Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Σημάτων, Ελέγχου και Ρομποτικής
Εργαστήριο Ρομποτικής - Εναέριων Συστημάτων

Έλεγχος οπτικής ανατροφοδότησης πολυκόπτερου με χρήση ενισχυτικής μάθησης για παρακολούθηση κινούμενου στόχου

Διπλωματική Εργασία

του

Ανδρέα Μητακίδη

Επιβλέπων: Κωνσταντίνος Σ. Τζαφέστας
Αν. Καθηγητής Ε.Μ.Π.

Αθήνα, 22 Οκτωβρίου 2022

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Σημάτων, Ελέγχου και Ρομποτικής
Εργαστήριο Ρομποτικής - Εναέριων Συστημάτων

Έλεγχος οπτικής ανατροφοδότησης πολυκόπτερου με χρήση ενισχυτικής μάθησης για παρακολούθηση κινούμενου στόχου

Διπλωματική Εργασία

του

Ανδρέα Μητακίδη

Επιβλέπων: Κωνσταντίνος Σ. Τζαφέστας
Αν. Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 18η Οκτωβρίου 2022:

.....	.....	.....
Κώνσταντίνος Σ. Τζαφέστας	Στέφανος Κόλλιας	Κωνσταντίνος Ι. Κυριακόπουλος
Αν. Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, 22 Οκτωβρίου 2022

.....

Ανδρέας Μητακίδης
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών, Ε.Μ.Π.

Περίληψη

Το αντικείμενο αυτής της διπλωματικής εργασίας είναι η ακολούθηση ενός επίγειου ρομποτ, που εκτελεί ρόλο οδηγού, από ένα εναέριο ρομπότ δηλαδή το πολυκόπτερο. Ο έλεγχος του πολυκόπτερου γίνεται αυτόνομα λαμβάνοντας την εικόνα του οδηγού από μία κάμερα με την οποία είναι εφοδιασμένο. Η χρήση της ενισχυτικής μάθησης επιτρέπει στο πολυκόπτερο να μάθει τη καλύτερη στρατηγική ακολούθησης του οδηγού και τελικά να μπορεί να εκτελέσει πιο απότομους ελιγμούς (aggressive maneuvering) σε σχέση με μία κλασική στρατηγική ελέγχου με οπτική ανατροφοδότηση. Για την εκπαίδευση ενός τέτοιου δικτύου είναι απαραίτητη η εκτέλεση πολλών επεισοδίων, γεγονός που καθιστά απαραίτητη τη χρήση προσομοίωσης όλων των συστημάτων. Αποδεικνύεται ότι το δίκτυο που είναι εκπαιδευμένο στη προσομοίωση είναι άμεσα εφαρμόσιμο σε πραγματικές συνθήκες επιτρέποντας έτσι την απευθείας πειραματική δοκιμή.

Για την εφαρμογή αυτή αξιοποιούνται πολλές διαφορετικές τεχνολογίες και μεθοδολογίες ώστε να πραγματοποιηθεί με επιτυχία. Απαραίτητη είναι η χρήση βαθιών συνελικτικών δικτύων για την αναγνώριση και προσδιορισμό της θέσης του οδηγού στην εικόνα που με τη σειρά της απαιτεί σημαντικούς υπολογιστικούς πόρους. Για την ικανοποίηση αυτών των απαιτήσεων το πολυκόπτερο φέρει έναν υπολογιστή τεχνητής νοημοσύνης (onboard AI computer) ειδικά σχεδιασμένο για τη γρήγορη και αποδοτική εφαρμογή της μηχανικής μάθησης.

Τέλος, πολύ σημαντική για την εφαρμογή αυτή και περαιτέρω για τη διπλωματική ειναι η πειραματική επιβεβαίωση των μεθοδολογιών που χρησιμοποιούνται και η ανάλυση των αποτελεσμάτων για την εξαγωγή χρήσιμων συμπερασμάτων.

Λέξεις Κλειδιά: Μηχανική Μάθηση, Ενισχυτική Μάθηση, Πολυκόπτερα, Βαθιά συνελικτικά δίκτυα, Ρομποτική, Έλεγχος, Οπτική ανατροφοδότηση, Σχήμα οδηγού - ακολουθητή, Προσομοίωση.

Abstract

The purpose of this diploma dissertation is the design of a leader-follower scheme where the leader is a ground vehicle and the follower an Unmanned Aerial System or simply multirotor. The multirotor's control is autonomous by utilizing the image received from the camera that the multirotor is equipped with. Reinforcement learning is deployed for the control of the multirotor by which the agent learns the best policy resulting in a more aggressive maneuvering compared to a classical visual servoing approach. The training of the agent requires a number of episodes to learn from and this makes simulation an essential part of the process. It is proven that the neural network that is trained in the simulation is directly transferable to real life applications and this allows us to test the trained system immediately.

For this application many different technologies and systems cooperate together to bring the final result. Indispensable is the use of Convolutional neural networks for the detection of the leader in the image plane. These neural networks demand some computational resources that are provided by the onboard Artificial Intelligence computer.

Finally, very important for this diploma's application is the experimental process where we test and confirm the validity of our methodology and analyze the results to extract useful information.

Keywords: Machine Learning, Reinforcement Learning, Multirotors, Convolutional Neural Networks, Robotics, Control, Visual Servoing, Leader-Follower scheme, Simulation.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή αυτής της διπλωματικής εργασίας, κ. Καθηγητή Κυριακόπουλο Κωνσταντίνο για την ευκαιρία που μου έδωσε να είμαι μέλος του εργαστηρίου Εναέριων Οχημάτων, για την πολύτιμη καθοδήγησή του και τις συμβουλές του. Μέσα από αυτή την ευκαιρία ήρθα σε επαφή με το πειραματικό κομμάτι, που δεν θα ήταν δυνατό χωρίς τον εξοπλισμό του εργαστηρίου και εξοικειώθηκα με πολλές έννοιες της ρομποτικής και της τεχνητής νοημοσύνης, και για αυτό είμαι ευγνώμων.

Επίσης θα ήθελα να ευχαριστήσω τον κ. Καθηγητή Γ. Καρρά για το συνεχές ενδιαφέρον και καθοδηγηση που παρείχε καθόλη τη διάρκεια αυτής της διπλωματικής εργασίας.

Τέλος θα ήθελα να ευχαριστήσω τους υποψήφιους διδάκτορες Σωτήρη Ασπράγκαθο και Φώτη Πανέτσο για την πολύτιμη συνεισφορά τους, καθώς αφιέρωσαν προσωπικό χρόνο για να με βοηθήσουν με την υλοποίηση της διπλωματικής και την διεξαγωγή των πειραμάτων .

Μητακιδης Ανδρέας
22 Οκτωβρίου 2022

Περιεχόμενα

Ευρετήριο Εικόνων

Κατάλογος Πινάκων

Κεφάλαιο 1

Εισαγωγή

1.1 Διατύπωση του προβλήματος

Η παρούσα διπλωματική εργασία καταπιάνεται με μία εφαρμογή κινηματικού ελέγχου με χρήση ενισχυτικής μάθησης (Reinforcement learning). Πιο συγκεκριμένα, η εφαρμογή αφορά την ακολούθηση ενός επίγειου ρομπότ-οδηγού από ένα Μη Επανδρωμένο Αεροσκάφος (UAV) εξοπλισμένο με κάμερα. Ο κινηματικός έλεγχος εφαρμόζεται στον εσωτερικό βρόχο (inner loop, attitude controller) που σημαίνει ότι δίνουμε ως εντολές τις γωνίες ως προς τους 3 άξονες περιστροφής (roll, pitch και yaw) και αυτό μας επιτρέπει πιο άμεση ανταπόκριση σε μετατοπίσεις του οδηγού (aggressive maneuvering). Η αναγνώριση του ρομπότ-στόχου (leader) γίνεται με χρήση βαθιών συνελικτικών νευρωνικών δικτύων (CNN) [2] που πραγματοποιούν κατάτμηση της εικόνας που λαμβάνεται από την κάμερα του πολυκόπτερου και εύρεση της θέσης του ρομπότ-οδηγού στο επίπεδο της εικόνας. Ο στόχος της εφαρμογής αυτής είναι να μπορεί το πολυκόπτερο να κινηθεί αυτόνομα ακολουθώντας τον οδηγό σε ένα σύνολο από διαφορετικά περιβάλλοντα και συνθήκες χωρίς να χρειάζεται εξωτερική παρέμβαση. Στα πλαίσια της ίδιας εφαρμογής υλοποιούμε και ένα ενδιάμεσο στάδιο που αφορά την ακολούθηση ενός σταθερού περιγράμματος (static contour). Καθώς οι αρχές λειτουργίας είναι οι ίδιες με την τελικη εφαρμογή αυτό το στάδιο ήταν χρήσιμο για την κλιμακούμενη αύξηση της δυσκολίας της εφαρμογής. Οι δύο παραπάνω εφαρμογές εκτελούνται πειραματικά προς απόδειξη της ορθής λειτουργίας τους.

1.2 Σημασία του προβλήματος

Η διπλωματική αυτή εργασία κατευθύνεται ομμοροπα με τις τάσεις της επιστήμης αξιοποιώντας και συνδυάζοντας την ενισχυτική μάθηση και τα Μη Επανδρωμένα Συστήματα.

Όσον αφορά την ενισχυτική μάθηση και γενικότερα την τεχνητή νοημοσύνη, τα τελευταία χρόνια υπάρχει ραγδαία εξέλιξη και αυξανόμενο ενδιαφέρον στο τομέα αυτό και τις εφαρμογές του. Με την ανάπτυξη της υπολογιστικής ισχύος και το πλήθος των δεδομένων τα οποία είναι διαθέσιμα, η τεχνητή νοημοσύνη έχει εισέλθει στις περισσότερες εφαρμογές δίνοντας νέες δυνατότητες και λύνοντας προβλήματα που δεν ήταν εφικτό μέχρι πρότερα. Παράδειγμα αυτού είναι η ανάπτυξη των βαθιών συνελικτικων δικτύων που επέτρεψαν την αναγνώριση εικόνων. Ενδιαφέρον αποτελεί ότι πολλές από αυτές τις τεχνικές εμπνεύστηκαν από τη φύση και τα βιολογικά συστήματα. Όπως τα συνελικτικά δίκτυα εμπνεύστηκαν από τις βιολογικές δομές επεξεργασίας εικόνας έτσι και η ενισχυτική μάθηση βασίστηκε σε βασικές αρχές εκπαίδευσης από το τομέα της ψυχολογίας. Η ενισχυτική μάθηση μας δίνει τη δυνατότητα να εκπαιδεύσουμε έναν πράκτορα (agent) σε ένα προσομοιωμένο περιβάλλον και έπειτα να τον μεταφέρουμε στο πραγματικό περιβάλλον.

Σχετικά με τα Μη Επανδρωμένα Αεροσκάφη τα τελευταία χρόνια συναντούν και αυτά όλο και περισσότερη απήχηση. Τα πολυκόπτερα αξιοποιούνται σε εφαρμογές γεωργίας, μεταφορών, επιθεώρησης, κινηματογραφίας, ασφάλειας και πολλές άλλες. Η ευελιξία της κίνησης τους και το συγκριτικά μικρότερο κόστος σε σχέση με τα επανδρωμένα αεροσκάφη, τα κατέστησε πάρα πολύ δημοφιλή. Ειδικότερα όταν αυτά συνδυάζονται με την τεχνητή νοημοσύνη και γίνονται αυτόνομα οι δυνατότητες πολλαπλασιάζονται. Με βάση τις παραπάνω εξελίξεις επιλέχθηκε και η εφαρμογή που υλοποιούμε στη παρούσα διπλωματική προάγοντας και ερευνώντας τους ραγδαία εξελισσόμενους αυτούς τομείς.

1.3 Προηγούμενη έρευνα

Ερευνητές από διαφορετικά πανεπιστήμια έχουν ασχοληθεί με εφαρμογές παρόμοιες με αυτές που καταπιάνεται η παρούσα διπλωματική. Ξεκινώντας από το Εργαστήριο Ρομποτικής του κ.Κυριακόπουλου βρίσκουμε εφαρμογές οπτικής ανατροφοδότησης σε Μη Επανδρωμένα συστήματα για την ακολούθηση μεταβαλλόμενων περιγραμμάτων όπως ακτές [3]. Όσον αφορά τον έλεγχο Μη Επανδρωμένων με χρήση ενισχυτικής μάθησης οι Sampedro και Rodriguez-Ramos παρουσιάζουν στο [4] μία εφαρμογή προσγείωσης σε ένα κινούμενο επίπεδο, ενώ οι ίδιοι παρουσιάζουν στο [5] και μία εφαρμογή ακολούθησης ενός

σημαδιού (marker) με χρήση ενισχυτικής μάθησης και ελέγχου στον εσωτερικό βρόχο (inner loop). Στη συνεχεία στο [6] ο Rodriguez-Ramos υλοποιεί μια μεθοδολογία ακολούθησης ενός μη επανδρωμένου αεροσκάφους από ένα άλλο μη Επανδρωμένο Αεροσκάφος (μΕΑ) με χρήση ενισχυτικής μάθησης δίνοντας έμφαση στην εκπαίδευση με χρήση συνθετικών δεδομένων για την καλύτερη γενίκευση του ελεγκτή. Η τελευταία αποτελεί και την πλησιέστερη εφαρμογή σε αυτό που υλοποιείται σε αυτή τη διπλωματική.

1.4 Συνεισφορά

Σε αυτή τη διπλωματική εφαρμόζουμε μεθόδους από τις προηγούμενες ερευνες και τις συνδυάζουμε για να υλοποιήσουμε την δική μας παραλλαγή. Πιο συγκεκριμένα αξιοποιούμε την ενισχυτική μάθηση εφαρμόζοντας τον έλεγχο στο εσωτερικό βρόχο και δεν κάνουμε έλεγχο ταχύτητας όπως στο [4]. Αυτό μας επιτρέπει την καλύτερη ακολούθηση του οδηγού σε σχέση με μια κλασική τεχνική οπτικής ανατροφοδότησης [7][8][9][10][11], γιατί η ενισχυτική μάθηση διασφαλίζει καλύτερη πρόβλεψη ταχύτητας στην ακανονιστη τροχιά του οδηγού. Επίσης αντί για σημάδι (marker) κανουμε χρήση συνελικτικών δικτύων για την αναγνώριση της θέσης του οδηγού-ρομπότ που μας δίνει περισσότερη ευελιξία και ευρωστία. Επίσης ο οδηγός-ρομποτ είναι επίγειος σε αντιθεση με τον εναέριο στόχο που καθιστά το πρόβλημα πιο δύσκολο λόγω του ανομοιόμορφου υποβάθρου. Τέλος η εκπαίδευση λαμβάνει χώρα σε προσομοίωση όπως και στις προηγούμενες έρευνες χωρίς την χρήση τυχαιότητας στην δημιουργία του περιβάλλοντος καθώς η μετάβαση από την προσομοίωση στο πραγματικό κόσμο είναι σχεδόν άμεση.

1.5 Δομή της εργασίας

Αυτή η διπλωματική είναι χωρισμένη σε 5 κεφάλαια. Το πρώτο ήταν η εισαγωγή που έδωσε μία σφαιρική εικόνα για το αντικείμενο της διπλωματικής, το στόχο και τη συνεισφορά του.

Στο δεύτερο κεφάλαιο αναλύεται το τεχνικό και μαθηματικό κομμάτι που πρέπει να υλοποιηθεί και το υπόβαθρο γνώσεων στο οποίο βασίζεται.

Στο τρίτο κεφάλαιο, αναφέρονται πιο αναλυτικά τα εργαλεία, τα μέσα και η γενικότερη αρχιτεκτονική του συστήματος που υλοποιήθηκε για να πραγματοποιήσουμε την εφαρμογή μας.

Στο τέταρτο κεφάλαιο, παρουσιάζουμε την πειραματική διαδικασία τόσο στην προσομοίωση όσο και στη πραγματικότητα και προβάλλουμε τα αποτελέσματα.

