ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

Πολυτεχνική Σχολή

Τμήμα Μηχανολόγων Μηχανικών

Τομέας Οργάνωσης Παραγωγής & Βιομηχανικής Διοίκησης

ΕΛΕΓΧΟΣ ΑΥΤΟΝΟΜΩΝ ΟΧΗΜΑΤΩΝ

Διπλωματική Εργασία

Μπέης Κωνσταντίνος

Επιβλέπων Καθηγητής: Αμπουντώλας Κωνσταντίνος – Αναπληρωτής Καθηγητής Τ.Μ.Μ.

Υπεβλήθη για την εκπλήρωση μέρους των απαιτήσεων για
την απόκτηση του Διπλώματος Μηχανολόγου Μηχανικού

Βόλος, 2022

Εγκρίθηκε από τα Μέλη της Τριμελούς Εξεταστικής Επιτροπής:

Πρώτος Εξεταστής (Επιβλέπων)	**Δρ. Κωνσταντίνος Αμπουντώλας** Αναπληρωτής Καθηγητής, Τμήμα Μηχανολόγων Μηχανικών, Πανεπιστήμιο Θεσσαλίας
Δεύτερος Εξεταστής	**Δρ. Δημήτριος Παντελής** Καθηγητής, Τμήμα Μηχανολόγων Μηχανικών, Πανεπιστήμιο Θεσσαλίας
Τρίτος Εξεταστής	**Δρ. Γεώργιος Σαχαρίδης** Αναπληρωτής Καθηγητής, Τμήμα Μηχανολόγων Μηχανικών, Πανεπιστήμιο Θεσσαλίας

Ευχαριστίες

Αρχικά θα ήθελα να εκφράσω τις ειλικρινείς ευχαριστίες μου στον επιβλέποντα της διπλωματικής εργασίας μου, Αναπληρωτή Καθηγητή κ. Κωνσταντίνο Αμπουντώλα, για την καθοδήγησή του και τις γνώσεις που μου έδωσε απλόχερα καθ' όλη τη διάρκεια της εκπόνησης της εργασίας μου.

Ένα μεγάλο ευχαριστώ στα υπόλοιπα μέλη της τριμελούς εξεταστικής επιτροπής, Καθηγητή Δημήτριο Παντελή και Αναπληρωτή Καθηγητή Γεώργιο Σαχαρίδη για την αποδοχή τους να είναι μέλη, την προσεκτική ανάγνωση της εργασίας μου και για τις χρήσιμες υποδείξεις τους.

Ακόμη, θα ήθελα να ευχαριστήσω τον φίλο μου Γεώργιο Τζιόκα για την πολύτιμη βοήθειά που μου παρείχε στον τομέα του προγραμματισμού σε γλώσσα Python και τη σύνδεσή της με τα κομμάτια της προσομοίωσης.

Τέλος, δεν θα μπορούσα να παραλείψω την ευγνωμοσύνη μου για τους φίλους και συναδέλφους μου που με συμβούλευαν όλη αυτή την περίοδο με εύστοχες υποδείξεις και παρατηρήσεις και μου παρείχαν βοήθεια όποτε τη χρειάστηκα. Την παρούσα εργασία την αφιερώνω στην οικογένειά μου που με στήριξε όλο αυτό τον καιρό στο εγχείρημα της ολοκλήρωσης των σπουδών μου και της διπλωματικής εργασίας μου.

Μπέης Κωνσταντίνος – Σεπτέμβριος 2022, Βόλος

ΕΛΕΓΧΟΣ ΑΥΤΟΝΟΜΩΝ ΟΧΗΜΑΤΩΝ

ΜΠΕΗΣ ΚΩΝΣΤΑΝΤΙΝΟΣ

Τμήμα Μηχανολόγων Μηχανικών, Πανεπιστήμιο Θεσσαλίας, Σεπτέμβριος 2022

Επιβλέπων Καθηγητής: Δρ. Κωνσταντίνος Αμπουντώλας,

Αναπληρωτής Καθηγητής

Περίληψη

Μόνο στις ΗΠΑ έχει υπολογιστεί ότι οι πολίτες ξοδεύουν 6,9 δισεκατομμύρια ώρες καθισμένοι στην κίνηση κάθε χρόνο. Η ιδέα της αυτόνομης οδήγησης έχει προσελκύσει την προσοχή πολλών ερευνητών τα τελευταία χρόνια καθώς πιστεύεται ότι θα παρέχει αρκετά πλεονεκτήματα όπως αυξημένη οδική ασφάλεια, μείωση ατυχημάτων, μείωση κυκλοφοριακής συμφόρησης κ.α. Η υλοποίηση αυτής της ιδέας κερδίζει όλο και περισσότερο έδαφος καθώς μεγάλες βιομηχανίες έχουν βαλθεί να αναπτύξουν συστήματα αυτοματισμού και να δώσουν λύσεις στα τεχνολογικά προβλήματα που προκύπτουν με τους ερευνητές να προβλέπουν ότι μέχρι το 2025 θα υπάρχουν 8 εκατομμύρια αυτόνομα οχήματα στους δρόμους. Ένα αυτόνομο όχημα είναι ικανό να ανιχνεύει το περιβάλλον του και να κινείται με ασφάλεια με ελάχιστη ή και καθόλου ανθρώπινη παρέμβαση.

Κύριος σκοπός αυτής της εργασίας είναι να διερευνήσει και να αναπτύξει μια στρατηγική ελέγχου για την πλοήγηση (αλλαγή λωρίδας) ενός ή πολλαπλών οχημάτων όσο το δυνατόν αποτελεσματικότερα μέσω πυκνής κυκλοφορίας σε αυτοκινητόδρομο χωρίς σύγκρουση με άλλα οχήματα.

Η στρατηγική ελέγχου θα αναπτυχθεί με τη βοήθεια μιας από τις πιο αποδοτικές τεχνικές προγραμματισμού για στοχαστικά προβλήματα, τον δυναμικό προγραμματισμό (Dynamic Programming). Με τη χρήση του αναδρομικού τύπου του Bellman θα γίνει επίλυση του Προβλήματος Στοχαστικού Βέλτιστου Ελέγχου Διακριτού Χρόνου για ένα όχημα που κινείται σε έναν αυτοκινητόδρομο. Η αποτελεσματικότητα της στρατηγικής ελέγχου που θα προκύψει θα αξιολογηθεί σε ένα περιβάλλον προσομοίωσης αστικής κινητικότητας της πλατφόρμας Eclipse Sumo. Θα προσομοιωθούν μία πληθώρα σεναρίων και θα ελεγχθεί η επιρροή διαφόρων παραμέτρων στην αποτελεσματικότητα των κανόνων ελέγχου.

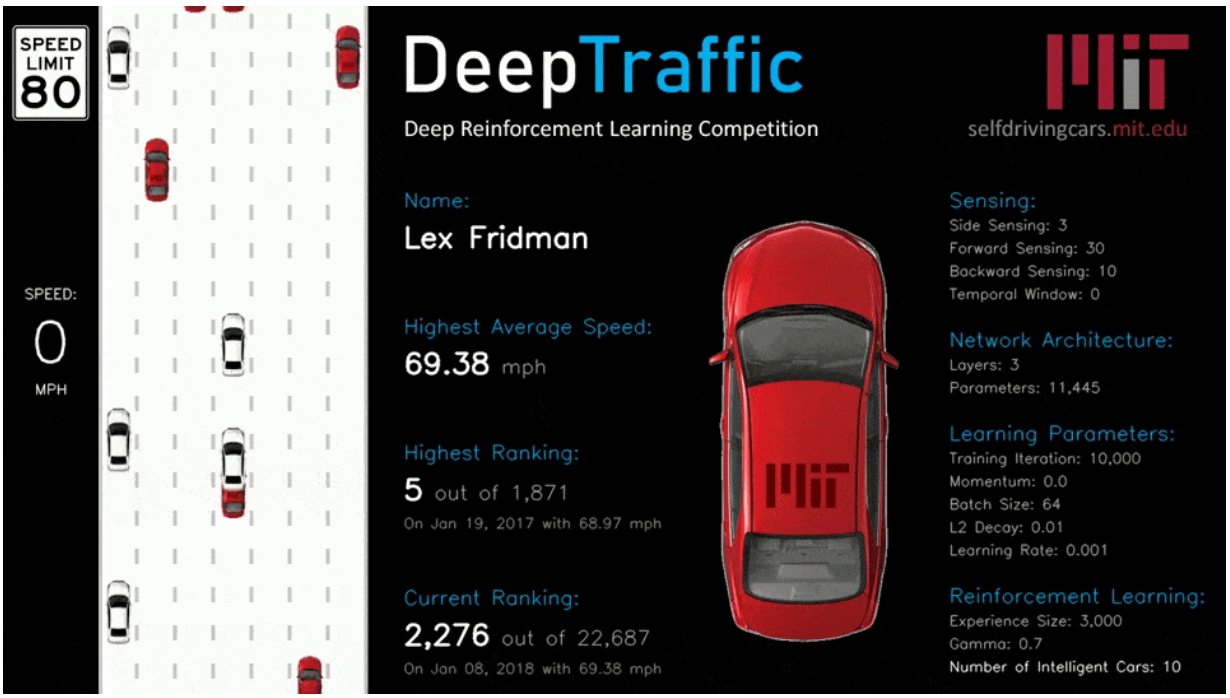
Περιεχόμενα

Κατάλογος Πινάκων

Κατάλογος Σχημάτων

1 ΕΙΣΑΓΩΓΗ

Στην παρούσα εργασία διερευνούμε την κίνηση ενός ή πολλαπλών οχημάτων σε ένα αυτοκινητόδρομο πέντε λωρίδων και αναπτύσσουμε μία στρατηγική οδικής συμπεριφοράς που αφορά στην εναλλαγή λωρίδων. Η ιδέα ξεκίνησε από ένα διαγωνισμό ενισχυτικής μάθησης του MIT που σύμφωνα με τον αλγόριθμο σχεδιασμού κίνησης που είχες αναπτύξει οπτικοποιούσε την πλοήγηση ενός οχήματος σε πυκνή κυκλοφορία. Ο στόχος ήταν να δημιουργηθεί ένα νευρωνικό δίκτυο που να οδηγεί ένα όχημα (ή πολλά οχήματα) όσο το δυνατόν γρηγορότερα μέσω πυκνής κυκλοφορίας σε αυτοκινητόδρομο. Η προσπάθεια ανάπτυξης στρατηγικής που εφαρμόσαμε στην εργασία βασίστηκε στον Διακριτό Δυναμικό Προγραμματισμό και αφορά ένα όχημα που αλλάζει λωρίδα για να επιτύχει τη μέγιστη ταχύτητα σε κορεσμένο αυτοκινητόδρομο.



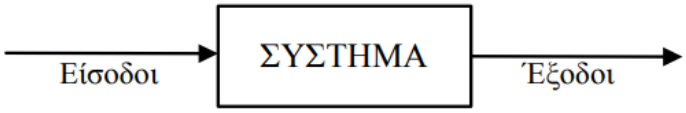
Σχήμα 1: Προσομοιωτής DeepTraffic που οπτικοποιεί τον υποβληθέντα αλγοριθμο αλλαγής λωρίδας.

Σε αυτό το κεφάλαιο ακολουθεί μία εισαγωγή στις έννοιες των δυναμικών συστημάτων και του δυναμικού προγραμματισμού. Παρουσιάζεται ο αναδρομικός τύπος του Bellman, η Αρχή του Βέλτιστου και γίνεται μία περιγραφή και προσπάθεια λύσης του Προβλήματος Βέλτιστου Ελέγχου Διακριτού Χρόνου.

1.1 Δυναμικά συστήματα και Αρχή του Βέλτιστου

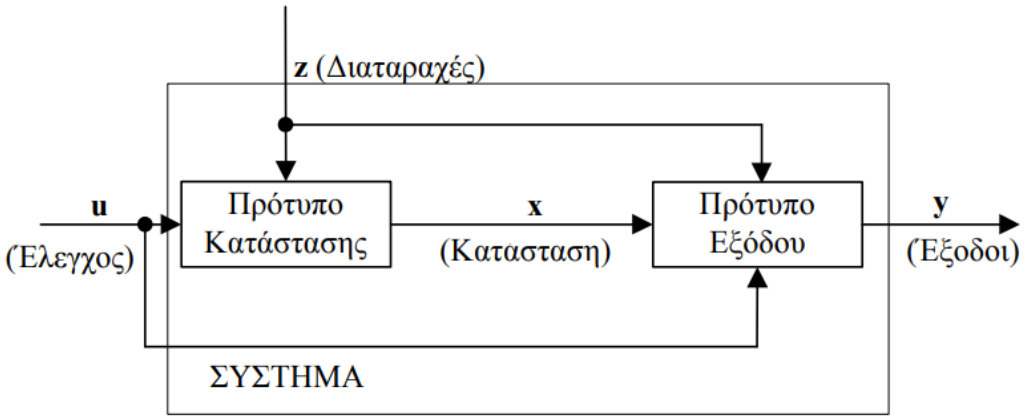
Ονομάζουμε σύστημα ένα σύνολο μαθηματικών σχέσεων και μεταβλητών κατάστασης που έχει συγκεκριμένη εσωτερική λειτουργία και σαφείς αλληλεπιδράσεις με το εξωτερικό περιβάλλον.

Το σύστημα δέχεται την επίδραση του περιβάλλοντος μέσω των εισόδων ενώ ανταποκρίνεται σ' αυτές δίνοντας αντίστοιχα κάποιες εξόδους.



Σχήμα 2: Γενική μορφή Συστήματος.

Οι είσοδοι μπορεί να διαχωριστούν σε δύο κατηγορίες. Στις ελεγχόμενες εισόδους ή εισόδους ελέγχου **u** και στις μη ελεγχόμενες ή αλλιώς διαταραχές **z**. Επίσης διακρίνουμε τις εξόδους **y** από τις εσωτερικές μεταβλητές κατάστασης του συστήματος **x.**



Σχήμα 3: Γενική Δομή Συστήματος.

Στην παρούσα εργασία θα ασχοληθούμε με δυναμικά συστήματα διακριτού χρόνου τα οποία περιγράφονται με μαθηματικές εξισώσεις.

Ένα δυναμικό σύστημα αποτελείται από έναν χώρο κατάστασης, του οποίου οι συντεταγμένες περιγράφουν την κατάσταση ανά πάσα στιγμή, και έναν δυναμικό κανόνα που καθορίζει τη συμπεριφορά όλων των μεταβλητών κατάστασης, λαμβάνοντας υπόψη μόνο τις παρούσες τιμές αυτών των ίδιων μεταβλητών κατάστασης.

Πρότυπο κατάστασης: $x(k+1) = f\left[x(k),\ u(k),\ z(k),\ k\right],\ x(k0) = x0$

Πρότυπο εξόδου: $y(k) = c\left[x(k),\ u(k),\ z(k),\ k\right]$

Όπου $k=k_0$, k_0+1,…. ο δείκτης διακριτού χρόνου που για όλες τις τιμές του ισχύει .(k)=.(kT), όπου T το διάστημα διακριτού χρόνου ή αλλιώς περίοδος.

Ο Δυναμικός Προγραμματισμός αποτελεί μία υπολογιστική μέθοδο για επίλυση προβλημάτων συνδυαστικής βελτιστοποίησης. Βασίζεται στην Αρχή του Βέλτιστου και ουσιαστικά έγκειται στην απλοποίηση ενός σύνθετου προβλήματος σε απλούστερα υποπροβλήματα και στην εύρεση αναδρομικά των βέλτιστων λύσεων. Το Πρόβλημα Βέλτιστου Ελέγχου αφορά δυναμικά συστήματα συνεχούς ή διακριτού χρόνου.

Για λόγους κατανόησης ας πάρουμε ως παράδειγμα ένα όχημα που ξεκινάει από ένα σημείο x_0 ενός δικτύου, τη χρονική στιγμή $t_0=0$ με στόχο να φτάσει σε ένα επιθυμητό σημείο x(K) ή τροχιά g[x(K),K] τη χρονική στιγμή K. Κάθε χρονική στιγμή t, ή βαθμίδα k, εμπεριέχει ένα σύνολο δυνατών εφικτών καταστάσεων x(k) από τις οποίες μπορεί να διέλθει το όχημα, το οποίο καθορίζεται από περιορισμούς. Ο αριθμός των εναλλακτικών αποφάσεων u(k) μέσω τον οποίων το όχημα έχει τη δυνατότητα να μεταβεί από μία κατάσταση μιας βαθμίδας στην επόμενη είναι πεπερασμένος. Η κάθε μετάβαση μαζί με την απόφαση από την οποία προέκυψε, χαρακτηρίζονται από ένα αντίστοιχο κόστος.

Στόχος είναι η εύρεση των τροχιών των μεταβλητών ελέγχου και κατάστασης u*(k) και x*(k), $k \in [0, K]$ που ελαχιστοποιούν το συνολικό κόστος για τη μετάβαση από την αρχική κατάσταση x_0 στην τελική βαθμίδα K.

Η **Αρχή του Βέλτιστου** βρίσκει άμεση εφαρμογή στην επίλυση προβλημάτων βελτιστοποίησης αποφάσεων πολλαπλών βαθμίδων διακριτού χρόνου.

Η **Αρχή του Βέλτιστου** λέει:

Κάθε υπόλοιπο u*(k), $k \in [k_1, K]$, $0 \le k_1 \le K$, της βέλτιστης τροχιάς ελέγχου u*(k), $k \in [0, K]$, αποτελεί μια βέλτιστη τροχιά για την μετάβαση από την αντίστοιχη ενδιάμεση κατάσταση $x^*(k_1)$ στο τελικό σημείο x(K) ή στην τελική τροχιά g[x(K),K]=0.

Η αρχή αυτή θα εφαρμοστεί σε παρακάτω πρόβλημα για την καλύτερη κατανόησή της.

1.2 Γενική Διατύπωση του Προβλήματος Βέλτιστου Ελέγχου Διακριτού Χρόνου

Το Πρόβλημα Βέλτιστου Ελέγχου (ΠΒΕ) απευθύνεται σε δυναμικά συστήματα συνεχούς ή διακριτού χρόνου.

Δεδομένης μιας αρχικής κατάστασης $x(0)=x_0$ και μιας επιθυμητής τελικής κατάστασης $x(K)=x_{desired}$ ζητείται να βρεθούν οι συναρτήσεις των μεταβλητών κατάστασης x(k) και ελέγχου u(k) που ελαχιστοποιούν το κριτήριο κόστους J.

Το συνολικό κόστος διαδρομής σε ένα σημείο x(K) του τελικού στόχου από μία αρχική κατάσταση x_0 δίνεται παρακάτω:

$$J = \vartheta[x(K)] + \sum_{k=0}^{K-1} \varphi[x(k), u(k),\ k] \qquad (1.1)$$

όπου ϕ[x(k), u(k), k] το κόστος μετάβασης από ένα σημείο x(k) στην επόμενη βαθμίδα k+1 , μέσω ενός ελέγχου u(k) και ϑ [x(k)] η συνάρτηση κόστους τελικών σημείων.

Θεωρούμε την ελαχιστοποίηση του κριτηρίου κόστους διακριτού χρόνου σε ένα καθορισμένο χρονικό ορίζοντα K υπό τους παρακάτω καταστατικούς περιορισμούς

$$x(k+1) = f[x(k), u(k), k] \qquad (1.2)$$

με την x(0) = x0 ως αρχική κατάσταση και τον τελικό στόχο να προσδιορίζεται από την

$$g[x(k)] = 0 \tag{1.3}$$

ή όποια άλλη διαδικασία ορίζει ένα σύνολο σημείων x(K) στο οποίο και μόνο δύναται να καταλήξει η κατάσταση του προβλήματος.

Η επιτρεπτή περιοχή τιμών του ελέγχου καθορίζεται με τον εξής τρόπο:

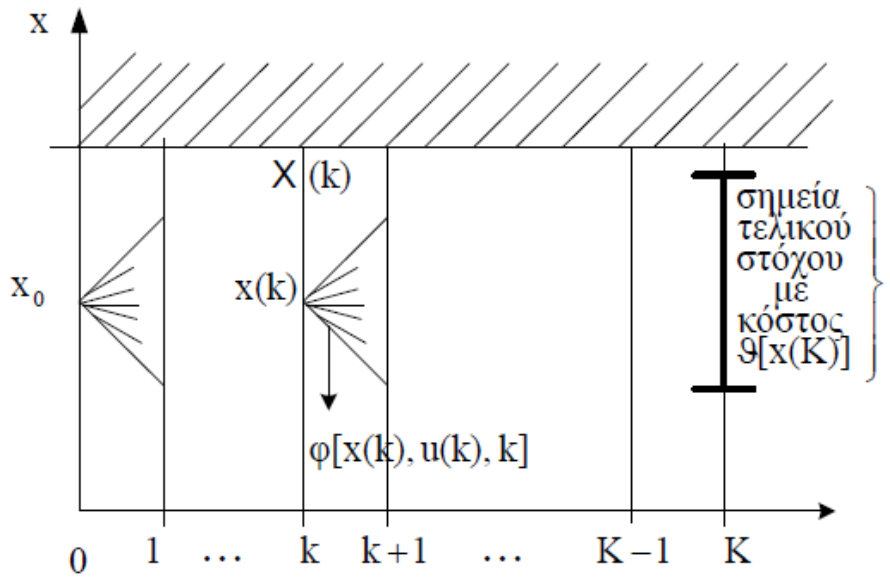
$$u(k) \in U[x(k),k] = \left\{ u(k) \middle| h[x(k),u(k),k] \le 0 \right\} \tag{1.4}$$

όπου έχουμε υποθέσει ότι ο Ιακωβιανός πίνακας ∂h/∂u είναι πλήρους βαθμού. Στην περίπτωση που έχουμε περιορισμούς ανισότητας κατάστασης δεν εκπληρούται αυτή η παραδοχή και θεωρούμε ακόμη τους παρακάτω περιορισμούς ανισότητας

$$x(k) \in X(k) = \left\{ x(k) \middle| h^x[x(k),k] \le 0 \right\} \tag{1.5}$$

με X(k) την επιτρεπτή περιοχή κατάστασης.

