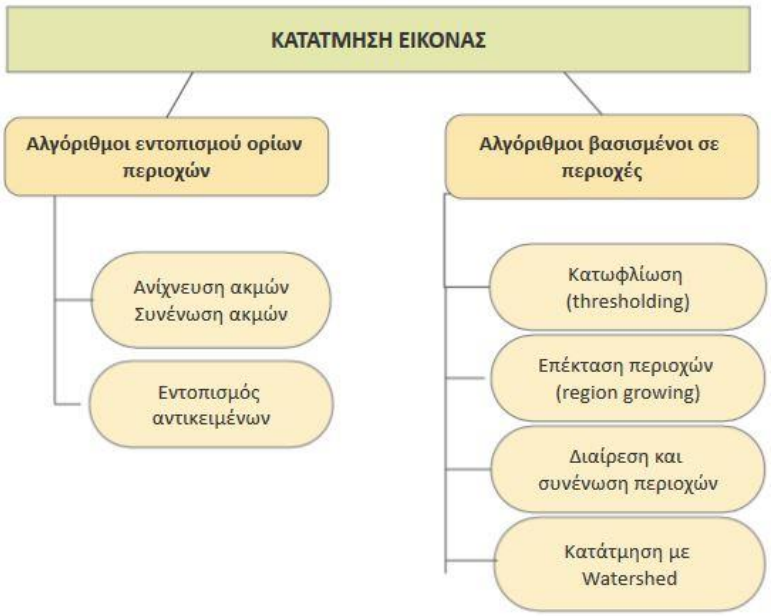
1.2 Τμηματοποίηση (Image Segmentation)

Τμηματοποίηση ή κατάτμηση εικόνας είναι ο χωρισμός της ψηφιακής εικόνας σε επιμέρους εικόνες. Αυτό βοηθά στην απλούστευση των πληροφοριών της εικόνας. Η διαδικασία αυτή χρησιμεύει στην αναγνώριση και παρακολούθηση αντικειμένων (Object detection-tracking).
Η κατάτμηση εικόνας πραγματοποιείται με δύο κατηγορίες αλγορίθμων. Με τους αλγορίθμους εντοπισμού ορίου περιοχών (boundary based segmentation) και με τους αλγορίθμους βασισμένους σε περιοχές (region based segmentation). [2]
Η κατηγορία αλγορίθμων εντοπισμού ορίων περιοχών εμπεριέχει δύο κλάδους μελέτης :

- Ανίχνευση ακμών – Συνένωση ακμών
- Εντοπισμός αντικειμένων με το μετασχηματισμό Hough

Η κατηγορία αλγορίθμων βασισμένων σε περιοχές εμπεριέχει τέσσερις κλάδους μελέτης:

- Κατωφλίωση (thresholding)
- Επέκταση περιοχών (region growing)
- Διαίρεση και συνένωση περιοχών (splitting and merging)
- Κατάτμηση με βάση τον μετασχηματισμό Watershed



Εικόνα 1.2 Κατηγορίες αλγορίθμων κατάτμησης εικόνας

Για λόγους συντομίας δε θα αναφερθούμε στους παραπάνω αλγορίθμους αναλυτικά.

Το αποτέλεσμα της κατάτμησης εικόνας είναι η εξαγωγή μιας δυαδικής εικόνας στην οποία μέσω της μαθηματικής μορφολογίας (στην οποία θα αναφέρουμε παρακάτω) μπορούμε να αναλύσουμε διαφορετικά αντικείμενα. Στην κατάτμηση της εικόνας βοηθούν διάφορα χαρακτηριστικά όπως η υφή και το χρώμα των αντικειμένων. Έτσι μπορούμε να κάνουμε κατάτμηση στην εικόνα με βάση το χρώμα, αλλά και με βάση τις εντάσεις των χρωμάτων. Για τις μεθόδους αυτές είναι απαραίτητο να προηγηθεί μια άλλη μέθοδος, η κατάτμηση ιστογράμματος (histogram-based method). Με αυτήν την μέθοδο, κατασκευάζεται ένα ιστόγραμμα από όλα τα pixels στην εικόνα. Τα κοίλα και οι κορυφές στο ιστόγραμμα χρησιμοποιούνται για να εντοπιστούν τα κοινά pixels της εικόνας. Ως μέτρο μπορεί να χρησιμοποιηθεί το χρώμα ή η ένταση του κάθε pixel.

1.2.1 Κατάτμηση μέσω χρώματος

Όταν η κατάτμηση γίνεται μέσω χρώματος, λαμβάνουμε μονοδιάστατο ιστόγραμμα της εικόνας για μονόχρωμες εικόνες και πολυδιάστατο για έγχρωμες. Η γενικότερη λογική είναι ότι η ανίχνευση κορυφών ή κοιλάδων στο ιστόγραμμα οδηγεί στην εύρεση αντικειμένων και φόντου αντίστοιχα. Στις πολύχρωμες εικόνες, βέβαια, η διαδικασία είναι πιο περίπλοκη, διότι τα τρία ιστογράμματα, του χρωματικού δηλαδή χώρου RGB (Red Green Blue – Κόκκινο Πράσινο Μπλε), μπορούν να υποστούν επεξεργασία χωριστά και να συνδυαστούν τα αποτελέσματα με στόχο να επικρατήσει η ισχυρότερη υπόθεση κατάτμησης, δηλαδή αυτή που μας δίνει καλύτερα το αντικείμενο που θέλουμε μέσω του συνδυασμού απόχρωσής του. [3]

1.2.2 Κατάτμηση μέσω έντασης

Όταν η κατάτμηση γίνεται μέσω έντασης του κάθε pixel, η διαδικασία είναι λιγότερο σύνθετη. Μιλώντας για μια ασπρόμαυρη (gray-scale) εικόνα, όπου φαίνεται πιο καθαρά η έννοια της έντασης, κάθε αντικείμενο με σχετικά μεγάλο μέγεθος δημιουργεί μια κατανομή pixel στο ιστόγραμμα της εικόνας γύρω από τη μέση τιμή έντασής του. [3]

Όπως είναι γνωστό, αλλά σημειώνεται και εδώ, για τις εικόνες επιπέδου γκρι με τον όρο «χαρακτηριστικά έντασης» αναφερόμαστε στις τιμές που μπορεί να λάβει ένα pixel. Συγκεκριμένα, υπάρχουν 256 αποχρώσεις του γκρι, με το 0 να συμβολίζει το μαύρο και το 255 το λευκό. Επομένως, 8 bits είναι απαραίτητα για την περιγραφή της έντασης ($2^8 = 256$).

1.3 Εξαγωγή Χαρακτηριστικών (Feature Extraction)

Μετά την κατάτμηση της εικόνας ακολουθεί η διαδικασία εξαγωγής χαρακτηριστικών (feature extraction). Με αυτή τη διαδικασία γνωρίζουμε ποια τμήματα της εικόνας είναι διακριτά, όπως γραμμές, γωνίες, σχήματα που μπορούν να περιγράψουν την εικόνα. Η ομάδα αλγορίθμων αυτής της διαδικασίας λαμβάνει υπόψη ότι κάθε pixel της εικόνας έχει τιμή 8bit. Άρα μια εικόνα διάστασης 640x320 θα έχει 640x320x8 = 1638400 bits πληροφορίας. Με αυτόν τον τρόπο μπορούμε να εστιάσουμε περισσότερο στην πληροφορία που μας ενδιαφέρει απομονώνοντας τα υπόλοιπα pixels, μετατρέποντας την εικόνα σε διάνυσμα χαρακτηριστικών. Μερικοί τύποι αλγορίθμων που κάνουν εξαγωγή χαρακτηριστικών είναι: [4]

- Ιστόγραμμα προσανατολισμένης κλίσης (Histogram of Oriented Gradients ή HOG)
- Μετασχηματισμός μεταβλητών χαρακτηριστικών κλίμακας (Scale-Invariant Feature Transform ή SIFT)

- Επιτάχυνση δυναμικού χαρακτηριστικού (Speeded Up Robust Feature ή SURF)

1.4 Ταξινόμηση Εικόνας (Image Classification)

Η διαδικασία αυτή λαμβάνει το διάνυσμα χαρακτηριστικών που έχει προκύψει από το προηγούμενο βήμα και το μετατρέπει σε ετικέτα κλάσης. Για παράδειγμα, μπορεί να καταλαβαίνει ότι μια εικόνα που έχει υποστεί διαδικασία εξαγωγής χαρακτηριστικών (feature extraction) περιέχει ένα ζώο που είναι γάτα. Αυτή η αναγνώριση ότι το ζώο είναι γάτα και όχι σκύλος χρειάζεται μια βάση δεδομένων από πίσω και κάποιους συγκεκριμένους τρόπους μηχανισμού ταξινόμησης.

Ένα μέρος της διαδικασίας αυτής είναι και η εκμάθηση (Learning). Αν στο προηγούμενο παράδειγμα η βάση δεδομένων δεν έχει αποθηκευμένη την πληροφορία «γάτα», τότε θα πρέπει να την αποθηκεύσει για μελλοντική χρήση.

Όλα αυτά επιτυγχάνονται με μια μεγάλη γκάμα αλγορίθμων ταξινόμησης και εκμάθησης (Learning and Classification Algorithms). Κανένας αλγόριθμος δεν μπορούμε να κρίνουμε ότι είναι καλύτερος από τους άλλους, γιατί το αποτέλεσμά τους εξαρτάται από τον χώρο εφαρμογής τους. Για παράδειγμα αν οι κλάσεις είναι γραμμικές ή όχι, αλλά και από άλλα χαρακτηριστικά. [5]

Κάποιοι αλγόριθμοι ταξινόμησης είναι οι παρακάτω:

- Ταξινόμηση Naïve Bayes (Naive Bayes Classifier)
- Δέντρα αποφάσεων (Decision Trees)
- Τυχαίο δάσος (Random Forest)
- Νευρωνικά Δίκτυα (Neural Network)
- Κοντινών γειτόνων (Nearest Neighbor ή K-means)

1.4.1 Ταξινόμηση Naïve Bayes

Πρόκειται για μια τεχνική ταξινόμησης βασισμένη στο Θεώρημα του Bayes. Με απλά λόγια, ένας ταξινομητής Naive Bayes υποθέτει ότι η παρουσία ενός συγκεκριμένου χαρακτηριστικού σε μια ομάδα ταξινόμησης δε σχετίζεται με την παρουσία οποιουδήποτε άλλου χαρακτηριστικού. Ακόμη και αν αυτά τα χαρακτηριστικά εξαρτώνται το ένα από το άλλο ή από την ύπαρξη των άλλων χαρακτηριστικών, όλες αυτές οι ιδιότητες έχουν έναν συγκεκριμένο ρόλο.

1.4.2 Δέντρα αποφάσεων

Ο αλγόριθμος αυτός ονομάζεται «δέντρο απόφασης» γιατί δημιουργεί μοντέλα ταξινόμησης με τη μορφή δομής δέντρου. Διαμερίζει ένα σύνολο δεδομένων σε μικρότερα και μικρότερα υποσύνολα, ενώ ταυτόχρονα αναπτύσσεται σταδιακά ένα συνδεδεμένο δέντρο αποφάσεων. Το τελικό αποτέλεσμα είναι ένα δέντρο με κόμβους απόφασης και κόμβους φύλλων. Ένας κόμβος απόφασης έχει δύο ή περισσότερους κλάδους και ένας κόμβος φύλλων αντιπροσωπεύει μια ταξινόμηση ή μια απόφαση. Ο κορυφαίος κόμβος απόφασης σε ένα δέντρο, που αντιστοιχεί στον καλύτερο προγνωστικό παράγοντα, ονομάζεται κόμβος ρίζας. Τα δέντρα αποφάσεων μπορούν να χειριστούν τόσο τα κατηγορηματικά όσο και τα αριθμητικά δεδομένα.

1.4.3 Τυχαίο δάσος

Τα τυχαία δάση αποφάσεων είναι μια μέθοδος μάθησης για την ταξινόμηση και άλλες διαδικασίες που λειτουργούν με την κατασκευή ενός πλήθους δέντρων αποφάσεων στον χρόνο εκμάθησης και ταξινόμησης δεδομένων. Τυχαία δάση αποφάσεων διορθώνουν τη λειτουργία των δέντρων αποφάσεων.

1.4.4 Νευρωνικά Δίκτυα

Ένα νευρωνικό δίκτυο αποτελείται από μονάδες διατεταγμένες σε στρώματα, τους επονομαζομένους νευρώνες. Οι νευρώνες μετατρέπουν ένα διάνυσμα εισόδου σε κάποια έξοδο. Κάθε νευρώνας λαμβάνει μια είσοδο, εφαρμόζει μια συνάρτηση σε αυτήν (συχνά μη γραμμική) και στη συνέχεια μεταδίδει την έξοδο στον επόμενο νευρώνα. Γενικά, τα νευρωνικά δίκτυα ορίζονται ως τροφοδοτικά, δηλαδή μια μονάδα τροφοδοτεί την παραγωγή της σε όλες τις μονάδες του επόμενου στρώματος, αλλά δεν υπάρχει ανάδραση στο προηγούμενο στρώμα.

1.4.5 Κοντινών γειτόνων (Nearest Neighbor ή K-means)

Ο αλγόριθμος K-means είναι ένας αλγόριθμος αυτόματης εκμάθησης που επιλύει το πρόβλημα της κατηγοριοποίησης σε συστάδες (ομάδες pixel) που αντιπροσωπεύουν ένα αντικείμενο της εικόνας. Ο K-means ακολουθεί μια τεχνική επαναληπτικής δομής που χρησιμοποιείται για να χωρίσει μια εικόνα σε K αριθμό συστάδων. Η διαδικασία που ακολουθεί είναι να ταξινομεί ένα δοσμένο σύνολο στοιχείων από μια συστάδα.

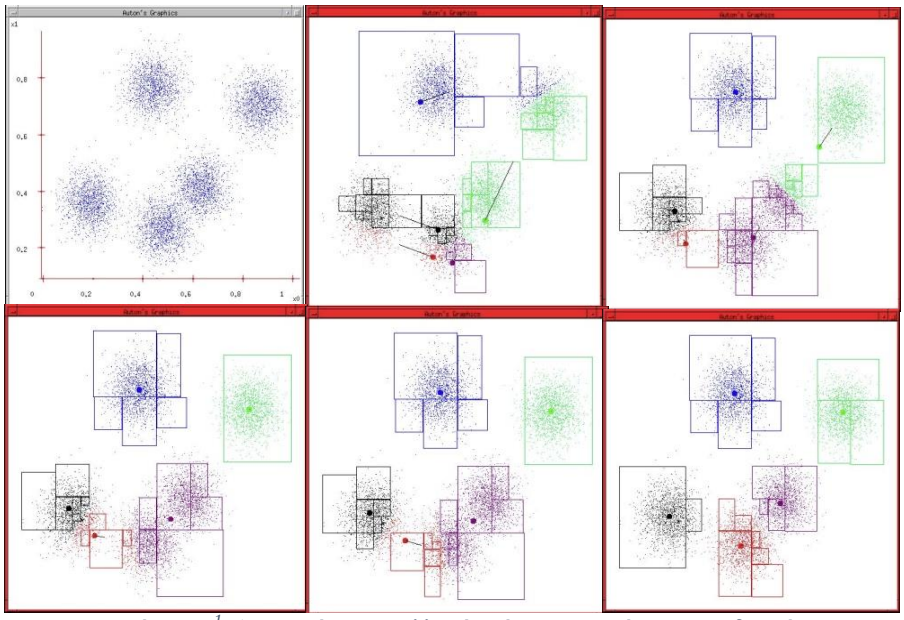
Τέλος, οι ιδιότητες του κάθε αντικειμένου αντιστοιχίζονται σε διανύσματα. Ο αλγόριθμος βρίσκει τις αποστάσεις μέσω μιας διαφοράς, η οποία είναι κατά κανόνα βασισμένη στο

χρώμα, την ένταση, τη σύσταση, ή τη θέση ενός pixel, ή έστω έναν ισορροπημένο συνδυασμό αυτών των παραγόντων. Ο στόχος που προσπαθούμε να επιτύχουμε είναι να ελαχιστοποιήσουμε τη συνολική διαφορά των συστάδων [3].

Η κύρια ιδέα είναι να καθοριστούν Κ κέντρα, ένα για κάθε συστάδα. Αυτά τα κέντρα πρέπει να τοποθετούνται με έναν έξυπνο τρόπο, γιατί η διαφορετική θέση των αρχικών κέντρων δημιουργεί και διαφορετικό αποτέλεσμα. Έτσι, για να μην ενωθούν οι συστάδες, η καλύτερη επιλογή είναι να τοποθετηθούν όσο το δυνατόν πιο μακριά το ένα κέντρο από το άλλο. Στη συνέχεια γίνεται λήψη του κάθε σημείου που ανήκει σε μια συστάδα στοιχείων και συνδέονται με το κοντινότερο σε αυτό κέντρο. Όταν όλα τα σημεία συνδεθούν τότε έχουμε μια πρώτη κατηγοριοποίηση.

Ύστερα ο αλγόριθμος υπολογίζει εκ νέου καινούργια Κ κέντρα ως κέντρα βάρους των συστάδων ως αποτέλεσμα του προηγούμενου βήματος. Αφού έχει αυτά τα νέα κέντρα, μια νέα σύνδεση πρέπει να γίνει μεταξύ των ίδιων καθορισμένων στοιχείων με το κοντινότερο νέο κέντρο. [3] Η ίδια διαδικασία επαναλαμβάνεται συνεχώς μέχρις ότου τα κέντρα σταθεροποιηθούν και αποτελέσουν τα τελικά κέντρα καθεμίας συστάδας.

Παρακάτω είναι η αναπαράσταση διαδοχικής αλλαγής κέντρου κατά τη συσταδοποίηση.



Εικόνα 1.3[1] *Αναπαράσταση αλλαγής κέντρων κατά τη συσταδοποίηση*

[1] https://apothesis.lib.teicrete.gr/bitstream/handle/11713/3089/Xylourgos2009.pdf?sequence=1

1.5 Ανίχνευση (Detection)

Όλα αυτά τα βήματα-διαδικασίες που αναφέρθηκαν παραπάνω είναι μια σειρά συνδυασμένων αλγορίθμων. Αυτοί οι αλγόριθμοι αντλούν όλες τις πληροφορίες από μία εικόνα. Εστιάζουν και αναδεικνύουν συγκεκριμένες από αυτές τις πληροφορίες. Το αποτέλεσμα είναι η εστίαση σε συγκεκριμένες πληροφορίες από μία εικόνα με ποικίλες πληροφορίες. Αν η πληροφορία εξόδου της διαδικασίας είναι αντικείμενο, τότε η διαδικασία ονομάζεται 'Object Detection' ή 'Recognition'. Αν το είδος της πληροφορίας που αντλούμε είναι κάτι πιο συγκεκριμένο, για παράδειγμα πρόσωπα ή αυτοκίνητα, τότε η διαδικασία θα ονομάζεται 'Face Detection' και 'Car Detection' αντίστοιχα. [4]

1.6 Παρακολούθηση αντικειμένων (Object tracking)

Η οπτική παρακολούθηση αντικειμένων είναι ένα από τα πιο σημαντικά στοιχεία στον τομέα των εφαρμογών της μηχανικής όρασης. Η παρακολούθηση αντικειμένων (Object tracking) γίνεται μετά τον εντοπισμό των αντικειμένων (Object detection), και είναι η αναγνώριση ενός αντικειμένου κάθε χρονική στιγμή καθώς αυτό κινείται σε μία οπτική σκηνή (frame). Ο βασικός στόχος της οπτικής παρακολούθησης αντικειμένων (που ενίοτε μπορεί να αναφέρεται και σε ανθρώπους, πρόσωπα κ.ά.) είναι η εκτίμηση των καταστάσεων του στόχου-αντικειμένου στα επόμενα καρέ (frame). Παρόλη την πρόοδο που έχει σημειωθεί τα τελευταία χρόνια, το πρόβλημα της οπτικής παρακολούθησης παραμένει πολύ δύσκολο. Λόγοι για τους οποίους συμβαίνει αυτό, προέρχονται από το ίδιο το περιβάλλον στο οποίο διαδραματίζεται η σκηνή, ή ακόμα και από το ίδιο το αντικείμενο. Για παράδειγμα, παραλλαγή φόντου και φωτισμού, άκαμπτα σημεία του αντικειμένου, μη γραμμική κίνηση του αντικειμένου, ή αλληλεπίδραση μεταξύ άλλων αντικειμένων. Ως εκ τούτου, είναι σημαντικό να αξιολογηθούν οι καλύτεροι μηχανισμοί που θα βοηθήσουν στον εντοπισμό μελλοντικών κατευθύνσεων των αντικειμένων-στόχων μέσω κάποιων αλγορίθμων παρακολούθησης.

1.7 Μαθηματική μορφολογία

Η μορφολογία είναι ένα ευρύ σύνολο λειτουργιών που έχει προσφέρει το μέρος των τελεστών στην επεξεργασία εικόνας. Οι τελεστές χρησιμεύουν στην ανάλυση εικόνων στο δυαδικό σύστημα. Οι πιο συνηθισμένες εφαρμογές τους είναι η ανίχνευση ακμών, η απομάκρυνση θορύβου, η ενίσχυση και τμηματοποίηση μιας εικόνας. [1]

1.7.1 Ανίχνευση Ακμών

Η κατάτμηση μπορεί να γίνει και με την ανίχνευση ακμών (Edge Detection) των ορίων των αντικειμένων. Ακμή (edge) είναι ένα σύνολο από σημεία ασυνέχειας τα οποία καθορίζουν το όριο ανάμεσα σε δύο ομοιόμορφες περιοχές. Οι ακμές είναι χαρακτηριστικά της εικόνας και φέρουν χρήσιμες πληροφορίες για τα όρια των αντικειμένων. Εικόνες που δεν έχουν έντονες αντιθέσεις είναι δύσκολο να αναλυθούν.

Η κατάτμηση των ασπρόμαυρων (gray-scale) εικόνων έχει μικρό βαθμό δυσκολίας σε σχέση με τις έγχρωμες εικόνες, που η κατάτμησή τους είναι πιο πολύπλοκη. Τρεις κύριες φιλοσοφίες επικρατούν: [2]

- Επεξεργασία και στα τρία χρώματα ταυτόχρονα.
- Επεξεργασία σε κάθε χρώμα χωριστά.
- Συνδυασμός των επιμέρους αποτελεσμάτων με διάφορα κριτήρια.

Μια ενδιαφέρουσα προσέγγιση είναι η πρόβλεψη των ορίων, παρακολουθώντας τις αλλαγές χρωμάτων των pixels και βρίσκοντας την κατεύθυνση αυτών των αλλαγών. Ακολουθώντας αυτές τις αλλαγές και συνδυάζοντάς τες μεταξύ τους, η εύρεση των ακμών είναι εφικτή.

Οι ανιχνευτές ακμών ομαδοποιούνται σε δύο κατηγορίες. Στις τοπικές τεχνικές, που χρησιμοποιούν τελεστές και δρουν σε τοπικές γειτονιές της εικόνας, και στις καθολικές τεχνικές, που χρησιμοποιούν καθολική πληροφορία από όλη την εικόνα και μεθόδους φιλτραρίσματος για την εξαγωγή πληροφοριών από ακμές.

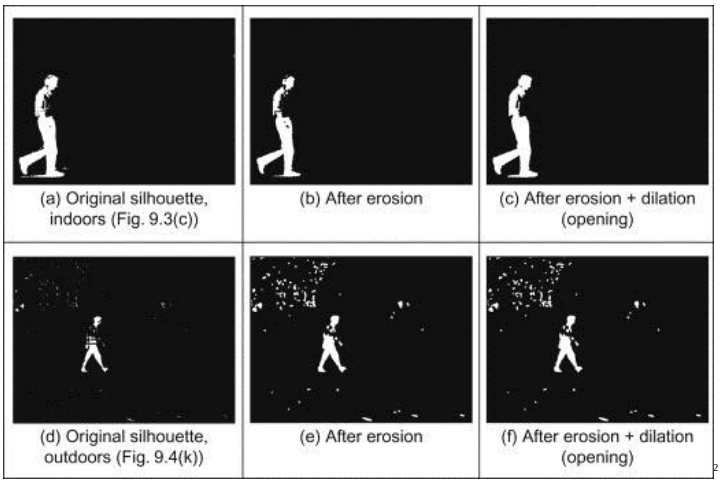
Από τις καθολικές τεχνικές, πολύ ξεχωριστή είναι η χρήση της κλίσης της εικόνας, η οποία βασίζεται στο γεγονός ότι οι ακμές έχουν οριστεί ως τοπικές μεταβολές της φωτεινότητας της εικόνας και, συνεπώς, τεχνικές διαφόρισης της εικόνας είναι δυνατό να εξαγάγουν την απαραίτητη πληροφορία σχετικά με τις ακμές. Η κλίση (gradient) της αρχικής εικόνας βρίσκεται εφαρμόζοντας φίλτρα, ή αλλιώς κατασκευάζοντας μάσκες, εύρεσης ακμών σε αυτήν.

Οι μάσκες είναι αριθμητικοί πίνακες μικρών διαστάσεων που εφαρμόζονται επανειλημμένα σε κάθε pixel της εικόνας με απλές μαθηματικές πράξεις, επηρεάζοντας τα γειτονικά pixels. Αυτή η διαδικασία είναι ουσιαστικά η εύρεση της πρώτης παραγώγου της αρχικής εικόνας και την φιλτράρει κατασκευάζοντας σαν έξοδο μια άλλη ασπρόμαυρη εικόνα (black & white) όπου απεικονίζονται με λευκό χρώμα οι ακμές της αρχικής, δηλαδή τα σημεία αλλαγής φωτεινότητάς της. Οι μάσκες είναι υπεύθυνες για το ποιες ακμές ανιχνεύονται. Κάθε μάσκα έχει αδυναμία σε συγκεκριμένα είδη θορύβου και σε αδύνατες ακμές. Αυτός είναι και ο λόγος που έχουμε διαφορετικές μάσκες και όχι μία που να είναι ιδανική για κάθε ιδιαιτερότητα κάθε εικόνας. [2]

1.7.2 Συστολή (Erosion) και Διαστολή (Dilation)

Οι πιο βασικές μορφολογικές λειτουργίες είναι η διαστολή (dilation) και η διάβρωση (erosion). Η διαστολή προσθέτει εικονοστοιχεία (pixels) στα όρια των αντικειμένων μιας εικόνας, ενώ η διάβρωση αφαιρεί εικονοστοιχεία στα όρια αντικειμένων. Ο αριθμός των εικονοστοιχείων που προστέθηκαν ή αφαιρέθηκαν από τα αντικείμενα μιας εικόνας εξαρτάται από το μέγεθος και το σχήμα του στοιχείου διαμόρφωσης που χρησιμοποιείται για την επεξεργασία της εικόνας. Στις λειτουργίες μορφολογικής διαστολής και διάβρωσης, η κατάσταση οποιουδήποτε δεδομένου εικονοστοιχείου στην εικόνα εξόδου προσδιορίζεται εφαρμόζοντας έναν κανόνα στο αντίστοιχο εικονοστοιχείο και των γειτονικών εικονοστοιχείων του στην εικόνα εισόδου. Ο κανόνας που χρησιμοποιείται για την επεξεργασία των εικονοστοιχείων ορίζει τη λειτουργία ως διαστολή ή διάβρωση. [6]

Ένα από τα κύρια προβλήματα που παρατηρήθηκαν μέχρι τώρα είναι τα σημεία θορύβου που μολύνουν το εντοπισμένο κινούμενο αντικείμενο και την εικόνα. Δεδομένου ότι τα σημεία αυτά προέρχονται από την κίνηση, είναι απίθανο η εικόνα να μπορεί να εξομαλυνθεί επαρκώς για να αφαιρεθούν χωρίς να απαλειφθούν οι λεπτομέρειες της εικόνας. Ως εκ τούτου, είναι συχνή η χρήση της μορφολογίας για την απομάκρυνσή τους. Αυτό μπορεί να επιτευχθεί είτε με την αφαίρεση των απομονωμένων λευκών σημείων, είτε με τη χρήση διάβρωσης και διαστολής (άνοιγμα και κλείσιμο). Η διάβρωση απομακρύνει τον θόρυβο ενώ διατηρεί το σχήμα. Υποθετικά, η διαστολή μπορεί να συνδέει τα χωριστά τμήματα (το κεφάλι και οι ώμοι διαχωρίζονται στην πραγματικότητα από το σώμα στο Σχήμα 9.4). Η επίδραση αυτών των λειτουργιών στις εικόνες που προκύπτουν από το προσωρινό διάμεσο φιλτράρισμα στην εσωτερική και την εξωτερική απεικόνιση παρουσιάζεται στο Σχήμα 9.3. Στο Σχήμα 9.3 (β) και 9.4 (ε), όπου μπορούμε να δούμε ότι η διάβρωση απομακρύνει κάποια από τη σκιά στην εσωτερική απεικόνιση και τα περισσότερα από τα αποτελέσματα του δέντρου στην εξωτερική απεικόνιση. Η διαστολή κατόπιν επιστρέφει τις σιλουέτες στο ίδιο μέγεθος, όπως φαίνεται στο Σχήμα 9.3 (c) και 9.4 (f). [6]



Εικόνα 1.4[2] Απεικονίσεις συστολής και διαστολής

1.8 Ανάλυση Blob (Blob Analysis)

Ο όρος «BLOB» σημαίνει Binary Large OBject και αναφέρεται σε μια ομάδα συνδεδεμένων pixel σε μια δυαδική εικόνα. Ο όρος «Large» δείχνει ότι μόνο αντικείμενα συγκεκριμένου μεγέθους και πάνω μάς ενδιαφέρουν, και ότι μικρά αντικείμενα είναι συνήθως θόρυβος.

Στη Μηχανική Όραση, μέθοδοι εντοπισμού blob χρησιμοποιούνται για την ανίχνευση περιοχών μιας εικόνας που διαφέρουν ως προς κάποιες ιδιότητες (όπως χρώμα ή φωτεινότητα) από τις γειτονικές τους. Σχηματικά, ένα blob είναι μια περιοχή της εικόνας όπου κάποιες ιδιότητες είναι σταθερές ή σχεδόν σταθερές. Όλα τα σημεία σε ένα blob μπορούν να θεωρηθούν υπό μία έννοια παρόμοια μεταξύ τους. Η πιο κοινή μέθοδος ανίχνευσης blobs είναι με χρήση συνέλιξης (convolution). [10]

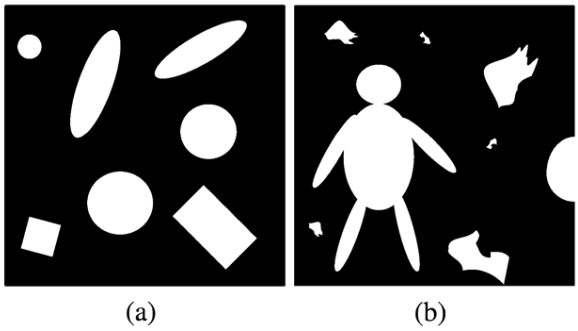
Υποθέτοντας ότι υπάρχει κάποια ιδιότητα που μας ενδιαφέρει και μπορεί να εκφραστεί ως συνάρτηση της θέσης πάνω στην εικόνα, υπάρχουν δύο κύριες κατηγορίες ανιχνευτών blob (blob detectors):

1. Διαφορικές μέθοδοι (differential methods), που βασίζονται στις παραγώγους της συνάρτησης ως προς τη θέση, και
2. Μέθοδοι που βασίζονται σε τοπικά ακρότατα (methods based on local extrema), που βασίζονται στην εύρεση των τοπικών μεγίστων και ελαχίστων της συνάρτησης.

Υπάρχουν πολλοί λόγοι για να μελετήσει και να αναπτύξει κανείς blob detectors. Ένας κύριος λόγος είναι για να πάρουμε συμπληρωματική πληροφορία για κάποιες περιοχές της

[2] https://www.sciencedirect.com/topics/engineering/detail-image

εικόνας, η οποία δεν μπορεί να δοθεί από ανιχνευτές ακμών ή ανιχνευτές γωνιών. Παλιότερα, η ανίχνευση blob χρησιμοποιούνταν για να πάρουμε περιοχές της εικόνας με ιδιαίτερο ενδιαφέρον, οι οποίες μπορεί να περιείχαν αντικείμενα ή τμήματα αντικειμένων (εφαρμογή σε object detection/recognition). Σε άλλα πεδία, όπως για παράδειγμα η ανάλυση ιστογράμματος, ανάλυση blob μπορεί να χρησιμοποιηθεί για ανίχνευση κορυφών (peak detection) με εφαρμογές στην τμηματοποίηση (segmentation). Μια άλλη συνήθης χρήση των blob detectors είναι για την ανάλυση και αναγνώριση υφής (texture).



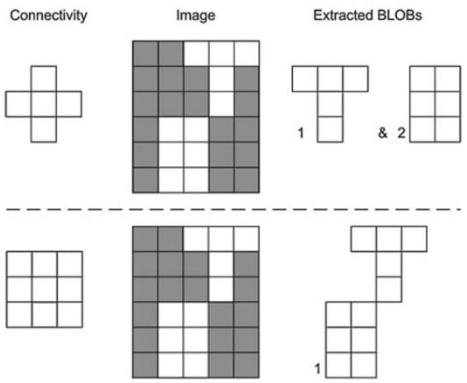
Εικόνα 1. 5 [3](a) Δυαδική εικόνα με διάφορα σχήματα (b) Δυαδική εικόνα με έναν άνθρωπο και θόρυβο.

Για παράδειγμα, blob analysis μπορεί να χρησιμοποιηθεί για τον σχεδιασμό ενός αλγορίθμου που θα βρίσκει πόσοι κύκλοι υπάρχουν στην Εικόνα 1. 5. Προφανώς η απάντηση είναι τρεις, αλλά πώς μπορεί αν το βρει αυτό ένας υπολογιστής; Ένα άλλο παράδειγμα είναι ο εντοπισμός του ανθρώπου στη Εικόνα 1. 5. Η διαδικασία απαιτεί δύο στάδια: εξαγωγή των blobs (blob extraction) και ταξινόμηση των blobs (blob classification). [7]

1.8.1 Εξαγωγή blobs (blob extraction)

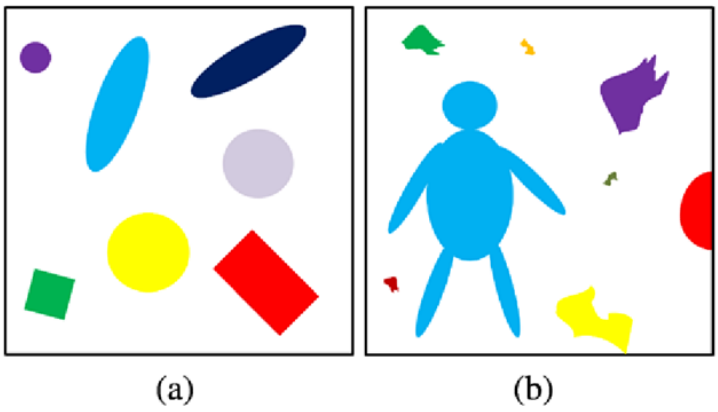
Ο σκοπός της εξαγωγής blobs είναι να απομονωθούν τα blobs (ουσιαστικά τα αντικείμενα) σε μια εικόνα. Όπως αναφέρθηκε, το blob αποτελείται από μια ομάδα συνδεδεμένων pixels. Το αν δύο pixels συνδέονται ή όχι καθορίζεται από το connectivity, δηλαδή ποια pixels είναι γείτονες και ποια όχι. Οι δυο πιο συχνά χρησιμοποιούμενοι τύποι connectivity παρουσιάζονται στην Εικόνα 1.6. Το 8-connectivity είναι πιο ακριβές από το 4-connectivity, αλλά το 4-connectivity χρησιμοποιείται συχνά επειδή απαιτεί λιγότερους υπολογισμούς και έτσι μπορεί να επεξεργαστεί την εικόνα ταχύτερα.

[2,3] http://what-when-how.com/introduction-to-video-and-image-processing/blob-analysis-introduction-to-video-and-image-processing-part-1/



Εικόνα 1.6 [4] 4- και 8-connectivity

Υπάρχουν αρκετοί διαφορετικοί αλγόριθμοι για την εύρεση των blobs, οι οποίοι συχνά αποκαλούνται connected component analysis ή connected component labeling. Ένας από αυτούς είναι ο αλγόριθμος Grass-Fire (που μπορεί να τροποποιηθεί και να εφαρμοστεί και σε grayscale ή έγχρωμες εικόνες). Μετά την εκτέλεση των αλγορίθμων αυτών, παίρνουμε ομάδες από pixels που η καθεμιά αντιστοιχεί σε ένα blob, μαζί με μία ετικέτα για το blob (αριθμό/χρώμα). Ένα παράδειγμα φαίνεται στην Εικόνα 1.7. [7]



Εικόνα 1.7[5] Δύο παραδείγματα εξαγωγής blobs με διαφορετικά χρώματα

1.8.2 Ταξινόμηση blobs (blob classification)

Το επόμενο βήμα μετά την εξαγωγή είναι η ταξινόμηση των blobs. Στην περίπτωση της πρώτης εικόνας, θέλουμε να ξέρουμε ποιο blob είναι κύκλος και ποιο όχι, ενώ στην περίπτωση της δεύτερης, ποιο blob είναι άνθρωπος και ποιο όχι. Η ταξινόμηση αποτελείται από δύο βήματα: την εξαγωγή χαρακτηριστικών (feature extraction) και την καθαυτό ταξινόμηση, δηλαδή το ταίριασμα με βάση τα feature vectors (classification). Και τα δύο

[4,5] http://what-when-how.com/introduction-to-video-and-image-processing/blob-analysis-introduction-to-video-and-image-processing-part-1/

στάδια έχουν περιγραφεί προηγουμένως. Έτσι, στο τέλος της διαδικασίας, μπορούμε να ξέρουμε ποια αντικείμενα υπάρχουν στην εικόνα. [7]

1.9 Έλεγχος Οπτικής Οδήγησης (Visual Servoing)

Η Οπτική οδήγηση ρομπότ (Visual Servoing robot) είναι ένα κλειστού βρόχου σύστημα ελέγχου θέσης ενός ρομποτικού βραχίονα με τη χρήση της μηχανικής όρασης. Πιο συγκεκριμένα, για τον έλεγχο του βραχίονα χρησιμοποιούνται πληροφορίες ανάδρασης (οπτικές πληροφορίες) που συλλέγονται από τους αισθητήρες όρασης. Υπάρχουν διαφόρων τύπων αισθητήρες όρασης, όπως κάμερες, σαρωτές, laser, αισθητήρες χρωμάτων, κίνησης κ.ά.

