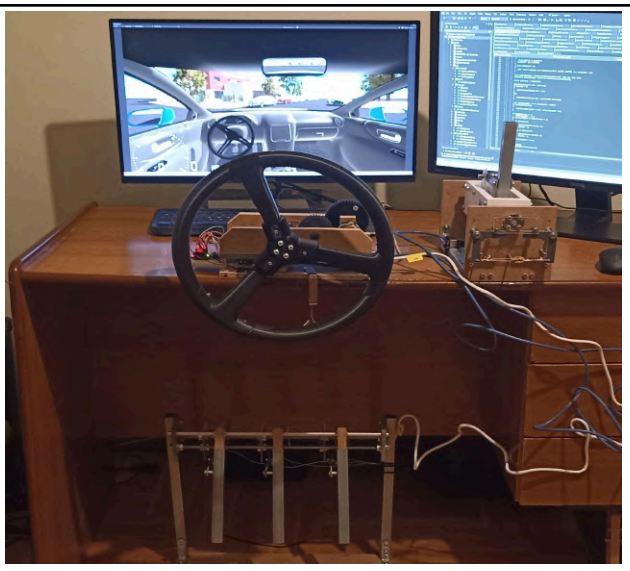
ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σχεδίαση και υλοποίηση συστήματος προσομοίωσης οδήγησης με τη χρήση ARDUINO



Του φοιτητή
Πουταχίδη Γεώργιου
Αρ. Μητρώου: 517320

Επιβλέπουσα
Παπαδοπούλου Μαρία
Επίκουρος Καθηγητής

Θεσσαλονίκη, Φεβρουάριος 2026

Τίτλος Δ.Ε. Σχεδίαση και υλοποίηση συστήματος προσομοίωσης οδήγησης με τη χρήση ARDUINO
Κωδικός Δ.Ε. 22306
Ονοματεπώνυμο φοιτητή Πουταχίδης Γεώργιος
Ονοματεπώνυμο εισηγητή Παπαδοπούλου Μαρία
Ημερομηνία ανάληψης Δ.Ε. 30-10-2022
Ημερομηνία περάτωσης Δ.Ε. 19-02-2026

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Περίληψη

Η συμβολή των προσομοιωτών οδήγησης στην εκπαίδευση των νέων οδηγών, αλλά και στην εξέλιξη των δυναμικών χαρακτηριστικών με τα οποία εφοδιάζονται τα σύγχρονα αυτοκίνητα, καθίσταται όλο και πιο σημαντική τα τελευταία χρόνια. Ο προσομοιωτής μπορεί να αποτελέσει ένα σημαντικό εργαλείο, τόσο στον τομέα της εκπαίδευσης όσο και της έρευνας, ενώ παράλληλα, μπορεί να συμβάλλει στη συνεχόμενη αναβάθμιση της οδηγικής ικανότητας των πολιτών, με στόχο την οδική ασφάλεια.

Σκοπός της παρούσας εργασίας είναι να σχεδιαστεί και κατασκευαστεί ένας προσομοιωτής οχημάτων, για την εκπαίδευση και αξιολόγηση υποψηφίων οδηγών, σε ένα ασφαλές, διαδραστικό περιβάλλον, με χαμηλό κόστος. Ο προσομοιωτής περιλαμβάνει φυσικό τιμόνι και πεντάλ πάνω στα οποία τοποθετήθηκαν αισθητήρες, οι τιμές των οποίων διαβάζονται από έναν μικροελεγκτή Arduino Leonardo και στη συνέχεια αποστέλλονται μέσω USB θύρας σε έναν υπολογιστή. Το πρόγραμμα της προσομοίωσης, είναι φτιαγμένο με τη μηχανή γραφικών Unity3D. Αναλυτικότερα, ο χρήστης θα διδάσκεται μία οδηγική δεξιότητα, μέσω animation, και στη συνέχεια θα καλείται να υλοποιήσει το εν λόγω σενάριο με τη χρήση του προσομοιωτή. Αφού ολοκληρώσει την άσκηση θα βαθμολογείται, ενώ παράλληλα θα μπορεί να παρακολουθεί την εξέλιξη της πορείας του. Η εργασία περιλαμβάνει τη σχεδίαση των φυσικών τμημάτων του προσομοιωτή, τη σχεδίαση των ηλεκτρονικών και τον προγραμματισμό του μικροελεγκτή, και τέλος την ανάπτυξη του λογισμικού του προσομοιωτή.

«Design and implementation of a driving simulation system using ARDUINO»

George Poutachidis

Περιεχόμενα

Κεφάλαιο 1. Τροχαία ατυχήματα και οι επιπτώσεις τους

Τα τροχαία ατυχήματα αποτελούν ένα από τα πιο σοβαρά προβλήματα δημόσιας υγείας και κοινωνικής ασφάλειας παγκοσμίως. Κάθε χρόνο, εκατομμύρια άνθρωποι χάνουν τη ζωή τους ή τραυματίζονται σε τροχαία, με συνέπειες που ξεπερνούν τα ατομικά όρια και επηρεάζουν τις οικογένειες, τα συστήματα υγείας και τις εθνικές οικονομίες. Η πολυπλοκότητα του φαινομένου οφείλεται σε πλήθος παραγόντων, όπως η ανθρώπινη συμπεριφορά, οι οδικές υποδομές και η κατάσταση των οχημάτων. Η μελέτη των επιπτώσεων των τροχαίων ατυχημάτων είναι κρίσιμη, καθώς δεν αφορά μόνο τις άμεσες απώλειες ζωών, αλλά και τις μακροχρόνιες συνέπειες σε κοινωνικό και οικονομικό επίπεδο.

1.1 Στατιστικα τροχαίων ατυχημάτων παγκοσμίως

Τα τροχαία ατυχήματα αποτελούν μια από τις σημαντικότερες αιτίες θανάτου και αναπηρίας σε παγκόσμιο επίπεδο. Σύμφωνα με τον Παγκόσμιο Οργανισμό Υγείας (ΠΟΥ), κάθε χρόνο περισσότεροι από 1,19 εκατομμύρια άνθρωποι χάνουν τη ζωή τους λόγω τροχαίων δυστυχημάτων, ενώ επιπλέον 20 έως 50 εκατομμύρια τραυματίζονται, πολλοί εκ των οποίων υφίστανται μακροχρόνιες ή μόνιμες αναπηρίες [1]. Η κλίμακα του προβλήματος καθιστά τα τροχαία ατυχήματα μία από τις κύριες αιτίες απωλειών υγείας, ιδίως στις νεότερες ηλικιακές ομάδες. Ενδεικτικά, τα στοιχεία του ΠΟΥ δείχνουν ότι τα τροχαία ατυχήματα αποτελούν την πρώτη αιτία θανάτου για παιδιά και νέους 5-29 ετών [2]. Η σοβαρότητα των περιστατικών είναι εξίσου ανησυχητική. Οι μη θανατηφόροι τραυματισμοί δεν περιορίζονται σε μικρές κακώσεις, αλλά συχνά οδηγούν σε μόνιμες αναπηρίες, παρατεταμένες νοσηλείες και μακροχρόνια απώλεια λειτουργικότητας. Οι συνέπειες αυτές μετρώνται σε χαμένα έτη ζωής με αναπηρία (DALYs), δείκτης που αποτυπώνει την πραγματική βαρύτητα του προβλήματος για τη δημόσια υγεία [3].