Για διακριτούς χρόνους k=0,1,...,K μπορούμε να θεωρήσουμε το πρόβλημα αυτό σαν πρόβλημα αποφάσεων πολλαπλών βαθμίδων και να επιχειρήσουμε τη λύση του μέσω της Αρχής του Βέλτιστου.



Σχήμα 4: Παραστατική επεξήγηση του ΠΒΕ διακριτού χρόνου.

1.3 Ο Αναδρομικός Τύπος του Bellman

Το παραπάνω πρόβλημα μπορεί να θεωρηθεί ως πρόβλημα αποφάσεων πολλαπλών βαθμίδων με την προϋπόθεση ότι ο χρόνος είναι διακριτός k=0,1,..., και θα επιχειρήσουμε να το λύσουμε με την Αρχή του Βέλτιστου. Στο πλαίσιο αυτό ορίζουμε αρχικά το υπολειπόμενο κόστος ή κόστος μεταφοράς J_k για τη μετάβαση μιας κατάστασης x(k) στον τελικό στόχο

$$J_k = \vartheta[x(K)] + \sum_{\kappa=k}^{K-1} \varphi[x(\kappa), u(\kappa),\ \kappa] \tag{1.6}$$

Ορίζουμε ως V[x(k), k] το ελάχιστο απ' τα κόστη μεταφοράς V[x(k),k]=min J_k μεταξύ όλων των τιμών u(κ), κ = k,...,K1, για τις οποίες ικανοποιούνται οι εξισώσεις (1.2)-(1.5). Το ελάχιστο κόστος εξαρτάται από το χρόνο και την κατάσταση μεταφοράς

$$V[x(k),k] = \min J_k = \min\{\varphi[x(k),u(k),k] + J_{k+1}\} \tag{1.7}$$

Εφαρμόζοντας την Αρχή του Βέλτιστου προηγούμενη σχέση προκύπτει

$$V[x(k),k] = \min\{\varphi[x(k),u(k),k] + V[x(k+1),k+1]\} \tag{1.8}$$

Από την (1.2) μπορούμε να αντικαταστήσουμε το x(k+1) στην (1.8) και έχουμε

$$V[x(k),k] = \min\{\varphi[x(k),u(k),k] + V[f(x(k),u(k),k),k+1]\} \tag{1.9}$$

Η εξίσωση αυτή ονομάζεται **αναδρομικός τύπος του Bellman**. Κοιτάζοντας το δεύτερο σκέλος της ισότητας γίνεται αντιληπτό ότι η ελαχιστοποίηση νοείται μόνο για την μεταβλητή ελέγχου της χρονικής βαθμίδας k. Δεν εξαρτάται από τον έλεγχο ούτε προηγούμενων ούτε επόμενων χρονικών βαθμίδων. Έτσι μπορεί να ξεκινήσει η μονοβαθμιαία ελαχιστοποίησή από την τελική χρονική βαθμίδα K και να συνεχίσει για χρόνους k=K-1,K-2,…,0 μετατρέποντας το πρόβλημα αποφάσεων πολλών βαθμίδων σε K προβλήματα μίας βαθμίδας. Βλέπουμε ότι

ο τύπος του Bellman αποτελεί μια αναδρομική διαδικασία η οποία υπολογίζει βέλτιστες τιμές συναρτήσεων κέρδους ή κόστους μέσω μιας κατάλληλης συναρτησιακής εξίσωσης.

1.4 Διατύπωση του Προβλήματος και Λύση

Θεωρούμε το Πρόβλημα Βέλτιστου Ελέγχου Διακριτού Χρόνου που περιλαμβάνει ένα όχημα που κινείται σε έναν αυτοκινητόδρομο πέντε (5) λωρίδων με στόχο να φτάσει σε ένα προκαθορισμένο επιθυμητό τελικό σημείο. Υποθέτουμε ότι φτάνει τον επιθυμητό στόχο την χρονική στιγμή K. Η θέση του κάθε χρονική στιγμή k μπορεί να πάρει τιμές x(k)=0,1,2,3,4. Κάθε στιγμή έχει τη δυνατότητα να αλλάξει λωρίδα είτε δεξιά είτε αριστερά, ή να διατηρήσει την πορεία του στη λωρίδα που βρίσκεται Οι τιμές του ελέγχου είναι αντίστοιχα u(k)=-1,0,1. Το κόστος που πρέπει να ελαχιστοποιηθεί δίνεται:

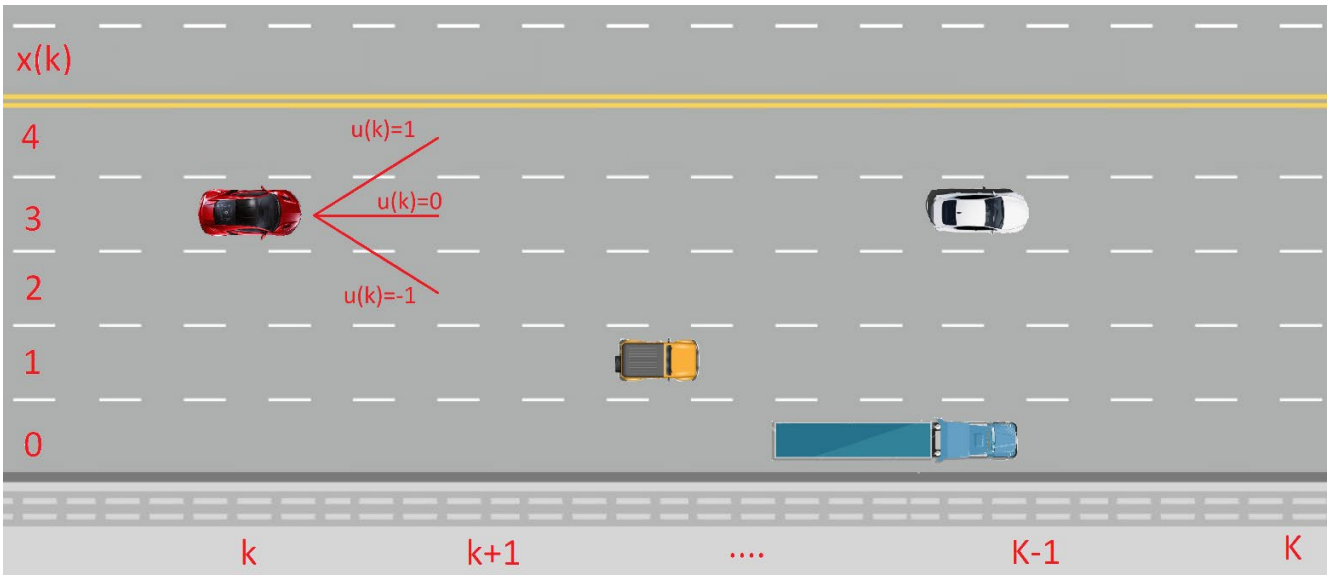
$$J = \vartheta[x(K)] + \sum_{k=0}^{K-1}[(x_{desired} - x(k))^2 + u^2(k)]$$

λαμβάνοντας υπόψη τους παρακάτω περιορισμούς

$$x(k+1) = x(k) + u(k)$$
$$\vartheta[x(K)] = 0, \quad x(K) = x_{desired}$$

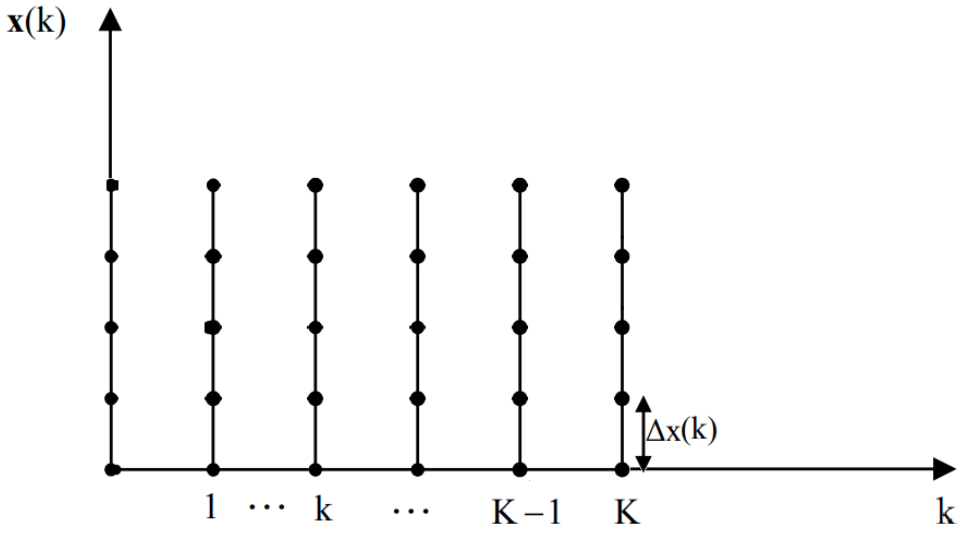
με τις ακόλουθες επιτρεπτές περιοχές κατάστασης και ελέγχου όπου και θα επιχειρήσουμε τη λύση του προβλήματος

$$X(k) = \{x(k) | 0 \le x(k) \le 4\} \qquad x(k) \in \mathbf{Z}$$
$$U(k) = \{u(k) | -1 \le u(k) \le 1\} \qquad u(k) \in \mathbf{Z}$$



Σχήμα 5: Αυτοκινητόδρομος πέντε λωρίδων με πέντε δυνατές καταστάσεις x(k) και τρεις δυνατούς ελέγχους u(k) για κάθε χρονική στιγμή k.

Θεωρούμε ένα διακριτό πλέγμα σημείων μέσα στην επιτρεπτή περιοχή και ορίζουμε το διακριτό διάστημα ίσο με ένα Δx(k)=1. Για κάθε σημείο του πλέγματος $x^i(k)$ έχουμε πεπερασμένο αριθμό μεταβάσεων $u^j(k)$ που καταλήγουν στην επόμενη βαθμίδα k+1. Η βελτιστοποίηση μπορεί έτσι να υλοποιηθεί άμεσα με την σύγκριση των κοστών των μεταβάσεων ενός σημείου του πλέγματος στην επόμενη βαθμίδα. Εφαρμόζοντας τον Δυναμικό Προγραμματισμό αρχίζουμε με το επιθυμητό σημείο προορισμού $x_{desired}$=x(K) και μεταβαίνοντας προς τα πίσω κατά μία βαθμίδα υπολογίζουμε για κάθε κόμβο ξεχωριστά ποια είναι η κατεύθυνση με το μικρότερο κόστος από αυτόν τον κόμβο προς το τελικό σημείο x(K).



Σχήμα 6: Διακριτοποίηση της επιτρεπτής περιοχής κατάστασης x(k).

Για κάθε διακριτό σημείο $x^i(K-1) \in X(K-1)$ πρέπει να εξεταστεί ποια τιμή ελέγχου $u^j(K-1) \in U[x^i(\mathrm{K}-1), K-1]$ ελαχιστοποιεί το συνολικό κόστος μετάβασης J_{K-1} από το σημείο $x^i(K-1)$ προς το $x_{desired}$ τηρώντας τους περιορισμούς:

$$J_{K-1} = \vartheta[x_{desired}] + \varphi[x^i(K-1), u^j(K-1),\ \mathrm{K\text{-}1}]$$
$$= (x(K) - x^i(K-1))^2 + (u^j(K-1))^2$$

Συγκρίνουμε και βρίσκουμε την τιμή $u^j(K-1)$ του διακριτού ελέγχου που ελαχιστοποιεί αυτό το μονοβαθμιαίο πρόβλημα ελαχιστοποίησης για το κάθε σημείο $x^i(K-1)$ της βαθμίδας K-1. Έτσι έχουμε το ελάχιστο κόστος $V[x^i(K-1),K-1]$ για κάθε $x^i(K-1)$ με τον ανάλογο έλεγχο.

Συνεχίζουμε στην βαθμίδα K-2 και με τον ίδιο ακριβώς τρόπο σε κάθε σημείο $x^i(K-1) \in X(K-1)$ βρίσκουμε ποιες τιμές ελέγχου $u^j(K-2) \in U[x^i(K-2),K-2]$ ελαχιστοποιούν το

$$J_{K-2} = V[x^i(K-1), K-1] + \varphi[x^i(K-2), u^j(K-2),\ \mathrm{K\text{-}2}]$$
$$= V[x^i(K-1), K-1] + (x^i(K-1) - x^i(K-2))^2 + (u^j(K-2))^2$$

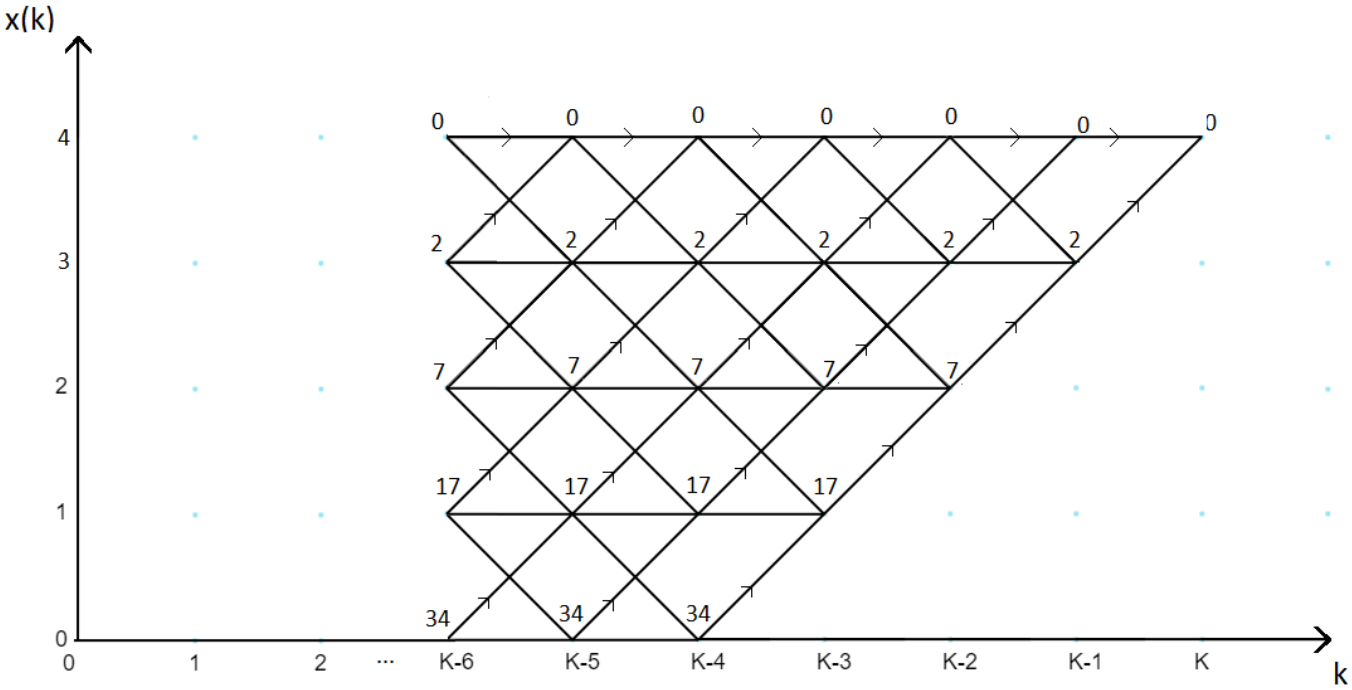
Συνεχίζουμε αυτή τη διαδικασία πηγαίνοντας βαθμιαία προς τα πίσω και για κάθε διακριτή κατάσταση $x^i(k)$ βρίσκουμε την αντίστοιχη διακριτή τιμή ελέγχου $u^j(k)$ που

πραγματοποιεί μια βέλτιστη μετάβαση στην επόμενη βαθμίδα k+1. Οπότε βρίσκουμε και τη βέλτιστη τροχιά από όποιο αρχικό σημείο x_0 επιλέξουμε προς τον τελικό στόχο.

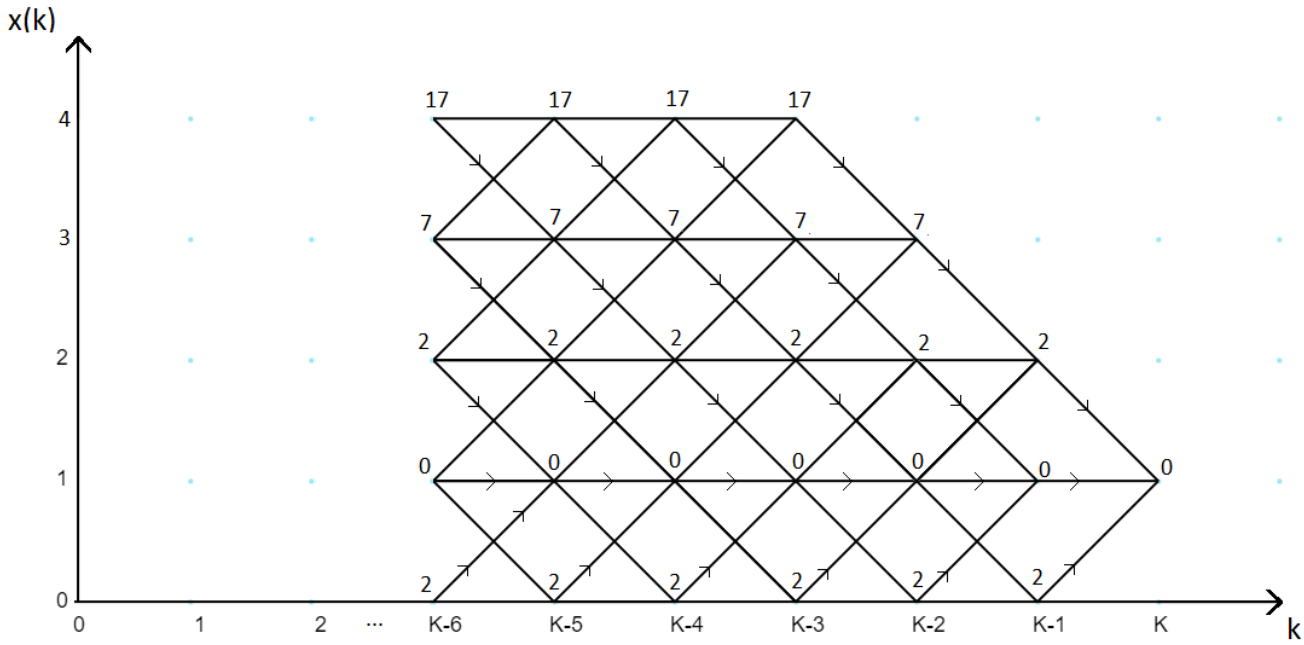
Παρακάτω παρατίθεται το πλέγμα του παραδείγματος με τα ελάχιστα κόστη μετάβασης $V[x^i(\kappa),\kappa]$ αναγραφόμενα πάνω από κάθε σημείο και τον αντίστοιχο βέλτιστο έλεγχο υποδεικνυόμενο με ένα βελάκι για τις εξής περιπτώσεις:

α) $\mathcal{G}[x(4)] = 0, \quad x(4) = x_{desired}$

β) $\mathcal{G}[x(1)] = 0, \quad x(1) = x_{desired}$



Σχήμα 7: Επίλυση του παραδείγματος για τελικό στόχο $x_{desired}$=x(4).



Σχήμα 8: Επίλυση του παραδείγματος για τελικό στόχο $x_{desired}$=x(1).

Παρατηρούμε ότι για όποιο σημείο του πλέγματος επιλέξουμε έχουμε το ελάχιστο κόστος μετάβασής του στον επιθυμητό στόχο και με τη βέλτιστη τροχιά, δηλαδή τα βήματα των αποφάσεων που πρέπει να πάρουμε για να φτάσουμε εκεί.