Τέλος, στο πέμπτο κεφάλαιο σχολιάζουμε τα αποτελέσματα και αναφέρουμε πτυχές που χρήζουν έρευνας στο μέλλον.

Κλείνοντας έχουμε το παράρτημα που περιέχει λεπτομέρειες υλοποίησης και εκκίνσης ρουτινών.

Κεφάλαιο 2

Τεχνική και θεωρητική διατύπωση του προβλήματος

Στο παρόν κεφάλαιο παρουσιάζεται η εφαρμογή της διπλωματικής μέσα από ένα πιο τεχνικό και μαθηματικό πρίσμα, εξηγώντας τις εξισώσεις που πρέπει να επιλυθούν για την επίτευξη της σωστής λειτουργίας της εφαρμογής.

2.1 Έλεγχος εσωτερικού βρόχου

Το ρομποτ που χρησιμοποιείται στην εφαρμογή και ελέγχουμε είναι ένα πολυκόπτερο (multirotor). Ο έλεγχος που υλοποιούμε όπως αναφέρθηκε και στην εισαγωγή γίνεται στον εσωτερικό βρόχο δηλαδή ελέγχουμε τις γωνίες γύρω από τους τρεις άξονες περιστροφής.

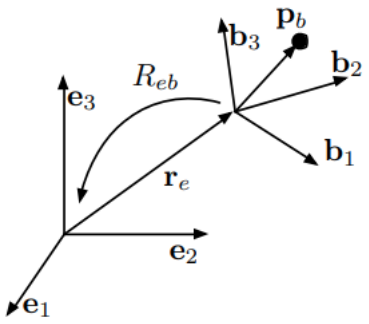
Το πολυκόπτερο είναι ένα υποεπενεργούμενο σύστημα που πρέπει να διατηρεί σε ισορροπία ένα σύνολο δυνάμεων και ροπών για να είναι δυνατή η πτήση του [12], [13]. Πιο συγκεκριμένα το δυναμικό μοντέλο του φαίνεται παρακάτω:

$$\begin{bmatrix} F \\ \tau \end{bmatrix} = \begin{bmatrix} m * 1_3 & 0_3 \\ 0 & I_3 \end{bmatrix} \begin{bmatrix} \mathrm{a} \\ \alpha \end{bmatrix} + \begin{bmatrix} \omega \times mv \\ \omega \times I_3\omega \end{bmatrix} \tag{2.1}$$

όπου:
F : συνολική δύναμη
τ : συνολική ροπή
m : μάζα
a : γραμμική επιτάχυνση
α : γωνιακή επιτάχυνση
I_3 : ροπή αδράνειας

1_3 : πίνακας με άσσους
0_3 : πίνακας με μηδενικά
v : γραμμική ταχύτητα
ω : γωνιακή ταχύτητα

Οι περιστροφές γύρω από τους τρεις άξονες παραμετροποιούνται ως γωνίες Euler όπως φαίνεται παρακάτω :



Σχήμα 2.1: Γραμμικός μετασχηματισμός. $p_e = \mathcal{R}_{eb}.p_b + r_e$. Σχήμα από τις διαφάνειες του πανεπιστημίου Carnegie Mellon University.

$$\mathcal{R}_{eb} = \mathcal{R}_z(\psi)\mathcal{R}_y(\theta)\mathcal{R}_x(\phi) \tag{2.2}$$

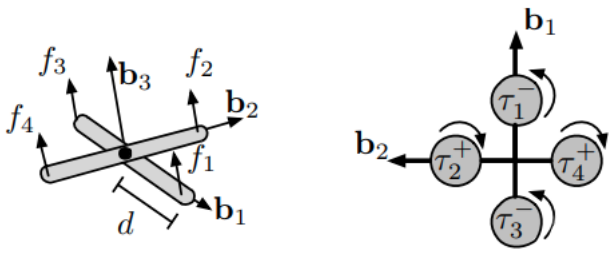
με $\mathcal{R}_z(\psi)$ να είναι το yaw, το $\mathcal{R}_y(\theta)$ να είναι το pitch, και το $\mathcal{R}_x(\phi)$ να είναι το roll.

Το σύνολο ροπών που ασκούνται στο πολυκόπτερο και τις οποίες ελέγχουμε μέσω της γωνιακής ταχύτητας των κινητήρων φαίνονται παρακάτω:

$$\tau_{b1} = d(f_2 - f_4) \tag{2.3}$$

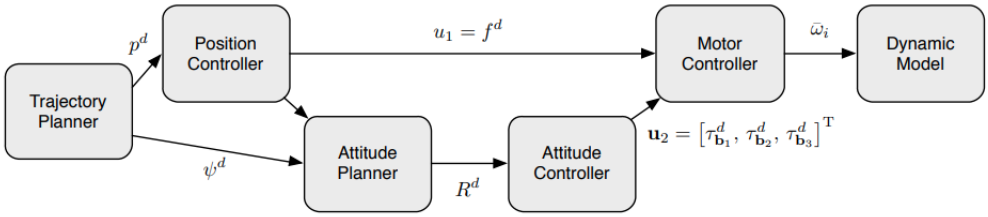
$$\tau_{b2} = d(f_1 - f_3) \tag{2.4}$$

$$\tau_{b3} = -\tau_1 + \tau_2 - \tau_3 + \tau_4 \tag{2.5}$$



Σχήμα 2.2: Ροπές που ασκούνται στο πολυκόπτερο.

Ο έλεγχος του συστήματος γίνεται με την χρήση ένος ελεγκτή PID (proportional - integral - derivative) και έχει πολλά επίπεδα ελέγχου τα οποία φαίνονται στο παρακάτω διάγραμμα:



Σχήμα 2.3: Διάγραμμα ελέγχου του πολυκόπτερου.

Ο έλεγχος του εσωτερικού βρόχου διέπεται από τις παρακάτω εξισώσεις:

$$u_2 = -k_R e_R - k_\omega e_\omega \tag{2.6}$$

όπου e_R είναι το σφάλμα στο προσανατολισμό του πολυκόπτερου, e_ω είναι το σφάλμα στη γωνιακή ταχύτητα και k_R, k_ω σταθερές.

Καθώς το σφάλμα προσανατολισμού είναι μη γραμμικό, γραμμικοποιούμε γύρω από τη θέση ισορροπίας και τελικά έχουμε ότι:

$$e_R \approx \begin{bmatrix} 0 & \Delta\psi & -\Delta\theta \\ -\Delta\psi & 0 & -\Delta\phi \\ \Delta\theta & -\Delta\phi & 0 \end{bmatrix} \tag{2.7}$$

$$e_R \approx \begin{bmatrix} \Delta\phi & \Delta\theta & \Delta\psi \end{bmatrix}^\top \tag{2.8}$$

2.2 Ενισχυτική μάθηση

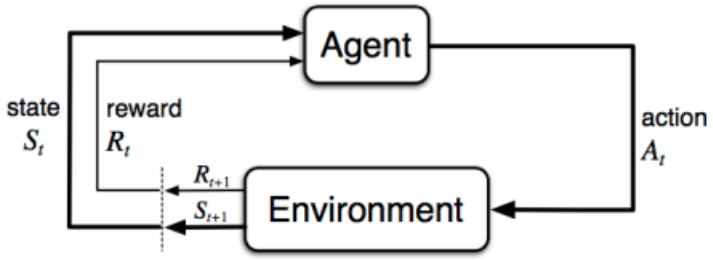
Στο προηγούμενο υποκεφάλαιο αναφέρεται πως δίνονται εντολές στις γωνίες roll, pitch, yaw. Στο παρόν υποκεφάλαιο θα αναφερθούμε στο σύστημα που παράγει αυτές τις εντολές. Όπως αναφέρθηκε και στην εισαγωγή ο έλεγχος γίνεται με χρήση ενισχυτικής μάθησης.

Η ενισχυτική μάθηση είναι ο τομέας της τεχνητής νοημοσύνης και της μηχανικής μάθησης στον οποίο ενας πράκτορας (agent) εκτελεί κινήσεις σε ένα περιβάλλον με σκοπό την μεγιστοποίηση της συνολικής ανταμοιβής δεδομένης μιας συνάρτησης ανταμοιβής. Ουσιαστικά η ενισχυτικη μάθηση προσπαθεί να μάθει να αντιστοιχεί καταστάσεις (states) σε κινήσεις (actions) με την μέγιστη ανταμοιβή. Η ενισχυτική μάθηση έχει καταφέρει να λύσει πολλά σύνθετα προβλήματα και είναι ένας συνεχώς αναπτυσσόμενος τομέας.

Ο στόχος της ενισχυτικής μάθησης είναι να εκπαιδεύσει τον πράκτορα (agent) σε μία βέλτιστη πολιτική (policy) που θα του επιτρέψει να λύσει το πρόβλημα. Συνεπώς ο πράκτορας (agent) μαθαίνει μέσα από μια διαδικασία δοκιμής και λάθους και λαμβάνοντας ανατροφοδότηση από το περιβάλλον του. Στην διαδικασία αυτή είναι πολύ σημαντική η ισορροπία ανάμεσα σε εξερεύνηση νέων καταστάσεων και στην αξιοποίηση των ήδη γνωστών.

Τα περισσότερα προβλήματα ενισχυτικής μάθησης μπορούν να μοντελοποιηθούν με μία διαδικασία Markov (Markov Decision Process) ή MDP [14] [15]. Σε ένα MDP η επόμενη κατάσταση εξαρτάται μόνο από την παρούσα. Ένα MDP αποτελείται από τα εξής στοιχεία $M = (S, A, P, R, \gamma)$ οπού S η κατάσταση (state), A η κίνηση (action), P η πιθανότητα μετάβασης από μία κατάσταση σε μία άλλη και γ ένας παράγοντας μείωσης (discount factor) που είναι χρήσιμος για να μειώνει την σημαντικότητα των μελλοντικών ανταμοιβών (reward) λόγω της αυξημένης τους αβεβαιότητας.

Η βασική λειτουργία του συστήματος φαίνεται στο παρακάτω διάγραμμα:



Σχήμα 2.4: Διάγραμμα αλληλεπίδρασης περιβάλλοντος και πράκτορα.

Η πολιτική που μαθαίνει το σύστημα μας είναι μια αντιστοίχηση που λέει στον πράκτορα (agent) τι κίνηση να κάνει σε μία κατάσταση s. Για αυτό το λόγο είναι αναγκαίο να ορίσουμε συναρτήσεις αξίας (value functions) που δίνουν μία μετρική για το πόσο καλή είναι μία κατάσταση και κάθε κίνηση σε καθεμία από αυτές. Έτσι ορίζουμε τη συνάρτηση αξίας-κατάστασης v (state-value) η οποία είναι η προσδοκώμενη επιστροφή (return) όταν βρισκόμαστε σε μια συγκεκριμένη κατάσταση και ακολουθούμε μία συγκεκριμένη πολιτική. Αντίστοιχα, ορίζουμε την συνάρτηση αξίας-κίνησης q (action-value) που είναι η προσδοκώμενη αξία όταν επιλέγουμε μία κίνηση σε μια συγκεκριμένη κατάσταση ακολουθώντας μία συγκεκριμένη πολιτκή. Πιο απλά οι δύο αυτές συναρτήσεις μας δείχνουν πόσο καλή είναι μια κατάσταση και μία κίνηση αντίστοιχα.

Συνεπώς έχουμε:

$$v_\pi(s) = E_\pi\left[\sum_{k=0}^{\infty} \gamma^k \mathcal{R}_{t+k} \middle| \mathcal{S}_t = s\right]$$
$$q_\pi(s,a) = E_\pi\left[\sum_{k=0}^{\infty} \gamma^k \mathcal{R}_{t+k} \middle| \mathcal{S}_t = s, \mathcal{A}_t = a\right]$$

Μία μέθοδος που χρησιμοποιείται συχνά στην ενισχυτική μάθηση είναι η actor-critic [16] μέθοδος που αξιοποιεί δύο βαθιά νευρωνικά δίκτυα τα οποία εκπαιδεύονται και μοντελοποιούν τις συναρτήσεις αξίας-κατάστασης (state-value) και αξίας-κίνησης (action-value). Η εκπαίδευση γίνεται με χρήση της παραγώγου μιας συνάρτησης επίδοσης και καθώς θέλουμε να μεγιστοποιήσουμε την επίδοση χρησιμοποιούμε gradient ascent.

Στη παρούσα εφαρμογή χρησιμοποιούμε Deep Deterministic Policy Gradient πράκτορα (agent) που ειναι κατάλληλος για συνεχές χώρο κινήσεων (continuous space), οπως αποδεικνύεται στο [17]. Ο ντετερμινιστικός αυτός αλγόριθμος είναι πιο αποδοτικός στον υπολογισμό σε σχέση με τον αντιστοιχο στοχαστικό και για να πετύχει επαρκή εξερεύνηση χρησιμοποιεί μία off-policy στρατηγική που σημαίνει ότι μαθαίνει ο actor από μία στρατηγική εξερεύνησης (exploratory policy). Πιο συγκεκριμένα, χρησιμοποιούνται target networks τα οποία διασφαλίζουν καλύτερη σύγκλιση της εκπαίδευσης και η εξερεύνηση (exploration) γίνεται με χρήση Ornstein-Uhlenbeck θορύβου όπως εξηγείται αναλυτικά στο Continuous Control with Deep Reinforcement Learning [18]. Επίσης ο αλγόριθμος αυτός κάνει χρήση του experience replay που σημαίνει ότι χρησιμοποιεί ένα καταχωρητή (buffer) στον οποίο και αποθηκεύει τις μεταβάσεις που συμβαίνουν και μετά επιλέγει τυχαία ένα σύνολο εγγραφών με τις οποίες εκπαιδεύει τα νευρωνικά. Αυτό ωφελεί στο ότι έχουμε καλύτερη αποσυσχέτιση των εγγραφών και τελικά καλύτερη σύγκλιση.

Ο αλγόριθμος φαίνεται αναλυτικά παρακάτω:

Randomly initialize critic network $Q(s, a|\theta^Q)$ and actor $\mu(s|\theta^\mu)$ with weights θ^Q and θ^μ.
Initialize target network Q' and μ' with weights $\theta^{Q'} \leftarrow \theta^Q$, $\theta^{\mu'} \leftarrow \theta^\mu$
Initialize replay buffer R
for episode = 1, M **do**
  Initialize a random process $\mathcal{N}$ for action exploration
  Receive initial observation state s_1
  for t = 1, T **do**
    Select action $a_t = \mu(s_t|\theta^\mu) + \mathcal{N}_t$ according to the current policy and exploration noise
    Execute action a_t and observe reward r_t and observe new state s_{t+1}
    Store transition (s_t, a_t, r_t, s_{t+1}) in R
    Sample a random minibatch of N transitions (s_i, a_i, r_i, s_{i+1}) from R
    Set $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$
    Update critic by minimizing the loss: $L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$
    Update the actor policy using the sampled policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N} \sum_i \nabla_a Q(s, a|\theta^Q)|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu} \mu(s|\theta^\mu)|_{s_i}$$

    Update the target networks:

$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$$
$$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$

  end for
end for

Σχήμα 2.5: Deep deterministic policy gradients αλγόριθμος.

Τέλος, πολύ σημαντικες παράμετροι για την εκπαίδευση ενός πράκτορα (agent) είναι η διαμόρφωση της συνάρτησης κόστους (reward function) και η επιλογή των μεταβλητών που ορίζουν μία κατάσταση (state). Αυτά είναι ειδικά για κάθε εφαρμογή και κρίσιμα για την επιτυχία της εκπαίδευσης.

Συνεπώς, ο πράκτορας της ενισχυτικής μάθησης (Reinforcement Learning agent) παίρνει είσοδο την κατάσταση (state), η οποία προκύπτει από την επεξεργασία της εικόνας που λαμβάνουμε από τη κάμερα, και υπολογίζει την ανταμοιβή (reward) για κάθε μετάβαση ώστε τελικά να μάθει μία πολιτική (policy) η οποία να λύνει το πρόβλημα που έχουμε θέσει.

