Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Διατμηματικό Πρόγραμμα Προηγμένων Συστημάτων Παραγωγής, Αυτοματισμού και Ρομποτικής

Βελτιστοποίηση ελεγκτών ρομποτικών οχημάτων με τη μέθοδο Βελτιστοποίησης Σμήνους Σωματιδίων (Particle Swarm Optimization)

Κοντοζούδης Θεόδωρος

Επιβλέπων Καθηγητής:

Αναπληρωτής Καθηγητής Ε. Δοϊτσίδης

Τμήμα Ηλεκτρονικών Μηχανικών Τ.Ε

Ηράκλειο, Απρίλιος 2017

Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών

Διατμηματικό Πρόγραμμα Μεταπτυχιακών Σπουδών (ΔΠΜΣ)

«Προηγμένα Συστήματα Παραγωγής, Αυτοματισμού και Ρομποτικής»

ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Βελτιστοποίηση ελεγκτών ρομποτικών οχημάτων με τη μέθοδο Βελτιστοποίησης Σμήνους Σωματιδίων (Particle Swarm Optimization)

Μεταπτυχιακός Φοιτητής: Κοντοζούδης Θεόδωρος

ΑΜ:MTH58

Επιβλέπων Καθηγητής:

Αναπληρωτής Καθηγητής Ε. Δοϊτσίδης

Παρουσιάστηκε δημόσια και εξετάστηκε στο Τμήμα…….στις

……/……/……

Εξεταστική Επιτροπή

1. Δοϊτσίδης Ελευθέριος, Αναπληρωτής Καθηγητής ΤΕΙ Κρήτης……..
2. Φασουλάς Ιωάννης, Επίκουρος Καθηγητής ΤΕΙ Κρήτης……
3. Παπαδουράκης Γεώργιος, Καθηγητής ΤΕΙ Κρήτης……

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω τους καθηγητές του μεταπτυχιακού προγράμματος για τις γνώσεις που αποκόμισα κατά τη διάρκεια της φοίτησης και κυρίως τον επιβλέποντα καθηγητή κ. Δοϊτσίδη Ελευθέριο για την άριστη συνεργασία και καθοδήγησή του σε όλο το χρονικό διάστημα εκπόνησης της παρούσας διπλωματικής εργασίας. Τέλος, θα ήθελα να ευχαριστήσω θερμά την οικογένειά μου και τα αγαπημένα μου πρόσωπα για τη στήριξη που παρείχαν και την υπομονή που έδειξαν.

ΠΕΡΙΛΗΨΗ

Σκοπός της παρούσας μεταπτυχιακής εργασίας είναι η μελέτη της επίδρασης αλγορίθμων βελτιστοποίησης σμήνους σωματιδίων, σε έντροχα ρομποτικά οχήματα που έχουν τη δυνατότητα να κινούνται αυτόνομα σε άγνωστο περιβάλλον με χρήση ελεγκτών ανεπτυγμένων με εργαλεία υπολογιστικής νοημοσύνης και συγκεκριμένα *ασαφών ελεγκτών*.

Αναλύεται η επίδραση της διαδικασίας βελτιστοποίησης στη συμπεριφορά των οχημάτων (ταχύτητα, χρόνος επίτευξης στόχου, αποφυγή εμποδίων κλπ.) καθώς και γίνεται διερεύνηση της επίδρασης διαφορετικών συναρτήσεων αξιολόγησης στην διαδικασία βελτιστοποίησης. Η συγκεκριμένη μελέτη υλοποιείται με συνδυασμένη χρήση του ρεαλιστικού προσομοιωτή V-REP για την αναπαράσταση ενός προσομοιωμένου ρομποτικού οχήματος Pioneer, με την ταυτόχρονη χρήση του υπολογιστικού πακέτου MATLAB για την πραγματοποίηση των σχετικών υπολογισμών.

Με τη συγκεκριμένη προσέγγιση οι βελτιστοποιημένοι ελεγκτές επέδειξαν εύρωστη συμπεριφορά και τα οχήματα ήταν σε θέση να πραγματοποιούν τον επιθυμητό σκοπό με μεγαλύτερη ταχύτητα από τους ευρετικά σχεδιασμένους ελεγκτές. Κατά τη διεξαγωγή των πειραμάτων αναδείχθηκε η σημαντικότητα του ρόλου της συνάρτησης αξιολόγησης στην εξέλιξη της βελτιστοποίησης και στην επίδραση της συμπεριφοράς του οχήματος.

ABSTRACT

In the context of this thesis, we have studied how we can use Particle Swarm Optimization (PSO) algorithms, to optimize the behavior of fuzzy logic controlled autonomous mobile robots, operating in an unknown environment. We have formally analyzed and studied the impact that this process has on the actual form of the derived controllers and to the overall behavior of the robots in terms of speed, ability to achieve certain targets and to avoid obstacles. The overall implementation was done using a client-server schema, were the roborealistic simulator V-REP was the server and was used to simulate the behavior of the mobile robot, while MATLAB was the client application in which all the necessary calculations were taking place. The optimized controllers exhibit robust behavior and they outperformed the heuristically designed controllers in all cases. With the aforementioned procedure the effect of different cost functions in the optimization process and therefore in the final controllers was highlighted.

Πίνακας περιεχομένων

Πίνακας σχημάτων

Πίνακας εικόνων

Πίνακας πινάκων

ΚΕΦΑΛΑΙΟ 1

Εισαγωγή

Η πλοήγηση και ο έλεγχος ρομποτικών οχημάτων είναι ένα θέμα που τα τελευταία χρόνια προσελκύει όλο και μεγαλύτερο ενδιαφέρον. Καθοριστικό ρόλο για αυτό έπαιξε η τεχνολογική εξέλιξη που επέτρεψε τη χρήση ρομποτικών συστημάτων για σκοπούς που μέχρι πριν λίγο καιρό, λόγω των περιορισμένων τους δυνατοτήτων, δεν ήταν δυνατόν να χρησιμοποιηθούν. Σήμερα όλο και συχνότερα εμφανίζονται ρομποτικά συστήματα με δυνατότητες που προσεγγίζουν αυτές που έχουν οι άνθρωποι-χειριστές, όπως για παράδειγμα τα αυτόνομα αυτοκίνητα που παρουσιάζουνται από διάφορες εταιρίες.

Ο κύριος στόχος όλων αυτών των προσπαθειών είναι να αναπτυχθούν συστήματα με υψηλή προσαρμοστικότητα σε μεταβαλλόμενες συνθήκες και όσο το δυνατόν μικρότερη εξάρτηση από τον αρχικό σχεδιασμό. Μια προσέγγιση είναι με την χρήση μεθοδολογιών που βασίζονται σε εργαλεία και τεχνικές υπολογιστικής νοημοσύνης (computational intelligence) και επιτρέπουν την ελεύθερη εξέλιξη της συμπεριφοράς των ρομποτικών οχημάτων. Τέτοιες μεθοδολογίες μπορούν να αναζητηθούν και στην περιοχή των *Αλγορίθμων Νοημοσύνης Σμήνους* (Swarm Intelligence), που είναι εμπνευσμένοι από τον τρόπο δράσης ομάδων έμβιων όντων όπως είναι τα σμήνη των πουλιών ή τα κοπάδια των ψαριών. Μια περίπτωση αλγορίθμου βελτιστοποίησης σμήνους είναι η *Βελτιστοποίηση Σμήνους Σωματιδίων* (Particle Swarm Optimization (PSO)). Πρόκειται για μια στοχαστική μέθοδο, που χρησιμοποιεί *πληθυσμούς για την αναζήτηση λύσεων εντός του χώρου αναζήτησης*. Ωστόσο, μια μεγάλη διαφορά με τους εξελικτικούς αλγόριθμους είναι η κίνηση κάθε μέλους του πληθυσμού με μια προσαρμόσιμη ταχύτητα (adaptable velocity) στον χώρο αναζήτησης. Επιπλέον, κάθε μέλος του πληθυσμού έχει μνήμη στην οποία διατηρεί την καλύτερη θέση που επισκέφτηκε ποτέ.

Σκοπός της εργασίας είναι να μελετηθεί η επίδραση Αλγορίθμων Βελτιστοποίησης Σμήνους Σωματιδίων, σε έντροχα ρομποτικά οχήματα που έχουν τη δυνατότητα να πλοηγούνται σε άγνωστο περιβάλλον με χρήση ελεγκτών ανεπτυγμένων με εργαλεία υπολογιστικής νοημοσύνης και συγκεκριμένα *ασαφών ελεγκτών*. Αναλύεται η επίδραση της διαδικασίας βελτιστοποίησης στη συμπεριφορά των οχημάτων (ταχύτητα, χρόνος επίτευξης στόχου, αποφυγή εμποδίων κλπ.) καθώς και γίνεται διερεύνηση της επίδρασης διαφορετικών συναρτήσεων αξιολόγησης στη διαδικασία βελτιστοποίησης.

Αναλυτικά, στο δεύτερο κεφάλαιο γίνεται εκτενής βιβλιογραφική αναφορά και παρουσιάζονται βασικές αρχές της ασαφούς λογικής καθώς και του τρόπου σχεδιασμού

ασαφών ελεγκτών για την πλοήγηση εντρόχων ρομποτικών οχημάτων. Στη συνέχεια γίνεται διεξοδική ανάλυση της διαδικασίας βελτιστοποίησης με χρήση σμήνους σωματιδίων και της επίδρασης των διαφόρων παραμέτρων σε αυτή.

Στο τρίτο κεφάλαιο γίνεται μια παρουσίαση του τρόπου κίνησης διαφορικών ρομποτικών οχημάτων και του τρόπου ανάπτυξης ενός ασαφούς ελεγκτή γι' αυτά. Παρουσιάζονται αναλυτικά πειράματα που πραγματοποιήθηκαν σε εξειδικευμένο περιβάλλον προσομοίωσης.

Στο τέταρτο κεφάλαιο παρουσιάζονται και αναλύονται διεξοδικά οι βιβλιοθήκες που αναπτύχθηκαν για τη βελτιστοποίηση ασαφών ελεγκτών με χρήση αλγορίθμων βελτιστοποίησης σμήνους σωματιδίων. Με τη χρήση των παραπάνω γίνεται διερεύνηση της επίδρασης διαφορετικών συναρτήσεων αξιολόγησης στη διαδικασία της βελτιστοποίησης και παρουσιάζεται μια συγκριτική ανάλυση των αποτελεσμάτων.

Στο πέμπτο κεφάλαιο γίνεται μια σύνοψη της εργασίας και παρουσιάζονται κάποιες σκέψεις για μελλοντική έρευνα στο συγκεκριμένο πεδίο.

Στο παράρτημα παρουσιάζονται αναλυτικά οι κώδικες που αναπτύχθηκαν στα πλαίσια της συγκεκριμένης εργασίας με χρήση του λογισμικού MATLAB.

ΚΕΦΑΛΑΙΟ 2

Βιβλιογραφική επισκόπηση

2.1 Εισαγωγή

Στο κεφάλαιο παρουσιάζονται οι βασικές αρχές της ασαφούς λογικής και πως χρησιμοποιείται στον έλεγχο συστημάτων και ειδικότερα σε έντροχα ρομποτικά οχήματα. Ακολούθως γίνεται αναφορά στα είδη βελτιστοποίησης, με έμφαση στον Αλγόριθμο Βελτιστοποίησης Σμήνους Σωματιδίων (Particle Swarm Optimization PSO) και τις παραλλαγές του. Τέλος γίνεται μια βιβλιογραφική αναφορά στις εφαρμογές του αλγορίθμου PSO στη ρομποτική και ειδικότερα στα έντροχα ρομποτικά οχήματα.

2.2 Ασαφής λογική

Ως υπολογιστική νοημοσύνη ορίζεται ένα σύνολο τεχνικών που έχουν τα εξής κοινά χαρακτηριστικά: (i) χρησιμοποιούν αριθμητική αναπαράσταση της γνώσης, (ii) επιδεικνύουν προσαρμοστικότητα και ανοχή σε σφάλματα και (iii) διαθέτουν ταχύτητα επεξεργασίας συγκρίσιμη με ανθρώπινες διαδικασίες νόησης. Κύριο γνώρισμα των τεχνικών αυτών είναι ότι με την χρήση τους λύνονται γνωσιακά προβλήματα δίχως να είναι απαραίτητη η διαθεσιμότητα μαθηματικών μοντέλων αλλά ξέροντας μόνο τα δεδομένα εισόδου και εξόδου [1]. Στόχος της υπολογιστικής νοημοσύνης είναι να μπορέσει ο άνθρωπος να μεταφέρει τον τρόπο σκέψης του σε μηχανές ή υπολογιστικά συστήματα και να τα καταστήσει ικανά να εκτελούν διάφορες λειτουργίες. Στο ευρύτερο γνωστικό αντικείμενο της υπολογιστικής νοημοσύνης ανήκει ένα σύνολο τεχνικών που περιλαμβάνει μεταξύ άλλων τα *νευρωνικά δίκτυα*, τον *εξελικτικό προγραμματισμό* και την *ασαφή λογική*.

Στη παράγραφο αυτή θα ασχοληθούμε με την ασαφή λογική που αποτελεί ένα κλάδο με ευρύ πεδίο εφαρμογών σε διάφορους τομείς όπως είναι ο έλεγχος συστημάτων, η ρομποτική, η λήψη αποφάσεων, τα συστήματα παραγωγής. Η εισαγωγή της ασαφούς λογικής και η σύλληψη της ιδέας του ασαφούς συνόλου έγινε από τον Lofti Zadeh το 1960, όπου διατυπώθηκε η άποψη ότι συνήθως στον κόσμο που ζούμε τα αντικείμενα γύρω μας ανήκουν σε διάφορα σύνολα με διαφορετικούς βαθμούς συμμετοχής. Βάση της προηγούμενης προσέγγισης διατυπώθηκε η αρχή του ασυμβίβαστου που υποστηρίζει ότι καθώς η πολυπλοκότητα ενός συστήματος αυξάνει, η ικανότητα μας να προβαίνουμε σε ακριβείς και σημαντικές δηλώσεις για τη συμπεριφορά του μειώνεται έως ότου να

φθάσουμε σε ένα όριο (κατώφλι-threshold) πέρα από το οποίο ακρίβεια και σημαντικότητα (ή σχετικότητα) καθίστανται σχεδόν αμοιβαία αποκλειόμενα χαρακτηριστικά. Η ασαφής λογική είναι ένα εργαλείο που αναπτύχθηκε προκειμένου να αντιμετωπιστεί η αρχή του ασυμβίβαστου του Zadeh. Τα εργαλεία που χρησιμοποιεί δανείζονται στοιχεία από την κλασσική θεωρία συνόλων αλλά χρησιμοποιώντας μια επέκταση τους την έννοια των ασαφών συνόλων.

Η θεωρία των συνόλων αρχικά αναπτύχθηκε από τον Cantor (1845-1918) και σύμφωνα με αυτή σύνολο είναι οποιαδήποτε συλλογή ή ομάδα ομοειδών πραγμάτων που έχουν ή ικανοποιούν μια συγκεκριμένη ιδιότητα. Τα μέλη της ομάδας αυτής καλούνται *στοιχεία του συνόλου*. Το πλήθος των στοιχείων ενός συνόλου καλείται *πληθικός αριθμός* του συνόλου και συμβολίζεται συνήθως με *N*. Υπάρχουν πεπερασμένα και άπειρα σύνολα, ανάλογα με το αν ο πληθικός τους αριθμός είναι πεπερασμένος ή άπειρος.

Η ασαφής λογική βασίζεται στην επέκταση της έννοιας του κλασικού συνόλου που ορίζεται στο δίτιμο σύνολο {0,1}, στη γενικευμένη έννοια του ασαφούς συνόλου που ορίζεται στο κλειστό απειροδιάστημα [0,1]. Ένα δίτιμο σύνολο Α ως προς το σύνολο αναφοράς Χ, μπορεί να παρασταθεί ισοδύναμα μέσω της χαρακτηριστικής συνάρτησής του I_A, δηλαδή:

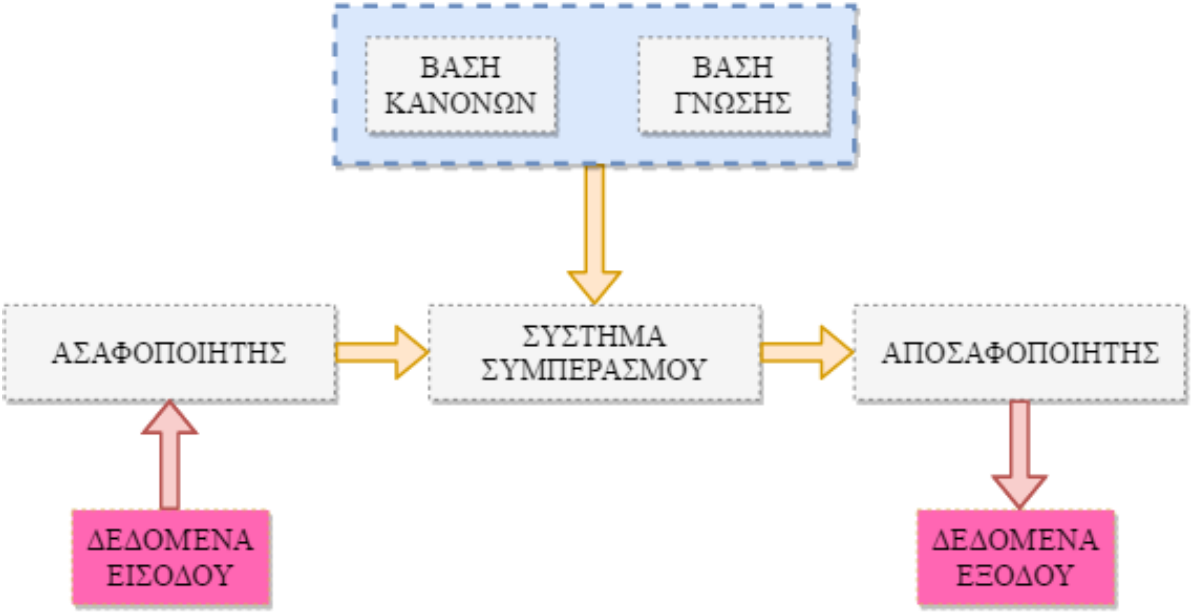
$$I_A: x \in X \to I_A(x) = \begin{cases} 1, \text{όταν } x \in A \\ 0, \text{όταν } x \notin A \end{cases} \quad (2.1) \qquad I_A: x \in X \to I_A(x) \in \{0,1\} \quad (2.2)$$

Για την αντιμετώπιση λεκτικών ασαφών εκφράσεων, το ασαφές σύνολο επεκτείνει την έννοια ενός δίτιμου συνόλου μέσω της συνάρτησης συμμετοχής, δηλαδή:

$$\mu_A: x \in X \to \mu_A(x) \in [0,1] \quad (2.3)$$

Ο αριθμός $\mu_A(x) \in [0,1]$, δηλώνει τον βαθμό συμμετοχής με τον οποίο το στοιχείο $x \in X$ ανήκει (συμμετέχει) στο ασαφές υποσύνολο Α του Χ. Δηλαδή $\mu_A(x) = 1$ σημαίνει ότι το x ανήκει ολοκληρωτικά στο Α, $\mu_A(x) = 0$ σημαίνει ότι το x δεν ανήκει καθόλου στο Α, και $0 < \mu_A(x) < 1$ σημαίνει ότι το x ανήκει μερικά δηλαδή κατά κάποιο βαθμό στο Α. Στην ασαφή λογική η τιμή αλήθειας μιας πρότασης δεν παίρνει μόνον μία από τις δύο τιμές 0 ή 1, αλλά μπορεί να παίρνει και οποιαδήποτε τιμή μεταξύ 0 και 1, οπότε μια πρόταση που δεν είναι αληθής δεν σημαίνει αναγκαία ότι είναι ψευδής, αλλά μπορεί να είναι μερικά αληθής και μερικά ψευδής, όπως ένα μισογεμάτο ποτήρι [2].

Ο Zadeh πρότεινε ένα διευρυμένο τρόπο αναπαράστασης όπου μια τιμή ανήκει ταυτόχρονα σε πολλά υποσύνολα, στο κάθε ένα με ένα *βαθμό συμμετοχής*. Τα *ασαφή σύνολα* (fuzzy sets) χρησιμοποιούνται για να αναπαραστήσουν τις μεταβλητές εισόδου και εξόδου με λεκτικούς όρους. Η περιγραφή μιας αυστηρά αριθμητικής τιμής με λεκτικούς όρους ονομάζεται *ασαφοποίηση* (fuzzyfication). Ο βαθμός συμμέτοχης μιας τιμής σ' ένα ασαφές σύνολο αποτελεί το *βαθμό βεβαιότητας* (degree of certainty) ότι η πρόταση είναι αληθής. Το πλάτος ενός ασαφούς συνόλου είναι το εύρος του πεδίου ορισμού του και αποτελεί ένα *μέτρο ασάφειας* (fuzziness) του όρου που περιγράφει. Με τη χρήση των παραπάνω εργαλείων είναι δυνατόν να συνθέσουμε ασαφής κανόνες, που αλληλοεπιδρούν και αντιστοιχούν σε ένα μηχανισμό αναπαράστασης της γνώσης με τρόπο λειτουργίας ανάλογο με τον ανθρώπινο τρόπο σκέψης. Ένα σύνολο κανόνων συνθέτει μια βάση γνώσης όπου βρίσκεται η πληροφορία για το πρόβλημα σε μορφή κανόνων "ΕΑΝ – ΤΟΤΕ". Οι κανόνες αποτελούνται από δυο μέρη, το τμήμα υπόθεσης και το τμήμα απόφασης. Για παράδειγμα εάν x είναι Α τότε y είναι Β, με Α και Β ασαφή σύνολα και x μια τιμή μιας μεταβλητής εισόδου η οποία ασαφοποιείται αποκτάει δηλαδή ένα βαθμό συμμετοχής στο ασαφές σύνολο Α. Η έξοδος y του συστήματος εκφράζει την απόφαση του κανόνα και παρέχεται από τον μηχανισμό του συμπεράσματος (inference) σε ασαφή μορφή. Το ασαφές συμπέρασμα απο-ασαφοποιείται και προκύπτει μια ευκρινή (crisp) τιμή που μπορεί να χειριστεί η υπολογιστική μηχανή [3].



Σχήμα 2.1 Βασική δομή ασαφούς ελεγκτή

2.3 Εφαρμογές της ασαφούς λογικής στον έλεγχο

Η ασαφής λογική μπορεί να χρησιμοποιηθεί ως εναλλακτική του κλασσικού ελέγχου και να αποτελέσει τη βάση για το σχεδιασμό στρατηγικών με τη βοήθεια ευφυών τεχνικών. Ο στόχος ενός ευφυούς ελεγκτή είναι να λειτουργεί με ανάλογο τρόπο με αυτόν που λειτουργεί ένας άνθρωπος που θα ήλεγχε την αντίστοιχη διεργασία. Ένας καλά σχεδιασμένος ευφυής ελεγκτής θα πρέπει να μπορεί να μιμηθεί τον καλύτερο άνθρωπο χειριστή της συγκεκριμένης διεργασίας. Προκειμένου να σχεδιαστεί ένας εύρωστος ασαφής ελεγκτής αρχικά ο σχεδιαστής θα πρέπει να καταγράψει του κανόνες με βάση τους οποίους λειτουργεί ο άνθρωπος χειριστής της διεργασίας ώστε το σύστημα να είναι εύρωστο, να έχει δηλαδή, την ικανότητα να παραμένει λειτουργικό κάτω από μη αναμενόμενες συνθήκες. Μέσω της αναπαράστασης των λεκτικών όρων από τα ασαφή σύνολα η ασαφής λογική αποτελεί τη γέφυρα επικοινωνίας ανάμεσα στον άνθρωπο και τη μηχανή. Η ασαφής λογική δημιουργήθηκε ώστε οι υπολογιστικές μηχανές να μπορούν να χειριστούν λεκτικούς όρους. Ο έλεγχος συστημάτων των οποίων η σχέση διέγερσης-απόκρισης χαρακτηρίζεται από ισχυρές μη γραμμικότητες είναι εφικτός στο πλαίσιο της ασαφούς λογικής.

Οι βασικές αρχές σχεδιασμού ενός ασαφούς ελεγκτή είναι :

- **Ορθότητα**: Η ικανότητα εκτέλεσης των λειτουργικών απαιτήσεων του συστήματος με ασφάλεια.
- **Ευρωστία**: Η ικανότητα του συστήματος να παραμένει λειτουργικό κάτω από μη αναμενόμενες συνθήκες.
- **Επεκτασιμότητα**: Η δυνατότητα επέκτασης του υλικού και του λογισμικού χωρίς επανασχεδίαση του συστήματος από την αρχή.

Τα βασικά δομικά στοιχεία ενός ασαφούς ελεγκτή είναι:

- **Η βάση γνώσης** (knowledge base) στην οποία είναι αποθηκευμένοι οι κανόνες (if-then rules) για τον έλεγχο της διαδικασίας.
- **Τα ασαφή σύνολα** (fuzzy sets) τα οποία χρησιμοποιούνται για να αναπαραστήσουν τις μεταβλητές εισόδου και εξόδου με τους λεκτικούς όρους.
- **Ο ασαφοποιητής** (fuzzyfier) ο οποίος μετατρέπει τις πραγματικές τιμές της εισόδου σε ασαφή σύνολα.
- **Ο μηχανισμός συμπερασμού** (inference engine) ο οποίος επεξεργάζεται τις εξόδους του ασαφοποιητή και με χρήση της βάσης γνώσης εξάγει τα ασαφή σύνολα των συμπερασμάτων.
- **Ο αποασαφοποιητής** (defuzzyfier) ο οποίος μετατρέπει τα συμπεράσματα που εξάγει ο μηχανισμός συμπερασμού σε πραγματικούς αριθμούς για να μπορεί να γίνει μετάδοση της δράσης ελέγχου στη διαδικασία.

Για τη σχεδίαση ενός ασαφούς ελεγκτή θα πρέπει να ακολουθηθούν τα παρακάτω βήματα:

- **Λεκτικός διαμερισμός (κατανομή)** των εισόδων και των εξόδων. Ο σχεδιαστής πρέπει να αναπαραστήσει τις μεταβλητές εισόδου και εξόδου με λεκτικούς όρους.
- **Διατύπωση των κανόνων.** Τα ασαφή σύνολα μετά την κατανομή των εισόδων και εξόδων αποθηκεύονται υπό τη μορφή συναρτήσεων συμμετοχής στον υπολογιστή και έπειτα ακολουθεί η διατύπωση των κανόνων.
- **Καθορισμός του τύπου της ασαφούς συνεπαγωγής.** Μετά τη διατύπωση των κανόνων είναι απαραίτητος ο καθορισμός του ασαφούς τύπου συνεπαγωγής. Οι πιο γνωστοί τύποι ασαφούς συνεπαγωγής είναι:

  α) του **Mamdani**, όπου χρησιμοποιείται ο *τελεστής max-min*, ο οποίος λαμβάνει το μικρότερο από τους βαθμούς συμμετοχής των ασαφοποιημένων τιμών και παράγει το βαθμό εκπλήρωσης (degree of fulfillment) του κάθε κανόνα. Ο βαθμός εκπλήρωσης του κανόνα δηλώνει τη βαρύτητα που έχει το αποτέλεσμα του κανόνα.

  β) του **Larsen**, όπου χρησιμοποιείται ο *τελεστής max-product*, ο οποίος υπολογίζει το βαθμό εκπλήρωσης του κανόνα πολλαπλασιάζοντας τους βαθμούς συμμετοχής των ασαφοποιημένων τιμών.
- **Επιλογή του τύπου της απο-ασαφοποίησης**. Η από-ασαφοποίηση παράγει μία ευκρινή (crisp) τιμή από ένα ασαφές σύνολο. Είναι με λίγα λόγια, η αντίθετη διαδικασία από την ασαφοποίηση. Ορισμένοι μέθοδοι από-ασαφοποίησης είναι:
  - ***Κέντρου βάρους*** (Centroid defuzzycation ή center of area ή COA), όπου υπολογίζεται το κέντρο βάρους της κατανομής του ασαφούς συνόλου της εξόδου:

$$z = \frac{\sum y_i \mu_C(y_i)}{\sum \mu_C(y_i)} \quad (2.4)$$

  - ***Μέσου του Μεγίστου*** (Middle of Maxima ή MOM). Είναι ο μέσος όρος όλων των στοιχείων y_i , $i=1,\ldots,N$ που παίρνουν την μέγιστη τιμή στο ασαφές σύνολο C [4].

$$z = \frac{1}{N}\sum_{i=1}^{N} y_i \quad (2.5)$$

Στις περισσότερες εφαρμογές χρησιμοποιούνται δύο τύποι ασαφών ελεγκτών, ο ελεγκτής τύπου Mamdani και τύπου Sugeno.

Για το σχεδιασμό ελεγκτή τύπου Mamdani η διαδικασία του ασαφούς συμπερασμού εκτελείται αρχικά με την ασαφοποίηση των τιμών των εισόδων (fuzzyfication), την εκτίμηση των κανόνων (rule evaluation), την συνάθροιση (aggregation) των

συμπερασμάτων των εξόδων και τέλος την από-ασαφοποίηση τους (defuzzification). Στη διαδικασία της ασαφοποίησης καθορίζεται ο βαθμός κατά τον οποίο οι τιμές των εισόδων ανήκουν στο καθένα από τα ασαφή σύνολα. Στη συνέχεια αφού οι είσοδοι ασαφοποιηθούν, εφαρμόζονται στα υποθετικά μέρη (antecedents) των κανόνων. Αν ένας κανόνας έχει πολλές υποθέσεις, τότε μέσω των τελεστών AND ή OR δίνεται ένα αριθμός που αντιπροσωπεύει το αποτέλεσμα της εκτίμησης του μέρους της υπόθεσης.

Αν θεωρήσουμε δυο ασαφή σύνολα Α και Β ορισμένα πάνω στο ίδιο κλασικό σύνολο Χ. Τότε η τομή $A \cap B$ αυτών είναι ένα ασαφές σύνολο με συνάρτηση συμμετοχής:

$$\mu_{A\cap B}(x) = \mu_A(x) \wedge \mu_B(x) \quad (2.6)$$

όπου $\wedge$ ο τελεστής ελαχίστου Mamdani $\mu_{A\cap B}(x) = \min(\mu_A(x), \mu_B(x))$ και δίνεται η μικρότερη τιμή που εκφράζει την εκτίμηση του κανόνα.

Αντίστοιχα η ένωση $A \cup B$ είναι ένα ασαφές σύνολο του Χ με συνάρτηση συμμέτοχης

$$\mu_{A\cup B}(x) = \mu_A(x) \vee \mu_B(x) \quad (2.7)$$

όπου $\vee$ ο τελεστής μεγίστου Mamdani $\mu_{A\cup B}(x) = \max(\mu_A(x), \mu_B(x))$ και δίνεται η μικρότερη τιμή που εκφράζει την εκτίμηση του κανόνα. Η τομή δυο συνόλων αντιστοιχεί στο λεκτικό ΚΑΙ (AND) ενώ η ένωση στο λεκτικό Ή (OR).

Στη συνέχεια, παράγεται ο βαθμός εκπλήρωσης (degree of fulfillment) του κανόνα και εκφράζεται με την αντίστοιχη τομή-α (α-cut) του ασαφούς συνόλου που αντιπροσωπεύει την βαρύτητα που έχει το αποτέλεσμα του κανόνα. Έστω ένα ασαφές σύνολο Α με συνάρτηση συμμετοχής $\mu_A(x)$, η τομή-α του ασαφούς συνόλου Α είναι ένα νέο ασαφές σύνολο με συνάρτηση συμμετοχής:

$$\mu_A(x) = \begin{cases} \mu_A(x), & 0 \le \mu_A(x) < \alpha \\ \alpha, & \alpha \le \mu_A(x) \le 1 \end{cases} \quad (2.8)$$

Τα αντίστοιχα α-cuts που δημιουργούνται στις εξόδους κάθε κανόνα συναθροίζονται με την εφαρμογή του τελεστή max ώστε να προκύψει ένα ασαφές σύνολο το οποίο αποσαφοποιείται με εφαρμογή κάποιας μεθόδου π.χ. του κέντρου βάρους (centroid) (2.4) ώστε να παραχθεί μια αυστηρά αριθμητική τιμή [5].

Στο μοντέλο του Sugeno οι συναρτήσεις συμμετοχής των εξόδων είναι είτε γραμμικές συναρτήσεις (πολυώνυμα) είτε σταθερές, άρα πάντα σαφείς (crisp), καθιστώντας τα

μοντέλα αυτά μία μίξη ασαφούς και σαφούς συστήματος σε αντίθεση με τα μοντέλα Mamdani στα οποία οι συναρτήσεις συμμετοχής της εξόδου είναι ασαφείς όπως ακριβώς και αυτές των εισόδων. Οι ασαφείς ελεγκτές τύπου Sugeno απαρτίζονται από κανόνες στους οποίους το τμήμα συμπεράσματος περιγράφεται από ένα γραμμικό μοντέλο, δηλαδή από ένα πολυώνυμο των εισόδων του μοντέλου και έχουν την παρακάτω μορφή:

$$Εάν\ x_i\ είναι\ A_1^i\ KAI\ \dots KAI\ x_n\ είναι\ A_n^i$$
$$Τότε\ y^i = c_0^i + c_1^i x_1 + \cdots + c_n^i x_n \quad με \quad i=1\dots m$$

Όπου m είναι ο ολικός αριθμός των κανόνων, x_k $k=1,2,\dots,n$ είναι η k είσοδος , y^i είναι η έξοδος του i κανόνα, A_κ^i είναι ασαφή σύνολα και c_k^i είναι οι παράμετροι (σταθερές) της εξόδου (του συμπεράσματος).

Το ολικό συμπέρασμα του ασαφούς αυτού συστήματος δίνεται από τη μέση τιμή των εξόδων y^i με βάρη w^i δηλαδή:

$$y = \frac{\sum_{i=1}^{m} w^i y^i}{\sum_{i=1}^{m} w^{\,i}} \quad (2.9)$$

Όπου w^i είναι η προσαρμοστικότητα του αριστερού μέλους του κανόνα i, δηλαδή:

$$w^i = \prod_{k}^{n} \mu_{A_k^i}(x_k) \quad (2.10)$$

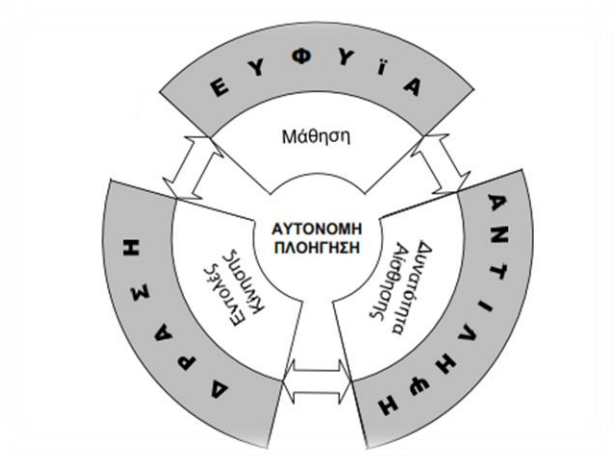
[6]

Η ασαφής λογική γνώρισε μεγάλη επιτυχία και εφαρμόστηκε σε ένα πλήθος εφαρμογών σε συστήματα με διαφορετικό βαθμό πολυπλοκότητας. Η πρώτη βιομηχανική εφαρμογή έγινε το 1982 στη Δανία και αφορούσε τον έλεγχο ενός κλιβάνου τσιμέντου [7], αν και ανακαλύφθηκε στη δύση, ο ασαφής έλεγχος βρήκε μεγάλη άνθηση και εφαρμογή στην Ασία και κυρίως στην Ιαπωνική βιομηχανία. Πολλές οικιακές συσκευές χρησιμοποιούν ασαφείς ελεγκτές όπως είναι τα κλιματιστικά και τα πλυντήρια.

2.4 Έλεγχος έντροχου ρομποτικού οχήματος με ασαφή λογική

Τα αυτόνομα έντροχα ρομποτικά οχήματα έχουν τη δυνατότητα να δρουν σε εχθρικό και μη δομημένο περιβάλλον με διαφορετικό βαθμό πολυπλοκότητας. Μέσα σε αυτό το

περιβάλλουν καλούνται να εκτελέσουν κάποια αποστολή που τους έχει ανατεθεί όπως εξερεύνηση, χαρτογράφηση, διάσωση, κ.α. Για να είναι ικανά να εκτελέσουν την αποστολή τους θα πρέπει να έχουν κάποια χαρακτηριστικά, με κύριο την *αντίληψη* του περιβάλλοντος στο οποίο βρίσκονται και τους κινδύνους που μπορεί να αντιμετωπίσουν. Τις πληροφορίες για το περιβάλλον τις λαμβάνουν από κατάλληλα αισθητήρια με τα οποία είναι εξοπλισμένα και συνήθως τις χρησιμοποιούν για τον υπολογισμό της θέσης τους για την κίνησή τους προς κάποιο στόχο, ή την αποφυγή εμποδίων κ.α. Άλλο χαρακτηριστικό που πρέπει να έχουν είναι η *ευφυία* ώστε να μπορούν να διαχειριστούν τα δεδομένα από τα αισθητήρια, να μαθαίνουν και να προσαρμόζονται στο άγνωστο περιβάλλον και να κινούνται αυτόνομα μέσα σ'αυτό. Επίσης θα πρέπει να *δρουν* χωρίς τη βοήθεια του ανθρώπου και να φέρουν εις πέρας τις αποστολές που τους έχουν δοθεί [8].

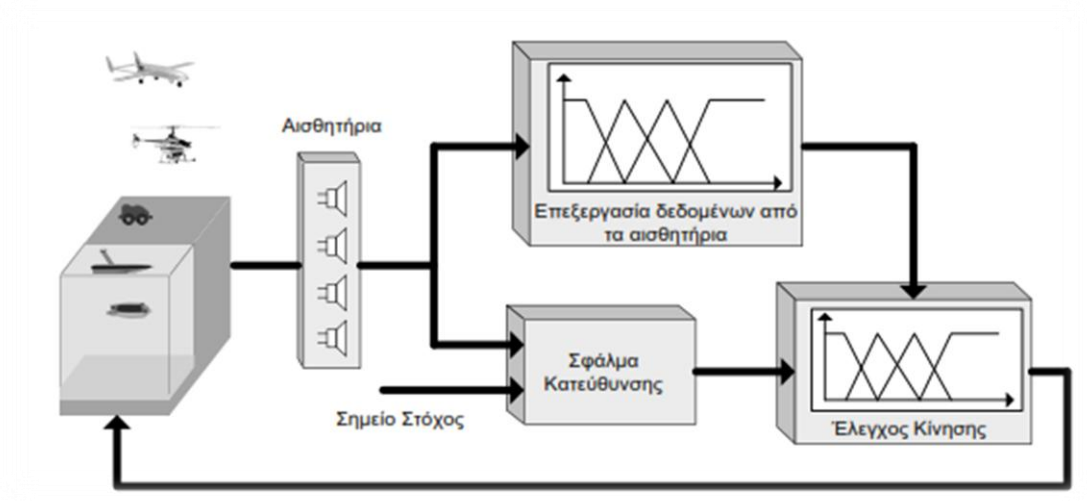


Σχήμα 2.2 Αλληλεπίδραση των απαιτούμενων χαρακτηριστικών για αυτόνομη πλοήγηση [8]

Η κίνηση ενός οχήματος σε ένα άγνωστο, εχθρικό και απρόβλεπτο περιβάλλον απαιτεί ελεγκτές που να μπορούν να διαχειριστούν την αβεβαιότητα, να δρουν γρήγορα και αποτελεσματικά, με τρόπο ανάλογο με αυτόν που θα δρούσε ένας χειριστής. Η ικανότητα της ασαφούς λογικής να διαχειριστεί την αβεβαιότητα και να μιμηθεί τον τρόπο δράσης ενός ανθρώπου-χειριστή[9], είναι ένας από τους κύριους λόγους που χρησιμοποιείται εκτενώς για την πλοήγηση όλων των τύπων ρομποτικών οχημάτων. Υπάρχουν διάφορες στρατηγικές υλοποίησης ασαφών ελεγκτών για την πλοήγηση έντροχων ρομποτικών οχημάτων. Συγκεκριμένα στη [10] προτάθηκε από τους X. Yang *et al.*, ένας ασαφής ελεγκτής που έχει διαφορετικά επίπεδα και είναι ικανός να καθοδηγεί ένα ρομποτικό όχημα σε προκαθορισμένο στόχο, δημιουργώντας ανάλογα με τα εμπόδια που συναντά υποστόχους έως να φτάσει στο τελικό του σημείο. Στην [11] χρησιμοποιήθηκε ασαφής ελεγκτής που συντονίζει ένα σύνολο υπό-ελεγκτών προκειμένου το όχημα να καθοδηγηθεί σε συγκεκριμένο στόχο. Αντίστοιχες τεχνικές για

την πλοήγηση ενός οχήματος με χρήση ασαφούς ελέγχου παρουσιάζονται και στις [12-17].

Η αυτόνομη πλοήγηση ρομποτικών οχημάτων μπορεί να χωριστεί σε δύο μεγάλες κατηγορίες. Η πρώτη κατηγορία αφορά το πρόβλημα συνολικά, δηλαδή τη δημιουργία ενός μονοπατιού που να οδηγεί στον τελικό στόχο και η δεύτερη κατηγορία αφορά το πρόβλημα τοπικά, δηλαδή πώς το όχημα ακολουθεί το μονοπάτι και ταυτόχρονα αποφεύγει εμπόδια που βρίσκονται κοντά ή πάνω στο μονοπάτι. Ένα άλλο ζήτημα είναι η σωστή διαχείριση των δεδομένων των αισθητηρίων, καθώς αυτά είναι η πηγή πληροφόρησής του για το περιβάλλον αλλά και για την κατάσταση του οχήματος. Τα δεδομένα αυτά συνήθως έχουν προβλήματα λόγω του θορύβου των μετρήσεων, αντικρουόμενων πληροφοριών από διαφορετικά αισθητήρια ή ακόμα μπορεί να χρειάζονται επεξεργασία προκειμένου να γίνουν χρήσιμα για την εξαγωγή κάποιου συμπεράσματος προκειμένου να δοθούν οι κατάλληλες εντολές κίνησης στο όχημα. Για να είναι δυνατή η χρήση αυτής της πληροφορίας και να μπορούν να εξαχθούν ασφαλείς εντολές κίνησης για ρομποτικά οχήματα έχει προταθεί στη βιβλιογραφία από τους Tsourveloudis *et al.* [18] η γενική αρχιτεκτονική πλοήγησης αυτόνομων ρομποτικών οχημάτων που παρουσιάζεται στο Σχήμα 2.3.



Σχήμα 2.3 Αρχιτεκτονική ελέγχου βασιζόμενη στην ασαφή [19]

Τα δεδομένα από τα αισθητήρια του αυτόνομου ρομποτικού οχήματος διοχετεύονται στο τμήμα που είναι υπεύθυνο για την επεξεργασία τους και στο τμήμα που είναι υπεύθυνο για τον υπολογισμό του σφάλματος κατεύθυνσης. Κατά την επεξεργασία τους εξάγεται πληροφορία για την τυχόν ύπαρξη εμποδίων κοντά στο ρομποτικό όχημα σε σχέση με την γωνία κατεύθυνσης του οχήματος και με την θέση του. Τα δεδομένα από τα αισθητήρια σε συνδυασμό με το σημείο στόχο χρησιμοποιούνται για την εξαγωγή του σφάλματος κατεύθυνσης του οχήματος. Όλη η ανωτέρω πληροφορία συντίθεται στο

δεύτερο επίπεδο και εξάγονται οι εντολές κίνησης του οχήματος. Αυτή η αρχιτεκτονική αυτόνομης πλοήγησης έχει χρησιμοποιηθεί με επιτυχία σε διαφορετικούς τύπους οχημάτων. Συγκεκριμένα οι Doitsidis *et al.* [20], χρησιμοποίησαν αυτήν την αρχιτεκτονική για την πλοήγηση και την ταυτόχρονη αποφυγή εμποδίων ενός οχήματος που στρίβει με ολίσθηση των τροχών. Η ίδια αρχιτεκτονική σε συνδυασμό με πεδία δυναμικού χρησιμοποιήθηκε για την πλοήγηση ενός οχήματος που κινείται σε εσωτερικό χώρο [21].

2.5 Αλγόριθμοι βελτιστοποίησης

Ως πρόβλημα βελτιστοποίησης στα μαθηματικά ορίζεται η αναζήτηση βέλτιστων παραμέτρων ενός συνήθως περίπλοκου συστήματος. Προβλήματα βελτιστοποίησης απαντώνται σε πολλά επιστημονικά πεδία όπως π.χ. στη φυσική, στη χημεία, στην οικονομία κ.α. Στα μαθηματικά διατυπώνεται ένα πρόβλημα βελτιστοποίησης σαν πρόβλημα ελαχιστοποίησης ή μεγιστοποίησης μιας συνάρτησης μίας ή πολλών μεταβλητών.

Οι αλγόριθμοι βελτιστοποίησης μπορούν να κατηγοριοποιηθούν σε δύο βασικές κατηγορίες: Ντετερμινιστικοί και Στοχαστικοί Αλγόριθμοι Βελτιστοποίησης.

Οι Ντερμινιστικοί αλγόριθμοι βασίζονται σε αναλυτικές και αλγεβρικές μεθόδους για τον ακριβή ορισμό ελαχίστων (ή μεγίστων) συνάρτησης πολλών μεταβλητών. Δεν εισάγουν καμία τυχαιότητα και φτάνουν στην ίδια τελική λύση εάν ξεκινήσουν από την ίδια αρχική λύση. Χαρακτηριστικό παράδειγμα είναι ο αλγόριθμος Αναρρίχησης Λόφου (Hill Climbing).

Οι Στοχαστικοί αλγόριθμοι εισάγουν τον όρο της τυχαιότητας με αποτέλεσμα σχεδόν κάθε φορά που θα εκτελεστεί ο αλγόριθμος να παραχθεί μια διαφορετική λύση παρ'όλο που έχουμε το ίδιο σημείο εκκίνησης. Χαρακτηριστικό παράδειγμα είναι οι *Γενετικοί Αλγόριθμοι* (Genetic Algorithms (GA)) και ο *Αλγόριθμος Σμήνους Σωματιδίων* (Particle Swarm Optimization (PSO)).

Οι Στοχαστικοί αλγόριθμοι μπορούν να ταξινομηθούν σε Ευρετικούς και Μεθευρετικούς. Η Ευρετική Αναζήτηση (Heuristic Search), ακολουθεί μια διαδικασία στην οποία περιορίζεται ο χώρος αναζήτησης της βέλτιστης λύσης. Αυτό γίνεται για να μειωθεί ο υπολογιστικός χρόνος με συνέπεια το αποτέλεσμα να μην είναι η θεωρητικά άριστη λύση αλλά μια πολύ καλή προσέγγιση αυτής. Αλγόριθμοι τέτοιου είδους είναι οι αλγόριθμοι Απληστίας (Greedy Algorithms), και οι Προσεγγιστικοί αλγόριθμοι (Approximation Algorithms).

Η μεθευριστική αναζήτηση κατευθύνει και τροποποιεί άλλες ευριστικές μεθόδους για να παράγει λύσεις πέραν αυτές που κανονικά επιτυγχάνονται κατά την αναζήτηση

τοπικού βέλτιστου. Η μεθευρετική διαδικασία εφαρμόζει κάποια μορφή στοχαστικής βελτιστοποίησης, έτσι ώστε η λύση που θα δώσει να εξαρτάται από το σύνολο των τυχαίων μεταβλητών που δημιουργήθηκαν. Αναζητώντας σε ένα μεγάλο σύνολο πιθανών λύσεων, η μεθευρετική διαδικασία μπορεί να βρει καλές λύσεις με λιγότερη υπολογιστική προσπάθεια από αλγόριθμους, επαναληπτικές μεθόδους, ή απλά ευρετικές διαδικασίες .

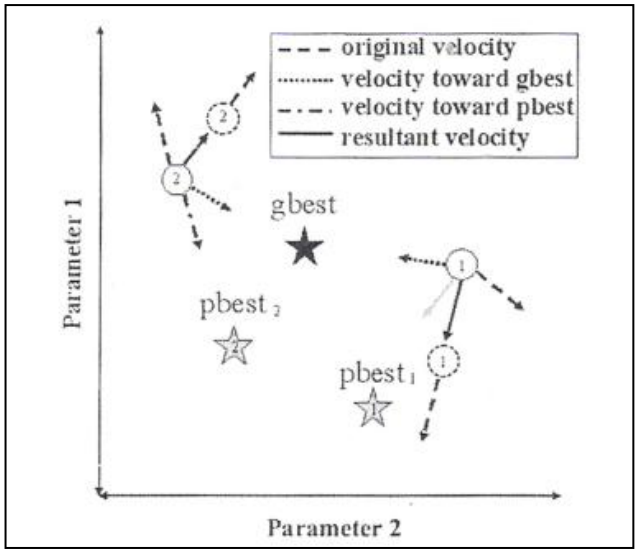
Οι Μεθευρετικοί αλγόριθμοι μπορούν να κατηγοριοποιηθούν σε αλγόριθμους μοναδικής λύσης όπως είναι η Προσομοιωμένη Ανόπτηση (Simulated Annealing), και η Περιορισμένη Αναζήτηση (TabuSearch (TS)), και πληθυσμού λύσεων όπως είναι οι Εξελικτικοί Αλγόριθμοι και οι Νοημοσύνης Σμήνους. Στους Εξελικτικούς ανήκουν οι Γενετικοί Αλγόριθμοι και η Διαφορική Εξέλιξη ενώ στους Νοημοσύνης Σμήνους ανήκουν ο Αλγόριθμος Βελτιστοποίησης Αποικίας Μελισσών (Bees Colony (BC)), ο αλγόριθμος Βελτιστοποίησης Αποικίας Μυρμηγκιών (Ant Colony Optimization algorithm (ACO)) και ο Αλγόριθμος Σμήνους Σωματιδίων (Particle Swarm Optimization (PSO)) [22].

2.6 Βελτιστοποίηση Σμήνους Σωματιδίων

Οι αλγόριθμοι Νοημοσύνης Σμήνους (Swarm Intelligence) έχουν σαν πηγή έμπνευσης την συλλογική συμπεριφορά (collective behavior) και την εκδηλούμενη νοημοσύνη (emergent intelligence) που εμφανίζεται σε πληθυσμούς πουλιών και ψαριών [23]. Ο συγκεκριμένος τύπος αλγορίθμου στηρίζεται στη κοινωνική ανταλλαγή πληροφορίας μεταξύ των μελών ενός πληθυσμού η οποία προσδίδει ένα εξελικτικό πλεονέκτημα στον πληθυσμό.

Μια περίπτωση αλγορίθμου βελτιστοποίησης σμήνους είναι η *Βελτιστοποίηση Σμήνους Σωματιδίων* (Particle Swarm Optimization (PSO)). Πρόκειται για μια στοχαστική μέθοδο, που χρησιμοποιεί *πληθυσμούς για την αναζήτηση λύσεων εντός του χώρου αναζήτησης*. Ωστόσο, μια μεγάλη διαφορά με τους εξελικτικούς αλγόριθμους είναι η κίνηση κάθε μέλους του πληθυσμού με μια προσαρμόσιμη ταχύτητα (adaptable velocity) στον χώρο αναζήτησης. Επιπλέον, κάθε μέλος του πληθυσμού έχει μια μνήμη στην οποία διατηρεί την καλύτερη θέση που επισκέφτηκε ποτέ. Σε προβλήματα ελαχιστοποίησης αυτή η θέση είναι το σημείο του χώρου με την μικρότερη συναρτησιακή τιμή που έχει επισκεφτεί ποτέ το μέλος του πληθυσμού [24]. Επιπροσθέτως, τα μέλη του πληθυσμού ανταλλάσσουν μεταξύ τους πληροφορία σχετικά με την καλύτερη θέση (μνήμη) που έχει το καθένα. Έτσι, ορίζοντας γειτονιές μεταξύ των μελών του πληθυσμού, υπάρχει ροή πληροφορίας μεταξύ των μελών που τις απαρτίζουν και η κίνηση του σμήνους είναι απόρροια της στοχαστικής επιτάχυνσης των μελών του προς τις προσωπικές καλύτερες θέσεις τους και προς τις καλύτερες θέσεις των

γειτόνων τους. Αυτή η έννοια της «επιτάχυνσης» η οποία χρησιμοποιούνταν ιδιαιτέρως σε συστήματα σωματιδίων της Σωματιδιακής Φυσικής [25] έδωσε την ιδέα στους Eberhart και Kennedy να ονομάσουν τα μέλη του πληθυσμού *σωματίδια* (particles). Επίσης, ο πληθυσμός ονομάστηκε *σμήνος* (swarm) λόγω της έμπνευσης που προήλθε από τον προσομοιωτή σμήνους που αναφέραμε παραπάνω. Έτσι, η μέθοδος ονομάστηκε Βελτιστοποίηση με Σμήνος Σωματιδίων (Particle Swarm Optimization) [26].



Σχήμα 2.4 Κίνηση σωματιδίων στο χώρο αναζήτησης σύμφωνα με τον αλγόριθμο PSO [27]

2.7 Είδη αλγορίθμων Βελτιστοποίησης Σμήνους Σωματιδίων

Έχοντας σαν πυρήνα τις βασικές αρχές που παρουσιάστηκαν στην παράγραφο 2.6 ξεκίνησε μια μεγάλη διαδικασία προσθαφαιρέσεως ιδεών και παραμέτρων του μοντέλου κίνησης που είχε αναπτυχθεί, ώσπου παρουσιάστηκε η πρώτη (απλή) έκδοση της μεθόδου, για προβλήματα αριθμητικής βελτιστοποίησης [24]. Στη συνέχεια θα αναφέρουμε τις παραλλαγές του αλγορίθμου και τις διαφορές που παρουσιάζουν μεταξύ τους.

Βασικός αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων

Ο αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων (PSO) είναι μια στοχαστική μέθοδος που βασίζεται σε έναν πληθυσμό (σμήνος) από σωματίδια που κινούνται μέσα σε έναν πολυδιάστατο χώρο αναζήτησης (πιθανές λύσεις). Η θέση του κάθε σωματιδίου αντιπροσωπεύει μια πιθανή λύση και μεταβάλλεται σύμφωνα με την εμπειρική γνώση τη δική του, αλλά και των γειτόνων του. Έστω $x_i(t)$ η θέση ενός σωματιδίου i στο χώρο

αναζήτησης τη διακριτή χρονική στιγμή t, η θέση του σωματιδίου μεταβάλλεται με την επίδραση της ταχύτητας $v_i(t)$ που έχει το σωματίδιο σύμφωνα με τη σχέση:

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (2.11)$$

Με $x_i(0) \sim U(x_{min}, x_{\max})$.

Το διάνυσμα της ταχύτητας οδηγεί τη διαδικασία βελτιστοποίησης και αντικατοπτρίζει τόσο την εμπειρική γνώση του σωματιδίου όσο και την κοινωνικά ανταλλασσόμενη πληροφορία από τη γειτονιά του σωματιδίου. Η εμπειρική γνώση του σωματιδίου αναφέρεται ως η *γνωσιακή συνιστώσα* (cognitive component), η οποία ωθεί το σωματίδιο στη δική του βέλτιστη θέση (personal best position) και η κοινωνικά ανταλλασσόμενη πληροφορία αναφέρεται ως η *κοινωνική συνιστώσα* (social component) της εξίσωσης ταχύτητας και ωθεί το σωματίδιο στη καλύτερη συνολικά θέση (global best position) [28].

Ως προς το μέγεθος της γειτονιάς έχουν προταθεί δύο παραλλαγές του αλγορίθμου: της *καθολικά βέλτιστης λύσης* (global best (gbest) PSO) και της *τοπικά βέλτιστης λύσης* (local best (lbest) PSO).

<u>Βελτιστοποίηση Σμήνους Σωματιδίων καθολικά βέλτιστης λύσης</u>

Για τον αλγόριθμο Βελτιστοποίησης Σμήνους Σωματιδίων καθολικά βέλτιστης λύσης (global best PSO), η γειτονιά κάθε σωματιδίου είναι ολόκληρο το σμήνος. Στην περίπτωση αυτή, η κοινωνική συνιστώσα της εξίσωσης μεταβολής της ταχύτητας του σωματιδίου αντικατοπτρίζει την πληροφορία από όλα τα σωματίδια του σμήνους. Η πληροφορία αυτή αντιστοιχεί στη βέλτιστη θέση του σμήνους $\hat{y}_j(t)$. Η ταχύτητα ενός σωματιδίου *i* δίνεται από τη σχέση:

$$v_{ij}(t+1) = v_{ij} + c_1 r_{1j}(t)\left[y_{ij}(t) - x_{ij}(t)\right] + c_2 r_{2j}(t)\left[\hat{y}_j(t) - x_{ij}(t)\right] \quad (2.12)$$

Όπου v_{ij} είναι η ταχύτητα του σωματιδίου, $i=1,\ldots,n_p$ με n_p το μέγεθος του πληθυσμού στη διάσταση $j=1,\ldots,n$, τη χρονική στιγμή t, $x_{ij}(t)$ είναι η θέση του σωματιδίου i στη διάσταση j τη χρονική στιγμή t, c_1 και c_2 θετικές σταθερές επιτάχυνσης, οι οποίες χρησιμοποιούνται για να ρυθμίσουν τις συνεισφορές της γνωσιακής και της κοινωνικής συνιστώσας αντίστοιχα, και $r_{1j}(t)$, $r_{2j}(t)$, τυχαίες τιμές στο διάστημα $[0,1]$, οι οποίες εισάγουν στοχαστικότητα στον αλγόριθμο. Θα πρέπει να τονιστεί η σημασία του όρου v_{ij} της εξίσωσης (2.12). Η ταχύτητα αυτή έχει το ρόλο της αδράνειας του σωματιδίου

και το εμποδίζει να μην εγκλωβιστεί σε τοπικά αλλά και ολικά ακρότατα. Για παράδειγμα στην περίπτωση που η τρέχουσα θέση είναι η προσωπική καλύτερη για το σωματίδιο αλλά και η συνολικά καλύτερη για το σμήνος τότε αν δεν υπήρχε ο όρος v_{ij} παρατηρούμε ότι σύμφωνα με την εξίσωση (2.12) ο γνωστικός και κοινωνικός όρος θα ήταν μηδέν και το σωματίδιο θα παρέμεινε εγκλωβισμένο στη συγκεκριμένη, έως κάποιο άλλο ολικό ελάχιστο ανακαλυπτόταν από ένα άλλο σωματίδιο. Αντιθέτως, λόγω αυτού του όρου, το σωματίδιο συνεχίζει την αναζήτησή του. Όμως ο όρος αυτός έχει σαν συνέπεια όταν το σωματίδιο έχει βρει το πραγματικό ολικό ακρότατο να μην μπορεί να παραμείνει στη θέση αυτή αλλά να εκτελεί μία ταλάντωση γύρω από αυτό το σημείο ανάλογη της ταχύτητας v_{ij}.

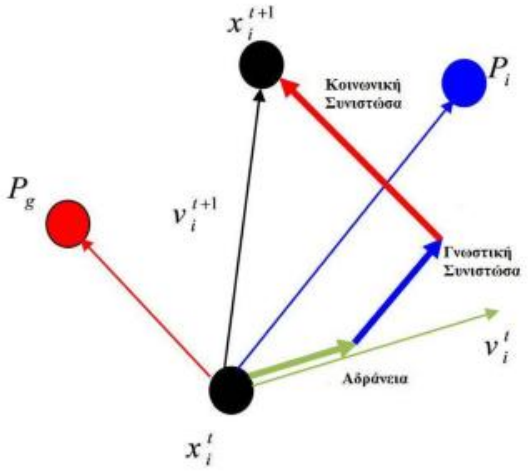
Η ατομική βέλτιστη θέση (pbest) ενός σωματιδίου i, y_i, είναι η καλύτερη θέση που έχει επισκεφθεί το σωματίδιο από την πρώτη χρονική στιγμή. Αν θεωρήσουμε πρόβλημα ελαχιστοποίησης, η θέση pbest την επόμενη χρονική στιγμή $(t+1)$ δίνεται από τη σχέση:

$$y_{ij}(t+1) = \begin{cases} y_{ij}(t), & f\left(x_{ij}(t+1)\right) \geq f\left(y_{ij}(t)\right) \\ x_{ij}(t+1), & f\left(x_{ij}(t+1)\right) < f\left(y_{ij}(t)\right) \end{cases} \quad (2.13)$$

Όπου $f\colon \mathbb{R}^n \to \mathbb{R}$ η συνάρτηση αξιολόγησης που παρέχει την σύνδεση μεταξύ του αλγόριθμου βελτιστοποίησης με τον φυσικό κόσμο και υπολογίζει πόσο κοντά στη βέλτιστη τιμή βρίσκεται μια λύση.