Ένα ακόμη πολύ σημαντικό συμπέρασμα που προκύπτει βλέποντας την γραφική επίλυση του Σχ.6 και Σχ.7 είναι ότι το ελάχιστο κόστος μεταφοράς και ο βέλτιστος έλεγχος είναι ανεξάρτητα του χρόνου k και εξαρτώνται μόνο από την τιμή x(k) της υπό μεταφοράν κατάστασης. Σε όλες τις χρονικές βαθμίδες ο έλεγχος προκύπτει πάντα ο ίδιος για σταθερή τιμή του x(k). Ο κανόνας ελέγχου που μας οδηγεί το ΠΒΕ επειδή είναι ανεξάρτητος του χρόνου ονομάζεται μη μεταβλητός.

$$u(k) = R[x(k)] = sign[x_{desired} - x(k)]$$

1.5 Σύνοψη Κεφαλαίου

Στο κεφάλαιο αυτό κάναμε μία εισαγωγή στα δυναμικά συστήματα και τον Δυναμικό Προγραμματισμό. . Ο κύριος στόχος του δυναμικού προγραμματισμού είναι η εύρεση ενός κανόνα επιλογής των ενεργειών που ελέγχει μια διαδικασία με το βέλτιστο τρόπο για την επίλυση ενός προβλήματος. Βασική ιδέα του είναι η αποσύνθεση του κύριου προβλήματος σε περισσότερο διαχειρίσιμα υποπροβλήματα. Σύμφωνα με την Αρχή του Βέλτιστου αν θεωρήσουμε μια διαδικασία βελτιστοποίησης καθενός από τα υποπροβλήματα ξεχωριστά τότε ο συνολικός βέλτιστος δρόμος του κύριου προβλήματος είναι αυτός που προκύπτει από το άθροισμα των επιμέρους βέλτιστων βημάτων. Οι υπολογισμοί γίνονται αναδρομικά όπου η βέλτιστη λύση του ενός υποπροβλήματος χρησιμοποιείται ως είσοδος στο επόμενο υποπρόβλημα.

Ο Bellman ανέπτυξε μία αναδρομική διαδικασία για τον υπολογισμό βέλτιστων τιμών συναρτήσεων κέρδους ή κόστους μέσω της ακόλουθης συναρτησιακής εξίσωσης.

$$V[x(k),k] = \min\left\{\varphi[x(k),u(k),k] + V[f(x(k),u(k),k),k+1]\right\}$$

Στη συνέχεια διατυπώθηκε το Πρόβλημα Βέλτιστου Ελέγχου διακριτού χρόνου της αλλαγής λωρίδας ενός αυτοκινήτου που κινείται σε έναν αυτοκινητόδρομο πέντε λωρίδων με σκοπό να φτάσει σε έναν επιθυμητό τελικό στόχο. Έπειτα επιλύθηκε με τον αναδρομικό τύπο του Bellman για την ελαχιστοποίηση της συνάρτησης κόστους του.

Στο επόμενο κεφάλαιο μελετάται η επίλυση του στοχαστικού προβλήματος όπου εισάγεται μία επιπλέον μεταβλητή διαταραχών της οποίας η τιμή δεν είναι σταθερή αλλά καθορίζεται από μια συνάρτηση πιθανολογικής κατανομής.

2 ΣΤΟΧΑΣΤΙΚΟΣ ΔΥΝΑΜΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

Μέχρι στιγμής το πρόβλημα βελτιστοποίησης που έχουμε ασχοληθεί είναι αιτιοκρατικό, δηλαδή οι τιμές όλων των μεγεθών θεωρούνται γνωστές. Όσο προσπαθούμε να κάνουμε τα μοντέλα πιο ρεαλιστικά και όταν έχουμε καταστάσεις που αφορούν το μέλλον ή περιέχουν τυχαιότητα τότε οι τιμές διαφόρων παραμέτρων παρουσιάζουν αβεβαιότητα. Ο προσδιορισμός των τιμών αυτών γίνεται με ρεαλιστικές υποθέσεις ως προς την πιθανολογική κατανομή τους και μπορεί να αποδοθεί με τυχαίες ή στοχαστικές μεταβλητές. Σε ένα τέτοιο πλαίσιο Στοχαστικού Δυναμικού Προγραμματισμού δεν μιλάμε πλέον για ελαχιστοποίηση του κριτήριου κόστους, αλλά της αναμενόμενης τιμής του.

2.1 Το Πρόβλημα Στοχαστικού Βέλτιστου Ελέγχου

Παρακάτω διατυπώνουμε ένα γενικό Πρόβλημα Στοχαστικού Δυναμικού Βέλτιστου Ελέγχου Διακριτού χρόνου του οποίου η επίλυση οδηγεί σε στοχαστικό βέλτιστο κανόνα ελέγχου.

Έστω ένα σύστημα διακριτού χρόνου

$$x(k+1) = f[x(k), u(k), z(k), k], \qquad k = 0, \ldots, K-1 \tag{2.1}$$

Το διάνυσμα z(k) των διαταραχών λαμβάνει τιμές από την περιοχή Z(k) μέσω μιας συνάρτησης πιθανολογικής κατανομής P[z|x(k),u(k),k]. Τα διανύσματα παίρνουν τιμές στις επιτρεπτές περιοχές τιμών $x(k) \in X(K)$, $u(k) \in U(K)$, $z(k) \in Z(K)$.

Η αναμενόμενη τιμή του κριτήριου κόστους ως προς την στοχαστική διαταραχή ορίζεται ως:

$$J = \mathrm{E}\left\{ \vartheta[x(K)] + \sum_{k=0}^{K-1} \varphi[x(k), u(k), z(k), k] \right\} \tag{2.2}$$

Η εύρεση ενός κανόνα ελέγχου που θα ελαχιστοποιήσει αυτό το κριτήριο κόστους αποτελεί τη λύση του προβλήματος στοχαστικού βέλτιστου ελέγχου

2.2 Μέθοδος Στοχαστικού Δυναμικού Προγραμματισμού

Όπως και πριν στην αιτιοκρατική περίπτωση, για την εφαρμογή της Αρχής του Βέλτιστου στο Στοχαστικό Πρόβλημα Βέλτιστου Ελέγχου διακριτού χρόνου ορίζουμε αρχικά το υπολειπόμενο αναμενόμενο κόστος μεταφοράς J_k ενός σημείου προς τον τελικό στόχο και το ελάχιστο κόστος V[x(k),k], έχοντας λάβει υπόψιν τους περιορισμούς ως εξής:

$$J_k = \underset{z(\kappa)}{\mathrm{E}}\left\{ \vartheta[x(K)] + \sum_{\kappa=k}^{K-1} \varphi[x(\kappa), u(\kappa), z(\kappa), \kappa] \right\} \tag{2.3}$$

$$V[x(k),k] = \min J_k = \min\left\{ \underset{z(k)}{\mathrm{E}} \left\{ \varphi[x(k), u(k), z(k), k] + \mathrm{J}_{k+1} \right\} \right\} \tag{2.4}$$

Οι μεταβλητές κατάστασης x(k), ελέγχου u(k) και η στοχαστική μεταβλητή z(k) δεν εξαρτώνται από επόμενες τιμές z(k+1),…z(K-1) και η αναμενόμενη τιμή λαμβάνεται μόνο ως προς z(k).

Εφαρμόζοντας την Αρχή του Βέλτιστου

$$V[x(k),k] = \min\left\{ \underset{z(k)}{\mathrm{E}} \left\{ \varphi[x(k), u(k), z(k), k] + V[x(k+1), k+1] \right\} \right\} \tag{2.5}$$

Όπου αντικαθιστούμε από την και παίρνουμε

$$V[x(k),k] = \min\left\{ \underset{z(k)}{\mathrm{E}} \left\{ \varphi[x(k), u(k), z(k), k] + V[f[x(k), u(k), z(k), k], k+1] \right\} \right\} \tag{2.6}$$

Με οριακή συνθήκη

$$V[x(K),K] = \vartheta[x(K)] \tag{2.7}$$

Και έτσι το πρόβλημα πολλαπλών βαθμίδων έχει μετατραπεί σε K μονοβαθμιαία προβλήματα αποφάσεων

Η εξ. (2.6) Αποτελεί την στοχαστική μορφή του αναδρομικού τύπου του Bellman.

2.3 Διακριτός Στοχαστικός Δυναμικός Προγραμματισμός

Ερχόμαστε στο πρόβλημα που εξετάσαμε προηγουμένως με το όχημα που κινείται σε αυτοκινητόδρομο πέντε (5) λωρίδων και εισάγουμε μία διαταραχή z(k). Η z(k) έιναι διακριτή μεταβλητή με διάστημα Δz(k)=1 και παίρνει τιμές z(k)=-1,0,1. Η διαταραχή αυτή εκφράζει τυχαίους εξωτερικούς παράγοντες που μπορεί να συμβούν κατά τη διάρκεια μιας πορείας με αποτέλεσμα μία αλλαγή λωρίδας να μην μπορεί να πραγματοποιηθεί είτε να πραγματοποιηθεί διαφορετική αλλαγή από αυτή της μεταβλητής ελέγχου.

Η καταστατική εξίσωση διαμορφώνεται ως εξής

$$x(k+1) = x(k) + u(k) + z(k) = f[x(k), u(k), z(k), k]$$

Η τελική θέση του οχήματος μπορεί να είναι οποιαδήποτε από τις 5 πιθανές λωρίδες. Μπορούμε να τιμωρήσουμε ενδεχόμενες διαφορές από τον επιθυμητό στόχο ορίζοντας την

$$\vartheta[x(K)] = [x(K) - x_{desired}]^2,$$

Με

$$X(k) = \{x(k) | 0 \le x(k) \le 4\} \qquad x(k) \in \mathbf{Z}$$
$$U(k) = \{u(k) | -1 \le u(k) \le 1\} \qquad u(k) \in \mathbf{Z}$$
$$Z(k) = \{z(k) | -1 \le z(k) \le 1\} \qquad z(k) \in \mathbf{Z}$$

το κόστος μετάβασης από ένα σημείο x(k) στην επόμενη βαθμίδα k+1 παραμένει

$$\varphi[x(k), u(k), z(k), k] = (x(k) - x_{desired})^2 + u^2(k)$$

Όπου είναι ανεξάρτητο του z(k)

Το ελάχιστο κόστος γίνεται

$$V[x(k),k]=\min\left\{\varphi[x(k),u(k),z(k),k]+\underset{z(k)}{\mathrm{E}}\left\{V[f[x(k),u(k),z(k),k],k+1\right\}\right\}$$

Αντικαθιστώντας το f[x(k),u(k),z(k),k] παίρνουμε

$$V[x(k),k]=\min\left\{\varphi[x(k),u(k),z(k),k]+\underset{z(k)}{\mathrm{E}}\left\{V[x(k)+u(k)+z(k),k+1\right\}\right\}$$

$$\begin{aligned}V[x(k),k]=\min_{u(k)}\{&\varphi[x(k),u(k),z(k),k]+\mathrm{p}(z=1|x(k),u(k))\cdot V(x(k)+u(k)+1)(k+1)+\\&+\mathrm{p}(z=0|x(k),u(k))\cdot V(x(k)+u(k))(k+1)+\\&+\mathrm{p}(z=-1|x(k),u(k))\cdot V(x(k)+u(k)-1)(k+1)\}\end{aligned}$$

Στο παράδειγμά μας θεωρούμε τις εξής πιθανολογικές κατανομές p(z(k),x(k),u(k)]

Το όχημα του προβλήματος σε κάθε μετάβαση μπορεί να μετακινηθεί μέχρι μία λωρίδα δεξιά η αριστερά και όχι παραπάνω. Γίνεται λοιπόν κατανοητό ότι οι περιπτώσεις

- u(k)=1 και z(k)=1
- u(k)=-1 και z(k)=-1

είναι ανέφικτες.

Σ' αυτές τις περιπτώσεις ορίζουμε p1 την πιθανότητα να ακολουθήσει την απόφαση που έχει παρθεί, 1-p1 την πιθανότητα να μείνει τελικά στην ίδια πορεία, δηλαδή z(k)=-u(k) οπότε x(k+1)=x(k)

Επίσης στις ακριανές λωρίδες x(k)=4 και x(k)=0 είναι αδύνατες οι u(k)=1 και u(k)=-1 αντίστοιχα από τους περιορισμούς καθώς καταλήγουν σε θέσεις εκτός πλέγματος.

Για τις ενδιάμεσες λωρίδες x=1,2,3 για u(k)=0 θεωρούμε p1 την πιθανότητα να ακολουθήσει την απόφαση που έχει παρθεί, p2 την πιθανότητα να μετακινηθεί μία λωρίδα δεξιά και 1-p1-p2 την πιθανότητα μία λωρίδα αριστερά

x=1,2,3	z / u	-1	0	1
	1	1-p1	p1	0
	0	p2	p1	1-p1-p2
	-1	0	p1	1-p1

Πίνακας 1: Πιθανολογική κατανομή διαταραχής για τις ενδιάμεσες λωρίδες x=1,2,3.

Για την ακριανή λωρίδα x=4 για u(k)=0 θεωρούμε p1 την πιθανότητα να ακολουθήσει την απόφαση που έχει παρθεί, 1-p1 την πιθανότητα να μετακινηθεί μία λωρίδα δεξιά

x=4	z / u	-1	0	1
	1	0	0	0
	0	1-p1	p1	0
	-1	0	p1	1-p1

Πίνακας 2: Πιθανολογική κατανομή διαταραχής για την ακριανή λωρίδα x=4.

Για την ακριανή λωρίδα x=0 για u(k)=0 θεωρούμε p1 την πιθανότητα να ακολουθήσει την απόφαση που έχει παρθεί, 1-p1 την πιθανότητα να μετακινηθεί μία λωρίδα αριστερά

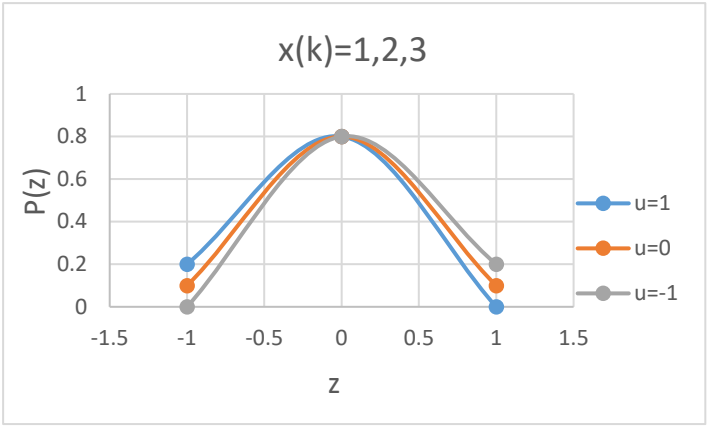
x=0	z / u	-1	0	1
	1	1-p1	p1	0
	0	0	p1	1-p1
	-1	0	0	0

Πίνακας 3: Πιθανολογική κατανομή διαταραχής για την ακριανή λωρίδα x=0.

Για p1=0,8 p2=0,1 οι κατανομές φαίνονται στον παρακάτω πίνακα

x=1,2,3	u \ z	-1	0	1
	1	0,2	0,8	0
	0	0,1	0,8	0,1
	-1	0	0,8	0,2

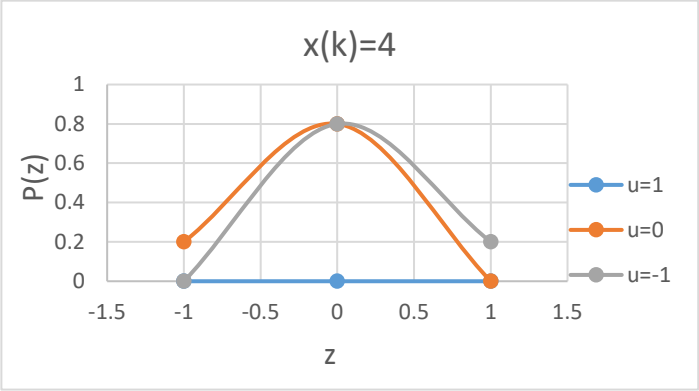
(α) x(k)=1,2,3



(α) x(k)=1,2,3

x=4	u \ z	-1	0	1
	1	0	0	0
	0	0,2	0,8	0
	-1	0	0,8	0,2

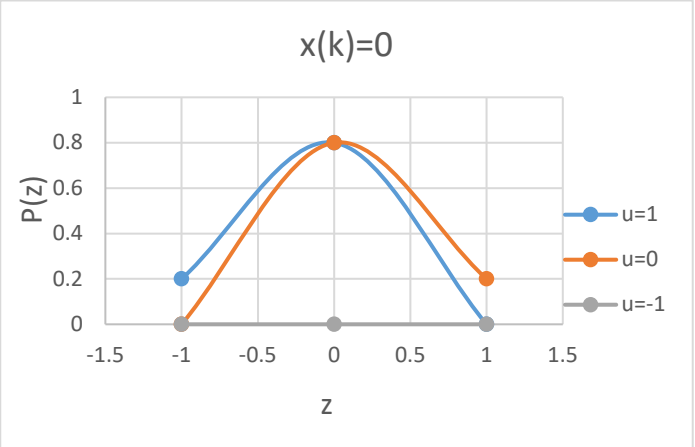
(β) x(k)=4



(β) x(k)=4

x=0	u \ z	-1	0	1
	1	0,2	0,8	0
	0	0	0,8	0,2
	-1	0	0	0

(γ) x(k)=0



(γ) x(k)=0

Πίνακας 4: Κατανομές πιθανοτήτων διαταραχής για p1=0,8 p2=0,1.

Σχήμα 9: Κατανομές πιθανοτήτων διαταραχής για p1=0,8 p2=0,1.

Επίλυση

Αρχικά υπολογίζουμε για $x(K) \in (0,1,2,3,4)$ την ποινή $\vartheta[x(K)] = [x(K) - x_{desired}]^2$ που είχαμε βάλει, για το επιθυμητό σημείο τελικού στόχου $x_{desired}$ που έχουμε ορίσει.

Για παράδειγμα, για $x_{desired} = 3$

$$\gamma\iota\alpha\ x(K) = 0,\ \ \vartheta[x(K)] = \vartheta[0] = [x(K) - x_{desired}]^2 = [0-3]^2 = 9$$
$$\gamma\iota\alpha\ x(K) = 1,\ \ \vartheta[x(K)] = \vartheta[1] = [x(K) - x_{desired}]^2 = [1-3]^2 = 4$$
$$\gamma\iota\alpha\ x(K) = 2,\ \ \vartheta[x(K)] = \vartheta[2] = [x(K) - x_{desired}]^2 = [2-3]^2 = 1$$
$$\gamma\iota\alpha\ x(K) = 3,\ \ \vartheta[x(K)] = \vartheta[3] = [x(K) - x_{desired}]^2 = [3-3]^2 = 0$$
$$\gamma\iota\alpha\ x(K) = 4,\ \ \vartheta[x(K)] = \vartheta[4] = [x(K) - x_{desired}]^2 = [4-3]^2 = 1$$

Για κάθε x(K-1) πρέπει να βρεθεί ο έλεγχος u(K-1) που ελαχιστοποιεί το

$$\begin{aligned} V[x(\mathrm{K}-1),\mathrm{K}-1] = \min_{u(K-1)} \{ & (x(K-1) - x_{desired})^2 + u^2(K-1) + \\ & + \mathrm{p}(z=1 \,|\, x(K-1), u(K-1)) \cdot V(x(K-1) + u(K-1) + 1)(K) + \\ & + \mathrm{p}(z=0 \,|\, x(K-1), u(K-1)) \cdot V(x(K-1) + u(K-1))(K) + \\ & + \mathrm{p}(z=-1 \,|\, x(K-1), u(K-1)) \cdot V(x(K-1) + u(K-1) - 1)(K) \} \end{aligned}$$

Π.χ. για $x(K-1) = 1$ πρέπει να υπολογιστεί το κόστος J_{K-1} για τις 3 τιμές ελέγχου

Για $u(K-1) = -1$

$$\begin{aligned} J_{K-1} = & \{ (1-3)^2 + (-1)^2 + \mathrm{p}(z=1 \,|\, 1,-1) \cdot V(1-1+1)(K) + \\ & + \mathrm{p}(z=0 \,|\, 1,-1) \cdot V(1-1)(K) + \mathrm{p}(z=-1 \,|\, 1,-1) \cdot V(1-1-1)(K) \} \\ = & 5 + (1-p1) * 4 + p1 * 9 \\ = & 9 + 5p1 \end{aligned}$$

Για $u(K-1)=0$

$$J_{K-1}=\{(1-3)^2+(0)^2+\mathrm{p}(z=1|1,0)\cdot V(1+1)(K)+$$
$$+\mathrm{p}(z=0|1,0)\cdot V(1)(K)+\mathrm{p}(z=-1|1,0)\cdot V(1-1)(K)\}$$
$$=4+(1-p1-p2)*1+p1*4+p2*9$$
$$=5+3p1+8p2$$

Για $u(K-1)=1$

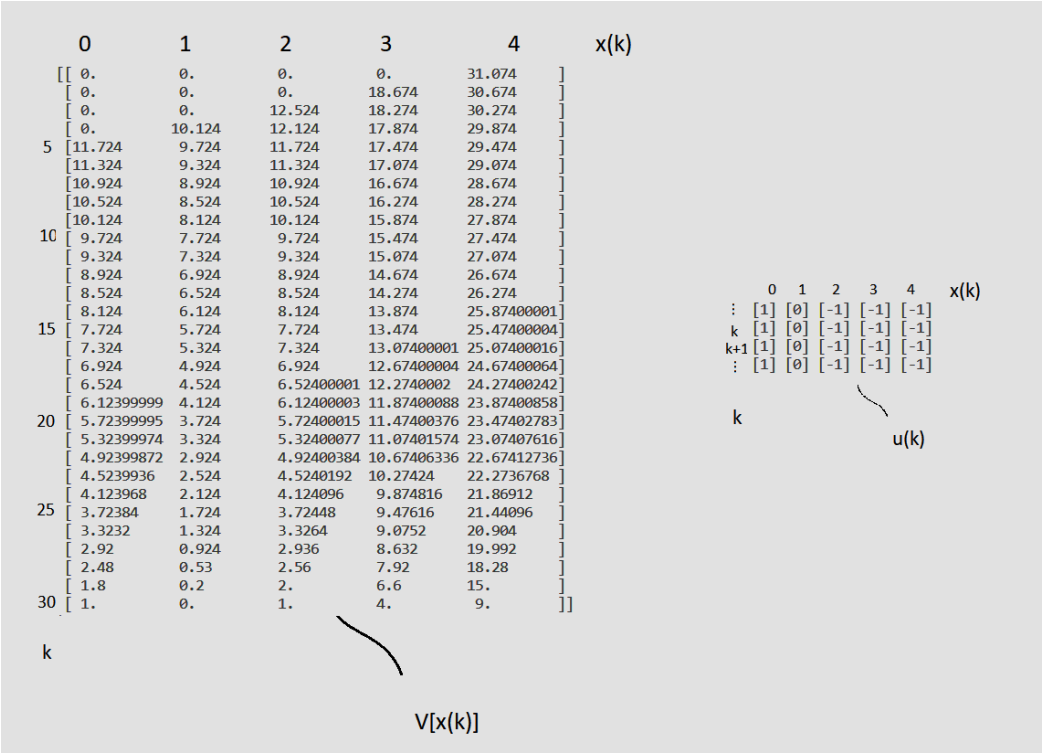
$$J_{K-1}=\{(1-3)^2+(1)^2+\mathrm{p}(z=1|1,1)\cdot V(1+1+1)(K)+$$
$$+\mathrm{p}(z=0|1,1)\cdot V(1+1)(K)+\mathrm{p}(z=-1|1,1)\cdot V(1+1-1)(K)\}$$
$$=5+p1*1+(1-p1)*4$$
$$=9-3p1$$

Το ελάχιστο των τριών αυτών αποτελεί το αναμενόμενο ελάχιστο κόστος V[1,K-1] και το u(K-1) από το οποίο προήλθε τον βέλτιστο έλεγχο.

