Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Πάρκινγκ Αυτόνομου Οχήματος με Χρήση Αλγορίθμων Βαθιάς Ενισχυτικής Μάθησης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΖΩΗ ΚΟΡΔΑ

Επιβλέπων: Ανδρέας-Γεώργιος Σταφυλοπάτης
Καθηγητής Ε.Μ.Π.

Αθήνα, Φεβρουάριος 2022

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Πάρκινγκ Αυτόνομου Οχήματος με Χρήση Αλγορίθμων Βαθιάς Ενισχυτικής Μάθησης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΖΩΗ ΚΟΡΔΑ

Επιβλέπων: Ανδρέας-Γεώργιος Σταφυλοπάτης
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 14η Φεβρουαρίου 2022.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
…….	…….	…….
Ανδρέας – Γεώργιος Σταφυλοπάτης Καθηγητής Ε.Μ.Π.	Γεώργιος Στάμου Καθηγητήε Ε.Μ.Π.	Στέφανος Κόλλιας Καθηγητής Ε.Μ.Π.

Αθήνα, Φεβρουάριος 2022

……

Ζωή Κόρδα
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Ένα ρομποτικό όχημα που πλοηγείται αυτόνομα αποτελεί έναν μακροχρόνιο στόχο τεχνητής νοημοσύνης. Γενικότερα, η οδήγηση ενός οχήματος είναι μια εργασία που απαιτεί υψηλό επίπεδο δεξιοτήτων, προσοχής και εμπειρίας από έναν άνθρωπο οδηγό. Αν και οι υπολογιστές έχουν αναπτύξει οξύτερες ικανότητες εστίασης από αυτές των ανθρώπων, η πλήρως αυτόνομη οδήγηση απαιτεί ένα επίπεδο πληροφοριών που ξεπερνούν αυτό που έχει επιτευχθεί μέχρι στιγμής από πράκτορες τεχνητής νοημοσύνης. Εντούτοις, ο τομέας της ενισχυτικής μάθησης (RL) συνιστά ένα ισχυρό μαθησιακό εργαλείο, ικανό πλέον να μαθαίνει πολύπλοκες πολιτικές σε περιβάλλοντα υψηλών διαστάσεων. Στην παρούσα διπλωματική εργασία, μελετάται η λειτουργία πάρκινγκ ενός αυτόνομου οχήματος στο περιβάλλον προσομοίωσης *highway-env* με τη χρήση αλγορίθμων βαθιάς ενισχυτικής μάθησης. Για τον σκοπό αυτό, σχεδιάστηκαν τέσσερις πράκτορες: ένας βελτιστοποιητής τροχιάς βασισμένος σε μοντέλο (model-based CEM planner) και τρεις πράκτορες ενισχυτικής μάθησης χωρίς μοντέλο (DQN, SAC, SAC with HER). Ειδικότερα, μελετήθηκε η συμπεριφορά καθενός από τους παραπάνω πράκτορες στο περιβάλλον, όπως και τα αποτελέσματα των προσομοιώσεων με βάση τις επιβραβεύσεις των επιμέρους πρακτόρων κατά την προσέγγιση κάποιας τυχαίας θέσης-στόχου, χωρίς την ύπαρξη άλλων πρακτόρων ή εμποδίων.

Λέξεις κλειδιά

Τεχνητή Νοημοσύνη, Ενισχυτική Μάθηση, Βαθιά Ενισχυτική Μάθηση, Αυτόνομη Πλοήγηση, Νευρωνικά Δίκτυα, Exploration- Exploitation dilemma, CEM, DQN, Δράστης - Κριτής, SAC, HER.

Abstract

A robotic vehicle that navigates autonomously is a long-term goal of artificial intelligence. Driving a vehicle is a task that requires a high level of skills, attention and experience from a human driver. Although computers have developed sharper focus skills than those of humans, fully autonomous driving requires a level of information that surpasses what has been achieved so far by AI agents. However, the field of reinforcement learning (RL) is a powerful tool, capable of learning complex policies in high-dimensional environments. In this diploma thesis, we study the parking function of an autonomous vehicle in the *highway-env* simulation environment, using deep reinforcement learning algorithms. Four agents were designed for this purpose: a model-based CEM planner and three model-free learning agents (DQN, SAC, SAC with HER). In particular, the behavior of each of the above agents in the environment was studied, as well the results of the simulations based on the rewards of the individual agents when approaching a random position-goal, without the existence of other agents or obstacles.

Key words

Artificial Intelligence, Reinforcement Learning, Deep Reinforcement Learning, Autonomous Navigation, Neural Network, Exploration- Exploitation dilemma, CEM, DQN, Actor-Critic, SAC, HER.

Ευχαριστίες

Ευχαριστώ θερμά τον επιβλέποντα καθηγητή αυτής της διπλωματικής εργασίας, κ. Ανδρέα-Γεώργιο Σταφυλοπάτη για την εμπιστοσύνη και τον χώρο που μου έδωσε να αναπτύξω τις δικές μου ιδέες. Ευχαριστώ επίσης τον κ. Θάνο Τασάκο για την πρόθυμη βοήθειά του και την άψογη συνεργασία και επικοινωνία που είχαμε. Θα ήθελα ακόμη να ευχαριστήσω την οικογένειά μου για την υποστήριξη και κατανόησή τους καθ' όλη τη διάρκεια των σπουδών μου. Εν κατακλείδι, ένα μεγάλο ευχαριστώ στους φίλους μου και σε όσους με βοήθησαν, λιγότερο ή περισσότερο, κατά την διάρκεια των φοιτητικών μου χρόνων.

Ζωή Κόρδα

Αθήνα, 14/03/ 2022

Περιεχόμενα

Κατάλογος Εικόνων

Κατάλογος Πινάκων

1 Εισαγωγή

1.1 Αυτόνομη Οδήγηση

Στα πλαίσια της αυτόνομης οδήγησης ένα όχημα καλείται να επεξεργαστεί το εξωτερικό του περιβάλλον, είτε με τη συνδρομή απομακρυσμένων συστημάτων πλοήγησης είτε μέσω εισερχόμενων πληροφοριών από αισθητήρες του ίδιου οχήματος, και με βάση τα στοιχεία που λαμβάνει, να σχεδιάσει την πορεία του και τελικά να κινηθεί πάνω σε μια προγραμματισμένη τροχιά χωρίς την παρέμβαση του ανθρώπου. Αφού το όχημα λάβει τα απαραίτητα δεδομένα του χώρου στον οποίο κινείται, ο εκάστοτε αλγόριθμος εφαρμόζεται διαμορφώνοντας ένα πλάνο διαδρομής. Σε αρκετές περιπτώσεις, ειδικότερα όταν πρόκειται για κίνηση σε αυτοκινητόδρομο με την παρουσία και άλλων οχημάτων, των οποίων οι προθέσεις δεν είναι εκ των προτέρων γνωστές, ούτε μπορούν να προβλεφθούν, η τροχιά του οχήματος ενδιαφέροντος πρέπει να προσαρμόζεται δυναμικά λαμβάνοντας υπόψη και τις ενέργειες των υπολοίπων στοιχείων του περιβάλλοντός του.

Τα αυτοκινούμενα οχήματα αποτελούν έναν από τους κυριότερους τομείς έρευνας στον σημερινό κόσμο. Από την εκπαίδευση των αυτοκινήτων για την στάθμευση μέχρι την κίνηση αυτών στον δρόμο είναι ένα μεγάλο ταξίδι με αρκετά εμπόδια. Η χρήση τεχνητής νοημοσύνης και νευρωνικών δικτύων έχει προσφέρει σημαντικά τόσο σε προβλήματα πλοήγησης σε σύνθετους χώρους καταστάσεων όσο και αποφυγής εμποδίων, ενώ, η ενισχυτική μάθηση (RL) αποτελεί ένα πλαίσιο γενικής χρήσης για τον σχεδιασμό ελεγκτών για μη γραμμικά συστήματα. Με τη βοήθεια δοκιμής και σφάλματος, προσπαθεί να μάθει έναν ελεγκτή (πολιτική). Ως αποτέλεσμα, συστήματα όπως τα κινούμενα ρομπότ, τα οποία είναι δύσκολο να ελεγχθούν χρησιμοποιώντας συμβατικές μεθοδολογίες ελέγχου, γίνονται διαχειρίσιμα με τη χρήση αλγορίθμων RL. Παραδοσιακά, η ενισχυτική μάθηση εφαρμόζεται σε προβλήματα με χαμηλό χώρο κατάστασης, ωστόσο η χρήση των Βαθιών Νευρωνικών Δικτύων ως εξομοιωτών λειτουργίας με RL έχει δείξει ανώτερα αποτελέσματα για τον έλεγχο συστημάτων υψηλής διάστασης. Ειδικότερα, έχει λειτουργήσει βοηθητικά επιλύοντας πλήθος ζητημάτων ενισχυτικής μάθησης, όπως η πλοήγηση σε απρόβλεπτο χώρο κατάστασης και η κίνηση σε πολυπρακτορικά περιβάλλοντα. Ωστόσο, η πρόκληση του ζητήματος αυτόνομης οδήγησης είναι αρκετά απαιτητική, μιας και οι χώροι κατάστασης του πραγματικού κόσμου είναι εξαιρετικά περίπλοκοι, ενώ οι χώροι δράσεις για το εκάστοτε όχημα είναι συνεχείς και συνεπώς απαιτούν καλό έλεγχο. Παράλληλα, τα αυτόνομα οχήματα οφείλουν κατά τη κίνησή τους να διατηρούν την λειτουργική ασφάλεια στο πλαίσιο ενός σύνθετου περιβάλλοντος.

1.2 Αντικείμενο διπλωματικής

Στην παρούσα εργασία υλοποιούνται και εξετάζονται αλγόριθμοι βαθιάς ενισχυτικής μάθησης για την εκπαίδευση ενός αυτόνομου πράκτορα-οχήματος σε περιβάλλον προσομοίωσης με στόχο την εκτέλεση της λειτουργίας πάρκινγκ σε δεδομένη θέση/στόχο και με συγκεκριμένη κατεύθυνση. Το περιβάλλον προσομοίωσης που χρησιμοποιήθηκε είναι το *«parking-v0»*, που αποτελεί ένα μόνο από τα πολλά περιβάλλοντα που περιλαμβάνει το *highway-env*. Συγκεκριμένα, για την εκπαίδευση του οχήματος μοντελοποιήθηκαν τρεις διαφορετικές τεχνικές βαθιάς ενισχυτικής μάθησης (model-based CEM planner, DQN, SAC), διαφοροποιήθηκαν κατάλληλα οι παράμετροι σε κάθε πείραμα και σχεδιάστηκαν συνολικά τέσσερις πράκτορες, των οποίων η απόδοση εξετάστηκε μέσα από μια σειρά επεισοδίων με στόχο την βέλτιστη δυνατή συμπεριφορά τους στο δεδομένο περιβάλλον.

1.3 Οργάνωση κειμένου

Η παρούσα διπλωματική εργασία αρθρώνεται σε πέντε κεφάλαια. Στο Κεφάλαιο 1 (Εισαγωγή) πραγματοποιείται μια σύντομη αναφορά στην συνεισφορά της τεχνητής νοημοσύνης στα ζητήματα αυτόνομης πλοήγησης και παρουσιάζεται συνοπτικά η κεντρική ιδέα της εργασίας. Στο Κεφάλαιο 2 (Θεωρητικό Υπόβαθρο) αναλύονται τα επιστημονικά πεδία της Μηχανικής Μάθησης, και ειδικότερα των τεχνητών νευρωνικών δικτύων και της Ενισχυτικής μάθησης, όπου παρουσιάζονται οι κυριότεροι και δημοφιλέστεροι αλγόριθμοί της. Στο Κεφάλαιο 3 (Βαθιά Ενισχυτική Μάθηση) παρατίθενται και επεξηγούνται οι αλγόριθμοι που υλοποιήθηκαν για την εκπαίδευση του πράκτορά μας, όπως και το μαθηματικό τους υπόβαθρο. Στο Κεφάλαιο 4 (Υλοποίηση και Αξιολόγηση Πειραμάτων) παρουσιάζεται το περιβάλλον προσομοίωσης, οι αλγόριθμοι που χρησιμοποιήθηκαν, καθώς και τα αποτελέσματα των πειραμάτων. Κλείνοντας, στο Κεφάλαιο 5 (Συμπεράσματα), γίνεται η ερμηνεία των παραπάνω αποτελεσμάτων, η σύγκριση των επιμέρους πειραμάτων και δίνονται κατευθυντήριες γραμμές πιθανών επεκτάσεων της παρούσας διπλωματικής σε μελλοντική έρευνα.

2 Θεωρητικό υπόβαθρο

2.1 Μηχανική Μάθηση

Ένα ανθρώπινο ον τείνει να βελτιώνει αυτόματα τον τρόπο αντιμετώπισης ενός προβλήματος σε αντίθεση με έναν υπολογιστή, εφόσον μαθαίνει από τα προηγούμενα λάθη του και τείνει να τα διορθώνει αναζητώντας νέες προσεγγίσεις για το εκάστοτε πρόβλημα. Τα παραδοσιακά προγράμματα υπολογιστών δεν εξετάζουν το αποτέλεσμα των εργασιών τους και συνεπώς δεν είναι σε θέση να βελτιώσουν τη συμπεριφορά τους.

Η Μηχανική μάθηση (Machine learning) ως κλάδος της Τεχνητής Νοημοσύνης αποτελεί το πεδίο που επικεντρώνεται στην κατασκευή αλγορίθμων οι οποίοι πραγματοποιούν προβλέψεις με βάση δεδομένα, χωρίς τη χρήση προκαθορισμένων οδηγιών [1]. Μια εργασία μηχανικής μάθησης έχει ως στόχο να προσδιορίσει μια συνάρτηση $g : X \rightarrow Y$,η οποία αντιστοιχίζει την είσοδο X (δεδομένα) στην έξοδο Y (πιθανές προβλέψεις). Ανάλογα με τον τύπο του αλγορίθμου μάθησης που χρησιμοποιείται στο εκάστοτε πρόβλημα, οι συναρτήσεις g επιλέγονται από ποικίλες κατηγορίες συναρτήσεων. Ο Mitchell (1997) ορίζει τον όρο "learning" ως εξής: «Ένα υπολογιστικό πρόγραμμα μαθαίνει από την εμπειρία E σχετικά με κάποια κατηγορία εργασιών T και μέτρο απόδοσης P, εάν η απόδοσή του στις εργασίες σε T, όπως μετριέται από το P, βελτιώνεται με την εμπειρία E» [2]. Ως μέτρο επιδόσεων επιλέγεται συνήθως η ακρίβεια του συστήματος, η οποία ορίζεται ως η αναλογία για την οποία το σύστημα παράγει ορθά την έξοδο, όταν αφορά πρόβλημα ταξινόμησης. Η εμπειρία E στην οποία υποβάλλονται οι αλγόριθμοι μηχανικής μάθησης συνίσταται από σύνολα δεδομένων, τα οποία περιέχουν ένα σύνολο παραδειγμάτων που χρησιμοποιούνται για την εκπαίδευση(train) και τη δοκιμή(test) των εν λόγω αλγορίθμων.

2.1.1 Μέθοδοι μηχανικής μάθησης

Οι αλγόριθμοι μηχανικής μάθησης μπορούν να ταξινομηθούν σε μεγάλη κλίμακα στις παρακάτω τρεις κατηγορίες ανά τύπο συνόλων δεδομένων που χρησιμοποιούνται ως εμπειρία.

Επιβλεπόμενη Μάθηση (Supervised Learning)

Οι εποπτευόμενοι αλγόριθμοι μάθησης εκπαιδεύονται χρησιμοποιώντας επισημασμένα παραδείγματα, όπως μια είσοδο για την οποία είναι γνωστή η επιθυμητή έξοδος. Ένας τέτοιος αλγόριθμος λαμβάνει ένα σύνολο εισροών μαζί με τις αντίστοιχες σωστές εξόδους, μαθαίνει συγκρίνοντας τες με τις πραγματικές για την εύρεση σφαλμάτων και τροποποιεί ανάλογα το μοντέλο. Μέσω μεθόδων όπως η ταξινόμηση (classification), η παλινδρόμηση (regression), η πρόβλεψη (prediction) και η ενίσχυση της διαβάθμισης (gradient boosting), η επιβλεπόμενη μάθηση χρησιμοποιεί μοτίβα για να προβλέψει τις τιμές της ετικέτας σε κάθε πρόσθετο δεδομένο χωρίς ετικέτα. Η εποπτευόμενη μάθηση χρησιμοποιείται συνήθως σε εφαρμογές όπου προγενέστερα δεδομένα προβλέπουν πιθανά μελλοντικά γεγονότα. [3], [4]

Μη επιβλεπόμενη Μάθηση (Unsupervised Learning)

Η μη επιβλεπόμενη μάθηση χρησιμοποιείται έναντι δεδομένων που δεν έχουν ετικέτες, δεν παρέχεται δηλαδή κάποια εμπειρία στον αλγόριθμο. Στόχος είναι ο αλγόριθμος να εξερευνήσει τα δεδομένα και να βρει κρυμμένες δομές μέσα σε αυτά ή χαρακτηριστικά που θα είναι χρήσιμα στη διαδικασία μάθησης. [3], [4] Επιγραμματικά, οι δημοφιλέστερες τεχνικές περιλαμβάνουν self-organizing maps, nearest-neighbor mapping, k-means clustering και singular value decomposition.

Ενισχυτική Μάθηση (Reinforcement Learning)

Ένας αλγόριθμος ενισχυτικής μάθησης ανακαλύπτει μέσω μιας διαδικασίας δοκιμών και σφαλμάτων, τις ενέργειες που αποδίδουν τις μεγαλύτερες ανταμοιβές. Ο εν λόγω τύπος

μάθησης έχει τρία βασικά συστατικά: τον πράκτορα/agent (τον εκπαιδευόμενο ή τον φορέα λήψης αποφάσεων), το περιβάλλον/environment (οτιδήποτε αλληλεπιδρά με τον πράκτορα) και τις ενέργειες/actions (οι δυνατές πράξεις που μπορεί να εκτελέσει ο πράκτορας). Θεμελιώδης στόχος στα πλαίσια της ενισχυτικής μάθησης είναι ο εκάστοτε πράκτορας να επιλέγει ενέργειες που μεγιστοποιούν την αναμενόμενη ανταμοιβή σε ένα δεδομένο χρονικό διάστημα. Ο πράκτορας επιτυγχάνει το στόχο πολύ γρηγορότερα ακολουθώντας μια καλή πολιτική (policy) και συνεπώς ζητούμενο στην ενίσχυση της μάθησης είναι να εκπαιδευτεί ο πράκτορας στην καλύτερη δυνατή πολιτική. [3], [4]

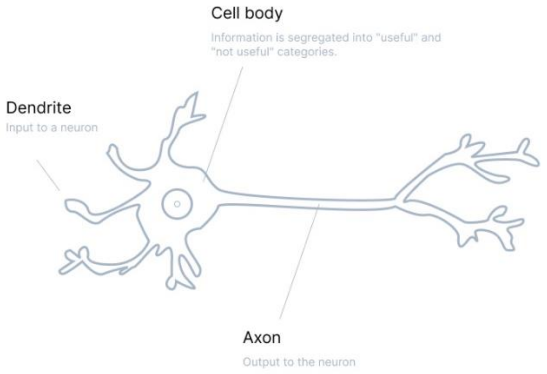
2.1.2 Βαθιά Μάθηση (Deep Learning)

Η Βαθιά Μάθηση αποτελεί πεδίο της Τεχνητής Νοημοσύνης που εκπαιδεύει έναν υπολογιστή να εκτελεί ανθρώπινες εργασίες, όπως η αναγνώριση ομιλίας, ο εντοπισμός εικόνων ή η πραγματοποίηση προβλέψεων. Αποτελεί υποσύνολο της μηχανικής μάθησης, όπου οι αλγόριθμοι δεν οργανώνουν τα δεδομένα για να τρέξουν μέσα από προκαθορισμένες εξισώσεις, αλλά περιλαμβάνουν δίκτυα ικανά να μαθαίνουν, χωρίς επίβλεψη, από δεδομένα που είναι αδόμητα ή μη επισημασμένα. [5] Τα δεδομένα μοντελοποιούνται με πολύπλοκες αρχιτεκτονικές, συνδυάζοντας πληθώρα μη γραμμικών μετασχηματισμών. Η Βαθιά Μάθηση (Deep Learning) εισήχθη σαν έννοια στον τομέα της Μηχανικής Μάθησης από τη Rina Dechter το 1986. [6] Οι αλγόριθμοί της βελτιώνουν την ικανότητα ταξινόμησης, αναγνώρισης και ανίχνευσης, ενώ επιτρέπουν στα συστήματα πρόγνωσης να προσαρμόζονται καλύτερα και να βελτιώνουν την απόδοσή τους με περισσότερα χαρακτηριστικά, καθώς χρησιμοποιούν -σε αντίθεση με τα απλά νευρωνικά δίκτυα (Artificial Neural Network)- περισσότερα επίπεδα στοιβαγμένα το ένα μετά το άλλο.

2.2 Νευρωνικά Δίκτυα

2.2.1 Τεχνητός Νευρώνας

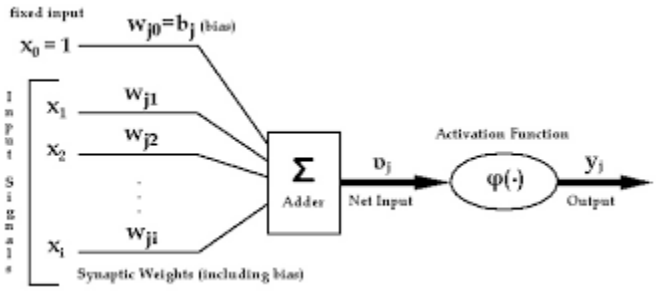
Πρώιμες βιολογικά εμπνευσμένες προσπάθειες μίμησης του εγκεφάλου με αλγοριθμικό/μαθηματικό τρόπο διευθετήθηκαν βάσει ορισμένων παραδοχών και περιορισμών. Οι βασικές ιδέες σχετίζονταν με τη σύγχρονη γνώση σχετικά με τη βιολογική δραστηριότητα των νευρικών συνδέσεων, των μοντέλων βιολογικού νευρώνα και με αντίστοιχες εμπειρίες που πραγματοποιήθηκαν. Οι παραδοχές επικεντρώθηκαν κυρίως στη σημασία της νευρωνικής δραστηριότητας, καθώς οι νευρώνες θα πρέπει να έχουν ένα μοτίβο σήματος "όλα ή καθόλου", εφόσον δεν είναι ενεργός ολόκληρος ο εγκέφαλός μας σε κάθε δεδομένη στιγμή.



***Εικόνα 1**: Neural network*

Ένας νευρώνας πρέπει να διεγείρεται υπό συγκεκριμένη τάση για να ενεργοποιηθεί, π.χ. το άθροισμα των συνδέσεων εισόδου θα πρέπει να οδηγεί σε ένα σήμα στην έξοδο, δημιουργώντας έτσι μια αιτιατή σχέση. Οι καθυστερήσεις στο τεχνητό νευρωνικό μοντέλο θα

πρέπει να λαμβάνονται υπόψη μόνο με σεβασμό στη συναπτική καθυστέρηση (αν και η σύγχρονη καθυστέρηση οφείλεται σε εντατικό υπολογισμό πινάκων). Μια τελευταία παρατήρηση σχετίζεται με τη στατική φύση του δικτύου, καθώς οι εγκέφαλοί μας δεν αλλάζουν συντριπτικά στον κάθε υπολογισμό.



***Εικόνα 2**: Δομή τεχνητού νευρώνα*

Κατά την ανάπτυξη ενός μοντέλου τεχνητού νευρώνα, οι δενδρίτες θα μπορούσαν να θεωρηθούν ως συνδέσεις εισόδου που οδηγούν στον πυρήνα, από τον οποίον ένα άθροισμα ηλεκτρικών παλμών θα μπορούσε να προστεθεί και να μεταβεί σε άλλους νευρώνες χρησιμοποιώντας έναν άξονα ή ακροδέκτες αξόνων. Οι άξονες αυτοί λειτουργούν ως σήμα εξόδου. Γενικότερα, οποιοσδήποτε μετασχηματισμός εισόδου είναι επιτρεπτός (εκτός από το μηδέν), αλλά δεδομένου ότι οι νευρώνες πιστεύεται ότι λειτουργούν με τρόπο «on-off», επιλέγεται αρχικά μια βηματική συνάρτηση. Λαμβάνοντας υπόψη έναν τεχνητό νευρώνα, κάθε σύνδεση αντιπροσωπεύεται από ένα βάρος w_i και κάθε είσοδος από x_i. Συνεπώς, μια νευρωνική είσοδος μεταφράζεται σε:

$$a = w_1x_1 + w_2x_2 + w_3x_3 + \dots + w_ix_i + b = \sum_{i=0}^{i=N} (w_ix_i) + b \qquad (2.1)$$

Η παραπάνω έκφραση περιλαμβάνει αυστηρά γραμμικό συνδυασμό μεταβλητών, π.χ. μοντέλο παλινδρόμησης. Ή σε μορφή μήτρας:

$$a = Wx + b \qquad (2.2)$$

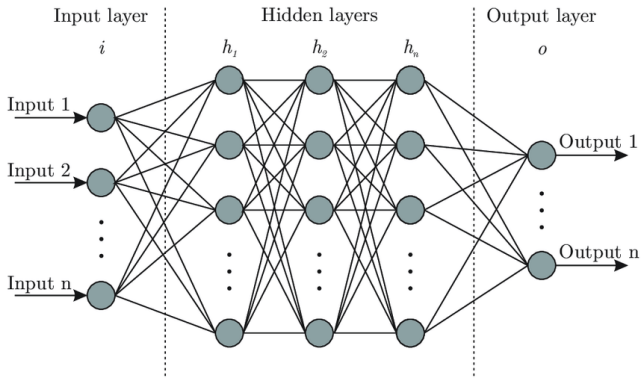
Στην πραγματικότητα, υπάρχουν μέθοδοι για την εκπαίδευση ενός νευρωνικού δικτύου χρησιμοποιώντας αναδρομικό αλγόριθμο ελαχίστων τετραγώνων. Εντούτοις, κάθε έξοδος ενός νευρώνα είναι ένας μη γραμμικός μετασχηματισμός συσχέτισης της n-οστής εισόδου. Μια έξοδος νευρώνων y μπορεί να υπολογιστεί ως εξής:

$$y = \Phi(a) = \Phi(Wx + b) \qquad (2.3)$$

όπου W είναι ο πίνακας βαρών, που αποδίδει βάρη στις συνδέσεις εισόδου, x οι ίδιες οι συνδέσεις και b η σταθερά bias.

2.2.2 Τεχνητά Νευρωνικά Δίκτυα

Τα νευρωνικά δίκτυα, σε αντίθεση με τους συμβατικούς αλγορίθμους, μπορούν να επιλύσουν προβλήματα που είναι περίπλοκα, με ευκολότερο τρόπο όσον αφορά την πολυπλοκότητα των αλγορίθμων. Ο κύριος λόγος για τη χρήση τεχνητών νευρωνικών δικτύων είναι η απλή δομή και η αυτοοργάνωσή τους που τους επιτρέπει να αντιμετωπίσουν ένα ευρύ φάσμα προβλημάτων χωρίς περαιτέρω παρέμβαση του προγραμματιστή. Ένα τυπικό Τεχνητό Νευρωνικό Δίκτυο αποτελείται από κόμβους, τους προαναφερθέντες νευρώνες, σταθμισμένες συνδέσεις μεταξύ αυτών των νευρώνων που μπορούν να προσαρμοστούν κατά τη διάρκεια της μαθησιακής διαδικασίας του δικτύου και μια συνάρτηση ενεργοποίησης που καθορίζει την τιμή εξόδου κάθε κόμβου ανάλογα με τις τιμές εισόδου του.



***Εικόνα 3**: Neural network [7]*

Το μοντέλο παραμετροποιείται από ένα σύνολο παραμέτρων θ που αντιστοιχίζει ένα διάνυσμα εισόδου M διαστάσεων ($\tilde{v}$) μέσω μιας σειράς κρυφών επιπέδων και ενεργοποιήσεων, σε ένα διάνυσμα εξόδου K διαστάσεων ($\tilde{y}$). Συγκεκριμένα, ένα νευρωνικό δίκτυο αποτελείται από διασυνδεδεμένα επίπεδα, όπου κάθε στρώμα υπολογίζει μια γραμμική αντιστοίχιση μεταξύ της εισόδου v και των βαρών της w, προσθέτοντας έναν σταθερό όρο b και χαρτογραφώντας το αποτέλεσμα μέσω μιας μη γραμμικής συνάρτησης ενεργοποίησης. Για παράδειγμα, αντιστοιχίζοντας ένα διάνυσμα εισόδου $\tilde{v}$ μέσω ενός κρυφού στρώματος με βάρη $W_0 \in \theta$, όρο bias $b_0 \in \theta$ και μη γραμμικότητα h_0, έχει ως αποτέλεσμα την ακόλουθη εξίσωση:

$$v_0 = h_0(W_0\, \tilde{v} + b_0) \tag{2.4}$$

Ένα επίπεδο L_i αντιπροσωπεύει ένα συγκεκριμένο σύνολο νευρώνων που μοιράζονται μια κοινή ιδιότητα. Έστω ότι κάθε νευρώνας συμβολίζεται $N_{i,t}$ μιας και όλοι οι t νευρώνες κατοικούν σε ένα επίπεδο i, επομένως:

$$\forall i\ \forall t\ \ N_{i,t} \in N_i \tag{2.5}$$

εάν κάθε νευρώνας που συνδέεται με την είσοδο αντιπροσωπεύεται ως $N_{input,t}$ τότε το σύνολο των νευρώνων εκπροσωπείται ως L_{input}. Το επίπεδο εισόδου (input layer) δέχεται τα δεδομένα εισόδου, δηλαδή λαμβάνει πληροφορίες από εξωτερικές προελεύσεις και τις παρέχει στο δίκτυο χωρίς περεταίρω επεξεργασία. Οι νευρώνες που το αποτελούν είναι υπεύθυνοι για την μεταβίβαση της πληροφορίας. Η πλειονότητα των επιπέδων μεταξύ μιας εισόδου και μιας εξόδου αναφέρονται ως κρυφά στρώματα (hidden layers). Το σύνολό τους αναπαριστά ένα εσωτερικό μπλοκ υπολογισμού, που δεν αλληλεπιδρά άμεσα με καμία εξωτερική μεταβλητή. Θεωρώντας ένα πλήρως συνδεδεμένο νευρωνικό δίκτυο, κάθε νευρώνας στο επίπεδο εισόδου συνδέεται με κάθε νευρώνα του αμέσως επόμενου κρυφού επιπέδου, όπου κάθε έξοδος είναι συνάρτηση της προηγούμενης μονάδας. Στα κρυφά επίπεδα πραγματοποιείται η επεξεργασία των δεδομένων εισόδου και η εξαγωγή κατάλληλων χαρακτηριστικών τα οποία στη συνέχεια μεταβιβάζονται στο επίπεδο εξόδου. Η αύξηση του βάθους των κρυφών επιπέδων οδηγεί σε αύξηση του βάθους των χαρακτηριστικών που εξάγονται. Αν το x_{input} είναι ένα διάνυσμα δεδομένων που τροφοδοτείται από το επίπεδο εισόδου και το $y_{Nhidden,t}$ μια έξοδος t του πρώτου κρυφού επιπέδου, τότε:

$$\forall t \quad y_{Nhidden,t} = \Phi(W_{input,t} * x_{input} + bt) \tag{2.6}$$

,όπου το L_{input} αντιπροσωπεύει το διάνυσμα εισόδου x_{input}. Ωστόσο, τα επίπεδα περιέχουν πλήθος νευρώνων και διαφορετικές διαστάσεις. Μπορεί να γίνει γενίκευση της παραπάνω σχέσης από έναν νευρώνα στο πλήρες στρώμα, λαμβάνοντας υπόψη όλες τις συνδέσεις προς όλους τους νευρώνες και συμπυκνώνοντάς τες σε μορφή μήτρας ως εξής:

$$y_{Lhidden} = \Phi(W_{input} * y_{Linput} + b) \tag{2.7}$$

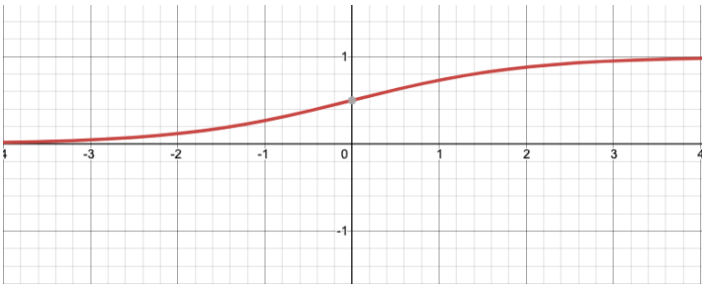
,όπου $y_{Lhidden}$ είναι ένα διάνυσμα στο οποίο κάθε καταχώριση αντιπροσωπεύει την έξοδο ενός συγκεκριμένου νευρώνα. Η μετάδοση δεδομένων προς τα εμπρός σε ένα νευρωνικό δίκτυο αναφέρεται συνήθως ως “forward-pass”. Το “forward-pass” είναι μια διαδοχική διαδικασία από επίπεδο σε επίπεδο, η οποία συνίσταται από πλήθος πολλαπλασιασμών και αθροισμάτων μήτρας. Διαδοχικά στρώματα με όλους τους νευρώνες συνδεδεμένους από ένα στρώμα σε ένα άλλο αναφέρονται στη βιβλιογραφία ως πλήρως συνδεδεμένα επίπεδα (fully connected layers). Αν και τα πλήρως συνδεδεμένα δίκτυα ήταν η πρώτη γενιά νευρωνικών δικτύων (πολυστρωματικά perceptrons), περαιτέρω έρευνες επικεντρώθηκαν στη διασπορά μεταξύ των συνδέσεων, που οδηγούν σε δημοφιλή περίπλοκα δίκτυα. Οι νευρώνες, λοιπόν, των κρυφών επιπέδων εκτελούν τους υπολογισμούς, όπως περιγράφηκαν παραπάνω, και τροφοδοτούν το αποτέλεσμα ως όρισμα στη συνάρτηση ενεργοποίησης που υλοποιεί εσωτερικά κάθε κόμβος. Η τιμή που λαμβάνει η συνάρτηση αυτή αποτελεί την έξοδο του νευρώνα για τις τρέχουσες εισόδους και βάρη και καθορίζει αν θα ενεργοποιηθεί ο εκάστοτε νευρώνας ή όχι.

2.2.3 Συνάρτηση ενεργοποίησης

Στα νευρωνικά δίκτυα η συνάρτηση ενεργοποίησης (activation/transfer function) καθορίζει τον τρόπο με τον οποίο το σταθμισμένο άθροισμα της εισόδου μετατρέπεται σε έξοδο από έναν κόμβο ή κόμβους σε ένα επίπεδο δικτύου. Ειδικότερα, εισάγει μια μη γραμμικότητα στην έξοδο και αποφασίζει αν ένας νευρώνας πρέπει να ενεργοποιηθεί ή όχι. Παρακάτω παρουσιάζονται ορισμένες κλασσικές συναρτήσεις ενεργοποίησης.

Σιγμοειδής συνάρτηση (sigmoid). Η σιγμοειδής συνάρτηση χρησιμοποιείται εκτενώς καθώς η λειτουργία της μπορεί να ερμηνευτεί σαν ποσοστό πυροδότησης ενός νευρώνα, δεδομένου ότι η τιμή του κυμαίνεται από 0 (δεν πυροδοτείται καθόλου) έως 1 (πλήρως κορεσμένη πυροδότηση σε υποτιθέμενη μέγιστη συχνότητα). Εκφράζεται από τον μαθηματικό τύπο:

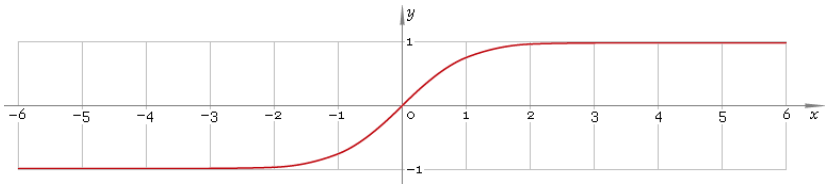
$$\Phi(z) = \text{sigmoid}(z) = \frac{1}{(1+e^{-z})} \tag{2.8}$$



***Εικόνα 4**: Sigmoid Activation function [8]*

Υπερβολική εφαπτομένη συνάρτηση (hyperbolic tangent). Εκφράζεται από τον μαθηματικό τύπο:

$$\Phi(z) = \text{tangenthyperbolic}(z) = \frac{2}{(1+e^{-2z})} - 1 \tag{2.9}$$



***Εικόνα 5**: Hyperbolic Tangent function [9]*

Softmax . Η Softmax δεν είναι μια συνεχής μαθηματική συνάρτηση όπως η sigmoid, η tanh ή η ReLU. Χρησιμοποιείται για να χαρτογραφήσει τις εξόδους του τελευταίου στρώματος ενός

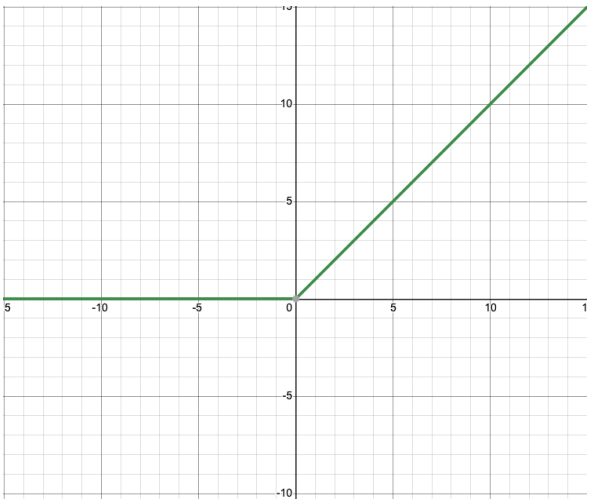
νευρωνικού δικτύου σε μια κατανομή πιθανοτήτων, δηλαδή, η σύνοψη των εξόδων του στρώματος softmax θα είναι 1 (ενότητα).Σε αντίθεση με άλλες συναρτήσεις ενεργοποίησης, η softmax λαμβάνει μια λίστα/πίνακα εισόδου και τις αντιστοιχίζει στην κατανομή πιθανοτήτων. Η έξοδος softmax, λοιπόν, μπορεί να θεωρηθεί ως κατανομή πιθανότητας σε ένα πεπερασμένο σύνολο αποτελεσμάτων. Προσθέτει μη γραμμικότητα στην έξοδο και χρησιμοποιείται κυρίως για την πρόβλεψη διακριτών πιθανοτήτων σε σχέση με τις κατηγορίες εξόδου. Εκφράζεται από τον μαθηματικό τύπο:

$$\Phi(z) = \text{softmax}(z) = \frac{e^{zi}}{\sum_j e^{zj}} \tag{2.10}$$

Rectifier Linear Unit (ReLU). Η μη γραμμικότητα της ReLU επιταχύνει τη σύγκλιση της Stochastic Gradient Descent και έχει μικρότερο υπολογιστικό κόστος σε σύγκριση με την σιγμοειδή και εφαπτομένη (sigmoid/tanh), ενώ η γραμμική της μορφή είναι υπεύθυνη για την ταχύτερη σύγκλιση. Παρόλα αυτά, οι συναρτήσεις ReLU έχουν ένα μειονέκτημα, το οποίο καλείται ανεπίσημα το πρόβλημα του "πεθαμένου ReLU" (dead ReLU), σύμφωνα με το οποίο είναι αρκετά εύθραυστη κατά τη διάρκεια της εκπαίδευσης και μπορεί να "πεθάνει". Για παράδειγμα μια μεγάλη κλίση (gradient) που ρέει μέσω ενός νευρώνα ReLU ενδέχεται να αναγκάσει τα βάρη να ενημερωθούν με τέτοιο τρόπο ώστε ο εκάστοτε νευρώνας να μην ενεργοποιηθεί ποτέ ξανά. Εάν συμβεί αυτό, η κλίση (gradient) που ρέει μέσα από τη μονάδα θα είναι για πάντα μηδέν από εκείνο το σημείο και μετά. Παρόλα αυτά, όταν $x > 0$ υπάρχει αρκετά μικρή πιθανότητα να εξαφανιστεί η κλίση, η οποία θα έχει σταθερή τιμή και συμβάλει στην ταχύτερη εκμάθηση του μοντέλου. Σε αντίθεση με την κλίση των σιγμοειδών συναρτήσεων η οποία γίνεται όλο και μικρότερη καθώς αυξάνεται η απόλυτη τιμή του x. Όταν δηλαδή όλες οι εισροές δεν ωθούν τη ReLU στο αρνητικό τμήμα, οι νευρώνες μπορούν να παραμείνουν ενεργοί, τα βάρη μπορούν να ενημερωθούν και το δίκτυο μπορεί να συνεχίσει να μαθαίνει.