Η καθολικά βέλτιστη θέση (gbest) $\hat{y}_j(t)$ τη χρονική στιγμή *t* ορίζεται ως εξής:

$$\hat{y}_j(t) = \min\left\{f\left(x_1(t)\right), \dots, f\left(x_{n_p}(t)\right)\right\} \quad (2.14) \text{ [28]}$$



Σχήμα 2.5 Συνεισφορά της αρχικής ταχύτητας και της γνωσιακής και κοινωνικής συνιστώσας για την επιλογή της επόμενης θέσης του σωματιδίου [29]

Στη συνέχεια παρουσιάζεται ένας γενικός αλγόριθμος με μορφή *ψευδοκώδικα* της μεθόδου Βελτιστοποίησης Σμήνους Σωματιδίων.

Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων

Αρχικοποίηση

- Επιλογή του μεγέθους του πληθυσμού των σωματιδίων
- Αρχικοποίηση της θέσης και της ταχύτητας κάθε σωματιδίου
- Υπολογισμός της αρχικής συναρτησιακής τιμής του κάθε σωματιδίου σύμφωνα με την συνάρτηση αξιολόγησης
- Εύρεση Βέλτιστης λύσης κάθε σωματιδίου
- Εύρεση Βέλτιστου σωματιδίου ολόκληρου του σμήνους

Κύρια Φάση

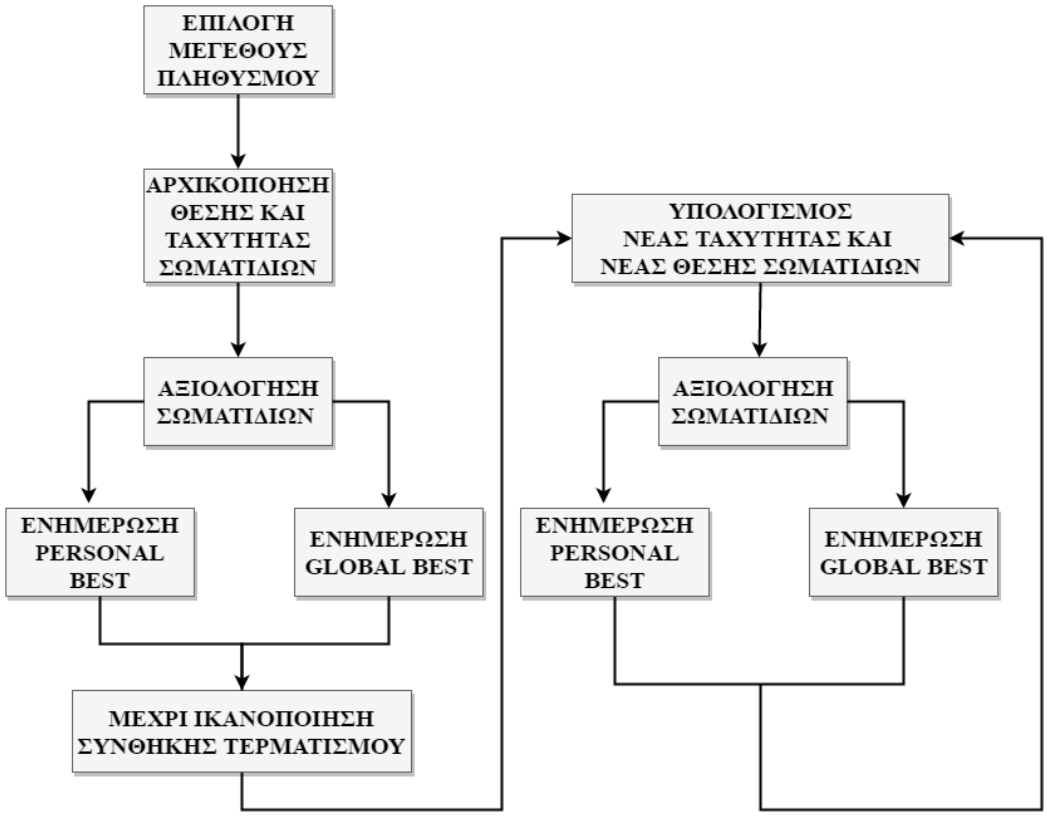
Do until μέχρι την ικανοποίηση της συνθήκης τερματισμού

- Υπολογισμός της ταχύτητας του κάθε σωματιδίου
- Υπολογισμός της νέας θέσης του κάθε σωματιδίου
- Υπολογισμός της νέας συναρτησιακής τιμής του κάθε σωματιδίου
- Ενημέρωση της βέλτιστης λύσης του κάθε σωματιδίου
- Εύρεση του βέλτιστου σωματιδίου ολόκληρου του σμήνους

Enddo

- Επιστροφή βέλτιστου σωματιδίου (βέλτιστης λύσης)

Πίνακας 2.1 Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων



Σχήμα 2.6 Διάγραμμα ροής αλγόριθμου Βελτιστοποίησης Σμήνους Σωματιδίων

Στη συνέχεια παρουσιάζονται κάποια στοιχεία σχετικά με τις παραμέτρους του αλγορίθμου:

Αρχικοποίηση πληθυσμού

Ένα πολύ σημαντικό στοιχείο για την επιτυχή και γρήγορη σύγκλιση του αλγορίθμου είναι η κατάλληλη αρχικοποίηση των σωματιδίων του σμήνους. Ο συνηθέστερος τρόπος είναι να αρχικοποιήσουμε τις θέσεις των σωματιδίων με τυχαίο τρόπο σε όλο τον χώρο λύσεων. Όταν αρχικοποιούμε τα σωματίδια είναι πολύ πιθανό να αφήσουμε κάποια μέρη του χώρου λύσεων ακάλυπτα και αυτό να οδηγήσει είτε σε αργή σύγκλιση προς το βέλτιστο είτε σε παγίδευση του αλγορίθμου σε τοπικό ελάχιστο. Έτσι, ανάλογα με το πρόβλημα που θέλουμε να επιλύσουμε πρέπει να προσπαθούμε να αρχικοποιούμε κατάλληλα τον πληθυσμό ώστε να καλύπτουμε όσο το δυνατόν μεγαλύτερο μέρος του χώρου αναζήτησης. Για παράδειγμα, αν το βέλτιστο βρίσκεται μεταξύ των τιμών 10 και 20 μία αρχικοποίηση ενός διανύσματος ανάμεσα στο 0 και στο 1 είτε δεν θα οδηγήσει ποτέ στο βέλτιστο είτε θα χρειαστεί πάρα πολλές επαναλήψεις για να βρει το βέλτιστο. Άρα σε αυτή την περίπτωση προσπαθούμε να αρχικοποιήσουμε τις τιμές των σωματιδίων με πιο εφικτό τρόπο. Ένας πολύ απλός αλλά και αποτελεσματικός τρόπος

παρουσιάζεται στο [28] όπου αν η ελάχιστη τιμή είναι το x_{min} και η μέγιστη τιμή είναι το x_{max} του χώρου αναζήτησης. τότε όλες οι τιμές του διανύσματος του κάθε σωματιδίου μπορούν να υπολογιστούν από τη σχέση:

$$x_{ij} = x_{min} + r_i(x_{max} - x_{min}) \quad (2.15)$$

Όπου *i=1...n, n* το μέγεθος του πληθυσμού, *r* τυχαίος αριθμός μεταξύ [0,1].

Συντελεστές επιτάχυνσης c_1και c_2

Οι διαφοροποιήσεις ως προς την κίνηση των σωματιδίων μέσα στο χώρο αναζήτησης σε σχέση με τις τιμές των *σταθερών επιτάχυνσης* c_1και c_2 , έχουν ως εξής:

- Εάν το $c_1 = c_2 = 0$ τα σωματίδια πετούν στην κατεύθυνση που τους δίνει η ταχύτητα τους.
- Εάν το $c_1 > 0$ και το $c_2 = 0$ τότε το σωματίδιο επηρεάζεται μόνο από τις προηγούμενες κινήσεις του και κινείται ανεξάρτητα από τα άλλα σωματίδια του σμήνους.
- Εάν $c_2 > 0$ και το $c_1 = 0$ ολόκληρο το σμήνος κυνηγάει ένα σωματίδιο, το βέλτιστο.
- Εάν το $c_1 = c_2$, οι δύο παράγοντες έχουν την ίδια επίδραση επομένως έλκουν το σωματίδιο ισόποσα.
- Εάν το $c_1 \gg c_2$ τότε το σωματίδιο έλκεται πολύ περισσότερο από τις προηγούμενες βέλτιστες τιμές του.
- Εάν $c_1 \gg c_2$ το σωματίδιο έλκεται πολύ περισσότερο από το βέλτιστο του σμήνους.
- Εάν c_1 *και* c_2 είναι μικρά τότε τα σωματίδιο κινείται με ομαλή τροχιά.
- Εάν c_1*και* c_2 είναι μεγάλα τότε τα σωματίδιο έχει μεγάλη επιτάχυνση και κινείται απότομα [30].

Μέγεθος πληθυσμού

Όσο μεγαλύτερο είναι το πλήθος των σωματιδίων, τόσο μεγαλύτερη είναι η αρχική ποικιλομορφία του σμήνους. Ένα μεγάλο σμήνος καλύπτει μεγαλύτερες περιοχές του χώρου αναζήτησης ανά επανάληψη του αλγόριθμου. Όμως, όσο μεγαλύτερος είναι ο αριθμός των σωματιδίων, τόσο αυξάνεται η υπολογιστική πολυπλοκότητα του αλγόριθμου ανά επανάληψη. Γενικότερα, το βέλτιστο μέγεθος σμήνους εξαρτάται από το πεδίο εφαρμογής του αλγόριθμου.

Αριθμός επαναλήψεων

Ο αριθμός των επαναλήψεων που απαιτείται για την επίτευξη βέλτιστης λύσης εξαρτάται από τη φύση του προβλήματος. Μικρός αριθμός επαναλήψεων οδηγεί σε πρόωρο τερματισμό του αλγόριθμου, ενώ μεγάλος αριθμός επαναλήψεων μπορεί να προσθέσει περιττή υπολογιστική πολυπλοκότητα.

Κριτήριο τερματισμού αλγορίθμου

Κάποια από τα κριτήρια τερματισμού [28] μπορούν να είναι:

- Ο μέγιστος αριθμός επαναλήψεων ή υλοποιήσεων της συνάρτησης βελτιστοποίησης.
- Η εύρεση αποδεκτής λύσης.
- Η μη ύπαρξη βελτίωσης για καθορισμένο αριθμό επαναλήψεων.
- Η κλίση της αντικειμενικής συνάρτησης να είναι κατά προσέγγιση μηδενική.

<u>Βελτιστοποίηση Σμήνους Σωματιδίων τοπικά βέλτιστης λύσης</u>

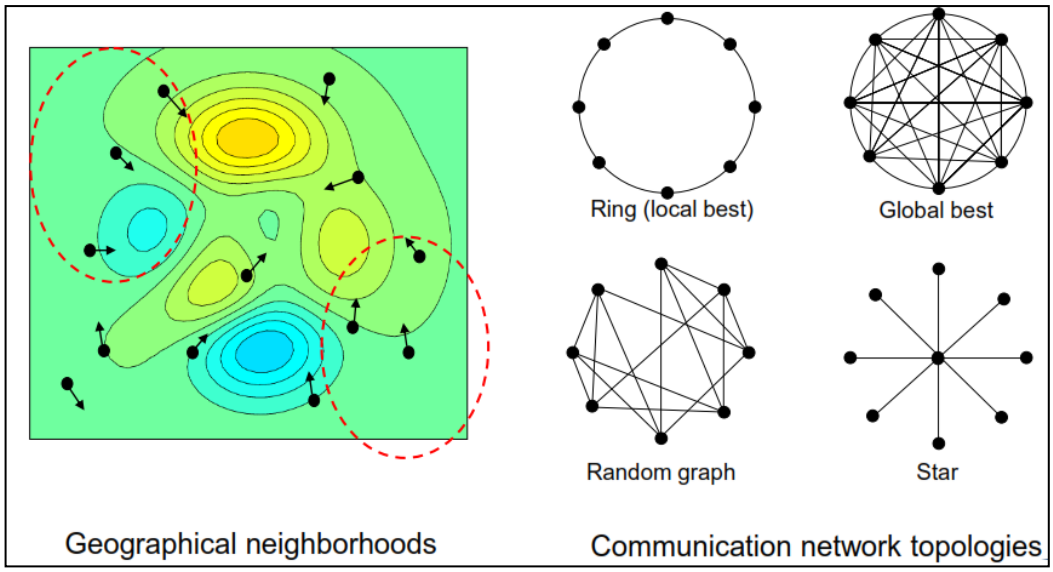
Στον αλγόριθμο Βελτιστοποίησης Σμήνους Σωματιδίων τοπικού βέλτιστου (local best PSO), ο πληθυσμός χωρίζεται σε γειτονιές και η κοινωνική συνιστώσα της εξίσωσης μεταβολής της ταχύτητας απεικονίζει την πληροφορία που ανταλλάσσεται μέσα στη γειτονιά του σωματιδίου, η οποία αντιστοιχεί σε τοπική γνώση του περιβάλλοντος. Για τον αλγόριθμο local best PSO, η ταχύτητα ενός σωματιδίου *i* δίνεται από τη σχέση:

$$v_{ij}(t+1) = v_{ij} + c_1 r_{1j}(t)\left[y_{ij}(t) - x_{ij}(t)\right] + c_2 r_{2j}(t)\left[\hat{y}_{kj}(t) - x_{ij}(t)\right] \quad (2.16)$$

Όπου $k=1,\ldots, n_k$ το πλήθος των γειτονιών, και $\hat{y}_{kj}(t)$ η βέλτιστη θέση της γειτονιάς *k*.

Ο τρόπος που χωρίζονται οι γειτονιές επηρεάζουν την κοινωνικότητα του σωματιδίου και την επικοινωνία του με τα υπόλοιπα επομένως και την απόδοση του αλγορίθμου. Υπάρχουν δύο βασικές τοπολογίες, i) οι γεωγραφικές και ii) των δικτύων επικοινωνίας, ωστόσο μπορεί η τοπολογία να αλλάζει με την πάροδο του χρόνου (δυναμικές τοπολογίες). Η απόδοση του PSO εξαρτάται από τη δομή του κοινωνικού δικτύου. Η ροή της πληροφορίας διά μέσου του κοινωνικού δικτύου καθορίζεται από τους εξής παράγοντες: α) το βαθμό συνδεσιμότητας μεταξύ των κόμβων (μελών) του δικτύου, β) το βαθμό συσσώρευσης (clustering) των κόμβων και γ) τη μέση μικρότερη απόσταση μεταξύ των κόμβων.

Σε ένα κοινωνικό δίκτυο υψηλής συνδεσιμότητας, τα περισσότερα μέλη του δικτύου επικοινωνούν μεταξύ τους, με αποτέλεσμα τη γρήγορη διάχυση της πληροφορίας. Αυτή οδηγεί σε ταχύτερη σύγκλιση σε μια λύση σε σχέση με ένα δίκτυο μικρότερης συνδεσιμότητας. Η ταχύτερη σύγκλιση έχει ως αντίτιμο την αυξημένη πιθανότητα εγκλωβισμού σε τοπικά ελάχιστα του χώρου αναζήτησης. Σε ένα κοινωνικό δίκτυο χαμηλής συνδεσιμότητας, με υψηλό βαθμό συσσώρευσης των σωματιδίων στις γειτονιές, ο χώρος αναζήτησης δεν καλύπτεται επαρκώς, με αποτέλεσμα την αδυναμία εύρεσης των βέλτιστων λύσεων [31]. Ο global best PSO είναι μια ειδική περίπτωση του local best PSO όπου *np=nk*.



Σχήμα 2.7 Διαχωρισμός γειτονιών σύμφωνα με γεωγραφικές τοπολογίες και δίκτυα επικοινωνίας [32]

<u>Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων με αποκοπή ταχύτητας</u>

Η πρώτη βελτίωση του αλγορίθμου ήταν η επιβολή ενός περιορισμού στην ταχύτητα των σωματιδίων (velocity clamping PSO). Αυτή η επιβολή κρίθηκε αναγκαία γιατί παρατηρήθηκε μία έκρηξη των τιμών της ταχύτητας σε πολύ μεγάλες τιμές, ειδικά στις περιπτώσεις σωματιδίων που βρίσκονται πολύ μακριά από τις βέλτιστες τιμές του σμήνους. Έτσι, τα συγκεκριμένα σωματίδια δεν μπορούσαν να ξαναβρούν το βέλτιστο, πράγμα που οδηγούσε σε αργή σύγκλιση του αλγορίθμου. Για να αντιμετωπιστεί αυτό το πρόβλημα προτάθηκε ένας περιορισμός της ταχύτητας με τη χρήση μιας παραμέτρου $v_{max.j}$ που καθορίζει τη μέγιστη επιτρεπόμενη ταχύτητα στη διάσταση *j*. Με αυτόν τον περιορισμό οι ταχύτητες ενημερώνονται από τον ακόλουθο τύπο:

$$v_{ij}(t+1) = \begin{cases} \acute{v}_{ij}(t+1), & \acute{v}_{ij}(t+1) < v_{max,j} \\ v_{max,j}, & \acute{v}_{ij}(t+1) \geq v_{max,j} \end{cases} \quad (2.17)$$

Όπου $\acute{v}_{ij}(t+1)$ η ταχύτητα που υπολογίζεται βάση της εξίσωσης (2.12) ή της (2.16)

Μεγάλη τιμή στο $v_{max.j}$ δίνει τη δυνατότητα στο σωματίδιο να εξερευνήσει μεγάλη περιοχή του χώρου αναζήτησης με μεγάλα βήματα με κίνδυνο όμως να χαθεί κάποιο σημείο που θα μπορούσε να δώσει ολικό ακρότατο, καθώς και να βρεθεί εκτός των ορίων αναζήτησης. Αντιθέτως πολύ μικρή τιμή στη $v_{max.j}$ έχει σαν αποτέλεσμα την εντατικοποίηση της εξερεύνησης γύρω από ένα πολύ καλό σημείο με τον κίνδυνο όμως η αναζήτηση να παγιδευτεί σε ένα τοπικό ακρότατο και είτε να υπάρχει μεγάλο κόστος στο χρόνο εξεύρεσης της βέλτιστη λύσης, είτε να μην μπορέσει να ξεφύγει ποτέ από το σημείο αυτό [24] [30].

<u>Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων με βάρος αδράνειας</u>

Παρότι ο περιορισμός της ταχύτητας βελτιώνει την αναζήτηση, ωστόσο δεν μπορεί να αντιμετωπίσει ικανοποιητικά την ταλάντωση που παρουσιάζει το σωματίδιο γύρω από ένα τοπικό ακρότατο με αποτέλεσμα να μην μπορεί να περιορίσει περισσότερο την περιοχή αναζήτησης και την σύγκλιση στο σημείο αυτό. Για το λόγο αυτό προτάθηκε ο αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων με συντελεστή βάρους αδράνειας (inertia weight (w) PSO)), σύμφωνα με τον οποίο η ταχύτητα του σωματιδίου για την περίπτωση του gbest PSO δίνεται από:

$$v_{ij}(t+1) = wv_{ij} + c_1 r_{1j}(t)\left[y_{ij}(t) - x_{ij}(t)\right] + c_2 r_{2j}(t)\left[\hat{y}_j(t) - x_{ij}(t)\right] \quad (2.18)$$

ενώ για την περίπτωση του lbest PSO, από:

$$v_{ij}(t+1) = wv_{ij} + c_1 r_{1j}(t)\left[y_{ij}(t) - x_{ij}(t)\right] + c_2 r_{2j}(t)\left[\hat{y}_{kj}(t) - x_{ij}(t)\right] \quad (2.19)$$

Εάν επιλεγούν σωστά τα *w*, c_1, c_2 μπορεί να επιτευχθεί σύγκλιση χωρίς τη χρήση του ορίου $v_{max.j}$ για τον περιορισμό της ταχύτητας. Το βάρος αδράνειας *w*, μπορεί να χρησιμοποιηθεί για τον έλεγχο της ισορροπίας μεταξύ της διασποράς της αναζήτησης στο χώρο λύσεων (exploration) και την ικανότητα εντατικοποίησης σε ένα πιθανό πολύ καλό σημείο του χώρου λύσεων (exploitation). Εάν το βάρος αδράνειας $w > 1$ οι ταχύτητες αυξάνονται με την πάροδο του χρόνου και το σμήνος αποκλίνει από τη βέλτιστη λύση. Εάν το βάρος αδράνειας $0 < w < 1$ τότε τα σωματίδια επιβραδύνουν και η σύγκλιση εξαρτάται από την επιλογή των συντελεστών επιτάχυνσης c_1 *και* c_2 [33].

Έχουν προταθεί αρκετές διαφορετικές παραλλαγές για την επιλογή του βάρους αδράνειας w, από σταθερές τιμές μέχρι διαφορετικούς τρόπους μείωσης ή αύξησης κατά τη διάρκεια των επαναλήψεων. Μία επιλογή είναι αρχικά το βάρος αδράνειας να έχει μία τιμή κοντά στη μονάδα, ώστε στα πρώτα στάδια να προωθείτε η εξερεύνηση με μεγάλες ταχύτητες και σε μεγάλες περιοχές, και μετά με μία γραμμική μείωση κατά τη διάρκεια των επαναλήψεων, να τείνει προς το μηδέν ώστε να εξαλειφθούν οι ταλαντώσεις των σωματιδίων στα τελικά στάδια και τα σωματίδια να καταφέρουν να συγκλίνουν στη βέλτιστη λύση. Συνήθως, η τελική τιμή επιλέγεται κοντά στο 0.1, ώστε να αποφευχθεί η ολοκληρωτική απαλοιφή του όρου v_{ij}. Η γραμμική μείωση της αδράνειας δίνεται από τον τύπο:

$$w(t) = w_{max} - \left(\frac{w_{max} - w_{min}}{iter_{max}}\right) \cdot t \quad (2.20)$$

Όπου t η τρέχουσα χρονική στιγμή, $w_{max,}$ w_{min}, είναι η μέγιστη και η ελάχιστη τιμή που μπορεί να πάρει το βάρος αδράνειας και $iter_{max}$ είναι ο μέγιστος αριθμός επαναλήψεων [34].

Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων με παράγοντα περιορισμού

Μια μη στοχαστική μέθοδος ώστε να αποφευχθεί η έκρηξη του σμήνους (δηλαδή η απομάκρυνση από το ολικό βέλτιστο), να εξασφαλιστεί η σύγκλιση και να εξαλειφθούν οι παράμετροι που περιορίζουν την ταχύτητα των σωματιδίων είναι να χρησιμοποιηθεί ένας παράγοντας περιορισμού (constriction factor PSO) στις ταχύτητες των σωματιδίων. Έτσι οι ταχύτητες των σωματιδίων υπολογίζονται για τη περίπτωση του *gbest PSO* σύμφωνα με τη σχέση 2.21, ενώ για τη περίπτωση του *lbest PSO* σύμφωνα με τη σχέση 2.22:

$$v_{ij}(t+1) = x\big(v_{ij} + \varphi_1\big[y_{ij}(t) - x_{ij}(t)\big] + \varphi_2\big[\hat{y}_j(t) - x_{ij}(t)\big]\big) \quad (2.21)$$

$$v_{ij}(t+1) = x\big(v_{ij} + \varphi_1\big[y_{ij}(t) - x_{ij}(t)\big] + \varphi_2\big[\hat{y}_{kj}(t) - x_{ij}(t)\big]\big) \quad (2.22)$$

$$\text{Όπου } x = \frac{2}{\left|2 - \varphi - \sqrt{\varphi(\varphi-4)}\right|} \quad \text{και} \quad \varphi = \varphi_1 + \varphi_2 > 4 \quad (2.23)$$

με $\varphi_1 = c_1 r_{1j}(t)$ και $\varphi_2 = c_2 r_{2j}(t)$.

Όπου $c_1 και\ c_2$ οι συντελεστές επιτάχυνσης και $r_{1j} και\ r_{2j}$ τυχαίοι αριθμοί ∈ [0,1].

Μια άλλη περίπτωση είναι ο συντελεστής περιορισμού να πολλαπλασιάζεται με έναν παράγοντα $k \in [0,1]$ οπότε:

$$x = \frac{2k}{|2-\varphi-\sqrt{\varphi(\varphi-4)}|} \text{ [35] (2.24)}$$

Για k κοντά στο μηδέν υπάρχει εντατικοποίηση της αναζήτησης γύρω από κάποιο πιθανό πολύ καλό σημείο ενώ αν το k πλησιάσει τη μονάδα ο αλγόριθμος εξερευνεί περισσότερες περιοχές στο χώρο αναζήτησης. Η χρήση του PSO με συντελεστή περιορισμού δίνει καλύτερα αποτελέσματα από τις προηγούμενες μορφές του αλγορίθμου.

Μια απλούστερη μορφή είναι η εξής:

$$v_{ij}(t+1) = x\left(v_{ij}(t) + c\big(p_{mj} - x_{ij}(t)\big)\right) \text{ (2.25)}$$

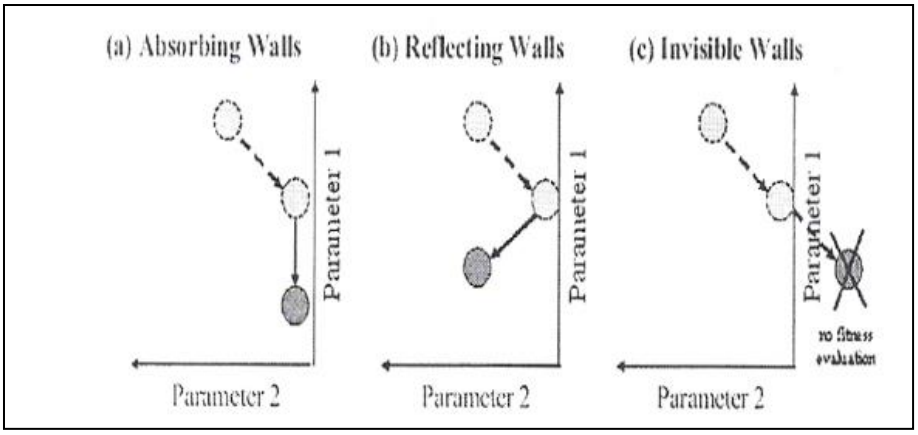
Όπου $x = \frac{2}{|2-c-\sqrt{c^2-4c}|}$ και $c = c_1 + c_2 > 4$ (2.26)

και $p_{mj} = \frac{c_1 p_{ij} + c_2 p_{gj}}{c}$ (2.27)

Όπου p_{ij} η καλύτερη τιμή της συνάρτησης αξιολόγησης του ίδιου του σωματιδίου (personal best) και p_{gj} η καθολικά καλύτερη τιμή για το σμήνος ή της γειτονιάς του σωματιδίου αναλόγως με την τοπολογία που ακολουθείτε στον αλγόριθμο.

Συνθήκες περιορισμών ταχύτητας

Συχνά σε μηχανολογικές εφαρμογές είναι επιθυμητό να περιορίζουμε την έρευνα σε αυτό που είναι φυσικά αποδεκτό. Η εμπειρία των ερευνητών σε τέτοια θέματα έχει δείξει ότι το $v_{max.j}$ (σχέση 2.17), ο παράγοντας περιστολής k (σχέση 2.24) και τα βάρη αδράνειας w (σχέση 2.18, 219), δεν περιορίζουν πάντα τα σωματίδια σε ένα χώρο αποδεκτών λύσεων. Για να λύσουν το πρόβλημα αυτό οι ερευνητές εισήγαγαν τρία διαφορετικά είδη περιοριστικών συνθηκών που είχαν ως στόχο η λύση που θα προκύπτει να είναι συμβατή με τους φυσικούς περιορισμούς, η περιγραφή αυτών γίνεται παρακάτω:



Σχήμα 2.8 α) απορροφητικά τοιχώματα, β) ανακλαστικά τοιχώματα, γ) αόρατα τοιχώματα [40]

i. **Απορροφητικά τοιχώματα:** όταν ένα σωματίδιο χτυπήσει το όριο του χώρου λύσεων σε μια από τις διαστάσεις του, η ταχύτητα του στη διεύθυνση αυτή μηδενίζεται ακαριαία και το σωματίδιο θα κινηθεί τελικά προς την διεύθυνση που βρίσκονται επιτρεπτές θέσεις στον χώρο λύσεων. Στην περίπτωση αυτή τα περιοριστικά τοιχώματα απορροφούν την ενέργεια του σωματιδίων που προσπαθούν να αποδράσουν από τον χώρο λύσεων.
ii. **Ανακλαστικά τοιχώματα**: όταν ένα σωματίδιο χτυπήσει το όριο του χώρου λύσεων σε μια από τις διαστάσεις του, το πρόσημο της ταχύτητας του στην διάσταση αυτή αλλάζει και το σωματίδιο ανακλάται πίσω προς τον χώρο λύσεων.
iii. **Αόρατα τοιχώματα**: τα σωματίδια επιτρέπεται να κινούνται χωρίς κανένα φυσικό περιορισμό. Παρόλα αυτά, τα σωματίδια που κινούνται εκτός του επιτρεπόμενου πεδίου δεν αξιολογούνται για την συνάρτηση κόστους. Το κίνητρο πίσω από αυτή την τεχνική είναι να μειωθεί ο υπολογιστικός χρόνος [40].

<u>Πλήρως Ενημερωμένος Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων</u>

Σε αυτή την εκδοχή του αλγορίθμου (full informed particle swarm (fips)) όλα τα γειτονικά σωματίδια επηρεάζουν την ταχύτητα του σωματιδίου και αποτελεί μια κανονικοποιημένη μορφή του αλγόριθμου Βελτιστοποίησης Σμήνους Σωματιδίων. Στην απόδοση του αλγορίθμου έχει σημαντικό ρόλο η τοπολογία των γειτονιών. Οι ταχύτητες των σωματιδίων δίνονται από τον παρακάτω τύπο:

$$v_{ij}(t+1) = x(v_{ij}(t) + \frac{1}{K_i}\sum_{n=1}^{K_i} cr_1(p_{nij} - x_{ij}(t))) \quad (2.28)$$

Όπου όπου το K_i είναι ο αριθμός των γειτονικών σωματιδίων του σωματιδίου i και το p_{nij} είναι η θέση του n−οστού γειτονικού σωματιδίου του i [36].

Ενοποιημένος Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων

Αυτή η εκδοχή (unified PSO) συνδυάζει τις ιδιότητες των εκδοχών local και global του αλγορίθμου PSO. Έστω $G(t+1)$ η ταχύτητα $v_{ij}(t+1)$ του σωματιδίου i για τον global PSO και $L(t+1)$ η ταχύτητα για τον local. Με τον συνδυασμό των δυο εξισώσεων η ταχύτητα $v_{ij}(t+1)$ υπολογίζεται σύμφωνα με τη σχέση:

$$v_{ij}(t+1) = (1-u)\cdot L(t+1) + u\cdot G(t+1) \quad (2.29)$$

Ο παράγοντας u ονομάζεται *unification factor*. Αυτός ο παράγοντας καθορίζει την επίδραση μεταξύ της τοπικής και της παγκόσμιας αναζήτησης [37].

Δυαδικός Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων

Για προβλήματα δυαδικής φύσης έχει προταθεί η εκδοχή του δυαδικού αλγόριθμου (Binary PSO). Κάθε λύση του προβλήματος, δηλαδή η θέση του σωματιδίου στο χώρο αναζήτησης αναπαριστάτε από μια ν-διάστατη δυαδική ακολουθία (binary string) όπου *ν* είναι ο αριθμός των παραμέτρων της λύσης του προβλήματος. Σε κάθε επανάληψη του αλγορίθμου οι ενημερώσεις για κάθε bit ξεχωριστά γίνονται πιθανότητά. Η πιθανότητα να είναι 1 ή 0 είναι συνάρτηση i) της τιμής του στην τρέχουσα θέση, ii) της ταχύτητας που έχει, iii) της τιμή του συγκεκριμένου bit της καλύτερης προσωπικής δυαδικής ακολουθίας, της τιμή του συγκεκριμένου bit της καλύτερης δυαδικής ακολουθίας όλου του σμήνους ή της γειτονιάς του, και δίνεται από τη σχέση:

Όπου,

$$P(x_{id}(t)=1)\ είναι\ f\left(x_{id}(t), v_{id}(t-1), p_{id}, p_{gd}\right) \quad (2.30)$$

Όπου $P(x_{id}(t)=1)$ είναι η πιθανότητα για ένα σωματίδιο να διαλέξει 1 για το bit της d-οστής θέσης της ακολουθίας των bits, $x_{id}(t)$ είναι η τρέχουσα κατάσταση (0 ή 1) του d-οστού bit, $v_{id}(t-1)$ είναι το μέτρο της τρέχουσας πιθανότητας της bit ακολουθίας να διαλέξει 1, p_{id} είναι η τιμή (0 ή 1) του d-οστού bit της καλύτερης προσωπικής τιμής του σωματιδίου i, p_{gd} είναι η τιμή (0 ή 1) του d-οστού bit της καλύτερης συνολικής τιμής για τη γειτονιά του σωματιδίου i (global ή local).

Η ταχύτητα για κάθε bit υπολογίζεται από τη σχέση (2.30) και στη συνέχεια υπολογίζεται η σιγμοειδής συνάρτηση αυτής, (σχέση 2.31). Αν τιμή της είναι μεγαλύτερη από ένα τυχαίο αριθμό *τ* που έχει τιμή από 0 έως 1 τότε το bit παίρνει τιμή 1 αλλιώς την τιμή 0 [38].

$$v_{id}(t) = v_{id}(t-1) + c_1 r_1(p_{id}(t-1) - x_{id}(t-1)) + c_2 r_2\big(p_{gd}(t-1) - x_{id}(t-1)\big) \quad (2.31)$$

$$f\big(v_{id}(t)\big) = \frac{1}{1+e^{-v_{id}(t)}} \quad (2.32)$$

$$x_{id} = \begin{cases} 1, & \varepsilon\acute{\alpha}\nu\ \tau < s\big(v_{id}(t)\big) \\ 0, & \varepsilon\acute{\alpha}\nu\ \tau \geq s\big(v_{id}(t)\big) \end{cases} \quad (2.33)$$

Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων λογικής άλγεβρας

Μια άλλη εκδοχή της δυαδικής μορφής του αλγορίθμου PSO είναι ο Boolean PSO, όπου η θέση και η ταχύτητα κάθε σωματίδιου αντιπροσωπεύονται από μια νιοστή δυαδική ακολουθία όπου *ν* είναι οι παράμετροι του προβλήματος. Για τη μεταβολή της θέσης και της ταχύτητας ισχύουν οι γνωστές σχέσεις της κλασικού αλγορίθμου PSO με τη διαφορά ότι οι πράξεις είναι λογικής άλγεβρας (*Boolean*).

$$v_{ij}(t+1) = w \cdot v_{ij} + c_1 r_{1j}(t) \cdot \big[y_{ij}(t) \oplus x_{ij}(t)\big] + c_2 r_{2j}(t) \cdot \big[\hat{y}_{kj}(t) \oplus x_{ij}(t)\big] \quad (2.34)$$

$$x_{ij}(t+1) = v_{ij}(t+1) \oplus x_{ij}(t) \quad (2.35)$$

Όπου $(\oplus)$ η λογική πράξη XOR, όπου $(\cdot)$ η λογική πράξη AND και όπου $(+)$ η λογική πράξη OR [39].

2.8 Εφαρμογή του αλγορίθμου Βελτιστοποίησης Σμήνους Σωματιδίων στη ρομποτική

Οι αλγόριθμοι PSO έχουν εφαρμοστεί σε πλήθος ρομποτικών εφαρμογών. Συγκεκριμένα έχουν χρησιμοποιηθεί για βελτιστοποίηση της κίνησης ρομποτικών βραχιόνων, έντροχων οχημάτων, εναέριων και υποβρύχιων ρομπότ, για τη βελτιστοποίηση του βαδίσματος ανθρωποειδών ρομπότ καθώς και για τη βελτιστοποίηση ελεγκτών PID, ασαφούς λογικής, νευρωνικών δικτύων κ.α.

2.8.1 Αλγόριθμος Βελτιστοποίησης Σμήνους Σωματιδίων και Έλεγχος

Οι Mahmud Iwan Solihin *et. al*. [41], παρουσίασαν το σχεδιασμό ενός PID ελεγκτή για DC κινητήρα με τη βοήθεια του αλγορίθμου PSO. Η απόδοση του PID ελεγκτή καθορίζεται από την ρύθμιση του αναλογικού, του ολοκληρωτικού και του διαφορικού κέρδους (K_p, K_i, K_d). Στην περίπτωση αυτή τα κέρδη του ελεγκτή αποτελούν τις 3 διαστάσεις του σωματιδίου και η θέση του στο χώρο αναζήτησης καθορίζεται από την συνάρτηση καταλληλότητας η οποία είναι το ολοκλήρωμα του απόλυτου σφάλματος με χρονικό βάρος:

$$ITAE = \int_0^\infty t|e(t)|dt \quad (2.36)$$

όπου $e(t)$ το το σφάλμα, τη χρονική στιγμή t

Οι Ching-Chang Wong *et. al*. [42], πρότειναν τη βελτιστοποίηση του ελεγκτή κίνησης ενός ρομποτικού οχήματος με τη χρήση του αλγορίθμου PSO. Ο έλεγχος του οχήματος αρχικά, γίνεται από δύο ασαφείς ελεγκτές όπου ο πρώτος λαμβάνει υπόψη την απόσταση, ενώ ο δεύτερος το σφάλμα κατεύθυνσης του οχήματος από τον επιθυμητό στόχο. Στη συνέχεια ένας αλγόριθμος PSO προσαρμοσμένος για το σχεδιασμό ασαφών συστημάτων επιλέγει αυτόματα τις κατάλληλες συναρτήσεις συμμετοχής για τις εισόδους και τις εξόδους των δύο ασαφών ελεγκτών.

Τα τεχνητά νευρωνικά δίκτυα (ΤΝΔ), ειδικά όταν έχουν συνδέσεις ανάδρασης, είναι σε θέση να αναπαράγουν πολύπλοκα δυναμικά μοντέλα, και ως εκ τούτου χρησιμοποιούνται σε εφαρμογές ελέγχου. Ο σχεδιασμός ενός δικτύου μπορεί να είναι ένα δύσκολο έργο, καθώς όσο πιο σύνθετο είναι το επιθυμητό δυναμικό μοντέλο, τόσο πιο δύσκολη είναι η σχεδίαση του δικτύου που απαιτείται. Πολλοί ερευνητές προσπάθησαν να αυτοματοποιήσουν τη διαδικασία σχεδιασμού ΤΝΔ με χρήση προγραμμάτων ηλεκτρονικών υπολογιστών. Το πρόβλημα της εύρεσης του καλύτερου συνόλου παραμέτρων για ένα δίκτυο ώστε να λύσει ένα πρόβλημα μπορεί να θεωρηθεί ως μια αναζήτηση και βελτιστοποίηση του προβλήματος. Οι παράμετροι μπορεί να είναι: τα βάρη των συνάψεων, η τοπολογία του δικτύου, οι συναρτήσεις μεταφοράς ή ακόμη και συνδυασμός αυτών. Ο Mahmood Rahmani *et.al.* [43] χρησιμοποιεί έναν αλγόριθμο PSO για την βελτιστοποίηση των παραμέτρων ενός νευρωνικού ελεγκτή για ένα ρομποτικό όχημα που προσπαθεί να πάει σε συγκεκριμένο στόχο αποφεύγοντας εμπόδια με τη χρήση αισθητήρα υπέρυθρών.

Οι Jim Pugh *et al.* [44], ερευνήσανε τη χρήση του PSO σε προβλήματα με θορυβώδες περιβάλλον και στη μη επιβλεπόμενη μάθηση ρομποτικών οχημάτων για την αποφυγή εμποδίων με τη χρήση αισθητήρων. Ο αλγόριθμος PSO που προτείνεται μεταβάλλει τα βάρη του νευρωνικού ελεγκτή του οχήματος, η απόδοση της συνάρτησης αξιολόγησης

βασίζεται πάνω στο τρόπο αποφυγής των εμποδίων, στο πόσο γρήγορα κινήθηκε το όχημα και στο πόσο κοντά βρίσκεται στα εμπόδια.

2.8.2 Διάφορες εφαρμογές με τον Αλγόριθμο Βελτιστοποίησης Σμήνους Σωματιδίων

Οι Thomas Rofer *et al.* [45], εφάρμοσαν έναν αλγόριθμο PSO για τη βελτιστοποίηση της βάδισης σε ένα ανθρωποειδές ρομπότ. Το δίποδο βάδισμα διαμορφώνεται από μια σειρά παραμετροποιήσιμων τροχιών. Για την επίτευξη της βάδισης σε όλες τις κατευθύνσεις διαφορετικά σύνολα παραμέτρων βάδισης βελτιστοποιούνται για διάφορες κατευθύνσεις.

Προκειμένου να προσδιοριστεί η θέση και ο προσανατολισμός ενός αντικειμένου ως προς το πλαίσιο του καρπού για ρομποτικό βραχίονα, θα πρέπει να προσδιοριστεί ο ομογενής μετασχηματισμός του συστήματος ματιού-χεριού, η οποία περιγράφεται τη στροφή και τη μετατόπιση του πλαισίου του καρπού ως προς το πλαίσιο της κάμερας. Οι Dong-Yuan Ge Hong Ji *et al.* [46], προτείνανε μια νέα προσέγγιση η οποία ενσωματώνει νευρωνικό δίκτυο με τον αλγόριθμο βελτιστοποίησης PSO σε συνδυασμό με διασταύρωση και μετάλλαξη για τη βαθμονόμηση του αισθητήρα του βραχίονα. Αρχικά φτιάχνεται ένα νευρωνικό δίκτυο με είσοδο έναν πίνακα στροφής όπου τα βάρη είναι οι συντελεστές που αντιστοιχούν στις στροφές κατά τους τρεις άξονες x, y, z. Στη συνέχεια, ο αλγόριθμος βελτιστοποίησης σωματιδίων σμήνους ενσωματώνεται στο πρόγραμμα επίλυσης του προβλήματος, όπου οι παράγοντες των βαρών αδράνειας και η πιθανότητα μετάλλαξης αυτο-προσαρμόζονται ανάλογα με την τροχιά κίνησης των σωματιδίων. Όταν το κριτήριο τερματισμού ικανοποιείται, ο πίνακας στροφής λαμβάνεται από τους συντελεστές των βαρών του νευρωνικού δικτύου. Στη συνέχεια υπολογίζεται το διάνυσμα μετατόπισης και έτσι επιτυγχάνεται η θέση και ο προσανατολισμός του πλαισίου του καρπού του βραχίονα σε σχέση με το πλαίσιο της κάμερας.

Ένας άλλος τομέας της ρομποτικής που έχει βρει εφαρμογή ο αλγόριθμος PSO είναι η βελτιστοποίηση της διαδικασίας συναρμολόγησης κάποιου προϊόντος. Η βέλτιστη εφικτή ρομποτική ακολουθία συναρμολόγησης οδηγεί σε αποτελεσματική διαδικασία κατασκευής με την ελαχιστοποίηση του κόστους συναρμολόγησης. Το κόστος συναρμολόγησης βασίζεται στην ενέργεια που απαιτείται για τη συναρμολόγηση των εξαρτημάτων χωρίς να υπάρχουν συγκρούσεις κατά τη διαδρομή των ρομπότ, καθώς και των αλλαγών κατεύθυνσης κατά τη διάρκεια των εργασιών συναρμολόγησης. Έτσι, ο προσδιορισμός μιας εφικτής ακολουθίας συναρμολόγησης με ελάχιστο κόστος συναρμολόγησης είναι ζωτικής σημασίας για τον βιομηχανικό κλάδο. Οι M. V. A. Raju. Bahubalendruni *et al.* [47], χρησιμοποίησαν τη μεθοδολογία PSO σε συνδυασμό με το περιβάλλον CAD για τη βελτίωση του κόστους μιας ρομποτικής ακολουθίας συναρμολόγησης. Σε αυτή τη μεθοδολογία, κάθε εξάρτημα του συναρμολογημένου

προϊόντος θεωρείται ως το σωματίδιο και μια διαδικασία μετάλλαξης εκτελείται για να δημιουργήσει μια νέα ακολουθία συναρμολόγησης για κάθε επανάληψη. Για να δημιουργηθεί η βέλτιστη ακολουθία συναρμολόγησης, μια συνάρτηση καταλληλότητας δημιουργείται, η οποία βασίζεται στην καταναλισκόμενη ενέργεια και στις αλλαγές κατεύθυνσης του ρομπότ που συνδέονται με την αλληλουχία συναρμολόγησης. Η αλληλουχία η οποία έχει την καλύτερη τιμή αξιολόγησης αντιμετωπίζεται ως η βέλτιστη ρομποτικό ακολουθία συναρμολόγησης.

2.8.3 Εφαρμογές του Αλγορίθμου Βελτιστοποίησης Σμήνους Σωματιδίων σε ρομποτικά οχήματα

Οι Dayal R Parhi *et al* [48], ανέπτυξαν ένα σύστημα πλοήγησης ρομποτικού οχήματος διαφορικής κίνησης βασισμένο στην ευφυΐα σμήνους. Μελετήθηκαν οι κινηματικοί περιορισμοί και αναπτύχθηκαν κινηματικά μοντέλα για τον υπολογισμό της ταχύτητας του ρομπότ. Μία νέα συνάρτηση καταλληλότητας έχει εισαχθεί για την προτεινόμενη μεθοδολογία που είναι συνάρτηση των θέσεων προς το πλησιέστερο εμπόδιο και τον προορισμό του ρομποτικού οχήματος.

Στη σημερινή εποχή υπάρχουν απαιτήσεις για εφαρμογές που αφορούν ρομποτικά οχήματα σε κλειστούς χώρους όπως τις κατοικίες και τα νοσοκομεία Για το σκοπό αυτό οι Yangmin Li *et al* [49], πρότειναν μια νέα μέθοδο σχεδιασμού διαδρομής με χρήση βελτιστοποίησης σμήνους σωματιδίων (PSO). Η μέθοδος αυτή μπορεί να παράγει ομαλή πορεία για να κάνει το ρομπότ να φτάσει στον προορισμό, χωρίς να αγγίξει τα εμπόδια κατά τη διαδρομή. Η ανίχνευση του αγνώστου περιβάλλοντος γίνεται χωρίς να χρειάζεται κάποιος χάρτης.

Οι Maryam Yarmohamadi *et al* [50], προτείνανε μια μέθοδο αλγορίθμου PSO για τον σχεδιασμό μονοπατιού (path planning) σε δυναμικά περιβάλλοντα με κινητά εμπόδια και στόχους. Ο PSO μπορεί εύκολα να τροποποιηθεί ώστε να επιτραπεί στο ρομποτικό όχημα να εξετάσει με επιτυχία τους κινδύνους ενός δυναμικού περιβάλλοντος και την αποφυγή εμποδίων ώστε να συνεχίσει μια αποτελεσματική πορεία προς στο στόχο τροποποιώντας τη διαδρομή από την αρχική θέση.

Οι Qirong Tang *et al* [51], ασχολήθηκαν με το πρόβλημα της συνεργατικής κίνησης ενός σμήνους ρομποτικών οχημάτων με σκοπό την αναζήτηση ενός στόχου σε ένα πολύπλοκο περιβάλλον. Η λύση είναι εμπνευσμένη από τον αλγόριθμο PSO σε συνδυασμό με την δυναμική του συστήματος που περιλαμβάνει την εξέταση των φυσικών ιδιοτήτων των οχημάτων όπως μάζα, αδράνεια, δύναμη, επιτάχυνση κ.λπ. Ολόκληρο το ρομποτικό σμήνος καθοδηγείται κυρίως από τη μέθοδο PSO και από μια ανεξάρτητη μονάδα αποφυγής εμποδίων που ενεργοποιείται όταν τα ρομπότ αντιμετωπίζουν τυχόν συγκρούσεις κατά τη διάρκεια των αποστολών. Ένα τεχνητό

σμήνος καλείται να εκτελέσει ασκήσεις αναζήτησης, κάθε μέλος του συστήματος μπορεί να αλληλεπιδράσει με τους γείτονές του ή το περιβάλλον. Τα αποτελέσματα των προσομοιώσεων για τον έλεγχο της στρατηγικής κίνησης δείχνουν ότι η μέθοδος αυτή δημιουργεί την επιθυμητή συμπεριφορά σε ικανοποιητικό επίπεδο.

Οι Lisa L. Smith *et al* [52], ανέπτυξαν έναν αλγόριθμο αναζήτησης στόχου όπου ένας τυχαίος αριθμός από ρομπότ-σωματίδια κατανέμονται σε μια συγκεκριμένη περιοχή και «πετάνε» μέσα από το χώρο αναζήτησης σε μία νέα θέση που υπολογίζεται για κάθε σωματίδιο ανά επανάληψη. Οι συντεταγμένες του στόχου είναι γνωστές και τα ρομπότ χρησιμοποιούν μια συνάρτηση καταλληλότητας, που βασίζεται στην Ευκλείδεια απόσταση των ρομπότ από τον στόχο. Οι συντεταγμένες του στόχου καθώς και των εμποδίων είναι γνωστές και για την αντίληψη τους χρησιμοποιούνται αισθητήρες. Εάν η επόμενη υποψήφια θέση ενός σωματιδίου βρίσκεται εντός του χώρου του εμποδίου, το σωματίδιο θα μετακινηθεί σε μια νέα θέση που υπολογίζεται βάση μιας συνάρτηση τόξου και των συντεταγμένων της πλησιέστερης γωνίας του εμποδίου σε σχέση με το όχημα, αντί να κατευθυνθεί πάνω στο εμπόδιο και προκύψει σύγκρουση. Καθώς τα ρομπότ - σωματίδια ψάχνουν την περιοχή με τη βοήθεια του αλγορίθμου PSO, χρησιμοποιούν την προσωπικά καλύτερα θέση τους p_{id}, και την καθολικά καλύτερη θέση p_{gd}, για να διατηρηθούν σε μια διαδρομή που τα οδηγεί στο στόχο. Βασικές σχέσεις γεωμετρίας και το Πυθαγόρειο θεώρημα χρησιμοποιούνται για να γίνει ανάλυση της τρέχουσας θέσης ενός σωματιδίου σε σχέση με το εμπόδιο κατά τις προσομοιώσεις. Σε πραγματική δοκιμή, ο σχετικισμός της θέσης του σωματιδίου με τον στόχο ή το εμπόδιο καθορίζονται με τη χρήση δεδομένων από αισθητήρες.

ΚΕΦΑΛΑΙΟ 3

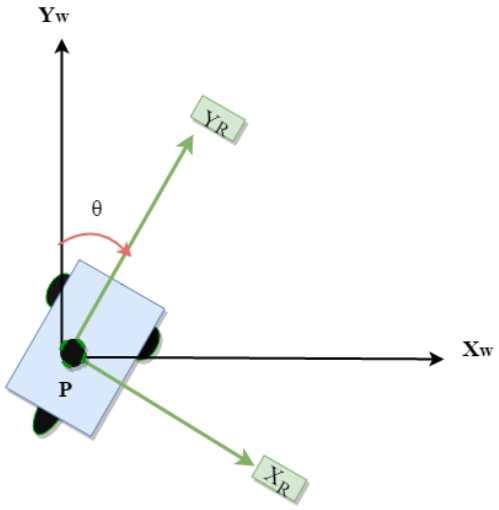
Έλεγχος ρομποτικών οχημάτων με τη βοήθεια ασαφούς λογικής

3.1 Εισαγωγή

Στο κεφάλαιο αυτό παρουσιάζεται το μοντέλο κίνησης ενός διαφορικά οδηγούμενου οχήματος. Στη συνέχεια γίνεται επεξήγηση της ανάπτυξης ενός απλού ασαφούς ελεγκτή για την κίνηση ενός ρομποτικού διαφορικού οχήματος από μια αρχική θέση σε ένα σημείο στόχο. Η πιστοποίηση καλής λειτουργίας γίνεται με τη χρήση προσομοίωσης σε ένα πραγματικό ρομποτικό όχημα τύπου Pioneer.

3.2 Κινηματικό μοντέλο ρομποτικού οχήματος διαφορικής κίνησης

Για να προσδιοριστεί η θέση ενός ρομποτικού οχήματος σε ένα δισδιάστατο επίπεδο, είναι απαραίτητο να προσδιοριστεί μια σχέση μεταξύ του παγκόσμιου (global) πλαισίου αναφοράς του επιπέδου και του σταθερού πλαισίου του ρομπότ, όπως φαίνεται στο σχήμα 3.1.



Σχήμα 3.1 Θέση και προσανατολισμός κινούμενου ρομπότ σε επίπεδο

Για να καθορίσουμε τη θέση που βρίσκεται το όχημα σε σχέση με το παγκόσμιο πλαίσιο αναφοράς O: $\{X_i, Y_i\}$, θεωρούμε ένα σημείο P στο σταθερό πλαίσιο ως σημείο αναφοράς για τη θέση του ρομπότ σε σχέση με το τοπικό πλαίσιο αναφοράς του $\{X_R, Y_R\}$. Η θέση του σημείου P ως προς το παγκόσμιο πλαίσιο αναφοράς μπορεί να καθοριστεί από τις συντεταγμένες x και y, και η γωνιακή διαφορά μεταξύ του παγκόσμιου και του τοπικού πλαισίου αναφοράς δίνεται από τη γωνία θ.

Επομένως η θέση και ο προσανατολισμός του ρομπότ δίνεται από το διάνυσμα:

$$\xi_1 = [x\ y\ \theta]^T \quad (3.1)$$

Τα δύο πλαίσια συσχετίζονται με τον πίνακα στροφής:

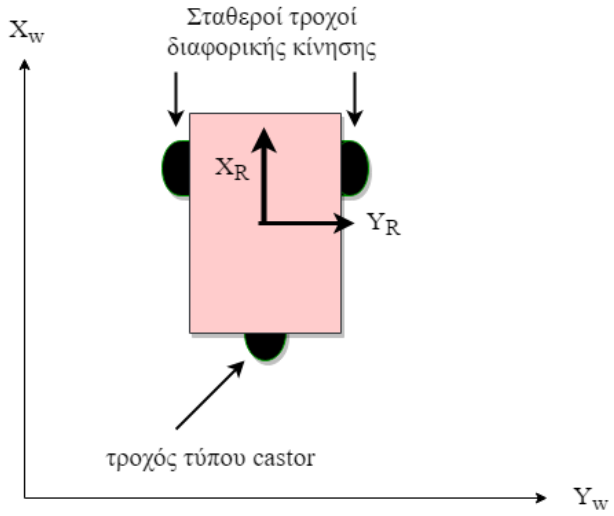
$$R(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \sin\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3.2)$$

Για τη θέση και την ταχύτητα κίνησης του ρομπότ εκφρασμένες στο παγκόσμιο σύστημα, σύμφωνα με τις σχέσεις 3.1 & 3.2 θα ισχύει :

$$\xi_R = R(\theta) \cdot \xi_1 = R(\theta) \cdot [x\ y\ \theta]^T \quad (3.3)$$

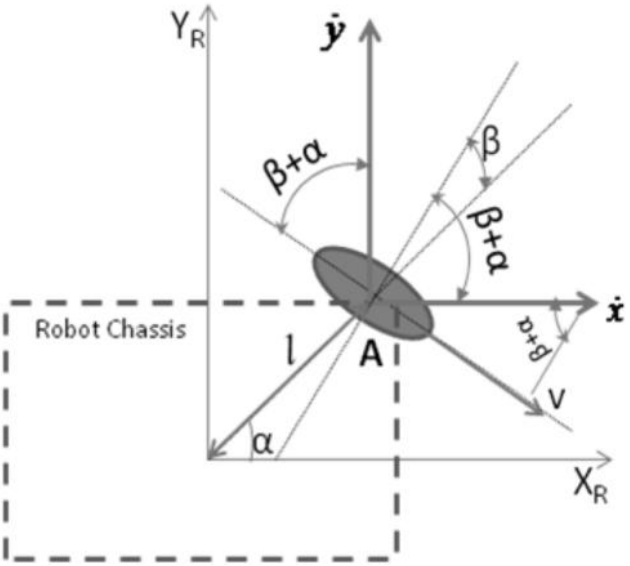
$$\dot{\xi_R} = R(\theta) \cdot \dot{\xi_1} = R(\theta) \cdot \left[\dot{x}\ \dot{y}\ \dot{\theta}\right]^T \quad (3.4)$$

Υπάρχουν διάφοροι τρόποι με τους οποίους μπορεί να κινηθεί ένα έντροχο ρομπότ ανάλογα με τη διαμόρφωση που έχουν οι τροχοί του και εάν έχει κατευθυντήριο τροχό ή όχι. Συγκεκριμένα για την περίπτωση της διαφορικής κίνησης έχουμε δύο κινητήριους σταθερούς τροχούς και έναν τροχό τύπου caster όπως φαίνεται στο σχήμα 3.2, ο σταθερός τροχός δεν έχει κατακόρυφο άξονα περιστροφής για την οδήγηση, η γωνία του ως προς το σταθερό πλαίσιο είναι σταθερή και η κίνησή του περιορίζεται σε εμπρός και πίσω κατά μήκος του επιπέδου του τροχού και σε περιστροφή γύρω από το σημείο επαφής του με το επίπεδο του εδάφους.

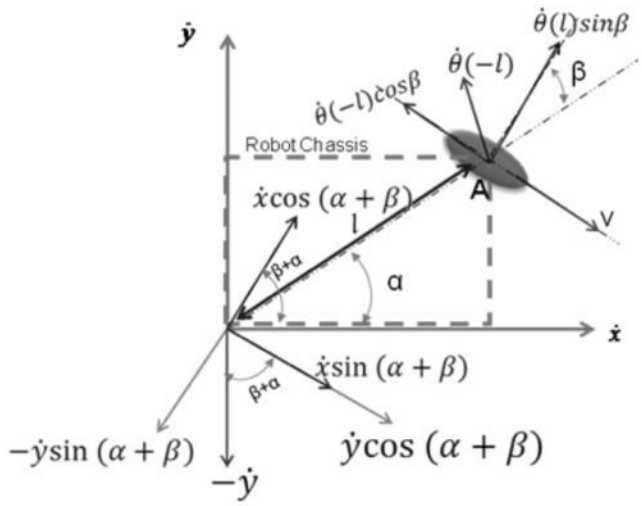


Σχήμα 3.2 Ρομποτικό όχημα με τροχούς διαφορικής κίνησης

Στο σχήμα 3.3 απεικονίζεται ένας σταθερός τροχός και η θέση του είναι εκφρασμένη σε πολικές συντεταγμένες με απόσταση *l* και γωνία *α* από την αρχή αξόνων του πλαισίου αναφοράς του σταθερού πλαισίου του ρομποτικού οχήματος. Η γωνία του επιπέδου του τροχού σε σχέση με το σταθερό πλαίσιο συμβολίζεται με *β*.