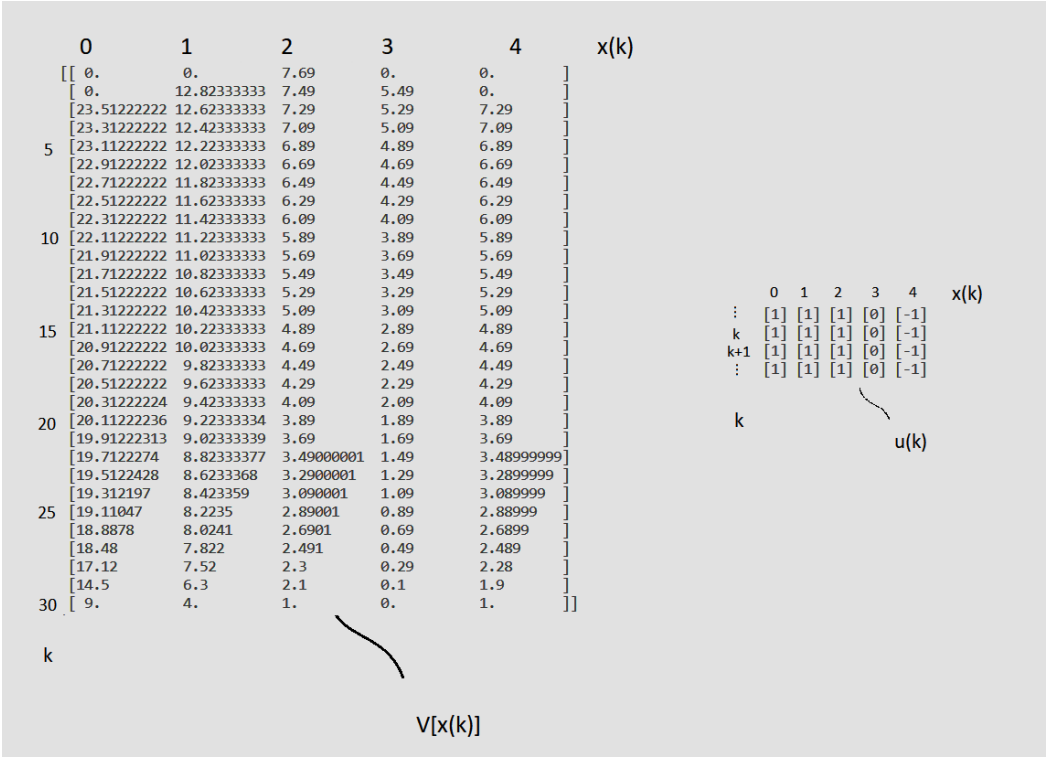
Επαναλαμβάνουμε αυτή τη διαδικασία για τα x(K-1)=0,2,3,4 της βαθμίδας K-1, και έπειτα προχωράμε βαθμιαία προς τα πίσω για όλα τα επιτρεπτά σημεία του πλέγματος βρίσκοντας την αντίστοιχη τιμή ελέγχου που οδηγεί στην επόμενη βαθμίδα κατά βέλτιστο τρόπο με το μικρότερο αναμενόμενο κόστος .

2.4 Αποτελέσματα

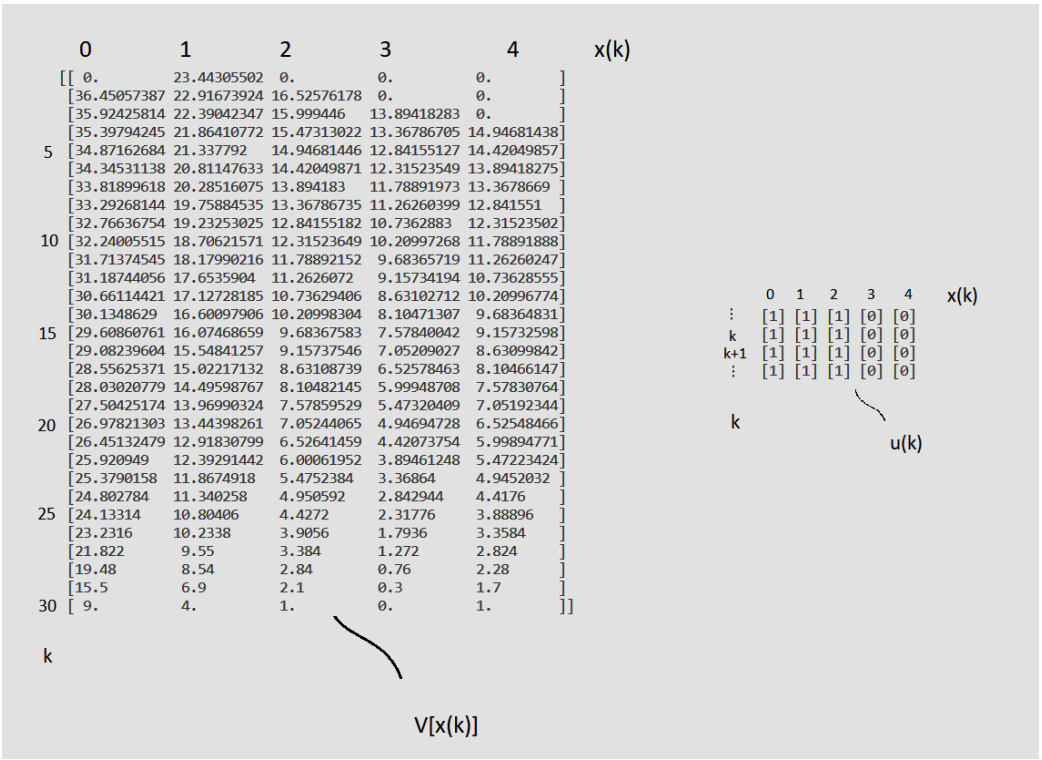
Πραγματοποιήθηκαν υπολογισμοί με τη βοήθεια κώδικα[1] σε περιβάλλον Python, για όλα τα επιθυμητά τελικά σημεία και για διάφορα ζεύγη πιθανοτήτων. Για τελικό στόχο τα σημεία x(K)=0,2,4 με όλα τα ζεύγη πιθανοτήτων προέκυψε ο κανόνας ελέγχου $R[x(k)] = sign(x_{desired} - x(k)]$. Για x(K)=1,3 ο κανόνας ελέγχου άλλαξε για p1≤0,75 όπως φαίνεται στους πίνακες παρακάτω.



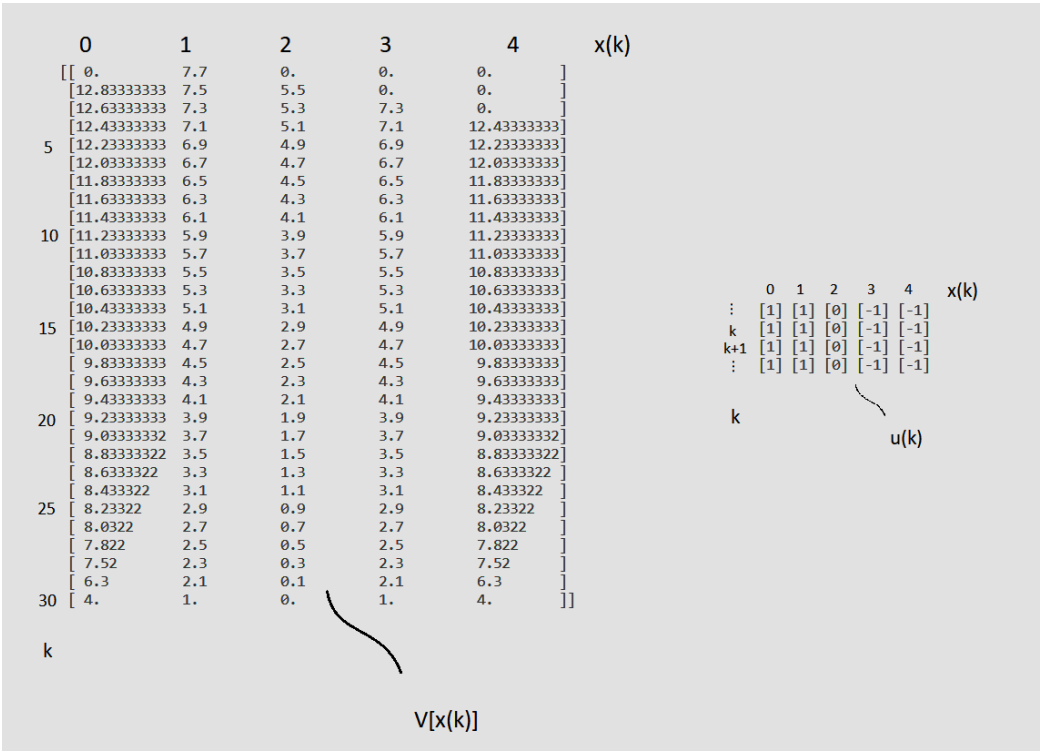
Σχήμα 10: Επίλυση του παραδείγματος για τελικό στόχο $x_{desired}$=x(1), θέση εκκίνησης x(0)=4, p1=0.8 και p2=0.15 σε τελικό χρόνο K=30.



Σχήμα 11: Επίλυση του παραδείγματος για τελικό στόχο $x_{desired}$=x(3), θέση εκκίνησης x(0)=2, p1=0.9, p2=0.05 σε τελικό χρόνο K=30.



Σχήμα 12: Επίλυση του παραδείγματος για τελικό στόχο $x_{desired}$=x(3), θέση εκκίνησης x(0)=1, p1=0.7, p2=0.1 σε τελικό χρόνο K=30.



Σχήμα 13: Επίλυση του παραδείγματος για τελικό στόχο $x_{desired}$=x(2), θέση εκκίνησης x(0)=1, p1=0.9, p2=0.05 σε τελικό χρόνο K=30.

			xdesired=xd=3					
p1	0,9	0,85	0,8	0,8	0,75	0,75	0,7	0,7
p2	0,05	0,1	0,1	0,15	0,15	0,2	0,2	0,15
1-p1-p2	0,05	0,05	0,1	0,05	0,1	0,05	0,1	0,15
R[x(k)]	sign(xd-x(k))		x(k)	R[x(k)]	x(k)	R[x(k)]		
			0	1	0	1		
			1	1	1	1		
			2	1	2	1		
			3	0	3	0		
			4	-1	4	0		

			xdesired=xd=1					
p1	0,9	0,85	0,8	0,8	0,75	0,75	0,7	0,7
p2	0,05	0,1	0,1	0,15	0,15	0,2	0,2	0,15
1-p1-p2	0,05	0,05	0,1	0,05	0,1	0,05	0,1	0,15
R[x(k)]	sign(xd-x(k))		x(k)	R[x(k)]	x(k)	R[x(k)]		
			0	1	0	0		
			1	0	1	0		
			2	-1	2	-1		
			3	-1	3	-1		
			4	-1	4	-1		

			xdesired=xd=4					
p1	0,9	0,85	0,8	0,8	0,75	0,75	0,7	0,7
p2	0,05	0,1	0,1	0,15	0,15	0,2	0,2	0,15
1-p1-p2	0,05	0,05	0,1	0,05	0,1	0,05	0,1	0,15
R[x(k)]	sign(xd-x(k))			x(k)	R[x(k)]			
				0	1			
				1	1			
				2	1			
				3	1			
				4	0			

			xdesired=xd=2					
p1	0,9	0,85	0,8	0,8	0,75	0,75	0,7	0,7
p2	0,05	0,1	0,1	0,15	0,15	0,2	0,2	0,15
1-p1-p2	0,05	0,05	0,1	0,05	0,1	0,05	0,1	0,15
R[x(k)]	sign(xd-x(k))			x(k)	R[x(k)]			
				0	1			
				1	1			
				2	0			
				3	-1			
				4	-1			

			xdesired=xd=0					
p1	0,9	0,85	0,8	0,8	0,75	0,75	0,7	0,7
p2	0,05	0,1	0,1	0,15	0,15	0,2	0,2	0,15
1-p1-p2	0,05	0,05	0,1	0,05	0,1	0,05	0,1	0,15
R[x(k)]	sign(xd-x(k))			x(k)	R[x(k)]			
				0	0			
				1	-1			
				2	-1			
				3	-1			
				4	-1			

Πίνακας 5: Κανόνες ελέγχου για πιθανούς συνδυασμούς πιθανοτήτων p1 και p2 και για διαφορετικούς επιθυμητούς τελικούς στόχους.

2.5 Σύνοψη Κεφαλαίου

Στο κεφάλαιο αυτό το Πρόβλημα Βέλτιστου Ελέγχου διακριτού χρόνου μετατρέπεται σε Στοχαστικό με την προσθήκη της διαταραχής z όπου η τιμή της δεν είναι σταθερή αλλά λαμβάνεται από μία συνάρτηση πιθανολογικής κατανομής. Η διαταραχή αυτή εκφράζει οποιοδήποτε εξωτερικό παράγοντα μπορεί να επηρεάσει μια απόφαση εναλλαγής λωρίδας. Έτσι με τη χρήση του αναδρομικού τύπου του Bellman ο Στοχαστικός πλέον Δυναμικός Προγραμματισμός ελαχιστοποιεί όχι την ίδια τη συνάρτηση κόστους αλλά την αναμενόμενη τιμή της.

Η επιλογή της τιμής της πιθανότητας p1 της πιθανολογικής κατανομής επηρεάζει τον κανόνα ελέγχου που προκύπτει με σημείο αλλαγής την τιμή 0,75, για επιθυμητό τελικό στόχο $x_{desired}$=1,3. Για τους υπόλοιπους επιθυμητούς στόχους ο κανόνας ελέγχου προκύπτει ίδιος.

Στη συνέχεια οι κανόνες ελέγχου που προέκυψαν θα προσομοιωθούν σε ένα πρόγραμμα αστικής κινητικότητας για να ελεγχθεί η αποτελεσματικότητά τους.

3 ΠΡΟΣΟΜΟΙΩΣΗ

Οι κανόνες ελέγχου που προέκυψαν από τον δυναμικό προγραμματισμό ουσιαστικά προτείνουν μία στρατηγική αλλαγής λωρίδας κατά την πορεία των οχημάτων προς τον τελικό στόχο. Η αποτελεσματικότητα αυτών θα εξακριβωθεί σε ένα πρόγραμμα προσομοίωσης όπου θα αναπαρασταθεί το σενάριο του οχήματός μας όσο το δυνατόν πιο κοντά σε ρεαλιστικές συνθήκες με διάφορες παραδοχές.

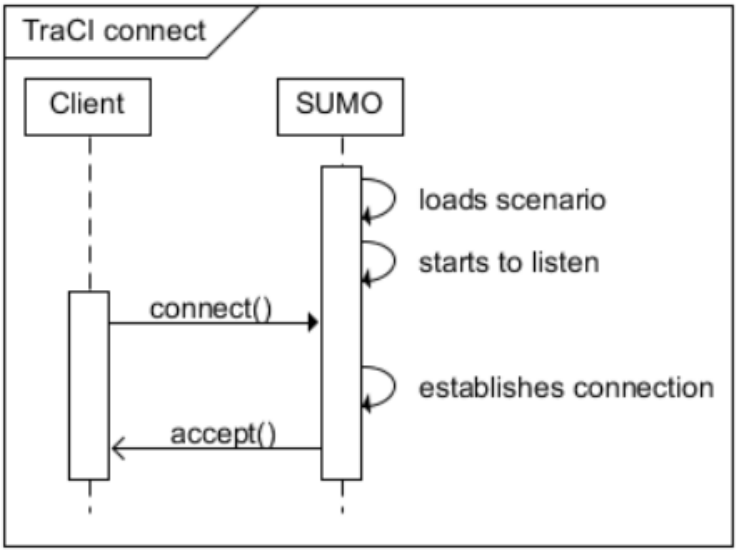
3.1 Προσομοιωτής – Sumo-Gui (Simulation of Urban Mobility)

Στην παρούσα εργασία το πρόγραμμα προσομοίωσης που χρησιμοποιήθηκε είναι το Eclipse Sumo (Simulation of Urban Mobility). Αποτελεί ένα ανοιχτού κώδικα, συνεχές πακέτο προσομοίωσης κυκλοφορίας πολλαπλών μέσων, σχεδιασμένο για να χειρίζεται μεγάλα δίκτυα. Επιτρέπει τη μοντελοποίηση συστημάτων κυκλοφορίας - συμπεριλαμβανομένων των οδικών οχημάτων, των μέσων μαζικής μεταφοράς και των πεζών. Το SUMO περιλαμβάνεται πληθώρα εργαλείων υποστήριξης που αυτοματοποιούν βασικές εργασίες για τη δημιουργία, την εκτέλεση και την αξιολόγηση προσομοιώσεων κυκλοφορίας, όπως εισαγωγή δικτύου, υπολογισμοί χαρακτηριστικών διαδρομών οχημάτων και οπτικοποίηση. Επίσης παρέχει τη δυνατότητα απομακρυσμένου ελέγχου της προσομοίωσης με διάφορους μηχανισμούς που επιτρέπουν σε δύο στοιχεία λογισμικού να επικοινωνούν μεταξύ τους χρησιμοποιώντας ένα σύνολο ορισμών και πρωτοκόλλων τα λεγόμενα API’s. Η δυνατότητα ελέγχου πολλών παραμέτρων προσαρμοσμένες στις ανάγκες του μοντέλου μας, ο εύκολος σχεδιασμός καθώς και η χρήση ρεαλιστικών μοντέλων οδικής κυκλοφορίας υπήρξαν οι λόγοι που επιλέχθηκε ως περιβάλλον προσομοίωσης

Η μελλοντική ανάπτυξη και ο οδικός χάρτης του SUMO διαμορφώνεται και προωθείται από την ομάδα εργασίας openMobility στο Ίδρυμα Eclipse. Αρκετοί εταίροι από τη βιομηχανία και τον ακαδημαϊκό χώρο εργάζονται μαζί για μια κοινή πλατφόρμα προσομοίωσης κινητικότητας για βιομηχανικές εφαρμογές και ακαδημαϊκή έρευνα.

3.2 TraCI (Traffic Control Interface)

Το Traffic Control Interface ή εν συντομία TraCI μας δίνει τη δυνατότητα να έχουμε πρόσβαση σε μια προσομοίωση τρέχουσας οδικής κυκλοφορίας και να επεμβαίνουμε σε αυτή. Επιτρέπει την ανάκτηση τιμών διαφόρων χαρακτηριστικών των προσομοιωμένων αντικειμένων και την αλλαγή της κατάστασής τους ελέγχοντας την συμπεριφορά τους "on-line". Επηρεάζει τη συμπεριφορά των οχημάτων και μπορεί να ζητήσει περιβαλλοντικές πληροφορίες. Χρησιμοποιεί το μοντέλο αρχιτεκτονικής λογισμικού πελάτη-διακομιστή βασισμένο στο πρωτόκολλο ελέγχου μεταφοράς (TCP) για να παρέχει πρόσβαση στο Sumo. Ως εκ τούτου το Sumo λειτουργεί ως διακομιστής που ξεκινά με πρόσθετες εντολές. Επιτρέπει τη σύνδεση σε πάνω από ένα χρήστες.