Ένα ακόμη πλεονέκτημα της ReLU θεωρείται η αραιότητα (sparsity), η οποία προκύπτει όταν $x <= 0$ και σύμφωνα με την οποία όσο περισσότερες μονάδες υπάρχουν σε ένα επίπεδο τόσο πιο αραιή είναι η προκύπτουσα αναπαράσταση. Εκφράζεται από τον μαθηματικό τύπο:

$$\Phi(z) = \max(0,z) \text{ or } \Phi(z) = (z)^{+} \tag{2.11}$$



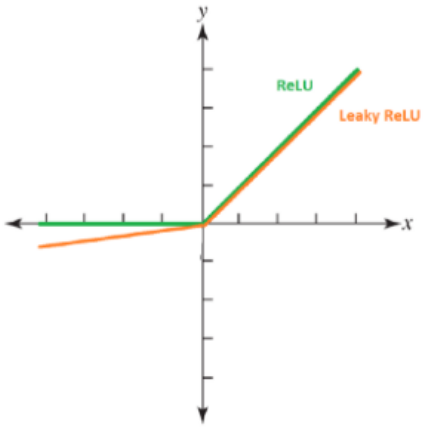
***Εικόνα 6**: ReLU Activation function [10]*

Αξίζει να αναφερθεί συμπληρωματικά με τα παραπάνω, ότι υπάρχουν διάφοροι τρόποι αντιμετώπισης του ζητήματος της «πεθαμένης» ReLU: [57]

i) Χρήση μικρότερου ποσοστού μάθησης: Ένα μεγάλο ποσοστό μάθησης έχει ως αποτέλεσμα μεγαλύτερη πιθανότητα αρνητικών βαρών, αυξάνοντας έτσι τις πιθανότητες θανάτου της ReLU. Έτσι, προτείνεται η μείωση του ποσοστού μάθησης κατά τη διάρκεια της εκπαίδευσης.

ii) Παραλλαγές της ReLU: Δεδομένου ότι είναι το επίπεδο τμήμα στο αρνητικό εύρος εισόδου που προκαλεί το πρόβλημα της «dead ReLU», εξετάζονται παραλλαγές της ReLU που προσαρμόζουν αυτό το επίπεδο τμήμα.

Το *Leaky ReLU* είναι μια κοινή αποτελεσματική μέθοδος για την επίλυση του «πεθαμένου» ReLU, προσθέτοντας μια μικρή κλίση στο αρνητικό εύρος. Αυτό τροποποιεί τη συνάρτηση ώστε να δημιουργεί μικρές αρνητικές εκροές όταν η είσοδος είναι μικρότερη από 0. Υπάρχουν και άλλες παραλλαγές όπως παραμετρικό ReLU (PReLU), εκθετική γραμμική μονάδα (ELU) και γραμμικές μονάδες σφάλματος Gaussian (GELU), όλες με κοινό παρονομαστή την αποφυγή τμημάτων μηδενικής κλίσης.



***Εικόνα 7**: Graphical comparison of ReLU and Leaky ReLU [57]*

iii) Τροποποίηση της διαδικασίας προετοιμασίας: Ένας κοινός τρόπος για την αρχικοποίηση βαρών και σταθερών όρων των νευρωνικών δικτύων είναι μέσω συμμετρικών κατανομών πιθανοτήτων. Ωστόσο, μια τέτοια μέθοδος είναι επιρρεπής στο πρόβλημα της «dead relu» λόγω κακών τοπικών ελαχίστων. Έχει αποδειχθεί ότι η χρήση μιας τυχαιοποιημένης ασύμμετρης αρχικοποίησης μπορεί να βοηθήσει στην πρόληψη του αυτού του ζητήματος.

2.2.4 Συνάρτηση Απώλειας (Loss function)

Αποτελεί συνήθως συνάρτηση του επιπέδου εξόδου, ενώ οι αναμενόμενες έξοδοι ορίζονται από ετικέτες ή στόχους και άλλες μεταβλητές που σχετίζονται με την εκπαίδευση των παραμέτρων του νευρωνικού δικτύου. Σκοπός της συνάρτησης απώλειας $L(\hat{y},y)$ είναι να ποσοτικοποιήσει την ζημιά που προκύπτει όταν η πρόβλεψη είναι $\hat{y}$ ενώ η πραγματική τιμή είναι y. Είναι κάτω φραγμένη και λαμβάνει την ελάχιστη τιμή της όταν η πρόβλεψη του εκάστοτε μοντέλου είναι σωστή. Θεωρώντας συνάρτηση εισόδου-εξόδου : $\hat{y} = f(x;w)$ τότε προκύπτει $L(\hat{y},y) = L(f(x;w),y) = L(w)$, όπου w τα βάρη του νευρωνικού δικτύου. Στόχος λοιπόν της εκπαίδευσης είναι η επιλογή κατάλληλων βαρών ώστε να επαληθευτεί η παρακάτω σχέση:

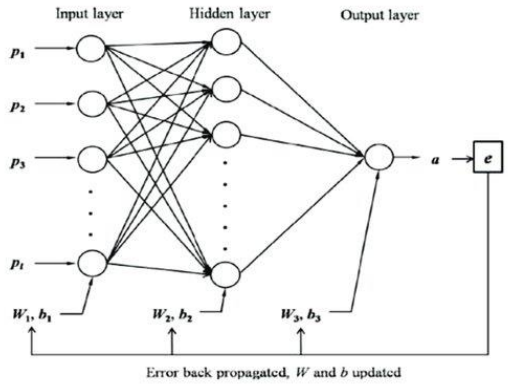
$$w_d = \arg_w \min L(\theta) \tag{2.12}$$

2.2.5 Οπίσθια Διάδοση (back propagation)

Ο αλγόριθμος της οπίσθιας διάδοσης χρησιμοποιείται κατά τη διαδικασία εκπαίδευσης ενός νευρωνικού δικτύου και εκτελεί την αναπροσαρμογή των βαρών ώστε να εξασφαλιστεί η σύγκλιση του εκάστοτε μοντέλου στο επιθυμητό αποτέλεσμα. Ένας αλγόριθμος

backpropagation λοιπόν υπολογίζει την απώλεια ή το κόστος που προκύπτει λόγω της διαφοράς πρόβλεψης y και ετικέτας ŷ, και μεταδίδει το σφάλμα στα προηγούμενα επίπεδα της δομής. Με αυτόν τον τρόπο υπολογίζεται το σφάλμα σε σχέση με τις παραμέτρους του μοντέλου.

Για τον αλγόριθμο ανάστροφης διάδοσης θεμελιώδες μαθηματικό εργαλείο θεωρείται το υπολογιστικό γράφημα. Ένα υπολογιστικό γράφημα είναι ένας κατευθυνόμενος γράφος του οποίου οι κόμβοι αντιστοιχούν σε συναρτήσεις ή μεταβλητές. Οι μεταβλητές μπορούν να τροφοδοτήσουν την τιμή τους στις συναρτήσεις, και οι συναρτήσεις μπορούν να τροφοδοτήσουν την έξοδο τους μέσα σε άλλες συναρτήσεις. Με αυτόν τον τρόπο, κάθε κόμβος στο γράφημα ορίζει μια συνάρτηση μεταβλητών. Επομένως, όλα τα νευρωνικά δίκτυα μπορούν να αναπαρασταθούν χρησιμοποιώντας ένα υπολογιστικό γράφημα. Σημειώνεται ότι στην περίπτωση εμπρόσθιας διάδοσης, το γράφημα δεν περιέχει κανέναν κύκλο.



***Εικόνα 8**: A typical structure of BPNN architecture [11]*

Η διαδικασία της οπίσθιας διάδοσης ακολουθεί τα παρακάτω βήματα [12]:

1. **Αρχικοποίηση**. Τα βάρη και τα κατώφλια του δικτύου αρχικοποιούνται εντός ομοιόμορφης κατανομής με μέση τιμή 0 και διασπορά τέτοια ώστε η τυπική απόκλιση των παραμέτρων των νευρώνων να είναι μεταξύ των ορίων της συνάρτησης ενεργοποίησης.
2. **Παραδείγματα εκπαίδευσης**. Παρουσιάζεται στο δίκτυο μια εποχή από δεδομένα εκπαίδευσης, τα οποία καθώς εκπαιδεύονται προκαλούν την αναπροσαρμογή των βαρών των νευρώνων.
3. **Ευθύς υπολογισμός**. Κατά τη διάρκεια της προώθησης, το σήμα εισόδου μεταδίδεται από τα ρηχότερα στα βαθύτερα επίπεδα της αρχιτεκτονικής, προωθώντας σταδιακά τα σήματα και καταλήγοντας σε ένα τελικό σήμα στην έξοδο του δικτύου. Οι ενδιάμεσες τιμές του σήματος αποθηκεύονται στα αντίστοιχα επίπεδα του νευρωνικού. Έστω μία είσοδος (x(n),d(n)), όπου x(n) εφαρμόζεται στην είσοδο του δικτύου και το d(n) εκφράζει την επιθυμητή έξοδο για την παραπάνω είσοδο. Το σφάλμα εξόδου δίνεται από τον τύπο : $e_i(n) = d_i(n) – y_i(n)$,όπου $y_i(n)$ η έξοδος για τον i νευρώνα.
4. **Υπολογισμός ανάδρασης**. Ο υπολογιστικός γράφος του δικτύου, έχει ένα τελικό επίπεδο u_n, το οποίο παριστάνει τη τελική απώλεια J. Η αξία του u_n εξαρτάται από τα προηγούμενα επίπεδα u_1, ..., u_n οπότε πρέπει να υπολογιστούν όλες οι παράγωγοι $\partial u_n/\partial u_i$, όπου $i \in \{1, ..., n\}$. Αντίστοιχα, κάθε μία από αυτές τις τιμές μπορεί να έχει επίσης προηγούμενες εξαρτήσεις στον γράφο, επομένως θα πρέπει να υπολογιστούν οι παράγωγοι που συλλαμβάνουν αυτές τις εξαρτήσεις. Σε αυτό συμβάλει ο αλγόριθμος ανάστροφης διάδοσης ο οποίος υπολογίζει όλες αυτές τις παραγώγους. Στο παρόν βήμα λοιπόν υπολογίζονται τα local gradients (δ_s) του δικτύου σύμφωνα με τους παρακάτω τύπους.

$\delta_j^{(l)}(n) = e_j^{(L)}(n)\varphi_j'(v_j^{(L)}(n))$,για το επίπεδο εξόδου (2.13)

$\delta_j^{(l)}(n) = \varphi_j'(v_j^{(l)}(n))\sum_k \delta_k^{(l+1)}(n)w_{kj}^{(l+1)}(n)$,για τα κρυφά επίπεδα (2.14)

Η προσαρμογή των βαρών πραγματοποιείται σύμφωνα με την εξίσωση:

$$W_{ji}^{(l)}(n+1) = w_{ji}^{(l)}(n) + \alpha[w_{ji}^{(l)}(n-1)] + \eta\delta_j(l)(n)y_i^{(l-1)}(n) \quad (2.15)$$

,όπου η: παράμετρος του ρυθμού εκμάθησης.

5. **Επανάληψη**. Τα παραπάνω βήματα επαναλαμβάνονται για νέες εποχές των παραδειγμάτων εκπαίδευσης έως και την επαλήθευση του εκάστοτε κριτηρίου τερματισμού που έχει τεθεί.

2.2.6 Βελτιστοποίηση

Οι αλγόριθμοι βελτιστοποίησης ελαχιστοποιούν μια συνάρτηση σφάλματος L(θ), η οποία εξαρτάται από τις εσωτερικές παραμέτρους του μοντέλου, που επηρεάζουν τη διαδικασία μάθησης και την έξοδό του. Η Gradient Descent είναι η πιο συχνή τεχνική και το θεμέλιο του τρόπου με τον οποίο εκπαιδεύονται και βελτιστοποιούνται τα Ευφυή Συστήματα. Χρησιμοποιείται για την ενημέρωση και τον συντονισμό των παραμέτρων του μοντέλου στην αντίθετη κατεύθυνση της κλίσης της αντικειμενικής συνάρτησης $\nabla_\theta L(\theta)$. Ο ρυθμός εκμάθησης (learning rate) η είναι μια υπερπαράμετρος που ελέγχει το βαθμό που προσαρμόζονται τα βάρη του δικτύου σε σχέση με την κλίση της απώλειας loss gradient. Επιπλέον, καθορίζει το πλήθος των βημάτων που λαμβάνονται για να προσεγγιστεί ένα (τοπικό) ελάχιστο. Ένας πολύ χαμηλός ρυθμός μάθησης οδηγεί σε πολύ αργή σύγκλιση, ενώ ένας αρκετά μεγάλος ρυθμός εκμάθησης μπορεί να εμποδίσει τη σύγκλιση και να προκαλέσει τη διακύμανση της συνάρτησης απώλειας στο ελάχιστο ή ακόμα και την απόκλιση. Υπάρχουν τρεις παραλλαγές της Gradient Descent που διαφέρουν ως προς το πλήθος των δεδομένων που χρησιμοποιούνται για να υπολογιστεί η κλίση της συνάρτησης (L(θ)).

Batch Gradient Descent

Η Vanilla/Batch gradient descent υπολογίζει την κλίση της συνάρτησης κόστους στις παραμέτρους θ για ολόκληρο το σύνολο δεδομένων εκπαίδευσης:

$$\theta \leftarrow \theta - \eta \cdot \nabla_\theta L(\theta) \quad (2.17)$$

Η batch gradient descent είναι πολύ αργή, διότι πρέπει να υπολογιστούν οι κλίσεις-gradients για όλο το σύνολο δεδομένων για να εκτελεστεί μόλις μια ενημέρωση. [13]

Stochastic Gradient Descent – SGD

Εκτελεί μία ενημέρωση παραμέτρων κάθε φορά για κάθε παράδειγμα εκπαίδευσης $x^{(i)}$ και ετικέτα $y^{(i)}$:

$$\theta \leftarrow \theta - \eta \cdot \nabla_\theta L(\theta; x^{(i)}; y^{(i)}) \quad (2.18)$$

Η SGD είναι ταχύτερη από την batch gradient descent διότι εκτελεί μια ενημέρωση τη φορά και μπορεί να χρησιμοποιηθεί για να μάθει online. Ωστόσο, εκτελώντας συχνές ενημερώσεις με μεγάλη διακύμανση, οι ενημερώσεις των παραμέτρων έχουν μεγάλη διακύμανση. Ενώ, λοιπόν, η διακύμανση της συνάρτησης σφάλματος σε διαφορετικές εντάσεις συντελεί στην ανακάλυψη νέων τοπικών ελαχίστων, παρόλα αυτά, η αργή μείωση του ρυθμού εκμάθησης οδηγεί στην σύγκλιση σε ένα τοπικό ή παγκόσμιο ελάχιστο για μη κυρτή και κυρτή βελτιστοποίηση αντίστοιχα. [13]

Adam

Αποτελεί επέκταση της SGD και χρησιμοποιείται για την ενημέρωση των βαρών του δικτύου με βάση τα δεδομένα εκπαίδευσης. Ο ADAM (Adaptive Moment Estimation) συνδυάζει τις μεθόδους AdaGrad και RMSProp. Ειδικότερα, αποθηκεύει τόσο έναν εκθετικά φθίνοντα μέσο όρο τετραγωνισμένων διαβαθμίσεων, όσο και έναν εκθετικά φθίνοντα μέσο όρο προηγούμενων διαβαθμίσεων. Οι μέσοι βαθμοί απόσβεσης των προηγούμενων και παρελθοντικών τετραγωνικών διαβαθμίσεων m_t και v_t υπολογίζονται ως εξής:

$$m^{(t+1)} = \beta_1 \cdot m^{(t)} + (1 - \beta_1) \cdot \nabla L_\theta \tag{2.19}$$

$$v^{(t+1)} = \beta_2 \cdot v^{(t)} + (1 - \beta_2) \cdot \nabla L_\theta^2 \tag{2.20}$$

,όπου β_1 και β_2 είναι υπερπαράμετροι και m_t, u_t είναι οι εκτιμήσεις του μέσου όρου και της μη συγκεντρωμένης απόκλισης των κλίσεων αντίστοιχα, εξ ου και το όνομα της μεθόδου. Δεδομένου ότι τα m_t και v_t αρχικοποιούνται ως μηδενικά διανύσματα, οι συγγραφείς του Adam παρατηρούν ότι οι παραπάνω παράμετροι είναι προκατειλημμένοι ως προς το μηδέν, ιδιαίτερα κατά τη διάρκεια των αρχικών χρονικών βημάτων και όταν οι ρυθμοί αποσύνθεσης είναι μικροί (δηλαδή β_1 και β_2 προσεγγίζουν το 1). Αυτές οι προκαταλήψεις αντισταθμίζονται με υπολογισμό των διορθωμένων προκαταρκτικών εκτιμήσεων πρώτης και δεύτερης στιγμής (μέσου όρου και απόκλισης):

$$\hat{m}^{(t+1)} = m^{(t)} / (1-\beta_1^t) \tag{2.21}$$

$$\hat{v}^{(t+1)} = v^{(t)} / (1-\beta_2^t) \tag{2.22}$$

Ο τελικός κανόνας ενημέρωσης είναι παρόμοιος με αυτούς που αναφέρθηκαν παραπάνω:

$$\theta^{(t+1)} = \theta^{(t)} - \{ \eta / [(\hat{v}^{(t+1)})^{1/2} + \varepsilon \} \cdot \hat{m}^{(t+1)} \tag{2.23}$$

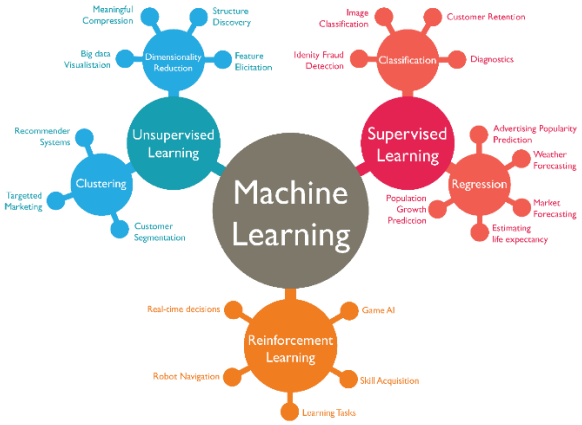
,όπου το ε είναι και πάλι μια σταθερά για την πρόληψη διαίρεσης με το 0. Οι συγγραφείς του Adam προτείνουν προκαθορισμένες τιμές 0.9 για β_1 και 0.999 για β_2 και 10^{-8} για ε. Δείχνουν εμπειρικά ότι ο Adam λειτουργεί καλά στην πράξη και συγκρίνεται ευνοϊκά με άλλους αλγορίθμους προσαρμοστικής μεθόδου μάθησης. [14], [15]

2.3 Ενισχυτική Μάθηση

2.3.1 Εισαγωγή

Η Ενισχυτική Μάθηση είναι ένα μαθησιακό πλαίσιο όπου ένας πράκτορας/μαθητής αλληλεπιδρά με το περιβάλλον του μέσω δοκιμών και σφαλμάτων. Σε αντίθεση με άλλες μεθόδους μηχανικής μάθησης, ο πράκτορας δεν είναι ενημερωμένος για τις κατάλληλες ενέργειες που πρέπει να ληφθούν. Αντ 'αυτού, ο πράκτορας εξερευνά το περιβάλλον για την επίτευξη του μέγιστου ποσού μελλοντικών ανταμοιβών (ή στατιστικά το υψηλότερο άθροισμα αναμενόμενων ανταμοιβών), συνήθως μέσω αναζήτησης στόχου/αντικειμένου (ή χώρου-στόχου) που αντιπροσωπεύεται αριθμητικά με μια μεγάλη ανταμοιβή. [16]

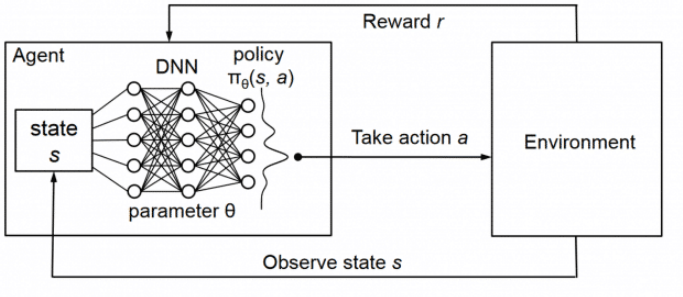
Η προσέγγιση αυτή διαφέρει ελαφρώς από την επιβλεπόμενη μάθηση, εφόσον δειγματοληπτεί απευθείας αλληλεπιδράσεις και εφαρμόζει στατιστική αναγνώριση μοτίβων. Ο πράκτορας δηλαδή ενεργεί και μαθαίνει ταυτόχρονα και λαμβάνοντας υπόψη την υπόθεση της μηδενικής εκ των προτέρων γνώσης για το περιβάλλον, απαιτείται κατάλληλη στρατηγική εξερεύνησης/εκμετάλλευσης (Exploration/Exploitation). Κύριος στόχος αποτελεί η σταδιακή βελτίωση μιας σειράς δράσεων, δεδομένων των καταστάσεων του περιβάλλοντος, επιτυγχάνοντας τελικά την καλύτερη πολιτική/συμπεριφορά για τον πράκτορά μας. [17]



***Εικόνα 9**: Fields of machine learning*

2.3.2 Μοντέλο προβλήματος

Κάθε πρόβλημα εκμάθησης αποτελείται από παρόμοιες ενότητες: ένας «μαθητής» (πράκτορας), ένας «δάσκαλος» (συνάρτηση ανταμοιβής), μια μέτρηση απόδοσης του πόσο καλά συμπεριφέρεται ο πράκτορας μάθησης (συνήθως οι δοκιμές αντιπροσωπεύονται αριθμητικά από ανταμοιβές). Στο κλασσικό μοντέλο ενισχυτικής μάθησης, ένας πράκτορας συνδέεται με το περιβάλλον του μέσω της παρατήρησης και της πράξης. Σε κάθε βήμα t της αλληλεπίδρασης, ο πράκτορας λαμβάνει ως είσοδο κάποια παρατήρηση της τωρινής του κατάστασης O_t του περιβάλλοντος. Στη συνέχεια, ακολουθώντας συγκεκριμένη στρατηγική αποφασίζει μια πράξη A_t ,η οποία λογίζεται ως έξοδός του. Η κάθε πράξη με την σειρά της, αλλάζει την κατάστασή του, και παράγει και μια αξία της συγκεκριμένης κατάστασης η οποία δίνεται στον πράκτορα ως βαθμωτό σήμα ενίσχυσης (ανταμοιβή) R_t . Στόχος του πράκτορα είναι η επιλογή πράξεων που στοχεύουν στην αύξηση του μακροπρόθεσμου αθροίσματος των ανταμοιβών. [17]



***Εικόνα 10**: Agent-Environment interaction [18]*

Περιβάλλον

Ένα περιβάλλον αποτελεί τον κόσμο όπου ο πράκτορας ενεργεί και μαθαίνει. Μια κατάσταση ενός περιβάλλοντος μπορεί να αναπαρασταθεί από ένα χώρο διαστάσεων N που περιέχεται στον χώρο κατάστασης ενός περιβάλλοντος D, π.χ.

$$s_t \subset D, \quad s_t \in R^N$$

όπου s_t είναι μια συγκεκριμένη κατάσταση στο χρόνο t και D ο χώρος κατάστασης που περιλαμβάνει τον κόσμο μας. Η διάσταση της κατάστασης R^N εξαρτάται από τη μορφή του προβλήματος. Η κατάλληλη διαστατική αναπαράσταση του περιβάλλοντος επηρεάζει την εκμάθηση του πράκτορα. Για κάθε πρόβλημα, μπορεί να υπάρχει ένας μεγάλος αριθμός διαφορετικών πιθανών αναπαραστάσεων.

Πράξη

Ως πράξη A_t ορίζονται οι δυνατές επιλογές που έχει ο πράκτορας να αλληλεπιδράσει με το περιβάλλον του.

Ανταμοιβή

Στα πλαίσια της ενισχυτικής μάθησης, ένας πράκτορας μαθαίνει με ενίσχυση. Πιο συγκεκριμένα, μια αρνητική ανταμοιβή οδηγεί σε ανεπιθύμητη συμπεριφορά. Ενώ, μια σειρά θετικών ανταμοιβών συνεπάγεται μια αξιόλογη πολιτική. Στόχος του πράκτορα είναι η μεγιστοποίηση του συνολικού αθροίσματος των ανταμοιβών:

$$R_t = r_{t+1} + r_{t+2} + r_{t+3} + \ldots + r_T \quad (2.24)$$

,όπου το R_t αντιπροσωπεύει την επιστροφή, t ένα συγκεκριμένο χρονικό βήμα μέχρι το T.

Για στοχαστικά περιβάλλοντα, ζητούμενο είναι να μεγιστοποιηθεί η αναμενόμενη αξία της επιστροφής μέσα σε μια εργασία. Επομένως, η παραπάνω εξίσωση γράφεται ως εξής:

$$R_t = \sum_{k=0}^{T} r_{t+k+1} \quad (2.25)$$

Ωστόσο, μια μελλοντική ανταμοιβή μπορεί να έχει διαφορετική παρούσα τιμή. Το ποσοστό της διαφοράς αυτής αναφέρεται στη βιβλιογραφία ως συντελεστής έκπτωσης. Ένας συντελεστής έκπτωσης μπορεί να θεωρηθεί μια παράμετρος μάθησης, που ποικίλει από $0 \leq \gamma \leq 1$. Κάθε μελλοντική ανταμοιβή κατά τη χρονική στιγμή t μειώνεται κατά γ^{k-1}. Όσο το γ πλησιάζει το μηδέν, ο πράκτορας θεωρεί τις κοντινές στην παρούσα κατάσταση ανταμοιβές πολύ πιο πολύτιμες. Αντίθετα, αν το γ προσεγγίζει τη μονάδα ο πράκτορας μεταβάλλεται ώστε να συμπεριφέρεται άπληστα και να θεωρεί τις μελλοντικές ανταμοιβές εξίσου σημαντικές. Η εκπτωτική συνολική ανταμοιβή ονομάζεται και Συνάρτηση Επιστροφής (G_t) κι ορίζεται ως:

$$G_t = \sum_{k=0}^{T} (\gamma^k r_{t+k+1}) \quad (2.26)$$

Ιστορικό / Κατάσταση

Ως ιστορικό ορίζεται μια αλληλουχία παρατηρήσεων O_t, πράξεων A_t και ανταμοιβών R_t, όπως φαίνεται παρακάτω:

$$H_t = O_1, R_1, A_1, \ldots, A_{t-1}, O_t, R_t \quad (2.27)$$

Ως κατάσταση S_t ορίζεται η πληροφορία που χρησιμοποιείται για να καθοριστεί τι θα συμβεί μελλοντικά και συγκεκριμένα ποια πράξη θα επιλέξει ο πράκτορας. Μια κατάσταση λοιπόν είναι συνάρτηση ιστορικού:

$$S_t = f(H_t) \quad (2.28)$$

Η κατάσταση του περιβάλλοντος στην πλειονότητα των περιπτώσεων δεν είναι γνωστή στον πράκτορα, αλλά ακόμη κι αν έχει πλήρη γνώση είναι πιθανόν να περιέχει περιττή πληροφορία.

2.3.3 Συναρτήσεις βελτίωσης συμπεριφοράς

Υπάρχουν πολλά στοιχεία που συνθέτουν τους αλγορίθμους ενισχυτικής μάθησης, τα οποία παρουσιάζονται παρακάτω.

2.3.3.1 Πολιτική (policy)

Η πολιτική, π, χαρτογραφεί τις αντιληπτές καταστάσεις του περιβάλλοντος στις ενέργειες που λαμβάνονται σε αυτές τις καταστάσεις. Μια πολιτική μπορεί να είναι ντετερμινιστική, να ακολουθεί δηλαδή τη λογική ενός πίνακα αναζήτησης όταν διασφαλίζεται η εκτέλεση μιας ενέργειας, ή στοχαστική, όταν εξετάζεται μια συγκεκριμένη πιθανότητα.

Ως εκ τούτου, μια ντετερμινιστική πολιτική αντιπροσωπεύεται μαθηματικά από τα ακόλουθα:

$$\pi(s_t) = \alpha_t , \qquad s_t \subset D \quad \alpha_t \subset A \tag{2.29}$$

Εάν οι ενέργειες είναι στοχαστικές, η πολιτική μετατρέπεται σε κατανομή πιθανοτήτων α_t δοσμένου του αντίστοιχου s_t. Έτσι, οι πληροφορίες πρέπει να συμπεριληφθούν στην πολιτική όπως φαίνεται παρακάτω:

$$\pi(\alpha_t \mid s_t) = p_i , \qquad s_t \subset D \quad \alpha_t \subset A \quad 0 \leq p_i \leq 1 \tag{2.30}$$

Η πολιτική που συγκεντρώνει το μεγαλύτερο ποσό ανταμοιβών για ένα συγκεκριμένο κόσμο, θεωρείται βέλτιστη και συμβολίζεται με π^*. Μετά την εφαρμογή ενός αλγορίθμου μάθησης σε συνδυασμό με μια κατάλληλη στρατηγική εξερεύνησης μέχρι τη σύγκλιση, δεδομένων επαρκών επεισοδίων, η πολιτική που αποκτάται μπορεί να θεωρηθεί βέλτιστη ή μη-βέλτιστη.

2.3.3.2 Συνάρτηση ανταμοιβής (reward function)

Μια συνάρτηση ανταμοιβής, $R_t(s_t, a_t)$ καθορίζει το στόχο στο πρόβλημα ενισχυτικής μάθησης. Χαρτογραφεί κάθε αντιληπτή κατάσταση ή ζεύγος κατάστασης-ενέργειας σε έναν μόνο αριθμό που υποδηλώνει την εγγενή σκοπιμότητα επίτευξης μιας συγκεκριμένης κατάστασης. Ανάλογα με το πρόβλημα, οι ανταμοιβές μπορεί να είναι αραιές, και έτσι παρουσιάζεται μεγάλη ανταμοιβή κατά την επίτευξη της τερματικής κατάστασης ,ενώ είναι μηδέν αλλού.

2.3.3.3 Μαρκοβιανές Διαδικασίες Αποφάσεων (MDP)

Στη γενική περίπτωση προβλήματος, η ανταμοιβή που αποκτάται από μια ενέργεια εξαρτάται από έναν συνδυασμό της προηγούμενης κατάστασης, της δράσης και της ανταμοιβής που οδηγεί στην κατάσταση αυτή. Έτσι, προσδιορίζοντας ολόκληρη τη διανομή ή το ιστορικό των βημάτων που συνέβησαν πριν ο πράκτορας φτάσει στην τρέχουσα κατάσταση, μπορεί να υπολογιστεί η δυναμική του συστήματος.

$$\Pr \{ s_{t+1} = s' , R_{t+1} = R' \mid s_0, a_0, R_1 \dots R_t , s_t , a_t \} \tag{2.31}$$

Για έναν αλγόριθμο εκμάθησης ενίσχυσης, είναι βοηθητική η ύπαρξη ενός σήματος κατάστασης που συνοψίζει τις παρελθοντικές λειτουργίες συμπυκνώνοντας παράλληλα όλες τις σχετικές πληροφορίες. Μια κατάσταση που διατηρεί επιτυχώς όλες τις παραπάνω πληροφορίες λέγεται ότι έχει την ιδιοκτησία Markov. [16] Υπό αυτό το πρίσμα, δεδομένης κατάστασης s και ενέργειας a, η πιθανότητα κάθε ζεύγους επόμενης κατάστασης και ανταμοιβής (s', r) δίνεται από τον παρακάτω τύπο:

$$\Pr \{ s_{t+1} = s' , R_{t+1} = R' \mid s_0, a_0, R_1 \dots R_t , s_t , a_t \} = \Pr \{ s_{t+1} = s' , R_{t+1} = r \mid s_t , a_t \} \tag{2.32}$$

Η παραπάνω εξίσωση ισχύει για κάθε s_t, a_t, s', R. Εάν ισχύει η ιδιότητα Markov για την κατάσταση, τότε το περιβάλλον και η διαδικασία στο σύνολό της ονομάζονται MDP (Markov Decision Process). Η δυναμική του ενός βήματος ενός τέτοιου προβλήματος επιτρέπει την πρόβλεψη της επόμενης κατάστασης και της αναμενόμενης ανταμοιβής από την επόμενη κατάσταση γνωρίζοντας μόνο την τρέχουσα κατάσταση και δράση. Με απλά λόγια, μια κατάσταση για να είναι Μαρκοβιανή, θα πρέπει το μέλλον να είναι ανεξάρτητο από το παρελθόν δεδομένου του παρόντος.

Μια MDP μπορεί να προσδιοριστεί σαν μία αλυσίδα πέντε στοιχείων <S, A, P_a, R_a, γ> ,όπου [19], [16]:

- S – ένα σετ καταστάσεων
- A – ένα σετ πράξεων
- $P_a(s,s')$ – η πιθανότητα του συστήματος σε κατάσταση s στον χρόνο t να φτάσει σε κατάσταση s' στον χρόνο t+1 όταν πραγματοποιεί ενέργεια a
- $R_a(s,s')$ – η άμεση ανταμοιβή που αποκτάται μετά την εναλλαγή από την κατάσταση s στην s'

- γ – ένας παράγοντας έκπτωσης που δίνει προτεραιότητα στη σημασία της άμεσης και μελλοντικής ανταμοιβής, με γ ϵ [0,1]

Η παραπάνω αλυσίδα ορίζεται ως Μαρκοβιανή Διαδικασία Αποφάσεων. Οι MDPs μπορούν να γενικευθούν σε μερικώς παρατηρήσιμες MDPs (ή POMDPs) στις οποίες η δυναμική του συστήματος παρατηρείται άμεσα από τον πράκτορα, αλλά οι υποκείμενες καταστάσεις δεν μπορούν να παρατηρηθούν άμεσα.

2.3.3.4 Συνάρτηση αξίας (value function)

Η τιμή μιας κατάστασης είναι η αναμενόμενη τιμή της συνολικής μειωμένης ανταμοιβής την οποία ένας πράκτορας μπορεί να ελπίζει να συσσωρεύσει στο μέλλον, όταν ακολουθεί μια συγκεκριμένη πολιτική ξεκινώντας από αυτή την κατάσταση. Οι τιμές λαμβάνουν υπόψη τη συσσώρευση ανταμοιβών από τις καταστάσεις που είναι πιθανό να ακολουθήσουν. Στην πραγματικότητα, οι συναρτήσεις αξίας βοηθούν τον πράκτορα να δώσει προτεραιότητα στη μακροπρόθεσμη συλλογή ανταμοιβών. Η συνάρτηση αξίας σε μια κατάσταση s όταν ακολουθείτε μια πολιτική π ορίζεται ως [20]:

$$V : V_\pi \rightarrow R, \quad V_\pi(s) = E_\pi[G_t \,|s_t = s] = E_\pi[\sum_{k=0}^{\infty}(\gamma^k R_{t+k+1})|\, s_t = s] \tag{2.33}$$

όπου το G_t αντιπροσωπεύει το κέρδος ή την απόδοση, δηλαδή τις μειωμένες ανταμοιβές που συσσωρεύονται με την πάροδο του χρόνου, και το $E_\pi[\cdot]$ αναπαριστά την αναμενόμενη αξία μιας τυχαίας μεταβλητής, δεδομένου ότι ο πράκτορας ακολουθεί την πολιτική π, με το t να είναι το χρονικό βήμα. Το επιτόκιο έκπτωσης για τις ανταμοιβές (γ ∈ [0, 1]) προσδιορίζει την παρούσα αξία των μελλοντικών ανταμοιβών. Ένα επιτόκιο έκπτωσης κοντά στο μηδέν ότι εξετάζονται μόνο οι ανταμοιβές του κοντινού μέλλοντος. Ενώ, επιτόκιο έκπτωσης πιο κοντά στη μονάδα σημαίνει ότι λαμβάνονται υπόψη ανταμοιβές και του πιο μακροπρόθεσμου μέλλοντος. Μια θεμελιώδης ιδιότητα των συναρτήσεων αξίας στην ενισχυτική μάθηση είναι η εγγενώς αναδρομική φύση. Για μια πολιτική, π, και οποιαδήποτε κατάσταση, s, η συνάρτηση αξίας ορίζεται με γνώμονα τις πιθανές διάδοχες καταστάσεις του ως ένας αναδρομικός κανόνας που δίδεται από:

$$\begin{aligned} V_\pi(s) &= E_\pi[G_t \,|s_t = s] = E_\pi[\sum_{k=0}^{\infty}(\gamma^k R_{t+k+1})|\, s_t = s] \\ &= E_\pi[(\, R_{t+1} + \gamma\sum_{k=0}^{\infty}(\gamma^k R_{t+k+2})|\, s_t = s] \\ &= \sum_\alpha \pi(\alpha|s) \sum_{s',r}(P_a(s,s')[R + \gamma V_\pi(s')]) \end{aligned} \tag{2.34}$$

2.3.3.5 Συνάρτηση ενέργειας-αξίας (action-value function)

Οι τιμές Q είναι εκτιμήσεις των αναμενόμενων αποδόσεων για κάθε πράξη που εκτελείται από μια μη τερματική κατάσταση. Η αξία της ανάληψης μιας δράσης α σε μια κατάσταση s, και μετά την εφαρμογή πολιτικής π, αντιπροσωπεύεται ως:

$$Q_\pi(s, \alpha) = E_\pi[G_t \,|s_t = s, \alpha_t = \alpha] = E_\pi[\sum_{k=0}^{\infty}(\gamma^k R_{t+k+1})|\, s_t = s, \alpha_t = \alpha] \tag{2.35}$$

Οι Q-τιμές διαθέτουν επίσης αναδρομικές ιδιότητες παρόμοιες με τη συνάρτηση αξίας.