Σχήμα 3.3 Οι γωνίες που σχηματίζει ο τροχός με το πλαίσιο συντεταγμένων του σταθερού πλαισίου[53]



Σχήμα 3.4 Ανάλυση των διανυσμάτων ταχυτήτων του τροχού[53]

Ενώ ένα διαφορικό ρομπότ είναι σε κίνηση, δύο περιορισμοί είναι δυνατόν να συμβαίνουν. Ο πρώτος περιορισμός επιβάλλεται από την επαφή κύλισης και ο δεύτερος από την πλευρική ολίσθηση. Σκοπός είναι να αποφευχθεί η πλευρική ολίσθηση, ενώ το ρομπότ είναι σε κίνηση.

Ο περιορισμός κύλισης για αυτόν τον τροχό επιβάλλει ότι όλες οι κινήσεις κατά μήκος της διεύθυνσης του επιπέδου του τροχού πρέπει να συνοδεύονται από την κατάλληλη ποσότητα περιστροφής έτσι ώστε να υπάρχει καθαρή κύλιση στο σημείο επαφής:

$$\dot{x_R}\sin(\alpha+\beta) - \dot{y_R}\cos(\alpha+\beta) - \dot{\theta_R} l\cos(\beta) = \dot{\varphi} r \quad (3.5)$$

Όπου $\dot{\varphi}$, r, η γωνιακή ταχύτητα και η ακτίνα του τροχού. Ο περιορισμός ολίσθησης για αυτόν τον τροχό επιβάλλει ότι η συνιστώσα της κίνησης του τροχού που είναι κάθετη προς το επίπεδο του τροχού πρέπει να είναι μηδέν:

$$\dot{x_R}\sin(\alpha+\beta) - \dot{y_R}\cos(\alpha+\beta) - \dot{\theta_R} l\cos(\beta) = 0 \quad (3.6)$$

Στα διαφορικά κινούμενα ρομπότ για τον αριστερό τροχό ισχύει ότι, $\alpha = 90°$, $\beta = 0°$ και $l = {}^{b}/_{2}$, οπότε :

$$\dot{x_R}\sin(90) - \dot{y_R}\cos(90) - \dot{\theta_R} l\cos(0) = \dot{\varphi_1} r \quad (3.7) \qquad \text{και}$$

$$\dot{x_R}\sin(90) - \dot{y_R}\cos(90) - \dot{\theta_R} l\cos(0) = 0 \quad (3.8)$$

Λύνοντας το σύστημα των δύο εξισώσεων προκύπτει ότι:

$\dot{x_R} - \dot{\theta_R}\, {}^{b}/_{2} = \dot{\varphi_1} r$ (3.9) και $\dot{y_R} = 0$ (3.10)

Για τον δεξιό τροχό ισχύει ότι, $\alpha = -90°$, $\beta = 0°$ και $l = {}^{b}/_{2}$, οπότε :

$\dot{x_R} - \dot{\theta_R}\, {}^{b}/_{2} = \dot{\varphi_2} r$ (3.11) και $\dot{y_R} = 0$ (3.12)

λύνοντας ως προς $\dot{x_R}$ και $\dot{y_R}$ παίρνουμε το κινηματικό μοντέλο του ρομπότ όπου:

$\dot{x_R} = (\dot{\varphi_1} + \dot{\varphi_2})\, {}^{r}/_{2}$ (3.13) $\dot{y_R} = 0$ (3.14) $\dot{\theta_R} = (\dot{\varphi_2} - \dot{\varphi_1})\, {}^{r}/_{b}$ (3.15) [53]

3.3 Δημιουργία ασαφούς ελεγκτή τύπου Mamdani για τον έλεγχο κίνησης από αρχική σε επιθυμητή θέση

Προκειμένου να σχεδιαστεί ένας ασαφής ελεγκτής για τον έλεγχο θέσης ενός ρομποτικού οχήματος διαφορικής οδήγησης, ακολουθήθηκε η προσέγγιση ενός ελεγκτή τύπου Mamdani, όπως αυτή περιγράφηκε στην παράγραφο 3.2. Για τον αρχικό πειραματισμό χρησιμοποιήθηκε ένα προσομοιωμένο μοντέλο του έντροχου ρομποτικού οχήματος Pioneer 3pdx με τα εξής χαρακτηριστικά:

Εικόνα 3.1 Το ρομποτικό όχημα Pioneer 3pdx

Operation
Robot Weight: 9 kg
Operating Payload: 17 kg
Differential Drive Movement
Turn Radius: 0 cm
Swing Radius: 26.7 cm
Max. Forward/Backward Speed: 1.2 m/s
Rotation Speed: 300°/s
Max. Traversable Step: 2.5 cm
Max. Traversable Gap: 5 cm
Max. Traversable Grade: 25%
Traversable Terrain: Indoor, wheelchair accessible
Microcontroller I/O
System Serial 32
digital inputs 8
digital outputs 7
analog inputs 3
serial expansion ports
User Control Panel
MIDI programmable piezo buzzer
Main power indicator
Battery charge indicator
2 AUX power switches
System reset
Motor enable pushbutton

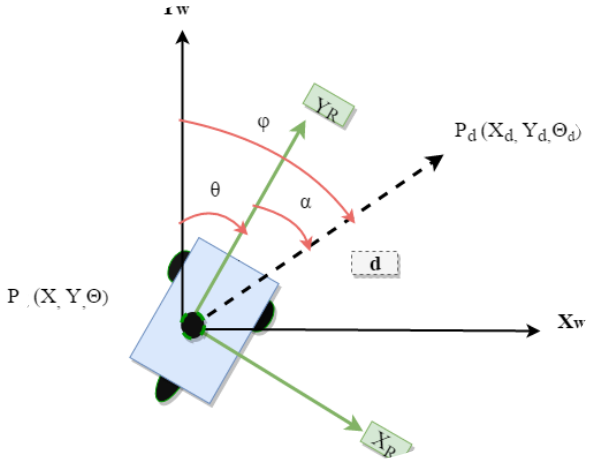
Πίνακας 3.1 Τεχνικά χαρακτηριστικά του Pioneer 3pdx

Ο ασαφής ελεγκτής πλοήγησης οδηγεί το όχημα από μια δεδομένη αρχική θέση σε ένα σταθερό ή κινούμενο σημείο-στόχο. Η απόκριση του ελεγκτή οδηγεί το όχημα είτε προς τα αριστερά, είτε προς τα δεξιά είτε προς τα εμπρός, ανάλογα με τη μεταβλητή εισόδου.

Η βασική λογική πάνω στην οποία στηρίζεται ο ελεγκτής είναι η εξής: Αν η επιθυμητή θέση είναι αριστερά της τρέχουσας, το ρομπότ στρίβει αριστερά, αν η επιθυμητή θέση είναι δεξιά της τρέχουσας στρίβει δεξιά, αν είναι στο κέντρο της τρέχουσας πηγαίνει ευθεία. Για να μπορέσουμε να κατευθύνουμε το όχημα είναι απαραίτητο να υπολογίσουμε το σφάλμα κατεύθυνσης.

Οι δύο τροχοί ενός διαφορικά οδηγούμενου ρομποτικού οχήματος είναι σταθεροί και κάθε τροχός ελέγχεται ανεξάρτητα από τον κινητήρα του, έτσι ώστε η διαδρομή του ελεγχόμενου ρομπότ να καθορίζεται από τις ταχύτητες των δύο κινητήρων. Το αντικείμενο του σχεδιασμού ενός ασαφούς ελεγκτή είναι να καθοριστούν οι ταχύτητες του αριστερού και δεξιού τροχού, έτσι ώστε το ρομπότ να μπορεί να κινείται αποτελεσματικά από την τρέχουσα θέση του (x,y), σε μια επιθυμητή θέση (x_d,y_d).

Ένα σχηματικό διάγραμμα ενός διαφορικά οδηγούμενο ρομποτικού οχήματος φαίνεται στο σχήμα (3.5), όπου X_w-Y_w είναι οι παγκόσμιες (global) συντεταγμένες και X_R-Y_R είναι οι τοπικές (local) συντεταγμένες, όπου η αρχή των αξόνων του τοπικού πλαισίου συντεταγμένων βρίσκεται στο κέντρο p του σταθερού πλαισίου του ρομποτικού οχήματος. Το σώμα του ρομπότ είναι συμμετρικό και το κέντρο μάζας βρίσκεται στο γεωμετρικό κέντρο p του σώματος. Τα σημεία (x,y) αντιπροσωπεύουν τη θέση του γεωμετρικού κέντρου p στο παγκόσμιο σύστημα συντεταγμένων X_w-Y_w και θ είναι η γωνία που δείχνει τον προσανατολισμό του ρομπότ. Η θ γωνία λαμβάνεται δεξιόστροφα από τον Y_w προς X_w άξονα.



Σχήμα 3.5 Θέση και προσανατολισμός ρομποτικού οχήματος στην αρχική και στην επιθυμητή

Το ρομπότ έχει δύο βαθμούς ελευθερίας σε σχέση με τη θέση και τον προσανατολισμό του και περιγράφονται από το διάνυσμα $p(x, y, \theta)$. Στο σχήμα 3.5 τα διανύσματα $p(x,y,\theta)$ και $p_d(x_d,y_d,\theta_d)$ περιγράφουν την τρέχουσα (x, y) και την επιθυμητή θέση (x_d, y_d) του ρομπότ, θ είναι ο τρέχον προσανατολισμός και θ_d είναι ο προσανατολισμός του ρομπότ στην επιθυμητή θέση. Η ευκλείδεια απόσταση μεταξύ της τρέχουσας (x, y) και επιθυμητής θέσης (x_d, y_d) είναι d, όπου

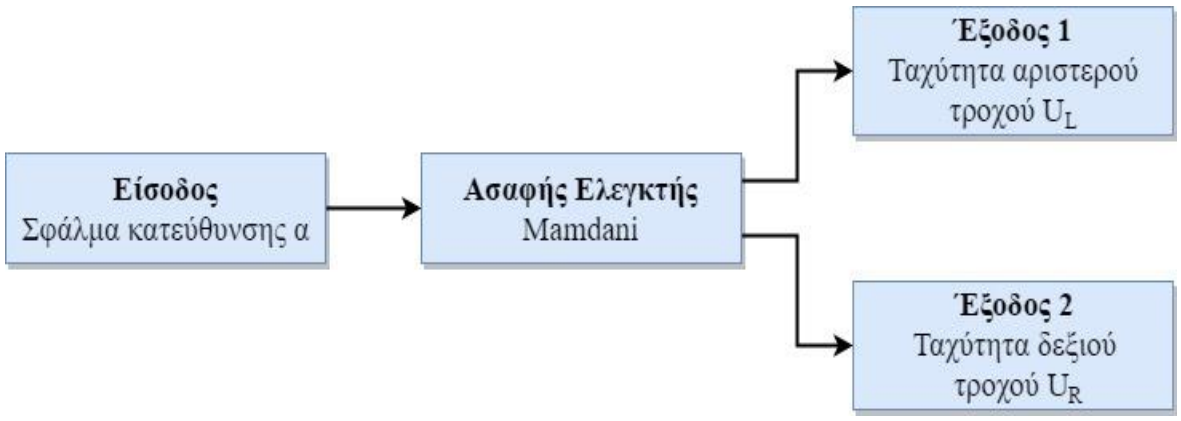
$$d = \sqrt{(x_d - x)^2 + (y_d - y)^2} \quad (3.16)$$

θ είναι η γωνία μεταξύ του Y_w άξονα και του προσανατολισμού του ρομπότ στην τρέχουσα θέση, φ είναι η γωνία μεταξύ του Y_w άξονα και του προσανατολισμού του ρομπότ στην επιθυμητή θέση και υπολογίζεται από τη σχέση:

$$\varphi = \tan^{-1}\frac{y_d - y}{x_d - x} \quad (3.17)$$

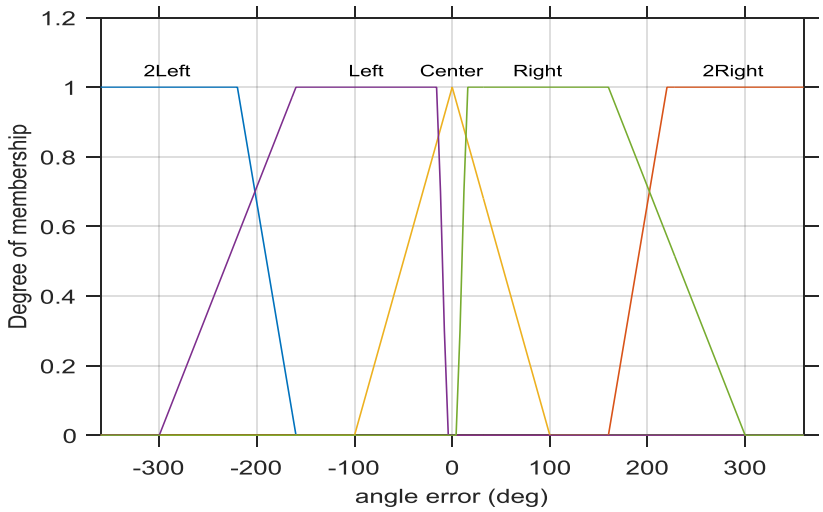
Η γωνία α είναι η γωνία μεταξύ του προσανατολισμού του ρομπότ και μεταξύ της επιθυμητής θέσης και ονομάζεται και σφάλμα της γωνίας κατεύθυνσης και δίνεται από τη σχέση:

$$\alpha = \begin{cases} \varphi - \theta, & \text{εάν} \quad \varphi - 180 < \theta < 180 \\ (\varphi - \theta) - 360, & \text{εάν} \quad -180 < \theta < \varphi - 180 \end{cases} \quad (3.18)$$



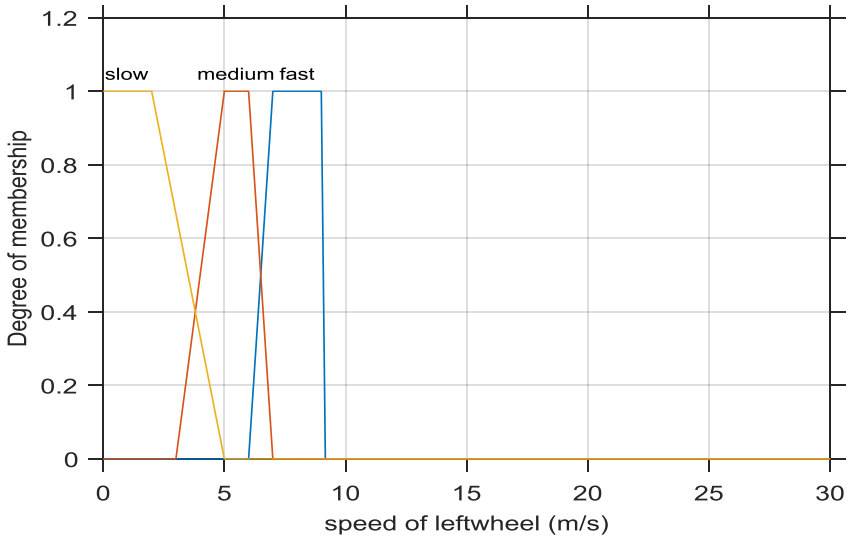
Σχήμα 3.6 Είσοδος και έξοδοι του ασαφούς ελεγκτή

Για να μπορέσουμε να χρησιμοποιήσουμε ως είσοδο τη γωνία σφάλματος, χωρίζουμε την είσοδο σε 5 ασαφή σύνολα που χαρακτηρίζονται από τις συναρτήσεις συμμετοχής τους: (Πολύ αριστερά, αριστερά, κέντρο, δεξιά, πολύ δεξιά), και έχει εύρος από -360 έως 360 μοίρες που ισοδυναμούν σε μία πλήρη περιστροφή είτε αριστερόστροφα είτε δεξιόστροφα, όπως φαίνεται στο σχήμα 3.7



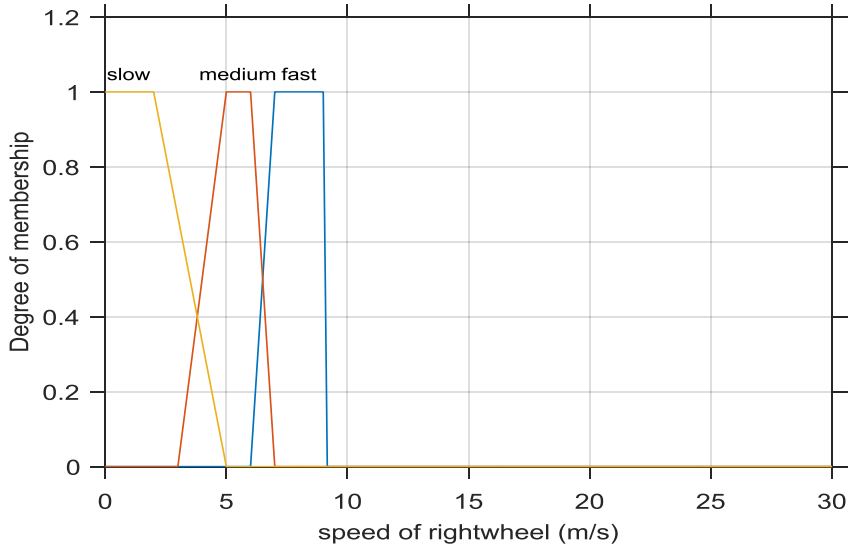
Σχήμα 3.7 Είσοδος «Γωνίας σφάλματος» ασαφούς ελεγκτή

Η πρώτη έξοδος του ελεγκτή που είναι η ταχύτητα του αριστερού τροχού U_L αποτελείται από 3 ασαφή σύνολα που χαρακτηρίζονται από τις συναρτήσεις συμμετοχής τους: (αργά, μέτρια, γρήγορα) με εύρος τιμών από από 0 έως 9 m/s, όπως φαίνεται στο σχήμα 3.8.



Σχήμα 3.8 Έξοδος 1 ταχύτητα αριστερού τροχού U_L

Ομοίως και για την δεύτερη έξοδο του ελεγκτή που είναι η ταχύτητα του δεξιού τροχού.



Σχήμα 3.9 Έξοδος 2 ταχύτητα δεξιού τροχού

Στη συνέχεια δημιουργήσαμε τους λεκτικούς κανόνες όπως φαίνονται στον πίνακα 3.2

ΛΕΚΤΙΚΟΙ ΚΑΝΟΝΕΣ					
	ΕΙΣΟΔΟΣ 1		**ΕΞΟΔΟΣ 1**		**ΕΞΟΔΟΣ 2**
	Γωνία Σφάλματος		**Ταχύτητα U_L**		**Ταχύτητα U_R**
Εάν	Πολύ αριστερά	**Τότε**	Αργά	**Και**	Γρήγορα
	Πολύ δεξιά		Γρήγορα		Αργά
	Κέντρο		Γρήγορα		Γρήγορα
	Αριστερά		Αργά		Μέτρια
	Δεξιά		Μέτρια		Αργά

Πίνακας 3.2 Λεκτικοί κανόνες ασαφούς ελεγκτή

Παρακάτω στα διαγράμματα του σχήματος 3.10, παρουσιάζεται πως αποσαφοποιούνται οι τιμές, ενεργοποιείται κάθε κανόνας και δημιουργούνται τα αντίστοιχα α-cuts στις εξόδους κάθε κανόνα. Τα 5 ασαφή συμπεράσματα συναθροίζονται σε ένα ασαφές συμπέρασμα με την εφαρμογή του τελεστή *max* και στη συνέχεια το ασαφές αυτό σύνολο αποσαφοποιείται με την εφαρμογή της μεθόδου centroid και προκύπτει η ακριβής τελική τιμή της κάθε εξόδου, που είναι αποτέλεσμα των 5 κανόνων.

Αν A_1 είναι το ασαφές σύνολο που που αντιστοιχεί στην είσοδο «γωνία σφάλματος» (angleError), U_{1i} και U_{2i} τα ασαφή σύνολα για την ταχύτητα του αριστερού τροχού και του δεξιού τροχού αντίστοιχα για τον i κανόνα και $\dot{U}_{1i}$ και $\dot{U}_{2i}$ οι α-cut τομές αυτών τότε ο βαθμός εκπλήρωσης των κανόνων είναι:

$$\alpha_1 = \{\mu_{A_1}(-200)\} = 0.67 \qquad \alpha_2 = \{\mu_{A_1}(-200)\} = 0.71$$
$$\alpha_3 = \{\mu_{A_1}(-200)\} = 0 \qquad \alpha_4 = \{\mu_{A_1}(-200)\} = 0$$
$$\alpha_5 = \{\mu_{A_1}(-200)\} = 0$$

Η έξοδος κάθε κανόνα για κάθε μία από τις εξόδους παράγεται ως εξής (τελεστής συνεπαγωγής)

$$\dot{U}_{11} = \min\{\alpha_1, \mu_{U_{11}}(x)\} \qquad \dot{U}_{21} = \min\{\alpha_1, \mu_{U_{21}}(x)\}$$
$$\dot{U}_{12} = \min\{\alpha_2, \mu_{U_{12}}(x)\} \qquad \dot{U}_{22} = \min\{\alpha_2, \mu_{U_{22}}(x)\}$$
$$\dot{U}_{13} = \min\{\alpha_3, \mu_{U_{13}}(x)\} \qquad \dot{U}_{23} = \min\{\alpha_3, \mu_{U_{23}}(x)\}$$
$$\dot{U}_{14} = \min\{\alpha_4, \mu_{U_{14}}(x)\} \qquad \dot{U}_{24} = \min\{\alpha_4, \mu_{U_{24}}(x)\}$$
$$\dot{U}_{15} = \min\{\alpha_5, \mu_{U_{15}}(x)\} \qquad \dot{U}_{25} = \min\{\alpha_5, \mu_{U_{25}}(x)\}$$

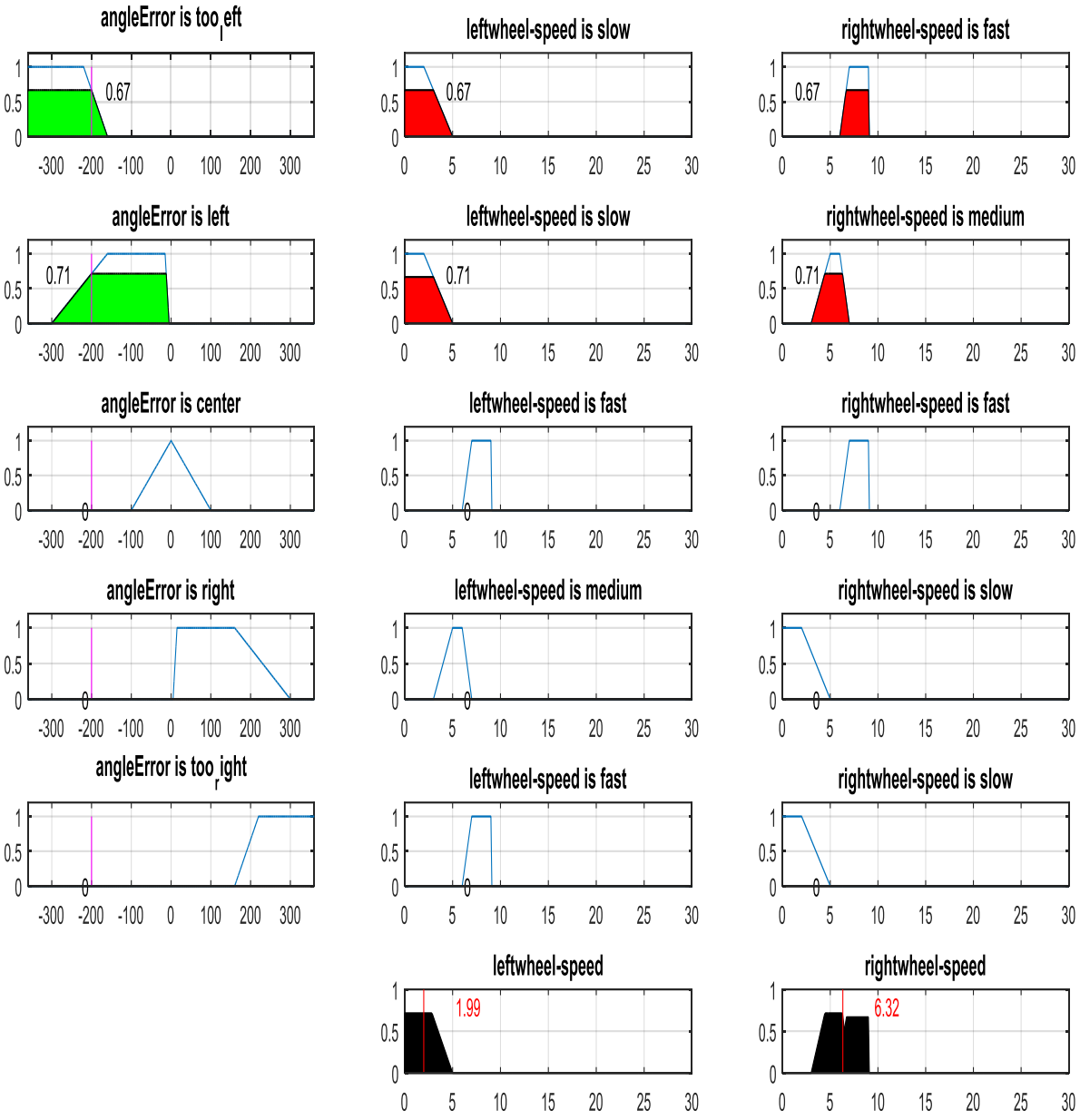
Τα αποτελέσματα των πέντε κανόνων θα είναι

$$U_1 = \dot{U}_{11} \cup \dot{U}_{12} \cup \dot{U}_{13} \cup \dot{U}_{14} \cup \dot{U}_{15}$$

$$U_2 = \dot{U}_{21} \cup 2_{22} \cup \dot{U}_{23} \cup \dot{U}_{24} \cup \dot{U}_{25}$$

και θα υπολογίζονται με τον τελεστή *max*

$$\mu_{U_1}(x) = max\{\mu_{\dot{U}_{11}}(x), \mu_{\dot{U}_{12}}(x), \mu_{\dot{U}_{13}}, (x), \mu_{\dot{U}_{14}}(x), \mu_{\dot{U}_{15}}(x)\}$$
$$\mu_{U_2}(x) = max\{\mu_{\dot{U}_{21}}(x), \mu_{\dot{U}_{22}}(x), \mu_{\dot{U}_{23}}, (x), \mu_{\dot{U}_{24}}(x), \mu_{\dot{U}_{25}}(x)\}$$



Σχήμα 3.10 Ενεργοποίηση των αντίστοιχων κανόνων για τιμές εισόδου angleError=-200 μοίρες, συνάθροιση των συμπερασμάτων τους και αποασαφοποίηση

Η υλοποίηση του ελεγκτή έγινε με τη βοήθεια του fuzzy toolbox στο MATLAB και η η προσομοίωση στο V-REP.

3.5 Τρόπος διασύνδεσης του προσομοιωτή V-REP με το Matlab

Το V-REP αποτελεί έναν ισχυρό προσομοιωτή ρομποτικών οχημάτων, διαθέτει αρκετά ευέλικτες μονάδες υπολογισμού (αντίστροφη κινηματική, φυσική-δυναμική, ανιχνεύσεις σύγκρουσης, υπολογισμούς ελάχιστης απόστασης, σχεδιασμό διαδρομή, κλπ), και διάφορους μηχανισμούς επέκτασης (plug-ins). Επίσης προσφέρει μια πληθώρα συναρτήσεων που μπορούν εύκολα να ενσωματωθούν και να συνδυαστούν με άλλα λογισμικά. Το V-REP είναι ικανό για γρήγορη προτυποποίηση και την επαλήθευση, την εξ αποστάσεως παρακολούθηση, τη γρήγορη ανάπτυξη αλγορίθμων της ρομποτικής που σχετίζεται με την εκπαίδευση, και την προσομοίωση των συστημάτων αυτοματισμού εργοστασίων. Επίσης διαθέτει έναν μηχανισμό για να επιτρέπεται η εύκολη πρόσβαση στη βιβλιοθήκη V-REP API από εξωτερικές εφαρμογές (π.χ. ρομπότ), και μπορεί να συνδέεται με διάφορες γλώσσες προγραμματισμού όπως Python, Java, Matlab, C++.

Το V-REP μπορεί να συνδυαστεί με το λογισμικό MATLAB το οποίο μπορεί να υλοποιήσει πολύπλοκες διαδικασίες, εκτελώντας πολύπλοκα script, συνδυάζοντας απλότητα στον τρόπο σύνταξης τους και συνδεσιμότητα. Συνδέοντας λοιπόν τα δύο αυτά προγράμματα μεταξύ τους γίνονται όλοι οι μαθηματικοί υπολογισμοί που είναι απαραίτητοι για την προσομοίωσης της κίνησης του οχήματος.

Από το V-REP μπορούμε να επιλέξουμε διάφορα ρομποτικά οχήματα για την εκτέλεση των προσομοιώσεων. Στη περίπτωση των πειραμάτων που εκτελέστηκαν στα πλαίσια της παρούσας εργασίας επιλέχθηκε το όχημα Pioneer 3pdx, καθώς πρόκειται για ένα τυπικό όχημα που κινείται βασιζόμενο στις αρχές της διαφορικής κίνησης. Αυτό επιτρέπει τον έλεγχο της διεύθυνσης του απλά ελέγχοντας την ταχύτητα στους δύο κινητήρες. Επίσης διαθέτει αρκετούς αισθητήρες με τους οποίους μπορεί να ελέγχει το περιβάλλον γύρω του για εμπόδια, όπως αισθητήρες υπερήχων, υπέρυθρών, laser scanner και οπτικό αισθητήρα εμπρός για αναγνώριση αντικειμένων.

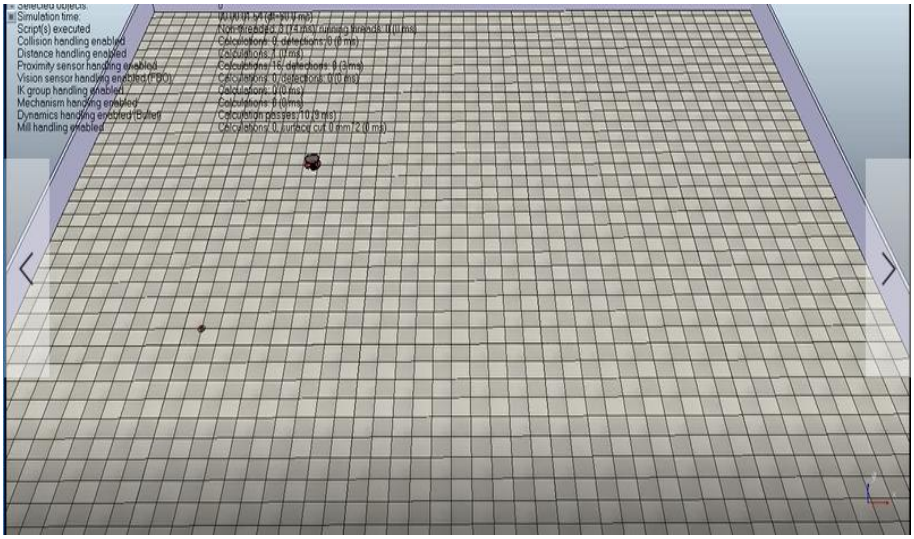
Για τη προσομοίωση στο V-REP αρχικά δημιουργήθηκε ο χώρος κίνησης του ρομποτικού οχήματος που ονομάζεται αρένα, και μπορεί να έχει διάφορα στατικά ή κινούμενα εμπόδια, καθώς και σημεία-στόχους που θα προσεγγίζει το όχημα. Δημιουργήθηκαν διάφορες αρένες με εμπόδια ή χωρίς, ανάλογα με το σκοπό της προσομοίωσης. Στη συνέχεια εισήχθει το Pioneer 3pdx, το οποίο έχει μια λίστα με όλα τα κατασκευαστικά μέρη από τα οποία αποτελείται (σύνδεσμοι, κινητήρες, τροχοί). Έπειτα επιλέχθηκαν τα αισθητήρια που θα φέρει στις προσομοιώσεις αποφυγής εμποδίων, και επιλέχθηκαν αισθητήρες υπερήχων με εμβέλεια 1 μέτρο και γωνία απόκλισης 30°.

Για να γίνει η σύζευξη μεταξύ των δυο προγραμμάτων αρχικά πρέπει να γίνει φόρτωση της βιβλιοθήκης remote API, στη συνέχεια γίνεται η αρχικοποίηση των συναρτήσεων για τη συνεργασία V-REP/ MATLAB (initialiseRoboticsToolbox), έπειτα

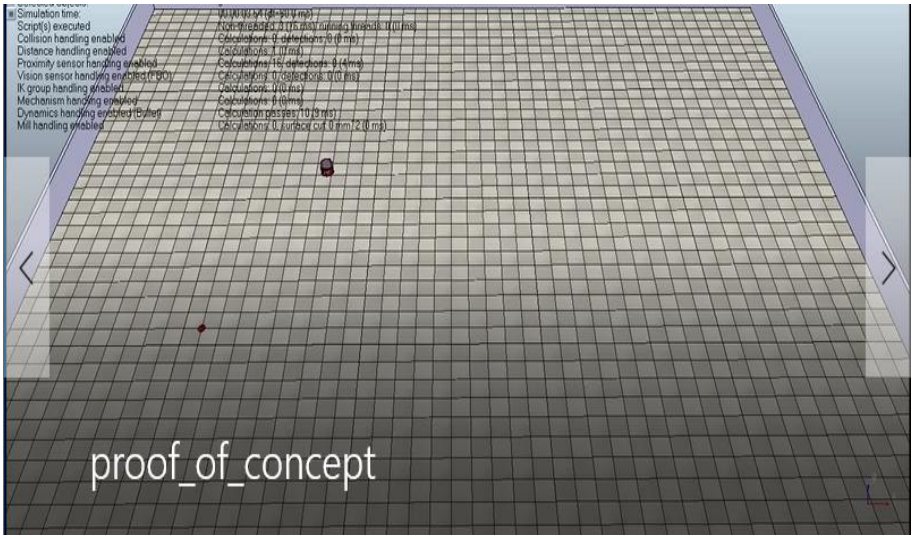
πρέπει να κλείσουν όλες οι ανοιχτές συνδέσεις και μέσω της κατάλληλης θύρας στέλνεται αίτημα για την εκκίνηση της προσομοίωσης. Στη συνέχεια αναπτύχθηκε κατάλληλος κώδικάς ώστε κάθε αντικείμενο που υπάρχει στην αρένα της προσομοίωσης του V-REP να μπορεί να αναγνωριστεί από το MATLAB. Το επόμενο βήμα είναι να γίνει η λήψη των δεδομένων για τη θέση και τον προσανατολισμό του οχήματος, τη θέση του στόχου, τις μετρήσεις των αισθητηρίων, εάν αυτά είναι ενεργοποιημένα. Κατά τη λήψη των δεδομένων πρέπει να εισαχθεί κατάλληλη χρονική καθυστέρηση για την σωστή ενημέρωση των δεδομένων. Στη συνέχεια ακολουθούν οι μαθηματικοί υπολογισμοί όπως της απόστασης από το στόχο, του σφάλματος προσανατολισμού σε σχέση με το στόχο, και ξεκινάει η διαδικασία για τον ασαφή έλεγχο. Αφού εισαχθούν τα δεδομένα που αποτελούν τις εισόδους του ελεγκτή, το MATLAB θα κάνει τους υπολογισμούς θα βγάλει τις τιμές εξόδου και θα στείλει τις τιμές των ταχυτήτων των τροχών στο V-REP ώστε να κινηθεί κατάλληλα το όχημα.

3.6 Επιβεβαίωση του σεναρίου

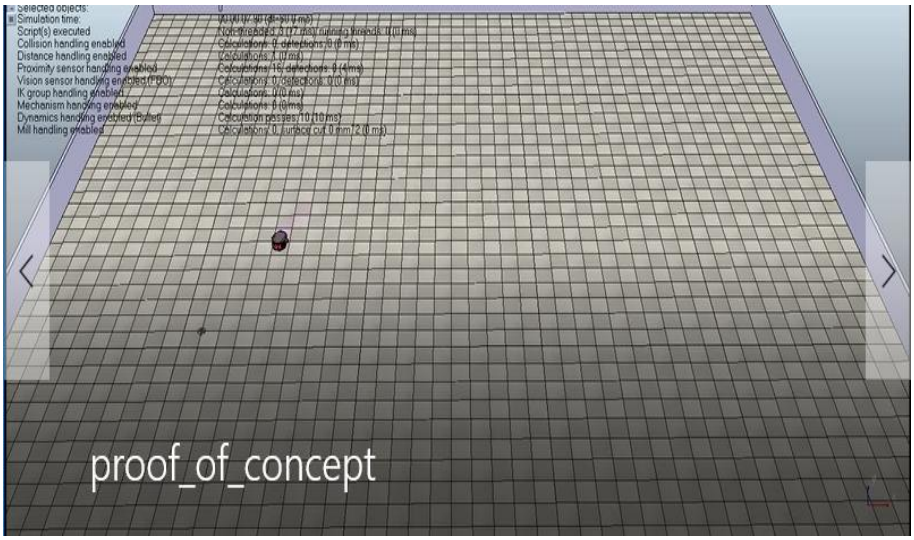
Στη συνέχεια έγινε η προσομοίωση του ρομποτικού οχήματος Pioneer 3pdx στο V-rep με τον ασαφή ελεγκτή που παρουσιάσαμε προηγουμένως. Σκοπός ήταν να μεταβεί από ένα αρχικό σημείο σε ένα τελικό σημείο-στόχο. Στις εικόνες 3.2-3.5 παρουσιάζονται στιγμιότυπα της κίνησης του οχήματος.



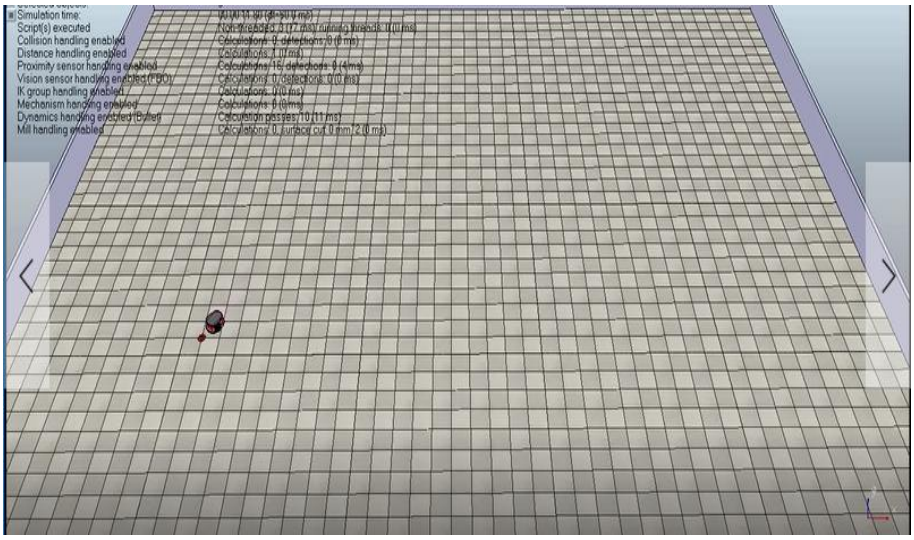
Εικόνα 3.2 Το όχημα στην αρχική θέση



Εικόνα 3.3 Ενδιάμεση θέση 1

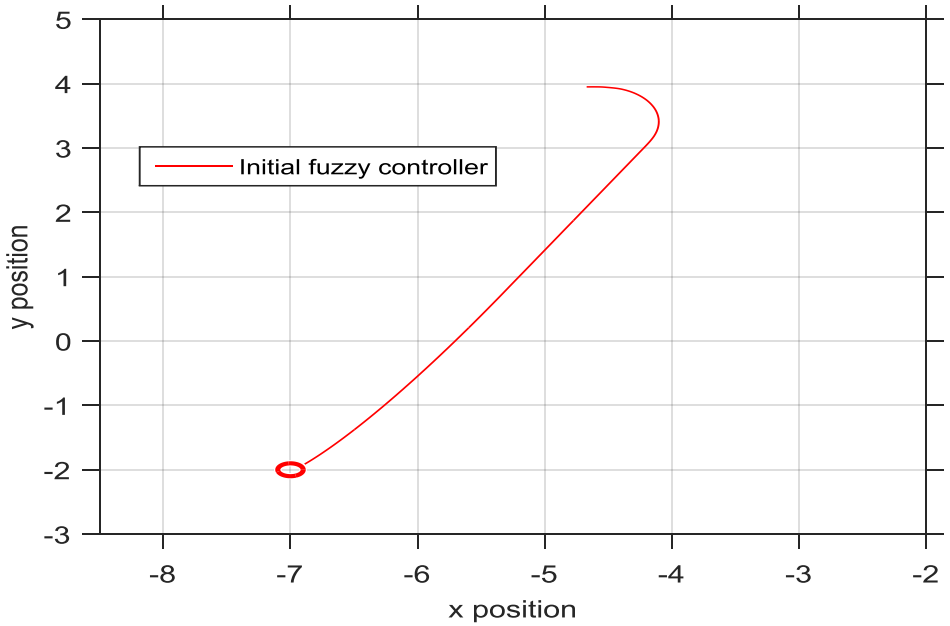


Εικόνα 3.4 Ενδιάμεση θέση 2

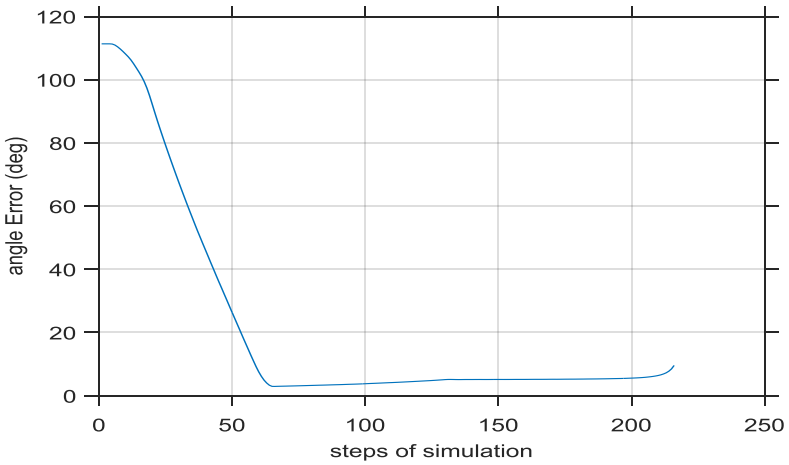


Εικόνα 3.5 Το όχημα λίγο πριν το τελικό σημείο-στόχο

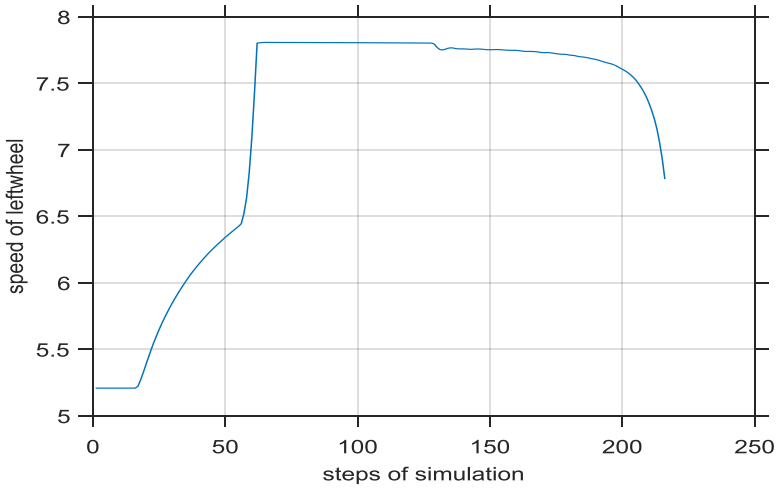
Στα σχήματα 3.11 & 3.14 παρουσιάζονται, η διαδρομή που ακολούθησε το όχημα, το σφάλμα της γωνίας κατεύθυνσης, καθώς και οι ταχύτητες του αριστερού και δεξιού τροχού αντίστοιχα. Ο κύκλος αντιπροσωπεύει το σημείο στόχο και θεωρείται ότι το ρομποτικό όχημα έχει επιτύχει το στόχο εάν έχει φτάσει σε μια απόσταση μικρότερη από ένα συγκεκριμένο κατώφλι.



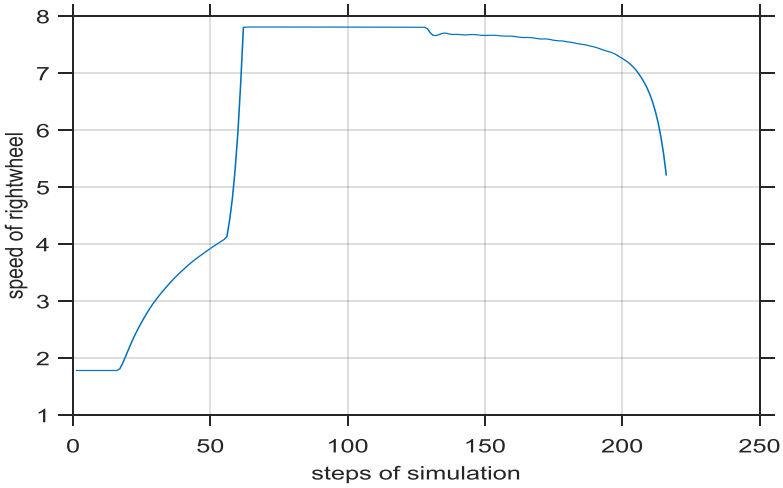
Σχήμα 3.11 Η διαδρομή του οχήματος



Σχήμα 3.12 Το σφάλμα της γωνίας κατεύθυνσης



Σχήμα 3.13 Η ταχύτητα του αριστερού τροχού



Σχήμα 3.14 Η ταχύτητα του δεξιού τροχού

Παρατηρούμε ότι το όχημα διανύει μια ομαλή τροχιά και καταφέρνει να προσεγγίσει το στόχο, το σφάλμα της γωνίας κατεύθυνσης μειώνεται έως ότου φτάσει κοντά στις 0 μοίρες, οι ταχύτητες των τροχών φτάνουν κοντά στα 7,5 m/s όπως ήταν αναμενόμενο σύμφωνα με τον σχεδιασμό του ασαφούς ελεγκτή. Η συγκεκριμένη προσομοίωση αποδεικνύει τόσο την λειτουργικότητα του ασαφούς ελεγκτή όσο και την ορθή λειτουργία του περιβάλλοντος προσομοίωσης που έχει επιλεγεί για τα πειράματα στα πλαίσια της συγκεκριμένης εργασίας.

ΚΕΦΑΛΑΙΟ 4

Βελτιστοποίηση της συμπεριφοράς ρομποτικών οχημάτων με τη χρήση του αλγορίθμου Σμήνους Σωματιδίων

4.1 Εισαγωγή

Στο κεφάλαιο αυτό μελετάται η επίδραση της εφαρμογής του αλγορίθμου Βελτιστοποίησης Σμήνους Σωματιδίων (PSO) σε ασαφείς ελεγκτές με στόχο να βελτιστοποιηθεί η συμπεριφορά τους. Επίσης γίνεται μελέτη της επίδρασης τους στην περίπτωση διαφορετικών όρων στη συνάρτηση αξιολόγησης. Αρχικά πραγματοποιήθηκαν απλές προσομοιώσεις βασισμένες αποκλειστικά στις εξισώσεις κίνησης, όπως αυτές περιγράφονται στην παράγραφο 3.2 και στη συνέχεια χρησιμοποιήθηκε ο ρεαλιστής προσομοίωσης V-REP για να μελετηθεί η επίδραση της συγκεκριμένης μεθοδολογίας σε ένα πραγματικό ρομποτικό όχημα, που προσομοιώνει το μέγιστο δυνατό βαθμό ακρίβειας. Το ρομποτικό όχημα που χρησιμοποιήθηκε ήταν το Pioneer 3pdx. Όλες οι προσομοιώσεις έγιναν στον ίδιο υπολογιστή με επεξεργαστή intel^{R} core™ i5-3337U στα 1.80GHz 64bit και με σταθερές συνθήκες. Στην συνέχεια παρουσιάζονται και αναλύονται τα δεδομένα των πειραμάτων.

4.2 Ταυτοποίηση λειτουργίας του αλγορίθμου Βελτιστοποίησης Σμήνους Σωματιδίων

Σκοπός του συγκεκριμένου πειράματος ήταν να μελετηθεί η επίδραση ενός αλγορίθμου PSO σε έναν ασαφή ελεγκτή εύρεσης θέσης και συγκεκριμένα πέντε (5) διαφορετικών σημείων, σε περιβάλλον χωρίς εμπόδια. Σκοπός είναι να βελτιστοποιηθεί ο ελεγκτής ως προς τον τρόπο συμπεριφοράς του, δηλαδή ως προς το μήκος της διαδρομής που διανύει, το χρόνο που χρειάζεται για εκτελέσει τη δοκιμασία και τον τρόπο με τον οποίο στρίβει.

Για το λόγο αυτό δημιουργήθηκε μια συνάρτηση αξιολόγησης που λαμβάνει υπόψη: (i) πόσα από τα σημεία – στόχους καταφέρνει να προσεγγίσει το όχημα, (ii) την ταχύτητα που κινείται, (iii) το πόσο απότομα στρίβει, (iv) το χρόνο που κάνει μέχρι να ολοκληρώσει την διαδρομή καθώς και (v) το μήκος της διαδρομής και αξιολογεί την επίδοση του κάθε ελεγκτή σύμφωνα με τη σχέση:

$$z = \frac{\frac{6}{j}\, final\ route}{V\left(^{250}/_{simulation\ steps}\right)\left(1 - \sqrt{dV}\right)} \quad (4.1)$$

Όπου j=1…6 τα σημεία (1-5) που καταφέρνει να προσεγγίσει το όχημα, $final\ route$ το μήκος της διαδρομής, V ο μέσος όρος της μέσης ταχύτητας των δύο τροχών του οχήματος, $simulation\ steps$ το άθροισμα των βημάτων σε κάθε προσομοίωση και dV ο απόλυτος μέσος όρος της διαφοράς των ταχυτήτων μεταξύ των δύο τροχών. Όσο ελαχιστοποιείται η τιμή της συνάρτησης αξιολόγησης τόσο καλύτερος είναι ο ελεγκτής. Όταν το όχημα δεν καταφέρνει να προσεγγίσει και τα πέντε σημεία τότε μπαίνει μία ποινή κατά την βαθμολόγηση του ελεγκτή.

Στη συγκεκριμένη μελέτη δημιουργήθηκε μια βιβλιοθήκη από κώδικες και συναρτήσεις στο Matlab με διαφορετικές λειτουργίες, και παρουσιάζονται στους παρακάτω πίνακα.

Κώδικες	
InitialiseHandleObject	Αρχικοποιεί το Pioneer 3pdx
PSO_for_fuzzy_without_sensors	Εφαρμόζει τον αλγόριθμο PSO.
main_for_fuzzy_without_sensors	Υλοποιεί το σύνολο του πειράματος

Πίνακας 4.1 Οι κώδικες που αναπτύχθηκαν για την υλοποίηση της προσομοίωσης

Συναρτήσεις	
create_x_from_fuzzy_without_sensors	Δημιουργεί το διάνυσμα που έχει τις τιμές των παραμέτρων των συναρτήσεων συμμετοχής από την είσοδο και τις εξόδους του αρχικού ασαφούς ελεγκτή.
create_x_PSO_new	Δημιουργεί ένα νέο διάνυσμα που έχει τις τιμές των παραμέτρων των συναρτήσεων συμμετοχής από την είσοδο και τις εξόδους του νέου ασαφούς ελεγκτή που δημιουργείται σε κάθε επανάληψη.
build_pop	Δημιουργεί τον αρχικό πληθυσμό του σμήνους.
convert_fuzzy	Μετατρέπει τα διανύσματα των τιμών των παραμέτρων της εισόδου και της εξόδου του αρχικού πληθυσμού σε ασαφείς ελεγκτές.
convert_fuzzy_new	Μετατρέπει τα διανύσματα των τιμών των παραμέτρων της εισόδου και της εξόδου κάθε νέου ελεγκτή που δημιουργείται σε

	κάθε επανάληψη σε ασαφή ελεγκτή.
convert_best_fuzzy	Μετατρέπει τα διανύσματα των τιμών των παραμέτρων της εισόδου και της εξόδου του καλύτερου ελεγκτή σε ασαφή ελεγκτή.
ob_fun_for_fuzzy_without_sensors	Αξιολογεί την επίδοση των σωματιδίων-ελεγκτών.
empty_arena_pso	Υλοποιεί την προσομοίωση στο V-REP και καταγράφει τα δεδομένα.

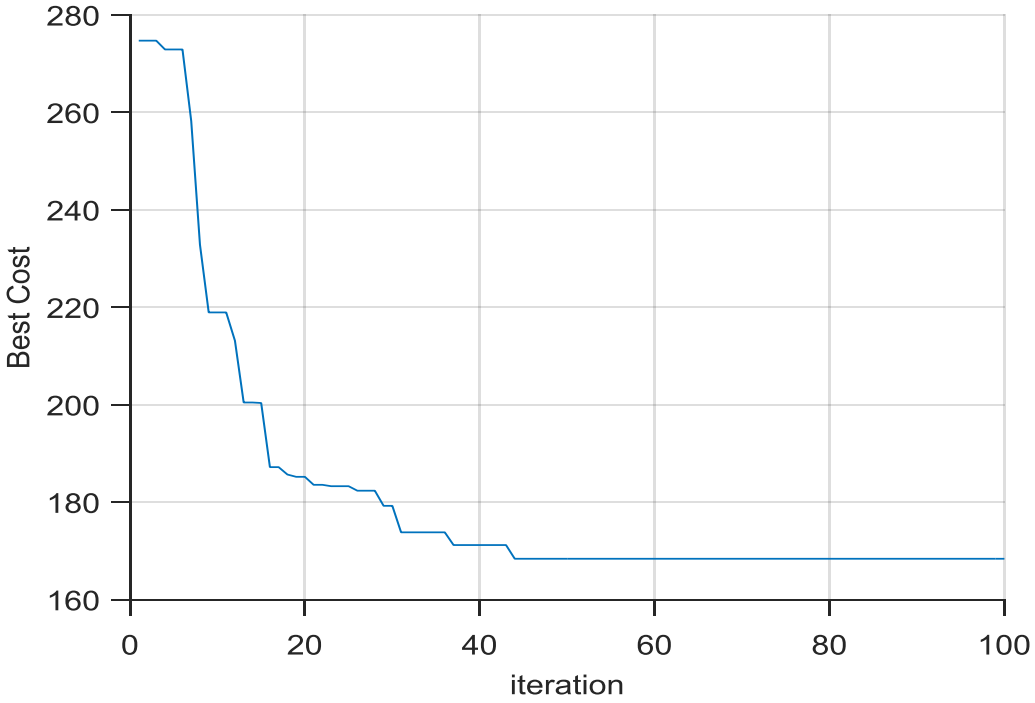
Πίνακας 4.2 Οι συναρτήσεις που αναπτύχθηκαν για την υλοποίηση της προσομοίωσης

Αρχικά με την συνάρτηση **create_x_from_fuzzy_without_sensors** δημιουργείται ένα διάνυσμα που έχει τις τιμές των παραμέτρων των συναρτήσεων συμμετοχής από την είσοδο και τις εξόδους του ασαφούς ελεγκτή. Το σύνολο των παραμέτρων του διανύσματος αποτελεί και τις διαστάσεις του χώρου αναζήτησης, στην περίπτωση μας ο ελεγκτής έχει 43 παραμέτρους. Στη συνέχεια δημιουργείται ο αρχικός πληθυσμός του σμήνους με τη χρήση της συνάρτησης **build_pop** και επιλέγεται το μέγεθος του, ίσο με τριάντα (30) σωματίδια. Η κατανομή των σωματιδίων στο χώρο αναζήτησης γίνεται σύμφωνα με τη σχέση: $x_{ij} = x_{min} + r_i(x_{max} - x_{min})$. Όπου *i=1...n, n* το μέγεθος του πληθυσμού, *r* τυχαίος αριθμός μεταξύ [0, 1], ώστε να υπάρχει όσο το δυνατόν μεγαλύτερη κάλυψη του χώρου. Ο πληθυσμός έχει την μορφή διανυσμάτων, οπότε με τη συνάρτηση **convert_fuzzy** τα μετατρέπουμε σε μορφή **.fis** δηλαδή σε ασαφείς ελεγκτές οι οποίοι αποτελούν τα σωματίδια του σμήνους. Μια άλλη συνάρτηση που χρησιμοποιείται είναι η **ob_fun_for_fuzzy_without_sensors** η οποία είναι η συνάρτηση αξιολόγησης της επίδοσης των σωματιδίων-ελεγκτών.

Έπειτα ακολουθεί ο κώδικας **PSO_for_fuzzy_without_sensors** στον οποίο εφαρμόζεται ο αλγόριθμος με παράγοντα περιορισμού (constriction factor) σχέση 2.21. Αρχικά επιλέγονται ο μέγιστος αριθμός επαναλήψεων και οι τιμές των παραμέτρων $\boldsymbol{\varphi_1 = 2.05,\ \varphi_2 = 2}$ για την γνωσιακή και την κοινωνική συνιστώσα, και **k=0.45**. Η επιλογή έγινε σύμφωνα με τη θεωρία και δίνεται μια μικρή βαρύτητα στην γνωσιακή συνιστώσα ώστε να γίνει πιο εντατικοποιημένη αναζήτηση. Ακολουθούν οι αρχικοποιήσεις των θέσεων και των ταχυτήτων των σωματιδίων, η αρχική ταχύτητα είναι ίση με μηδέν, των προσωπικών καλύτερων θέσεων (personal best), και της συνολικά καλύτερης θέσης (global best). Το επόμενο στάδιο είναι η προσομοίωση της πρώτης επανάληψης για κάθε έναν από τους ελεγκτές στο V-REP, η καταγραφή των δεδομένων και η αξιολόγησή τους σύμφωνα με την συνάρτηση που αναφέρθηκε πιο πάνω. Στη συνέχεια γίνεται ενημέρωση των προσωπικών καλύτερων θέσεων (personal best), και της συνολικά καλύτερης θέσης (global best). Έπειτα ακολουθεί η επόμενη επανάληψη του αλγορίθμου με την ανανέωση της ταχύτητας και της θέσης κάθε σωματιδίου σύμφωνα με

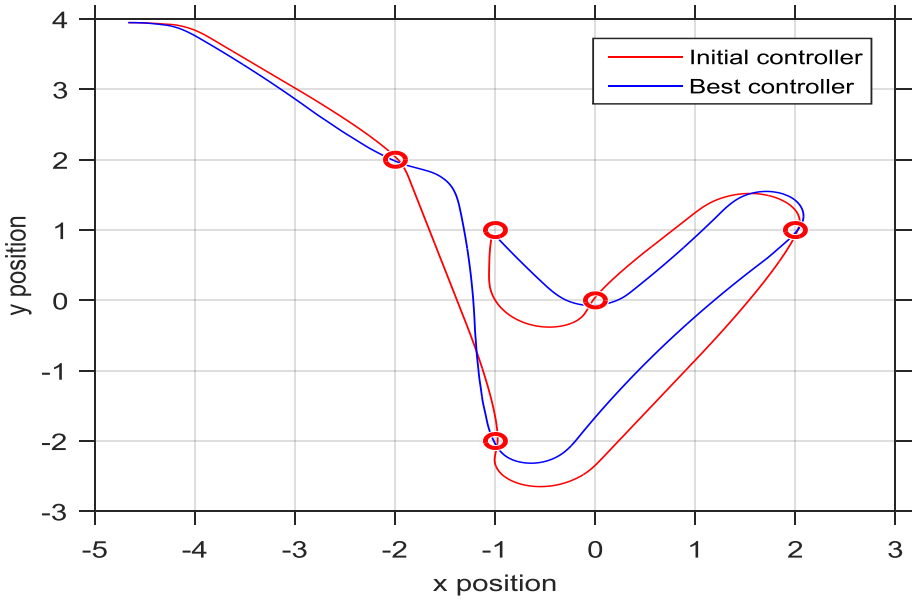
τις σχέσεις 2.11 & 2.21, γίνεται προσομοίωση κάθε ελεγκτή στο V-REP, ακολουθεί η αξιολόγηση και η ενημέρωση των προσωπικών καλύτερων θέσεων (personal best), και της συνολικά καλύτερης θέσης (global best). Ο αλγόριθμος επαναλαμβάνεται μέχρι την ικανοποίηση της συνθήκης τερματισμού, που στην περίπτωση αυτή είναι ο μέγιστος αριθμός επαναλήψεων που είναι ίσος με εκατό (100).

