Πολυτεχνειο Κρητης
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων

ΠΛΗ 513 – Αυτόνομοι Πράκτορες – 2017

Διδάσκων: Μ. Γ. Λαγουδάκης **2η Εργαστηριακή Άσκηση** Παράδοση: **30/10/17,11μμ**

Εισαγωγή

Έχοντας ήδη μια μικρή εμπειρία με τον ρομποτικό προσομοιωτή Webots, είστε πλέον σε θέση να σχεδιάζετε απλούς ελεγκτές (controllers) για διαφορετικά ρομποτικά συστήματα, λαμβάνοντας υπόψιν τα ιδιαίτερα χαρακτηριστικά τους, ως προς τους αισθητήρες και τους επενεργητές. Στόχος της παρούσας εργαστηριακής άσκησης είναι η γνωριμία με το e-puck, ένα τυπικό μοντέλο τροχοφόρου κινητού ρομπότ με διαφορική κίνηση, και η ανάπτυξη ενός βασικού ελεγκτή για την ασφαλή πλοήγησή του εντός ενός απλού λαβυρίνθου.

Εγκατάσταση

Για να δουλέψετε με τον προσομοιωτή Webots, θα χρειαστεί να κατεβάσετε:

- το κατάλληλο αρχείο εγκατάστασης για την έκδοση **7.4.3**[1] από www.cyberbotics.com/archive
- τα εγχειρίδια χρήσης για την έκδοση **7.4.3** από την ενότητα εργαστηριακού υλικού στο courses

Λογικά, τα παραπάνω τα έχετε ήδη κάνει από την προηγούμενη εργαστηριακή άσκηση, οπότε είστε έτοιμοι.

Το Ρομπότ e-puck

Το ρομπότ e-puck (www.e-puck.org) είναι ένα μικρό τροχοφόρο ρομπότ, με διάμετρο μόλις 7 εκατοστά, ύψος 5 εκατοστά και βάρος κάτω από 200 γραμμάρια, που κινείται με διαφορική κίνηση σε δύο τροχούς. Σχεδιάστηκε αρχικά από τους Michael Bonani και Francesco Mondada σε συνεργασία με τρία ερευνητικά εργαστήριά του Ecole Polytechnique Federale de Lausanne (EPFL) στην Ελβετία. Τόσο το υλικό, όσο και το λογισμικό, του e-puck είναι open source, ωστόσο κατασκευάζεται και κυκλοφορεί εμπορικά. Οι αισθητήρες που διαθέτει είναι 8 αισθητήρες υπερύθρων στην περιφέρειά του για μέτρηση απόστασης σε κοντικά εμπόδια, ένα επιταχυνσιόμετρο τριών διαστάσεων, τρία μικρόφωνα, και μία έγχρωμη κάμερα χαμηλής ανάλυσης. Οι επενεργητές που διαθέτει είναι δύο βηματικοί κινητήρες 1000 θέσεων, ένας για κάθε τροχό, ένα ηχείο, και μια σειρά από διόδους LED στην περιφέρεια του σώματος, στο σώμα και στην κάμερα. Διατίθενται επίσης και διάφορα πρόσθετα στοιχεία: πυργίσκος με 1D ή 2D omni-directional camera για μελέτη οπτικών ροών, αισθητήρες χρώματος εδάφους για παρακολούθηση γραμμών, πυργίσκος με χρωματιστές διόδους LED για οπτική επικοινωνία, και μαγνητικοί τροχοί για κάθετη αναρρίχηση.

Εγχειρίδια

Το βασικό εγχειρίδιο που είναι διαθέσιμο για το e-puck είναι ένα άρθρο που δημοσιεύθηκε το 2009 στο διεθνές συνέδριο Robotica 2009. Θα το βρείτε και στο εργαστηριακό υλικό στο courses.

- Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J.-C., Floreano, D. and Martinoli, A. *The e-puck, a Robot Designed for Education in Engineering.* Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions, 2009.

Σημαντικές τεχνικές πληροφορίες ωστόσο περιλαμβάνονται στην Ενότητα 8.1 του Webots User Guide. Ρίξτε μια ματιά για να δείτε την χωροθέτηση των αισθητήρων απόστασης στο σώμα του ρομπότ, την σχέση απόστασης-τιμών, και τα interfaces που παρέχονται στον κώδικα για κίνηση και μετρήσεις αποστάσεων.

Rat's Life

Το Rat's Life (ratslife.org) είναι ένας ρομποτικός διαγωνισμός που διεξήχθη επίσημα κατά τα έτη 2009–2011 στο περιβάλλον προσομοίωσης Webots. Ουσιαστικά, πρόκειται για ένα παιχνίδι επιβίωσης, όπου δύο ρομπότ e-puck (αρουραίοι) ανταγωνίζονται μεταξύ τους για τους πόρους που βρίσκονται διάσπαρτοι μέσα σε έναν άγνωστο (τυχαίο) λαβύρινθο. Όπως και οι αρουραίοι στη φύση, τα δύο ρομπότ πρέπει να ψάξουν για τροφές που θα τους επιτρέψουν να ζήσουν περισσότερο από τον αντίπαλό τους. Στην περίπτωσή μας, κάθε ρομπότ μπορεί να «τραφεί» σε κάποιον από τους τέσσερις σταθμούς ενέργειας που είναι τοποθετημένοι σε τυχαίες θέσεις μέσα στον λαβύρινθο, όπου μπορεί να φορτιστεί η μπαταρία του. Μόλις εντοπίσει και πλησιάσει έναν σταθμό ενέργειας, το ρομπότ αντλεί την διαθέσιμη ενέργεια, αναζωογονείται, και ο σταθμός δεν είναι πλέον διαθέσιμος για φόρτιση για κάποιο χρονικό διάστημα έως ότου συσσωρεύσει νέα ενέργεια και είναι έτοιμος να την παραδώσει. Ως εκ τούτου, το ρομπότ θα πρέπει να εξερευνήσει τον λαβύρινθο, ψάχνοντας και για άλλους σταθμούς, ενώ ταυτόχρονα θα πρέπει να θυμάται που είχε εντοπίσει πρωτύτερα σταθμούς ενέργειας, ώστε ακολουθώντας κατάλληλες διαδρομές να ανεφοδιάζεται συνεχώς. Ταυτόχρονα, το ίδιο κάνει και ο αντίπαλος, οπότε ακόμη κι αν φτάσει ο αρουραίος μας επιτυχώς σε κάποιον σταθμό, δεν αποκλείεται να τον βρει (προσωρινά) άδειο. Ο χρόνος τρέχει ανελέητα, η ενέργεια μειώνεται ακάθεκτα και η αναζήτηση για τροφή συνεχίζεται. Το ερώτημα είναι: ποιος από τους δύο αρουραίους θα μπορέσει να ζήσει περισσότερο; Ένας έξυπνος διαγωνισμός, όπου υπεισέρχονται τα

[1] Η τελευταία έκδοση (8.3.0) δυστυχώς δεν παρέχει πλέον δωρεάν λειτουργικότητα.

βασικά προβλήματα ενός ρομποτικού πράκτορα: επεξεργασία δεδομένων αισθητήρων, έλεγχος κίνησης, σχεδιασμός διαδρομής, εντοπισμός θέσης, χαρτογράφηση, στρατηγική λήψης αποφάσεων. Το Rat's Life έχει υλοποιηθεί και έχει τρέξει επίσης και σε πραγματικό περιβάλλον με λαβύρινθο από Lego.