Οι οικονομικές επιπτώσεις είναι επίσης ιδιαίτερα βαριές. Ο ΠΟΥ εκτιμά ότι τα τροχαία ατυχήματα κοστίζουν σε πολλές χώρες περίπου 3% του Ακαθάριστου Εγχώριου Προϊόντος (ΑΕΠ) ετησίως . Οι απώλειες αυτές προέρχονται τόσο από τις άμεσες δαπάνες υγειονομικής περίθαλψης, όσο και από την απώλεια παραγωγικότητας των θυμάτων και των οικογενειών τους [1].

Η γενικότερη τάση δείχνει ότι, χωρίς την εφαρμογή στοχευμένων μέτρων, οι θάνατοι και οι τραυματισμοί από τροχαία αναμένεται να αυξηθούν σημαντικά στο μέλλον. Ήδη από το 2004, από την Παγκόσμια Έκθεση για την Πρόληψη των Τροχαίων Ατυχημάτων (WHO & World Bank), είχε εκτιμηθεί ότι οι θάνατοι από τροχαία θα μπορούσαν να αυξηθούν κατά περίπου 65% μέσα σε διάστημα 20 ετών, από την περίοδο 2000-2020, και σε χαμηλού και μεσαίου εισοδήματος χώρες, αυτός ο δείκτης αναμένεται να φτάσει το 80% [3]. Αν και οι Στόχοι Βιώσιμης Ανάπτυξης (SDG 3.6) θέτουν ως στόχο τη μείωση των τροχαίων θανάτων και τραυματισμών κατά το ήμισυ μέχρι το 2030, η πρόοδος μέχρι σήμερα είναι ανεπαρκής και απαιτείται σημαντική ενίσχυση των πολιτικών οδικής ασφάλειας [4].

Εν κατακλείδι, τα στατιστικά στοιχεία υπογραμμίζουν τη δραματική διάσταση του προβλήματος των τροχαίων ατυχημάτων. Δεν πρόκειται μόνο για μεμονωμένα περιστατικά, αλλά για μια παγκόσμια κρίση δημόσιας υγείας, με σοβαρές επιπτώσεις στην κοινωνία, την οικονομία και την ποιότητα ζωής.

1.2 Επιπτώσεις τροχαίων ατυχημάτων στην κοινωνία

Τα τροχαία ατυχήματα επιβαρύνουν την κοινωνία με πολλαπλούς, αλληλοσυνδεόμενους τρόπους: Άμεσες ανθρώπινες απώλειες, σοβαροί τραυματισμοί που οδηγούν σε μακροχρόνιες αναπηρίες, οικονομικές δαπάνες για υγειονομική περίθαλψη και αποκατάσταση, απώλεια παραγωγικότητας και επιβάρυνση των οικογενειών και των συστημάτων κοινωνικής πρόνοιας. Σύμφωνα με τις πιο

πρόσφατες επίσημες εκτιμήσεις, οι ετήσιοι παγκόσμιοι θάνατοι από τροχαία ανέρχονται στο περίπου 1.19 εκατομμύριο και τα μη θανατηφόρα τραύματα σε δεκάδες εκατομμύρια, με σημαντικό ποσοστό αυτών να επιφέρουν μόνιμη ή μακροχρόνια αναπηρία [1]. Η οικονομική επιβάρυνση είναι επίσης μεγάλη. Εκτιμήσεις του ΠΟΥ τοποθετούν το άμεσο και έμμεσο κόστος των τροχαίων σε πολλές χώρες κοντά στο ~3% του ΑΕΠ ετησίως, ποσοστό που υπογραμμίζει τον μακροπρόθεσμο αντίκτυπο στην ανάπτυξη και την κοινωνική ευημερία [1].

Οι υψηλότερες τιμές θνησιμότητας ανά 100.000 κατοίκους εντοπίζονται συστηματικά σε χώρες χαμηλού και μεσαίου εισοδήματος, ιδίως στην υπο-Σαχάρια Αφρική και σε μέρη της Ανατολικής Μεσογείου και της Νότιας Ασίας, όπου οι αναφερόμενες τιμές είναι πολύ μεγαλύτερες του παγκόσμιου μέσου όρου. Η γεωγραφική ανισότητα είναι εμφανής: Η Αφρικανική περιοχή παρουσιάζει από τις υψηλότερες θνησιμότητες (π.χ. χώρες με διψήφιες τιμές/100.000), ενώ οι υψηλού εισοδήματος χώρες καταγράφουν πολύ χαμηλότερα ποσοστά. Ταυτόχρονα, οι μεγαλύτερες απόλυτες απώλειες σε αριθμό νεκρών παρατηρούνται σε πληθυσμιακά μεγάλες χώρες με υψηλή κυκλοφορία οχημάτων. Ενδεικτικά, χώρες όπως η Ινδία και η Κίνα καταγράφουν πολύ μεγάλο απόλυτο αριθμό θανάτων, λόγω του μεγάλου πληθυσμού και του όγκου κινητικότητας, ακόμη κι όταν οι κατά κεφαλήν δείκτες είναι μικρότεροι από κάποιες μικρότερες χώρες με πολύ υψηλότερο δείκτη θνησιμότητας. Αυτές οι παρατηρήσεις βασίζονται στις αναλυτικές εκτιμήσεις και χάρτες του ΠΟΥ για τον ρυθμό θνησιμότητας και τα βασικά επιδημιολογικά μεγέθη [2].
Τα τροχαία ατυχήματα έχουν σημαντικό αντίκτυπο σε όλες τις χώρες στις οποίες συναντιούνται, αλλά όταν αυτά συγκεντρώνονται σε χώρες με περιορισμένους πόρους, ο αντίκτυπος πολλαπλασιάζεται:

- Υγεία και μακροχρόνιες συνέπειες από τροχαία: Τα τροχαία ατυχήματα επηρεάζουν δραστικά την ατομική και δημόσια υγεία πέρα από τις άμεσες απώλειες ζωής. Κάθε χρόνο περίπου 1,19 εκατομμύριο άνθρωποι χάνουν τη ζωή τους σε οδικά δυστυχήματα και άλλοι 20–50 εκατομμύρια τραυματίζονται, με τις τελευταίες εκτιμήσεις του Παγκόσμιου Οργανισμού Υγείας να δείχνουν ότι η επιβάρυνση σε «χαμένα έτη υγιούς ζωής λόγω αναπηρίας» (DALYs) είναι τεράστια σε παγκόσμιο επίπεδο, ιδίως σε χώρες χαμηλού και μεσαίου εισοδήματος όπου τα συστήματα υγείας και οι υπηρεσίες αποκατάστασης είναι περιορισμένοι. Ο όγκος των τραυματισμών είναι τέτοιος που καθιστά τα τροχαία έναν από τους κύριους παράγοντες που συμβάλλουν στην παγκόσμια επιβάρυνση με ασθένειες και αναπηρίες, υπερβαίνοντας πολλές άλλες συχνές αιτίες θνησιμότητας και νοσηρότητας σε νεότερες ηλικιακές ομάδες [5].
- Οικονομία και εργασιακή παραγωγικότητα: Η απώλεια εργαζομένων της παραγωγικής ηλικίας μειώνει το εισόδημα νοικοκυριών και το διαθέσιμο εργατικό δυναμικό, ενώ τα έξοδα υγείας και η απώλεια εισοδήματος επιβαρύνουν τόσο τα νοικοκυριά όσο και το δημόσιο σύστημα κοινωνικής προστασίας. Η εκτίμηση κόστους ~3% του ΑΕΠ σε πολλές χώρες δείχνει τη σημαντική μακροοικονομική επίπτωση [1].
- Κοινωνικές ανισότητες: Η επίδραση των τροχαίων ατυχημάτων δεν κατανέμεται ομοιόμορφα μέσα στις κοινωνίες· οι κοινωνικοοικονομικές ανισότητες αντανακλώνται στα αποτελέσματα της οδικής ασφάλειας. Οι χώρες χαμηλού και μεσαίου εισοδήματος καταγράφουν τις υψηλότερες τιμές θανάτων και επιβαρύνσεων από τροχαία [1]. Οι αναλύσεις που διερευνούν το μέγεθος της βλάβης των τροχαίων υπογραμμίζουν ότι τα θύματα με σοβαρούς τραυματισμούς αντιμετωπίζουν πολύπλοκες ανάγκες υγειονομικής περίθαλψης και μακροχρόνια αποκατάσταση, ενώ η έλλειψη κατάλληλων υπηρεσιών στις πιο ευάλωτες χώρες αυξάνει τη μόνιμη αναπηρία και τις επιπτώσεις στη λειτουργική ποιότητα ζωής. οι φτωχότερες ομάδες πλήττονται δυσανάλογα, εντείνοντας κυκλικά τη φτώχεια και την