Μετά το πέρας της προσομοίωσης έγινε καταγραφή και ανάλυση των δεδομένων. Στο σχήμα 4.1 παρουσιάζεται πως μεταβάλλεται η καλύτερη τιμή της συνάρτησης αξιολόγησης σε κάθε επανάληψη. Η αρχική καλύτερη τιμή είναι **274.69** και η συνολικά καλύτερη τιμή είναι **168.37.** Η μείωση είναι της τάξης του **38.70%** Παρατηρούμε ότι μετά τη $44_{η}$ επανάληψη δεν υπάρχει περαιτέρω βελτίωση και ο αλγόριθμος συγκλίνει σε μια βέλτιστη τιμή.



Σχήμα 4.1 Μεταβολή συνάρτησης αξιολόγησης

Στο σχήμα 4.2 παρουσιάζεται μια σύγκριση της διαδρομής που ακολουθεί το όχημα με τον αρχικό και τον καλύτερο ελεγκτή προσεγγίζοντας τα πέντε σημεία-στόχους.



Σχήμα 4.2 Διαδρομή οχήματος με τον αρχικό και τον καλύτερο ελεγκτή

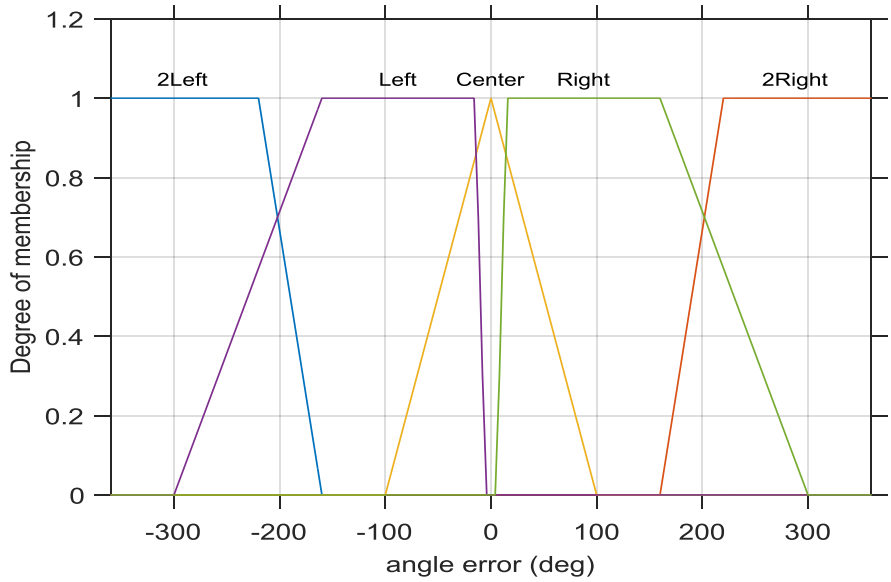
Όσον αφορά τη διαφοροποίηση στη συμπεριφορά του ελεγκτή με βάση της επίδοσης τους, τα αποτελέσματα παρουσιάζονται στον πίνακα 4.3.

	Αρχικός ασαφής ελεγκτής	**Καλύτερος ασαφής ελεγκτής**	**Ποσοστό μεταβολής %**
Επανάληψη	1	44	
Σωματίδιο	1	17	
Τιμή συνάρτησης αξιολόγησης	274.69	168.37	-38.70
Βήματα προσομοίωσης (steps)	602	479	-20.43
Χρόνος διαδρομής (sec)	32.39	26.49	-18.21
Μήκος διαδρομής (m)	18.39	17.23	-6.30
Μέση ταχύτητα (m/s)	6.23	7.57	+21.50
dV	1.34	1.60	+19.40

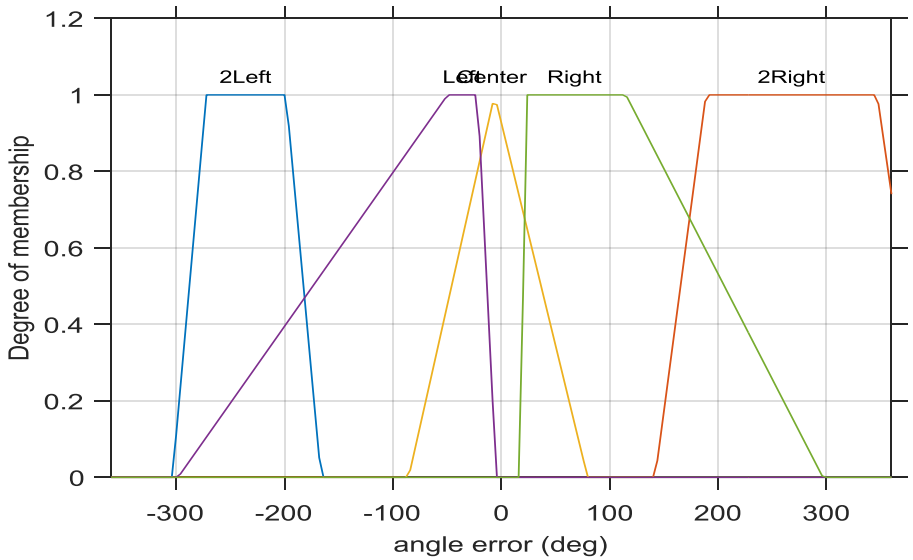
Πίνακας 4.3 Δεδομένα προσομοίωσης 1

Συμπερασματικά ο ελεγκτής παρουσιάζει σημαντική βελτίωσή με το όχημα να κινείται με μεγαλύτερη ταχύτητα, γι' αυτό και στρίβει πιο απότομα, διανύοντας ταυτόχρονα μικρότερη διαδρομή με αποτέλεσμα να προσεγγίζει τους στόχους σε λιγότερο χρόνο.

Στα παρακάτω σχήματα παρουσιάζονται πως έχουν διαφοροποιηθεί οι τιμές των παραμέτρων των συναρτήσεων συμμετοχής της εισόδου και των εξόδων του ελεγκτή.

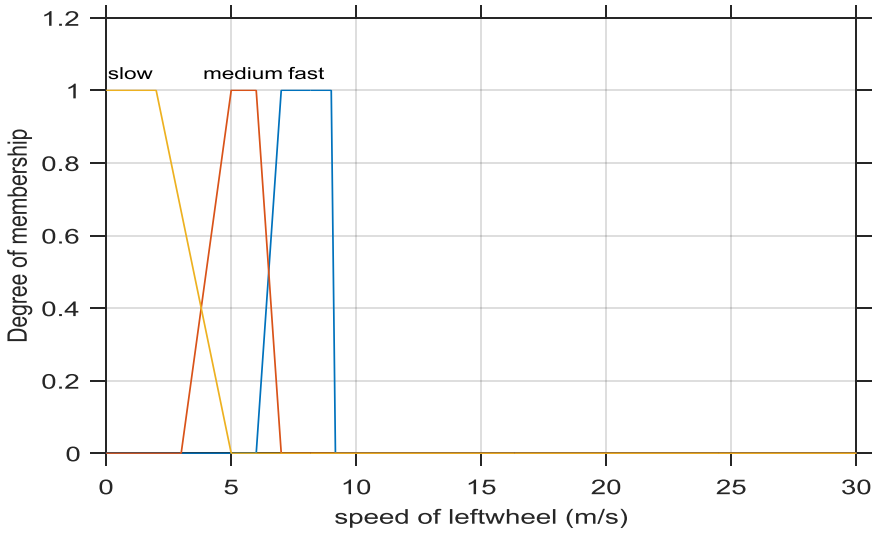


Σχήμα 4.3 Είσοδος «Σφάλμα γωνίας» αρχικού ελεγκτή

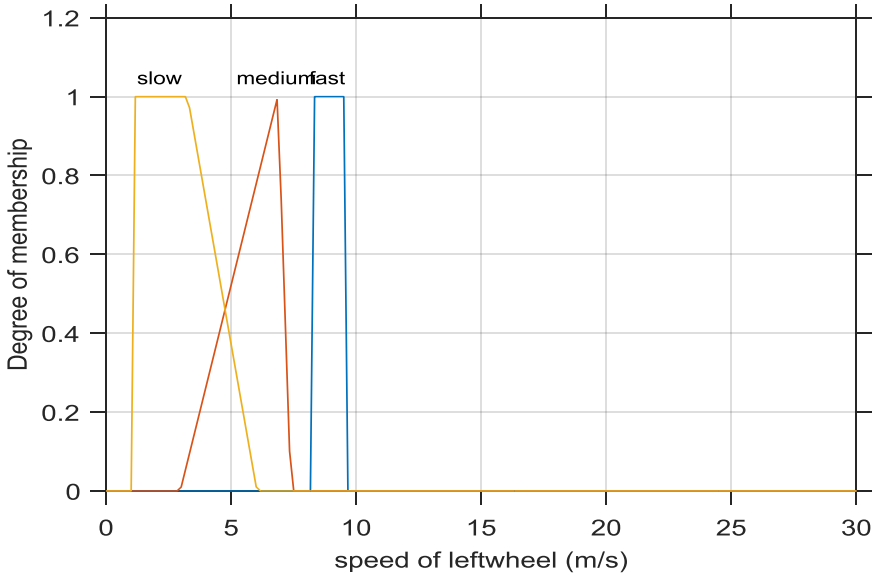


Σχήμα 4.4 Είσοδος «Σφάλμα γωνίας» βέλτιστου ελεγκτή

Στο ασαφές σύνολο «γωνία σφάλματος», παρατηρούμε ότι υπάρχει μια σύμπτυξη του εύρους των συναρτήσεων συμμετοχής με αποτέλεσμα το όχημα να στρίβει πιο απότομα.

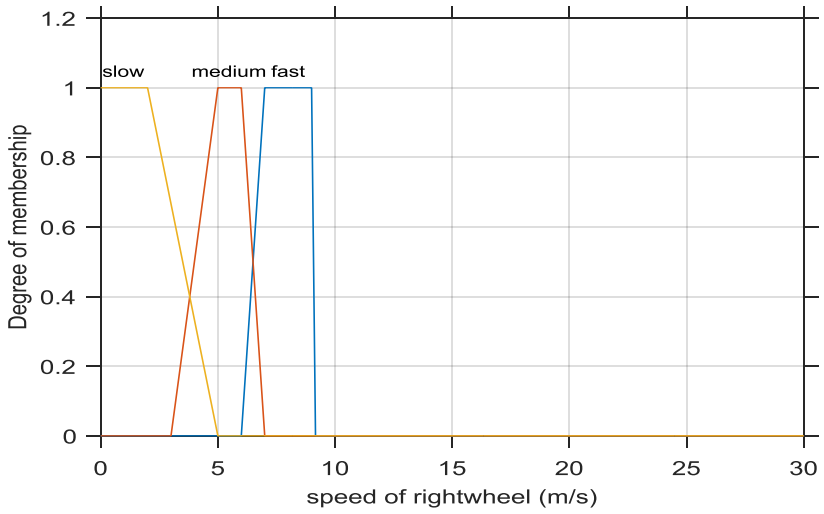


Σχήμα 4.5 Έξοδος «ταχύτητα αριστερού τροχού» αρχικού ελεγκτή

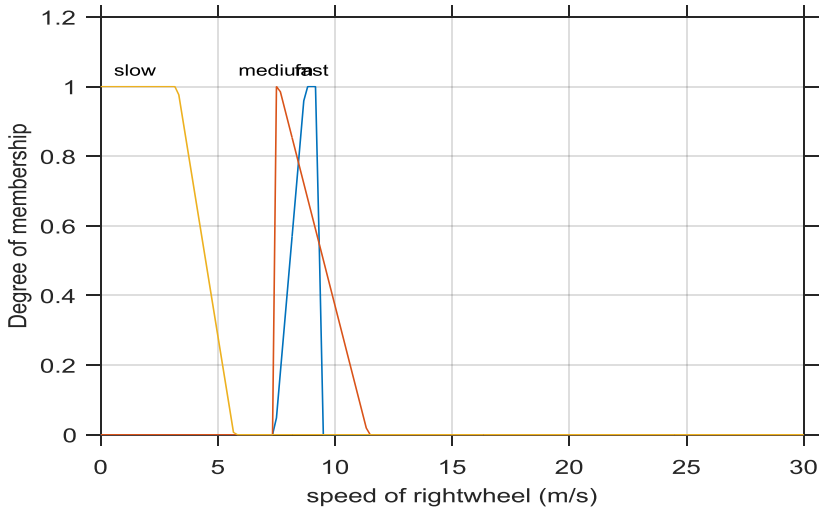


Σχήμα 4.6 Έξοδος «ταχύτητα αριστερού τροχού» βέλτιστου ελεγκτή

Στο ασαφές σύνολο «ταχύτητα αριστερού τροχού» παρατηρούμε ότι η συνάρτηση συμμετοχής «αργά» μετακινήθηκε προς τα δεξιά και επομένως περιέχει μεγαλύτερες τιμές ταχυτήτων, η συνάρτηση συμμετοχής «μέτρια» έγινε πιο ασαφής σε αντίθεση με την «γρήγορα» που έγινε πιο συγκεκριμένη.



Σχήμα 4.7 Έξοδος «ταχύτητα δεξιού τροχού» αρχικού ελεγκτή



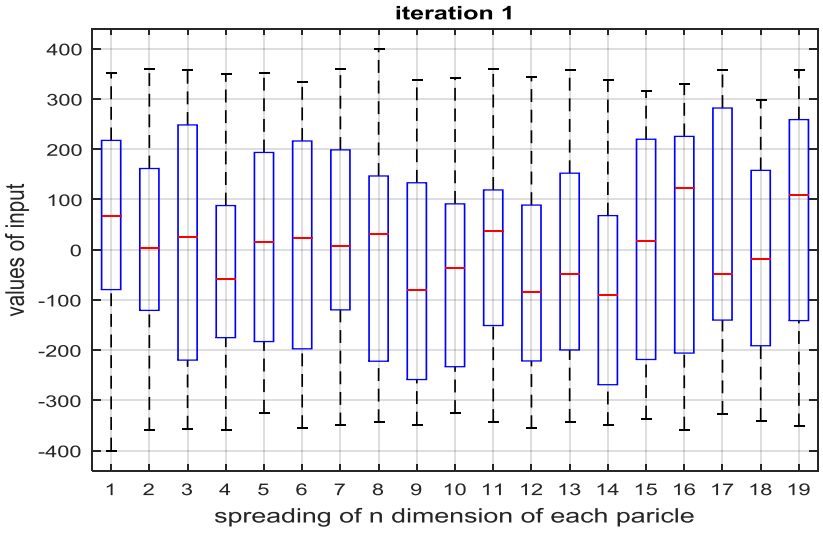
Σχήμα 4.8 Έξοδος «ταχύτητα δεξιού τροχού» βέλτιστου ελεγκτή

Στο ασαφές σύνολο «ταχύτητα δεξιού τροχού» παρατηρούμε ότι η συνάρτηση συμμετοχής «αργά» αύξησε το εύρος της λίγο περισσότερο από 5 m/s που ήταν αρχικά, η συνάρτηση συμμετοχής «μέτρια» μετακινήθηκε προς τα δεξιά και σχεδόν αλληλοκαλύπτεται με την «αργά» αυξάνοντας έτσι την ασάφεια του συστήματος.

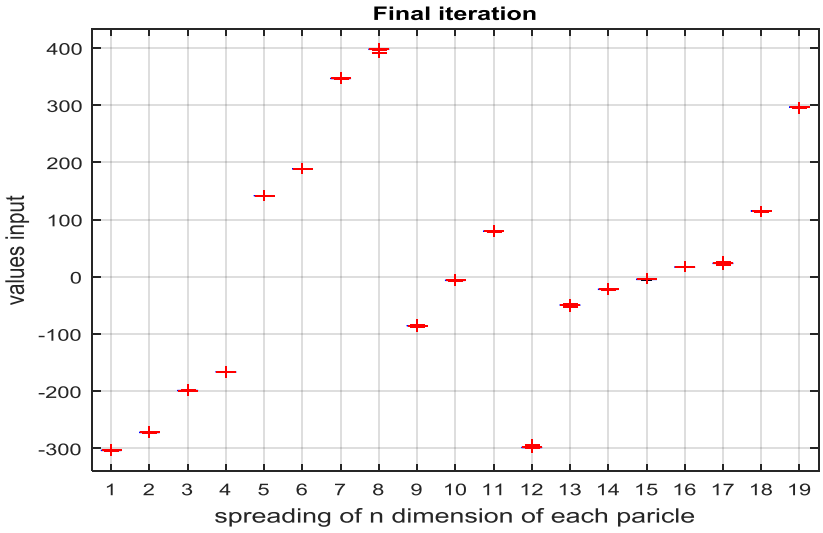
Η αύξηση του μέγιστου ορίου του εύρους τιμών των ασαφών συνόλων: «ταχύτητα αριστερού τροχού» και «ταχύτητα δεξιού τροχού» είχε ως αποτέλεσμα το όχημα να κινείται με μεγαλύτερη ταχύτητα.

Στα σχήματα 4.9 και 4.10 παρουσιάζεται η κατανομή των παραμέτρων των συναρτήσεων των συμμετοχής της εισόδου της γωνίας σφάλματος των ελεγκτών κατά τη δημιουργία του πληθυσμού και στην τελική επανάληψη αντίστοιχα. Παρατηρούμε ότι

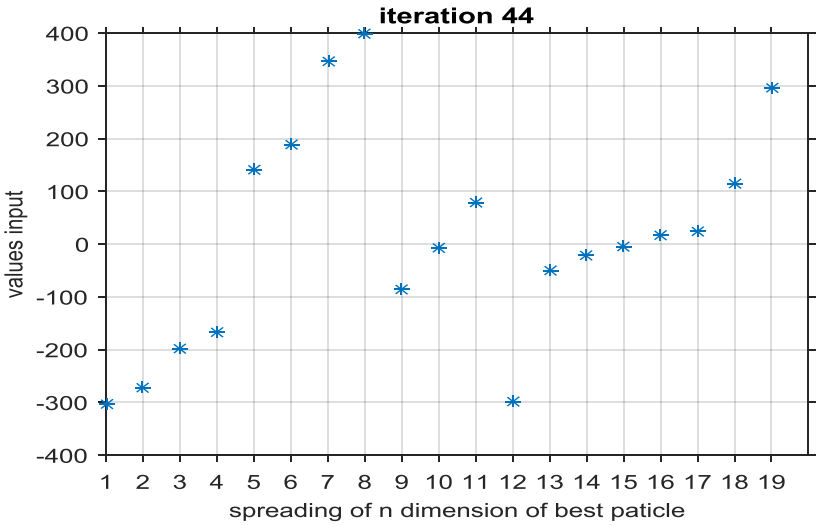
ενώ στην πρώτη επανάληψη η κατανομή είναι μεγάλη και έχει εύρος σχεδόν από -360 έως 360 σε κάθε διάσταση του διανύσματος, που ήταν και το επιθυμητό, στην τελική επανάληψη οι τιμές των παραμέτρων έχουν συγκλίνει σχεδόν σε συγκεκριμένη τιμή. Στο σχήμα 4.11 παρουσιάζονται οι παράμετροι των συναρτήσεων συμμετοχής του βέλτιστου ελεγκτή μετά από 44 επαναλήψεις. Παρατηρούμε πως όλοι οι παράμετροι όλων των σωματιδίων-ελεγκτών έχουν συγκλίνει στη βέλτιστη λύση.



Σχήμα 4.9 Κατανομή παραμέτρων του συνόλου των σωματιδίων για την είσοδο «Γωνία σφάλματος»

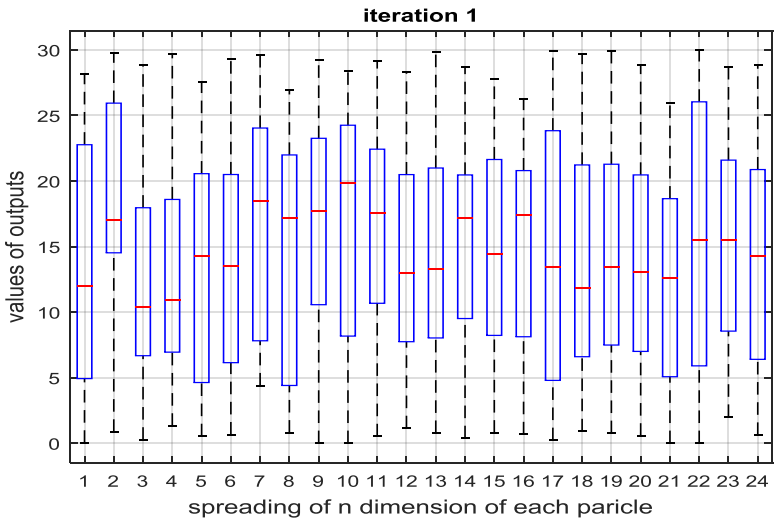


Σχήμα 4.10 Κατανομή παραμέτρων του συνόλου των σωματιδίων για την είσοδο «Γωνία σφάλματος»

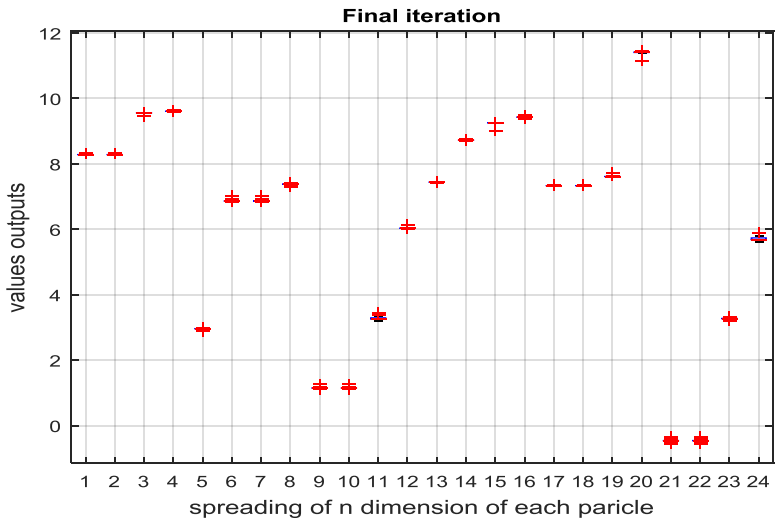


Σχήμα 4.11 Κατανομή παραμέτρων του βέλτιστου ελεγκτή για την είσοδο «Γωνία σφάλματος»

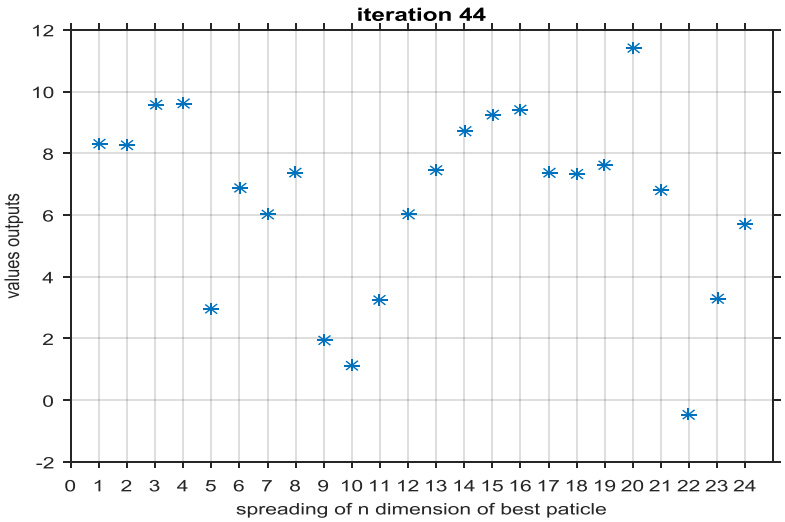
Στα σχήματα 4.12 και 4.13 παρουσιάζεται η κατανομή των παραμέτρων των συναρτήσεων των συμμετοχής των εξόδων των ταχυτήτων των ελεγκτών κατά τη δημιουργία του πληθυσμού και στην τελική επανάληψη αντίστοιχα. Παρατηρούμε ότι ενώ στην πρώτη επανάληψη η κατανομή είναι μεγάλη και έχει εύρος σχεδόν από 0 έως 30 σε κάθε διάσταση του διανύσματος, που ήταν και το επιθυμητό, στην τελική επανάληψη οι τιμές των παραμέτρων έχουν συγκλίνει σχεδόν σε συγκεκριμένη τιμή. Στο σχήμα 4.14 παρουσιάζονται οι παράμετροι των συναρτήσεων συμμετοχής του βέλτιστου ελεγκτή μετά από 44 επαναλήψεις. Παρατηρούμε πως όλοι οι παράμετροι όλων των σωματιδίων-ελεγκτών έχουν συγκλίνει στη βέλτιστη λύση.



Σχήμα 4.12 Κατανομή παραμέτρων του συνόλου των σωματιδίων των εξόδων «Ταχύτητα αριστερού τροχού» και «Ταχύτητα δεξιού τροχού»

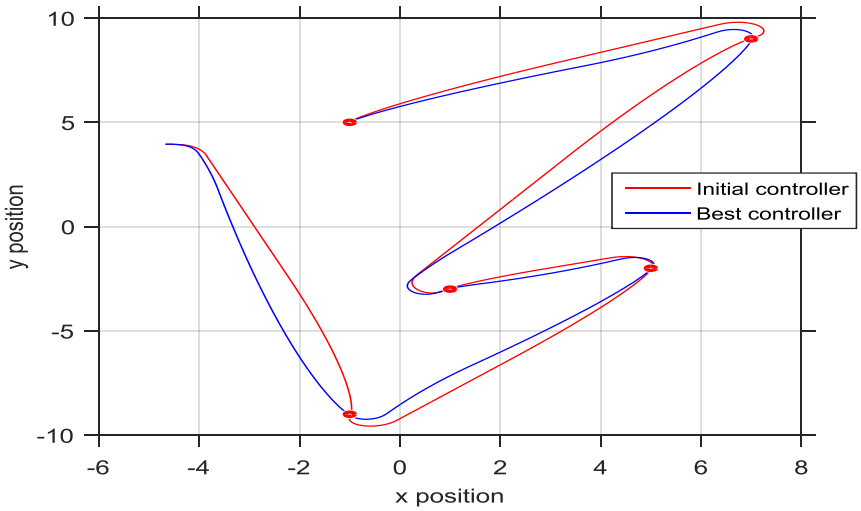


Σχήμα 4.13 Κατανομή παραμέτρων του συνόλου των σωματιδίων των εξόδων «Ταχύτητα αριστερού τροχού» και «Ταχύτητα δεξιού τροχού»



Σχήμα 4.14 Κατανομή παραμέτρων του συνόλου βέλτιστου ελεγκτή των εξόδων «Ταχύτητα αριστερού τροχού» και «Ταχύτητα δεξιού τροχού»

Προκειμένου να δοκιμάσουμε πως συμπεριφέρεται ο ελεγκτής στη περίπτωση διαφορετικών σημείων-στόχων έγινε νέα προσομοίωση. Στη συνέχεια παρουσιάζονται τα αποτελέσματα με τον αρχικό και βέλτιστο ελεγκτή.



Σχήμα 4.15 Διαδρομή οχήματος με τον αρχικό και τον καλύτερο ελεγκτή

	Αρχικός ασαφής ελεγκτής	**Καλύτερος ασαφής ελεγκτής**	**Ποσοστό μεταβολής %**
Βήματα προσομοίωσης (steps)	1579	1327	-15.96
Χρόνος διαδρομής (sec)	80	67	-16.25
Μήκος διαδρομής (m)	53.86	52.60	-2.34
Μέση ταχύτητα (m/s)	7.01	8.19	+16.83
dV (m/s)	0.67	0.83	+23.88

Πίνακας 4.4 Δεδομένα προσομοίωσης 2

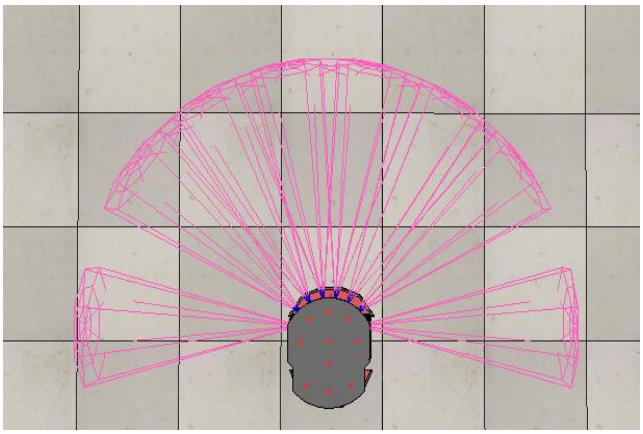
Ο βέλτιστος ελεγκτής παρουσιάζει ευρωστία καθώς έχει την ίδια συμπεριφορά και σε διαφορετικό περιβάλλον δοκιμής.

4.3 Εφαρμογή του αλγορίθμου Βελτιστοποίησης Σμήνους Σωματιδίων σε ασαφή ελεγκτή αποφυγής εμποδίων

Σκοπός του δεύτερου πειράματος είναι η βελτιστοποίηση ενός ασαφούς ελεγκτή που οδηγεί ένα διαφορικό όχημα σε επιθυμητές θέσεις αποφεύγοντας εμπόδια με τη χρήση αισθητήρων υπερήχων. Η δημιουργία του ελεγκτή έγινε με τη βοήθεια του fuzzy toolbox του Matlab καθώς επίσης και όλοι μαθηματικοί υπολογισμοί, οι κώδικες που είναι απαραίτητοι για τις προσομοιώσεις, για την καταγραφή και την ανάλυση των δεδομένων.

4.3.1 Δημιουργία ελεγκτή αποφυγής εμποδίων

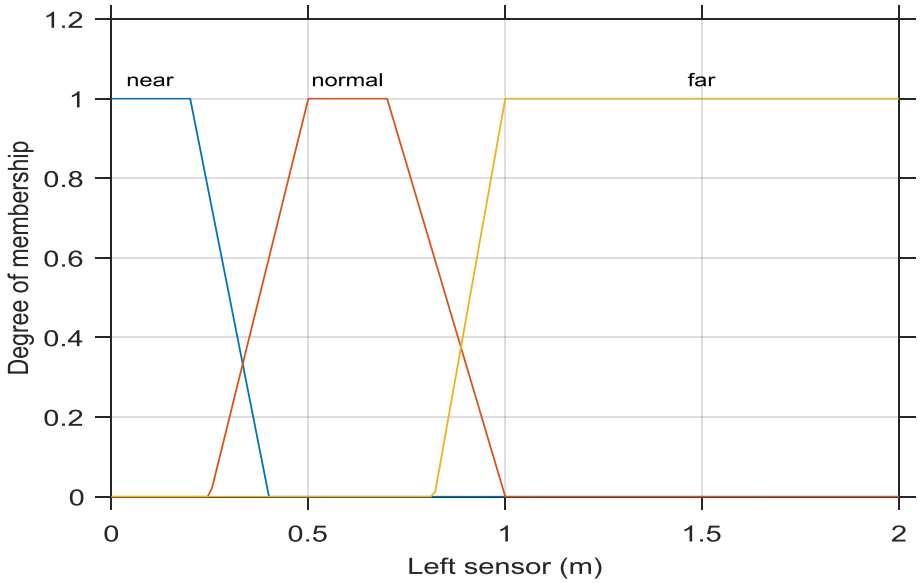
Το όχημα της προσομοίωσης έχει οκτώ (8) αισθητήρες καλύπτοντας μια περιοχή 180° μοιρών, κάθε αισθητήρας έχει ακτίνα ανίχνευσης από 0 έως 1 μέτρο, γωνία κάλυψης 30° μοιρών, η περιοχή που καλύπτουν στο χώρο είναι κωνική με βάση ακτίνας 0,005 μέτρα, στην εικόνα 4.1 φαίνεται πως είναι κατανεμημένοι.



Εικόνα 4.1 Η κατανομή των αισθητήρων

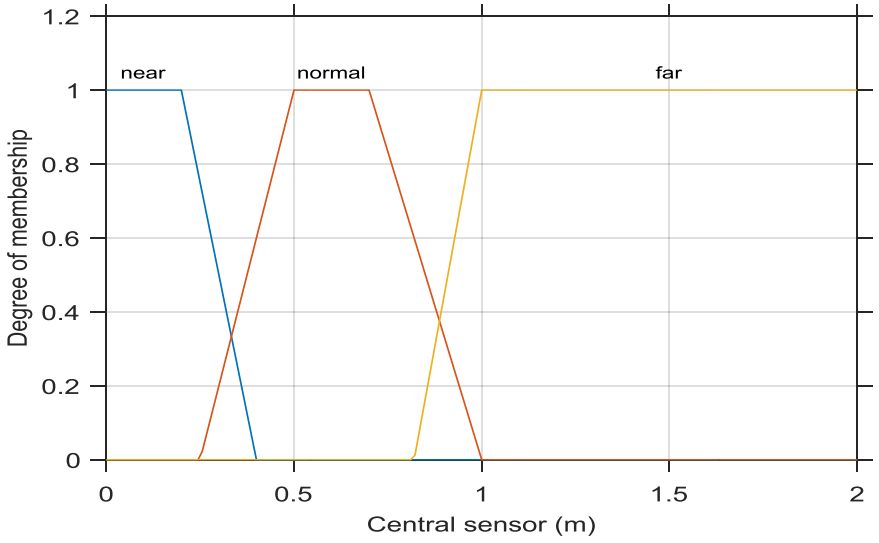
Οι αισθητήρες έχουν ομαδοποιηθεί, ξεκινώντας από αριστερά προς τα δεξιά, οι τρεις πρώτοι αποτελούν την αριστερή πλευρά ανίχνευσης, οι επόμενοι δύο την κεντρική και οι υπόλοιποι τρεις την δεξιά, οι ομάδες αυτές αποτελούν και τις τρεις από τις τέσσερις εισόδους του ασαφούς ελεγκτή. Για κάθε μία από τις ομάδες των αισθητήρων, ως είσοδος για τον ελεγκτή λαμβάνεται η μικρότερη τιμή που καταγράφεται από τους αισθητήρες της ομάδας, σε κάθε βήμα προσομοίωσης. Η τέταρτη είσοδος του ελεγκτή είναι γωνία μεταξύ του οχήματος και της επιθυμητής θέσης όπως είδαμε και στον πρώτο ελεγκτή, και οι δύο έξοδοι είναι η ταχύτητα του αριστερού και του δεξιού τροχού. Στα παρακάτω σχήματα παρουσιάζονται οι συναρτήσεις συμμετοχής των εισόδων και των εξόδων του ασαφούς ελεγκτή τύπου Mamdani που δημιουργήθηκε για την κίνηση ρομποτικού οχήματος διαφορικής κίνησης σε επιθυμητά σημεία με αποφυγή εμποδίων.

Η πρώτη είσοδος αποτελείται από 3 ασαφή σύνολα που χαρακτηρίζονται από τις συναρτήσεις συμμετοχής: (κοντά, κανονικά, μακριά). Το εύρος τιμών του ασαφούς συνόλου είναι από 0 έως 2 μέτρα ενώ η περιοχή ανίχνευσης εμποδίων των αισθητήρων είναι από 0-1 μέτρο.



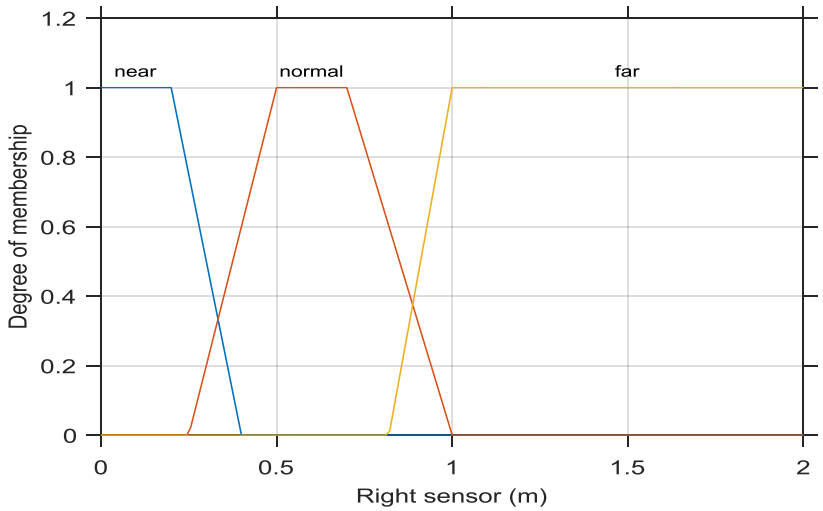
Σχήμα 4.16 Είσοδος «Αισθητήρας αριστερά» ασαφούς ελεγκτή

Ομοίως η δεύτερη είσοδος



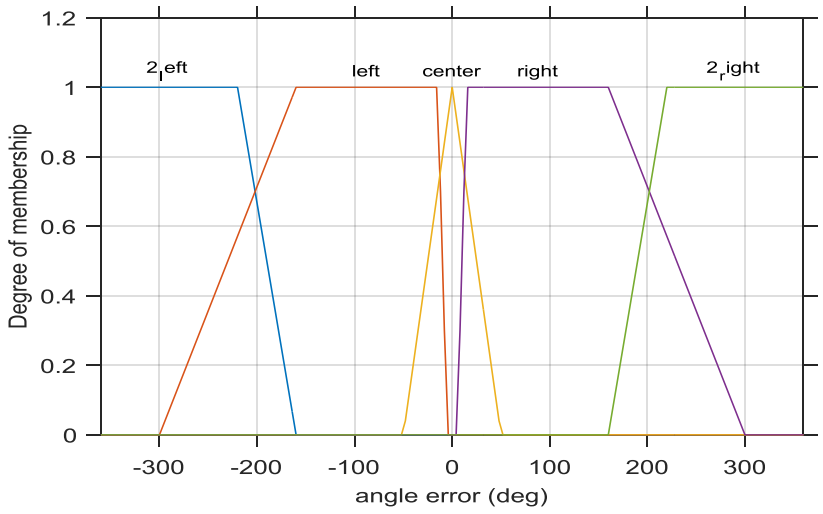
Σχήμα 4.17 Είσοδος «Αισθητήρας κέντρο» ασαφούς ελεγκτή

Ομοίως η τρίτη είσοδος



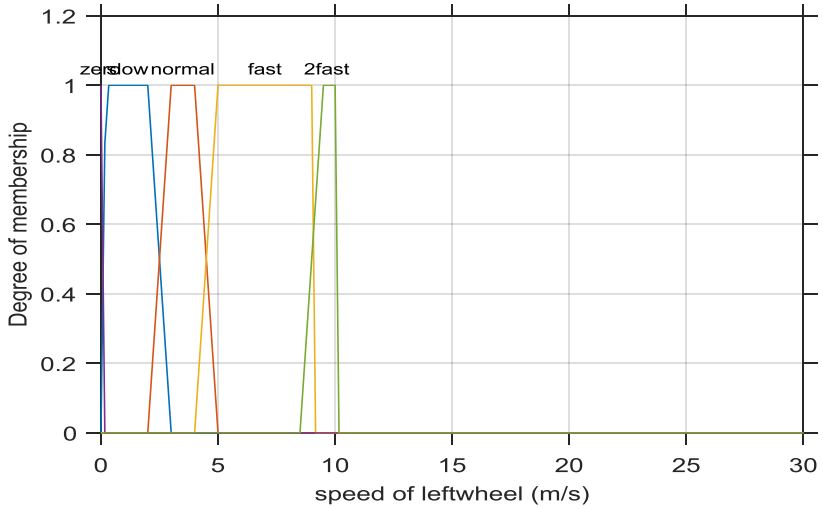
Σχήμα 4.18 Είσοδος «Αισθητήρας δεξιά» ασαφούς ελεγκτή

Η τέταρτη είσοδος αποτελείται από 5 ασαφή σύνολα που χαρακτηρίζονται από τις συναρτήσεις συμμετοχής: (Πολύ αριστερά, αριστερά, κέντρο, δεξιά, πολύ δεξιά), με εύρος από -360 έως 360 μοίρες όπως φαίνεται στο σχήμα 4.19.



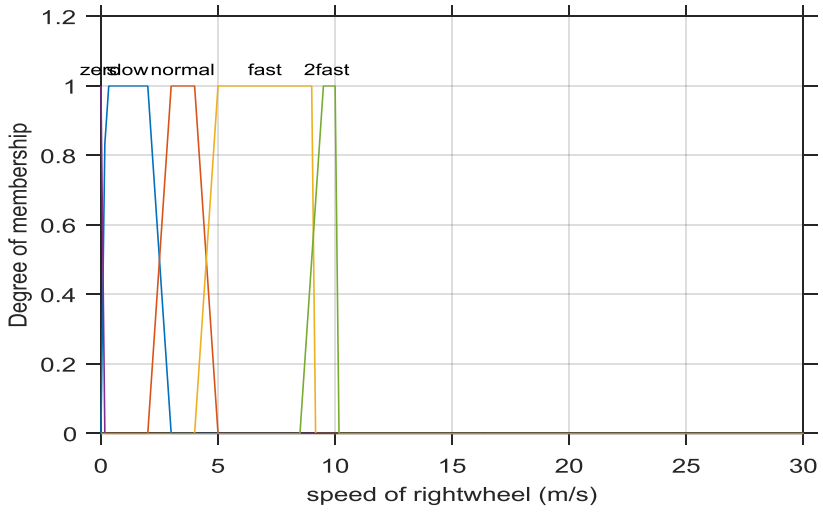
Σχήμα 4.19 Είσοδος «Γωνία σφάλματος» ασαφούς ελεγκτή

Η πρώτη έξοδος του ελεγκτή είναι η ταχύτητα του αριστερού τροχού U_L και αποτελείται από 5 ασαφή σύνολα που χαρακτηρίζονται από τις συναρτήσεις συμμετοχής: (μηδέν, αργά, μέτρια, γρήγορα, πολύ γρήγορα), με εύρος από 0 έως 10 m/s όπως φαίνεται στο σχήμα 4.20.



Σχήμα 4.20 Έξοδος 1 ταχύτητα αριστερού τροχού U_L

Ομοίως και για την δεύτερη έξοδο του ελεγκτή που είναι η ταχύτητα του δεξιού τροχού.



Σχήμα 4.21 Έξοδος 1 ταχύτητα δεξιού τροχού U_R

Στη συνέχεια δημιουργήσαμε 32 λεκτικούς κανόνες, στον πίνακα 4.3 αναφέρονται ενδεικτικά μερικοί από αυτούς.

ΛΕΚΤΙΚΟΙ ΚΑΝΟΝΕΣ								
	ΕΙΣΟΔΟΣ 1	**ΕΙΣΟΔΟΣ 2**	**ΕΙΣΟΔΟΣ 3**	**ΕΙΣΟΔΟΣ 4**		**ΕΞΟΔΟΣ 1**		**ΕΞΟΔΟΣ 2**
	Αισθητήρας αριστερά	**Αισθητήρας κέντρο**	**Αισθητήρας δεξιά**	**Γωνία Σφάλματος**		**Ταχύτητα U_L**		**Ταχύτητα U_R**
Εάν	Μακριά	Μακριά	Μακριά	-	**Τότε**	Μέτρια	**Και**	Γρήγορα
	Μακριά	Κανονικά	Κοντά	-		Μηδέν		Αργά
	Κανονικά	Μακριά	Κανονικά	-		Αργά		Αργά
	Κανονικά	Κανονικά	Κοντά	-		Μηδέν		Αργά
	Κοντά	Μακριά	Κανονικά	-		Μέτρια		Μηδέν
	Κοντά	Κοντά	Μακριά	-		Αργά		Μηδέν
	Μακριά	Μακριά	Μακριά	Πολύ αριστερά		Μηδέν		Γρήγορα
	Μακριά	Μακριά	Μακριά	Δεξιά		Πολύ Γρήγορα		Αργά

Πίνακας 4.5 Τμήμα της βάσης κανόνων του ασαφούς ελεγκτή αποφυγής εμποδίων

4.3.2 Επιλογή συναρτήσεων αξιολόγησης

Στη συνέχεια αναπτύχθηκε το σενάριο της προσομοίωσης που έχει ως εξής: το όχημα ξεκινάει από μια αρχική θέση και μεταβαίνει στο σημείο-στόχο αποφεύγοντας ενδιάμεσα εμπόδια. Σκοπός είναι η βελτιστοποίηση της συμπεριφοράς του οχήματος σε σχέση με τον συνολικό χρόνο που χρειάζεται για να φτάσει στο σημείο-στόχο, το μήκος της διανυθείσας διαδρομής, την ταχύτητα με την οποία κινείται, την απόσταση προσέγγισης των εμποδίων και την ομαλότητα που στρίβει.

Οι συναρτήσεις αξιολόγησης μπορούν να ταξινομηθούν ανάλογα με τη γνώση που υπάρχει πριν την έναρξη της διαδικασίας εξέλιξης και ενσωματώνεται σ'αυτές προτού αρχίσει η διαδικασία της εξέλιξης. Οι συναρτήσεις αξιολόγησης χωρίζονται σε επτά κατηγορίες που παρουσιάζονται στον πίνακα 4.6 [54].

	Κατηγορία συνάρτησης αξιολόγησης	**Επίπεδο πρότερης γνώσης που έχει ενσωματωθεί στη συνάρτηση αξιολόγησης**
1	Συναρτήσεις προσαρμογής που χρησιμοποιούν δεδομένα εκπαίδευσης	Πολύ υψηλό
2	Συναρτήσεις προσαρμογής βασισμένες σε συμπεριφορές	Υψηλό

3	Συγκεντρωτικές συναρτήσεις προσαρμογής	Χαμηλό
4	Συναρτήσεις προσαρμογής για συγκεκριμένο πρόβλημα	Μέτριο
5	Αυξητικές συναρτήσεις προσαρμογής βασισμένες στον τρόπο λειτουργίας	Μέτριο έως Υψηλό
6	Αυξητικές συναρτήσεις προσαρμογής βασισμένες στο περιβάλλον	Χαμηλό έως Μέτριο
7	Ανταγωνιστικές συναρτήσεις προσαρμογής	Χαμηλό έως Μέτριο

Πίνακας 4.6 Κατηγοριοποίηση των συναρτήσεων προσαρμογής [54]

Για την αξιολόγηση του ελεγκτή χρησιμοποιήθηκαν δύο συναρτήσεις αξιολόγησης με διαφορετική φιλοσοφία ώστε να γίνει σύγκριση της συμπεριφοράς του ρομποτικού οχήματος.

Η πρώτη συνάρτηση αξιολόγησης, αξιολογεί την επίδοση του κάθε ελεγκτή λαμβάνοντας υπόψη τον συνολικό χρόνο που χρειάζεται για να φτάσει στο σημείο-στόχο, το μήκος της διανυθείσας διαδρομής, την ταχύτητα με την οποία κινείται, την απόσταση προσέγγισης των εμποδίων και την ομαλότητα που στρίβει. Η συνάρτηση αυτή είναι βασισμένη σε συμπεριφορές και είναι σχεδιασμένη ώστε να μετράει ποιοτικά τις διάφορες δραστηριότητες ενός ρομποτικού οχήματος και τον τρόπο με τον οποίο εκτελούνται. Τέτοιου τύπου συναρτήσεις απαρτίζονται συνήθως από πολλές διαφορετικές υποσυναρτήσεις που συνυπολογίζονται συνήθως ως ένα σταθμισμένο άθροισμα, το επίπεδο πρότερης γνώσης που έχει ενσωματωθεί στη συνάρτηση αξιολόγησης είναι υψηλό [55-57].

$$ob\ fun\ 1$$

$$= \frac{1}{\left(V \cdot \left(1 - \frac{final_{route}}{40}\right) \cdot \left(1 - \sqrt{dV}\right) \cdot i_{sensors} \cdot \left(1 - \frac{vp2goal}{0.2}\right) \cdot \left(1 - near_{coef}\right) \cdot \left(\sqrt{1 - \frac{simulation_{steps}}{1400}}\right)\right)} \quad (4.2)$$

Όπου **V** ο μέσος όρος της μέσης ταχύτητας των δύο τροχών του οχήματος, $\boldsymbol{final_{route}}$ το μήκος της διαδρομής, $\boldsymbol{dV}$ ο απόλυτος μέσος όρος της διαφοράς των ταχυτήτων μεταξύ των δύο τροχών, $\boldsymbol{i_{sensors}}$ η ελάχιστη τιμή ενεργοποίησης από όλους τους αισθητήρες, $\boldsymbol{vp2goal}$ η απόσταση από το στόχο, $\boldsymbol{near_{coef}}$ είναι ένας συντελεστής με τιμή από 0 έως 1 που υπολογίζει πόσες φορές το όχημα προσέγγισε τα εμπόδια σε απόσταση μικρότερη ή ίση των 50 cm και $\boldsymbol{simulation_{steps}}$ το άθροισμα των βημάτων σε κάθε προσομοίωση. Όσο ελαχιστοποιείται η τιμή της συνάρτησης αξιολόγησης τόσο καλύτερος είναι ο ελεγκτής. Η συνάρτηση αυτή ενεργοποιείται μόνο όταν ισχύουν ταυτόχρονα τρεις συνθήκες: όταν το όχημα προσεγγίσει το στόχο σε απόσταση μικρότερη ή ίση του 0.18 m, όταν το όχημα έχει αποφύγει τα εμπόδια σε απόσταση μεγαλύτερη ή ίση των 0.25 m και όταν ο συντελεστής $\boldsymbol{near_{coef}}$ είναι μικρότερος ή ίσος του 0.4. Σε διαφορετική περίπτωση μπαίνει μια ποινή που αυξάνει την τιμή της συνάρτησης αξιολόγησης και έτσι ο ελεγκτής δεν μπορεί να είναι ο καλύτερος ούτε για το ίδιο το σωματίδιο ούτε συνολικά για τον πληθυσμό.

Ο σχεδιασμός της δεύτερης συνάρτησης αξιολόγησης έγινε με την εξής λογική: όσο μικρότερο χρόνο ενεργοποιούνται τα αισθητήρια τόσο καλύτερος γίνεται ο ελεγκτής. Για να επιτευχθεί αυτό, καταγράφονται οι τιμές και των οχτώ (8) αισθητήρων με τη μορφή πίνακα, στη συνέχεια αθροίζονται όσες τιμές είναι μικρότερες του ενός (1) που σημαίνει ότι ενεργοποιήθηκε ο αισθητήρας, το άθροισμα είναι και η τιμή της συνάρτησης αξιολόγησης, όσο ελαχιστοποιείται η τιμή , τόσο καλύτερος είναι ο ελεγκτής. Η συνάρτηση αξιολόγησης αυτή μπορεί να χαρακτηριστεί ως συγκεντρωτική γιατί βασίζεται σε ένα συνολικό κριτήριο επιτυχίας ή αποτυχίας του ρομποτικού οχήματος χωρίς να περιλαμβάνουν πληροφορία για το πώς επιτεύχθηκε ο συγκεκριμένος σκοπός ή στόχος. Στις συναρτήσεις αυτού του τύπου είναι μειωμένη η επίδραση του σχεδιαστή καθώς συγκεντρώνονται οι επιμέρους συμπεριφορές σε ένα τελικό αποτέλεσμα που τελικά αξιολογείται [58, 59].

$$ob\ fun\ 2 = m1.. \quad (4.3)$$

Όπου $m1$ το άθροισμα που αναφέρθηκε προηγουμένως. Η συνάρτηση αυτή ενεργοποιείται μόνο όταν το όχημα φτάσει στο σημείο-στόχο, σε διαφορετική περίπτωση επιβάλλεται ποινή.

4.3.3 Ανάπτυξη του κώδικα και επιλογή παραμέτρων του αλγορίθμου

Στη συνέχεια δημιουργήθηκε μια νέα βιβλιοθήκη συναρτήσεων και κώδικα με διαφορετικές λειτουργίες για την υλοποίηση της προσομοίωσης.

Κώδικες	
InitialiseHandleObject	Αρχικοποιεί το Pioneer 3pdx
PSO_for_fuzzy_with_sensors	Εφαρμόζει τον αλγόριθμο PSO.
main_for_fuzzy_with_sensors	Υλοποιεί το σύνολο του πειράματος

Πίνακας 4.7 Οι κώδικες που αναπτύχθηκαν για την υλοποίηση της προσομοίωσης

Συναρτήσεις	
create_x_from_fuzzy_with_sensors	Δημιουργεί το διάνυσμα που έχει τις τιμές των παραμέτρων των συναρτήσεων συμμετοχής από τις εισόδους και τις εξόδους του αρχικού ασαφούς ελεγκτή.
create_x_PSO_new	Δημιουργεί ένα νέο διάνυσμα που έχει τις τιμές των παραμέτρων των συναρτήσεων συμμετοχής από την είσοδο και τις εξόδους του νέου ασαφούς ελεγκτή που δημιουργείται σε κάθε επανάληψη.
build_pop	Δημιουργεί τον αρχικό πληθυσμό του σμήνους.
convert_fuzzy	Μετατρέπει τα διανύσματα των τιμών των παραμέτρων της εισόδου και της εξόδου του αρχικού πληθυσμού σε ασαφείς ελεγκτές.
convert_fuzzy_new	Μετατρέπει τα διανύσματα των τιμών των παραμέτρων της εισόδου και της εξόδου κάθε νέου ελεγκτή που δημιουργείται σε κάθε επανάληψη σε ασαφή ελεγκτή.
convert_best_fuzzy	Μετατρέπει τα διανύσματα των τιμών των παραμέτρων της εισόδου και της εξόδου του καλύτερου ελεγκτή σε ασαφή ελεγκτή.
ob_fun_for_fuzzy_with_sensors_	Αξιολογεί την επίδοση των σωματιδίων-ελεγκτών.
ob_fun_for_fuzzy_with_sensors_4	Αξιολογεί την επίδοση των σωματιδίων-ελεγκτών.
arena_with_obstacles	Υλοποιεί την προσομοίωση στο V-REP και καταγράφει τα δεδομένα.

Πίνακας 4.8 Οι συναρτήσεις που αναπτύχθηκαν για την υλοποίηση της προσομοίωσης

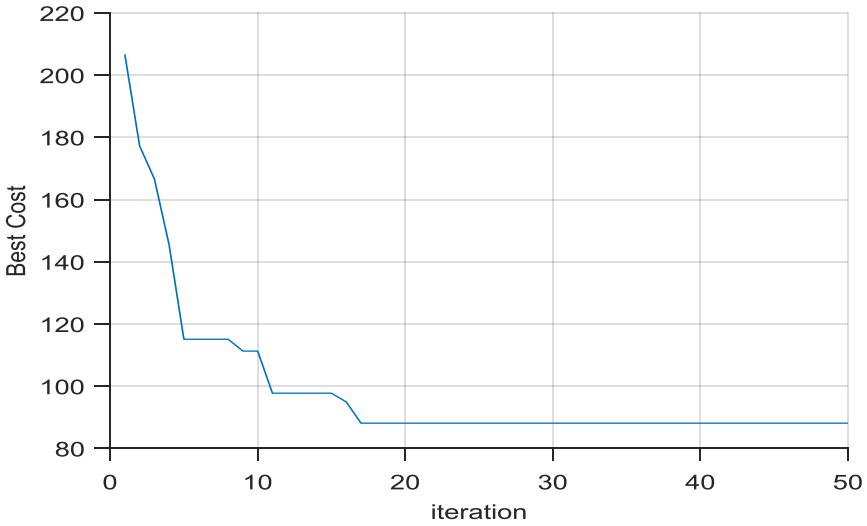
Η φιλοσοφία, και η ροή είναι οι ίδιες όπως και με την πρώτη προσομοίωση. Αρχικά με την συνάρτηση **create_x_from_fuzzy_with_sensors** δημιουργείται ένα διάνυσμα που έχει τις τιμές των παραμέτρων των συναρτήσεων συμμετοχής από τις εισόδους και τις εξόδους του ασαφούς ελεγκτή. Το σύνολο των παραμέτρων του διανύσματος αποτελεί και τις διαστάσεις του χώρου αναζήτησης, στην περίπτωση μας ο ελεγκτής έχει 95 παραμέτρους. Στη συνέχεια δημιουργείται ο αρχικός πληθυσμό με τη χρήση της συνάρτησης **build_pop** και επιλέγεται το μέγεθος του σμήνους, ίσο με τριάντα (35) σωματίδια. Η διασπορά των σωματιδίων στο χώρο αναζήτησης γίνεται σύμφωνα με τη σχέση: $x_{ij} = x_{min} + r_i(x_{max} - x_{min})$ όπου $i=1...n$, n το μέγεθος του πληθυσμού, r τυχαίος αριθμός μεταξύ [0,1], ώστε να υπάρχει όσο το δυνατόν μεγαλύτερη κάλυψη του χώρου. Ο πληθυσμός έχει την μορφή διανυσμάτων, οπότε με τη συνάρτηση **convert_fuzzy** τα μετατρέπουμε σε μορφή **.fis** δηλαδή σε ασαφείς ελεγκτές οι οποίοι αποτελούν τα σωματίδια του σμήνους. Για την περίπτωση αυτή η **ob_fun_for_fuzzy_with_sensors** είναι η συνάρτηση αξιολόγησης των σωματιδίων-ελεγκτών.

Έπειτα ακολουθεί ο κώδικας **PSO for fuzzy with sensors** στον οποίο εφαρμόζεται ο αλγόριθμος με παράγοντα περιορισμού (constriction factor) σχέση 2.21. Αρχικά επιλέγονται ο μέγιστος αριθμός επαναλήψεων και οι τιμές των παραμέτρων $\boldsymbol{\varphi_1 = 2,}$ $\boldsymbol{\varphi_2 = 2.03}$ για την γνωσιακή και την κοινωνική συνιστώσα, και **k=0.52**. Η επιλογή έγινε σύμφωνα με τη θεωρία και δίνεται μια μικρή βαρύτητα στην κοινωνική συνιστώσα ώστε να γίνει πιο γρήγορη αναζήτηση. Ακολουθούν οι αρχικοποιήσεις των θέσεων και των ταχυτήτων των σωματιδίων, η αρχική ταχύτητα είναι ίση με μηδέν, των προσωπικών καλύτερων θέσεων (personal best), και της συνολικά καλύτερης θέσης (global best). Το επόμενο στάδιο είναι η προσομοίωση της πρώτης επανάληψης για κάθε έναν από τους ελεγκτές στο V-REP, η καταγραφή των δεδομένων και η αξιολόγησή τους σύμφωνα με την συνάρτηση που αναφέρθηκε πιο πάνω. Στη συνέχεια γίνεται ενημέρωση των προσωπικών καλύτερων θέσεων (personal best), και της συνολικά καλύτερης θέσης (global best). Έπειτα ακολουθεί η επόμενη επανάληψη του αλγορίθμου με την ανανέωση της ταχύτητας και της θέσης κάθε σωματιδίου σύμφωνα με τις σχέσεις 2.11 & 2.21, γίνεται προσομοίωση κάθε ελεγκτή στο V-REP, ακολουθεί η αξιολόγηση και η ενημέρωση των προσωπικών καλύτερων θέσεων (personal best), και της συνολικά καλύτερης θέσης (global best). Ο αλγόριθμος επαναλαμβάνεται μέχρι την ικανοποίηση της συνθήκης τερματισμού, που στην περίπτωση αυτή είναι ο μέγιστος αριθμός επαναλήψεων που είναι ίσος με πενήντα (50).

