ΑΥΤΟΝΟΜΑ ΡΟΜΠΟΤΙΚΑ ΟΧΗΜΑΤΑ

ΘΕΜΑ ΕΡΓΑΣΙΑΣ:

ΥΠΟΛΟΓΙΣΜΟΣ ΘΕΣΗΣ & ΠΡΟΣΑΝΑΤΟΛΙΣΜΟΥ ΤΡΟΧΟΦΟΡΟΥ ΡΟΜΠΟΤΙΚΟΥ ΟΧΗΜΑΤΟΣ ΣΤΟ ΠΕΡΙΒΑΛΛΟΝ ΠΡΟΣΟΜΟΙΩΣΗΣ COPPELIASIM (VREP) ΜΕ ΧΡΗΣΗ PYTHON

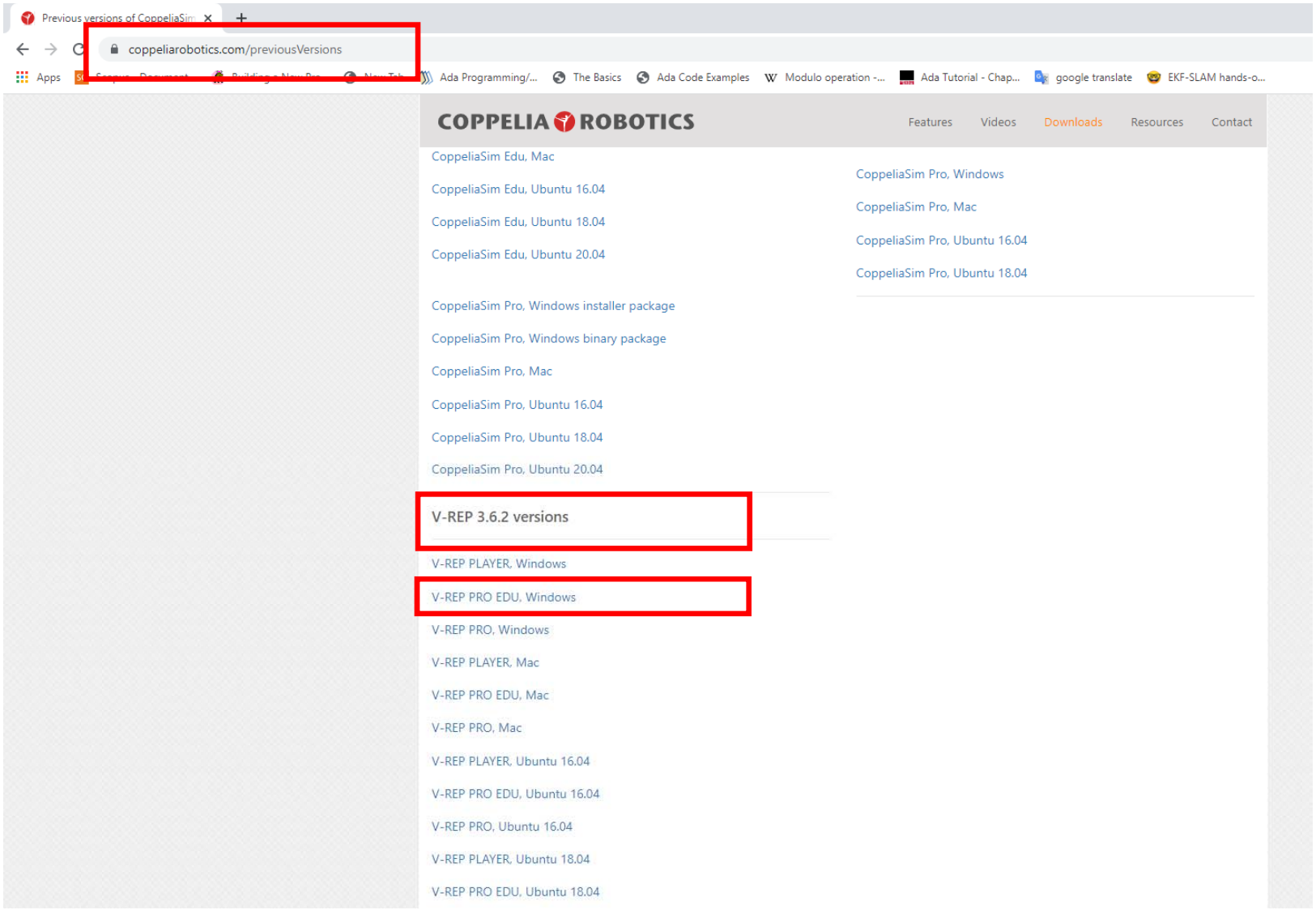
Απώτερος σκοπός της εργασίας

- Βαθύτερη κατανόηση των μεθοδολογιών που παρουσιάζονται στο μάθημα
- Υλοποίηση διδασκόμενων αλγορίθμων σε γλώσσα προγραμματισμού (Python)
- Επίλυση ενός συνολικού κλασικού προβλήματος ρομποτικής
- Ενσωμάτωση αλγορίθμων σε ρεαλιστικό ρομποτικό σύστημα και μελέτη της συμπεριφοράς αυτού