Διαδικασία

Ξεκινήστε τον προσομοιωτή Webots (επιλέξτε Continue with the free version of Webots)[2]. Αντιγράψτε τον φάκελο `webots/projects/contests/ratslife` σε κάποιο δικό σας χώρο χωρίς Ελληνικούς χαρακτήρες στο path (όλες οι αναφορές στο εξής θα είναι στο δικό σας αντίγραφο). Από το File επιλέξτε να ανοίξετε τον κόσμο `ratslife.wbt` που βρίσκεται στο φάκελο `ratslife/worlds`. Μόλις ξεκινήσετε την προσομοίωση, στο γραφικό περιβάλλον θα δημιουργηθεί ο τυχαίος λαβύρινθος και θα δείτε τα δύο ρομπότ e-puck να ξεκινούν από τυχαίες θέσεις. Τα δύο ρομπότ ελέγχονται από τον κώδικα Java που βρίσκεται στους φακέλους `ratslife/controllers/Rat0` και `ratslife/controllers/Rat1` αντίστοιχα.

Πειραματισμός

Παρακολουθήστε για λίγο την προσομοίωση. Ανοίξτε και το `Rat0.java` (το `Rat1.java` είναι ίδιο, μόνο το όνομα διαφέρει) και προσπαθήστε να καταλάβετε τη λογική που περιέχεται στον κώδικα. Αγνοήστε τα τμήματα που έχουν να κάνουν με την κάμερα και τα LEDs (ακόμη καλύτερα, σχολιάστε τις γραμμές 90–128) και εστιάστε μόνο σ΄ αυτά που έχουν να κάνουν με τις μετρήσεις απόστασης και την κίνηση των τροχών. Ο controller που δίνεται ουσιαστικά υλοποιεί ένα Braitenberg Vehicle[3], όπου οι τιμές των αισθητήρων τροφοδοτούνται απευθείας στον ένα ή και στους δύο επενεργητές μετά από κάποιο ζύγισμα με κατάλληλα βάρη για να προκύψει μια επιθυμητή συμπεριφορά πλοήγησης. Στον κώδικα υπάρχουν δύο διανύσματα βαρών· εντοπίστε τα και προσπαθήστε να καταλάβετε τον ρόλο τους. Σε κάθε κύκλο ελέγχου, το ρομπότ διαβάζει τις τιμές των αισθητήρων απόστασης και υπολογίζει κάποιες τιμές για τις ταχύτητες των τροχών. Παρατηρήστε πώς αυτές οι τιμές μεταβάλλονται, όταν το ρομπότ αποφασίσει να μπει σε διαδικασία επιτόπιας στροφής για την αποφυγή κάποιων εμποδίων. Παίξτε με τις τιμές των βαρών και δοκιμάστε τα δικά σας διανύσματα τιμών. Επίσης, αλλάξτε τον κώδικα, ώστε το ρομπότ να υλοποιεί μια συμπεριφορά πλοήγησης τύπου left (ή right) wall following, όπου το ρομπότ κινείται στον λαβύρινθο ακολουθώντας πάντα τον τοίχο στα αριστερά (δεξιά) του. Με άλλα λόγια, κινείται συνεχώς κατά μήκος του τοίχου, έχοντας πάντα το αριστερό (ή δεξί) χέρι ακουμπισμένο στον τοίχο στα αριστερά (δεξιά), δηλαδή περίπου ότι θα κάνατε κι εσείς αν σας έβαζε κανείς σε έναν θεοσκότεινο λαβύρινθο. Τι το ιδιαίτερο έχει μια τέτοια συμπεριφορά, όταν χρησιμοποιείται σε έναν λαβύρινθο; Ψάξτε στο διαδίκτυο ή αλλού να βρείτε την απάντηση. Σε κάθε αλλαγή, πατήστε Clean και Build για να δημιουργηθεί ξανά το εκτελέσιμο και ξεκινήστε πάλι την προσομοίωση με το Revert. Διαγράψτε τον άλλο παίκτη από το scene tree στα αριστερά για να μειώσετε τον υπολογιστικό φόρτο και μερικούς τοίχους αν θέλετε να έχετε μεγαλύτερο χώρο για τον παίκτη σας. Μην αποθηκεύσετε όμως τον τροποποιημένο κόσμο!