4.3.4 Αποτελέσματα προσομοίωσης αποφυγής εμποδίων 1

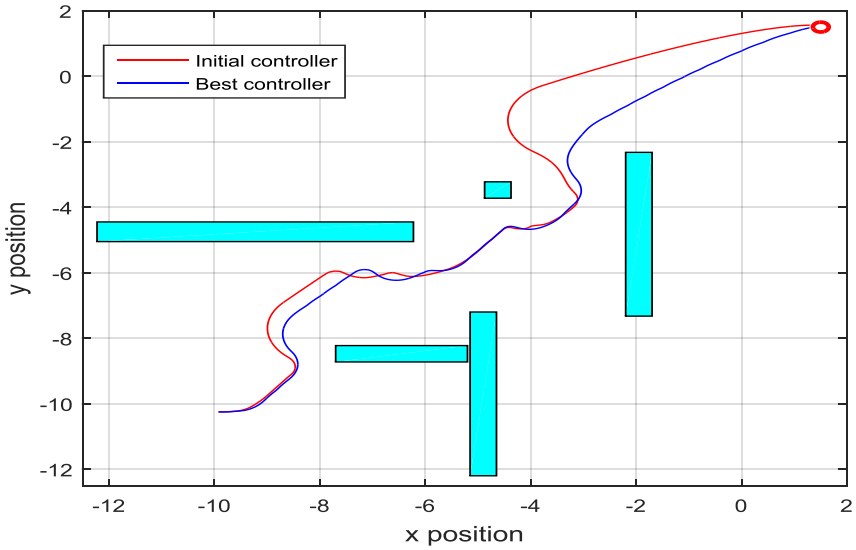
Στο σχήμα 4.22 παρουσιάζεται πως μεταβάλλεται η καλύτερη τιμή της συνάρτησης αξιολόγησης σε κάθε επανάληψη. Η αρχική καλύτερη τιμή είναι **206.68** και η συνολικά καλύτερη τιμή είναι **88.12.** Η βελτίωση είναι της τάξης **του 57.36%** Παρατηρούμε ότι

μετά τη $18_{η}$ επανάληψη δεν υπάρχει περαιτέρω βελτίωση. Η σχετικά γρήγορη εύρεση της βέλτιστης θέσης οφείλεται στο ότι η κοινωνική συμπεριφορά του σωματιδίου είναι μεγαλύτερη από την γνωσιακή.



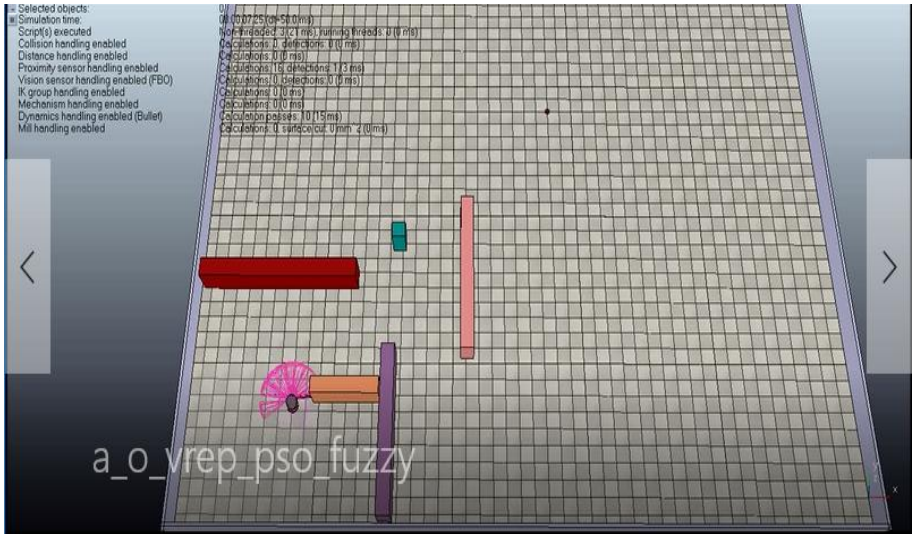
Σχήμα 4.22 Μεταβολή συνάρτησης αξιολόγησης

Στο σχήμα 4.23 παρουσιάζεται ο χώρος που κινήθηκε το όχημα, οι θέσεις των εμποδίων και η διαδρομή που ακολούθησε με τον αρχικό και με τον βέλτιστο ελεγκτή. Φαίνεται ξεκάθαρα ότι το όχημα με τον βελτιωμένο ελεγκτή προσεγγίζει τα εμπόδια σε μικρότερη απόσταση αυτό όμως έχει ως αποτέλεσμα να κάνει πιο «κλειστές στροφές» και να διανύει μικρότερη συνολική απόσταση.

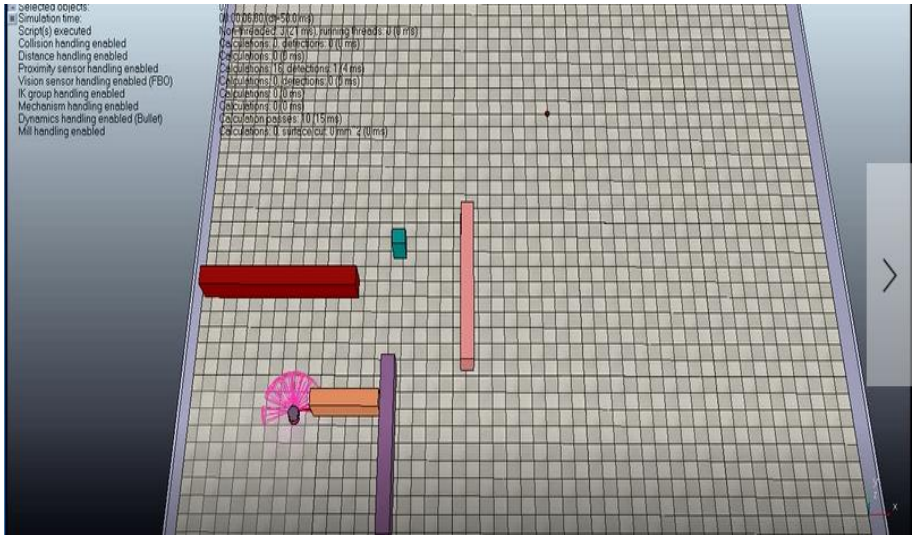


Σχήμα 4.23 Διαδρομή οχήματος με τον αρχικό και τον καλύτερο ελεγκτή

Στη συνέχεια και συγκεκριμένα στις εικόνες 4.2 έως 4.19 παρουσιάζονται μια σειρά από στιγμιότυπα της προσομοίωσης προκειμένου να αναδειχθούν οι διαφορές στη συμπεριφορά του οχήματος.

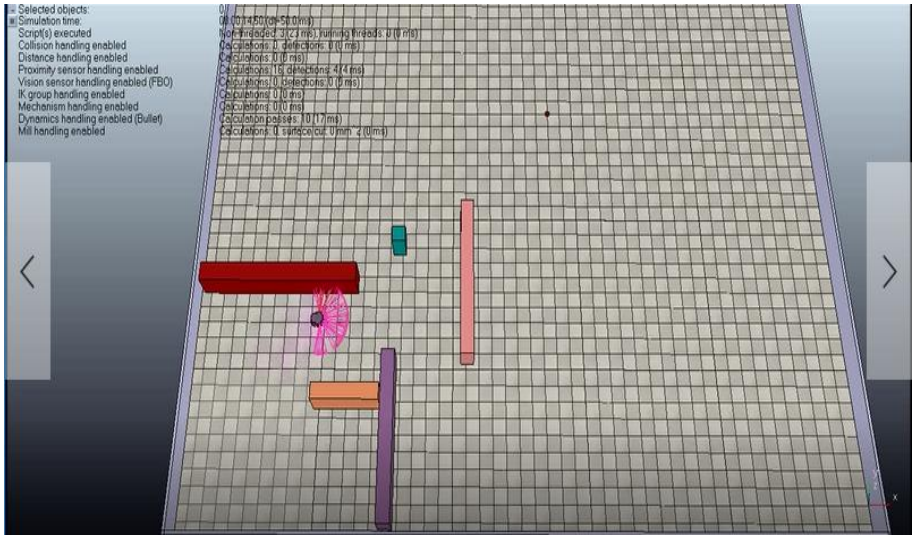


Εικόνα 4.2 Αρχικός ελεγκτής 7 sec

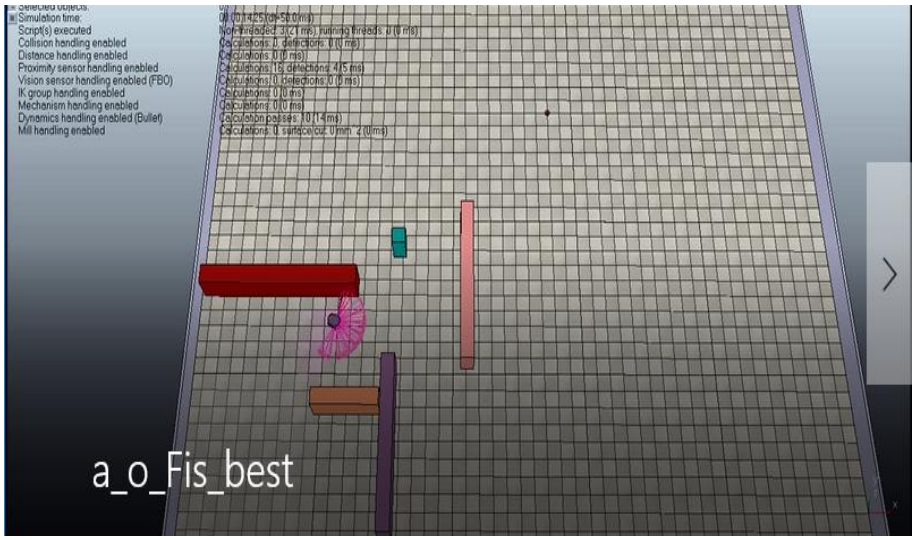


Εικόνα 4.3 Βέλτιστος ελεγκτής 7 sec

Στις εικόνες 4.2 και 4.3 παρατηρούμε ότι το όχημα με τον βέλτιστο ελεγκτή πλησιάζει πιο κοντά στο εμπόδιο

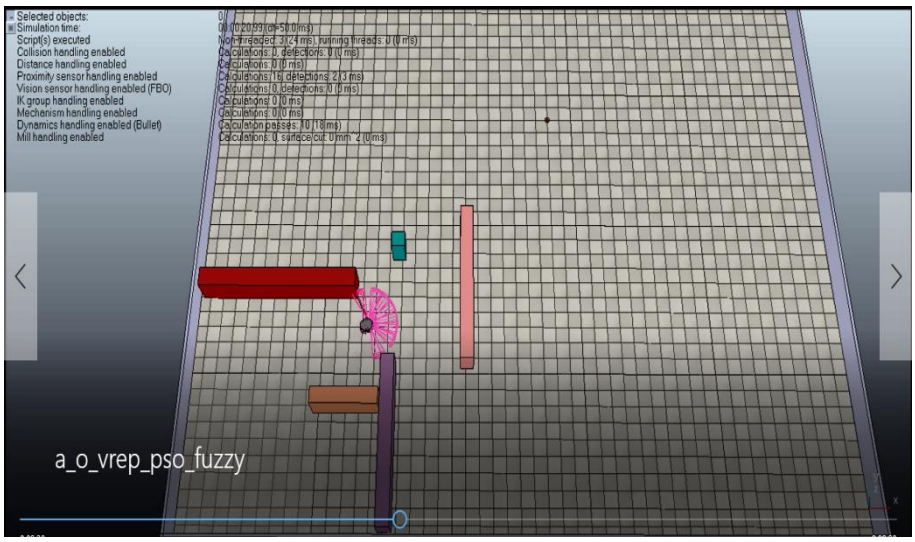


Εικόνα 4.4 Αρχικός ελεγκτής 14 sec

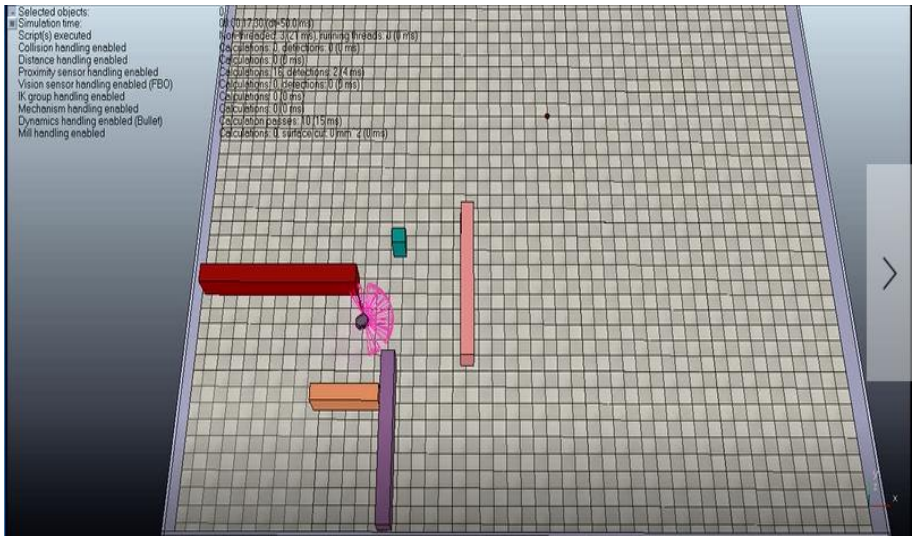


Εικόνα 4.5 Βέλτιστος ελεγκτής 14 sec

Στις εικόνες 4.4 και 4.5 παρατηρούμε ότι το όχημα με τον βέλτιστο ελεγκτή στα 14 sec έχει αρχίσει να προηγείται του αρχικού.

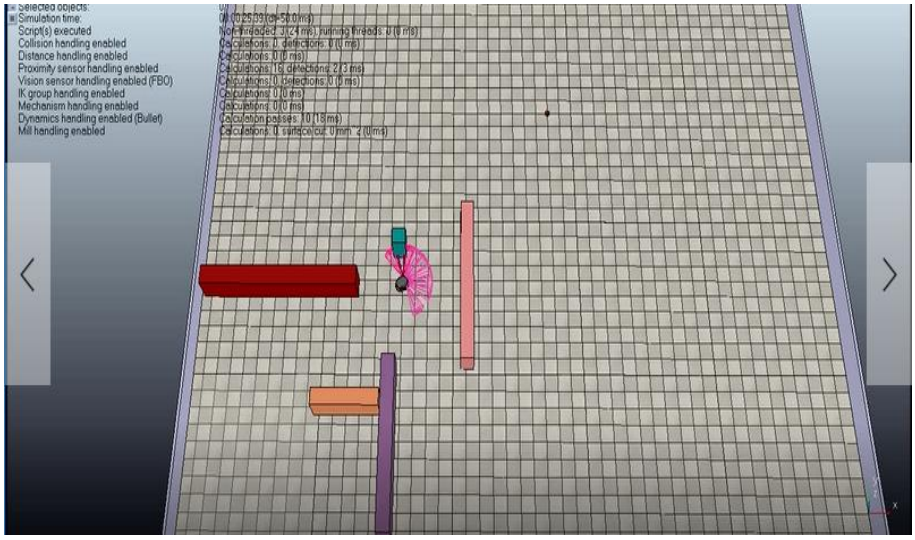


Εικόνα 4.6 Αρχικός ελεγκτής 20 sec

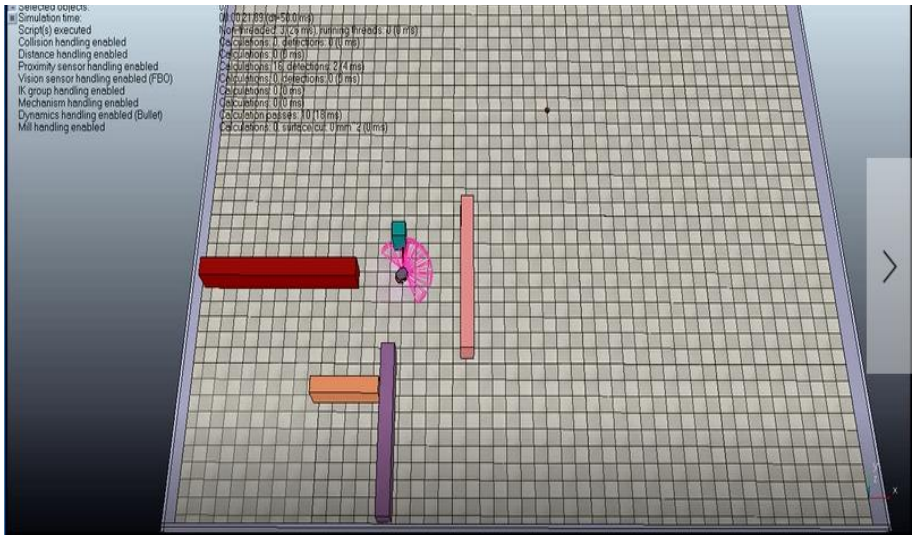


Εικόνα 4.7 Βέλτιστος ελεγκτής 17 sec

Στις εικόνες 4.6 και 4.7 παρατηρούμε ότι το όχημα με τον βέλτιστο ελεγκτή έχει αποφύγει το κόκκινο εμπόδιο στα 17 sec ενώ με τον αρχικό ελεγκτή στα 20 sec

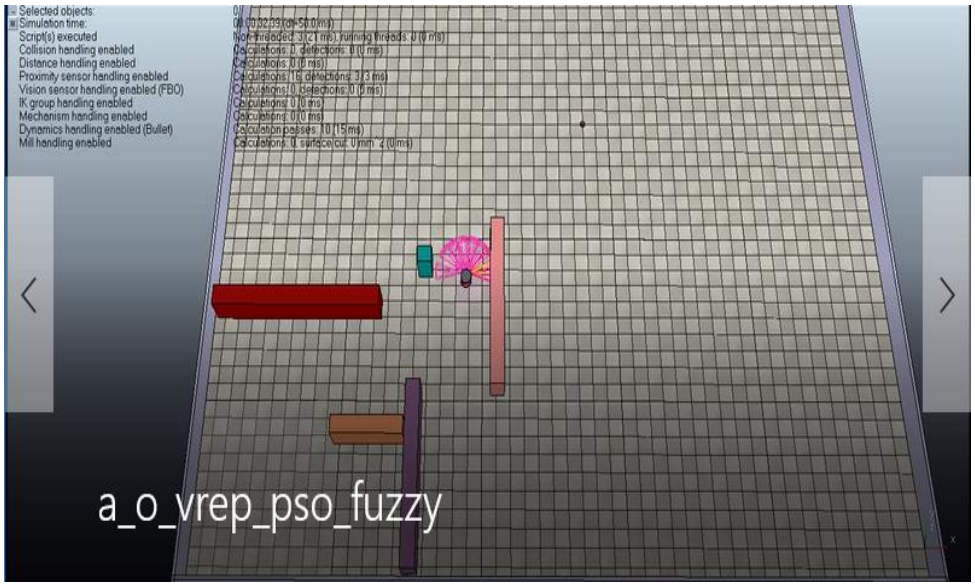


Εικόνα 4.8 Αρχικός ελεγκτής 25 sec

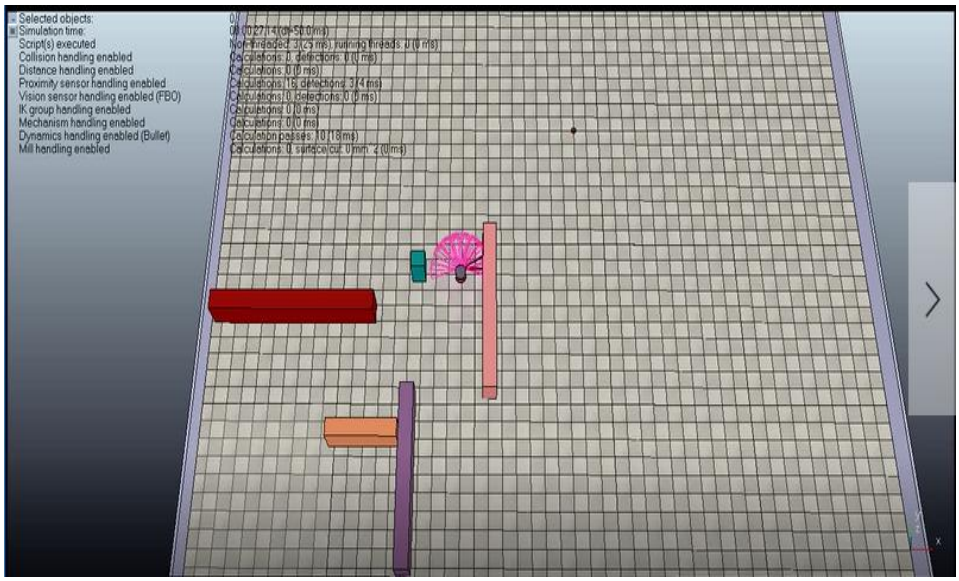


Εικόνα 4.9 Βέλτιστος ελεγκτής 22 sec

Στις εικόνες 4.8 και 4.9 παρατηρούμε ότι το όχημα με τον βέλτιστο ελεγκτή αποφεύγει το κυανό εμπόδιο στα 22 sec και από μεγαλύτερη απόσταση σε σχέση με τον αρχικό ελεγκτή που το αποφεύγει στα 25 sec

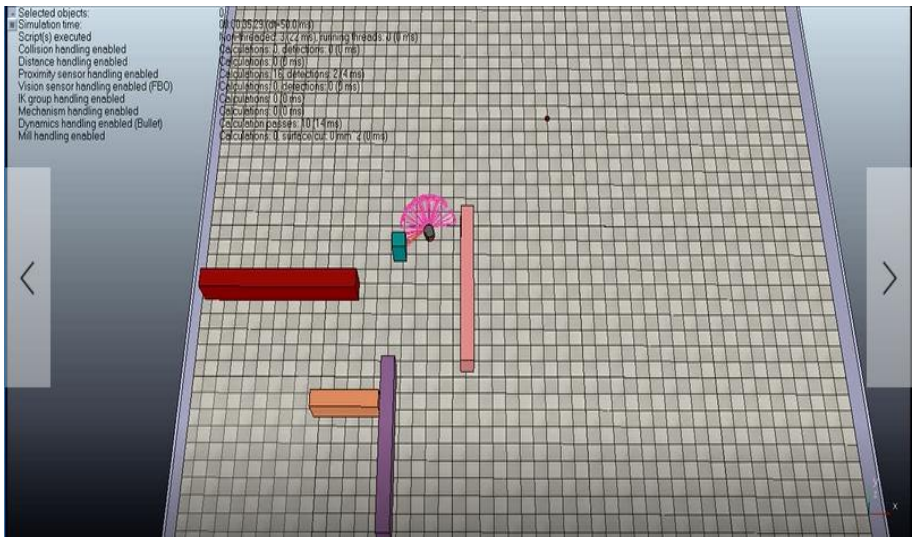


Εικόνα 4.10 Αρχικός ελεγκτής 32 sec

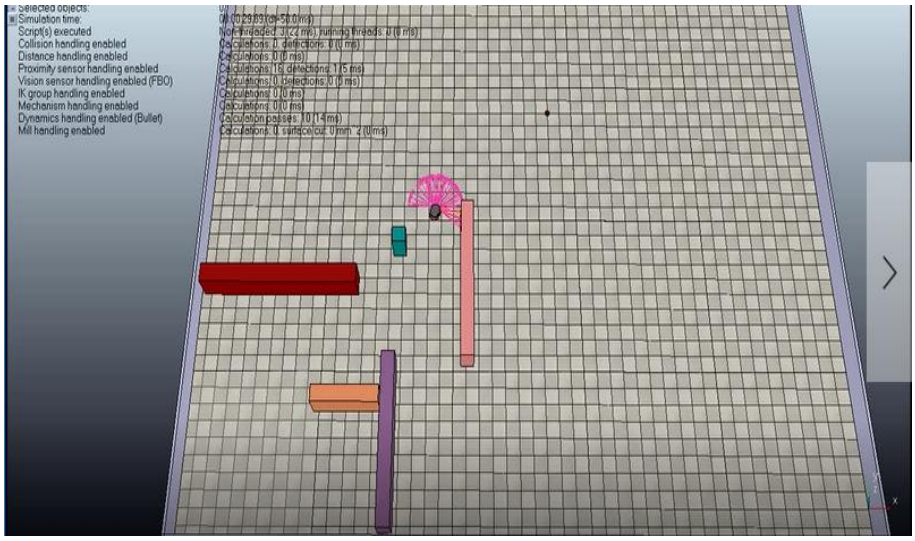


Εικόνα 4.11 Βέλτιστος ελεγκτής 27 sec

Στις εικόνες 4.10 και 4.11 παρατηρούμε ότι το όχημα με τον βέλτιστο ελεγκτή βρίσκεται παράλληλα με το εμπόδιο στα δεξιά του στα 27 sec, σε αντίθεση με το όχημα με τον αρχικό ελεγκτή που βρίσκεται στα 32 sec και το προσεγγίζει σε κοντινότερη απόσταση.

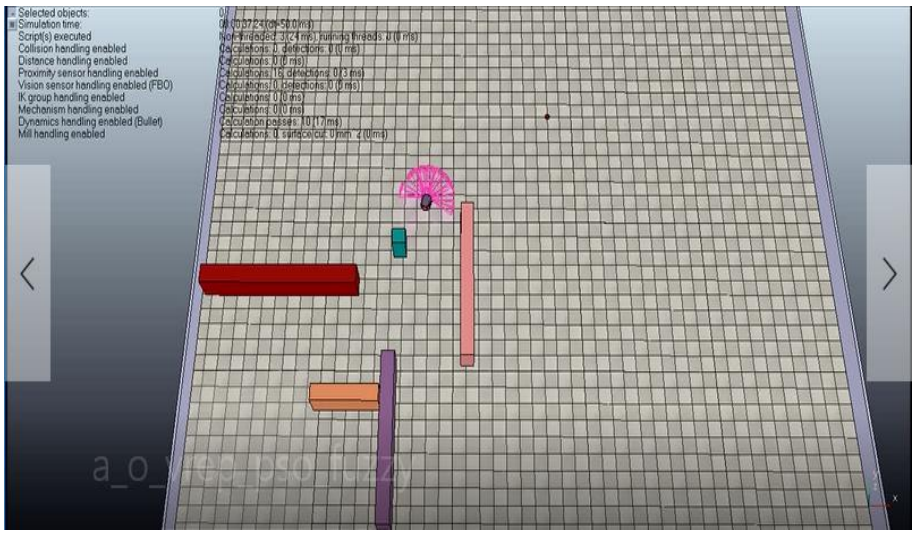


Εικόνα 4.12 Αρχικός ελεγκτής 35 sec

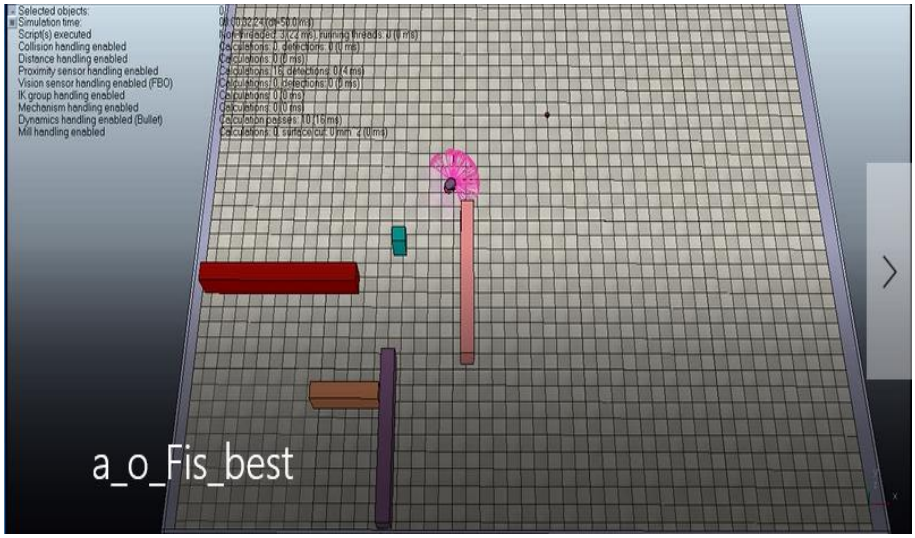


Εικόνα 4.13 Βέλτιστος ελεγκτής 30 sec

Στις εικόνες 4.12 και 4.13 τα οχήματα βρίσκονται 3 sec μετά την προηγούμενη θέση τους, παρατηρούμε ότι το όχημα με τον αρχικό ελεγκτή αποφεύγει το εμπόδιο πιο απότομα από το όχημα με το βέλτιστο ελεγκτή και παίρνει τη στροφή πιο ανοιχτά σε αντίθεση με το άλλο όχημα που θα πλησιάσει το εμπόδιο ώστε να πάρει πιο κλειστή στροφή ώστε να διανύσει μικρότερη διαδρομή ως το τελικό σημείο-στόχο.



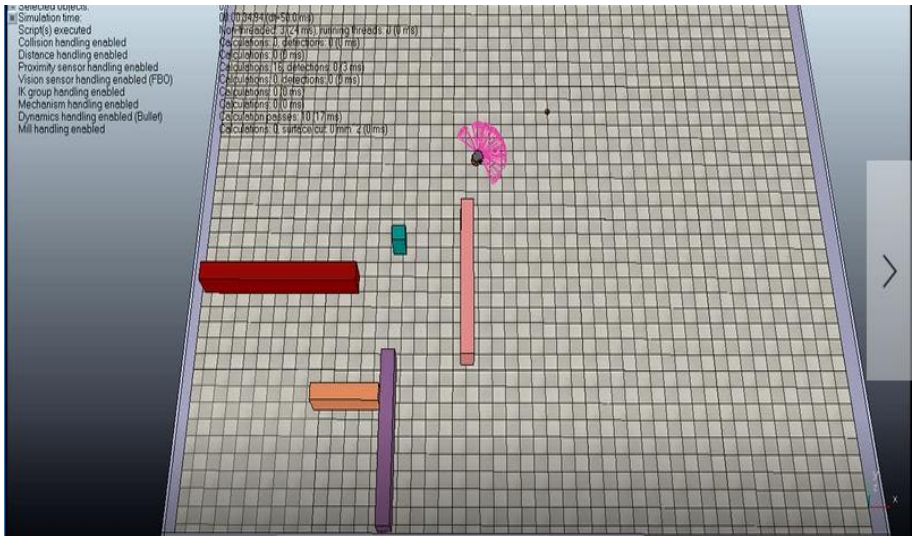
Εικόνα 4.14 Αρχικός ελεγκτής 37 sec



Εικόνα 4.15 Βέλτιστος ελεγκτής 32 sec

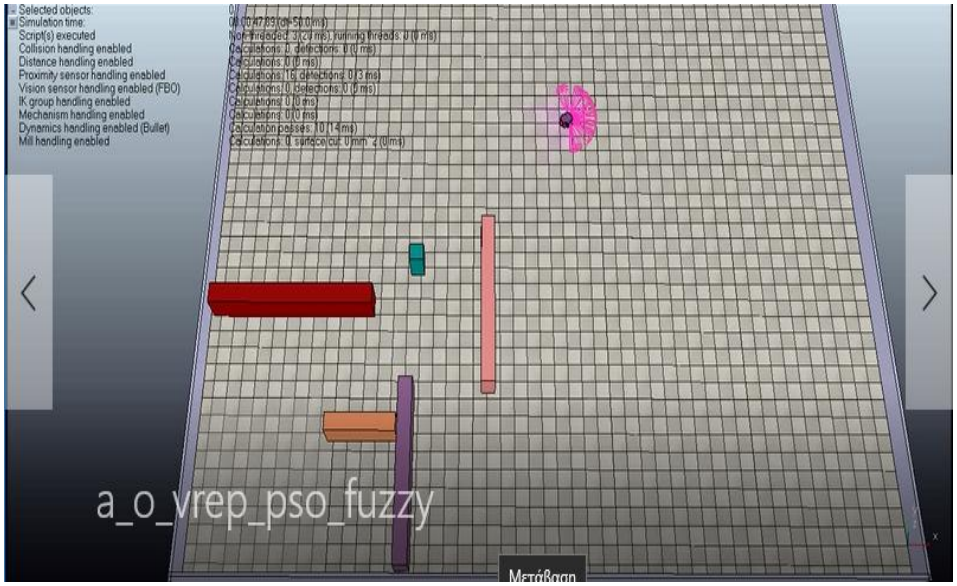
Στις εικόνες 4.14 και 4.15 τα οχήματα βρίσκονται 2 sec μετά την προηγούμενη θέση τους, παρατηρούμε καλύτερα ότι το όχημα με τον αρχικό ελεγκτή αποφεύγει το εμπόδιο πιο έντονα από το όχημα με το βέλτιστο ελεγκτή και παίρνει τη στροφή πιο ανοιχτά σε αντίθεση με το άλλο όχημα που θα πλησιάσει το εμπόδιο ώστε να πάρει πιο κλειστή στροφή με αποτέλεσμα να διανύσει μικρότερη διαδρομή ως το τελικό σημείο-στόχο.

Εικόνα 4.16 Αρχικός ελεγκτής 42 sec

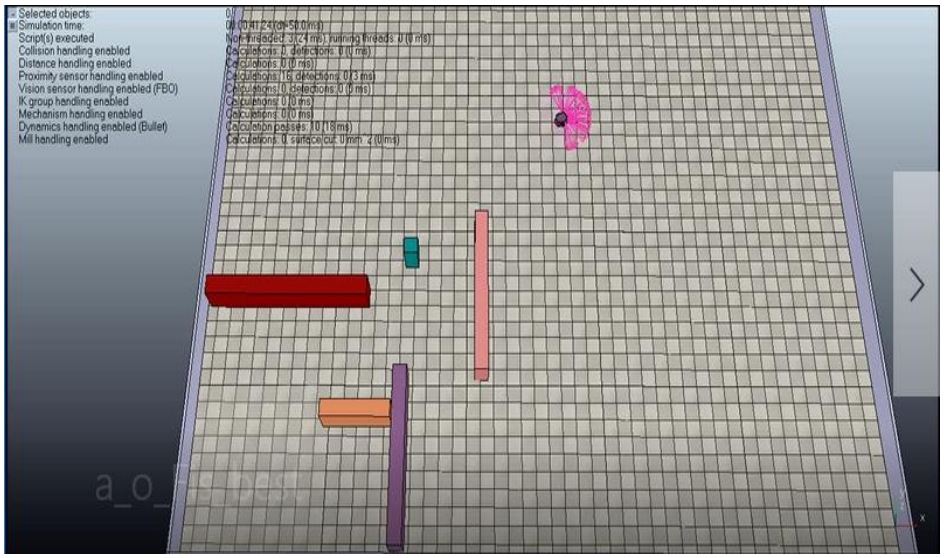


Εικόνα 4.17 Βέλτιστος ελεγκτής 35 sec

Στις εικόνες 4.16 και 4.17 τα οχήματα βρίσκονται στην τελική ευθεία, στην ίδια περίπου απόσταση πριν το τελικό σημείο στόχο με το όχημα με τον αρχικό ελεγκτή να έχει χρόνο 42 sec ενώ με τον βέλτιστο 35 sec.



Εικόνα 4.18 Αρχικός ελεγκτής 48 sec



Εικόνα 4.19 Βέλτιστος ελεγκτής 42 sec

Στις εικόνες 4.18 και 4.19 τα οχήματα έχουν φτάσει στο τελικό σημείο στόχο με το όχημα με τον αρχικό ελεγκτή να έχει χρόνο 48 sec ενώ με τον βέλτιστο 42 sec.

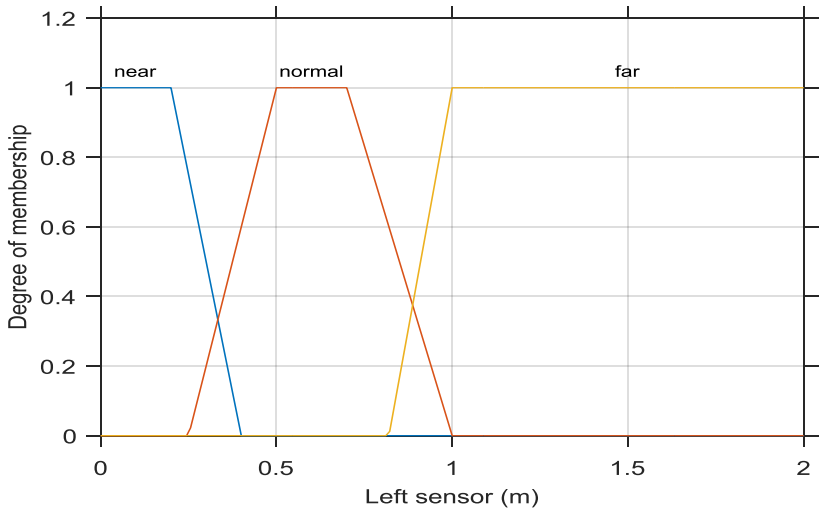
Στον πίνακα 4.9 παρουσιάζονται τα αποτελέσματα από την ανάλυση των καταγεγραμμένων δεδομένων, καθώς και μια σύγκριση της συμπεριφοράς του καλύτερου ελεγκτή που προέκυψε από τη διαδικασία βελτιστοποίησης σε σχέση με τον αρχικό ασαφή ελεγκτή.

	Αρχικός ασαφής ελεγκτής	**Καλύτερος ασαφής ελεγκτής**	**Ποσοστό μεταβολής %**
Επανάληψη	1	17	
Σωματίδιο	1	5	
Τιμή συνάρτησης αξιολόγησης	206.68	88.12	-57.36
Βήματα προσομοίωσης (steps)	998	794	-20.44
Χρόνος διαδρομής (sec)	48	42	-12.50
Μήκος διαδρομής (m)	21.12	18.81	-10.94
Μέση ταχύτητα (m/s)	4.46	5.03	+12.78
dV (m/s)	1.28	1.72	+34.37
Ελάχιστη ενεργοποίηση αισθητήρων (m)	0.62	0.57	-8.06
Μέσος όρος ενεργοποίησης αισθητήρων (m)	0.93	0.95	+2.15
Near_coef<=0.5 (m)	0.00	0.00	0.00

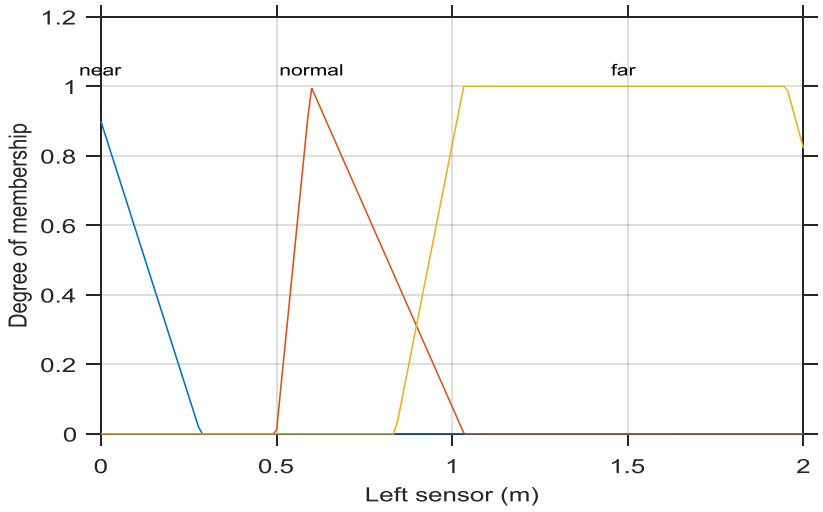
Πίνακας 4.9 Δεδομένα προσομοίωσης

Συμπερασματικά ο βελτιωμένος ελεγκτής σε σχέση με τον αρχικό κινείται με μεγαλύτερη ταχύτητα, αποφεύγει τα εμπόδια σε μικρότερη απόσταση από τον αρχικό αλλά όχι μικρότερη από 0.5 m ώστε να υπάρχει το ενδεχόμενο σύγκρουσης και αυτό του δίνει το πλεονέκτημα να «παίρνει πιο κλειστές στροφές» και να διανύει μικρότερη διαδρομή, με αποτέλεσμα να φτάνει στο στόχο πιο γρήγορα, στρίβει πιο απότομα από τον αρχικό αλλά αυτό είναι απόρροια της αυξημένης ταχύτητας και του ότι προσπαθεί να φτάσει στο στόχο σε μικρότερο χρονικό διάστημα

Στα παρακάτω σχήματα παρουσιάζονται πως έχουν διαφοροποιηθεί οι τιμές των παραμέτρων των συναρτήσεων συμμετοχής των εισόδων και των εξόδων του ελεγκτή.

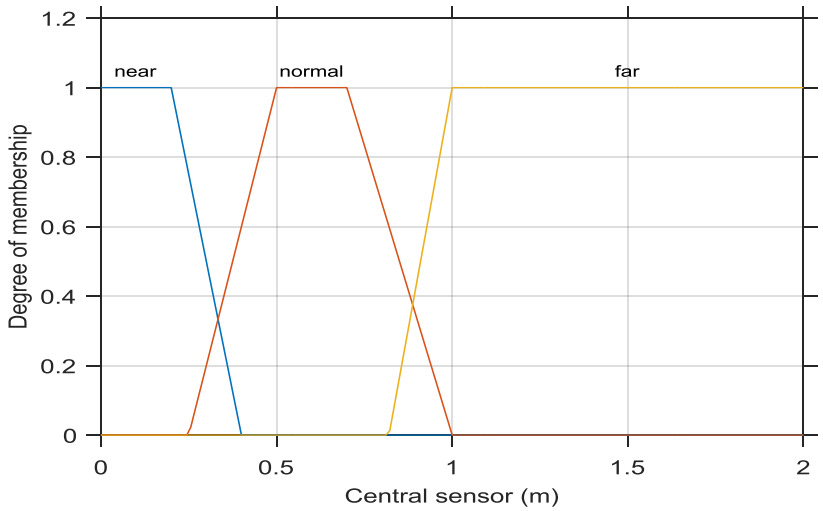


Σχήμα 4.24 Είσοδος «Αισθητήρας αριστερά» αρχικού ελεγκτή

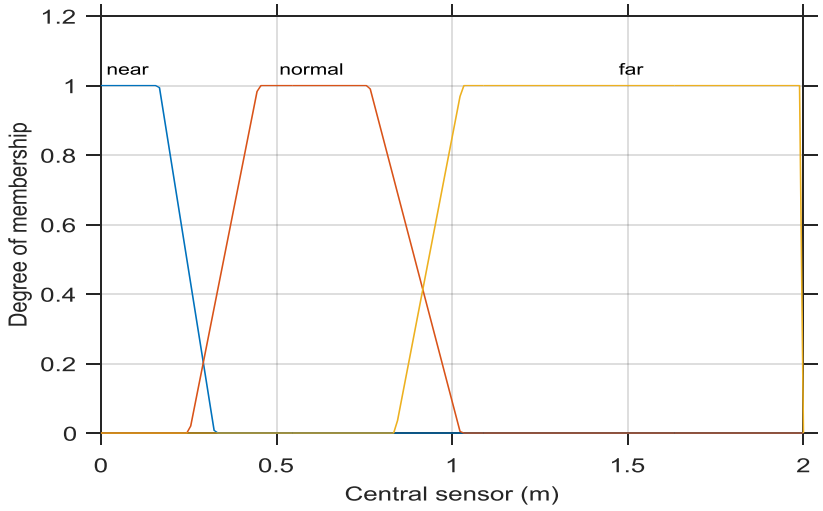


Σχήμα 4. 25 Είσοδος «Αισθητήρας αριστερά» βέλτιστου ελεγκτή

Στο ασαφές σύνολο «αισθητήρας αριστερά», παρατηρούμε ότι η συνάρτηση συμμετοχής «κοντά» έχει μεταφερθεί αριστερά μειώνοντας το εύρος της, και υπάρχει ένα κενό μεταξύ 0.3 και 0.5 m. Εξήγηση γι'αυτό είναι ότι το όχημα ποτέ δεν βρέθηκε να προσεγγίζει εμπόδιο κάτω από το 0.5 m και προσπαθεί να προσαρμόσει τις συναρτήσεις συμμετοχής σε νέο εύρος για το ασαφές σύνολο. Επίσης η συνάρτηση συμμετοχής «κανονικά» έχει μεταφερθεί λίγο δεξιά έχει γίνει τριγωνική από τραπεζοειδής αυξάνοντας την ασάφεια του συνόλου. Η συνάρτηση συμμετοχής «μακριά» έχει μειώσει την περιοχή όπου είναι σίγουρα «μακριά» αυξάνοντας και αυτή με τη σειρά της την ασάφεια του συνόλου.

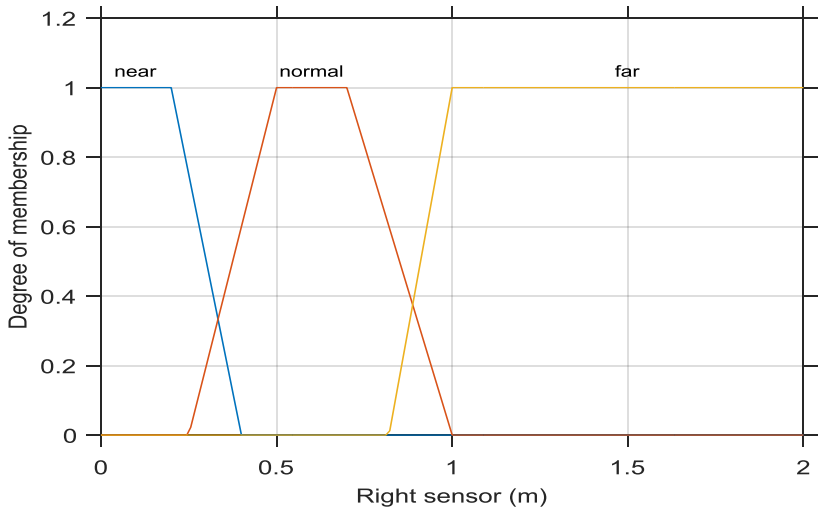


Σχήμα 4.26 Είσοδος «Αισθητήρας κέντρο» αρχικού ελεγκτή

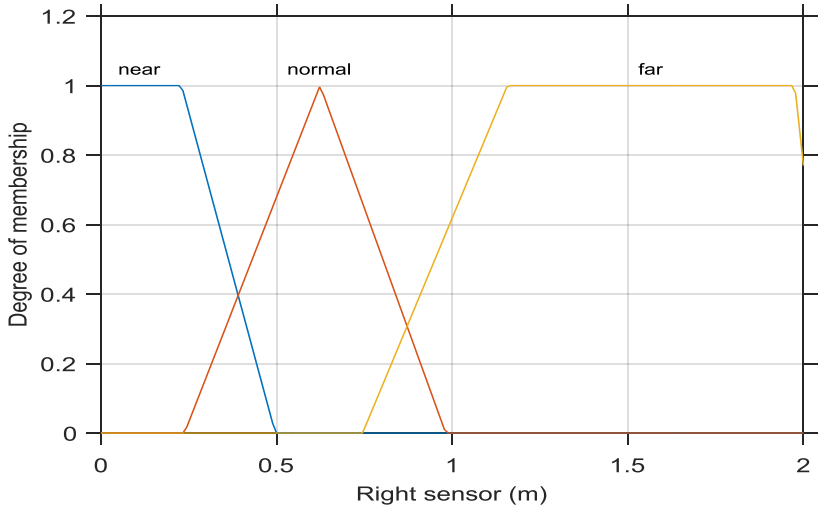


Σχήμα 4.27 Είσοδος «Αισθητήρας κέντρο» βέλτιστου ελεγκτή

Στο ασαφές σύνολο «αισθητήρας κεντρικά», παρατηρούμε ότι η συνάρτηση συμμετοχής «κοντά» έχει μεταφερθεί αριστερά μειώνοντας το εύρος ενώ η συνάρτηση συμμετοχής «κανονικά» έχει αυξήσει το εύρος της και προς τα δεξιά και προς τα αριστερά. Η συνάρτηση συμμετοχής «μακριά» έχει μειώσει την περιοχή όπου είναι σίγουρα «μακριά» αυξάνοντας και αυτή με τη σειρά της την ασάφεια του συνόλου.

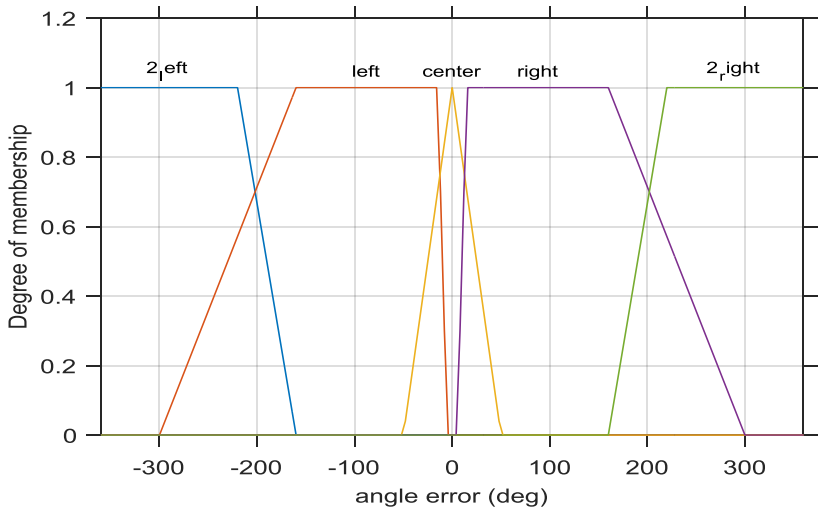


Σχήμα 4.28 Είσοδος «Αισθητήρας δεξιά» αρχικού ελεγκτή

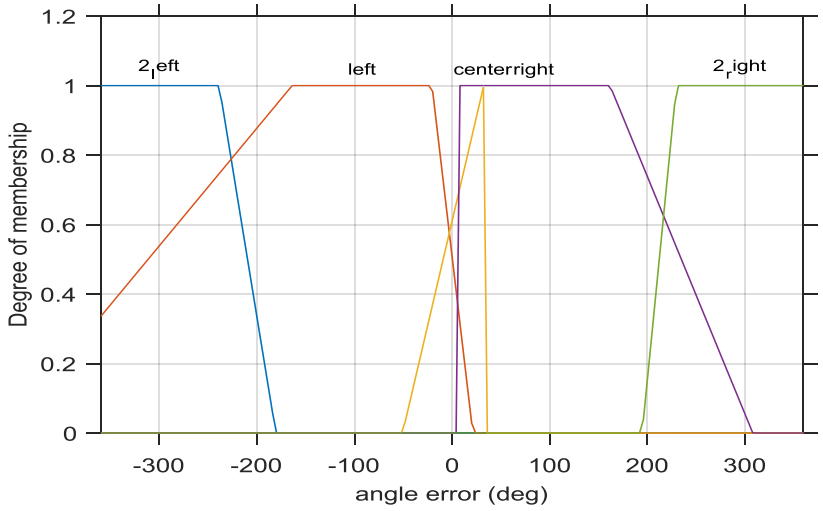


Σχήμα 4.29 Είσοδος «Αισθητήρας δεξιά» βέλτιστου ελεγκτή

Στο ασαφές σύνολο «αισθητήρας δεξιά», παρατηρούμε ότι η συνάρτηση συμμετοχής «κοντά» έχει μετακινηθεί δεξιά αυξάνοντας το εύρος της, η συνάρτηση συμμετοχής «κανονικά» τριγωνική από τραπεζοειδής αυξάνοντας την ασάφεια του συνόλου. Η συνάρτηση συμμετοχής «μακριά» έχει μειώσει την περιοχή όπου είναι σίγουρα «μακριά» αυξάνοντας και αυτή με τη σειρά της την ασάφεια του συνόλου.

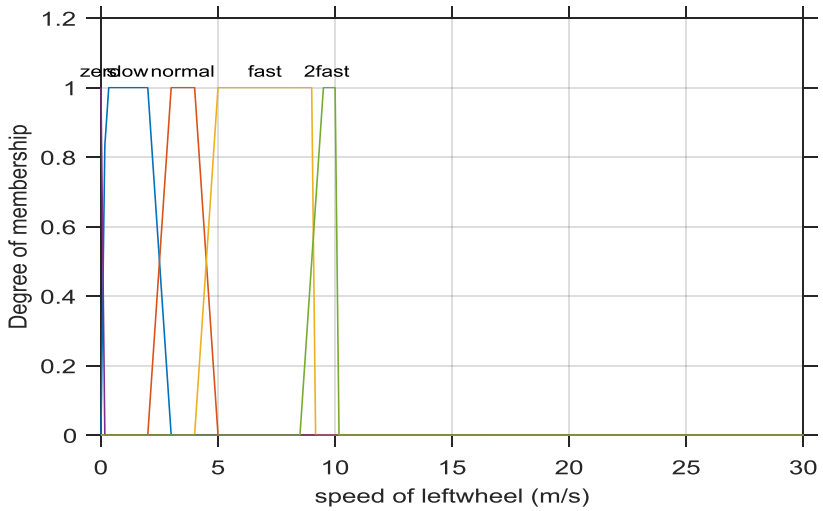


Σχήμα 4.30 Είσοδος «Γωνία σφάλματος» αρχικού ελεγκτή

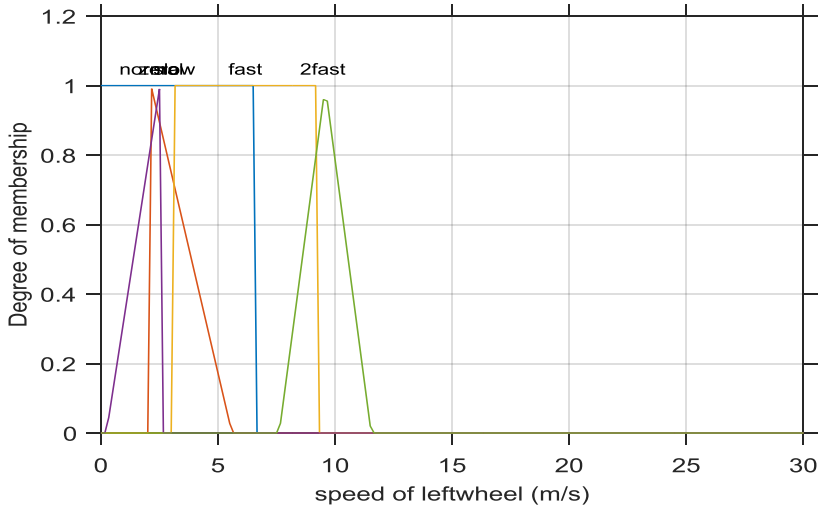


Σχήμα 4.31 Είσοδος «Γωνία σφάλματος» βέλτιστου ελεγκτή

Στο ασαφές σύνολο «Γωνία σφάλματος», παρατηρούμε ότι οι συναρτήσεις «αριστερά» και «δεξιά» έχουν αυξήσει αρκετά το εύρος έχοντας υπερκαλύψει σχεδόν ολοκληρωτικά την συνάρτηση «κέντρο» αυξάνοντας την ασάφεια του συνόλου.

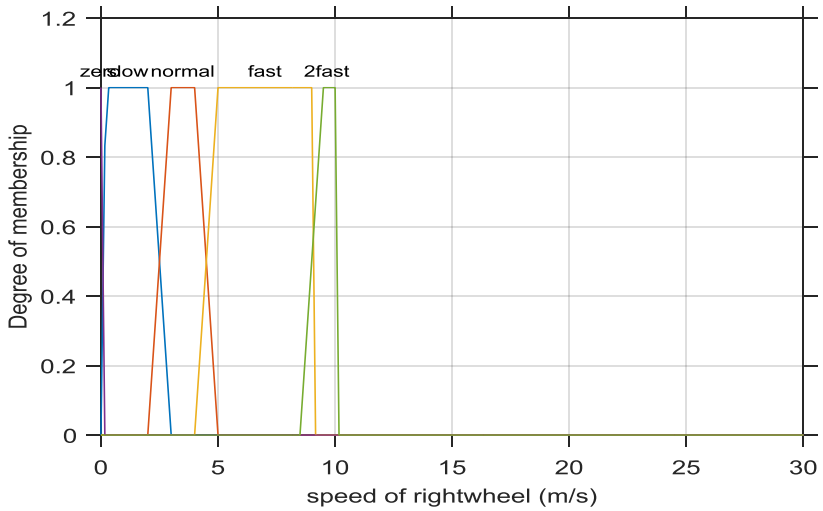


Σχήμα 4.32 Έξοδος «ταχύτητα αριστερού τροχού» αρχικού ελεγκτή

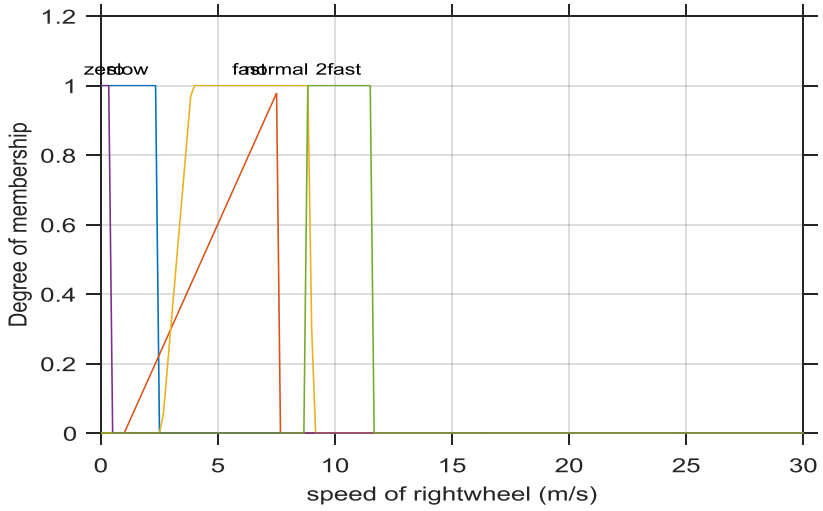


Σχήμα 4.33 Έξοδος «ταχύτητα αριστερού τροχού» βέλτιστου ελεγκτή

Στο ασαφές σύνολο «ταχύτητα αριστερού τροχού», παρατηρούμε ότι οι συναρτήσεις τείνουν να καλύψουν τις διπλανές τους, αυτό συμβαίνει γιατί ο ελεγκτής προσπαθεί να γίνει πιο απλός με λιγότερες. συναρτήσεις συμμετοχής. Άλλη βασική διαφοροποίηση είναι ότι η συνάρτηση συμμετοχής «πολύ γρήγορά» αυξάνει το μέγιστο όριό της από 10 m/s που ήταν αρχικά σε περίπου 12 m/s, αυτό συμβαίνει γιατί το όχημα προσπαθεί να κινηθεί γρηγορότερα.



Σχήμα 4. 34 Έξοδος «ταχύτητα δεξιού τροχού» αρχικού ελεγκτή

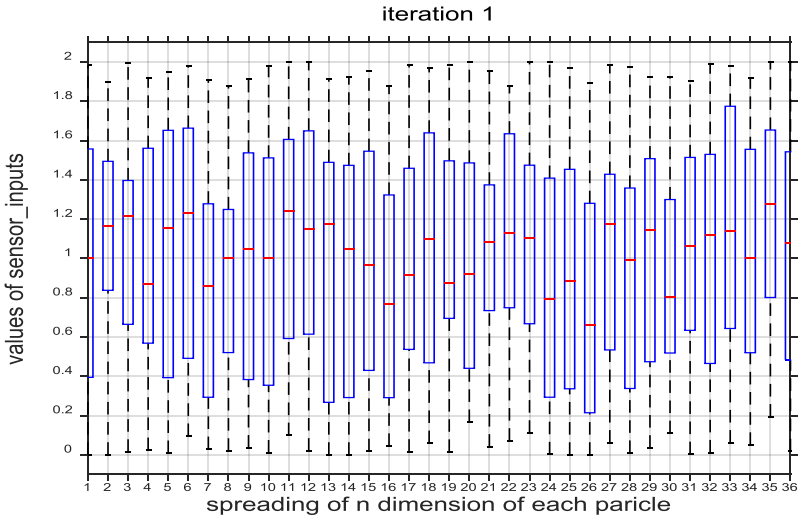


Σχήμα 4.35 Έξοδος «ταχύτητα δεξιού τροχού» βέλτιστου ελεγκτή

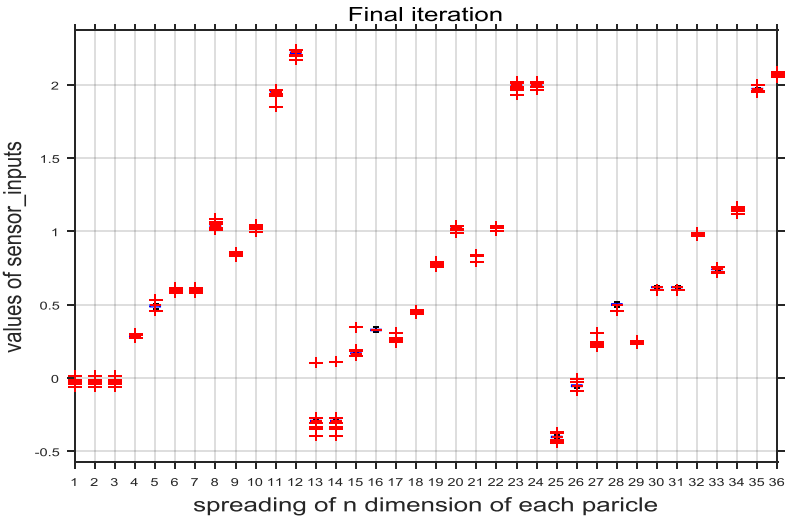
Στο ασαφές σύνολο «ταχύτητα δεξιού τροχού», παρατηρούμε ότι οι συναρτήσεις τείνουν να καλύψουν τις διπλανές τους, αυτό συμβαίνει γιατί ο ελεγκτής προσπαθεί να γίνει πιο απλός με λιγότερες. συναρτήσεις συμμετοχής. Άλλη βασική διαφοροποίηση είναι ότι η συνάρτηση συμμετοχής «πολύ γρήγορά» αυξάνει το μέγιστο όριό της από 10 m/s που ήταν αρχικά σε περίπου 12 m/s, αυτό συμβαίνει γιατί το όχημα προσπαθεί να κινηθεί γρηγορότερα.