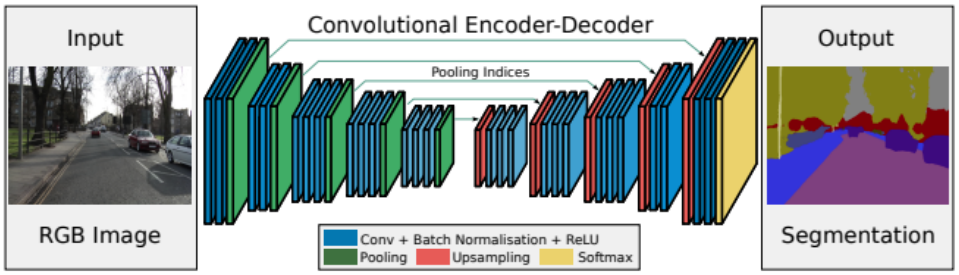
2.3 Κατάτμηση εικόνας

Για την εύρεση του επίγειου ρομπότ μέσα στην εικόνα που λαμβάνουμε από την κάμερα χρησιμοποιούμε μία τεχνική που ονομάζεται κατάτμηση εικόνας (image

Σχήμα 2.6: Παράδειγμα κατάτμησης εικόνας.

segmentation)[19]. Η κατάτμηση εικόνας (image segmentation) είναι η διαδικασία κατηγοριοποίησης του κάθε εικονοκυττάρου (pixel) της εικόνας σε μία κλάση. Στο δικό μας πρόβλημα έχουμε μόνο δύο κλάσεις καθώς προσπαθούμε να διαχωρίσουμε το επίγειο ρομπότ από το υπόβαθρο του (background). Η υλοποίηση της κατάτμησης εικόνας (image segmentation) γίνεται με χρήση συνελικτικών νευρωνικών δικτύων με πολλά επίπεδα και δομή κωδικοποιητή-αποκωδικοποιητή (encoder-decoder). Ο κωδικοποιητής (encoder) εξάγει χαρακτηριστικά από την εικόνα με φίλτρα ενώ ο αποκωδικοποιητής (decoder) παράγει το τελικό αποτέλεσμα το οποίο συνήθως είναι μία μάσκα κατάτμησης. Για την εκπαίδευση αυτού του δικτύου χρησιμοποιούμε την cross entropy συνάρτηση κόστους (loss function) με βάση τις αληθινές κλάσεις των εικονοκυττάρων (pixel) που έχουν κατηγοριοποιηθεί (annotated) στο σύνολο των δεδομένων μας (dataset). Επίσης η επαύξηση των εικόνων (image augmentation) [20] είναι αρκετά χρήσιμη στην αποτελεσματικότερη εκπαίδευση του δικτύου και για αυτό και αξιοποιείται στην εφαρμογή αυτή.

Για την εφαρμογή αυτής της διπλωματικής επιθυμούμε να έχουμε μικρό σε μέγεθος μοντέλο ώστε να μπορεί να υποστηριχθεί από την μνήμη του υπολογιστή που έχει το πολυκόπτερο (multirotor) μας καθώς και χαμηλό χρόνο απόκρισης (inference time) καθώς η εφαρμογή είναι πραγματικού χρόνου. Για αυτούς τους λόγους επιλέχθηκε το MobileNet του οποίου η αρχιτεκτονική παρουσιάζεται στη δημοσίευση Mobilenets: Efficient convolutional neural networks for mobile vision applications [21].

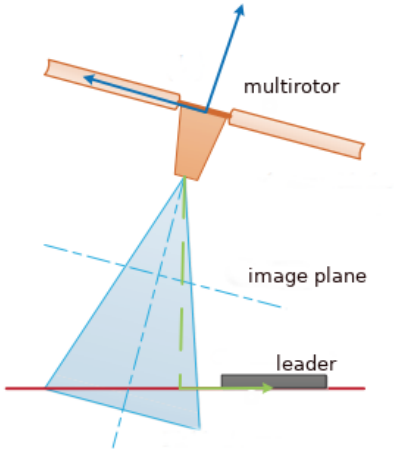


Σχήμα 2.7: SegNet: Βαθιά συνελικτική αρχιτεκτονική κωδικοποιητή-αποκωδικοποιητή για κατάτμηση εικόνας, από το [1].

2.4 Προβολή εικόνας - Εικονικός σταθεροποιητής

Για τον υπολογισμό της θέσης του επίγειου οχήματος μέσω της εικόνας που λαμβάνουμε απαιτείται να λάβουμε υπόψη τη κλίση του πολυκόπτερου (multirotor), διότι διαφορετικά η σχετική θέση φαίνεται μετατοπισμένη. Σε αυτό το υποκεφάλαιο εξηγείται πως αντιμετωπίζεται το παραπάνω πρόβλημα της μετατόπισης της εικόνας λόγω της κλίσης του πολυκόπτερου (multirotor). Το πρόβλημα αυτό προκύπτει καθώς για να κινηθεί το πολυκόπτερο προς μία κατεύθυνση, στρέφεται γύρω από έναν άξονα του με αποτέλεσμα η εικόνα της κάμερας, η οποία είναι προσανατολισμένη προς τα κάτω, να μετατοπίζεται επίσης. Η μετατόπιση αυτή κάνει το ρομπότ-οδηγό να φαίνεται ότι κινείται προς την αντίθετη κατεύθυνση από ότι συμβαίνει στην πραγματικότητα. Πιο απλά, αν το πολυκόπτερο κινούνταν προσπαθώντας να μειώσει την απόσταση του οδηγού από το κέντρο της εικόνας τότε λόγω της στροφής του πολυκόπτερου, στην εικόνα θα φαινόταν αρχικά να αυξάνεται η απόσταση από το κέντρο. Για να το αντιμετωπίσουμε αυτό πρέπει να κάνουμε ένα γραμμικό μετασχηματισμό στην εικόνα λαμβάνοντας υπόψη το roll και pitch του πολυκόπτερου και προβάλλοντας την εικόνα στο κάθετο επίπεδο.

Στην εικόνα 2.8 φαίνεται ένα χαρακτηριστικό παράδειγμα του προβλήματος που περιγράψαμε παραπάνω. Το πολυκόπτερο θέλει να κινηθεί προς τα δεξία όπως βλέπουμε στην εικόνα και για αυτό αποκταέι την κλίση αυτή. Την επόμενη χρονική στιγμή η απόσταση του πολυκόπτερου από το όχημα-οδηγό έχει μειωθεί αλλά στο επίπεδο της εικόνας η απόσταση του ρομπότ-οδηγού από το κέντρο της εικόνας έχει αυξηθεί. Καθώς η πληροφορία για τη θέση του οδηγού λαμβάνεται από την εικόνα, η απευθείας χρήση της πληροφορίας αυτής είναι προβλήματική



Σχήμα 2.8: Παράδειγμα εικονικού σταθεροποιητή (Virtual gimbal).

και για αυτό κάνουμε προβολή της εικονας στο κάθετο επίπεδο κάτω από το πολυκόπτερο.

Για να το υλοποιήσουμε αυτό πρέπει αρχικά να μεταφερθούμε από συντεταγμένες εικονοστοιχείων (pixel coordinates) σε συντεταγμένες κόσμου (world coordinates), να εφαρμόσουμε τον γραμμικό μετασχηματισμό που λαμβάνει υπόψιν τις περιστροφές roll και pitch, την οποία διαδικασία ονομάζουμε εικονικό σταθεροποιητή (virtual gimbal) και έπειτα να επιστρέψουμε σε συντεταγμένες εικονοστοιχείων (pixel coordinates).

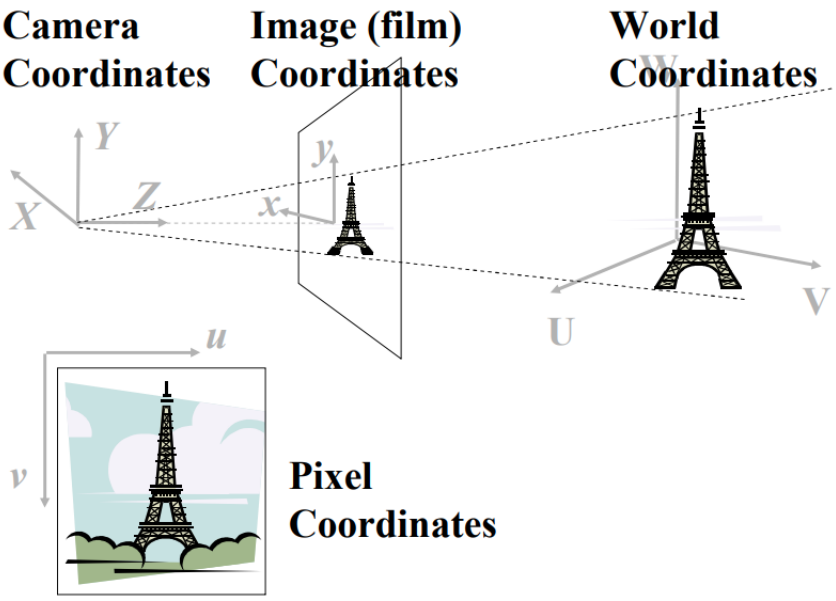
Αυτό γίνεται μέσα από μία σειρά μετασχηματισμών που φαίνονται στο Σχήμα 2.10.

Πρώτα έχουμε την μετατροπή από συντεταγμένες εικονοστοιχείων (pixel coordinates) σε συντεταγμένες εικόνας (image coordinates):

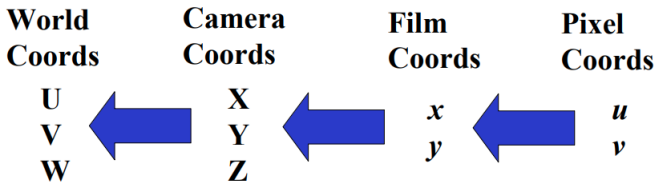
$$u = \frac{1}{s_x}x + O_x \quad v = \frac{1}{s_y}y + O_y \tag{2.9}$$

όπου s_x, s_y είναι οι κλίμακες (effective scales) και αφορούν την αναλογία των εικονοστοιχείων (pixels).

Η μετατροπή από συντεταγμένες εικόνας (image coordinates) σε συντεταγμένες κάμερας (camera coordinates) διέπεται από τη παρακάτω σχέση που



Σχήμα 2.9: Συστήματα συντεταγμένων, από τις διαλέξεις του Penn State university.



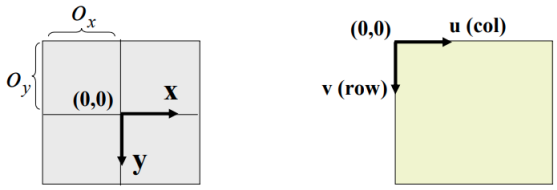
Σχήμα 2.10: Μετασχηματισμοί συστημάτων συντεταγμένων, από τις διαλέξεις του Penn State university.

προκύπτει από την προβολική γεωμετρία:

$$x = f\frac{X}{Z} \quad y = f\frac{Y}{Z} \tag{2.10}$$

Συνδιάζοντας τις (2.9) και (2.10) έχουμε:

$$u = \frac{1}{s_x}f\frac{X}{Z} + O_x \quad v = \frac{1}{s_y}f\frac{Y}{Z} + O_y \tag{2.11}$$



Σχήμα 2.11: Συντεταγμένες εικόνας και εικονοστοιχείων.

για τις οποίες τα f, s_x, s_y, O_x, O_y είναι εσωτερικά χαρακτηριστικά της κάμερας και τα έχουμε δεδομένα, και το Z είναι το ύψος στο οποίο βρίσκεται το πολυκόπτερο.

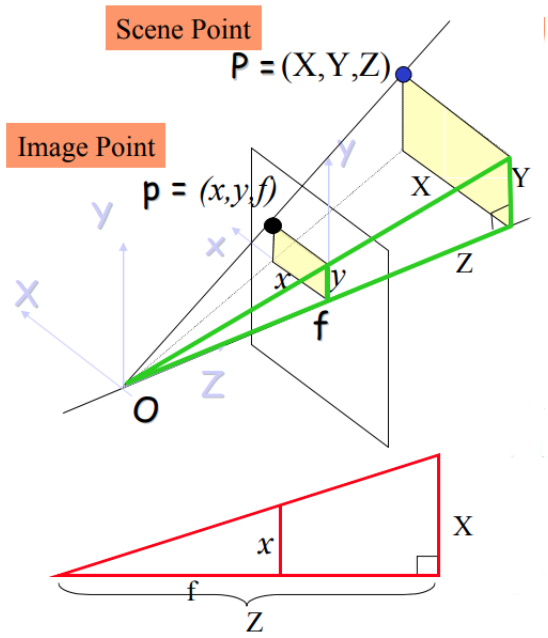
Για τη μετατροπή από συντεταγμένες εικόνας (camera coordinates) σε πραγματικές συντεταγμένες (world coordinates) θεωρούμε την αρχη των αξόνων στο κέντρο του πολυκόπτερου με z άξονα προς τα κάτω και x άξονα προς τα μπροστά. Έτσι πρέπει να περιστρέψουμε το σύστημα συντεταγμένων της κάμερας (camera frame) κατά $Rot_z(\pi/2)$ και να το μετατοπίσουμε κατά $Tra_z(0.2m)$.

Στη συνέχεια πρέπει να υλοποιήσουμε το γραμμικό μετασχηματισμό των συντεταγμένων δηλαδή τις περιστροφές γύρω από το pitch και roll που γίνεται με χρήση του παρακάτω μετασχηματισμού:

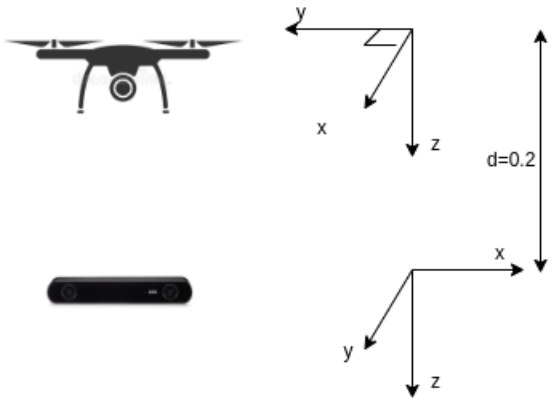
$$\begin{bmatrix} c_\phi & -s_\theta s_\phi & c_\theta s_\phi \\ 0 & c_\theta & s_\theta \\ -s_\theta & -s_\theta c_\phi & c_\theta c_\phi \end{bmatrix} \tag{2.12}$$

όπου $c_* = cos(*), s_* = sin(*), \theta = pitch, \phi = roll$.

Τέλος, πρέπει από πραγματικές συντεταγμένες (world coordinates) να επιστρέψουμε σε συντεταγμένες εικονοστοιχείων (pixel coordinates) χρησιμοποιώντας τις ίδιες σχέσεις με προηγουμένως.



Σχήμα 2.12: Προοπτική προβολή εικόνας.



Σχήμα 2.13: Σύστημα συντεταγμένων πολυκόπτερου και κάμερας.

Κεφάλαιο 3

Προσέγγιση επίλυσης

Η εφαρμογή της διπλωματικής αυτής εργασίας για να υλοποιήσει τις μεθοδολογίες που παρουσιάζονται στο παραπάνω κεφάλαιο και να πραγματοποιηθεί διαθέτει ένα σύνολο από απαιτήσεις σε λογισμικό (software) και μηχανημάτων (hardware) καθώς και τη σωστή επιλογή σημαντικών παραμέτρων στις μαθηματικές φόρμουλες.

3.1 Μηχανήματα

3.1.1 Πολυκόπτερο

Το βασικό όχημα για την εφαρμογή αυτή είναι το πολυκόπτερο. Το εργαστήριο διαθέτει δύο, το ένα με τέσσερις κινητήρες και το άλλο με οχτώ. Η πτήση ελέγχεται από ένα μικροϋπολογιστή με χρήση διαφόρων αισθητήρων όπως IMU (inertial measurement unit), GPS (global positioning system) και βαρόμετρο. Οι διαστάσεις τους είναι 80 και 140 εκατοστά διάμετρος αντίστοιχα και έχουν δυνατότητα να σηκώσουν τουλάχιστον 2 και 5 κιλά αντίστοιχα ώστε να υποστηρίξουν τον εξοπλισμό που φέρουν.