2.3.3.6 Συνάρτηση πλεονεκτήματος (advantage function)

Η έννοια της συνάρτησης πλεονεκτήματος ορίζεται ώστε να προωθεί προς τα εμπρός τις διαφορές μεταξύ των ανταμοιβών που αναμένονται από την ανάληψη διαφορετικών ενεργειών από την ίδια κατάσταση. Ως εκ τούτου, η συνάρτηση πλεονεκτήματος είναι ένα μέτρο με το οποίο η αναμενόμενη αξία της ανάληψης συγκεκριμένης δράσης είναι διαφορετική από την αναμενόμενη αξία της ανάληψης δράσης που επί του παρόντος θεωρείται βέλτιστη. [21], [22]

$$A(s, \alpha) = (Q(s, \alpha) - V(s)) = (Q(s,\alpha) - \max_{\alpha'} Q(s,\alpha')) \tag{2.36}$$

Εξ ορισμού, σε μια συγκεκριμένη κατάσταση, η τιμή του πλεονεκτήματος ,όταν λαμβάνεται η βέλτιστη πράξη από μια κατάσταση, είναι μηδέν.

Εάν $\alpha = \text{argmax}_\alpha A^*(s,\alpha)$, με το $A^*(s,\alpha)$ να είναι το βέλτιστο πλεονέκτημα της συνάρτησης στα (s, α):

$$A^*(s, \alpha) = (Q^*(s,\alpha) - \max_\alpha Q^*(s,\alpha)) = (\max_\alpha Q^*(s,\alpha) - \max_\alpha Q^*(s,\alpha)) = 0 \qquad (2.37)$$

2.3.3.7 Τελεστής Bellman (Bellman operator)

Οι τελεστές Bellman είναι "χειριστές" οι οποίοι αντιστοιχίζουν το ένα σημείο σε ένα άλλο μέσα σε έναν διανυσματικό χώρο τιμών καταστάσεων, R^N. Η επανεγγραφή των εξισώσεων Bellman ως τελεστών είναι χρήσιμη για την απόδειξη ότι ορισμένοι δυναμικοί αλγόριθμοι προγραμματισμού (π.χ. επανάληψη πολιτικής, επανάληψη αξίας) συγκλίνουν σε ένα μοναδικό σταθερό σημείο. [20] Όταν πρόκειται για αλγορίθμους επανάληψης αξίας (value iteration), όπως οι Q-learning , ο τελεστής Bellman τ χρησιμεύει σαν ένα σύστημα ένδειξης της επανάληψης, χωρίς να είναι απαραίτητος ο καθορισμός των δεικτών για τον αριθμό επανάληψης.

Ο τελεστής Bellman $\tau Q \rightarrow Q$ ορίζεται κατά σημείο ως:

$$\tau Q(s, a) = R(s, a) + \gamma E_{Pa} \max_{\alpha' \epsilon A} Q(s' , a') \qquad (2.38)$$

,με τον όρο $E_{Pa}[]$ αναφερόμαστε στην προσδοκία σχετικά με τη συνάρτηση πιθανότητας μετάβασης $P_\alpha(s,s')$ από την κατάσταση s στην s'.

Ο τελεστής Bellman συγκλίνει στις βέλτιστες τιμές όταν πληρούνται από ορισμένες προϋποθέσεις από τον πράκτορα, οι οποίες παρουσιάζονται παρακάτω:

- Άπειρη εξερεύνηση, δηλαδή, κάθε ζεύγος κατάστασης-δράσης επισκέπτεται απείρως συχνά ο πράκτορας.
- Συντελεστής έκπτωσης $\gamma < 1$.
- Συνθήκες Robbins-Monro [23]: $\sum_{t=0}^{\infty} \alpha_t(s,\alpha) = \infty$ & $\sum_{t=0}^{\infty} \alpha_t^2(s,\alpha) < \infty$,όπου α_t είναι ο ρυθμός εκμάθησης την χρονική στιγμή t.

2.4 Exploration-exploitation dilemma

Το δίλημμα εξερεύνησης-εκμετάλλευσης και το ποσοστό με το οποίο πρέπει η καθεμία να εφαρμόζεται σε ένα πρόβλημα ενισχυτικής μάθησης αποτελεί σημαντικό αντικείμενο μελέτης. Οι πράκτορες πρέπει να εξερευνήσουν για να βελτιώσουν την κατάσταση που ενδεχομένως αποφέρει υψηλότερες ανταμοιβές στο μέλλον ή να εκμεταλλευτούν την κατάσταση που αποφέρει την υψηλότερη ανταμοιβή με βάση τις υπάρχουσες γνώσεις για το περιβάλλον. [24]

Η εξερεύνηση μπορεί να περιγραφεί ως προσπάθεια ανακάλυψης νέων δυνατοτήτων σχετικά με το περιβάλλον εκτελώντας ακόμη και μη βέλτιστες ενέργειες. Η κεντρική ιδέα είναι να εκτελεστεί μια ενέργεια διαφορετική από την εμπειρία του παρελθόντος που υποδεικνύει τη σωστή απόφαση δεδομένου ενός συγκεκριμένου περιβάλλοντος. Έτσι, η διερεύνηση της ανεξερεύνητης δράσης οδηγεί σε δύο αποτελέσματα: είτε σε μειωμένη ανταμοιβή, δηλαδή τιμωρία (αρνητική αξία ανταμοιβής), είτε σε αύξηση της αξίας ανταμοιβής. Η εξερεύνηση είναι απαραίτητη επειδή μη βέλτιστες ενέργειες στο ίδιο περιβάλλον μπορεί να επιφέρουν αυξημένη ανταμοιβή. Από την άλλη πλευρά, η εκμετάλλευση είναι η έννοια της επανειλημμένης εκτέλεσης των ίδιων ενεργειών στο ίδιο περιβάλλον, εφόσον αυτές είναι γνωστό ότι συνεισφέρουν στην τρέχουσα μέγιστη ανταμοιβή. [25] Ως εκ τούτου, η ιδέα είναι αντίθετη από την εξερεύνηση, η οποία καταλήγει στο συμπέρασμα ότι η αύξηση της εκμετάλλευσης οδηγεί σε φθίνουσα εξερεύνηση. Έτσι, ο πράκτορας συναντά το δίλημμα εάν θα διερευνήσει ανεξερεύνητες ενέργειες ή θα εκμεταλλευτεί την εμπειρία του παρελθόντος. Στην πράξη, η εκμετάλλευση δείχνει ότι η λήψη αποφάσεων του πράκτορα βασίζεται σε μια πολιτική που έχει μάθει σε ένα περιβάλλον. Μία από τις κύριες προκλήσεις της ενίσχυσης της μάθησης είναι ο συμβιβασμός εξερεύνησης και εκμετάλλευσης. [24] Η εφαρμογή αμιγούς εξερεύνησης υποβαθμίζει τη μάθηση του πράκτορα, ωστόσο αυξάνει την ευελιξία του να προσαρμόζεται σε

ένα δυναμικό περιβάλλον. Από την άλλη, μια αμιγής εκμετάλλευση οδηγεί τη διαδικασία μάθησης του πράκτορα σε τοπικά βέλτιστες λύσεις. [26] Έχουν μελετηθεί διάφορες πολιτικές μάθησης για την αντιμετώπιση αυτού του ζητήματος. Ένας πράκτορας θέλει να μεγιστοποιήσει την ανταμοιβή του για κάθε ενέργεια που αναλαμβάνει - να εκμεταλλευτεί την εμπειρία του παρελθόντος - αλλά αυτό έχει ως στόχο να μειώσει την εξερεύνηση νέων δράσεων για τη βελτίωση της γνώσης σχετικά με τη διαδικασία δημιουργίας ανταμοιβών. Όταν ένας πράκτορας αυξάνεται στην εξερεύνηση δεν χρειάζεται να μεγιστοποιήσει την ανταμοιβή του.

2.4.1 Στρατηγικές εξερεύνησης

Ζήτημα είναι να εξερευνηθεί ο πιο ικανοποιητικός τρόπος που πετυχαίνει τον στόχο και παράλληλα να συνεχίζει η εκμετάλλευση μιας συγκεκριμένης ενέργειας. Η εξερεύνηση ωστόσο είναι αρκετά δύσκολη. Χωρίς κατάλληλες λειτουργίες ανταμοιβής, οι αλγόριθμοι θα ακολουθούν αέναα την «ουρά» τους. Με την πάροδο των ετών, πολλές στρατηγικές εξερεύνησης έχουν διατυπωθεί με την ενσωμάτωση μαθηματικών προσεγγίσεων. Παρακάτω, παρατίθενται δημοφιλείς στρατηγικές εξερεύνησης που χρησιμοποιούνται στα μοντέλα εκμάθησης ενίσχυσης.

Epsilon-Greedy Method

Ο Epsilon-Greedy (ε-greedy) είναι ο πιο κοινός και απλούστερος αλγόριθμος για την εξισορρόπηση του συμβιβασμού μεταξύ εξερεύνησης και εκμετάλλευσης επιλέγοντάς τες τυχαία. Υποθέτουμε ότι εξερευνούμε τις n επιλογές στο αρχικό στάδιο. Με τον ε-greedy αλγόριθμο, το (1- ε) τοις εκατό του χρόνου εκμεταλλευόμαστε άπληστα την καλύτερη επιλογή k μεταξύ των επιλογών n, ενώ το υπόλοιπο ε τοις εκατό του χρόνου διερευνώνται τυχαία οι υπόλοιπες επιλογές για μια καλύτερη απόφαση από την προηγουμένως καλύτερη επιλογή k. Η τιμή του ε ορίζεται συνήθως στο 10%. [25]

Epsilon Decreasing Method

Η προσαύξηση του epsilon είναι παρόμοια με την ε-greedy μέθοδο. Το ε στη μέθοδο ε-greedy παραμένει σταθερό, ενώ στη μέθοδο μείωσης του epsilon η τιμή του μειώνεται σταδιακά με την πάροδο του χρόνου. Ο αριθμός των νέων επιλογών που διερευνώνται μειώνεται σταδιακά με τη μείωση της αξίας ε και συνεπώς η καλύτερη επιλογή γίνεται όλο και πιο βέβαιη στη διαδικασία. [25]

Εξερεύνηση με βάση την περιέργεια (Curiosity based Exploration)

Σε πολλά σενάρια του πραγματικού κόσμου, οι εξωγενείς ανταμοιβές στον πράκτορα είναι σημαντικά αραιές ή απουσιάζουν εντελώς. Σε τέτοιες περιπτώσεις, η περιέργεια χρησιμεύει ως ένα εγγενές σήμα ανταμοιβής για να επιτρέψει στον πράκτορα να εξερευνήσει το περιβάλλον του και να αποκτήσει εμπειρίες που θα του φανούν χρήσιμες στην μετέπειτα «ζωή» του. Η περιέργεια στα μοντέλα RL παρουσιάστηκε για πρώτη φορά από τον Dr Juergen Schmidhuber το 1991, μέσω ενός πλαισίου «curious» νευρικών ελεγκτών, το οποίο περιγράφει πώς ένας συγκεκριμένος αλγόριθμος μπορεί να οδηγηθεί από περιέργεια και πλήξη. Αυτό γίνεται εισάγοντας (καθυστερημένη) ενίσχυση για δράσεις που αυξάνουν τη γνώση του μοντέλου δικτύου για τον κόσμο. Υπό αυτό το πλαίσιο, το πρότυπο δίκτυο πρέπει να διαμορφώσει τη δική του άγνοια, δείχνοντας έτσι μια στοιχειώδη μορφή αυτοεπαναστατικής συμπεριφοράς. [26], [27]

Ανώτατο όριο εμπιστοσύνης (Upper confidence bound)

Το άνω όριο εμπιστοσύνης ορίζεται ως εξής: $Q^{t(\alpha)} + U^{t(\alpha)}$, όπου $Q^{t(\alpha)}$ είναι οι μέσες ανταμοιβές που προκύπτουν από τη δράση α μέχρι το χρόνο t και $U^{t(\alpha)}$ είναι μια συνάρτηση αντιστρόφως ανάλογη με τον πόσες φορές έχει πραγματοποιηθεί μια ενέργεια α. Η μεγιστοποίηση του

ανώτατου ορίου εμπιστοσύνης είναι μια στρατηγική που χρησιμοποιούν οι πράκτορες για να κινηθούν προς την επίτευξη του στόχου τους. [28]

Εξερεύνηση Boltzmann (Boltzmann Exploration)

Ο πράκτορας αντλεί τις ενέργειές του από μια κατανομή boltzmann (softmax) πάνω από τις τιμές Q, που ρυθμίζονται από μια παράμετρο θερμοκρασίας. [28] Στο πλαίσιο της ενισχυτικής μάθησης, τα εκθετικά συστήματα στάθμισης χρησιμοποιούνται ευρέως για την εξισορρόπηση της εξερεύνησης και της εκμετάλλευσης και αναφέρονται αντίστοιχα ως πολιτικές εξερεύνησης Boltzmann, Gibbs ή softmax. Στην πιο κοινή εκδοχή της εξερεύνησης του Boltzmann, η πιθανότητα επιλογής μιας δράσης είναι ανάλογη με την εκθετική συνάρτηση του εμπειρικού μέσου ανταμοιβής που προκύπτει από την δράση αυτή, η οποία προκύπτει ως εξής:

$$p_{t,i} \propto e^{\eta_t \widehat{\mu}_{t,i}}$$

, όπου $p_{t,i}$ είναι η πιθανότητα να επιλεχθεί η πράξη i τη χρονική στιγμή t, $\mu_{t,i}$ ο εμπειρικός μέσος των ανταμοιβών που αποκτώνται από την πράξη i μέχρι την χρονική στιγμή t και $\eta_t>0$ είναι ο παράγοντας εκμάθησης.

Δειγματοληψία Thompson (Thompson sampling)

Ο πράκτορας παρακολουθεί την πεποίθηση σχετικά με την πιθανότητα βέλτιστων ενεργειών και δειγμάτων από αυτή τη διανομή. Συνεχίζοντας να διερευνά εναλλακτικές δράσεις, ένας πράκτορας να κερδίζει υψηλότερες ανταμοιβές στο μέλλον διασφαλίζοντας ότι δεν έχει προσκολληθεί σε μια στρατηγική και εξερευνώντας άλλες ενώ εκμεταλλεύεται μία. Η δειγματοληψία Thompson είναι ένας αλγόριθμος για προβλήματα διαδικτυακών αποφάσεων όπου οι ενέργειες λαμβάνονται διαδοχικά με τρόπο που πρέπει να εξισορροπηθεί μεταξύ της εκμετάλλευσης αυτού που είναι γνωστό ότι μεγιστοποιεί την άμεση απόδοση και της επένδυσης/εξερεύνησης για τη συσσώρευση νέων πληροφοριών που μπορούν να βελτιώσουν τις μελλοντικές επιδόσεις. [28] Ο παράγοντας παρακολουθεί την πιθανότητα βέλτιστων ενεργειών και δειγμάτων από αυτή τη διανομή.

Σε κάθε βήμα, επιλέγεται μια ενέργεια χρησιμοποιώντας την ακόλουθη πιθανολογική συνάρτηση:

$$\pi(\alpha|h_t) = P[Q(\alpha)>Q(\alpha') , \forall\ \alpha'\neq\alpha \mid h_t] = E_{R|h_t}[1(\alpha = \text{argmax}_{\alpha\epsilon A}Q(\alpha))] \qquad (2.39)$$

Όπου $\pi(\alpha|h_t)$ είναι η πιθανότητα ανάληψης δράσης, δεδομένου του ιστορικού h_t. Εμπνευσμένα από τη δειγματοληψία Thompson, τα δίκτυα Bootstrapped DQN εισήγαγαν μια έννοια αβεβαιότητας στην προσέγγιση των τιμών της Q στο κλασικό DQN μοντέλο χρησιμοποιώντας τη μέθοδο bootstrapping. Το Bootstrapping επιστρατεύεται για να προσεγγιστεί μια κατανομή δειγματοληπτώντας με αντικατάσταση από τον ίδιο πληθυσμό πολλές φορές και στη συνέχεια να συγκεντρώνοντας τα αποτελέσματα.

2.5 Ταξινόμηση RL

Οι αλγόριθμοι ενισχυτικής μάθησης μπορούν να κατηγοριοποιηθούν με διαφορετικά κριτήρια. Ένα από αυτά στηρίζεται στο αν ο εκάστοτε αλγόριθμος βασίζεται σε κάποιο μοντέλο περιβάλλοντος κατά τη διάρκεια της διαδικασίας μάθησης. Έτσι, οι αλγόριθμοι ταξινομούνται σε αυτούς που η μάθηση βασίζεται σε κάποιο μοντέλο (model-based learning) και σε όσους μαθαίνουν άνευ μοντέλου (model-free learning).

2.5.1 Model-based learning

Τα συστήματα ενισχυτικής μάθησης που βασίζονται σε κάποιο μοντέλο συλλέγουν πληροφορίες σχετικά με τις μεταβάσεις από το περιβάλλον, οι οποίες αξιοποιούνται για την παραγωγή ενός μοντέλου που θα προβλέπει τα μελλοντικά βήματα. Με αυτόν τον τρόπο, ο πράκτορας έχει τη δυνατότητα να εκτελεί εικονικές εξερευνήσεις στον χώρο κατάστασης-ενέργειας και να αποφασίσει για μια φαινομενικά βέλτιστη πορεία δράσης από την τρέχουσα κατάσταση του πράκτορα. Για τους παραπάνω αλγορίθμους χρησιμοποιείται δυναμικός προγραμματισμός (dynamic programming) , ο οποίος υπολογίζει τις βέλτιστες πολιτικές σε μια Μαρκοβιανή Διαδικασία Αποφάσεων. [29] Χρησιμοπούνται, λοιπόν, συναρτήσεις αξίας (value functions) για την οργάνωση του χώρου των πολιτικών, με στόχο την αποδοτική αναζήτηση των βέλτιστων από αυτές.

2.5.1.1 Δυναμικός Προγραμματισμός

Θεμέλια του δυναμικού προγραμματισμού αποτελούν η αξιολόγηση πολιτικής (policy evaluation) και η βελτίωση πολιτικής (policy improvement). Η αξιολόγηση πολιτικής συνεπάγεται τον επαναλαμβανόμενο υπολογισμό των συναρτήσεων αξίας. Η παραπάνω διαδικασία συναντάται στην βιβλιογραφία ως πρόβλημα πρόβλεψης (prediction problem). [30] Η συνάρτηση αξίας κατάστασης V_π , δεδομένης πολιτικής π, υπολογίζεται από την εξίσωση Bellman ως εξής:

$$V_\pi(s) = \sum_a \pi(s,a) \sum_{s'} (P_{ss'}^{a}[R_s^{a} + \gamma V_k(s')]) \text{ , } s \epsilon S \qquad (2.40)$$

Δεδομένης μιας συνάρτησης αξίας στόχος αποτελεί η βελτίωση της πολιτικής. Ειδικότερα, για συγκεκριμένη κατάσταση s διερευνάται αν είναι συμφέρουσα η τροποποίηση της εκάστοτε πολιτικής ώστε να επιλεγεί με ντετερμινιστικό τρόπο ενέργεια α = π(s). Υπολογίζεται λοιπόν η αξία επιλογής της δράσης α στην τρέχουσα κατάσταση s και έπειτα εφαρμόζεται η τρέχουσα πολιτική. Η συνάρτηση τιμών ενέργειας υπολογίζεται ως εξής:

$$Q_\pi(s,\alpha) = \sum_{s'} (P_{ss'}^{a}[R_s^{a} + \gamma V_\pi(s')]) \qquad (2.41)$$

Χρησιμοποιώντας τις παραπάνω εξισώσεις πρέπει να εξεταστεί αν ισχύει $Q_\pi(s,\alpha) \geq V_\pi(s)$. Αν επαληθεύεται αυτή η σχέση συμπεραίνεται ότι θα ήταν καλύτερη η επιλογή της α κάθε φορά που ο πράκτορας βρίσκεται στην κατάσταση s, και ότι η νέα πολιτική είναι στη γενική περίπτωση πιο αποδοτική από την προηγούμενη. Γενικότερα, σύμφωνα με το θεώρημα βελτίωσης πολιτικής (policy improvement theorem) για κάθε πιθανό ζεύγος ντετερμινιστικών πολιτικών π, π’ εάν ισχύει:

$$Q_\pi(s,\alpha) \geq V_\pi(s), \ \forall s \in S, \ \text{τότε } V_{\pi'} \geq V_\pi \qquad (2.42)$$

,δηλαδή αν υπάρχει απόφαση α για την κατάσταση s, ώστε ακολουθώντας την α όταν βρισκόμαστε στην s και την πολιτική π για κάθε άλλη κατάσταση, να έχουμε καλύτερη συνάρτηση αξιολόγησης για το σύνολο των καταστάσεων, τότε λαμβάνουμε ως νέα πολιτική την π’ και θα ισχύει ότι η πολιτική π’ θα είναι καλύτερη από την π και συνεπώς η εφαρμογή της επιφέρει μεγαλύτερη συνολική ανταμοιβή.

Συνδυάζοντας τις δύο προαναφερθέντες μεθόδους, προκύπτουν η επανάληψη ως προς την πολιτική (policy iteration) και η επανάληψη ως προς την αξία (value iteration), οι οποίες έχοντας πλήρης γνώση του μοντέλου περιβάλλοντος και για πεπερασμένες Μαρκοβιανές διαδικασίες, υπολογίζουν τις βέλτιστες συναρτήσεις αξίας και πολιτικές

Policy Iteration

Η επανάληψη ως προς την πολιτική πραγματοποιείται με την συνεχή εναλλαγή αξιολόγησης πολιτικής με βελτίωση πολιτικής, μέχρι να φτάσει το μοντέλο στην σύγκλιση. Έτσι, θα βρεθεί

η βέλτιστη πολιτική π*. Παρακάτω ορίζεται ο αλγόριθμος «Policy Iteration» που περιγράφει αυτήν την διαδικασία. [31]

Algorithm 1: Dynamic Programming – Policy Iteration

For each sϵS, an arbitrary policy πϵA(s) is defined
For each sϵS, $V_\pi(s) = 0$
Repeat #evaluation policy
  δ ← 0
  for each sϵS:
    old_value ← V(s)
    $V(s) \leftarrow r(s, \pi(s)) + \gamma\sum_{s'}(P_{ss'}^{\pi(s)}V(s'))$
    δ ← max(δ, |old_value – V(s)|)
  until δ < ε
  policy_state ← True #policy improvement
  For each sϵS:
    β ← π(s)
    $\pi(s) \leftarrow \arg_\alpha\max\{ r(s, \alpha) + \gamma\sum_{s'}(P_{ss'}^{\alpha}V(s')) \}$
    if β ≠ π(s) then policy_state ← False
    if policy_state then EXIT
Output π(s) for each sϵS

Η παράμετρος ε επηρεάζει αρκετά την ταχύτητα σύγκλισης, σύμφωνα με το παρακάτω μοτίβο. Αν είναι αρκετά μικρό ώστε να προσεγγίζει το 0, η ακολουθία των $V_n^{\pi i}(s)$ θα είναι κατά ε κοντά στην πραγματική τιμή $V^{\pi i}(s)$ για κάθε s. Παρόλα αυτά, μια τέτοια μέθοδος συνεπάγεται μεγάλο υπολογιστικό κόστος καθώς σε κάθε επανάληψη εκτελείται ένα βήμα αξιολόγησης πολιτικής. Αν πάλι το ε είναι αρκετά μεγάλο τότε ενδέχεται σε κάποια επανάληψη να μην βελτιώνει τις τιμές $V_n^{\pi i}(s)$, sϵS.

Value Iteration

Σε αντίθεση με την παραπάνω μέθοδο, η επανάληψη ως προς την αξία παρουσιάζει σημαντική βελτίωση, εφόσον προσεγγίζει επαναληπτικά τη βέλτιστη συνάρτηση V, χωρίς να περιμένει να επιτευχθεί η ε-σύγκλιση. Παρουσιάζει ικανοποιητικό αποτέλεσμα έστω και από μια ανανέωση της $V_\kappa^{\pi i}(s)$ για μια πολιτική π_i. Πραγματοποιεί λοιπόν μόνο μία ενημέρωση των τιμών της συνάρτησης αξίας, σύμφωνα με τον παρακάτω αλγόριθμο:

$$V_{k+1}^{\pi} = \max\{ r(s, \alpha) + \gamma\sum_{s'}(P_{ss'}^{\alpha}V_\kappa^{\pi i}(s')) \}, s\epsilon S \qquad (2.43)$$

, όπου V_{k+1} η εκτίμηση της συνάρτησης αξίας στο βήμα k+1. Έτσι, θα βρεθεί η βέλτιστη V^* και συνεπώς και η βέλτιστη πολιτική π*. Παρακάτω ορίζεται ο αλγόριθμος «Value Iteration» που περιγράφει αυτήν την διαδικασία. [31]

Algorithm 2: Dynamic Programming – Value Iteration

For each sϵS, $V_\pi(s) = 0$
ε ← 0,001
Repeat #evaluation policy
  δ ← 0
  for each sϵS:
    old_value ← V(s)
    $V(s) \leftarrow \max\{ r(s, \alpha) + \gamma\sum_{s'}(P_{ss'}^{\alpha}V(s')) \}$
    δ ← max(δ, |old_value – V(s)|)
  until δ < ε
Output $\pi(s)$ for each sϵS , so that: $\pi(s) = \arg_\alpha\max\{ r(s, \alpha) + \gamma\sum_{s'}(P_{ss'}^{\alpha}V(s')) \}$

Αξίζει να αναφερθεί πως μέσω του δυναμικού προγραμματισμού ενημερώνονται οι εκτιμήσεις για τις αξίες των καταστάσεων με βάση εκτιμήσεις για τις αξίες των μελλοντικών καταστάσεων. Η πρακτική αυτή ενημέρωσης εκτιμήσεων διάδοχων καταστάσεων βάσει άλλων

καταστάσεων ονομάζεται bootstapping και έχει μεγάλη υπολογιστική πολυπλοκότητα, καθώς οι τιμές της συνάρτησης αξιολόγησης υπολογίζονται για το ολόκληρο το σύνολο των καταστάσεων. Ενώ όταν εφαρμόζονται μη ακριβή μοντέλα οδηγούν σε μειωμένες επιδόσεις.

2.5.2 Model-free learning

Σε μια τέτοια προσέγγιση δεν είναι απαραίτητη η εκμάθηση κάποιου μοντέλου για την παραγωγή εμπειριών, ενώ πρωταρχικός στόχος είναι η εκμάθηση βελτιώνοντας μια πολιτική συμπεριφοράς που μεγιστοποιεί ένα αριθμητικό σήμα ανταμοιβών. [30] Μιας και αυτή η προσέγγιση δεν απαιτεί ενδιάμεσο βήμα για των υπολογισμό των εικονικών εμπειριών, δεν έχουν τόσο μεγάλο υπολογιστικό κόστος και μαθαίνουν απευθείας από τα επεισόδια των εμπειριών.

2.5.2.1 Αξιολόγηση πολιτικής άνευ μοντέλου

2.5.2.1.1 Monte-Carlo

Οι μέθοδοι που βασίζονται στο μοντέλο Monte-Carlo (MC) μαθαίνουν απευθείας από πλήρη επεισόδια εμπειρίας (sample episodes) και χαρακτηρίζονται από την απλότητα στην κατανόηση και στη χρήση τους. Εφαρμόζονται μόνο σε εργασίες επεισοδίων, όπου υπάρχει σαφής τερματική κατάσταση και θα μπορούσε να ξεκινήσει την επαναφορά της εκάστοτε εργασίας. Με απλά λόγια, για να λειτουργήσει ο αλγόριθμος, τα επεισόδια πρέπει να τερματίζουν. Οι μέθοδοι MC στην πλειοψηφία τους έχουν υψηλές διακυμάνσεις, μηδενικό bias και διασφαλίζουν θεωρητικά την σύγκλιση. Ένας Monte-Carlo αλγόριθμος ενημερώνει την συνάρτηση αξίας $V(s_t)$ προς την πραγματική απόδοση G_t. Δεν απαιτεί πλήρη γνώση του περιβάλλοντος, σε αντίθεση με τον δυναμικό προγραμματισμό, αλλά το μοντέλο περιβάλλοντος είναι απαραίτητο μιας και από αυτό απαιτείται η δυνατότητα παραγωγής δειγμάτων μεταβάσεων (ακολουθιών καταστάσεων, ενεργειών, ανταμοιβών). Η αξία λοιπόν μιας κατάστασης παράγεται ως εξής:

$$V(s_t) \leftarrow V(s_t) + \alpha[G_t - V(s_t)] \tag{2.44}$$

Όσον αφορά την αξιολόγηση πολιτικής, κάθε εμφάνιση μιας κατάστασης s σε ένα επεισόδιο, καλείται επίσκεψη στην κατάσταση. Η $V_\pi(s)$ υπολογίζεται από την μέθοδο Monte-Carlo, ως μέσος όρος των επιστροφών όλων των επισκέψεων στην s σε ένα σύνολο επεισοδίων. Η μέθοδος Monte-Carlo πρώτης επίσκεψης, συμπεριλαμβάνει στον υπολογισμό της μέσης τιμής μόνο τις επιστροφές των πρώτων επισκέψεων στην κατάσταση s. Παρακάτω παρουσιάζεται ο αλγόριθμος αξιολόγησης πολιτικής Monte-Carlo. [33]

Algorithm 3: Monte-Carlo evaluation policy

```
For each sϵS, αϵA(s):
   π(s) ← policy to be evaluated
   Vπ(s) arbitrarily defined
   Returns(s,α) ← an empty list, for all s ϵ S
   Repeat:
      Generate an episode with random initial state using π
      For each (s,α) appearing in the episode:
         R ← return following the first occurrence of (s,α)
         Append R to Returns(s,α)
      V(s) ← average(Returns(s))
   Until convergence condition = True
```

Η συνθήκη σύγκλισης θα πρέπει να είναι τέτοια ώστε να εξασφαλίζει πως οι τελικές εκτιμήσεις είναι αμερόληπτες και η τυπική απόκλιση αρκετά μικρή.

2.5.2.1.2 Temporal difference learning (TD)

Η μέθοδος χρονικών διαφορών (TD) αποτελεί θεμελιώδη ιδέα που χρησιμοποιείται από τους αλγορίθμους ενισχυτικής μάθησης, η οποία συνδυάζει τα πλεονεκτήματα του δυναμικού προγραμματισμού και των μεθόδων Monte-Carlo. Συνοψίζοντας όσα έχουν ήδη αναφερθεί, για την Monte Carlo, η οποία προϋποθέτει προβλήματα πεπερασμένου χρόνου, η πραγματοποίηση πλήθους επεισοδίων γίνεται με την παρατήρηση των ανταμοιβών και καταστάσεων, χωρίς απαραιτήτως την γνώση του μοντέλου του περιβάλλοντος, ώστε μετά το πέρας ενός αριθμού επεισοδίων να είναι διαθέσιμο ένα δείγμα αθροιστικών αμοιβών για κάθε ζεύγος (s,α), και συνεπώς να προκύψει μία εκτίμηση της Q(s ,α) από τον αντίστοιχο δειγματικό μέσο. Από την άλλη, ο Δυναμικός Προγραμματισμός προϋποθέτει πλήρη γνώση του μοντέλου του περιβάλλοντος, αλλά μπορεί να εφαρμοστεί σε προβλήματα πεπερασμένου και απείρου χρονικού ορίζοντα. Εμπλέκοντας την τιμή της συνάρτησης αξιολόγησης μίας κατάστασης με κάθε άλλη τιμή της συνάρτησης αξιολόγησης της αμέσως επόμενης κατάστασης (Bootstrap), συνεπάγεται υψηλό υπολογιστικό κόστος. Τα συστήματα μάθησης χρονικών διαφορών (TD) μαθαίνουν από ελλιπή/ανολοκλήρωτα επεισόδια δίνοντας βάση στις υφιστάμενες εκτιμήσεις (μέχρι τη χρονική στιγμή εκείνη) και ενημερώνοντας μια εικασία προς μια άλλη. Όπως και στις μεθόδους Monte-Carlo δεν απαιτείται πλήρη γνώση του περιβάλλοντος όπου δρα ο πράκτορας, παρά μόνο εμπειρία αλληλεπίδρασης με αυτό. [34] Ωστόσο, σε αντιδιαστολή με αυτές λειτουργούν και σε περιβάλλοντα που δεν τερματίζουν, εφόσον μπορούν να μαθαίνουν χωρίς γνώση της οριστικής κατάστασης, από ελλιπείς ακολουθίες. Οι μέθοδοι TD έχουν την δυνατότητα μάθησης πριν την γνώση του τελικού αποτελέσματος, αλλά και online έπειτα από κάθε βήμα. [34] Στα πλαίσια μιας TD μεθόδου, ως παρατήρηση-δείγμα λαμβάνει από ένα μοναδικό βήμα t την αμοιβή r_t και με βάση αυτήν ανανεώνεται και η συνάρτηση αξίας για την κατάσταση που βρισκόταν (σταθμισμένος μέσος/Monte-Carlo), ενώ λαμβάνει υπόψη και τις ήδη αποθηκευμένες τιμές της συνάρτησης αξίας των καταστάσεων που μεταπηδά (bootstrap/Δυναμικός Προγραμματισμός). [34] Συνήθως, παρουσιάζουν χαμηλή διακύμανση και κάποια bias, και έτσι είναι στατιστικά πιο αποτελεσματικές από τις μεθόδους MC και πιο ευαίσθητες στις αρχικές τιμές.

2.5.2.1.2.1 Μέθοδος Χρονικών Διαφορών TD(0)

Η απλούστερη μέθοδος της οικογένειας των αλγορίθμων μάθησης με χρονικές διαφορές αποτελεί η TD(0). Λειτουργεί ενημερώνοντας την τρέχουσα εκτίμηση της συνάρτησης αξίας $V_t(s_t)$ στο άθροισμα της παρατηρήσιμης ανταμοιβής R_{t+1} και της μειωμένης εκτιμώμενης συνάρτησης αξίας του επόμενου βήματος $V_t(s_{t+1})$, χρησιμοποιώντας την παρακάτω αναδρομική εξίσωση [34]:

$$V_t(s_t) \leftarrow V_t(s_t) + \alpha[R_{t+1} + \gamma V_t(s_{t+1}) - V_t(s_t)] =>$$

$$V_t(s_t) \leftarrow V_t(s_t) + \alpha[R_{t+1} + \gamma V_t(s_{t+1})] - \alpha V_t(s_t) =>$$

$$V_t(s_t) \leftarrow (1-\alpha)V_t(s_t) + \alpha[R_{t+1} + \gamma\, V_t(s_{t+1})] \quad (2.45)$$

Η μεταβλητή α συμβολίζει το βήμα ανανέωσης τιμών και ανήκει στο διάστημα $\alpha \in [0,1]$, ενώ το γ είναι ο εκπτωτικός παράγοντας. Η ποσότητα $G_t^{(1)} = R_{t+1} + \gamma\, V_t(s_{t+1})$ καλείται στόχος «ενός βήματος» (one-step target), ενώ η $\delta(t) = R_{t+1} + \gamma\, V_t(s_{t+1}) - V_t(s_t)$ ορίζεται σαν σφάλμα TD «ενός βήματος» (one-step TD error), εφόσον η ενημέρωση πραγματοποιείται στον στόχο που έχει αποκτηθεί εκτελώντας μόνο ένα βήμα. Ο τελευταίος τύπος δείχνει ότι η νέα τιμή της συνάρτησης αξίας προκύπτει ως σταθμισμένος μέσος της παλιάς τιμής και του στόχου (target) με βάρη (1-α) και α αντίστοιχα, που ορίζονται από το βήμα ανανέωσης. Μικρότερη τιμή του βήματος α συνεπάγεται μικρότερη επιρροή της νέας παρατήρησης στην V(s). Παρακάτω παρουσιάζεται ο αλγόριθμος αξιολόγησης πολιτικής Χρονικών Διαφορών (TD) [34]:

Algorithm 4: Temporal Difference (0) evaluation policy

For each sϵS, αϵA(s):
  α ← step size ϵ (0,1]
  $V_\pi(s)$ arbitrarily defined for policy π
  For each one of N episodes $s_{initial} = s$
    Repeat:
      α ← π(s)
      observation of R, s'
      V(s) ← V(s) + α[R + γV(s') – V(s)]
      s ← s'
    until $s = s_{terminal}$

Οι αλγόριθμοι Monte-Carlo και TD(0) συγκλίνουν στην ίδια τελικά βέλτιστη λύση, ωστόσο παρουσιάζουν μια διαφορά ως προς τα αποτελέσματα μέχρι την οριστική ταύτισή τους. Ενώ η Monte-Carlo αναζητά εκτίμηση που να ελαχιστοποιεί το μέσο τετραγωνικό σφάλμα (mst), η TD(0) αναζητά την εκτίμηση που είναι κατάλληλη για το μοντέλο μέγιστης πιθανοφάνειας της Μαρκοβιανής διαδικασίας.

2.5.2.1.2.2 Μέθοδος Χρονικών Διαφορών TD(λ)

Οι μέθοδοι χρονικών διαφορών με παράμετρο λ αποτελούν συνδυασμό της TD(0) και Monte Carlo. Για να εκτιμηθεί η συνάρτηση αξιολόγησης ο αλγόριθμος TD(λ) χρησιμοποιεί τις άμεσες αμοιβές μιας απόφασης προσάπτοντάς τους διαφορετικά βάρη. Αφενός, η μέθοδος Monte-Carlo αναμένει το τέλος του κάθε επεισοδίου οπότε θα έχει πλήρη γνώση των αμοιβών των καταστάσεων που έχουν προσπελαστεί, και με βάση αυτές εκτιμώνται οι τιμές της συνάρτησης αξιολόγησης. Αφετέρου, η TD(0) λαμβάνει ως παρατήρηση μόνο την άμεση ανταμοιβή μιας κατάστασης και με βάση αυτή ενημερώνεται η τιμής της συνάρτησης αξιολόγησης της κατάστασης που μόλις προσπελάστηκε. Η αξιολόγηση πολιτικής TD(λ) χρησιμοποιεί την λογική των ιχνών επιλεξιμότητας (eligibility traces). [35] Ενσωματώνοντας ίχνη επιλεξιμότητας στις μεθόδους χρονικών διαφορών, αυτές λαμβάνουν χαρακτηριστικά των Monte-Carlo, όπως η ανοχή σε διαδικασίες μάθησης που δεν είναι Μαρκοβιανές.