Στο σχήμα 4.36 και 4.37 παρουσιάζεται η κατανομή των παραμέτρων των συναρτήσεων των συμμετοχής των εισόδων των αισθητηρίων των ελεγκτών κατά τη δημιουργία τη του πληθυσμού και στην τελική επανάληψη αντίστοιχα. Παρατηρούμε ότι ενώ στην πρώτη επανάληψη η κατανομή είναι μεγάλη και έχει εύρος σχεδόν από 0 έως 2

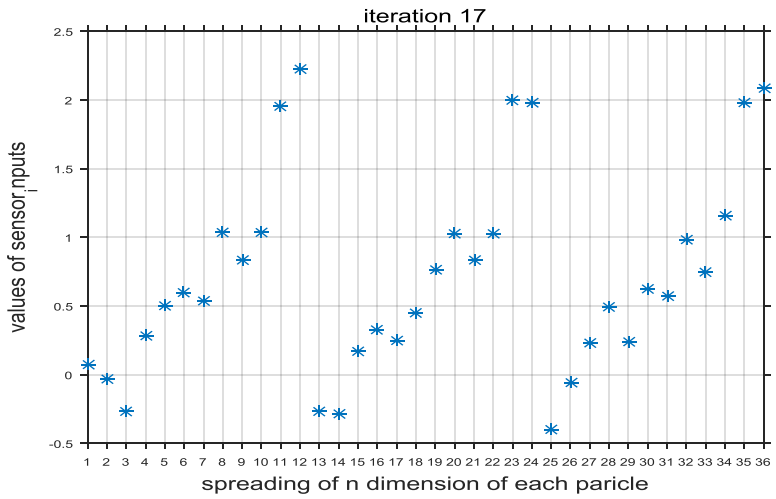
σε κάθε διάσταση, που ήταν και το επιθυμητό, στην τελική επανάληψη τείνουν να συγκλίνουν σε κάποιο συγκεκριμένο αριθμό. Στο σχήμα 4.38 παρουσιάζονται οι παράμετροι των συναρτήσεων συμμετοχής του βέλτιστου ελεγκτή μετά από 17 επαναλήψεις. Παρατηρούμε πως όλοι οι παράμετροι των σωματιδίων έχουν συγκλίνει στη βέλτιστη λύση.



Σχήμα 4.36 Κατανομή παραμέτρων του συνόλου των σωματιδίων για τις εισόδους «Αισθητήρας αριστερά», «Αισθητήρας κέντρο», «Αισθητήρας δεξιά»

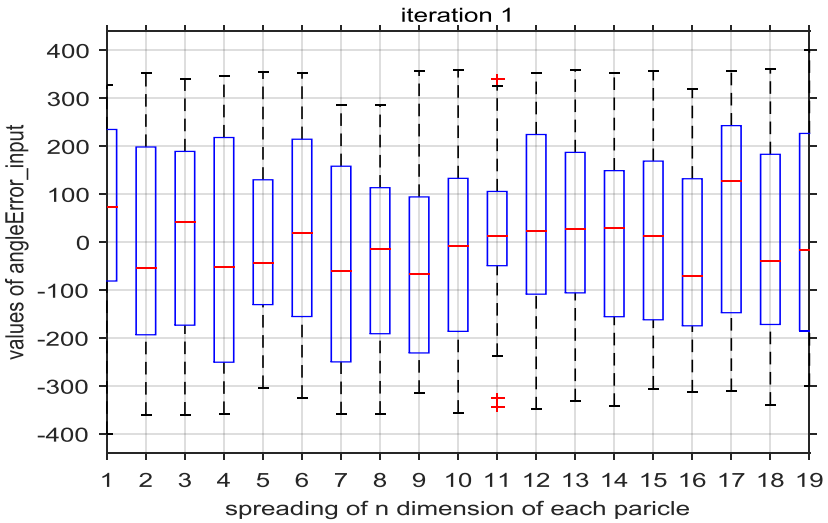


Σχήμα 4.37 Κατανομή παραμέτρων του συνόλου των σωματιδίων για τις εισόδους «Αισθητήρας αριστερά», «Αισθητήρας κέντρο», «Αισθητήρας δεξιά»

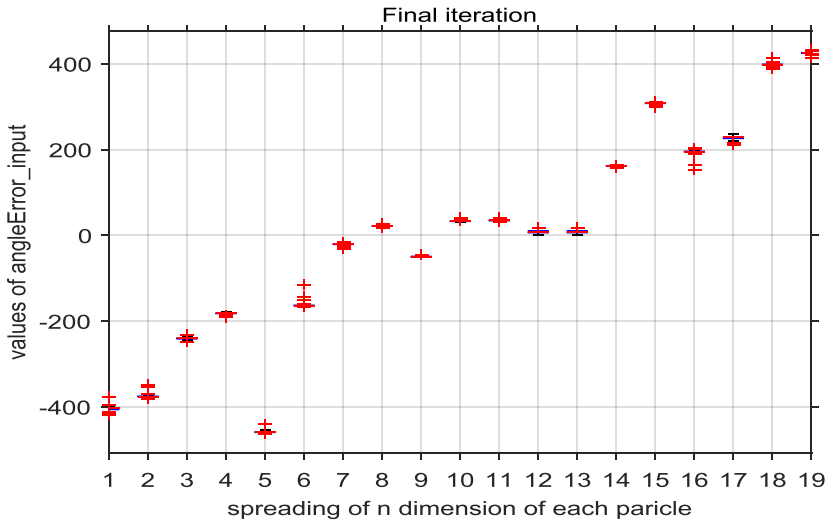


Σχήμα 4.38 Κατανομή παραμέτρων του βέλτιστου ελεγκτή για τις εισόδους «Αισθητήρας αριστερά», «Αισθητήρας κέντρο», «Αισθητήρας δεξιά»

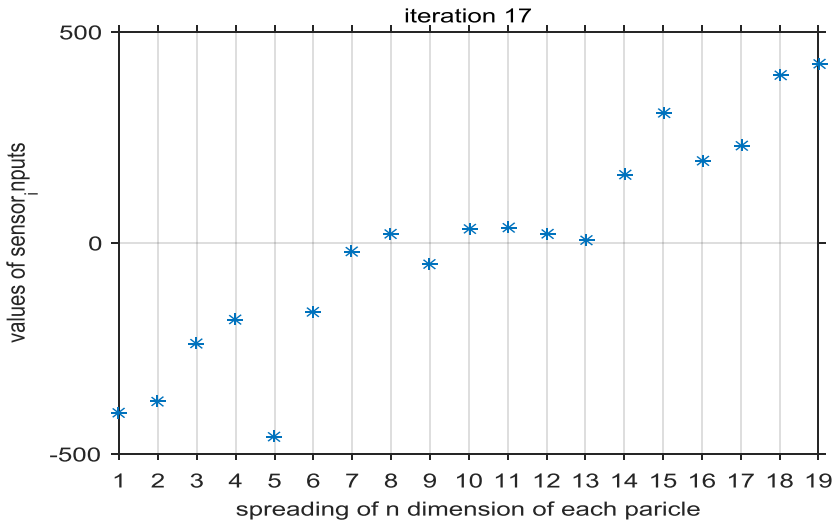
Στο σχήμα 4.39 και 4.40 παρουσιάζεται η κατανομή των παραμέτρων των συναρτήσεων συμμετοχής της εισόδου της γωνίας σφάλματος των ελεγκτών κατά τη δημιουργία τη του πληθυσμού και στην τελική επανάληψη αντίστοιχα. Παρατηρούμε ότι ενώ στην πρώτη επανάληψη η διασπορά είναι μεγάλη και έχει εύρος σχεδόν από -360 έως 360 σε κάθε διάσταση, που ήταν και το επιθυμητό, στην τελική επανάληψη έχουν συγκλίνει σε σχεδόν σε κάποιο συγκεκριμένο αριθμό. Στο σχήμα 4.41 παρουσιάζονται οι παράμετροι των συναρτήσεων συμμετοχής του βέλτιστου ελεγκτή μετά από 17 επαναλήψεις. Παρατηρούμε πως όλοι οι παράμετροι των σωματιδίων έχουν συγκλίνει στη βέλτιστη λύση.



Σχήμα 4.39 Κατανομή παραμέτρων του συνόλου των σωματιδίων για την είσοδο «Γωνία σφάλματος»

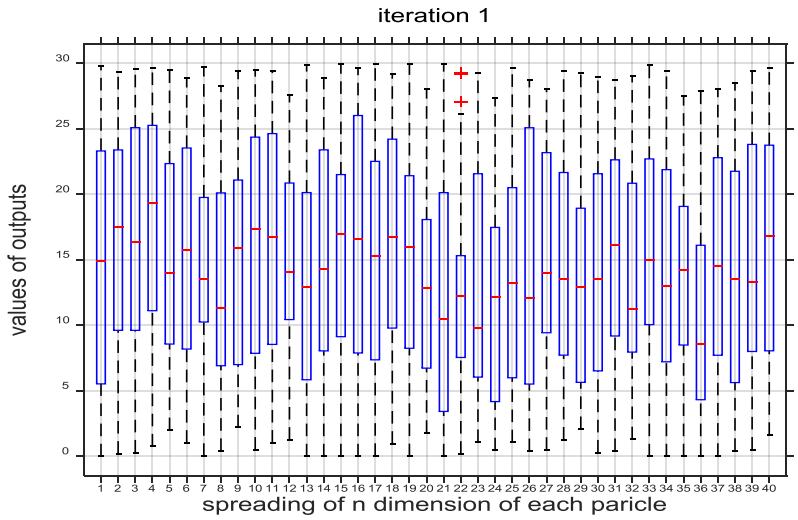


Σχήμα 4.40 Κατανομή παραμέτρων του συνόλου των σωματιδίων για την είσοδο «Γωνία σφάλματος»

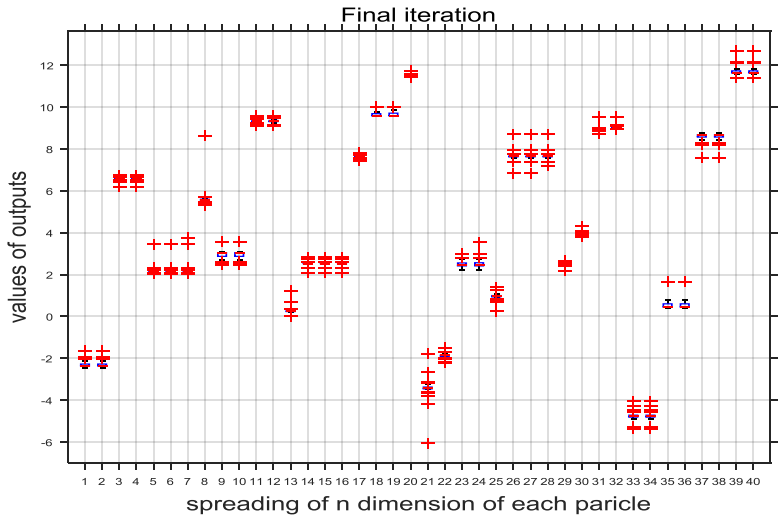


Σχήμα 4.41 Κατανομή παραμέτρων του βέλτιστου ελεγκτή για την είσοδο «Γωνία σφάλματος»

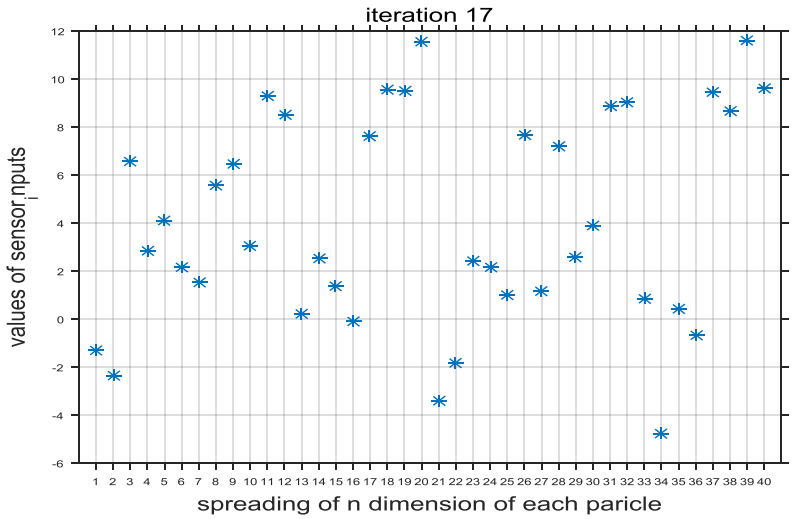
Στο σχήμα 4.42 και 4.43 παρουσιάζεται η κατανομή των παραμέτρων των συναρτήσεων των συμμετοχής των εξόδων των ταχυτήτων των ελεγκτών κατά τη δημιουργία του πληθυσμού και στην τελική επανάληψη αντίστοιχα. Παρατηρούμε ότι ενώ στην πρώτη επανάληψη η διασπορά είναι μεγάλη και έχει εύρος σχεδόν από 0 έως 30 σε κάθε διάσταση, που ήταν και το επιθυμητό, στην τελική επανάληψη έχουν συγκλίνει σε σχεδόν σε κάποιο συγκεκριμένο αριθμό. Στο σχήμα 4.44 παρουσιάζονται οι παράμετροι των συναρτήσεων συμμετοχής του βέλτιστου ελεγκτή μετά από 17 επαναλήψεις. Παρατηρούμε πως όλοι οι παράμετροι των σωματιδίων έχουν συγκλίνει στη βέλτιστη λύση.



Σχήμα 4.42 Κατανομή παραμέτρων του συνόλου των σωματιδίων των εξόδων «Ταχύτητα αριστερού τροχού» και «Ταχύτητα δεξιού τροχού»



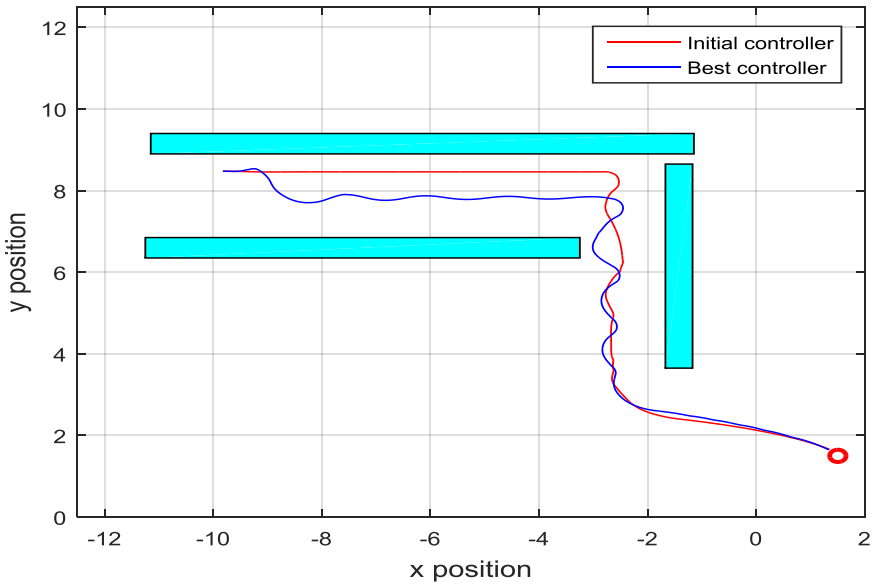
Σχήμα 4.43 Κατανομή παραμέτρων του συνόλου των σωματιδίων των εξόδων «Ταχύτητα αριστερού τροχού» και «Ταχύτητα δεξιού τροχού»



Σχήμα 4.44 Κατανομή παραμέτρων του βέλτιστου ελεγκτή των εξόδων «Ταχύτητα αριστερού τροχού» και «Ταχύτητα δεξιού τροχού»

4.3.5 Αποτελέσματα προσομοίωσης αποφυγής εμποδίων 2

Στη συνέχεια έγινε σύγκριση του αρχικού ελεγκτή με τον βέλτιστο σε νέο περιβάλλον με διαφορετικά εμπόδια και νέο τελικό-σημείο στόχο, στο σχήμα 4.45 παρουσιάζονται οι διαδρομές που ακολούθησε το όχημα με τους δύο ελεγκτές.



Σχήμα 4.45 Διαδρομή οχήματος με τον αρχικό και τον καλύτερο ελεγκτή

Στον πίνακα 4.10 παρουσιάζονται τα αποτελέσματα από την ανάλυση των καταγεγραμμένων δεδομένων, καθώς και μια σύγκριση της συμπεριφοράς του καλύτερου ελεγκτή που προέκυψε από τη διαδικασία βελτιστοποίησης σε σχέση με τον αρχικό ασαφή ελεγκτή.

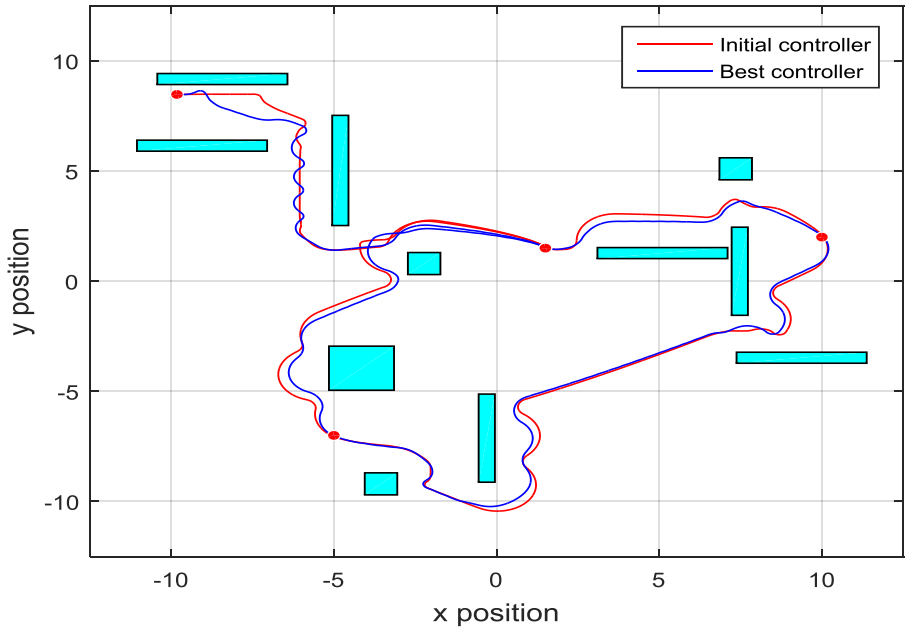
	Αρχικός ασαφής ελεγκτής	**Καλύτερος ασαφής ελεγκτής**	**Ποσοστό μεταβολής %**
Βήματα προσομοίωσης (steps)	1316	860	-34.65
Χρόνος διαδρομής (sec)	67	45	-32.83
Μήκος διαδρομής (m)	17.13	17.51	+2.21
Μέση ταχύτητα (m/s)	2.72	4.50	+65.44
dV (m/s)	0.54	1.75	+224.07
Ελάχιστη ενεργοποίηση αισθητήρων (m)	0.28	0.09	-67.85
Μέσος όρος ενεργοποίησης αισθητήρων (m)	0.60	0.84	+40.00
Near_coef<=0.5 (m)	0.47	0.10	-78.72

Πίνακας 4.10 Δεδομένα προσομοίωσης 2

Το όχημα με τον βέλτιστο ελεγκτή κινείται με μεγαλύτερη ταχύτητα, στρίβει πιο απότομα για να αποφύγει πιο γρήγορα τα εμπόδια ,πλησιάζει όταν χρειαστεί πολύ κοντά το εμπόδιο για να κάνει μικρότερη διαδρομή αλλά γενικά κινείται σε μεγαλύτερη απόσταση από τα εμπόδια, και όλη αυτή η συμπεριφορά έχει σαν αποτέλεσμα να φτάνει στο σημείο στόχο πιο γρήγορα.

4.3.6 Αποτελέσματα προσομοίωσης αποφυγής εμποδίων 3

Στη συνέχεια έγινε σύγκριση του αρχικού ελεγκτή με τον βέλτιστο σε άλλο περιβάλλον με διαφορετικά εμπόδια και νέο τελικό-σημείο στόχο, στο σχήμα 4.46 παρουσιάζονται οι διαδρομές που ακολούθησε το όχημα με τους δύο ελεγκτές.



Σχήμα 4.46 Διαδρομή οχήματος με τον αρχικό και τον καλύτερο ελεγκτή

Στον πίνακα 4.11 παρουσιάζονται τα αποτελέσματα από την ανάλυση των καταγεγραμμένων δεδομένων, καθώς και μια σύγκριση της συμπεριφοράς του καλύτερου ελεγκτή που προέκυψε από τη διαδικασία βελτιστοποίησης σε σχέση με τον αρχικό ασαφή ελεγκτή.

	Αρχικός ασαφής ελεγκτής	**Καλύτερος ασαφής ελεγκτής**	**Ποσοστό μεταβολής %**
Βήματα προσομοίωσης (steps)	3638	3049	-16.19
Χρόνος διαδρομής (sec)	184	153	-16.84
Μήκος διαδρομής (m)	73.84	72.23	-2.18
Μέση ταχύτητα (m/s)	4.24	5.00	+17.92
dV (m/s)	1.10	1.65	+50.00
Ελάχιστη ενεργοποίηση αισθητήρων (m)	0.30	0.14	-53.33
Μέσος όρος ενεργοποίησης αισθητήρων (m)	0.89	0.93	+4.49
Near_coef<=0.5 (m)	0.06	0.03	-50.00

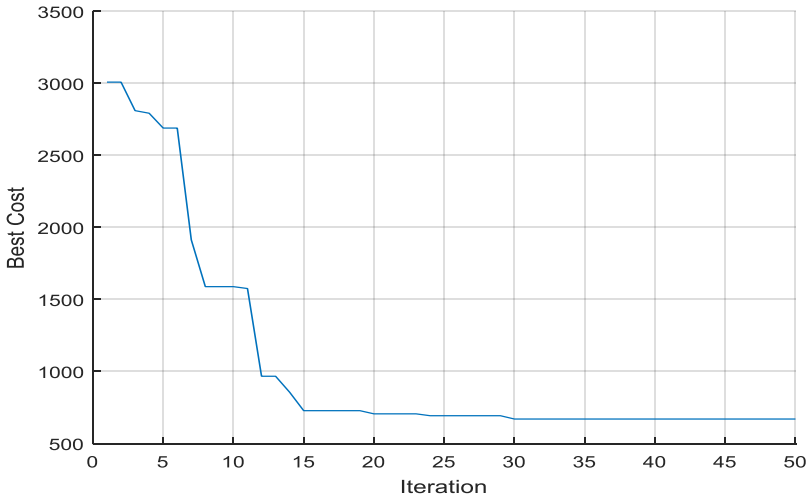
Πίνακας 4.11 Δεδομένα προσομοίωσης 3

Το όχημα με τον βέλτιστο ελεγκτή κινείται με μεγαλύτερη ταχύτητα, στρίβει πιο απότομα για να αποφύγει πιο γρήγορα τα εμπόδια ,πλησιάζει όταν χρειαστεί πολύ κοντά το εμπόδιο για να κάνει μικρότερη διαδρομή αλλά γενικά κινείται σε μεγαλύτερη απόσταση από τα εμπόδια, και όλη αυτή η συμπεριφορά έχει σαν αποτέλεσμα να φτάνει στο σημείο στόχο πιο γρήγορα

Συμπερασματικά ο βέλτιστος ελεγκτής παρουσιάζει ευρωστία καθώς έχει την ίδια συμπεριφορά και σε διαφορετικά περιβάλλοντα δοκιμής.

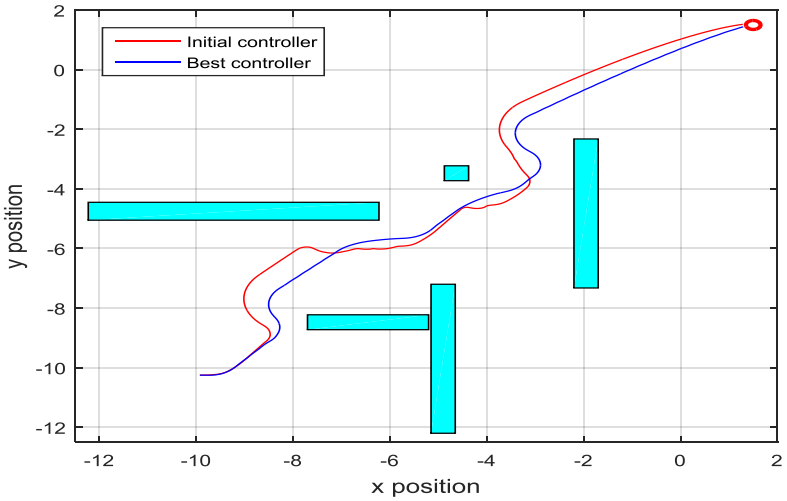
4.3.7 Αποτελέσματα προσομοίωσης με τη δεύτερη συνάρτηση αξιολόγησης

Στο σχήμα 4.47 παρουσιάζεται πως μεταβάλλεται η καλύτερη τιμή της συνάρτησης αξιολόγησης σε κάθε επανάληψη. Η αρχική καλύτερη τιμή είναι **3009** και η συνολικά καλύτερη τιμή είναι **670** Η μείωση είναι της τάξης του **77.73%** Παρατηρούμε ότι μετά τη $30_{η}$ επανάληψη δεν υπάρχει περαιτέρω βελτίωση. Η σχετικά γρήγορη εύρεση της βέλτιστης θέσης οφείλεται στο ότι η κοινωνική συμπεριφορά του σωματιδίου είναι μεγαλύτερη από την γνωσιακή.



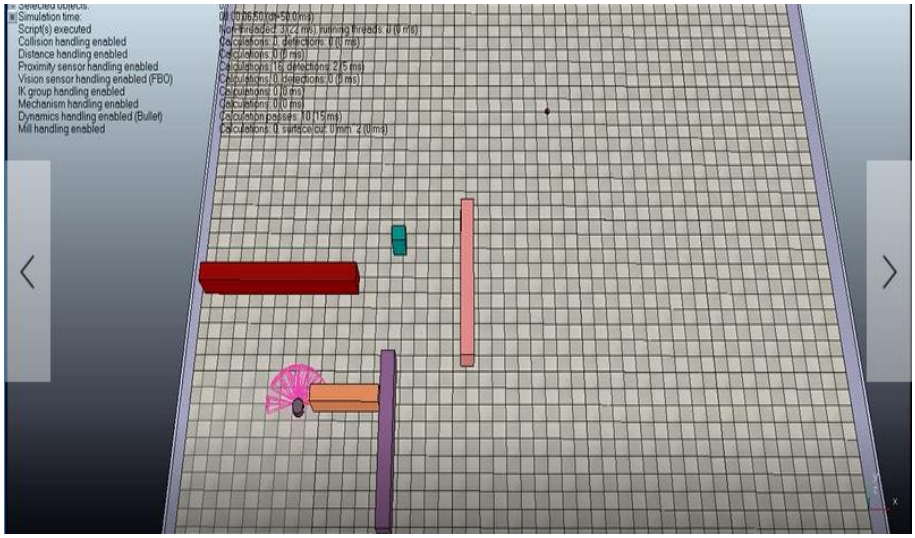
Σχήμα 4.47 Μεταβολή συνάρτησης αξιολόγησης

Στο σχήμα 4.48 παρουσιάζεται ο χώρος που κινήθηκε το όχημα, οι θέσεις των εμποδίων και η διαδρομή που ακολούθησε με τον αρχικό και με τον βέλτιστο ελεγκτή. Φαίνεται ξεκάθαρα ότι το όχημα με τον βελτιωμένο ελεγκτή προσεγγίζει τα εμπόδια σε μικρότερη απόσταση αυτό όμως έχει ως αποτέλεσμα να κάνει πιο «κλειστές στροφές» και να διανύει μικρότερη συνολική απόσταση, επίσης όταν το εμπόδιο βρίσκεται από τα αριστερά το ξεπερνάει στρίβοντας λιγότερο από όταν το εμπόδιο είναι στη δεξιά πλευρά.



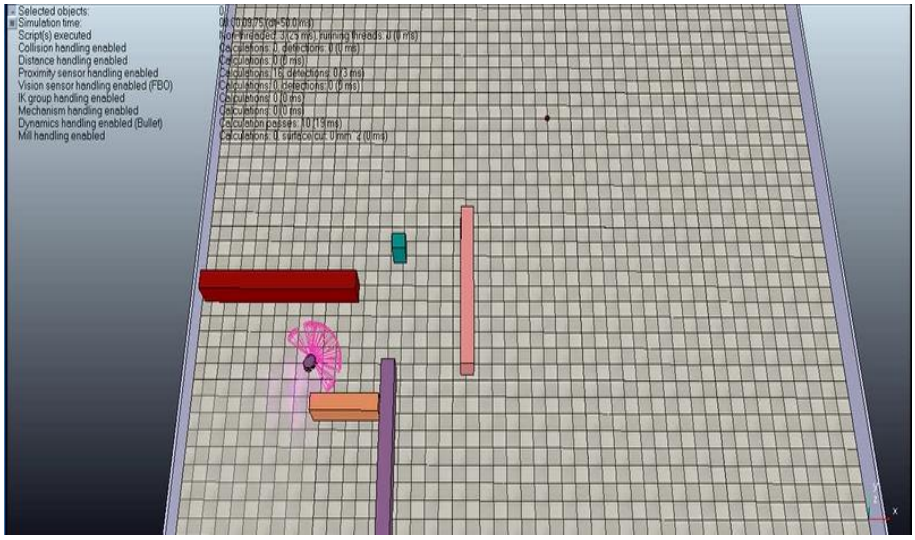
Σχήμα 4.48 Διαδρομή οχήματος με τον αρχικό και τον καλύτερο ελεγκτή

Παρακάτω παρουσιάζονται στιγμιότυπα από την προσομοίωση



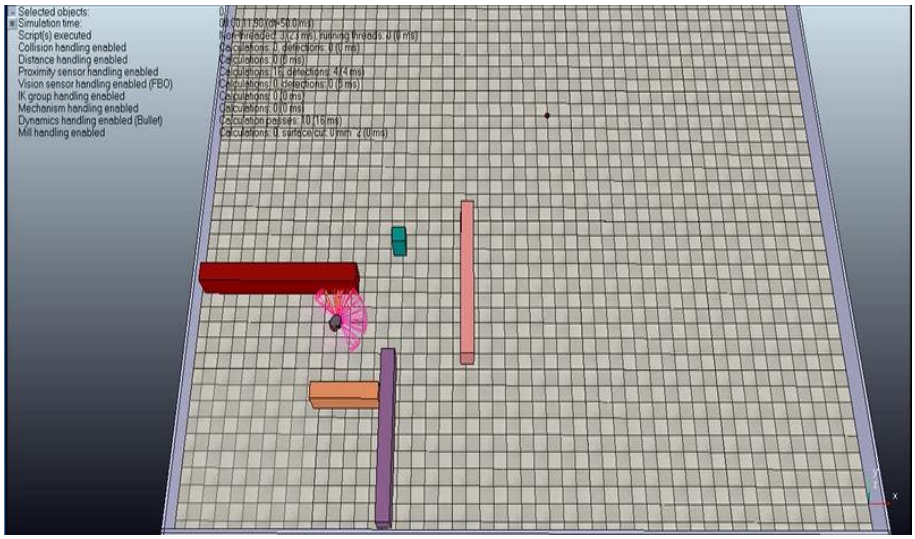
Εικόνα 4.20 Βέλτιστος ελεγκτής 6 sec

Στην εικόνα 4.20 παρατηρούμε ότι το όχημα με τον βέλτιστο ελεγκτή πλησιάζει πολύ κοντά στο εμπόδιο

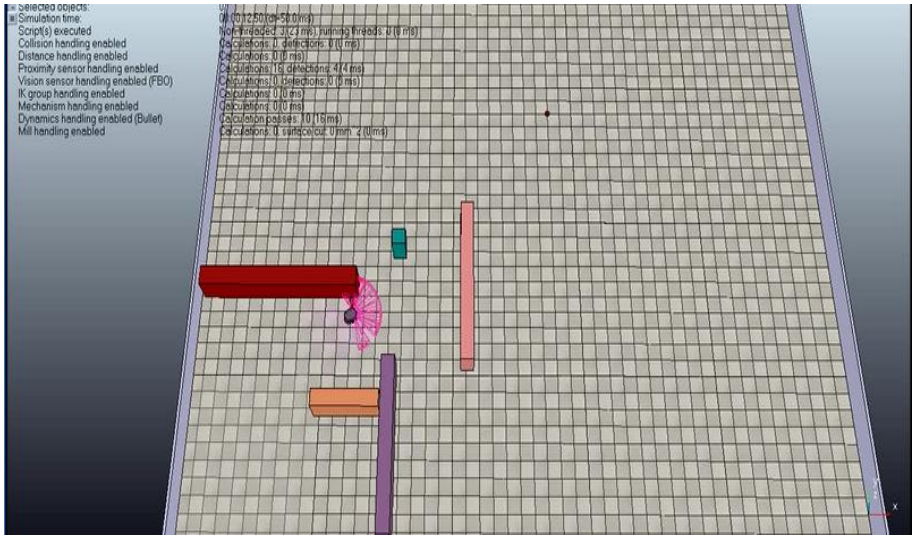


Εικόνα 4.21 Βέλτιστος ελεγκτής 9sec

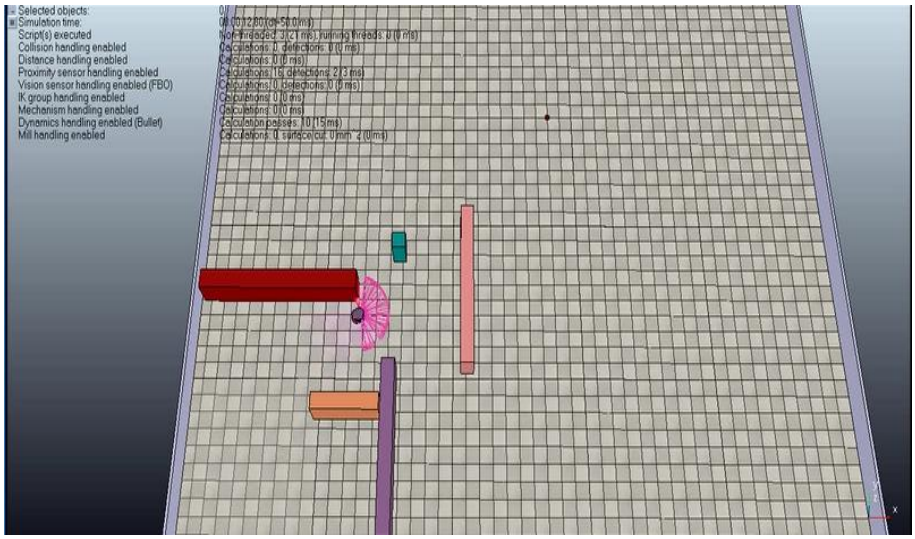
Στην εικόνα 4.21 παρατηρούμε ότι το όχημα πήρε αρκετά κλειστά τη στροφή



Εικόνα 4.22 Βέλτιστος ελεγκτής 11 sec

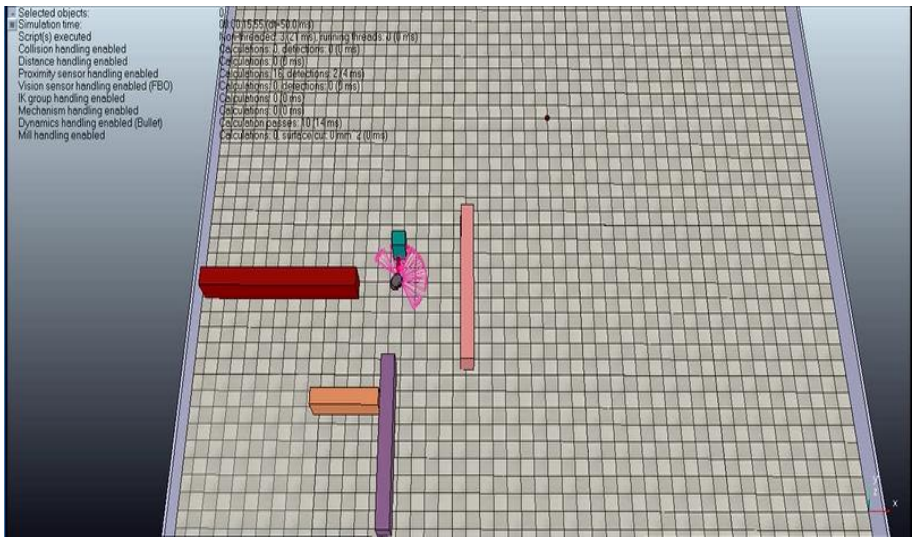


Εικόνα 4.23 Βέλτιστος ελεγκτής 12 sec

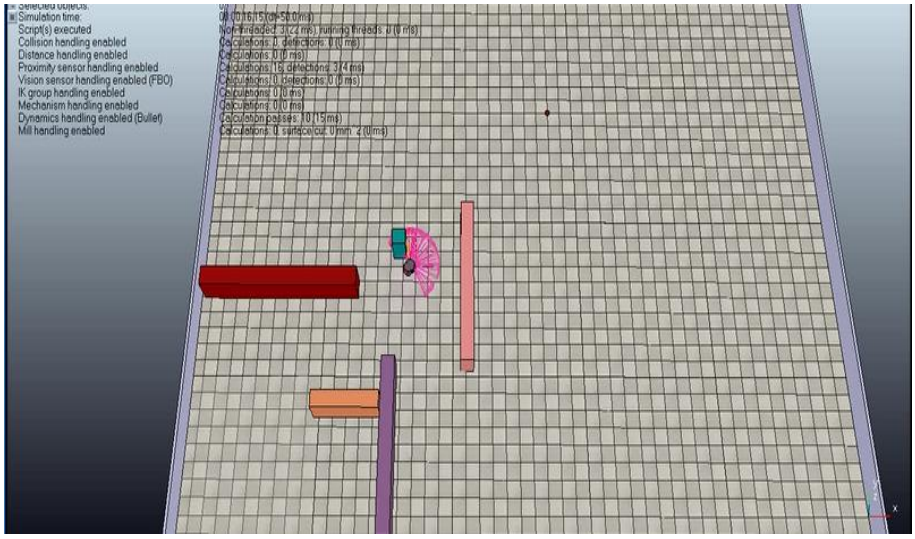


Εικόνα 4.24 Βέλτιστος ελεγκτής 13 sec

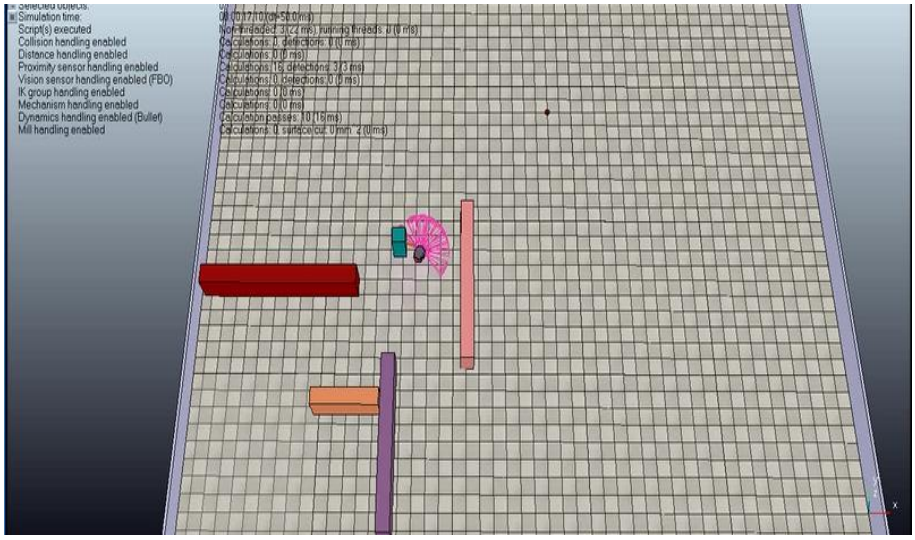
Στις εικόνες 4.22, 4.23, 4.24 παρατηρούμε πως αποφεύγει το όχημα το κόκκινο εμπόδιο από τα αριστερά και ότι δεν στρίβει πάρα πολύ άλλα το προσεγγίζει όσο πιο κοντά μπορεί ώστε να κάνει πορεία όσο το δυνατόν πιο ευθεία.



Εικόνα 4.25 Βέλτιστος ελεγκτής 15 sec

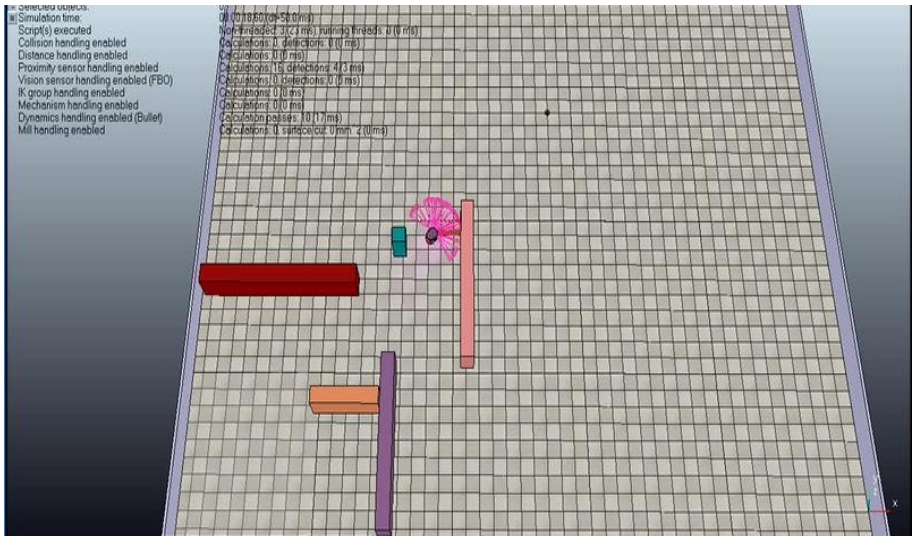


Εικόνα 4.26 Βέλτιστος ελεγκτής 16 sec

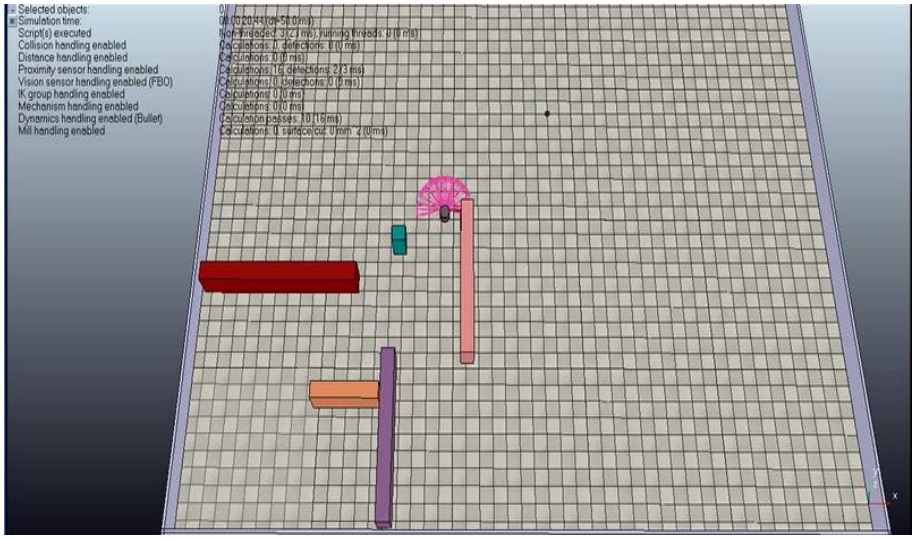


Εικόνα 4.27 Βέλτιστος ελεγκτής 17 sec

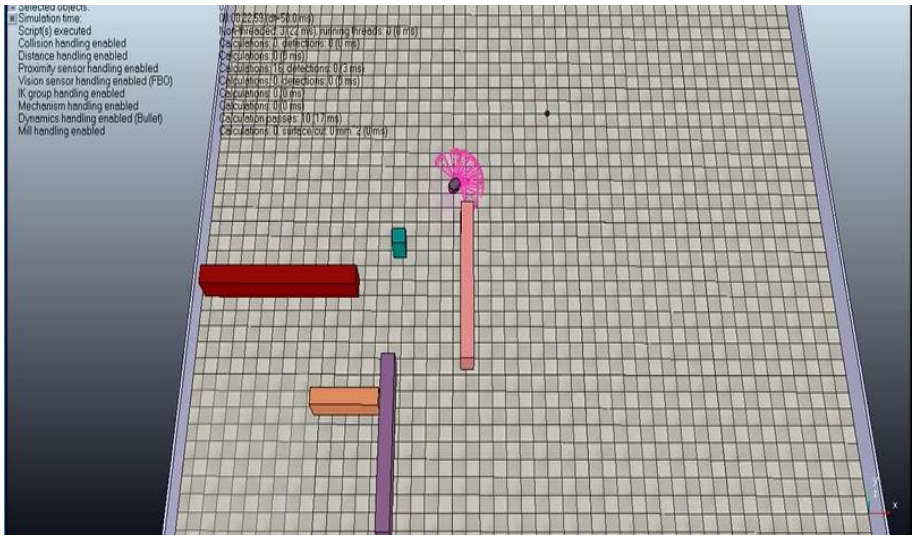
Στις εικόνες 4.25, 4.26, 4.27 παρατηρούμε πως αποφεύγει το όχημα το κυανό εμπόδιο από τα αριστερά και ότι δεν στρίβει πάρα πολύ άλλα το προσεγγίζει όσο πιο κοντά μπορεί ώστε να κάνει πορεία όσο το δυνατόν πιο ευθεία.



Εικόνα 4.28 Βέλτιστος ελεγκτής 18 sec

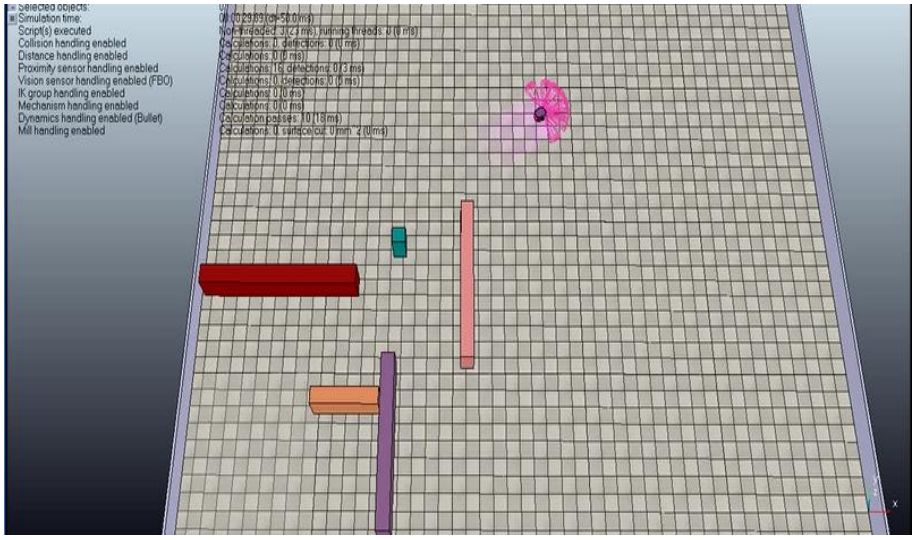


Εικόνα 4.29 Βέλτιστος ελεγκτής 20 sec



Εικόνα 4.30 Βέλτιστος ελεγκτής 22 sec

Στις εικόνες 4.28, 4.29, 4.30 παρατηρούμε πως αποφεύγει το όχημα το εμπόδιο από τα δεξιά, στρίβει πιο πολύ σε σχέση με πριν και παίρνει κλειστά την στροφή ώστε να ευθυγραμμιστεί με το σημείο-στόχο με αποτέλεσμα να κάνει τη λιγότερο δυνατή διαδρομή.



Εικόνα 4.31 Βέλτιστος ελεγκτής 30 sec

Στην εικόνα 4.31 παρατηρούμε το όχημα να φτάνει στο σημείο-στόχο στα 30 sec. Στον πίνακα 4.12 παρουσιάζονται τα αποτελέσματα από την ανάλυση των καταγεγραμμένων δεδομένων

	Αρχικός ασαφής ελεγκτής	**Καλύτερος ασαφής ελεγκτής**	**Ποσοστό μεταβολής %**
Επανάληψη	1	30	
Σωματίδιο	1	6	
Τιμή συνάρτησης αξιολόγησης	3009	670	-77.73
Βήματα προσομοίωσης (steps)	983	591	-39.87
Χρόνος διαδρομής (sec)	50	30	-40.00
Μήκος διαδρομής (m)	19.76	18.30	-7.38
Μέση ταχύτητα (m/s)	4.25	6.37	+49.88
dV (m/s)	1.31	1.58	+20.61
Ελάχιστη ενεργοποίηση αισθητήρων (m)	0.62	0.42	-32.25
Μέσος όρος ενεργοποίησης αισθητήρων (m)	0.93	0.95	+2.15

Πίνακας 4.12 Δεδομένα προσομοίωσης

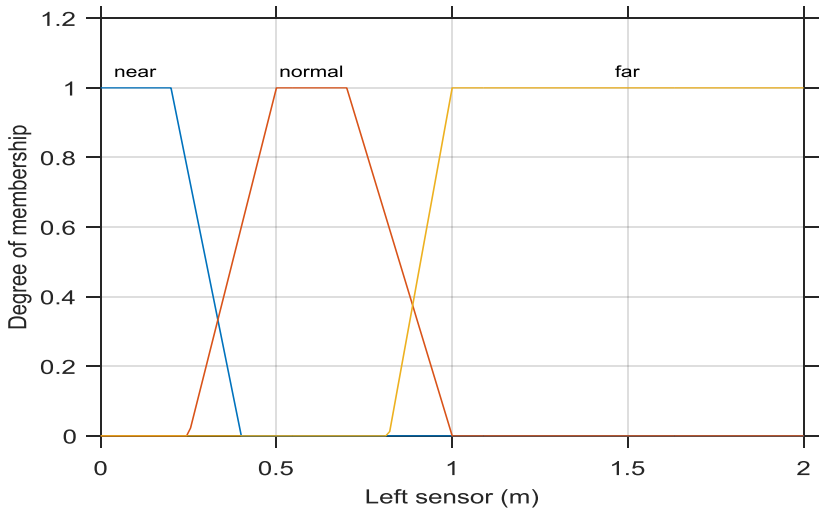
Στον πίνακα 4.13 συγκρίνονται τα δεδομένα του προηγούμενου βέλτιστου ελεγκτή με τον νέο.

	Καλύτερος Ασαφής Ελεγκτή Συνάρτησης Αξιολόγησης 2	**Καλύτερος Ασαφής Ελεγκτή Συνάρτησης Αξιολόγησης 1**	**Ποσοστό μεταβολής %**
Επανάληψη	30	17	
Σωματίδιο	6	5	
Βήματα προσομοίωσης (steps)	591	794	-25.56
Χρόνος διαδρομής (sec)	30	42	-28.57
Μήκος διαδρομής (m)	18.30	18.81	-2.71
Μέση ταχύτητα (m/s)	6.37	5.03	26.64
dV (m/s)	1.58	1.72	-8.13
Ελάχιστη ενεργοποίηση αισθητήρων (m)	0.42	0.57	-26.31
Μέσος όρος ενεργοποίησης αισθητήρων (m)	0.95	0.95	0.00

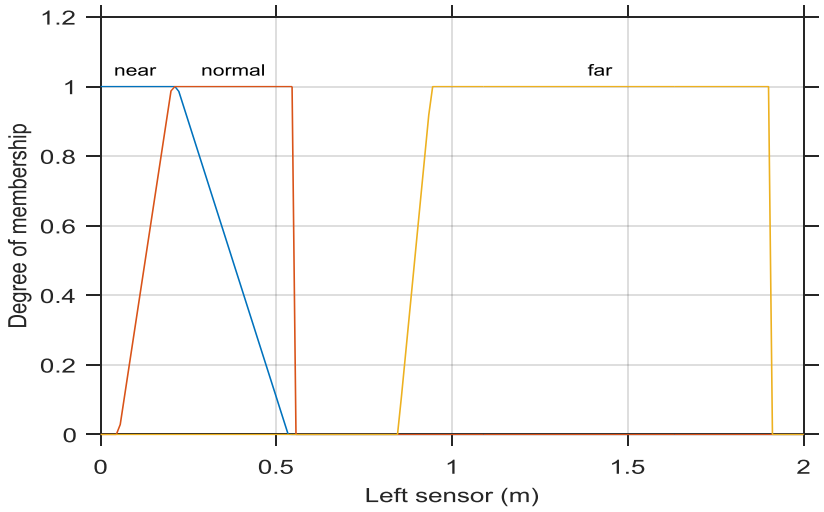
Πίνακας 4.13 Σύγκριση βέλτιστων ελεγκτών

Ο νέος βελτιστοποιημένος ελεγκτής σε σχέση με τον προηγούμενο κινείται με μεγαλύτερη ταχύτητα, αποφεύγει τα εμπόδια σε μικρότερη απόσταση και αυτό του δίνει το πλεονέκτημα να διανύει μικρότερη διαδρομή, με αποτέλεσμα να φτάνει στο στόχο πιο γρήγορα, δεν στρίβει τόσο απότομα και προσπαθεί να κάνει πορεία όσο το δυνατόν ευθύγραμμη ώστε να μπορεί να αναπτύξει μεγαλύτερη ταχύτητα, όλα αυτά έχουν αποτέλεσμα να φτάνει στο στόχο σε αρκετά μικρότερο χρονικό διάστημα. Συμπερασματικά η βελτίωση που παρουσιάζει είναι μεγαλύτερη από τον προηγούμενο βέλτιστο ελεγκτή.

Στα παρακάτω σχήματα παρουσιάζονται πως έχουν διαφοροποιηθεί οι τιμές των παραμέτρων των συναρτήσεων συμμετοχής των εισόδων και των εξόδων του νέου βελτιστοποιημένου ελεγκτή.

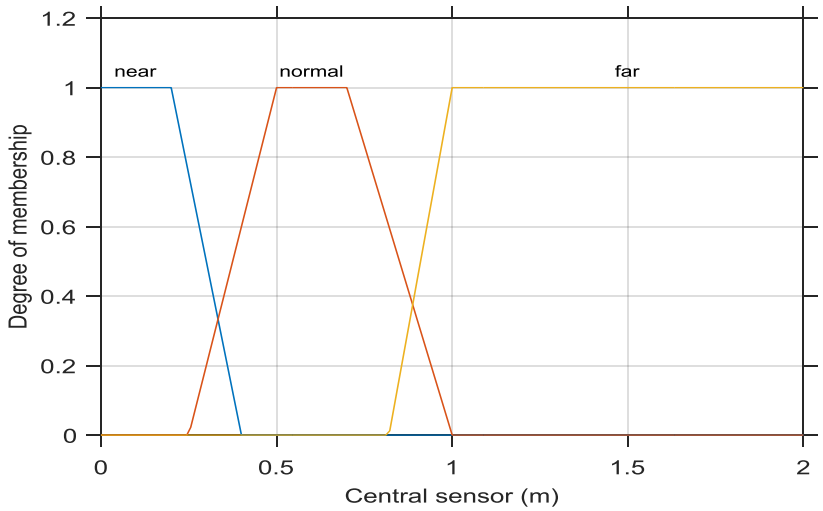


Σχήμα 4.49 Είσοδος «Αισθητήρας αριστερά» αρχικού ελεγκτή

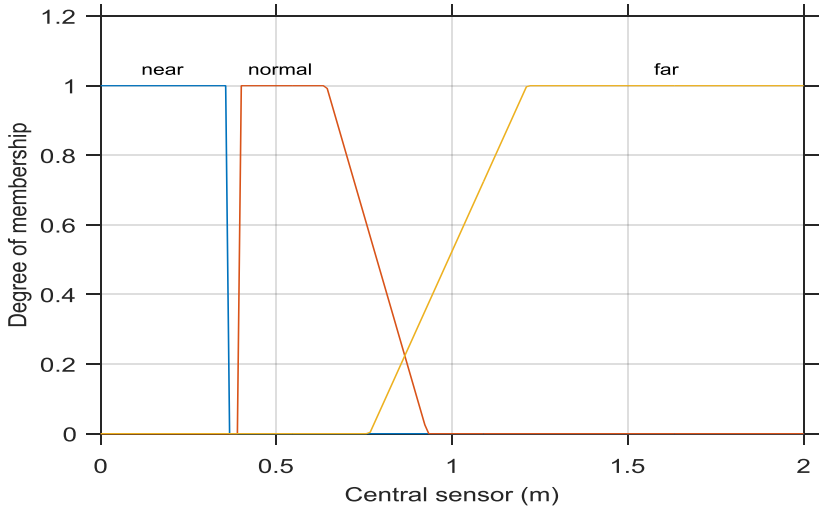


Σχήμα 4.50 Είσοδος «Αισθητήρας αριστερά» νέου βέλτιστου ελεγκτή

Στα σχήματα 4.49 και 4.50 παρατηρείται ότι στη συνάρτηση συμμετοχής «αργά», το τελικό όριό της έχει αυξηθεί πέρα του 0.5 m αυξάνοντας την επιρροή της στους κανόνες. Η συνάρτηση συμμετοχής «κανονικά» μετακινήθηκε αριστερά και καλύπτει μεγάλος μέρος της «αργά» αυξάνοντας την ασάφεια του συνόλου Η συνάρτηση συμμετοχής «μακριά» μείωσε το εύρος τη κάτω από 2 m.

