

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ - ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ  
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ  
ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΠΟΛΟΓΙΣΤΩΝ



ΠΑΝΕΠΙΣΤΗΜΙΟ  
ΠΑΤΡΩΝ  
UNIVERSITY OF PATRAS

ΤΟΜΕΑΣ: ΤΟΜΕΑΣ ΣΥΣΤΗΜΑΤΩΝ ΚΑΙ ΑΥΤΟΜΑΤΟΥ ΕΛΕΓΧΟΥ  
ΕΡΓΑΣΤΗΡΙΟ: ΕΡΓΑΣΤΗΡΙΟ ΑΥΤΟΜΑΤΟΥ ΕΛΕΓΧΟΥ

### ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του φοιτητή του Τμήματος Ηλεκτρολόγων Μηχανικών και Τεχνολογίας  
Υπολογιστών της Πολυτεχνικής Σχολής του Πανεπιστημίου Πατρών

ΜΙΧΑΗΛ-ΆΓΓΕΛΟΥ ΤΡΙΑΝΤΑΦΥΛΛΗ ΤΟΥ ΆΓΓΕΛΟΥ  
(ΣΕ ΓΕΝΙΚΗ ΠΤΩΣΗ)

ΑΡΙΘΜΟΣ ΜΗΤΡΩΟΥ: 228214

### Θέμα

Υλοποίηση αυτο-ρυθμιζόμενων PID ελεγκτών με χρήση Labview

### Επιβλέπων

Επίκουρος Καθηγητής Καζάκος Δημοσθένης

Πάτρα, Ιανουάριος 2018



# ΠΙΣΤΟΠΟΙΗΣΗ

Πιστοποιείται ότι η διπλωματική εργασία με θέμα

Υλοποίηση αυτο-ρυθμιζόμενων PID ελεγκτών με χρήση Labview

του φοιτητή του Τμήματος Ηλεκτρολόγων Μηχανικών και  
Τεχνολογίας Υπολογιστών

Μιχαήλ-Άγγελου Τριανταφύλλη του Άγγελου  
(σε γενική πτώση)

(Α.Μ.: 228214)

παρουσιάστηκε δημόσια και εξετάστηκε στο τμήμα Ηλεκτρολόγων  
Μηχανικών και Τεχνολογίας Υπολογιστών στις

\_\_/\_\_/\_\_

Ο Επιβλέπων

Ο Διευθυντής του Τομέα

Καζάκος Δημοσθένης  
Επίκουρος Καθηγητής

Κούσουλας Νικόλαος  
Καθηγητής



## Στοιχεία διπλωματικής εργασίας

Θέμα: Υλοποίηση αυτο-ρυθμιζόμενων PID ελεγκτών με χρήση  
Labview

Φοιτητής: Μιχαήλ-Άγγελος Τριανταφύλλης του Άγγελου  
(σε ονομαστική πτώση)

Ομάδα επίβλεψης  
Επίκουρος Καθηγητής Καζάκος Δημοσθένης  
Βαθμίδα και Ονοματεπώνυμο Συνεπιβλέποντα  
Ονοματεπώνυμο Διδακτορικού Φοιτητή

Εργαστήρια  
Εργαστήριο Αυτομάτου Ελέγχου

Περίοδος εκπόνησης της εργασίας:  
Μήνας Έτος - Μήνας Έτος

Η εργασία αυτή γράφτηκε στο  $\text{\LaTeX}$  και χρησιμοποιήθηκε η  
γραμματοσειρά GFS Didot του Greek Font Society.



## Περίληψη

**Η** εργασία αυτή ασχολείται με την αυτόματη ρύθμιση (self-regulation) PID ελεγκτών. Παρόλο που οι PID ελεγκτές αποτελούν ένα πολύ διαδεδομένο είδος ελεγκτών με ευρεία χρήση σε βιομηχανικές, και όχι μόνο, εφαρμογές η σωστή ρύθμιση τους απαιτεί εμπειρία από το χειριστή και συνήθως αποτελεί χρονοβόρα διαδικασία. Μέσω προσομοίωσης στο περιβάλλον Labview γίνεται προσπάθεια να αυτοματοποιηθεί η διαδικασία αυτή και να φανεί ποιοι είναι οι περιορισμοί ενός αυτο-ρυθμιζόμενου PID ελεγκτή.





