Υπολογιστική Νοημοσύνη

Σχεδίαση Ασαφούς Ελεγχτή (FLC)

Θεόδωρος Κατζάλης
AEM: 9282
katzalis@ece.auth.gr

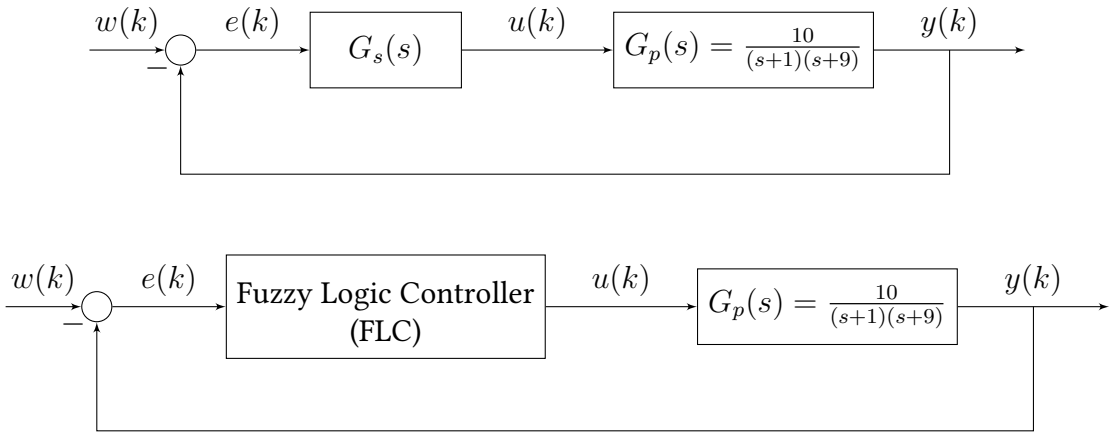
Αριστοτέλειο Πανεπιστήμιο Θεσσαλονικής

February 25, 2023

Περιεχόμενα

1 Εισαγωγή

Σκοπός αυτής της εργασίας είναι η σχεδίαση ενός ασαφούς ελεγκτή για το σύστημα που περιγράφεται από το παρακάτω σχήμα. Έχοντας ως αφετηρία την σχεδίαση ενός γραμμικού ελεγκτή (PI), θα μελετήσουμε την απόκριση και την συμπεριφορά του ασαφούς (FZ-PI) για διάφορες εισόδους. Η ανάλυση έγινε με την χρήση `Matlab` και τα αρχεία προσομοίωσης μπορούν να βρεθούν στο τέλος της αναφοράς.



Σχήμα 1: Σύστημα κλειστού βρόγχου μοναδιαίας αρνητικής ανάδρασης

2 Σχεδίαση γραμμικού ελεγκτή

Για να έχουμε μηδενικό σφάλμα μόνιμης κατάστασης για την ταχύτητα, ο γραμμικός ελεγκτής που θα χρησιμοποιήσουμε είναι PI της μορφής:

$$G_c(s) = K_p + \frac{K_I}{s} = \frac{K_p(s+c)}{s}, \quad c = \frac{K_I}{K_p} \tag{1}$$

Οι προδιαγραφές που θα πρέπει να πληρούνται ειναι:

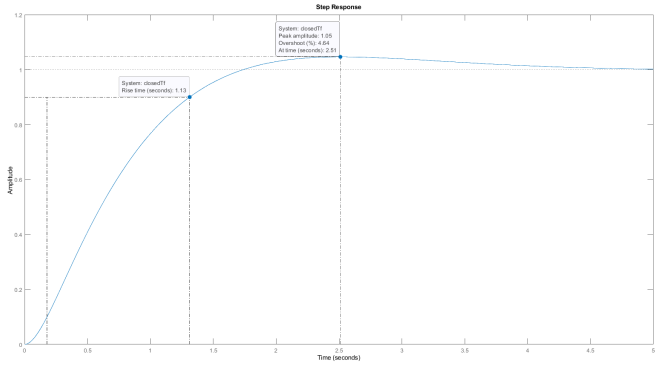
- Overshoot < 10%
- Rise time < 1.2 seconds

Χρησιμοποιώντας το `Control Systems` toolbox του `Matlab` (`controlSystemDesigner()`), προσθέσαμε τις προδιαγραφές και με διαδραστικό γραφικό τρόπο αλλάξαμε τις παραμέτρους του ελεγκτή προκειμένου να βρεθούμε εντός των επιτρεπτών περιοχών λειτουργίας. Τυπικά επιλέξαμε έναν από τους διαθέσιμους συνδυασμούς που ικανοποιούν τις παραπάνω προδιαγραφές με τιμές: $c = 1.5$ και $Kp = 1$ (οπότε $K_I = K_p * c = 1.5$).

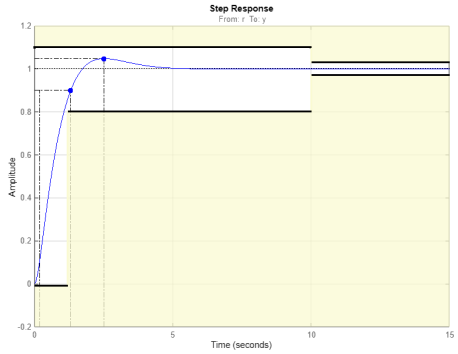
Η συνάρτηση κλειστού βρόγχου με τον συγκεκριμένο ελεγκτή είναι:

$$G(s) = \frac{10(s+1.5)}{(s+7.814)(s+2.186)(s+1.92)} \tag{2}$$

Στη συνέχεια απεικονίζονται η απόκριση του συστήματος σε βηματική είσοδο και ο γεωμετρικός τόπος ριζών του (root locus) για τις παραπάνω σχεδιαστικές παραμέτρους.