Παρατηρώντας τις n άμεσες αμοιβές της παρούσας κατάστασης $s_t = s$, ανανεώνουμε τις τιμές της συνάρτησης αξιολόγησης ορίζοντας ως στόχο(επιστροφή) n βημάτων το άθροισμα για τα n επόμενα βήματα συν την εκάστοτε εκτίμηση για τη συνάρτηση αξιολόγησης της κατάστασης s_{t+n}:

$$G_t^{(n)} = R_t + \gamma R_{t+1} + \dots + \gamma^{n-1} R_{t+n-1} + \gamma^n R(S_{t+n}) \text{ , } n \leq T\text{-}t \qquad (2.46)$$

Για κάθε n > T-t ο στόχος ταυτίζεται με τον στόχο της μεθόδου Monte-Carlo ως εξής: [36]

$$G_t^{(n)} = R_t + \gamma R_{t+1} + \dots + \gamma^{n-1} R_{t+n-1} + \gamma^n R_T \text{ , } n > T\text{-}t \qquad (2.47)$$

Ενώ η μέθοδος των χρονικών διαφορών n ακολουθεί τον παρακάτω αλγόριθμο: [35]

$$V(S_t) \leftarrow V(S_t) + \alpha(G_t^{(n)} - V(S_t)) \text{ , } n \leq T\text{-}t \qquad (2.48)$$

Ως λ-επιστροφή (λ-return), ορίζεται ο βεβαρημένος μέσος όρος όλων των στόχων n-βημάτων, ως εξής:

$$G_t^{\lambda} = (1-\lambda)\sum_{n=1}^{\infty}(\lambda^{n-1} G_t^{(n)}) \qquad (2.49)$$

Συμβολίζουμε το ίχνος της κατάστασης s_t την χρονική στιγμή t, με $e_t(s) \in R^+$. Σε κάθε χρονικό βήμα, οι καταστάσεις φθίνουν κατά παράγοντα γλ, εκτός του ίχνους της τρέχουσας κατάστασης του πράκτορα, το οποίο αυξάνεται κατά 1. Για κάθε κατάσταση sϵS με λ ϵ [0,1] ισχύουν οι παρακάτω σχέσεις: [35]

$$e_t(s) = \gamma\lambda e_{t-1}(s) \quad , s \neq s_1 \qquad (2.50)$$

$$e_t(s) = \gamma\lambda e_{t-1}(s) + 1 \ , s = s_1 \quad (2.51)$$

Τα ίχνη επιλεξιμότητας εκφράζουν κάθε χρονική στιγμή ποιες καταστάσεις έχει επισκεφτεί ο πράκτορας σε όρους γλ. Ενώ, όσο το λ τείνει στη μονάδα, η μέθοδος των χρονικών διαφορών TD(λ) προσεγγίζει τη συμπεριφορά των μεθόδων Monte-Carlo. Η ανανέωση των τιμών της συνάρτησης αξίας γίνεται ως εξής:

$$V_{t+1}(s_t) = V_t(s_t) + \alpha[R(s_{t+1}) + \gamma V_t(s_{t+1}) - V_t(s_t)]e_t(s_t) \quad (2.52)$$

Ο αλγόριθμος αξιολόγησης πολιτικής χρονικών διαφορών TD(λ), παρουσιάζεται παρακάτω: [36]

Algorithm 5: Temporal Difference (λ) evaluation policy

For each sϵS:
  V(s) arbitrarily defined
  e(s) = 0
  For each one of N episodes
    $s_{initial}$ = s
    Repeat:
      α ← π(s)
      implementation of α, observation of R, s'
      δ ← R + γV(s') – V(s)
      e(s) ← e(s) + 1
      for each s:
        V(s) ← V(s) + αδe(s)
        e(s) ← γλe(s)
      s ← s'
    until s = $s_{terminal}$

2.5.2.2 Έλεγχος άνευ μοντέλου

Ο έλεγχος άνευ μοντέλου στοχεύει στη βελτιστοποίηση της συνάρτησης τιμών, όταν δεν μπορεί να χρησιμοποιηθεί η Μαρκοβιανή διαδικασία απόφασης. Μια τέτοια μέθοδος εκμεταλλεύεται δείγματα από εμπειρίες που έχουν ληφθεί από το περιβάλλον. Πρέπει λοιπόν όλες οι δυνατές ενέργειες να επιλέγονται αέναα, διαδικασία που επιτυγχάνεται ακολουθώντας τις παρακάτω μεθόδους:

- *On-policy*: μέθοδος που προσπαθεί να αξιολογήσει ή να βελτιώσει την πολιτική που χρησιμοποιείται για να παρθούν οι αποφάσεις. Είναι ιδιαίτερα χρήσιμη μέθοδος όταν επιθυμούμε να βελτιστοποιήσουμε την τιμή ενός πράκτορα που εξερευνεί. [37]
- *Off-policy*: μέθοδος που αξιολογεί ή βελτιώνει μια πολιτική διαφορετική από αυτήν που χρησιμοποιείται για να παραχθούν τα δεδομένα. Η μέθοδος αυτή χρησιμοποιείται όταν ο πράκτορας δεν εξερευνεί αρκετά το περιβάλλον. [16]

2.5.2.2.1 Έλεγχος Monte-Carlo

Χρησιμοποιεί όπως και στον Δυναμικό Προγραμματισμό την λογική της γενικευμένης επανάληψης με βάση την πολιτική (generalized policy iteration/ GPI), στην οποία επιχειρείται η προσέγγιση τόσο της καλύτερης εκάστοτε πολιτικής βάση της τρέχουσας συνάρτησης τιμών όσο και η συνεχής ανανέωση της συνάρτησης τιμών ώστε να προσεγγίζει την τρέχουσα. Η βελτιστοποίηση της πολιτικής επιτυγχάνεται κάνοντάς την άπληστη σε σχέση με την τρέχουσα συνάρτηση τιμών χωρίς περεταίρω γνώση του περιβάλλοντος. Επιλέγεται ως βέλτιστη πολιτική εκείνη που για κάθε συνάρτηση πράξης-αξίας Q, αποφασίζει την πράξη που επιφέρει την μέγιστη τιμή της Q. [33]

2.5.2.2.1.1 On-policy έλεγχος Monte-Carlo

Η μέθοδος αυτή για να εκτιμήσει μια πολιτική, ακολουθεί μια ελαφρώς τροποποιημένη στοχαστική, ώστε να προσεγγιστεί η βέλτιστη. Παρακάτω παρουσιάζεται μια ε-greedy πολιτική , με ε ϵ (0,1), η οποία επιζητά την καλύτερη δυνατή απόφαση, δηλαδή την ενέργεια με τη μέγιστη εκτιμώμενη αξία, χωρίς να λαμβάνει υπόψιν άλλες παραμέτρους. Μια greedy ντετερμινιστική πολιτική ορίζεται από την σχέση $\pi(s) = \max_{\alpha}\{Q_{\pi}(s,\alpha)\} = \alpha^{*}$, και η ε-greedy στοχαστική πολιτική ορίζεται βάση αυτής και έχει ίσες πιθανότητες επιλογής $\frac{\varepsilon}{A(s)}$ για όλες τις μη άπληστες ενέργειες και πιθανότητα $(1 - \varepsilon + \frac{\varepsilon}{A(s)})$ δίνεται στην άπληστη ενέργεια. Παρακάτω παρουσιάζεται ο αλγόριθμος Monte-Carlo on-policy (ε-greedy policy): [33]

Algorithm 6: On-policy Monte-Carlo control

For each sϵS, αϵA(s):
  Q(s,α) ← 0 (arbitrarily defined)
  π(s) ← policy to be evaluated, ε-greedy
  Returns(s,α) ← an empty list, for all s ϵ S
  Repeat:
    Generate an episode using policy π
    G ← 0
    For each (s,α) appearing in the episode:
      G ← G + R_{t+1}
      Append G to Returns(s,α)
      Q(s,α) ← mean{ Returns(s,α) }
      α^{*} = argmax_{α}Q(s,α)
      for each α ϵ A(s):
        if α = α^{*}:
          $\pi(s,\alpha) = 1 - \varepsilon + \frac{\varepsilon}{|A(s)|}$
        elsif α ≠ α^{*}:
          $\pi(s,\alpha) = \frac{\varepsilon}{|A(s)|}$
  Until convergence condition = True

Οι τελικές εκτιμήσεις του παραπάνω αλγορίθμου πρέπει να δίνουν αμερόληπτο και αξιόπιστο αποτέλεσμα, ώστε να επέλθει η σύγκλιση. Η συνθήκη ,λοιπόν, σύγκλισης πρέπει να επιλέγεται με αυτό το κριτήριο.

2.5.2.2.1.2 Off-policy έλεγχος Monte-Carlo

Σε αντίθεση με την on-policy η οποία βελτιώνει μια συγκεκριμένη πολιτική με το πέρας κάθε επεισοδίου και είναι η ίδια πολιτική που ακολουθεί για τη λήψη αποφάσεων, η off-policy μέθοδος εκτιμά και ανανεώνει τις τιμές της συνάρτησης αξιολόγησης ($V_{\pi}(s)$) μιας πολιτικής π, ενώ για την παραγωγή των επεισοδίων ακολουθεί μια άλλη σταθερή πολιτική π'. Η πολιτική που ακολουθείτε καλείται πολιτική συμπεριφοράς και εκείνη που αξιολογείται και βελτιώνεται, πολιτική εκτίμησης. Με αυτή την μέθοδο η τελευταία πολιτική μπορεί να είναι ντετερμινιστική όταν η πολιτική συμπεριφοράς δοκιμάζει κάθε πιθανή ενέργεια.
Για να λειτουργήσει η παραπάνω μέθοδος, δηλαδή να εκτιμηθούν με αξιοπιστία οι τιμές της συνάρτησης $V_{\pi}(s)$ ακολουθώντας την πολιτική π', πρέπει κάθε απόφαση που λαμβάνεται όντας σε κατάσταση s με πολιτική π να έχει μη μηδενική πιθανότητα να ληφθεί και από την π'. Η συνθήκη αυτή περιγράφεται από τον παρακάτω τύπο:

$$\pi(s,\alpha) > 0 \Rightarrow \pi'(s,\alpha) \tag{2.53}$$

Παρακάτω παρουσιάζεται αναλυτικά ο αλγόριθμος: [33]

Algorithm 7: Off-policy Monte-Carlo control

For each sϵS, αϵA(s):
  Q(s,α) ← 0 (arbitrarily defined)

$N(s,\alpha) \leftarrow 0$
$\pi(s) \leftarrow \text{argmax}_{\alpha} Q(s,\alpha)$
Repeat:
  $\pi' \leftarrow$ behavioral policy
  Generate an episode using policy π'
  $G \leftarrow 0$
  $W \leftarrow 1$
  For each (s,α) appearing in the episode:
    $G \leftarrow \gamma G + R_{t+1}$
    $N(s_t,\alpha_t) \leftarrow N(s_t,\alpha_t) + W$
    $Q(s_t,\alpha_t) \leftarrow Q(s_t,\alpha_t) + \frac{W}{C(st,at)} [\, G - Q(s_t,\alpha_t) \,]$
    $\pi(s_t) \leftarrow \text{argmax}_{\alpha} Q(s_t,\alpha)$
    if $\alpha_t \neq \pi(s_t)$:
      EXIT the repeat
    $W \leftarrow W \frac{1}{\pi'(st,at)}$

Αξίζει να αναφερθεί πως όταν η πολιτική εκτίμησης π είναι ντετερμινιστική ενδέχεται μα παράγονται επεισόδια που δεν είναι χρήσιμα στην εκτίμηση και αυτό διότι η μέθοδος off-policy κόβει το επεισόδιο και συλλέγει δεδομένα μόνο από το τέλος του. Για να αντιμετωπιστεί αυτό το φαινόμενο μπορεί να χρησιμοποιηθεί μια στοχαστική πολιτική εκτίμησης π.

2.5.2.2.2 Έλεγχος Χρονικών Διαφορών TD

Ο έλεγχος Χρονικών διαφορών TD ακολουθεί μια διαδικασία που καλείται «ενημέρωση παρτίδας». Σύμφωνα με αυτήν, κατά την εκπαίδευση του μοντέλου επαναλαμβάνονται οι ίδιες εμπειρίες μέχρι να επέλθει η σύγκλιση, εφόσον ο αριθμός τους είναι πεπερασμένος και συνεπώς η ενημέρωση της συνάρτησης τιμών V γίνεται κάθε φορά που ο πράκτορας επισκέπτεται μια μη τερματική κατάσταση, ως εξής:

$$V(s_t) \leftarrow V(s_t) + \alpha[G_t - V(s_t)] =>$$

$$V(s_t) \leftarrow V(s_t) + \alpha[R_{t+1} + \gamma\, V(s_{t+1})] - V(s_t)] \tag{2.54}$$

Υπό το παραπάνω πρίσμα, ένας αλγόριθμος TD(0) θα συγκλίνει με ντετερμινιστικό τρόπο σε μια λύση που θα είναι ανεξάρτητη της παραμέτρου α.

2.5.2.2.2.1 On-policy έλεγχος TD

Αλγόριθμος Sarsa

Ο αλγόριθμος αυτός βελτιώνει την πολιτική σύμφωνα με την οποία αλληλεπιδρά και με το περιβάλλον και από την οποία παράγονται οι ανταμοιβές και οι επόμενες παρατηρήσεις. Αναζητά, λοιπόν, την βέλτιστη πολιτική προσανατολίζοντας την εκμάθηση όχι στην συνάρτηση κατάστασης-αξίας V(s) αλλά εκτιμώντας την συνάρτηση αξιολόγησης καταστάσεων-αποφάσεων Q(s,α) για συγκεκριμένη πολιτική π και για όλες τις καταστάσεις s και δράσεις α. Η τιμή της Q(s,α) ανανεώνεται έπειτα από κάθε μετάβαση σε μια μη τερματική κατάσταση, ενώ όταν η s είναι τερματική τότε η Q θεωρείται μηδενική. Ο κανόνας ανανέωσης ακολουθεί τον παρακάτω αλγόριθμο:

$$Q(s_t,\alpha_t) \leftarrow Q(s_t,\alpha_t) + \alpha[R_{t+1} + \gamma Q(s_{t+1}, \alpha_{t+1}) - Q(s_t,\alpha_t)] \tag{2.55}$$

, όπου το $(R_{t+1} + \gamma Q(s_{t+1}, \alpha_{t+1}))$ είναι ο στόχος(target). [38]

Ο αλγόριθμος Sarsa αξιοποιεί στην διαδικασία ενημέρωσης κάθε στοιχείο από το σύνολο εμπειριών $<s_t, \alpha_t, s_{t+1}, \alpha_{t+1}>$ και από εκέι έχει κληρονομήσει και το όνομά του αν αναλογιστούμε ότι λαμβάνει υπόψη και την άμεση αμοιβή (reward-R_t). Η πολιτική συγκλίνει στην ε-greedy με το ε να τείνει στο 0. Παρακάτω παρουσιάζεται ο αλγόριθμος για την εν λόγω μέθοδο:

Algorithm 8: Sarsa (on-policy TD) control

For each sϵS, αϵA(s):
  Q(s,α) ϵ R (arbitrarily defined)
  $Q(s_{terminal},*) = 0$
  π(s) ← arbitrary, ε-greedy policy
  For each episode repeat:
    Initialization of S
    select a decision α for statement s following ε-greedy policy
    Q(s,α) ← Q(s,α) + α[R + γQ(s', α') - Q(s,α)]
    s ← s'
    α ← α'
  until $s = s_{terminal}$
until convergence condition = True

Αλγόριθμος Sarsa(λ)

Αντίστοιχα με τον Sarsa και ο Sarsa(λ) είναι ένας on-policy αλγόριθμος που χρησιμοποιεί τα ίχνη επιλεξιμότητας του TD(λ), καθώς εκτιμώντας τις τιμές της Q(s,α) για κάθε sϵS και αϵA χρειάζονται τα ίχνη $e_t(s,α)$ για τις χρονικές στιγμές t για κάθε ζεύγος (s,α). Τα ίχνη επιλεξιμότητας εκφράζουν το ποσό της επιρροής μιας ανταμοιβής στην ανανέωση της τιμής της συνάρτηση αξιολόγησης n βήματα πριν την τρέχουσα κατάσταση. Η ανανέωση της Q πραγματοποιείται σύμφωνα με την παρακάτω εξίσωση:

$$Q(s_t,\alpha_t) \leftarrow Q(s_t,\alpha_t) + \alpha[R_{t+1} + \gamma Q(s_{t+1}, \alpha_{t+1}) - Q(s_t,\alpha_t)]\, e_t(s,\alpha) \qquad (2.56)$$

,όπου τα ίχνη επιλεξιμότητας ορίζονται ως εξής: $e_t(s,\alpha) = (\gamma\lambda)\cdot e_{t-1}(s,\alpha) + I_{sst} \cdot I_{\alpha\alpha t}$

,με $I_{sst} = 1$ για $s_t = s$ και $I_{\alpha\alpha t} = 1$ για $\alpha_t = \alpha$. [38]

Ο αλγόριθμος επίλυσης του παραπάνω προβλήματος περιγράφεται παρακάτω: [34]

Algorithm 9: Sarsa(λ) (on-policy TD) control

For each sϵS, αϵA(s):
  Q(s,α) ϵ R (arbitrarily defined)
  $Q(s_{terminal},*) = 0$
  For each episode repeat:
    For each sϵS, αϵA(s): e(s,α) = 0
    Initialization of s, α
    For each episode step repeat:
      Implementation of α, Observe R, s'
      select a decision α' for statement s' following ε-greedy policy
      target ← R + γQ(s', α') - Q(s,α)
      e(s,α) ← e(s,α) + 1
      For each (s,α) ϵ episode:
        Q(s,α) ← Q(s,α) + α·target·e(s,α)
        e(s,α) ← (γλ) · e(s,α)
      s ← s'
      α ← α'
    until $s = s_{terminal}$
  until convergence condition = True

2.5.2.2.2.2 Off-policy έλεγχος TD

Q-learning

Ο αλγόριθμος Q-learning αναζητά την βέλτιστη πολιτική π*, η οποία όμως δεν ακολουθείται στην παραγωγή των επεισοδίων. Η εκπαιδευόμενη συνάρτηση Q προσεγγίζει απευθείας την ιδανική συνάρτηση πράξης-αξίας Q^*, χωρίς να ενδιαφέρεται για την τρέχουσα πολιτική. [39] Το ζεύγος κατάστασης-πράξης που θα επιλεχθεί από το μοντέλο για επίσκεψη και ανανέωση,

επηρεάζεται από την πολιτική. [40] Ενώ, κρίσιμο για την σύγκλιση της Q με πιθανότητα 1 στην βέλτιστη Q^* ,είναι να διασφαλιστεί η συνεχής ανανέωση όλων των ζευγών. [41]

Η ανανέωση των τιμών της Q γίνεται σύμφωνα με τον παρακάτω αλγόριθμο:

$$Q(s_t,\alpha_t) \leftarrow Q(s_t,\alpha_t) + \alpha[R_{t+1} + \gamma \max_{\alpha'} Q(s_{t+1}, \alpha') - Q(s_t,\alpha_t)] \quad (2.57)$$

Ο αλγόριθμος Q-learning παρουσιάζεται παρακάτω: [34]

Algorithm 10: Q-learning (off-policy TD) control

For each sϵS, αϵA(s):
  Q(s,α) ϵ R (arbitrarily defined)
  $Q(s_{terminal},*) = 0$
  π(s) ← arbitrary, ε-greedy policy
  For each episode repeat:
    Initialization of S
    select a decision α for statement s following ε-greedy policy
    for each episode step repeat:
      implementation of α, observation of R, s'
      $Q(s,\alpha) \leftarrow Q(s,\alpha) + \alpha[R + \gamma \max_{\alpha'} Q(s', \alpha') - Q(s,\alpha)]$
      s ← s'
  until $s = s_{terminal}$
until convergence condition = True

Q(λ)-learning

Ο αλγόριθμος αυτός εκμεταλλευόμενος τα ίχνη επιλεξιμότητας ακολουθεί πολιτική π για την εκτίμηση της συνάρτησης αξιολόγησης Q, η οποία τελικά συγκλίνει στην βέλτιστη Q^*, μέσω της οποίας προκύπτει και η βέλτιστη greedy πολιτική π'. [42] Η πολιτική π που ακολουθείται στη διάρκεια του επεισοδίου είναι soft ή ε-greedy για να διασφαλιστεί ότι θα προσπελαστεί κάθε πιθανή κατάσταση ακόμη κι αν οι αποφάσεις που λαμβάνονται δεν είναι οι βέλτιστες ως προς τις εκτιμήσεις της εκάστοτε Q (exploration). Αντίθετα, για την ανανέωση των τιμών, οι ανταμοιβές που προκύπτουν θα ακολουθούν μια αλληλουχία greedy αποφάσεων από τις οποίες θα προκύψει και η τελική βέλτιστη greedy πολιτική (exploitation). Υπό αυτό το πλαίσιο, τα ίχνη επιλεξιμότητας είναι ιδιαίτερα χρήσιμα, διότι μηδενίζονται κάθε φορά που λαμβάνεται μια exploratory απόφαση και όπως προκύπτει από τον παρακάτω τύπο, δεν πραγματοποιείται ανανέωση της τιμής της Q:

$$Q_{t+1}(s,\alpha) \leftarrow Q_t(s,\alpha) + \alpha \cdot target_t \cdot e_t(s,\alpha) \quad , \forall\ (s,\alpha) \epsilon \text{ episode} \quad (2.58)$$

,όπου $target_t = R_{t+1} + \gamma \max_{\alpha'} Q_t(s', \alpha') - Q_t(s,\alpha)$ [42]

Τα ίχνη επιλεξιμότητας ορίζονται ως εξής: $e_t(s,\alpha) = (\gamma\lambda) \cdot e_{t-1}(s,\alpha) \cdot I_{\alpha^*\alpha} + I_{sst} \cdot I_{\alpha\alpha t}$ (2.59)

με $I_{sst} = 1$ για $s_t = s$

$I_{\alpha\alpha t} = 1$ για $\alpha_t = \alpha$ και

$I_{\alpha^*\alpha t} = 1$ για $\alpha_t = \alpha^*$, με α^* η βέλτιστη πολιτική όπως προκύπτει από την Q.

Ο αλγόριθμος επίλυσης του παραπάνω προβλήματος περιγράφεται παρακάτω: [42]

Algorithm 11: Q(λ)-learning (off-policy TD) control

For each sϵS, αϵA(s):
  Q(s,α) ϵ R (arbitrarily defined)
  $Q(s_{terminal},*) = 0$
  For each episode repeat:
    For each sϵS, αϵA(s): e(s,α) = 0

Initialization of s, α
For each episode step repeat:
Implementation of α, Observe R, s'
select a decision α' for statement s' following ε-greedy policy
$\alpha^* \leftarrow \text{argmax}_{\pi'}\{ Q(s', \pi') \}$
target ← $R + \gamma Q(s', \alpha^*) - Q(s,\alpha)$
$e(s,\alpha) \leftarrow e(s,\alpha) + 1$
For each (s,α) ϵ episode:
$Q(s,\alpha) \leftarrow Q(s,\alpha) + \alpha \cdot \text{target} \cdot e(s,\alpha)$
If $\alpha' = \alpha^*$ then $e(s,\alpha) \leftarrow (\gamma\lambda) \cdot e(s,\alpha)$
else $e(s,\alpha) \leftarrow 0$
$s \leftarrow s'$
$\alpha \leftarrow \alpha'$
until $s = s_{terminal}$
until convergence condition = True

3 Deep reignforcement learning

Η Βαθιά Ενισχυτική Μάθηση (Deep Reinforcement Learning) αναφέρεται στην ενίσχυση της μάθησης με βαθιά νευρωνικά δίκτυα ως συναρτήσεις προσέγγισης. Η ενίσχυση της μάθησης με νευρωνικά δίκτυα επιτρέπει την εκμάθηση ενός μεγάλου φάσματος συναρτήσεων αποφάσεων και έχει ως συνέπεια προσεγγιστικές συναρτήσεις που είναι προσαρμοσμένες εν γένει να αντιμετωπίζουν αποτελεσματικά συνεχείς και μεγάλους χώρους καταστάσεων. Οι πιο επιτυχημένες προσεγγίσεις εκπαιδεύονται άμεσα από τις ακατέργαστες εισόδους, χρησιμοποιώντας ελαφριές ενημερώσεις με βάση τη στοχαστική κάθοδο διαβάθμισης (stochastic gradient descent). Τροφοδοτώντας επαρκή δεδομένα σε βαθιά νευρωνικά δίκτυα, είναι συνήθως πιο πιθανή η εκμάθηση καλύτερων αναπαραστάσεων.

Η αρχιτεκτονική TD-Gammon του Tesauro [43] αποτέλεσε σημείο εκκίνησης για μια τέτοια προσέγγιση. Η παραπάνω αρχιτεκτονική ενημερώνει τις παραμέτρους ενός δικτύου που υπολογίζει τη συνάρτηση τιμής Q, απευθείας από on-policy δείγματα εμπειρίας: s_t, a_t, r_t, s_{t+1}, a_{t+1}, που προέρχονται από την αλληλεπίδραση του αλγορίθμου με το περιβάλλον. Σε αντίθεση με το TD-Gammon, οι σύγχρονες μέθοδοι Βαθιάς Ενισχυτικής Μάθησης χρησιμοποιούν την τεχνική της επανάληψη εμπειρίας (experience replay) σύμφωνα με την οποία οι εμπειρίες του πράκτορα αποθηκεύονται σε κάθε βήμα, $e_t = (s_t, a_t, r_t, s_{t+1})$ σε ένα σύνολο δεδομένων $D = e_1$, ..., e_N , όπου συγκεντρώνονται πολλά επεισόδια σε μια μνήμη επανάληψης (replay memory/buffer). [44] Κατά τη διάρκεια του εσωτερικού βρόχου του αλγορίθμου, εφαρμόζονται ενημερώσεις Q-learning, ή ενημερώσεις minibatch, σε δείγματα εμπειρίας, e∼D, που τυχαία επιλέχθηκε από τη «δεξαμενή» αποθηκευμένων δειγμάτων. Μετά την εκτέλεση επανάληψης εμπειρίας, ο εκάστοτε πράκτορας επιλέγει και εκτελεί μια ενέργεια σύμφωνα με μια ε-άπληστη πολιτική.

3.1 Experience replay

Στην μέθοδο επανάληψης εμπειρίας ο πράκτορας αποθηκεύει το διάνυσμα εμπειρίας (s, α, r, s') στον buffer επανάληψης D. Μια έκδοση της μεθόδου αυτής αποθηκεύει τις τελευταίες N μεταβάσεις σε μια βάση δεδομένων «συρόμενου παραθύρου», ενώ μια άλλη έκδοση αποθηκεύει όλες τις εμπειρίες. [44] Σε κάθε βήμα ανανέωσης, ο πράκτορας δειγματοληπτεί ομοιόμορφα μια μίνι-παρτίδα (mini batch) εμπειρίας από τον buffer και τη χρησιμοποιεί για να ενημερωθούν τα βάρη του δικτύου τιμών Q_t. Εφόσον η σειρά δειγματοληψίας είναι τυχαία, αποφεύγονται οι συσχετίσεις μεταξύ των διαδοχικών δειγμάτων. Επιπλέον, τα δείγματα εντός μιας μίνι παρτίδας αξιολογούνται με την ίδια Q_t και έτσι βαθμολογούνται ομοίως. Αξίζει να σημειωθεί πως οι εμπειρίες μπορούν, θεωρητικά, να δειγματοληφθούν αρκετές φορές πριν εγκαταλείψουν τον buffer(μνήμη) και έτσι σπάνιες εμπειρίες επαναχρησιμοποιούνται περισσότερες από μία φορές, γεγονός ιδιαίτερα πολύτιμο εάν ορισμένες εμπειρίες είναι δαπανηρές. [45]

Παρόλα αυτά, εφόσον στην επανάληψη εμπειρίας τα δείγματα εμπειρίας ανασύρονται από τον buffer ομοιόμορφα, τα κοινά δείγματα εμπειρίας λαμβάνονται συχνότερα, μιας και εμφανίζονται στη μνήμη πιο συχνά. Από την άλλη, οι σπάνιες - και συνήθως υψηλής πληροφόρησης - εμπειρίες είναι σημαντικά λιγότερο πιθανό να δειγματοληπτηθούν από τις κοινές εμπειρίες. Συμπερασματικά, η ομοιόμορφη δειγματοληψία από τη μνήμη επανάληψης δεν χρησιμοποιεί αποτελεσματικά τις αποθηκευμένες εμπειρίες.

3.1.1 Prioritized Experience replay

Το πρόβλημα της ομοιόμορφης δειγματοληψίας του buffer αναπαραγωγής εμπειρίας είναι η δειγματοληψία με βάση το σφάλμα χρονικής διαφοράς (temporal difference error-TD) δ. Για ένα σημείο δεδομένων i ισχύει:

$$\delta_i = Q_t(s_i, \alpha_i) - r_i + \gamma \max_{\alpha'} Q_t(s_t', \alpha') \quad (3.1)$$

Αυτή είναι μια εναλλακτική μέθοδος δειγματοληψίας από τη μνήμη επανάληψης με βάση την ιδέα της σάρωσης κατά προτεραιότητα. Ωστόσο, αντί να χρησιμοποιηθεί μια άπληστη προσέγγιση όπου οι εμπειρίες με το υψηλότερο TD, η επανάληψη εμπειρίας κατά προτεραιότητα υπολογίζει μια πιθανότητα δειγματοληψίας για κάθε εμπειρία i χρησιμοποιώντας μια κατανομή Boltzmann: [46]

$$P(i) = p_i^{\alpha} / \sum_k (p_i^{\alpha}) \quad (3.2)$$

,όπου α είναι παράμετρος θερμοκρασίας, που χρησιμοποιείται για την εξισορρόπηση μεταξύ των απολύτως άπληστων προτεραιοτήτων (α = 1) και της ομοιόμορφης δειγματοληψίας (α = 0). Η αξία p_i υπολογίζεται σύμφωνα με την αναλογική μέθοδο ή την μέθοδο που βασίζεται σε κατάταξη.

- Η αναλογική μέθοδος (proportional method) υπολογίζεται από τον τύπο $p_i = |\delta_i| + \varepsilon$, όπου $\varepsilon > 0$ αρκετά μικρό ώστε να διασφαλίζεται πιθανότητα για τις εμπειρίες που παρουσιάζουν $\delta_i = 0$.
- Από την άλλη, η μέθοδος κατάταξης (rank-based method) υπολογίζεται από τον τύπο $p_i = 1/\text{rank}(i)$, όπου rank(i) είναι η κατάταξη τα εμπειρίας i όταν η μνήμη αναπαραγωγής εμπειρίας είναι ταξινομημένη με βάση το δ_i.

Η ιεράρχηση βάσει κατάταξης παρουσιάζει καλύτερα αποτελέσματα όταν οι διαφορές στα δ_i είναι αρκετά μεγάλες, εφόσον τα απροσδόκητα μεγάλα TD-λάθη δεν λαμβάνουν σημαντικά μεγάλο τμήμα του χώρου πιθανοτήτων, αλλά η πιθανότητα βασίζεται στην κατάταξή τους, που είναι το ανώτατο όριο.

3.1.2 Replay Memory Composition

Η τεχνική αυτή αποτελεί ένα υβριδικό μοντέλο των δύο τεχνικών που αναφέρθηκαν για την κλασική μνήμη αναπαραγωγής εμπειρίας, σύμφωνα με την οποία στο πρώτο μισό της μνήμης εντοπίζονται οι παλιές εμπειρίες του εκάστοτε πράκτορα, ενώ στο δεύτερο τμήμα αποθηκεύονται οι νέες εμπειρίες. [47] Η μέθοδος αυτή είναι ιδιαίτερα χρήσιμη ώστε να επανεξετάζονται οι παλαιότερες διερευνητικές εμπειρίες, χωρίς να αντικαθίστανται από τις νέες. Με αυτόν τον τρόπο, λύνεται το πρόβλημα της «καταστροφικής λήθης» (“catastrophic forgetting”), σύμφωνα με το οποίο η προγενέστερη κατάρτιση του πράκτορα λησμονείται , αφού πανογράφεται από τις νέες εμπειρίες. Εκτός των άλλων, η τεχνική αυτή αυξάνει την σταθερότερα της συνάρτησης Q.

3.2 Μέθοδος DQN

Στα πλαίσια της ενισχυτικής μάθησης η εκμάθηση πρακτόρων πραγματοποιείται μέσα από βαθμωτές ανταμοιβές, οι οποίες παρουσιάζουν σποραδικότητα, θόρυβο και καθυστέρηση. Εκτός αυτών, εφόσον τα γεγονότα είναι διαδοχικά, είναι φυσικό να εντοπίζεται μεγάλη συσχέτιση δεδομένων, ενώ η συνεχής εκμάθηση νέων συμπεριφορών οδηγεί σε μεταβολή της κατανομής των δεδομένων. Το γεγονός αυτό δεν είναι συνεπές με τη σταθερή κατανομή που προϋποθέτει η βαθιά μάθηση.

Τα βαθιά Q-δίκτυα (DQN) χρησιμοποιούν βαθιά νευρωνικά δίκτυα για να προσεγγίσουν τη συνάρτηση Q εκπαιδεύοντας τον εκάστοτε πράκτορα μέσα από πολιτικές ελέγχου ακατέργαστων δεδομένων. Κατά τη χρήση μη γραμμικής συνάρτησης προσέγγισης, τα νευρωνικά δίκτυα είναι ιδιαίτερα επιρρεπή σε αποκλίσεις. Οι τελευταίες οφείλονται είτε σε συσχετίσεις που υπάρχουν στις ακολουθίες των παρατηρήσεων, είτε σε συσχετίσεις μεταξύ των τιμών της συνάρτησης πράξης-αξίας Q(s, α) και της βέλτιστης συνάρτησης πράξης-αξίας $Q^*(s,\alpha)$ που υπακούει την εξίσωση Bellman ως εξής: $Q^*(s,\alpha) = E_{s',\epsilon E}[r+\gamma \max_{\alpha'} Q^*(s', \alpha')|s,\alpha]$.

Για την ανανέωση των βαρών του νευρωνικού, αυτό εκπαιδεύεται με μια παραλλαγή της Q-learning μεθόδου, η οποία χρησιμοποιεί στοχαστική κάθοδο κατά την κλίση (stochastic gradient descent). Ενώ, η επανάληψη εμπειρίας (experience replay) είναι μια τεχνική που βοηθά στην αφαίρεση των χρονικών συσχετισμών της ακολουθίας παρατήρησης και στην εξομάλυνση της κατανομής δεδομένων. [48] Ο DQN αλγόριθμος είναι model-free καθώς αξιοποιεί δείγματα από το περιβάλλον χωρίς να προβαίνει σε εκτιμήσεις αυτού, και off-policy, μιας και μαθαίνει μέσω μιας ε-άπληστης στρατηγικής (επιλέγει τυχαία πράξη με πιθανότητα ε), εξασφαλίζοντας παράλληλα επαρκή εξερεύνηση του χώρου κατάστασης στο οποίο κινείται ο πράκτορας. Υπό αυτό το πρίσμα, οι εμπειρίες που συλλέγονται από τα βήματα μετάβασης αποθηκεύονται σε ένα buffer/memory επανάληψης, από τον οποίον ανασύρονται ομοιόμορφα για να χρησιμοποιηθούν στα βήματα ενημέρωσης. Κατά τη διάρκεια εκτέλεσης του αλγορίθμου πραγματοποιούνται ανανεώσεις βαρών του δικτύου με χρήση τυχαίων δειγμάτων από τον buffer. Η επαναληπτική ενημέρωση που ανανεώνει τις τιμές Q προς τις τιμές-στόχους, ενημερώνει τις τελευταίες επίσης μόνο περιοδικά για να μειωθούν οι συσχετίσεις με το στόχο. Έτσι, αυξάνεται η σταθερότητα του πράκτορα.

Η μέθοδος χρησιμοποιεί ως κριτήριο κατάρτισης το μέσο τετραγωνικό σφάλμα μεταξύ της τρέχουσας εκτίμησης Q-value και των στόχων ενδιαφέροντός μας (ενημέρωση Q-learning). Το σφάλμα λοιπόν υπολογίζεται από τον παρακάτω τύπο:

$$L(\theta_i) = E(s_{t+1},\alpha_{t+1})[(r_{t+1} + \gamma max_{\alpha(t+1)}Q(s_{t+1}, \alpha_{t+1}, \theta_{i-1}) - Q(s_t, \alpha_t, \theta_i))^2] \quad (3.3)$$

,όπου η ποσότητα $(r_{t+1} + \gamma max_{\alpha(t+1)}Q(s_{t+1}, \alpha_{t+1}, \theta_{i-1}) = m_i$ αντιπροσωπεύει έναν σταθερό όρο ενημέρωσης της συνάρτησης πράξης-αξίας Q. Στη συνέχεια, τα σφάλματα διαδίδονται στο τρέχον δίκτυο Q, όπου η πρώτη Ιακωβιανή που προκύπτει από το επίπεδο εξόδου είναι η εξής:

$$\nabla_{\theta i}L(w) = E(s_{t+1},\alpha_{t+1})[(m_i - Q(s_t, \alpha_t, \theta_i))\cdot \nabla_w Q(s_t, \alpha_t, \theta_i)] \quad (3.4)$$

Με αυτόν τον τρόπο, ολόκληρο το δίκτυο αντιστοιχίζει τις πράξεις χρησιμοποιώντας μια παραλλαγή του batch learning (εκμάθηση με τη χρήση μιας ποσότητας προγενέστερων εμπειριών) για να αποσυμπιέσει τις παρατηρήσεις εξομαλύνοντας τις ταλαντώσεις κατά την διάρκεια των επαναλήψεων.

Συγκριτικά με τον online Q-learning αλγόριθμο, του οποίου αποτελεί μια παραλλαγή, ο DQN παρουσιάζει αρκετά πλεονεκτήματα. Αρχικά, η αποδοτικότητα βελτιώνεται μιας και τα βάρη ανανεώνονται με κάθε βήμα εμπειριών που χρησιμοποιείται για την εκμάθηση. Επιπλέον, με την μέθοδο αναπαραγωγής εμπειρίας και τη χρήση τυχαίων και όχι συνεχόμενων δειγμάτων μειώνεται σημαντικά η συσχέτισή τους. Παρακάτω παρουσιάζεται ο ψευδοκώδικας για τον αλγόριθμο DQN.