Το πολυκόπτερο (multirotor) είναι εφοδιασμένο με ένα υπολογιστή ειδικά σχεδιασμένο για νευρωνικά δίκτυα ονομαζόμενος Jetson Xavier και μία κάμερα βάθους, την ZED2.

Σχήμα 3.1: Πολυκόπτερα.

3.1.2 Υπολογιστής τεχνητής νοημοσύνης (Nvidia Jetson Xavier)

Ο Nvidia Jetson Xavier είναι ένας υπολογιστής που προορίζεται για χρήση σε εφαρμογές τεχνητής νοημοσύνης και ρομποτικής. Προσφέρει ταχύτητα στους υπολογισμούς του, με μικρή κατανάλωση ενέργειας και μικρό βάρος που το καθιστά κατάλληλο για κινητές (mobile) εφαρμογές.

Πίνακας 3.1: Jetson Xavier χαρακτηριστικά.

GPU	512-core Volta GPU με Tensor Cores
CPU	8-core ARM v8.2 64-bit CPU, 8MB L2 + 4MB L3
Μνήμη	32GB 256-Bit LPDDR4x — 137GB/s
Δίσκος	32GB eMMC 5.1
Μέγεθος	105 mm x 105 mm x 65 mm
Βάρος	$1548g$

3.1.3 Κάμερα βάθους (RGB-D, ZED2)

Η ZED2 κάμερα αναπτύχθηκε από την Stereo Labs και είναι μία κάμερα που δίνει δυνατότητες χωρικής αντίληψης των αντικειμένων. Χρησιμοποιεί τεχνητή νοημοσύνη (AI , aritficial intelligence) για την πραγματοποίηση κάποιων ενσωματομένων (built in) λειτουργιών καθώς και περιέχει ένα σύνολο από αισθητήρες που την καθιστούν χρήσιμη σε ένα μεγάλο φάσμα εφαρμογών. Σημαντικά χαρακηριστικά για την υλοποίηση της εφαρμογής είναι οι διαστάσεις

Σχήμα 3.2: Jetson Xavier.

της εικόνας που λαμβάνουμε από τη κάμερα καθώς και το εστιακό μήκος (focal length), χαρακτηριστικά τα οποία ρυθμίζονται στην ZED2. Άλλα σημαντικά χαρακτηριστικά της κάμερας φαίνονται στο Πίνακα 3.3.

Πίνακας 3.2: ZED2 χαρακτηριστικά.

Οπτικό πεδίο	110 μοίρες οριζόντια, 70 κατακόρυφα, 120 διαγωνία μέγιστο
Εύρος βάθους	20 cm εώς 20 m
Ανάλυση	720p εώς 2K
FPS	15, 30, 60

Σχήμα 3.3: ZED2.

3.1.4 Επίγειο ρομπότ οδηγός (Summit XL)

Το Summit XL στην εφαρμογή μας αποτελεί το όχημα οδηγό που ακολουθείται από το πολυκόπτερο (multirotor). Το Summit XL είναι ένα όχημα που ανέπτυξε η Robotnik που έχει πολλές δυνατότητες και εφαρμογές. Διαθέτει πολλούς αισθητήρες που το κάνουν κατάλληλο για αυτόνομες εφαρμογές αλλά

στα πλαίσια αυτής της διπλωματικής ο έλεγχος του γίνεται ασύρματα με χρήση τηλεχειριστηρίου.

Σχήμα 3.4: Summit XL.

Κάποια τεχνικά χαρακτηριστικά φαίνονται παρακάτω στο Πίνακα 3.3.

Πίνακας 3.3: Summit XL χαρακτηριστικά.

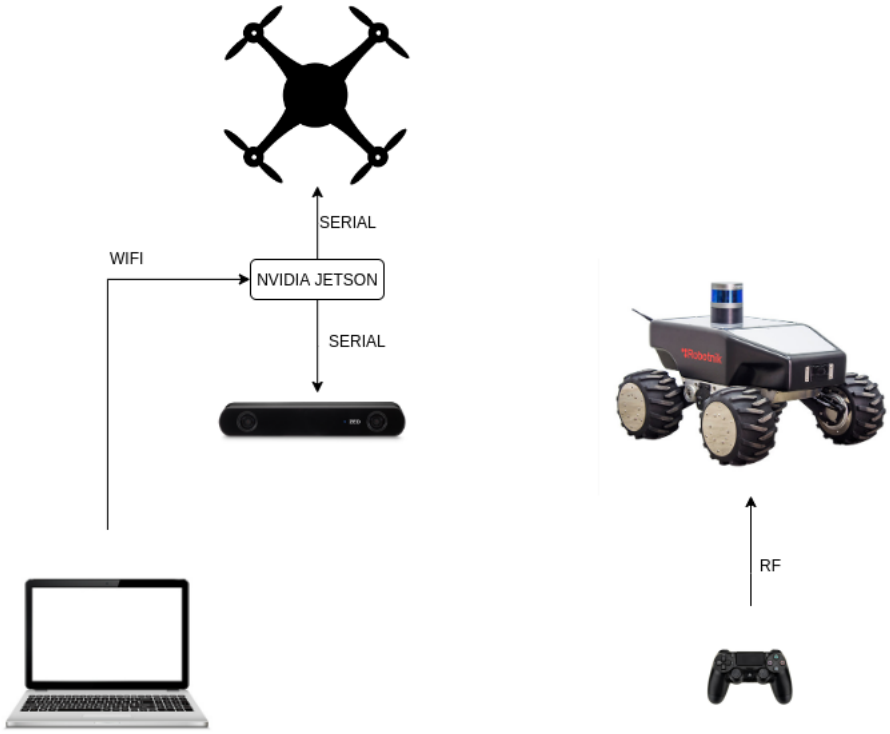
Διαστάσεις	720 x 614 x 416 mm
Βάρος	65kg
Ταχύτητα	3 m/s
Αυτονομία	10 ώρες
Μοτέρ	4 x 500 W brushless

3.1.5 Υπολογιστής - Σταθμός εδάφους

Τέλος για την εκκίνηση και έλεγχο των συστημάτων διαθέτουμε ένα υπολογιστή που λειτουργεί ως σταθμός ελέγχου και με τον οποίο συνδεόμαστε απομακρυσμένα με ssh (secure shell protocol) στον υπολογιστή του πολυκόπτερου.

3.2 Λογισμικό

Το λογισμικό (software) μπορεί να διαφοροποιηθεί ανάλογα με το περιβάλλον εκτέλεσης της εφαρμογής εννοώντας το πραγματικό κόσμο και την προσομοίωση.



Σχήμα 3.5: Διάγραμμα συνδέσεων.

3.2.1 Προσομοίωση

Στη προσομοίωση (simulation) όλες οι οντότητες που περιγράψαμε στα μηχανήματα (hardware) υλοποιούνται μέσω λογισμικού. Η μηχανή προσομοίωσης που χρησιμοποιούμε είναι το gazebo simulator και οι οντότητες-μοντέλα επικοινωνούν μεταξύ τους χρησιμοποιώντας το ROS (robot operating sustem) [22]. Πιο συγκεκριμένα, χρησιμοποιούμε ROS melodic, σε Ubuntu 18.04 με gazebo9.

3.2.2 Ενδιάμεσο λογισμικό ρομποτικής (ROS)

Το Robot Operating System (ROS) είναι ένα ενδιάμεσο λογισμικό ανοιχτού κώδικα, πολύ χρήσιμο για την υλοποίηση εφαρμογών ρομποτικής. Αυτό προσφέρει το πλαίσιο επικοινωνίας μεταξύ των διάφορων οντοτήτων τόσο στη προσομοίωση όσο και στην πραγματική εφαρμογή. Η αρχή λειτουργίας του βασίζεται σε θέματα (topics) και κόμβους (nodes), όπου οι κόμβοι (nodes) είναι

οι διάφορες οντότητες που εκδίδουν ή διαβάζουν μηνύματα από συγκεκριμένα θέματα (topics). Έτσι επιτρέπεται η συνεχής ενημέρωση τιμών ανάμεσα στα διάφορα τμήματα του κώδικα. Αυτό το σύστημα επιτρέπει πολλούς κόμβους (nodes) να διαβάζουν το ίδιο θέμα (topic) καθώς και να ενημερώνονται όταν υπάρχουν ανανεωμένες τιμές. Το ROS τρέχει στον υπολογιστή του πολυκόπτερου (onboard computer) δηλαδή στον nvidia xavier όταν έχουμε πραγματική εφαρμογή και στον υπολογιστή για την προσομοίωση.

3.2.3 Λογισμικό πτήσης (Ardupilot)

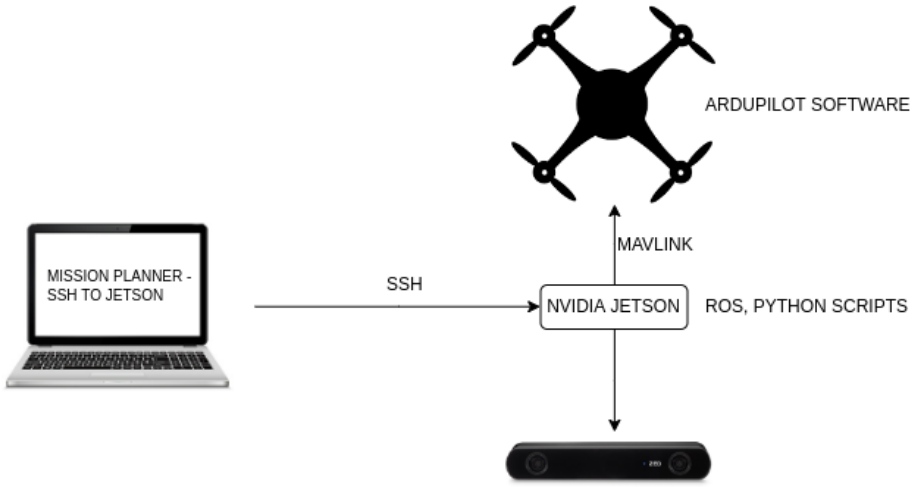
Το Ardupilot λογισμικό είναι ένα ανοιχτού κώδικα λογισμικό που προορίζεται για τον αυτόνομο έλεγχο διαφόρων οχημάτων εκ των οποίων και πολυκόπτερων όπως στην περίπτωση μας. Το λογισμικό τρέχει στον μικρουπολογιστή που φέρει το πολυκόπτερο μας και χρησιμοποιεί το σύνολο των αισθητήρων για τον έλεγχο της πτήσης. Η επικοινωνία με το ardupilot γίνεται με χρήση του πρωτόκολλου MAVLink που μας επιτρέπει να δίνουμε εντολές στον ελεγκτή εσωτερικού βρόχου (attitude controller) όπως επιθυμούμε σε αυτή την εφαρμογή καθώς και επιστρέφει τηλεμετρία για την εποπτία της πτήσης. Τέλος, το MAVROS είναι το πακέτο που μετατρέπει τα MAVLink μηνύματα σε ROS θέματα topics.

3.2.4 Λογισμικό σταθμού εδάφους (Mission Planner)

Όπως αναφέρθηκε παραπάνω το ardupilot επιστρέφει τηλεμετρία από το πολυκόπτερο. Τα δεδομένα αυτά είναι πολύ σημαντικά για την γενικότερη εποπτία και έλεγχο της πτήσης για αυτό και χρησιμοποιούμε το Missiion planner που είναι ένα λογισμικό που λαμβάνει και παρουσιάζει τις πληροφορίες αυτές με γραφικό τρόπο. Ταυτόχρονα μέσα από το λογισμικό αυτό μπορούν να ρυθμιστούν σημαντικές παράμετροι του πολυκόπτερου.

3.2.5 Python κώδικες για αναγνώριση

Στην εφαρμογή μας όπως έχει ήδη αναφερθεί κάνουμε χρήση νευρωνικών δικτύων για την εύρεση του ρομπότ - οδηγού στο εσωτερικό της εικόνας. Για την επίτευξη αυτού έχει γραφεί ένας python κώδικας (script) που διαβάζει το θέμα (topic) που έχει την εικόνα από την κάμερα και πάνω σε αυτή τρέχει το νευρωνικό δίκτυο. Έτσι λαμβάνει τη μάσκα που δημιουργείται από την κατάτμηση εικόνας (image segmentation) και χρησιμοποιώντας τη βιβλιοθήκη OpenCV [23] βρίσκει το πλαίσιο ελάχιστου εμβαδού που περιέχει τον οδηγό. Τέλος εκδίδει τις συντεταγμενες αυτού του πλαισίου στο θέμα (topic) με όνομα box ώστε να το λάβει ο ελεγκτής (controller). Το περιβάλλον στο οποίο εκτελείται



Σχήμα 3.6: Διάγραμμα λογισμικού και πρωτοκόλλων.

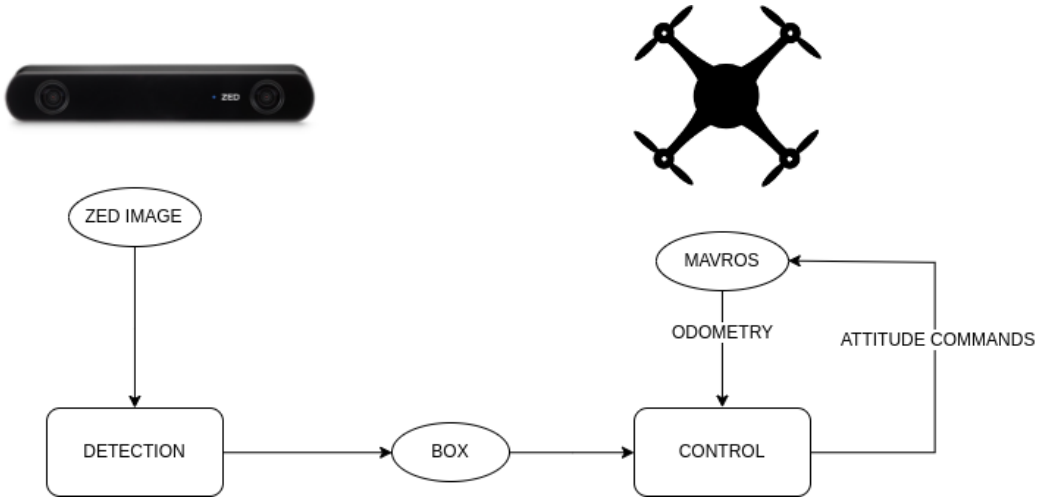
ο παραπάνω κώδικας (script) απαιτεί tensorflow 1 , python 3.6 και άλλα που περιγράφονται αναλυτικά στο αρχείο build.md .

3.2.6 Python κώδικες για έλεγχο

Για τον έλεγχο του πολυκόπτερου έχουμε επίσης ένα python κώδικα που εκτελείται στο ίδιο περιβάλλον με το κώδικα (script) της αναγνώρισης (detection). Ο συγκεκριμένος κώδικας (script) διαβάζει το θέμα (topic) του box και το θέμα (topic) του odometry του πολυκόπτερου (drone) και με βάση αυτό υπολογίζει τη κατάσταση (state) στο οποίο βρίσκεται. Έχοντας τη κατάσταση (state) και τον εκπαιδευμένο target actor υπολογίζουμε μία κίνηση (action) η οποία στέλνεται στο mavros θέμα (topic) η οποία και τελικά φτάνει στο ardupilot ως mavlink μήνυμα (message). Να σημειωθεί επίσης ότι ο κώδικας (script) αυτός χρησιμοποιεί μία κλάση που υλοποιεί την εικονική σταθεροποίηση της εικόνας (virtual gimbal) που συζητήσαμε στο προηγούμενο κεφάλαιο.