ανισότητα σε περιοχές όπου η πρόσβαση σε αποζημίωση και ιατρική περίθαλψη είναι περιορισμένη [6].

Στις χώρες όπως η Νορβηγία, η Σουηδία, η Ελβετία, η Ισλανδία και άλλες βόρειες/ δυτικές ευρωπαϊκές χώρες παρατηρούνται σταθερά πολύ χαμηλοί δείκτες θνησιμότητας (θανάτων ανά 100.000 κατοίκους) και σοβαρών τροχαίων συγκρούσεων. Οι συνήθεις εξηγήσεις που αναφέρονται στη βιβλιογραφία περιλαμβάνουν:

- Συστηματική νομοθεσία και επιβολή κανόνων (όρια ταχύτητας, χρήση ζωνών/ κρανών, αυστηρή αστυνόμευση)[2].
- Υψηλά πρότυπα οδικής υποδομής (διαχωρισμένες λωρίδες για ευάλωτους χρήστες, ασφαλή πεζοδρόμια, κόμβοι ασφαλείας όπως κυκλικοί κόμβοι) [2], [7].
- Υψηλά πρότυπα ασφάλειας οχημάτων και εφαρμογή τεχνολογιών ασφαλείας (π.χ. ηλεκτρονικός έλεγχος ευστάθειας) [2], [7].
- Ολοκληρωμένες πολιτικές οδικής ασφάλειας που συνδυάζουν σχεδιασμό, επιβολή και ποιοτική φροντίδα μετά το ατύχημα. Οι διεθνείς συγκρίσεις και οι εκθέσεις οργανισμών όπως το ITF/ OECD δείχνουν ότι χώρες με τέτοια χαρακτηριστικά εμφανίζουν πολύ χαμηλές τιμές θανάτων ανά 100.000 [2], [7].
- Υψηλού επιπέδου εκπαίδευση οδηγών. Στις χώρες αυτές, η προαδειοδοτική εκπαίδευση είναι συστηματική, με σχολές οδηγών που εφαρμόζουν καλά πρότυπα, σχολικές διδασκαλίες/μάθηση οδικής αγωγής, δοκιμασίες που δεν αρκούν μόνο στη θεωρία αλλά και στην πράξη, καθώς και συνεχιζόμενη εκπαίδευση ή αξιολογήσεις μετά την απόκτηση της άδειας σε ορισμένες περιπτώσεις [2], [7].

1.3 Σημασία εκπαίδευσης υποψήφιων οδηγών

Η εκπαίδευση των υποψήφιων οδηγών αποτελεί έναν από τους πιο κρίσιμους παράγοντες για τη βελτίωση της οδικής ασφάλειας και τη μείωση των τροχαίων ατυχημάτων. Η απόκτηση άδειας οδήγησης δεν θα πρέπει να θεωρείται μια τυπική διαδικασία, αλλά ένα στάδιο εντατικής εκπαίδευσης που διαμορφώνει τις οδηγικές συμπεριφορές των νέων οδηγών. Η ποιότητα της εκπαίδευσης πριν από την αδειοδότηση καθορίζει σε μεγάλο βαθμό τη στάση απέναντι στην οδική ασφάλεια, την ικανότητα αναγνώρισης κινδύνων και την συνολική υπευθυνότητα στο δρόμο.
Οι χώρες με τις χαμηλότερες τιμές τροχαίων ατυχημάτων, έχουν υιοθετήσει συνεπή πρότυπα στην εκπαίδευση υποψήφιων οδηγών. Η ένταξη μαθημάτων οδικής αγωγής στο σχολικό πρόγραμμα, εξετάσεις που αξιολογούν όχι μόνο τη θεωρητική γνώση αλλά και την πρακτική ετοιμότητα, καθώς και η συνεχής ενημέρωση οδηγών για νέες τεχνολογίες και κανονισμούς, συμβάλλουν στη διαμόρφωση υπεύθυνων και ασφαλών οδηγικών συνηθειών [8].

Η χρήση προσομοιωτών οδήγησης εμφανίζεται στην βιβλιογραφία ως ένα πολύτιμο συμπληρωματικό εργαλείο στην εκπαίδευση οδηγών, με δυνατότητα να ενισχύσει την παραδοσιακή εκπαίδευση διδάσκοντας δεξιότητες, βελτιώνοντας την ικανότητα αναγνώρισης κινδύνων και προσφέροντας ένα ασφαλές, επαναλήψιμο περιβάλλον εκπαίδευσης χωρίς πραγματικό ρίσκο. Ο πρόσφατος ανασκοπικός έλεγχος 17 εμπειρικών μελετών έδειξε ότι, παρά την σχετικά περιορισμένη» επιστημονική βιβλιογραφία και μεθοδολογικές προκλήσεις (όπως μικρά δείγματα και σπάνιες μετρήσεις μετεκπαίδευσης), η πλειονότητα των μελετών υποστήριξε ότι οι εκπαιδευόμενοι που χρησιμοποίησαν προσομοιωτές εμφάνισαν σημαντική βελτίωση στην απόδοση σε εκπαιδευτικές δοκιμασίες και σε σύγκριση με ομάδες ελέγχου ή προ-εμπλοκή εκπαίδευσης, γεγονός που δείχνει εμπειρική αξία για τη

βελτίωση της οδικής συμπεριφοράς υπό ελεγχόμενες συνθήκες κινδύνου. Τα ευρήματα τόνισαν επίσης ότι η πραγματοποίηση και αίσθηση ρεαλισμού του προσομοιωτή μπορεί να ενισχύσει την εκπαίδευση, ενώ παράλληλα υπογραμμίζονται τα κενά στην τεκμηρίωση της μεταφοράς αυτών των δεξιοτήτων στην πραγματική οδήγηση [9]. Παράλληλα, αναφορές οργανώσεων όπως το CIECA επισημαίνουν ότι οι προσομοιωτές είναι ιδιαίτερα χρήσιμοι ως προσθήκη στην εκπαίδευση των μαθητών οδηγών, ενισχύοντας δεξιότητες όπως η αντίληψη κινδύνου και συμπληρώνοντας τα παραδοσιακά μαθήματα, ενώ αναγνωρίζουν ότι δεν μπορούν ακόμη να υποκαταστήσουν πλήρως τις δοκιμές σε πραγματικές συνθήκες οδικής κυκλοφορίας [10].