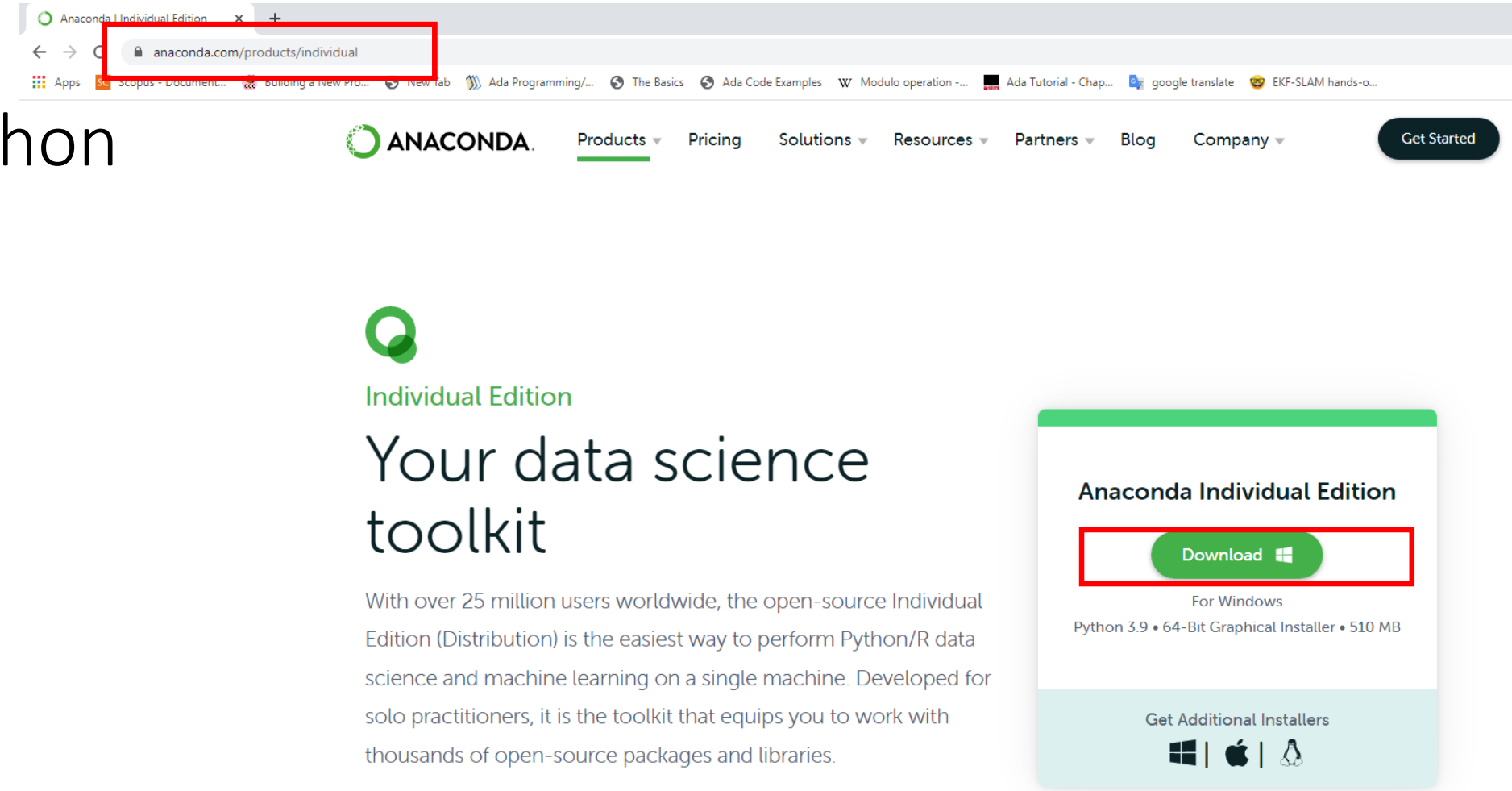
Περιγραφή Θέματος

- Δίνεται τροχοφόρο ρομποτικό όχημα (unicycle-like) το οποίο κινείται στο επίπεδο.
- Στο χώρο υπάρχουν ορόσημα (landmarks) των οποίων η θέση είναι γνωστή στο ρομπότ (χάρτης).
- Το όχημα διαθέτει κατάλληλο αισθητήρα που αναγνωρίζει τα ορόσημα και παρέχει την απόσταση (range) και τον προσανατολισμό (bearing) του ρομπότ ως προς κάθε ορόσημο.
- Το ορόσημο γίνεται αντιληπτό μόνο εάν βρίσκεται εντός του εύρους μέτρησης του αισθητήρα.
- Ο αισθητήρας μπορεί ταυτόχρονα να αναγνωρίζει παραπάνω από ένα ορόσημα.