3.3 Μεθοδολογία

Στη διπλωματική αυτή υλοποιούμε πέρα από το βασικό σενάριο ακολούθησης του ρομπότ - οδηγού, και ένα πιο απλό σενάριο κατά το οποίο ακολουθούμε μια σταθερή γραμμή. Αυτό υπηρετεί ως εισαγωγικό και ενδιάμεσο στάδιο καθώς η μετάβαση από αυτό στο τελικό σενάριο καθίσταται πολύ πιο έυκολη.



Σχήμα 3.7: Διάγραμμα επικοινωνίας των θεμάτων (topics) .

3.3.1 Ακολούθηση σταθερού περιγράμματος (Static Contour)

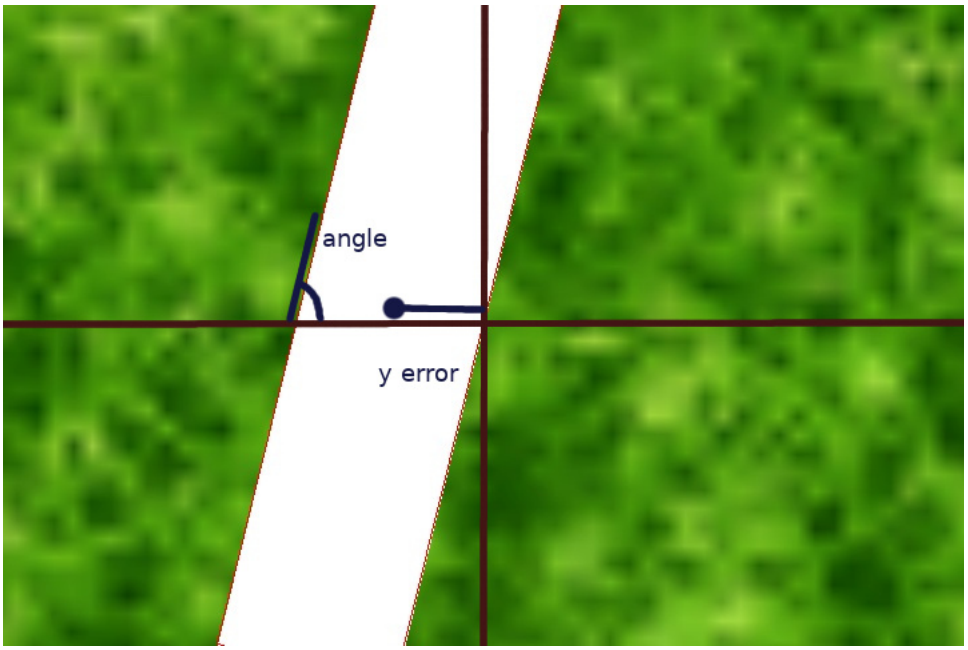
Στο σενάριο αυτό ακολουθούμε ένα πεζοδρόμιο με σταθερή ταχύτητα και σε σταθερό υψός. Οι πιο σημαντικές παράμετροι που πρέπει να οριστούν για την επιτυχή εκπαίδευση του πράκτορα ενισχυτικής μάθησης (Reinforcement Learning agent) είναι η κατάσταση (state), οι κινήσεις (actions) και η συνάρτηση ανταμοιβής (reward function). Ο κώδικας μας διαβάζει τις συντεταγμένες του πλαισίου (box) που λαμβάνει από τον αλγόριθμο αναγνώρισης (detector) και με βάση αυτές υπολογίζει την οριζόντια απόσταση του πεζοδρομίου από το κέντρο της εικόνας καθώς και την κλίση του ώς προς την κατακόρυφο. Αυτές οι δύο πληροφορίες κρίνονται απαραίτητες για την ευθυγράμμιση του πολυκόπτερου πάνω από το πεζοδρόμιο, για αυτό και αποτελούν μέρος του διανύσματος κατάστασης.

Πιο αναλυτικά το διάνυσμα κατάστασης φαίνεται παρακάτω:

$$state = [y\_distance, y\_velocity, angle, x\_velocity\_error] \tag{3.1}$$

όπου $x\_velocity\_error = x\_velocity\_desired - x\_velocity$.

Τα $y\_velocity$ και $x\_velocity\_error$ τα υπολογιζουμε από την οδομετρία που επιστρέφεται από το mavros. Το πρώτο είναι απαραίτητο στο διάνυσμα κατάστασης διότι χωρίς αυτό το πολυκόπτερο επιταχύνει προς την θέση ισορροπίας με αποτέλεσμα να έχουμε ταλαντώσεις γύρω από την επιθυμητή θέση. Δηλαδή



Σχήμα 3.8: Σφάλμα στην ακολούθηση πεζοδρομίου.

η γνώση της ταχύτητας στον άξονα y επιτρέπει στον αλγόριθμο να συγκλίνει καλύτερα σε συνδυασμό με το σωστό σχεδιασμό της συνάρτησης ανταμοιβής (reward function) που αναλύεται παρακάτω. Τέλος, το $x\_velocity\_error$ χρησιμοποιείται για να πετύχουμε τη σταθερή ταχύτητα κίνησης.

Όλες οι τιμές του state είναι κανονικοποιημένες στο διάστημα [-1,1] γεγονός που κρίνεται πολύ σημαντικό για την εκπαίδευση των νευρωνικών δικτύων.

Το διάνυσμα κινήσεων (action vector) αποτελείται από τις στροφές στους τρεις άξονες, δηλαδή:

$$action = [roll, pitch, \Delta yaw] \tag{3.2}$$

οπου $\Delta yaw = yaw\_old - yaw\_new$, η μεταβολή στο yaw.

Οι τιμές αυτές είναι περιορισμένες σε ένα μικρό φάσμα ώστε να μην έχουμε μεγάλες και απότομες κινήσεις. Στη περίπτωση μας έχουμε για roll και pitch το διάστημα [-3,+3] μοίρες και για το Δyaw [-5,+5] μοίρες.

Όσον αφορά την συνάρτηση ανταμοιβής (reward function) θέλουμε να δώσουμε πέναλτι στις καταστάσεις που θέλουμε να αποφύγουμε και άρα ελαχιστοποιώντας το πέναλτι να έχουμε την επιθυμητή συμπεριφορά.

Πιο αναλυτικά έχουμε:

$$position\_penatly = distance\_penalty + angle\_penalty \tag{3.3}$$

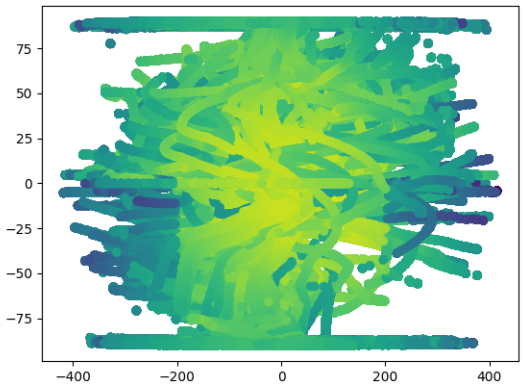
που σημαίνει ότι όσο πιο μεγάλη η οριζόντια απόσταση από το κέντρο της εικόνας και όσο πιο μεγάλη η κλίση του πεζοδρομίου ώς προς την κατακόρυφο της εικόνας τόσο μεγαλύτερο το πέναλτι.

$$velocity\_penalty = x\_velocity + y\_velocity \tag{3.4}$$

όπου επιβάλλεται ποινή σε υψηλές τιμές ταχύτητας ώστε να κινούμαστε πάνω από το πεζοδρόμιο με μηδενική ταχύτητα στον y άξονα και σταθερή ταχύτητα στον x άξονα.

$$action\_penalty = roll + pitch + yaw \tag{3.5}$$

όπου επιβάλλεται ποινή για μεγάλες κινήσεις καθώς θέλουμε πιο ομαλή κίνηση του πολυκόπτερου.



Σχήμα 3.9: Στο διάγραμμα αυτό φαίνεται η ανταμοιβή (reward) σε συνάρτηση με το σφάλμα στην απόσταση (x άξονας) και το σφάλμα στην κλίση του πεζοδρομίου (y άξονας). Όπως φαίνεται έχουμε μεγαλύτερη ανταμοιβή (reward) όσο πιο μικρά είναι τα σφάλματα.

Όλες οι τιμές που αναφέρονται στη συνάρτηση ανταμοιβής (reward function) είναι σε απόλυτη τιμή και κανονικοποιημένες στο [0,1].

Βάζοντας όλες τις ποινές μαζί έχουμε για την ανταμοιβή (reward) :

$$reward = -A * position\_penalty - B * velocity\_penalty - C * action\_penalty \tag{3.6}$$

$$reward = reward/350 \tag{3.7}$$

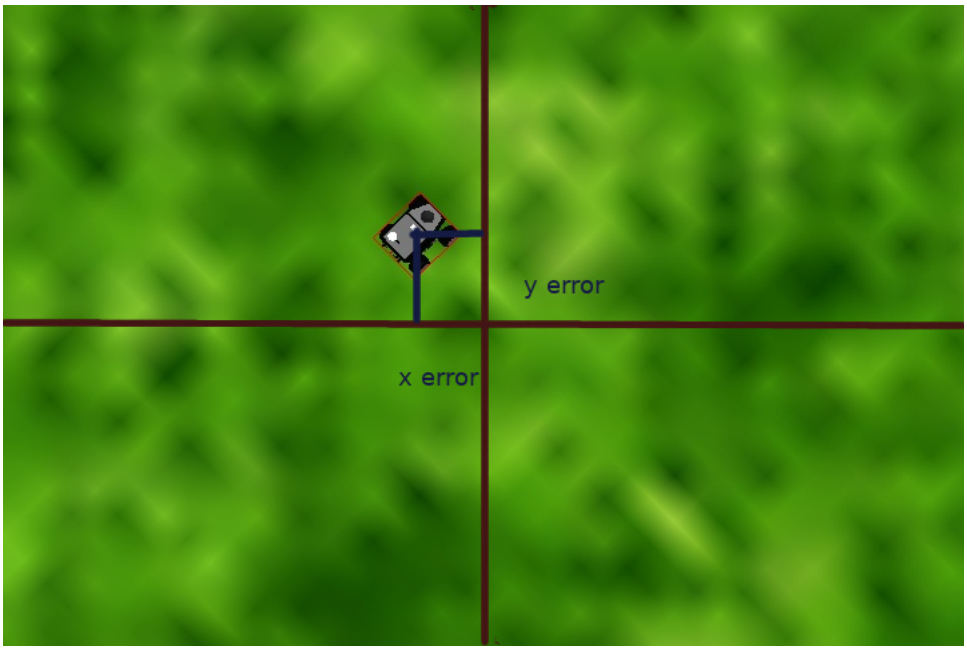
οπου A = 100, B = 60 , C=10 τιμές που έχουν επιλεχθεί εμπειρικά με βάση τη προτεραιότητα και τη βαρύτητα που έχει το κάθε πέναλτι στην εκπαίδευση της ενισχυτικής μάθησης.

Άλλες παράμετροι που χρειάστηκε να επιλεχθούν είναι ο αριθμός των νευρώνων και ο ρυθμός εκπαίδευσης για τον actor και τον critic. Οι τιμές που επιλέχθηκαν είναι οι εξής:
actor : 2 κρυφά επίπεδα νευρώνων (hidden layers) με 256 νευρώνες το καθένα και ρυθμό μάθησης (learning rate) 10^{-4} και συνάρτηση ενεργοποιήσης (activation function) tanh.
critic : 2 κρυφά επίπεδα νευρώνων (hidden layers) με 256 νευρώνες το καθένα και ρυθμό μάθησης (learning rate) 10^{-3} και συνάρτηση ενεργοποιήσης (activation function) ReLU.
Τέλος η τ (tau) παράμετρος της ενημέρωσης των target networks επιλέχθηκε ίση με 0.01.

3.3.2 Ακολούθηση κινούμενου οδηγού (Target Following)

Έχοντας υλοποιήσει το σενάριο με την ακολούθηση σταθερής επιφάνειας, η ακολούθηση κινούμενου στόχου είναι πιο άμεση. Σε αυτή τη περίπτωση ο κώδικας μας λαμβάνει τις συντεταγμένες του πλαισίου και από αυτές υπολογίζει την απόσταση του κέντρου του πλαισίου από το κέντρο της εικόνας τόσο στον x όσο και στον y άξονα. Επίσης υπολογίζει την κλίση του πλαισίου ώστε να γνωρίζουμε τον προσανατολισμό του ρομπότ-οδηγού και να μπορούμε να ευθυγραμμιστουμε με αυτόν. Αυτές οι πληροφορίες είναι πολύ σημαντικές για να μπορούμε να κρατήσουμε τον οδηγό στο κέντρο της εικόνας.

Όπως και στο προηγούμενο σενάριο έτσι και εδώ δημιουργείται το πρόβλημα των ταλαντώσεων γύρω από την θέση ισορροπίας. Για να το εξαλείψουμε εισάγουμε επιπλεον τη μεταβολή της απόστασης του box από το κέντρο της εικόνας (παράγωγος του σφάλματος) το οποίο μας δίνει μια τιμή που σχετίζεται με την ταχύτητα το ρομποτ-οδηγού που δεν έχουμε πιο άμεσο τρόπο να την προσεγγίσουμε. Αυτή η τιμή σε συνδυασμό με τις ταχύτητες του πολυκόπτερου στους δύο άξονες του επιπέδου μας δίνουν μια πιο ολοκληρωμένη εικόνα για την σχετική κίνηση των δύο οχημάτων.



Σχήμα 3.10: Σφάλμα στην ακολούθηση του κινούμενου οδηγού.

Λαμβάνοντας όλα τα παραπάνω υπόψη το διάνυσμα κατάστασης ειναι το ακόλουθο:

$$state = [distance\_x, distance\_y, angle, deriv\_dist\_x, deriv\_dist\_y, velocity\_x, velocity\_y] \tag{3.8}$$

Όλες οι τιμές της κατάστασης (state) είναι επίσης κανονικοποιημένες στο διάστημα [-1,1] .

Το διάνυσμα κινήσεων (action vector) αποτελείται ομοίως από τις στροφές στους τρεις άξονες, δηλαδή:

$$action = [roll, pitch, \Delta yaw] \tag{3.9}$$

οι οποίες όπως και προηγουμένως είναι περιορισμένες σε μικρό σύνολο τιμών.

Όσον αφορά την συνάρτηση ανταμοιβής (reward fuction) ακολουθούμε την ίδια νοοτροπία με προηγουμένως. Πιο αναλυτικά έχουμε:

$$position\_penatly = distance\_penalty + 0.5 * angle\_penalty \tag{3.10}$$

$$derivative\_penalty = deriv\_dist\_x + deriv\_dist\_y \tag{3.11}$$

$$action\_penalty = roll + pitch + yaw \tag{3.12}$$

Παρομοίως ολες οι τιμές που αναφέρονται στην συνάρτηση ανταμοιβής (reward function) είναι σε απόλυτη τιμή και κανονικοποιημένες στο [0,1].

Συνολικά έχουμε:

$$reward = -A*position\_penalty - B*derivative\_penalty - C*action\_penalty \tag{3.13}$$

$$reward = reward/340 \tag{3.14}$$

οπου A = 100, B = 30 , C=10 που επιλέχθηκαν εμπειρικά.
Οι παράμετροι που αφορούν την εκπαίδευση των νευρωνικών δικτύων παραμένουν οι ίδιες.