Η οπτική πληροφορία είναι το σφάλμα της θέσης του ρομποτικού βραχίονα από την επιθυμητή τιμή. Το σφάλμα μετράται σε καρτεσιανές συντεταγμένες ή στο πλαίσιο αναφοράς του αισθητήρα, στο χώρο δράσης του βραχίονα και εισέρχεται σε κάποιον ελεγκτή για περεταίρω έλεγχο του συστήματος. Οι κατηγορίες συστημάτων οπτικής οδήγησης είναι πέντε και χρησιμοποιούνται με βάση δύο κριτήρια. Τη δομή του ελέγχου και το πλαίσιο αναφοράς του σφάλματος. [8]

1.9.1 Ιεραρχικά συστήματα οπτικής οδήγησης (ή dynamic look–and–move)

Όταν ο έλεγχος του συστήματος οπτικής οδήγησης είναι ιεραρχικός, η οπτική πληροφορία χρησιμοποιείται για να παραχθούν σήματα, τα οποία εφαρμόζονται στην είσοδο ενός εσωτερικού ελεγκτή των ρομποτικών αρθρώσεων. Ο ελεγκτής αυτός είναι απαραίτητος για την ευστάθεια του ρομποτικού συστήματος, αφού η οπτική πληροφορία δεν επηρεάζει κατευθείαν τις αρθρώσεις του ρομπότ. [8]

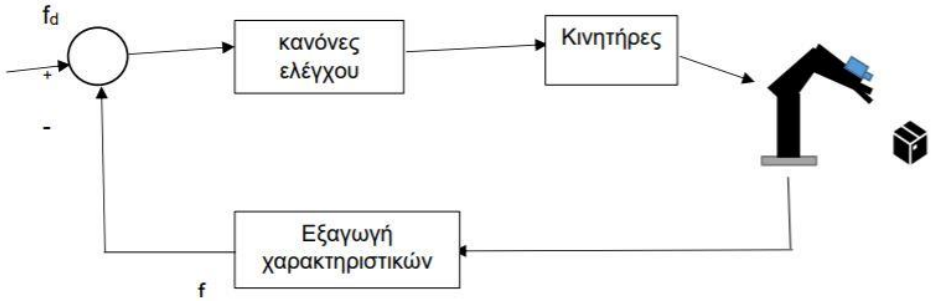
1.9.2 Συστήματα με απευθείας έλεγχο (ή direct visual servo)

Στην περίπτωση των συστημάτων με direct visual servo, ο εσωτερικός ελεγκτής των ρομποτικών αρθρώσεων δε χρησιμοποιείται καθόλου. Η οπτική πληροφορία εισάγεται σε έναν ελεγκτή οπτικής οδήγησης, ο οποίος υπολογίζει τα σήματα αναφοράς των ρομποτικών αρθρώσεων. Η ευστάθεια του συστήματος εξαρτάται αποκλειστικά από τη μηχανική όραση. [8]

Η αρχιτεκτονική dynamic look–and–move χρησιμοποιείται πιο συχνά σε σχέση με την direct visual servo αρχιτεκτονική. Στα συστήματα με direct visual servo ο ρυθμός δειγματοληψίας (άρα και ελέγχου) είναι αυτός της λήψης οπτικής πληροφορίας από την κάμερα και είναι αρκετά χαμηλός. Αυτό δυσκολεύει πάρα πολύ τον απ' ευθείας έλεγχο ενός ρομπότ του οποίου οι κινηματικές εξισώσεις εμφανίζουν μεγάλη πολυπλοκότητα και μη γραμμικότητες. [8]

1.9.3 Συστήματα με βάση την εικόνα (IBVS)

Η τεχνική IBVS βασίζεται στο σφάλμα μεταξύ των χαρακτηριστικών της εικόνας που υφίσταται επεξεργασία και στα χαρακτηριστικά της επιθυμητής εικόνας, και δεν περιλαμβάνει καμία εκτίμηση της θέσης του στόχου, παρά μόνο για τον άμεσο έλεγχο των βαθμών ελευθερίας (DOF) του ρομπότ. Τα χαρακτηριστικά μπορεί να είναι οι συντεταγμένες των οπτικών χαρακτηριστικών, γραμμές ή ακόμα και περιοχές. Η τεχνική αυτή είναι ιδιαίτερα ανθεκτική στον θόρυβο και στα σφάλματα. Ένα αρνητικό σημείο του συστήματος είναι ότι, με την εφαρμογή αυτής της μεθόδου, δεν μπορεί το ρομπότ να εκτελέσει μεγάλες περιστροφές. [9]

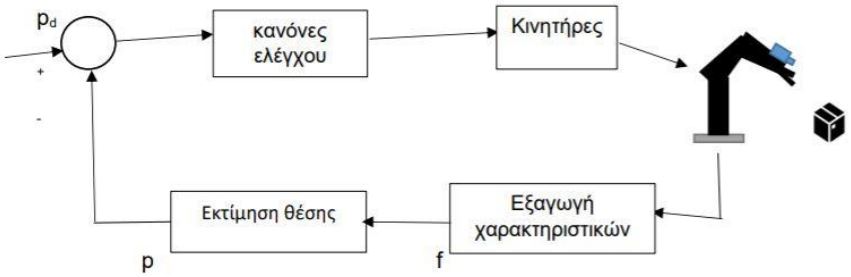


Εικόνα 1.8[6] IBVS σύστημα servoing

1.9.4 Συστήματα με βάση τη θέση (PBVS)

Το PBVS είναι μια τεχνική ελέγχου βασισμένη στην εκτίμηση της θέσης του αντικειμένου σε σχέση με τη θέση της κάμερας. Αυτή η εκτίμηση γίνεται με βάση τις καρτεσιανές συντεταγμένες της κάμερας και του αντικειμένου. Στη συνέχεια αποστέλλεται στον ελεγκτή ρομπότ, ο οποίος με τη σειρά του ελέγχει το ρομπότ. Επίσης, είναι απαραίτητη η 3D αναπαράσταση του μοντέλου του αντικειμένου. Στα μειονεκτήματα αυτής της τεχνικής μπορούμε να προσάψουμε την ευαισθησία στον θόρυβο και τα πολλά σφάλματα λόγω πολλών εναλλαγών θέσεων. [9]

[6] http://hellanicus.lib.aegean.gr/handle/11610/17197



Εικόνα 1.9[7] PBVS σύστημα servoing

1.9.5 Υβριδικά συστήματα

Υπάρχουν βέβαια και υβριδικά συστήματα που αξιοποιούν τα πλεονεκτήματα και των δύο συστημάτων, IBVS και PBVS. Τέτοια συστήματα βασίζονται στο σφάλμα ανάμεσα στα χαρακτηριστικά της εικόνας που υφίσταται επεξεργασία και στα χαρακτηριστικά της επιθυμητής εικόνας, που όμως περιστρέφονται με βάση το καρτεσιανό σύστημα συντεταγμένων. Τέτοια συστήματα αποσκοπούν στο να είναι πιο ανθεκτικά στον θόρυβο. [9]

1.9.6 Κατηγορίες χρήσης κάμερας

Γενικά, τα οπτικά δεδομένα μπορούν να ληφθούν είτε από μια κάμερα (monocular vision) που βρίσκεται πάνω στον βραχίονα ή σε ένα κινητό μέρος ενός ρομποτικού συστήματος (eye-in-hand) είτε να παρακολουθεί το ρομπότ από μία σταθερή θέση (eye-to-hand).

Επίσης, μπορούμε να χρησιμοποιήσουμε δύο ή περισσότερες κάμερες (binocular vision ή redundant camera system, αντίστοιχα) ώστε να συνδυάσουμε τα οπτικά δεδομένα που λαμβάνουμε. Η τοποθέτησή τους μπορεί να γίνει σε σταθερές θέσεις του χώρου εργασίας είτε σε συνδυαστική διάταξη (stand alone και eye-in-hand). Σε κάθε πρόβλημα οπτικής οδήγησης η επιλογή του πλήθους και της διάταξης των καμερών εξαρτάται από τις απαιτήσεις του ίδιου του προβλήματος. Για παράδειγμα η χρήση άνω των δύο καμερών δίνει στο σύστημα την αντίληψη του βάθους και της ακρίβειας ενώ σε αντίθετη περίπτωση η χρήση μίας κάμερας δίνει πιο απλουστευμένη επεξεργασία και άρα είναι πιο ευέλικτο έλεγχο.[8]

Εικόνα 1.10[8] Eye-to-hand και eye-in-hand διαδικασία.

[7,8] http://hellanicus.lib.aegean.gr/handle/11610/17197

Κεφάλαιο 2: Επιπλέον θεωρητικό υπόβαθρο

2.1 Λίγα λόγια για τα Σ.Α.Ε.

2.1.1 Η εξέλιξη των Σ.Α.Ε.

Από την εποχή της ύπαρξης του ανθρώπου πάνω στη γη μέχρι σήμερα, ο άνθρωπος έχει καταφέρει να εισάγει στη ζωή του την τεχνολογία. Εκμεταλλευόμενος όλη αυτή την επιστήμη, σιγά - σιγά με τη τεχνική του σκέψη, η οποία διευρύνεται με την παρατήρηση, τη δοκιμή, τον πειραματισμό και την έρευνα, καταφέρνει να εισαγάγει και τον αυτοματισμό στη ζωή του.

Η περίοδος μέχρι το 1868 χαρακτηρίζεται ως μια περίοδος όπου ο αυτοματισμός αναπτύχθηκε αρκετά. Για περεταίρω ιστορική πληροφόρηση προτείνουμε την πηγή [6]. Μέχρι το 1868 δεν υπάρχει κάποια θεωρητική μαθηματική βάση. Το κενό αυτό συμπληρώνουν ο Maxwell [7], ο Vyshnegradskii [8] και ο Routh [9].

Τα τελευταία εβδομήντα χρόνια τα συστήματα αυτομάτου ελέγχου (Σ.Α.Ε.) αναπτύχθηκαν ραγδαία καθώς εμφανίστηκαν αξιόλογα θεωρητικά και πρακτικά αποτελέσματα όπως αυτά του Nyquist [10] και του Black [11].

Τα επόμενα έτη και κυρίως κατά την περίοδο του Β΄ παγκοσμίου πολέμου και μέχρι το 1957 περίπου, σημειώθηκε αξιόλογη έρευνα που χαρακτηρίζεται σήμερα ως κλασσική θεωρία των συστημάτων αυτομάτου ελέγχου (Σ.Α.Ε.). Από το 1957 μέχρι και σήμερα τα επιτεύγματα που συνέβησαν έδωσαν μια νέα διάσταση και ώθηση στον αυτοματισμό και χαρακτηρίζονται ως σύγχρονη θεωρία των συστημάτων αυτομάτου ελέγχου (Σ.Α.Ε.).

Τα συστήματα αυτομάτου ελέγχου (Σ.Α.Ε.) είναι ένας από τους σημαντικότερους κλάδους της τεχνολογίας καθώς ο αυτοματισμός είναι συνυφασμένος με την ανάπτυξη σχεδόν κάθε μορφής τεχνολογίας. Ο αυτόματος έλεγχος εφαρμόζεται σήμερα από τα πιο απλά μέχρι και τα πιο σύνθετα και πολύπλοκα συστήματα που κατασκευάζει ο άνθρωπος. Για παράδειγμα το αυτοκίνητο, το αεροπλάνο, ο ηλεκτρονικός υπολογιστής, το ρομπότ, οι τηλεπικοινωνίες, οι βιομηχανικές μονάδες, κ.ά.

Σύστημα αυτομάτου ελέγχου (Σ.Α.Ε) είναι ένα σύστημα, που τα διάφορα μέρη του είναι συνδεδεμένα μεταξύ τους έτσι ώστε να συμπεριφέρονται αυτόματα κατά έναν προκαθορισμένο επιθυμητό τρόπο. Η μείωση της συμμετοχής του ανθρώπου στη λειτουργία των μηχανημάτων και των εργοστασίων είναι ένας από τους κύριους σκοπούς της τεχνολογίας, καθώς ο άνθρωπος έχει περιορίσει τις ώρες εργασίας, αυξάνοντας έτσι τον διαθέσιμο χρόνο για ανάπαυση και αξιοποίησή του. Οι βιομηχανίες πλέον λειτουργούν εν

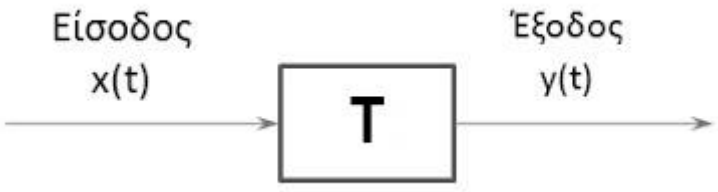
μέρει ή εξολοκλήρου αυτόματα όπως π.χ. οι βιομηχανίες λιπάσματος, ζάχαρης, χάρτου, τσιμέντου, αυτοκινήτων κ.λπ. [12]

2.1.2 Τρόπος λειτουργίας Σ.Α.Ε. – Βασική δομή ενός Σ.Α.Ε.

Σύστημα είναι ένα σύνολο στοιχείων τα οποία είναι κατάλληλα συνδεδεμένα μεταξύ τους για να επιτελέσουν κάποιο έργο. Για να φέρει όμως ένα σύστημα σε πέρας το έργο του, θα πρέπει να του δοθεί η κατάλληλη διέγερση. Το σχήμα 2.1, μαζί με τη χρησιμοποιούμενη επιστημονική ορολογία και συμβολισμό, δίνει μια εικόνα της παραπάνω έννοιας. Η απόκριση *y(t)* καλείται επίσης και συμπεριφορά του συστήματος ή έξοδος.

Αν συμβολίσουμε το σύστημα με τον τελεστή T , τότε η έξοδός του *y(t)* συνδέεται με την είσοδό του *x(t)* με τη σχέση:

$$y(t) = T\ x(t)$$



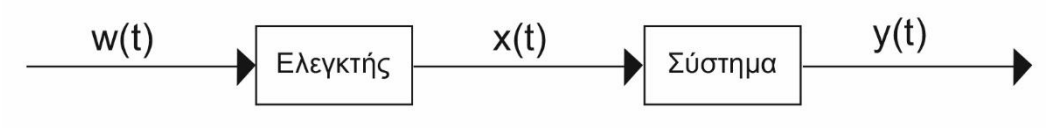
Εικόνα 2.1 Σχηματικό διάγραμμα συστήματος με είσοδο και έξοδο.

Τα Σ.Α.Ε. μπορούν να διακριθούν σε δύο κατηγορίες, στα ανοιχτά και στα κλειστά συστήματα:

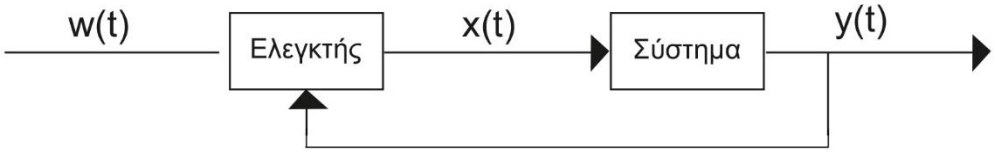
- Το ανοιχτό σύστημα λειτουργεί χωρίς ανάδραση (ανατροφοδότηση) και παράγει απευθείας το αντίστοιχο σήμα εξόδου ως απόκριση του συστήματος σε συγκεκριμένο σήμα εισόδου.
- Το κλειστό σύστημα λειτουργεί με ανάδραση (ανατροφοδότηση) καθώς λαμβάνεται συνεχώς μια μέτρηση του σήματος εξόδου το οποίο και συγκρίνεται με την επιθυμητή έξοδο του συστήματος (σήμα εισόδου) έτσι ώστε να παράγεται ένα σήμα διαφοράς που εφαρμόζεται στη διαδικασία.

Έτσι, σε ένα Σ.Α.Ε. το σήμα εισόδου *x(t)* δεν παράγεται απ΄ευθείας από μία γεννήτρια, αλλά είναι η έξοδος ενός πρόσθετου συστήματος που ονομάζουμε αντιστάθμιση ή ρυθμιστή ή ελεγκτή.

Στα ανοιχτά συστήματα, σχήμα 2.2, ο ελεγκτής διεγείρεται από μία εξωτερική διέγερση *w(t)*, η οποία μπορεί να είναι το σήμα μιάς γεννήτριας. Είναι δε κατασκευασμένος έτσι ώστε η έξοδός του *x(t)* να είναι η κατάλληλη διέγερση στο υπό έλεγχο σύστημα που θα προκαλέσει την επιθυμητή έξοδο *y(t)*. Στα κλειστά συστήματα, σχήμα 2.3, ο ελεγκτής διεγείρεται και από την έξοδο *y(t)* του υπό έλεγχο συστήματος, οπότε η *w(t)* είναι συνάρτηση και της *y(t)*.



Εικόνα 2.2 Ανοιχτό σύστημα Σ.Α.Ε.



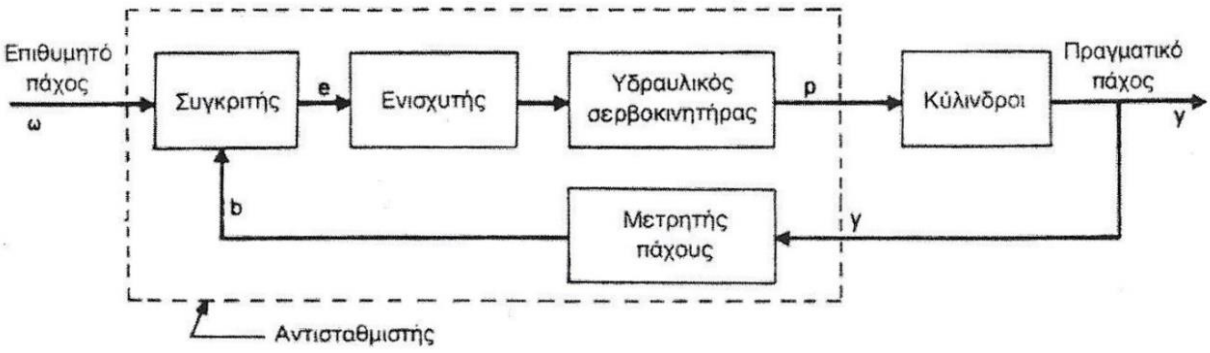
Εικόνα 2.3 Κλειστό σύστημα Σ.Α.Ε.

Τα κλειστά συστήματα διαφέρουν ουσιαστικά στην αρχή λειτουργίας από τα ανοιχτά συστήματα. Η διαφορά αυτή οφείλεται στην τροφοδοσία πληροφοριών από την έξοδο στην είσοδο του συστήματος. Η διαδικασία αυτή ονομάζεται ανατροφοδότηση ή ανάδραση και διαδραματίζει βασικότατο ρόλο στον αυτοματισμό. [12]

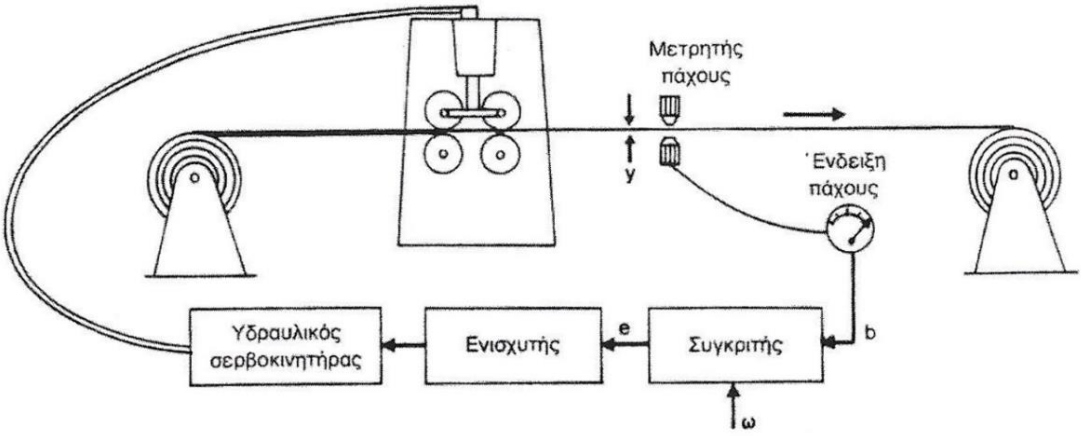
2.1.3 Παράδειγμα ανάδρασης με αισθητήρα

Για την καλύτερη κατανόηση της ανάδρασης θα χρησιμοποιήσουμε ένα σύστημα αυτομάτου ελέγχου πάχους λαμαρίνας, σχήμα 2.4. Το σύστημα αυτό είναι σχεδιασμένο έτσι ώστε το πάχος *y* της λαμαρίνας, που είναι και η έξοδος του συστήματος, να παραμένει σταθερό.

Αυτό επιτυγχάνεται ως εξής: Το πάχος *y* καθορίζεται από την πίεση *p* που ασκούν οι κύλινδροι πάνω στη λαμαρίνα. Η πίεση αυτή ελέγχεται από την ένδειξη *b* του οργάνου που μετρά το πάχος *y* της λαμαρίνας (αισθητήρας). Έτσι, όταν το σφάλμα $e=\omega-b\neq 0$, όπου το σήμα ω αντιπροσωπεύει το επιθυμητό πάχος, τότε ο υδραυλικός σερβοκινητήρας αυξάνει ή μειώνει την πίεση στους κυλίνδρους με αποτέλεσμα το πάχος *y* να μικραίνει ή να μεγαλώνει αντίστοιχα. Η διαδικασία αυτή έχει ως αποτέλεσμα το πάχος *y* της λαμαρίνας να διατηρείται σταθερό και ίσο προς το επιθυμητό πάχος ω. Στο σχήμα 2.5 δίνεται το σχηματικό διάγραμμα του κλειστού συστήματος. [12]



Εικόνα 2.4[8] Εποπτική εικόνα του συστήματος παραγωγής λαμαρίνας.



Εικόνα 2.5[9] Σχηματικό διάγραμμα του συστήματος παραγωγής λαμαρίνας.

2.2 Λειτουργία περιφερειακού εξοπλισμού εφαρμογών

Στην παρούσα πτυχιακή εργασία γίνεται χρήση πολλών εξαρτημάτων για την υλοποίηση των εφαρμογών που θα αναφερθούν. Αυτά είναι:

- Σερβοκινητήρας
- Οδηγός κινητήρα Συνεχούς Ρεύματος
- Αισθητήρας Υπερήχων
- Οπτικός κωδικοποιητής θέσης
- Κάμερα παρακολούθησης (camera)

[8,9] Π.Ν. Παρασκευοπούλου, 'ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΜΑΤΟΥ ΕΛΕΓΧΟΥ' ΘΕΩΡΙΑ ΚΑΙ ΕΦΑΡΜΟΓΕΣ, ΤΟΜΟΣ Α: Σ.Α.Ε. ΣΥΝΕΧΟΥΣ ΧΡΟΝΟΥ, Εκδόσεις Marcel Dekker, ΑΘΗΝΑ, 2007

2.2.1 Σερβοκινητήρες

Οι σερβοκινητήρες χρησιμοποιούνται εδώ και αρκετά χρόνια στα συστήματα αυτομάτου ελέγχου κλειστού βρόχου και σε εφαρμογές, στις οποίες απαιτείται έλεγχος της ταχύτητας, της θέσης και της ροπής του άξονα του κινητήρα. Κλασσικά παραδείγματα χρησιμοποίησης σερβοκινητήρων είναι στους ρομποτικούς βραχίονες, στις αυτόματες εργαλειομηχανές, στα τηλεκατευθυνόμενα μοντέλα, καθώς και στα αυτόματα συστήματα πλοήγησης πλοίων και αεροπλάνων. Κύριο γνώρισμα των σερβοκινητήρων είναι η ικανότητά τους να αναπτύσσουν μεγάλες επιταχύνσεις, όταν ξεκινούν από πλήρη ακινησία, δηλαδή να έχουν μικρή ροπή αδράνειας και μεγάλη ροπή στρέψης. Όταν στους σερβοκινητήρες εφαρμοστεί η τάση λειτουργίας τους, τότε αυτοί περιστρέφονται με μια συγκεκριμένη ταχύτητα (σύμφωνα και με τις προδιαγραφές τους). Για να περιστραφεί ένας σερβοκινητήρας με διαφορετικές ταχύτητες, χρησιμοποιείται η τεχνική PWM (Pulse Width Modulation). Σύμφωνα με αυτή την τεχνική, ο κινητήρας οδηγείται όχι με σταθερή τάση, αλλά με παλμούς σταθερής συχνότητας και ύψους, η διάρκεια των οποίων καθορίζει και την ταχύτητα περιστροφής του κινητήρα. Στην παρούσα εργασία χρησιμοποιούμε servo – κινητήρα [13] και η ταχύτητα περιστροφής του υπολογίζεται σε μοίρες ανά λεπτό.

Εικόνα 2.6 [9]Ο σερβοκινητήρας MG995

Πλεονεκτήματα σερβοκινητήρα:

- Έχει τη δυνατότητα παραγωγής μεγάλων τιμών ροπής.
- Μπορεί να περιστρέφεται σε υψηλές ταχύτητες.
- Ελέγχεται εύκολα από τους σύγχρονους ελεγκτές ρομποτικών εφαρμογών.
- Διατίθεται σε μεγάλη ποικιλία εμπορικών μοντέλων.

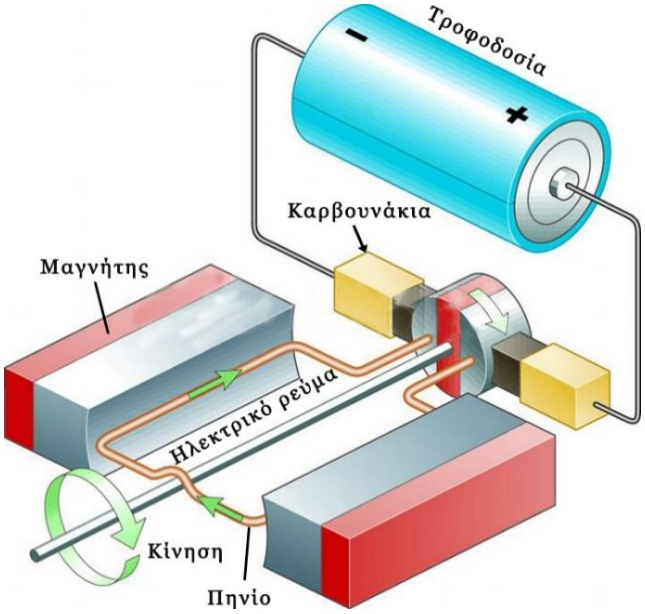
Μειονεκτήματα σερβοκινητήρα:

- Υψηλό κόστος

[9] https://robu.in/product/towerpro-mg995-metal-gear-servo-motor/

- Υψηλή κατανάλωση

Ο σερβοκινητήρας (servomotor) είναι ένας ηλεκτρικός κινητήρας συνεχούς ρεύματος εφοδιασμένος με αισθητήρα προσδιορισμού της θέσης του άξονα περιστροφής του. Ο ηλεκτρικός κινητήρας μετατρέπει την ηλεκτρική ενέργεια σε μηχανική ροπή. Στην παρακάτω εικόνα βλέπουμε τη βασική αρχή λειτουργίας ενός ηλεκτρικού κινητήρα. [14]



Εικόνα 2.7 Αρχή λειτουργίας σερβοκινητήρα.

Στη λειτουργία του κινητήρα μπορούμε να γυρίζουμε έναν άξονα από τις 0 μέχρι τις 180 μοίρες. Η λειτουργία αυτή είναι χρήσιμη, όπως και στην παρούσα εργασία, για να κινείται κάποιο μέρος ελεγχόμενα. Αν σταθεροποιήσουμε κάπου τη βάση του σερβοκινητήρα, μπορούμε να τον χρησιμοποιήσουμε, προκειμένου να πετύχουμε στην κατασκευή μας κίνηση σε εύρος 180 μοιρών. Ο κινητήρας συνδέεται με ένα καλώδιο στην πηγή, δεύτερο στη γείωση και τρίτο σε ένα pin του ελεγκτή, ώστε να μπορούμε να δίνουμε εντολές, για το πώς και πόσο θα στραφεί. Θα το μελετήσουμε αναλυτικότερα παρακάτω. [14]

2.2.2 Οδηγός κινητήρα Συνεχούς Ρεύματος

Ο οδηγός κινητήρα (motor driver) είναι μια ηλεκτρονική διάταξη που συχνά βρίσκει εφαρμογή σε ρομποτικά συστήματα και όχι μόνο. Η λειτουργία του είναι να λαμβάνει σήμα ελέγχου μικρού ρεύματος και να το μετατρέπει σε σήμα ελέγχου μεγάλου ρεύματος. Το σήμα ελέγχου μεγάλου ρεύματος στέλνεται σε κινητήρα και είναι ικανό να τον περιστρέψει. Τα χαρακτηριστικά της κάθε ηλεκτρονικής πλακέτας εξαρτώνται από το chip ελεγκτή που εμπεριέχεται σε αυτήν. Υπάρχουν πολλά είδη οδηγών κινητήρων και η επιλογή τους

εξαρτάται από τον τύπο και τα ονομαστικά χαρακτηριστικά του κινητήρα που θα περιστρέψει. Οδηγοί βηματικών κινητήρων, σερβοκινητήρων αλλά και κινητήρων συνεχούς ρεύματος είναι κάποια παραδείγματα οδηγών κινητήρων. Εμείς σε κάποιες εφαρμογές θα χρησιμοποιήσουμε τον οδηγό κινητήρων που εμπεριέχει το chip L298N, δηλαδή τον Dual Motor Driver Module L298N. [15]

Ο συγκεκριμένος οδηγός έχει τη δυνατότητα να περιστρέφει δύο κινητήρες συνεχούς τάσης ή βηματικούς κινητήρες. Η μικρότερη τάση τροφοδοσίας του είναι τα 5 VDC και η μέγιστη τα 35VDC. Λειτουργεί για ισχύς μικρότερες των 25 Watt.

Κάνουμε χρήση της συγκεκριμένης διάταξης για την αποφυγή βλάβης στην πλακέτα Arduino καθώς το μέγιστο ρεύμα που μπορεί να δώσει το Arduino είναι 20mA, ενώ ο οδηγός μας μπορεί να δώσει μέχρι 2A.

Εικόνα 2.8[10] Οδηγός κινητήρα Συνεχούς Τάσης

[10]

2.2.3 Αισθητήρες Υπερήχων

Ο αισθητήρας υπερήχων (Ultrasonic sensor) χρησιμοποιεί δύο διατάξεις, μία για να στείλει ένα υπερηχητικό σήμα και μία για να το λάβει. Έτσι, από τον χρόνο που μεσολαβεί από το να στείλει μέχρι να λάβει το σήμα πίσω μπορούμε να υπολογίσουμε την απόσταση στην οποία μπροστά μας βρίσκεται κάποιο αντικείμενο. Στην παρούσα εργασία χρησιμοποιήσαμε έναν κλασσικό τέτοιο αισθητήρα, τον HC-SR04 Ultrasonic Distance Sensor [16].

[10] https://www.makerlab-electronics.com/product/l298n-dual-h-bridge-motor-driver-module/

Εικόνα 2.9 Αισθητήρας υπερήχων.

Οι ακροδέκτες του αισθητήρα είναι οι εξής: ground (GND), Echo Pulse Output (ECHO), Trigger Pulse Input (TRIG), 5V Supply (Vcc). Δίνουμε τάση στη συσκευή χρησιμοποιώντας τον ακροδέκτη Vcc και το γειώνουμε με το GND. Με τη βοήθεια του Arduino που θα αναφέρουμε παρακάτω στέλνουμε ένα σήμα με προκαθορισμένο παλμό στον ακροδέκτη TRIG, το οποίο με τη σειρά του στέλνει υπερήχους απ' τη συσκευή, με τον ρυθμό παλμών που ορίσαμε στον ακροδέκτη TRIG. Αυτά τα κύματα υπερήχων, προσπίπτουν πάνω στα αντικείμενα που υπάρχουν σχετικά κοντά στη συσκευή και αντανακλώνται. Μ' αυτόν τον τρόπο τα ανακλώμενα κύματα γυρνούν στη συσκευή μας και μόλις τα ανιχνεύσει ο αισθητήρας μας, στέλνει ένα σήμα τάσεως 5V απ' τον ακροδέκτη ECHO πίσω στην πλακέτα μας.

```
digitalWrite(trigPin, HIGH)     //δίνω τάση, αρχίζω να στέλνω σήμα
delayMicroseconds(10)           //αφήνω λίγο χρόνο για την αποστολή
digitalWrite(trigPin, LOW)      //μηδενίζω την τάση, σταματά η αποστολή σήματος
```

Η αποστολή σήματος γίνεται δίνοντας τάση στο trigger, ενώ παίρνουμε τον χρόνο που πέρασε μέχρι να επιστρέψει το σήμα στο echo με την εντολή:

```
duration = pulseIn(echoPin, HIGH)
```

Ο υπολογισμός της απόστασης (σε cm) γίνεται βασιζόμενος στην ταχύτητα του ήχου, δηλαδή:

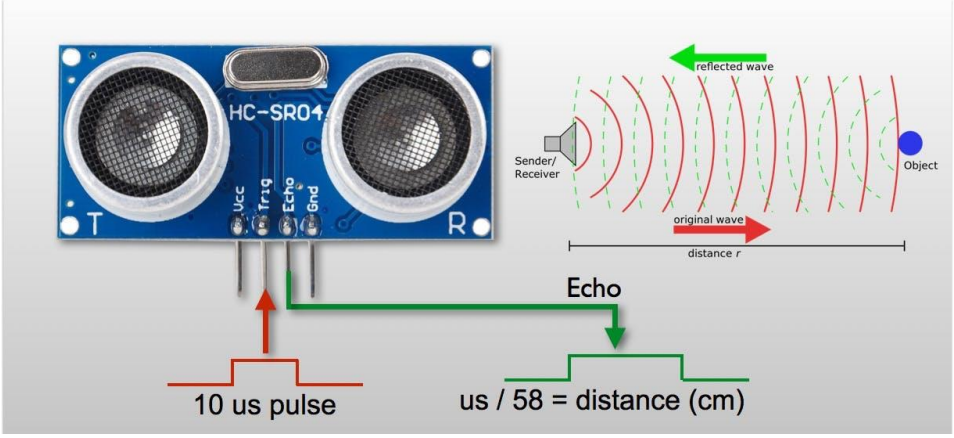
```
distance = duration/58.138      ή distance = duration/58.2
```

Η τιμή 58.138 έρχεται από τον κατασκευαστή και προκύπτει από την ταχύτητα του ήχου. Επειδή οι μετρήσεις είναι εμπειρικές, λεπτομερής παρατήρηση θα μας δώσει ίσως λεπτομερέστερες τιμές – για παράδειγμα άλλη εφαρμογή με τον ίδιο αισθητήρα προτείνει τη

διαίρεση με 58.2 (στην πρώτη περίπτωση έχουμε trigger (αποστολή) για 2μs και στη δεύτερη για 10μs). Γενικότερα, ο τύπος είναι:

distance = ((διάρκεια παλμού HIGH)(ταχύτητα ήχου: 340m/s)) / 2*

Διαιρούμε με το 2 επειδή το σήμα μας στάλθηκε και ελήφθη, δηλαδή έχει διανύσει διπλάσια απόσταση. Η διάρκεια του παλμού έχει δοθεί από την pulseIn σε microseconds (10-6 sec). [17]



Εικόνα 2.10 Αρχή λειτουργίας αισθητήρα υπερήχων.

[11]

2.2.4 Οπτικοί κωδικοποιητές θέσης

Οι οπτικοί κωδικοποιητές θέσης (encoders) διακρίνονται σε Περιστροφικούς και Γραμμικούς κωδικοποιητές και μετατρέπουν την κίνηση σε ψηφιακά σήματα. Η γραμμική μέτρηση είναι δυνατή και με περιστροφικούς κωδικοποιητές σε συνδυασμό με γρανάζι και οδοντωτό κανόνα ή τροχούς μέτρησης. Στο εμπόριο διατίθενται ζεύγη φωτοεκπομπών διόδων-φωτοτρανζίστορ σε ένα ενιαίο εξάρτημα σχήματος Π γνωστό ως φωτοανασχετήρας (photointerruptor). [18]

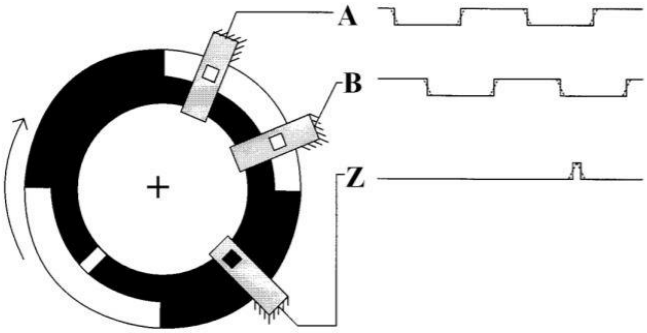
Εικόνα 2.11 α) Photointerruptor 1041 Omron και β) H5S-360 USDIGITAL

[11] https://www.hackster.io/powerberry/like-a-bat-with-hc-sr04-829486

Στις περισσότερες ρομποτικές εφαρμογές προτιμάται η χρήση αυξητικών κωδικοποιητών διότι από κατασκευαστικής απόψεως είναι πιο απλοί και κατά συνέπεια πιο φθηνοί. Οι αυξητικοί κωδικοποιητές παράγουν έναν προκαθορισμένο με ακρίβεια αριθμό παλμών ανά περιστροφή. Μετρούν τη γωνιακή ή γραμμική απόσταση κίνησης.

Ο κωδικοποιημένος δίσκος (περιστροφική και γραμμική κίνηση αντίστοιχα) διαιρείται σε χωριστά τμήματα που είναι διαδοχικά διαφανή ή αδιαφανή (εγκοπές). Μια πηγή φωτός με LED εκπέμπει μια παράλληλα προσανατολισμένη φωτεινή δέσμη που φωτίζει όλα τα τμήματα του κωδικοποιημένου δίσκου. Φωτοστοιχεία λαμβάνουν το διαμορφωμένο φως και το μετατρέπουν σε δύο ημιτονοειδή σήματα. Τα μαύρα κελιά αναπαριστούν τη μη ανίχνευση παλμού και έχουν τιμή ‘0’ ενώ τα λευκά (ανίχνευση παλμού) αναπαριστούν το ‘1’.

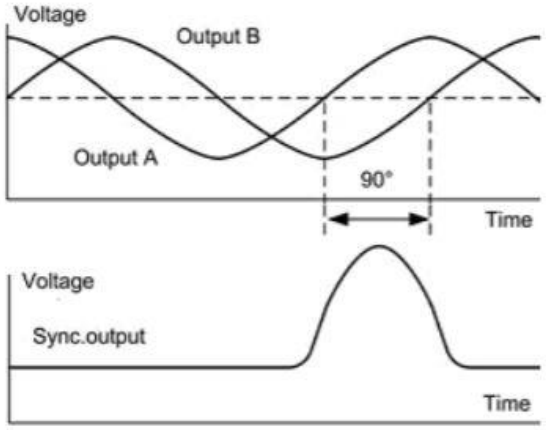
Τα σήματα A και B (βλέπε ακόλουθο σχήμα) είναι μετατοπισμένα το ένα από το άλλο κατά 90°. Η διαφορά φάσης μεταξύ των σημάτων επιτρέπει την αξιολόγηση της διεύθυνσης περιστροφής (αριστερά-δεξιά). Για να το πετύχουμε αυτό αρκεί να τοποθετήσουμε τα δύο ζεύγη φωτεινής πηγής- φωτοτρανζίστορ μετατοπισμένα μεταξύ τους κατά ¼ της απόστασης των οπών. Αλλιώς δύο δίσκους μετατοπισμένων κατά ¼ της απόστασης των οπών αντίστοιχα. Έτσι, αν η κίνηση είναι προς τα δεξιά η μια παλμοσειρά προηγείται της άλλης κατά T/4, όπου T η περίοδος του παλμού - ενώ το αντίθετο συμβαίνει όταν η κίνηση είναι προς τα αριστερά.



Εικόνα 2.12[12] Στοιχεία λειτουργίας αυξητικού κωδικοποιητή.

Για τον ορισμό της απόλυτης θέσης αναφοράς (απόλυτο μηδέν) οι κωδικοποιητές αυτού του τύπου φέρουν συνήθως και ένα τρίτο ίχνος (δείκτης αναφοράς), το οποίο αποτελείται από ένα μόνο αδιαφανή τομέα. Μια τυπική μορφή της εξόδου ενός τέτοιου αυξητικού γραμμικού κωδικοποιητή φαίνεται στο σχήμα που ακολουθεί:

[12] http://nefeli.lib.teicrete.gr/browse/sefe/hlk/2012/TritsonisAntonios,TertisGeorgios/attached-document-1342087665-580594-5982/TritsonisAntonios_TertisGeorgios2012.pdf

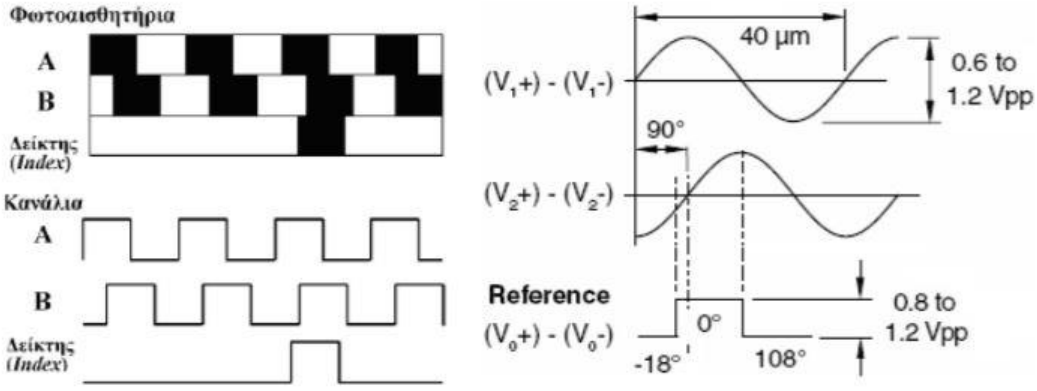


Εικόνα 2.13 [13]Έξοδος γραμμικού κωδικοποιητή.