Κεφάλαιο 2. Σχεδίαση και υλοποίηση συστήματος

Η ανάπτυξη ενός προσομοιωτή οδήγησης απαιτεί τη σύζευξη μηχανικών τμημάτων, ηλεκτρονικών και λογισμικού με στόχο τη μέγιστη ρεαλιστικότητα, την εργονομία και την εύκολη κλιμάκωση του συστήματος. Το επιλεγμένο μοντέλο υλοποίησης ακολουθεί μια προσέγγιση χαμηλού κόστους αλλά υψηλής πιστότητας, ώστε να προσφέρει στον υποψήφιο οδηγό μια εμπειρία που προσεγγίζει την πραγματική οδήγηση όσο το δυνατόν περισσότερο.

Στο κομμάτι του ηλεκτρονικού υλικού χρησιμοποιείται η πλακέτα Arduino micro, η οποία ενσωματώνει τον μικροελεγκτή ATmega32U4. Ο μικροελεγκτής, ο οποίος λαμβάνει αναλογικά και ψηφιακά σήματα από τα χειριστήρια οδήγησης (τιμόνι, πεντάλ και λοιπά χειριστήρια οχήματος). Μέσω θύρας USB, ο μικροελεγκτής επικοινωνεί με τον υπολογιστή στον οποίο εκτελείται το λογισμικό του προσομοιωτή. Χρησιμοποιώντας ειδική βιβλιοθήκη, το Arduino αναγνωρίζεται από τον υπολογιστή ως γενική συσκευή εισόδου, επιτρέποντας τη διαλειτουργικότητα με το λογισμικό χωρίς την ανάγκη προσαρμοσμένων οδηγών.

Ο σχεδιασμός των μηχανικών τμημάτων πραγματοποιήθηκε στο πρόγραμμα Blender, με χρήση επεκτάσεων όπως το Precision Gear, που του παρέχουν λειτουργίες παραμετρικού CAD. Οι φυσικές διαστάσεις ελήφθησαν από πραγματικό όχημα ώστε το τελικό αποτέλεσμα να είναι εργονομικό και να μιμείται την εμπειρία οδήγησης σε όσο το δυνατόν υψηλότερη πιστότητα. Η κατασκευή έγινε τμήματα απο τρισδιάστατη εκτύπωση, PVC, μεταλλικά μέρη, καθώς και ινοσανίδα διατηρώντας χαμηλό κόστος αλλά υψηλή δομική ακεραιότητα.

Το λογισμικό υλοποιήθηκε σε γλώσσα C# χρησιμοποιώντας την Unity3D ως μηχανή γραφικών, σε συνδυασμό με το Blender για τη δημιουργία τρισδιάστατων μοντέλων για κάθε εικονιζόμενο αντικείμενο από τα μέρη του οχήματος μέχρι τις οδικές υποδομές. Το λογισμικό περιλαμβάνει εργαλεία εσωτερικής ανάπτυξης, τα οποία επιτρέπουν την ταχεία δημιουργία περιεχομένου, όπως animated σκηνές και σενάρια ποικίλης δυσκολίας. Έτσι, το λογισμικό μπορεί να επεκτείνεται και να προσαρμόζεται εύκολα σε διαφορετικές εκπαιδευτικές ανάγκες.

Σχεδιαστικά, υιοθετήθηκε εκτεταμένα το Singleton pattern, το οποίο εξασφαλίζει ότι συγκεκριμένες λειτουργικές μονάδες (όπως ο διαχειριστής ήχου, η φυσική προσομοίωση ή ο ελεγκτής των σεναρίων) υπάρχουν σε μία μόνο παρουσία καθ’ όλη τη διάρκεια εκτέλεσης. Η χρήση του singleton σε αυτό το πλαίσιο απλοποιεί τη διαχείριση πόρων, αποτρέπει διενέξεις μεταξύ πολλαπλών παρουσιών και επιτρέπει την εύκολη συντήρηση της βάσης κώδικα. Επιπλέον, κάποιες λειτουργίες του λογισμικού ακολουθούν μια event-driven αρχιτεκτονική, όπου γεγονότα ενεργοποιούν συγκεκριμένες αντιδράσεις από διάφορες κλάσεις. Η αρχιτεκτονική αυτή είναι σημαντική γιατί μειώνει την πολυπλοκότητα του κώδικα και καθιστά το σύστημα πιο ευέλικτο.

Ένα ακόμη βασικό στοιχείο είναι η διαδικαστική δημιουργία περιεχομένου (procedural content generation), η οποία επιτρέπει τη δημιουργία αστικών σκηνών και οδικών περιβαλλόντων. Εργαλεία αυτόματης παραγωγής περιεχομένου αναπτύχθηκαν για χρήση κατα την διάρκεια της ανάπτυξης καθώς και κατα την διάρκεια εκτέλεσης. Μέσα από τη χρήση αρθρωτών τρισδιάστατων μοντέλων (modular assets), το λογισμικό μπορεί να συνθέτει ποικίλες και ρεαλιστικές διαδρομές για εξάσκηση. Σημαντικός αλγόριθμος που αξιοποιείται είναι ο Wave Function Collapse, ιδιαίτερα διαδεδομένος στη βιομηχανία των βιντεοπαιχνιδιών και των προσομοιώσεων, ο οποίος επιτρέπει τη δημιουργία περιβαλλόντων με βάση προκαθορισμένους κανόνες. Τεχνικές οπως η αυτόματη δημιουργία τρισδιάστατων σχημάτων (procedural mesh generation) χρησιμοποιούνται εκτενώς σε διάφορα στάδια της δημιουργίας περιεχομένου. Με τις τεχνικές αυτές, ο εκπαιδευόμενος οδηγός έχει τη δυνατότητα να

εξασκείται σε πληθώρα ρεαλιστικών σεναρίων, αυξάνοντας τη μεταφερσιμότητα των δεξιοτήτων του στην πραγματική οδήγηση και μειώνοντας σημαντικά το κόστος παραγωγής του λογισμικού.

2.1 Υλοποίηση του λογισμικού προσομοίωσης

Το λογισμικό που παρουσιάζεται σε αυτό το έργο αναπτύχθηκε με στόχο να καλύψει συγκεκριμένες απαιτήσεις που κρίνονται κρίσιμες για την αποτελεσματικότητα του προσομοιωτή. Πρώτη προτεραιότητα αποτελεί ο ρεαλισμός. Τα σενάρια που παρουσιάζονται σε όλα τα μέρη του προσομοιωτή πρέπει να αποτυπώνουν με ακρίβεια πραγματικές καταστάσεις που μπορεί να συναντήσει ο οδηγός στην καθημερινή οδήγηση. Η διδασκαλία των τεχνικών υποστηρίζεται από σαφές οπτικό και ακουστικό περιεχόμενο σε μορφή animation, τα οποία καθοδηγούν τον εκπαιδευόμενο στην εκτέλεση της κάθε άσκησης. Το έργο αυτό περιλαμβάνει επίσης τη δυνατότητα δημιουργίας διαδικαστικά παραγόμενων περιβαλλόντων, επιτρέποντας στον εκπαιδευόμενο να ασκεί τις δεξιότητές του σε έναν πρακτικά απεριόριστο αριθμό απο σενάρια.