## Ευχαριστίες

**Ό**σο κι αν φαίνεται σαν ατομική δουλειά η παρούσα εργασία, στην πραγματικότητα βοήθησαν αρκετοί άνθρωποι (ο καθένας με το δικό του τρόπο) για να ολοκληρωθεί.



# ΠΕΡΙΕΧΟΜΕΝΑ

1	LabVIEW	3
1.1	Εισαγωγή στο LabVIEW . . . . .	3
2	PID ελεγκτές	7
2.1	Εισαγωγή στους PID ελεγκτές . . . . .	7
2.1.1	Πρώτη χρήση . . . . .	8
3	Super Resolution μέσω Image Registration	9
3.1	Μοντελοποίηση προβλήματος . . . . .	9
	Βιβλιογραφία	11



# ΕΙΣΑΓΩΓΗ

**Η** εργασία αυτή έχει γίνει προσπάθεια να γραφεί σε ανεξάρτητα κεφάλαια, τα οποία θα δώσουν στον αναγνώστη τις απαιτούμενες γνώσεις ώστε να καταλάβει σε βάθος τις τεχνικές που χρησιμοποιούνται. Σε κάθε κεφάλαιο γίνεται αναλυτική παρουσίαση των τεχνικών καθώς και του υπόβαθρου που πρέπει να έχει κάποιος ώστε τις κατανοήσει, ωστόσο θεωρείται πως ο αναγνώστης έχει ήδη κάποιες γνώσεις στο χώρο του αυτομάτου ελέγχου και στην ανάλυση συστημάτων. Έτσι, βασικές έννοιες και μηχανισμοί της ανωτέρω περιοχής θα θεωρούνται δεδομένοι και δε θα γίνει κάποια ανάλυσή τους στο κείμενο αυτό, εκτός αν κρίνεται απαραίτητο.



## ΚΕΦΑΛΑΙΟ

# 1

# LABVIEW

## 1.1 Εισαγωγή στο LabVIEW

**T**ο λογισμικό που χρησιμοποιήθηκε για την υλοποίηση αυτής της εργασίας είναι το LabVIEW από την εταιρία National Instruments (NI). Συνεπώς κρίνεται χρήσιμη μια σύντομη αναφορά σε αυτό και στον τρόπο που λειτουργεί. Καθώς το LabVIEW είναι ένα πολύ διαδεδομένο λογισμικό, με ευρεία χρήση στους κλάδους των μηχανικών, κάποιος που θέλει περισσότερες πληροφορίες μπορεί να τις βρει εύκολα στο διαδίκτυο. Κάποιες πηγές που χρησιμοποιήθηκαν για τη συγγραφή αυτής της εργασίας είναι [1], [2] και [3]. Το LabVIEW (Laboratory Virtual Instrument Engineering Workbench) είναι ένα περιβάλλον ανάπτυξης για μία οπτική γλώσσα προγραμματισμού. Σε αντίθεση με τα κοινά προγραμματιστικά περιβάλλοντα, στο LabVIEW δε χρησιμοποιείται κώδικας για να γραφτούν οι εντολές που θα εκτελεστούν αλλά γραφικά όπως κουτιά και σύμβολα. Για παράδειγμα, υπάρχουν πολλές οπτικές γλώσσες, που είναι γνωστές σαν γλώσσες ροής δεδομένων (dataflow), που βασίζονται στην ιδέα "τετράγωνα και βέλη" ("boxes and arrows"), όπου τα τετράγωνα (ή άλλου τύπου αντικείμενα) της οθόνης θεωρούνται οντότητες που συνδέονται από βέλη, γραμμές ή ακμές, που αναπαριστούν σχέσεις μεταξύ τους.