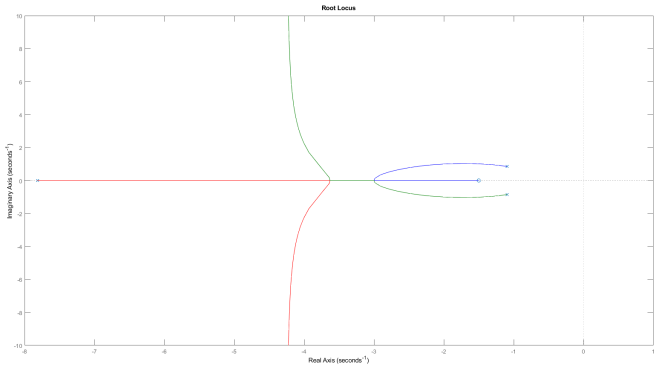


(a)

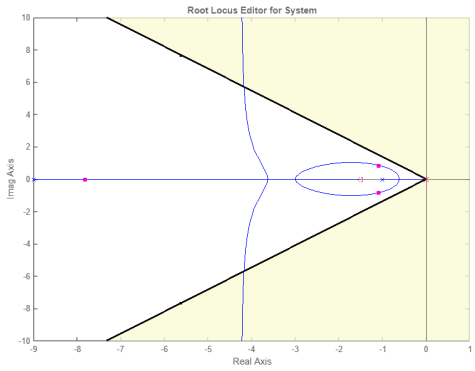


(b)

Σχήμα 2: Βηματική απόκριση ελεγχτή PI



(a)



(b)

Σχήμα 3: Γεωμετρικός τόπος ριζών ελεγχτή PI

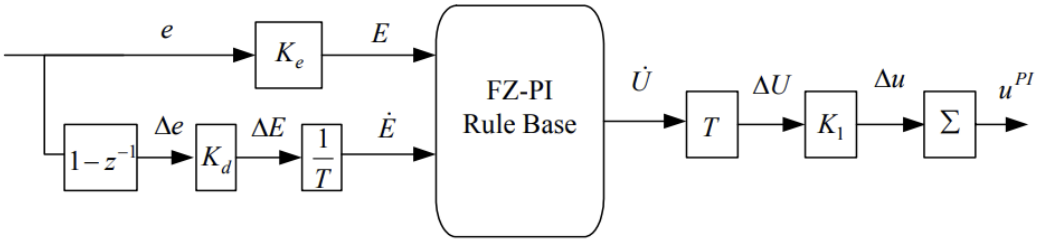
3 Σχεδίαση ασαφούς ελεγκτή (FLC)

Η σχεδίαση του ασαφούς ελεγκτή βασίζεται στα ακόλουθα βήματα:

Βήμα 1 Επιλογή εισόδων και εξόδων. Ο τύπος του ελεγκτή που θα χρησιμοποιήσουμε, ομοίως με την γραμμική περίπτωση, είναι FZ-PI για να μηδενίσουμε το σφάλμα ταχύτητας στην μόνιμη κατάσταση. Η μαθηματική περιγραφή του PI στον χρόνο είναι:

$$\dot{u}^{PI}(t) = K_p\dot{e}(t) + K_I e(t) \tag{3}$$

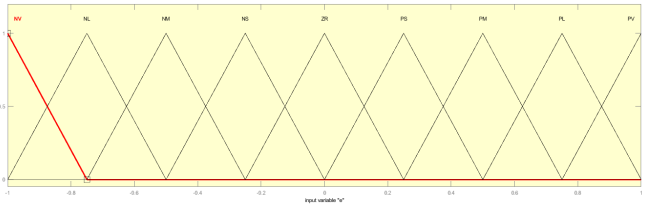
Έτσι με αντίστοιχο τρόπο, η ψηφιακή υλοποίηση του FZ-PI, προσεγγίζοντας τις παραγώγους ως $\dot{e}(t) \approx \frac{\Delta e}{T}$, όπου T ο χρόνος δειγματοληψίας, είναι:



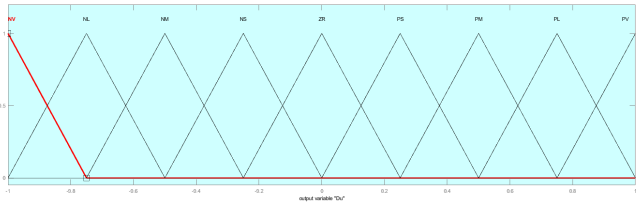
Σχήμα 4: Αρχιτεκτονική ψηφιακού FZ-PI ελεγκτή

, όπου K_e, K_d κέρδη κλιμακοποίησης εισόδων και K_1 κέρδος κλιμακοποίησης εξόδου.

Βήμα 2 Κανονικοποίηση και ασαφοποίηση εισόδου και εξόδου (**fuzzification**). Αρχικά κανονικοποιούμε τις crisp τιμές στο διάστημα [-1,1] όπως συνηθίζεται για να υπάρχει ομοιομορφία στην σχεδίαση μεταξύ ασαφών ελεγκτών. Στη συνέχεια με την βοήθεια των λεγόμενων συναρτήσεων συμμετοχής (**membership functions**) μετατρέπουμε αυτές τις τιμές σε fuzzy (αντιστοίχηση crisp set σε fuzzy set). Οι συναρτήσεις που χρησιμοποιήθηκαν ήταν τριγωνικές (`trimf`) και η διαμόρφωση τους για τις εισόδους (η διαφορά του σφάλματος έχει ακριβώς την ίδια διαμέριση με το σφάλμα) και την έξοδο φαίνονται παρακάτω.