Ο εν λόγω γραμμικός οπτικός κωδικοποιητής αυξητικού τύπου αποτελείται από έναν άξονα όπου έχουν ανοιχτεί μία ή συνηθέστερα δύο σειρές εγκοπών και από ένα σύστημα πηγής με δυο φωτοαισθητήρες. Κατά μήκος του άξονα ολισθαίνει ο παλμοδότης σε μορφή Π που 'αγκαλιάζει' την πλάκα. Από τη μια μεριά του Π υπάρχει φωτεινή πηγή (2 LEDs) που εκπέμπει λεπτή δέσμη φωτός και από την άλλη 2 φωτοδίοδοι ή φωτοτρανζίστορ. Η φωτοευαίσθητη μονάδα συνδέεται κατάλληλα σε κύκλωμα στην έξοδο του οποίου παίρνουμε είτε ψηλή τάση (όταν το Π βρίσκεται μπροστά από οπή) είτε χαμηλή (στην αντίθετη περίπτωση). Η εναλλαγή αυτή ονομάζεται παλμός - τάσης. Ο αριθμός των παλμών αντιπροσωπεύει τον αριθμό των οπών που έχει συναντήσει ο παλμοδότης κατά την κίνησή του. Επειδή η απόσταση μεταξύ των οπών είναι συγκεκριμένη, η απόσταση που έχει διανύσει ο παλμοδότης/δείκτης θα είναι:

απόσταση = *αριθμός παλμών* * *απόσταση μεταξύ οπών*.

[13] http://nefeli.lib.teicrete.gr/browse/sefe/hlk/2012/TritsonisAntonios,TertisGeorgios/attached-document-1342087665-580594-5982/TritsonisAntonios_TertisGeorgios2012.pdf



Εικόνα 2.14[14] Κωδικοποιημένος δίσκος αυξητικού κωδικοποιητή θέσης των 2 bits.

Η διακριτική ικανότητα του οργάνου είναι η απόσταση μεταξύ δύο οπών. Στο διάστημα αυτό ο απαριθμητής δίνει έναν παλμό, που είναι και το ελάχιστο που μπορεί να αναγνώσει ο απαριθμητής παλμών. [18]

Στις εφαρμογές που θα εξετάσουμε στην εργασία αυτή, χρησιμοποιούμε έναν κωδικοποιητή θέσης δύο καναλιών με δείκτη αναφοράς (index) καθώς και έναν φωτοανασχετήρα (photointerruptor) για την παρουσίαση του DC motor. [19], [20]

2.2.5 Κάμερα παρακολούθησης (camera)

Η κάμερα τροφοδοτεί την εικόνα της σε πραγματικό χρόνο μέσω ενός υπολογιστή. Κάθε εικόνα ανεξάρτητα από τον τύπο της είναι μια δισδιάστατη μήτρα αριθμών που ανάλογα με το είδος της εικόνας οι αριθμοί μπορούν να αντιπροσωπεύουν εντάσεις φωτεινότητας (όπως στην παρούσα εφαρμογή), αποστάσεις, είτε άλλες φυσικές ποσότητες.

Για τη λήψη ψηφιακής εικόνας χρησιμοποιείται μία κάμερα, ένας ψηφιοποιητής εικόνας και ένας υπολογιστής στον οποίο γίνεται η επεξεργασία της. Από εκεί, μέσω κατάλληλων διαδικασιών μπορούμε να περιγράψουμε τις μαθηματικές σχέσεις που λαμβάνει η κάμερα και να τις απεικονίσουμε στον υπολογιστή προκειμένου να κάνουμε προβολή εικόνας, αναγνώριση χρωμάτων και αναγνώριση αντικειμένων με σύστημα συντεταγμένων, όπως θα αναφερθούν στην παρακάτω εργασία.

Στην παρουσίαση του *Ball on Beam* χρησιμοποιούμε μία camera *Logitech c170* όπου την συνδέουμε μέσω θύρας USB στον υπολογιστή και επεξεργαζόμαστε τα δεδομένα που λαμβάνει από το περιβάλλον μέσω του *MatLab (Simulink).* [21] Κάμερα επίσης θα

[14] http://nefeli.lib.teicrete.gr/browse/sefe/hlk/2012/TritsonisAntonios,TertisGeorgios/attached-document-1342087665-580594-5982/TritsonisAntonios_TertisGeorgios2012.pdf

χρησιμοποιηθεί και στις εφαρμογές για μελλοντική έρευνα που θα αναφέρουμε στο Κεφάλαιο 4.

Εικόνα 2.15[15] Web Camera Logitech c170

2.3 Η βασική λειτουργία του Arduino

Το Arduino είναι μια «ανοικτού κώδικα» πλατφόρμα ηλεκτρονικών, βασισμένη σε ένα ευέλικτο και εύκολο στη χρήση hardware και software που προορίζεται για δημιουργία διαδραστικών αντικειμένων.

Είναι ένα ηλεκτρονικό κύκλωμα που βασίζεται στον μικροελεγκτή ATmega της Atmel και του οποίου όλα τα σχέδια, καθώς και το software που χρειάζεται για τη λειτουργία του, διανέμονται ελεύθερα και δωρεάν ώστε να μπορεί να κατασκευαστεί από τον καθένα. Αφού κατασκευαστεί, μπορεί να συμπεριφερθεί σαν ένας μικροσκοπικός υπολογιστής, αφού ο χρήστης μπορεί να συνδέσει επάνω του πολλαπλές μονάδες εισόδου/εξόδου και να προγραμματίσει τον μικροελεγκτή να δέχεται δεδομένα από τις μονάδες εισόδου, να τα επεξεργάζεται και να στέλνει κατάλληλες εντολές στις μονάδες εξόδου. Το Arduino βασίζεται στον ATmega328 [22], έναν 8-bit RISC μικροελεγκτή, τον οποίο χρονίζει στα 16MHz.

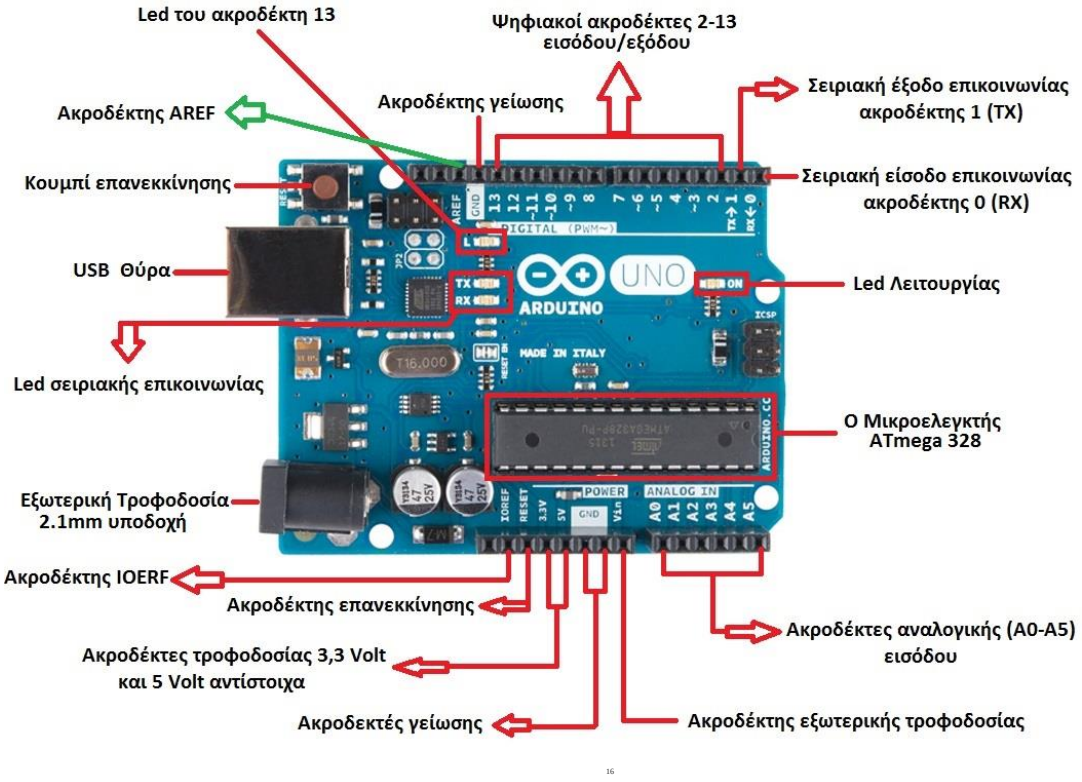
2.3.1 Είσοδοι – Έξοδοι

Το Arduino διαθέτει σειριακό interface. Ο μικροελεγκτής ATmega υποστηρίζει σειριακή επικοινωνία, την οποία το Arduino προωθεί μέσα από έναν ελεγκτή Serial-over-USB ώστε να συνδέεται με τον υπολογιστή μέσω USB. Η σύνδεση αυτή χρησιμοποιείται για τη μεταφορά των προγραμμάτων που σχεδιάζονται από τον υπολογιστή στο Arduino, αλλά και για αμφίδρομη επικοινωνία του Arduino με τον υπολογιστή μέσα από το πρόγραμμα την ώρα που

[15] https://amarpc.com/product/logitech-webcam-c170/

εκτελείται. Τη διαδικασία αυτή θα ακολουθήσουμε και εμείς στην παρούσα εργασία όπως θα εξηγηθεί πιο αναλυτικά στα παρακάτω κεφάλαια.

Επιπλέον, στην πάνω πλευρά του Arduino βρίσκονται 14 θηλυκά pin, αριθμημένα από 0 έως 13, που μπορούν να λειτουργήσουν ως ψηφιακές είσοδοι και έξοδοι. Λειτουργούν στα 5V και το καθένα μπορεί να παρέχει ή να δεχτεί το πολύ 40mA. Ως ψηφιακή έξοδος, ένα από αυτά τα pin μπορεί να τεθεί από το πρόγραμμα σε κατάσταση HIGH ή LOW, οπότε το Arduino θα ξέρει αν πρέπει να διοχετεύσει ή όχι ρεύμα στο συγκεκριμένο pin. Με αυτόν τον τρόπο μπορεί να ανάψει και να σβήσει ένα LED που έχει συνδεθεί στο συγκεκριμένο pin. Αν πάλι ρυθμιστεί ένα από αυτά τα pin ως ψηφιακή είσοδο μέσα από το πρόγραμμα, μπορούμε με την κατάλληλη εντολή να διαβάσουμε την κατάστασή του (HIGH ή LOW) ανάλογα με το αν η εξωτερική συσκευή που έχει συνδεθεί σε αυτό το pin διοχετεύει ή όχι ρεύμα στο pin. Μερικά από αυτά τα 14 pin, εκτός από ψηφιακές είσοδοι/έξοδοι έχουν και δεύτερη λειτουργία.



Εικόνα 2.16 [16] Στοιχεία του Arduino.

[16]https://projectmaniacs.wordpress.com/2014/11/29/%CE%B1%CE%BD%CE%AC%CE%BB%CF%85%CF%83%CE%B7-%CF%84%CE%BF%CF%85-arduino-uno/

Συγκεκριμένα:

- Τα pin 0 και 1 λειτουργούν ως RX και TX της σειριακής επικοινωνίας όταν το πρόγραμμα ενεργοποιεί τη σειριακή θύρα. Έτσι, όταν λόγου χάρη το πρόγραμμα στέλνει δεδομένα στη σειριακή πόρτα, αυτά προωθούνται και στη θύρα USB μέσω του ελεγκτή Serial-Over-USB, αλλά και στο pin 0 για να τα διαβάσει ενδεχομένως μια άλλη συσκευή.
- Τα pin 2 και 3 λειτουργούν ως εξωτερικά interrupt (interrupt 0 και 1 αντίστοιχα). Με άλλα λόγια, μπορεί να ρυθμιστεί μέσα από το πρόγραμμα, ώστε να λειτουργούν αποκλειστικά ως ψηφιακές είσοδοι στις οποίες όταν συμβαίνουν συγκεκριμένες αλλαγές, η κανονική ροή του προγράμματος σταματάει και εκτελείται μια συγκεκριμένη συνάρτηση.
- Τα pin 3, 5, 6, 9, 10 και 11 μπορούν να λειτουργήσουν ως ψευδοαναλογικές έξοδοι με το σύστημα PWM (Pulse Width Modulation). Έτσι, μπορεί να συνδεθεί λόγου χάρη ένα LED σε κάποιο από αυτά τα pin και να ελεγχθεί πλήρως η φωτεινότητά του με ανάλυση 8bit (256 καταστάσεις από 0-σβηστό ως 255-πλήρως αναμμένο). Το PWM δεν είναι πραγματικά αναλογικό σύστημα και θέτοντας στην έξοδο την τιμή 127, δεν σημαίνει ότι η έξοδος θα δίνει 2.5V αντί της κανονικής τιμής των 5V, αλλά ότι θα δίνει έναν παλμό που θα εναλλάσσεται με μεγάλη συχνότητα και για ίσους χρόνους μεταξύ των τιμών 0 και 5V.

Στην κάτω πλευρά του Arduino, με τη σήμανση ANALOG IN υπάρχει μια ακόμη σειρά από 6 pin, αριθμημένα από το 0 ως το 5. Το καθένα από αυτά λειτουργεί ως αναλογική είσοδος κάνοντας χρήση του ADC (Analog to Digital Converter) που είναι ενσωματωμένο στον μικροελεγκτή. Για παράδειγμα, μπορεί να τροφοδοτήσει ένα από αυτά με μια τάση την οποία μπορούμε να κυμαίνουμε με ένα ποτενσιόμετρο από 0V ως μια τάση αναφοράς Vref η οποία, αν δε γίνει κάποια αλλαγή είναι προρυθμισμένη στα 5V. Τότε, μέσα από το πρόγραμμα μπορούμε να «διαβάσουμε» την τιμή του pin ως ένα ακέραιο αριθμό ανάλυσης 10-bit, από 0 (όταν η τάση στο pin είναι 0V) μέχρι 1023 (όταν η τάση στο pin είναι 5V). Η τάση αναφοράς μπορεί να ρυθμιστεί με μια εντολή στο 1.1V, ή σε όποια τάση επιθυμούμε (μεταξύ 2 και 5V) τροφοδοτώντας εξωτερικά με αυτή την τάση το pin με τη σήμανση AREF που βρίσκεται στην απέναντι πλευρά της πλακέτας. Έτσι, αν τροφοδοτήσουμε το pin AREF με 3.3V και στη συνέχεια δοκιμάσουμε να διαβάσουμε κάποιο pin αναλογικής εισόδου στο οποίο εφαρμόζεται τάση 1.65V, το Arduino θα μας επιστρέψει την τιμή 512. Τέλος, καθένα από τα 6 αυτά pin, με κατάλληλη εντολή μέσα από το πρόγραμμα μπορεί να

μετατραπεί σε ψηφιακό pin εισόδου/εξόδου όπως τα 14 που βρίσκονται στην απέναντι πλευρά και τα οποία περιγράφηκαν πριν. Σε αυτή την περίπτωση τα pin μετονομάζονται από 0~5 σε 14~19 αντίστοιχα. [23]

2.3.2 Τροφοδοσία

Το Arduino μπορεί να τροφοδοτηθεί με ρεύμα είτε από τον υπολογιστή μέσω της σύνδεσης USB, είτε από εξωτερική τροφοδοσία που παρέχεται μέσω μιας υποδοχής φις των 2.1mm (θετικός πόλος στο κέντρο) και βρίσκεται στην κάτω-αριστερή γωνία του Arduino. Για να μην υπάρχουν προβλήματα, η εξωτερική τροφοδοσία πρέπει να είναι από 7 ως 12V και μπορεί να προέρχεται από έναν κοινό μετασχηματιστή του εμπορίου, από μπαταρίες ή οποιαδήποτε άλλη πηγή DC. Δίπλα από τα pin αναλογικής εισόδου, υπάρχει μια ακόμα συστοιχία από 6 pin με τη σήμανση POWER.

Η λειτουργία του καθενός έχει ως εξής: [23]

- Το πρώτο, με την ένδειξη RESET, όταν γειωθεί (σε οποιοδήποτε από τα 3 pin με την ένδειξη GND που υπάρχουν στο Arduino) έχει ως αποτέλεσμα την επανεκκίνηση του Arduino.
- Το δεύτερο, με την ένδειξη 3.3V, μπορεί να τροφοδοτήσει τα εξαρτήματα με τάση 3.3V.
- Το τρίτο, με την ένδειξη 5V, μπορεί να τροφοδοτήσει τα εξαρτήματα με τάση 5V. Ανάλογα με τον τρόπο τροφοδοσίας του ίδιου του Arduino, η τάση αυτή προέρχεται είτε άμεσα από τη θύρα USB (που ούτως ή άλλως λειτουργεί στα 5V), είτε από την εξωτερική τροφοδοσία, αφού αυτή περάσει από έναν ρυθμιστή τάσης για να την «φέρει» στα 5V.
- Το τέταρτο και το πέμπτο pin, με την ένδειξη GND, είναι γειώσεις.
- Το έκτο και τελευταίο pin, με την ένδειξη Vin έχει διπλό ρόλο. Σε συνδυασμό με το pin γείωσης δίπλα του, μπορεί να λειτουργήσει ως μέθοδος εξωτερικής τροφοδοσίας του Arduino. Αν όμως έχει ήδη συνδεδεμένη εξωτερική τροφοδοσία μέσω του φις, μπορεί να χρησιμοποιηθεί αυτό το pin για να τροφοδοτήσει εξαρτήματα με την πλήρη τάση της εξωτερικής τροφοδοσίας (7~12V), πριν αυτή περάσει από τον ρυθμιστή τάσης όπως γίνεται με το pin των 5V.

2.3.3 Arduino IDE και σύνδεση με τον υπολογιστή

Ό,τι χρειάζεται για τη διαχείριση του Arduino από τον υπολογιστή το παρέχει το Arduino IDE. 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. 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. [24], [25]

Το Arduino IDE παρέχει:

- ένα πρακτικό περιβάλλον για τη συγγραφή των προγραμμάτων (τα οποία ονομάζονται sketch στην ορολογία του Arduino) με συντακτική χρωματική σήμανση,
- αρκετά έτοιμα παραδείγματα,
- μερικές έτοιμες βιβλιοθήκες για προέκταση της γλώσσας και για να χειριζόμαστε εύκολα μέσα από τον κώδικά μας τα εξαρτήματα που συνδέουμε στο Arduino,
- τον compiler για τη μεταγλώττιση των sketch,
- ένα serial monitor που παρακολουθεί τις επικοινωνίες της σειριακής (USB), αναλαμβάνει να στείλει αλφαριθμητικά της επιλογής σας στο Arduino μέσω αυτής και είναι ιδιαίτερα χρήσιμο για το debugging των sketch
- και την επιλογή να ανεβάσουμε το μεταγλωττισμένο sketch στο Arduino.

Για τα δύο τελευταία χαρακτηριστικά βέβαια, το Arduino πρέπει να έχει συνδεθεί σε μια από τις θύρες USB του υπολογιστή και, λόγω του ελεγκτή Serial-over-USB, θα πρέπει να αναγνωριστεί από το λειτουργικό σύστημα ως εικονική σειριακή θύρα. Για τη σύνδεση θα χρειαστεί ένα καλώδιο USB.

Βασικές λειτουργίες του IDE:

	Έλεγχος του κώδικα για λάθη.
	Τερματισμός της σειριακής κονσόλας.
	Δημιουργία νέου έργου (sketch).
	Παρουσίαση μενού με όλα τα αποθηκευμένα έργα. Πατώντας σε ένα από αυτά ανοίγει για επεξεργασία.
	Αποθήκευση του έργου.
	Μεταγλώττιση του κώδικα και ανέβασμα του στο Arduino.
	Εμφάνιση της σειριακής κονσόλας. Αποστολή και λήψη δεδομένων που στάλθηκαν μέσω της σειριακής θύρας,

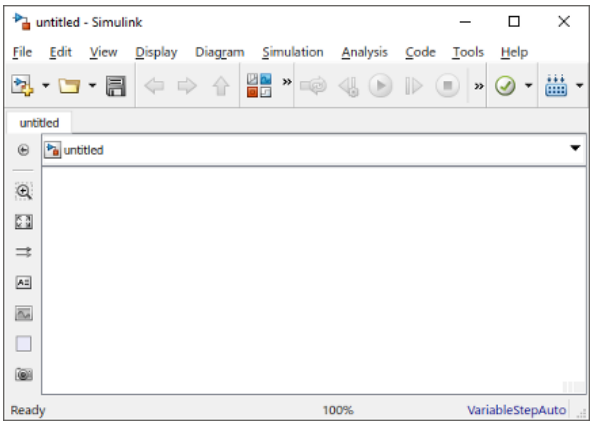
Εικόνα 2.17 Βασικές λειτουργίες του IDE.

2.4 Η βασική λειτουργία του Simulink

Μια γλώσσα υψηλού επιπέδου που χρησιμοποιείται για μαθηματικούς υπολογισμούς, ανάπτυξη αλγορίθμων, προσομοίωση, μοντελοποίηση και προτυποποίηση είναι το Matlab. Στην παρούσα πτυχιακή για τη μοντελοποίηση και προσομοίωση των εφαρμογών μας θα χρησιμοποιήσουμε το Simulink, το οποίο είναι ένα πρόγραμμα ενσωματωμένο στο Matlab, για την ανάλυση και την οπτικοποίηση δεδομένων.

Το Simulink είναι ένα λογισμικό πακέτο που επιτρέπει τη μοντελοποίηση, προσομοίωση και ανάλυση δυναμικών συστημάτων. Υποστηρίζει γραμμικά και μη γραμμικά συστήματα, μοντελοποιημένα σε συνεχή ή διακριτό χρόνο, ή ακόμη και υβριδικά συστήματα (εν μέρει μοντελοποιημένα σε συνεχή και εν μέρει σε διακριτό χρόνο) [18]. Υποστηρίζονται ακόμη συστήματα με τμηματικά διαφορετικούς χρόνους δειγματοληψίας.

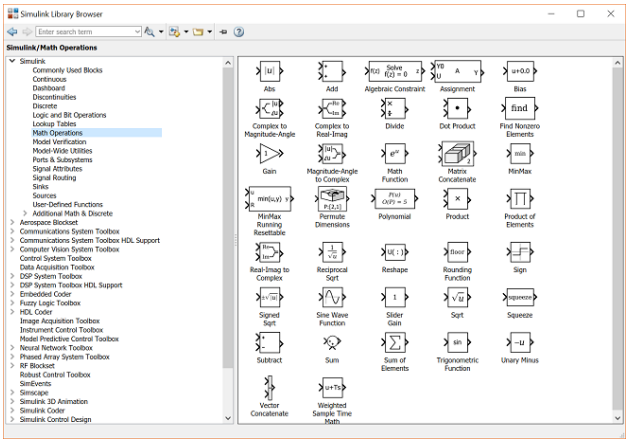
Για τη μοντελοποίηση, το Simulink παρέχει ένα γραφικό περιβάλλον διεπαφής (GUI) που επιτρέπει την κατασκευή μοντέλων ως δομικών διαγραμμάτων, χρησιμοποιώντας λειτουργίες click-and-drag του ποντικιού. Το Simulink περιλαμβάνει ένα πλήθος βιβλιοθηκών δομικών στοιχείων (blocks), τα βασικότερα από τα οποία είναι οι πηγές (sources), τα στοιχεία «απορρόφησης» (sinks), τα συνεχή γραμμικά στοιχεία, τα μη γραμμικά στοιχεία και τα στοιχεία σημάτων και συστημάτων. Είναι επίσης δυνατή η τροποποίηση και η δημιουργία νέων δομικών στοιχείων από τον χρήστη.



Εικόνα 2.18 Επιφάνεια εργασίας-σχεδίασης στο Simulink.

Τα μοντέλα Simulink είναι ιεραρχικά (ένα μοντέλο μπορεί να περιέχει μπλοκ τα οποία περιέχουν με τη σειρά τους άλλα μπλοκ), έτσι μπορούν να ιδωθούν σε διάφορα επίπεδα. Μετά τη δημιουργία ενός μοντέλου, είναι δυνατή η προσομοίωσή του, χρησιμοποιώντας μια από τις διάφορες μεθόδους ολοκλήρωσης που παρέχει το Simulink. Χρησιμοποιώντας παλμογράφους (scopes) και άλλα μπλοκ απεικόνισης, μέσω της βιβλιοθήκης που

εμπεριέχεται στο πρόγραμμα, είναι δυνατή η παρακολούθηση των αποτελεσμάτων της προσομοίωσης, καθώς αυτή εξελίσσεται. Επιπλέον, είναι δυνατή η εξαγωγή αποτελεσμάτων της προσομοίωσης στον χώρο εργασίας της Matlab για περεταίρω επεξεργασία. Είναι ακόμη δυνατή η χρήση του Simulink για προσομοίωση, αλλά και έλεγχο συστημάτων σε πραγματικό χρόνο, μέσω της εργαλειοθήκης πραγματικού χρόνου (Real Time Workshop). [27]



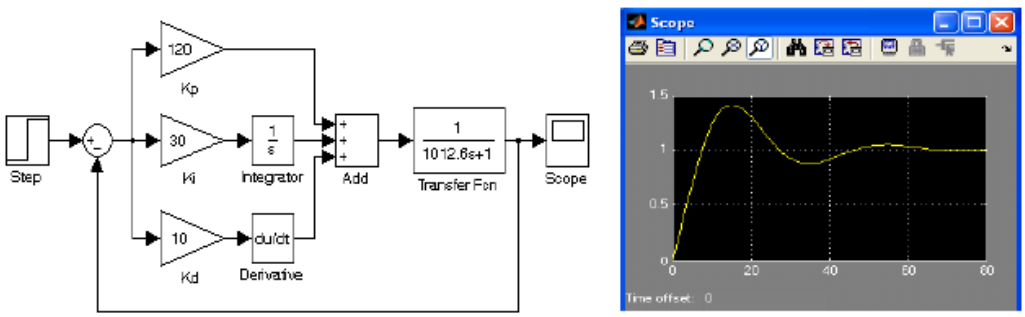
Εικόνα 2.19 Βιβλιοθήκες του Simulink.

Περαιτέρω πληροφορίες για τα δομικά στοιχεία, τις βιβλιοθήκες και ό,τι πληροφορίες σχετικά με το Simulink, είναι διαθέσιμες μέσω της βοήθειας της Matlab και του ιστοτόπου https://www.mathworks.com/ .

Αφότου καθοριστεί ένα πρότυπο ο χρήστης έχει τη δυνατότητα να δει τα αποτελέσματα της προσομοίωσης καθώς αυτή τρέχει. Επίσης, του είναι επιτρεπτό να αλλάξει τις παραμέτρους και να ξαναδεί τα αποτελέσματα της προσομοίωσης που τον ενδιαφέρουν.

Γενικά το Simulink, μας ενθαρρύνει να χτίσουμε πρότυπα από την αρχή ή να κάνουμε διαφοροποιήσεις σε υπάρχοντα, ώστε να πάρουμε διάφορα αποτελέσματα να τα αναλύσουμε και να τα απεικονίσουμε σε γραφική παράσταση.

[17]



Εικόνα 2.20 [17]Μοντέλο PID στο Simulink με scope.

[17] https://www.researchgate.net/figure/a-MATLAB-Simulink-model-of-the-liquid-level-control-system-with-PID-controller-and-b_fig3_313375177

Κεφάλαιο 3: Μεθοδολογία έρευνας

3.1 Επικοινωνία Arduino – Simulink

Σε όλες τις εφαρμογές της εργασίας μας χρησιμοποιούνται τα δύο παραπάνω εργαλεία. Το Simulink για τον έλεγχο των διεργασιών από τον χρήστη και το Arduino για τη μεσολάβηση των αισθητηρίων, διαφόρων κυκλωμάτων και μοτέρ με τον ηλεκτρονικό υπολογιστή, πράγμα που σημαίνει ότι το Simulink και το Arduino επικοινωνούν μεταξύ τους ταυτόχρονα. Ένας τρόπος επικοινωνίας μεταξύ τους είναι η σειριακή σύνδεση. Το Arduino έχει τη δυνατότητα σειριακής επικοινωνίας με υπολογιστή ή άλλες συσκευές μέσω USB ή μέσω των ψηφιακών ακίδων 0 (RX) και 1(TX), όπως αναφερθήκαμε. Εμείς θα κάνουμε χρήση του καλωδίου USB για ευκολία. Συνδέουμε, λοιπόν, το καλώδιο και ρυθμίζουμε το Arduino μέσω του κώδικα του ρυθμού μετάδοσης των δεδομένων (baud rate).

Παράλληλα σε ένα blank model του Simulink χρησιμοποιούμε τρία blocks.

Στο block “Serial Configuration” ρυθμίζουμε τις παραμέτρους της επικοινωνίας που θέλουμε να πετύχουμε όπως:

“Communication port”: τη θύρα usb που έχουμε συνδέσει την πλακέτα Arduino

“Baud rate”: τον ρυθμό μετάδοσης των bits

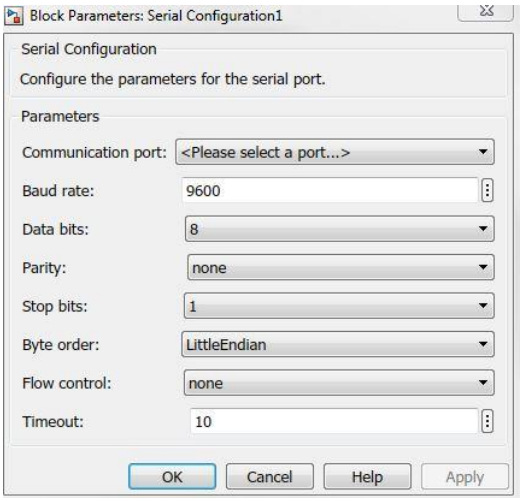
“Data bits”: τον αριθμό των bits που θα μεταδοθούν μέσω της σειριακής διασύνδεσης.

“Stop bits”: τον αριθμό των bits που θα υποδεικνύουν το τέλος ενός byte.

“Byte order”: τη διεύθυνση αποθήκευσης των byte

“Flow control”: τη διαχείριση της ταχύτητας μετάδοσης δεδομένων στη σειριακή θύρα.

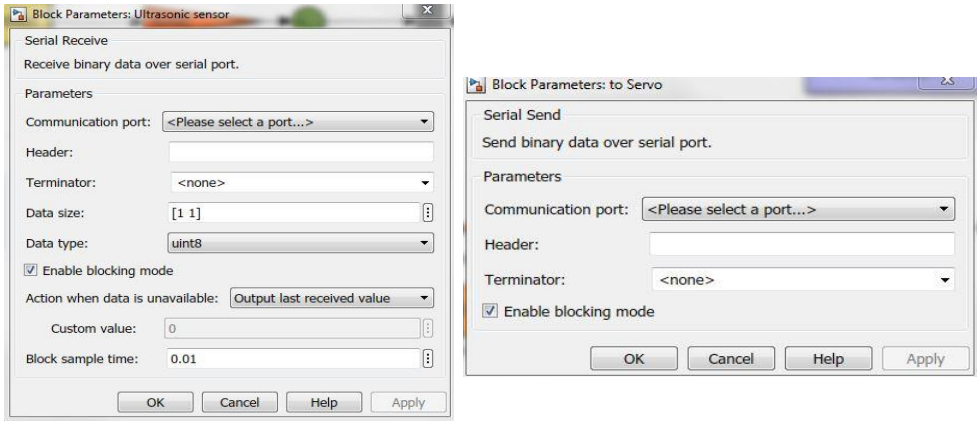
“Timeout”: το χρονικό διάστημα που το μοντέλο θα περιμένει για τα δεδομένα κατά τη διάρκεια κάθε βήματος χρόνου προσομοίωσης.



Εικόνα 3.1 Ρυθμίσεις "Serial Configuration Block"

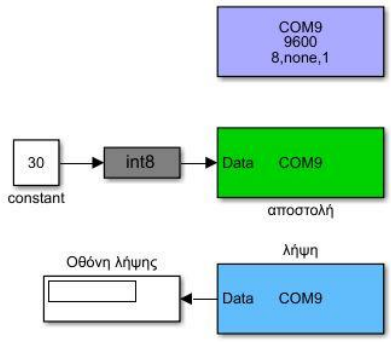
Στο Block "Serial Receive" ρυθμίζουμε τη θύρα USB που θα λαμβάνει δεδομένα ο υπολογιστής μας από το Arduino, τον χρόνο δειγματοληψίας καθώς και τον τύπο της μεταβλητής των ληφθέντων δεδομένων.

Στο block "Serial Send" απλά επιλέγουμε τη θύρα USB του υπολογιστή που θα στέλνει δεδομένα στο Arduino.



Εικόνα 3.2 Ρυθμίσεις "Serial Receive" (αριστερά) και "Serial Send" (δεξιά)

Όμως επειδή τα δεδομένα που λαμβάνει το Arduino είναι 8-μπιτοι αριθμοί μετατρέπουμε τα δεδομένα προς αποστολή σε 8-μπιτους ακεραίους. Αυτό γίνεται προσθέτοντας το block "Data Type Conversion" .



Εικόνα 3.3 Block "Data Type Conversion"

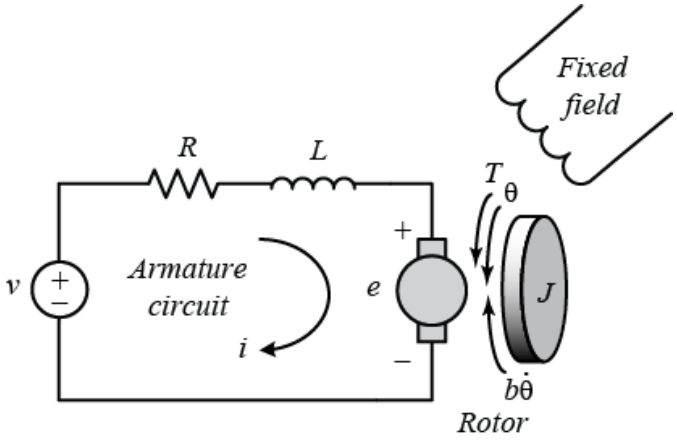
3.2 DC motor εφαρμογές

3.2.1 DC motor – διαδικασία προσδιορισμού συνάρτησης μεταφοράς

Αν καθίσουμε και αναλογιστούμε απλά συστήματα μηχατρονικής το πρώτο που ίσως μας έρθει στο μυαλό θα είναι ο ηλεκτρικός κινητήρας ή αλλιώς μοτέρ. Ο κινητήρας είναι ένα στοιχείο που παίρνει ηλεκτρική ενέργεια και αποδίδει κινητική ενέργεια. Υπάρχουν πολλοί τύποι κινητήρων. Κάποιοι από αυτούς είναι οι κινητήρες συνεχούς τάσης, οι κινητήρες εναλλασσόμενης τάσης, οι σερβοκινητήρες -αναφερθήκαμε παραπάνω- και οι βηματικοί κινητήρες.

Στο παρόν μέρος της εργασίας θα μελετήσουμε ένα Σύστημα Αυτόματου Ελέγχου όπου η διεργασία μας θα είναι ένας ηλεκτροκινητήρας συνεχούς ρεύματος (DC motor).

Στο block «Διεργασίας» θα πάρει θέση η συνάρτηση μεταφοράς του κινητήρα μας. Οι κινητήρες τέτοιου τύπου έχουν ισοδύναμο ηλεκτρικό κύκλωμα, αυτό που φαίνεται στην εικόνα:



Εικόνα 3.4[18] Κύκλωμα DC μοτέρ.

[18] http://ctms.engin.umich.edu/CTMS/index.php?example=MotorPosition&section=SystemModeling

Η αρχή λειτουργίας τους είναι μια πηγή συνεχούς τάσης V που τροφοδοτεί με ρεύμα i έναν αντιστάτη R, ένα πηνίο L και το κύκλωμα οπλισμού e. Η αύξηση του ρεύματος i προκαλεί αύξηση της μαγνητικής ροής που βρίσκεται στο εσωτερικό του κινητήρα με αποτέλεσμα την αύξηση της ταχύτητας περιστροφής του ρότορα και του άξονα του μοτέρ, αφού είναι συνδεδεμένα μεταξύ τους και υποθέτουμε ότι είναι άκαμπτοι. Επίσης υποθέτουμε ότι η ροπή τριβής είναι ανάλογη της γωνιακής ταχύτητας του άξονα.

Οπότε συμπεραίνουμε ότι το σύστημα του ηλεκτροκινητήρα συνεχούς τάσης παίρνει σαν είσοδο τάση και βγάζει σαν έξοδο κίνηση στον άξονα.

Μένει τώρα να προσδιορίσουμε την εξίσωση της διεργασίας-μοτέρ.

Η ροπή του κινητήρα T είναι ανάλογη με τη σταθερά ροπής Kt και το ρεύμα του κινητήρα i.

$$T = Kt\, i$$

Η emf (e) είναι ανάλογη της γωνιακής ταχύτητας του άξονα bθ με την ηλεκτροκινητική σταθερά δύναμης Kb.

$$e = Kb\, \dot{\theta}$$

Όμως στο σύστημα SI η σταθερά ροπής Kt και η ηλεκτροκινητική σταθερά δύναμης Kb είναι ίσες Kt=Kb και γι’ αυτό τον λόγο θα χρησιμοποιήσουμε τον όρο K και θα εννοούμε τόσο το Kt όσο και το Kb.

Χρησιμοποιώντας τον δεύτερο νόμο του Kirchhoff στο παραπάνω κύκλωμα, έχουμε τις παρακάτω εξισώσεις.

$$J\ddot{\theta} + b\dot{\theta} = Ki$$

$$L\frac{di}{dt} + Ri = V - K\dot{\theta}$$

Όπου J είναι η ροπή αδρανείας του ρότορα.