Ασκήσεις

Τώρα είναι η σειρά σας να προγραμματίσετε μια διαφορετική και ενδιαφέρουσα συμπεριφορά πλοήγησης στον αρουραίο e-puck του Rat's Life. Αν υποθέσουμε ότι ο δικός σας αρουραίος είναι λίγο ανάποδος και θέλει να κινείται με την όπισθεν, τροποποιήστε τον κώδικα του `Rat0.java` ώστε να υλοποιήσετε τη συμπεριφορά **left** wall following αλλά με κίνηση **όπισθεν**! Προσέξτε, διότι η διάταξη των αισθητήρων απόστασης στο πίσω μέρος του ρομπότ είναι διαφορετική από αυτή στο εμπρός μέρος, εκτός του ότι είναι και λιγότεροι στο πίσω μέρος! Συνεπώς, θα πρέπει να δημιουργήσετε τα δικά σας διανύσματα βαρών και κατά πάσα πιθανότητα να τροποιήσετε τον τρόπο με τον οποίο στρίβει. Με διπλό κλικ πάνω στο ρομπότ μπορείτε να δείτε σε ξεχωριστό παράθυρο τις τρέχουσες τιμές των αισθητήρων ανά πάσα τιμή. Κρατώντας πατημένο το Shift, με το ποντίκι μπορείτε να μετακινήσετε το ρομπότ και να το τοποθετήσετε όπου θέλετε στον λαβύρινθο. Θα προσέξατε ίσως ότι με την αρχική συμπεριφορά κάποιες φορές το ρομπότ «κολλάει» σε κάποιες θέσεις. Φυσικά, εσείς θα φροντίσετε αυτό να μην συμβεί, αλλά αν παρ΄ ελπίδα συμβεί, πώς θα μπορούσατε να το αντιμετωπίσετε; Σκεφτείτε λίγο μήπως κάποια τυχαιότητα είναι χρήσιμη ενίοτε... Ο κώδικάς σας θα πρέπει να λειτουργεί σε οποιονδήποτε τυχαίο λαβύρινθο, όπως δημιουργούνται από τον controller στην αρχή. Παρακολουθείτε συνεχώς στην προσομοίωση αν επιτυγχάνεται ο στόχος σας και διορθώνετε ανάλογα. Όταν τα καταφέρετε, τροποποιήστε και τον κώδικα του `Rat1.java`, ώστε ο άλλος αρουραίος να υλοποιεί **right** wall following πάλι με κίνηση **όπισθεν** και βάλτε τους να τρέξουν μαζί...

Αναφορά/Παράδοση/Βαθμολογία

Συμπιέστε τον φάκελο εργασίας `ratslife` που περιέχει τον κώδικά σας (`Rat0.java` και `Rat1.java`). Καταγράψτε ένα βίντεο της τελικής συμπεριφοράς των αρουραίων σας μέσα από το Webots. Γράψτε μια σύντομη αναφορά (το πολύ μία σελίδα, σε PDF), όπου θα περιγράφετε τη δουλειά σας και θα απαντάτε το ερώτημα που τέθηκε παραπάνω σχετικά με το left (ή right) wall following. Τέλος, παραδώστε κώδικα, αναφορά και βίντεο μέσω του courses. Η βαθμολογία θα προκύψει από την ποιότητα της εργασίας σας.

[2] Επιλέξτε Pause για Startup Mode μέσα από το Tools-Preferences για να μην ξεκινάει αυτόματα η προσομοίωση.

[3] Δείτε το σχετικό κείμενο (Braitenberg Vehicle) στην Wikipedia ή στο εργαστηριακό υλικό στο courses.