(a) e



(b) Δu

Σχήμα 5: Membership functions

Βήμα 3 Καθορισμός βάσης κανόνων (rule base). Η βάση κανόνων αποτελεί ένα πολύ σημαντικό κομμάτι της σχεδίασης καθώς συσχετίζει τις λεκτικές τιμές της εισόδου με την έξοδο. Το πρότυπο που χρησιμοποιήσαμε για τον καθορισμό των κανόνων βασίζεται στους Mc Vicar και Wheelan και μπορεί να συνοψιστεί στον πίνακα 1. Η μορφή των κανόνων για παράδειγμα είναι:

IF e is NL **AND** Δe is PL **THEN** Δu is ZR

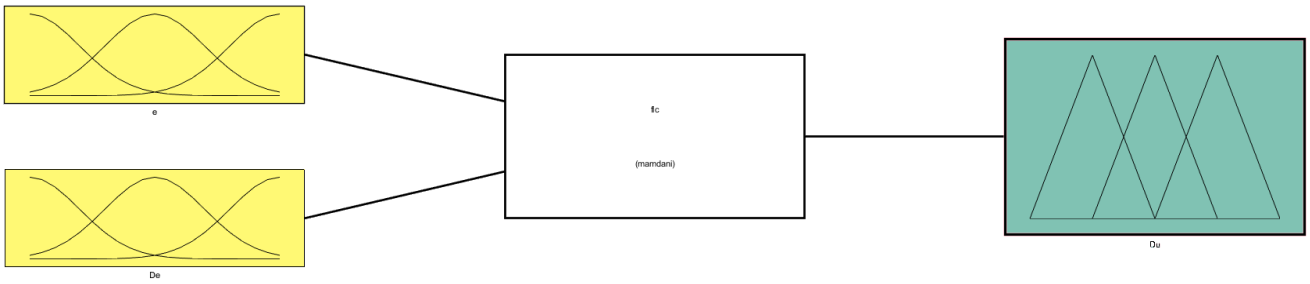
Δe \ e	NV	NL	NM	NS	ZR	PS	PM	PL	PV
PV	ZR	PS	PM	PL	PV	PV	PV	PV	PV
PL	NS	ZR	PS	PM	PL	PV	PV	PV	PV
PM	NM	NS	ZR	PS	PM	PL	PV	PV	PV
PS	NL	NM	NS	ZR	PS	PM	PL	PV	PV
ZR	NV	NL	NM	NS	ZR	PS	PM	PL	PV
NS	NV	NV	NL	NM	NS	ZR	PS	PM	PL
NM	NV	NV	NV	NL	NM	NS	ZR	PS	PM
NL	NV	NV	NV	NV	NL	NM	NS	ZR	PS
NV	NV	NV	NV	NV	NV	NL	NM	NS	ZR

Πίνακας 1: Template rule base
N: Negative, P: Postiive, V: Very > L: Large, M: Medium, S: Small

Βήμα 4 Επιλογή τελεστή συμπερασμού και αποασαφοποίηση (**defuzzification**). Σε αυτό το βήμα θα χρησιμοποιήσουμε τον τελεστή **Mamdani**. Ο τελεστής σύνθεσης θα είναι ο max-min, το λογικό `AND` υλοποιείται με min και το λογικό `ALSO` με max. Για την αποασαφοποίηση, δηλαδή την μετάβαση απο fuzzy set σε crisp value, θα χρησιμοποιηθεί ο τελεστής COA (center of average), γνωστός ως centroid.

Βήμα 5 Επιλογή κερδών κλιμακοποίησης. Ο προσδιορισμός αυτών των παραμέτρων όπως και στον γραμμικό ελεγκτή καθορίζει την ταχύτητα απόκρισης, την εμφάνιση ταλαντώσεων και την σταθερότητα του. Στην σχεδίαση, επιλέγουμε ως αρχικές συνθήκες τα κέρδη του γραμμικού PI ελεγκτή. Για αυτήν την συσχέτιση κερδών PI και FZ-PI θα μιλήσουμε στην συνέχεια.

Βήμα 6 Αποκανονικοποίηση εξόδου. Όπως αναφέραμε στο Βήμα 1, κανονικοποιούμε τις εισόδους στο διάστημα [-1,1], οπότε προκειμένου να επιστρέψουμε στην κλίμακα του σήματος αναφοράς, αποκανονικοποιούμε την έξοδο.



Σχήμα 6: Block diagram του ασαφούς ελεγκτή μεταξύ εισόδων και εξόδου

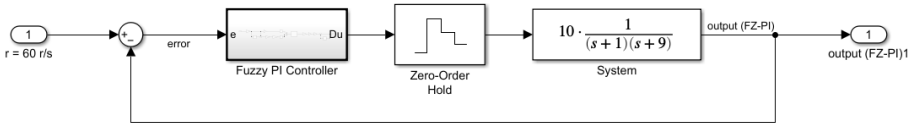
Αξίζει να σημειωθεί ότι η σχεδίαση του ελεγκτή ως προς τις συναρτήσεις συμμετοχής και την βάση κανόνων υλοποιήθηκαν προγραμματιστικά χρησιμοποιώντας το `Fuzzy toolbox`. Η γραφική διεπαφή χρησιμοποιήθηκε για λόγους επαλήθευσης.

Στη συνέχεια, προκειμένου να προσομοιώσουμε την λειτουργία του συστήματος κλειστού βρόγχου Σχήμα 1, στο οποίο πλέον ο ελεγκτής είναι ασαφής, χρησιμοποιήσαμε `Simulink` και έχουμε τα ακόλουθα αποτελέσματα.