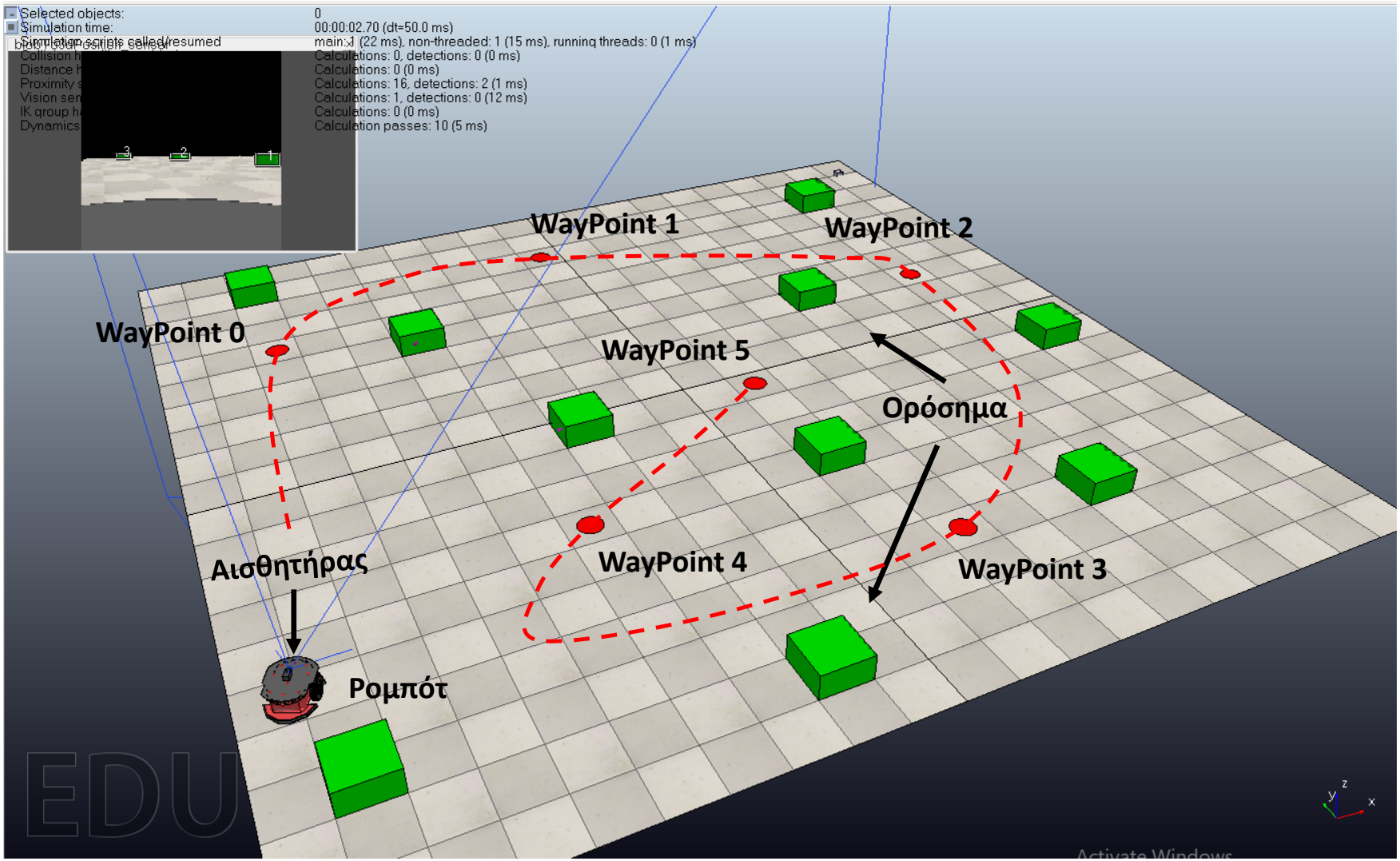
VREP


Previous versions of CoppeliaSim
coppeliarobotics.com/previousVersions
Apps
Ada Programming/...
The Basics
Ada Code Examples
Modulo operation -...
Ada Tutorial - Chap...
google translate
EKF-SLAM hands-o...
COPPELIA ROBOTICS
Features
Videos
Downloads
Resources
Contact
CoppeliaSim Edu, Mac
CoppeliaSim Edu, Ubuntu 16.04
CoppeliaSim Edu, Ubuntu 18.04
CoppeliaSim Edu, Ubuntu 20.04
CoppeliaSim Pro, Windows installer package
CoppeliaSim Pro, Windows binary package
CoppeliaSim Pro, Mac
CoppeliaSim Pro, Ubuntu 16.04
CoppeliaSim Pro, Ubuntu 18.04
CoppeliaSim Pro, Ubuntu 20.04
CoppeliaSim Pro, Windows
CoppeliaSim Pro, Mac
CoppeliaSim Pro, Ubuntu 16.04
CoppeliaSim Pro, Ubuntu 18.04
V-REP 3.6.2 versions
V-REP PLAYER, Windows
V-REP PRO EDU, Windows
V-REP PRO, Windows
V-REP PLAYER, Mac
V-REP PRO EDU, Mac
V-REP PRO, Mac
V-REP PLAYER, Ubuntu 16.04
V-REP PRO EDU, Ubuntu 16.04
V-REP PRO, Ubuntu 16.04
V-REP PLAYER, Ubuntu 18.04
V-REP PRO EDU, Ubuntu 18.04

Python


Anaconda | Individual Edition
anaconda.com/products/individual
Apps
Ada Programming/...
The Basics
Ada Code Examples
Modulo operation -...
Ada Tutorial - Chap...
google translate
EKF-SLAM hands-o...
ANACONDA.
Products
Pricing
Solutions
Resources
Partners
Blog
Company
Get Started
Individual Edition
Your data science toolkit
With over 25 million users worldwide, the open-source Individual Edition (Distribution) is the easiest way to perform Python/R data science and machine learning on a single machine. Developed for solo practitioners, it is the toolkit that equips you to work with thousands of open-source packages and libraries.
Anaconda Individual Edition
Download
For Windows
Python 3.9 • 64-Bit Graphical Installer • 510 MB
Get Additional Installers