Σχήμα 14: Λειτουργία σύνδεσης TraCI με Sumo.

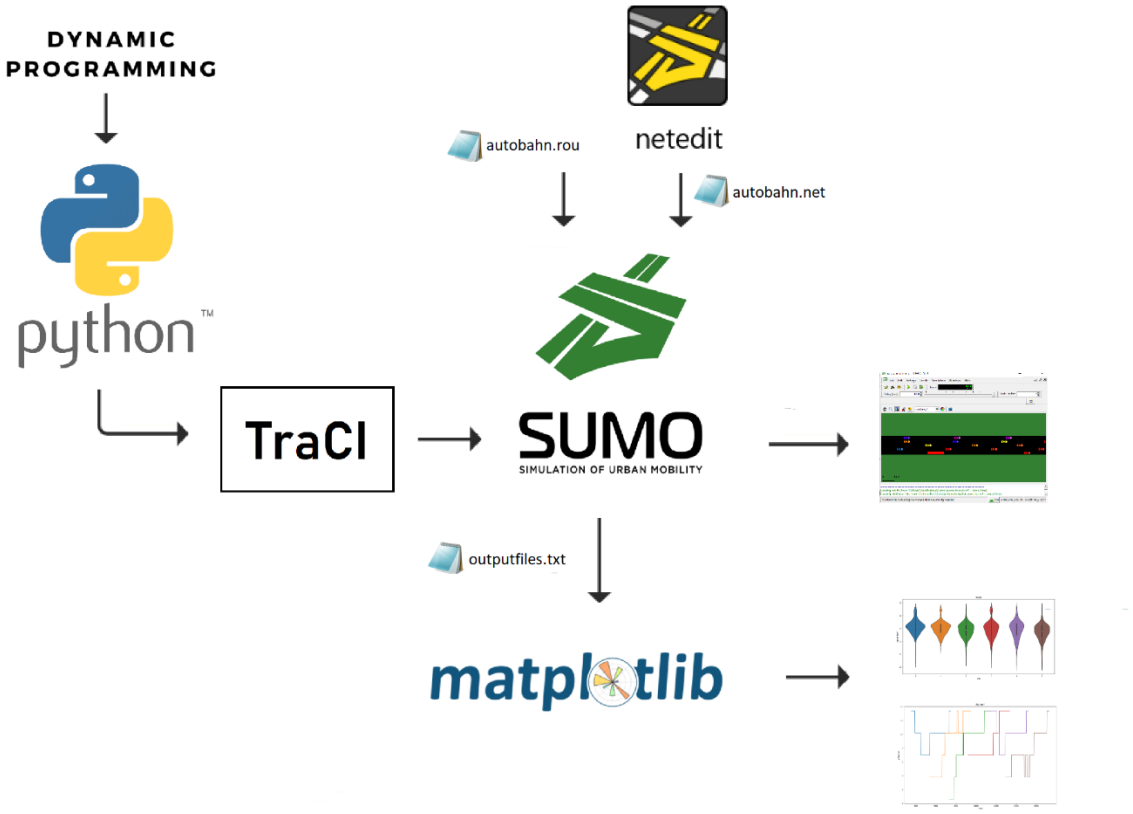
Κατά την εκκίνηση το Sumo προετοιμάζει μόνο την προσομοίωση και περιμένει όλες τις εξωτερικές εφαρμογές να συνδεθούν και να πάρουν τον έλεγχο

3.3 Netedit

Το Netedit αποτελεί ένα πρόγραμμα επεξεργασίας γραφικών δικτύου. Μπορεί να χρησιμοποιηθεί για τη δημιουργία δικτύων από την αρχή, για την τροποποίηση όλων των πτυχών των υπαρχόντων δικτύων καθώς και για τον εντοπισμό σφαλμάτων χαρακτηριστικών δικτύου. Το netedit είναι χτισμένο πάνω από το netconvert. Ως γενικός εμπειρικός κανόνας, οτιδήποτε μπορεί να κάνει το netconvert, το netedit μπορεί επίσης να κάνει. Το netedit έχει απεριόριστες δυνατότητες αναίρεσης/επανάληψης και έτσι επιτρέπει τη γρήγορη διόρθωση των λαθών επεξεργασίας. Η διεπαφή χρήστη ακολουθεί πιστά αυτή του sumo-gui.

3.4 Αλληλεπίδραση των εργαλείων μεταξύ τους

Αρχικά χρησιμοποιούμε το Netedit για να σχεδιάσουμε και να διαμορφώσουμε το οδικό μας δίκτυο σύμφωνα με τις απαιτήσεις μας. Σχεδιάζοντας το δίκτυο μας το Netedit δημιουργεί ένα αρχείο *.net το οποίο παρέχει όλες τις πληροφορίες και τα χαρακτηριστικά του δικτύου που φτιάξαμε. Επίσης δημιουργούμε ένα ακόμα αρχείο κειμένου *.rou το οποίο καθορίζει και διαμορφώνει τον τύπο και τα χαρακτηριστικά των ροών των οχημάτων που πρόκειται να προσομοιωθούν. Για να μπορέσουμε να χρησιμοποιήσουμε τα δύο προαναφερθέντα αρχεία, διαμορφώνουμε ένα τρίτο αρχείο *.cfg το οποίο ουσιαστικά συνδέει μαζί τα δύο προηγούμενα και κατευθύνει το Sumo να βρει τους ορισμούς των οχημάτων, των ροών και τα χαρακτηριστικά του δικτύου που έχει σχεδιαστεί. Έτσι το Sumo τρέχει την προσομοίωση που έχουμε δημιουργήσει με τα χαρακτηριστικά που θέλουμε. Μέσω της Python και των κανόνων ελέγχου που έχουν προκύψει από το δυναμικό προγραμματισμό ελέγχουμε την προσομοίωση που τρέχει, online μέσω του TraCI και πραγματοποιούμε αλλαγές στη λωρίδα συγκεκριμένων οχημάτων. Επίσης καθώς η προσομοίωση του Sumo τρέχει παράγει κάποια αρχεία output με πληροφορίες και τιμές χαρακτηριστικών των οχημάτων που εμείς έχουμε ορίσει. Αυτά χρησιμοποιούνται ως δεδομένα για τη δημιουργία διαγραμμάτων και πινάκων μέσω μιας βιβλιοθήκη της Python, την Matplotlib.



Σχήμα 15: Αλληλεπίδραση των εργαλείων που χρησιμοποιήθηκαν.

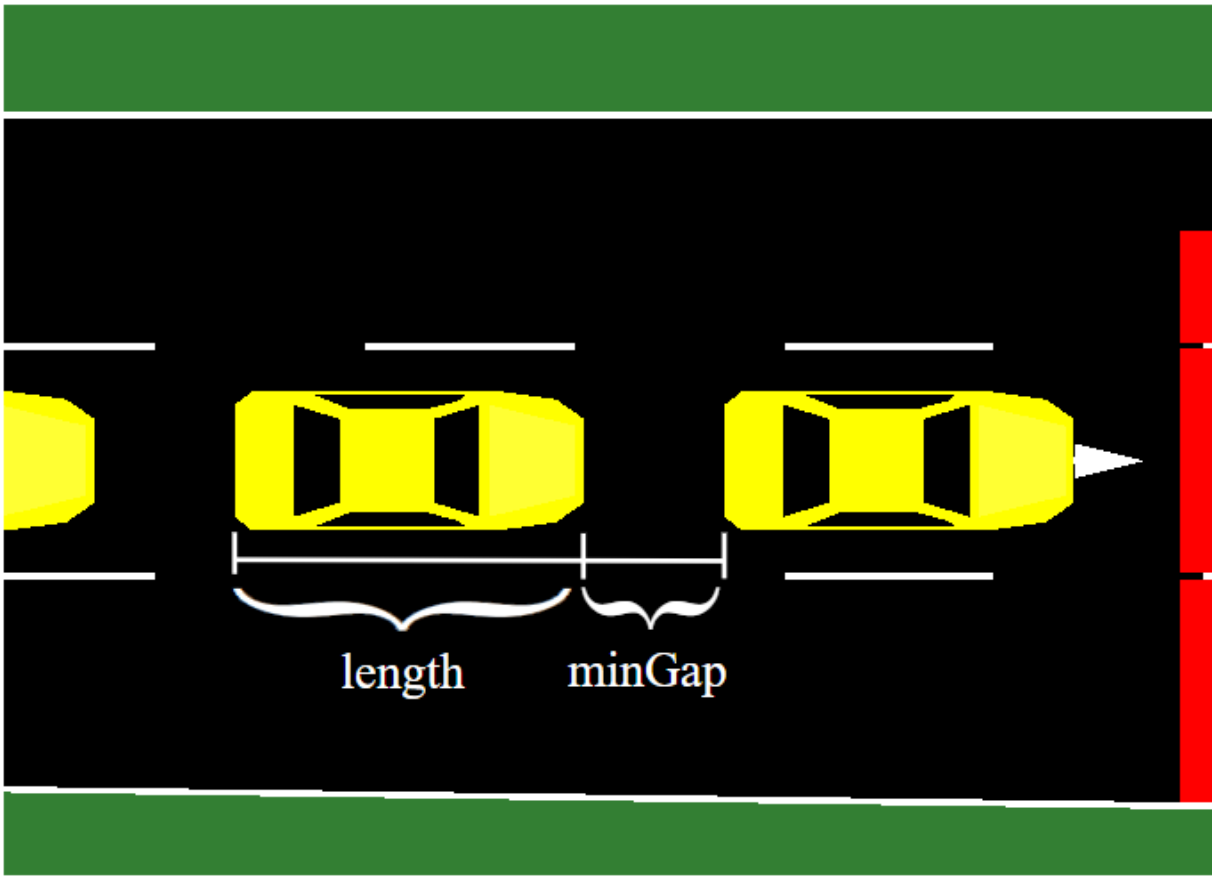
3.5 Σχεδίαση δικτύου και προσομοίωση

Το πρώτο βήμα για την υλοποίηση της προσομοίωσης είναι η σχεδίαση του δικτύου του αυτοκινητόδρομου. Όπως προαναφέρθηκε χρησιμοποιούμε το πρόγραμμα Netedit και δημιουργούμε κάτι απλό. Δημιουργούμε τέσσερις κόμβους στη σειρά οι οποίοι ενώνονται μεταξύ τους και φτιάχνουν τρία τμήματα. Το πρώτο τμήμα αποτελεί το κομμάτι εισαγωγής των οχημάτων στον αυτοκινητόδρομο, μήκους 600 μέτρων. Το δεύτερο τμήμα μήκους 9400 μέτρων είναι ουσιαστικά το κύριο μέρος του αυτοκινητόδρομου και το τρίτο και τελευταίο, αποτελεί την έξοδο μήκους 200 μέτρων, όπου τα οχήματα ουσιαστικά φτάνουν στον προορισμό τους, τερματίζεται η διαδρομή τους και απομακρύνονται από το δίκτυο . Ο αυτοκινητόδρομος του σεναρίου που θα προσομοιώσουμε έχει συνολικό μήκος 10 χιλιόμετρα και 200 μέτρα συμπεριλαμβανομένων των κομματιών εισαγωγής και εξαγωγής

από αυτόν και αποτελείται από πέντε λωρίδες. Αυτές τις παραμέτρους, και διάφορες άλλες όπως για παράδειγμα το μέγιστο επιτρεπτό όριο ταχύτητας μπορούμε να τις ρυθμίσουμε εμείς με την εντολή inspect ανάλογα με τις ανάγκες του σεναρίου.

Το κομμάτι της σχεδίασης κάπου εδώ τελειώνει καθώς έχουμε ένα πολύ απλό σύστημα. Στη συνέχεια για να τρέξει η προσομοίωση στο Sumo θα πρέπει να δημιουργήσουμε ένα επιπλέον αρχείο με κατάληξη *.rou όπου δηλώνουμε τα είδη των οχημάτων τα οποία θα τρέξουν στον αυτοκινητόδρομο μας καθώς και τις ροές τους. Μέσα σε αυτό έχουμε τη δυνατότητα να ορίσουμε μία πληθώρα χαρακτηριστικών που αφορούν στα οχήματα που θα εισέλθουν στην προσομοίωση όπως για παράδειγμα:

- την ικανότητα επιτάχυνσης και επιβράδυνσης
- τη μέγιστη δυνατή φυσική επιβράδυνση σε έκτακτη ανάγκη
- το μήκος του οχήματος
- την απόσταση σε μέτρα ενός οχήματος από το προπορευόμενο
- τη μέγιστη ταχύτητα που μπορεί να αναπτύξει
- το κατά πόσο θα μπορεί να παραβιάζει το όριο ταχύτητας του αυτοκινητόδρομου
- τις εκπομπές καυσαερίων
- το πλάτος του το ύψος του
- την ελάχιστη απόσταση από το προπορευόμενο όχημα που πρέπει να τηρεί για να αποφύγει μία πιθανή σύγκρουση
- το μοντέλο με τον οποίο γίνονται οι αλλαγές λωρίδας το οποίο είναι από προεπιλογή το πρότυπο του Krauss
- τον αριθμό των ανθρώπων που μπορούν να επιβιβαστούν



Σχήμα 16: Μήκος οχήματος και ελάχιστη απόσταση από προπορευόμενο όχημα

Όσον αφορά τώρα στα χαρακτηριστικά των ροών των οχημάτων μπορούμε να ορίσουμε:

- τη χρονική στιγμή την οποία θα αρχίσουν οι αφίξεις των συγκεκριμένων οχημάτων στον αυτοκινητόδρομο
- τη διάρκεια εισαγωγής της ροής
- το σημείο εισαγωγής στο δίκτυο (λωρίδα άφιξης)
- τη θέση από την οποία θα εξέλθουν από αυτόν ή αλλιώς τον επιθυμητό τελικό στόχο
- την ταχύτητα άφιξης και εισόδου
- τον αριθμό οχημάτων ανά ώρα τα οποία θα τρέχουν στην προσομοίωση

Ακολουθεί μία περιγραφή της επιλογής κάποιων παραμέτρων και διαφόρων παραδοχών που έγιναν για να στηθεί το μοντέλο μας . Η ροή μας αποτελείται από τέσσερα είδη οχημάτων. Τα συμβατικά επιβατικά οχήματα τα οποία μπορούν να αναπτύξουν τελική ταχύτητα 33 m/s ή 118,8 km/h, τα πιο γρήγορα οχήματα τα οποία μπορούν να αναπτύξουν τελική ταχύτητα 40 m/s ή 144 km/h, μεγαλύτερά και πιο δυσκίνητα οχήματα τύπου λεωφορεία που αναπτύσσουν μικρότερες ταχύτητες έως 27 m/s ή 97,2 km/h και μία ακόμα ροή κανονικών επιβατικών αυτοκινήτων με τελική ταχύτητα 33 m/s ή 118,8 km/h τα οποία στη συνέχεια θα τα αποκαλούμε “tester”. Οι τρεις πρώτες ροές θα αποτελούνται από μεγάλο αριθμό οχημάτων και ουσιαστικά θα δημιουργούν την κυκλοφορία και η τέταρτη θα αποτελείται από μικρό αριθμό και θα εξετάζεται η συμπεριφορά της. Τα μισά περίπου “tester” αποτελούν τα οχήματα στα οποία θα εφαρμόσουμε διάφορους κανόνες ελέγχου οδικής συμπεριφοράς και θα συγκριθούν με τα υπόλοιπα μισά που δρουν χωρίς παρέμβαση. Στην ροή των tester ορίζουμε μία λωρίδα την οποία θέλουμε να τελειώσουν την διαδρομή τους, τον επιθυμητό τελικό στόχο δηλαδή, και ορίζουμε ακόμα τρία σημεία διέλευσης, δημιουργώντας έτσι ένα σενάριο κυλιόμενου ορίζοντα. Αυτά τα σημεία διέλευσης ή way points βρίσκονται στο 3ο και στο 6,5ο χιλιόμετρο σε διάφορες λωρίδες που θα ορίζουμε. Ο αριθμός οχημάτων της ροής των tester είναι 17 , σε 6 από τα οποία θα τα ελέγχουμε εμείς μέσω του TraCI και θα τους δίνουμε εντολές για να αλλάζουμε την θέση τους στον αυτοκινητόδρομο δηλαδή τη λωρίδα τους. Θα αναφερόμαστε στα 6 αυτά ελεγχόμενα ως tester οχήματα και στα υπόλοιπα 11 ως normal ή default ή κανονικών. Κανόνας με τον οποίο θα ορίζουμε την αλλαγή λωρίδας είναι αυτός που έχει προκύψει από το δυναμικό προγραμματισμό και τον στοχαστικό αναδρομικό τύπου του Bellman.

*autobahn.rou - Σημειωματάριο

Αρχείο Επεξεργασία Μορφή Προβολή Βοήθεια

```
<routes>

    <vType id="normal_car" vClass="passenger" maxSpeed="33" speedFactor="1"  />
    <vType id="sporty_car" vClass="passenger" maxSpeed="40" speedFactor="1" />
    <vType id="tester" vClass="passenger" maxSpeed="33" speedFactor="1"  />
    <vType id="coach" vClass="coach"  maxSpeed="27" speedFactor="1."  />


    <flow id="normal" type="normal_car" begin="0" end="1400" number="2460" from="entry" to="exit"
        departPos="last" departLane="best" departSpeed="max" />
    <flow id="sporty" type="sporty_car" begin="0" end="1400" number="700" from="entry" to="exit"
        departPos="last" departLane="best" departSpeed="max" />
    <flow id="tester" type="tester" begin="0" end="1400" number="17" from="entry" to="exit"
        departPos="last" departLane="best" departSpeed="max" arrivalLane="4" tau="0.5" >
                <stop lane="longedge_2" startPos="3000" endPos="3010" speed="25" />
                <stop lane="longedge_3" startPos="6500" endPos="6510" speed="25" />
    </flow>
    <flow id="coach" type="coach" begin="0" end="1400" number="330" from="entry" to="exit"
        departPos="last" departLane="best" departSpeed="max" />
```

Σχήμα 17: Αρχείο *.rou με τις πληροφορίες για τους τύπους και τις ροές οχημάτων και τα χαρακτηριστικά τους.

Ουσιαστικά θέλουμε να συγκρίνουμε τη συμπεριφορά των οχημάτων ίδιων χαρακτηριστικών καθώς και την πορεία τους όταν υπάρχει επιθυμητός στόχος για τους κανόνες ελέγχου πού αναπτύξαμε προηγουμένως. Στο σενάριο όμως δεν θα έχουμε έναν τελικό επιθυμητό στόχο στο τέλος του αυτοκινητόδρομου αλλά θα λειτουργήσουμε με την ύπαρξη ενός κυλιόμενου ορίζοντα όπου θα μεταβάλλεται ο στόχος των οχημάτων κατά τη διάρκεια της προσομοίωσης. Στο σενάριο μας ο στόχος διέλευσης θα μεταβληθεί τρεις φορές μέσα στην προσομοίωση.

Για την εξαγωγή δεδομένων από την επιθυμητή ροή οχημάτων την οποία θα ελέγξουμε, το Sumo μας δίνει τη δυνατότητα να προσκολλήσουμε πάνω στα οχήματά τις λεγόμενες συσκευές οι οποίες μπορούν να εξάγουν με λεπτομέρεια δεδομένα διαφόρων χαρακτηριστικών. Έτσι μειώνεται ο υπολογιστικός φόρτος σε ένα βαθμό και τα εξαγόμενα αρχεία που παράγονται είναι σημαντικά μικρότερα περιέχοντας τις πληροφορίες μόνο των επιθυμητών οχημάτων και όχι όλων όσων θα τρέξουν στο δίκτυο.