3.1 Σενάριο 1

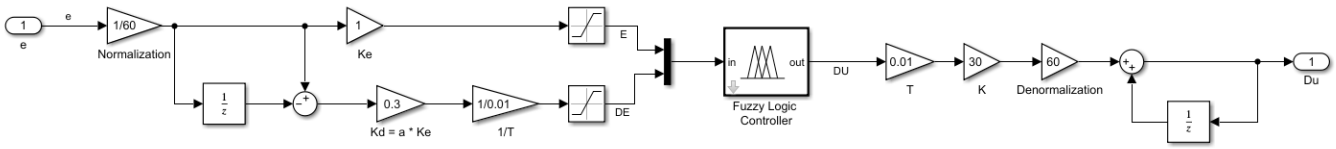
3.1.1 Σχεδίαση του συστήματος κλειστού βρόγχου και απόκριση

(a) System



(b) closedLoopFuzzyPI subsystem

Σχήμα 7



Σχήμα 8: Fuzzy PI Controller subsystem

Στόχος της παραπάνω σχεδίασης με τον ασαφή ελεγκτή ήταν να επιτύχουμε καλύτερη απόδοση σε σχέση με τον γραμμικό ρυθμίζοντας τα κέρδη εισόδου και εξόδου. Πιο συγκεκριμένα οι προδιαγραφές είναι: rise time < 0.8 seconds και overshoot < 7%. Αρχικά οι τιμές των κερδών του ασαφούς ελεγκτή, έχοντας ως δεδομένα τις τιμές του γραμμικού, τέθηκαν με τον εξής τρόπο:

$$K_d = \alpha * K_e \tag{4}$$

$$K_e = 1 \tag{5}$$

$$a \approx \frac{K_p}{K_I} = \frac{1}{1.5} \approx 0.66 \tag{6}$$

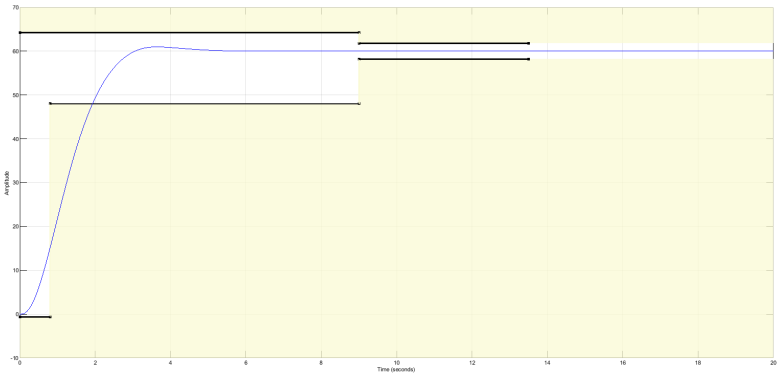
$$K = \frac{K_p}{F\{a * K_e\}} = \frac{1}{F\{0.66 * 1\}} \tag{7}$$

$$= \frac{1}{F\{0.66\} * F\{1\}} = \frac{1}{0.66 * 1} = 1.5 \tag{8}$$

, όπου $F\{\}$ fuzzy transfer function.

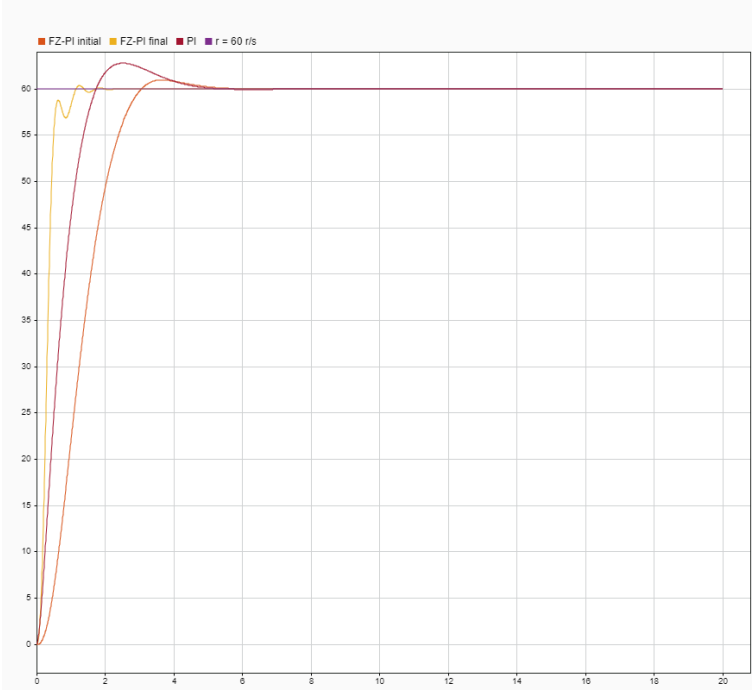
Η σχέση (5) ισχύει επειδή η βάση των κανόνων είναι γραμμική (linear rule base). Η τιμή του K_e ισούται με 1 για να αποφεύγουμε φαινόμενα κορεσμού.