Algorithm 12: Deep Q-learning with experience replay

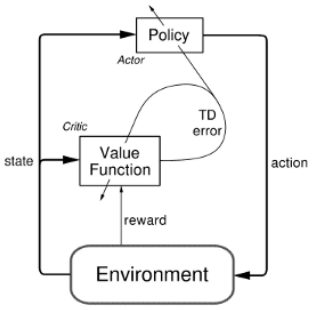
Initialize Q-networks n^{base} and n^{target} with random weights
$s_t \leftarrow$ InitialStateVector
Initialize action
Initialize experience replay memory D to capacity N
Take the action α
For i=0; i<|D|; i++ do
  Receive reward r_t
  Observe next state s_{t+1}
  Store transition < s_t, α, r_t, s_{t+1}> to experience replay buffer D
  Sample random action α ~U(A)
  Take the action α
End for
For i=|D|; i<1e6; i++ do
  Interact with environment
  Receive reward r_t

Observe next state s_{t+1}
Store transition $< s_t, \alpha, r_t, s_{t+1}>$ to experience replay buffer D
With propability ε:
Select action using ε-greedy policy $\alpha \sim U(A)$
Otherwise:
$\alpha = \text{argmax}_\alpha Q(s,\alpha ; n^{target})$
Take action α
Perform updates:
Sample mini-batch of transitions from D $(s_m, \alpha_m, r_m, s_{m+1}) \sim U(D)$
Update n^{target} using $(Q(s_m, \alpha_m) - r_m + \gamma[\max_{\alpha'} Q_t(s_{m+1},\alpha_{m+1} ; n^{target})])^2$
Every M steps:
Copy base network weights to target network $n^{target} = n^{base}$
End for

3.3 Μέθοδοι Actor-Critic

Οι αλγόριθμοι actor-critic διατηρούν δύο συγκεκριμένα σύνολα παραμέτρων, διαχωρίζοντας έτσι τη συνάρτηση πράξης-αξίας Q και την πολιτική. Ο εκπαιδευόμενος πράκτορας χωρίζεται σε δύο ξεχωριστές οντότητες: τον δράστη (πολιτική) και τον κριτή (συνάρτηση αξίας). Ο δράστης λαμβάνει ως είσοδο την τρέχουσα κατάσταση και εξάγει τη βέλτιστη δράση. [49] Συγκεκριμένα, ελέγχει τη συμπεριφορά του πράκτορα και έτσι ενημερώνει κατάλληλα την πολιτική ώστε να είναι η βέλτιστη. Ενώ ο ρόλος του κριτή είναι να εκτιμά τις συναρτήσεις αξίας που συνδέονται με την πολιτική του δράστη χρησιμοποιώντας δείγματα, δηλαδή αξιολογεί πόσο αποτελεσματική είναι η πολιτική π_θ για τις τρέχουσες παραμέτρους θ με στόχο τη βελτίωση της απόδοσης. Ουσιαστικά, ο κριτής παίζει το ρόλο της "αξιολόγησης" ενός τυπικού DQN μοντέλου, λαμβάνοντας την κατάσταση του περιβάλλοντος και μια δράση και επιστρέφοντας μια ανταμοιβή που αντιπροσωπεύει πόσο εύστοχη είναι η ενέργεια για τη δεδομένη κατάσταση. Με άλλα λόγια, υπολογίζει την αναμενόμενη ανταμοιβή, ενώ βρίσκεται σε μια δεδομένη κατάσταση και αξιολογώντας την ποιότητα της τρέχουσας πολιτικής, προσαρμόζει την εκτίμηση της συνάρτησης αξίας. Μετά από έναν αριθμό βημάτων αξιολόγησης πολιτικής από τον κριτή, ο δράστης ενημερώνεται χρησιμοποιώντας πληροφορίες από τον κριτή. Με αυτόν τον τρόπο, τα δύο παραπάνω μοντέλα εκπαιδεύονται και γίνονται συνεχώς καλύτερα το καθένα στον ρόλο του. [50]

Ο δράστης μπορεί να αναπαρασταθεί από ένα οποιασδήποτε αρχιτεκτονικής νευρωνικό δίκτυο, του οποίου στόχος αποτελεί η παραγωγή της βέλτιστης πράξης δεδομένης μια κατάστασης. Από την άλλη πλευρά, ο κριτής εκπροσωπείται από μια προσεγγιστική συνάρτηση που παίρνει ως είσοδο το περιβάλλον και τη δράση του δράστη και μετά από κατάλληλη επεξεργασία εξάγει τη μέγιστη δυνατή μελλοντική αμοιβή, δηλαδή την εκάστοτε αξία της συνάρτησης Q για το εκάστοτε ζεύγος (s,α). Η εκπαίδευση των δύο δικτύων πραγματοποιείται ξεχωριστά, ενώ για την ενημέρωση των βαρών τους χρησιμοποιείται gradient ascent. Η ιδέα πίσω από την τεχνική gradient ascent έγκειται στο ότι αντί να αναζητηθεί η βέλτιστη επιλογή και έπειτα να προσαρμοστεί το μοντέλο σε αυτήν, στόχος είναι η προσέγγιση του παγκόσμιου μεγίστου μέσω της αφελής τεχνικής να ακολουθούνται τα επιμέρους τοπικά μέγιστα, μέθοδος που φαίνεται να λειτουργεί καλά στην πλειονότητα των πρακτικών προβλημάτων. Αξίζει να σημειωθεί πως η ενημέρωση των βαρών γίνεται σε κάθε βήμα και όχι μόνο στο τέλος του εκάστοτε επεισοδίου, αντίθετα με τις μεθόδους policy gradient. χρησιμοποιήσαμε το μοντέλο-στόχο. Και έτσι, πρέπει να επικαιροποιούμε τα βάρη του σε κάθε βήμα. Η ανανέωση των βαρών πραγματοποιείται αργά και συγκεκριμένα, διατηρείται η τιμή του μοντέλου-στόχου κατά ένα κλάσμα *tau* και ενημερώνεται για να είναι το αντίστοιχο βάρος μοντέλου, το υπόλοιπο *(1- tau)* κλάσμα.



***Εικόνα 11**: Actor-Critic model*

Οι μέθοδοι δράστη-κριτή (Actor-Critic) επιδιώκουν να συνδυάσουν τα πλεονεκτήματα των actor-only και critic-only μεθόδων. Όπως και οι μέθοδοι actor-only, οι μέθοδοι actor-critic είναι ικανές να παράγουν συνεχείς δράσεις, ενώ η μεγάλη διακύμανση της πολιτικής κλίσεων (policy gradient) των actor-only μεθόδων αντισταθμίζονται με την προσθήκη ενός κριτή. Αυτές οι μέθοδοι συνήθως διατηρούν τις επιθυμητές ιδιότητες σύγκλισης των μεθόδων διαβάθμισης πολιτικής, σε αντίθεση με τις μεθόδους critic-only. Στις μεθόδους actor-critic, η πολιτική ενημερώνεται στην κατεύθυνση διαβάθμισης πολιτικής (policy gradient) χρησιμοποιώντας μόνο ένα μικρό βήμα, το οποίο συνεπάγεται ότι μια αλλαγή στη συνάρτηση αξίας θα έχει ως αποτέλεσμα μόνο μια μικρή αλλαγή στην πολιτική, που οδηγεί σε λιγότερο ή καθόλου ταλαντωτική συμπεριφορά την τελευταία. Οι εν λόγω αλγόριθμοι απαιτούν ελάχιστο υπολογισμό για την επιλογή δράσης, μιας και η πολιτική αποθηκεύεται ξεχωριστά, γεγονός που εξαλείφει την ανάγκη αναζήτησης για την επιλογή ενεργειών κατά τη διάρκεια κάθε βήματος.

3.3.1 Μέθοδος Soft Actor-Critic

Ο αλγόριθμος soft actor-critic (SAC) είναι ένας off-policy αλγόριθμος ενισχυτικής μάθησης βασισμένος στη μέγιστη εντροπία του πλαισίου μάθησης ενίσχυσης. Υπό αυτό το πρίσμα, όπως αναφέρθηκε παραπάνω, ο δράστης στοχεύει στη μεγιστοποίηση της αναμενόμενης ανταμοιβής, μεγιστοποιώντας παράλληλα την εντροπία. Με αυτόν τον τρόπο, ο πράκτορας προσπαθεί να πετύχει το στόχο του ενεργώντας όμως όσο το δυνατόν πιο τυχαία. Ο αλγόριθμος SAC είναι ένας off-policy μέγιστης εντροπίας αλγόριθμος που προσφέρει τόσο τη δειγματοληπτική μάθηση όσο και τη σταθερότητα. [51] Ένας soft actor-critic αλγόριθμος ξεπερνά την πολυπλοκότητα και την πιθανή αστάθεια που εκδηλώνεται σε προγενέστερους off-policy αλγορίθμους μέγιστης εντροπίας που βασίζονται στην Q-learning μέθοδο. Ενώ, από σχετικές μελέτες έχει αποδειχθεί πειραματικά ότι ο εν λόγω αλγόριθμος επιτυγχάνει σημαντική βελτίωση στις επιδόσεις και στην αποτελεσματικότητα δείγματος σε σχέση με προηγούμενες off-policy και on-policy μεθόδους. Ένας SAC αποτελείται συνήθως από τα παρακάτω στοιχεία-κλειδιά που συμβάλλουν στην αποδοτικότητά του:

- Μια αρχιτεκτονική actor-critic με χωριστή πολιτική και δίκτυα που προσεγγίζουν τη συνάρτηση αξίας
- Μια off-policy διαμόρφωση που επιτρέπει την επαναχρησιμοποίηση δεδομένων και εμπειριών που έχουν συλλεχθεί σε παρελθοντικό χρόνο
- Μεγιστοποίηση της εντροπίας για να εξασφαλιστεί η σταθερότητα και η επαρκής εξερεύνηση

Στα πλαίσια υλοποίησης ενός SAC αλγορίθμου, χρησιμοποιούνται νευρωνικά δίκτυα τόσο για την συνάρτηση Q όσο και για την πολιτική, ενώ αντί της εκτέλεσης της αξιολόγησης και βελτίωση της σύγκλισης, πραγματοποιείται μια εναλλαγή προς τη βελτιστοποίησης των δύο δικτύων με στοχαστική κάθοδο κλίσης. [51] Για παράδειγμα, οι συναρτήσεις αξίας μπορούν να μοντελοποιηθούν ως νευρωνικά δίκτυα, και η πολιτική σαν μία Γκαουσιανή με μέση τιμή

και συνδιακύμανση να προκύπτουν από τα νευρωνικά δίκτυα. Η συνάρτηση κατάστασης-αξίας προσεγγίζει την soft value, όπως καλείται, ενώ δεν είναι απαραίτητο να αναπτυχθεί επιπλέον νευρωνικό δίκτυο για την προσέγγιση της συνάρτησης κατάστασης-αξίας, εφόσον αυτή σχετίζεται με τη συνάρτηση Q και την πολιτική. Η ποσότητα αυτή μπορεί να εκτιμηθεί από ένα μόνο δείγμα δράσης από την τρέχουσα πολιτική χωρίς να εισάγει προκατάληψη (bias). Ωστόσο, η ύπαρξη χωριστού νευρωνικού δικτύου για την προσέγγιση της soft value, μπορεί να σταθεροποιήσει την διαδικασία εκπαίδευσης και συμβάλει στην ταυτόχρονη εκπαίδευση και άλλων δικτύων. Η συνάρτηση soft value εκπαιδεύεται με στόχο της ελαχιστοποίηση του τετραγωνικού υπολειπόμενου σφάλματος (squared residual error).

Η μέθοδος εναλλάσσεται μεταξύ της συλλογής εμπειριών από το περιβάλλον με την τρέχουσα πολιτική και της ενημέρωσης των νευρωνικών δικτύων χρησιμοποιώντας τις στοχαστικές κλίσεις από παρτίδες που ελήφθησαν από την μνήμη επανάληψης (replay buffer). Η χρήση off-policy δεδομένων από τον replay buffer είναι εφικτή, διότι τόσο οι εκτιμητές αξίας όσο και η πολιτική μπορούν να εκπαιδευτούν εξ ολοκλήρου σε δεδομένα off-policy. Ο αλγόριθμος είναι αγνωστικιστής στην παραμετροποίηση της πολιτικής, εφόσον αυτή μπορεί να αξιολογηθεί για οποιαδήποτε αυθαίρετο ζεύγος κατάστασης-δράσης.

Αλγόριθμοι που βασίζονται στην αρχή της μέγιστης εντροπίας μπορούν να ξεπεράσουν συμβατικές μεθόδους RL σε δύσκολες εργασίες. Παρακάτω παρατίθενται τα θεμελιώδη στοιχεία των αλγορίθμων SAC που επηρεάζουν την απόδοση. [51]

Stochastic vs Deterministic policy

Ένας soft actor-critic αλγόριθμος μαθαίνει στοχαστικές πολιτικές μέσω ενός στόχου μέγιστης εντροπίας. Η εντροπία εμφανίζεται τόσο στην πολιτική όσο και στην συνάρτηση αξίας. Στην πολιτική, αποτρέπει την πρόωρη σύγκλιση της διακύμανσης πολιτικής, ενώ στη συνάρτηση αξίας ενθαρρύνει την εξερεύνηση αυξάνοντας την αξία των περιοχών του χώρου κατάστασης που οδηγούν σε συμπεριφορά υψηλής εντροπίας (Εξίσωση 5). Για να αποδειχθεί πώς η στοχαστικότητα της πολιτικής και η μεγιστοποίηση της εντροπίας επηρεάζουν την απόδοση, σχετικές μελέτες έχουν συγκρίνει τον αλγόριθμο SAC με μια ντετερμινιστική παραλλαγή του που δεν μεγιστοποιεί την εντροπία. Η εν λόγω παραλλαγή αποτελείται από δύο Q συναρτήσεις, χρησιμοποιεί σκληρές ενημερώσεις στόχων και χρησιμοποιεί σταθερό θόρυβο εξερεύνησης.

Ο SAC συμπεριφέρεται με μεγαλύτερη συνέπεια, ενώ η ντετερμινιστική παραλλαγή παρουσιάζει πολύ υψηλή μεταβλητότητα, υποδεικνύοντας σημαντικά χειρότερη σταθερότητα. Συνεπώς, η διαδικασία μάθησης μιας στοχαστικής πολιτικής με μεγιστοποίηση της εντροπίας μπορεί να σταθεροποιήσει δραστικά την εκπαίδευση. Το γεγονός αυτό είναι ιδιαίτερα σημαντικό σε δύσκολα προβλήματα, όπου ο συντονισμός των υπερπαραμέτρων είναι δύσκολος.

Policy evaluation

Δεδομένου ότι το SAC συγκλίνει σε στοχαστικές πολιτικές, είναι οφέλιμο η τελική πολιτική που θα ακολουθηθεί να κατασταθεί ντετερμινιστική για τις καλύτερες επιδόσεις. Για την αξιολόγηση, προσεγγίζεται η μέγιστη εκ των υστέρων δράση επιλέγοντας τον μέσο όρο της κατανομής πολιτικής.

Reward scale

Ο SAC αλγόριθμος είναι ιδιαίτερα ευαίσθητος στην κλιμάκωση του σήματος ανταμοιβής, δηλαδή η απόδοση της διαδικασίας μάθησης μεταβάλλεται καθώς αλλάζει η η κλίμακα ανταμοιβής. Για μικρές ανταμοιβές, η πολιτική γίνεται σχεδόν ομοιόμορφη, και έτσι αποτυγχάνει να εκμεταλλευτεί το σήμα ανταμοιβής, με συνέπεια να υποβαθμίζεται σημαντικά η επίδοση. Από την άλλη, για μεγάλες ανταμοιβές, το μοντέλο μαθαίνει γρήγορα στην αρχή

αλλά η πολιτική στη συνέχεια γίνεται σχεδόν ντετερμινιστική, οδηγώντας σε φτωχά τοπικά ελάχιστα λόγω έλλειψης επαρκούς εξερεύνησης. Με τη σωστή κλιμάκωση ανταμοιβής, το μοντέλο εξισορροπεί ανάμεσα στην εξερεύνηση και στην εκμετάλλευση, οδηγώντας σε ταχύτερη μάθηση και καλύτερη ασυμπτωτική απόδοση.

Target network update

Είναι σύνηθες να χρησιμοποιείται χωριστή δίκτυο αξίας στόχου που παρακολουθεί αργά την πραγματική συνάρτηση αξίας για τη βελτίωση της σταθερότητας. Χρησιμοποιείται συνήθως μια εκθετικά μεταβαλλόμενος μέσος όρος, με μια σταθερά εξομάλυνσης τ, για να ενημερώσει τα βάρη του δικτύου-στόχου. Όταν η σταθερά εξομάλυνσης λάβει την τιμή ένα αντιστοιχεί σε σκληρή ενημέρωση όπου αντιγράφονται τα βάρη απευθείας σε κάθε επανάληψη και όταν ισούται με μηδέν δεν ενημερώνεται καθόλου το δίκτυο-στόχος. Μεγάλη τιμή του τα οδηγεί σε αστάθειες, ενώ αρκετά μικρές τιμές καθιστούν την εκπαίδευση πιο αργή.

Παρακάτω παρουσιάζεται ο ψευδοκώδικας SAC.

Algorithm 14: Soft Actor-Critic

Input: initial policy parameters θ, Q-function parameters φ_1, φ_2, V-function parameters ψ, empty replay buffer D
$\psi_{target} \leftarrow \psi$
repeat:
  Observe state s , select action $\alpha \sim \pi_\theta(\cdot|s)$
  Execute α
  Observe next state s', reward r, and done signal d to indicate whether s' is terminal state
  Store (s, α, r, s', d) in replay buffer D
  If s'= $s_{terminal}$ reset environment state
  If it's time to update then
    For j in range(however many updates) do
      Randomly sample a batch of transitions, B = {(s, α, r, s', d)} from D
      Compute targets for Q and V functions:
        $y_q(r, s', d) = r + \gamma(1-d)V_{\varphi target}(s')$
        $y_v(s) = \min_{i=1,2} Q_{\varphi 1}(s,\tilde{\alpha}) - \alpha \log \pi_\theta(\tilde{\alpha}|s)$, $\tilde{\alpha} \sim \pi_\theta(\cdot|s)$
      Update Q-functions by one step of gradient descent using
        $\nabla_{\varphi i} \frac{1}{|B|} \sum_{(s,\alpha,r,s',d)\epsilon B} (Q_{\varphi,i}(s,\alpha) - y_q(r, s', d))^2$ for i=1,2
      Update V-function by one step of gradient descent using
        $\nabla_\psi \frac{1}{|B|} \sum_{s\epsilon B} (V_\psi(s) - y_v(s))^2$
      Update policy by one step of gradient ascent using
        $\nabla_\theta \frac{1}{|B|} \sum_{s\epsilon B} (Q_{\varphi,1}(s, \tilde{\alpha}_\theta(s)) - \alpha \log \pi_\theta(\tilde{\alpha}_\theta(s)|s))$, where $\tilde{\alpha}_\theta(s)$ is a sample from $\pi_\theta(\tilde{\alpha}|s)$
      Update target value network with
        $\psi_{target} \leftarrow \rho\psi_{target} + (1-\rho)\psi$
    end for
  end if
until convergence

3.4 Hindsight Experience Replay

Η διαχείριση των αραιών ανταμοιβών ήταν μία από τις πιο διαδεδομένες προκλήσεις της Ενισχυτικής Μάθησης. Η τεχνική της «εκ των υστέρων επανάληψης εμπειρίας» (Hindsight Experience Replay – HER) επιτρέπει την αποτελεσματική εκμάθηση δειγμάτων από ανταμοιβές που είναι αραιές. [52] Παραδοσιακά συνδυάζεται με κάποιον αυθαίρετο off-policy αλγόριθμο RL.

Ο άνθρωπος έχει την ικανότητα να μαθαίνει τόσο πετυχαίνοντας τον επιθυμητό στόχο, όσο και αποτυγχάνοντας. Αντιθέτως, οι κλασσικοί model-free αλγόριθμοι ενισχυτικής μάθησης σε ένα ενδεχόμενο αποτυχίας θεωρούν ότι η σειρά πράξεων-αποφάσεων δεν οδήγησε στο επιθυμητό αποτέλεσμα και έτσι ο εκάστοτε πράκτορας μαθαίνει ελάχιστα, αν όχι απολύτως τίποτα.

Προκειμένου να υλοποιηθεί η ικανότητα επιθεώρησης προηγούμενων δράσεων και να συμπεραθούν χρήσιμες πληροφορίες από τις δράσεις αυτές, παρόλο που τελικά οδήγησαν στην αποτυχία, θα επιστρατευθούν τεχνικές off-policy learning. Στόχος των RL αλγορίθμων είναι η μάθηση μιας πολιτικής που θα οδηγήσει στην μεγιστοποίηση της μελλοντικής αθροιστικής ανταμοιβής. Αλληλεπιδρώντας με το περιβάλλον μέσω δοκιμών και σφαλμάτων, χρησιμοποιούνται οι εμπειρίες που συλλέγονται στη διαδικασία μάθησης για να τελειοποιηθεί η πολιτική. Οι off-policy αλγόριθμοι μπορούν να μάθουν και να βελτιώσουν μια πολιτική από δεδομένα που έχουν συλλεχθεί από άλλη πολιτική. Ο σκοπός της πολιτικής εξερεύνησης είναι να διερευνήσει επαρκώς τον χώρο κατάστασης, ώστε ο αλγόριθμος μάθησης να μπορεί να συμπεράνει από αυτούς ποιες ενέργειες πρέπει να ληφθούν σε διαφορετικά καταστάσεις. Για τη καταχώρηση των εμπειριών χρησιμοποιείται ένας buffer, δηλαδή μια μνήμη παρελθοντικών εμπειριών, ο οποίος βοηθά στην ενημέρωση του νευρωνικού δικτύου. Σε ένα multi-goal πρόβλημα, αποθηκεύεται και ο εκάστοτε στόχος, ενώ η ανταμοιβή είναι συνάρτηση εκτός της κατάστασης και της πράξης, και του στόχου.

Η ιδέα πίσω από τον αλγόριθμο HER (εκ των υστέρων επανάληψη εμπειρίας) βασίζεται στην αποθήκευση στον buffer επανάληψης κάθε μετάβασης $s_t \rightarrow s_{t+1}$ μετά από κάθε επεισόδιο s_0, s_1, …, s_T , και όχι μόνο τον αρχικό στόχο για το εκάστοτε επεισόδιο, αλλά και ένα υποσύνολο άλλων ενδιάμεσων στόχων. [52] Παρατηρείται ότι ο επιδιωκόμενος στόχος ενώ επηρεάζει τις ενέργειες του πράκτορα, ωστόσο δεν μεταβάλλει τη δυναμική του περιβάλλοντος. Με βάση το γεγονός αυτό, η κάθε μετάβαση μπορεί να επαναληφθεί με αυθαίρετο στόχο υποθέτοντας ότι ακολουθείται ένας off-policy αλγόριθμος ενισχυτικής μάθησης. Με άλλα λόγια, υλοποιώντας τον HER φανταζόμαστε ότι ο στόχος ήταν εξ αρχής η κατάσταση s’ στην οποία καταλήγει ο πράκτορας στο τέλος του επεισοδίου, και συνεπώς ότι έχει επιτύχει τον επιθυμητό στόχο και για αυτό λαμβάνει θετική ανταμοιβή. Επομένως, εκτός από την αποθήκευση της πραγματικής τροχιάς, αποθηκεύεται και η τροχιά που έχει ως στόχο την κατάσταση στην οποία καταλήγει το επεισόδιο. Η εν λόγω τροχιά είναι η φανταστική και εδράζεται στην ανθρώπινη ικανότητα να μαθαίνει χρήσιμα πράγματα από αποτυχημένες προσπάθειες. Είναι σημαντικό να επισημανθεί ότι στη φανταστική πορεία, η ανταμοιβή που λήφθηκε στο τελικό βήμα του επεισοδίου είναι τώρα μια θετική ανταμοιβή που αποκτήθηκε από την επίτευξη του φανταστικού στόχου και επομένως ακόμη κι αν η ακολουθούμενη πολιτική είναι κακή, ο πράκτορας θα έχει πάντα κάποιες θετικές ανταμοιβές για να μαθαίνει από αυτές. Προφανώς, στην αρχή της διαδικασίας μάθησης, οι παραπάνω φανταστικοί στόχοι θα είναι μόνο οι καταστάσεις που θα μπορούσε να επιτύχει η τυχαία και προφανώς όχι βέλτιστη αρχική πολιτική, οι οποίες δεν χρησιμεύουν στην πράξη. Ωστόσο, τα νευρωνικά δίκτυα διασφαλίζουν ότι η πολιτική θα μπορούσε επίσης να προσεγγίσει καταστάσεις παρόμοιες με αυτές που έχει δει στο παρελθόν. [52] Αυτή είναι η ιδιότητα γενίκευσης, χαρακτηριστική της επιτυχημένης βαθιάς μάθησης. Αρχικά, ο πράκτορας θα είναι σε θέση να προσεγγίσει καταστάσεις σε μια σχετικά μικρή περιοχή γύρω από την αρχική κατάσταση και προοδευτικά επεκτείνει την προσβάσιμη περιοχή του χώρου κατάστασης μέχρι τελικά να μάθει να φτάνει στις καταστάσεις-στόχους που ενδιαφέρουν πραγματικά. Ο HER δεν απαιτεί από τον προγραμματιστή να προσαρμόσει το πρόβλημα ή να σχεδιάσει ένα πρόγραμμα κατάρτισης, αλλά ο πράκτορας προμηθεύεται με προβλήματα που είναι όντως σε θέση να λύσει και να αυξήσουμε σταδιακά το φάσμα αυτών των προβλημάτων.

Στην απλούστερη έκδοση του αλγορίθμου επαναλαμβάνεται κάθε μετάβαση με τον στόχο $m(s_T)$, δηλαδή τον στόχο που επιτυγχάνεται στην τελική κατάσταση του επεισοδίου. Επίσης, επαναλαμβάνεται κάθε τροχιά με τον αρχικό στόχο που ακολουθήθηκε στο επεισόδιο. Ο αλγόριθμος HER δεν απαιτεί να έχει κανέναν έλεγχο στην κατανομή των αρχικών καταστάσεων του περιβάλλοντος, ενώ μαθαίνει με εξαιρετικά σποραδικές ανταμοιβές και όπως έχει αποδειχθεί από σχετικά πειράματα, η κατάρτιση είναι καλύτερη σε σύγκριση με διαμορφωμένες αμοιβές. [52]

4 Υλοποίηση και Αξιολόγηση Πειραμάτων

Στην παρούσα διπλωματική μελετάται η συμπεριφορά και η απόδοση ποικίλων αλγορίθμων βαθιάς ενισχυτικής μάθησης σε ένα περιβάλλον πλοήγησης συνεχούς χώρου και χρόνου. Ειδικότερα, ο πράκτοράς μας είναι ένα αυτόνομο όχημα το οποίο εκπαιδεύεται κατάλληλα ώστε να παρκάρει μόνο του στον επιθυμητό χωρικό στόχο του περιβάλλοντος πλοήγησης μέσα στο οποίο εργαζόμαστε. Το πρόβλημα πλοήγησης αυτόνομων οχημάτων (AVs) αντιστοιχεί στον υπολογισμό των βέλτιστων ενεργειών που επιτρέπουν στο εκάστοτε AV να ξεκινήσει από το σημείο εκκίνησης A και να φτάσει στον προορισμό B μέσω μιας ομαλής τροχιάς. Στο περιβάλλον όπου εκπαιδεύεται ο πράκτοράς μας δεν υπάρχουν άλλα οχήματα ή εμπόδια που θα απαιτούσαν κατάλληλη διαχείριση από μέρος του προγραμματιστή. Η στάθμευση αποτελεί ένα έργο συνεχούς ελέγχου με συγκεκριμένο αρχικό στόχο εξ αρχής καθορισμένο, όπου ο εκάστοτε πράκτορας οδηγεί ένα αυτοκίνητο ελέγχοντας το πετάλι γκαζιού και τη γωνία διεύθυνσης, ώστε να σταθμεύσει σε μια δεδομένη θέση με την κατάλληλη κατεύθυνση. Για την υλοποίηση του προβλήματος χρησιμοποιούνται πειραματικά διάφορες μέθοδοι βαθιάς ενισχυτικής μάθησης: μία βασισμένη σε μοντέλο (model-based) με εκ των προτέρων σχεδιασμό τροχιάς και δύο μέθοδοι δίχως μοντέλο (model-free) DQN και SAC. Ενώ η γλώσσα προγραμματισμού που χρησιμοποιήθηκε για την μοντελοποίηση του προβλήματος είναι η python.

Υπάρχουν εκτεταμένες εργασίες σχετικά με τους αλγορίθμους πλοήγησης και σχεδιασμού στη ρομποτική και την αυτόνομη οδήγηση. Σε ένα ευρύτερο επίπεδο, μπορούν να ταξινομηθούν ως τεχνικές για τον έλεγχο του οχήματος (vehicle control) , τεχνικές σχεδιασμού κίνησης (motion planning) και μεθόδους που βασίζονται στην «από άκρη σε άκρη» μάθηση (end-to-end learning), όπου τα δεδομένα εισόδου επεξεργάζονται για την παραγωγή ενός διανύσματος δράσης στην έξοδο σε πραγματικό χρόνο. Οι μέθοδοι ελέγχου βασίζονται σε ένα πολύ ακριβές μοντέλο του οχήματος, το οποίο πρέπει να είναι γνωστό εκ των προτέρων για πλοήγηση σε υψηλές ταχύτητες ή κατά τη διάρκεια πολύπλοκων ελιγμών. Οι μέθοδοι σχεδιασμού κίνησης βασίζονται είτε αλγεβρική είτε σε πιθανολογική αναζήτηση ή στη χρήση μη γραμμικής βελτιστοποίησης ελέγχου. Αρκετές τεχνικές που βασίζονται στη μάθηση περιλαμβάνουν αλγορίθμους ενισχυτικής μάθησης που στοχεύουν στην εύρεση βέλτιστης πολιτικής που χαρτογραφεί απευθείας τις μετρήσεις των αισθητήρων για τον έλεγχο εντολών όπως η ταχύτητα, η επιτάχυνση ή η γωνία διεύθυνσης. Οι βέλτιστες πολιτικές για πλοήγηση μπορούν να μάθουν με κατάλληλες ανταμοιβές για τους διάφορους ελιγμούς του οχήματος. Άλλες μέθοδοι χρησιμοποιούν βαθιά πρόβλεψη τροχιάς που βασίζεται στη μάθηση μοντέλων για την πρόβλεψη της μελλοντικής κατάστασης.

Δεδομένης της τρέχουσας κατάστασης του περιβάλλοντος, που αποτελείται από συγκεκριμένη θέση, ταχύτητα, επιτάχυνση και γωνία τη στιγμή (t), ο στόχος είναι να προβλεφθεί η δράση για το όχημα στο επόμενο βήμα (t+1) με χρήση εκπαιδευμένης πολιτικής. Αυτές οι ενέργειες συνήθως ανήκουν σε ένα πεπερασμένο σύνολο διαμήκους κινήσεων (επιτάχυνσης, επιβράδυνσης, διατήρηση ταχύτητας) και πλευρικών ενεργειών (στροφή αριστερά/δεξιά).

4.1 Σχετικές Μελέτες

Αντίληψη του περιβάλλοντος

Ως χώρος στάθμευσης ορίζεται ο κενός χώρος που διαμορφώνεται από δύο εμπόδια ή βαμμένες στο δρόμο γραμμές. Για μια θέση στάθμευσης που σχηματίζεται από εμπόδια, έχουν αναπτυχθεί μέθοδοι που βασίζονται σε αισθητήρες εύρους [58],[59]. Αισθητήρες υπερήχων και LiDAR έχουν χρησιμοποιηθεί για να προσδιοριστεί το σχήμα των εκάστοτε εμποδίων. Αν και η ακρίβεια ανίχνευσης έχει βελτιωθεί, όπως αποδεικνύεται από πρόσφατη μελέτη [58], οι μέθοδοι αυτές δεν μπόρεσαν να εντοπίσουν το χώρο στάθμευσης μετά την έναρξη της

διαδικασίας του παρκαρίσματος. Οι προσεγγίσεις που βασίζονται στην όραση αναπτύχθηκαν για το χώρο στάθμευσης που σχηματίζεται από σημασμένο οδόστρωμα. Το σύστημα AVM (Around View Monitoring), ένα σύστημα αισθητήρων που ανιχνεύει τον χώρο, μπορεί να παρακολουθεί συνεχώς την θέση στάθμευσης. Για να επιτευχθεί η συνεχής επίβλεψη της θέσης πάρκινγκ κατά τη διάρκεια της πλοήγησης του οχήματος, έχουν προταθεί μέθοδοι κατάτμησης βάσει βαθιών συνελικτικών δικτύων [60],[61]. Για την παρακολούθηση του οχήματος, η ακρίβεια πρόβλεψης επιδεινώνεται από θορύβους και θα πρέπει να εξετάζεται το ενδεχόμενο απόφραξης. Σε σχετική μελέτη έχει χρησιμοποιηθεί φίλτρο συσχέτισης με πυρήνα για τη διερεύνηση εγγενών χαρακτηριστικών από δείγματα που παρενέβαιναν στο παρασκήνιο [62]. Κατά τη διάρκεια της διαδικασίας του πάρκινγκ, έχει αναπτυχθεί ταυτόχρονος εντοπισμός και χαρτογράφηση (SLAM) για τον εντοπισμό του οχήματος στο επίπεδο εκατοστών [63].

Παραγωγή κίνησης

Πολλές μέθοδοι σχεδιασμού και ελέγχου μονοπατιών έχουν προταθεί για τη παραγωγή της λειτουργίας στάθμευσης. Η πρώτη περιλαμβάνει αριθμητικές βελτιστοποιήσεις, αναζήτηση A*, εξερευνώντας ταχέως τυχαία οικογένεια δέντρων (RRT) και σχεδιαστές καμπυλών. Στις μεθόδους βελτιστοποίησης, αναπτύχθηκε τοπική βελτιστοποίηση [60],[64] για την εύρεση της συντομότερης διαδρομής. Παρόλο που δεν μπορεί να διασφαλιστεί το παγκόσμιο βέλτιστο, συνήθως παράγονται εφικτές διαδρομές σε σύντομο χρονικό διάστημα. Αντίθετα, η μέθοδος παγκόσμιας βέλτιστης λειτουργίας [65] χρειάστηκε πολύ χρόνο για να βρει τον βέλτιστο ελιγμό στάθμευσης. Η μέθοδος αναζήτησης A* [66] διακριτοποιεί το χώρο αναζήτησης, ο οποίος εγγυήθηκε την πληρότητα και τη βέλτιστη λειτουργία. Ωστόσο, η επιλογή του μεγέθους του πλέγματος απαιτεί εξειδικευμένες γνώσεις. Οι μέθοδοι RRT [67] μπορούν να σχεδιάσουν στο συνεχές χώρο χρησιμοποιώντας ευρετικές πληροφορίες. Οι σχεδιαστές καμπυλών [68] δημιουργούν συνεχείς καμπύλες συνδυάζοντας γραμμές και τόξα. Για τις μεθόδους πορείας, χρησιμοποιήθηκε γραμμικός τετραγωνικός ρυθμιστής (LQR) [69], έλεγχος λειτουργίας ολίσθησης [70] και ασαφής ελεγκτής [71]. Αυτές οι μέθοδοι ελέγχου, ωστόσο, δεν μπορούν να αντιμετωπίσουν περιορισμούς του συστήματος, όπως τα όρια ενεργοποίησης του συστήματος διεύθυνσης. Για να διασφαλιστεί ότι η σχεδιασμένη διαδρομή μπορεί να ακολουθηθεί χονδρικά, ο σχεδιασμός διαδρομής πρέπει να κατασκευάσει μια αποδεκτή διαδρομή που οριοθετείται από το μέγιστο ρυθμό αλλαγής καμπυλότητας και καμπυλότητας κάτω από τη συντηρητική υπόθεση ταχύτητας. Αν και η καμπυλότητα μπορεί να είναι συνεχής, οι πραγματικές τροχιές εξακολουθούν να αποκλίνουν από την αναμενόμενη διαδρομή λόγω των σφαλμάτων παρακολούθησης. Το σύστημα δεν είναι αρκετά ευέλικτο για να αντιμετωπίσει αντιληπτικές πληροφορίες σε πραγματικό χρόνο και να μεγιστοποιήσει την προσαρμοστικότητα του οχήματος.

Για τη βελτίωση της προσαρμοστικότητας του APS, έχουν αναπτυχθεί συστήματα APS ενισχυτικής μάθησης (RL) χωρίς μοντέλα, στα οποία ο αλγόριθμος μαθαίνει να κατευθύνει δοκιμάζοντας άμεσα ενέργειες σε μια προσπάθεια επίτευξης μέγιστης αθροιστικής ανταμοιβής. Το πλεονέκτημα της RL έγκειται στην ικανότητά της να αντιμετωπίζει τους περιορισμούς του συστήματος, καθώς και τον τρόπο συνεχούς μάθησης που καθοδηγείται από δεδομένα και τον άνθρωπο. Η μέθοδος RL λαμβάνει ρητά υπόψη τους περιορισμούς του συστήματος, συνήθως με τη μορφή ορίων ενεργοποιητών, σε κάθε χρονικό διάστημα. Ο αλγόριθμος λαμβάνει αποφάσεις με βάση τις καταστάσεις κίνησης σε πραγματικό χρόνο και τις αντιληπτικές πληροφορίες. Εν τω μεταξύ, τα σφάλματα του ελέγχου που ακολουθεί τη διαδρομή μπορούν να αποφευχθούν θεμελιωδώς με την ενσωμάτωση του σχεδιασμού και του ελέγχου χρησιμοποιώντας το RL, το οποίο αυξάνει το ανώτερο όριο του συστήματος. Το μοντέλο Deep Q-learning(DQN) [72] έχει χρησιμοποιηθεί για να μάθει τις ευρετικές πληροφορίες για υβριδικό A*, το οποίο παρέχει υπολογιστικά πλεονεκτήματα έναντι του συμβατικού σχεδιασμού μονοπατιών. Έχουν συγκριθεί διαφορετικές ρυθμίσεις για σχεδιασμό κίνησης που

βασίζεται σε DQN [73] , γεγονός που κατέδειξε την αποδεκτή απόδοση του χρόνου εκτέλεσης σε διάφορες συσκευές. Σε σύγκριση με τη μέθοδο DQN που βασίζεται στην αξία (value-based method), η μέθοδος που βασίζεται σε πολιτική (policy-based method) βελτιστοποιεί άμεσα την πολιτική με βάση τη μέθοδο δειγματοληψίας και υπολογίζει συνεχώς την κλίση της πολιτικής προσδοκίας ανταμοιβής σχετικά με τις παραμέτρους του δικτύου πολιτικής κατά τη διάρκεια της διαδικασίας εκπαίδευσης [78]. Αν και η μέθοδος που βασίζεται στην πολιτική εφαρμόζεται σε χώρους συνεχούς δράσης υψηλών διαστάσεων, σε κάθε επαναληπτικό βήμα θα πρέπει να δειγματοληπτείται δέσμη ακολουθίας για την επικαιροποίηση των παραμέτρων, προκαλώντας μεγάλη διακύμανση της πολιτικής εκτίμησης διαβάθμισης και διευκολύνοντας την πτώση στο τοπικό βέλτιστο. Η μέθοδος Actor-Critic, η οποία συνδυάζει τη μέθοδο βάσει αξίας και τη μέθοδο βάσει πολιτικής, υιοθετεί τη βασιζόμενη στην πολιτική μέθοδο για την επικαιροποίηση της πολιτικής και τη συνάρτηση αξίας ως μέθοδος της πολιτικής [79]. Ο αλγόριθμος Deep Deterministic Policy Gradient (DDPG) [80] έχει κάνει κάποιες βελτιώσεις συγκριτικά με τη μέθοδο Actor-Critic. Το μοντέλο DDPG [74] με έλεγχο προεπισκόπησης, ο οποίος βασίζεται σε σήμα αναφοράς, έχει προταθεί για την επίλυση του βέλτιστου προβλήματος ελέγχου της κάθετης στάθμευσης. Το DDPG [75] με χειροκίνητη εξερεύνηση οδηγών και διαφορετικούς αγωγούς επανεκπαίδευσης κύκλου ελέγχου έχει χρησιμοποιηθεί για την επίτευξη αντιδραστικού χώρου στάθμευσης από άκρο σε άκρο σε μια πραγματική πλατφόρμα οχημάτων. Ωστόσο, το APS χωρίς μοντέλο RL απαιτεί μεγάλες ποσότητες δεδομένων, γεγονός που καθιστά την εκπαίδευση στο πραγματικό όχημα ανέφικτη και ανυπολόγιστη.