Εφαρμόζουμε λοιπόν μετασχηματισμό Laplace στις εξισώσεις:

$$s\,(Js+b)\,\Theta(s) = K\,I(s)$$

$$(Ls+R)\,I(s) = V(s) - Ks\,\Theta(s)$$

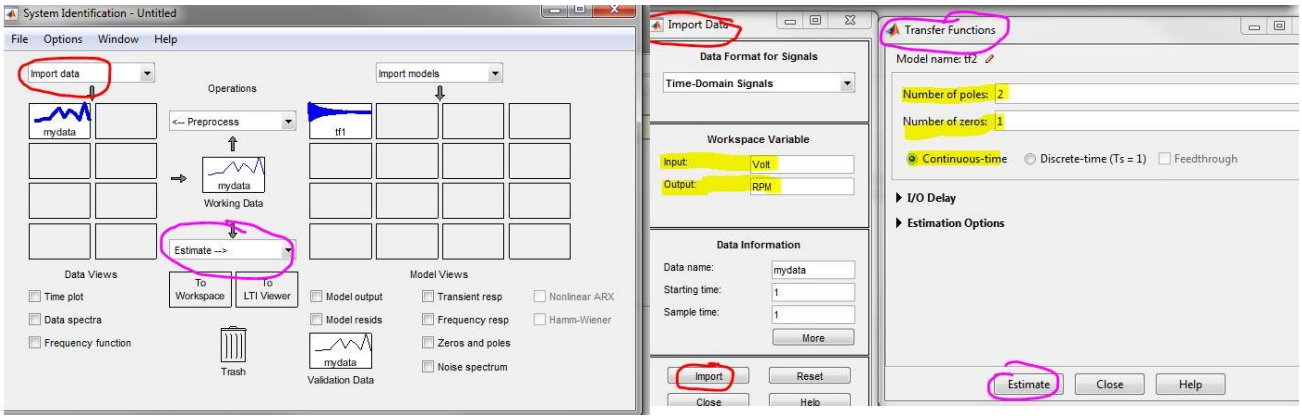
Απαλείφουμε το ***I(s)*** και θεωρούμε ως έξοδο του συστήματος την ταχύτητα περιστροφής και ως είσοδο την τάση του οπλισμού.

$$P(s) = \frac{\dot{\Theta}(s)}{V(s)} = \frac{K}{(Js+b)(Ls+R)+K^2} \quad \frac{rad/sec}{V}$$

Επειδή όμως θα προσπαθήσουμε να κάνουμε έλεγχο θέσης του άξονα περιστροφής και όχι έλεγχο ταχύτητας, απλά πρέπει να διαιρέσουμε την παραπάνω λειτουργία με s

$$\frac{\Theta(s)}{V(s)} = \frac{K}{(Js+b)(Ls+R)+K^2} \quad \frac{rad}{V}$$

Αν, λοιπόν, αντικαταστήσουμε τις μεταβλητές K, J, L, R και b στην παραπάνω σχέση με τα στοιχεία οποιουδήποτε μοτέρ, θα πάρουμε την αντίστοιχη συνάρτηση μεταφοράς. [28]
Εμείς επειδή δε γνωρίζουμε τα στοιχεία του μοτέρ που έχουμε στα χέρια μας και άρα δεν μπορούμε να βρούμε τη συνάρτηση μεταφοράς του συστήματος, ακολουθούμε την εξής διαδικασία:



Εικόνα 3.5 Απόκτηση συνάρτηση μεταφοράς από System Identification

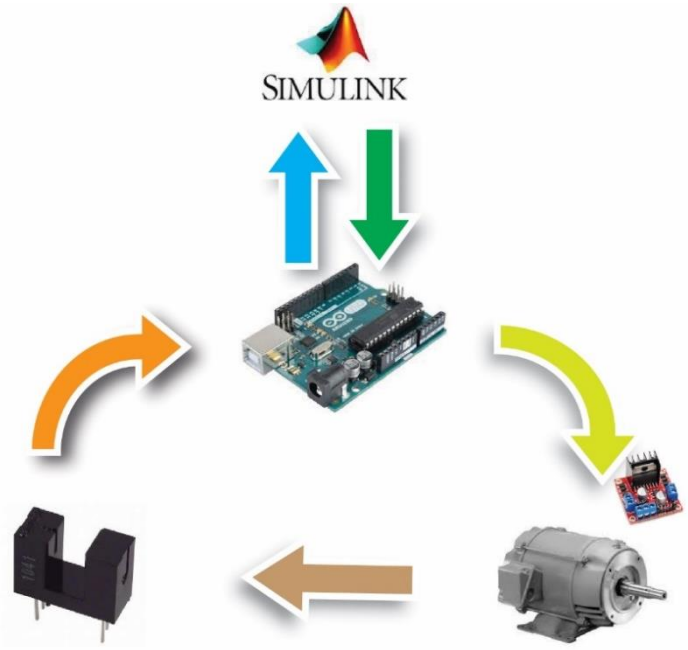
Μετράμε τις στροφές του κινητήρα μας με διαφορετική τάση εισόδου και καταγράφουμε τις τιμές με τη μορφή διανυσμάτων στο Workspace του Matlab. Ανοίγουμε την εφαρμογή System Identification του Matlab για να προσδιορίζουμε τη συνάρτηση μεταφοράς. Η συνάρτησή μας είναι η παρακάτω:

$$\frac{5658\,s + 20440}{s^2 + 2.456 * 10^{-5}s + 2.81}$$

3.2.2 DC motor με photo interrupter

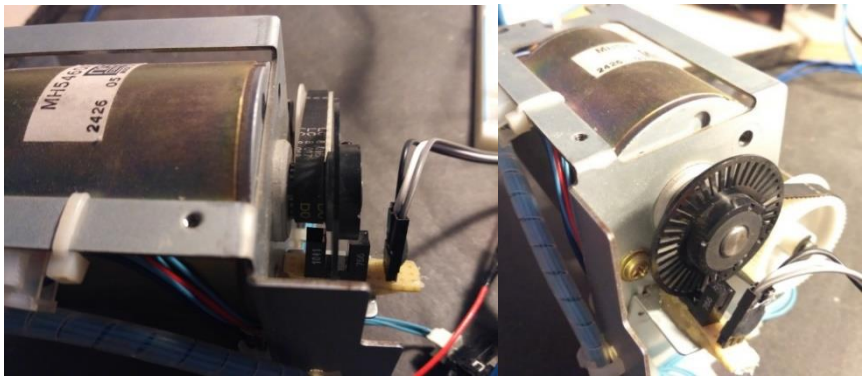
Η ανάδραση που χρησιμοποιούμε στην παρούσα διεργασία είναι η γωνία περιστροφής του άξονα του μοτέρ. Για να επιτευχθεί η μέτρηση της γωνίας χρησιμοποιούμε έναν κωδικοποιητή φωτεινών διακοπών ή αλλιώς έναν photo interrupter encoder, τεσσάρων ακίδων. Η αρχή λειτουργίας τέτοιων αισθητήρων εξηγήθηκε παραπάνω. Εμείς θα χρησιμοποιήσουμε το 1041 της εταιρίας Omron. Επίσης θα χρησιμοποιηθεί ένα κύκλωμα αντιστάσεων για τη λειτουργία του αισθητήρα και μία πλακέτα Arduino Uno Rev3 για την καταγραφή των δεδομένων του αισθητήρα.

Το παρακάτω block διάγραμμα δίνει μία γενική επεξήγηση του τρόπου λειτουργίας του συστήματος.



Εικόνα 3.5 Διάγραμμα γενικής λειτουργίας DC motor-photo interrupter encoder

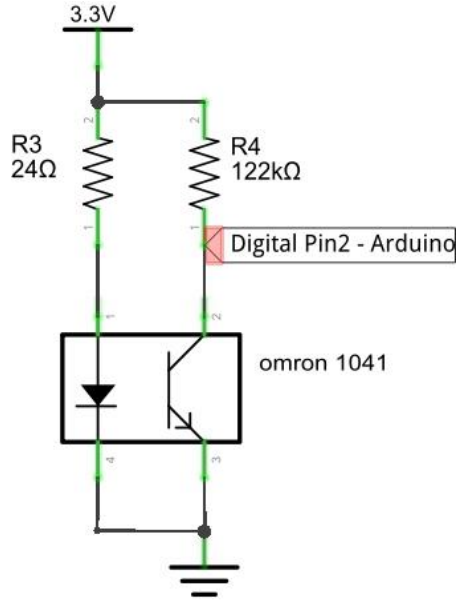
Το Simulink με την εκκίνησή του, θα στέλνει σειριακά στο Arduino την επιθυμητή γωνία. Το Arduino μέσω του κώδικά του θα δίνει κίνηση στο μοτέρ, χάρη στην πλακέτα γέφυρα Η, και θα διαβάζει παράλληλα τις εναλλαγές των εγκοπών που στέλνει ο αισθητήρας κατά την περιστροφή του άξονα του μοτέρ. Έπειτα θα στέλνει στο Simulink σειριακά τον αριθμό των εναλλαγών. Το Simulink θα μετατρέπει τον αριθμό αυτό σε γωνία και θα την τοποθετεί στην ανάδραση του συστήματος.



Εικόνα 3.6 Φωτογραφία DC motor-photo interrupter encoder

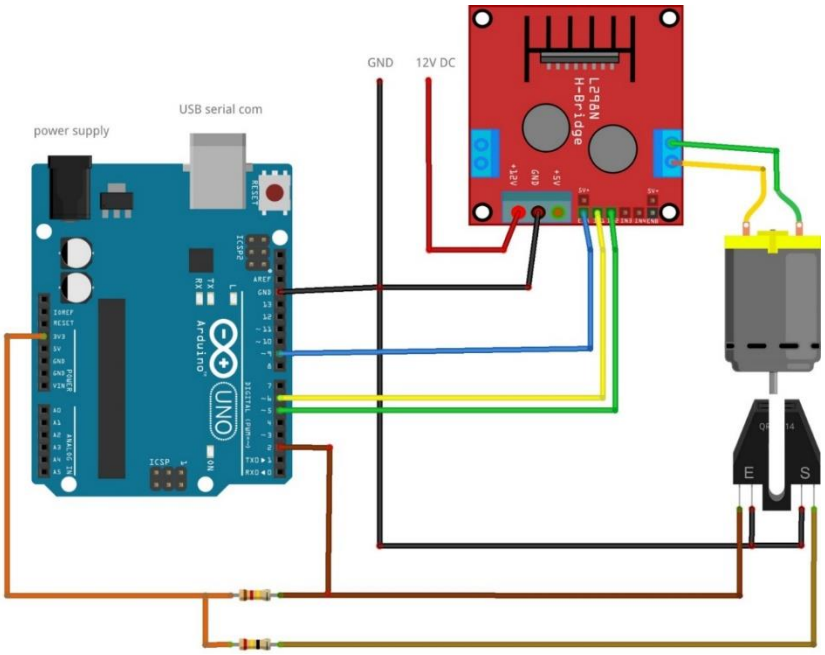
Ο αισθητήρας είναι τοποθετημένος κατά μήκος του άξονα του μοτέρ με σκοπό την καταμέτρηση των εναλλαγών των εγκοπών που προκύπτουν από το διάτρητο δίσκο που είναι τοποθετημένος πάνω στον άξονα.

Η πλακέτα Arduino τις εναλλαγές του αισθητήρα τις διαβάζει ως HIGH και LOW τιμές διαβάζοντας την τάση στο ψηφιακό πινάκι 2 (digital pin 2) της πλακέτας Arduino. Η τιμή HIGH αναφέρεται για τάση 5 Volt και η τιμή LOW για τάση 0 Volt. Τι θα γίνει όμως αν το ψηφιακό πινάκι παίρνει ενδιάμεσες τιμές τάσης; Τις τιμές, λοιπόν, που είναι από 3V έως 5V το Arduino τις θεωρεί ως 5 Volt και τις τιμές που είναι από 0V έως 3V τις θεωρεί 0 Volt.
Για να μπορέσουν να γίνουν εφικτές αυτές οι τιμές τάσεων συνδέουμε τον αισθητήρα με την παρακάτω διάταξη αντιστάσεων.



Εικόνα 3.7 Ηλεκτρικό κύκλωμα photo interrupter encoder

Οι τιμές των αντιστάσεων R3 και R4 προέκυψαν πειραματικά, αφού για την αδιάφανη περιοχή του κωδικοποιημένου δίσκου, το ψηφιακό πινάκι 2 λαμβάνει 3,19V και για τη διαφανή περιοχή 2,04V, πράγμα που βοηθά στον εύκολο διαχωρισμό HIGH και LOW τιμών. Η παρακάτω εικόνα δείχνει τη συνδεσμολογία όλου του hardware συστήματος.



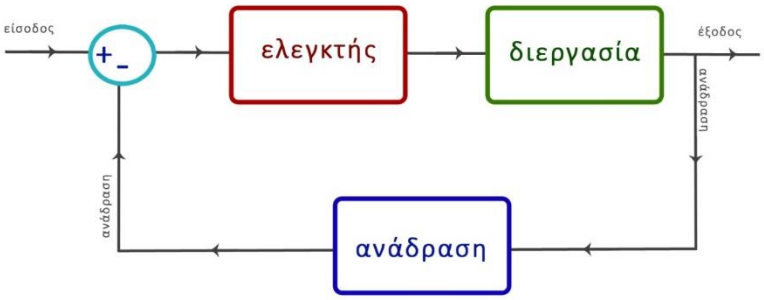
Εικόνα 3.8 Συνδεσμολογία hardware συστήματος DC motor – photo interrupter encoder.

Όταν το Arduino θέλει να περιστραφεί το μοτέρ δίνει στο ψηφιακό πινάκι 5 την τιμή «1», στο ψηφιακό πινάκι 6 την τιμή «0» και στο ψηφιακό πινάκι 9 μια τιμή PWM. Η τιμή PWM έχει εύρος 0-255 και ορίζει το πλάτος της τάσης εξόδου της γέφυρας Η, που είναι και η τάση εισόδου του μοτέρ.

Το κύκλωμα αντιστάσεων τροφοδοτείται με τάση 3.3V από την πλακέτα Arduino. Η πλακέτα Arduino και η γέφυρα Η (H-bridge ή motor driver module) τροφοδοτούνται με συνεχή τάση 12V DC από εξωτερικό τροφοδοτικό. Τέλος το Arduino συνδέεται σειριακά με καλώδιο USB, με υπολογιστή που τρέχει το αρχείο Simulink.

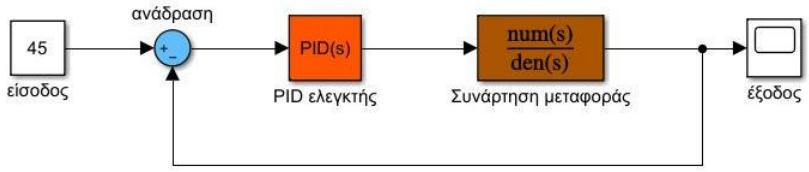
3.2.3 DC motor – Simulink και παραμετροποίηση ελεγκτή

Όπως αναφερθήκαμε στην ενότητα 1.2, τα Συστήματα Αυτομάτου Ελέγχου (Σ.Α.Ε.) έχουν πέντε βασικά μέρη. 1) Είσοδος, 2) Έξοδος, 3) Ελεγκτής, 4) Διεργασία και 5) Ανάδραση όπως στην παρακάτω εικόνα:



Εικόνα 3.9 Κλειστό σύστημα Σ.Α.Ε.

Δημιουργούμε, λοιπόν, στο Simulink ένα Blank model αρχείο και κατασκευάζουμε το θεωρητικό Σ.Α.Ε..

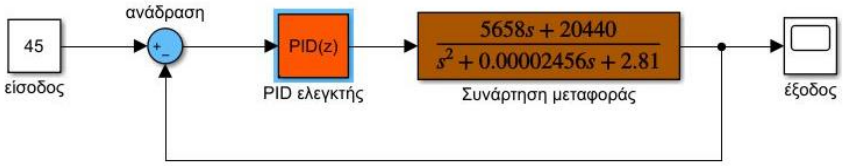


Εικόνα 3.10 Θεωρητικό κλειστό κύκλωμα Σ.Α.Ε. στο Simulink.

Η είσοδος του συστήματος θα ορίζεται από το block "constant" που θα περιέχει τη γωνία σε μοίρες που επιθυμούμε να περιστραφεί ο άξονας του μοτέρ.

Στο block "sum" προσθέτουμε την είσοδο του συστήματος και αφαιρούμε την ανάδραση που έρχεται από την έξοδο.

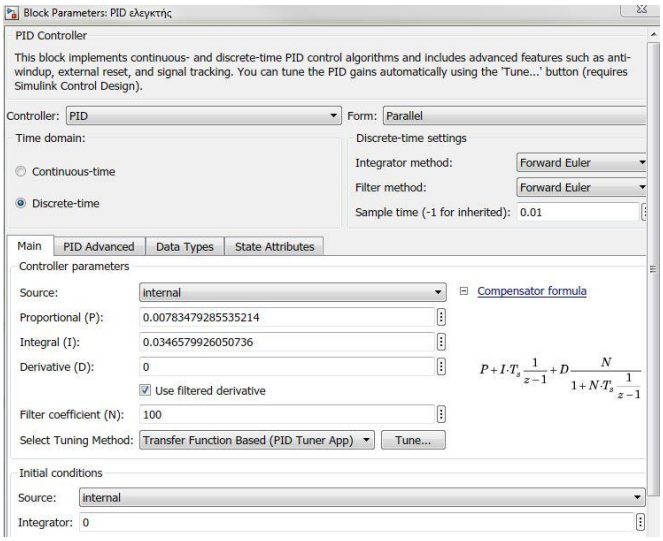
Στο block "Transfer function" γράφουμε τη συνάρτηση μεταφοράς της διεργασίας που υπολογίσαμε με την πιο πάνω διαδικασία.



Εικόνα 3.11 Κλειστό κύκλωμα ΣΑΕ με συνάρτηση μεταφοράς

Την έξοδο την οδηγούμε στο block "scope" για να παρατηρήσουμε τα αποτελέσματα της διεργασίας με τον ελεγκτή.

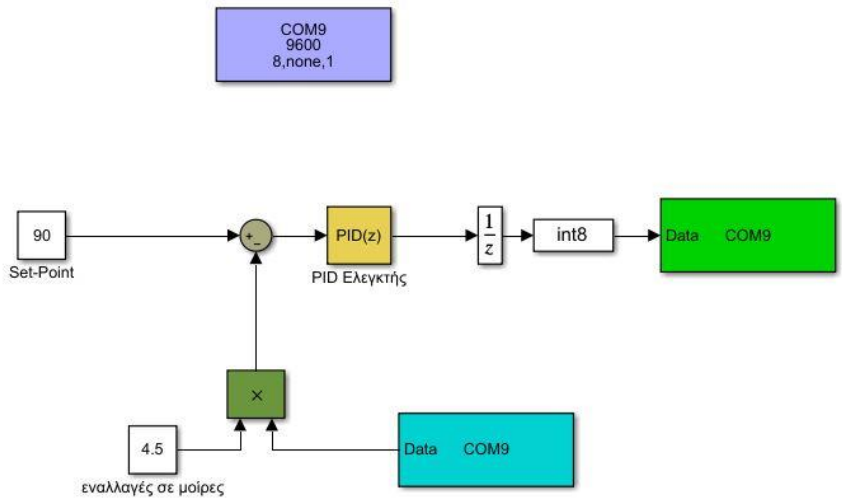
Και τέλος στο block "PID Controller" επιλέγουμε Discrete-Time (διακριτός χρόνος) και ορίζουμε το Sample time (περίοδος δειγματοληψίας) που επιθυμούμε. Ύστερα πατάμε tune για να επιλέξουμε την απόκριση στο χρόνο και την ακρίβεια που θέλουμε. Έτσι το block υπολογίζει από μόνο του τις παραμέτρους του ελεγκτή.



Εικόνα 3.12 PID Controller block

Αφού υπολογίσουμε τις παραμέτρους του PID ελεγκτή ξεκινάμε να διαμορφώνουμε το blank model προσθέτοντας και αφαιρώντας blocks, ώστε να πετύχουμε επικοινωνία με την πλατφόρμα Arduino.

Προσθέτουμε λοιπόν τα τρία βασικά blocks που εξυπηρετούν τη σειριακή επικοινωνία μεταξύ Simulink και Arduino, "Serial Configuration", "Serial Send", "Serial Receive", "Unit Delay", "Data Type Conversion" και ρυθμίζουμε το κάθε block όπως είπαμε στη 3.1 ενότητα. Επίσης προσθέτουμε ένα block επιπλέον "Constant" και ένα "Product", και αφού αφαιρέσουμε το block "Transfer Function" συνδέουμε τα blocks όπως στην εικόνα.



Εικόνα 3.13 Ολοκληρωμένο μοντέλο DC motor με σειριακή επικοινωνία.

Το block "Serial Receive" θα επιστέφει στο simulink το πλήθος των εναλλαγών (της αδιάφανης περιοχής στη διαφανή) που αντιλαμβάνεται ο αισθητήρας κατά την κίνηση του

μοτέρ. Επειδή όμως στην ανάδραση θέλουμε γωνία σε μονάδα μέτρησης «μοίρες» θα πρέπει να μετατρέψουμε το πλήθος των εναλλαγών σε μοίρες. Η μετατροπή γίνεται πολλαπλασιάζοντας το πλήθος με την τιμή 4.5. Η τιμή αυτή υπολογίζεται ως εξής. Μετράμε πόσες εγκοπές έχει ο κωδικοποιημένος δίσκος. Πολλαπλασιάζουμε με το 2 για να υπολογίσουμε τις εναλλαγές του αισθητήρα σε μία πλήρη περιστροφή. Ύστερα αυτό που θα βρούμε το διαιρούμε από το 360 που είναι μοίρες ενός κύκλου. Έτσι υπολογίζουμε πόσες μοίρες αντιστοιχούν σε κάθε εναλλαγή.

$$\frac{360^{\circ}}{k * 2}$$

Όπου k: ο αριθμός των εγκοπών του δίσκου.

Στη δική μας εφαρμογή επειδή ό δίσκος έχει 40 εγκοπές προκύπτει το 4,5.

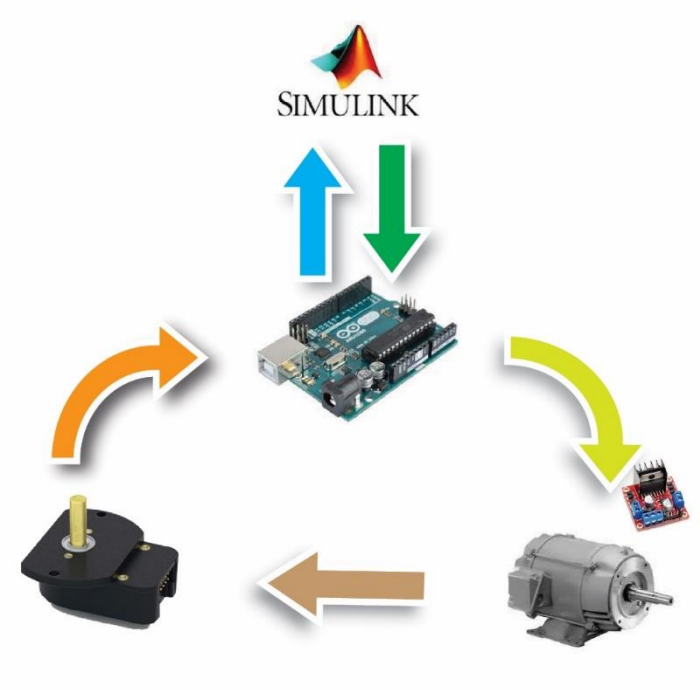
Με αυτόν τον τρόπο επιτυγχάνουμε να έχουμε στην ανάδραση τις μοίρες που έχει περιστραφεί ο άξονας του μοτέρ.

Το block “είσοδος” το μετονομάζουμε σε “set-point” και θα επιλέγει ο χρήστης την γωνία (σε μοίρες) που επιθυμεί να περιστραφεί ο άξονας του μοτέρ κατά την εκτέλεση του Simulink. Το “sum” αφαιρεί από την επιθυμητή γωνία την πραγματική που λαμβάνουμε από την ανάδραση και έτσι δημιουργείται το error, δηλαδή το πόσο θέλει ακόμα να περιστραφεί ο άξονας για να επιτευχθεί η επιθυμητή γωνία. Ο ελεγκτής ανάλογα με τις παραμέτρους του, ρυθμίζει την τάση που θα λάβει το μοτέρ για να ολοκληρωθεί η περιστροφή.

3.2.4 DC motor με Optical encoder

Παρατηρήσαμε στην προηγούμενη εφαρμογή ότι ο αισθητήρας μας μετράει με ακρίβεια 4,5 μοίρες. Αυτό το προκαλεί ο κωδικοποιημένος δίσκος που περιστρέφεται μαζί με τον άξονα. Όσες περισσότερες εγκοπές έχει ο δίσκος τόσο μεγαλώνει η ακρίβεια του αισθητήρα. Αν για παράδειγμα ο δίσκος μας είχε 180 εγκοπές τότε ο αισθητήρας θα είχε ακρίβεια 1 μοίρα με κάθε εναλλαγή.

Εδώ, λοιπόν, θα ασχοληθούμε με την αντικατάσταση του αισθητήρα μας με άλλο αισθητήρα. Ο νέος μας αισθητήρας λειτουργεί με παρόμοιο τρόπο με τον προηγούμενο και είναι κωδικοποιητής οπτικού άξονα και συγκεκριμένα ο H5S-360 της USDIGITAL. Ο τρόπος λειτουργίας του αναφέρθηκε παραπάνω. Το block διάγραμμα της γενικής λειτουργίας είναι το παρακάτω.



Εικόνα 3.14 Διάγραμμα γενικής λειτουργίας DC motor-Optical encoder

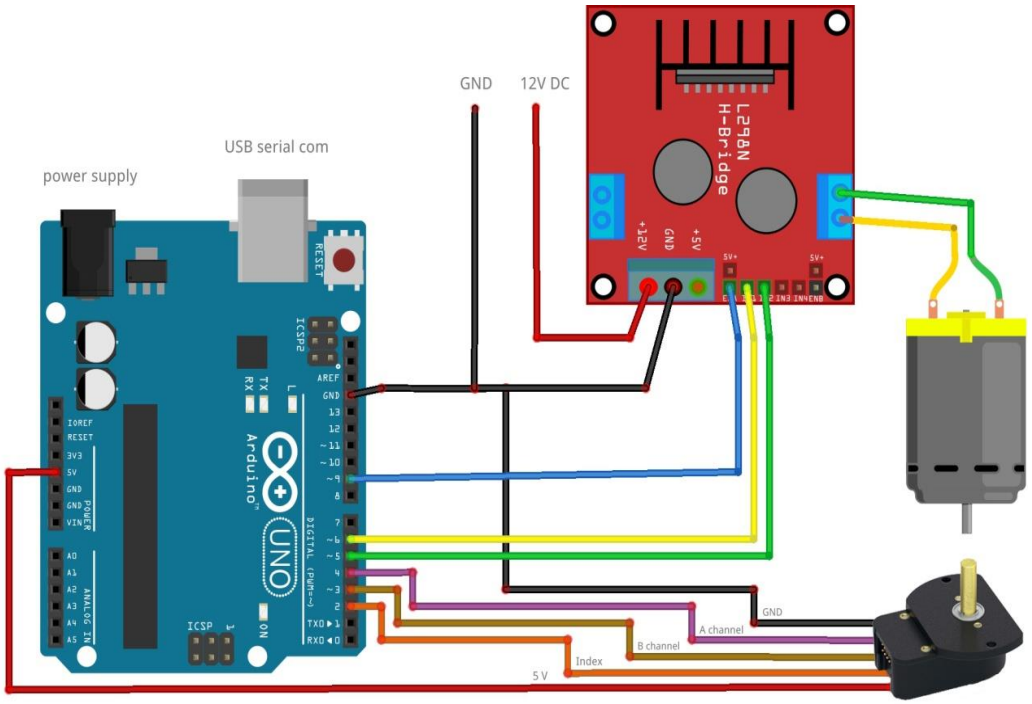
Το Simulink με την εκκίνησή του, στέλνει σειριακά στο Arduino την επιθυμητή γωνία. Το Arduino μέσω του νέου του κώδικα δίνει κίνηση στο μοτέρ, χάρη στην πλακέτα γέφυρα Η. Το Arduino παράλληλα διαβάζει από τον νέο αισθητήρα τις εναλλαγές που προκύπτουν κατά την περιστροφή του κωδικοποιημένου δίσκου αλλά και τη φορά. Έπειτα στέλνει στο Simulink σειριακά τον αριθμό των εναλλαγών. Το Simulink μετατρέπει τον αριθμό αυτό σε γωνία και την τοποθετεί στην ανάδραση του συστήματος.

Ο νέος αισθητήρας δε χρειάζεται κωδικοποιημένο δίσκο και κύκλωμα αντιστάσεων καθώς τους έχει ενσωματωμένους. Επομένως θα αφαιρέσουμε τον encoder της εταιρείας Omron καθώς και τον διάτρητο δίσκο και τον άξονα του μοτέρ και θα συνδέσουμε τον νέο κωδικοποιητή.

Η πλακέτα Arduino τις εναλλαγές του αισθητήρα τις καταλαβαίνει ως HIGH και LOW τιμές διαβάζοντας την τάση στα ψηφιακά πινάκια 2,3 και 4 (digital pin 2, digital pin 3 και digital pin 4) της πλακέτας Arduino, αντίστοιχα. Η τιμή HIGH αναφέρεται για τάση 5 Volt και η τιμή LOW για τάση 0 Volt. Το ψηφιακό πινάκι 4 είναι συνδεδεμένο με το κανάλι Α του αισθητήρα και το ψηφιακό πινάκι 3 είναι συνδεδεμένο με το κανάλι Β του αισθητήρα.

Ο ακροδέκτης index του αισθητήρα είναι συνδεδεμένος με το ψηφιακό πινάκι 2 (digital pin 2) του Arduino για να ορίζει τη θέση.

Παρακάτω φαίνεται η συνδεσμολογία του κάθε στοιχείου του συστήματός μας:



Εικόνα 3.15 Συνδεσμολογία hardware συστήματος DC motor - Optical encoder.

Το Simulink αρχείο παραμένει το ίδιο με αυτό της προηγούμενης υποενότητας, γιατί η λήψη των δεδομένων από την πλακέτα του Arduino είναι ίδια. Το μόνο που αλλάζει είναι ο κώδικας του Arduino καθώς έχει διαφορετική σύνδεση με τον κωδικοποιητή και διαφορετικό τρόπο λειτουργίας.

3.2.5 Αποτελέσματα και συμπεράσματα αποκρίσεων για το DC motor

Όπως ήδη γνωρίζουμε από τη θεωρία των συστημάτων αυτόματου ελέγχου, ο ελεγκτής είναι μία από τις βασικότερες μονάδες. Στα συστήματα ανοικτού βρόχου η είσοδος του ελεγκτή είναι η ίδια με την είσοδο του συστήματος (set-point), ενώ στα συστήματα κλειστού βρόγχου είναι το σφάλμα (error) που παράγεται από τη διαφορά του set-point και της εξόδου. Στόχος του είναι η σταθεροποίηση της εξόδου σε οποιαδήποτε διαταραχή με μηδενισμό του σφάλματος, επίσης ο ελεγκτής πετυχαίνει καλύτερες επιδόσεις ταχύτητας, ευστάθειας και ακρίβειας στο σύστημα ανάλογα με την επιλογή ελεγκτή που θα κάνουμε αλλά και τη ρύθμισή τους.

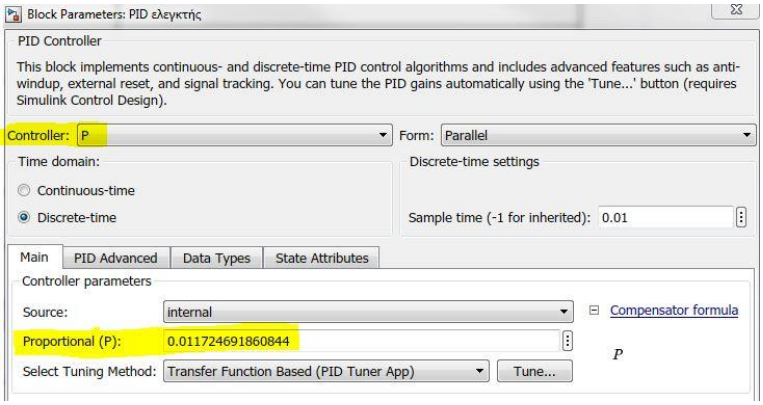
Υπάρχουν πολλοί τύποι ελεγκτών, αλλά εμείς θα ασχοληθούμε με τους ελεγκτές βηματικής απόκρισης. Αυτοί είναι οι Αναλογικοί ελεγκτές (ή Proportional controller), οι Ολοκληρωτικοί ελεγκτές (ή Integral controller), οι Διαφορικοί ελεγκτές (ή Differential controller), οι

Αναλογικοί-Ολοκληρωτικοί ελεγκτές, οι Αναλογικοί-Διαφορικοί ελεγκτές και οι Αναλογικοί-Ολοκληρωτικοί-Διαφορικοί ελεγκτές.

Στην εφαρμογή μας, χρησιμοποιήσαμε όλους τους παραπάνω ελεγκτές ακολουθώντας τη διαδικασία tuning του Simulink και λάβαμε τις αποκρίσεις τους με μέριμνα την ταχύτητα, την ακρίβεια και την ευστάθεια του συστήματος.

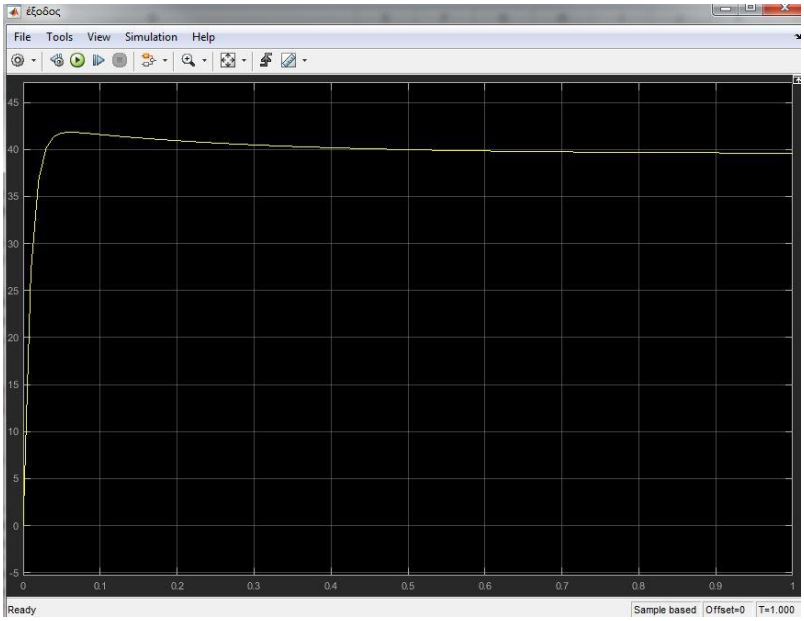
3.2.5.1 Αναλογικός ελεγκτής P

Ρυθμίσαμε το block "ελεγκτής" να χρησιμοποιεί μόνο την παράμετρο P. Με τη λειτουργία tune επιλέξαμε την ταχύτερη απόκριση στο χρόνο (sec.) και η τιμή της παραμέτρου P που προκύπτει είναι 0.011724691860844.



Εικόνα 3.16 Ρυθμίσεις P ελεγκτή για DC motor

Για είσοδο 40 μοίρες κατά την εκτέλεση του Simulink λάβαμε την παρακάτω απόκριση.

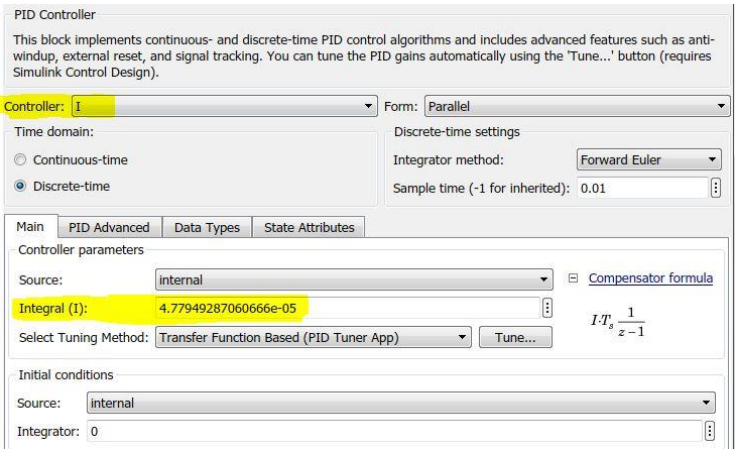


Εικόνα 3.17 Απόκριση με P ελεγκτή στο DC motor

Παρατηρούμε ότι η απόκριση έχει μία υπερύψωση από 0.03 έως 0.446 δευτερόλεπτα που δεν ξεπερνά το 20% της εισόδου, δηλαδή την τιμή 48. Από 0.446 και έπειτα η έξοδος έχει την τιμή της εισόδου.

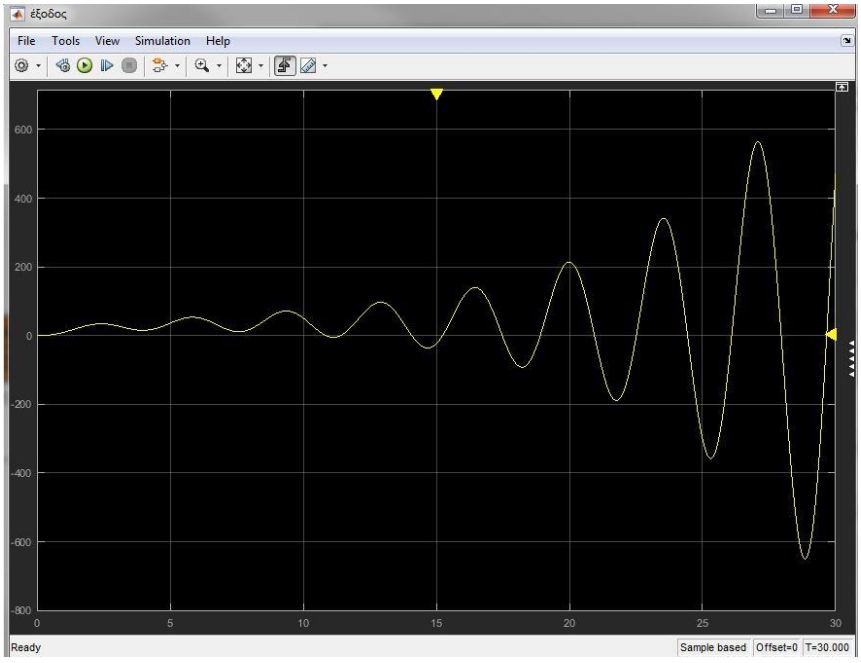
3.2.5.2 Ολοκληρωτικός ελεγκτής I

Ρυθμίσαμε το block "ελεγκτής" να χρησιμοποιεί μόνο την παράμετρο I. Με τη λειτουργία tune επιλέξαμε μία τυχαία απόκριση στον χρόνο (sec.) και η τιμή της παραμέτρου I που προκύπτει είναι 4.77949287060666e-05.



Εικόνα 3.18 Ρυθμίσεις I ελεγκτή για DC motor

Για την ίδια είσοδο 40 μοιρών κατά την εκτέλεση του Simulink λάβαμε την παρακάτω απόκριση.



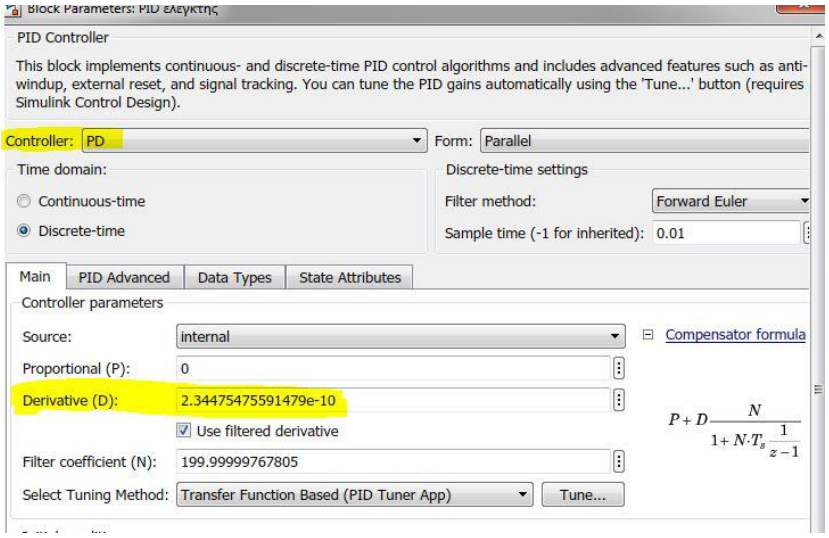
Εικόνα 3.19 Απόκριση με I ελεγκτή στο DC motor

Παρατηρούμε ότι το σύστημά μας με αυτόν τον ελεγκτή παρουσιάζει αστάθεια, διότι με το πέρας του χρόνου δεν ισορροπεί ποτέ στην τιμή της εισόδου (40 μοίρες). Αντίθετα όλο και αυξάνεται. Συμπεραίνουμε ότι ο Ολοκληρωτικός ελεγκτής I είναι ακατάλληλος για το σύστημά μας.