Ιδιαίτερα σημαντική είναι και η απαιτούμενη επεκτασιμότητα και συντηρησιμότητα του λογισμικού. Τα προγραμματιστικά μοτίβα που χρησιμοποιούνται επιτρέπουν την εύκολη προσθήκη νέων χαρακτηριστικών και σεναρίων στον προσομοιωτή, κάτι που επιτυγχάνεται σε μεγάλο βαθμό μέσω εσωτερικών εργαλείων που διευκολύνουν την ανάπτυξη πρόσθετων κεφαλαίων και σκηνών με διαφορετικά επίπεδα δυσκολίας. Τέλος, καθοριστικός παράγοντας για τη συνολική εμπειρία είναι η απόδοση. Το λογισμικό πρέπει να εκτελείται ομαλά σε ευρύ σύνολο συστημάτων, ώστε να μην αποκλείει χρήστες με περιορισμένο προϋπολογισμό. Αυτό επιτυγχάνεται μέσω διαφόρων τεχνικών, όπως caching, χρήση επιπέδων λεπτομέρειας (LOD) και βελτιστοποίηση των αλγορίθμων που χρησιμοποιούνται.

2.1.1 Ανάγνωση εισόδων και βαθμονόμηση

Η αλληλεπίδραση του χρήστη με το εικονικό όχημα στηρίζεται θεμελιωδώς στη σωστή ανάγνωση και επεξεργασία των δεδομένων εισόδου από το υλικό (π.χ. τιμόνι, πεντάλ). Για τον σκοπό αυτό αξιοποιείται το Unity Input System, ένα πακέτο που παρέχει ένα ευέλικτο και επεκτάσιμο πλαίσιο διαχείρισης εισόδων. Στον πυρήνα του συστήματος βρίσκεται η έννοια του Input Action, δηλαδή μια αφηρημένη ενέργεια που συνδέεται με έναν ή περισσότερους φυσικούς χειρισμούς (π.χ. το πάτημα ενός κουμπιού ή η περιστροφή του τιμονιού)[11].

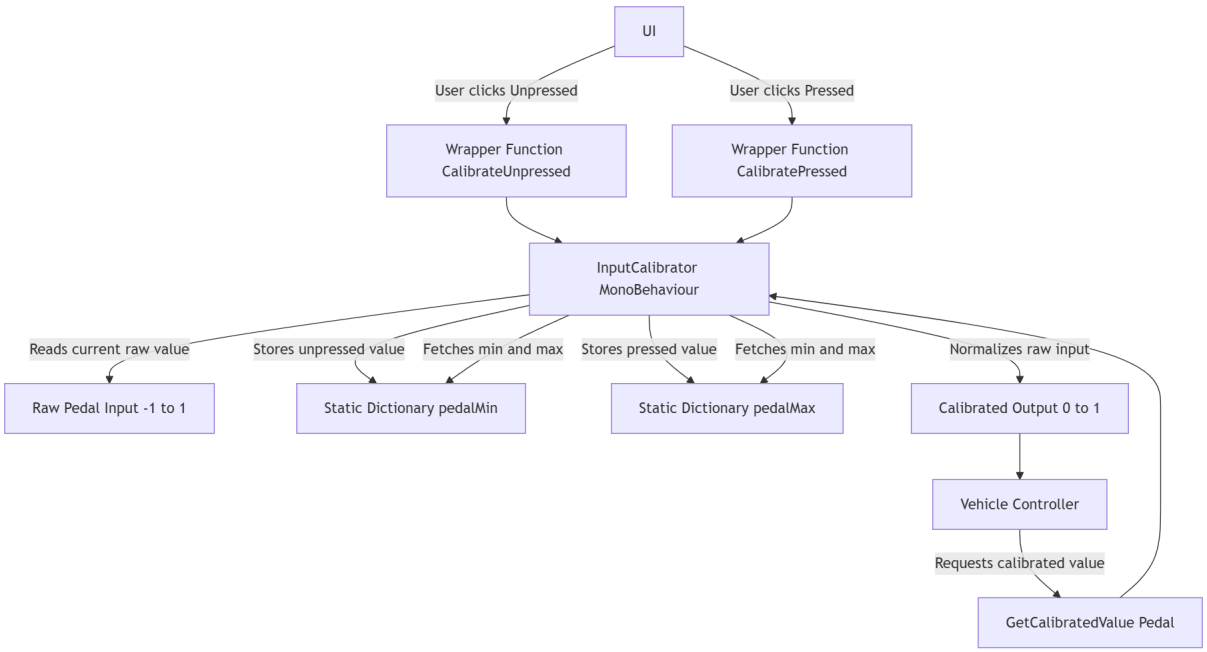
Όλες οι ενέργειες εισόδου που απαιτούνται από το σύστημα ορίζονται στο αρχείο vehicleInput, το οποίο περιέχει τους ορισμούς για το γκάζι, το φρένο, τον συμπλέκτη, το τιμόνι και τον επιλογέα ταχυτήτων. Το αρχείο αυτό συνδέεται με την κλάση PlayerInput που παρέχεται από τη Unity, η οποία αναλαμβάνει να ακούει τις εισόδους που ορίζονται και να τις διοχετεύει στο κατάλληλο σημείο του προγράμματος.

Στην κλάση vehicleController υλοποιούνται συναρτήσεις που καλούνται απευθείας από την κλάση PlayerInput, με σκοπό να εξάγουν τις αριθμητικές τιμές των εισόδων και να τις αποθηκεύουν σε ξεχωριστές μεταβλητές τύπου float, όπως clutch, brake, gas και steering

Τέλος, αξίζει να σημειωθεί ότι δεν απαιτείται περαιτέρω επεξεργασία ή ανάπτυξη προσαρμοσμένων οδηγών για την επικοινωνία του συστήματος με το υλικό. Αυτό επιτυγχάνεται επειδή η βιβλιοθήκη που χρησιμοποιείται από τον μικροελεγκτή φροντίζει ώστε ο υπολογιστής να αναγνωρίζει το χειριστήριο ως ένα γενικό USB input device. Έτσι, η Unity, μέσω του Input System, μπορεί να προσπελάσει τις εισόδους απευθείας, εξασφαλίζοντας απλότητα στη ρύθμιση.