Στη μέθοδο RL που βασίζεται σε μοντέλα, το MCTS με αλυσίδα μνήμης προτάθηκε στο [76] και [77] για να απαλλαγούμε από την ανθρώπινη εμπειρία. Το MCTS περιλαμβάνει βήματα επιλογής, επέκτασης, προσομοίωσης και δημιουργίας αντιγράφων ασφαλείας. Η τιμή κατάστασης αποκτήθηκε με προσομοίωση πολλών βημάτων από τον κόμβο φύλλων. Η πιθανότητα επιλογής δράσης στην προσομοίωση δόθηκε από μια πολιτική ANN, της οποίας οι εισροές ήταν οι καταστάσεις στάθμευσης και η έξοδος ήταν η κατανομή πιθανότητας. Κατά την επιλογή δράσης στην αναζήτηση δέντρων, το βάρος της πολιτικής ορίστηκε σε 0. Η αναγνώριση του συστήματος χρησιμοποιήθηκε για τη βελτίωση της ακρίβειας του μοντέλου. Η πολυ-αντικειμενική βέλτιστη πραγματοποιήθηκε συνδυάζοντας MCTS, διαμήκεις και πλευρικές πολιτικές. Ένας μεγάλος όγκος δεδομένων μπορεί να αποκτηθεί, αν και αυτή η μέθοδος έχει υψηλές απαιτήσεις για την ακρίβεια του μοντέλου του οχήματος, γεγονός που περιορίζει την εφαρμογή του.

4.2 Μοντελοποίηση προβλήματος

4.2.1 Αυτόνομη Πλοήγηση

Ένα ρομποτικό όχημα που οδηγεί αυτόνομα είναι ένας μακροχρόνιος στόχος τεχνητής νοημοσύνης. Στην παρούσα διπλωματική εργασία εξετάζεται το πρόβλημα της αυτόνομης οδήγησης στην λειτουργία πάρκινγκ υπό το πρίσμα της ενισχυτικής μάθησης χρησιμοποιώντας αλγορίθμους βαθιάς ενισχυτικής μάθησης. Τα συστήματα αυτόνομης οδήγησης (AD) αποτελούν εργασίες πολλαπλών επιπέδων αντίληψης που επιτυγχάνουν υψηλή ακρίβεια λόγω των αρχιτεκτονικών βαθιάς μάθησης. Το ζήτημα της αυτόνομης πλοήγησης γενικότερα, απαιτεί τον καθορισμό και την μεγιστοποίηση μιας συνάρτησης κόστους. Ο εκάστοτε πράκτορας οφείλει να μαθαίνει τις νέες διαμορφώσεις του περιβάλλοντος, καθώς και να προβλέπει τη βέλτιστη απόφαση κάθε στιγμή. Το περιβάλλον αντιπροσωπεύει έναν χώρο υψηλών διαστάσεων, ενώ δεδομένου του αριθμού των μοναδικών διαμορφώσεων, βάσει των οποίων ο πράκτορας και το περιβάλλον παρατηρούνται, αυτός είναι συνδυαστικά μεγάλος. Συνεπώς, απαιτείται η επίλυση μιας διαδοχικής διαδικασίας λήψης αποφάσεων, η οποία διαμορφώνεται στα πλαίσια της ενισχυτικής μάθησης, όπου ο πράκτορας μαθαίνει και ενεργεί

βέλτιστα στο περιβάλλον του. Η δημιουργία ενός πράκτορα αυτόνομης οδήγησης θεμελιώνεται σε τρεις βασικούς πυλώνες, όπως περιγράφονται παρακάτω.

Αναγνώριση

Η αναγνώριση αποτελεί τον προσδιορισμό των στοιχείων του περιβάλλοντος χώρου. Παράδειγμα αυτού αποτελεί ο εντοπισμός πεζών, εμποδίων, επιπλέον πρακτόρων, κ.λπ. . Αποτελεί σχετικά εύκολο έργο χάρη στην πρόοδο των αλγορίθμων βαθιάς μάθησης (DL), οι οποίοι επιτρέπουν αναγνώριση σε ανθρώπινο επίπεδο ή και παραπάνω σε διάφορα προβλήματα ανίχνευσης και ταξινόμησης αντικειμένων. Τα μοντέλα βαθιάς μάθησης είναι σε θέση να μάθουν σύνθετα χαρακτηριστικά αναπαραστάσεων από ακατέργαστα δεδομένα εισόδου (με τα δίκτυα CNN να συνιστούν ίσως το πιο επιτυχημένο μοντέλο βαθιάς μάθησης).

Πρόβλεψη

Για έναν πράκτορα αυτόνομης οδήγησης δεν αρκεί η αναγνώριση του περιβάλλοντός του. Οφείλει επίσης να χτίσει εσωτερικά μοντέλα που προβλέπουν τις μελλοντικές του καταστάσεις. Παραδείγματα αυτής της κατηγορίας προβλήματος περιλαμβάνουν τη δημιουργία ενός χάρτη του χώρου ή την παρακολούθηση ενός αντικειμένου. Για να επιτευχθεί αυτό απαιτείται η πρόβλεψη του μέλλοντος μέσω της αξιοποίησης παρελθοντικών εμπειριών και την ενσωμάτωση προηγούμενων πληροφοριών στο εκάστοτε μοντέλο.

Σχεδιασμός

Ο σχεδιασμός αποτελεί τη δημιουργία ενός αποτελεσματικού μοντέλου που ενσωματώνει αναγνώριση και πρόβλεψη για την διαμόρφωση της μελλοντικής σειράς ενεργειών οδήγησης που θα επιτρέψουν στο όχημα να πλοηγηθεί επιτυχώς στο περιβάλλον. Ο σχεδιασμός θεωρείται ίσως το δυσκολότερο έργο των τριών. Κάτι το οποίο εδράζεται στην ικανότητα του μοντέλου να κατανοεί το περιβάλλον (αναγνώριση) και τη δυναμική του (πρόβλεψη) με τρόπο που του επιτρέπει να προγραμματίζει τις μελλοντικές ενέργειες έτσι ώστε να αποφεύγει τις ανεπιθύμητες καταστάσεις (κυρώσεις) και να οδηγεί με ασφάλεια και ακρίβεια στον προορισμό/στόχο του (ανταμοιβές).

4.2.2 Προσομοίωση Περιβάλλοντος

Σε αυτή την διπλωματική, χρησιμοποιείται το περιβάλλον προσομοίωσης highway-env για την λειτουργία του πάρκινγκ (parking-v0), που αποτελεί μια εργασία συνεχούς ελέγχου υπό συνθήκες στόχου, στην οποία το όχημα-πράκτορας πρέπει να σταθμεύει σε ένα δεδομένο χώρο με την κατάλληλη κατεύθυνση. Στο παρόν περιβάλλον για την εργασία του πάρκινγκ δεν υπάρχουν στον χώρο άλλα οχήματα ή εμπόδια που να δυσκολεύουν περαιτέρω την κίνηση του πράκτορά μας στον χώρο και να προσθέτουν πολυπλοκότητα στο πρόβλημα.

Το περιβάλλον είναι ένα GoalEnv, δηλαδή ο πράκτορας λαμβάνει ένα λεξικό που περιέχει τόσο την τρέχουσα παρατήρηση (observation), δηλαδή την κατάσταση στην οποία βρίσκεται ο πράκτορας το εκάστοτε στιγμιότυπο, όσο και τον επιθυμητό στόχο (desired_goal) που προϋποτίθεται για την πολιτική του. Οι παρατηρήσεις που λαμβάνονται από το περιβάλλον χρησιμοποιούνται για την εκπαίδευση του πράκτορα και την εκμάθηση της εκάστοτε πολιτικής. Εντός του εν λόγω λεξικού περιέχεται επίσης και ένας πίνακας κατάστασης που ονομάζεται achieved_goal, ο οποίος όμως δεν είναι χρήσιμος εν προκειμένω, καθώς χρησιμοποιείται μόνο όταν οι χώροι κατάστασης και στόχου είναι διαφορετικοί, ως προβολή από τον χώρο παρατήρησης στο χώρο στόχου.

4.2.2.1 Χώρος κατάστασης

Ο χώρος κατάστασης του οχήματος είναι συνεχής και αποτελείται από έξι στοιχεία (έξι διαστάσεις). Η κινηματική των οχημάτων αντιπροσωπεύεται από την κλάση Kinematic Bicycle Model, όπως περιγράφεται παρακάτω:

$$\dot{x} = v\cos(\psi+\beta)$$
$$\dot{y} = v\sin(\psi+\beta)$$
$$\ddot{x} = \alpha$$
$$\dot{w} = \frac{v}{l}\sin\beta$$
$$\beta = \tan^{-1}(\frac{1}{2}\tan\delta)$$

,όπου:
(x,y) → η θέση του οχήματος-πράκτορα
v → η ταχύτητα του κατά μήκος του νοητού άξονα x
w → η κατεύθυνση (η κατεύθυνση προς την οποία το όχημα δείχνει ανά πάσα στιγμή. Εκφράζεται ως γωνιακή απόσταση σε σχέση με το βορρά)
α → η επιτάχυνση του πράκτορα
β → η γωνία ολίσθησης στο κέντρο της βαρύτητας
δ → η γωνία του μπροστινού τροχού που χρησιμοποιείται ως εντολή διεύθυνσης

Οι παραπάνω υπολογισμοί εμφανίζονται αν καλέσουμε τη μέθοδο step().

4.2.2.2 Χώρος δράσης

Ο χώρος δράσης του οχήματος είναι συνεχής (ContinuousAction), γεγονός που επιτρέπει στον πράκτορα να ρυθμίζει απευθείας τους ελεγκτές χαμηλού επιπέδου της κινηματικής του οχήματος, δηλαδή το γκάζι και τη γωνία διεύθυνσης .Σημειώνεται ότι στο παρόν περιβάλλον ο έλεγχος του γκαζιού και του τιμονιού μπορεί να ενεργοποιηθεί ή να απενεργοποιηθεί μέσω των διαμήκων και πλευρικών διαμορφώσεων, αντίστοιχα. Έτσι, ο χώρος δράσης μπορεί να είναι είτε μιας διάστασης είτε δύο διαστάσεων. Στο πλαίσιο των πειραμάτων που υλοποιήθηκαν στην παρούσα εργασία, εργαστήκαμε με δισδιάστατο χώρο δράσης.

Στο περιβάλλον που χρησιμοποιείται υπάρχει και ο τύπος DiscreteMetaAction, ο οποίος προσθέτει ένα στρώμα ελεγκτών ταχύτητας και διεύθυνσης πάνω από τον συνεχή έλεγχο χαμηλού επιπέδου που περιγράφηκε παραπάνω, έτσι ώστε ο πράκτορας να μπορεί να ακολουθήσει αυτόματα το δρόμο με την επιθυμητή ταχύτητα. Ειδικότερα, οι διαθέσιμες ενέργειες του εν λόγω τύπου συνίστανται στην αλλαγή της λωρίδας προορισμού και της ταχύτητας που χρησιμοποιούνται ως σημεία ρύθμισης για τους ελεγκτές χαμηλού επιπέδου, δράσεις οι οποίες ωστόσο δεν είναι διαθέσιμες για τη λειτουργία του πάρκινγκ. Ο κατάλογος των διαθέσιμων δράσεων μπορεί να προσεγγιστεί με get_available_actions() μέθοδο, ενώ η λήψη μη διαθέσιμης ενέργειας ισοδυναμεί με την ανάληψη δράσης IDLE.

Ο χώρος συνεχούς δράσης περιλαμβάνει το γκάζι ή/και τη γωνία διεύθυνσης. Εφόσον όπως αναφέρθηκε και παραπάνω, είναι ενεργοποιημένα τόσο το γκάζι όσο και το τιμόνι, τίθενται με αυτή τη σειρά: [γκάζι, τιμόνι].Τα διαστήματα χώρου είναι πάντα [-1, 1], αλλά χαρτογραφούνται σε διαστήματα γκαζιού/διεύθυνσης μέσω των παρακάτω διαμορφώσεων.

ACCELERATION_RANGE= (-5, 5,0)

Εύρος επιτάχυνσης: [-x, x], σε m/s^2.

STEERING_RANGE= (-0,7853981633974483, 0,7853981633974483)

Εύρος γωνίας διεύθυνσης: [-x, x], σε rad.

4.2.2.3 Μοντελοποίηση ανταμοιβών

Η επιλογή μιας κατάλληλης συνάρτησης ανταμοιβής που αποδίδει ρεαλιστική βέλτιστη οδηγική συμπεριφορά είναι ένα δύσκολο πρόβλημα, το οποίο δεν αντιμετωπίζεται σε αυτή την εργασία. Συγκεκριμένα, δεν θέλουμε να προσδιορίσουμε κάθε πτυχή της αναμενόμενης οδηγικής συμπεριφοράς στο εσωτερικό της λειτουργίας ανταμοιβής. Αντίθετα, θα προτιμούσαμε να καθορίσουμε μόνο μια λειτουργία ανταμοιβής όσο το δυνατόν απλούστερη και απλή, προκειμένου να δούμε να προκύπτει επαρκής συμπεριφορά από τη μάθηση.

Δεδομένου ότι οι ανταμοιβές πρέπει να οριοθετηθούν και η βέλτιστη πολιτική δεν επηρεάζεται από την κλιμάκωση και τη εναλλαγή ανταμοιβών, επιλέγεται η ομαλοποίηση στην περιοχή [0,1], κατά σύμβαση. Άλλωστε, η ομαλοποίηση των ανταμοιβών έχει επίσης παρατηρηθεί ότι είναι πρακτικά επωφελής στη βαθιά ενισχυτική μάθηση. Σημειώνεται ότι απαγορεύουμε τις αρνητικές ανταμοιβές, καθώς μπορεί να ενθαρρύνουν τον πράκτορα να προτιμήσει να τερματίσει ένα επεισόδιο νωρίς αντί να διακινδυνεύσει να υποστεί αρνητική απόδοση εάν δεν μπορεί να βρεθεί ικανοποιητική τροχιά.

Στο περιβάλλον στάθμευσης, η συνάρτηση ανταμοιβής πρέπει επίσης να καθορίσει τον επιθυμητό προορισμό στόχου. Έτσι, ο όρος ταχύτητας αντικαθίσταται από έναν σταθμισμένο κανόνα p μεταξύ της κατάστασης παράγοντα και της κατάστασης στόχου.

$$R(s,\alpha) = -\|s\text{-}s_g\|_{W,p}^{p} - b\text{collision}$$

,όπου $s = [x, y, v_x, v_y, \cos\psi, \sin\psi]$, $s_g = [x_g, y_g, 0, 0, \cos\psi_g, \sin\psi_g]$

,και $\|x\|_{W,p} = (\sum_i |W_i x_i|^p)^{1/p}$

Χρησιμοποιούμε έναν κανόνα/νόρμα p και όχι έναν ευκλείδειο κανόνα για να έχουμε μια στενότερη αύξηση των ανταμοιβών στο στόχο.

4.2.3 Σχεδίαση Πρακτόρων

4.2.3.1 Model-based agent

Η μάθηση ενίσχυσης χωρίς μοντέλα συνιστά μόνο την πράξη που πρέπει να ληφθεί επί του τρέχοντος χρόνου χωρίς να είναι σε θέση να προβλέψει τις συνέπειές της. Συνεπώς, δεν μπορεί να προβλεφθεί η τροχιά με την εκτέλεση της πολιτικής. Αντίθετα, μέσω των μεθόδων που βασίζονται σε μοντέλα μπορεί να διερευνηθεί η πολιτική για την προσδοκώμενη και προβλεπόμενη τροχιά της. Το πρόβλημα της αυτόνομης στάθμευσης αποτελεί μια Μαρκοβιανή Διαδικασία Αποφάσεων (MDP), οι ιδιότητες της οποίας δικαιολογούν τη χρήση μεθόδων που βασίζονται σε μοντέλα. Η πολιτική εξαρτάται σε μεγάλο βαθμό από τον τελικό στόχο και επομένως το γεγονός αυτό προσθέτει ένα σημαντικό επίπεδο πολυπλοκότητας σε μια διαδικασία μάθησης χωρίς μοντέλα, ενώ η δυναμική του συστήματος είναι εντελώς ανεξάρτητη από τον στόχο και ως εκ τούτου μπορεί να είναι απλούστερη στην εκμάθηση. Επιπλέον, στα πλαίσια μιας βιομηχανικής εφαρμογής, για λόγους ασφαλείας η προγραμματισμένη πορεία πρέπει να είναι γνωστή εκ των προτέρων, πριν από την εκτέλεση.

Το μοντέλο που σχεδιάστηκε αντιπροσωπεύει τη δυναμική του συστήματος. Επιλέχθηκε ένα δομημένο μοντέλο εμπνευσμένο από τα συστήματα Γραμμικού Αμετάβλητου-Χρόνου (Linear Time-Invariant / LTI), σύμφωνα με την παρακάτω εξίσωση:

$$\dot{x} = f_\theta(x, u) = A_\theta(x, u)x + B_\theta(x, u)u$$

,όπου το ζεύγος (x,u) αντιπροσωπεύει το ζεύγος κατάστασης-δράσης (s, α). Διαισθητικά, μαθαίνουμε σε κάθε σημείο (x_t, u_t) τη γραμμικοποίηση της πραγματικής δυναμικής f σε σχέση με το (x, u). Παραμετροποιούμε τους συντελεστές A_θ και B_θ ως δύο πλήρως συνδεδεμένα

δίκτυα με ένα κρυφό στρώμα 64 νευρώνων το καθένα και συνάρτηση ενεργοποίησης την Relu. Στα δεδομένα εισόδου καθενός εκ των δύο νευρωνικών δικτύων εφαρμόζεται γραμμικός μετασχηματισμός ($y=xA^T+b$) μέσω της κλάσης Linear.

Ο πράκτορας αλληλεπιδρά με το περιβάλλον και συλλέγει ssυνολικά 1000 τυχαίες εμπειρίες, οι οποίες οργανώθηκαν σε πλειάδες (Tuple), καθεμία από τις οποίες περιλαμβάνει την προηγούμενη εμπειρία, την ενέργεια που εκτέλεσε ο πράκτορας στο δεδομένο στιγμιότυπο, και την τρέχουσα παρατήρηση.

```
Tuple(torch.Tensor(previous_observation["observation"]), torch.Tensor(action), torch.Tensor(observation["observation"]))
```

Οι εμπειρίες αυτές αποθηκεύονται σε μία μνήμη από την οποία χρησιμοποιείται το 70% των δειγμάτων για την εκπαίδευση του μοντέλου. Το μοντέλο εκπαιδεύεται με εποπτευόμενο τρόπο για να ελαχιστοποιηθεί η απώλεια MSE(Mean Squared Error) $L^2(f_\theta;D)$ πάνω από την παρτίδα εμπειριών D με στοχαστική κάθοδο κλίσης:

$$L^2(f_\theta;D) = \frac{1}{|D|}\sum_{st,at,st+1\epsilon D} ||s_{t+1} + f_\theta(s_t, \alpha_t)||^2$$

Ενώ, για τη βελτιστοποίηση του μοντέλου χρησιμοποιήθηκε η μέθοδος Adam.

Το μοντέλο ανταμοιβής που χρησιμοποιήθηκε αποτελεί μια ζυγισμένη L1 νόρμα μεταξύ της τρέχουσας κατάστασης και του στόχου και δίνεται από την παρακάτω συνάρτηση.

$$\text{Ανταμοιβή} = -(|\Delta x| + |\Delta y|)$$

Η L1-νόρμα, γνωστή και ως Απόσταση Μανχάταν ή Κανόνας Taxicab, αποτελεί το άθροισμα των μεγεθών των φορέων σε ένα χώρο. Είναι ο πιο φυσικός τρόπος μέτρησης της απόστασης μεταξύ των φορέων, δηλαδή το άθροισμα της απόλυτης διαφοράς των συστατικών των φορέων. Παραδοσιακά, σε αυτόν τον κανόνα, όλα τα συστατικά του διανύσματος σταθμίζονται εξίσου, ωστόσο για τον παρόν πράκτορα χρησιμοποιούνται τα παρακάτω βάρη του έχουν προβλεφθεί από τους σχεδιαστές του συγκεκριμένου περιβάλλοντος προσομοίωσης για καθένα από τα στοιχεία του πίνακα κατάστασης.

REWARD_WEIGHTS : [1., 0.3, 0., 0., 0.02, 0.02]

Για τον σχεδιασμό της τροχιάς του πράκτορα, χρησιμοποιούμε δυναμικό μοντέλο εκμάθησης. Για την επίλυση του προβλήματος βέλτιστου ελέγχου, χρησιμοποιείται ένας αλγόριθμος βελτιστοποίησης βάσει δειγματοληψίας που ονομάζεται Μέθοδος Διασταυρούμενης Εντροπίας (Cross-Entropy Method/CEM) και εφαρμόζεται συνήθως σε προβλήματα συνδυαστικά και συνεχή, όπως και το παρόν στο οποίο αναζητείται η βέλτιστη ακολουθία ενεργειών.

4.2.3.1.1 CEM planner

Βελτιστοποιητές τροχιάς για ενισχυτική μάθηση βάσει μοντέλου, όπως η μέθοδος Cross-Entropy (CEM), μπορούν να αποφέρουν σημαντικά αποτελέσματα ακόμη και σε εργασίες πολλών διαστάσεων ελέγχου και με περιβάλλοντα αραιών ανταμοιβών. Η CEM είναι μια στοχαστική τεχνική βελτιστοποίησης που λειτουργεί καλά ακόμη και με μοντέλα που έχουν μάθει, παράγοντας συγκρίσιμες ή ακόμη και υψηλότερες επιδόσεις από τις μεθόδους ενισχυτικής μάθησης χωρίς μοντέλα. [53] Αποτελεί μια τεχνική βελτιστοποίησης χωρίς παράγωγα (Derivative-Free Optimization – DFO) που βρίσκει το βέλτιστο χωρίς να υπολογίζει επιμέρους παραγώγους, και συνεπώς δεν απαιτείται backpropagation.

Η CEM λαμβάνει ένα σωρό εισόδων, βλέπει τις παραγόμενες εξόδους, επιλέγει τις εισόδους που έχουν οδηγήσει στις καλύτερες εξόδους και ενημερώνει διαρκώς τον πράκτορα μέχρι οι έξοδοι του μοντέλου να είναι ικανοποιητικές. [54] Με άλλα λόγια, η καρδιά του εν λόγω

αλγορίθμου είναι η παραγωγή ενός σωρού επεισοδίων από τα οποία πετάει τα «κακά» όσον αφορά τις ανταμοιβές επεισόδια και εκπαιδεύει το νευρωνικό δίκτυο του πράκτορα με τα καλύτερα. Με αυτόν τον τρόπο, το νευρωνικό μαθαίνει να επαναλαμβάνει ενέργειες που οδηγούν σε όλο και καλύτερες αποφάσεις καθώς χρησιμοποιούνται κάθε φορά νέες παρτίδες ελίτ επεισοδίων. Ο πράκτορας πρέπει να εκπαιδευτεί μέχρι μια ορισμένη μέση ανταμοιβή για την παρτίδα των επεισοδίων να προσεγγίσει το όριο (threashold) που έχει τεθεί.

Η παραπάνω μέθοδος επιδιώκει την εύρεση των καλύτερων βαρών ώστε το νευρωνικό να αποφασίσει τις σωστές ενέργειες με βάση την κατάσταση του πράκτορα. Τα βήματα της μεθόδου CEM για την εύρεση των βαρών αυτών περιγράφεται παρακάτω: [55]

Βήμα 1: Σχεδιάζεται ένας σωρός με αρχικά βάρη από μια τυχαία κατανομή. Επιλέγεται γενικά η κατανομή αυτή να είναι Gaussian. Θεωρείται αρχικά ότι ο μέσος είναι μ=0 και η διακύμανση $\sigma^2=1$.

Βήμα 2: Ο πράκτορας συλλέγει ενέργειες από το δίκτυο πολιτικής βασισμένος στα βάρη που επιλέχθηκαν παραπάνω, τρέχει το επεισόδιο και συλλέγει τις ανταμοιβές που παράχθηκαν από το περιβάλλον. Η μέθοδος αξιολόγησης για έναν πράκτορα παίρνει ένα υποψήφιο βάρος ως είσοδο, αναπαράγει ένα επεισόδιο και παράγει την αθροιστική ανταμοιβή από αυτό το επεισόδιο.

Βήμα 3: Εντοπίζονται τα βάρη που παράγουν τις καλύτερες ανταμοιβές, τα οποία και καλούνται “elite” βάρη.

Βήμα 4: Επιλογή νέων βαρών από μια κατανομή που ορίζεται από τα “elite” βάρη. Τα νέα βάρη θα ανασυρθούν από μια γκαουσιανή κατανομή με μέσο μ’ και διακύμανση σ'^2 που προκύπτουν από τα “elite” βάρη που έχουν ήδη υπολογιστεί.

Βήμα 5: Επαναλαμβάνονται τα βήματα 2 με 4 μέχρι οι αμοιβές που προκύπτουν να είναι ικανοποιητικές.

Παρακάτω δίνεται ένας ψευδοκώδικας για την λειτουργία της μεθόδου CEM που περιγράφηκε παραπάνω.

Algorithm 15: CEM planner

Initialize $\mu \epsilon R^d$, $\sigma \epsilon R^d$ ($\mu=0$ and $\sigma=1$)
For iteration = 1, 2,…. do
  Collect n samples of θ_i ~N(μ, diag(σ))
  Perform a noisy evaluation R_i ~θ_i
  Select the top k% of samples which is the elite set
  Fit a Gaussian distribution to elite set and obtain new values for μ, σ
end for
Return the final μ

4.2.3.2 DQN agent

Στον αλγόριθμο DQN εκπαιδεύεται ένα νευρωνικό δίκτυο ενός κρυφού επιπέδου 256 νευρώνων με συνάρτηση ενεργοποίησης την Relu. Όπως λοιπόν και στο παραπάνω μοντέλο, έτσι και κατά την εκπαίδευση του DQN πράκτορα, στα δεδομένα εισόδου του νευρωνικού δικτύου εφαρμόζεται γραμμικός μετασχηματισμός μέσω της κλάσης Linear. Ο πράκτορας αλληλεπιδρά με το περιβάλλον, συλλέγει εμπειρίες και τις αποθηκεύει σε μία μνήμη από την οποία χρησιμοποιείται το 70% των δειγμάτων για την εκπαίδευση του μοντέλου. Οι εμπειρίες σε αντίθεση με το παραπάνω μοντέλο αποθηκεύονται υπό την παρακάτω μορφή, η οποία περιλαμβάνει την τρέχουσα κατάσταση, την ενέργεια του πράκτορα, την ανταμοιβή, την επόμενη κατάσταση, καθώς και μια μεταβλητή (done) που υποδεικνύει τον τερματισμό του επεισοδίου.

```
Transition = namedtuple('Transition', ('state', 'action', 'reward', 'next_state', 'done'))
```

Το μοντέλο εκπαιδεύεται πάλι με εποπτευόμενο τρόπο για να ελαχιστοποιηθεί η απώλεια MSE(Mean Squared Error) $L^2(f_\theta;D)$ πάνω από την παρτίδα εμπειριών D με στοχαστική κάθοδο κλίσης. Ενώ, για τη βελτιστοποίηση του μοντέλου χρησιμοποιήθηκε, όπως και παραπάνω, η μέθοδος Adam.

Ακόμη, χρησιμοποιήθηκε η τεχνική αναπαραγωγής εμπειριών (replay experience), σύμφωνα με την οποία ο πράκτορας αποθηκεύει παρελθοντικά στιγμιότυπα εμπειρίας, χωρητικότητας 100.000 δειγμάτων (των πιο πρόσφατων) και η εκπαίδευση του νευρωνικού δικτύου πραγματοποιείται από 2.500 τυχαία δείγματα των στιγμιότυπων, κάθε φορά.

Ένα σημαντικό ζήτημα για έναν DQN πράκτορα αποτελεί η εξισορρόπηση μεταξύ εκμετάλλευσης και εξερεύνησης για την λήψη της βέλτιστης απόφασης κάθε φορά. Υπό το παραπάνω πρίσμα, χρησιμοποιείται ε-άπληστη τεχνική, σύμφωνα με την οποία ορίζεται μια παράμετρος ε, που αποτελεί την πιθανότητα εκλογής μιας πράξης τυχαία. Ενώ κατά την εκκίνηση του αλγορίθμου, η τιμή της παραμέτρου ορίζεται ίση με τη μονάδα, όσο περνούν οι εποχές εκπαίδευσης μειώνεται κατά ένα μέγεθος ελάττωσης κάθε φορά (ε-decay = 0.999995). Αρχικά, λοιπόν, η επιλογή των ενεργειών είναι απολύτως τυχαία και σταδιακά με το πέρας του χρόνου ο αλγόριθμος επιλέγει δράσεις όλο και περισσότερο με βάση την πολιτική του.

4.2.3.3 Soft Actor-Critic agent

Στον αλγόριθμο Soft Actor-Critic (SAC) εκπαιδεύονται δύο νευρωνικά δίκτυα, ένα για τον δράστη(actor) και ένα για τον κριτή(critic). Αναφορικά με το νευρωνικό δίκτυο του δράστη, αυτό αποτελείται από ένα κρυφό επίπεδο 256 νευρώνων με συνάρτηση ενεργοποίησης την Relu. Στα δεδομένα εισόδου, λοιπόν, του νευρωνικού δικτύου του δράστη εφαρμόζεται γραμμικός μετασχηματισμός μέσω της κλάσης Linear. Το δίκτυο που εκπαιδεύεται ακολουθεί γκαουσιανή πολιτική (GaussianPolicy network), δίνει δηλαδή μια γκαουσιανή κατανομή πιθανότητας όλων των δυνατών πράξεων (όπως στο policy gradient). Ενώ το νευρωνικό δίκτυο του κριτή(critic) αποτελείται από δύο όμοια γραμμικά νευρωνικά δίκτυα (QNetworks) ενός κρυφού επιπέδου 256 νευρώνων το καθένα με συνάρτηση ενεργοποίησης την Relu. Το δίκτυο αυτό ονομάζεται TwinnedQNetwork και δίνει μια εκτίμηση αξίας της κατάστασης στην οποία βρίσκεται ο πράκτορας κάθε στιγμή.

Ο πράκτορας αλληλεπιδρά με το περιβάλλον, συλλέγει εμπειρίες και τις αποθηκεύει σε μία μνήμη(buffer) την οποία χρησιμοποιεί για την εκπαίδευση του μοντέλου. Οι εμπειρίες σε ακολουθούν την παρακάτω δομή, η οποία περιλαμβάνει την τρέχουσα κατάσταση, την ενέργεια του πράκτορα, την ανταμοιβή, την επόμενη κατάσταση, καθώς και μια μεταβλητή (episode_done) που υποδεικνύει τον τερματισμό του επεισοδίου.

```
(state, action, reward, next_state, episode_done)
```

Ενώ κάθε στοιχείο του παραπάνω λεξικού αποθηκεύεται στην μνήμη ξεχωριστά ανά κατηγορία μαζί με τα υπόλοιπα επιμέρους δεδομένα προγενέστερων εμπειριών.

Η συνάρτηση απώλειας είναι διαφορετική για τα δύο ήδη νευρωνικών δικτύων που επιστρατεύονται για την εκπαίδευση του πράκτορα SAC. Αναλυτικότερα, για το δίκτυο του κριτή, ως συνάρτηση απώλειας λογίζεται ο μέσος όρος των σταθμισμένων τετραγωνικών διαφορών μεταξύ των τιμών της τρέχουσας κατάστασης όπως υπολογίζεται από το μοντέλο του κριτή και των τιμών της κατάστασης στόχου. Χρησιμοποιείται, λοιπόν, η παρακάτω συνάρτηση:

```
mean(( current_q1 - target_q )**2 * weights )
```

Για το δίκτυο του δράστη, στόχος της πολιτικής που χρησιμοποιήθηκε αποτελεί η μεγιστοποίηση της ποσότητας *(q + alpha * εντροπία)* με βάρη προτεραιότητας, όπου q είναι οι προσδοκώμενες τιμές q, alpha μια παράμετρος εκμάθησης που καθορίζει μια ποσότητα αλλαγής στους συντελεστές σε κάθε ενημέρωση. Επομένως, η συνάρτηση απώλειας που πρέπει να ελαχιστοποιηθεί δίνεται από την παρακάτω συνάρτηση:

*mean((-q – alpha * entropy) * weights)*

Η προσέγγιση που χρησιμοποιήθηκε για τον πράκτορα είναι η εξής. Με το πέρας κάθε επεισοδίου, αποθηκεύονται οι διαδοχικές καταστάσεις, οι ενέργειες και οι ανταμοιβές του πράκτορα ως ιστορικό. Το ιστορικό αυτό χρησιμοποιείται στην διαδικασία της εκπαίδευσης, η οποία πραγματοποιείται από 256 τυχαία δείγματα στιγμιοτύπων κάθε φορά. Τα βάρη του νευρωνικού δικτύου ανανεώνονται ανά κάποιο συγκεκριμένο αριθμό βημάτων/στιγμιότυπων (20000 στην δεδομένη περίπτωση), οπότε και το ιστορικό διαγράφεται για να δεχθεί νέες εμπειρίες. Ο πράκτορας δρα με πλήρη τυχαιότητα, εξερευνώντας το περιβάλλον του, πριν φτάσει τα 1000 στιγμιότυπα (10 επεισόδια) και έπειτα αρχίζει να εκμεταλλεύεται και να μαθαίνει από τις προγενέστερες εμπειρίες του ανανεώνοντας τα βάρη και τις επιμέρους παραμέτρους του δικτύου.

4.2.3.4 Soft Actor-Critic with HER

Το HER είναι ένας αλγόριθμος που λειτουργεί με off-policy μεθόδους (DQN, SAC, DDPG). Όπως έχει περιγραφεί και στο αντίστοιχο κεφάλαιο, η HER βασίζεται στο ότι ακόμα και αν δεν επιτευχθεί ο επιθυμητός στόχος, μπορεί να έχει επιτευχθεί ένας άλλος κατά τη διάρκεια της προσομοίωσης. Δημιουργεί, λοιπόν, "εικονικές" μεταβάσεις , αλλάζοντας τον προσδοκώμενο εξ αρχής στόχο από προηγούμενα επεισόδια. Η λογική του Soft Actor-Critic είναι αυτή που αναλύθηκε και παραπάνω, με τη διαφορά ότι ο παρόν πράκτορας διαχειρίζεται με διαφορετικό τρόπο τις ανταμοιβές. Πιο συγκεκριμένα, η τεχνική της «εκ των υστέρων επανάληψης εμπειρίας» (Hindsight Experience Replay), βασίζεται στην αποθήκευση κάθε μετάβασης του πράκτορα (ολόκληρης της τροχιάς του) είτε αυτή έχει το επιθυμητό αποτέλεσμα είτε όχι και στην υπόθεση ότι ο στόχος εξ αρχής ήταν η κατάσταση στην οποία κατέληξε το όχημα στο τέλος του επεισοδίου. Υπό το παραπάνω πρίσμα, όλες οι ανταμοιβές είναι θετικές και έχουν στόχο την εκμάθηση του πράκτορα μέσα από τα λάθη του.

Αξίζει να σημειωθεί ότι το HER δεν αποτελεί πλέον ξεχωριστό αλγόριθμο, αλλά μια κλάση μνήμης επανάληψης(replay buffer),που ονομάζεται HerReplayBuffer, και η οποία περνά σε έναν off-policy αλγόριθμο (εν προκειμένω στον SAC), όταν χρησιμοποιείτε το MultiInputPolicy. Ο σχεδιασμός του παρόντος πράκτορα βασίστηκε στην υλοποίηση του μοντέλου SAC όπως αυτό περιγράφεται στην βιβλιοθήκη Stable-baselines3.

4.3 Πειραματικά αποτελέσματα

4.3.1 Προσέγγιση πειραμάτων

Παρακάτω ακολουθεί η συνοπτική περιγραφή της διαδικασίας που ακολουθήθηκε για τον σχεδιασμό των πρακτόρων και την εκτέλεση των πειραμάτων. Για τους τέσσερις αλγόριθμους που υλοποιήθηκαν, έγινε μελέτη των επιμέρους παραμέτρων, οι οποίες ορίστηκαν έτσι ώστε να επιτευχθεί η βέλτιστη αποτελεσματικότητα του εκάστοτε πειράματος. Σημειώνεται ότι για όλους τους πράκτορες που σχεδιάστηκαν, το περιβάλλον ήταν μονοπρακτορικό (δεν υπήρχαν επιπλέον οχήματα που να κινούνται στο χώρο) και δεν περιλάμβανε εμπόδια που να απαιτούν πολυπλοκότερη διαχείριση. Συνεπώς, το όχημα σε κάθε πείραμα κινείται σε άδειο χάρτη, ενώ η αρχική (εκκίνηση) και τελική θέση (στόχος) είναι τυχαίες και μεταβάλλονται από επεισόδιο σε επεισόδιο.