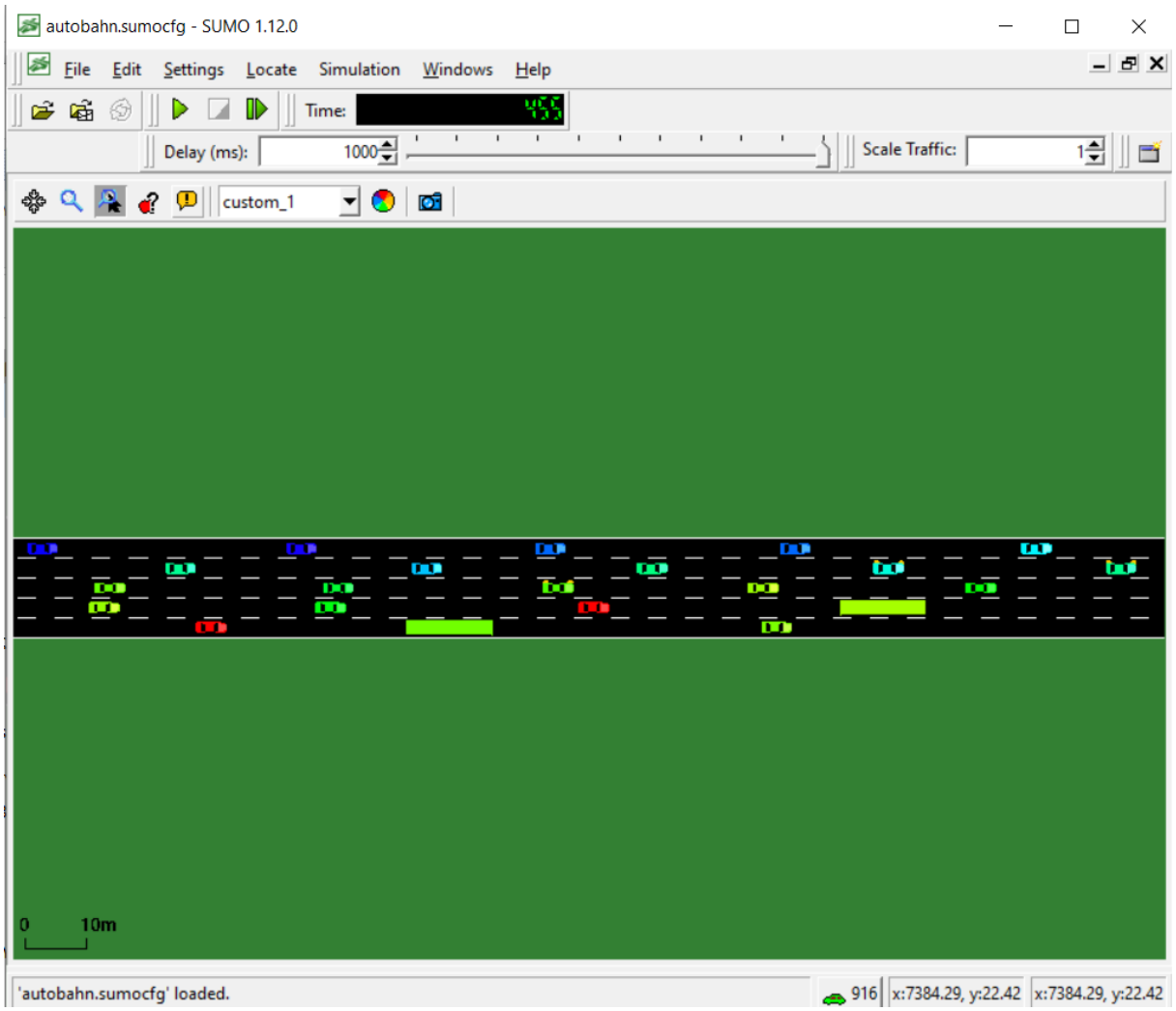
Στο τέλος της προσομοίωσης δημιουργούνται αρχεία κειμένου με στοιχεία των οχημάτων που κινούνται μέσα σε αυτή και περιέχουν πληροφορίες σχετικά με τη λωρίδα που βρίσκεται το κάθε όχημα σε κάθε timestep προσομοίωσης, διάφορες στατιστικές πληροφορίες για το μέσο όρο του χρόνου των οχημάτων που κινήθηκαν στην προσομοίωση, στοιχεία σχετικά με την ταχύτητα και την κάθε θέση που πέρασαν τα οχήματα του αυτοκινητοδρόμου και άλλα.

Εμείς μέσω αυτών των αρχείων και με τη βοήθεια της Python της βιβλιοθήκης Matplotlib μπορούμε να απομονώσουμε τα στοιχεία τα οποία μας ενδιαφέρουν στην περίπτωσή μας, της ροής των tester των πειραματικών αλλά και των κανονικών οχημάτων και να επεξεργαστούμε τις πληροφορίες τους και να συγκρίνουμε τη συμπεριφορά τους. Αυτά που θα επικεντρωθούμε θα είναι η αλλαγή λωρίδων που θα πραγματοποιήσουν κατά τη διάρκεια του ταξιδιού τους, ο χρόνος τον οποίο έκαναν για να ολοκληρώσουν τη διαδρομή καθώς και η μέση ταχύτητα τους.

Τα σενάρια τα οποία θα τρέξουμε την προσομοίωση μας θα διαφέρουν ως προς το χρόνο ελέγχου με τον οποίο θα επιβάλλουμε εμείς στα πειραματικά οχήματα να αλλάζουν τη λωρίδα τους ο οποίος θα κυμανθεί από 10 έως 90 δευτερόλεπτα. Επίσης θα υπάρξει διαφοροποίηση ως προς τον κανόνα ελέγχου που εφαρμόζεται όπου σχετίζεται με τις τιμές της στοχαστικής μεταβλητής p1 και p2 που είχαμε αναφέρει στο προηγούμενο κεφάλαιο και ως προς την κυκλοφοριακή συμφόρηση που θα επικρατεί στον αυτοκινητόδρομο. Για την

κυκλοφοριακή συμφόρηση θα χρησιμοποιηθεί ένα σενάριο χαμηλής κίνησης και ένα σενάριο υψηλής. Η υψηλή κίνηση θα αποτελείται περίπου από 2.200 οχήματα ανά λωρίδα ανά ώρα ενώ στη χαμηλή θα υπάρχουνε 1.800 οχήματα ανά λωρίδα ανά ώρα. Τα αποτελέσματα θα οπτικοποιηθούν σε διάφορα διαγράμματα και πίνακες για την καλύτερη κατανόηση και επεξεργασία τους.

Η εκτέλεση της προσομοίωσης αλλά και η σύνδεση με το TraCI για η on-line παρεμβολή στην τρεχούμενη προσομοίωση για έλεγχο των οχημάτων έγινε με τη βοήθεια κώδικα[2] σε περιβάλλον Python.



Σχήμα 18: Προσομοίωση κυκλοφορίας στο περιβάλλον Sumo.

3.6 Σύνοψη Κεφαλαίου

Στο παρόν κεφάλαιο γίνεται μία παρουσίαση των εργαλείων που χρησιμοποιούνται για το σχεδιασμό του δικτύου και την εκτέλεση της προσομοίωσης στο πρόγραμμα Eclipse SUMO. Ορίζονται τέσσερις διαφορετικές ροές οχημάτων που διαφέρουν ως προς το μέγεθος, την τελική ταχύτητα, τον αριθμό και τον τύπο των οχημάτων. Αναλύονται τα χαρακτηριστικά της προσομοίωσης που αφορούν την κυκλοφοριακή συμφόρηση, τον χρόνο ελέγχου και τα σενάρια κυλιόμενου ορίζοντα επιθυμητού στόχου των οχημάτων και γίνεται μία περιγραφή των παραμέτρων της προσομοίωσης.

Στο επόμενο κεφάλαιο ακολουθεί η παρουσίαση των αποτελεσμάτων της προσομοίωσης και ο σχολιασμός τους.

4 ΑΠΟΤΕΛΕΣΜΑΤΑ ΚΑΙ ΔΙΑΓΡΑΜΜΑΤΑ

Στο κεφάλαιο αυτό παρουσιάζουμε αριθμητικά αποτελέσματα διάφορων σεναρίων που προσομοιώσαμε. Τα σενάρια διαφέρουν ως προς το χρόνο ελέγχου (control time), τον αριθμό οχημάτων ανά λωρίδα ανά ώρα (μποτιλιάρισμα), τον κανόνα ελέγχου που προκύπτει από την τιμή των πιθανοτήτων της στοχαστικής μεταβλητής και το εκάστοτε σενάριο κυλιόμενου ορίζοντα, τα σημεία δηλαδή μέσα στον αυτοκινητόδρομο από τα οποία καθορίζουμε να διέλθει το όχημα.

4.1 Σενάρια Προσομοίωσης

Στο σύνολο έχουμε 48 σενάρια καθώς προσομοιώνουμε για 4 διαφορετικούς χρόνους ελέγχου, 2 είδη κίνησης στο δρόμο, 2 κανόνες ελέγχου (στοχαστική μεταβλητή) και 3 σενάρια διέλευσης σημείων (κυλιόμενος ορίζοντας) .

Ο χρόνος ελέγχου, το χρονικό διάστημα δηλαδή ανά το οποίο θα εφαρμόζουμε τον κανόνα ελέγχου παίρνει τιμές 10, 30, 60, 90 δευτερόλεπτα. Όσο αυξάνεται η τιμή, ασκούμε σπανιότερα τον έλεγχο και άρα η συμπεριφορά των οχημάτων τείνει στη φυσιολογική. Στο σενάριο μέτριας κίνησης έχουμε περίπου 1800 οχήματα ανά λωρίδα ανά ώρα ενώ ο αυτοκινητόδρομος γεμίζει στα 2100 οχήματα ανά λωρίδα ανά ώρα στο δεύτερο σενάριο. Εφαρμόζονται οι δύο διαφορετικοί κανόνες ελέγχου που αναπτύχθηκαν στο 2^{ο} κεφάλαιο και τα τρία σενάρια διέλευσης σημείων κυλιόμενου ορίζοντα όπως φαίνονται στον παρακάτω πίνακα. Στο SUMO ισχύει ο κανόνας οδηγείται δεξιά, οπότε αναμένεται κίνηση στις πρώτες ή δεξιά λωρίδες να οδηγούν σε μικρότερες ταχύτητες.

Control time (s)	veh/lane/hour	p1 στοχαστικής μεταβλητής	Σενάρια διέλευσης σημείων	**σύνολο σεναρίων**
10	2122	>=0,75	2-3-4	
30			0-3-1	
60	1803	<=0,75	4-1-3	
90				
4	2	2	3	**48**

Πίνακας 6: Σενάρια Προσομοίωσης.

4.2 Πίνακες και διαγράμματα

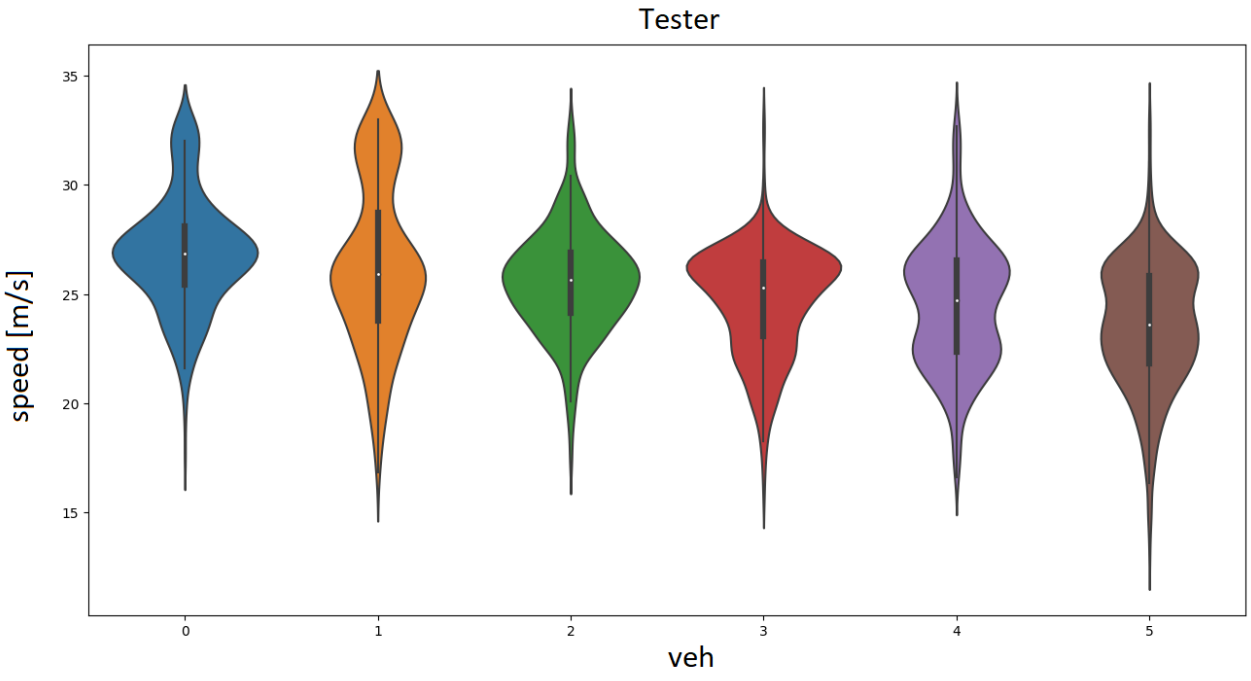
Παρακάτω ακολουθεί σύγκριση των αποτελεσμάτων 5 οχημάτων tester στα οποία εφαρμόσαμε τον κανόνα ελέγχου και 6 κανονικών οχημάτων με τα ίδια χαρακτηριστικά που κινούνταν με τους προεπιλεγμένους κανόνες του προσομοιωτή παραθέτοντας δύο τύπους διαγραμμάτων. Στο ένα απεικονίζεται η κατανομή και η μέση τιμή των ταχυτήτων του κάθε οχήματος και στο δεύτερο η τροχιά τους, η θέση τους δηλαδή στον y άξονα (λωρίδα αυτοκινητόδρομου) μέσα στο χρόνο προσομοίωσης.

Σημειώνουμε ότι η αντιστοιχία λωρίδας θέσης στον y άξονα είναι η εξής

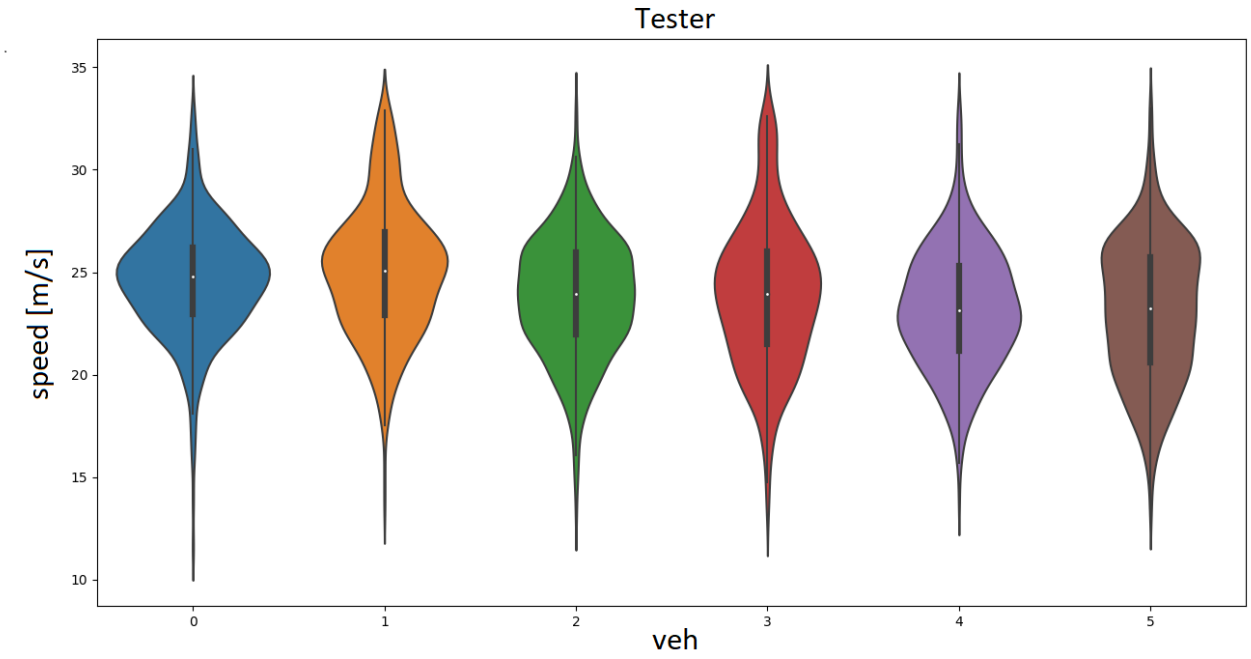
Λωρίδα	**Θέση στον y άξονα**
4	13,4
3	10,2
2	7
1	3,8
0	0,6

Πίνακας 7: Αντιστοιχία λωρίδας αυτοκινητόδρομου με συντεταγμένη y στα διαγράμματα πορείας.

Παρακάτω ακολουθούν ενδεικτικά μερικά διαγράμματα.



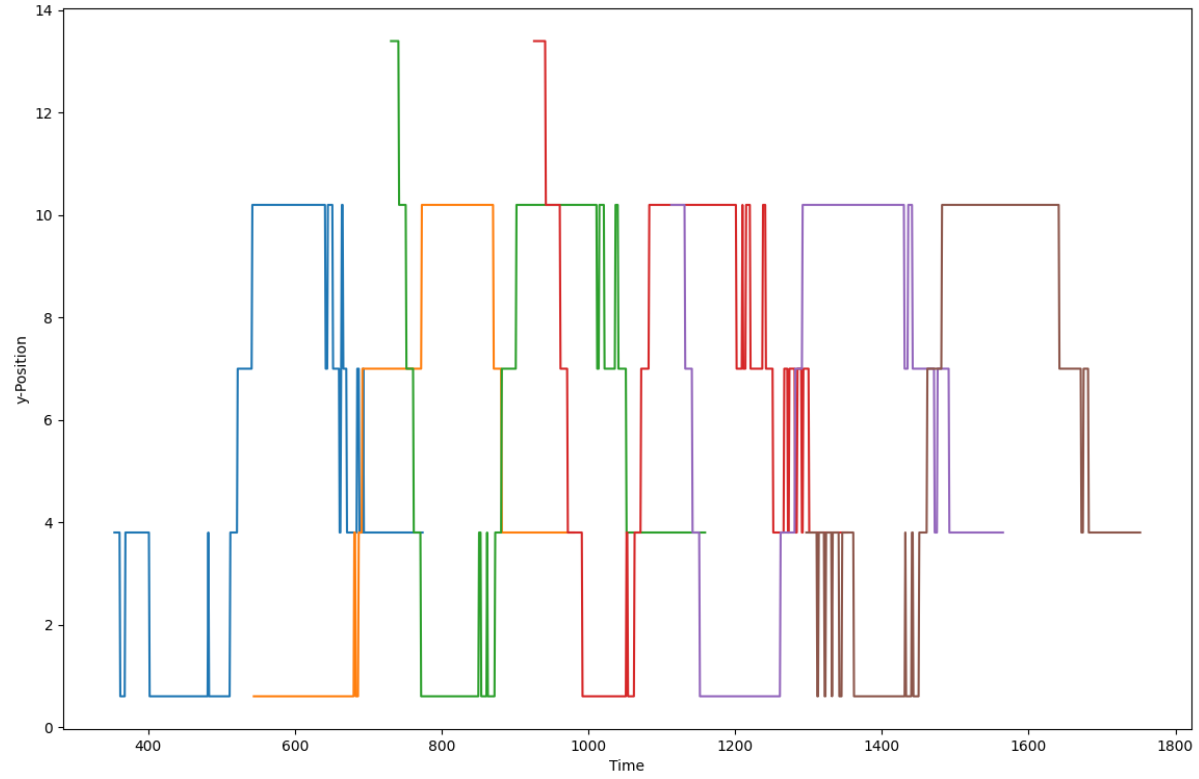
α)



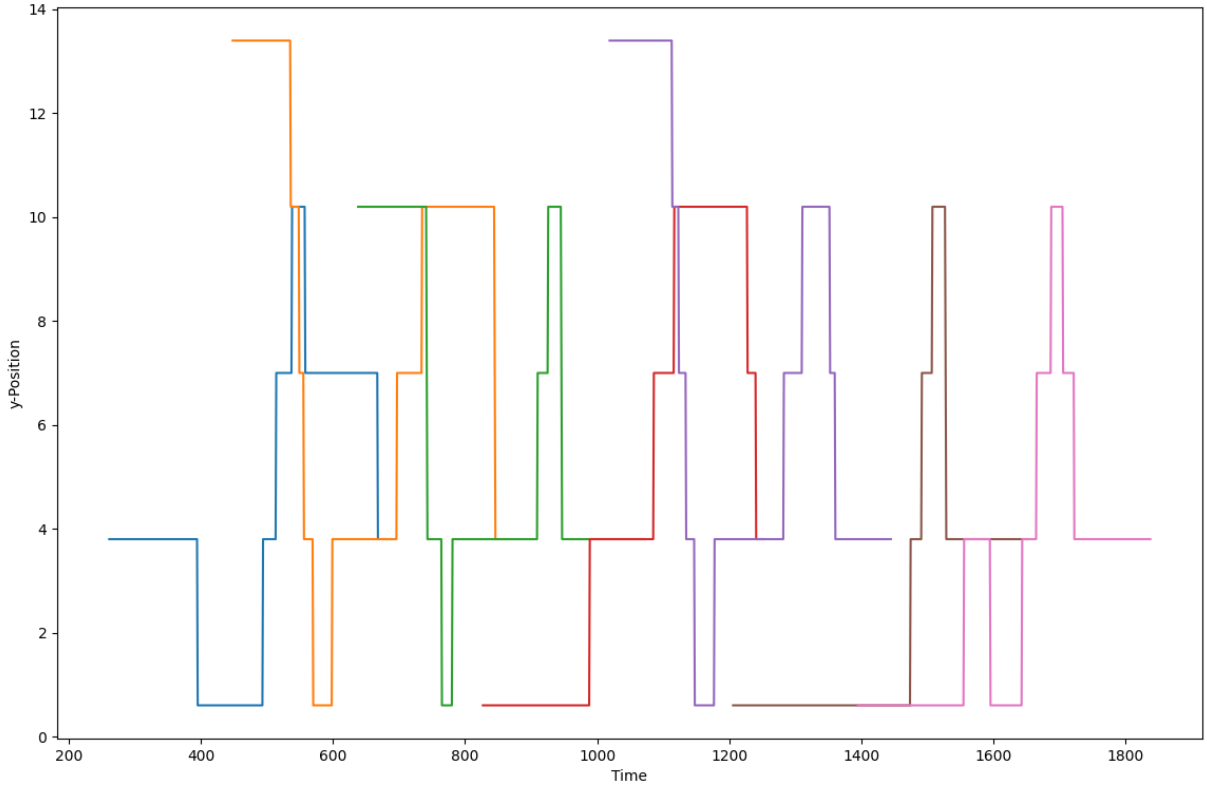
β)