Ο χώρος εργασίας στο VREP (Project_Scene.ttt)



Ζητούμενα

1. Να υλοποιηθεί κατάλληλη συνάρτηση που να μετατρέπει τις επιθυμητές ταχύτητες στο τοπικό σύστημα του ρομπότ, u (μεταφορική) και ω (περιστροφική), σε γωνιακές ταχύτητες των τροχών (αριστερός, δεξιός) του ρομπότ.

```
def servo_controller(u, w):
    ...
    ...
    ...
    return wl, wr
```

Δίνονται τα γεωμετρικά χαρακτηριστικά του ρομπότ:

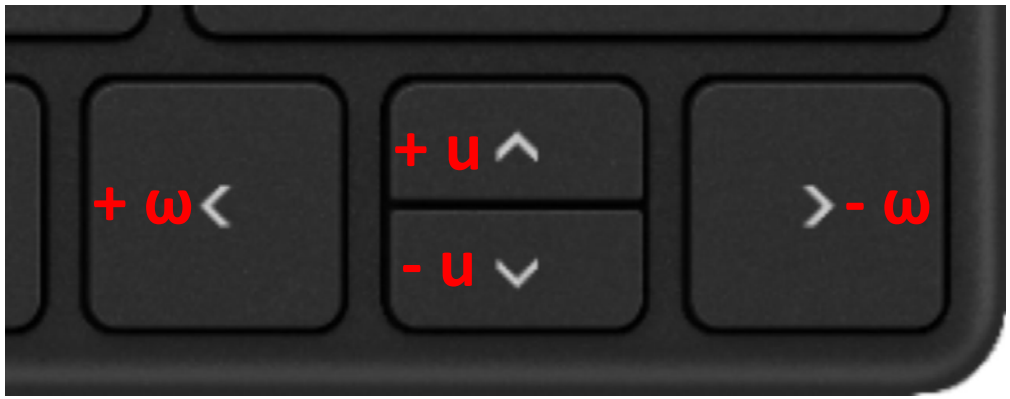
- Ακτίνα ροδών R = 0.111 m
- Μήκος άξονα ροδών L = 0.381 m

Ζητούμενα

2. Να υλοποιηθεί κατάλληλη συνάρτηση που να επιτρέπει τον τηλεχειρισμό του ρομπότ από το πληκτρολόγιο π.χ:

```
def keyboard_teleop():
    ...
    ...
    ...
    return u, w
```

Όσο τα πλήκτρα είναι πατημένα το ρομπότ θα αποκτά **σταθερές ταχύτητες** (u, ω), τις οποίες επιλέγετε κατά βούληση.



Σημείωση: Να χρησιμοποιήσετε το πακέτο **keyboard** της Python για την ανίχνευση συμβάντων (events) του πληκτρολογίου.

Ζητούμενα

3. Να υλοποιηθεί κατάλληλη συνάρτηση που να υπολογίζει τις μεταβλητές οδομετρίας $\delta_{rot1}, \delta_{trans}, \delta_{rot2}$ από την τρέχουσα και προηγούμενη θέση του ρομπότ καθώς και από τους θορύβους $\alpha_1, \alpha_2, \alpha_3, \alpha_4$.

```
def get_odometry_from_pose(pose, pose_prev, a):
    ...
    ...
    ...
    return np.array([delta_rot_1, delta_trans, delta_rot_2])
```

a = [α_1, α_2, α_3, α_4]

Όπου είναι δυνατόν προτιμήστε χρήση πινάκων μέσω του πακέτου **numpy** αντί για lists. Θα βοηθήσει ιδιαίτερα στους μαθηματικούς υπολογισμούς.

import **numpy** as **np**

Ζητούμενα

4. Να υλοποιηθεί κατάλληλη συνάρτηση που να υλοποιεί το μοντέλο κίνησης του ρομπότ με βάση την οδομετρία

```
def motion_model(odometry, robot_pose):
    ...
    ...
    ...
    return np.array([x_mm, y_mm, theta_mm])
```