Η βαθμονόμηση των εισόδων που δέχεται το πρόγραμμα από τα πεντάλ υλοποιείται στην κλάση inputCalibrator η οποία:

- Μέσω των συναρτήσεων OnGas, OnBrake, OnClutch λαμβάνει τιμές εισόδου από το Input System.
- Παρέχει συναρτήσεις οι οποίες καλούνται απο το UI για να ρυθμίσει την μέγιστη και την ελάχιστη τιμή για κάθε πεντάλ.
- Αποθηκέυει τις μέγιστες και ελάχιστες τιμές για κάθε πεντάλ σε ένα dictionary
- αποθηκεύει το κάθε dictionary σε στατική μορφή για να είναι προσβάσιμες οι πληροφορίες καθ'ολη την διάρκεια εκτέλεσης του προγράμματος
- Παρέχει μια συνάρτηση η οποία μετατρέπει τις ακατέργαστες τιμές εισόδου σε βαθμονομημένες



Σχήμα 2.1.1: Διάγραμμα συστήματος βαθμονόμησης

2.1.2 Λειτουργία του εικονικού οχήματος

Η πιστή αναπαράσταση της λειτουργίας ενός οχήματος αποτελεί θεμελιώδη προϋπόθεση για τη δημιουργία μιας εκπαιδευτικής προσομοίωσης υψηλής ποιότητας. Στην παρούσα ενότητα αναλύονται οι μηχανισμοί που υλοποιούν τη συμπεριφορά του εικονικού οχήματος, τόσο στο επίπεδο της φυσικής κίνησης όσο και στην οπτικοποίηση των κινούμενων μερών της καμπίνας. Ιδιαίτερη έμφαση δίνεται στην ακρίβεια με την οποία προσομοιώνεται η απόκριση του οχήματος σε εξωτερικές και εσωτερικές δυνάμεις, καθώς και στη ρεαλιστική απεικόνιση της αλληλεπίδρασης του οδηγού με τα στοιχεία ελέγχου.

Η φυσική του οχήματος περιλαμβάνει την απόδοση της κίνησης, της αδράνειας, της επιτάχυνσης και της πέδησης, καθώς και την αλληλεπίδραση με την επιφάνεια του δρόμου μέσω ενός συστήματος ανάρτησης που προσομοιώνει πραγματικές παραμέτρους άνεσης και σταθερότητας. Η λειτουργία των εσωτερικών μερών, όπως το τιμόνι, τα πεντάλ και ο επιλογέας ταχυτήτων, αποδίδεται με δυναμικό τρόπο, ώστε κάθε κίνηση του οδηγού να αντικατοπτρίζεται άμεσα στο εικονικό περιβάλλον. Παράλληλα, οι εικονικοί καθρέπτες παρέχουν μια απεικόνιση του χώρου γύρω από το όχημα,

επιτρέποντας στον εκπαιδευόμενο να αναπτύξει τις απαραίτητες δεξιότητες ελέγχου του περιβάλλοντος, όπως ακριβώς θα έκανε σε πραγματικές συνθήκες οδήγησης.

2.1.2.1 Φυσική του οχήματος

Για τον χειρισμό της φυσικής συμπεριφοράς του εικονικού οχήματος αξιοποιείται το Physics API της Unity, το οποίο βασίζεται στην τεχνολογία PhysX της NVIDIA. Το PhysX αποτελεί μια από τις πιο διαδεδομένες και αποδοτικές μηχανές φυσικής στον χώρο της αλληλεπιδραστικής γραφικής απεικόνισης, προσφέροντας υψηλή ακρίβεια και βελτιστοποιημένη απόδοση σε πραγματικό χρόνο. Η Unity ενσωματώνει το PhysX στον πυρήνα της, παρέχοντας στον προγραμματιστή έτοιμες και ευέλικτες δομές για τη διαχείριση δυνάμεων, αδράνειας, συγκρούσεων και αλληλεπιδράσεων αντικειμένων. Αυτό καθιστά το εργαλείο ιδανικό για εφαρμογές προσομοίωσης, όπου η αξιοπιστία και η ρεαλιστική αναπαράσταση της φυσικής συμπεριφοράς είναι καίριας σημασίας.

Στην παρούσα εφαρμογή, δύο βασικές κλάσεις του Unity Physics API αξιοποιούνται για την υλοποίηση του οχήματος: η Rigidbody, η οποία προσδίδει μάζα, κέντρο βάρους και αδράνεια στο όχημα ώστε να υπόκειται σε ρεαλιστικούς νόμους κίνησης, και η WheelCollider, η οποία προσομοιώνει τη δυναμική των τροχών, συμπεριλαμβανομένης της τριβής, της πρόσφυσης, της ανάρτησης και της μεταφοράς κίνησης από τον κινητήρα στο δρόμο.

Η κλάση vehicleController λειτουργεί ως ο ενδιάμεσος μηχανισμός που οδηγεί το Physics API, αξιοποιώντας τόσο παραμέτρους που έχουν οριστεί από τον προγραμματιστή κατά το στάδιο ανάπτυξης (όπως η ισχύς του κινητήρα, η σκληρότητα της ανάρτησης ή τα όρια διεύθυνσης), όσο και τις τιμές εισόδου που καταγράφονται σε πραγματικό χρόνο από τον οδηγό κατά την προσομοίωση (π.χ. πίεση στα πεντάλ, στροφή του τιμονιού, αλλαγή ταχυτήτων). Με αυτόν τον τρόπο, επιτυγχάνεται μια πιστή και δυναμική αναπαράσταση της οδηγικής εμπειρίας, όπου η φυσική συμπεριφορά του οχήματος ανταποκρίνεται άμεσα στις ενέργειες του χρήστη.

Η επικοινωνία με το physics api της unity γίνεται στον βρόχο update της κλάσης vehicleController. Η εφαρμογή της επιτάχυνσης και της επιβράδυνσης γίνεται μέσω της επικοινωνίας με τα instances της κλάσης wheelCollider των τροχών. συγκεκριμένα τίθενται τιμές στα πεδία motorTorque και brakeTorque στα instances, οι οποίες αναπαριστούν την ροπή επιτάχυνσης και πέδησης αντίστοιχα σε μονάδες του SI.

```
if(Mathf.RoundToInt(brake) == 0){
    frontLeftCollider.brakeTorque = 0;
    frontRightCollider.brakeTorque = 0;
    rearLeftCollider.brakeTorque = 0;
    rearRightCollider.brakeTorque = 0;

    frontLeftCollider.motorTorque = gas*torqueMultiplier;
    frontRightCollider.motorTorque = gas*torqueMultiplier;
    rearLeftCollider.motorTorque = gas*torqueMultiplier;
    rearRightCollider.motorTorque = gas*torqueMultiplier;
}else{
    frontLeftCollider.brakeTorque = brake*brakeMultiplier;
    frontRightCollider.brakeTorque = brake*brakeMultiplier;
    rearLeftCollider.brakeTorque = brake*brakeMultiplier;
    rearRightCollider.brakeTorque = brake*brakeMultiplier;
}
```

Η εφαρμογή της στροφής γίνεται θέτοντας μια τιμή στο πεδίο steerAngle της κλάσης WheelCollider η οποία είναι πολλαπλάσια της εισόδου του χρήστη. Επειτα εξάγεται η περιστροφή και η τοποθεσία του WheelCollider στις μεταβλητές FrontLeftRotation και FrontLeftPosition αντίστοιχα, και θέτουμε τις τιμές αυτές στο αντικείμενο που αναπαριστά τον τροχό frontLeftMesh. Επίσης θέτουμε μια τιμή περιστροφής στις δαγκάνες των φρένων ώστε να περιστρέφονται ταυτόχρονα με τους εμπρόσθιους τροχούς. Με παρόμοιο τρόπο γίνονται οι υπολογισμοί για τους τρεις εναπομείναντες τροχούς.

```
frontLeftCollider.steerAngle = steering*maxSteeringAngle;

Quaternion FrontLeftRotation;
Vector3 FrontLeftPosition;
frontLeftCollider.GetWorldPose(out FrontLeftPosition, out FrontLeftRotation);
frontLeftMesh.transform.position = FrontLeftPosition;
frontLeftMesh.transform.rotation = FrontLeftRotation;

caliperFL.transform.localRotation =
Quaternion.Euler(caliperFL.transform.localRotation.eulerAngles.x,
steering*maxSteeringAngle, caliperFL.transform.localRotation.eulerAngles.z);
```