Με τις παραπάνω αρχικές συνθήκες έχουμε την ακόλουθη βηματική απόκριση ύψους 60 rad/s:

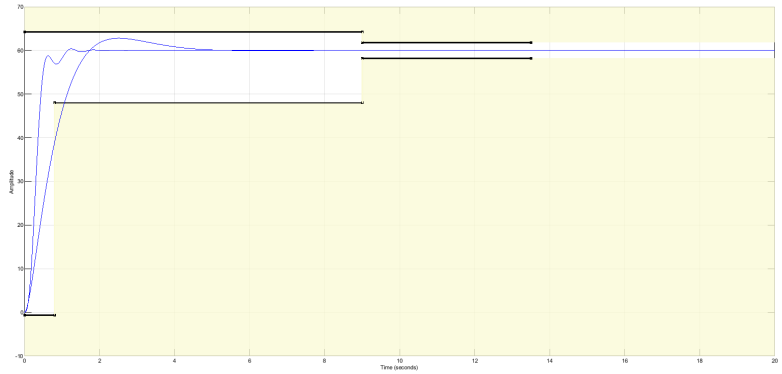


Σχήμα 9: Βηματική απόκριση FZ-PI με αρχικές συνθήκες

Βλέπουμε ότι απέχουμε από την επιθυμητή περιοχή λειτουργίας και πιο συγκεκριμένα παρατηρούμε ότι στοχεύουμε να κάνουμε το σύστημα μας πιο γρήγορο. Για να το κάνουμε αυτό θα μειώσουμε την τιμή του α και θα ανεβάσουμε και το κέρδος εξόδου K. Έτσι έπειτα απο κάποιες δοκιμές (trial and error) για $a = 0.3$ και $K = 30$ έχουμε:



(a) Σύγκριση βηματικών αποκρίσεων



(b) Βηματικές αποκρίσεις FZ-PI και PI με όρια προδιαγραφών

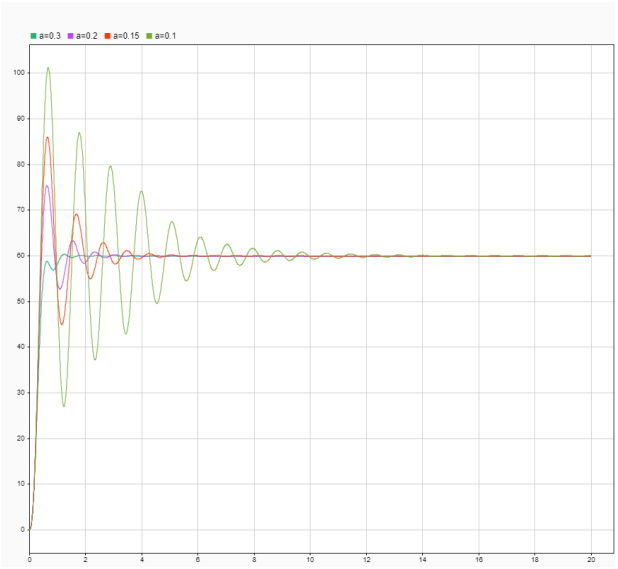
Σχήμα 10

Συνοπτικά έχουμε:

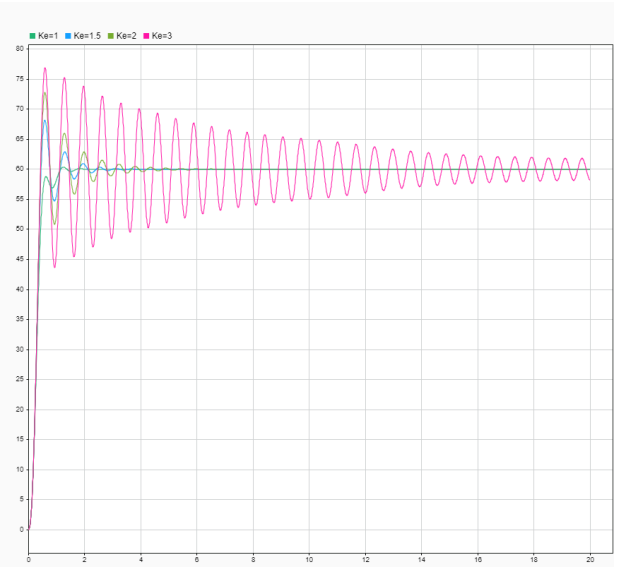
	K_p	K_I	K_e	α	K	Rise time	Overshoot	Settling time
PI	1	1.5	-	-	-	1.13s	4.63%	3.72s
FZ-PI Initial	-	-	1	0.66	1	1.8s	1.61%	2.81s
FZ-PI Final	-	-	1	0.3	30	0.33s	0.6%	1.04s

Πίνακας 2: Σύγκριση χαρακτηριστικών PI και FZ-PI

Εδώ θα δούμε την επίδραση που έχει το κάθε κέρδος κλιμακοποίησης στην απόκριση του ελεγχτή. Θα μεταβάλλουμε ένα απο τα α, K_e, K και θα κρατάμε τα άλλα σταθερά με βάση την τελική ρύθμιση του FZ-PI. Απο το Σχήμα 11 α) παρατηρούμε ότι όσο μειώνεται το μέγεθος α, η ταχύτητα απόκρισης αυξάνεται. Ωστόσο εμφανίζονται ταλαντώσεις εξαιτίας αυτής της ταχύτητας μέχρι που οδηγούμαστε πλέον σε αστάθεια. Στο σχήμα β) κρατώντας το γινόμενο K_d σταθερό, έχουμε παρόμοια συμπεριφορά με την μεταβολή του α μέχρι να φτάσουμε και πάλι σε αστάθεια. Τέλος στο σχήμα 13, το κέρδος εξόδου όσο αυξάνεται βελτιώνει την απόκριση τόσο ως προς την ταχύτητα αλλά και ως προς τις ταλαντώσεις. Αν αυξήσουμε ιδιαίτερα πολύ το K, τότε θα παρατηρήσουμε πολλές μικρές ταλαντώσεις οι οποίες αποσβένουν σταδιακά.