Εκτίμηση τη χρονική στιγμή k

odometry: οδομετρία [δrot1, δtrans, δ, δrot2]

robot_pose: θεση και προσανατολισμός του ρομποτ [x, y, θ] τη χρονική στιγμή k-1

Ζητούμενα

5. Υλοποίηση EKF Localization αλγορίθμου για την εκτίμηση της θέσης και προσανατολισμού του ρομπότ με βάση το μοντέλο κίνησης και τις μετρήσεις από τα ορόσημα. Το ρομπότ να κινείται στο χώρο με το σχήμα τηλεχειρισμού που υλοποιήθηκε στο βήμα 2 και να ακολουθεί την τροχιά που περνά από τα WayPoints.

```
def ekf_algorithm(ekf_state, Sigma, robot_odometry,a, id_landmarks, map_world):

    ...
    ...
    ...
    return ekf_state, Sigma
```

Σημείωση: Προσπαθήστε να έχετε ορόσημα όσο το δυνατό συχνότερα στο οπτικό πεδίο του αισθητήρα.

Δεδομένα

1. Ο χώρος εργασίας στο VREP (Project_Scene.ttt)
   - Ρομπότ
   - Αισθητήρας (κάμερα που όμως λειτουργεί ως range/bearing sensor)
   - Ορόσημα
   - WayPoints
2. Έτοιμη διασύνδεση της επικοινωνίας VREP και Python μέσω RemoteAPI
   - Τα αρχεία για τη διασύνδεση βρίσκονται στο φάκελο της εργασίας (δεν κάνετε κάτι με αυτά)
   - Η επικοινωνία είναι υλοποιημένη στο αρχείο main_code_template.py
3. Βοηθητικές συναρτήσεις
   - Υλοποιημένες συναρτήσεις για απευθείας χρήση
   - Βρίσκονται στο αρχείο help_pkg.py
   - import help_pkg
4. Template Python κώδικας όπου θα ενσωματώσετε τον δικό σας κώδικα
   - Αρχείο main_code_template.py
     - Βασική αρχιτεκτονική προγράμματος
     - Υποδείξεις για την κλήση των συναρτήσεων που θα υλοποιηθούν για τα ζητούμενα 1 – 5
     - Τα σχόλια εντός του κώδικα είναι ιδιαίτερα επεξηγηματικά
   - Αρχείο amr_loc_template.py
     - Δίνεται template των συναρτήσεων που πρέπει να υλοποιήσετε
5. Βοηθητικό βίντεο εκκίνησης/λειτουργίας

Βοηθητικές συναρτήσεις

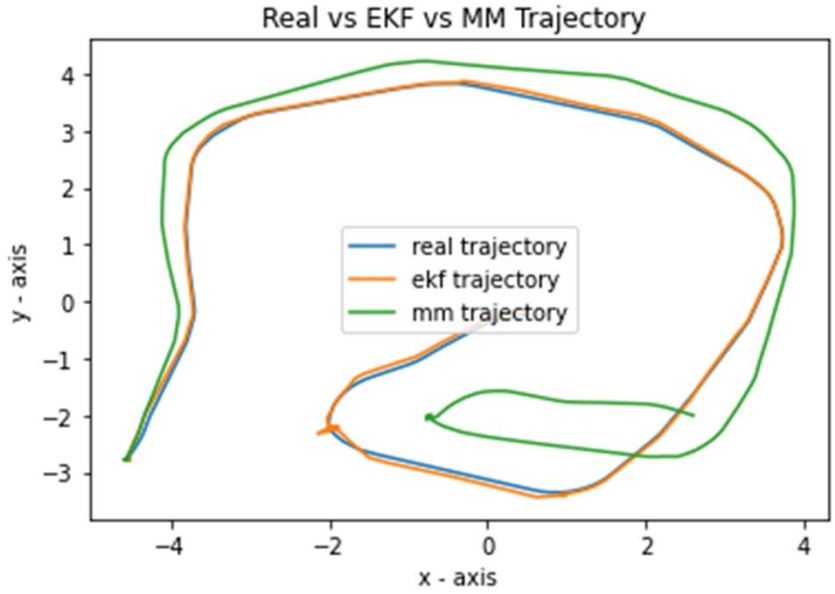
- **vrep_connection()**:
  - Περιγραφή: Δημιουργεί την επικοινωνία VREP – Python
  - Είσοδος: κενή
  - Έξοδος: handler επικοινωνίας
- **sample_normal_twelve(mu, sigma)**:
  - Περιγραφή: Δείγμα από κανονική κατανομή
  - Είσοδος: μέση τιμή (mu), διασπορά (sigma)
  - Έξοδος: δείγμα
- **get_robot_pose(clientID, robot_handle)**:
  - Περιγραφή: Επιστρέφει την τρέχουσα θέση και προσανατολισμό του ρομπότ σε πίνακα [x, y, theta]
  - Είσοδος: handler επικοινωνίας (clientID), handler ρομπότ (robot_handler)
  - Έξοδος: θέση και προσανατολισμός του ρομπότ σε πίνακα [x, y, theta]
- **map_create(clientID)**:
  - Περιγραφή: Δημιουργεί τον χάρτη που περιέχει τις θέσεις των οροσήμων
  - Είσοδος: handler επικοινωνίας (clientID)
  - Έξοδος: Πίνακας που περιέχει τις θέσεις των οροσήμων