2.1.2.2 Σύστημα διαχείρισης διαχείρισης των στροφών του κινητήρα

Στην περίπτωση που έχει επιλεχθεί αυτόματο κιβώτιο ταχυτήτων υλοποιήθηκε σύστημα διαχείρησης στροφών. Το υποσύστημα που διαχειρίζεται τις στροφές ανά λεπτό (RPM) του κινητήρα έχει ως βασική λειτουργία τον υπολογισμό της κατάλληλης σχέσης μετάδοσης, προκειμένου να διατηρείται το όχημα εντός του βέλτιστου εύρους λειτουργίας. Συγκεκριμένα, η ταχύτητα του κινητήρα αξιοποιείται τόσο για την αλλαγή ταχυτήτων όσο και για τον καθορισμό του αντίστοιχου φάσματος στροφών, λαμβάνοντας υπόψη τις εκάστοτε συνθήκες κίνησης. Για την αποφυγή υπερβολικά συχνών αλλαγών ταχυτήτων εφαρμόζεται μηχανισμός υστέρησης, ο οποίος εξομαλύνει τη διαδικασία μετάβασης μεταξύ διαφορετικών σχέσεων.

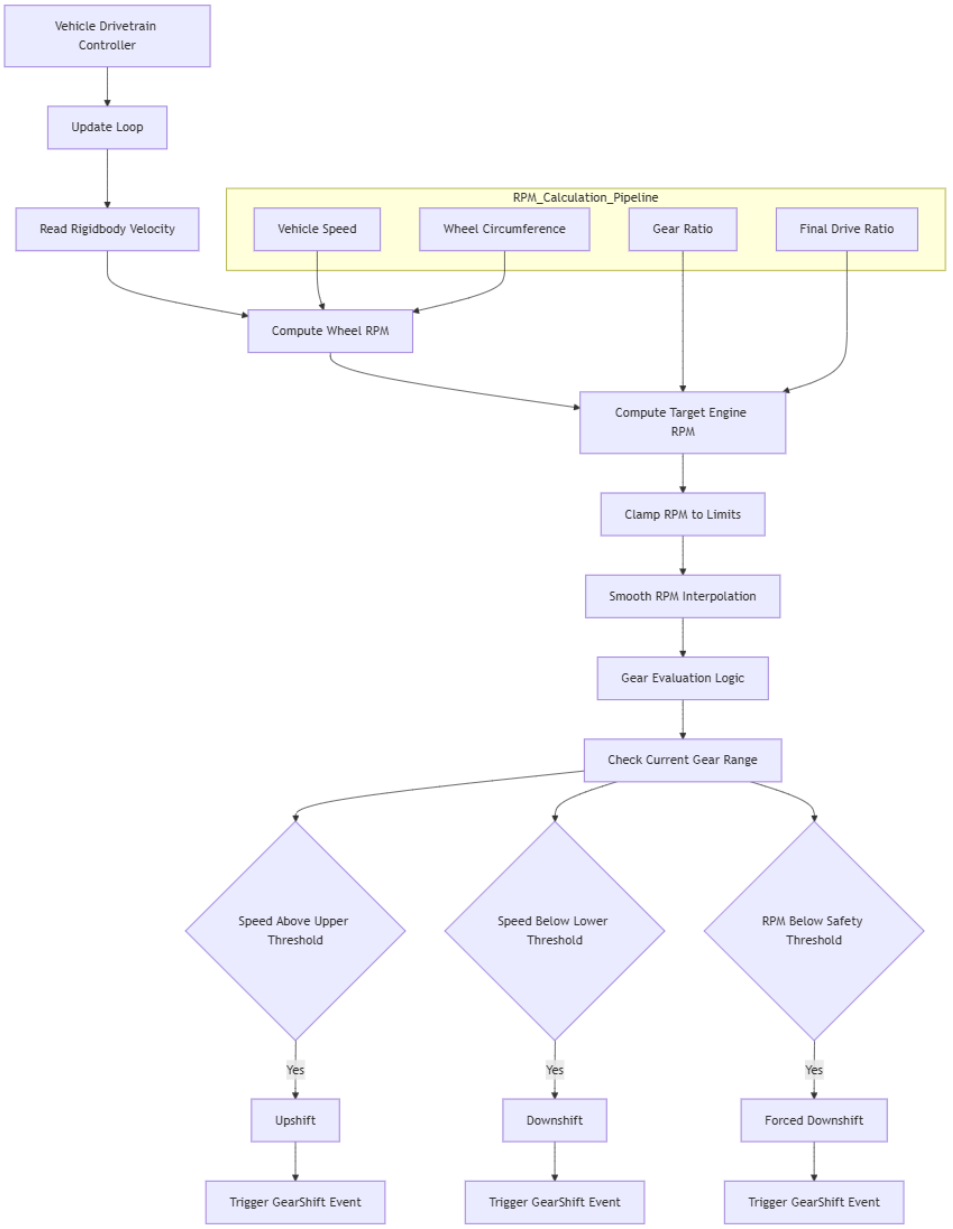
Οι σχέσεις μετάδοσης αποθηκεύονται σε έναν πίνακα δεκαδικών τιμών, με κάθε στοιχείο να αντιστοιχεί σε μία ταχύτητα. Ο τελικός λόγος μετάδοσης (final drive ratio), ο οποίος αναπαριστά το διαφορικό του οχήματος, φυλάσσεται σε ξεχωριστή μεταβλητή. Οι στροφές των τροχών υπολογίζονται βάσει της ταχύτητας του οχήματος, η οποία διαιρείται με την περιφέρεια του τροχού και έπειτα με το 60, ώστε να μετατραπεί σε περιστροφές ανά λεπτό.

Στον βρόχο Update της κλάσης vehicleController, η τιμή targetRPM υπολογίζεται χρησιμοποιώντας τις στροφές των τροχών, τον λόγο της τρέχουσας σχέσης μετάδοσης και τον τελικό λόγο μετάδοσης. Στη συνέχεια, η πραγματική τιμή του rpm εξομαλύνεται και προσεγγίζεται σταδιακά προς την targetRPM με τη χρήση της συνάρτησης Mathf.Lerp, διασφαλίζοντας ρεαλιστική συμπεριφορά. Τέλος, καλείται η συνάρτηση UpdateGearWithHysteresis, η οποία λαμβάνει ως είσοδο την ταχύτητα του οχήματος σε km/h και την τρέχουσα τιμή rpm, καθορίζοντας με ομαλό τρόπο την αλλαγή σχέσης μετάδοσης.

Η συνάρτηση UpdateGearWithHysteresis:

- Για κάθε σχέση μετάδοσης στον πίνακα gearRatios, υπολογίζει τη μέγιστη ταχύτητα (maxSpeedForGear) και την ελάχιστη ταχύτητα (minSpeedForGear) που αντιστοιχούν σε εκείνη τη σχέση, με βάση το εύρος στροφών (minRPM–maxRPM), τον τελικό λόγο μετάδοσης (finalDriveRatio) και την διάμετρο του τροχού.