(a) Ως προς το α



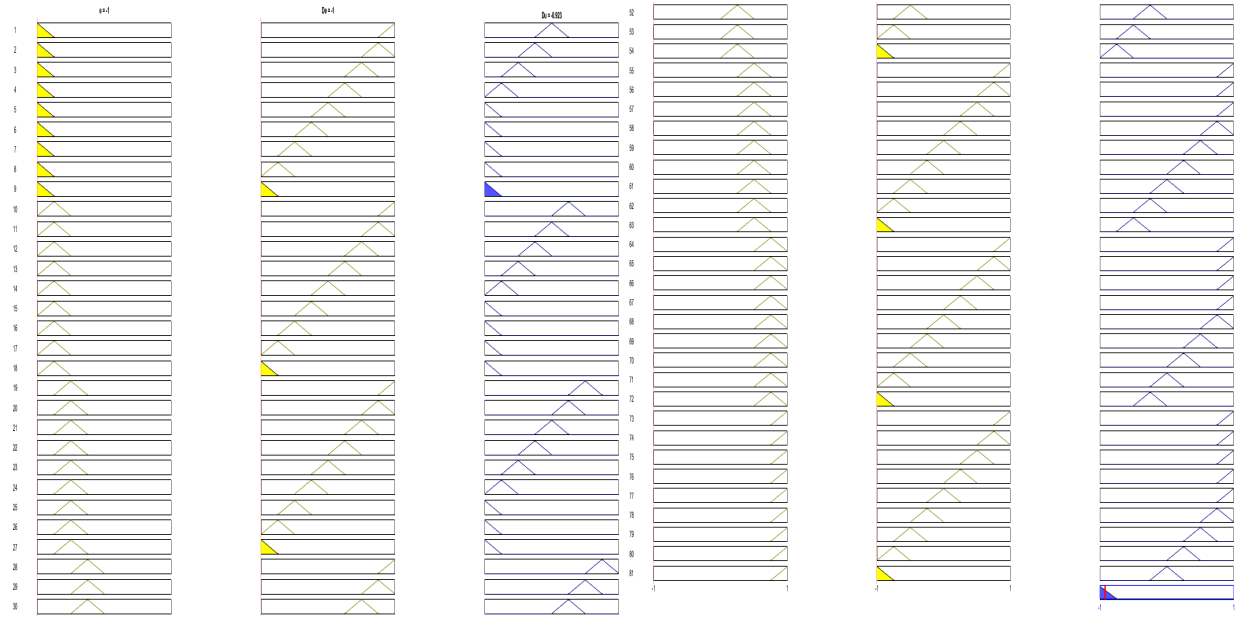
(b) Ως προς το K_e

(c) Ως προς το K

Σχήμα 11: Σύγκριση βηματικών αποκρίσεων

3.1.2 Λειτουργία της βάσης του ελεγκτή και συμπεράσματα

Για διέγερση $e = NV$ και $De = NV$ με crisp τιμές -1 και -1 αντίστοιχα, έχουμε:

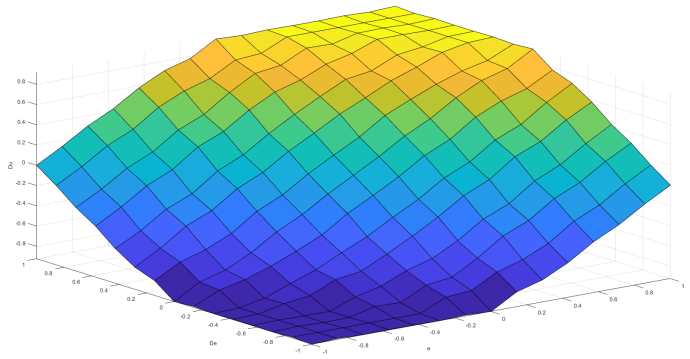


(a) κανόνες 1-30 (b) κανόνες 52-81

Σχήμα 12: Mamdani τελεστής συμπερασμού

Από το Σχήμα 12 (b), στο τελευταίο γράφημα βλέπουμε το aggregation (max) όλων των εξόδων για διέγερση του κάθε κανόνα και τελικό στάδιο την αποασαφοποίηση με τον τελεστή centroid για να πάρουμε την τελική τιμή. Επειδή $e = -1$ και $\Delta e = -1$, αυτό σημαίνει ότι έχουμε ανοδική πορεία και απομακρυνόμαστε από το σημείο αναφοράς. Έτσι η έξοδος μας θα πρέπει να είναι κατάλληλη για να αντισταθμίσει αυτήν την πορεία και πιο συγκεκριμένα πρέπει να μειώσουμε το σήμα ελέγχου. Επομένως $\Delta u < 0$ (-0.923). Το σήμα ελέγχου πρέπει να μειωθεί όσο πιο πολύ είναι εφικτό (NV), γιατί η μεταβολή του σφάλματος έχει την μέγιστη αρνητική τιμή (NV).