Σχήμα 19: Κατανομή ταχυτήτων για τα οχήματα ελέγχου για p1=0,9, 2100 veh/lane/hr, χρόνος ελέγχου 30 δευτερόλεπτα: α) target lanes 4-1-3, β) target lanes 0-3-1

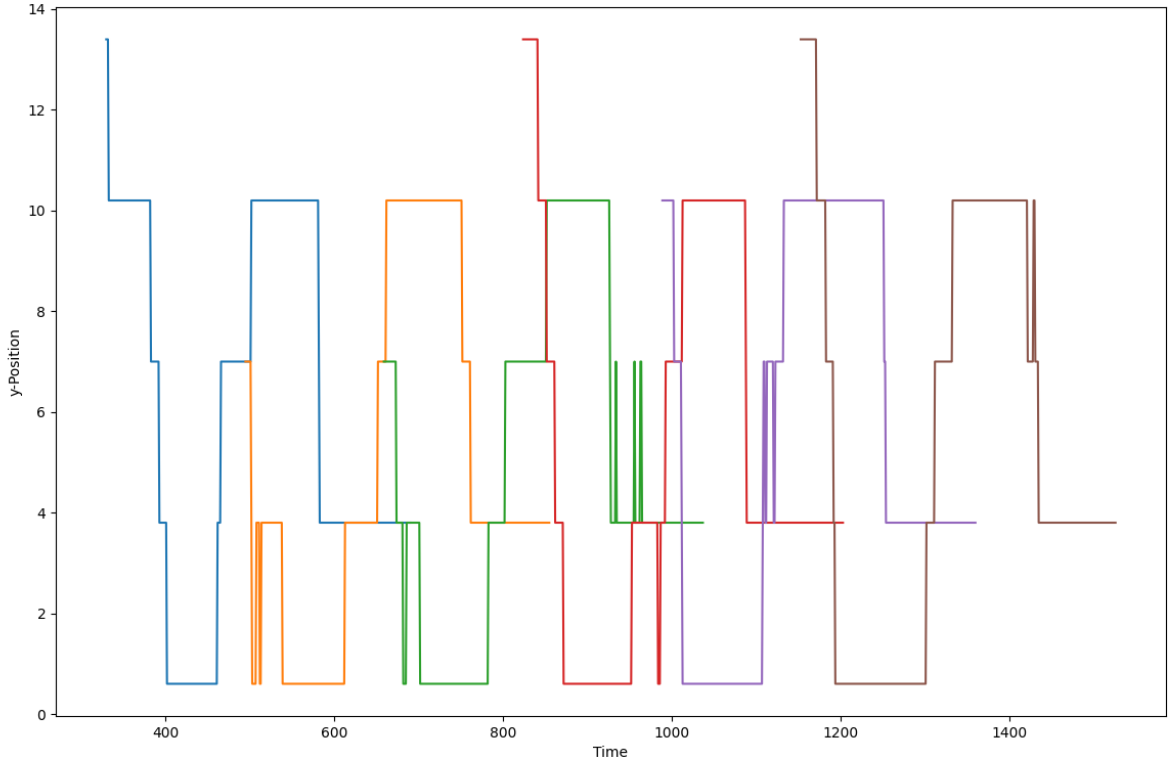
Σε πυκνή κυκλοφορία και για έναν μέσο χρόνο ελέγχου 30 δευτερολέπτων το σενάριο κυλιόμενου ορίζοντα στις γρήγορες λωρίδες έδωσε μεγαλύτερη μέση ταχύτητα των ελεγχόμενων οχημάτων (90,45 km/h) απ' ότι στο σενάριο με τις αργές λωρίδες (85,89 km/h). Το αποτέλεσμα αυτό μπορεί να θεωρηθεί αναμενόμενο καθώς ο κανόνας οδηγείται δεξιά και προσπερνάτε από αριστερά έχει ως αποτέλεσμα την κίνηση οχημάτων στις δεξιά λωρίδες με χαμηλότερες ταχύτητες. Επίσης στο πρώτο σενάριο οι τιμές των ταχυτήτων έχουν μεγαλύτερο εύρος και κάποια οχήματα κινούνται σε αρκετά υψηλές ταχύτητες.


y-Position
Time
0
2
4
6
8
10
12
14
400
600
800
1000
1200
1400
1600
1800

α)


y-Position
Time
0
2
4
6
8
10
12
14
200
400
600
800
1000
1200
1400
1600
1800

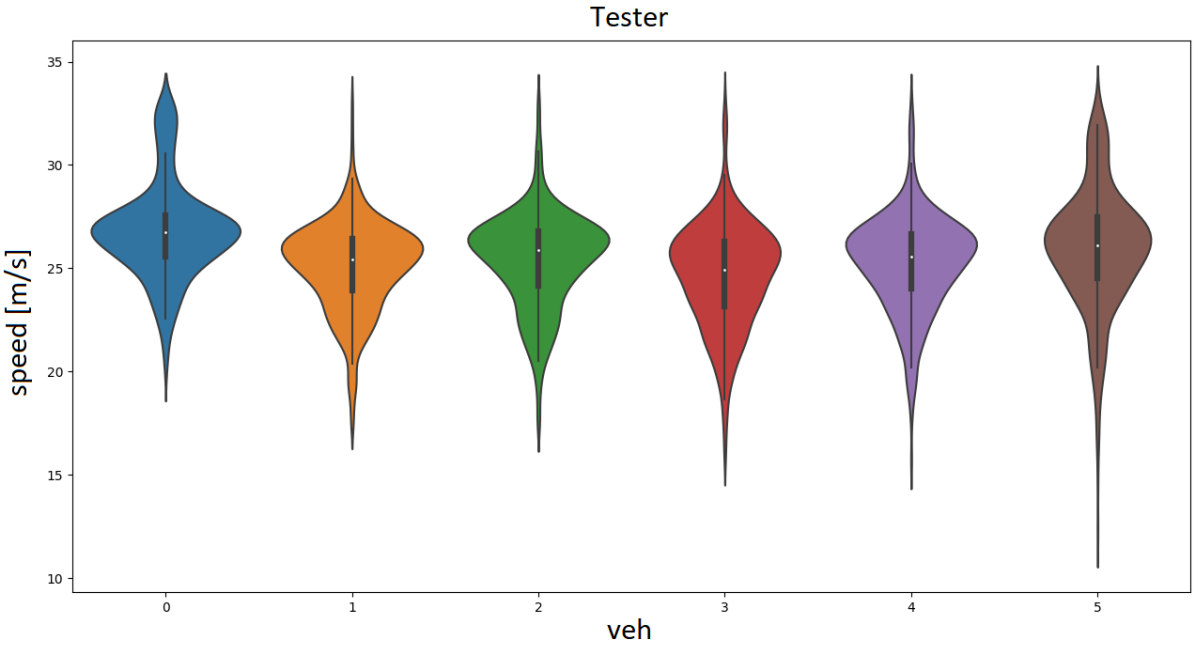
β)



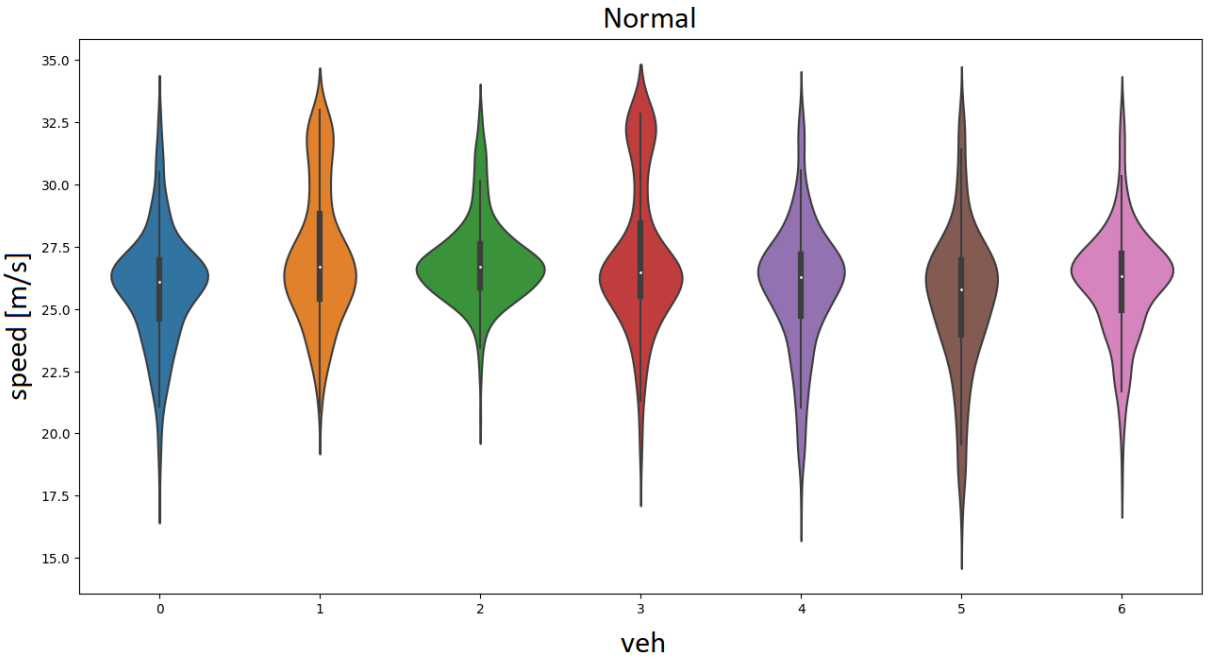
γ)

Σχήμα 20: Διαγράμματα τροχιάς οχημάτων σε σχέση με το χρόνο όπου αποτυπώνονται οι αλλαγές λωρίδας για p1=0,9, σημεία διέλευσης 0-3-1, χρόνο ελέγχου 10 δευτερολέπτων και: α) 2100 veh/lane/hour για οχήματα tester, β) 2100 veh/lane/hour για οχήματα default, γ) 1800 veh/lane/hour για οχήματα tester

Κινούμενοι στις λωρίδες 0, 1 και 3 και για μικρό χρόνο ελέγχου 10 δευτερολέπτων ο μέσος αριθμός αλλαγής λωρίδας των tester είναι μεγαλύτερος στο σενάριο πυκνής κυκλοφορίας. Στο σενάριο των 2100 veh/lane/hour είναι 16,83 (Σχήμα 19.α) ενώ στων 1800 veh/lane/hour είναι 11,50 (Σχήμα 19.γ). Το γεγονός αυτό οφείλεται στο ότι τα οχήματα σε πυκνή κυκλοφορία προσπαθούν να κάνουν περισσότερες προσπεράσεις και μετακινούνται αριστερά προς τις γρηγορότερες λωρίδες αλλά ο κανόνας ελέγχου τα επαναφέρει στις αργές λωρίδες ώστε να περάσουν από τα σημεία διέλευσης. Και στις δύο περιπτώσεις και γενικά στα περισσότερα σενάρια τα tester πραγματοποιούν περισσότερες αλλαγές λωρίδας απ' ότι τα οχήματα που δεν ασκείται έλεγχος. Σημειώνεται ότι στο σενάριο του Σχήμα 19.α παρατηρήθηκε ο μεγαλύτερος αριθμός εναλλαγών λωρίδας και ο μεγαλύτερος χρόνος ταξιδίου. Στην περίπτωσή μας τα κανονικά οχήματα πραγματοποίησαν κατά μέσο όρο 7 αλλαγές λωρίδας (Σχήμα 19.γ)



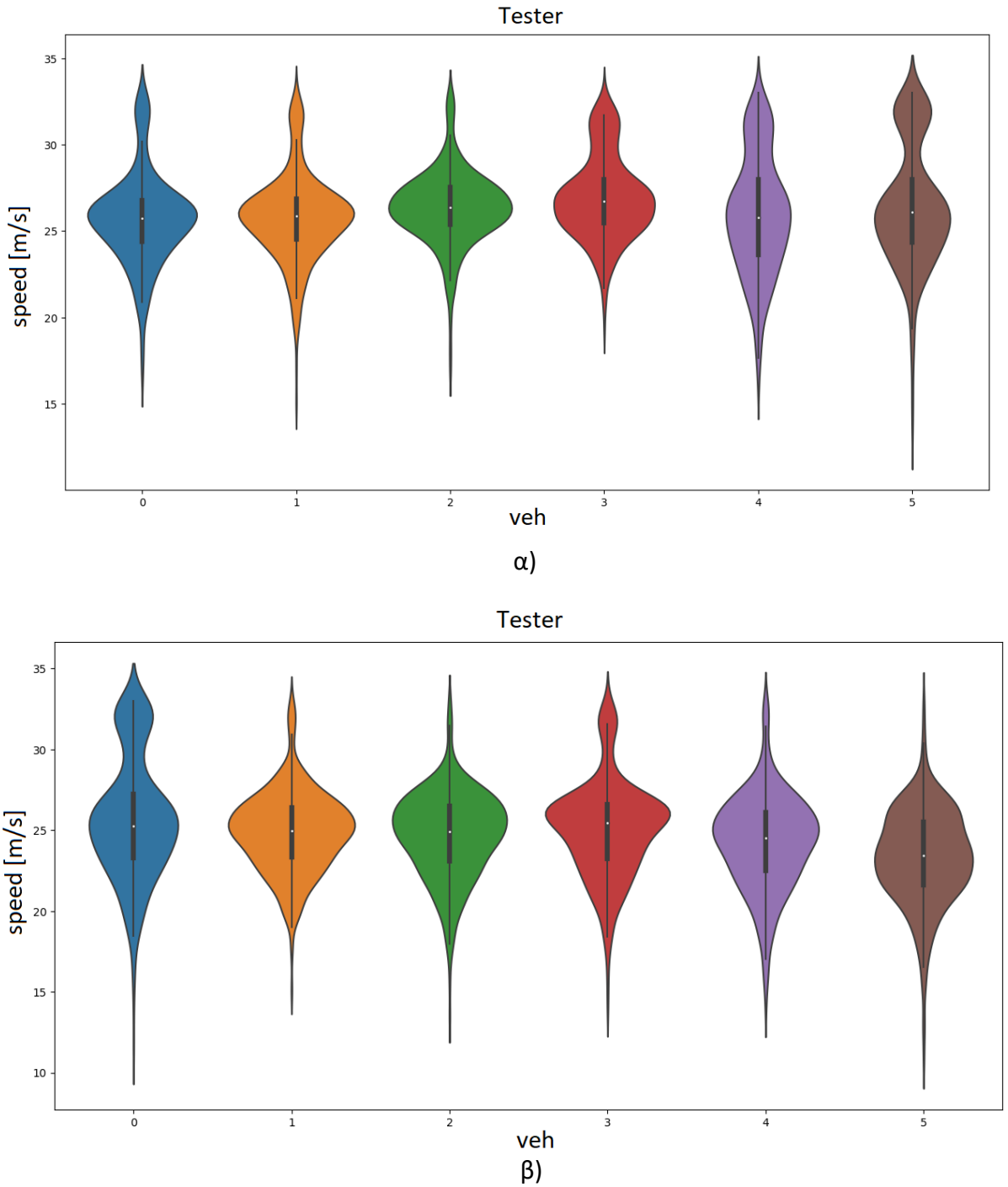
α)



β)

Σχήμα 21: Κατανομή ταχυτήτων για p1=0,7, target lanes 4-1-3, 2100 veh/lane/hr, χρόνος ελέγχου 60 δευτερόλεπτα: α) Οχήματα ελέγχου tester, β) Οχήματα Normal

Άξιο παρατήρησης είναι το σενάριο κυλιόμενου ορίζοντα των λωρίδων 4-1-3, για χρόνο ελέγχου 60 δευτερολέπτων και p1=0,7 όπου τα οχήματα ελέγχου ανέπτυξαν σημαντικά μικρότερες ταχύτητες από τα κανονικά. Ο μέσος αριθμός αλλαγής λωρίδας ήταν σχεδόν ίδιος κοντά στο 7.



Σχήμα 22: Κατανομή ταχυτήτων για τα οχήματα ελέγχου για p1=0,7, target lanes 2-3-4 χρόνος ελέγχου 10 δευτερόλεπτα: α) 2100 veh/lane/hour, β) 1800 veh/lane/hour

Στο σενάριο σημείων διέλευσης 2-3-4, για χρόνο ελέγχου 10 δευτερολέπτων και για p1=0,7 φαίνεται πόσο σημαντικό ρόλο παίζει η κυκλοφοριακή συμφόρηση στην ανάπτυξη ταχύτητας. Στο σενάριο πυκνής συμφόρησης τα οχήματα ταξιδεύουν με αισθητά πιο χαμηλές ταχύτητες ενώ καταγράφουν σχεδόν διπλάσιες αλλαγές λωρίδας (5,5 έναντι 3).

4.3 Αριθμητικά αποτελέσματα

Παρακάτω ακολουθούν συγκεντρωτικοί πίνακες των αποτελεσμάτων των διάφορων σεναρίων. Θεωρούμε έναν κανόνα ελέγχου επιτυχή με κριτήριο το αν οδήγησε σε μικρότερο μέσο χρόνο ταξιδίου, μεγαλύτερη μέση ταχύτητα στα οχήματα που εφαρμόζεται, αν πέρασε επιτυχώς από τα σημεία διέλευσης και αν οδήγησε σε μικρό αριθμό αλλαγών λωρίδας. Ο χρόνος και η ταχύτητα ταξιδίου συνδέονται άμεσα.

veh/lane/hour	control time (s)	num. lane changes		travel time		speed (km/h)	
		tester	default	tester	default	tester	default
2100	10	10,16	**7,00**	399	**389**	91,58	**93,93**
	30	8,50	**7,57**	404	**400**	90,45	**91,35**
	60	7,83	**7,42**	401	**389**	91,12	**93,93**
	90	7,50	**7,00**	402	**397**	90,90	**92,04**
1800	10	**6,66**	7,71	**360**	364	**96,50**	95,44
	30	**5,83**	6,71	366	**362**	94,92	**95,97**
	60	7,00	**6,85**	**362**	363	**95,97**	95,70
	90	8,00	**7,00**	**367**	**367**	**94,66**	**94,66**

α)

veh/lane/hour	control time (s)	num. lane changes		travel time		speed (km/h)	
		tester	default	tester	default	tester	default
2100	10	5,50	**3,42**	**412**	415	**88,69**	88,05
	30	4,66	**3,71**	**413**	417	**88,47**	87,63
	60	4,50	**4,14**	402	**401**	90,90	**91,12**
	90	**3,66**	3,85	397	**392**	92,04	**93,21**
1800	10	**3**	3,57	**369**	**369**	**94,15**	**94,15**
	30	**3.33**	4.14	370	**367**	93,89	**94,66**
	60	**3,16**	4,00	**363**	368	**95,70**	94,40
	90	3,50	**3,42**	**366**	368	**94,92**	94,40

β)

veh/lane/hour	control time (s)	num. lane changes		travel time		speed (km/h)	
		tester	default	tester	default	tester	default
2100	10	16,83	**7**	437	**431**	83,62	**84,78**
	30	10,66	**7,85**	425	**423**	85,98	**86,38**
	60	8.66	**7,57**	420	**417**	87,00	**87,63**
	90	9,5	**8,28**	420	**416**	87,00	**87,84**
1800	10	11,50	**7.86**	**371**	372	**93,64**	93,39
	30	10,83	**8,57**	371	**369**	93,64	**94,15**
	60	9,00	**7,42**	**365**	369	**95,18**	94,15
	90	8,16	**7,14**	**370**	372	**93,89**	93,39

γ)

Πίνακας 8: Συγκεντρωτικός πίνακας αποτελεσμάτων για πιθανότητα στοχαστικής μεταβλητής p1=0,9 και p2=0,05 για τα τρία σενάρια κυλιόμενου ορίζοντα: α) 4-1-3, β) 2-3-4, γ) 0-3-1

veh/lane/hour	control time (s)	num. lane changes		travel time		speed (km/h)	
		tester	default	tester	default	tester	default
2100	10	10,33	**7,86**	397	**387**	92,04	**94,42**
	30	**7,83**	8,14	401	**394**	91,12	**92,74**
	60	7,66	**7,57**	398	**386**	91,81	**94,66**
	90	7,50	**7,00**	402	**397**	90,90	**92,04**
1800	10	**5,66**	7.28	**363**	364	**95,70**	95,44
	30	**6,00**	6,85	365	**364**	95,18	**95,44**
	60	**7,33**	8,00	367	**366**	94,66	**94,92**
	90	8,00	**7,00**	368	**367**	94,40	**94,66**

α)

veh/lane/hour	control time (s)	num. lane changes		travel time		speed (km/h)	
		tester	default	tester	default	tester	default
2100	10	5,5	**3,42**	**412**	415	**88,69**	88,05
	30	4,66	**3.71**	**413**	417	**88,47**	87,63
	60	4,50	**4.14**	402	**401**	90,90	**91,12**
	90	**3,66**	3,85	397	**392**	92,04	**93,21**
1800	10	**3**	3,57	**369**	**369**	**94,15**	**94,15**
	30	**3,33**	4,14	370	**367**	93,89	**94,66**
	60	**3,16**	4,00	**363**	368	**95,70**	94,40
	90	3,50	**3,42**	**366**	368	**94,92**	94,40

β)

veh/lane/hour	control time (s)	num. lane changes		travel time		speed (km/h)	
		tester	default	tester	default	tester	default
2100	10	13	**7,42**	427	**425**	85,57	**85,98**
	30	10,83	**7**	423	**418**	86,38	**87,42**
	60	9,83	**8**	420	**412**	87,00	**88,69**
	90	9,5	**8,28**	420	**416**	87,00	**87,84**
1800	10	11,16	**8.28**	**372**	373	**93,10**	92,85
	30	10,16	**6,57**	**369**	373	**93,85**	92,85
	60	9,00	**7,42**	**365**	369	**94,88**	93,85
	90	7,83	**7,14**	**368**	370	**94,11**	93,60

γ)

Πίνακας 9: Συγκεντρωτικός πίνακας αποτελεσμάτων για πιθανότητα στοχαστικής μεταβλητής p1=0,7 και p2=0,2 για τα τρία σενάρια κυλιόμενου ορίζοντα: α) 4-1-3, β) 2-3-4, γ) 0-3-1

α)

σενάριο κυλιόμενου ορίζοντα	p1=0.9>0,75		p1=0.7<0,75	
	4_1_3			
	veh/lane/hr	control step	veh/lane/hr	control step
	2100	10	2100	10
		30		30
		60		60
		90		90
	1800	**10**	1800	**10**
		30		30
		60		60
		90		90

β)

σενάριο κυλιόμενου ορίζοντα	p1=0.9>0,75		p1=0.7<0,75	
	2_3_4			
	veh/lane/hr	control step	veh/lane/hr	control step
	2100	**10**	2100	**10**
		30		**30**
		60		60
		90		90
	1800	**10**	1800	**10**
		30		30
		60		**60**
		90		**90**

γ)

σενάριο κυλιόμενου ορίζοντα	p1=0.9>0,75		p1=0.7<0,75	
	0_3_1			
	veh/lane/hr	control step	veh/lane/hr	control step
	2100	10	2100	10
		30		30
		60		60
		90		90
	1800	**10**	1800	**10**
		30		**30**
		60		**60**
		90		**90**

Πίνακας 10: Συγκεντρωτικά αποτελέσματα των διαφόρων σεναρίων. Με έντονη γραφή και υπογράμμιση τα σενάρια στα οποία ο κανόνας ελέγχου έδωσε μικρότερους χρόνους ταξιδίου και μεγαλύτερες μέσες ταχύτητες . Σενάρια κυλιόμενου ορίζοντα: α) 4-1-3, β) 2-3-4, γ) 0-3-1.