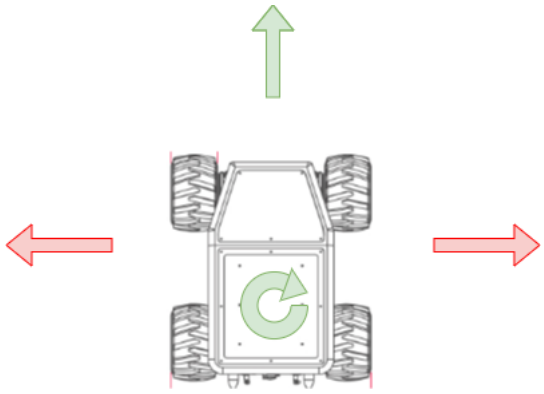
Το παραπάνω σενάριο ακολούθησης είναι λειτουργικό όταν η πληροφορία των συντεταγμένων του πλαισίου (box) είναι εύρωστη (robust) και αξιόπιστη. Αυτό είναι απαραίτητο ώστε η μακρύτερη πλευρά του box να ταυτίζεται με τη μακρύτερη διάσταση του ρομπότ-οδηγού και άρα να έχουμε έγκυρη πληροφορία σχετικά με τον προσανατολισμό του οδηγού στην εικόνα που λαμβάνουμε. Εφόσον αυτό συχνά δεν είναι δυνατό σε πραγματικές συνθήκες, αναπτύχθηκε το παρακάτω σέναριο που δεν κάνει χρήση της κλίσης του πλαισίου (box) αλλά παρόλα αυτά δεν αποκλίνει σημαντικά από τον προσανατολισμό του οδηγού.

Το σενάριο αυτό εξάγει από τις συντεταγμένες του πλαισίου (box) μόνο το κέντρο του και υπολογίζει την απόσταση του από το κέντρο της εικόνας. Το διάνυσμα κατάστασης (state) συνεπώς είναι το ίδιο με προηγουμένως με τη μόνη διαφορά ότι δεν έχουμε την κλίση (angle). Η σημαντική διαφορά είναι ότι θέλουμε η ταχύτητα στον y άξονα να είναι μηδενική και άρα συνεχώς το πολυκόπτερο να κινείται προς το μπροστά στρίβωντας με το yaw για να αλλάξει κατεύθυνση. Το ρομπότ-οδηγός κινείται πάντα προς τη κατεύθυνση στην οποία προσανατολίζεται, αφού δεν μπορεί να κινηθεί πλαγίως και θεωρούμε ότι δεν πηγαίνει και όπισθεν. Έτσι αν το κέντρο του ρομπότ απέκλινε προς μια κατεύθυνση από το κέντρο της εικόνας οδηγεί στο συμπέρασμα ότι το ρομπότ προσανατολίστηκε και το ίδιο προς εκείνη την κατεύθυνση. Και βάζοντας το πολυκόπτερο να ακολουθεί το σφάλμα (error) χωρίς κίνηση στον y άξονα καταφέρνουμε να ακολουθεί προσανατολισμένο με το ρομπότ-οδηγό.

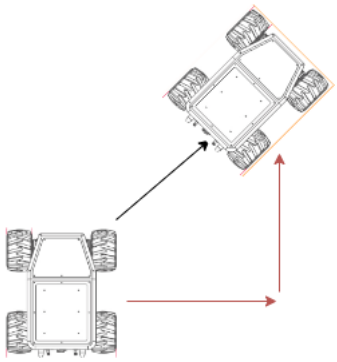
Σημαντικό για το σενάριο αυτό είναι η συνάρτηση ανταμοιβής (reward function) που ορίζεται όπως παραπάνω και επιπλέον έχει τον παρακάτω όρο:

$$velocity\_penalty = velocity\_y + clip(velocity\_x, 0, -1) \tag{3.15}$$

που δίνει ποινή για μη μηδενικές ταχύτητες στον y άξονα και για αρνητικές ταχύτητες στον άξονα x για να μην πηγαίνουμε όπισθεν.



Σχήμα 3.11: Επιτρεπτές κινήσεις του Summit. Με πράσινο οι δυνατές κινήσεις ενώ με κόκκινο οι μη.



Σχήμα 3.12: Δυνατή τροχιά κίνησης του Summit. (Με κόκκινο η μη δυνατή τροχιά)

3.4 Εκπαίδευση στη προσομοίωση

Για την εκπαίδευση των πρακτόρων (agent) και την καλύτερη γενίκευση χρησιμοποιήθηκαν διάφορες τεχνικές που κρίνεται χρήσιμο να αναφερθούν. Η εκπαίδευση γίνεται σε ένα σύνολο επεισοδίων τα οποία ολοκληρώνονται είτε όταν το ρομπότ οδηγός βρεθεί εκτός της εικόνας είτε ολοκληρωθεί ένας αριθμός βημάτων (timesteps) που στη περίπτωση μας έχουμε επιλέξει να είναι 1024. Όταν συμβεί ένα από αυτά τα δύο το πολυκόπτερο οδηγείται και πάλι πάνω από το ρομπότ οδηγό. Καθώς είμαστε σε περιβάλλον προσομοίωσης έχουμε τόσο την πληροφορία για τις συντεταγμένες του ρομπότ οδηγού όσο και μπορούμε να οδηγήσουμε το πολυκόπτερο ακριβώς πάνω από αυτό με χρήση

mavros εντολών (commands). Να σημειωθεί ότι ο κώδικας (script) που κινεί το robot οδηγό στην προσομοίωση παρέχει μια ψευδοτυχαία τροχιά και ενημερώνεται μέσω του ROS θέματος (topic) box για το αν έχει χαθεί το ρομπότ από την εικόνα ώστε να σταματήσει και να περιμένει την επανεύρεση. Χρήσιμο για την καλύτερη γενίκευση αποδείχθηκε η χρήση ομοιόμορφου θορύβου στην πληροφορία της θέσης του ρομπότ οδηγού κατά την επανεύρεση. Αυτό διότι ο πράκτορας (agent) μαθαίνει να ακολουθεί το στόχο ακόμα και όταν στην αρχική κατάσταση ο οδηγός δεν είναι στο κέντρο της εικόνας, γεγονός που συμβαίνει στις πραγματικές συνθήκες.

Όσον αφορά την αποθήκευση των βαρών και δεδομένων εκπαίδευσης, στο τέλος κάθε βήματος (timestep) αποθηκεύουμε σε αρχείο τα σφάλματα (error) και τις κινήσεις (action), ενώ στο τέλος κάθε επεισοδίου τις μέσες τιμές ανταμοιβών (average reward). Αυτό είναι απαραίτητο για τον έλεγχο και την οπτικοποίηση της πορείας της εκπαίδευσης. Επίσης όταν η μέση τιμή ανταμοιβής (average reward) είναι καλύτερη από την προηγουμένως καλύτερη τότε αποθηκεύουμε τα βάρη των νευρωνικών. Ανά 200 επεισόδια κρατάμε τα καλύτερα βάρη από αυτά και τέλος ανά δέκα επεισόδια ενημερώνουμε τις γραφικές παραστάσεις που δείχνουν την πορεία της εκπαίδευσης.

Για το εισαγωγικό πρόβλημα του σταθερού περιγράμματος (static contour) οι διαδικασίες που ακολουθούμε είναι παρόμοιες με τη μόνη διαφορά ότι όταν χάνουμε το πεζοδρόμιο επιστρέφουμε στην τελευταία θέση στην οποία αυτό βρισκόταν στο κέντρο της εικόνας προσθέτοντας και θόρυβο όπως αναφέρθηκε παραπάνω.

Κεφάλαιο 4

Αποτελέσματα

Στο παρόν κεφάλαιο παρουσιάζουμε τα αποτελέσματα της εκπαίδευσης του πράκτορα ενισχυτικής μάθησης (Reinforcement Learning agent) στα σενάρια που παρουσιάστηκαν αναλυτικά παραπάνω καθώς και την επίδοση των εκπαιδευμένων actors τόσο σε προσομοιωμένες (simulated) όσο και πραγματικές συνθήκες.

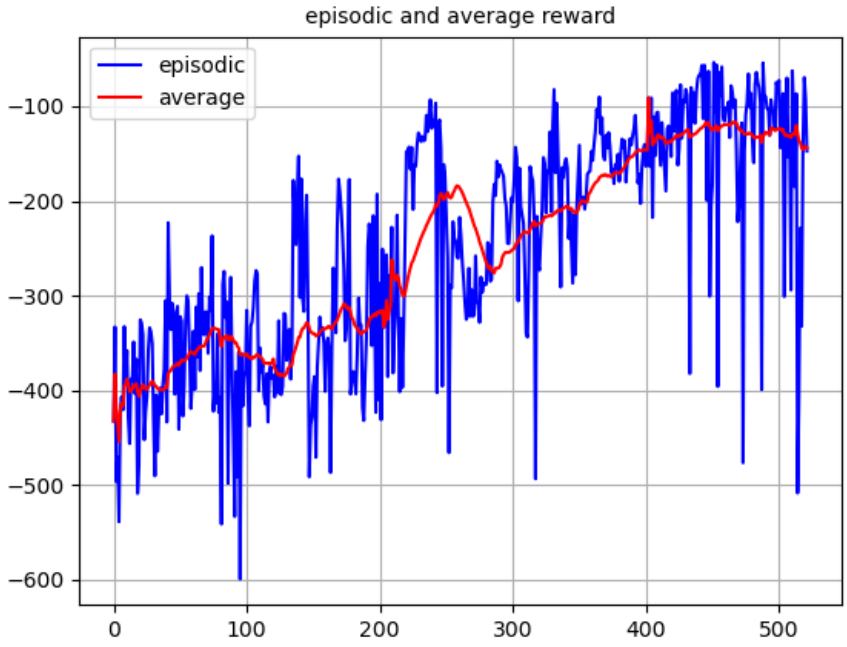
4.1 Προσομοίωση - Εκπαίδευση

Η ενισχυτική μάθηση (Reinforcement Learning) καθώς είναι μία μεθοδολογία Βαθιάς Μάθησης (Deep Learning) χρειάζεται ένα πλήθος επεισοδίων για εκπαίδευση ώστε τα βαθιά νευρωνικά δίκτυα της να συγκλίνουν. Έτσι και στη περίπτωση μας χρειάστηκαν πολλά επειδόσια τα οποία πραγματοποιηθήκαν στο περιβάλλον προσομοίωσης. Πιο αναλυτικά παρουσιάζονται στη συνέχεια.

4.1.1 Σταθερό περίγραμμα (Static Contour)

Για το σενάριο ακολούθησης της σταθερής επιφάνειας χρειάστηκαν λιγότερο από 600 επισόδεια για την εκπαίδευση του πράκτορα (agent) ώστε να πάρουμε τη μέγιστη δυνατή ανταμοιβή (reward) αποφεύγοντας ταυτόχρονα την υπερεκπαίδευση (overfitting).

Στο σχήμα 4.1 βλέπουμε την μεταβολή της επεισοδιακής (episodic) και μέσης (average) ανταμοιβής (reward) κατά την εκπαίδευση του πράκτορα (agent) και πως αυτή σταθεροποιείται γύρω από μία μέγιστη τιμή υποδεικνύοντας έτσι την επιτυχία της εκπαίδευσης. Παρατηρούμε ότι οι τιμές της ανταμοιβής (reward) είναι αρνητικές καθώς ο τρόπος που έχουμε σχεδιάσει τη συνάρτηση ανταμοιβής (reward function) είναι ώς το αρνητικό του πέλαντι. Ουσιαστικά θέλουμε



Σχήμα 4.1: DDPG επεισοδιακή και μέση τιμή ανταμοιβής (episodic and average reward) για το σταθερό περίγραμμα (static contour).

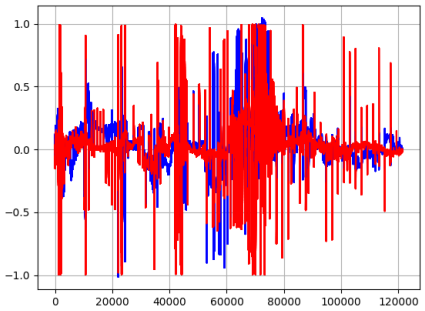
να ελαχιστοποιήσουμε το πέλαντι πλησιάζοντας έτσι το 0 που είναι η μέγιστη τιμή της ανταμοιβής (reward).

Παρακάτω στο σχήμα 4.2 φαίνεται και η μεταβολή του σφάλματος απόστασης και κλίσης (distance, angle error) κατα την πρόοδο της εκπαίδευσης, όπου και παρατηρούμε ότι μειώνονται σημαντικά προς το τέλος.

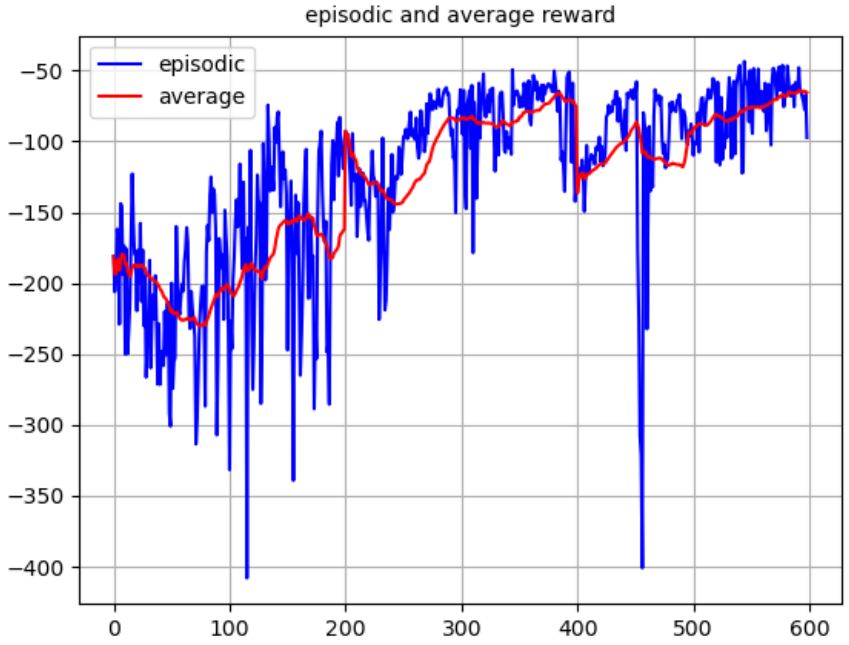
4.1.2 Κινούμενος οδηγός (Moving Target Following)

Για το σενάριο ακολούθησης του κινούμενου στόχου η εκπαίδευση του πράκτορα (agent) έλαβε επίσης 600 επεισόδια εώς ότου συνέκλινε σε μία μέγιστη τιμή. Πιο αναλυτικά η εξέλιξη της εκπαίδευσης φαίνεται στο σχήμα 4.3.

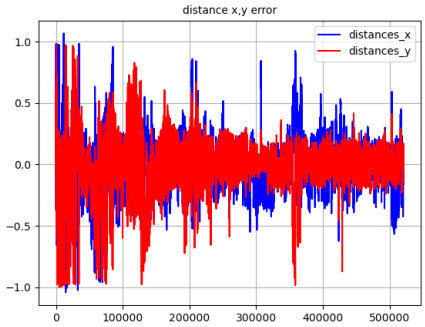
Όπως και για την περίπτωση του σταθερού περιγράμματος (static contour) έτσι και εδώ παρουσιάζουμε στο σχήμα 4.4 το σφάλμα στους δύο άξονες δηλαδή την απόσταση του κινούμενου στόχου από το κέντρο της εικόνας στο x και y άξονα κατά την πρόοδο της εκπαίδευσης.



Σχήμα 4.2: Σφάλμα απόστασης και κλίσης προς τα τελευταία επισόδεια.



Σχήμα 4.3: DDPG επεισοδιακή και μέση τιμή ανταμοιβής (episodic and average reward) για τον κινούμενο οδηγό (moving target following).