**Dataflow Programming** Η οπτική γλώσσα προγραμματισμού του LabVIEW ονομάζεται "G" και βασίζεται στη λογική του dataflow προγραμματισμού που αναφέρθηκε προηγουμένως. Αυτό σημαίνει ότι αν υπάρχουν αρκετά δεδομένα διαθέσιμα σε μία συνάρτηση ή ένα subVI (σύνολο συναρτήσεων)

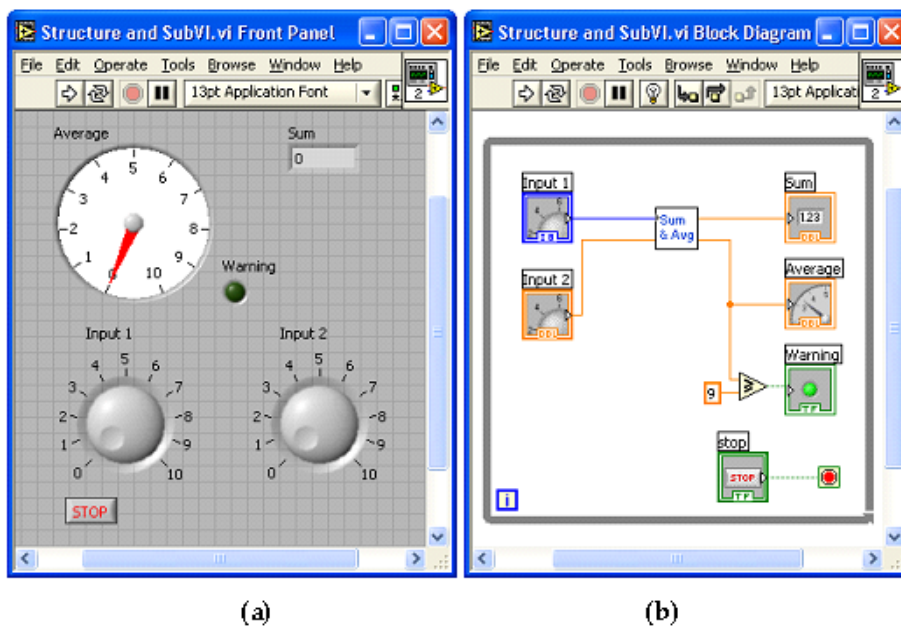
τότε αυτή η συνάρτηση ή το subVI θα εκτελεστεί. Η ροή της εκτέλεσης του προγράμματος καθορίζεται από τη δομή ενός γραφικού μπλοκ διαγράμματος (block diagram), που στην ουσία αποτελεί τον πηγαίο κώδικα του LabVIEW. Σε αυτό ο προγραμματιστής συνδέει διαφορετικές συναρτήσεις-κόμβους (function-nodes) τραβώντας καλώδια. Αυτά τα καλώδια διαδίδουν τις μεταβλητές και κάθε κόμβος μπορεί να εκτελεστεί μόλις όλα τα δεδομένα στην είσοδό του είναι διαθέσιμα. Δεδομένου ότι αυτό μπορεί να συμβαίνει για πολλαπλούς κόμβους ταυτόχρονα, το LabVIEW μπορεί να εκτελεστεί εγγενώς παράλληλα. Περισσότερα για το dataflow programming μπορείτε να βρείτε στο [4].