Βοηθητικές συναρτήσεις

- **read_sensor_data (initialCall, clientID, map_world, robot_pose)**:
  - Περιγραφή: Διαβάζει τα δεδομένα από τον αισθητήρα
  - Είσοδος: Boolean μεταβλητή αν ο αισθητήρας διαβάζει για πρώτη φορά (initialCall), handler επικοινωνίας (clientID), χάρτης του χώρου (map_world), θέση και προσανατολισμός του ρομποτ [x, y, theta] (robot_pose)
  - Έξοδος: Boolean μεταβλητή (initialCall), πίνακας (id_landmarks) που περιέχει τα ορόσημα που βλέπει ο αισθητήρας με την ετικέτα τους [id1, range1, bearing1
    id2, range2, bearing2
    … , … , …]
- **get_associated_landmarks**:
  - Περιγραφή: Ταυτοποιεί τα ορόσημα. Την καλεί η read_sensor_data. Δεν είναι απαραίτητες περισσότερες λεπτομέρειες. Δεν θα την καλέσετε απευθείας.
- **set_motor_cmds (clientID, l_motor_handle, r_motor_handle, wl, wr)**:
  - Περιγραφή: Στέλνει εντολές ταχύτητας περιστροφής στους κινητήρες του ρομπότ
  - Είσοδος: handler επικοινωνίας (clientID), handlers αριστερου (l_motor_handle) και δεξιού (r_motor_handle) κινητήρα, επιθυμητές γωνιακές ταχύτητες περιστροφής αριστερού (wl) και δεξιού (wr) κινητήρα.
  - Έξοδος: Κενό

Παράδοση

- Οι ζητούμενες συναρτήσεις υλοποιημένες σε Python στο αρχείο amr_loc_template.py το οποίο να γίνεται import στο main_code_template.py.
- Συμπληρωμένο κατάλληλα το αρχείο main_code_template.py και πλήρως λειτουργικό.
- Συνοπτική έκθεση με την επεξήγηση των συναρτήσεων που υλοποιήθηκαν και το θεωρητικό υπόβαθρο (εξισώσεις/αλγόριθμοι) που υιοθετήθηκαν.
- Για την υποδεικνυόμενη τροχιά (μέσω τηλεχειρισμού) να δημιουργηθούν γραφήματα με την πραγματική θέση και προσανατολισμό του ρομπότ, τη θέση και προσανατολισμό που θα είχαμε αν χρησιμοποιούσαμε μόνο το μοντέλο κίνησης, τη θέση και προσανατολισμό που εκτιμά ο αλγόριθμος EKF.



Βοηθητικό Βίντεο Εκκίνησης/Λειτουργίας