Από τους παραπάνω συγκεντρωτικούς πίνακες εξάγονται τα εξής αποτελέσματα:

- Ο χρόνος ελέγχου των 90 δευτερολέπτων οδηγεί σε συμπεριφορά σχεδόν ίδια με τα κανονικά οχήματα. Το γεγονός αυτό θα μπορούσαμε να πούμε ότι είναι αναμενόμενο δεδομένου της απόστασης μεταξύ των σημείων διέλευσής που βρίσκονται ανά 3 περίπου χιλιόμετρα. Σε χρόνο 90 δευτερολέπτων και με μία μέση ταχύτητα από 23 m/s έως 25 m/s στο εκάστοτε όχημα επιβάλλεται έλεγχος το πολύ μία φορά.
- Όλα τα οχήματα κατάφεραν να περάσουν από τα σημεία διέλευσης επιτυχώς.
- Στα περισσότερα σενάρια τα tester πραγματοποιούν περισσότερες αλλαγές λωρίδας απ’ ότι τα οχήματα που δεν ασκείται έλεγχος.
- Στο σενάριο κυλιόμενου ορίζοντα 2-3-4 μόνο ο κανόνας ελέγχου για τη λωρίδα 3 διαφέρει για τις δύο πιθανότητες της στοχαστικής μεταβλητής οπότε οδηγούμαστε σε παρόμοια αποτελέσματα.
- Κινούμενοι στις αργές λωρίδες οι δύο κανόνες ελέγχου εφαρμόζονται με επιτυχία μόνο για το σενάριο χαμηλής κίνησης για όλους σχεδόν τους χρόνους ελέγχου.
- Κινούμενοι στις γρήγορες λωρίδες πιο αποδοτικός χρόνος ελέγχου είναι τα 10 και 30 sec.
- Από όλους τους χρόνους ελέγχου αυτός των 10 δευτερολέπτων εφαρμόζεται πιο αποδοτικά στα περισσότερα σενάρια.
- Για κίνηση σε πυκνή κυκλοφορία οι κανόνες ελέγχου οδηγούν πάντα σε περισσότερες εναλλαγές λωρίδας.
- Σε πυκνή κυκλοφορία επιτυχής έλεγχος παρατηρείται σε λιγότερα σενάρια απ’ ότι σε χαμηλή συμφόρηση.
- Ο χρόνος ελέγχου των 10 sec εφαρμόζεται με επιτυχία σε όλα τα σενάρια χαμηλής κίνησης.
- Κινούμενοι στις αργές λωρίδες, στο σενάριο πυκνής κυκλοφορίας κανένας κανόνας ελέγχου δεν εφαρμόζεται επιτυχώς.
- Στο σενάριο των στόχων διέλευσης 2-3-4, καθώς οι επιθυμητές λωρίδες ήταν διαδοχικές τα οχήματα δεν χρειάστηκε να κάνουν μεγάλες μετακινήσεις και ο έλεγχος εφαρμόστηκε πιο αποδοτικά σε σχέση με τα άλλα δύο.

- Οι δύο κανόνες ελέγχου εφαρμόζονται επιτυχώς σχεδόν στα ίδια σενάρια όσον αφορά στο μέσο χρόνο ταξιδίου και ταχύτητας, αλλά όχι στον μέσο αριθμό αλλαγής λωρίδας.

4.4 Σύνοψη Κεφαλαίου

Στο παρόν κεφάλαιο έγινε μία ανάλυση των παραμέτρων των διάφορων σεναρίων που προσομοιώθηκαν. Διαφορετικοί χρόνοι ελέγχου, σενάρια κυλιόμενου ορίζοντα, κανόνες ελέγχου και τύποι κυκλοφοριακής συμφόρησης οδήγησαν σε μία πληθώρα αποτελεσμάτων.

Παρουσιάστηκαν τα αριθμητικά αποτελέσματα, απεικονίστηκαν σε πίνακες και διαγράμματα και σχολιάστηκαν

Τα πιο βασικά συμπεράσματα ακολουθούν στο επόμενο κεφάλαιο.

5 ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΜΕΛΛΟΝΤΙΚΕΣ ΕΡΕΥΝΕΣ

Η παρούσα εργασία αναφέρεται στην επίλυση του Προβλήματος Στοχαστικού Βέλτιστου Ελέγχου Διακριτού Χρόνου με την χρήση του αναδρομικού τύπου του Bellman και στον προσδιορισμό ενός κανόνα ελέγχου, για την αλλαγή λωρίδας ενός οχήματος που κινείται σε έναν αυτοκινητόδρομο πέντε λωρίδων. Στόχος είναι να φτάσει στο όχημα σε έναν επιθυμητό τελικό στόχο, περνώντας και από ενδιάμεσα επιθυμητά σημεία σε πυκνή κυκλοφορία επιτυγχάνοντας μέγιστη ταχύτητα.

Από τα σενάρια που αναλύθηκαν προέκυψαν τα παρακάτω συμπεράσματα:

- Ο έλεγχος μπορεί να εφαρμοστεί επιτυχώς για ένα όχημα κινούμενο στις αργές λωρίδες μόνο σε μέτρια κυκλοφοριακή συμφόρηση.
- Πολύ συχνός έλεγχος οδηγεί σε μεγάλο αριθμό αλλαγής λωρίδας με αποτέλεσμα να επηρεάζεται και ο χρόνος ταξιδίου.
- Κινούμενοι στις αργές λωρίδες, στο σενάριο πυκνής κυκλοφορίας κανένας κανόνας ελέγχου δεν εφαρμόζεται επιτυχώς.
- Ο χρόνος ελέγχου των 10 sec εφαρμόζεται με επιτυχία σε όλα τα σενάρια μέτριας κίνησης.
- Μεγάλοι χρόνοι ελέγχου επηρεάζουν ελάχιστα τη συμπεριφορά του οχήματος.
- Οι δύο κανόνες ελέγχου οδηγούν σε παρόμοια αποτελέσματα.
- Σε μέτρια κυκλοφοριακή συμφόρηση ο χρόνος ελέγχου των 30 δευτερολέπτων πρέπει να αποφεύγεται, ενώ σε πυκνή κυκλοφορία λιγότερο αποδοτικός είναι αυτός των 60.
- Τα αποτελέσματα δείχνουν ότι ο έλεγχος αποδίδει αρκετά περισσότερο στην μέτρια κυκλοφοριακή συμφόρηση.
- Στις γρήγορες λωρίδες, για το ίδιο διάστημα ελέγχου έχουμε λιγότερες εναλλαγές λωρίδας απ’ ότι στις αργές καθώς στις τελευταίες λόγω χαμηλής ταχύτητας αυξάνεται η συχνότητα των προσπεράσεων.
- Στο σενάριο που οι επιθυμητές λωρίδες στον κυλιόμενο ορίζοντα είναι διαδοχικές και δεν απαιτούνται μεγάλες πλάγιες μετακινήσεις ο έλεγχος εφαρμόζεται πιο αποδοτικά.

Η διπλωματική αυτή ανοίγει το δρόμο για την μελέτη της αναλυτικής λύσης του ντετερμινιστικού προβλήματος και την επαλήθευση του μη μεταβλητού κανόνα ελέγχου. Όσον αφορά το στοχαστικό πρόβλημα που μοντελοποιήθηκε , η αναζήτηση της αναλυτικής λύσης αποτελεί μία πρόκληση. Στο στοχαστικό πρόβλημα της εργασίας αυτής η πιθανολογική κατανομή εξαρτάται και από την μεταβλητή κατάστασης x(k) καθιστώντας την αναλυτική λύση πιο πολύπλοκη. Πρώτα θα πρέπει να γίνει κατάλληλη διατύπωση των πιθανολογικών κατανομών με πιο απλοϊκό τρόπο για την επίλυση της αναδρομικής εξίσωσης Bellman στη στοχαστική μορφή.

Περιθώρια βελτίωσης υπάρχουν επίσης στην πιθανολογική κατανομή ώστε να προσεγγιστούν πιο ρεαλιστικά σενάρια. Η επέκταση του ελέγχου στην αυξομείωση της ταχύτητας (επιτάχυνση/επιβράδυνση) ταυτόχρονα με την αλλαγή λωρίδας ενός οχήματος μπορεί να αποτελέσει το επόμενο βήμα στην εξέλιξη των στρατηγικών ελέγχου οδικής κυκλοφορίας.

Βιβλιογραφία

[1] Bellman, R.E. (1957) **Dynamic Programming**. Princeton University Press, Princeton.

[2] Papageorgiou, M. (2009) Dynamic Programming. Crete.

[3] Larson, R.E., Casti, J.L. (1978) Principles of Dynamic Programming - Part I: Basic Analytic and Computational Methods. Dekker, New York.

[4] https://thalis.math.upatras.gr/~bountis/files/dynamical_systems.pdf

[5] Ασημάκης, Ν., & Αδάμ, Μ. (2015). *Συστήματα διακριτού και συνεχούς χρόνου*

[6] https://www.webpages.uidaho.edu/niatt_labmanual/chapters/capacityandlos/theoryandconcepts/freeflowspeedandflowrate.htm

[7] https://github.com/lexfridman/deeptraffic

[8] Operations Research - An Introduction, 9th Edition Hamdy A. Taha. University of Arkansas, Fayetteville.

[9] http://www.samos.aegean.gr/math/dimitheo/DYNAMIC_PROGRAMMING_SHMEIOSEIS.pdf

Αλγόριθμος Δυναμικού Προγραμματισμού για το Στοχαστικό ΠΒΕ διακριτού χρόνου σε Python

```
import math
import numpy as np

# Basic Inputs

sp=1                # starting position
xd=3                # x_desired
t=100               # epithimitos xronos
numl=5              # number of lanes
p1=0.8              # probability to execute the decision
p2=0.15             # probability to take another decision

#---------------------------------------------------------------------

# Arxikh sunthiki

V=np.zeros([t+1,numl+2])   # V(k,x)  t+1 xronikes katastaseis 0,1,2,...t
k=3
for i in range(0,numl,1):
    V[k][i]=pow(i-xd,2)


# DISTRIBUTION OF Z

p=np.zeros([3,3,numl])       # p(z,j,i)/p(z,u,x)


for i in range(1,numl-1):    # arithmos lwridwn=numl, oi lwrides=0,1,..numl-1
    p[0][2][i]=1-p1          # oi akrianes lwrides einai oi 0 kai numl-1
    p[1][2][i]=p1            # oi endiameses einai oi  1,2,..numl-2
    p[2][2][i]=0

    p[0][1][i]=p2
    p[1][1][i]=p1
    p[2][1][i]=1-p1-p2

    p[0][0][i]=0
    p[1][0][i]=p1
```

```
    p[2][0][i]=1-p1

p[0][2][numl-1]=0           # sthn panw akrianh lwrida den mporei u=1 (J=2)
p[1][2][numl-1]=0
p[2][2][numl-1]=0

p[0][1][numl-1]=1-p1
p[1][1][numl-1]=p1
p[2][1][numl-1]=0

p[0][0][numl-1]=0
p[1][0][numl-1]=p1
p[2][0][numl-1]=1-p1

p[0][2][0]=1-p1             # sthn katw akrianh lwrida den mporei u=-1 (J=0)
p[1][2][0]=p1
p[2][2][0]=0

p[0][1][0]=0
p[1][1][0]=p1
p[2][1][0]=1-p1

p[0][0][0]=0
p[1][0][0]=0
p[2][0][0]=0


#---------------------------------------------------------------------


# MAIN LOOP

directions_list = []            # edw apothikeyw veltistes apofaseis i
for k in range(t-1,-1,-1):
    for i in range(0,numl):
        temp_v = []
        for j in [-1,0,1]:
            if i+j >=0 and i+j <=numl-1:
                F = (pow(i-xd,2)+pow(j,2))    # ϕ[x(k),u(k),z(k),k]
                J=j+1                         # apofash

                temp_v.append((F+p[2][J][i]*V[k+1][i+j+1]+p[1][J][i]*V[k+1][i+
j+0]+p[0][J][i]*V[k+1][i+j-1],j))

               # ϕ[x(k),u(k),z(k),k] + E{V[x(k+1),k+1]}


            else:
                temp_v.append((math.inf,j))
```

```
        min_v = min(temp_v, key = lambda t: t[0])[0]    # Veltistes apofaseis
        directions_list.append((k,i,[item[1] for item in temp_v if min_v ==
item[0]]))

        V[k][i] = min_v


#---------------------------------------------------------------------


# Veltiwsh Emfanishs

V= np.delete(V, numl+1, 1)
V= np.delete(V, numl, 1)
V= np.delete(V, t, 0)

for m in range(0,numl,1):
    if m != sp:
        V[0][m]=0
    if m not in (sp+1,sp,sp-1):
        V[1][m]=0
    if m not in (sp+2,sp+1,sp,sp-1,sp-2):
        V[2][m]=0
    if m not in (sp+3,sp+2,sp+1,sp,sp-1,sp-2,sp-3):
        V[3][m]=0



#-----------------------------------------------------------


#APOTELESMATA

print(V)         # kosth
print("*")


#print(directions_list)

for i in range(numl*(t-1),0,-numl):     #pinakas me veltistes apofaseis

    print(directions_list[i][2],directions_list[i+1][2],directions_list[i+2][2
],directions_list[i+3][2],directions_list[i+4][2])
```

Αλγόριθμος διασύνδεσης TraCI με προσομοίωση SUMO σε Python

```
import os
import sys
import traci

# Global settings
sumoBinary = os.path.join('sumo-gui')
sumoCmd = [sumoBinary, "-c",os.path.join('autobahn.sumocfg')]

# we need to import python modules from the $SUMO_HOME/tools directory
def set_env():
    if 'SUMO_HOME' in os.environ:
        tools = os.path.join(os.environ['SUMO_HOME'], 'tools')
        sys.path.append(tools)
    else:
        sys.exit("please declare environment variable 'SUMO_HOME'")

#-------FUNCTIONS------------------------------------------------------------


simstep=1
controltime=10              # control time in seconds
tc=controltime/simstep      # time control in simulation steps

# def target lane=4
def lch4(x,step):

    lst=traci.vehicle.getIDList()      # list of vehs IDs in the highway
    if lst.count(x)>0:
            if (step%tc)==0:
                nol4=traci.vehicle.getLaneIndex(x)
                if nol4==0 or nol4==1 or nol4==2 or nol4==3:
                    traci.vehicle.changeLane(x, nol4+1, 1)
                else:
                    traci.vehicle.changeLane(x, nol4, 1)

# def target lane=3 p=0,9 [1,1,1,0,-1]
def lch309(x,step):

    lst=traci.vehicle.getIDList()      # list of vehs IDs in the highway
    if lst.count(x)>0:
        if (step%tc)==0:
            nol4=traci.vehicle.getLaneIndex(x)
```

```
            if nol4==0 or nol4==1 or nol4==2:
                traci.vehicle.changeLane(x, nol4+1, 1)
            elif nol4==4:
                traci.vehicle.changeLane(x, nol4-1, 1)
            else:
                traci.vehicle.changeLane(x, nol4, 1)


# def target lane=3 p=0,7 [1,1,1,0,0]
def lch307(x,step):

    lst=traci.vehicle.getIDList()      # list of vehs IDs in the highway
    if lst.count(x)>0:
        if (step%tc)==0:
            nol5=traci.vehicle.getLaneIndex(x)
            if nol5==0 or nol5==1 or nol5==2:
                traci.vehicle.changeLane(x, nol5+1, 1)
            else:
                traci.vehicle.changeLane(x, nol5, 1)

# def target lane=2 [1,1,0,-1,-1]
def lch2(x,step):

    lst=traci.vehicle.getIDList()      # list of vehs IDs in the highway
    if lst.count(x)>0:
        if (step%tc)==0:
            nol5=traci.vehicle.getLaneIndex(x)
            if nol5==0 or nol5==1:
                traci.vehicle.changeLane(x, nol5+1, 1)
            elif nol5==3 or nol5==4:
                traci.vehicle.changeLane(x, nol5-1, 1)
            else:
                traci.vehicle.changeLane(x, nol5, 1)

# def target lane=1 p=0,9 [1,0,-1,-1,-1]
def lch109(x,step):

    lst=traci.vehicle.getIDList()      # list of vehs IDs in the highway
    if lst.count(x)>0:
        if (step%tc)==0:
            nol5=traci.vehicle.getLaneIndex(x)
            if nol5==2 or nol5==3 or nol5==4:
                traci.vehicle.changeLane(x, nol5-1, 1)
            elif nol5==0:
                traci.vehicle.changeLane(x, nol5+1, 1)
            else:
                traci.vehicle.changeLane(x, nol5, 1)

# def target lane=1 p=0,7 [0,0,-1,-1,-1]
```

```
def lch107(x,step):

    lst=traci.vehicle.getIDList()      # list of vehs IDs in the highway
    if lst.count(x)>0:
        if (step%tc)==0:
            nol5=traci.vehicle.getLaneIndex(x)
            if nol5==2 or nol5==3 or nol5==4:
                traci.vehicle.changeLane(x, nol5-1, 1)
            else:
                traci.vehicle.changeLane(x, nol5, 1)

# def target lane=0 [0,-1,-1,-1,-1]
def lch0(x,step):

    lst=traci.vehicle.getIDList()      # list of vehs IDs in the highway
    if lst.count(x)>0:
        if (step%tc)==0:
            nol5=traci.vehicle.getLaneIndex(x)
            if nol5==1 or nol5==2 or nol5==3 or nol5==4:
                traci.vehicle.changeLane(x, nol5-1, 1)
            else:
                traci.vehicle.changeLane(x, nol5, 1)


# def kuliomenos orizontas
def ko(x,step):
    lst=traci.vehicle.getIDList()      # list of vehs IDs in the highway
    if lst.count(x)>0:
        dst=traci.vehicle.getDistance(x)
        if dst <= 3000:
            lch2(x,step)
        elif dst < 6500 and dst > 3000:
            lch307(x,step)
        else:
            lch4(x,step)


def run():
    traci.start(sumoCmd)

    step = 0
    while step < 4000/simstep:
        traci.simulationStep()

        ko("tester.4",step)          # Elegxos epithimitwn oxhmatwn gia kinhsh
        ko("tester.6",step)          # Eme ton kathorismeno kuliomeno orizonta
        ko("tester.8",step)
        ko("tester.10",step)
        ko("tester.12",step)
```

```
        ko("tester.14",step)



        step += 1
        numberloaded=traci.vehicle.getIDCount()
        if numberloaded==0 :
            traci.close()

if __name__=="__main__":
    set_env()
    run()

    sys.exit(1)
```