3.2.5.3 Διαφορικός ελεγκτής D

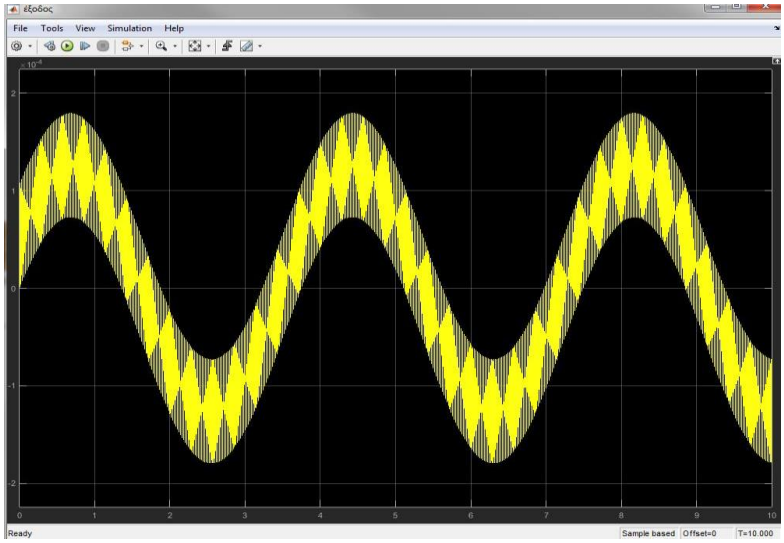
Ακολουθήσαμε τη συνηθισμένη διαδικασία για να ρυθμίσουμε το block "ελεγκτής" να χρησιμοποιεί μόνο την παράμετρο D. Επειδή το block του Simulink δε μας δίνει τη δυνατότητα αυτή, επιλέξαμε έναν PD ελεγκτή, κάναμε tune για μια απόκριση που η παράμετρος P ήταν πολύ κοντά στο μηδέν. Ύστερα μηδενίσαμε την παράμετρο P και κρατήσαμε μόνο την παράμετρο D που ήταν:

Παράμετρος D:	**2.34475475591479e-10**



Εικόνα 3.20 Ρυθμίσεις D ελεγκτή για DC motor

Για την ίδια είσοδο 40 μοιρών κατά την εκτέλεση του Simulink λάβαμε την παρακάτω απόκριση.

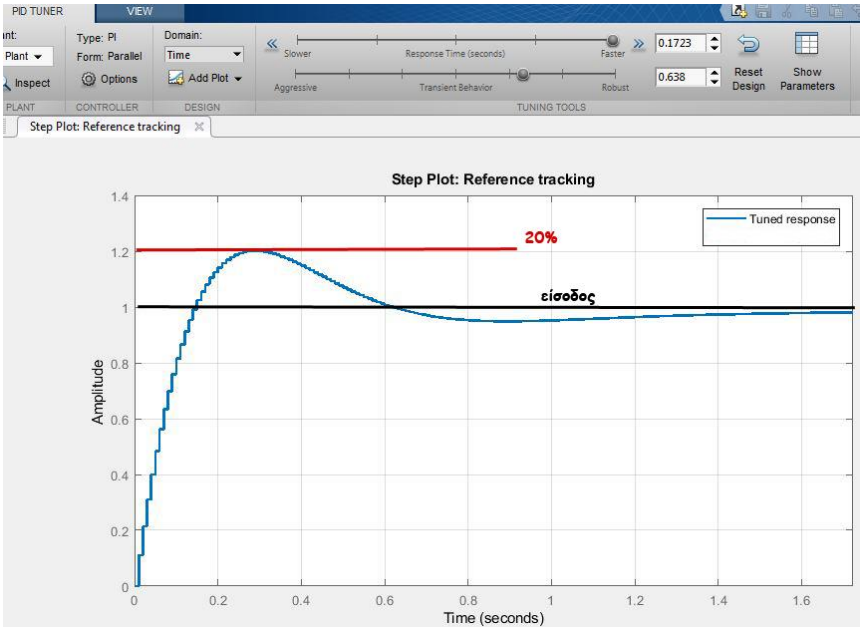


Εικόνα 3.21 Απόκριση με D ελεγκτή στο DC motor

Πράγμα που δείχνει ότι και πάλι το σύστημά μας δεν είναι ευσταθές και ότι ο Διαφορικός ελεγκτής είναι και αυτός ακατάλληλος.

3.2.5.4 Αναλογικός - Ολοκληρωτικός ελεγκτής PI

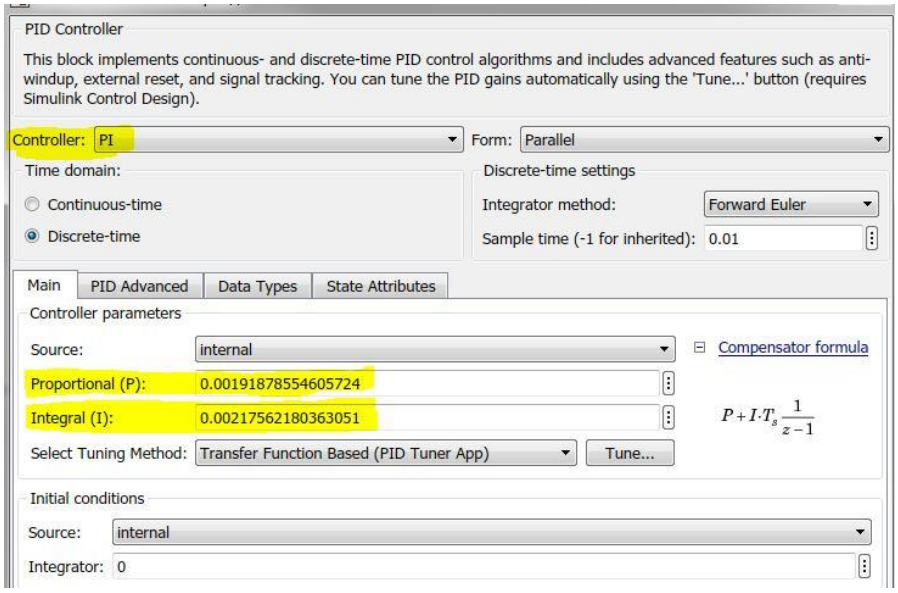
Ρυθμίσαμε το block "ελεγκτής" να χρησιμοποιεί μόνο την παράμετρο PI. Με τη λειτουργία "tune" επιλέξαμε την ταχύτερη απόκριση στον χρόνο (sec.) και τη μεταβατική συμπεριφορά, ώστε η απόκριση να μην ξεπερνά το 20% της εισόδου.



Εικόνα 3.22 Η λειτουργία "tune" για PI ελεγτή

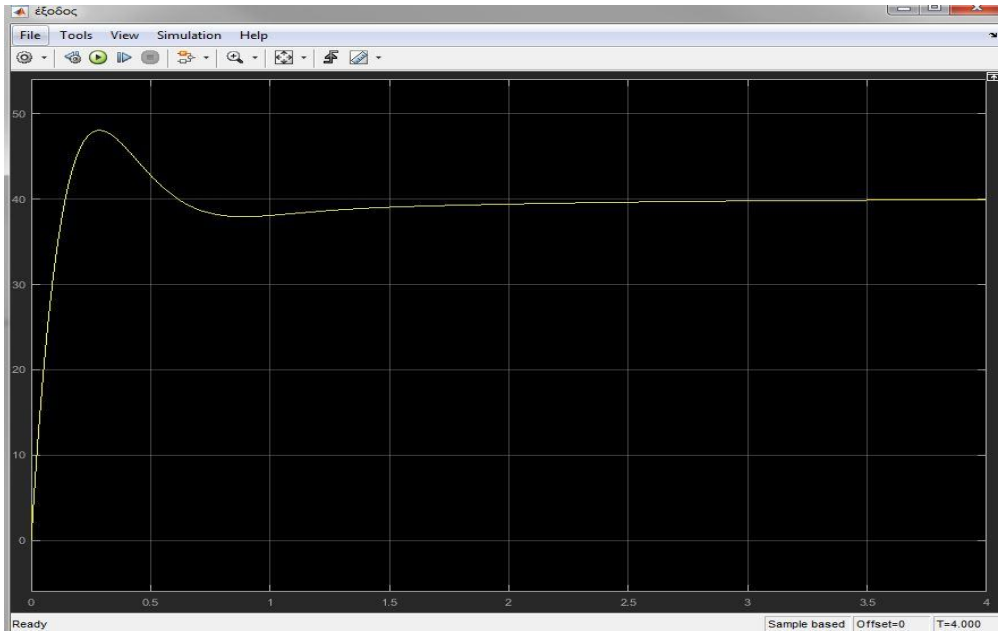
Οι τιμές των παραμέτρων PI που προκύπτουν:

Παράμετρος P:	**0.00191878554605724**
Παράμετρος I:	**0.00217562180363051**



Εικόνα 3.23 Ρυθμίσεις PI ελεγκτή για DC motor

Για είσοδο 40 μοιρών κατά την εκτέλεση του Simulink λάβαμε την παρακάτω απόκριση.



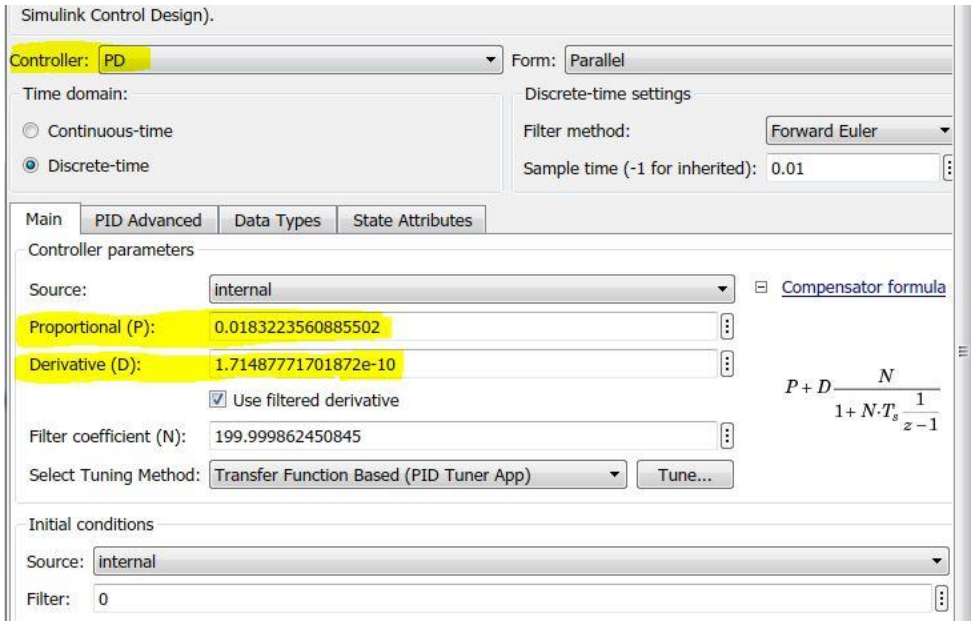
Εικόνα 3.24 Απόκριση με PI ελεγκτή στο DC motor

Παρατηρούμε ότι η απόκριση έχει μία υπερύψωση από 0.143 έως 0.619 δευτερόλεπτα που φτάνει στο 20% της εισόδου, δηλαδή την τιμή 48. Από 0.619 έως 3 δευτερόλεπτα η απόκριση τείνει να έχει την τιμή της εισόδου.

3.2.5.5 Αναλογικός - Διαφορικός ελεγκτής PD

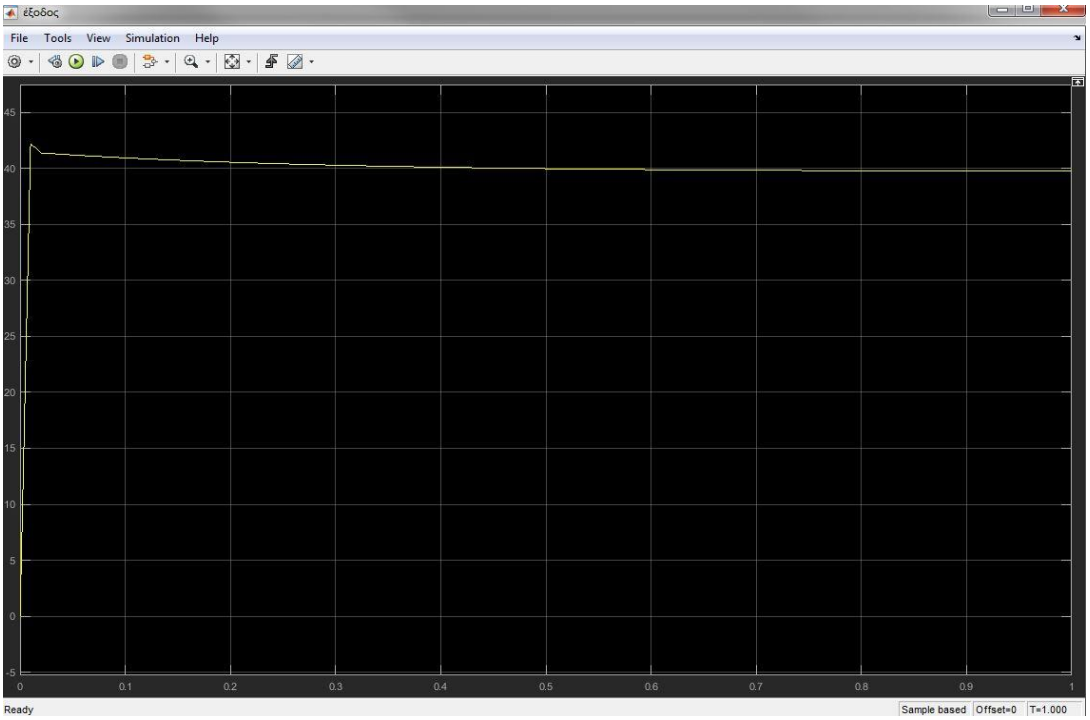
Ρυθμίσαμε το block “ελεγκτής” να χρησιμοποιεί μόνο την παράμετρο PD. Με τη λειτουργία tune επιλέξαμε την ταχύτερη απόκριση στον χρόνο (sec.) και τη μεταβατική συμπεριφορά, ώστε να πετύχουμε τη μεγαλύτερη ακρίβεια. Οι τιμές των παραμέτρων PD είναι:

Παράμετρος P:	**0.0183223560885502**
Παράμετρος D:	**1.71487771701872e-10**



Εικόνα 3.25 Ρυθμίσεις PD ελεγκτή για DC motor

Για είσοδο 40 μοιρών κατά την εκτέλεση του Simulink λάβαμε την παρακάτω απόκριση.



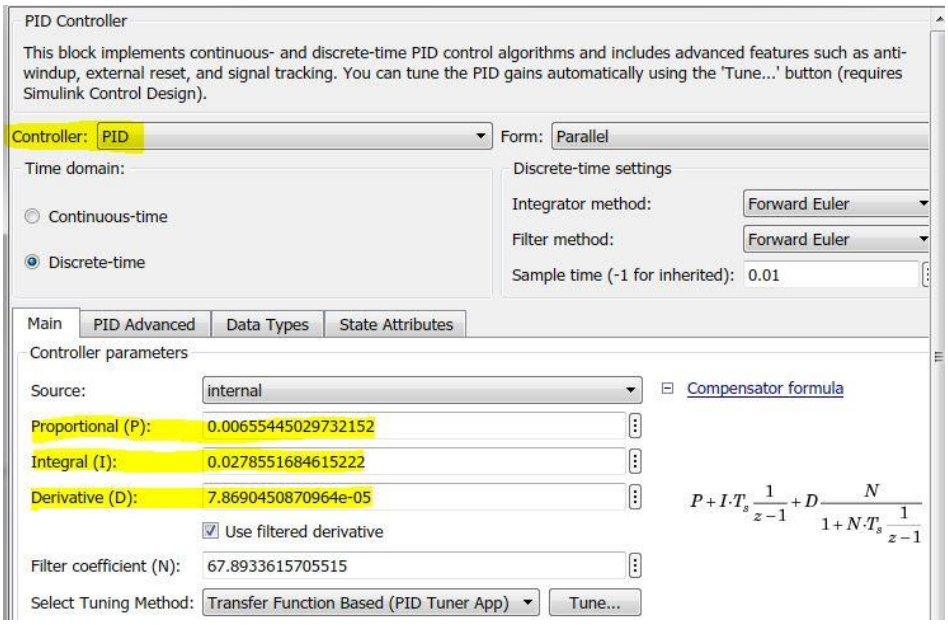
Εικόνα 3.26 Απόκριση με PD ελεγκτή στο DC motor

Παρατηρούμε ότι η απόκριση εμφανίζει μία υπερύψωση από 0.001 έως 0.398 δευτερόλεπτα που δεν ξεπερνά το 20% της εισόδου, δηλαδή την τιμή 48. Από 0.398 και μετά η απόκριση παίρνει την τιμή της εισόδου.

3.2.5.6 Αναλογικός – Ολοκληρωτικός - Διαφορικός ελεγκτής PID

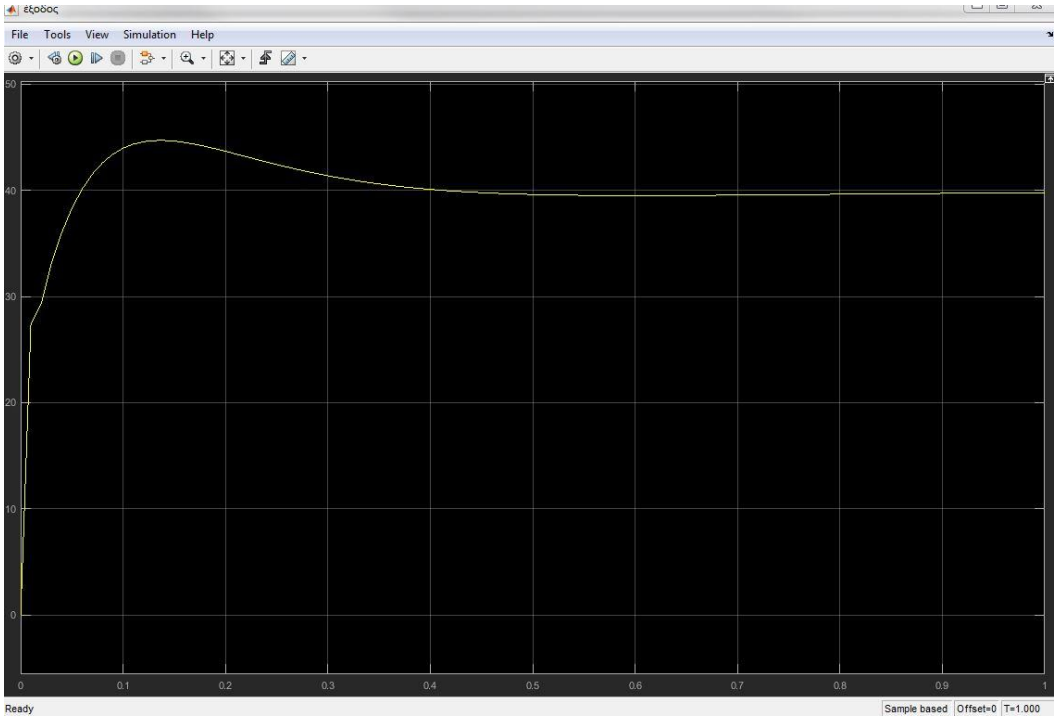
Ρυθμίσαμε το block "ελεγκτής" να χρησιμοποιεί μόνο την παράμετρο PID. Με τη λειτουργία tune επιλέξαμε τη μεταβατική συμπεριφορά, ώστε να πετύχουμε τη μεγαλύτερη ακρίβεια και την απόκριση στον χρόνο (sec.) τέτοια ώστε το σύστημα να είναι ευσταθές. Διότι παρατηρήθηκε ότι με την αύξηση της χρονικής απόκρισης το σύστημα ασταθοποιείται. Οι τιμές των παραμέτρων PID είναι:

Παράμετρος P:	**0.00655445029732152**
Παράμετρος I:	**0.0278551684615222**
Παράμετρος D:	**7.8690450870964e-05**



Εικόνα 3.27 Ρυθμίσεις PID ελεγκτή για DC motor

Για είσοδο 40 μοιρών κατά την εκτέλεση του Simulink λάβαμε την παρακάτω απόκριση.



Εικόνα 3.28 Απόκριση με PID ελεγκτή στο DC motor

Παρατηρούμε ότι η απόκριση εμφανίζει μία υπερύψωση από 0.059 έως 0.432 δευτερόλεπτα που δεν ξεπερνά το 20% της εισόδου, δηλαδή την τιμή 48. Από 0.432 έως 0.993 δευτερόλεπτα η απόκριση τείνει να πάρει την τιμή της εισόδου και μετά από 1 δευτερόλεπτο η απόκριση παίρνει την τιμή της εισόδου.

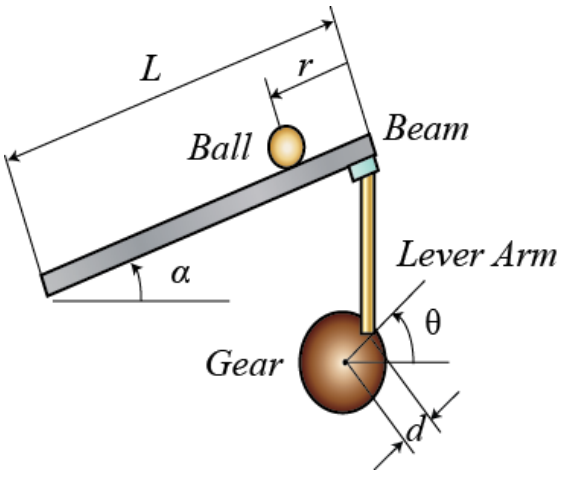
Αφού είδαμε τις αποκρίσεις όλων των πιθανών ελεγκτών για τη διεργασία μας καταλήγουμε σε κάποια συμπεράσματα.

- Ο ελεγκτής με τη μεγαλύτερη ταχύτητα εκτέλεσης είναι ο PD ελεγκτής καθώς η απόκριση παίρνει την τιμή της εισόδου σε 0.398 δευτερόλεπτα. Ο μικρότερος χρόνος με τους υπόλοιπους ελεγκτές.
- Οι ελεγκτές που οδηγούν το σύστημα σε αστάθεια με όποιες τιμές κι αν πάρουν οι παράμετροί τους είναι ο ελεγκτής I και ο ελεγκτής D, και ο λόγος είναι ότι δεν μπορούν να λειτουργήσουν σε διεργασίες που ο βαθμός τους είναι δευτέρου βαθμού.

3.3 Ball on Beam

3.3.1 Ball on Beam διαδικασία προσδιορισμού συνάρτησης μεταφοράς

Στο παρόν μέρος της εργασίας θα μελετήσουμε σύστημα μπάλας πάνω σε ράβδο ή αλλιώς Ball on Beam. Έχουμε δηλαδή μία ράβδο που πάνω της κινείται μία μπάλα ανάλογα με την κλίση της ράβδου. Το ένα άκρο της ράβδου είναι σταθερό και το άλλο είναι συνδεδεμένο με βραχίονα. Ο βραχίονας παίρνει κίνηση από γρανάζι ενός σερβοκινητήρα. Αναλυτικά στην εικόνα.



Εικόνα 3.29 [19]Σύστημα Ball on Beam.

Η διαφορά αυτού του συστήματος με τα προηγούμενα είναι ότι στον έλεγχο η είσοδος και η έξοδος διαφέρουν. Η είσοδος είναι απόσταση και η έξοδος γωνία. Ως απόσταση ορίζεται η μετατόπιση της μπάλας πάνω στη ράβδο και ως γωνία η γωνία περιστροφής του

[19] http://ctms.engin.umich.edu/CTMS/index.php?example=BallBeam&section=SystemModeling

σερβοκινητήρα. Θα προσπαθήσουμε, λοιπόν, να ελέγξουμε τη θέση της μπάλας δημιουργώντας τον κατάλληλο ελεγκτή. Για να κάνουμε όμως κάτι τέτοιο θα πρέπει πρώτα να προσδιορίσουμε τη συνάρτηση μεταφοράς του συστήματος.

Για διευκόλυνσή μας υποθέτουμε ότι η μπάλα κινείται πάνω στη ράβδο χωρίς ολίσθηση και άρα χωρίς να ασκούνται πάνω της δυνάμεις τριβής.

Το μήκος της ράβδου ορίζεται ως "L", η μετατόπιση της μπάλας ως "r", η κλίση της ράβδου ως "α", η ακτίνα του γραναζιού του μοτέρ ως "d" και η γωνία περιστροφής του γραναζιού ως "θ".

Για να μπορέσουμε όμως να αναλύσουμε το σύστημα σε δυνάμεις και να κάνουμε τον νόμο του Νεύτωνα θα χρειαστούμε επιπλέον μεταβλητές. Έτσι, λοιπόν, τη μάζα της μπάλας την ορίζουμε ως "m", την ακτίνα της μπάλας ως "R", τη σταθερά βαρύτητας ως "g" και τη ροπή αδρανείας της μπάλας ως "J".

L	*Μήκος ράβδου*
r	*Μετατόπιση της μπάλας*
α	*Κλίση της ράβδου*
d	*Ακτίνα γραναζιού μοτέρ*
θ	*Γωνία περιστροφής γραναζιού μοτέρ*
m	*Μάζα μπάλας*
R	*Ακτίνα μπάλας*
g	*Σταθερά βαρύτητας*
J	*Ροπή αδρανείας μπάλας*

Αγνοώντας ότι η δεύτερη παράγωγος της γωνίας "α" επηρεάζει την επιτάχυνση της μπάλας υπολογίζουμε την εξής εξίσωση:

$$0 = \left(\frac{J}{R^2} + m\right)\ddot{r} + mg\alpha - mr\dot{\alpha}^2$$

Κάνοντας γραμμικοποίηση στην παραπάνω σχέση λαμβάνουμε:

$$(2) \qquad \left(\frac{J}{R^2}+m\right)\ddot{r} = -mg\alpha$$

Η εξίσωση που συνδέει τη γωνία του γραναζιού με τη γωνία της ράβδου είναι γραμμική.

$$(3) \qquad a = \frac{d}{L}\theta$$

Από τις σχέσεις 2,3 έχουμε:

$$\left(\frac{J}{R^2}+m\right)\ddot{r} = -mg\frac{d}{L}\theta$$

Κάνουμε, λοιπόν, μετασχηματισμό Laplace:

$$\left(\frac{J}{R^2}+m\right)R(s)\,s^2 = -mg\frac{d}{L}\Theta(s)$$

Με μια μικρή αναδιάταξη της μορφής έξοδος προς είσοδο, έχουμε προσδιορίσει τη γενική μορφή της συνάρτησης μεταφοράς του συστήματός μας. [29]

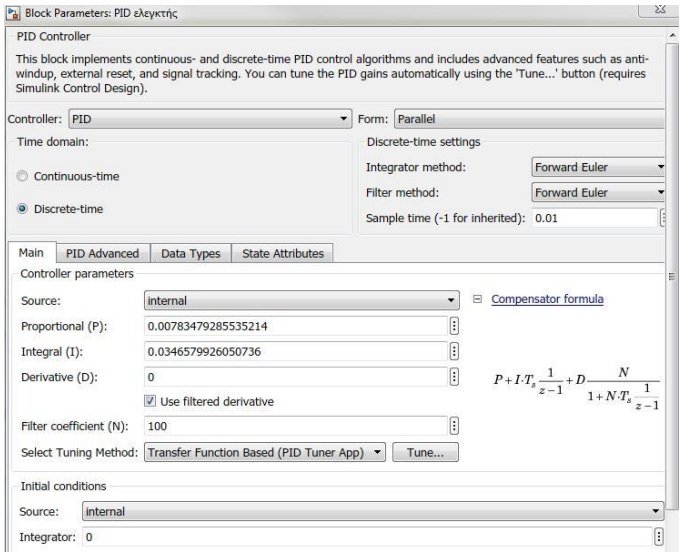
$$P(s) = \frac{R(s)}{\Theta(s)} = -\frac{mgd}{L\left(\frac{J}{R^2}+m\right)}\frac{1}{s^2}$$

Αντικαθιστώντας, λοιπόν, τις μεταβλητές με τις τιμές που γνωρίζουμε από το σύστημά μας καταλήγουμε στην παρακάτω συνάρτηση μεταφοράς της διεργασίας μας. Ο υπολογισμός έγινε στο "command window" του Matlab.

$$P(s) = \frac{0.6364}{s^2}$$

3.3.2 Ball on Beam παραμετροποίηση ελεγκτή

Για να υπολογίσουμε τις παραμέτρους του PID ελεγκτή πρέπει πρώτα να δημιουργήσουμε ένα blank model αρχείο στο Simulink ακολουθώντας την ίδια διαδικασία με την ενότητα 2.2.2, μόνο που στο block "Transfer Function" θα συμπληρώσουμε τη συνάρτηση μεταφοράς που υπολογίσαμε. Έτσι, λοιπόν, κάνοντας tune στον PID ελεγκτή παίρνουμε τις παρακάτω τιμές του ελεγκτή μας.

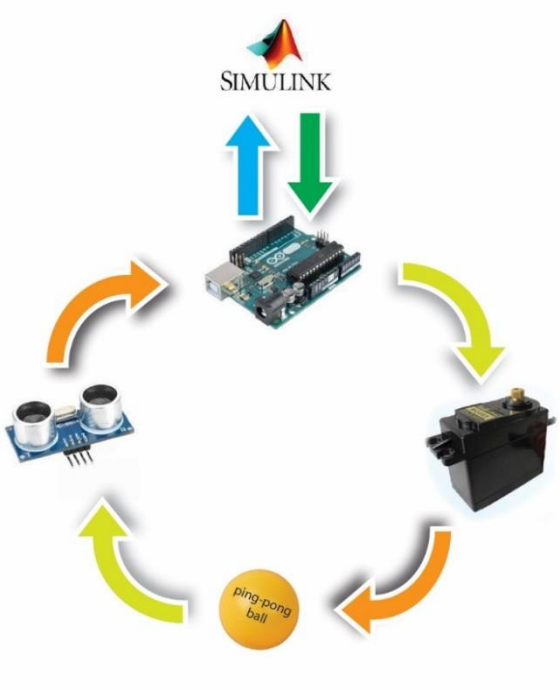


Εικόνα 3.30 block "PID controller"

Ο συγκεκριμένος ελεγκτής αντιπροσωπεύει τη διεργασία μας. Μπορεί να λειτουργήσει με οποιονδήποτε αισθητήρα στην ανάδραση, αρκεί στην είσοδό του να μπαίνουν τιμές μεταφρασμένες σε μετατόπιση και μετά την έξοδό του να μετατρέπονται σε γωνία. Εμείς θα μελετήσουμε τη διεργασία με δύο αισθητήρες απόστασης, με αισθητήρα υπερήχων και με camera αντίστοιχα. Για να γίνει όμως αυτό θα πρέπει να δημιουργήσουμε ξεχωριστά blank models αρχεία στο Simulink, γιατί ο τρόπος δειγματοληψίας του κάθε αισθητήρα είναι διαφορετικός.

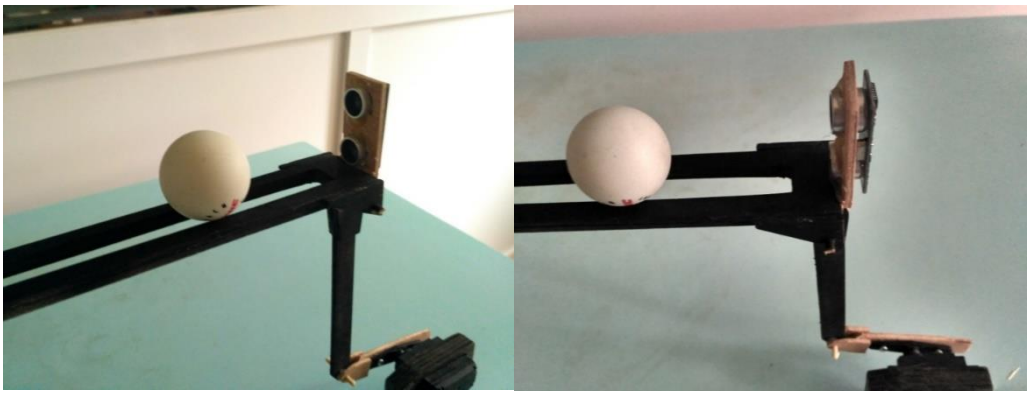
3.3.3 Ball on Beam με αισθητήρα υπερήχων

Αρχικά μελετάμε ως ανάδραση το σήμα του αισθητήρα υπερήχων ή αλλιώς και ultrasonic sensor που έχουμε εξηγήσει παραπάνω τον τρόπο λειτουργίας του. Για αυτή την εφαρμογή θα χρησιμοποιήσουμε έναν αισθητήρα υπερήχων HC-SR04, μία πλακέτα Arduino Uno Rev3, ένα σερβοκινητήρα, ένα μπαλάκι ping-pong και την κατασκευή μας με τη ράβδο. Η γενική επεξήγηση του τρόπου λειτουργίας του συστήματος δίνεται παρακάτω.



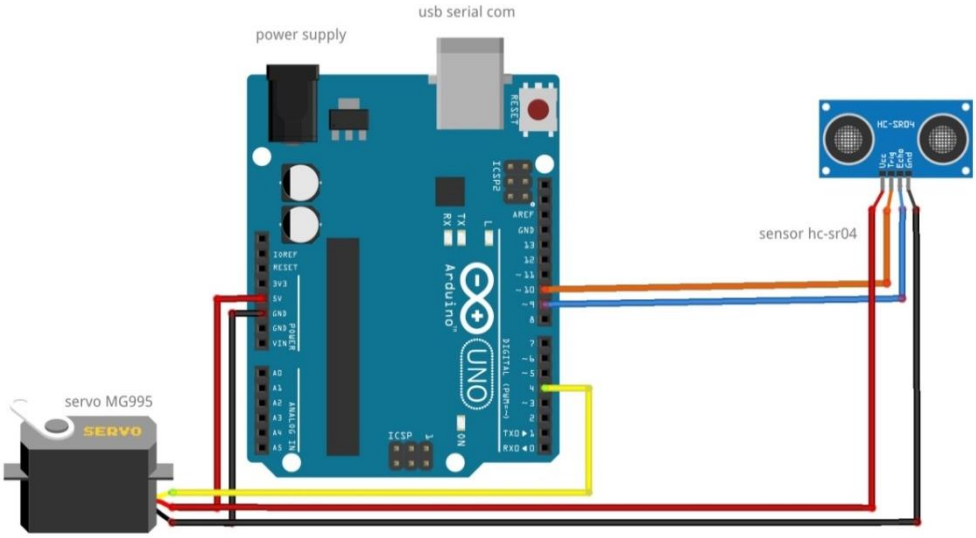
Εικόνα 3.31 Διάγραμμα γενικής λειτουργίας Ball on Beam με αισθητήρα υπερήχων.

Το Simulink με την εκκίνησή του, στέλνει σειριακά στο Arduino την επιθυμητή γωνία. Το Arduino μέσω του κώδικά του δίνει κίνηση στον σερβοκινητήρα. Ο σερβοκινητήρας κινεί τη ράβδο και κινείται η μπάλα. Ο αισθητήρας διαβάζει τη μετατόπιση της μπάλας μέσω των υπερήχων του και στέλνει τη μετατόπιση στο Arduino σε μορφή παλμών. Έπειτα το Arduino μετατρέπει τον παλμό σε μετατόπιση και το στέλνει σειριακά στο Simulink. Το Simulink χρησιμοποιεί τη μετατόπιση αυτή ως ανάδραση στο σύστημα. Ο αισθητήρας είναι τοποθετημένος στο ένα άκρο της ράβδου σε όρθια θέση κοιτώντας προς τη μπάλα με σκοπό τη συνεχή μέτρηση της μετατόπισης της μπάλας.



Εικόνα 3.32 Φωτογραφία Ball on Beam με αισθητήρα υπερήχων της εφαρμογής μας.

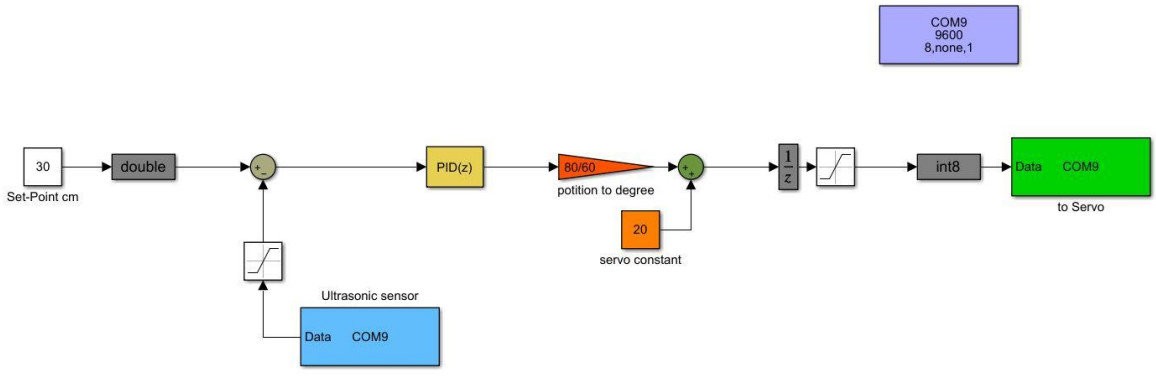
Συνδέουμε τον σερβοκινητήρα και τον αισθητήρα με το Arduino. Το signal pin του σερβοκινητήρα θα συνδεθεί με το digital pin 4 του Arduino, το Vcc pin του σερβοκινητήρα με το 5V pin tou Arduino και το GND pin του σερβοκινητήρα με το GND pin του Arduino. Το echo pin του αισθητήρα με το digital pin 9 του Arduino, το triger pin του αισθητήρα με το digital pin 10 του Arduino και η τροφοδοσία του αισθητήρα στα ίδια pins με την τροφοδοσία του σερβοκινητήρα αντίστοιχα. Η πλακέτα Arduino τροφοδοτείται από εξωτερική πηγή με τάση 12 Volt DC και τη συνδέουμε σειριακά με USB καλώδιο με τον υπολογιστή μας.



Εικόνα 3.33 Συνδεσμολογία hardware συστήματος Ball on Beam με αισθητήρα υπερήχων.

Δημιουργούμε, λοιπόν, ένα νέο blank model στο Simulink με τον ελεγκτή που υπολογίσαμε προηγουμένως, με τα blocks σειριακής επικοινωνίας που χρησιμοποιήσαμε και στην ενότητα 2.2.2 (Serial Configuration, Serial Send, Serial Receive), με ένα block “gain”, δύο blocks “constant” και δύο “sum”, δύο “Saturation” και δύο “Data Type Conversion” και ένα “unit Delay”.

Η σύνδεση των blocks είναι η ακόλουθη:



Εικόνα 3.34 Ball on Beam με αισθητήρα υπερήχων και σειριακή επικοινωνία

Παρατηρούμε ότι η ανάδραση χρησιμοποιείται απευθείας μετά τη λήψη χωρίς να γίνεται κάποια μετατροπή. Αυτό γιατί στον κώδικα Arduino έχουμε προγραμματίσει, ώστε να στείλει στο Simulink τη μετατόπιση που διαβάζει ο αισθητήρας. Το σημείο που θέλει ο χρήστης να μετατοπιστεί η μπάλα πάνω στη ράβδο ορίζεται από το "Set-Point" block σε εκατοστά. Επίσης, μετά τον ελεγκτή φτιάχνουμε τη συγκεκριμένη διάταξη για να μετατρέψουμε τη διορθωμένη μετατόπιση του ελεγκτή σε γωνία. Αυτή η γωνία είναι η γωνία που θα περιστραφεί ο σερβοκινητήρας. Πώς υπολογίζεται;

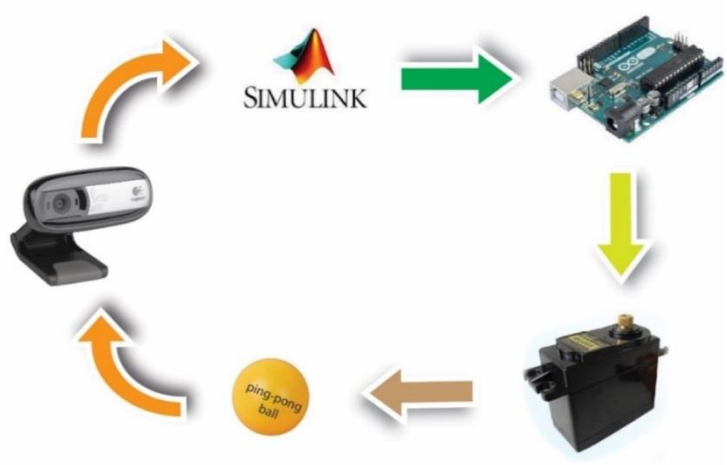
Η διορθωμένη μετατόπιση y πολλαπλασιάζεται με το πηλίκο του εύρους της γωνίας του σερβοκινητήρα Δθ με την απόσταση της ράβδου L και προστίθεται η ελάχιστη γωνία που μπορεί να περιστραφεί ο σερβοκινητήρας "θ΄".

$$\theta = y\frac{\Delta\theta}{L} + \theta'$$

Τέλος χρησιμοποιούμε τα "Saturation" blocks για τη μείωση πιθανού θορύβου, λειτουργώντας ως ζωνοπερατά φίλτρα.