Έπειτα, λοιπόν, από αρκετή μελέτη και εκτέλεση πλήθος πειραμάτων και δοκιμών με τις παραμέτρους των επιμέρους διαφορετικών μοντέλων, κατορθώθηκε η επιτυχής επίτευξη αρκετών εξ αυτών και η προσέγγιση του επιθυμητού στόχου με βέλτιστο τρόπο κάθε φορά για το εκάστοτε μοντέλο. Στις παρακάτω ενότητες, αναλύονται τα πειραματικά αποτελέσματα των προσομοιώσεων και τα μοντέλα εκπαίδευσης που χρησιμοποιήθηκαν για κάθε αλγόριθμο.

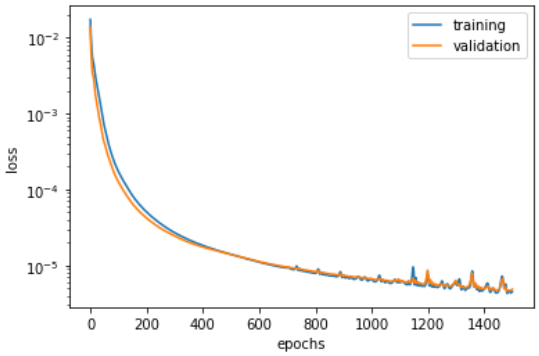
4.3.2 Model-based agent

Οι κυριότερες παράμετροι που επηρεάζουν την εκπαίδευση του μοντέλου παρουσιάζονται στον παρακάτω πίνακα με τις τιμές που μετά από πλήθος πειραμάτων επιλέχθηκαν ως βέλτιστες για την εκπαίδευση του αλγορίθμου.

***Πίνακας 1**: Παράμετροι model-based CEM planner agent*

ΠΑΡΑΜΕΤΡΟΙ	***Model-based agent (CEM planner)***
reward_type	weighted L1-norm (απόλυτη απόσταση από στόχο)
learning rate	0.1
gamma	None
loss function	MSELoss (mean squared error)
train epochs	1500
optimizer	Adam
episodes	10
episode steps	100

Όπως διαπιστώνεται από την παρακάτω γραφική, η συνάρτηση απώλειας σταθεροποιείται τείνοντας στο μηδέν κατά τη διάρκεια κατά την εκπαίδευση του μοντέλου.



***Εικόνα 12**: Training and validation losses*

Evaluation

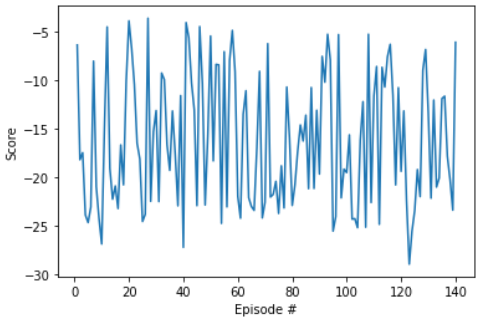
Στη συνέχεια εκτελέστηκε πλήθος 10000 στιγμιοτύπων με στόχο την αξιολόγηση του μοντέλου. Για κάθε επεισόδιο που τερματίζει ή για κάθε επεισόδιο που ο πράκτορας δεν καταφέρνει να φτάσει στον στόχο του, εκτυπώνονται οι ανταμοιβές και η πληροφορία της επιτυχίας ή αποτυχίας. Όπως φαίνεται και από τα πειραματικά αποτελέσματα που παρατίθενται παρακάτω, από τα 120 επεισόδια που εκτελέστηκαν, στα 55 ο πράκτορας δεν έφτασε στον προσδοκώμενο στόχο (αν και όπως διαπιστώθηκε από τα πειράματα πάντα κατέληγε πολύ κοντά σε αυτόν), γεγονός που μπορεί να συμπεραθεί παρατηρώντας και μόνο τις ανταμοιβές που προκύπτουν στο τέλος κάθε επεισοδίου. Συγκεκριμένα, σε όσα επεισόδια το όχημα δεν κατόρθωσε να παρκάρει στην θέση-στόχο, οι ανταμοιβές παρουσιάζουν αξιοσημείωτη απόκλιση από αυτές των επιτυχημένων επεισοδίων.

```
Reward: -6.408624742813211 Success? True
Reward: -18.21881441701877 Success? False
Reward: -17.49331918229207 Success? False
Reward: -23.905949314873173 Success? False
Reward: -24.68279804324033 Success? False
Reward: -23.078696744424146 Success? False
Reward: -8.064599523652982 Success? True
Reward: -20.98690628222053 Success? False
Reward: -24.25096574712189 Success? False
Reward: -26.890928441281847 Success? False
Reward: -14.953547359034124 Success? True
Reward: -4.547140350140137 Success? True
Reward: -19.11516973189291 Success? False
Reward: -22.279569718296038 Success? False
Reward: -20.9148884691467 Success? False
Reward: -23.24310168988562 Success? False
Reward: -16.692354956545756 Success? True
Reward: -20.802345426219798 Success? False
Reward: -10.139444910164448 Success? True
Reward: -3.922147669848006 Success? True
Reward: -6.7629834648467035 Success? True
Reward: -10.486651185471654 Success? True
Reward: -16.520145994661316 Success? True
Reward: -18.131828661467225 Success? False
Reward: -24.575982428359424 Success? False
Reward: -23.826180662952545 Success? False
Reward: -3.6542691963378373 Success? True
```

```
Reward: -10.784872315626385 Success? True
Reward: -21.177927971802745 Success? False
Reward: -13.125568691003243 Success? True
Reward: -19.693222231886676 Success? False
Reward: -7.588044829984178 Success? True
Reward: -10.240445596823776 Success? True
Reward: -5.302105116413477 Success? True
Reward: -7.940010835675963 Success? True
Reward: -25.56611269800598 Success? False
Reward: -24.06183115137172 Success? False
Reward: -5.333024563365768 Success? True
Reward: -22.14796187817502 Success? False
Reward: -19.1847465555925933 Success? False
Reward: -19.539243764986196 Success? False
Reward: -15.6564665554654888 Success? True
Reward: -24.332780539405235 Success? False
Reward: -24.27868333155332 Success? False
Reward: -25.213166906863194 Success? False
Reward: -16.213215557111965 Success? False
Reward: -12.241446926136032 Success? True
Reward: -25.163656456712808 Success? False
Reward: -5.299597640508014 Success? True
Reward: -22.623120206794916 Success? False
Reward: -11.667955759025867 Success? True
Reward: -8.621270771436656 Success? True
Reward: -24.868780533043058 Success? False
Reward: -8.704129511897024 Success? True
Reward: -10.73493958817552 Success? True
Reward: -7.615868531097444 Success? True
Reward: -6.338148259584246 Success? True
Reward: -11.577571067932066 Success? True
Reward: -20.813616164768668 Success? False
Reward: -10.799427487788817 Success? True
Reward: -19.410498695270704 Success? False
```

Παρακάτω φαίνεται η γραφική παράσταση των συνολικών ανταμοιβών κάθε επεισοδίου της παραπάνω αξιολόγησης του αλγορίθμου και παρά το ικανοποιητικό ποσοστό επιτυχίας του μοντέλου, παρατηρείται έντονη διακύμανσή αυτών.

Σημειώνεται ότι η ανταμοιβή κάθε επεισοδίου προκύπτει αθροιστικά από τις επιμέρους ανταμοιβές των στιγμιοτύπων του, που μετρώνται σε κλίμακα από 0 έως 1. Συνεπώς, αναλογικά θα μπορούσε να διαιρεθούν οι παρακάτω τιμές με το 100 (τα στιγμιότυπα κάθε επεισοδίου), ώστε να προκύψει η ανταμοιβή καθενός από αυτά στην ίδια κλίμακα. Επιπρόσθετα, αξίζει να επισημανθεί ότι ικανοποιητική θεωρείται μια ανταμοιβή μεγαλύτερη από -0.25 προσεγγιστικά.

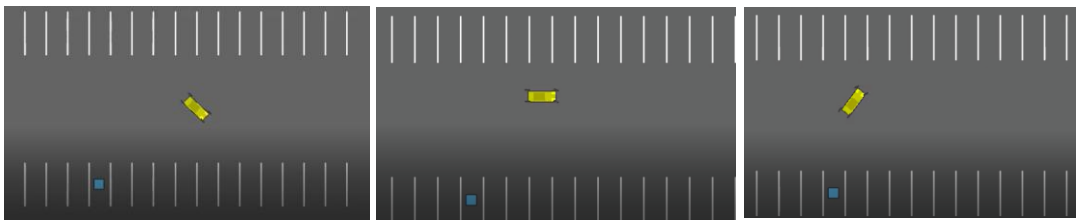


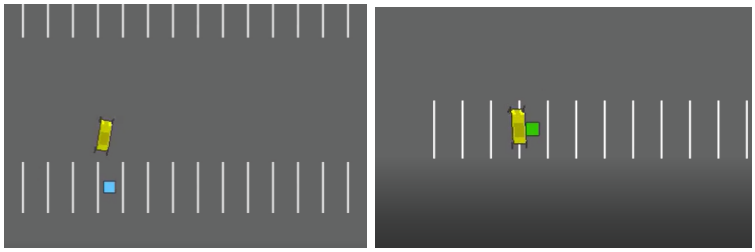
***Εικόνα 13**: Model-based CEM planner evaluation rewards*

Optimization

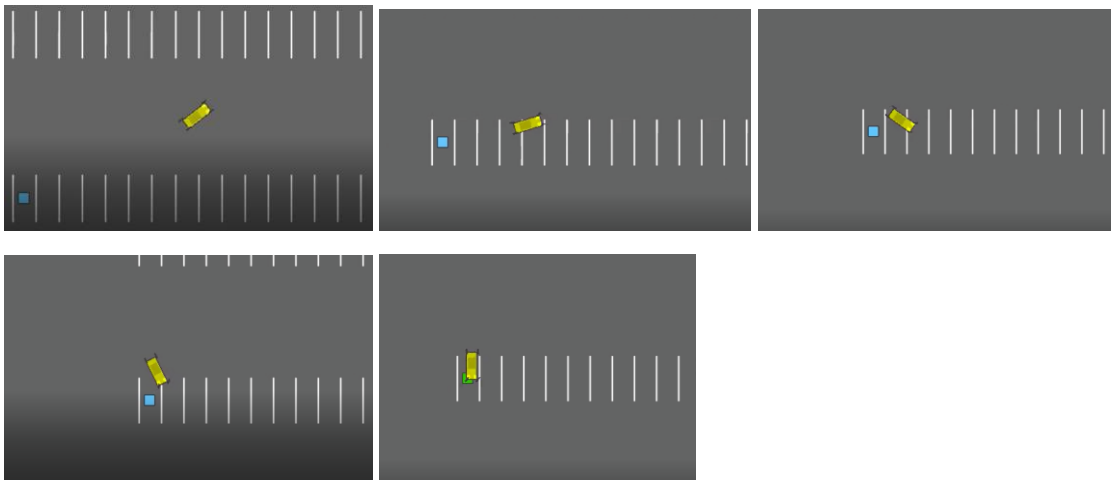
Παρουσιάζονται στιγμιότυπα από κάποια επιτυχημένα επεισόδια του πράκτορα, όπου και παρατηρείται πως η αρχική και τελική θέση του οχήματος είναι τυχαία και διαφέρει από επεισόδιο σε επεισόδιο, ενώ η τροχιά που ακολουθείται σε κάθε περίπτωση είναι η εκάστοτε βέλτιστη όπως προέκυψε από τον CEM planner.

Παράδειγμα Α.

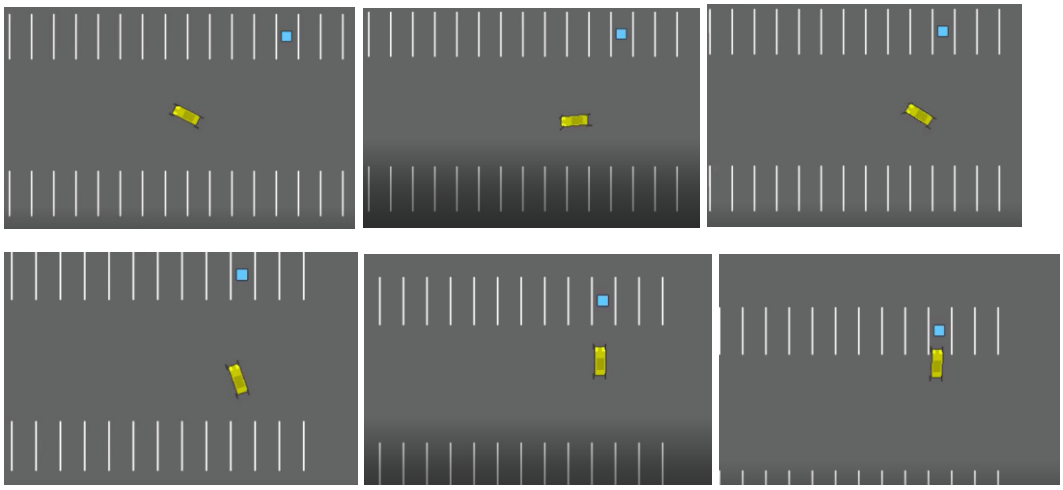




Παράδειγμα Β.



Παράδειγμα C.



Αξίζει να σημειωθεί ότι κατά την διάρκεια των δοκιμών η πλειονότητα των επεισοδίων ήταν επιτυχημένα και τελικά το όχημα έφτανε στην θέση-στόχο αρκετά σύντομα και με τον ζητούμενο προσανατολισμό. Επομένως μπορεί να θεωρηθεί ότι ο εν λόγω αλγόριθμος επιλύει το ζήτημα του πάρκινγκ με αυτόνομη πλοήγηση.

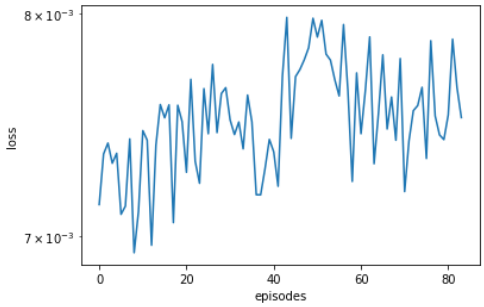
4.3.3 DQN agent

Οι κυριότερες παράμετροι που επηρεάζουν την εκπαίδευση του μοντέλου παρουσιάζονται στον παρακάτω πίνακα με τις τιμές που μετά από πλήθος πειραμάτων επιλέχθηκαν ως βέλτιστες για την εκπαίδευση του αλγορίθμου.

Πίνακας 2: Παράμετροι DQN agent

ΠΑΡΑΜΕΤΡΟΙ	***DQN agent***
reward_type	Lp-norm (όπως περιγράφηκε στην αντίστοιχη ενότητα μοντελοποίησης ανταμοιβών)
learning rate	0.0001
gamma	0.99
loss function	MSELoss (mean squared error)
optimizer	Adam
total episodes	200
steps/ episode	100
batch size	2500
epsilon min	0.999995
episodes before train (steps)	100 (10000)
episodes before update the target model	150

Η εκπαίδευση του DQN πράκτορα απαιτούσε μεγαλύτερο υπολογιστικό κόστος και χρόνο συγκριτικά με αυτήν του πράκτορα βασισμένου σε μοντέλο που περιγράφηκε στην προηγούμενη ενότητα και τα αποτελέσματα δεν ήταν επαρκώς ικανοποιητικά. Συγκεκριμένα, χρειάστηκαν, μετά από δοκιμές, τουλάχιστον 100 επεισόδια, ώστε ο πράκτορας να προσεγγίζει εν τέλει έστω και ένα 40% επιτυχίας. Ένα πρόβλημα που αντιμετωπίστηκε κατά την διάρκεια των πειραμάτων και ίσως οφείλεται και στην μη επιτυχία του παρόντος μοντέλου είναι οι κυκλικές τροχιές. Ειδικότερα, λόγω της άπληστης πολιτικής που ακολουθείται στην επιλογή των ενεργειών, ο πράκτορας στην πλειονότητα των περιπτώσεων διάλεγε δράσεις που τον οδηγούν σε θέσεις που έχει βρεθεί ήδη στο παρελθόν με αποτέλεσμα αρκετές φορές να αποπροσανατολίζεται από τον στόχο του. Μετά από αρκετές δοκιμές το ελάχιστο κόστος ,που κατορθώθηκε να προσεγγιστεί, έτεινε στο 0.007-0.008 όπως φαίνεται από την παρακάτω γραφική, παρουσιάζοντας κάποιες διακυμάνσεις, ωστόσο.



Εικόνα 14: Episode losses

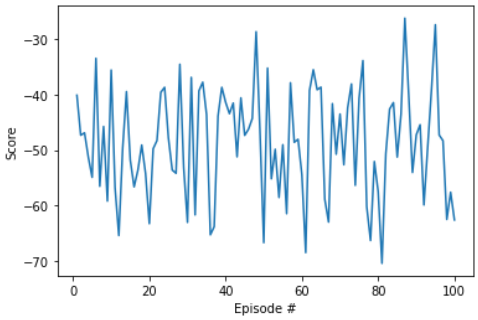
Evaluation

Στη συνέχεια εκτελέστηκε πλήθος 10000 στιγμιοτύπων με στόχο την αξιολόγηση του μοντέλου. Για κάθε επεισόδιο που τερματίζει ή για κάθε επεισόδιο που ο πράκτορας δεν καταφέρνει να φτάσει στον στόχο του, εκτυπώνονται οι ανταμοιβές και η πληροφορία της επιτυχίας ή αποτυχίας. Όπως φαίνεται και από τα πειραματικά αποτελέσματα που παρατίθενται παρακάτω, από τα 100 επεισόδια που εκτελέστηκαν, κανένα από αυτά δεν στέφτηκε με επιτυχία, γεγονός που μπορεί να συμπεραθεί παρατηρώντας και μόνο τις ανταμοιβές που προκύπτουν στο τέλος κάθε επεισοδίου. Παρόλα αυτά, κατά την οπτικοποίηση κάποιων επεισοδίων βρέθηκαν κάποια, αν και ελάχιστα, που προσέγγιζαν τον στόχο.

```
Reward: -40.16192648569911 Success? False
Reward: -47.346952562184015 Success? False
Reward: -46.909478779344155 Success? False
Reward: -51.29596077777571 Success? False
Reward: -54.906298136561125 Success? False
Reward: -33.530599042456366 Success? False
Reward: -56.50782457878583 Success? False
Reward: -45.77184761673793 Success? False
Reward: -59.16547708167197 Success? False
Reward: -35.63704925942772 Success? False
Reward: -56.733690125376924 Success? False
Reward: -65.33724370024254 Success? False
Reward: -49.730206568722664 Success? False
Reward: -39.49868060062353 Success? False
Reward: -51.80101658267526 Success? False
Reward: -56.60273033867546 Success? False
Reward: -53.606402219368 Success? False
Reward: -49.09555304383664 Success? False
Reward: -54.17106846690726 Success? False
Reward: -63.214256877023644 Success? False
Reward: -49.77166903556794 Success? False
Reward: -48.288768190463 59 Success? False
Reward: -39.64180391710701 Success? False
Reward: -38.73798843390877 Success? False
Reward: -47.831418145974204 Success? False
Reward: -53.58287725150667 Success? False
Reward: -54.16760024260912 Success? False
Reward: -34.602949995501 71 Success? False
Reward: -53.1056206998739 Success? False
Reward: -62.98338194188334 Success? False
Reward: -36.95184752474107 Success? False
Reward: -61.64139076008384 Success? False
Reward: -39.377953307857915 Success? False
```

```
Reward: -35.284428990033774 Success? False
Reward: -55.16125719535203 Success? False
Reward: -49.85967840819468 Success? False
Reward: -58.51576133506348 Success? False
Reward: -49.03974424027474 Success? False
Reward: -61.42154657239745 Success? False
Reward: -37.91622511035905 Success? False
Reward: -48.604575245814345 Success? False
Reward: -48.06890601231271 Success? False
Reward: -54.43884865968843 Success? False
Reward: -68.41101434242648 Success? False
Reward: -39.31169022418018 Success? False
Reward: -35.5827500279315 Success? False
Reward: -39.18101781573237 Success? False
Reward: -38.70809646848129 Success? False
Reward: -58.8165730906981 Success? False
Reward: -62.93092202156625 Success? False
Reward: -41.669614246818085 Success? False
Reward: -50.75092442387994 Success? False
Reward: -43.531386562540085 Success? False
Reward: -52.64330823460739 Success? False
Reward: -42.42284347246532 Success? False
Reward: -38.13264025336005 Success? False
Reward: -56.3342948680 56965 Success? False
Reward: -40.71967927710517 Success? False
Reward: -33.934188665783964 Success? False
Reward: -60.14551101744993 Success? False
Reward: -66.23922895218061 Success? False
```

Παρακάτω φαίνεται η γραφική παράσταση των συνολικών ανταμοιβών κάθε επεισοδίου της παραπάνω αξιολόγησης του αλγορίθμου και διαπιστώνεται η έντονη διακύμανσή τους, ενδεικτικό της αποτυχίας του μοντέλου.

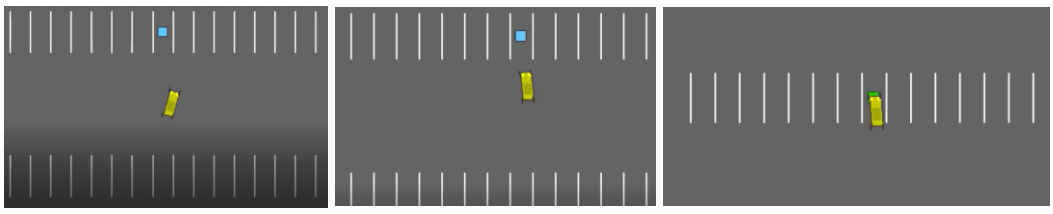


***Εικόνα 15**: DQN evaluation rewards*

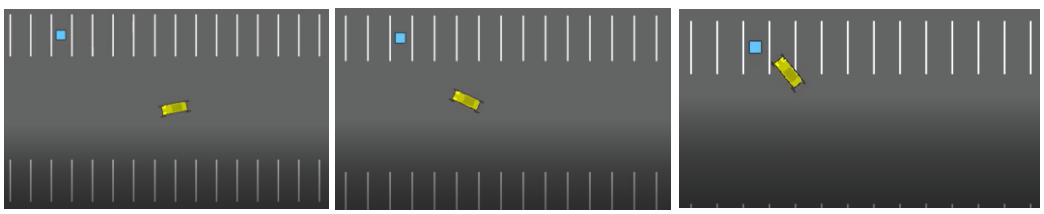
Optimization

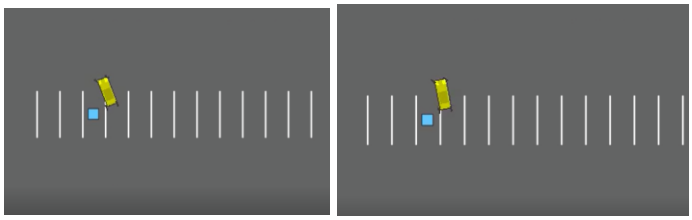
Παρουσιάζονται στιγμιότυπα από κάποια επιτυχημένα επεισόδια του πράκτορα, όπου και παρατηρείται πως η αρχική και τελική θέση του οχήματος είναι τυχαία και διαφέρει από επεισόδιο σε επεισόδιο.

Παράδειγμα Α.

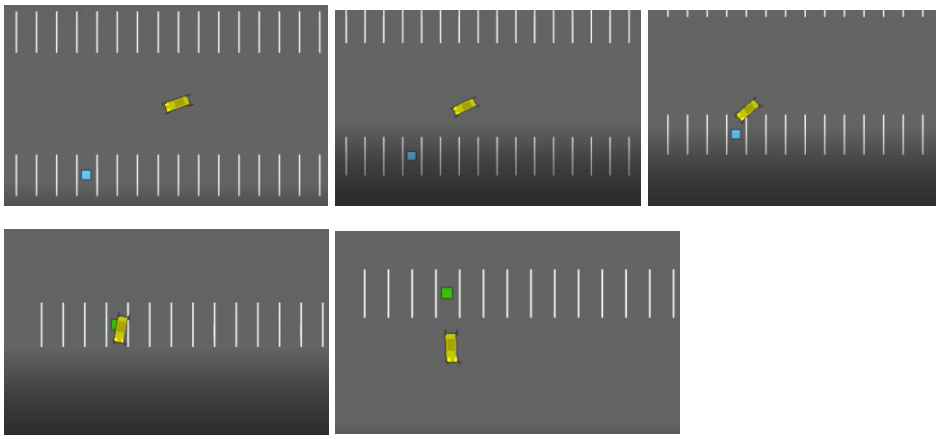


Παράδειγμα Β.





Παράδειγμα C.



Να επισημανθεί, ότι όταν το όχημα έπρεπε να αλλάξει τον προσανατολισμό του για να παρκάρει ορθά, γεγονός που απαιτούσε πολυπλοκότερους ελιγμούς, ο πράκτορας δεν κατόρθωνε να προσεγγίσει την θέση-στόχο. Αντίθετα, σε περιπτώσεις οπότε η προσέγγιση του στόχου μπορούσε να πραγματοποιηθεί με λιγότερες κινήσεις και όχι πολύπλοκους ελιγμούς, ο πράκτορας συμπεριφερόταν καλύτερα.

4.3.4 Soft Actor-Critic

Οι κυριότερες παράμετροι που επηρεάζουν την εκπαίδευση του μοντέλου παρουσιάζονται στον παρακάτω πίνακα με τις τιμές που μετά από πλήθος πειραμάτων επιλέχθηκαν ως βέλτιστες για την εκπαίδευση του αλγορίθμου.

***Πίνακας 3**: Παράμετροι Soft Actor-Critic agent*

ΠΑΡΑΜΕΤΡΟΙ	***SAC agent***
reward_type	Lp-norm (όπως περιγράφηκε στην αντίστοιχη ενότητα μοντελοποίησης ανταμοιβών)
learning rate	0.0001
gamma	0.99
critic loss function	*(όπως περιγράφηκε σε προηγούμενη ενότητα)*
actor loss function	*(όπως περιγράφηκε σε προηγούμενη ενότητα)*
optimizer	Adam
total episodes	200
steps/ episode	100
batch size	256
tau	0.005
alpha	0.6
episodes before train (steps)	100 (10000)
episodes before update the target model	150

Evaluation

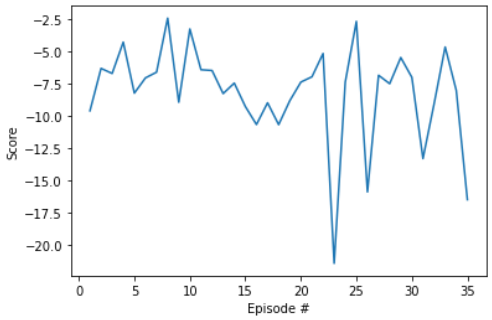
Στη συνέχεια εκτελέστηκε πλήθος 10000 στιγμιοτύπων με στόχο την αξιολόγηση του μοντέλου. Για κάθε επεισόδιο που τερματίζει ή για κάθε επεισόδιο που ο πράκτορας δεν καταφέρνει να φτάσει στον στόχο του, εκτυπώνονται οι ανταμοιβές και η πληροφορία της

επιτυχίας ή αποτυχίας. Όπως φαίνεται και από τα πειραματικά αποτελέσματα που παρατίθενται παρακάτω, από τα 100 επεισόδια που εκτελέστηκαν, στα 24 ο πράκτορας δεν έφτασε στον προσδοκώμενο στόχο, γεγονός που μπορεί να συμπεραθεί παρατηρώντας και μόνο τις ανταμοιβές που προκύπτουν στο τέλος κάθε επεισοδίου. Συγκεκριμένα, σε όσα επεισόδια το όχημα δεν κατόρθωσε να παρκάρει στην θέση-στόχο, η ανταμοιβές παρουσιάζουν αξιοσημείωτη απόκλιση από αυτές των επιτυχημένων επεισοδίων.

```
Reward: -9.2949254223896 Success? True
Reward: -9.97043353518907 Success? True
Reward: -16.63955730461978 Success? False
Reward: -19.317919306495128 Success? False
Reward: -13.198637550428378 Success? True
Reward: -17.654807979448847 Success? False
Reward: -22.962835945376245 Success? False
Reward: -11.611776498773029 Success? True
Reward: -27.23389875321821 Success? False
Reward: -4.102836893817231 Success? True
Reward: -5.659166774748313 Success? True
Reward: -10.288989625071473 Success? True
Reward: -13.064129809584056 Success? True
Reward: -22.946622885198494 Success? False
Reward: -4.510239038343615 Success? True
Reward: -10.373007547489955 Success? True
Reward: -22.878765884366768 Success? False
Reward: -15.610118332937958 Success? False
Reward: -5.483347067134567 Success? True
Reward: -18.339910402842875 Success? False
Reward: -8.401764962599282 Success? True
Reward: -8.458427182624819 Success? True
Reward: -24.770673977604158 Success? False
Reward: -7.094379439559621 Success? True
Reward: -23.07443341304664 Success? False
Reward: -7.980026501566595 Success? True
Reward: -4.908198118102768 Success? True
Reward: -9.167085768488095 Success? True
```

```
Reward: -5.302105116413477 Success? True
Reward: -7.940010835675963 Success? True
Reward: -25.56611269800598 Success? False
Reward: -24.06183115137172 Success? False
Reward: -5.333024563365768 Success? True
Reward: -22.14796187817502 Success? False
Reward: -19.184746555925933 Success? False
Reward: -19.539243764986196 Success? False
Reward: -15.656466555465888 Success? True
Reward: -24.332780539405235 Success? False
Reward: -24.27868333155332 Success? False
Reward: -25.213166906863194 Success? False
Reward: -16.213215557111965 Success? False
Reward: -12.241446926136032 Success? True
Reward: -25.163656456712808 Success? False
Reward: -5.299597640508014 Success? True
Reward: -22.623120206794916 Success? False
Reward: -11.667955759025867 Success? True
Reward: -8.621270771436656 Success? True
Reward: -24.868780533043058 Success? False
Reward: -8.704129511897024 Success? True
Reward: -10.73493958817552 Success? True
Reward: -7.615868531097444 Success? True
Reward: -6.338148259584246 Success? True
Reward: -11.577571067932066 Success? True
Reward: -20.81361616476868 Success? False
Reward: -10.79942748778817 Success? True
```

Παρακάτω φαίνεται η γραφική παράσταση των συνολικών ανταμοιβών κάθε επεισοδίου της παραπάνω αξιολόγησης του αλγορίθμου και παρά το ικανοποιητικό ποσοστό επιτυχίας του μοντέλου, παρατηρείται έντονη διακύμανσή αυτών.

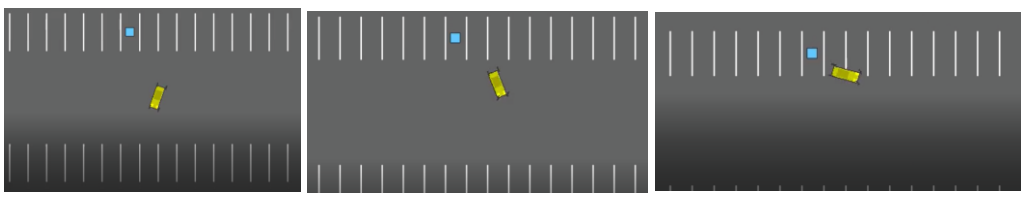


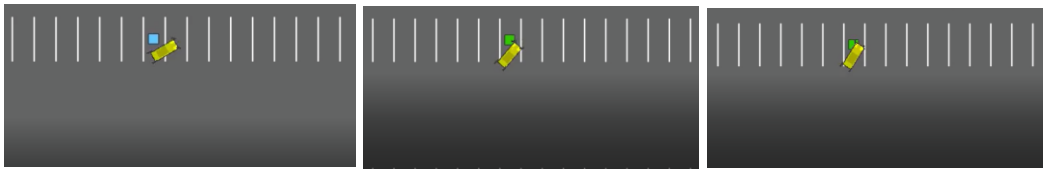
***Εικόνα 16**: SAC evaluation rewards*

Optimization

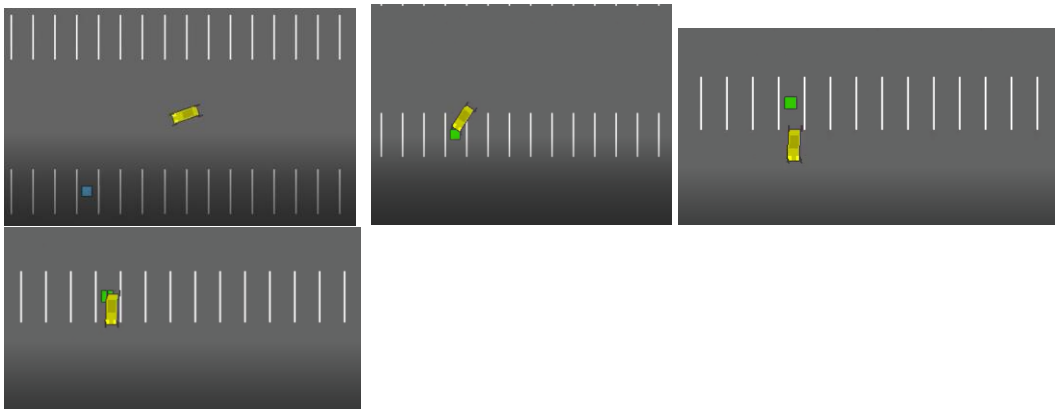
Παρουσιάζονται στιγμιότυπα από κάποια επιτυχημένα επεισόδια του πράκτορα, όπου και παρατηρείται πως η αρχική και τελική θέση του οχήματος είναι τυχαία και διαφέρει από επεισόδιο σε επεισόδιο.

Παράδειγμα Α.

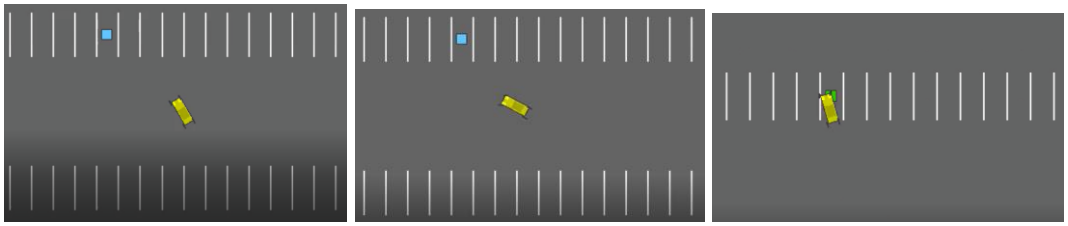




Παράδειγμα Β.



Παράδειγμα C.



4.3.5 Soft Actor-Critic –HER

Ο σχεδιασμός του παρόντος αλγορίθμου βασίστηκε στην υλοποίηση του μοντέλου SAC όπως αυτό περιγράφεται στην βιβλιοθήκη Stable-baselines3 και στον αλγόριθμο που υλοποιήσαμε κλήθηκε ως εξής:

her_kwargs=dict(n_sampled_goal=4,goal_selection_strategy='future',online_sampling=True,max_episode_length=100)

model = SAC('MultiInputPolicy', env, replay_buffer_class=HerReplayBuffer, replay_buffer_kwargs=her_kwargs, verbose=1, buffer_size=int(1e6), learning_rate=1e-3, gamma=0.95, batch_size=1024, tau=0.05, policy_kwargs=dict(net_arch=[512, 512, 512]))

Όπου, για τις παραμέτρους που αφορούν το μοντέλο HER ισχύουν τα παρακάτω:

n_sampled_goal	Αριθμός εικονικών μεταβάσεων που δημιουργούνται ανά πραγματική μετάβαση, με δειγματοληψία νέων στόχων.
goal_selection_strategy	Στρατηγική για τη δειγματοληψία στόχων για επανάληψη.
online_sampling	Εάν είναι True, τα δείγματα HER προστίθενται κατά τη δειγματοληψία, διαφορετικά προστίθενται στο τέλος ενός επεισοδίου.
max_episode_length	Το μέγιστο πλήθος των βημάτων ανά επεισόδιο.

Στον παρακάτω πίνακα παρουσιάζονται οι παράμετροι που ορίστηκαν από εμάς και απέδιδαν καλύτερα στην εκπαίδευση του αλγορίθμου μετά από πλήθος πειραμάτων.