3.2 Ερμηνεία του νόμου ελέγχου του FLC

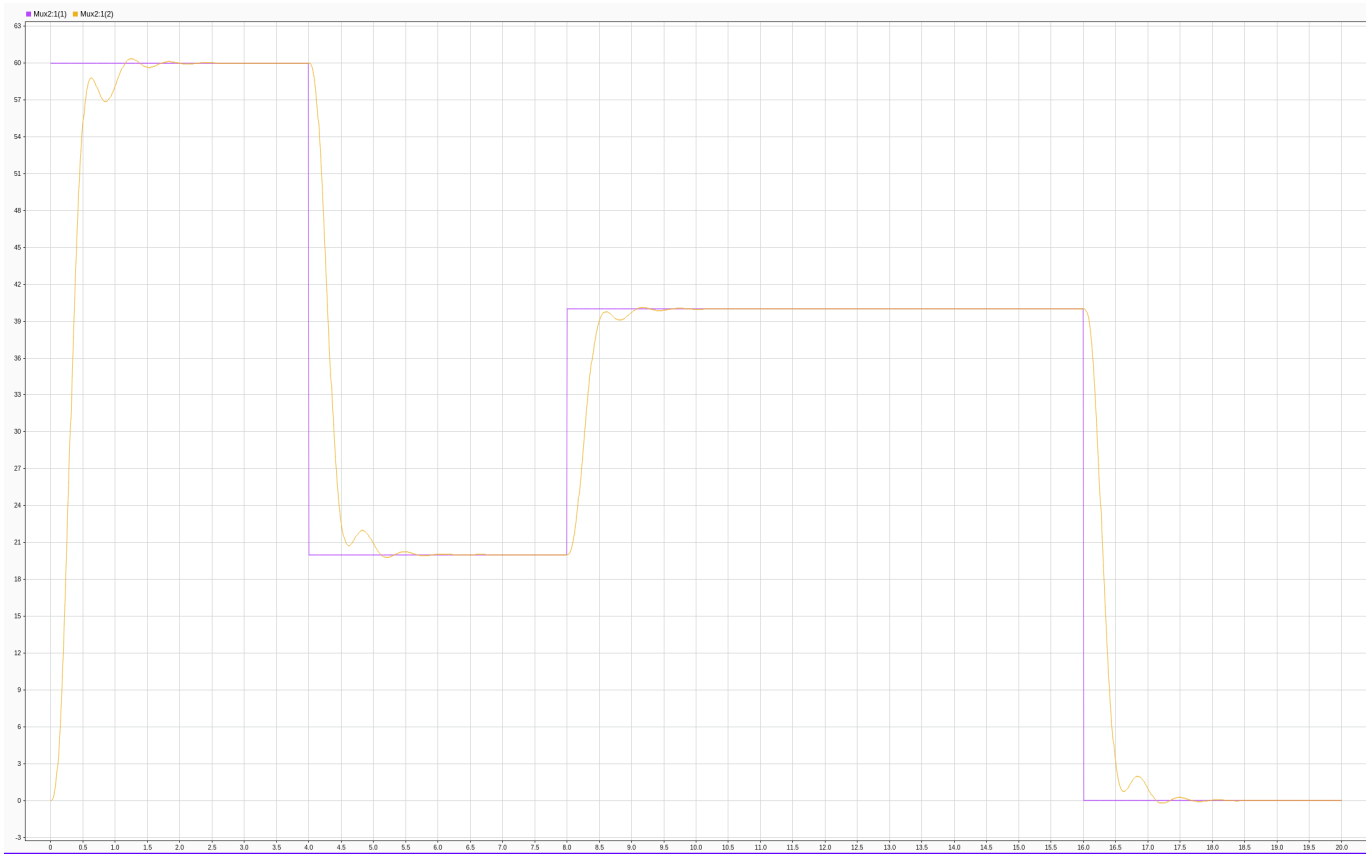


Σχήμα 13: Τρισδιάστατη επιφάνεια συσχέτισης εξόδου Δu με εισόδους e και Δe

Η παραπάνω τρισδιάστατη απεικόνιση μας υποδεικνύει την εξάρτηση της εξόδου απο τις εισόδους και το γενικότερο σκεπτικό της βάσης κανόνων. Πιο συγκεκριμένα μπορούμε να παρατηρήσουμε ότι σε χαμηλές αρνητικές τιμές σφάλματος και μεταβολής σφάλματος, δηλαδή όταν απομακρυνόμαστε απο το σημείο αναφοράς, η έξοδος είναι εξίσου αρνητική για να μειώσει το σήμα ελέγχου και να μας επαναφέρει προς το σημείο αναφοράς. Αντίστοιχα, στις μεγάλες θετικές τιμές, έχουμε καθοδική πορεία και απομακρυνόμαστε απο το σημείο αναφοράς. Έτσι η έξοδος θα πρέπει να διορθώσει και το σήμα ελέγχου να αυξηθεί, άρα $\Delta u > 0$. Ομοίως αναλογιζόμαστε για τις ενδιάμεσες τιμές.

3.3 Σενάριο 2

Σχήμα 14: Τετραγωνοειδές σήμα εισόδου και απόκριση του συστήματος



Σχήμα 15: Τραπεζοειδές σήμα εισόδου και απόκριση του συστήματος

Παρατηρούμε ότι το σύστημα μας έχει πολύ καλή απόδοση ως προς το τραπεζοειδές σήμα αναφοράς διότι δεν υπάρχουν απότομες αλλαγές οπότε η απόκριση του συστήματος προσαρμόζεται γρήγορα. Συνεπώς ο ασαφής ελεγκτής μας έχει την ικανότητα να παρακολουθεί εισόδους ράμπας.