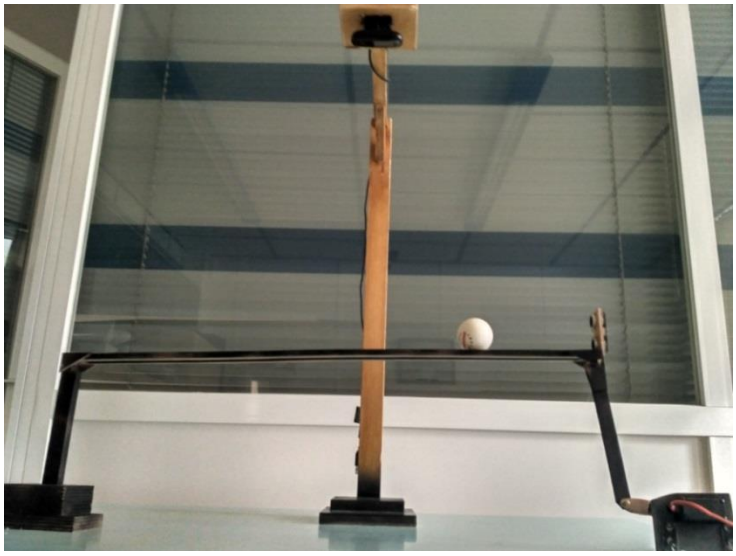
3.3.4 Ball on Beam με camera

Σε αυτή την υποενότητα θα μελετήσουμε ως ανάδραση το σήμα μιας απλής κάμερας. Εμείς στην εφαρμογή μας χρησιμοποιούμε μια διαδικτυακή κάμερα ή web camera και πιο συγκεκριμένα την C170 της εταιρίας Logitech. Επίσης χρησιμοποιούμε ό,τι και στην προηγούμενη εφαρμογή, δηλαδή, μία πλακέτα Arduino Uno Rev3, έναν σερβοκινητήρα, ένα μπαλάκι ping-pong και την κατασκευή μας με τη ράβδο. Η γενική επεξήγηση του τρόπου λειτουργίας του συστήματος δίνεται στο παρακάτω διάγραμμα.



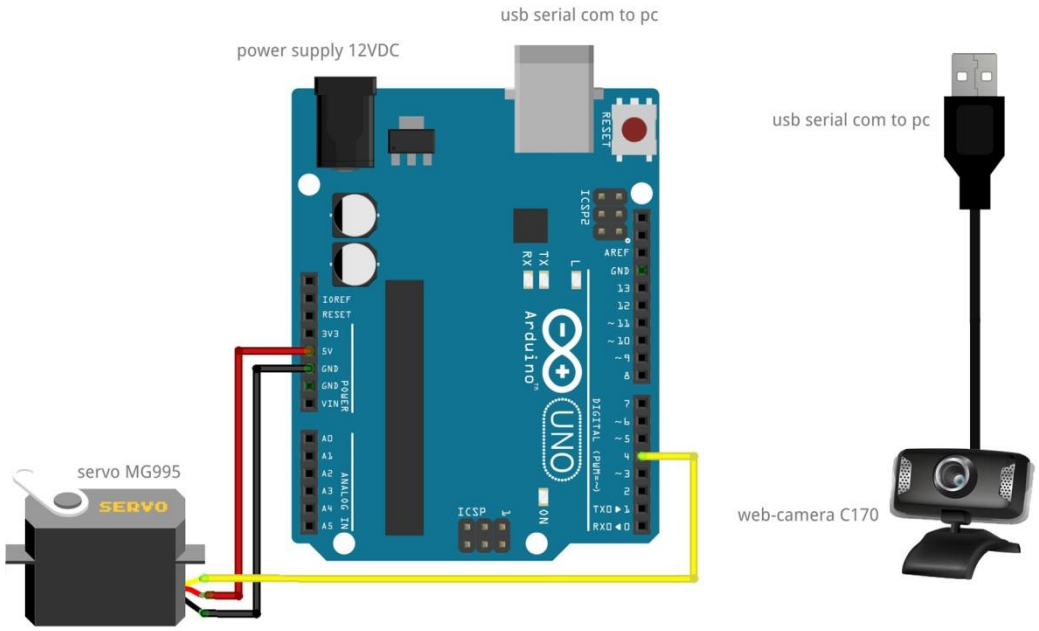
Εικόνα 3.35 Διάγραμμα γενικής λειτουργίας Ball on Beam με camera.

Παρατηρούμε ότι το παραπάνω διάγραμμα διαφέρει από αντίστοιχα διαγράμματα στη λειτουργία του Arduino. Το Simulink με την εκκίνησή του, στέλνει σειριακά στο Arduino την επιθυμητή γωνία. Το Arduino μέσω του κώδικά του δίνει κίνηση στον σερβοκινητήρα. Ο σερβοκινητήρας κινεί τη ράβδο και έτσι κινείται η μπάλα. Η κάμερα στέλνει την εικόνα στο Simulink. Το Simulink φιλτράρει την εικόνα (με κάποιες λειτουργίες που θα τις δούμε παρακάτω) και κάνει αναγνώριση της θέσης της μπάλας στην εικόνα. Έτσι, αν η μπάλα μετατοπιστεί, αυτόματα καταλαβαίνει τη νέα θέση της μπάλας . Το Simulink χρησιμοποιεί τη μετατόπιση αυτή ως ανάδραση στο σύστημα.
Η camera είναι τοποθετημένη πάνω σε μια βάση για να καταγράφει από ψηλά την κίνηση της μπάλας πάνω στη ράβδο.



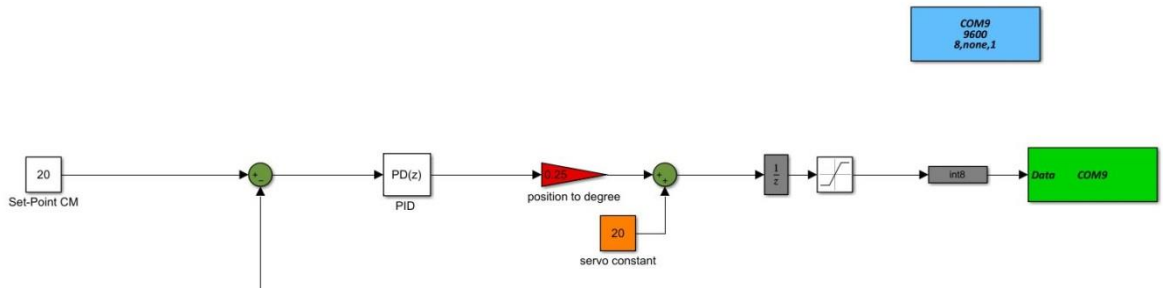
Εικόνα 3.36 Φωτογραφία Ball on Beam με camera

Συνδέουμε την camera σειριακά με τον υπολογιστή μας και το σερβοκινητήρα όπως στην προηγούμενη υποενότητα. Το “signal pin” του σερβοκινητήρα θα συνδεθεί με το digital pin 4 του Arduino, το Vcc pin του σερβοκινητήρα με το 5V pin του Arduino και το GND pin του σερβοκινητήρα με το GND pin του Arduino. Η πλακέτα Arduino τροφοδοτείται από εξωτερική πηγή με τάση 12 V DC και τη συνδέουμε σειριακά με USB καλώδιο με τον υπολογιστή μας.



Εικόνα 3.37 Συνδεσμολογία hardware συστήματος Ball on Beam με camera.

Δημιουργούμε, λοιπόν, ένα νέο "blank model" στο Simulink με τον ελεγκτή που υπολογίσαμε προηγουμένως, με τα blocks σειριακής επικοινωνίας που χρησιμοποιήσαμε και στην ενότητα 3.2.1 και 2.3.2 (Serial Configuration και Serial Send), με ένα block "gain", δύο blocks "constant και δύο "sum", ένα "Saturation" και ένα "Data Type Conversion" και ένα "unit Delay".

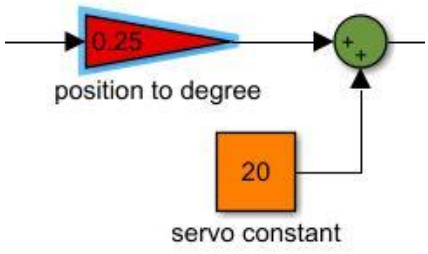


Εικόνα 3.38 Ολοκληρωμένο μοντέλο Ball on Beam με camera με σειριακή επικοινωνία.

Το set-point του συστήματος είναι η είσοδος του συστήματος, δηλαδή η θέση της μπάλας που επιθυμούμε να μετατοπιστεί. Έτσι, λοιπόν, το set-point το ρυθμίζουμε από ένα block "constant". Η επιλογή θα γίνεται σε μονάδα μέτρησης εκατοστά (cm). Το block "constant" το συνδέουμε με το "+" του block "sum". Στην έξοδο του block "sum" έχουμε το error που προκύπτει από τη διαφορά εισόδου (set-point) με ανάδρασης (camera). Το error απόστασης

μπαίνει στον ελεγκτή (block “PID Controller”) και από εκεί βγαίνει η διορθωμένη θέση της μπάλας.

Επειδή, όμως, το Simulink στέλνει στο Arduino τη γωνία περιστροφής του σερβοκινητήρα, θα πρέπει να μετατρέψουμε τη διορθωμένη θέση σε γωνία. Αυτό επιτυγχάνεται με αυτή τη σύνδεση των blocks.



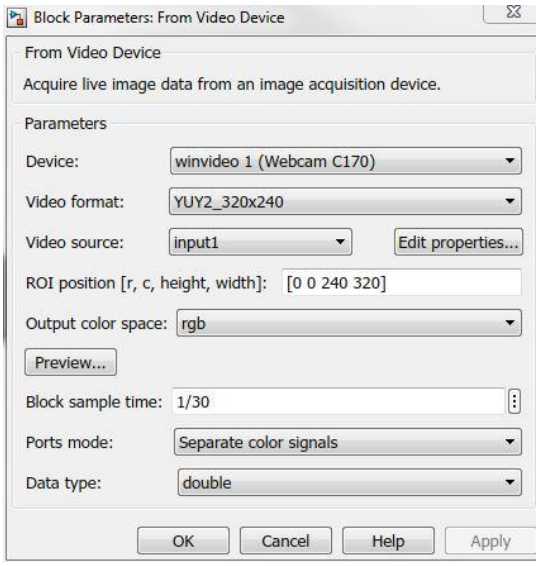
Εικόνα 3.39 Μετατροπέας cm σε μοίρες

Η διορθωμένη μετατόπιση y πολλαπλασιάζεται με το πηλίκο του εύρους της γωνίας του σερβοκινητήρα Δθ με την απόσταση της ράβδου L και προστίθεται η μικρότερη γωνία που γυρίζει ο σερβοκινητήρας “θ”.

$$\theta = y\frac{\Delta\theta}{L} + \theta'$$

Τέλος χρησιμοποιούμε το “Saturation” block για τη μείωση πιθανού θορύβου.

Όπως είπαμε παραπάνω η κάμερα συνδέεται απευθείας με το Simulink χωρίς τη διαμεσολάβηση του Arduino, με σκοπό την εφαρμογή των διαδικασιών της μηχανικής όρασης. Έτσι, λοιπόν, για την εισαγωγή της εικόνας της κάμερας στο Simulink δε χρησιμοποιούμε το “Serial Receive” block που χρησιμοποιούσαμε στις προηγούμενες εφαρμογές, αλλά το “From Video Device” block. Το συγεκριμένο block μάς δίνει τη δυνατότητα να εισαγάγουμε στο Simulink τη ζωντανή λήψη εικόνας της κάμερας. Η προεπεξεργασία της εικόνας γίνεται με την εισαγωγή της στο Simulink καθώς επιλέγεται ο χρωματικός χώρος. Αυτό επιτυγχάνεται και με τις παρακάτω ρυθμίσεις.



Εικόνα 3.40 Ρυθμίσεις "From Video Device" block

Στην επιλογή **"Device"** επιλέγουμε τη συσκευή που θέλουμε να μας δίνει την εικόνα.

Στο **"Video format"** επιλέγουμε την ανάλυση και μορφή της εικόνας. Εμείς επιλέγουμε σε YUY2 τη χαμηλότερη ανάλυση για να πετύχουμε μεγαλύτερη ταχύτητα ανταπόκρισης του συστήματος.

Το **"ROI position"** χρησιμοποιείται για να εισαγάγουμε ένα διάνυσμα γραμμών που καθορίζει την περιοχή απόκτησης στην εικόνα βίντεο. Η μορφή είναι [σειρά, στήλη, ύψος, πλάτος]. Οι προεπιλεγμένες τιμές για τη σειρά και τη στήλη είναι 0. Οι προεπιλεγμένες τιμές για το ύψος και το πλάτος έχουν οριστεί στη μέγιστη επιτρεπόμενη τιμή που υποδεικνύεται από την ανάλυση της μορφής βίντεο.

Στο **"Output color space"** ρυθμίζουμε τον τύπο χρωμάτων της εικόνας, εμείς χρησιμοποιούμε RGB για να μπορέσουμε πιο εύκολα να φιλτράρουμε τα χρώματα της εικόνας αργότερα.

Στο **"Block sample time"** καθορίζουμε τον χρόνο δειγματοληψίας του μπλοκ κατά τη διάρκεια της προσομοίωσης. Αυτός είναι ο ρυθμός με τον οποίο το μπλοκ εκτελείται κατά τη διάρκεια της προσομοίωσης. Η προεπιλογή είναι 1/30.

Αφού πατήσουμε "Apply" και "ok" παρατηρούμε ότι το block έχει τρεις εξόδους. Η κάθε έξοδος είναι και ένα διάνυσμα που αντιπροσωπεύει το αντίστοιχο χρώμα από την προεπιλογή RGB. Η "R" έξοδος είναι το κόκκινο χρώμα της εικόνας, το "G" το πράσινο και το "B" το μπλε. Χρησιμοποιώντας, λοιπόν, αυτές τις τρεις εξόδους ταυτόχρονα θα λάβουμε την εικόνα που τραβάει η κάμερα κάθε χρονική στιγμή.

Πρέπει να έχουμε υπόψη μας δύο πολύ σημαντικές παρατηρήσεις.

- Η φωτεινότητα του χώρου επιδρά σημαντικά στα χρώματα της εικόνας της κάμερας.

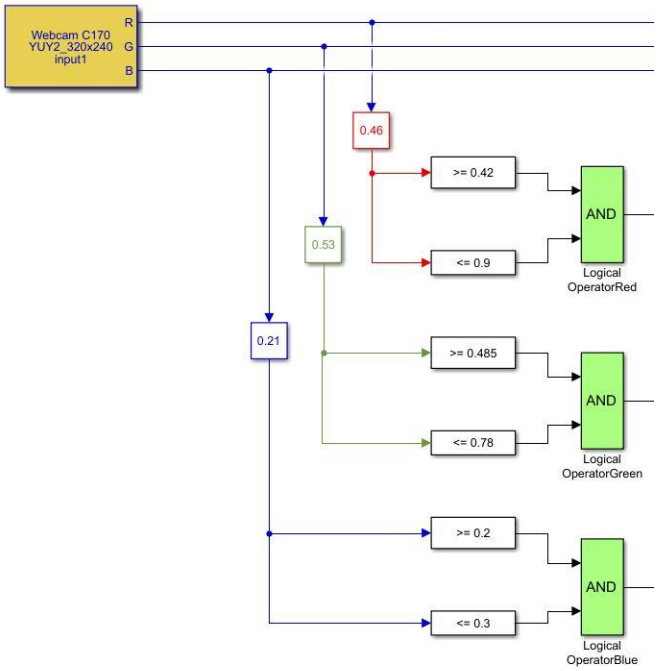
- Στην εικόνα της κάμερας περιλαμβάνονται κι άλλα αντικείμενα εκτός από την μπάλα.

Για τη λύση του πρώτου προβλήματος τοποθετούμε δίπλα στην κάμερα και την κατασκευή φωτιστικά για τη σταθεροποίηση των μεταβολών των χρωμάτων της εικόνας ανεξάρτητα από τη φωτεινότητα του χώρου.

Για τη λύση του δεύτερου προβλήματος ακολουθούμε την εξής διαδικασία.

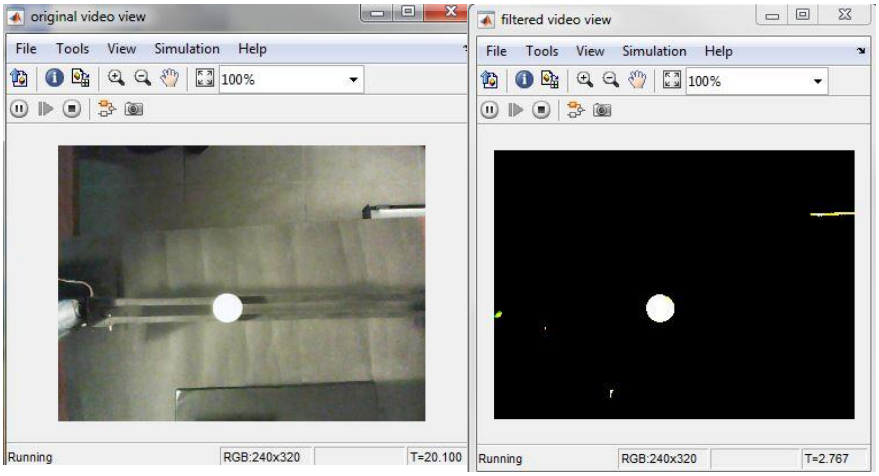
Εμείς θέλουμε να γνωρίζουμε τις συντεταγμένες της θέσης της μπάλας πάνω στην εικόνα. Αυτό που μπορούμε να κάνουμε στην αρχή είναι, να φιλτράρουμε τα χρώματα της εικόνας ξεχωρίζοντας το χρώμα της μπάλας από τα υπόλοιπα χρώματα και στη συνέχεια να ορίσουμε τις συντεταγμένες της μπάλας πάνω στην εικόνα που θα είναι με ξεχωριστό χρώμα.

Το φιλτράρισμα θα γίνει με blocks: ένα “Slider Gain” , δύο “Compare to constant” και ένα “Logical Operator” για κάθε RGB χρώμα και θα συνδεθούν όπως στην εικόνα.



Εικόνα 3.41 Κατάτμηση εικόνας με βάση το χρώματος της μπάλας.

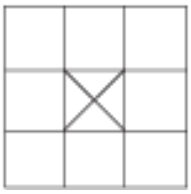
Οι τιμές που δώσαμε εμείς στα συγκεκριμένα blocks έγινε κατόπιν πειραμάτων καθώς έπαιζε μεγάλο ρόλο η φωτεινότητα του χώρου. Ξεχωρίσαμε το λευκό χρώμα της μπάλας «μαυρίζοντας» όλα τα άλλα χρώματα. Μας παρείχε μεγάλη βοήθεια το block “Video Viewer”. Κατά την εκτέλεση του Simulink, το συγκεκριμένο block αν το συνδέσουμε με τα RGB κανάλια, ανοίγει αυτόματα ένα παράθυρο με την εικόνα που λαμβάνει.



Εικόνα 3.42 Φωτογραφία α) εικόνας από την κάμερα και β) κατατμημένη εικόνα

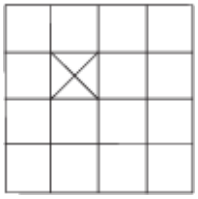
Παρατηρούμε, λοιπόν, ότι στη φιλτραρισμένη εικόνα φαίνεται ξεκάθαρα η θέση της μπάλας. Αυτή, λοιπόν, τη θέση θα προσπαθήσουμε να την απεικονίσουμε με συντεταγμένες στην αρχική εικόνα.

Συνδέουμε τα φιλτραρισμένα RGB χρώματα με την είσοδο ενός ακόμη "Logical Operator" block και μέχρι στιγμής κάναμε και τη διαδικασία της κατάτμησης της εικόνας με βάση το χρώμα. Την έξοδό του "Logical Operator" block τη συνδέουμε με ένα "Erosion" block για να κάνουμε μορφολογική διάβρωση και να βρούμε τα τοπικά ελάχιστα της εικόνας. Το μπλοκ ολισθαίνει το δομικό στοιχείο πάνω σε μια εικόνα, βρίσκει τα τοπικά ελάχιστα και δημιουργεί τη μήτρα εξόδου από αυτές τις ελάχιστες τιμές. Αν το στοιχείο έχει κεντρικό στοιχείο, το μπλοκ τοποθετεί τα ελάχιστα εκεί, όπως φαίνεται στο παρακάτω σχήμα.



Εικόνα 3.43 Απεικόνιση δομικού στοιχείου στο κέντρο.

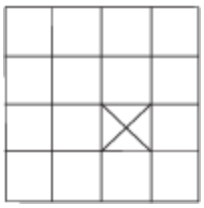
Αν το δομικό στοιχείο δεν έχει ακριβές κέντρο, το μπλοκ έχει μια προκατάληψη προς την άνω αριστερή γωνία και τοποθετεί τα ελάχιστα εκεί, όπως φαίνεται στο παρακάτω σχήμα.



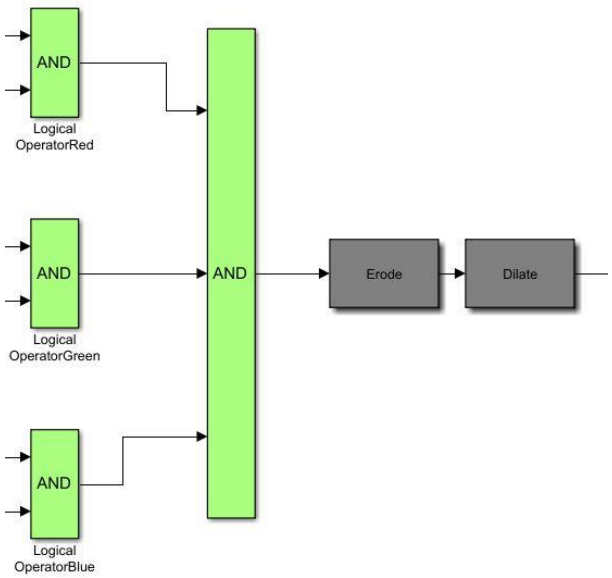
Εικόνα 3.44 Δομικό στοιχείο με προκατάληψη προς την άνω αριστερή γωνία.

Την έξοδο από το “Erosion” block (διάβρωση) τη συνδέουμε με την είσοδο ενός “Dilation” block (διαστολή). Αυτό για να κάνουμε μορφολογική διαστολή της εικόνας. Το μπλοκ αυτό περιστρέφει το δομικό στοιχείο κατά 180 μοίρες. Στη συνέχεια, ολισθαίνει το στοιχείο διαμόρφωσης πάνω σε μια εικόνα, βρίσκει τα τοπικά μέγιστα και δημιουργεί τη μήτρα εξόδου από αυτές τις μέγιστες τιμές. Αν το στοιχείο δομής έχει κεντρικό στοιχείο, το μπλοκ τοποθετεί τα μέγιστα εκεί, όπως φαίνεται στην Εικόνα 2.3.16..

Εάν το στοιχείο δομής δεν έχει ακριβές κέντρο, το μπλοκ έχει προκατάληψη προς την κάτω δεξιά γωνία ως αποτέλεσμα της περιστροφής. Το μπλοκ τοποθετεί τα μέγιστα εκεί, όπως φαίνεται στο παρακάτω σχήμα.

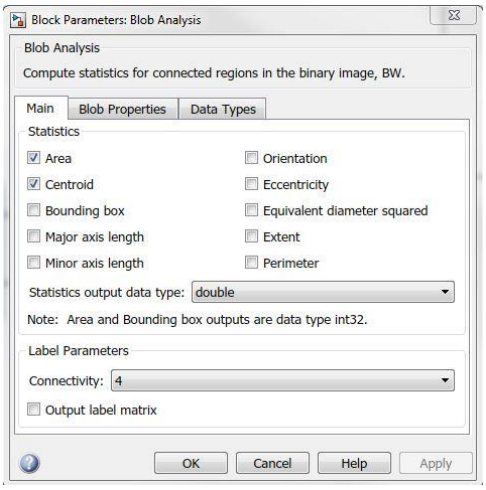


Εικόνα 3.45 Δομικό στοιχείο με προκατάληψη προς την κάτω δεξιά γωνία.



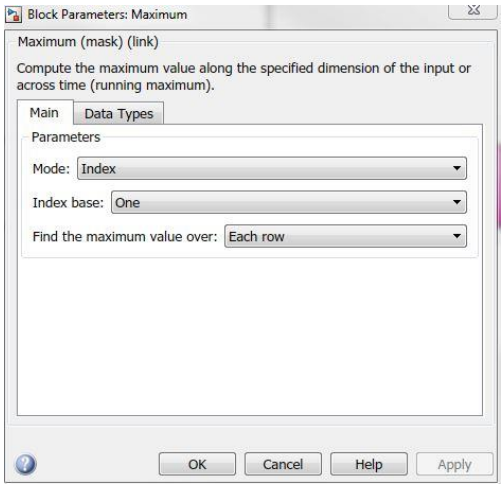
Εικόνα 3.46 Μορφολογική διαστολή και συστολή

Στη συνέχεια μπαίνουμε στη διαδικασία εξαγωγής χαρακτηριστικών (feature extraction) καθώς την έξοδο από το “Dilation” block την οδηγούμε στο “Blob Analysis” block. Χρησιμοποιούμε αυτό το block για να υπολογίσουμε τα στατιστικά στοιχεία των pixels της εικόνας. Το μπλοκ επιστρέφει ποσότητες όπως το centroid, bounding box, label matrix και blob count. Το block υποστηρίζει σήματα μεταβλητού μεγέθους εισόδου και εξόδου.



Εικόνα 3.47 Ρυθμίσεις "Blob Analysis" block

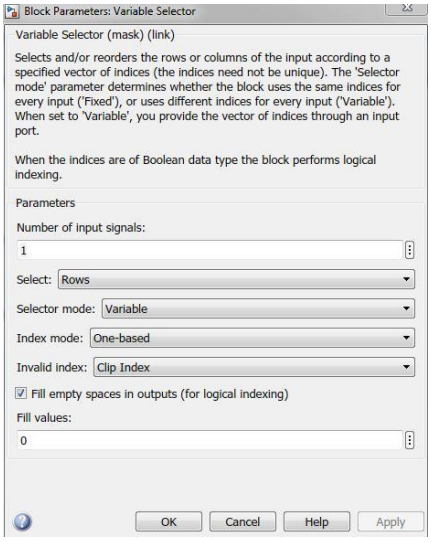
Επιλέγουμε τα checkboxes: "Area" και "Centroid". Το "Area" εξάγει ένα διάνυσμα που αντιπροσωπεύει τον αριθμό των εικονοστοιχείων (pixels) σε επισημασμένες περιοχές. Το "Centroid" εξάγει μια μήτρα των συντεταγμένων (*x*, *y*). Ορίζουμε τον τύπο των δεδομένων **(**Statistics output data type) να είναι double. Στο "connectivity" ορίζουμε ποια εικονοστοιχεία (pixels) συνδέονται μεταξύ τους. Επιλέγουμε «4» γιατί θέλουμε να συνδέσουμε τα pixels που βρίσκονται στην επάνω, κάτω, αριστερή και δεξιά πλευρά της εικόνας. Την έξοδο "Area" του block "Blob Analysis" τη συνδέουμε με ένα block "Maximum", ώστε να μας επιστρέψει τη συστοιχία των μέγιστων τιμών της διάστασης που μπαίνει σαν είσοδο.



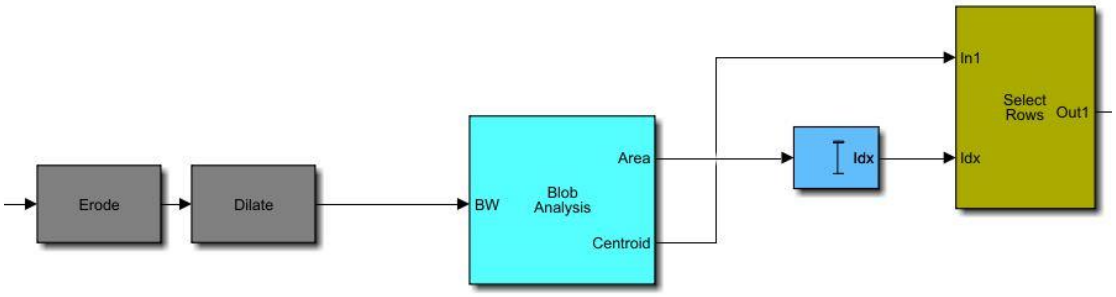
Εικόνα 3.48 Ρυθμίσεις"Maximum" Block

Την έξοδο "Centroid" του block "Blob Analysis" και την έξοδο "Index" του block "Maximum" τις συνδέουμε με ένα block "Variable Selector". Το block εξάγει ένα υποσύνολο γραμμών ή στηλών από τη μήτρα εισόδου M-by-N σε κάθε είσοδο. Επιλέγουμε στο "Select"-

> Row για να εξάγει το block γραμμές από κάθε είσοδο. Το “Selector mode” -> “Variable” για να επιλέγονται τόσες σειρές ή στήλες στην έξοδο, όσο είναι και το διάνυσμα που δέχεται από την Index θύρα.

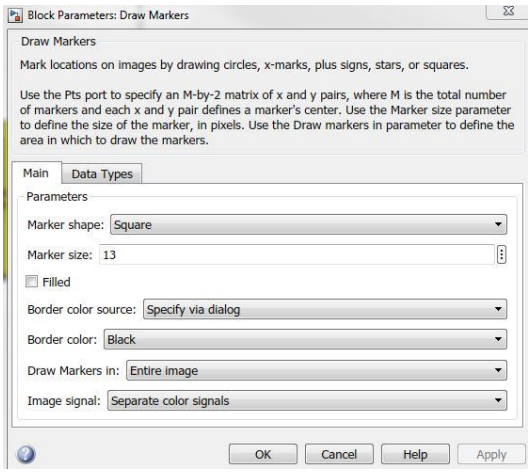


Εικόνα 3.49 Ρυθμίσεις “Variable Selector” Block



Εικόνα 3.50 Μαθηματική μορφολογία-Blob ανάλυση-Maximum-Variable Selector

Πλέον η έξοδος του“Variable Selector” μας δίνει τις συντεταγμένες της μπάλας. Εδώ σταματά η διαδικασία της εξαγωγής χαρακτηριστικών (feature extraction) και ακολουθεί η διαδικασία της αναγνώρισης και της παρακολούθησης (detection – tracking). Έτσι, λοιπόν, την έξοδο του “Variable Selector” τη συνδέουμε στη θύρα Pts του “Draw Markers” block. Αυτό το block επεμβαίνει στην εικόνα που του εισάγουμε χαράσσοντας τις συντεταγμένες από το “Variable Selector”.



Εικόνα 3.51 Ρυθμίσεις "Draw Markers" block

Επιλέγουμε:

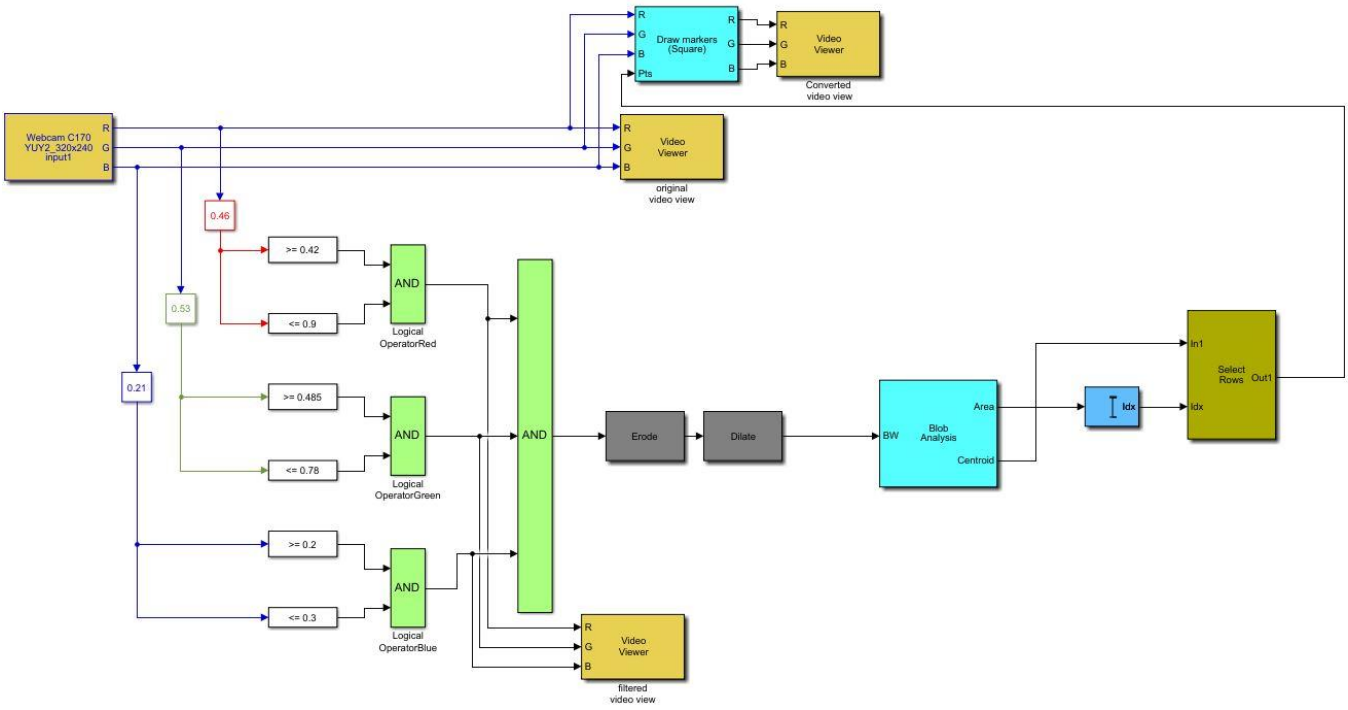
Στο "Marker shape"->Square για να σημαδεύεται το σημείο των συντεταγμένων με το κέντρο ενός τετραγώνου.

Στο "Marker size"->13 για να ορίσουμε το μέγεθος του σημαδιού.

Και από τις επόμενες επιλογές ρυθμίζουμε το χρώμα του σημαδιού.

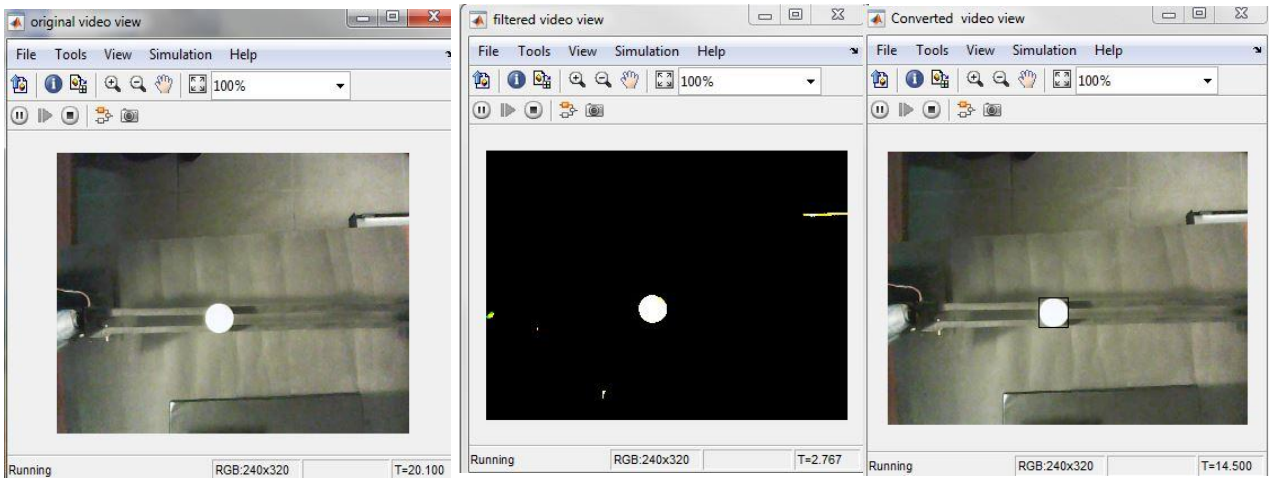
Τέλος, την έξοδο την οδηγούμε σε ένα "Video Viewer" για την απεικόνιση του βίντεο με τις συντεταγμένες της μπάλας κατά την εκτέλεση του Simulink.

Με αυτόν τον τρόπο κάνουμε αναγνώριση και παρακολούθηση στην μπάλα.



Εικόνα 3.52 Ολοκληρωμένο block διάγραμμα αναγνώρισης μπάλας με συντεταγμένες.

Παρακάτω βλέπουμε τη λήψη της κάμερας, τη λήψη της κάμερας μετά από φιλτράρισμα και τη λήψη της κάμερας μετά το όρισμα των συντεταγμένων της μπάλας.



Εικόνα 3.53 α) εικόνα από τη κάμερα β) κατατμημένη εικόνα γ) αναγνώριση μπάλας

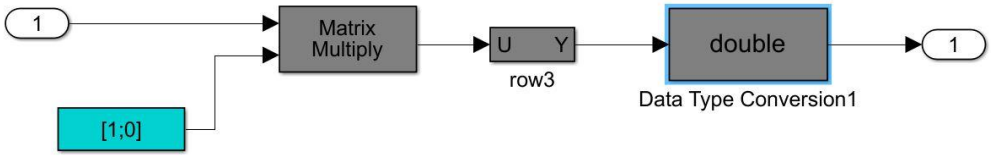
Η εφαρμογή μας δεν έχει ολοκληρωθεί. Θα πρέπει το παραπάνω κύκλωμα και συγκεκριμένα οι συντεταγμένες να συνδεθούν ως ανατροφοδότηση του ελεγκτή που σχεδιάσαμε στην προηγούμενη υποενότητα.

Για να επιτευχθεί αυτό θα πρέπει να λάβουμε υπόψη μας δύο ακόμα παρατηρήσεις.

Η πρώτη παρατήρηση είναι ότι η έξοδος του "Variable Selector" που είναι οι συντεταγμένες της θέσης της μπάλας στην εικόνα είναι πίνακας δύο στηλών. Η πρώτη στήλη παίρνει τις τιμές της τετμημένης x και η δεύτερη της τεταγμένης y της θέσης της μπάλας. Εμείς χρειαζόμαστε μόνο την τετμημένη καθώς το σύστημά μας είναι ενός βαθμού ελευθερίας.

Η δεύτερη παρατήρηση είναι ότι αυτή η έξοδος είναι μονάδα μέτρησης εικονοστοιχείων (pixels) και όχι σε εκατοστά (cm) που χρησιμοποιούμε στον ελεγκτή.

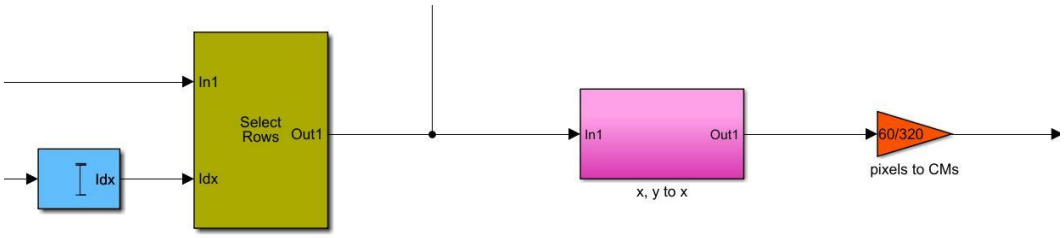
Τα προβλήματα αυτά τα επιλύουμε με μια σειρά επιπλέον blocks. Για το πρώτο πρόβλημα αρκεί να πολλαπλασιάσουμε την έξοδο του "Variable Selector" με έναν πίνακα [1 0]. Με αυτόν τον τρόπο μηδενίζουμε τις τεταγμένες που δε χρειαζόμαστε και κρατάμε τις τετμημένες. Δημιουργούμε, λοιπόν, ένα "Subsystem" και συνδέουμε στο εσωτερικό του την έξοδο του "Variable Selector" με ένα "Product" και έναν πίνακα [1 0] και στη συνέχεια με ένα "row" και με ένα "Data type conversion".