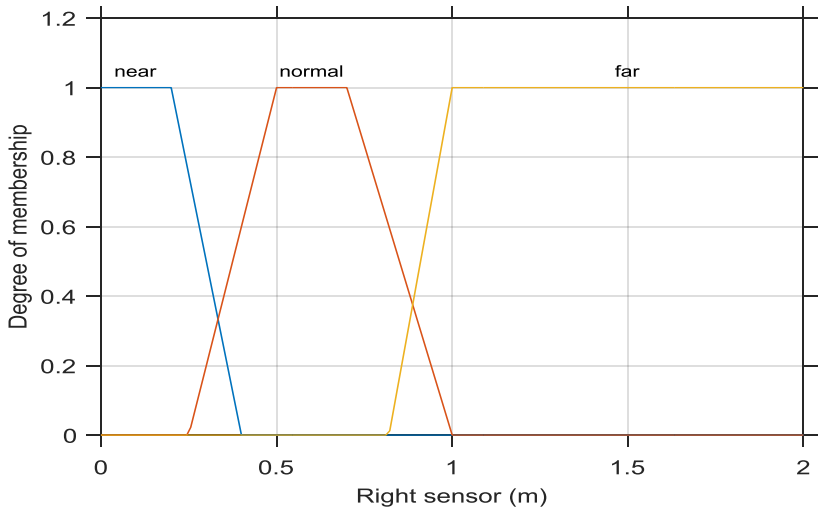


Σχήμα 4.51 Είσοδος «Αισθητήρας κέντρο» αρχικού ελεγκτή

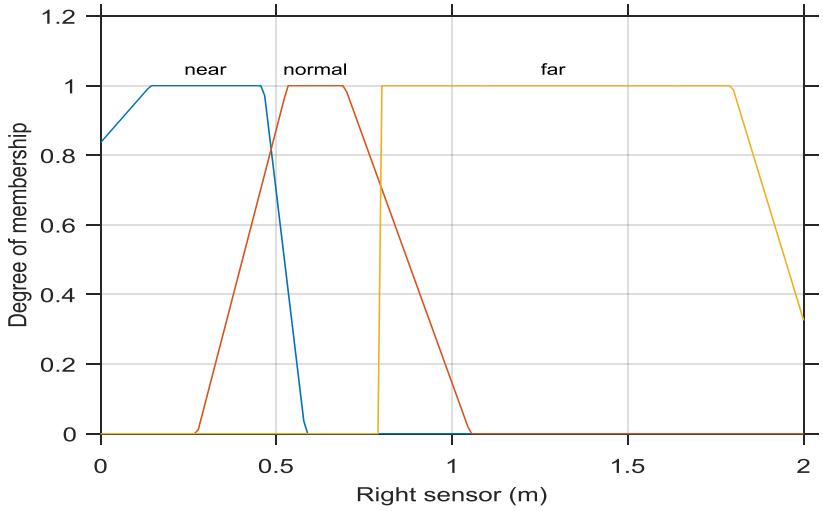


Σχήμα 4.52 Είσοδος «Αισθητήρας κέντρο» νέου βέλτιστου ελεγκτή

Στα σχήματα 4.51 και 4.52 παρατηρείται ότι στη συνάρτηση συμμετοχής «αργά», το τελικό όριό της έχει μειώθηκε ελάχιστα και έγινε πιο σαφής. Η συνάρτηση συμμετοχής «κανονικά» μείωσε το εύρος της και έγινε και αυτή πιο σαφής. Στη συνάρτηση συμμετοχής «μακριά» αυξήθηκε η ασάφεια των αρχικών τιμών.

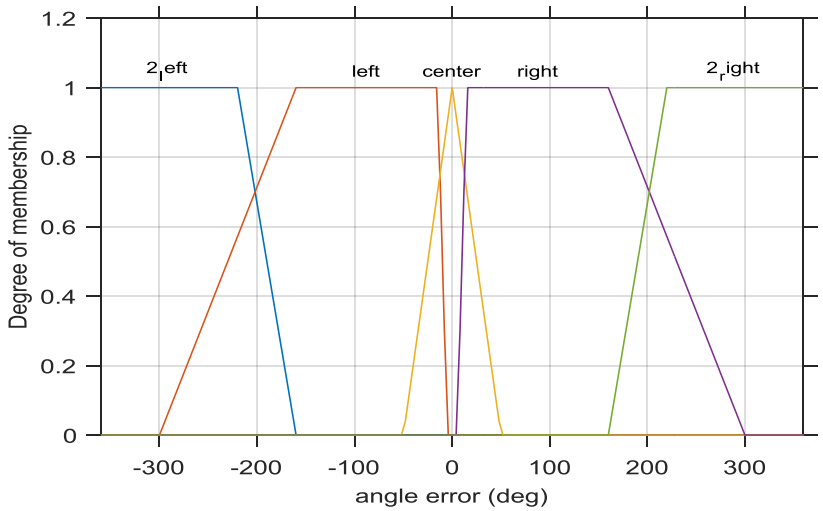


Σχήμα 4.53 Είσοδος «Αισθητήρας δεξιά» αρχικού ελεγκτή

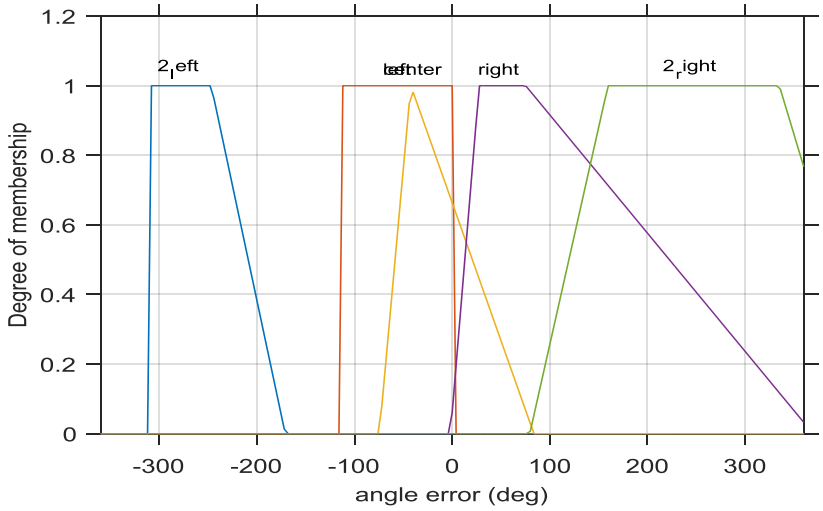


Σχήμα 4.54 Είσοδος «Αισθητήρας δεξιά» νέου βέλτιστου ελεγκτή

Στα σχήματα 4.53 και 4.54 παρατηρείται ότι στη συνάρτηση συμμετοχής «αργά», το τελικό όριό της έχει αυξηθεί πέρα του 0.5 m αυξάνοντας την επιρροή της στους κανόνες. Η συνάρτηση συμμετοχής «κανονικά» αύξησε τα όρια της και μείωσε το εύρος των τιμών όπου ο βαθμός συμμετοχής της είναι ίσος με ένα με αποτέλεσμα να αυξάνει η ασάφεια. Η συνάρτηση συμμετοχής αύξησε τις αρχικές τιμές της όπου είναι σίγουρα ένα (1).

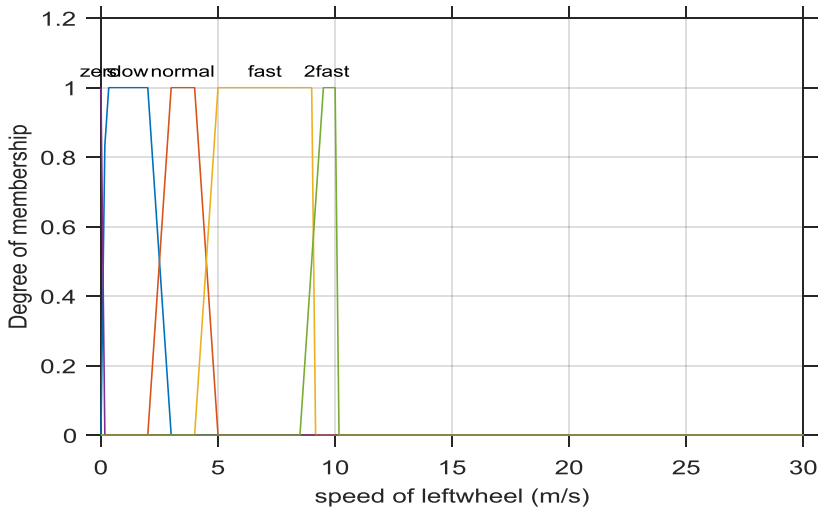


Σχήμα 4.55 Είσοδος «Γωνία σφάλματος» αρχικού ελεγκτή

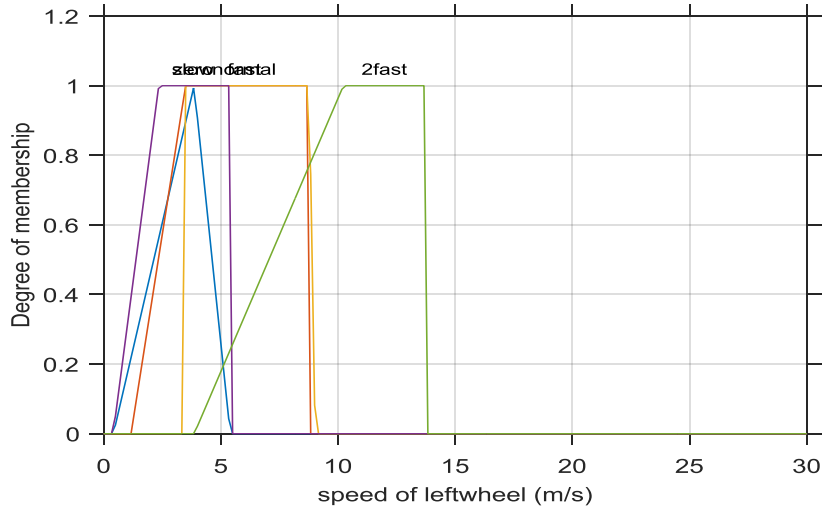


Σχήμα 4.56 Είσοδος «Γωνία σφάλματος» νέου βέλτιστου ελεγκτή

Στα σχήματα 4.55 και 4.56 παρατηρείται ότι στη συνάρτηση συμμετοχής «πολύ αριστερά» μείωσε το εύρος της και την ασάφειά της, η συνάρτηση συμμετοχής «αριστερά» μείωσε και αυτή το εύρος της και την ασάφειά της αλλά παράλληλα αύξησε το τελικό της όριο λίγο μετά του μηδενός, η επιρροή της συνάρτησης συμμετοχής «κέντρο» στους κανόνες ελαχιστοποιήθηκε καθώς καλύπτεται σχεδόν ολοκληρωτικά από τις γειτονικές της, η συνάρτηση συμμετοχής «δεξιά» αύξησε το εύρος της παράλληλα μείωσε το εύρος τιμών για τις οποίες ο βαθμός συμμετοχής είναι ίσος με ένα (1) αυξάνοντας την ασάφειά της, η συνάρτηση συμμετοχής «πολύ δεξιά» μείωσε τις αρχικές της τιμές, αύξησε το εύρος τιμών για τις οποίες ο βαθμός συμμετοχής είναι ίσος με ένα (1) αυξάνοντας την επιρροή της στους κανόνες.

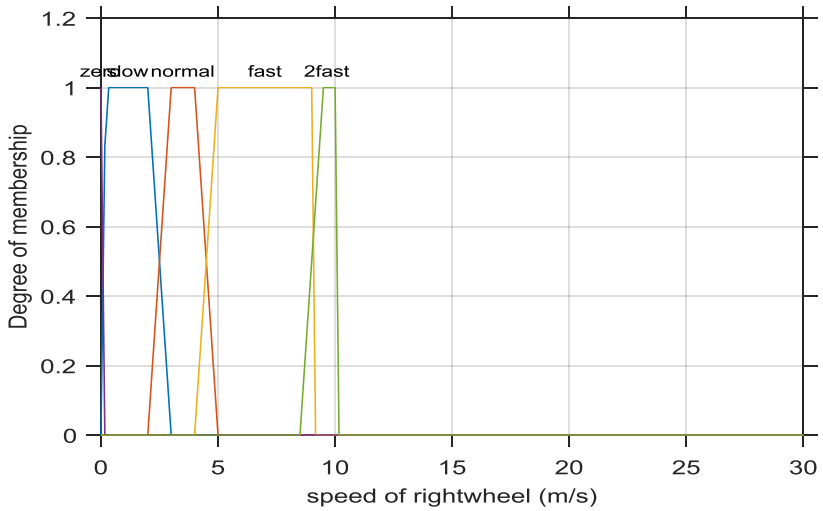


Σχήμα 4.57 Έξοδος «ταχύτητα αριστερού τροχού» αρχικού ελεγκτή

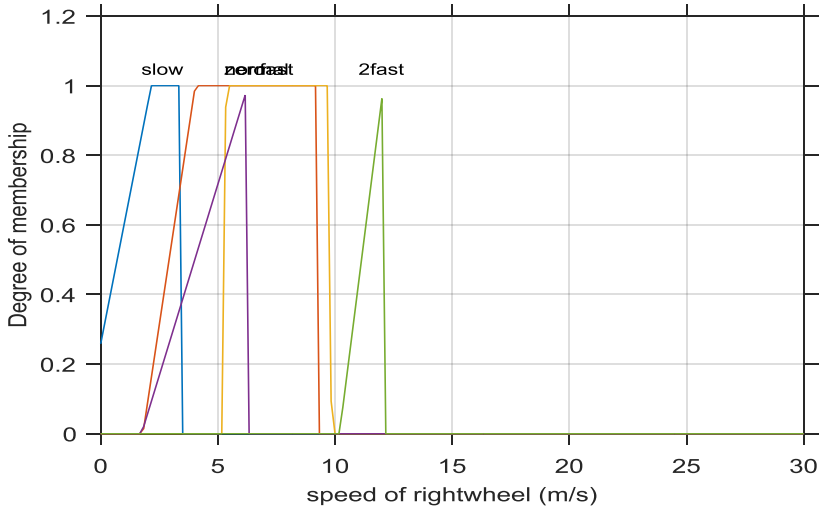


Σχήμα 4.58 Έξοδος «ταχύτητα αριστερού τροχού» νέου βέλτιστου ελεγκτή

Στα σχήματα 4.57 και 4.58 παρατηρείται ότι η συνάρτηση συμμετοχής «μηδέν» έχει μετακινηθεί προς τα δεξιά και από μία συγκεκριμένη (crisp) τιμή που ήταν ασαφοποιήθηκε, γενικά υπάρχει τάση αλληλοκάλυψης και ασαφοποίησης των συναρτήσεων συμμετοχής. Σημαντική διαφορά είναι η αύξηση του μέγιστου ορίου της συνάρτησης συμμετοχής «πολύ γρήγορα» περίπου στα 14m/s που έχει σαν αποτέλεσμα και την αύξηση της ταχύτητας του οχήματος.



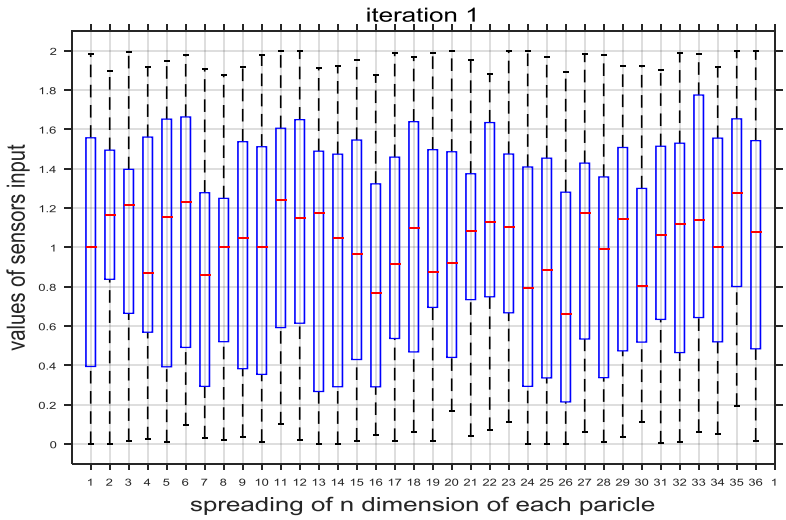
Σχήμα 4.59 Έξοδος «ταχύτητα δεξιού τροχού» αρχικού ελεγκτή



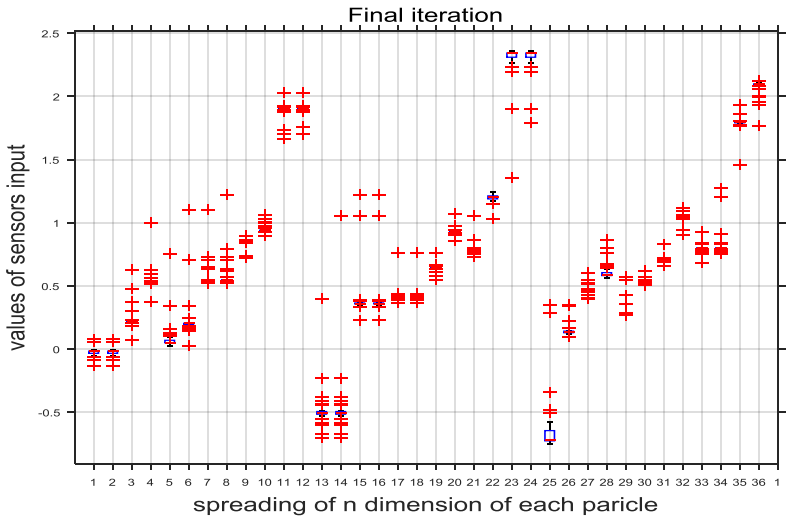
Σχήμα 4.60 Έξοδος «ταχύτητα δεξιού τροχού» νέου βέλτιστου ελεγκτή

Στα σχήματα 4.59 και 4.60 παρατηρείται ότι η συνάρτηση συμμετοχής «μηδέν» έχει μετακινηθεί προς τα δεξιά και από μία συγκεκριμένη (crisp) τιμή που ήταν ασαφοποιήθηκε, όμως καλύπτεται από άλλες και δεν επηρεάζει τους κανόνες, γενικά υπάρχει τάση αλληλοκάλυψης και ασαφοποίησης των συναρτήσεων συμμετοχής. Σημαντική διαφορά είναι η αύξηση του μέγιστου ορίου της συνάρτησης συμμετοχής «πολύ γρήγορα» περίπου στα 12m/s που έχει σαν αποτέλεσμα και την αύξηση της ταχύτητας του οχήματος.

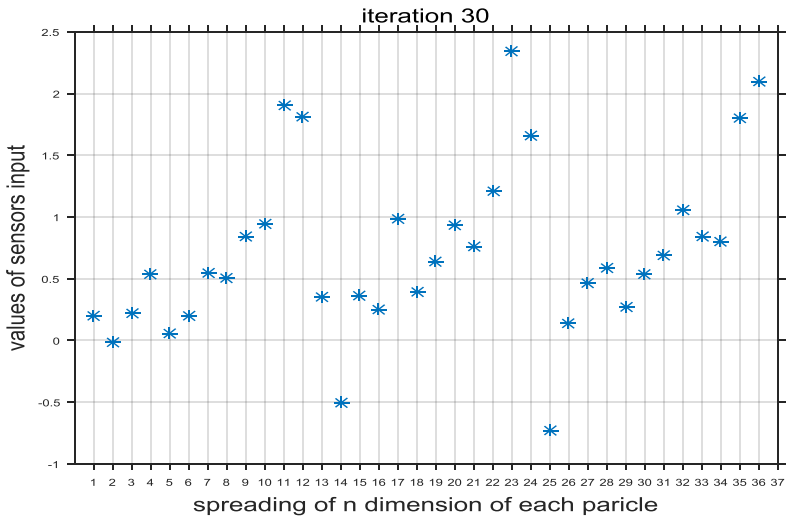
Στα σχήματα 4.61 και 4.62 παρουσιάζεται η κατανομή των παραμέτρων των συναρτήσεων των συμμετοχής των εισόδων των αισθητηρίων των ελεγκτών κατά τη δημιουργία τη του πληθυσμού και στην τελική επανάληψη αντίστοιχα. Παρατηρούμε ότι ενώ στην πρώτη επανάληψη η κατανομή είναι μεγάλη και έχει εύρος σχεδόν από 0 έως 2 σε κάθε διάσταση, που ήταν και το επιθυμητό, στην τελική επανάληψη τείνουν να συγκλίνουν σε κάποιο συγκεκριμένο αριθμό. Στο σχήμα 4.63 παρουσιάζονται οι παράμετροι των συναρτήσεων συμμετοχής του βέλτιστου ελεγκτή μετά από 17 επαναλήψεις. Παρατηρούμε πως όλοι οι παράμετροι των σωματιδίων έχουν συγκλίνει στη βέλτιστη λύση.



Σχήμα 4.61 Κατανομή παραμέτρων του συνόλου των σωματιδίων για τις εισόδους «Αισθητήρας αριστερά», «Αισθητήρας κέντρο», «Αισθητήρας δεξιά»

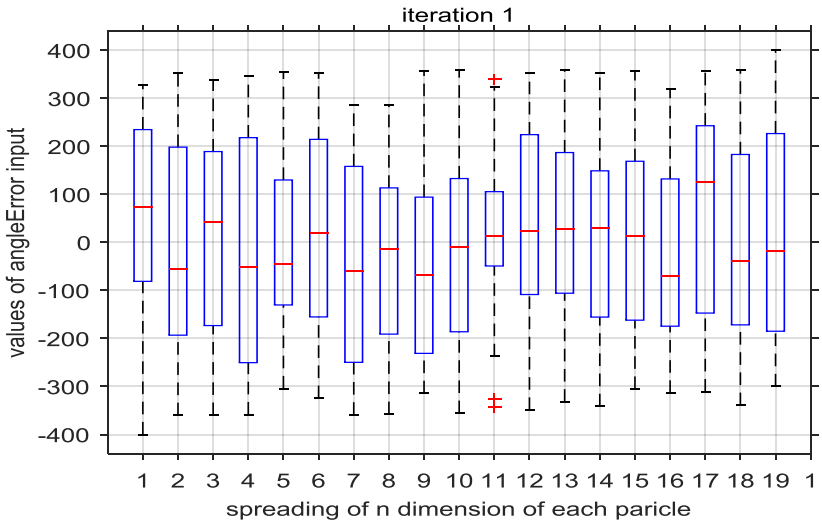


Σχήμα 4.62 Κατανομή παραμέτρων του συνόλου των σωματιδίων για τις εισόδους «Αισθητήρας αριστερά», «Αισθητήρας κέντρο», «Αισθητήρας δεξιά»

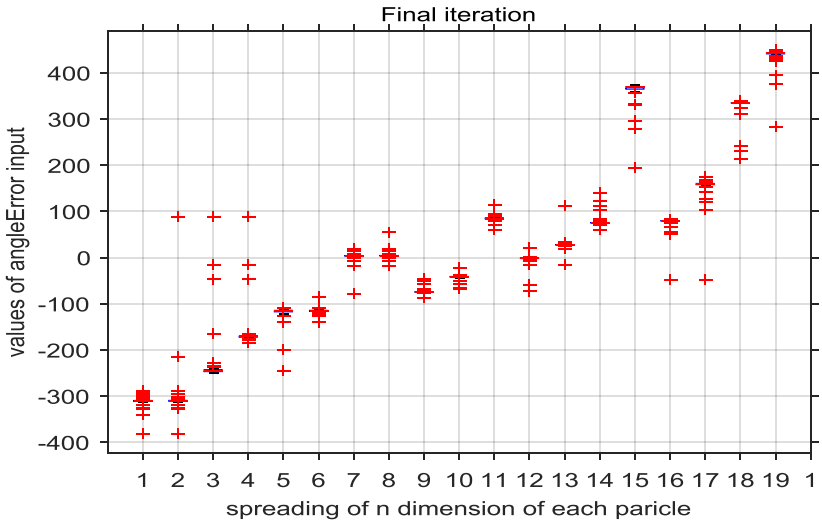


Σχήμα 4.63 Κατανομή παραμέτρων του νέου βέλτιστου ελεγκτή για τις εισόδους «Αισθητήρας αριστερά», «Αισθητήρας κέντρο», «Αισθητήρας δεξιά»

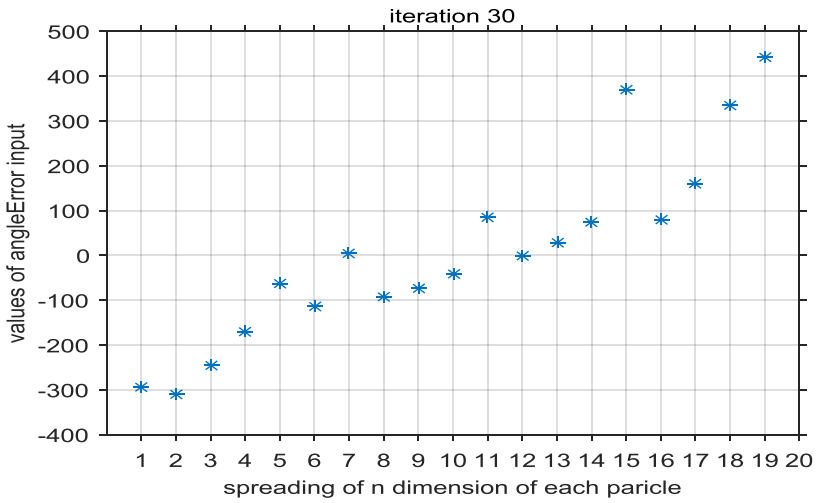
Στα σχήματα 4.64 και 4.65 παρουσιάζεται η κατανομή των παραμέτρων των συναρτήσεων συμμετοχής της εισόδου της γωνίας σφάλματος των ελεγκτών κατά τη δημιουργία τη του πληθυσμού και στην τελική επανάληψη αντίστοιχα. Παρατηρούμε ότι ενώ στην πρώτη επανάληψη η διασπορά είναι μεγάλη και έχει εύρος σχεδόν από -360 έως 360 σε κάθε διάσταση, που ήταν και το επιθυμητό, στην τελική επανάληψη τείνουν να συγκλίνουν σε κάποιο συγκεκριμένο αριθμό. Στο σχήμα 4.66 παρουσιάζονται οι παράμετροι των συναρτήσεων συμμετοχής του βέλτιστου ελεγκτή μετά από 17 επαναλήψεις. Παρατηρούμε πως όλοι οι παράμετροι των σωματιδίων έχουν συγκλίνει στη βέλτιστη λύση.



Σχήμα 4.64 Κατανομή παραμέτρων του συνόλου των σωματιδίων για την είσοδο «Γωνία σφάλματος

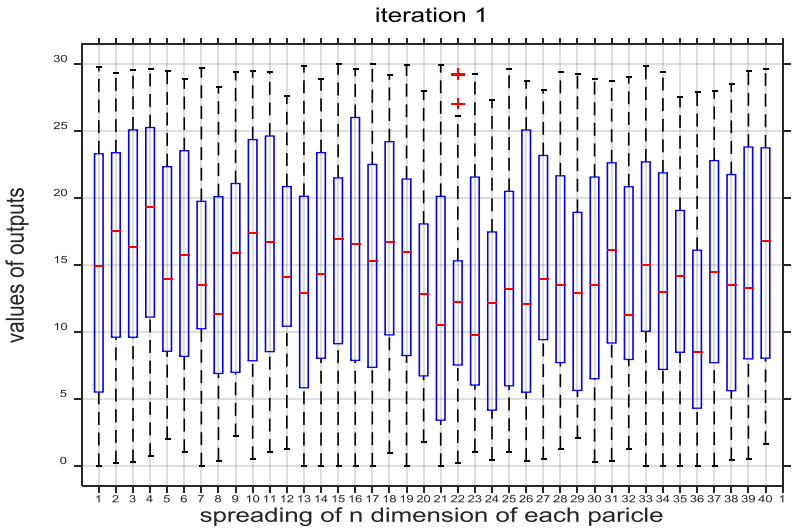


Σχήμα 4.65 Κατανομή παραμέτρων του συνόλου των σωματιδίων για την είσοδο «Γωνία σφάλματος»

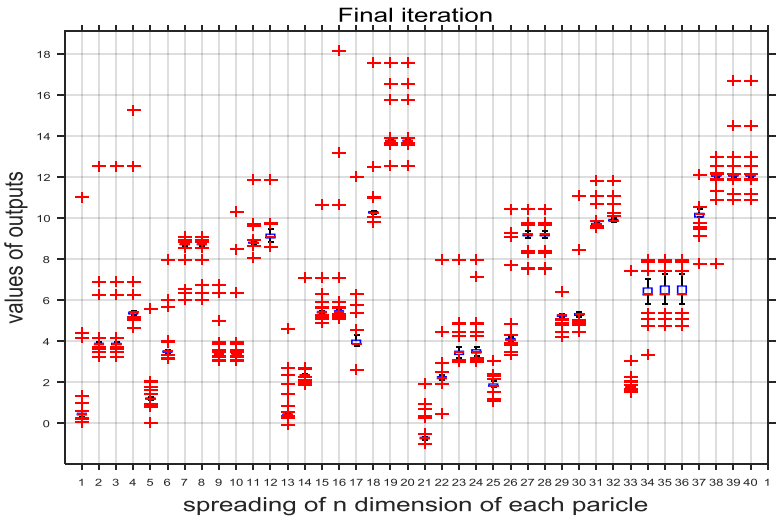


Σχήμα 4.66 Κατανομή παραμέτρων του νέου βέλτιστου ελεγκτή για την είσοδο «Γωνία σφάλματος»

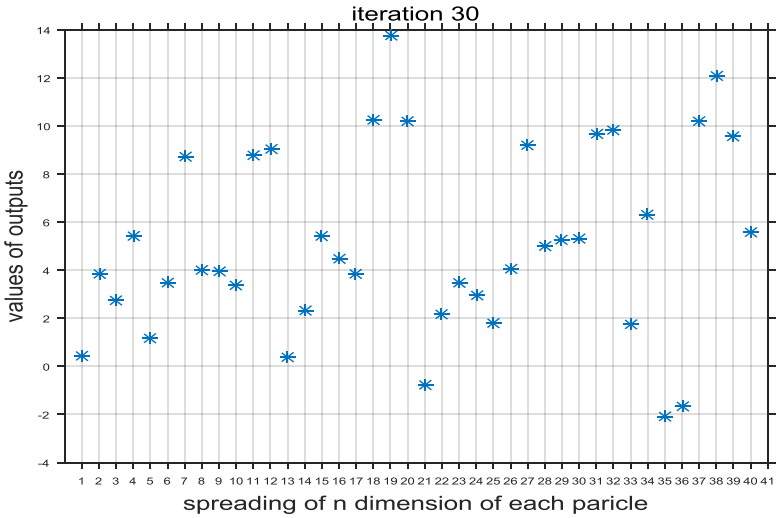
Στα σχήματα 4.67 και 4.68 παρουσιάζεται η κατανομή των παραμέτρων των συναρτήσεων των συμμετοχής των εξόδων των ταχυτήτων των ελεγκτών κατά τη δημιουργία του πληθυσμού και στην τελική επανάληψη αντίστοιχα. Παρατηρούμε ότι ενώ στην πρώτη επανάληψη η διασπορά είναι μεγάλη και έχει εύρος σχεδόν από 0 έως 30 σε κάθε διάσταση, που ήταν και το επιθυμητό, στην τελική επανάληψη τείνουν να συγκλίνουν σε σχεδόν σε κάποιο συγκεκριμένο αριθμό. Στο σχήμα 4.69 παρουσιάζονται οι παράμετροι των συναρτήσεων συμμετοχής του βέλτιστου ελεγκτή μετά από 17 επαναλήψεις. Παρατηρούμε πως όλοι οι παράμετροι των σωματιδίων έχουν συγκλίνει στη βέλτιστη λύση.



Σχήμα 4.67 Κατανομή παραμέτρων του συνόλου των σωματιδίων των εξόδων «Ταχύτητα αριστερού τροχού» και «Ταχύτητα δεξιού τροχού»



Σχήμα 4.68 Κατανομή παραμέτρων του συνόλου των σωματιδίων των εξόδων «Ταχύτητα αριστερού τροχού» και «Ταχύτητα δεξιού τροχού»



Σχήμα 4.69 Κατανομή παραμέτρων του νέου βέλτιστου ελεγκτή των εξόδων «Ταχύτητα αριστερού τροχού» και «Ταχύτητα δεξιού τροχού»

ΚΕΦΑΛΑΙΟ 5

Συμπεράσματα/μελλοντικές επεκτάσεις

5.1 Συμπεράσματα

Σκοπός της παρούσας εργασίας ήταν να μελετηθεί η επίδραση Αλγορίθμων Βελτιστοποίησης Σμήνους Σωματιδίων, σε έντροχα ρομποτικά οχήματα που έχουν τη δυνατότητα να πλοηγούνται σε άγνωστο περιβάλλον με χρήση ελεγκτών ανεπτυγμένων με εργαλεία υπολογιστικής νοημοσύνης και συγκεκριμένα *ασαφών ελεγκτών*. Αναλύεται η επίδραση της διαδικασίας βελτιστοποίησης στη συμπεριφορά των οχημάτων (ταχύτητα, χρόνος επίτευξης στόχου, αποφυγή εμποδίων κλπ.) καθώς και γίνεται διερεύνηση της επίδρασης διαφορετικών συναρτήσεων αξιολόγησης στην διαδικασία βελτιστοποίησης.

Στα πλαίσια της εργασίας αναπτύχθηκαν μια σειρά από ελεγκτές ασαφούς λογικής για ένα προσομοιωμένο ρομποτικό όχημα τύπου Pioneer που του επέτρεπαν να κινηθεί σε ένα δομημένο περιβάλλον όπου υπήρχαν εμπόδια δίχως να συγκρουστεί με αυτά και να βρει σημεία στόχους. Οι συγκεκριμένοι ελεγκτές βελτιστοποιηθήκαν με χρήση αλγορίθμου βελτιστοποίησης σμήνους σωματιδίων και με χρήση διαφορετικών συναρτήσεων αξιολόγησης. Οι συναρτήσεις που χρησιμοποιήθηκαν ήταν βασισμένες σε συμπεριφορές και συγκεντρωτικές και οδήγησαν σε νέους ελεγκτές με σαφώς καλύτερη συμπεριφορά από αυτούς που είχαν σχεδιαστεί ευρετικά από τον άνθρωπο σχεδιαστή.

Συγκεκριμένα με χρήση των νέων ελεγκτών το όχημα είχε τη δυνατότητα να κινείται με μεγαλύτερη ταχύτητα και να αποφεύγει τα εμπόδια σε μικρότερη απόσταση και αυτό του έδινε το πλεονέκτημα να διανύει μικρότερη διαδρομή, με αποτέλεσμα να φτάνει στο στόχο πιο γρήγορα, δεν έστριβε τόσο απότομα και προσπαθούσε να κάνει πορεία όσο το δυνατόν ευθύγραμμη ώστε να μπορεί να αναπτύξει μεγαλύτερη ταχύτητα. Κατά τη διεξαγωγή των πειραμάτων αναδείχθηκε η σημαντικότητα του ρόλου της συνάρτησης αξιολόγησης στην εξέλιξη της βελτιστοποίησης και στην επίδραση της συμπεριφοράς του οχήματος.

Στο μέλλον προτείνεται η επέκταση της παρούσας εργασίας με την μελέτη της επίδρασης περισσότερων συναρτήσεων αξιολόγησης καθώς επίσης την μελέτη της εφαρμογής αντιστοίχων τεχνικών σε ομάδες έντροχων οχημάτων που επιδεικνύουν συνεργατική συμπεριφορά για την επίτευξη διάφορων αποστολών. Η ορθή λειτουργία των παραπάνω ελεγκτών αναμένεται να δοκιμαστεί και σε πραγματικά ρομποτικά οχήματα.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] L. A. Zadeh, "Fuzzy Sets", *Information and Control*, vol. 8, pp. 338-353, 1965.

[2] Ι. Α. Θεοδώρου, Εισαγωγή στην Ασαφή λογική, σελ.23-46, Εκδόσεις Τζίολα, 2010.

[3] Σ. Παπαδάκης, Π. Αδαμίδης, "Ασαφή Συστήματα Θεωρία και Εργαστηριακές ασκήσεις", ΤΕΙ Θεσσαλονικής, σελ. 3-15, 2004.

[4] Σ. Βολογιαννίδης, "Ευφυής Έλεγχος, Θεωρία και Εφαρμογές", Διδακτικές σημειώσεις, Τμήματος Πληροφορικής και Επικοινωνιών, ΤΕΙ Σερρών, 2009.

[5] Καλυκάκης Γ. Εμμανουήλ, "Ανάπτυξη και αυτόνομη πλοήγηση ρομποτικής πλατφόρμας αγρού σε καλλιέργεια σειρών", Μεταπτυχιακή εργασία, Διατμηματικό Πρόγραμμα Προηγμένα Συστήματα Παραγωγής, Αυτοματισμού και Ρομποτικής, 2016.

[6] Σ. Γ. Τζαφέστα, Υπολογιστική νοημοσύνη: Τόμος Α, κεφάλαιο 3, σελ 63-66, 2002.

[7] Carlos Dualibe, Michel Verleysen and Paul G.A. Jespers, Design of Analog Fuzzy Logic Controllers in CMOS Technologies – Implementation, Test and Application, Kluwer Academic Publishers, 2003.

[8] Ε. Δοϊτσίδης, "Μεθοδολογία Μοντελοποίησης και Βελτιστοποίησης Υπολογιστικής Νοημοσύνης Ομάδας Αυτόνομων Οχημάτων", Διδακτορική Διατριβή, Τμήμα Μηχανικών Παραγωγής και Διοίκησης, Πολυτεχνείο Κρήτης, 2008.

[9] A. Saffiotti, "Handling uncertainty in control of autonomous robots", *Lecture Notes in Artificial Intelligence 1455*, A. Hunter and S. Parsons, Eds.: Springer-Verlag, pp. 198–224, 1998.

[10] X. Yang, M. Moallem, and R. V. Patel, "A layered Goal-Oriented Fuzzy Motion Planning Strategy for Mobile Robot Navigation", *IEEE Transactions on Systems, Man, Cybernetics Part B: Cybernetics*, vol. 35, pp. 1214-1224, Dec. 2005.

[11] P. Resu, E. M. Petriu, T. M. Whalen, A. Cornell, and H. J. W. Spoelder, "Behavior-Based NeuroFuzzy Controller for Mobile Robot Navigation", *IEEE Transactions on Instrumentation and Measurement*, vol. 52, pp. 1335-1340 Aug. 2003.

[12] E. Aguire and A. Gonzalez, "Fuzzy Behaviors for Mobile Robot Navigation: Design, Coordination and Fusion", *International Journal of Approximate Reasoning*, vol. 25, pp. 225-289, 2000.

[13] S. C. Goodridge and R. C. Luo, "Fuzzy Behavior Fusion for Reactive Control of an Autonomous Mobile Robot: MARGE", *in Proceedings of the IEEE International Conference on Robotics and Automation*, San Diego, pp. 1622-1627, 1994.

[14] W. Li, "Fuzzy Logic-based 'Perception-Action' Behavior Control of a Mobile Robot in Uncertain Environment", *in Proceedings of the 3rd IEEE Conference on Fuzzy Systems*, San Francisco, pp. 1626-1631, 1994.

[15] G. Oriolo, G. Ulivi, and M. Vindittelli, "Real-time Map Builting and Navigation for Autonomous Robots in Unknown Environments", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 28 pp. 316-333, 1998.

[16] O. S. Ishikawa, "A Method of Indoor Mobile Robot Navigation by Using Fuzzy Control", *in Proceeding of the IEEE/RSJ International Workshop on Intelligent Robots and Systems*, Osaka, Japan, pp. 1013-1018, 1991.

[17] H. Seraji and A. Howard, "Behavior-Based Robot Navigation on Challenging Terrain: A Fuzzy Logic Approach", *IEEE Transactions on Robotics and Automation*, vol. 18, pp. 308-321, 2002.

[18] N. Tsourveloudis, L. Doitsidis, K. Valavanis, "Autonomous Navigation of Unmanned Vehicles: A Fuzzy Logic Perspective," *Cutting Edge Robotics*, pp. 291-310, Pro Literatur Verlag, ISBN:3-86611-038-3, 2005.

[19] N. Tsourveloudis, L. Doitsidis, K. Valavanis, "Autonomous Navigation of Unmanned Vehicles: A Fuzzy Logic Perspective," *Cutting Edge Robotics*, pp. 291-310, Pro Literatur Verlag, ISBN:3-86611-038-3, 2005.

[20] L. Doitsidis, K. P. Valavanis, and N. C. Tsourveloudis, "Fuzzy Logic Based Autonomous Skid Steering Vehicle Navigation", *in Proceedings of the IEEE International Conference on Robotics and Automation*, Washington DC, U.S.A, pp. 2171-2177, 2002.

[21] N. C. Tsourveloudis, K. P. Valavanis, and T. Hebert, "Autonomous Vehicle Navigation Utilizing Electrostatic Potential Fields and Fuzzy Logic", *IEEE Transactions on Robotics and Automation*, vol. 17, pp. 490-497, 2001.

[22] Χ. Φούντας, "Αλγόριθμοι τεχνητής ευφυίας σμήνους και εφαρμογές σε μη-ντετερμινιστικά πολυωνυμικά προβλήματα", 2008.

[23] J. Kennedy and R.C. Eberhart, Swarm Intelligence, Morgan Kaufmann Publishers, 2001.

[24] R. C. Eberhart, P. Simpson, and R. Dobbins, Computational Intelligence, PC Tools, Academic Press, 1996.

[25] W. T. Reeves. "Particle systems–a technique for modelling a class of fuzzy objects", *ACM Transactions on Graphics*, 2(2): pp. 91–108, 1983.

[26] J. Kennedy and R. C. Eberhart, "Particle swarm optimization", *In Proc. IEEE Int. Conf. Neural Networks*, vol IV, pp. 1942–1948, 1995.

[27] Χ. Μπαχτσεβανίδης, "Ανάθεση κυψελίδων σε ασύρματα δίκτυα με τη χρήση αλγορίθμων βελτιστοποίησης σμήνους σωματιδίων", Μεταπτυχιακή Εργασία, Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης, 2009.

[28] Andries P. Engelbrech, Computational Intelligence: An Intoduction, Wiley, 2007.

[29] Π. Γκαϊδατζής, Ε. Κοντοστάθης, "Εύρεση Βέλτιστης Θέσης και Διαστασιολόγησης Εγκατεστημένης Ισχύος Μονάδων Διανεμημένης Παραγωγής με χρήση Τεχνικής Βελτιστοποίησης Σμήνους Σωματιδίων", Διπλωματική εργασία, Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης, 2014.

[30] Ι.Μαρινάκης, Μ. Μαρινάκη, "Εξελικτικοί Αλγόριθμοι και Βελτιστοποίηση Συστημάτων Μεγάλης Κλίμακας", Σημειώσεις Μεταπτυχιακού Μαθήματος Εξελικτικοί Αλγόριθμοι και Βελτιστοποίηση Συστημάτων Μεγάλης Κλίμακας, Σχολή Μηχανικών Παραγωγής και Διοίκησης, Πολυτεχνείο Κρήτης, 2010.

[31] Andries P. Engelbrecht, Fundamentals of Computational Swarm Intelligence, Wiley, 2005.

[32] Kennedy J and Mendes R, "Population structure and particle swarm performance," *Proceeding of IEEE conference on Evolutionary Computation*, pp. 1671-1676, 2002.

[33] Shi, Y. Eberhart, R., "A modified particle swarm optimizer", *In Proceedings of the IEEE World Congress on Computational Intelligence*, pp. 69-73, 1998.

[34] Eberhart, R. C. Shi, Y., "Comparing inertia weights and constriction factors in particle swarm optimization", *In proceedings of the 2000 Congress on Evolutionary Computation*, vol 1, pp. 8488, 2000.

[35] M. Clerc, J. Kennedy, "The particle swarm explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58-73, 2002.

[36] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: Simpler, maybe better," *IEEE Transactions on Evolutionary Computation*, vol. 8, pp. 204–210, 2004.

[37] Parsopoulos, K.E., Vrahatis, M.N. "UPSO: A unified particle swarm optimization scheme," *In: Lecture Series on Computer and Computational Sciences*, vol.1, pp. 868-873, 2004.

[38] J. Kennedy and R. Eberhart, "A discrete binary version of the particle swarm algorithm," *In Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, 4104-4108, IEEE Press, 1997.

[39] Z. D. Zaharis, S. K. Goudos, and T. V. Yioultsis, "Application of Boolean PSO with adaptive velocity mutation to the design of optimal linear antenna arrays excited by uniform-amplitude current distribution", *J. of Electromagn. Waves and Appl.,* Vol. 25, 1422–1436, 2011.

[40] Χ. Μπαχτσεβανίδης, "Ανάθεση κυψελίδων σε ασύρματα δίκτυα με τη χρήση αλγορίθμων βελτιστοποίησης σμήνους σωματιδίων", Μεταπτυχιακή Εργασία, Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης, 2009.

[41] Mahmud Iwan Solihin, Lee Fook Tack, Moey Leap Kean, 'Tuning of PID Controller Using Particle Swarm Optimization," *In proceedings of the International Conference on Advanced Science, Engineering and Information Technology*, pp. 458-461, 2011.

[42] Ching-Chang Wong, Hou-Yi Wang, and Shih, An Li, "PSO-based Motion Fuzzy Controller Design for Mobile Robots," *International Journal of Fuzzy Systems*, vol. 10, no. 1, pp. 284-292, 2008.

[43] Mahmood Rahmani, "Particle swarm optimization of artificial neural networks for autonomous robots", M.Sc. Thesis, Department of Applied Physics, Chalmers University of Technology, 2008.

[44] Jim Pugh, Alcherio Martinol, Yizhen Zhang "Particle swarm optimization for unsupeρvised robotic learning," *In proceedings of the Swarm Intelligence Symposium*, pp. 92-99, 2005.

[45] Thomas Rofer, Tim Laue, Cord Niehaus, "Gait Optimization on a Humanoid Robot using Particle Swarm Optimization," *In Proceedings of the Second Workshop on Humanoid Soccer*, 2007.

[46] Dong-Yuan Ge, Xi-Fan Yao, Qing-He Yao, Hong Jin, "Robot sensor calibration via neural network and particle swarm optimization enhanced with crossover and mutation", *Tehnički vjesnik*, vol. 21, no. 5, pp. 1025-1033, 2014.

[47] M. V. A. Raju. Bahubalendruni, B. B. Biswal, B. B. V. L Deepak, "Optimal Robotic Assembly Sequence Generation Using Particle Swarm Optimization", *Journal of Automation and Control Engineering*, vol. 4, No. 2, 2016.

[48] Dayal R Parhi, BBVL Deepak, "Path Generation of a Differential Mobile Robot Using Particle Swarm Optimization", *International Journal of Artificial Intelligence and Computational Research*, vol. 4, no. 1, pp. 7-11, 2012.

[49] Yangmin Li, Xin Chen, "Mobile Robot Navigation Using Particle Swarm Optimization and Adaptive NN," *In proceedings of the International Confeence on Natural Computation,* 2005.

[50] Maryam Yarmohamadi, H. Haj Seyyed Javadi, Hossein Erfani, "Improvement of Robot Path Planning Using Particle Swarm Optimization in Dynamic Environments with Mobile Obstacles and Target," *Advanced Studies in Biology*, vol. 3, no. 1, pp. 43-53, 2011.

[51] Qirong Tang and Peter Eberhard, "Cooperative Motion of Swarm Mobile Robots Based on Particle Swarm Optimization and Multibody System Dynamics", *Mechanics Based Design of Structures and Machines*, vol. 32, no. 2, pp. 179-193, 2011.

[52] Lisa L. Smith, Ganesh K. Venayagamoorthy, Phillip G. Holloway, "Obstacle Avoidance in Collective Robotic Search Using Particle Swarm Optimization", *Swarm Intelligence,* 2006.

[53] Δρ. Μ. Καββουσανός, "Κινηματική ανάλυση τροχοφόρων ρομπότ", Σημειώσεις μαθήματος: Προηγμένα Ρομποτικά Συστήματα, Διατμηματικό Μεταπτυχιακό Πρόγραμμα Προηγμένα Συστήματα Παραγωγής, Αυτοματισμού και Ρομποτικής, ΤΕΙ Κρήτης, 2016.

[54] A. Nelson, G. Barlow, L. Doitsidis, "Fitness Functions in Evolutionary Robotics: A Survey and Analysis," *Robotics and Autonomous Systems,* vol. 57, Issue 4, pp. 345-370, 2009.

[55] W. Banzhaf, P. Nordin, and M. Olmer, "Generating adaptive behavior using function regression within genetic programming and a real robot", *in Proceedings of the Second International Conference on Genetic Programming*, San Francisco, pp. 35-43, 1997.

[56] N. Jakobi, "Running across the reality gap: Octopod locomotion evolved in a minimal simulation", *in Evolutionary Robotics: First European Workshop, EvoRobot98*, pp. 39-58, 1998.

[57] H. H. Lund and O. Miglino, "From simulated to real robots", *in Proceedings of IEEE International Conference on Evolutionary Computation*, pp. 362-365, 1996.

[58] F. Hoffmann and J. C. S. Z. Montealegre, "Evolution of a tactile wall-following behavior in real time " *in The 6th Online World Conference on Soft Computing in Industrial Applications (WSC6),* 2001.

[59] G. S. Hornby, S. Takamura, J. Yokono, O. Hanagata, M. Fujita, and J. Pollack, "Evolution of controllers from a high-level simulator to a high dof robot", *in Evolvable Systems: from Biology to Hardware; Proceedings of the Third International Conference (ICES 2000)*, pp. 80-89, 2000.

ΠΑΡΑΡΤΗΜΑ

Παρακάτω αναγράφονται και επεξηγούνται τα οι κώδικες και οι συναρτήσεις που δημιουργήθηκαν για την υλοποίηση των πειραμάτων.

ΚΩΔΙΚΕΣ

InitialiseHandleObject

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%
% Στο αρχείο αυτό γίνεται η αρχικοποίηση των αντικειμένων
από τα οποία
% αποτελείται η σκηνή στο V-rep.
%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Pioneer
[errorCode, h.Pioneer] =
vrep.simxGetObjectHandle(clientID,'Pioneer',...
    vrep.simx_opmode_oneshot_wait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function for Pionner
object');
end

%Left Motor
[errorCode, h.leftMotor] =
vrep.simxGetObjectHandle(clientID,...
    'Pioneer_p3dx_leftMotor',
vrep.simx_opmode_oneshot_wait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function with left
motor Pionner object');
end

%Right Motor
[errorCode, h.rightMotor] =
vrep.simxGetObjectHandle(clientID,...
    'Pioneer_p3dx_rightMotor',
vrep.simx_opmode_oneshot_wait);
if errorCode ~= 0
```

```
    error('Error simxGetObjectHandle function with right
motor Pionner object');
end

%Goal
[errorCode, h.goal] =
vrep.simxGetObjectHandle(clientID,'goal',...
    vrep.simx_opmode_oneshot_wait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function for goal
object');
end

[errorCode, h.US1] = vrep.simxGetObjectHandle(clientID,...

'Pioneer_p3dx_ultrasonicSensor1',vrep.simx_opmode_oneshot_w
ait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function for US1
object');
end


[errorCode, h.US2] = vrep.simxGetObjectHandle(clientID,...

'Pioneer_p3dx_ultrasonicSensor2',vrep.simx_opmode_oneshot_w
ait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function for US2
object');
end

[errorCode, h.US3] = vrep.simxGetObjectHandle(clientID,...

'Pioneer_p3dx_ultrasonicSensor3',vrep.simx_opmode_oneshot_w
ait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function for US3
object');
else display('sensor3 OK!')
end

[errorCode, h.US4] = vrep.simxGetObjectHandle(clientID,...

'Pioneer_p3dx_ultrasonicSensor4',vrep.simx_opmode_oneshot_w
ait);
if errorCode ~= 0
```

```
    error('Error simxGetObjectHandle function for US4
object');
end

[errorCode, h.US5] = vrep.simxGetObjectHandle(clientID,...

'Pioneer_p3dx_ultrasonicSensor5',vrep.simx_opmode_oneshot_w
ait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function for US5
object');
end

[errorCode, h.US6] = vrep.simxGetObjectHandle(clientID,...

'Pioneer_p3dx_ultrasonicSensor6',vrep.simx_opmode_oneshot_w
ait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function for US6
object');
else display('sensor6 OK!')
end

[errorCode, h.US7] = vrep.simxGetObjectHandle(clientID,...

'Pioneer_p3dx_ultrasonicSensor7',vrep.simx_opmode_oneshot_w
ait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function for US7
object');
end

[errorCode, h.US8] = vrep.simxGetObjectHandle(clientID,...

'Pioneer_p3dx_ultrasonicSensor8',vrep.simx_opmode_oneshot_w
ait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function for US8
object');
end

[errorCode, h.US9] = vrep.simxGetObjectHandle(clientID,...

'Pioneer_p3dx_ultrasonicSensor9',vrep.simx_opmode_oneshot_w
ait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function for US9
```

```
object');
end

[errorCode, h.US10] = vrep.simxGetObjectHandle(clientID,...

'Pioneer_p3dx_ultrasonicSensor10',vrep.simx_opmode_oneshot_
wait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function for US10
object');
end

[errorCode, h.US11] = vrep.simxGetObjectHandle(clientID,...

'Pioneer_p3dx_ultrasonicSensor11',vrep.simx_opmode_oneshot_
wait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function for US11
object');
end

[errorCode, h.US12] = vrep.simxGetObjectHandle(clientID,...

'Pioneer_p3dx_ultrasonicSensor12',vrep.simx_opmode_oneshot_
wait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function for US12
object');
end

[errorCode, h.US13] = vrep.simxGetObjectHandle(clientID,...

'Pioneer_p3dx_ultrasonicSensor13',vrep.simx_opmode_oneshot_
wait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function for US13
object');
end

[errorCode, h.US14] = vrep.simxGetObjectHandle(clientID,...

'Pioneer_p3dx_ultrasonicSensor14',vrep.simx_opmode_oneshot_
wait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function for US14
object');
end
```

```
[errorCode, h.US15] = vrep.simxGetObjectHandle(clientID,...

'Pioneer_p3dx_ultrasonicSensor15',vrep.simx_opmode_oneshot_
wait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function for US15
object');
end

[errorCode, h.US16] = vrep.simxGetObjectHandle(clientID,...