**Graphical Programming** Το LabVIEW ενσωματώνει τη δημιουργία διεπαφών χρήστη, που ονομάζονται εμπρόσθιοι πίνακες (front panels) στον κύκλο ανάπτυξης. Τα προγράμματα-υπορουτίνες LabVIEW ονομάζονται εικονικά όργανα (VIs). Κάθε VI διαθέτει τρία στοιχεία: ένα block diagram, ένα front panel και ένα πάνελ σύνδεσης (connection panel). Το τελευταίο χρησιμοποιείται για να αντιπροσωπεύει το VI στα block diagrams άλλων, καλώντας τα VI. Το front panel κατασκευάζεται με χειριστήρια (controls) και δείκτες (indicators). Τα controls είναι είσοδοι: επιτρέπουν σε ένα χρήστη να παρέχει πληροφορίες στο VI. Τα indicators είναι έξοδοι: υποδηλώνουν ή εμφανίζουν τα αποτελέσματα με βάση τις εισόδους που δίδονται στο VI. Το πίσω πλαίσιο, το οποίο είναι ένα block diagram, περιέχει τον γραφικό πηγαίο κώδικα. Όλα τα αντικείμενα που τοποθετούνται στο front panel εμφανίζονται στην πίσω πλευρά ως τερματικά (terminals). Ο πίσω πίνακας περιέχει επίσης δομές και λειτουργίες οι οποίες εκτελούν εργασίες στα controls και παρέχουν δεδομένα στα indicators. Οι δομές και οι λειτουργίες βρίσκονται στην παλέτα λειτουργιών και μπορούν να τοποθετηθούν στον πίσω πίνακα. Οι συλλογικοί έλεγχοι, οι δείκτες, οι δομές και οι λειτουργίες θα αναφέρονται ως κόμβοι. Οι κόμβοι συνδέονται μεταξύ τους με τη χρήση καλωδίων, π.χ. δύο controls και μια ενδεικτική λυχνία μπορούν να συνδεθούν με τη λειτουργία προσθήκης (addition function) έτσι ώστε η ένδειξη να εμφανίζει το άθροισμα των δύο controls. Έτσι, ένα VI μπορεί να λειτουργήσει είτε ως πρόγραμμα, με τον μπροστινό πίνακα να λειτουργεί ως διεπαφή χρήστη, είτε, όταν πέσει ως κόμβος στο block diagram, το front panel ορίζει τις εισόδους και εξόδους του κόμβου μέσω του παραθύρου σύνδεσης. Αυτό σημαίνει ότι κάθε VI μπορεί εύκολα να δοκιμαστεί πριν να ενσωματωθεί ως υπορουτίνα σε ένα μεγαλύτερο πρόγραμμα.

Η γραφική προσέγγιση επιτρέπει επίσης στους μη προγραμματιστές να χτίσουν προγράμματα με μεταφορά και απόθεση εικονικών αναπαραστάσεων του εργαστηριακού εξοπλισμού με τον οποίο είναι ήδη εξοικειωμένοι. Το περιβάλλον προγραμματισμού LabVIEW, με τα παραδείγματα και την τεκμηρίωση που περιλαμβάνονται, καθιστά απλή τη δημιουργία μικρών εφαρμογών. Αυτό είναι ένα πλεονέκτημα από τη μια πλευρά, αλλά υπάρχει επίσης ο κίνδυνος να υποτιμηθεί η εμπειρογνομοσύνη που



απαιτείται για τον προγραμματισμό G υψηλής ποιότητας. Για σύνθετους αλγορίθμους ή κώδικα μεγάλης κλίμακας, είναι σημαντικό ο προγραμματιστής να έχει εκτεταμένη γνώση της σύνταξης του LabVIEW και της τοπολογίας της διαχείρισης μνήμης της. Τα πιο εξελιγμένα συστήματα ανάπτυξης LabVIEW προσφέρουν τη δυνατότητα δημιουργίας αυτόνομων εφαρμογών. Επιπλέον, είναι δυνατή η δημιουργία κατανεμημένων εφαρμογών, οι οποίες επικοινωνούν με ένα μοντέλο πελάτη-εξυπηρετητή και έτσι είναι ευκολότερο να εφαρμοστούν λόγω της εγγενώς παράλληλης φύσης της προγραμματιστικής γλώσσας G. Ένα τυπικό περιβάλλον προγραμματισμού στο LabVIEW φαίνεται στο σχήμα 1.1