Εικόνα 3.54 Απόκτηση μόνο τετμημένης από τις συντεταγμένες θέσης

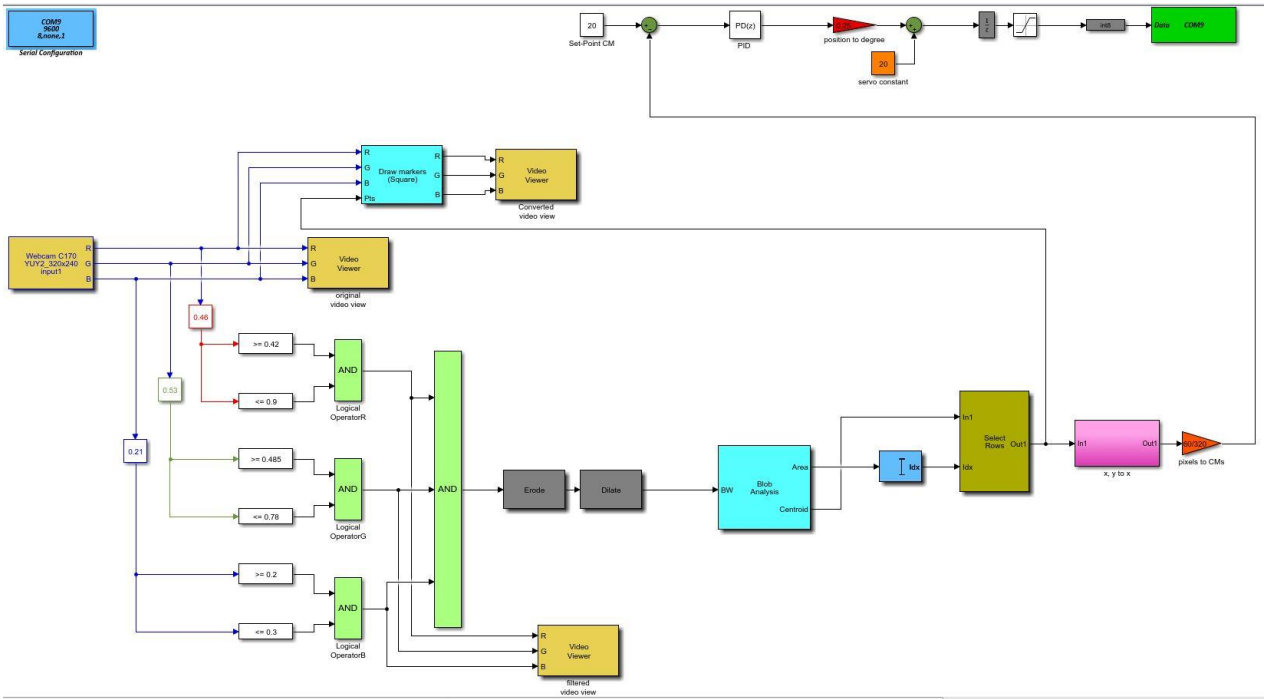
Τα παραπάνω blocks τα επιλέγουμε και τα τοποθετούμε μέσα σε ένα subsystem για περισσότερη ευκολία. Έτσι η έξοδος του "Subsystem" είναι η τετμημένη της θέσης της μπάλας σε pixels.

Λύνοντας το δεύτερο πρόβλημα συνδέουμε την έξοδο του "Subsystem" με ένα "gain" που μετατρέπει τα εικονοστοιχεία pixels σε εκατοστά cm.



Εικόνα 3.55 Απόκτηση τετμημένης και μετατροπή από pixels σε cm

Την έξοδο του "Gain" τη συνδέουμε στην ανατροφοδότηση του ελεγκτή μας και το σύστημά μας είναι ολοκληρωμένο.



Εικόνα 3.56 Ολοκληρωμένο σύστημα Ball on Beam με camera ως ανατροφοδότηση

3.3.5 Αποτελέσματα και συμπεράσματα αποκρίσεων για το Ball on Beam

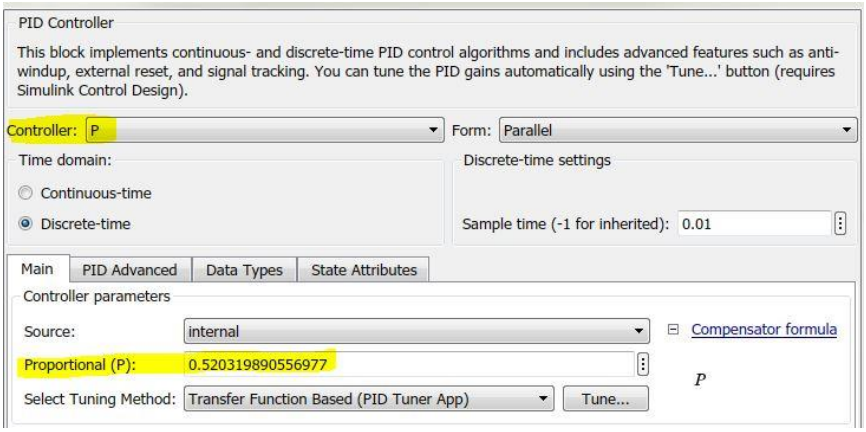
Στην ενότητα αυτή παραθέτουμε τις παρατηρήσεις και τα συμπεράσματα της συγκεκριμένης διεργασίας, για τους παρακάτω ελεγκτές ακολουθώντας τη διαδικασία "tuning" του Simulink, όπως στην ενότητα 3.2.5..

Αναλογικός, Ολοκληρωτικός, Διαφορικός, Αναλογικός-Ολοκληρωτικός, Αναλογικός-Διαφορικός και Αναλογικός-Ολοκληρωτικός-Διαφορικός ελεγκτής.

Οι παράμετροι των ελεγκτών καθορίζονται με στόχο την ταχύτητα απόκρισης, την ακρίβεια και την ευστάθεια του συστήματος.

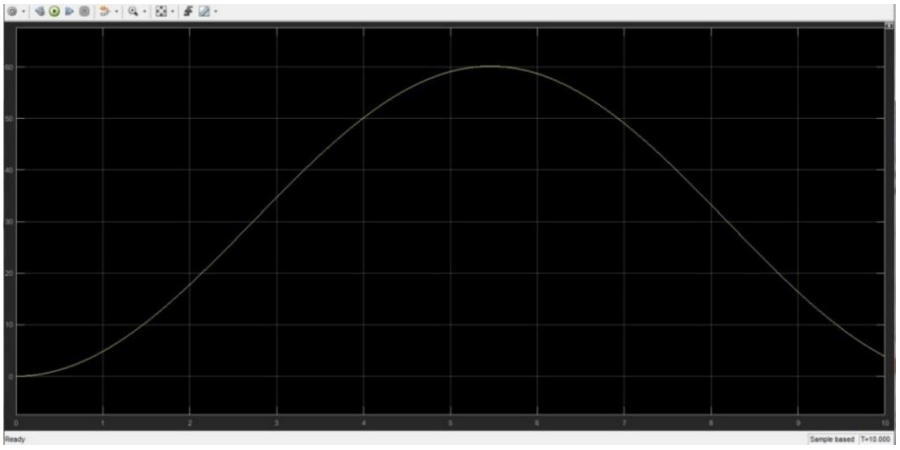
3.3.5.1 Αναλογικός ελεγκτής P

Ακολουθώντας την ίδια διαδικασία με την υποενότητα 3.2.5.1 ρυθμίσαμε το block "ελεγκτής" να χρησιμοποιεί μόνο την παράμετρο P. Με τη λειτουργία "tune" επιλέξαμε την απόκριση στον χρόνο (sec.) ώστε να έχουμε τη μέγιστη ακρίβεια και η τιμή της παραμέτρου P που προκύπτει είναι 0.520319890556977.



Εικόνα 3.57 Ρύθμιση P ελεγκτή για το Ball on Beam

Για είσοδο 30 cm κατά την εκτέλεση του Simulink λάβαμε την παρακάτω απόκριση.

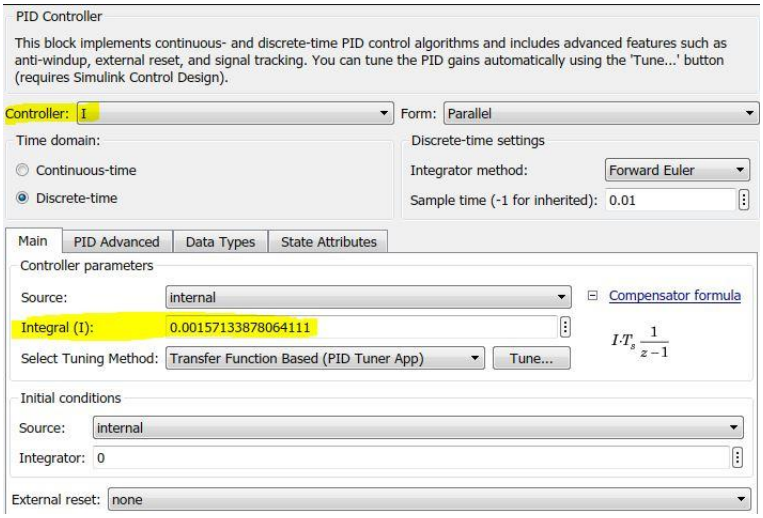


Εικόνα 3.58 Απόκριση P ελεγκτή του Ball on Beam

Παρατηρούμε ότι η απόκριση φτάνει στο διπλάσιο της εισόδου και δε σταθεροποιείται ποτέ. Ο ελεγκτής αυτός κρίνεται ακατάλληλος γιατί προσφέρει αστάθεια στο σύστημα.

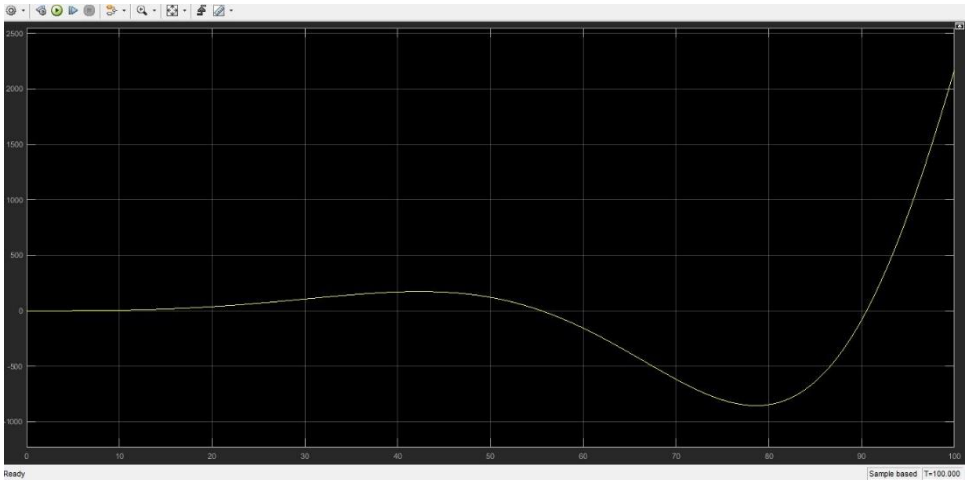
3.3.5.2 Ολοκληρωτικός ελεγκτής I

Ρυθμίσαμε το block "ελεγκτής" να χρησιμοποιεί μόνο την παράμετρο I. Με τη λειτουργία tune επιλέξαμε την αργότερη απόκριση στον χρόνο (sec.) για να πετύχουμε μεγαλύτερη ακρίβεια και η τιμή της παραμέτρου I που προκύπτει είναι 0.00157133878064111.



Εικόνα 3.59 Ρύθμιση I ελεγκτή για το Ball on Beam

Για την ίδια είσοδο 60 cm κατά την εκτέλεση του Simulink λάβαμε την παρακάτω απόκριση.



Εικόνα 3.60 Απόκριση I ελεγκτή του Ball on Beam

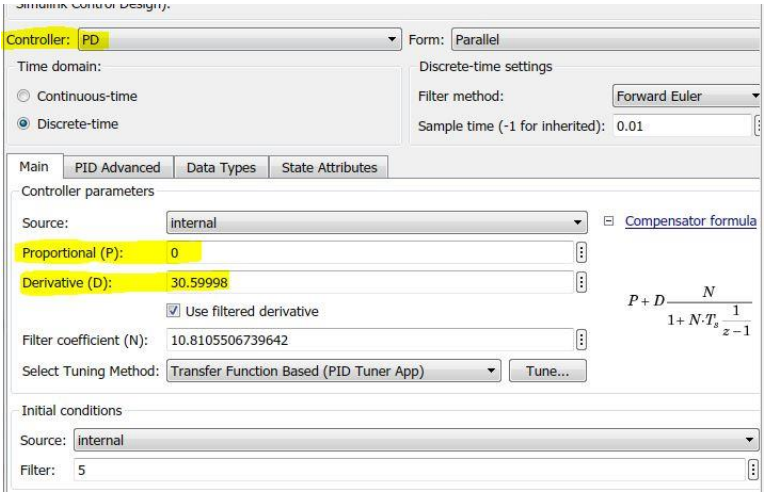
Παρατηρούμε ότι το σύστημά μας με αυτόν τον ελεγκτή παρουσιάζει αστάθεια διότι με το πέρας του χρόνου δεν ισορροπεί ποτέ στην τιμή της εισόδου (60 cm), αντίθετα όλο και

αυξάνεται. Συμπεραίνουμε ότι ο Ολοκληρωτικός ελεγκτής I είναι ακατάλληλος για το σύστημά μας.

3.3.5.3 Διαφορικός ελεγκτής D

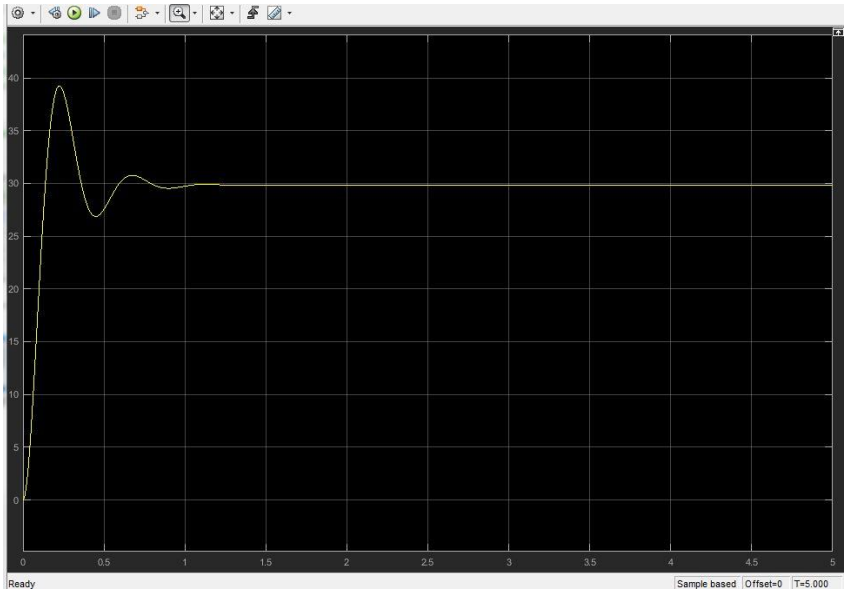
Ακολουθήσαμε τη συνηθισμένη διαδικασία για να ρυθμίσουμε το block "ελεγκτής" να χρησιμοποιεί μόνο την παράμετρο D. Επειδή το block του Simulink δε μας δίνει τη δυνατότητα αυτή, επιλέξαμε έναν PD ελεγκτή, κάναμε tune για μια απόκριση που η παράμετρος P ήταν πολύ κοντά στο μηδέν. Ύστερα μηδενίσαμε την παράμετρο P και κρατήσαμε μόνο την παράμετρο D που ήταν:

Παράμετρος D: 30.59998



Εικόνα 3.61 Ρύθμιση D ελεγκτή για το Ball on Beam

Για την ίδια είσοδο 30 cm κατά την εκτέλεση του Simulink λάβαμε την παρακάτω απόκριση.



Εικόνα 3.62 Απόκριση D ελεγκτή του Ball on Beam

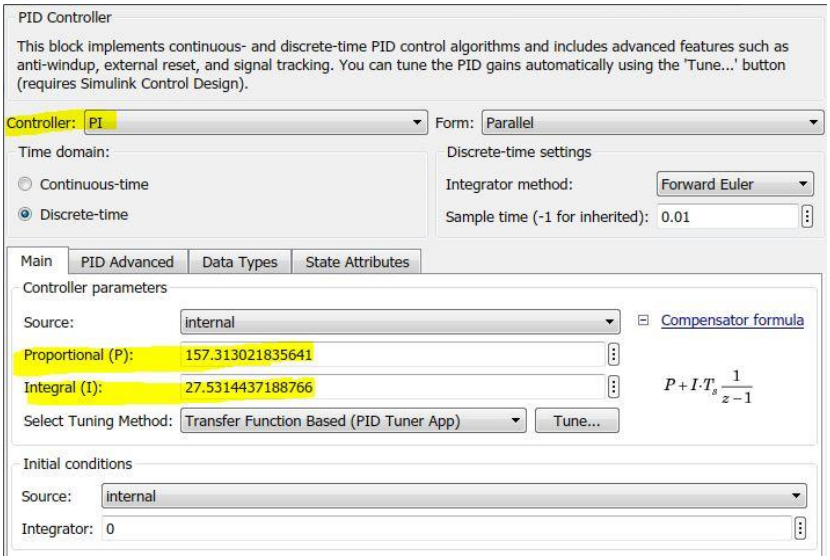
Παρατηρούμε ότι το σύστημα με αυτόν τον ελεγκτή παρουσιάζει μία υπερύψωση άνω του 20% της εισόδου (36 cm) από 0.14 sec έως 0.354 sec. Στη συνέχεια ακολουθούν κάποιες καμπυλώσεις γύρω από την τιμή της εισόδου και καταλήγουμε να ισορροπεί το σύστημα μετά το 1.5 sec.

3.3.5.4 Αναλογικός - Ολοκληρωτικός ελεγκτής PI

Ρυθμίσαμε το block "ελεγκτής" να χρησιμοποιεί μόνο την παράμετρο PI. Με τη λειτουργία tune επιλέξαμε την ταχύτερη απόκριση στον χρόνο (sec.) και την καλύτερη μεταβατική συμπεριφορά.

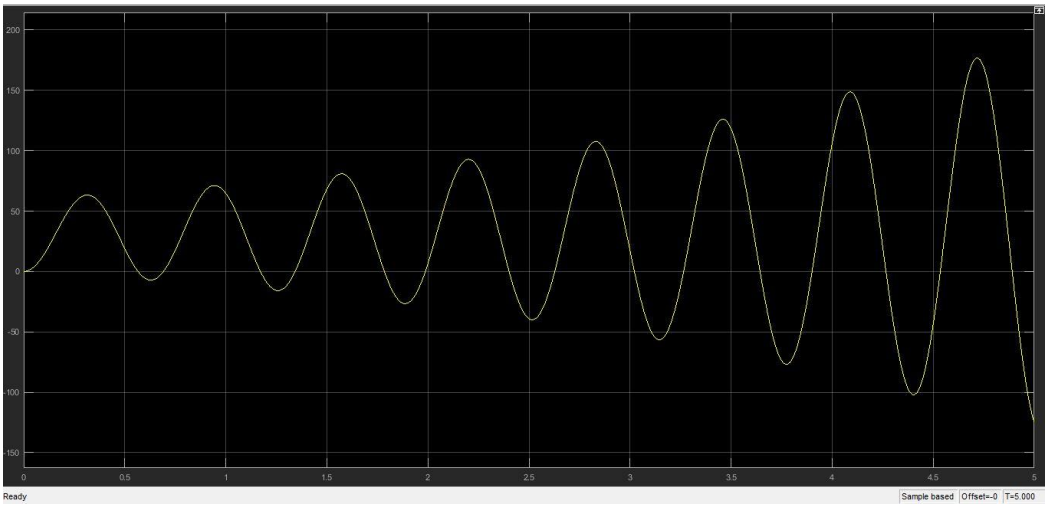
Οι τιμές των παραμέτρων PI που προκύπτουν:

Παράμετρος P:	**157.313021835641**
Παράμετρος I:	**27.5314437188766**



Εικόνα 3.63 Ρύθμιση PI ελεγκτή για το Ball on Beam

Για είσοδο 30 cm κατά την εκτέλεση του Simulink λάβαμε την παρακάτω απόκριση.



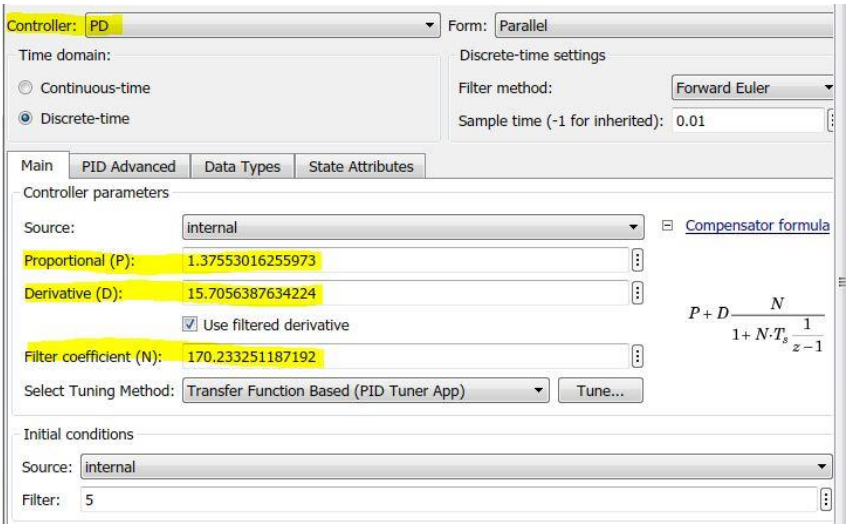
Εικόνα 3.64 Απόκριση PI ελεγκτή του Ball on Beam

Παρατηρούμε ότι το σύστημα με τον συγκεκριμένο τύπο ελεγκτή παρουσιάζει αστάθεια και άρα είναι ακατάλληλος για αυτό το σύστημα.

3.3.5.5 Αναλογικός - Διαφορικός ελεγκτής PD

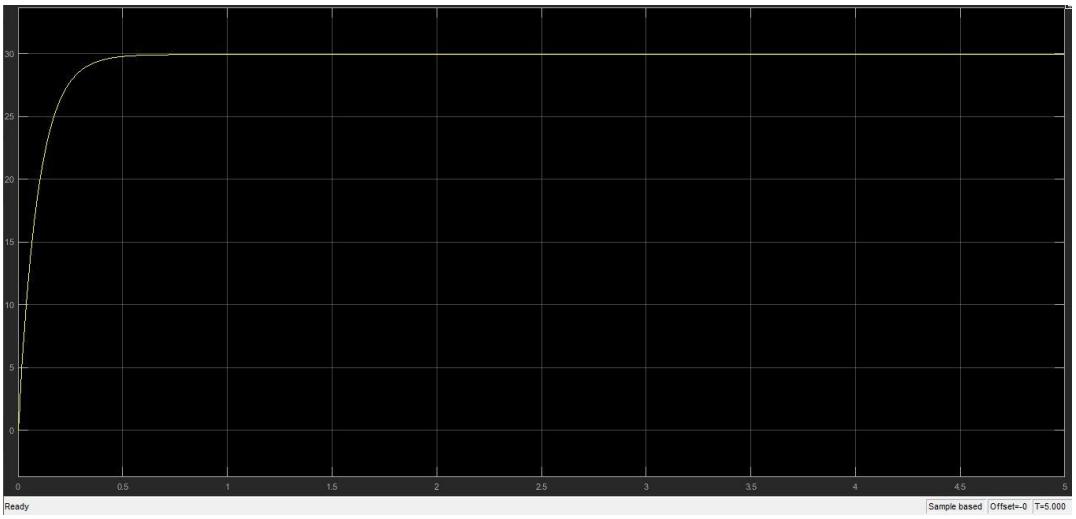
Ρυθμίσαμε το block "ελεγκτής" να χρησιμοποιεί μόνο την παράμετρο PD. Με τη λειτουργία tune επιλέξαμε την ταχύτερη απόκριση στον χρόνο (sec.) και τη μεταβατική συμπεριφορά, ώστε να πετύχουμε τη μεγαλύτερη ακρίβεια. Οι τιμές των παραμέτρων PD είναι:

Παράμετρος P:	**1.37553016255973**
Παράμετρος D:	**15.7056387634224**



Εικόνα 3.65 Ρύθμιση PD ελεγκτή για το Ball on Beam

Για είσοδο 30 cm κατά την εκτέλεση του Simulink λάβαμε την παρακάτω απόκριση.



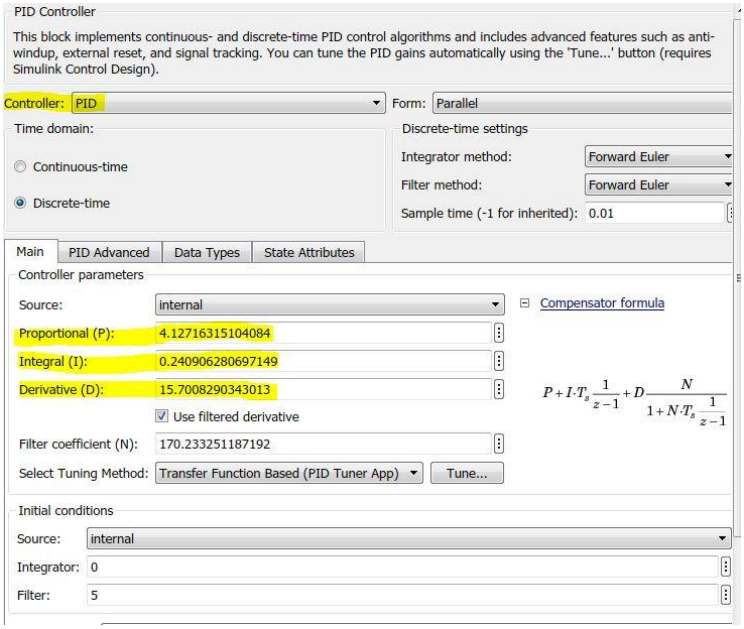
Εικόνα 3.66 Απόκριση PD ελεγκτή του Ball on Beam

Παρατηρούμε ότι η απόκριση δεν έχει υπερύψωση και ότι στο 1sec έχει πάρει η έξοδος την τιμή της εισόδου.

3.3.5.6 Αναλογικός – Ολοκληρωτικός - Διαφορικός ελεγκτής PID

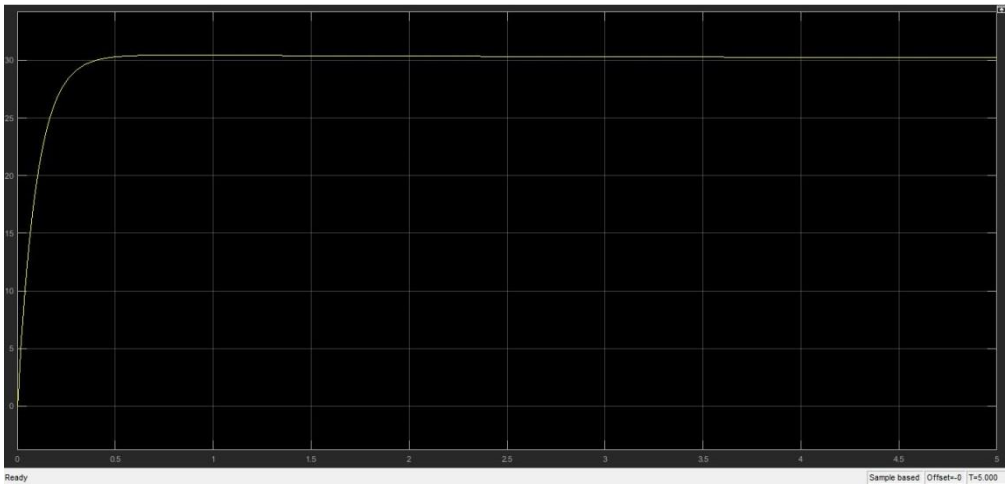
Ρυθμίσαμε το block "ελεγκτής" να χρησιμοποιεί μόνο την παράμετρο PID. Με τη λειτουργία tune επιλέξαμε τη μεταβατική συμπεριφορά, ώστε να πετύχουμε τη μεγαλύτερη ακρίβεια και την ταχύτερη απόκριση στον χρόνο (sec.) . Οι τιμές των παραμέτρων PID είναι:

Παράμετρος P:	**4.12716315104084**
Παράμετρος I:	**0.240906280697149**
Παράμετρος D:	**15.7008290343013**



Εικόνα 3.67 Ρύθμιση PID ελεγκτή για το Ball on Beam

Για είσοδο 30 cm κατά την εκτέλεση του Simulink λάβαμε την παρακάτω απόκριση.



Εικόνα 3.68 Απόκριση PID ελεγκτή του Ball on Beam

Παρατηρούμε ότι η απόκριση προσπερνά την τιμή της εισόδου στο 0.388 sec. και ύστερα τείνει να την ισοφαρίσει. Το προσπέρασμα είναι κατά 0.16% της τιμής εισόδου.

Αφού είδαμε τις αποκρίσεις όλων των πιθανών ελεγκτών για τη διεργασία μας, καταλήγουμε σε κάποια συμπεράσματα.

- Ο ελεγκτής με τη μεγαλύτερη ταχύτητα εκτέλεσης είναι ο PD ελεγκτής καθώς η απόκριση παίρνει την τιμή της εισόδου σε περίπου 0,5 δευτερόλεπτα. Ο μικρότερος χρόνος με τους υπόλοιπους ελεγκτές.

- Ο ελεγκτής με τη μεγαλύτερη ακρίβεια είναι ο PID ελεγκτής καθώς με μεγαλύτερη χρονική απόκριση μπορεί να πετύχει την ακριβή τιμή της εισόδου από τον PD ελεγκτή.
- Οι ελεγκτές που οδηγούν το σύστημα σε αστάθεια, όποιες τιμές κι αν πάρουν οι παράμετροί τους, είναι ο ελεγκτής P, ο ελεγκτής I και ο ελεγκτής PD.

Κεφάλαιο 4: Άλλες εφαρμογές και μελλοντική έρευνα

4.1 Μελλοντικές εφαρμογές με βάση τις διεργασίες που υλοποιήσαμε

Έχει παρατηρηθεί ότι σε γραμμές παραγωγής διαφόρων προϊόντων, μετά την ολοκλήρωση της παραγωγικής τους διαδικασίας, κάποια από τα προϊόντα κρίνονται ελαττωματικά. Σε μια γραμμή παραγωγής υπάρχουν πολλοί τρόποι που μπορεί το τελικό προϊόν να ελεγχθεί, κι αυτό εξαρτάται από τη δυνατότητα της κάθε επιχείρησης για λεπτομερή και γρήγορο έλεγχο. Για παράδειγμα σε μία γραμμή παραγωγής τον έλεγχο του τελικού προϊόντος μπορεί να τον κάνει ένας υπάλληλος, σε αντίθεση με κάποια άλλη γραμμή παραγωγής που για το ίδιο τελικό προϊόν τον κάνει ένας ρομποτικός βραχίονας σε συνεργασία με διάφορα αισθητήρια. Βέβαια, ο τρόπος ελέγχου έχει αμφίδρομη σχέση με την οικονομική ευχέρεια της κάθε επιχείρησης (και την ακρίβεια της μέτρησης που θέλει να κάνει).

Η μηχανική όραση είναι ένας κλάδος με ποικίλες δυνατότητες που όλο και περισσότερο βρίσκει εφαρμογή σε γραμμές παραγωγής ανά τον κόσμο. Παρέχει λεπτομερέστερη μελέτη και ανάλυση του παραγόμενου προϊόντος, με σχεδόν μηδενικό περιθώριο σφάλματος, μέσα σε πολύ λίγο χρόνο σε σχέση με την ανθρώπινη όραση. Επιτυγχάνεται με τη χρήση κάμερας ή αισθητήρων, με έναν υπολογιστή ή PLC και με κατάλληλο software.

Μία τέτοια δυνατότητα είναι η αναγνώριση χρωμάτων (color detection) που μας βοήθησε, στην προηγούμενη ενότητα (ενότητα 3.3.4), ως ανάδραση, για τον υπολογισμό της ακριβής θέσης της μπάλας πάνω στη ράβδο. Οι εφαρμογές που σκεφτήκαμε με την παρούσα λειτουργία είναι:

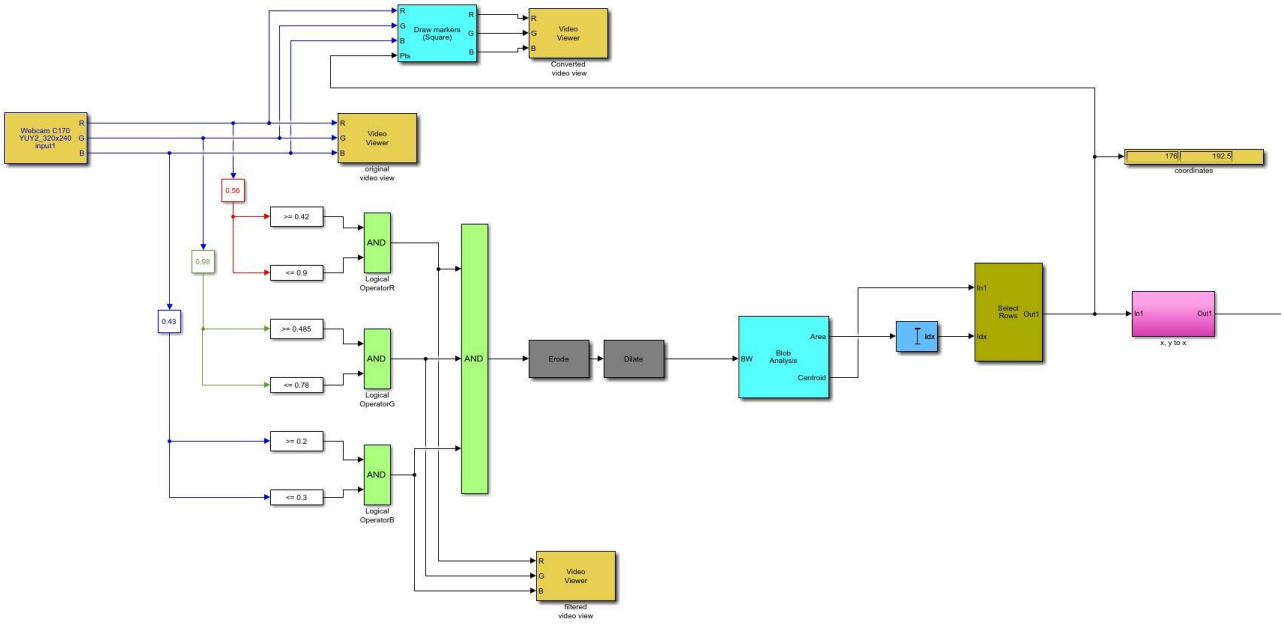
- **Έλεγχος ποιότητας βαψίματος προϊόντων μετά τη διαδικασία βαφής τους**
- **Διαχωρισμός προϊόντων πολλαπλών χρωμάτων σε διαφορετικές διεργασίες**

Οι δύο αυτές εφαρμογές μπορούν να βρουν χρήση από τη βιομηχανία τροφίμων μέχρι και τη βιομηχανία πλαστικών, υφασμάτων και μετάλλων.

Μπορούμε, λοιπόν, κάλλιστα να χρησιμοποιήσουμε τη δομή του προηγούμενου blank model κάνοντας μερικές μετατροπές, ώστε να μας βοηθήσει σε κάποιες εφαρμογές γραμμών παραγωγής.

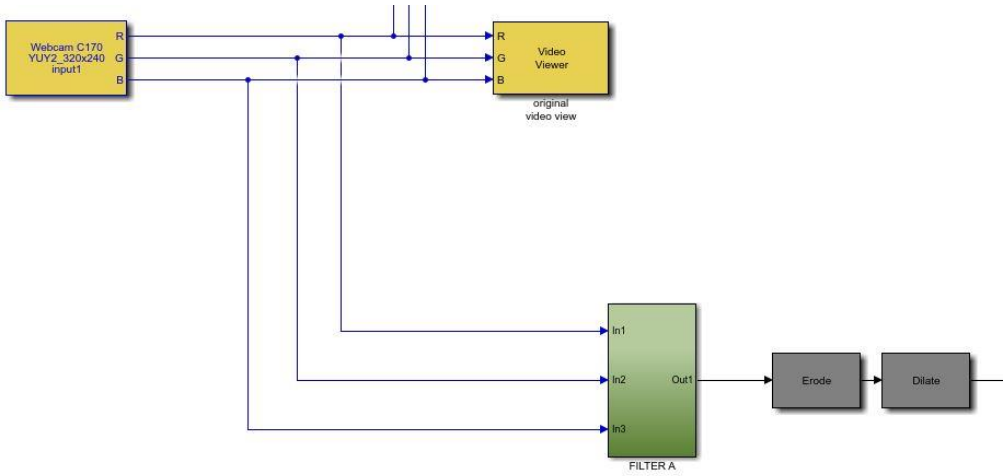
4.1.1 Έλεγχος ποιότητας βαψίματος προϊόντων μετά τη διαδικασία βαφής τους

Για την πρώτη εφαρμογή λοιπόν, χρησιμοποιούμε το προηγούμενο blank model αφαιρώντας όλα τα block στοιχεία μετά το block "x, y to x" και κρατώντας όλα τα υπόλοιπα blocks.



Εικόνα 4.1 Το σύστημα Ball on Beam μέχρι το block "x , y to x".

Επιλέγουμε τα blocks που κάνουν το φιλτράρισμα της εικόνας και τα ενοποιούμε σε ένα subsystem και το ονομάζουμε "FILTER"

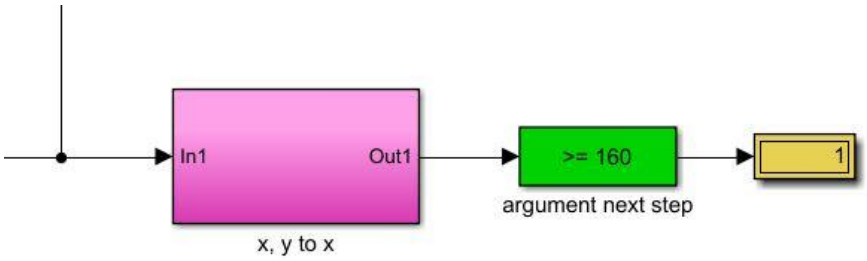


Εικόνα 4.2 Δημιουργία subsystem "FILTER".

Στο “FILTER” ρυθμίζοντας κατάλληλα τα “slider gain” και τα “compare to constant” blocks, επιλέγουμε το χρώμα που επιθυμούμε να αναγνωρίζει η κάμερα και να διεγείρεται το σύστημα.

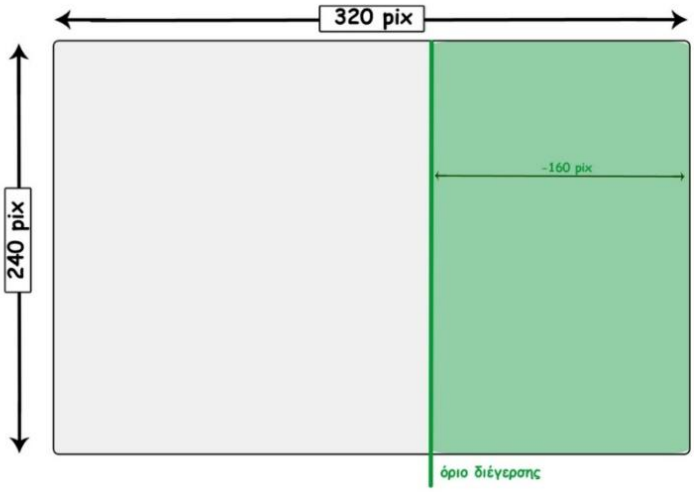
Ας υποθέσουμε ότι σε μία γραμμή παραγωγής, ένα απάρτημα κινείται πάνω σε έναν ταινιόδρομο. Ο ταινιόδρομος το περνά από διάφορα στάδια διεργασίας ώστε να βγει το παραγόμενο προϊόν στην τελική του μορφή. Μπορεί ένα από αυτά τα στάδια να είναι κάποιος φούρνος βαφής ή δωμάτιο αμμοβολής ή βαφείο υφασμάτων ή φούρνος ψησίματος. Στο επόμενο στάδιο ο ταινιόδρομος περνά το προϊόν μέσα από έναν χώρο (π.χ. κουτί) που έχει σταθερό φωτισμό. Εκεί μέσα είναι τοποθετημένη μια κάμερα και καταγράφει το κάθε αντικείμενο που περνά από μπροστά της. Η εικόνα αναλύεται στο Simulink και ανάλογα με τη ρύθμιση του φίλτρου, επιλέγουμε το εύρος χρωμάτων που θέλουμε να αναγνωρίζει το σύστημα ως λειτουργικό ή ελαττωματικό προϊόν. Το πρόγραμμα κλειδώνει το προϊόν αυτό δίνοντάς του συντεταγμένες πάνω στην εικόνα. Μόλις ο ταινιόδρομος πάει το προϊόν σε σημείο που μπορεί να αλλάξει διεργασία (π.χ κάδος απορριμμάτων ή άλλος ταινιόδρομος), το Simulink δείχνει ότι έχει πάρει συγκεκριμένες συντεταγμένες. Στη συνέχεια ένα έμβολο ή ένα μοτέρ παίρνει εντολή να μετακινήσει το προϊόν στην επόμενη διεργασία.