***Πίνακας 4**: Παράμετροι Soft Actor-Critic with HER agent*

ΠΑΡΑΜΕΤΡΟΙ	***SAC agent - HER***
policy	MultiInputPolicy
reward_type	Lp-norm (όπως περιγράφηκε στην αντίστοιχη ενότητα μοντελοποίησης ανταμοιβών)
learning rate	0.0001

gamma	0.95
optimizer	Adam
total episodes	200
steps/ episode	100
batch size	1024
tau	0.05
alpha	0.6
episodes before train (steps)	100 (10000)
episodes before update the target model	150

Το μοντέλο εκπαιδεύεται όπως περιγράφηκε παραπάνω. Ανά 4 επεισόδια τυπώνονται οι κυριότερες παράμετροι που επηρεάζουν την εκπαίδευση του μοντέλου και δίνουν ένα μέτρο της επιτυχίας του. Παραδειγματικά, παρουσιάζονται ο συνολικός αριθμός των επεισοδίων, τα μέχρι την εκάστοτε ενημέρωση στιγμιότυπα, ο χρόνος εκπαίδευσης, ο μέσος όρος ανταμοιβής των πιο πρόσφατων τεσσάρων επεισοδίων, οι απώλειες του κριτή και δράστη, όπως και το μέτρο επιτυχίας (success rate) που είναι ενδεικτικό της αποδοτικότητας του αλγορίθμου. Η μεταβλητή αυτή είναι χαρακτηριστική για τη μέθοδο HER καθώς στις τιμές αυτής βασίζεται ο πράκτορας, μιας και είναι ένα μέτρο του πόσου προσέγγισε τον στόχο και έτσι ο πράκτορας μαθαίνει ακόμη και από τις λανθασμένες αποφάσεις και μη επιτυχημένες τροχιές του. Αξίζει να σημειωθεί πως στο παρόν πείραμα που παρουσιάζεται για την επιτυχή εκμάθηση του μοντέλου και την προσέγγιση του στόχου εντέλει με 97% επιτυχία (success rate = 0.97), χρειάστηκαν 908 επεισόδια συνολικά. Παρακάτω παρατίθενται οι τρεις πρώτες και τρεις τελευταίες ενημερώσεις που λήφθηκαν κατά την διάρκεια της εκπαίδευσης, με τις τιμές των παραμέτρων που περιγράφηκαν παραπάνω.

```
| rollout/           |          |
|    ep_len_mean     | 100      |
|    ep_rew_mean     | -68.3    |
|    success rate    | 0        |
| time/              |          |
|    episodes        | 4        |
|    fps             | 5        |
|    time_elapsed    | 78       |
|    total timesteps | 400      |
| train/             |          |
|    actor_loss      | -8.43    |
|    critic_loss     | 0.0679   |
|    ent_coef        | 0.742    |
|    ent_coef_loss   | -1       |
|    learning_rate   | 0.001    |
|    n_updates       | 299      |
```

```
| rollout/           |          |
|    ep_len_mean     | 100      |
|    ep_rew_mean     | -55.9    |
|    success rate    | 0        |
| time/              |          |
|    episodes        | 8        |
|    fps             | 4        |
|    time_elapsed    | 183      |
|    total timesteps | 800      |
| train/             |          |
|    actor_loss      | -8.61    |
|    critic_loss     | 0.0465   |
|    ent_coef        | 0.499    |
|    ent_coef_loss   | -2.32    |
|    learning_rate   | 0.001    |
|    n_updates       | 699      |
```

```
| rollout/           |          |
|    ep_len_mean     | 100      |
|    ep_rew_mean     | -52.4    |
|    success rate    | 0        |
| time/              |          |
|    episodes        | 12       |
|    fps             | 4        |
|    time_elapsed    | 295      |
|    total timesteps | 1200     |
| train/             |          |
|    actor_loss      | -5.59    |
|    critic_loss     | 0.0215   |
|    ent_coef        | 0.336    |
|    ent_coef_loss   | -3.55    |
|    learning_rate   | 0.001    |
|    n_updates       | 1099     |
```

```
| rollout/           |          |
|    ep_len_mean     | 24.9     |
|    ep_rew_mean     | -7.81    |
|    success rate    | 0.97     |
| time/              |          |
|    episodes        | 900      |
|    fps             | 3        |
|    time_elapsed    | 13871    |
|    total timesteps | 49825    |
| train/             |          |
|    actor_loss      | 1.49     |
|    critic_loss     | 0.0586   |
|    ent_coef        | 0.00583  |
|    ent_coef_loss   | -0.14    |
|    learning_rate   | 0.001    |
|    n_updates       | 49724    |
```

```
| rollout/           |          |
|    ep_len_mean     | 25.1     |
|    ep_rew_mean     | -7.91    |
|    success rate    | 0.97     |
| time/              |          |
|    episodes        | 904      |
|    fps             | 3        |
|    time_elapsed    | 13894    |
|    total timesteps | 49909    |
| train/             |          |
|    actor_loss      | 1.47     |
|    critic_loss     | 0.006    |
|    ent_coef        | 0.00586  |
|    ent_coef_loss   | -0.0798  |
|    learning_rate   | 0.001    |
|    n_updates       | 49808    |
```

```
| rollout/           |          |
|    ep_len_mean     | 25       |
|    ep_rew_mean     | -7.92    |
|    success rate    | 0.97     |
| time/              |          |
|    episodes        | 908      |
|    fps             | 3        |
|    time_elapsed    | 13918    |
|    total timesteps | 49995    |
| train/             |          |
|    actor_loss      | 1.55     |
|    critic_loss     | 0.00748  |
|    ent_coef        | 0.00573  |
|    ent_coef_loss   | -0.186   |
|    learning_rate   | 0.001    |
|    n_updates       | 49894    |
```

Όπως φαίνεται από τα πειραματικά αποτελέσματα, με την πάροδο των επεισοδίων εκπαίδευσης, διαπιστώνεται ότι ο πράκτορας ανταποκρίνεται στην διαδικασία της εκμάθησης, εφόσον αυξάνεται η ανταμοιβή του (άρα οι ενέργειες που λαμβάνει οδηγούν πιο κοντά στον επιθυμητό στόχο) και το success rate προσεγγίζει με σχετικά σταθερό ρυθμό και χωρίς απότομες αυξομειώσεις την μονάδα, η οποία θεωρείται και η απόλυτη επιτυχία για το μοντέλο.

Evaluation

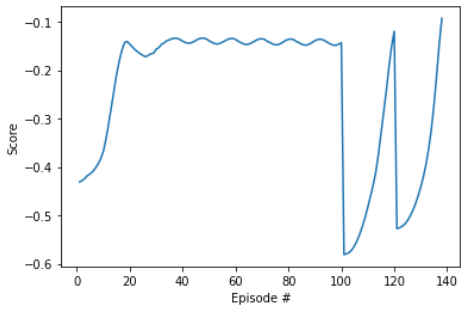
Στη συνέχεια εκτελέστηκε πλήθος 1000 στιγμιοτύπων με στόχο την αξιολόγηση του μοντέλου. Για κάθε επεισόδιο που τερματίζει ή για κάθε επεισόδιο που ο πράκτορας δεν καταφέρνει να

φτάσει στον στόχο του, εκτυπώνονται οι ανταμοιβές και η πληροφορία της επιτυχίας ή αποτυχίας. Όπως φαίνεται και από τα πειραματικά αποτελέσματα που παρατίθενται παρακάτω, από τα 31 επεισόδια που εκτελέστηκαν, μόλις στα 5 ο πράκτορας δεν έφτασε στον προσδοκώμενο στόχο, γεγονός που μπορεί να συμπεραθεί παρατηρώντας και μόνο τις ανταμοιβές που προκύπτουν στο τέλος κάθε επεισοδίου. Συγκεκριμένα, σε όσα επεισόδια το όχημα δεν κατόρθωσε να παρκάρει στην θέση-στόχο, η ανταμοιβές παρουσιάζουν αξιοσημείωτη απόκλιση από αυτές των επιτυχημένων επεισοδίων.

```
Reward: -3.408758628586496 Success? True
Reward: -17.94016041147533 Success? False
Reward: -5.270604628472361 Success? True
Reward: -15.448484367044447 Success? False
Reward: -6.7477829433283025 Success? True
Reward: -3.410180688067002 Success? True
Reward: -5.782514087257013 Success? True
Reward: -7.698361430303921 Success? True
Reward: -2.693937323833027 Success? True
Reward: -8.108420206128086 Success? True
Reward: -7.425471343276711 Success? True
Reward: -9.320737057504887 Success? True
Reward: -3.2026077841597727 Success? True
Reward: -3.7467316254599203 Success? True
Reward: -3.585310317921535 Success? True
Reward: -7.853212603869396 Success? True
Reward: -9.001823712972884 Success? True
```

```
Reward: -4.160802838760347 Success? True
Reward: -3.5076063306327603 Success? True
Reward: -7.213635860020949 Success? True
Reward: -17.282612334122025 Success? False
Reward: -8.681254819597383 Success? True
Reward: -6.99502582490864 Success? True
Reward: -6.330844907109852 Success? True
Reward: -4.889168565738305 Success? True
Reward: -9.723382075407068 Success? True
Reward: -5.577657041167964 Success? True
Reward: -16.370581628375266 Success? False
Reward: -7.335061378751388 Success? True
Reward: -2.6071203551504256 Success? True
Reward: -17.020346897334854 Success? False
```

Παρακάτω φαίνεται η γραφική παράσταση των συνολικών ανταμοιβών κάθε επεισοδίου της παραπάνω αξιολόγησης του αλγορίθμου και διαπιστώνεται ότι αυτές σταθεροποιούνται σε μια περιοχή πολύ κοντά στο μηδέν, που αποτελεί την πλήρη επιτυχία του μοντέλου.

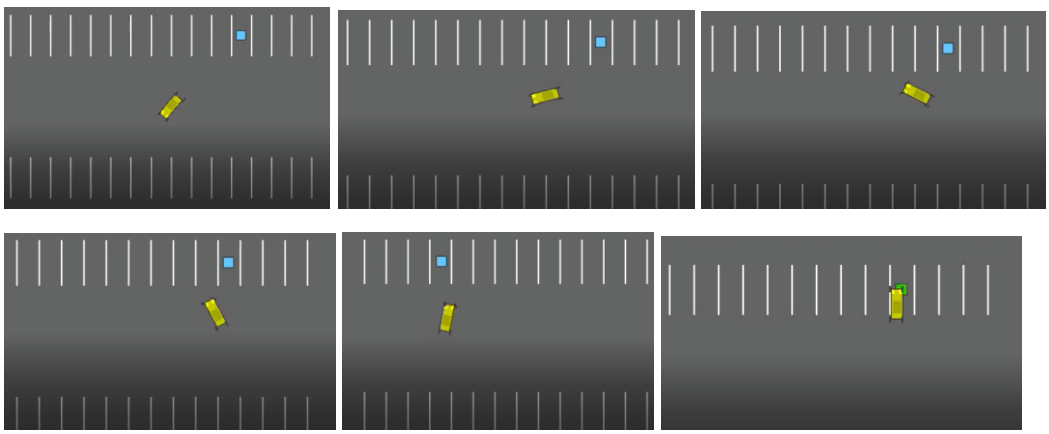


***Εικόνα 17**: SAC-Her evaluation rewards*

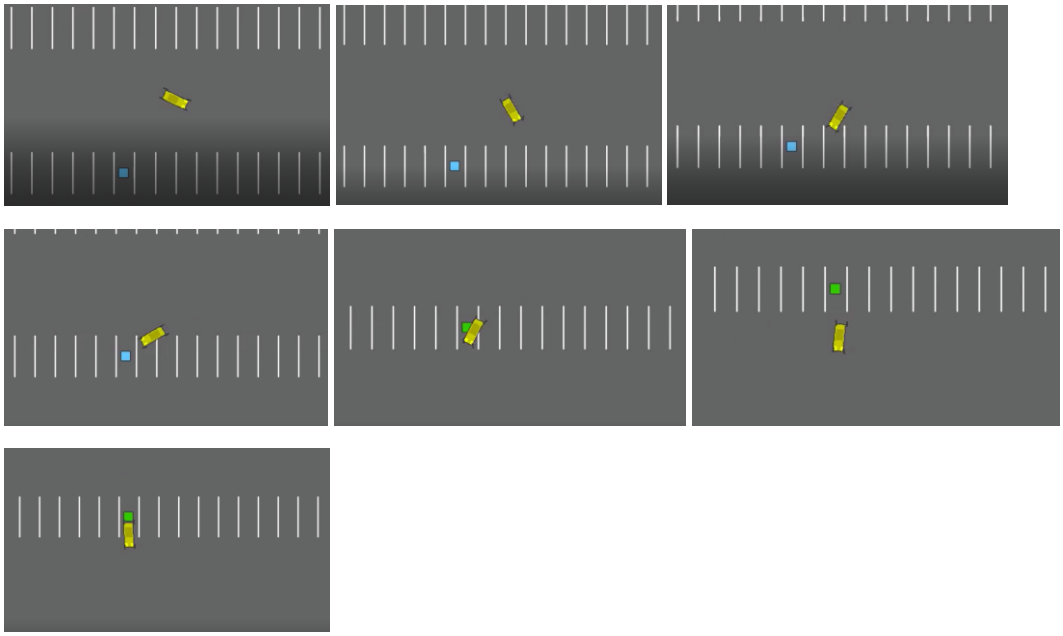
Optimization

Ακολουθούν στιγμιότυπα από κάποια επιτυχημένα επεισόδια του πράκτορα, όπου και παρατηρείται πως η αρχική και τελική θέση του οχήματος είναι τυχαία και διαφέρει από επεισόδιο σε επεισόδιο.

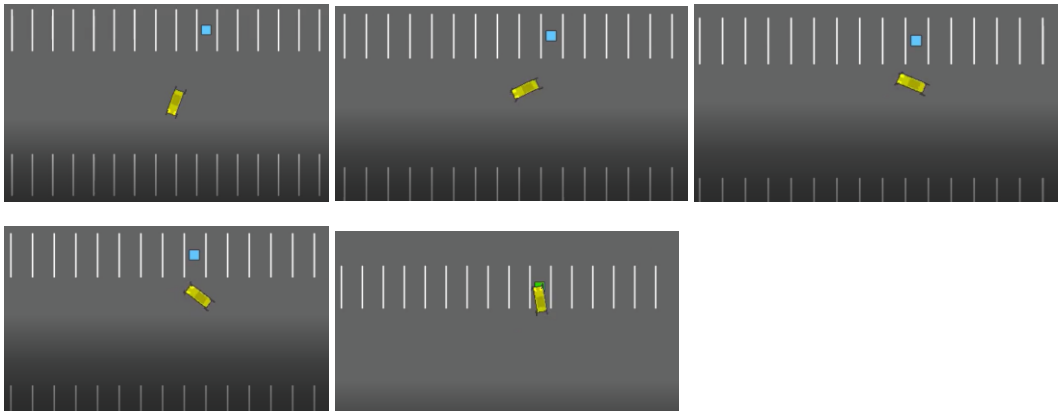
Παράδειγμα Α.



Παράδειγμα Β.



Παράδειγμα C.



5 Συμπεράσματα

Συνοψίζοντας, στην παρούσα διπλωματική παρουσιάστηκε το θεωρητικό υπόβαθρο της Ενισχυτικής μάθησης, ενώ έγινε και μια περιεκτική μελέτη και παρουσίαση των κυριότερων αλγορίθμων της Βαθιάς Ενισχυτικής μάθησης. Υπό το παραπάνω πλαίσιο, σχεδιάστηκαν τέσσερις πράκτορες (model-based CEM planner, DQN, SAC, SAC with HER), καθένας από τους οποίους χρησιμοποιεί διαφορετικό μοντέλο μάθησης ή συνδυασμό διαφορετικών μεθόδων, με κοινό παρονομαστή σε κάθε περίπτωση: την εκπαίδευση του εκάστοτε πράκτορα να πλοηγηθεί σε άδειο χάρτη (μονοπρακτορικό περιβάλλον, δίχως ύπαρξη εμποδίων) και να παρκάρει σε συγκεκριμένη θέση-στόχο. Στο κεφάλαιο λοιπόν αυτό συζητώνται και συγκρίνονται τα αποτελέσματα των τεσσάρων αλγορίθμων ως προς την εκπαίδευση και την αποδοτικότητά τους, ενώ γίνεται αναφορά στο μη επιτυχώς εκπαιδευμένο μοντέλο DQN και στα πιθανά αίτια της αποτυχίας του. Κλείνοντας, προτείνονται κατευθυντήριοι άξονες για την εξέλιξη των πειραμάτων που πραγματοποιήθηκαν ώστε να καταστούν αποδοτικότερα και πολυπλοκότερα, όπως και ιδέες για μελλοντική πληρέστερη έρευνα.

5.1 Απόδοση αλγορίθμων

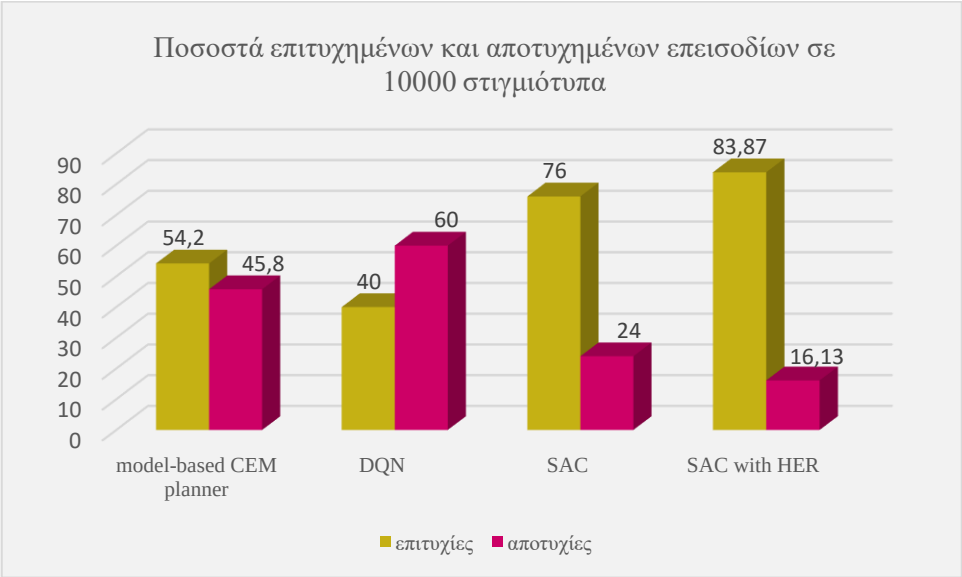
Στην παρούσα διπλωματική πραγματεύεται το ζήτημα της προσομοίωσης πλοήγησης ενός οχήματος σε συνεχή μονοπρακτορικό περιβάλλον χωρίς εμπόδια. Στόχος κάθε μοντέλου που εκπαιδεύτηκε αποτέλεσε η εκμάθηση του πράκτορα κατά τέτοιον τρόπο ώστε να είναι σε θέση να προσεγγίζει, ιδανικά με βέλτιστο τρόπο, κάποια τελική θέση-στόχο, οπουδήποτε κι αν εντοπιζόταν στον χώρο και ανεξάρτητα με το σημείο εκκίνησής του.

Προς αυτήν την πρόκληση, σχεδιάστηκαν οι εξής τέσσερις ευφυείς πράκτορες: model-based CEM planner, DQN, SAC, SAC with HER. Κατά την εκπαίδευσή τους, όλοι συνέκλιναν στην συνθήκη τερματισμού, παρόλο που η απόδοσή τους διέφερε με τον SAC-HER να ξεχωρίζει σημαντικά από τους υπόλοιπους, γεγονός που οφείλεται κυρίως στην χρήση της τεχνικής Hindsight Experience Replay. Στη συνέχεια επαληθεύτηκαν τα εκπαιδευμένα μοντέλα και διαπιστώθηκε η διαφορετική συμπεριφορά τους τόσο μεταξύ τους, όσο και σε κάποιες περιπτώσεις με την απόδοση που παρουσίασαν τα ίδια κατά την εκπαίδευσή τους. Η αξιολόγηση (evaluation) πραγματοποιήθηκε σε δείγμα 10000 στιγμιότυπων, ενώ τα αποτελέσματά της παρουσιάστηκαν στο προηγούμενο κεφάλαιο για καθέναν από αυτούς.

Ειδικότερα, στα αποτελέσματα του model-based πράκτορα το 54,2% ήταν επιτυχίες. Ωστόσο, όπως έχει ήδη αναφερθεί, ακόμη και για τα μη επιτυχημένα επεισόδια, ο πράκτορας κατόρθωνε να φτάνει αρκετά κοντά στον επιθυμητό στόχο, εκτελώντας τροχιές με πολύπλοκους ελιγμούς, αν και όχι βέλτιστες στην πλειονότητα των περιπτώσεων.Αναφορικά με τον DQN πράκτορα, κατά την εκπαίδευσή του κατόρθωνε να φτάνει τη θέση-στόχο με ποσοστό επιτυχιών κοντά στο 40%. Ωστόσο, όπως φάνηκε από τα αντίστοιχα αποτελέσματα, κατά την αξιολόγησή του κανένα από τα 100 επεισόδια που εκτελέστηκαν δεν ήταν επιτυχημένο, γεγονός που οφείλεται στο φαινόμενο των «κυκλικών τροχιών». Κατά τη διάρκεια αυτών ο πράκτορας επιλέγει δράσεις ώστε να επισκέπτεται θέσεις που είχε βρεθεί και στο παρελθόν και λόγω της ντετερμινιστικής άπληστης πολιτικής που ακολουθεί, δεν μπορεί να ξεφύγει από την επαναλαμβανόμενη τροχιά. Αξίζει να επισημανθεί ότι κατά και την εκπαίδευση και καθώς τα βάρη του δικτύου ανανεώνονται μετά από κάθε βήμα κίνησης, ο πράκτορας δεν πέφτει σε «κυκλικές τροχιές». Με άλλα λόγια, αφού αντιληφθεί ότι πρέπει να κινηθεί προς τον στόχο, τότε αν τύχει και πέσει σε μια ατέρμονη επανάληψη, τα βάρη θα αλλάξουν μετά από ορισμένο αριθμό βημάτων, έτσι ώστε να διαλέγει διαφορετική ενέργεια σε μια κατάσταση. Ο πράκτορας SAC έχει ποσοστό επιτυχίας περίπου 76%, σημαντικά μεγαλύτερο από αυτό των προηγούμενων μοντέλων, ενώ η πλειοψηφία αυτών συγκλίνουν με βέλτιστο τρόπο στον στόχο. Κλείνοντας, για τον πράκτορα SAC – HER, το ποσοστό επιτυχιών είναι εξαιρετικό, 83,87%,

ενώ η συντριπτική πλειοψηφία αυτών είναι βέλτιστες τροχιές και για τα μη επιτυχημένα επεισόδια το όχημα κατόρθωσε να φτάσει πολύ κοντά στον εκάστοτε αρχικό στόχο.

Παρακάτω παρουσιάζεται ένα συγκεντρωτικό γράφημα επιτυχημένων και αποτυχημένων επεισοδίων που προέκυψαν από προσομοίωση 10.000 στιγμιοτύπων, για καθέναν από τους τέσσερις εκπαιδευμένους πράκτορες



5.2 Κατεύθυνση μελλοντικής έρευνας

Όπως έχει αναφερθεί αρκετές φορές στα προηγούμενα κεφάλαια, το ζήτημα του αυτόνομου παρκαρίσματος μοντελοποιήθηκε ώστε να αφορά την πλοήγηση σε μονοπρακτορικό περιβάλλοντα χώρο χωρίς εμπόδια με τυχαία αρχική θέση και τυχαία τελική επιθυμητή θέση-στόχο. Ενώ, σε μεγαλύτερη κλίμακα πρόβλημα αφορά την κίνηση του πράκτορα σε έναν άγνωστο χάρτη πολυπρακτορικού περιβάλλοντος με στατικά ή/και δυναμικά εμπόδια. Επομένως, μελλοντική επέκταση της παρούσας διπλωματικής θα μπορούσε να αποτελεί η έρευνα και η δημιουργία μοντέλων με κατεύθυνση την γενικότερη περίπτωση.

Σε πρώτο επίπεδο, σημαντικό πεδίο έρευνας θα μπορούσε να αποτελέσει η αντιμετώπιση του ζητουμένου σε ένα περιβάλλον με στατικά και στη συνέχεια δυναμικά εμπόδια, των οποίων η θέση μεταβάλλεται από επεισόδιο σε επεισόδιο και δεν μπορεί να προβλεφθεί εκ των προτέρων. Για το παρόν πρόβλημα συνεχούς κίνησης, ίσως δεν θα αρκούσε η πρόσδοση αρνητικών ανταμοιβών στον πράκτορα σε περίπτωση σύγκρουσης. Το πρόβλημα αυτό θα πρέπει να εστιάζει στην εξέταση του περιβάλλοντος χώρου πριν ληφθεί οποιαδήποτε απόφαση κίνησης. Ειδικότερα, δίνεται έμφαση στα δεδομένα εισόδου, που θα βοηθήσουν στην αναγνώριση κινδύνου σύγκρουσης, και όχι στις παραμέτρους εκπαίδευσης. Συνεπώς, προτείνεται η χρήση αισθητήρων, οι οποίοι θα παρέχουν στον πράκτορα την εικόνα του περιβάλλοντος την δεδομένη χρονική στιγμή, τη σχετική θέση των εμποδίων και τον προσανατολισμό τους. Οι πληροφορίες αυτές θα επεξεργαστούν κατάλληλα και το εκάστοτε επιλεγμένο μοντέλο θα τις λάβει υπόψη έτσι ώστε το όχημα εντέλει να επιλέγει δράσεις που όχι μόνο στοχεύουν στην βέλτιστη τροχιά προς την θέση-στόχο, αλλά παράλληλα θα εξασφαλίζουν την αποφυγή των εμποδίων.

Σε δεύτερο επίπεδο, μία πιο απαιτητική προσέγγιση θα ήταν η εκπαίδευση του πράκτορα σε ένα πολυπρακτορικό περιβάλλον στάθμευσης, όπου το κάθε όχημα θα έχει διαφορετικό στόχο. Στον χώρο αυτό, οι πράκτορες θα κινούνται ταυτόχρονα και οφείλουν να λαμβάνουν υπόψη

τις ενέργειες όλων των υπολοίπων προκειμένου να αποφασίσουν την βέλτιστη για αυτούς ενέργεια, αποφεύγοντας ταυτόχρονα και τις πιθανές συγκρούσεις. Εν κατακλείδι, σε ένα ρεαλιστικότερο και αρκετά πολυπλοκότερο πρόβλημα, θα μπορούσε να μελετηθεί η συνύπαρξη πολλών πρακτόρων παράλληλα, σε έναν χώρο με στατικά και δυναμικά εμπόδια.

Παραπομπή

Στον παρακάτω σύνδεσμο παρουσιάζονται αναλυτικά οι αλγόριθμοι που σχεδιάστηκαν για την εκπαίδευση και αξιολόγηση των τεσσάρων πρακτόρων:

zoikorda/self-driving-cars (github.com)

Βιβλιογραφία

[1] Bishop C. M., *Pattern recognition and machine learning*, springer, 2006.

[2] T. Mitchell, *Machine Learning*, McGraw Hill, 1997.

[3] Hunter, Heidenreich, *«What are the types of machine learning?,»* towards data science, 2018.

[4] S. J. Russell, P. Norvig, *Artificial intelligence: a modern approach*., Malaysia: Pearson Education Limited, 2016.

[5] Y. LeCun, Y. Bengio, G. Hinton, *"Deep learning"*, nature, vol. 521, no. 7553, p. 436, 2015.

[6] Dechter, Rina, *Learning while searching in constraint-satisfaction-problems*, Artificial Intelligence Center, Hughes Aircraft Company, Calabasas and Cognitive Systems Laboratory, Computer Science Department , 1986.

[7] Facundo Bre, Juan M. Gimenez, Victor D. Fachinotti, *"Prediction of wind pressure coeeficints on building surfaces using Artificial Neural Networks"*, Energy and Buildings 158, 29 November 2017.

[8] Saurabh, Singh, *«Sigmoid as an Activation Function in Neural Networks»*, Deep Learning University, 2021.

[9] «Librow,» [Ηλεκτρονικό]. Available: http://www.librow.com/articles/article-11/appendix-a-31.

[10] Saurabh, Singh, *«ReLU as an Activation Function in Neural Networks»*, Deep Learning University, 2021.

[11] Mislan, Haviluddin, Sigin Hardwinarto, Sumaryono, Marlon Aipassa, *«Rainfall Monthly Prediction Based on Artificial Neural Network – A Case Study Tenggarong Station, East Kalimantan – Indonesia,»* Procedia Computer Science 59, 25 August 2015.

[12] S. Haykin, *Neural networks: a comprehensive foundation*, Prentice Hall PTR, 1994.

[13] Sebastian Ruder, *«An overview of gradient descent optimization algorithms»*, 2016.

[14] Vitaly Bushaev, *«Adam — latest trends in deep learning optimization.»*, towards data science, 2018.

[15] Jimmy Lei Ba, Diederik P. Kingma, *«ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION»*, conference paper at ICLR 2015.

[16] A. G. Barto, R. S. Shaton ,*«Reinforcement learning: An introduction.»*, MIT press, 2018.

[17] M. L. Littman, A. W. Moore, L. P. Kaelbling, *«Reinforcement learning: A survey»*, Journal of artificial intelligence research, vol. 4, 1996, p. 237–285.

[18] Philip Ossenkopp, *«Reinforcement learning – Part 2: Getting started with Deep Q-Networks»*, NOVATEC, 2018.

[19] Dan C., Marinescu, *«Mobile Edge Cloud and Markov Decision Processes»*, Cloud Computing (Second Edition), 2018.

[20] M. Gasic, *«Introduction to reinforcement learning»*, Dialogue Systems Group, Cambridge University Engineering Department.

[21] L. C. Baird, *«Reinforcement learning in continuous time: Advantage updating»*, 1994.

[22] L. Baird et al., *«Residual algorithms: Reinforcement learning with function approximation»*, Proceedings of the twelfth international conference on machine learning, 1995, p. 30–37.

[23] Dmitry B. Rokhlin, *«Robbins-Monro conditions for persistent exploration learning strategies»*, 2018.

[24] Wang Chi Cheung, *«Exploration-Exploitation Trade-off in Reinforcement Learning on Online Markov Decision Processes with Global Concave Rewards»*, 2019.

[25] *«Intro to Reinforcement Learning: The Explore-Exploit Dilemma»*, towards data science, 2020.

[26] Erik J. Peterson, Timothy D. Verstynen, *«Curiosity eliminates the exploration-exploitation dilemma»*, Cold Spring Harbor Laboratory, 2020.

[27] Deepak Pathak, Pulkit Argawal, Alexei A. Efros, Trevol Darrell, *«Curiosity-driven Exploration by Self-supervised Prediction»*, 2017.

[28] L. Weng, *«Exploration Strategies in Deep Reinforcement Learning»*, Lil'Log, 2020.

[29] P. A. Gagniuc, *«Markov chains: from theory to implementation and experimentation.»*, John Wiley & Sons, 2017.

[30] R. Bellman et al., *«The theory of dynamic programming»*, Bulletin of the American Mathematical Society, MA, Athena scientific Belmont, 1995, p. vol.1.

[31] M. Lee, *«Dynamic Programming»*, incompleteideas.net, 2005.

[32] T. Levine, I. Sutskever, S. Lillicrap, S. Gu, «*Continuous deep q-learning with model based acceleration*», International Conference on Machine Learning, pp. 2829–2838, 2016.

[33] M. Lee, *«Monte Carlo Methods»*, incompleteideas.net, 2005.

[34] A. G. Barto, R. S. Sutton, *«Temporal-Difference Learning»*, Reinforcement Learning: An Introduction, MIT Press, 1998.

[35] A. Violante, *«Simple Reinforcement Learning: Temporal Difference Learning»*, 2018.

[36] F. Kunz, *«An Introduction to Temporal Difference Learning»*, Seminar on Autonomous Learning Systems, Department of Computer Science, TU Darmstadt.

[37] J. Peters, J. Kober, *«Reinforcement learning in robotics: A survey»*, Reinforcement Learning: State of the Art, Springer, 2012, pp. pp. 579-610.

[38] M. Niranjan, G. A. Rummery, *«On-line Q-learning using connectionist systems.»*, University of Cambridge, Department of Engineering, 1994.

[39] C. J. Watkins, *«Learning from delayed rewards. PhD thesis»*, University of Cambridge England, 1989.

[40] C. J. Watkins, P. Dayan, *«Q-learning»*, Machine learning, vol. 8, no. 3-4, 1992, pp. 279-292.

[41] M. L. Littman, *«Algorithms for sequential decision making»*, 1996.

[42] K. M. Gupta, *«Performance Comparison of Sarsa(λ) and Watkin's Q(λ) Algorithms»*, Department of Computer Science , Texas Tech University.

[43] G. Tesauro, *«Temporal difference learning and TD-Gammon»*, Communications of the ACM, vol. 38, no. 3, 1995, pp. pp. 58-68.

[44] F. L. Lewis, M. Sistani, B. Naghibi, H. Modares, *«Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems»*, Automatica, vol. 50, no. 1, 2014, pp. pp. 193-202.

[45] L. J. Lin, *«Self-improving reactive agents based on reinforcement learning, planning and teaching»*, Machine learning, vol. 8(3-4), 1992, p. pp. 293–321.

[46] T. Quan, J. Antonoglu, S. Schaul, *«D. Prioritized experience replay»*, Internasional Conference on Neural Information Processing Systems (NeurIPS), 2019.

[47] T. de Bruin, J. Kober, K. Tuyls, R. Babuska, *«The importance of experience replay database composition in deep reinforcement learning»*, Deep Reinforcement Learning Workshop, NIPS, 2015.

[48] L. J. Lin, *«Reinforcement learning for robots using neural networks»*, Carnegie-Mellon Univ Pittsburgh PA School of Computer Science, Tech. Rep., 1993.

[49] J. Peters, S. Schaal, *«Natural actor-critic»*, Neurocomputing, vol. 71, no. 7, 2018, p. pp. 1180–1190.

[50] A. Gruslys, M. G. Azar, M. G. Bellemare, R. Munos, *«The reactor: A sample-efficient actor-critic architecture»*, arXiv:1704.04651, 2017.

[51] T. Haarnoja, A. Zhou, P. Abbeel, S. Levine, *«Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor»*, arXiv:1801.01290, 2018.

[52] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, W. Zaremba, *«Hindsight Experience Replay»*, arXiv:1707.01495v3, 2018.

[53] W. Davidson, R. Rubinstein, *«The cross-entropy method for combinatorial and continuous optimization»*, Methodology and Computing in Applied Probability, 1999.

[54] C. Pinneri, S. Sawant, S. Blaes, J. Achterhold, J. Stuckler, M. Rolinek, G. Martius, *«Sample-efficient Cross-Entropy Method for Real-time Planning»*, arXiv:2008.06389v1, 2020.

[55] Pieter-Tjerk de Boer, Dirk P. Kroese, S. Mannor, Reuven Y. Rubinstein, *«A Tutorial on the Cross-Entropy Method»*, Ann Oper Res 134, 19–67, 2005.

[56] H. G. Katzgraber, *«Introduction to Monte Carlo Methods»*, Department of Physics and Astronomy, Texas A&M University College Station and Theoretische Physik, ETH Zurich, 2011.

[57] Kenneth Leung, *«The Dying ReLU Problem»*, Clearly Explained, towards data science, 2021

[58] Yang Q., Chen H., Su J., Li J., *«Towards High Accuracy Parking Slot Detection for Automated Valet Parking System»*, Proceedings of the SAE 2019 New Energy and Intelligent Connected Vehicle Technology Conference; Shanghai, China. 21–22 May 2019; pp. 1–10.

[59] Song J., Zhang W.W., Wu X.C., Cao H.T., Gao Q.M., Luo S.Y., *«Laser-based SLAM automatic parallel parking path planning and tracking for passenger vehicle.»*, IET Intell. Transp. Syst. 2019;13:1557–1568. doi: 10.1049/iet-its.2019.0049.

[60] Jang C., Kim C., Lee S., Kim S., Lee S., Sunwoo M., *«Re-Plannable Automated Parking System With a Standalone Around View Monitor for Narrow Parking Lots.»,* IEEE Trans. Intell. Transp. Syst. 2019;21:1–14. doi: 10.1109/TITS.2019.2891665.

[61] Yan W., Tao Y., Junqiao Z., Linting G., Wei J., *«VH-HFCN based Parking Slot and Lane Markings Segmentation on Panoramic Surround View»,* Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV); Piscataway, NJ, USA. 26–30 June 2018; pp. 1767–1772.

[62] Chen X., Xu X., Yang Y., Wu H., Tang J., Zhao J., *«Augmented Ship Tracking Under Occlusion Conditions From Maritime Surveillance Videos.»,* IEEE Access. 2020; 8:42884–42897. doi: 10.1109/ACCESS.2020.2978054.

[63] Qin T., Chen T., Chen Y., Su Q., *«AVP-SLAM: Semantic Visual Mapping and Localization for Autonomous Vehicles in the Parking Lot.»,* Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2020; Las Vegas, NV, USA. 25–29 October 2020.

[64] Lee S, Lim W, Sunwoo M, *«Robust Parking Path Planning with Error-Adaptive Sampling under Perception Uncertainty.»,* Sensors (Basel). 2020 Jun 23.

[65] Li B., Wang K., Shao Z., *«Time-Optimal Maneuver Planning in Automatic Parallel Parking Using a Simultaneous Dynamic Optimization Approach.»,* IEEE Trans. Intell. Transp. Syst. 2016;17:3263–3274. doi: 10.1109/TITS.2016.2546386.

[66] Chao C., Rickert M., Knoll A., *«Path planning with orientation-aware space exploration guided heuristic search for autonomous parking and maneuvering.»* Proceedings of the 2015 IEEE Intelligent Vehicles Symposium (IV); Piscataway, NJ, USA. 28 June–1 July 2015; pp. 1148–1153.

[67] Banzhaf H., Sanzenbacher P., Baumann U., Zoellner J.M., *«Learning to Predict Ego-Vehicle Poses for Sampling-Based Nonholonomic Motion Planning.»,* IEEE Robot. Autom. Lett. 2019;4:1053–1060. doi: 10.1109/LRA.2019.2893975.

[68] Vorobieva H., Glaser S., Minoiu-Enache N., Mammar S., *«Automatic Parallel Parking in Tiny Spots: Path Planning and Control.»,* IEEE Trans. Intell. Transp. Syst. 2015;16:396–410. doi: 10.1109/TITS.2014.2335054.

[69] Fan Z., Chen H., *«Study on Path Following Control Method for Automatic Parking System Based on LQR.»,* SAE Int. J. Passeng. Cars Electron. Electr. Syst. 2016;10:41–49. doi: 10.4271/2016-01-1881.

[70] Du X.X., Tan K.K., *«Autonomous Reverse Parking System Based on Robust Path Generation and Improved Sliding Mode Control.»,* IEEE Trans. Intell. Transp. Syst. 2015;16:1225–1237. doi: 10.1109/TITS.2014.2354423.

[71] Ballinas E., Montiel O., Castillo O., Rubio Y., Aguilar L.T., *«Automatic parallel parking algorithm for a car-like robot using fuzzy pd+i control.»,* Eng. Lett. 2018;26:447–454.

[72] Bernhard J., Gieselmann R., Esterle K., Knoll A., *«Experience-Based Heuristic Search: Robust Motion Planning with Deep Q-Learning.»,* Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems; Maui, HI, USA. 4–7 November 2018; pp. 3175–3182.

[73] Du Z., Miao Q., Zong C., *«Trajectory Planning for Automated Parking Systems Using Deep Reinforcement Learning.»,* Int. J. Automot. Technol. 2020;21:881–887. doi: 10.1007/s12239-020-0085-9.

[74] Bejar E., Moran A., *«Reverse Parking a Car-Like Mobile Robot with Deep Reinforcement Learning and Preview Control.»,* Proceedings of the 2019 IEEE 9th Annual Computing and

Communication Workshop and Conference; Las Vegas, NV, USA. 7–9 January 2019; pp. 377–383.

[75] Zhang P, Xiong L, Yu Z, Fang P, Yan S, Yao J, Zhou Y, *«Reinforcement Learning-Based End-to-End Parking for Automatic Parking System.»*, Sensors (Basel). 2019 Sep 16

[76] Zhang J.R., Chen H., Song S.Y., Hu F.W., *«Reinforcement Learning-Based Motion Planning for Automatic Parking System.»*, IEEE Access. 2020;8:154485–154501. doi: 10.1109/ACCESS.2020.3017770.

[77] Hu F., Chen H., Zhang J., *«Study on Robust Motion Planning Method for Automatic Parking Assist System Based on Neural Network and Tree Search.»,* Proceedings of the SAE 2019 New Energy and Intelligent Connected Vehicle Technology Conference; Shanghai, China. 21–22 May 2019.

[78] Jagodnik K., Thomas, P., Branicky, M., Kirsch, R., *«Training an Actor-Critic reinforcement learning controller for arm movement using human-generated rewards.»,* IEEE Trans. Neural Syst. Rehabil. Eng. 2017, 25, 1892–1905.

[79] Fan, Q.; Yang, G.; Ye, D., *«Quantization-Based Adaptive Actor-Critic Tracking Control With Tracking Error Constraints.»,* IEEE Trans. Neural Netw. Learn. Syst. 2018, 29, 970–980.

[80] Xu J., Hou Z., Wang W., Xu B., Zhang K., Chen K., *«Feedback deep deterministic policy gradient with fuzzy reward for robotic multiple peg-in-hole assembly tasks.»*, IEEE Trans. Ind. Inform. 2019, 15, 1658–1667.