Σχήμα 1.1: LabVIEW: Front Panel και Block Diagram



## ΚΕΦΑΛΑΙΟ

# 2

## PID ΕΛΕΓΚΤΕΣ

### 2.1 Εισαγωγή στους PID ελεγκτές

**Έ**ΝΑΣ αναλογικός-ολοκληρωτικός-παραγωγικός ελεγκτής (proportional-integral-derivative controller) ή όπως είναι πιο γνωστός PID controller, είναι ένας μηχανισμός ανάδρασης (feedback) βρόχου ελέγχου (control loop) που χρησιμοποιείται ευρέως σε βιομηχανικά συστήματα ελέγχου καθώς και σε μια ποικιλία άλλων εφαρμογών που απαιτούν συνεχή διαμορφωμένο έλεγχο. Η διαδικασία λειτουργίας είναι κοινή για όλους τους ελεγκτές αυτού του είδους. Ένας PID ελεγκτής υπολογίζει συνεχώς μια τιμή σφάλματος  $e(t)$  ως διαφορά μεταξύ μιας επιθυμητής τιμής ρύθμισης (setpoint ή SP) και μεταξύ μιας μεταβλητής της διαδικασίας υπό έλεγχο (process value ή PV) και εφαρμόζει μια διόρθωση βασισμένη στον αναλογικό, ολοκληρωτικό και παραγωγικό όρο του (P, I, D αντίστοιχα) οι οποίοι δίνουν και στον ελεγκτή το όνομά του.

Στην πράξη, εφαρμόζει αυτόματα διορθωμένα και ακριβή διόρθωση σε μια λειτουργία ελέγχου. Ένα καθημερινό παράδειγμα είναι ο έλεγχος ταχύτητας σε οδικό όχημα. όπου εξωτερικές επιδράσεις, όπως κλίσεις, θα προκαλούσαν αλλαγές στην ταχύτητα του οχήματος. Ο αλγόριθμος PID επαναφέρει την ταχύτητα του αυτοκινήτου στην επιθυμητή από τον οδηγό τιμή της με τον βέλτιστο τρόπο, χωρίς καθυστέρηση ή υπέρβαση, ελέγχοντας την ισχύ εξόδου του κινητήρα του οχήματος.

### 2.1.1 Πρώτη εφαρμογή

Η πρώτη θεωρητική ανάλυση και πρακτική εφαρμογή αφορούσε το σύστημα αυτόματου συστήματος διεύθυνσης πλοίων, το οποίο αναπτύχθηκε από τις αρχές της δεκαετίας του 1920 και μετά. Στη συνέχεια χρησιμοποιήθηκε για τον αυτόματο έλεγχο της διαδικασίας στη μεταποιητική βιομηχανία, όπου εφαρμόστηκε ευρέως σε πνευματικούς, και στη συνέχεια ηλεκτρονικούς, ελεγκτές. Σήμερα υπάρχει γενική χρήση της ιδέας PID σε εφαρμογές που απαιτούν ακριβή και βελτιστοποιημένο αυτόματο έλεγχο.

# SUPER RESOLUTION ΜΕΣΩ IMAGE REGISTRATION

## 3.1 Μοντελοποίηση προβλήματος

**Π**ΡΙΝ δούμε στην πράξη πώς λειτουργούν οι τεχνικές super resolution θα πρέπει να περιγράψουμε το πρόβλημα με το οποίο θα δουλέψουμε με μαθηματικούς όρους. Η μοντελοποίηση αυτή θα μας επιτρέψει να χρησιμοποιήσουμε μαθηματικά εργαλεία και τεχνικές και να ορίσουμε σε μια αυστηρή “γλώσσα” (αυτή των μαθηματικών) τις λειτουργίες που επιτελούνται από κάθε μέθοδο, ούτως ώστε να πετύχουμε το τελικό αποτέλεσμα.