Για να μπορέσει το Simulink να στείλει εντολή στο μικροϋπολογιστή ή στο PLC που κινεί το μοτέρ ή το έμβολο προσθέτουμε στην έξοδό του ένα block “compare to constant” και επιλέγουμε την τετμημένη που θέλουμε να αλλάξει διεργασία το προϊόν.



Εικόνα 4.3 Όρισα συγκεκριμένου ορίου διέγερσης

Όταν η τετμημένη πάρει την τιμή που θέλουμε (στο παράδειγμά μας είναι 160 pixels) τότε η έξοδος του “compare to constant” block βγάζει τιμή «1», ενώ σε άλλη περίπτωση τιμή «0». Η τιμή «1» σηματοδοτεί την ενεργοποίηση της διέγερσης.

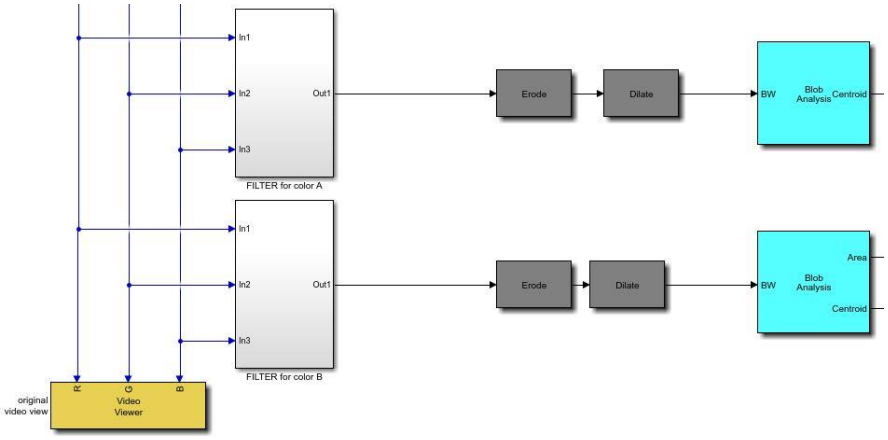


Εικόνα 4.4 Παράδειγμα διέγερσης στα 160 pixels.

Έτσι, σε μια γραμμή παραγωγής, μπορεί να επιτευχθεί ο έλεγχος ποιότητας βαφής προϊόντων ή πιο γενικά ο διαχωρισμός ενός συγκεκριμένου χρώματος αντικειμένου ανάμεσα σε πολλά άλλα.

4.1.2 Διαχωρισμός προϊόντων πολλαπλών χρωμάτων σε διαφορετικές διεργασίες

Η δεύτερη εφαρμογή δε διαφέρει πολύ από την πρώτη καθώς το σύστημα μάς παρέχει περισσότερη πληροφορία. Για να μπορέσει να αναγνωρίσει το κάθε χρώμα, θα χρειαστεί να εισαγάγουμε ένα φίλτρο για το κάθε χρώμα που επιθυμούμε.

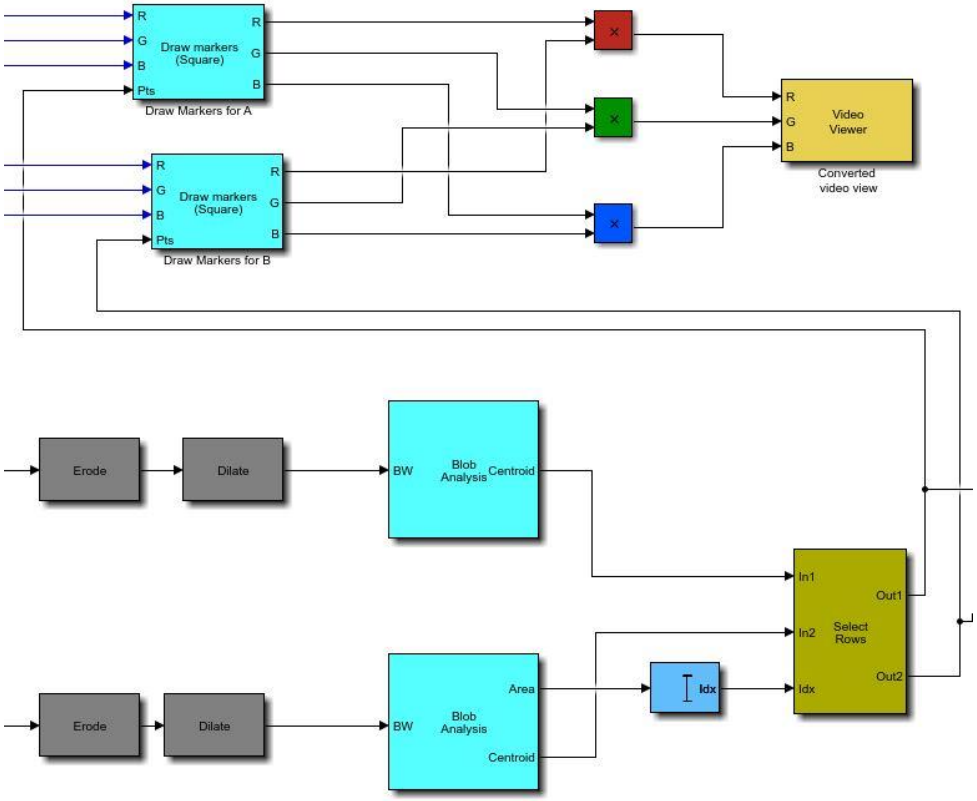


Εικόνα 4.5 Κατάτμηση εικόνας με βάση το χρώμα για δύο αντικείμενα

Χρησιμοποιώντας το προηγούμενο μοντέλο Simulink προσθέτουμε ένα ακόμα φίλτρο, ένα “erode”, “dilate” block και ένα “blob Analysis” block και τα συνδέουμε μεταξύ τους όπως παρακάτω.

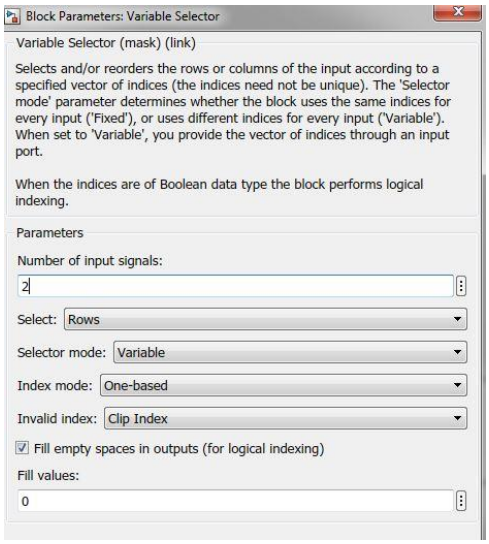
Στο “blob Analysis” block κλικάρουμε μόνο την επιλογή “centroid”, διότι στο επόμενο block που είναι το “Variable Selector” θα πρέπει να μπουν τρία σήματα. Η συντεταγμένη του A

προϊόντος, η συντεταγμένη του B προϊόντος και ο χώρος "Area" που δραστηριοποιούνται. Θα είχαμε το ίδιο αποτέλεσμα αν κρατάγαμε το "Area" και το "Centroid" από το A και το "Centroid" από το B.



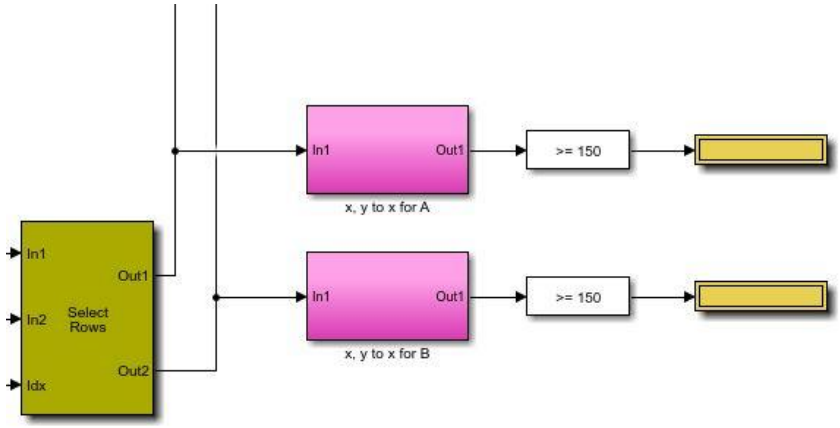
Εικόνα 4.6 Διαδικασία αναγνώρισης δύο αντικειμένων μετά από κατάτμηση εικόνας

Στο "Variable Selector" block επιλέγουμε δύο εισερχόμενα σήματα και αυτόματα θα έχουμε και δύο εξόδους, όπου η κάθε έξοδος θα είναι οι συντεταγμένες του κάθε αντικειμένου.



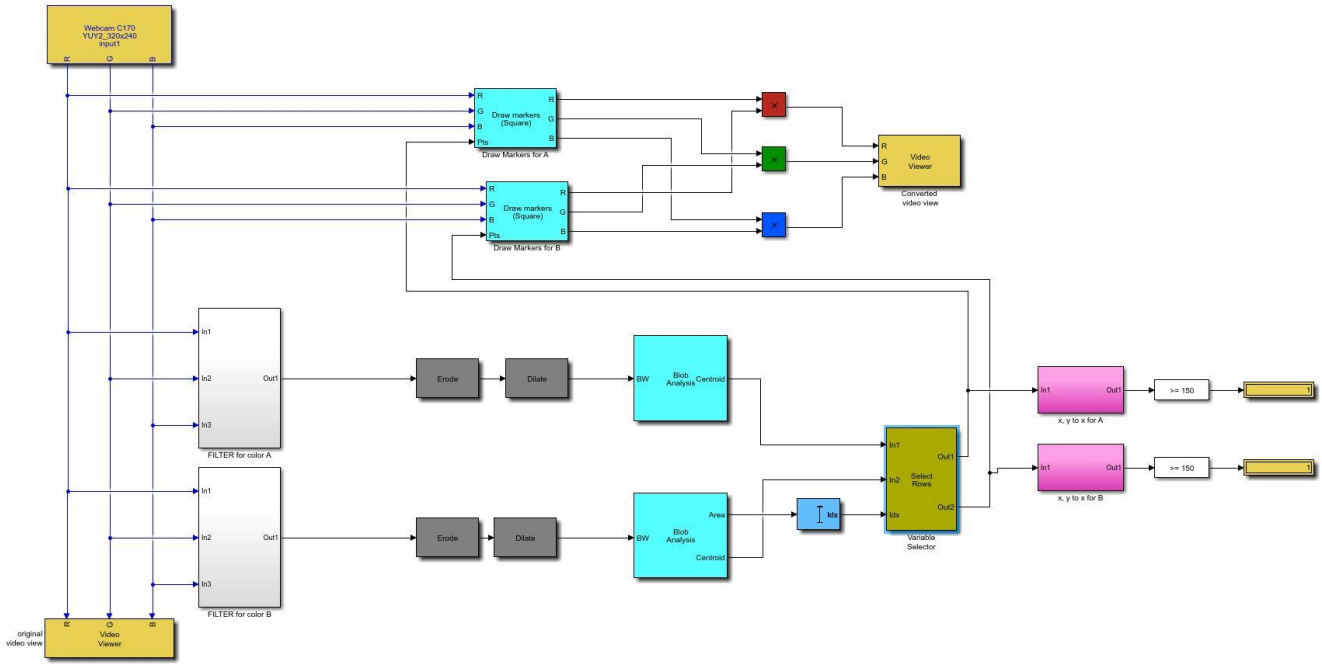
Εικόνα 4.7 Ρυθμίσεις "Variable Selector" block για δύο σήματα

Η κάθε έξοδος θα οδηγηθεί σε ένα “Draw Maker” block για την απεικόνιση της θέσης του κάθε αντικειμένου πάνω στη ζωντανή εικόνα. Ύστερα οι χρωματικοί έξοδοι θα συνδεθούν μεταξύ τους με “product” και στη συνέχεια σε ένα “Video Viewer”. Την κάθε έξοδο του “Variable Selector” block τη χρησιμοποιούμε για να ορίσουμε σε ποιες συντεταγμένες θα μπει το προϊόν στην επόμενη διεργασία με τη βοήθεια μικροϋπολογιστή ή PLC.



Εικόνα 4.8 Όρισα συγκεκριμένου ορίου διέγερσης δύο αντικειμένων

Το ολοκληρωμένο Simulink αρχείο για την αναγνώριση δύο χρωμάτων φαίνεται παρακάτω.



Εικόνα 4.9 Ολοκληρωμένο μοντέλο αναγνώρισης δύο αντικειμένων

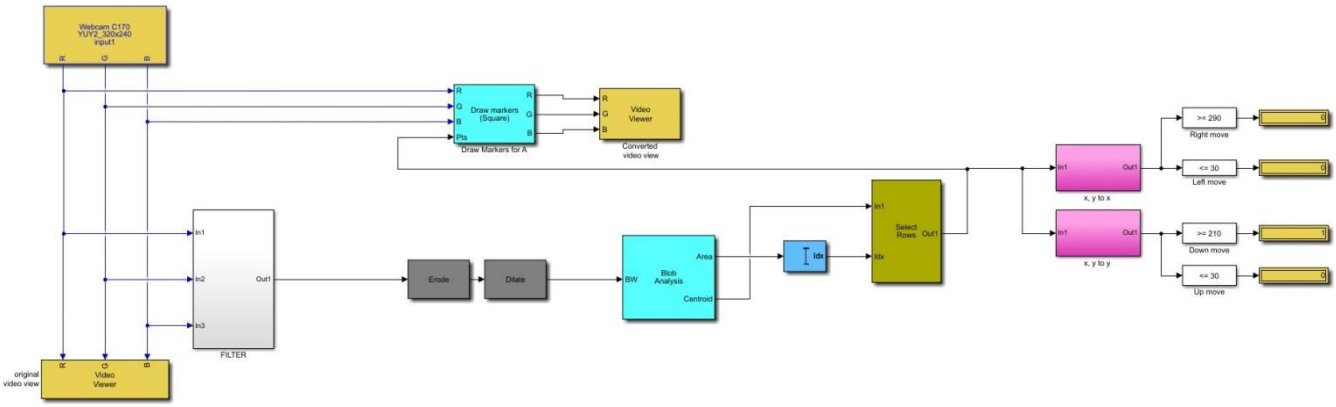
Σε περίπτωση που θέλουμε παραπάνω χρώματα, απλά προσθέτουμε φίλτρα, αυξάνουμε τον αριθμό εισόδων του "Variable Selector" block, προσθέτουμε "Draw maker" blocks και συνδέουμε τις εξόδους τους με "product" τόσες φορές, ώστε το κάθε χρώμα να έχει μία σύνδεση με το "Video Viewer".

Στη βιομηχανία όμως δε γίνεται χρήση του προγράμματος Matlab και άρα του Simulink. Αντίθετα, είναι πιο συνήθης η χρήση αρχείων που έχουν κατάληξη exe. Γι' αυτό προτείνουμε μετά τη δημιουργία του μοντέλου Simulink να χρησιμοποιηθεί το Simulink Coder ή Real-Time Workshop, μια εφαρμογή του Simulink, ώστε να μετατραπεί το μοντέλο Simulink σε κώδικα C. Αυτόν τον κώδικα τον καλούμε από την εφαρμογή GUI (Graphical User Interface) και λαμβάνουμε το μοντέλο μας σε αρχείο exe.

Βέβαια το ίδιο Simulink αρχείο μπορεί να χρησιμοποιηθεί και σε άλλες εφαρμογές του τύπου: «αν δει η κάμερα αυτό το χρώμα κάνε κάτι». Για παράδειγμα, αν η κάμερα αναγνωρίσει το πράσινο χρώμα τότε να δίνει εντολή το σύστημα να ξεκινά ένα μοτέρ, ενώ αν αναγνωρίσει το κόκκινο χρώμα τότε το σύστημα να σταματά το μοτέρ.

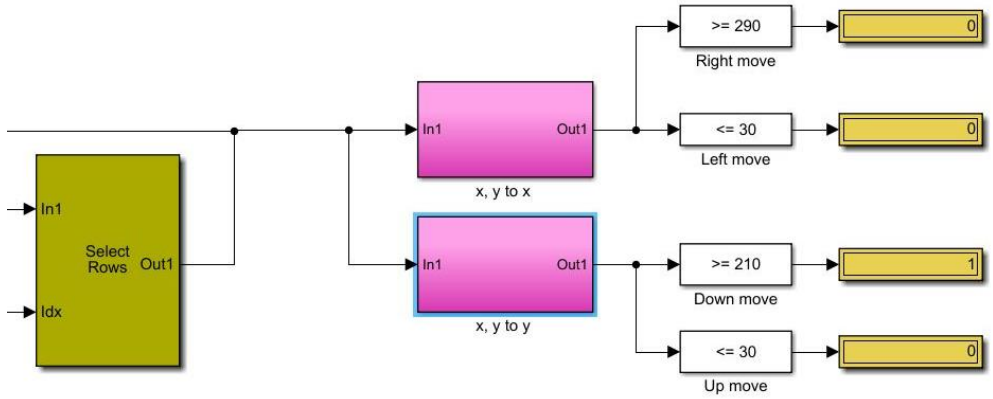
4.1.3 Παρακολούθηση συγκεκριμένου χρώματος (color tracking)

Μια τρίτη εφαρμογή που σκεφτήκαμε, είναι η παρακολούθηση χρώματος (color tracking). Στην παρούσα εφαρμογή, αν το χρώμα που διεγείρει την κάμερα πάει να ξεφύγει από την εικόνα, τότε το σύστημα περιστρέφει την κάμερα προς το μέρος του αντικειμένου αυτού με τη χρήση κάποιων μοτέρ. Αυτό μπορεί να είναι ένας ρομποτικός βραχίονας ή οι ρόδες ενός αυτόνομου αμαξιδίου. Στο παρακάτω Simulink αρχείο φαίνεται η σύνδεση των "block" που θα χρησιμοποιήσουμε για την παρακολούθηση συγκεκριμένου χρώματος από ρομποτικό βραχίονα με δύο βαθμούς ελευθερίας.



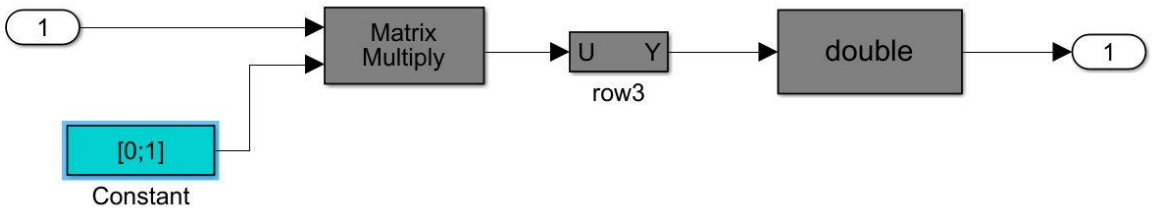
Εικόνα 4.10 Block διάγραμμα αναγνώρισης χρώματος και διέγερσης συγκεκριμένων συντεταγμένων

Το παραπάνω σύστημα παρατηρούμε ότι είναι σχεδόν ολόιδιο με την πρώτη εφαρμογή (αναγνώριση ενός χρώματος), με διαφορά την έξοδο του συστήματος. Έχουμε προσθέσει ένα "block" που μας δίνει την τετμημένη y και κάποια "compare to constant" blocks για την κάθε κίνηση που μπορεί να κάνει ο ρομποτικός βραχίονας, περιστροφή αριστερά-δεξιά και κίνηση πάνω-κάτω.



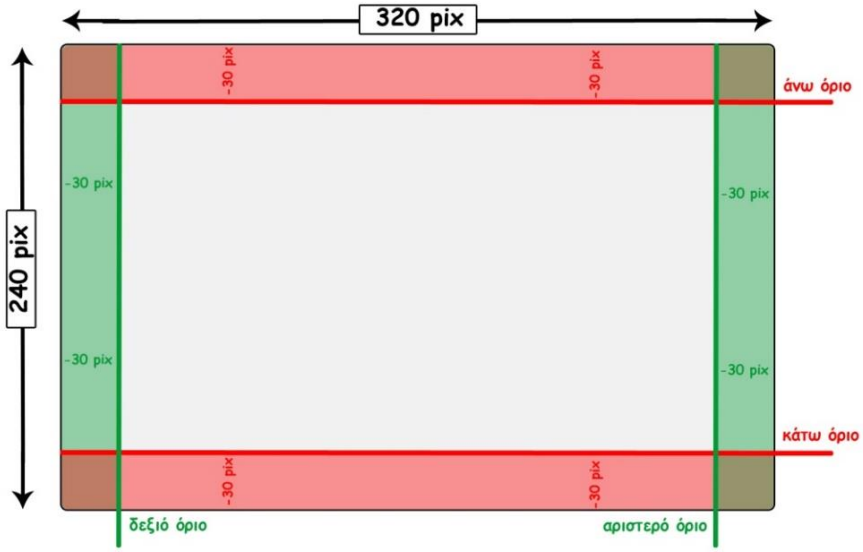
Εικόνα 4.11 Όρισα συγκεκριμένου ορίου διέγερσης τετμημένης και τεταγμένης χρώματος

Το block "x,y to y" είναι ίδιο με το "x,y to x" με τη διαφορά στο "constant" να είναι [0;1] και όχι [1;0].



Εικόνα 4.12 Απόκτηση μόνο τεταγμένης από τις συντεταγμένες θέσης

Οι τιμές που δίνουμε στα "compare to constant" blocks είναι τα όρια που βάζουμε στην εικόνα για την ενεργοποίηση της κάθε κίνησης. Η εικόνα που χρησιμοποιούμε εμείς είναι 320x240 pixels. Η κίνηση προς τα δεξιά ενεργοποιείται όταν το χρωματικό αντικείμενο φτάσει στα 290 pixels, δηλαδή 30 pixels πριν την έξοδό του από την εικόνα. Το ίδιο ισχύει για κάθε κίνηση.



Εικόνα 4.13 Παράδειγμα διέγερσης συγκεκριμένων ορίων

Με λίγα λόγια, όταν το αντικείμενο βρίσκεται στην γκρι περιοχή δεν κινείται ο βραχίονας. Όταν μεταβεί σε μία από τις κόκκινες περιοχές κινείται πάνω-κάτω αντίστοιχα, ενώ όταν μεταβεί σε μία από τις πράσινες κινείται δεξιά-αριστερά.

Επίλογος

Στο κομμάτι αυτό ολοκληρώσαμε τη μελέτη μας πάνω στο θέμα των εφαρμογών της μηχανικής όρασης. Καταλήξαμε σε συμπεράσματα για το ποιος είναι ο καταλληλότερος τρόπος ελέγχου τόσο στην ενότητα του DC μοτέρ όσο και στην αναγνώριση της μπάλας στη δοκό (Ball on Beam) και μπορούμε πλέον να παραμετροποιήσουμε τις εφαρμογές μας με βάση τις επιθυμίες μας. Ο έλεγχος αυτός που ακολουθήσαμε (PID) μέσω της μηχανικής όρασης μάς βοήθησε να αντιληφθούμε στο κομμάτι αυτό των σπουδών μας, πόσο σημαντικός είναι για έναν μηχανικό αυτοματισμού να μπορεί να κατανοήσει, να ελέγξει, να παραμετροποιήσει και να τα εφαρμόσει σε ό,τι εφαρμογές και εργασίες του ζητηθούν τα παραπάνω στοιχεία. Η επιστήμη του αυτοματισμού έχει αρχίσει και ακμάζει ραγδαία καθώς οι συνθήκες της αγοράς το απαιτούν και πρέπει μέσω τέτοιων εφαρμογών και μελετών να εξελίσσεται όσο το δυνατόν περισσότερο.

Βιβλιογραφία

1. Μπελέρης Β-Σ. «Μηχανική όραση», Πτυχιακή εργασία, Τμήμα Μηχανικών Αυτοματισμού, Σχολή Τεχνολογικών Εφαρμογών, Α.Ε.Ι. Πειραιά Τ.Τ., σελ.6, 14-15,20
   http://okeanis.lib.puas.gr/xmlui/bitstream/handle/123456789/3074/auto_39545.pdf?sequence=1&isAllowed=y
   (προσπέλαση 07/1/2019)
2. ΚΕΣ 03 – Αναγνώριση Προτύπων και Ανάλυσης Εικόνας, Τμήμα Επιστήμης και Τεχνολογίας Τηλεπικοινωνιών, Πανεπιστήμιο Πελοποννήσου, σελ.2
   https://eclass.uop.gr/modules/document/file.php/TST191/lectures/pr03.pdf
3. Ξυλουργός Ν., «Κατάτμηση εικόνων υπερήχων για την εύρεση ανατομικών αναφορών», Πτυχιακή εργασία, Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων, Σχολή Τεχνολογικών Εφαρμογών, Τ.Ε.Ι. Κρήτης, σελ. 36-51
   https://apothesis.lib.teicrete.gr/bitstream/handle/11713/3089/Xylourgos2009.pdf?sequence=1
   (προσπέλαση 19/12/2018)
4. Satya Mallick, Image Recognition and Object Detection – Part 1, November 14 (2016)
   https://www.learnopencv.com/image-recognition-and-object-detection-part1/
   (προσπέλαση 07/1/2019)
5. Βλαχοστάθης Σ. (Ιούνιος 2014), Ανίχνευση ανθρώπου και παρακολούθηση της κίνησής του, Πτυχιακή Εργασία, Τμήμα Ηλεκτρονικής και Επεξεργασίας της Πληροφορίας, Πανεπιστήμιο Πατρών
   http://nemertes.lis.upatras.gr/jspui/bitstream/10889/8221/1/MSc_%CE%A0%CF%84%CF%85%CF%87%CE%B9%CE%B1%CE%BA%CE%B7_Final.pdf
   (προσπέλαση 29/12/2018)
6. Mark S. Nixon, Alberto S. Aguado in Feature Extraction & Image Prrocessing for Computer Vision (Third Edition), 2012, “Moving object detection and description”
   https://www.sciencedirect.com/topics/engineering/detail-image
   (Προσπέλαση 07/1/2019)
   https://www.sciencedirect.com/topics/engineering/detail-image

7. http://what-when-how.com/introduction-to-video-and-image-processing/blob-analysis-introduction-to-video-and-image-processing-part-1/ (προσπέλαση 14/1/2019)
8. Δημήτριος Ε. Παπαγεωργίου (Αθήνα, Ιούλιος 2011), «Έλεγχος οπτικής οδήγησης ρομποτικού βραχίονα με χρήση φίλτρου Kalman και φίλτρου σωματιδίων για παρακολούθηση κινούμενου στόχου», Πτυχιακή εργασία, Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Εθνικού Μετσόβιου Πολυτεχνίου, σελ. 30-34
9. Μπαρτζελιώτης Ε. (Σάμος, Απρίλιος 2017), «Ρομποτικό σύστημα με τρεις βαθμούς ελευθερίας που ακολουθεί μία γραμμή μέσω ελέγχου οπτικής ανατροφοδότησης (visual servo control)», Πτυχιακή εργασία, Τμήμα Μηχανικών Πληροφοριακών και Επικοινωνιακών Συστημάτων, Πανεπιστήμιο Αιγαίου, σελ.25-27 http://hellanicus.lib.aegean.gr/handle/11610/17197 (προσπέλαση 30/12/2018)
10. https://en.wikipedia.org/wiki/Blob_detection (προσπέλαση 14/1/2019)
11. https://el.wikipedia.org/wiki/Μηχανική_όραση (προσπέλαση 22/11/2018)
12. Δ. Καλλιγερόπουλος, Σ. Βασιλειάδου, '*Ιστορία της Τεχνολογίας & των Αυτομάτων*', Σύγχρονη Εκδοτική, Αθήνα, 2005.
13. J.C. Maxwell, "*On Governors*", Proceedings of the Royal Society of London, Vol.16, 1868.
14. J.C. Maxwell, "*On Governors*", Philosophical Magazine, Vol.35, pp.385-398, 1868
15. E.J. Routh, "*A treatise on the Stability of a Given State of Motion"*, MacMillan and Co., London 1877
16. H. Nyquist, "*Certain Topics in Telegraph Transmission Theory*", Transactions AIEE, Vol.47, pp. 617-644,1928
    H. Nyquist, *"Regeneration Theory*", Bell System Technical Journal, Vol.11, pp.126-147,1932
17. H.S. Black, "*Stabilized Feedback Amplifiers",* Bell System Technical Journal, 1934
    H.S. Black, *"Inverting the Negative Feedback Amplifier",* IEEE Spectrum, Vol.14, pp.54-60, 1977
18. Π.Ν. Παρασκευοπούλου, 'ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΜΑΤΟΥ ΕΛΕΓΧΟΥ' ΘΕΩΡΙΑ ΚΑΙ ΕΦΑΡΜΟΓΕΣ, ΤΟΜΟΣ Α: Σ.Α.Ε. ΣΥΝΕΧΟΥΣ ΧΡΟΝΟΥ, Εκδόσεις Marcel Dekker, ΑΘΗΝΑ, 2007 , σελ. 15 -22

19. MG995 High Speed Metal Gear Dual Ball Bearing Servo https://www.electronicoscaldas.com/datasheet/MG995_Tower-Pro.pdf (προσπέλαση 29/11/2018)
20. Στέφανος Οικονόμου. Υπουργείο Πολιτισμού, Παιδείας και Θρησκευμάτων, Ινστιτούτο Τεχνολογίας Υπολογιστών και εκδόσεων «ΔΙΟΦΑΝΤΟΣ», Μηχατρονική Γ΄ΕΠΑΛ, Συστήματα Δράσης. http://iep.edu.gr:8080/images/school_books/24-0530_Mixatroniki_G-EPAL_BM.pdf , σελ. 30 -37 (προσπέλαση 29/11/2018)
21. Tutorial – L298N Dual Motor Controller Module 2A and Arduino https://hackerstore.nl/PDFs/Tutorial298.pdf
22. HC – SR04 User Guide https://www.mpja.com/download/hc-sr04_ultrasonic_module_user_guidejohn.pdf (προσπέλαση 03/12/2018)
23. Εμμανουήλ Πουλάκης, (Ηράκλειο, Ιανουάριος 2015), Προγραμματίζοντας με τον μικροελεγκτή Arduino. http://users.sch.gr/manpoul/docs/arduino/ProgrammingArduino.pdf (προσπέλαση 29/11/2018)
24. Τσιτσώνης Α. & Τέρτης Γ. (Ιούνιος 2012), Αισθητήρες κίνησης και μετατόπισης, Πτυχιακή Εργασία, Τμήμα Ηλεκτρονικής, Τ.Ε.Ι. Κρήτης, σελ.54 -59 http://nefeli.lib.teicrete.gr/browse/sefe/hlk/2012/TritsonisAntonios,TertisGeorgios/attached-document-1342087665-580594-5982/TritsonisAntonios_TertisGeorgios2012.pdf (προσπέλαση 03/12/2018)
25. US Digital. H5 Ball Bearing Optical Shaft Encoder https://cdn.usdigital.com/assets/datasheets/H5_datasheet.pdf?k=636734135647339553 (προσπέλαση 03/12/2018)
26. Omron electronics, INC. EE-SX 198/199/1018/1025/1041/1042/1070/1071 https://www.mouser.com/ds/2/307/sx1025-1190149.pdf (προσπέλαση 03/12/2018)
27. Logitech https://www.logitech.com/el-gr/product/webcam-c170 (προσπέλαση 03/12/2018)
28. SparkFun Electronics [US]. ATMEL. 8-bit AVR Microcontroller with 4/8/16/32K Bytes In-System Programmable Flash, ATMega328.

https://www.sparkfun.com/datasheets/Components/SMD/ATMega328.pdf
(προσπέλαση 28/11/2018)

29. https://deltahacker.gr/arduino-intro/
(προσπέλαση 28/11/2018)
30. https://en.wikipedia.org/wiki/Arduino_IDE
(προσπέλαση 28/11/2018)
31. https://www.arduino.cc/en/Guide/Environment
(προσπέλαση 28/11/2018)
32. Δρ. Μιχάλης Παρασκευάς. Τεχνολογικό Εκπαιδευτικό Ίδρυμα Δυτικής Ελλάδας. Τμήμα Μηχανικών Πληροφορικής Τ.Ε.. Σήματα και Συστήματα (Διάλεξη 3: Εισαγωγή στα Συστήματα).
http://opencourses.teiwest.gr/modules/document/file.php/CIED107/1.%20%CE%94%CE%B9%CE%B1%CE%BB%CE%AD%CE%BE%CE%B5%CE%B9%CF%82/Signals_and_Systems_Lectures_3.pdf .
(προσπέλαση 27/11/2018)
33. Χειράκης Δ. (Οκτώβριος 2011), Έλεγχος Ανάστροφου Εκκρεμούς, Πτυχιακή Εργασία, Τμήμα Ηλεκτρολογίας, Τ.Ε.Ι. Καβάλας
http://digilib.teiemt.gr/jspui/bitstream/123456789/1671/1/012011172.pdf
(προσπέλαση 27/11/2018)
34. Control Tutorials for Matlab & Simulink, DC Motor Position: System Modeling
http://ctms.engin.umich.edu/CTMS/index.php?example=MotorPosition&section=SystemModeling
(προσπέλαση 27/11/2018)
35. Control Tutorials for Matlab & Simulink, Ball & Beam: System Modeling
http://ctms.engin.umich.edu/CTMS/index.php?example=BallBeam&section=SystemModeling
(προσπέλαση 27/11/2018)

Παράρτημα

Κώδικας Arduino για Ball on Beam με το Ultrasonic αισθητήριο:

```
#include < Servo.h >
     Servo myservo1;
 const int servo_pin1 = 4;
 const int trigPin = 9;
 const int echoPin = 10;

 // defined variables for ultrasonic
 long duration;
 int distance;
 void setup() {
     pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
     pinMode(echoPin, INPUT); // Sets the echoPin as an Input
     Serial.begin(9600);
     myservo1.attach(servo_pin1); //the pin for the servo1 control
 }
 void loop() {
     digitalWrite(trigPin, LOW); // Clears the trigPin
     delayMicroseconds(2);
     digitalWrite(trigPin, HIGH); // Sets the trigPin on HIGH state for 10
micro seconds
     delayMicroseconds(10);
     digitalWrite(trigPin, LOW);
     duration = pulseIn(echoPin, HIGH); // Reads the echoPin, returns the
sound wave travel time in microseconds

     // Calculating the distance
     distance = duration * 0.034 / 2;
     delay(100);
     if (distance > 60) // 60 cm is the maximum position for the ball
     {
         distance = 60;
     }
     Serial.write(distance); // send to simulink the distance
     if (Serial.available() > 0) {
         int c = Serial.read(); //gets one byte from serial buffer
         byte x = byte(c);
         myservo1.write(x); // rotate servo motor
         delay(5);
     }
 }
```

Κώδικας Arduino για Ball on Beam με κάμερα:

```
#include < Servo.h >
    Servo myservo; // create servo object to control a servo
const int servo_pin = 4;
int servo_angle = 0;
void setup() {
    Serial.begin(9600);
    myservo.attach(servo_pin); //the pin for the servo control
}
void loop() {
    if (Serial.available() > 0) {
        int c = Serial.read(); //gets one byte from serial buffer
        byte x = byte(c); //makes the string readString
        myservo.write(x);
        delay(10);
    }
}
```

Κώδικας Arduino για μετρήσεις Volt-Time-RPM DC μοτέρ:

```
int hallsensor = 2; // Hall sensor at pin 2
volatile byte counter;
unsigned int rpm;
unsigned long passedtime;
float input_voltage = 0.0;
void isr() {
    //Each rotation, this interrupt function is run twice, so take that
into consideration for
    //calculating RPM
    //Update count
    counter++;
}
void setup() {
    Serial.begin(9600); //Intiates Serial communications
    attachInterrupt(0, isr, RISING); //Interrupts are called on Rise of
Input
    pinMode(hallsensor, INPUT); //Sets hallsensor as input
    counter = 0;
    rpm = 0;
    passedtime = 0; //Initialise the values
}
void loop() {
    delay(1000); //Update RPM every second
    detachInterrupt(0); //Interrupts are disabled
```

```
    rpm = 60 * 1000 / (millis() - passedtime) * counter;
    passedtime = millis();
    counter = 0;
    int analog_value = analogRead(A1);
    input_voltage = (analog_value * 5.0) / 1024.0;
    Serial.print("v= "); //Print Volt
    Serial.println(input_voltage);
    Serial.print("time    "); //Print time
    Serial.println(millis());
    Serial.print("RPM= ");
    Serial.println(rpm); //Print out result to monitor
    attachInterrupt(0, isr, RISING); //Restart the interrupt processing
}
```

Κώδικας Arduino για DC μοτέρ με photo interrupter encoder (Omron 1041):

```
const int photointerrupterPin = 2; // the pin that the photointerrupter is
attached to
const int ledPin = 13;
int photointerrupterState = 0; // current state of the photointerrupter
int photointerrupterLastState = 0;
int steps = 0;
byte tsteps = 0;
byte c = 0;
//Motor
const int motorPin1 = 5; // Pin 7 of L293
const int motorPin2 = 6; // Pin 2 of L293
const int motorenable = 9; // Pin 1 of L293
byte buffer = 00000000;
int length = 8;
void setup() {
    pinMode(photointerrupterPin, INPUT); // initialize the photointerrupter
pin as a input
    pinMode(ledPin, OUTPUT); // initialize the LED as an output
    Serial.begin(9600); // initialize serial communication
    //Set pins for L293 as outputs
    pinMode(motorPin1, OUTPUT);
    pinMode(motorPin2, OUTPUT);
    pinMode(motorenable, OUTPUT);
    digitalWrite(motorenable, LOW);
    analogWrite(motorPin1, 0);
    digitalWrite(motorPin2, LOW);
}
void loop() {
    digitalWrite(ledPin, LOW);
    digitalWrite(motorenable, LOW);
    analogWrite(motorPin1, 0);
```

```
    digitalWrite(motorPin2, LOW);
    if (Serial.available() > 0) {
        // read the incoming byte:
        c = Serial.read();
        byte t_angle = byte(c);
        double tsteps = t_angle / 4.5;
        while (steps < tsteps) {
            digitalWrite(ledPin, HIGH);
            digitalWrite(motorenable, HIGH);
            analogWrite(motorPin1, 210); //power supply 5v and L298 input
12v with conector
            digitalWrite(motorPin2, LOW);
            delay(5);
            // read the photointerrupter input pin:
            photointerrupterState = digitalRead(photointerrupterPin);
            // compare the photointerrupterState to its previous state
            if (photointerrupterState != photointerrupterLastState) {
                // if the state has changed, increment the score counter
                if (photointerrupterState == HIGH) {
                    // if the current state is HIGH then the
photointerrupter went from off to on:
                    steps++;
                } else {
                    // if the current state is LOW then the button
                    steps++;
                }
                delay(5); // Delay a little bit to avoid bouncing
            }
            // save the current state as the last state, for next time
through the loop
            photointerrupterLastState = photointerrupterState;
            Serial.write(steps);
        }
    }
}
```