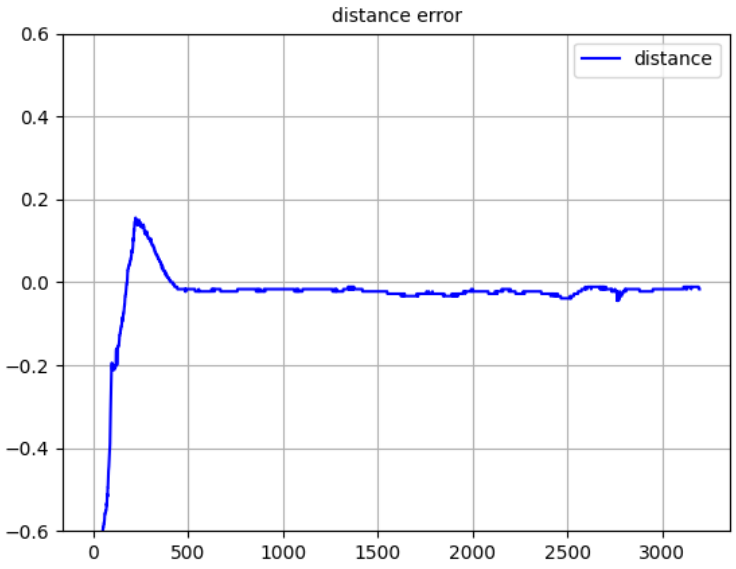
Σχήμα 4.4: Σφάλμα απόστασης στον x και y άξονα κατά τη εξέλιξη των επεισοδίων.

4.2 Προσομοίωση - Δοκιμή

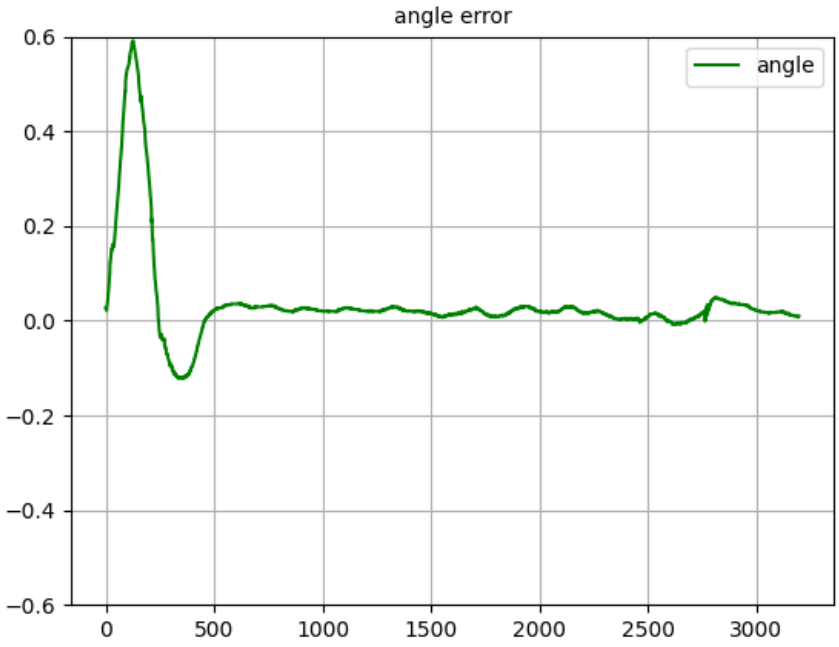
Έχοντας εκπαιδεύσει και κρατήσει τα καλύτερα βάρη για τους actors εκτελούμε την εφαρμογή του νευρωνικού (inference) δηλαδή την εφαρμογή του πράκτορα (agent) και παρακολούθηση της επίδοσης του στο εκάστοτε σενάριο.

4.2.1 Σταθερό περίγραμμα

Στο σενάριο ακολούθησης του πεζοδρομίου στο περιβάλλον προσομοίωσης παρατηρούμε ότι ο πράκτορας (agent) έχει μάθει μία επιτυχημένη στρατηγική η οποία καταφέρνει να κρατάει το πεζοδρόμιο στο κέντρο της εικόνας και ευθυγραμμισμένο με αυτή. Μάλιστα αυτό το καταφέρνει διατηρώντας το σφάλμα σε πολύ μικρές τιμές της τάξης του 5% . Το αρχικό σφάλμα που φαίνεται στα διαγράμματα δόθηκε επίτηδες στις αρχικές συνθήκες ώστε ο πράκτορας (agent) να μάθει να ακολουθεί το πεζοδρόμιο ακόμα και όταν η αρχικοποίηση είναι με μεγάλο σφάλμα.



Σχήμα 4.5: Σφάλμα απόστασης στην εφαρμογή για σταθερό περίγραμμα (Inference distance error).

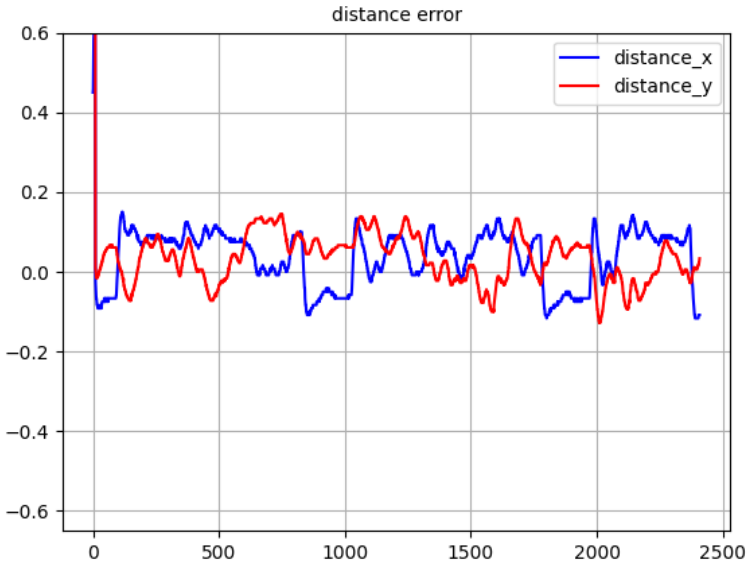


Σχήμα 4.6: Σφάλμα γωνίας στην εφαρμογή για σταθερό περίγραμμα (Inference angle error).

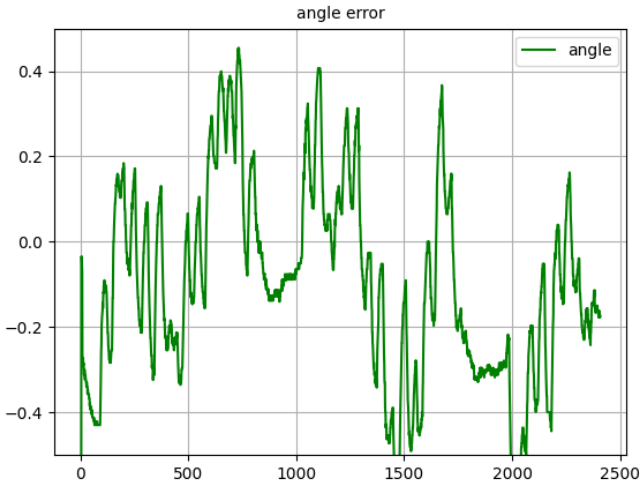
4.2.2 Κινούμενος οδηγός

Αντίστοιχα και για το σενάριο ακολούθησης του κινούμενου στόχου έχουμε μία επιτυχημένη πολιτική (policy) που καταφέρνει να διατηρεί το Summit πάντα εντός της εικόνας και κρατώντας κατά το δυνατό κοινό προσανατολισμό με αυτό. Το σφάλμα της θέσης του Summit είναι μικρό της τάξης του 10% ενώ το σφάλμα της γωνίας φτάνει εώς και 40% λόγω της διαφοράς φάσης που υπάρχει μέχρι να γίνει εμφανής η κατεύθυνση κίνησης του Summit.

Να σημειωθεί ότι και στα δύο προσομοιούμενα σενάρια το ύψος διατηρήθηκε σταθερό μέσω των εντολών που δίνονται αυτόματα και ορίζουν το τρόπο λειτουργίας (mode) πτήσης του πολυκόπτερου.



Σχήμα 4.7: Σφάλμα απόστασης στην εφαρμογή για ακολούθηση κινούμενου οδηγού (Inference distance error).



Σχήμα 4.8: Σφάλμα γωνίας στην εφαρμογή για την ακολούθηση του κινούμενου οδηγού (Inference angle error).

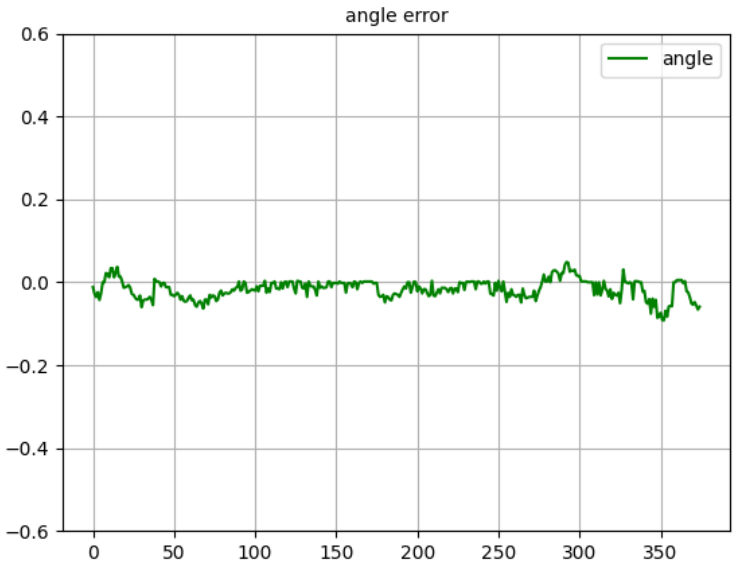
4.3 Πειραματικά αποτελέσματα

Η επιτυχής πραγματοποίηση των σενάριων αυτών στη προσομοίωση αποτελεί απαραίτητη και καλή ένδειξη για την μετάβαση στο πραγματικό κόσμο. Έτσι σε αυτό το υποκεφάλαιο παρουσιάζονται τα πειραματικά αποτελέσματα της διπλωματικής αυτής εργασίας.

4.3.1 Σταθερό περίγραμμα

Το πείραμα ακολούθησης του πεζοδρομίου πραγματοποιήθηκε σε πραγματικές συνθήκες με συμπεριφορά παρόμοια με την προσδοκόμενη της προσομοίωσης. Το σφάλμα λαμβάνει λίγο μεγαλύτερες τιμές λόγω του θορύβου που εισέρχεται λόγω των πραγματικών συνθηκών. Η κύρια πηγή θορύβου όπως αποδείχθηκε είναι στην ανίχνευση του πεζοδρομίου από το νευρωνικό δίκτυο κατάτμησης εικόνας (Image Segmentation Neural Network) όπου δεν είναι απόλυτα ακριβές το σύνολο των σημείων που ανήκουν στο πεζοδρόμιο. Πέρα από αυτό ο ελεγκτής (controller) συμπεριφέρεται όπως θα περιμέναμε επιβεβαιώνοντας ότι η μετάβαση από προσομοίωση σε πραγματικές συνθήκες μπορεί να είναι άμεση.

Σχήμα 4.9: Σφάλμα απόστασης στο πραγματικό πείραμα.

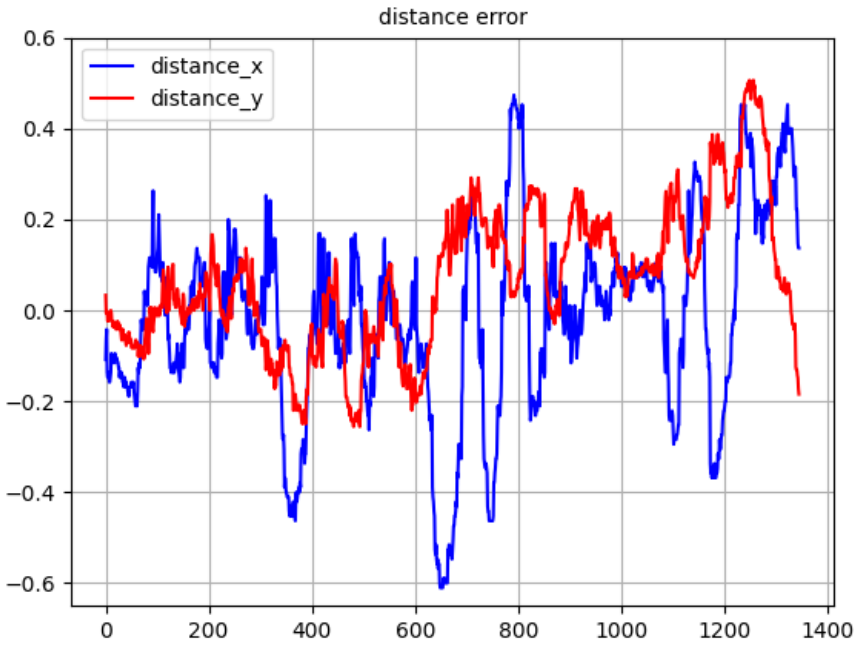


Σχήμα 4.10: Σφάλμα κλίσης στο πραγματικό πείραμα.

Σχήμα 4.11: Παράδειγμα ανίχνευσης πεζοδρομίου.

4.3.2 Κινούμενος οδηγός

Το πείραμα ακολούθησης του Summit πραγματοποιήθηκε και αυτό σε πραγματικές συνθήκες με χειροκίνητο χειρισμό του Summit. Το σφάλμα (error) στο πείραμα αυτό λαμβάνει μεγαλύτερες τιμές αλλά χωρίς να χάνει ποτέ το ρομπότ οδηγό από την εικόνα. Ο λόγος που έχουμε μεγαλύτερο σφάλμα σε αυτή την εφαρμογή είναι οι πιο απότομες αλλαγές στην πορεία και την ταχύτητα του Summit σε σχέση με την πιο ομαλή οδήγηση του στην προσομοίωση. Η ανίχνευση (Detection) ήταν επιτυχημένη καθώς έβρισκε τον οδηγό συνεχώς χωρίς ψευδώς θετικά (false positives) ή αληθώς αρνητικά (true negatives). Όπως συζητήθηκε και στο κεφάλαιο της μεθοδολογίας το σύνολο σημείων (pointcloud) που δημιουργείται από την ανίχνευση (detection) δεν επαρκεί για να προσδιορίσουμε το προσσανατολισμό του οδηγού και άρα κινούμαστε με τη στρατηγική που περιγράφηκε. Το σφάλμα στο προσανατολισμό στο πραγματικό πείραμα είναι αντίστοιχο με εκείνο της προσομοίωσης αλλά δεν υπάρχει τρόπος να το παρουσιάσουμε σε διάγραμμα καθώς δεν μπορούμε να γνωρίζουμε το πραγματικό προσαναλοτισμό του Summit με βάση την ανίχνευση (detection).



Σχήμα 4.12: Σφάλμα απόστασης στο πραγματικό πείραμα.

Σχήμα 4.13: Παράδειγμα ανίχνευσης Summit.

Κεφάλαιο 5

Συμπεράσματα και μελλοντικές προοπτικές

5.1 Συμπέρασματα - Συζήτηση

Σε αυτή την εργασία πραγματοποιήθηκε ένα σχήμα ελέγχου με χρήση ενισχυτικής μάθησης. Η ενισχυτική μάθηση δίνει νέες δυνατότητες ελέγχου οι οποίες προκύπτουν από τα διαφορετικά σενάρια στα οποία εκπαιδεύεται ο ελεγκτής (controller) σε αντίθεση με ένα σταθερό κανόνα ελέγχου όπως στην περίπτωση του κλασικού ελέγχου οπτικής ανατροφοδότησης. Η εφαρμογή αυτή δείχνει τη δυνατότητα της ενισχυτικής μάθησης να δημιουργεί στρατηγικές οι οποίες σε σύνθετα σενάρια δεν θα ήταν προφανείς επιβεβαιώνοντας την χρησιμότητα της σε πλήθος εφαρμογών. Στη περίπτωση μας η ενισχυτική μάθηση αξιοποιήθηκε με σκοπό την καλύτερη αποκρισιμότητα στις αλλαγές της ταχύτητας και της κατεύθυνσης του ρομπότ-οδηγού καθώς επιτυγχάνει καλύτερη πρόβλεψη της ταχύτητας του ρομπότ οδηγού σε σχέση με το κλασικό σχήμα ελέγχου ώστε να ακολουθήσει τον οδηγό. Αυτό το εγχείρημα επιτεύχθηκε σε σημαντικό βαθμό όπως υποδεικνύουν τα πειραματικά αποτελέσματα. Το ρομπότ οδηγός μένει εντός της εικόνας της κάμερας του πολυκόπτερου και κατ΄ επέκταση το πολυκόπτερο ακολουθεί το Summit.