'Pioneer_p3dx_ultrasonicSensor16',vrep.simx_opmode_oneshot_
wait);
if errorCode ~= 0
    error('Error simxGetObjectHandle function for US16
object');
end
```

main_for_fuzzy_without_sensors

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Ο κώδικας αυτός είναι ο κύριος κώδικας της προσομοίωσης,
που καλεί τις συναρτήσεις για τη δημιουργία των διανυσμάτων
των παραμέτρων, τη δημιουργία του πληθυσμού των ελεγκτών
και την εκτέλεση του αλγορίθμου PSO και της προσομοίωσης
στο V-rep.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear; close all; clc;

[init_array_input,init_array_output,init_array_merge,
init_controller,particles]...
 =create_x_from_fuzzy_without_sensors (1);

[particles] =
build_pop(init_array_input,init_array_output,init_array_mer
ge, init_controller,30);

[Fis]=convert_fuzzy(particles,init_array_merge,init_control
ler,init_array_input,init_array_output);
```

```
PSO_for_fuzzy_without_sensors;
```

main_for_fuzzy_with_sensors

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Ο κώδικας αυτός είναι ο κύριος κώδικας της προσομοίωσης, 
που καλεί τις συναρτήσεις για τη δημιουργία των διανυσμάτων 
των παραμέτρων, τη δημιουργία του πληθυσμού των ελεγκτών 
και την εκτέλεση του αλγορίθμου PSO και της προσομοίωσης 
στο V-rep.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear; close all; clc;

[init_array_input,init_array_output,init_array_merge,
init_controller,particles]...
 =create_x_from_fuzzy_with_sensors (1);

[particles] = 
build_pop(init_array_input,init_array_output,init_array_mer
ge, init_controller,35);

[Fis]=convert_fuzzy(particles,init_array_merge,init_control
ler,init_array_input,init_array_output);

 PSO_for_fuzzy_with_sensors;
```

PSO_for_fuzzy_without_sensors

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Ο κώδικας αυτός εκτελεί τον αλγόριθμο PSO και την 
προσομοίωση στο V-rep.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Problem Definition
CostFunction=@(simulation_steps,vp2goal,Pioneer)ob_fun_for_
```

```
fuzzy_without_sensors(simulation_steps,vp2goal,Pioneer); % 
Cost Function

[dum1,nVar]=size(init_array_merge); % Number of Decision 
Variables

VarSize=[1 nVar]; % Size of Decision Variables Matrix

nPop=size(particles,2); % Population Size (Swarm Size)

%% PSO Parameters
MaxIt=100;      % Maximum Number of Iterations

% % Constriction Coefficients
c1=2.05;     % Personal Learning Coefficient
c2=2;     % Global Learning Coefficient
phi1=2.05;
phi2=2;
phi=phi1+phi2;
kapa=0.45;
chi=2*kapa/abs(2-phi-sqrt(phi*(phi-4)));
w=chi;       % Inertia Weight
wdamp=1;     % Personal Learning Coefficient


%% Initialization
GlobalBest.Cost=inf;

for i=1:nPop

     % Initialize Position
     particles(i).Position=particles(i).merge_input_output;

     % Run simulation for number of particles
     [simulation_steps,vp2goal,Pioneer] = 
empty_arena_pso(Fis(i));
     particles(i).Pioneer=Pioneer;

     % Initialize Velocity
     particles(i).Velocity=zeros(VarSize);

     % Evaluation
     particles(i).Cost = 
CostFunction(simulation_steps,vp2goal,Pioneer);

     % Update Personal Best
     particles(i).Best.Position=particles(i).Position;
```

```
     particles(i).Best.Cost=particles(i).Cost;

     % Update Global Best
     if particles(i).Best.Cost<GlobalBest.Cost
         GlobalBest=particles(i).Best;
     end
 end

iteration.Best_Cost(1)=GlobalBest.Cost;
iter(1).part=particles;

%% PSO Main Loop
for it=2:MaxIt

    for i=1:nPop

       % Update Velocity with constriction factor x
        particles(i).Velocity = w*(particles(i).Velocity
...
            +c1*rand(VarSize).*(particles(i).Best.Position-
particles(i).Position) ...
            +c2*rand(VarSize).*(GlobalBest.Position-
particles(i).Position));

        % Update Position
        particles(i).Position = particles(i).Position +
particles(i).Velocity;

        % Run simulation for the new particle position
        x_array = particles(i).Position;
        [Fis_new] =
convert_fuzzy_new(x_array,init_controller);
        [array_input,array_output,
merge_input_output]=create_x_PSO_new(1,Fis_new);
        particles(i).array_input=array_input;
        particles(i).array_output=array_output;
        particles(i).
merge_input_output=merge_input_output;
        [simulation_steps,vp2goal,Pioneer] =
empty_arena_pso(Fis_new);
        particles(i).Pioneer=Pioneer;
        particles(i).controller=Fis_new;

        % Evaluation
        particles(i).Cost =
CostFunction(simulation_steps,vp2goal,Pioneer);
```

```
        % Update Personal Best
        if particles(i).Cost<particles(i).Best.Cost


particles(i).Best.Position=particles(i).Position;
            particles(i).Best.Cost=particles(i).Cost;
            particles(i).Best.part=[it i];

            % Update Global Best
            if particles(i).Best.Cost<GlobalBest.Cost

                GlobalBest=particles(i).Best;
           end
        end
    end

  w=w*wdamp;
  iteration.Best_Cost(it)=GlobalBest.Cost;
  iter(it).part=particles;

end

 %% Best Fuzzy
 BestSol = GlobalBest;
 x_best_array=GlobalBest.Position;
 [Fis_best] = convert_best_fuzzy(
x_best_array,init_controller);
```

PSO_for_fuzzy_with_sensors

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Ο κώδικας αυτός εκτελεί τον αλγόριθμο PSO και την
προσομοίωση στο V-rep.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Problem Definition
CostFunction=@(simulation_steps,vp2goal,Pioneer)ob_fun_for_
fuzzy_with_sensors(simulation_steps,vp2goal,Pioneer); %
Cost Function
%
```

```
CostFunction=@(simulation_steps,vp2goal,Pioneer)ob_fun_for_
fuzzy_with_sensors_2(simulation_steps,vp2goal,Pioneer); %
Cost Function

[dum1,nVar]=size(init_array_merge); % Number of Decision
Variables

VarSize=[1 nVar]; % Size of Decision Variables Matrix

nPop=size(particles,2); % Population Size (Swarm Size)

%% PSO Parameters
MaxIt=50;      % Maximum Number of Iterations

% % Constriction Coefficients
c1=2;     % Personal Learning Coefficient
c2=2.03;     % Global Learning Coefficient
phi1=2;
phi2=2.03;
phi=phi1+phi2;
kapa=0.52;
chi=2*kapa/abs(2-phi-sqrt(phi*(phi-4)));
w=chi;      % Inertia Weight
wdamp=1;


%% Initialization
GlobalBest.Cost=inf;

for i=1:nPop

     % Initialize Position
     particles(i).Position=particles(i).merge_input_output;

     % Run simulation for number of particles
    [simulation_steps,vp2goal,Pioneer] =
arena_with_obstacles(Fis(i));
     particles(i).Pioneer=Pioneer;

     % Initialize Velocity
     particles(i).Velocity=zeros(VarSize);

     % Evaluation
     particles(i).Cost =
CostFunction(simulation_steps,vp2goal,Pioneer);

     % Update Personal Best
```

```
     particles(i).Best.Position=particles(i).Position;
     particles(i).Best.Cost=particles(i).Cost;

     % Update Global Best
     if particles(i).Best.Cost<GlobalBest.Cost
         GlobalBest=particles(i).Best;
     end
 end

iteration.Best_Cost(1)=GlobalBest.Cost;
iter(1).part=particles;

%% PSO Main Loop
for it=2:MaxIt

    for i=1:nPop

        % Update Velocity with constriction factor x
        particles(i).Velocity = w*(particles(i).Velocity
...
            +c1*rand(VarSize).*(particles(i).Best.Position-
particles(i).Position) ...
            +c2*rand(VarSize).*(GlobalBest.Position-
particles(i).Position));

        % Update Position
        particles(i).Position = particles(i).Position +
particles(i).Velocity;

        % Run simulation for the new particle position
        x_array = particles(i).Position;
        [Fis_new] =
convert_fuzzy_new(x_array,init_controller);
        [array_input,array_output,
merge_input_output]=create_x_PSO_new(1,Fis_new);
        particles(i).array_input=array_input;
        particles(i).array_output=array_output;
        particles(i).
merge_input_output=merge_input_output;
        [simulation_steps,vp2goal,Pioneer] =
arena_with_obstacles(Fis_new);

        particles(i).Pioneer=Pioneer;
        particles(i).controller=Fis_new;

        % Evaluation
        particles(i).Cost =
```

```
CostFunction(simulation_steps,vp2goal,Pioneer);

        % Update Personal Best
        if particles(i).Cost<particles(i).Best.Cost


particles(i).Best.Position=particles(i).Position;
            particles(i).Best.Cost=particles(i).Cost;
            particles(i).Best.part=[it i];

            % Update Global Best
            if particles(i).Best.Cost<GlobalBest.Cost

                GlobalBest=particles(i).Best;
           end
        end

    end

  w=w*wdamp;
  iteration.Best_Cost(it)=GlobalBest.Cost;
  iter(it).part=particles;

end

 %% Best Fuzzy
 BestSol = GlobalBest;
 x_best_array=GlobalBest.Position;
 [Fis_best] = convert_best_fuzzy(
x_best_array,init_controller);
```

ΣΥΝΑΡΤΗΣΕΙΣ

create_x_from_fuzzy_without_sensors

Inputs: [number_of_particles]

Outputs: [init_array_input,init_array_output,init_array_merge, init_controller,particles]

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Η συνάρτηση αυτή παίρνει σαν είσοδο των αριθμό 1, έπειτα
επιλέγεται ο αρχικός fuzzy controller που θα
βελτιστοποιηθεί και που είναι σε μορφή .fis και θα αναλυθεί
στις παραμέτρους από τις οποίες αποτελούνται οι είσοδοι και
οι έξοδοί του, με τη μορφή διανυσμάτων. Η έξοδος της
συνάρτησης είναι τα αρχικά διανύσματα των παραμέτρων των
εισόδων, των εξόδων, του συνόλου των παραμέτρων,ο αρχικός
fuzzy controller και ένα structure με την ονομασία particle
με όλες τις απαραίτητες πληροφορίες.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function
[init_array_input,init_array_output,init_array_merge,
init_controller,particles]=create_x_from_fuzzy_without_sens
ors(number_of_particles)

% number_of_particles=1;
counter_iter_input=1;
counter_iter_output=1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Epilogh poiou controller 8elw na xrhsimopoihsw
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for count_number_of_particles=1:number_of_particles


controller(count_number_of_particles)=readfis('vrep_fuzzy_w
```

```
ithout_sensors');

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
%Epilogh twn input gia na topo8eth8oun san ari8moi sto
xrwmmosoma
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%

for counter_fuzzy=1:count_number_of_particles


[dummu1,input_number]=size(controller(counter_fuzzy).input)
;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
    %Afou dw posa input exei o controller pairnw tis
parametrous twn mf
    %gia na mpoyn sto dianisma

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%

    for counter_1=1:input_number %loop gia ka8e input


[dummy2,number_mf]=size(controller(counter_fuzzy).input(cou
nter_1).mf);

        for counter_2=1:number_mf %loop gia ta mf


number_params=length(controller(counter_fuzzy).input(counte
r_1).mf(counter_2).params);

            for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf


particles(count_number_of_particles).array_input(counter_it
er_input)=controller(counter_fuzzy).input(counter_1).mf(cou
nter_2).params(counter_3);
```

```
                counter_iter_input=counter_iter_input+1;

            end
        end
    end



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%
    %Antistoinh diadikasia kai gia ta output

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%


[dummu1,output_number]=size(controller(counter_fuzzy).outpu
t);

    for counter_1=1:output_number %loop gia ka8e output


[dummy3,number_mf]=size(controller(counter_fuzzy).output(co
unter_1).mf);

        for counter_2=1:number_mf %loop gia ta mf


number_params=length(controller(counter_fuzzy).output(count
er_1).mf(counter_2).params);

            for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf


particles(count_number_of_particles).array_output(counter_i
ter_output)=controller(counter_fuzzy).output(counter_1).mf(
counter_2).params(counter_3);

                counter_iter_output=counter_iter_output+1;
            end
        end
    end


particles(count_number_of_particles).controller=controller(
count_number_of_particles);
```

```
particles(count_number_of_particles).merge_input_output=...
    [particles(count_number_of_particles).array_input
particles(count_number_of_particles).array_output];

end
%%%%%% output: input aray, output array, controller, merge
input_output

         init_array_input=particles.array_input;
         init_array_output=particles.array_output;
         init_array_merge=particles.merge_input_output;
         init_controller=particles.controller;
         particles;
end
```

create_x_from_fuzzy_with_sensors

Inputs: [number_of_particles]

Outputs: [init_array_input,init_array_output,init_array_merge, init_controller,particles]

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%
Η συνάρηση αυτή παίρνει σαν είσοδο των αριθμό 1, έπειτα
επιλέγεται ο αρχικός fuzzy controller που θα
βελτιστοποιηθεί και που είναι σε μορφή .fis και θα αναλυθεί
στις παραμέτρους από τις οποίες αποτελούνται οι είσοδοι και
οι έξοδοί του, με τη μορφή διανυσμάτων. Η έξοδος της
συνάρτησης είναι τα αρχικά διανύσματα των παραμέτρων των
εισόδων, των εξόδων, του συνόλου των παραμέτρων,ο αρχικός
fuzzy controller και ένα structure με την ονομασία particle
με όλες τις απαραίτητες πληροφορίες.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%

function
[init_array_input,init_array_output,init_array_merge,
init_controller,particles]=create_x_from_fuzzy_with_sensors
(number_of_particles)
```

```
% number_of_particles=1;
counter_iter_input=1;
counter_iter_output=1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%
%Epilogh poiou controller 8elw na xrhsimopoihsw
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%

for count_number_of_particles=1:number_of_particles


controller(count_number_of_particles)=readfis('vrep_pso_fuz
zy');

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
%Epilogh twn input gia na topo8eth8oun san ari8moi sto
xrwmmosoma
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%

for counter_fuzzy=1:count_number_of_particles


[dummu1,input_number]=size(controller(counter_fuzzy).input)
;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
    %Afou dw posa input exei o controller pairnw tis
parametrous twn mf
    %gia na mpoyn sto dianisma

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%

    for counter_1=1:input_number %loop gia ka8e input


[dummy2,number_mf]=size(controller(counter_fuzzy).input(cou
nter_1).mf);
```

```
        for counter_2=1:number_mf %loop gia ta mf


number_params=length(controller(counter_fuzzy).input(counte
r_1).mf(counter_2).params);

            for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf


particles(count_number_of_particles).array_input(counter_it
er_input)=controller(counter_fuzzy).input(counter_1).mf(cou
nter_2).params(counter_3);

                counter_iter_input=counter_iter_input+1;

            end
        end
    end



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%
    %Antistoinh diadikasia kai gia ta output

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%


[dummu1,output_number]=size(controller(counter_fuzzy).outpu
t);

    for counter_1=1:output_number %loop gia ka8e output


[dummy3,number_mf]=size(controller(counter_fuzzy).output(co
unter_1).mf);

        for counter_2=1:number_mf %loop gia ta mf


number_params=length(controller(counter_fuzzy).output(count
er_1).mf(counter_2).params);

            for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf
```

```
particles(count_number_of_particles).array_output(counter_i
ter_output)=controller(counter_fuzzy).output(counter_1).mf(
counter_2).params(counter_3);

                counter_iter_output=counter_iter_output+1;
            end
        end
    end


particles(count_number_of_particles).controller=controller(
count_number_of_particles);


particles(count_number_of_particles).merge_input_output=...
    [particles(count_number_of_particles).array_input
particles(count_number_of_particles).array_output];

end
%%%%%% output: input aray, output array, controller, merge
input_output

         init_array_input=particles.array_input;
         init_array_output=particles.array_output;
         init_array_merge=particles.merge_input_output;
         init_controller=particles.controller;
         particles;
end
```

create_x_PSO_new

Inputs: [number_of_particles,Fis_new]

Outputs: [array_input,array_output, merge_input_output]

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%
Η συνάρηση αυτή παίρνει σαν είσοδο των αριθμό 1, και τον
νέο fuzzy controller που δημιουργείται σε κάθε επανάληψη
και είναι σε μορφή .fis. Στη συνέχεια αναλύεται στις
παραμέτρους από τις οποίες αποτελούνται οι είσοδοι και οι
έξοδοί του, με τη μορφή διανυσμάτων. Η έξοδος της
```

```
συνάρτησης είναι τα διανύσματα των παραμέτρων των εισόδων,
των εξόδων, του συνόλου των παραμέτρων.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%

function [array_input,array_output,
merge_input_output]=create_x_PSO_new(number_of_particles,Fi
s_new)

% number_of_particles=1;
counter_iter_input=1;
counter_iter_output=1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%
%Epilogh poiou controller 8elw na xrhsimopoihsw
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%

for count_number_of_particles=1:number_of_particles
% %
     controller(count_number_of_particles)=Fis_new;
%
controller(count_number_of_particles)=readfis('FLC_2');
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
%Epilogh twn input gia na topo8eth8oun san ari8moi sto
xrwmmosoma
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%

for counter_fuzzy=1:count_number_of_particles


[dummu1,input_number]=size(controller(counter_fuzzy).input)
;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
    %Afou dw posa input exei o controller pairnw tis
parametrous twn mf
    %gia na mpoyn sto dianisma

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%

    for counter_1=1:input_number %loop gia ka8e input


[dummy2,number_mf]=size(controller(counter_fuzzy).input(cou
nter_1).mf);

        for counter_2=1:number_mf %loop gia ta mf


number_params=length(controller(counter_fuzzy).input(counte
r_1).mf(counter_2).params);

            for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf


particles(count_number_of_particles).array_input(counter_it
er_input)=controller(counter_fuzzy).input(counter_1).mf(cou
nter_2).params(counter_3);

                counter_iter_input=counter_iter_input+1;

            end
        end
    end



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%
    %Antistoinh diadikasia kai gia ta output

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%


[dummu1,output_number]=size(controller(counter_fuzzy).outpu
t);

    for counter_1=1:output_number %loop gia ka8e output


[dummy3,number_mf]=size(controller(counter_fuzzy).output(co
unter_1).mf);
```

```
        for counter_2=1:number_mf %loop gia ta mf


number_params=length(controller(counter_fuzzy).output(count
er_1).mf(counter_2).params);

            for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf


particles(count_number_of_particles).array_output(counter_i
ter_output)=controller(counter_fuzzy).output(counter_1).mf(
counter_2).params(counter_3);

                counter_iter_output=counter_iter_output+1;
            end
        end
    end


particles(count_number_of_particles).controller=controller(
count_number_of_particles);


particles(count_number_of_particles).merge_input_output=...
    [particles(count_number_of_particles).array_input
particles(count_number_of_particles).array_output];

end
%%%%%% output: input aray, output array, controller, merge
input_output

         array_input=particles.array_input;
         array_output=particles.array_output;
         merge_input_output=particles.merge_input_output;
         init_controller=particles.controller;
end
```

convert_fuzzy

Inputs: [particles,init_array_merge,init_controller,init_array_input,init_array_output]

Outputs: [Fis]

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Η συνάρτηση αυτή αυτή παίρνει σαν εισόδο τα διανύσματα των
παραμέτρων των εισόδων και των εξόδων του αρχικού ελεγκτή
και τον μετατρέπει σε ασαφή ελεγκτή με τη μορφή .fis που
είναι και η έξοδος της συνάρτησης.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


 function
[Fis]=convert_fuzzy(particles,init_array_merge,init_control
ler,init_array_input,init_array_output);
min_spread=0.00001;

%H e3wterikh anakyklwsh 8a doulepsei gia to noumero twn
%particles. Ena paricle --> Enas Elegkths
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%

for counter_id=1:size(particles,2);

counter_gene_input=1;
counter_gene_input_output=1;
counter_gene_output=1;

   controller(counter_id)=init_controller;

merge(counter_id).input_output=particles(counter_id).merge_
input_output;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
    %Epilogh twn input

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%


[dummu1,input_number]=size(controller(counter_id).input);



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%
    %Afou dw posa input exei o controller pairnw tis
parametrous twn mf

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%

    for counter_1=1:input_number %loop gia ka8e input

        %%%%%%%%%%%%%%%%%%%Edw bazw ton elegxo gia to
flag%%%%%%%%%%%%

        a=controller(counter_id).input(counter_1).name;
        b=double(a);
        b=sum(b);

        if b==410

            checker=1;

        else

            checker=2;

        end



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%

[dummy2,number_mf]=size(controller(counter_id).input(counte
r_1).mf);


        for counter_2=1:number_mf %loop gia ta mf


number_params=length(controller(counter_id).input(counter_1
).mf(counter_2).params);


            for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf
                if checker==1
                    if counter_2==1
```

```
                        m_in(counter_3)=1;
                    end
                else


m_in(counter_3)=merge(counter_id).input_output(counter_gene
_input_output);
                end


counter_gene_input_output=counter_gene_input_output+1;

            end


            if number_params==4

                m1=m_in(1);
                m2=m_in(2);
                m3=m_in(3);
                m4=m_in(4);

                if (m1>=m2)
                    m1=m2-min_spread;
                end

                if (m2>=m3)
                    m3=m2+min_spread;
                end

                if (m3>=m4)
                    m4=m3+min_spread;
                end

                m_in(1)=m1;
                m_in(2)=m2;
                m_in(3)=m3;
                m_in(4)=m4;


                for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf


controller(counter_id).input(counter_1).mf(counter_2).param
s(counter_3)=m_in(counter_3);
```

```
                end
                clear m1 m2 m3 m4 m_in
            end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            %Prosoxh bazw elegxo gia na dw ti 8a ginei me 0
kai to 1

%1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%

            if checker==2 %%%%%%%%%%%%


                if number_params==3

                    m1=m_in(1);
                    m2=m_in(2);
                    m3=m_in(3);


                    if  (m1>=m2)&(m2<m3)

                        m1=m2-min_spread;

                    end


                    if    (m1>=m2)&(m2>=m3)

                        m1=m2-min_spread;
                        m3=m2+min_spread;

                    end


                    if     (m1<m2)&(m2>=m3)

                        m3=m2+min_spread;

                    end


                    m_in(1)=m1;
                    m_in(2)=m2;
                    m_in(3)=m3;
```

```
                    for counter_3=1:number_params %loop gia
tis parametrous toy ka8e mf


controller(counter_id).input(counter_1).mf(counter_2).param
s(counter_3)=m_in(counter_3);
                    end

                    clear m1 m2 m3 m_in

                end

            else

                for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf


controller(counter_id).input(counter_1).mf(counter_2).param
s(counter_3)=m_in(counter_3);

                end

            end


            clear m1 m2 m3 m_in

        end
    end

    %%       to idio gia ta output


[dummu1,output_number]=size(controller(counter_id).output);



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
    %Afou dw posa output exei o controller pairnw tis
parametrous twn mf
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%

    for counter_1=1:output_number %loop gia ka8e output

        %%%%%%%%%%%%%%%%%%%Edw bazw ton elegxo gia to
flag%%%%%%%%%%%%

        a=controller(counter_id).output(counter_1).name;
        b=double(a);
        b=sum(b);

        if b==410

            checker=1;

        else

            checker=2;

        end



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%

[dummy2,number_mf]=size(controller(counter_id).output(count
er_1).mf);


        for counter_2=1:number_mf %loop gia ta mf


number_params=length(controller(counter_id).output(counter_
1).mf(counter_2).params);


            for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf
                if checker==1
                    if counter_2==1
                        m_in(counter_3)=1;
                    end
                else
```

```
m_in(counter_3)=merge(counter_id).input_output(counter_gene
_input_output);
                end


counter_gene_input_output=counter_gene_input_output+1;

            end


            if number_params==4

                m1=m_in(1);
                m2=m_in(2);
                m3=m_in(3);
                m4=m_in(4);

                if (m1>=m2)
                    m1=m2-min_spread;
                end

                if (m2>=m3)
                    m3=m2+min_spread;
                end

                if (m3>=m4)
                    m4=m3+min_spread;
                end

                m_in(1)=m1;
                m_in(2)=m2;
                m_in(3)=m3;
                m_in(4)=m4;


                for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf


controller(counter_id).output(counter_1).mf(counter_2).para
ms(counter_3)=m_in(counter_3);

                end
                clear m1 m2 m3 m4 m_in
            end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            %Prosoxh bazw elegxo gia na dw ti 8a ginei me 0
kai to 1

%1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%

            if checker==2 %%%%%%%%%%%%


                if number_params==3

                    m1=m_in(1);
                    m2=m_in(2);
                    m3=m_in(3);


                    if  (m1>=m2)&(m2<m3)

                        m1=m2-min_spread;

                    end


                    if    (m1>=m2)&(m2>=m3)

                        m1=m2-min_spread;
                        m3=m2+min_spread;

                    end


                    if     (m1<m2)&(m2>=m3)

                        m3=m2+min_spread;

                    end


                    m_in(1)=m1;
                    m_in(2)=m2;
                    m_in(3)=m3;



                    for counter_3=1:number_params %loop gia
```

```
tis parametrous toy ka8e mf


controller(counter_id).output(counter_1).mf(counter_2).para
ms(counter_3)=m_in(counter_3);
                    end

                    clear m1 m2 m3 m_in

                end

            else

                for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf


controller(counter_id).output(counter_1).mf(counter_2).para
ms(counter_3)=m_in(counter_3);

                end

            end


            clear m1 m2 m3 m_in

        end
    end

end

              Fis=controller;
end
```

convert_fuzzy_new

Inputs: [x_array,init_controller]

Outputs: [Fis_new]

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Η συνάρτηση αυτή αυτή παίρνει σαν εισόδο τα διανύσματα των
```

```
παραμέτρων των εισόδων και των εξόδων του νέου ελεγκτή που
δημιουργείται σε κάθε επανάληψη καιμε τη βοήθεια της δομής
του αρχικού ελεγκτή τον μετατρέπει σε ελεγκτή με τη μορφή
.fis που είναι και η έξοδος της συνάρτησης
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [Fis_new] =
convert_fuzzy_new(x_array,init_controller )
min_spread=0.00001;

%H e3wterikh anakyklwsh 8a doulepsei gia to noumero twn
%particles. Ena paricle --> Enas Elegkths
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%

for counter_id=1:size(x_array,1);

counter_gene_input=1;
counter_gene_input_output=1;
counter_gene_output=1;

   controller(counter_id)=init_controller;
   merge(counter_id).input_output=x_array;


[dummu1,input_number]=size(controller(counter_id).input);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
    %Afou dw posa input exei o controller pairnw tis
parametrous twn mf

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%

    for counter_1=1:input_number %loop gia ka8e input

        %%%%%%%%%%%%%%%%%%%Edw bazw ton elegxo gia to
flag%%%%%%%%%%%%

        a=controller(counter_id).input(counter_1).name;
        b=double(a);
        b=sum(b);

        if b==410

            checker=1;
```

```
        else

            checker=2;

        end


[dummy2,number_mf]=size(controller(counter_id).input(counte
r_1).mf);


        for counter_2=1:number_mf %loop gia ta mf


number_params=length(controller(counter_id).input(counter_1
).mf(counter_2).params);


            for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf
                if checker==1
                    if counter_2==1
                        m_in(counter_3)=1;
                    end
                else


m_in(counter_3)=merge(counter_id).input_output(counter_gene
_input_output);
                end


counter_gene_input_output=counter_gene_input_output+1;

            end


            if number_params==4

                m1=m_in(1);
                m2=m_in(2);
                m3=m_in(3);
                m4=m_in(4);

                if (m1>=m2)
                    m1=m2-min_spread;
```

```
                end

                if (m2>=m3)
                    m3=m2+min_spread;
                end

                if (m3>=m4)
                    m4=m3+min_spread;
                end

                m_in(1)=m1;
                m_in(2)=m2;
                m_in(3)=m3;
                m_in(4)=m4;


                for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf


controller(counter_id).input(counter_1).mf(counter_2).param
s(counter_3)=m_in(counter_3);

                end
                clear m1 m2 m3 m4 m_in
            end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            %Prosoxh bazw elegxo gia na dw ti 8a ginei me 0
kai to 1

%1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%

            if checker==2 %%%%%%%%%%%%


                if number_params==3

                    m1=m_in(1);
                    m2=m_in(2);
                    m3=m_in(3);


                    if (m1>=m2)&(m2<m3)
```

```
                        m1=m2-min_spread;

                    end


                    if    (m1>=m2)&(m2>=m3)

                        m1=m2-min_spread;
                        m3=m2+min_spread;

                    end


                    if     (m1<m2)&(m2>=m3)

                        m3=m2+min_spread;

                    end


                    m_in(1)=m1;
                    m_in(2)=m2;
                    m_in(3)=m3;



                    for counter_3=1:number_params %loop gia
tis parametrous toy ka8e mf


controller(counter_id).input(counter_1).mf(counter_2).param
s(counter_3)=m_in(counter_3);
                    end

                    clear m1 m2 m3 m_in

                end

            else

                for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf


controller(counter_id).input(counter_1).mf(counter_2).param
s(counter_3)=m_in(counter_3);
```

```
                end

            end


            clear m1 m2 m3 m_in

        end
    end

    %%       to idio gia ta output


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
    %Epilogh twn input gia na topo8eth8oun san ari8moi

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%


[dummu1,output_number]=size(controller(counter_id).output);



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
    %Afou dw posa output exei o controller pairnw tis
parametrous twn mf

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%

    for counter_1=1:output_number %loop gia ka8e output

        %%%%%%%%%%%%%%%%%%%Edw bazw ton elegxo gia to
flag%%%%%%%%%%%%

        a=controller(counter_id).output(counter_1).name;
        b=double(a);
        b=sum(b);

        if b==410

            checker=1;

        else
```

```
            checker=2;

        end


[dummy2,number_mf]=size(controller(counter_id).output(count
er_1).mf);


        for counter_2=1:number_mf %loop gia ta mf


number_params=length(controller(counter_id).output(counter_
1).mf(counter_2).params);


            for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf
                if checker==1
                    if counter_2==1
                        m_in(counter_3)=1;
                    end
                else


m_in(counter_3)=merge(counter_id).input_output(counter_gene
_input_output);
                end


counter_gene_input_output=counter_gene_input_output+1;

            end


            if number_params==4

                m1=m_in(1);
                m2=m_in(2);
                m3=m_in(3);
                m4=m_in(4);

                if (m1>=m2)
                    m1=m2-min_spread;
                end
```

```
                if (m2>=m3)
                    m3=m2+min_spread;
                end

                if (m3>=m4)
                    m4=m3+min_spread;
                end

                m_in(1)=m1;
                m_in(2)=m2;
                m_in(3)=m3;
                m_in(4)=m4;


                for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf


controller(counter_id).output(counter_1).mf(counter_2).para
ms(counter_3)=m_in(counter_3);

                end
                clear m1 m2 m3 m4 m_in
            end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            %Prosoxh bazw elegxo gia na dw ti 8a ginei me 0
kai to 1

%1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%

            if checker==2 %%%%%%%%%%%%


                if number_params==3

                    m1=m_in(1);
                    m2=m_in(2);
                    m3=m_in(3);


                    if (m1>=m2)&(m2<m3)

                        m1=m2-min_spread;
```

```
                    end


                    if    (m1>=m2)&(m2>=m3)

                        m1=m2-min_spread;
                        m3=m2+min_spread;

                    end


                    if     (m1<m2)&(m2>=m3)

                        m3=m2+min_spread;

                    end


                    m_in(1)=m1;
                    m_in(2)=m2;
                    m_in(3)=m3;



                    for counter_3=1:number_params %loop gia
tis parametrous toy ka8e mf


controller(counter_id).output(counter_1).mf(counter_2).para
ms(counter_3)=m_in(counter_3);
                    end

                    clear m1 m2 m3 m_in

                end

            else

                for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf


controller(counter_id).output(counter_1).mf(counter_2).para
ms(counter_3)=m_in(counter_3);

                end
```

```
            end


            clear m1 m2 m3 m_in

        end
    end

end

              Fis_new=controller;
end
```

convert best fuzzy

Inputs : [x_best_array,init_controller]

Outputs : [Fis_best]

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Έχει ως είσοδο τα διανύσματα των τιμών των παραμέτρων της
εισόδου και της εξόδου του καλύτερου ελεγκτή και με τη
βοήθεια της δομής του αρχικού ελεγκτή σε ασαφή ελεγκτή
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [Fis_best] = convert_best_fuzzy
( x_best_array,init_controller )

min_spread=0.00001;

%H e3wterikh anakyklwsh 8a doulepsei gia to noumero twn
%particles. Ena paricle --> Enas Elegkths
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%

for counter_id=1:size(x_best_array,1);


counter_gene_input=1;
counter_gene_input_output=1;
counter_gene_output=1;

   controller(counter_id)=init_controller;
   merge(counter_id).input_output=x_best_array;
```

```
  [dummu1,input_number]=size(controller(counter_id).input);



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
    %Afou dw posa input exei o controller pairnw tis
parametrous twn mf
    %gia na mpoyn sto xrwmoswma

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%

    for counter_1=1:input_number %loop gia ka8e input

        %%%%%%%%%%%%%%%%%%%Edw bazw ton elegxo gia to
flag%%%%%%%%%%%%

        a=controller(counter_id).input(counter_1).name;
        b=double(a);
        b=sum(b);

        if b==410

            checker=1;

        else

            checker=2;

        end




%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%

[dummy2,number_mf]=size(controller(counter_id).input(counte
r_1).mf);


        for counter_2=1:number_mf %loop gia ta mf


number_params=length(controller(counter_id).input(counter_1
```

```
).mf(counter_2).params);


            for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf
                if checker==1
                    if counter_2==1
                        %
controller(cre_count).input(counter_1).mf(counter_2).params
(counter_3)=merge_input_output(counter_gene_input_output);


%                         m_in(counter_3)=0;
%                     else
                        m_in(counter_3)=1;
                    end
                else


m_in(counter_3)=merge(counter_id).input_output(counter_gene
_input_output);
                end


counter_gene_input_output=counter_gene_input_output+1;

            end


            if number_params==4

                m1=m_in(1);
                m2=m_in(2);
                m3=m_in(3);
                m4=m_in(4);

                if (m1>=m2)
                    m1=m2-min_spread;
                end

                if (m2>=m3)
                    m3=m2+min_spread;
                end

                if (m3>=m4)
                    m4=m3+min_spread;
                end
```

```
                m_in(1)=m1;
                m_in(2)=m2;
                m_in(3)=m3;
                m_in(4)=m4;


                for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf


controller(counter_id).input(counter_1).mf(counter_2).param
s(counter_3)=m_in(counter_3);

                end
                clear m1 m2 m3 m4 m_in
            end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            %Prosoxh bazw elegxo gia na dw ti 8a ginei me 0
kai to 1

%1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%

            if checker==2 %%%%%%%%%%%%


                if number_params==3

                    m1=m_in(1);
                    m2=m_in(2);
                    m3=m_in(3);


                    if  (m1>=m2)&(m2<m3)

                        m1=m2-min_spread;

                    end


                    if    (m1>=m2)&(m2>=m3)

                        m1=m2-min_spread;
                        m3=m2+min_spread;
```

```
                    end


                    if     (m1<m2)&&(m2>=m3)

                        m3=m2+min_spread;

                    end


                    m_in(1)=m1;
                    m_in(2)=m2;
                    m_in(3)=m3;



                    for counter_3=1:number_params %loop gia
tis parametrous toy ka8e mf


controller(counter_id).input(counter_1).mf(counter_2).param
s(counter_3)=m_in(counter_3);
                    end

                    clear m1 m2 m3 m_in

                end

            else

                for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf


controller(counter_id).input(counter_1).mf(counter_2).param
s(counter_3)=m_in(counter_3);

                end

            end


            clear m1 m2 m3 m_in

        end
    end
```

```
    %%       to idio gia ta output

%    controller(counter_id)=init_controller;
%    merge(counter_id).input_output=x_best_array;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
    %Epilogh twn input gia na topo8eth8oun san ari8moi sto
xrwmmosoma

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%


[dummu1,output_number]=size(controller(counter_id).output);



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
    %Afou dw posa output exei o controller pairnw tis
parametrous twn mf
    %gia na mpoyn sto xrwmoswma

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%

    for counter_1=1:output_number %loop gia ka8e output

        %%%%%%%%%%%%%%%%%%%Edw bazw ton elegxo gia to
flag%%%%%%%%%%%%

        a=controller(counter_id).output(counter_1).name;
        b=double(a);
        b=sum(b);

        if b==410

            checker=1;

        else

            checker=2;

        end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%

[dummy2,number_mf]=size(controller(counter_id).output(count
er_1).mf);


        for counter_2=1:number_mf %loop gia ta mf


number_params=length(controller(counter_id).output(counter_
1).mf(counter_2).params);


            for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf
                if checker==1
                    if counter_2==1
                        %
controller(cre_count).input(counter_1).mf(counter_2).params
(counter_3)=merge_input_output(counter_gene_input_output);


%                           m_in(counter_3)=0;
%                     else
                        m_in(counter_3)=1;
                    end
                else


m_in(counter_3)=merge(counter_id).input_output(counter_gene
_input_output);
                end


counter_gene_input_output=counter_gene_input_output+1;

            end


            if number_params==4

                m1=m_in(1);
```

```
                m2=m_in(2);
                m3=m_in(3);
                m4=m_in(4);

                if (m1>=m2)
                    m1=m2-min_spread;
                end

                if (m2>=m3)
                    m3=m2+min_spread;
                end

                if (m3>=m4)
                    m4=m3+min_spread;
                end

                m_in(1)=m1;
                m_in(2)=m2;
                m_in(3)=m3;
                m_in(4)=m4;


                for counter_3=1:number_params %loop gia tis
parametrous toy ka8e mf


controller(counter_id).output(counter_1).mf(counter_2).para
ms(counter_3)=m_in(counter_3);

                end
                clear m1 m2 m3 m4 m_in
            end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            %Prosoxh bazw elegxo gia na dw ti 8a ginei me 0
kai to 1

%1%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%

            if checker==2 %%%%%%%%%%%%


                if number_params==3

                    m1=m_in(1);
```

```
                    m2=m_in(2);
                    m3=m_in(3);


                    if (m1>=m2)&(m2<m3)

                        m1=m2-min_spread;

                    end


                    if   (m1>=m2)&(m2>=m3)

                        m1=m2-min_spread;
                        m3=m2+min_spread;

                    end


                    if    (m1<m2)&(m2>=m3)

                        m3=m2+min_spread;

                    end


                    m_in(1)=m1;
                    m_in(2)=m2;
                    m_in(3)=m3;



                    for counter_3=1:number_params %loop gia
tis parametrous toy ka8e mf


controller(counter_id).output(counter_1).mf(counter_2).para
ms(counter_3)=m_in(counter_3);
                    end

                    clear m1 m2 m3 m_in

                end

            else

                for counter_3=1:number_params %loop gia tis
```

```
parametrous toy ka8e mf


controller(counter_id).output(counter_1).mf(counter_2).para
ms(counter_3)=m_in(counter_3);

                end

            end


            clear m1 m2 m3 m_in

        end
    end

end

       Fis_best=controller;

end
```

build_pop

Inputs: [init_array_input,init_array_output,init_array_merge, init_controller,num]

Outputs: [particles]

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Η συνάρτηση αυτή παίρνει σαν είσοδο τον αρχικό fuzzy
controller τη μορφή .fis, τα αρχικά διανύσματα με τις
παραμέτρους των εισόδων, των εξόδων και του συνόλου των
παραμέτρων του αρχικού ελεγκτή καθώς και τον αριθμό του
επιθυμητού πλήθους των σωματιδίων. Στη συνέχεια δημιουργεί
τον πληθυσμό των σωματιδίων-ελεγκτών και σαν εξοδο δίνει τα
σωματίδια-ελεγκτές με τη μορφή διανυσμάτων.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function [ particles ] = build_pop
( init_array_input,init_array_output,init_array_merge,
init_controller,num)
```

```
        controller=init_controller;
        flag=1;


for      counter_init=1:num

        particles(counter_init).controller=controller;


    if flag==1

            particles(counter_init).merge_input_output =
init_array_merge;
            particles(counter_init).array_input =
init_array_input;
            particles(counter_init).array_output =
init_array_output;

    else

           [dummu1,input_number]=size(controller.input);


  for     counter_1=1:input_number %loop gia ka8e input


min_input(counter_1)=controller.input(counter_1).range(1);

max_input(counter_1)=controller.input(counter_1).range(2);


[dummy2,number_mf_in]=size(controller.input(counter_1).mf);

      for counter_2=1:number_mf_in %loop gia ta mf


number_params_1(counter_2)=length(controller.input(counter_
1).mf(counter_2).params);
%
      end

             size_of_input(counter_1)=sum(number_params_1);
             number_params_1=[];


      for    zita_1=1:size_of_input(counter_1)
```

```
particles(counter_init).input(counter_1).x_out(zita_1)=(min
_input(counter_1)+rand*(max_input(counter_1)-
min_input(counter_1)));
      end



table_input(1,counter_1)=struct2table(particles(counter_ini
t).input(counter_1));
             array_input=table2array(table_input);

particles(counter_init).array_input=array_input;

  end

        %%%%%%%%%%%%%%% output %%%%%%%%%%%%%

            [dummu3,output_number]=size(controller.output);


for         counter_4=1:output_number %loop gia ka8e output


min_output(counter_4)=controller.output(counter_4).range(1)
;

max_output(counter_4)=controller.output(counter_4).range(2)
;


[dummy4,number_mf_out]=size(controller.output(counter_4).mf
);

      for   counter_5=1:number_mf_out %loop gia ta mf


number_params_2(counter_5)=length(controller.output(counter
_4).mf(counter_5).params);


      end

            size_of_output(counter_4)=sum(number_params_2);
            number_params_2=[];
```

```
        for    zita_2=1:size_of_output(counter_4)


particles(counter_init).output(counter_4).y_out(zita_2)=(mi
n_output(counter_4)+rand*(max_output(counter_4)-
min_output(counter_4)));

        end


table_output(1,counter_4)=struct2table(particles(counter_in
it).output(counter_4));
             array_output=table2array(table_output);

particles(counter_init).array_output=array_output;
end


particles(counter_init).merge_input_output=[array_input
array_output];


    end
             flag=2;


end
             particles;

end
```

ob_fun_for_fuzzy_without_sensors

Inputs: [simulation_steps,vp2goal,Pioneer]

Outputs: [z]

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Η συνάρτηση αυτή έχει σαν είσοδο τα δεδομένα της
προσομοίωσης, και σαν έξοδο μια τιμή με την οποία
αξιολογείται η απόδοση του ελεγκτή.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
function
z=ob_fun_for_fuzzy_without_sensors(simulation_steps,vp2goal
,Pioneer)

 global j
 global final_route

 mean_left_speed=mean(Pioneer.Data.speed.leftwheel);
 mean_right_speed=mean(Pioneer.Data.speed.rightwheel);

 V=(mean([mean_left_speed mean_right_speed]))/30;
 dV=(mean(abs((Pioneer.Data.speed.leftwheel-
Pioneer.Data.speed.rightwheel)/30)));

   if j<=5

z=(0*vp2goal)+10^4*(1*6/j)/((250/simulation_steps)*(1-
sqrt(dV)));
   else

z=(0*vp2goal)+(1*6/j)*final_route/(V*(250/simulation_steps)
*(1-sqrt(dV)));
   end

end
```

ob_fun_for_fuzzy_with_sensors

Inputs: [simulation_steps,vp2goal,Pioneer]
Outputs: [z]

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Η συνάρτηση αυτή έχει σαν είσοδο τα δεδομένα της
προσομοίωσης, και σαν έξοδο μια τιμή με την οποία
αξιολογείται η απόδοση του ελεγκτή.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function
z=ob_fun_for_fuzzy_with_sensors(simulation_steps,vp2goal,Pi
oneer)
```

```
     global final_route;

      mean_left_speed=mean(Pioneer.Data.speed.leftwheel);
      mean_right_speed=mean(Pioneer.Data.speed.rightwheel);

      V=(mean([mean_left_speed mean_right_speed]))/30;


      dV=(mean(abs((Pioneer.Data.speed.leftwheel-
Pioneer.Data.speed.rightwheel)/30)));


i_sensors=min([min([Pioneer.Data.sensors.centralInput])
min([Pioneer.Data.sensors.leftInput])...
                min([Pioneer.Data.sensors.rightInput])]);

      a=[Pioneer.Data.sensors.leftInput
Pioneer.Data.sensors.centralInput...
         Pioneer.Data.sensors.rightInput];

      [m,~]=find(a<=0.5);
       m1=length(m);
       near_coef= abs(m1/(3*(simulation_steps-1))); % oso
megalitero toso xirotero


      if
(vp2goal<=0.18)&&(i_sensors>0.25)&&(near_coef<=0.4)

           z=1/(V*(1-(final_route/40))*(1-
sqrt(dV))*(i_sensors)...
               *(1-(vp2goal/0.2))*(1-near_coef)*sqrt((1-
(simulation_steps/1400))));

           if (z==inf)||(z<=0)
               z=10^6;
           end
      else
           z=10^4/((1-(vp2goal/100))*(1-near_coef));

           if (z==inf)||(z<=0)
               z=10^6;
           end
      end

end
```

ob_fun_for_fuzzy_with_sensors_2

Inputs: [simulation_steps,vp2goal,Pioneer]
Outputs: [z]

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Η συνάρτηση αυτή έχει σαν είσοδο τα δεδομένα της
προσομοίωσης, και σαν έξοδο μια τιμή με την οποία
αξιολογείται η απόδοση του ελεγκτή.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function
z=ob_fun_for_fuzzy_with_sensors_2(simulation_steps,vp2goal,
Pioneer)



sens=[Pioneer(1).Data.sensors.s1;Pioneer(1).Data.sensors.s2
;...

Pioneer(1).Data.sensors.s3;Pioneer(1).Data.sensors.s4;...

Pioneer(1).Data.sensors.s5;Pioneer(1).Data.sensors.s6;...

Pioneer(1).Data.sensors.s7;Pioneer(1).Data.sensors.s8];

       m=find(sens<1);
       m1=length(m);


      if (vp2goal<=0.15)

                z=m1;

      else

                z=vp2goal*10^5;

      end

end
```

empty_arena_pso

Inputs: [Fis]

Outputs: [simulation_steps,vp2goal,Pioneer]

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Η συνάρτηση παίρνει σαν είσοδο έναν fuzzy controller
εκτελεί την προσομοίωση στο V-rep και δίνει σαν έξοδο τα
steps της προσομοίωσης, την τελική απόσταση από τον στόχο
και ένα structure Pioneer με όλα τα δεδομένα της
προσομοίωσης.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [simulation_steps,vp2goal,Pioneer] =
empty_arena_pso(Fis)
% Connection Matlab/Vrep
disp('Program strated')
vrep=remApi('remoteApi');%build the object and load the
library
setenv('VREP','C:\Program Files (x86)\V-REP3\V-
REP_PRO_EDU')
v=VREP();
v.loadscene('empty_arena');
v.simstart();

%build the object and load the library
%initialiseRoboticsToolbox
%We use the loopback
%IP adress / connection Port / wait Until Connected /
%do Not Reconnect Once Disconnected / time Out In (ms)
vrep.simxFinish(-1); % close all opened connections
clientID=vrep.simxStart('127.0.0.1',19999,true,true,2000,5)
;

if clientID == -1
    error('the connection to the server was not possible');
else
    disp('Connected to remote API server');

    %Requests a start of a simulation (clientID /
operationMode)
    [errorCode]=vrep.simxStartSimulation(clientID,...
        vrep.simx_opmode_oneshot_wait);
```

```
    if errorCode ~=0
        error('Vrep error of requests a start of a
simulation');
    else
        disp('OK request simulation');
    end
end

%% Retrieves an object handle based on his name
initialiseHandleObject

%% Retrieves goal position
[errorCode, goal_pos] =
vrep.simxGetObjectPosition(clientID,h.goal,-1,...
    vrep.simx_opmode_streaming);%first call: streaming
pause(0.1)
[errorCode, goal_pos] =
vrep.simxGetObjectPosition(clientID, h.goal, -1,...
    vrep.simx_opmode_buffer);%other call: buffer
if errorCode ~= 0
    error('Error with simxGetObjectPosition for goal');
end

%% Retrieve the position of the car
[errorCode, actu_pos] =
vrep.simxGetObjectPosition(clientID,h.Pioneer,-1,...
    vrep.simx_opmode_streaming);%first call: streaming
pause(0.1) %Why we need a Pause?
[errorCode, actu_pos] =
vrep.simxGetObjectPosition(clientID, h.Pioneer,-1,...
    vrep.simx_opmode_buffer);%other call: buffer
if errorCode ~= 0
    error('Error, can not find the actual position of the
car');
end

%% Retrieve the orientation (Euler angles) of the car
[errorCode,actu_orien]=vrep.simxGetObjectOrientation(client
ID,h.Pioneer,...
    -1,vrep.simx_opmode_streaming);
pause(0.1) %Why we need a Pause?
[errorCode,actu_orien]=vrep.simxGetObjectOrientation(client
ID,h.Pioneer,...
    -1,vrep.simx_opmode_buffer);
if errorCode ~= 0
    error('Error, can not find the actual orientation of
the car');
```

```
end

%%
motorSpeed = [0 0];%Initialize motor speed (1) left, (2)
Right

xD=[-2 -1  2  0 -1];
yD=[2  -2  1  0  1];

i=1;
global j;
j=1;
global final_route;

stop = 0;
steps=1;

[errorCode, actu_pos] =
vrep.simxGetObjectPosition(clientID,...
 h.Pioneer, -1, vrep.simx_opmode_buffer);

prev_x_position=actu_pos(1);
prev_y_position=actu_pos(2);



%% Control loop
tic
while (vrep.simxGetConnectionId(clientID)~=-
1)&&(stop~=1)&&(steps<800)&&(toc<45)

     % Set the position of goal
      goal_position=[xD(j),yD(j),0];

      [errorCode]=vrep.simxSetObjectPosition(clientID,
h.goal,-1,...
        goal_position(i,:),vrep.simx_opmode_oneshot_wait);
      if errorCode ~= 0

         error('Error with simxSetObjectPosition for
goal');
      end

   %Actualise actual orientation and actual position of the
car

[errorCode,actu_orien]=vrep.simxGetObjectOrientation(client
```

```
ID,...
        h.Pioneer,-1,vrep.simx_opmode_buffer);
    [errorCode, actu_pos] =
vrep.simxGetObjectPosition(clientID,...
        h.Pioneer, -1, vrep.simx_opmode_buffer);
    [errorCode, goal_pos] =
vrep.simxGetObjectPosition(clientID, h.goal,...
        -1, vrep.simx_opmode_buffer);

    Pioneer(1).Data.x_position(steps)=actu_pos(1);
    x_position=actu_pos(1);

    Pioneer(1).Data.y_position(steps)=actu_pos(2);
    y_position=actu_pos(2);

    %Calcul the route
    d(steps)=sqrt((y_position-
prev_y_position)^2+(x_position-prev_x_position)^2);

    prev_x_position=x_position;
    prev_y_position=y_position;

    route=sum(d);
    Pioneer(1).Data.route(steps)=route;

    %Calcul distance from vehicle point to goal
    vp2goal = sqrt((goal_position(i,1)-actu_pos(1))^2+...
        (goal_position(i,2)-actu_pos(2))^2);

    Pioneer(1).Data.distance(steps)=vp2goal;

    %Calcul angle the vehicule position orientation and the
goal
    curdiff = atan2(goal_position(i,2)-actu_pos(2),...
        goal_position(i,1)-actu_pos(1));

    %Calcul error angle
    angle_error =  actu_orien(3)-curdiff;

    if (angle_error < -pi) %turn left
            angle_error = angle_error + 2*pi;
        elseif (angle_error >= pi)%turn right
            angle_error = angle_error - 2*pi;
    end

     angle_error_deg=rad2deg(angle_error);
     Pioneer(1).Data.angleError(steps)=angle_error_deg;
```

```
     output = evalfis (double(angle_error_deg),Fis)

     motorSpeed(1)=output(1);
     motorSpeed(2)=output(2);

     Pioneer(1).Data.speed.leftwheel(steps)= motorSpeed(1);
     Pioneer(1).Data.speed.rightwheel(steps)=
motorSpeed(2);


[errorCode]=vrep.simxSetJointTargetVelocity(clientID,h.left
Motor,...
        motorSpeed(1),vrep.simx_opmode_streaming);

[errorCode]=vrep.simxSetJointTargetVelocity(clientID,h.righ
tMotor,...
        motorSpeed(2),vrep.simx_opmode_streaming);

      steps=steps+1;
      Pioneer(1).Data.steps(steps)=steps;

      final_route=sum(d);
      Pioneer(1).Data.final_route(steps)=final_route;

      if(vp2goal < 0.1)
           j=j+1;
        if j>5
           stop=1;
        end
      end
toc
end

%% Stop and end of program
[errorCode]=vrep.simxSetJointTargetVelocity(clientID,h.left
Motor,0,...
    vrep.simx_opmode_streaming);
[errorCode]=vrep.simxSetJointTargetVelocity(clientID,h.righ
tMotor,0,...
    vrep.simx_opmode_streaming);

v.simstop();
vrep.simxFinish(-1);

simulation_steps= Pioneer(1).Data.steps(steps);
Pioneer(1).Data.final_route=final_route;
```

```
disp('program end')
end
```

arena_with_obstacles

Inputs: [Fis]

Outputs: [simulation_steps,vp2goal,Pioneer]

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Η συνάρτηση παίρνει σαν είσοδο έναν fuzzy controller
εκτελεί την προσομοίωση στο V-rep και δίνει σαν έξοδο τα
steps της προσομοίωσης, την τελική απόσταση από τον στόχο
και ένα structure Pioneer με όλα τα δεδομένα της
προσομοίωσης.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [simulation_steps,vp2goal,Pioneer] =
arena_with_obstacles (Fis)
%% Connection Matlab/Vrep

disp('Program strated')
vrep=remApi('remoteApi');%build the object and load the
library
setenv('VREP','C:\Program Files (x86)\V-REP3\V-
REP_PRO_EDU')
v=VREP();
v.loadscene('pso_arena_10');
v.simstart();

%initialiseRoboticsToolbox
%We use the loopback
%IP adress / connection Port / wait Until Connected /
%do Not Reconnect Once Disconnected / time Out In (ms)
vrep.simxFinish(-1); % close all opened connections
clientID=vrep.simxStart('127.0.0.1',19999,true,true,2000,5)
;


if clientID == -1
    error('the connection to the server was not possible');
```

```
else
    disp('Connected to remote API server');

    %Requests a start of a simulation (clientID /
operationMode)
    [errorCode]=vrep.simxStartSimulation(clientID,...
    vrep.simx_opmode_oneshot_wait);

     if errorCode ~=0 %~= = !=
        error('Vrep error of requests a start of a
simulation');
    else
        disp('OK request simulation');
    end
end

%% Retrieves an object handle based on his name
initialiseHandleObject

%% Retrieves goal position
[errorCode, goal_pos] =
vrep.simxGetObjectPosition(clientID,h.goal,-1,...
    vrep.simx_opmode_streaming);%first call: streaming
pause(0.1)
[errorCode, goal_pos] =
vrep.simxGetObjectPosition(clientID, h.goal, -1,...
    vrep.simx_opmode_buffer);%other call: buffer
if errorCode ~= 0
    error('Error with simxGetObjectPosition for goal');
end

%% Retrieve the position of the car
[errorCode, actu_pos] =
vrep.simxGetObjectPosition(clientID,h.Pioneer,-1,...
    vrep.simx_opmode_streaming);%first call: streaming
pause(0.1) %Why we need a Pause?
[errorCode, actu_pos] =
vrep.simxGetObjectPosition(clientID, h.Pioneer,-1,...
    vrep.simx_opmode_buffer);%other call: buffer
if errorCode ~= 0
    error('Error, can not find the actual position of the
car');
end

%% Retrieve the orientation (Euler angles) of the car
[errorCode,actu_orien]=vrep.simxGetObjectOrientation(client
ID,h.Pioneer,...
```

```
    -1,vrep.simx_opmode_streaming);
pause(0.1) %Why we need a Pause?
[errorCode,actu_orien]=vrep.simxGetObjectOrientation(client
ID,h.Pioneer,...
    -1,vrep.simx_opmode_buffer);
if errorCode ~= 0
    error('Error, can not find the actual orientation of
the car');
end

%% Recieves data sensor

f=000;
f2=000;
z=0.05;
g=0;

[errorCode,detect1,detectedP1,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US1,vrep.simx_opmod
e_streaming+f);
pause(z);

[errorCode,detect1,detectedP1,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US1,vrep.simx_opmod
e_buffer);
pause(g);

[errorCode,detect2,detectedP2,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US2,vrep.simx_opmod
e_streaming+f2);
pause(z);

[errorCode,detect2,detectedP2,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US2,vrep.simx_opmod
e_buffer);
pause(g);

[errorCode,detect3,detectedP3,detectedObjectHandle,detected
SurfaceNV]=...
```

```
vrep.simxReadProximitySensor(clientID,h.US3,vrep.simx_opmod
e_streaming+f2);
pause(z);
[errorCode,detect3,detectedP3,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US3,vrep.simx_opmod
e_buffer);
pause(g);

[errorCode,detect4,detectedP4,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US4,vrep.simx_opmod
e_streaming+f2);
pause(z);

[errorCode,detect4,detectedP4,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US4,vrep.simx_opmod
e_buffer);
pause(g);

[errorCode,detect5,detectedP5,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US5,vrep.simx_opmod
e_streaming+f2);
pause(z);

[errorCode,detect5,detectedP5,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US5,vrep.simx_opmod
e_buffer)
pause(g);

[errorCode,detect6,detectedP6,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US6,vrep.simx_opmod
e_streaming+f2);
pause(z);

[errorCode,detect6,detectedP6,detectedObjectHandle,detected
```

```
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US6,vrep.simx_opmod
e_buffer);
pause(g);
[errorCode,detect7,detectedP7,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US7,vrep.simx_opmod
e_streaming+f2);
pause(z);

[errorCode,detect7,detectedP7,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US7,vrep.simx_opmod
e_buffer);
pause(g);

[errorCode,detect8,detectedP8,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US8,vrep.simx_opmod
e_streaming+f2);
pause(z);

[errorCode,detect8,detectedP8,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US8,vrep.simx_opmod
e_buffer);
pause(g);

if errorCode ~= 0
    error('Error, sensor problem');
end
%%
motorSpeed = [0 0];%Initialize motor speed (1) left, (2)
Right

xD=[1.5]; yD=[1.5];
i=1;
j=1;

steps=1;
stop=0;
xi=0;
```

```
xi2=0;

[errorCode, actu_pos] =
vrep.simxGetObjectPosition(clientID,...
 h.Pioneer, -1, vrep.simx_opmode_buffer);

prev_x_position=actu_pos(1);
prev_y_position=actu_pos(2);
prev_vp2goal=25;
prev_dV=0;

global final_route;

%% Control loop
tic
while (vrep.simxGetConnectionId(clientID)~=-
1)&&(stop~=1)&&(steps<1350)&&(toc<100)
     % Set the position of goal
     goal_position=[xD(j),yD(j),0];

    [errorCode]=vrep.simxSetObjectPosition(clientID,
h.goal,-1,...
        goal_position(i,:),vrep.simx_opmode_oneshot_wait);
    if errorCode ~= 0
        error('Error with simxSetObjectPosition for goal');
    end

     %% Sensors
      u=0;


[errorCode,detect1,detectedP1,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US1,vrep.simx_opmod
e_buffer);
       pause(u);


[errorCode,detect2,detectedP2,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US2,vrep.simx_opmod
e_buffer);
       pause(u);
```

```
[errorCode,detect3,detectedP3,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US3,vrep.simx_opmod
e_buffer);
       pause(u);


[errorCode,detect4,detectedP4,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US4,vrep.simx_opmod
e_buffer);
      pause(u);


[errorCode,detect5,detectedP5,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US5,vrep.simx_opmod
e_buffer);
       pause(u);


[errorCode,detect6,detectedP6,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US6,vrep.simx_opmod
e_buffer);
      pause(u);


[errorCode,detect7,detectedP7,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US7,vrep.simx_opmod
e_buffer);
       pause(u);


[errorCode,detect8,detectedP8,detectedObjectHandle,detected
SurfaceNV]=...

vrep.simxReadProximitySensor(clientID,h.US8,vrep.simx_opmod
e_buffer);
       pause(u);
```

```
    %% Actualise actual orientation and actual position of
the car

[errorCode,actu_orien]=vrep.simxGetObjectOrientation(client
ID,...
        h.Pioneer,-1,vrep.simx_opmode_buffer);
    [errorCode, actu_pos] =
vrep.simxGetObjectPosition(clientID,...
        h.Pioneer, -1, vrep.simx_opmode_buffer);
    [errorCode, goal_pos] =
vrep.simxGetObjectPosition(clientID, h.goal,...
        -1, vrep.simx_opmode_buffer);

    Pioneer(1).Data.x_position(steps)=actu_pos(1);
    x_position=actu_pos(1);

    Pioneer(1).Data.y_position(steps)=actu_pos(2);
    y_position=actu_pos(2);

    %Calcul the route
    d(steps)=sqrt((y_position-
prev_y_position)^2+(x_position-prev_x_position)^2);

    prev_x_position=x_position;
    prev_y_position=y_position;

    route=sum(d);
    Pioneer(1).Data.route(steps)=route;

    %Calcul distance from vehicle point to goal
    vp2goal = sqrt((goal_position(i,1)-actu_pos(1))^2+...
        (goal_position(i,2)-actu_pos(2))^2);
    Pioneer(1).Data.distance(steps)=vp2goal;

      if (steps>=300)&&(vp2goal>=14)
          stop=1;
      end

      if (prev_vp2goal==round(vp2goal,4))
             xi=xi+1;
          if (xi==50)
              stop=1;
          end
      end
    prev_vp2goal=round(vp2goal,4);

    %Calcul angle the vehicule position orientation and the
```

```
goal
    curdiff = atan2(goal_position(i,2)-actu_pos(2),...
        goal_position(i,1)-actu_pos(1));

    %Calcul error angle
    angle_error =  actu_orien(3)-curdiff;

    if (angle_error < -pi) %turn left
            angle_error = angle_error + 2*pi;
        elseif (angle_error >= pi)%turn right
            angle_error = angle_error - 2*pi;
    end

        angle_error_deg=rad2deg(angle_error)
        Pioneer(1).Data.angleError(steps)=angle_error_deg;


         if  detectedP1(3)<0 || detectedP1(3)>1

              detectedP1(3)=1;
         end

         if detectedP2(3)<0 || detectedP2(3)>1

              detectedP2(3)=1;
         end

         if  detectedP3(3)<0 || detectedP3(3)>1

              detectedP3(3)=1;
         end

         if detectedP4(3)<0 || detectedP4(3)>1

              detectedP4(3)=1;
         end

         if  detectedP5(3)<0 || detectedP5(3)>1

              detectedP5(3)=1;
         end

         if detectedP6(3)<0 || detectedP6(3)>1

              detectedP6(3)=1;
         end
```

```
          if  detectedP7(3)<0 || detectedP7(3)>1

               detectedP7(3)=1;
          end

          if detectedP8(3)<0 || detectedP8(3)>1

               detectedP8(3)=1;
          end

          Pioneer(1).Data.sensors.s1(steps)=detectedP1(3);
          Pioneer(1).Data.sensors.s2(steps)=detectedP2(3);
          Pioneer(1).Data.sensors.s3(steps)=detectedP3(3);
          Pioneer(1).Data.sensors.s4(steps)=detectedP4(3);
          Pioneer(1).Data.sensors.s5(steps)=detectedP5(3);
          Pioneer(1).Data.sensors.s6(steps)=detectedP6(3);
          Pioneer(1).Data.sensors.s7(steps)=detectedP7(3);
          Pioneer(1).Data.sensors.s8(steps)=detectedP8(3);

          %% controller Inputs

          input1 = min([(detectedP3(3)) (detectedP2(3))
(detectedP1(3))])  %% left
          input2 = min([detectedP4(3) detectedP5(3)])  %%
center
          input3 = min([(detectedP6(3)) (detectedP7(3))
(detectedP8(3))])  %% right
%
          Pioneer(1).Data.sensors.leftInput(steps)=input1;

Pioneer(1).Data.sensors.centralInput(steps)=input2;
          Pioneer(1).Data.sensors.rightInput(steps)=input3;
%

i_sensors=min([min([Pioneer.Data.sensors.centralInput])...
          min([Pioneer.Data.sensors.leftInput])
min([Pioneer.Data.sensors.rightInput])]);

          if (i_sensors<=0.25)
                stop=1;
          end

          %%  Fuzzy controller

          output = evalfis
(double([input1;input2;input3;angle_error_deg]),Fis);
          motorSpeed(1)=output(1)
```

```
          motorSpeed(2)=output(2)
          Pioneer(1).Data.speed.leftwheel(steps)=
motorSpeed(1);
          Pioneer(1).Data.speed.rightwheel(steps)=
motorSpeed(2);

          dV= abs(motorSpeed(1)-motorSpeed(2));

            if (dV>=3)&& (prev_dV==round(dV,3))
                   xi2=xi2+1;
                if (xi2==500)
                    stop=1;
                end
            end
             prev_dV=round(dV,4);


[errorCode]=vrep.simxSetJointTargetVelocity(clientID,h.left
Motor,...

motorSpeed(1),vrep.simx_opmode_streaming);

[errorCode]=vrep.simxSetJointTargetVelocity(clientID,h.righ
tMotor,...

motorSpeed(2),vrep.simx_opmode_streaming);

          steps=steps+1;
          Pioneer(1).Data.steps(steps)=steps;

          final_route=sum(d);
          Pioneer(1).Data.final_route(steps)=final_route;

    if    (vp2goal <= 0.15)
           j=j+1;
       if j>1
          stop=1;
       end
    end
 toc
end

[errorCode]=vrep.simxSetJointTargetVelocity(clientID,h.left
Motor,0,...
    vrep.simx_opmode_streaming);
[errorCode]=vrep.simxSetJointTargetVelocity(clientID,h.righ
tMotor,0,...
```

```
    vrep.simx_opmode_streaming);

v.simstop();
vrep.simxFinish(-1);

simulation_steps= Pioneer(1).Data.steps(steps);
Pioneer(1).Data.final_route=final_route;

disp('program end')

end
```