Ξεκινώντας, θα πρέπει να περιγράψουμε τον τρόπο με τον οποίο λαμβάνουμε εικόνες χαμηλής ανάλυσης από μια φυσική σκηνή μέσω μιας κάμερας. Η κάμερα σαν όργανο καταγραφής εισάγει κάποιες ατέλειες, όπως είδαμε στο προηγούμενο κεφάλαιο. Τέτοιες ατέλειες μπορεί να είναι σφάλματα καταγραφής από τον αισθητήρα της κάμερας, θόλωμα λόγω αστοχιών των οπτικών στοιχείων κλπ. Επιπλέον, αν λάβουμε διαδοχικές εικόνες μιας φυσικής σκηνής, οι εικόνες αυτές περιμένουμε να έχουν κάποιες μετατοπίσεις ως προς αυτό που απεικονίζουν. Οι μετατοπίσεις αυτές μπορεί να οφείλονται είτε στον ανθρώπινο παράγοντα (που χειρίζεται την κάμερα) είτε στο αντικείμενο της φυσικής σκηνής που μπορεί να μην είναι σταθερό. Στην εικόνα που λαμβάνεται τελικά, λαμβάνονται δείγματα σε χαμηλή χωρική συχνότητα και λόγω ατελειών του οργάνου μπορεί να έχουμε και παρουσία θορύβου. Για μια εικόνα υψηλής ανάλυσης  $X$  (την

οποία θα προσπαθήσουμε να ανακατασκευάσουμε), μπορούμε να αναπτύξουμε το παραπάνω μοντέλο ως εξής:

$$Y_i = S_i T_i H_i X + n_i \quad (3.1)$$

όπου ορίζουμε για την  $i$ -οστή εικόνα χαμηλής ανάλυσης  $Y_i$  τους τελεστές που ενεργούν στην  $X$  :

- $S_i$  για υποδειγματοληψία,
- $T_i$  για μετατόπιση,
- $H_i$  για θόλωμα (blurring),
- $n_i$  για προσθετικό θόρυβο.

Οι υποθέσεις που κάνουμε για το παραπάνω πρόβλημα είναι ότι το blurring είναι ίδιο σε όλο το χώρο και είναι γνωστό στον αλγόριθμο super resolution, ο θόρυβος είναι λευκός Gaussian με την ίδια διασπορά σε όλες τις εικόνες χαμηλής ανάλυσης και ότι ο γεωμετρικός μετασχηματισμός αφορά μόνο την καθολική μετατόπιση.

Στην περίπτωση που θεωρήσουμε αμελητέο θόρυβο και θόλωμα, τα παραπάνω βήματα αρκούν για να υπολογίσουμε την εικόνα υψηλής ανάλυσης  $X$ , την οποία λαμβάνουμε σαν αποτέλεσμα του αλγορίθμου.

Data:  $dx, dy, Y_i, N, W, H, S$

Output: High resolution reconstructed  $X$

$X \leftarrow 0$ ;

for  $n \leftarrow 1$  to  $N$  do

$\tilde{Y} \leftarrow Y_i$ ;  
 $i \leftarrow 1 : W/S$ ;  
 $j \leftarrow 1 : H/S$ ;  
 $px = i * S + dx_n$ ;  
 $py = j * S + dy_n$ ;  
 $X_{px,py} = \tilde{Y}_{i,j}$ ;

end

Αλγόριθμος 1: Ανακατασκευή shift-add fusion

Συγκρίνοντας τη μέθοδο αυτή με την ανακατασκευή μέσω του μετασχηματισμού Fourier, παρατηρούμε την απλότητα και την αποδοτικότητά της καθώς βρίσκει την εικόνα υψηλής ανάλυσης μόνο με μετακινήσεις pixel.

# ΒΙΒΛΙΟΓΡΑΦΙΑ

- [1] <https://en.wikipedia.org/wiki/LabVIEW>
- [2] <http://www.ni.com/en-us/shop/labview.html>
- [3] Jeffrey Travis, Jim Kring, “LabVIEW for Everyone: Graphical Programming Made Easy and Fun (3rd ed.)”, Prentice Hall Professional, July 27, 2006, ISBN-13: 978-0-13-185672-1.
- [4] Johnston, W.M., Hanna, J.R.P. and Millar, R.J. “Advances in dataflow programming languages”. ACM Computing Surveys 36 (1): 1–34. 2004. DOI: 10.1145/1013208.1013209