Το παραπάνω σχήμα ακολούθησης σίγουρα έχει περιθώρια βελτίωσης και κυρίως όσον αφορά την εκπαίδευση σε σενάρια με πιο απότομες αλλαγές ταχύτητας και κατεύθυνσης. Πιο αναλυτικά, ο χειρισμός του ρομπότ-οδηγού στη προσομοίωση ήταν μεν ψευδοτυχαία αλλά παρέμενε σε ένα βαθμό ομαλή ως προς το εύρος των αλλαγών, ειδικά σε σύγκριση με τον χειροκίνητο τηλεχειρισμό στο πραγματικό πείραμα. Έτσι έχοντας εκπαιδεύσει τον ελεγκτή (controller) σε πιο ακραίες συνθήκες θα είχαμε έναν πιο αποκρίσιμο ελεγκτή

(controller) και περαιτέρω μικρότερη διακύμανση του σφάλματος.

Οι υπόλοιπες αδυναμίες της εφαρμογής αυτής αφορούν κυρίως το κομμάτι της αναγνώρισης του οδηγού. Αρχικά, η έλλειψη της πληροφορίας για το προσανατολισμό του οδηγού μας έθεσε το περιορισμό να μην μπορούμε να προσανατολιστούμε επακριβώς με βάση αυτό αλλά να αρκεστούμε στην πληροφορίας της κατεύθυνσης της μετατόπισης του. Αυτό δημιούργησε ένα σημαντικό σφάλμα στην κατεύθυνση αλλά δεν επηρεάσε την ακολούθηση του οδηγού. Μία προσέγγιση λύσης του παραπάνω προβλήματος θα ήταν η εκπαίδευση της κατάτμησης εικόνας με κάποιο καλύτερο λογισμικό (framework) που θα επέστρεφε πιο ακριβές σύνολο σημείων (pointcloud) και άρα από αυτό θα μπορούσαμε να εξάγουμε την κατεύθυνση του Summit. Μία διαφορετική προσέγγιση είναι η δημιουργία ενός προσαρμοσμένου (custom) νευρωνικού δικτύου το οποίο θα βασίζεται και πάλι στο MobileNets αλλά θα έχει στην έξοδο του δύο νευρώνες οι οποίες θα δίνουν τις συντεταγμένες δύο χαρακτηριστικών σημείων (landmarks) πάνω στο Summit και έτσι ενώνοντας τα δύο αυτά σημεία θα έχουμε το προσανατολισμό του Summit.

Επίσης, παρατηρήθηκε αδυναμία του νευρωνικού δικτύου της αναγνώρισης, να ανιχνεύσει ορισμένα σημεία του πεζοδρομίου τα οποία λόγω σκίασης είχαν φαινομενικά διαφορετικό χρώμα. Αυτό θα μπορούσε να βελτιωθεί με διεύρυνση του συνόλου δεδομένων (dataset) εκπαίδευσης για το πεζοδρόμιο ώστε να αναγνωρίζει το πεζοδρόμιο σε διαφορετικές συνθήκες φωτισμού.

5.2 Μελλοντικές προοπτικές

Η παρούσα εφαρμογή έχει πολλές δυνατότητες εξέλιξης και βελτίωσης. Πέρα από τις αδυναμίες που αναφέρθηκαν παραπάνω υπάρχουν και χρήσιμες επεκτάσεις που δίνουν νέες δυνατότητες στο σχήμα ακολούθησης. Αρχικά, χρήσιμη θα ήταν η δυνατότητα αποφυγής εμποδίων του πολυκόπτερου με χρήση αισθητήρων απόστασης και επαναπροσδιορισμός του μονοματιού ακολούθησης με χρήση της πληροφορίας της απόστασης από το εμπόδιο. Επιπλέον, μία σημαντική επέκταση του παρόντος σχήματος είναι ο ελεγκτής (controller) να μπορεί να κάνει πρόβλεψη τροχιάς (trajectory estimation) του οδηγού και με βάση αυτό να ξέρει που να κινηθεί σε περίπτωση που το ρομπότ οδηγός βρεθεί εκτός της εικόνας ή κάποιο αντικείμενο το καλύψει, με σκοπό την επανεύρεση του. Τέλος, μία ακόμα εξέλιξη θα ήταν η επιπλεον εκπαίδευση (fine-tuning) ενός ήδη εκπαιδευμένου δικτύου (pre-trained network) όπως το YOLO έχοντας έτσι τη δυνατότητα να αναγνωρίσουμε ένα μεγαλύτερο σύνολο κλάσεων και επιλέγοντας από αυτές να διαλέγουμε τον οδηγό που θα ακολουθήσουμε.

Κεφάλαιο 6

Παράρτημα

6.1 Εκκίνηση ρουτινών

Η πραγματοποίηση των πειραμάτων και της προσομοίωσης απαιτεί ένα σύνολο εντολών για να εκκινηθεί. Αυτές οι οδηγίες παρουσιάζονται αναλυτικά στο github στο αρχείο run.md

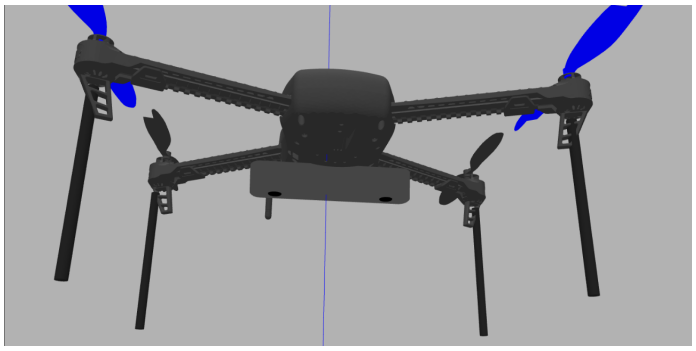
6.2 Εκπαίδευση της κατάτμησης εικόνας

Η κατάτμηση εικόνας (Image Segmentation) που χρησιμοποιούμε για την εύρεση ενός αντικειμένου στην εικόνα (detection) χρειάζεται μία διαδικασία για να την εκπαιδεύσουμε. Αυτή η διαδικασία παρουσιάζεται σε αυτό το υποκεφάλαιο. Το λογισμικό (framework) το οποίο αξιοποιούμε είναι το image-segmentation keras του οποίου το repository μπορεί να βρεθεί εδώ [24]. Για να ξεκινήσουμε τη διαδικασία εκπαίδευσης του νευρωνικού δικτύου χρειάζεται να δημιουργήσουμε ένα σύνολο δεδομένων (dataset) στο οποίο θα ορίσουμε και τις πραγματικές ετικέτες (labels). Για το σκοπό αυτό καταγράφουμε ένα βίντεο που περιλαμβάνει τόσο το αντικείμενο που μας ενδιαφέρει όσο και ένα σύνολο από διαφορετικά τοπία. Στη περίπτωση μας αυτό το αντικείμενο είναι το Summit και το βίντεο καταγράφηκε σε ένα χωμάτινο τοπίο με κάποια βλάστηση σε ορισμένα σημεία. Έπειτα από το βίντεο εξάγουμε στιγμιότυπα (frames) τα οποία θα χρησιμοποιήσουμε για την εκπαίδευση. Σε αυτά τα στιγμιότυπα (frames) πρέπει να ορίσουμε τις μάσκες οι οποίες δηλώνουν που βρίσκεται το Summit στην εικόνα. Αυτές οι μάσκες δημιουργούνται με τη χρήση εργαλείων όπως το labelme και χρησιμεύουν ώς ετικέτες (labels) για την εκπαίδευση. Στη συνέχεια ακολουθούμε κάποιες διαδικασίες οι οποίες διευκολύνουν τη καλύτερη εκπαίδευση και που οδηγεί σε καλύτερη γενίκευση. Αυτές είναι η μείωση των

διαστάσεων των εικόνων και η επαύξηση εικόνων (image augmentation) με χρήση πολλών τεχνικών όπως αναστροφή, περιστροφή, αλλαγή χρωμάτων και αντίθεσης. Τέλος ορίζουμε το πλήθος των κλάσεων δηλαδή στη περίπτωση μας δύο (το τοπίο και το Summit) και εκπαιδεύουμε για 50 εποχές. Έχοντας εκπαιδεύσει το νευρωνικό το τεστάρουμε σε ένα σύνολο από στιγμιότυπα (frames) τα οποία δεν έχει συναντήσει στην εκπαίδευση (training) και παρατηρούμε ότι βρίσκει σωστά το Summit χωρίς αληθώς αρνητικά (true negatives) ή ψευδώς θετικά (false positives). Οδηγίες και οι κώδικες (scripts) που υλοποιούν όλα όσα αναφέρθηκαν βρίσκονται στο repository του vision.

6.3 Σχεδιασμός της προσομοίωσης

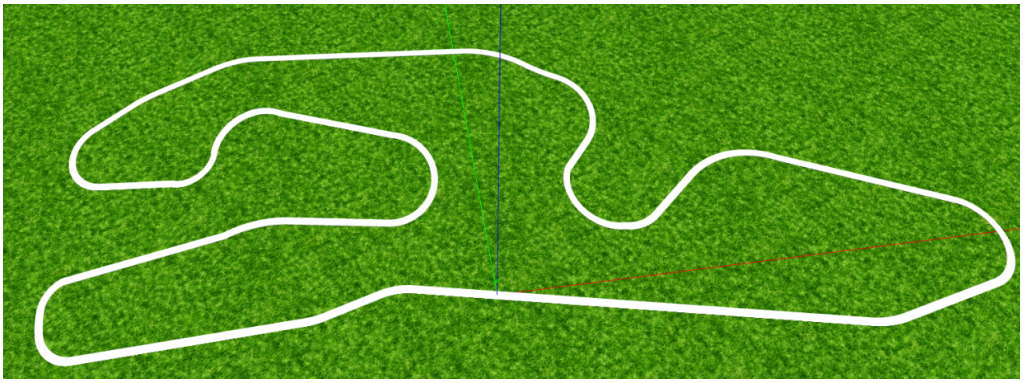
Για να γίνει δυνατή η εκπαιδεύση στο περιβάλλον προσομοίωσης είναι αναγκαίο να σχεδιαστούν οι κόσμοι και τα μοντέλα για το κάθε σενάριο. Το βασικό μοντέλο της προσομοιώσης είναι το πολυκόπτερο με την ZED stereocamera που ελέγχεται από το ardupilot software in the loop (SITL). Αυτό φαίνεται παρακάτω στην εικόνα 6.1.



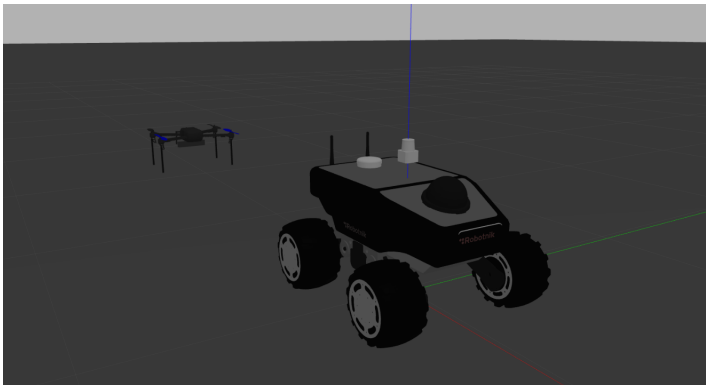
Σχήμα 6.1: Μοντέλο του πολυκόπτερου για την προσομοίωση.

Για το σενάριο του σταθερού περιγράμματος (static contour) δημιουργήθηκε ο παρακάτω κόσμος racetrack (Σχήμα 6.2) που δίνει πλήθος ευθειών και στροφών στις οποίες μπορει να εκπαιδευτεί το πολυκόπτερο μας.

Τέλος για το σενάριο ακολούθησης του κινούμενου οδηγού χρησιμοποιήσαμε το μοντέλο του Summit που δίνεται από την κατασκευάστρια εταιρεία σε ένα κενό κόσμο όπως φαίνεται στο σχήμα 6.3.



Σχήμα 6.2: Κόσμος racetrack



Σχήμα 6.3: Κόσμος με Summit.

Βιβλιογραφία

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[2] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *2017 international conference on engineering and technology (ICET)*, pp. 1–6, Ieee, 2017.

[3] S. N. Aspragkathos, G. C. Karras, and K. J. Kyriakopoulos, "A visual servoing strategy for coastline tracking using an unmanned aerial vehicle," in *2022 30th Mediterranean Conference on Control and Automation (MED)*, pp. 375–381, IEEE, 2022.

[4] A. Rodriguez-Ramos, C. Sampedro, H. Bavle, P. De La Puente, and P. Campoy, "A deep reinforcement learning strategy for uav autonomous landing on a moving platform," *Journal of Intelligent & Robotic Systems*, vol. 93, no. 1, pp. 351–366, 2019.

[5] C. Sampedro, A. Rodriguez-Ramos, I. Gil, L. Mejias, and P. Campoy, "Image-based visual servoing controller for multirotor aerial robots using deep reinforcement learning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 979–986, IEEE, 2018.

[6] A. Rodriguez-Ramos, A. Alvarez-Fernandez, H. Bavle, P. Campoy, and J. P. How, "Vision-based multirotor following using synthetic learning techniques," *Sensors*, vol. 19, no. 21, p. 4794, 2019.

[7] D. Lee, T. Ryan, and H. J. Kim, "Autonomous landing of a vtol uav on a moving platform using image-based visual servoing," in *2012 IEEE international conference on robotics and automation*, pp. 971–976, IEEE, 2012.

[8] G. C. Karras, C. P. Bechlioulis, G. K. Fourlas, and K. J. Kyriakopoulos, "Target tracking with multi-rotor aerial vehicles based on a robust visual servo controller with prescribed performance," in *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 480–487, IEEE, 2020.

[9] P. Vlantis, P. Marantos, C. P. Bechlioulis, and K. J. Kyriakopoulos, "Quadrotor landing on an inclined platform of a moving ground vehicle," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2202–2207, IEEE, 2015.

[10] H. J. Asl and H. Bolandi, "Robust vision-based control of an underactuated flying robot tracking a moving target," *Transactions of the Institute of Measurement and Control*, vol. 36, no. 3, pp. 411–424, 2014.

[11] H. Jabbari Asl, G. Oriolo, and H. Bolandi, "Output feedback image-based visual servoing control of an underactuated unmanned aerial vehicle," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 228, no. 7, pp. 435–448, 2014.

[12] R. Mahony, V. Kumar, and P. Corke, "Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor," *IEEE Robotics and Automation magazine*, vol. 19, no. 3, pp. 20–32, 2012.

[13] F. Sabatino, "Quadrotor control: modeling, nonlinearcontrol design, and simulation," 2015.

[14] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[15] M. L. Puterman, "Markov decision processes," *Handbooks in operations research and management science*, vol. 2, pp. 331–434, 1990.

[16] V. Konda and J. Tsitsiklis, "Actor-critic algorithms," *Advances in neural information processing systems*, vol. 12, 1999.

[17] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *International conference on machine learning*, pp. 387–395, PMLR, 2014.

[18] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[19] S. Minaee, Y. Y. Boykov, F. Porikli, A. J. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," *IEEE transactions on pattern analysis and machine intelligence*, 2021.

[20] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of big data*, vol. 6, no. 1, pp. 1–48, 2019.

[21] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[22] J. Meyer, A. Sendobry, S. Kohlbrecher, U. Klingauf, and O. v. Stryk, "Comprehensive simulation of quadrotor uavs using ros and gazebo," in *International conference on simulation, modeling, and programming for autonomous robots*, pp. 400–411, Springer, 2012.

[23] G. Bradski, "The opencv library dr dobb's journal of software tools," 2020.

[24] D. Gupta, "A beginner's guide to deep learning based semantic segmentation using keras," *Divam Gupta*, vol. 6, 2019.