4 Matlab

Ο κώδικας για την εργασία μπορεί να βρεθεί παρακάτω. Τα αρχεία προσομοίωσης μπορούν να σταλθούν εφόσον ζητηθούν.

```
% AEM: 9282
% Theodoros Katzalis

clear; clc

% plant transfer function
GP = zpk([], [-1 -9], 10)

% setup path
rootDir = fileparts(matlab.desktop.editor.getActiveFilename); % FIX: this ...
    requires to run within matlab desktop editor
addpath([rootDir '/simulink'])
addpath(rootDir)

%% LINEAR CONTROLLER

% controlSystemDesigner(GP)
% With Control System Toolbox, interactively we found the configuration of
```

```
% the PI "k*(s+c)/s" controler to satisfy our design requirements:
% 1) Rise time < 1.2 seconds
% 2) Overshoot < 10%

% The configuration is saved to "dcControllerTuning.mat"
% Uncomment the following line to launch the toolbox with the configuration:
% controlSystemDesigner("dcControllerTuning.mat")

c = 1.5;
k = 1;

% controller transfer function
%GC = tf([k * 1, k * c], [1, 0])
GC = zpk(-c, 0, k)

% closed loop transfer function
closedTf = feedback(GP * GC, 1)

% plots
figure; step(closedTf)
figure; rlocus(closedTf)

%% FUZZY CONTROLLER

% Design fuzzy controller
fis = mamfis('AndMethod', "min", 'ImplicationMethod', "min", ...
    'AggregationMethod', "max", 'DefuzzificationMethod', "centroid");

numMfs = 9;
fis = addInput(fis, [-1 1], 'Name', "e", 'NumMFs', numMfs, 'MFType', "trimf");
fis = addInput(fis, [-1 1], 'Name', "De", 'NumMFs', numMfs, 'MFType', "trimf");
mfNames = ["NV", "NL", "NM", "NS", "ZR", "PS", "PM", "PL", "PV"];

% I/O fuzzy variables and membership functions
middleNode = -1;
leftNode = middleNode - 1/4;
rightNode = middleNode + 1/4;

for index = 1:numel(mfNames)
    fis.Input(1).MembershipFunctions(index).Name = mfNames(index);
    fis.Input(1).MembershipFunctions(index).Parameters = [leftNode middleNode ...
        rightNode];
    fis.Input(2).MembershipFunctions(index).Name = mfNames(index);
    fis.Input(2).MembershipFunctions(index).Parameters = [leftNode middleNode ...
        rightNode];

    middleNode = middleNode + 1/4;
    leftNode = middleNode - 1/4;
    rightNode = middleNode + 1/4;
end

fis = addOutput(fis, [-1 1], 'Name', "Du", 'NumMFs', numMfs, 'MFType', "trimf");
zeroIdxOutputMf = 5;

middleNode = -1;
leftNode = middleNode - 1/4;
rightNode = middleNode + 1/4;

for index = 1:numel(mfNames)
    fis.Output(1).MembershipFunctions(index).Name = mfNames(index);
    fis.Output(1).MembershipFunctions(index).Parameters = [leftNode middleNode ...
        rightNode];

```

```
77      middleNode = middleNode + 1/4;
78      leftNode = middleNode - 1/4;
79      rightNode = middleNode + 1/4;
80  end
81
82  numPoints = 5000;
83  figure
84  subplot(311); plotmf(fis, 'input', 1, numPoints); title("Input e")
85  subplot(312); plotmf(fis, 'input', 2, numPoints); title("Input De")
86  subplot(313); plotmf(fis, 'output', 1, numPoints); title("Output Du")
87  sgtitle("Membership functions I/O")
88
89  % Rule base
90  rowDim = 9;
91  colDim = 9;
92  rules = strings(rowDim, colDim);
93
94  % iterate over array diagonally and fill the matrix based on Mc Vicar and ...
        Wheelan template
95  numDiagonals = 9;
96
97  for idxDiag = 1:numDiagonals
98      rowIdx = idxDiag;
99      for j = 1:numDiagonals - idxDiag + 1
100         if idxDiag == 1
101             rules(rowIdx, j) = sprintf("De=%s & e=%s => Du=%s", mfNames(numMfs ...
                    - rowIdx + 1), mfNames(j), "ZR");
102             %rules(rowIdx,j) = sprintf("%s,%s,%s", mfNames(numMfs - rowIdx + 1), ...
                    mfNames(j),"ZR");
103         elseif idxDiag <= zeroIdxOutputMf - 1
104             rules(rowIdx, j) = sprintf("De=%s & e=%s => Du=%s", mfNames(numMfs ...
                    - rowIdx + 1), mfNames(j), mfNames(zeroIdxOutputMf - idxDiag + 1));
105             %rules(rowIdx, j) = sprintf("%s,%s,%s", mfNames(numMfs - rowIdx + ...
                    1), mfNames(j), mfNames(zeroIdxOutputMf - idxDiag + 1));
106
107             rules(j, rowIdx) = sprintf("De=%s & e=%s => Du=%s", mfNames(numMfs ...
                    - j + 1), mfNames(rowIdx), mfNames(zeroIdxOutputMf + idxDiag - 1));
108             %rules(j, rowIdx) = sprintf("%s,%s,%s", mfNames(numMfs - j + 1), ...
                    mfNames(rowIdx), mfNames(zeroIdxOutputMf + idxDiag - 1));
109         else
110             rules(rowIdx, j) = sprintf("De=%s & e=%s => Du=%s", mfNames(numMfs ...
                    - rowIdx + 1), mfNames(j), "NV");
111             %rules(rowIdx, j) = sprintf("%s,%s,%s", mfNames(numMfs - rowIdx + ...
                    1), mfNames(j), "NV");
112
113             rules(j, rowIdx) = sprintf("De=%s & e=%s => Du=%s", mfNames(numMfs ...
                    - j + 1), mfNames(rowIdx), "PV");
114             %rules(j, rowIdx) = sprintf("%s,%s,%s", mfNames(numMfs - j + 1), ...
                    mfNames(rowIdx), "PV");
115         end
116         rowIdx = rowIdx + 1;
117     end
118 end
119
120 fis = addRule(fis, rules(:));
121 writeFIS(fis, "flc")
122 gensurf(fis)
123
124 % Uncomment these lines for GUI representation of fis object, fuzzy inference ...
        system
125 %plotfis(fis)
126 %ruleview(fis)
127 %fuzzy(fis)
```

```
%showrule(fis)

%% Simulink closed loop unity negative feedback system
open_system("simulink_simulation.slx")
```