- Ελέγχει εάν το όχημα βρίσκεται αυτή τη στιγμή στη συγκεκριμένη σχέση μετάδοσης, αν δεν είναι, προχωράει στην επόμενη σχέση
- Αν η ταχύτητα του οχήματος είναι μεγαλύτερη από το μέγιστο όριο για αυτήν τη σχέση (συν ένα περιθώριο υστέρησης – hysteresis) και υπάρχει διαθέσιμη υψηλότερη σχέση, το κιβώτιο ανεβάζει ταχύτητα.
- Αν η ταχύτητα του οχήματος είναι μικρότερη από το ελάχιστο όριο (μείον το hysteresis) και υπάρχει διαθέσιμη χαμηλότερη σχέση, το κιβώτιο κατεβάζει ταχύτητα.
- Αν οι στροφές του κινητήρα είναι κάτω από το όριο downShiftRPM και η τρέχουσα σχέση είναι μεγαλύτερη από την πρώτη, το κιβώτιο κατεβάζει ταχύτητα για να μην σβήσει ο κινητήρας.
- Κάθε φορά που αλλάζει σχέση, γίνεται ενημέρωση μέσω του γεγονότος gearShiftEvent, το οποίο λαμβάνει ως ορίσματα την προηγούμενη και την τρέχουσα σχέση.



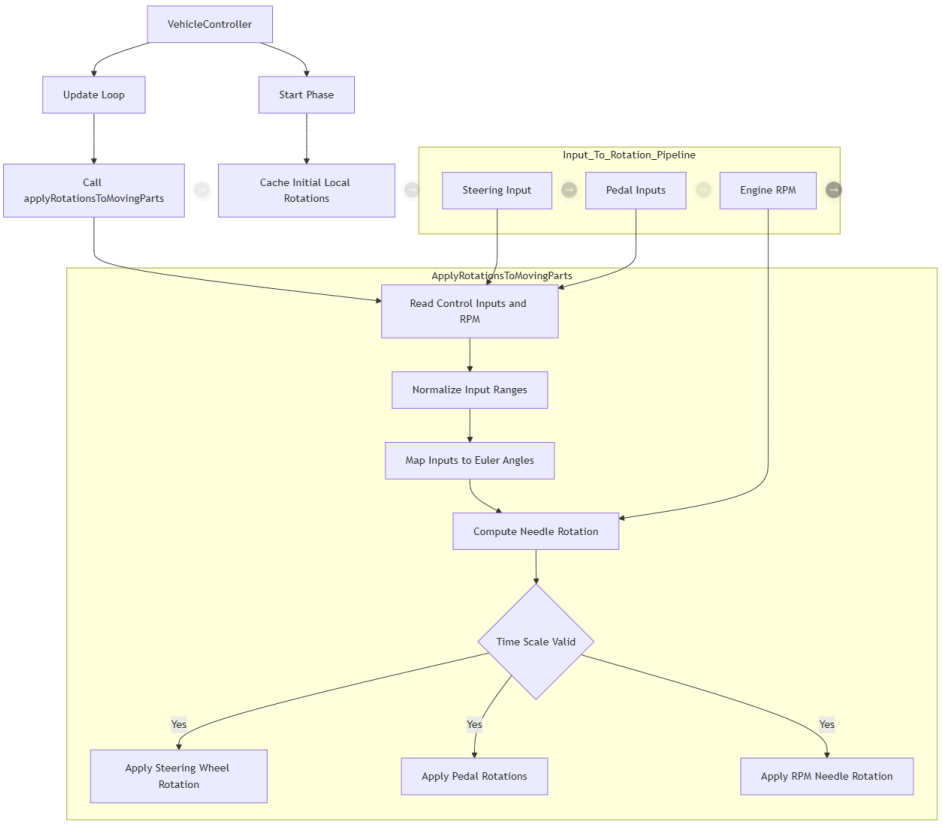
Σχήμα 2.1.2.2: Λογικό διάγραμμα αυτόματης μετάδοσης

2.1.2.3 Κινούμενα μέρη στο εσωτερικό του οχήματος

Για την ενίσχυση της εμβύθισης του χρήστη στην προσομοίωση, ορισμένα στοιχεία του εσωτερικού του οχήματος αποδίδονται με περιστροφική κίνηση, η οποία συγχρονίζεται με τις πραγματικές εισόδους του οδηγού. Συγκεκριμένα, κινούμενα μέρη όπως το τιμόνι, τα πεντάλ επιτάχυνσης, φρένου και συμπλέκτη, καθώς και ο δείκτης στροφών, περιστρέφονται σε πραγματικό χρόνο. Κατά την εκκίνηση της εφαρμογής καλείται η συνάρτηση initializeMovingParts(), η οποία θέτει τις αρχικές τιμές περιστροφής σε μορφή τετραδόνιου (quaternion) για όλα τα κινούμενα μέρη. Στη συνέχεια, μέσα στον βρόγχο Update, εκτελείται η συνάρτηση applyRotationsToMovingParts(), η οποία εφαρμόζει τις αντίστοιχες περιστροφές βάσει των δεδομένων εισόδου του χρήστη (π.χ. στροφή τιμονιού, πίεση πεντάλ), και τις μοίρες περιστροφής του κάθε αντικειμένου ξεχωριστά. Με αυτό τον τρόπο διασφαλίζεται ότι η οπτική αναπαράσταση της καμπίνας παραμένει σε απόλυτη συμφωνία με τις πραγματικές ενέργειες του οδηγού, ενισχύοντας τη ρεαλιστικότητα της μαθησιακής διαδικασίας.

Η συνάρτηση applyRotationsToMovingParts:

- Υπολογίζει κανονικοποιημένες τιμών τιμές εισόδου. Το τιμόνι κανονικοποιείται από το εύρος -1, 1 σε 0,1. Τα πεντάλ (συμπλέκτης, φρένο, γκάζι) κανονικοποιούνται από το εύρος 0,1
- Μετατρέπει κανονικοποιημένες τιμές σε γωνίες περιστροφής. Για κάθε στοιχείο υπολογίζεται η αντίστοιχη γωνία περιστροφής με χρήση της Mathf.Lerp, ώστε να προκύψει η κατάλληλη ενδιάμεση τιμή.
- Εφαρμόζει περιστροφές στα Αντικείμενα. Το τιμόνι περιστρέφεται γύρω από τον τοπικό άξονα Y, συνδυάζοντας την αρχική περιστροφή με την τρέχουσα. Τα πεντάλ γκαζιού, φρένου και συμπλέκτη περιστρέφονται γύρω από τον τοπικό άξονα X, προσομοιώνοντας το πάτημά τους. Ο δείκτης στροφών περιστρέφεται γύρω από τον άξονα Z, αντικατοπτρίζοντας τις στροφές του κινητήρα.



Σχήμα 2.1.2.3: Λογικό διάγραμμα κίνησης των εσωτερικών μερών του οχήματος

2.1.2.4 Λειτουργία μοχλού ταχυτήτων

Στο εσωτερικό της καμπίνας του οδηγού έχει υλοποιηθεί ένας μοχλός ταχυτήτων, ο οποίος ανταποκρίνεται σε πραγματικό χρόνο στις αλλαγές σχέσεων μετάδοσης και κινείται αναλόγως. Το τρισδιάστατο μοντέλο που αναπαριστά τον μοχλό έχει rigged δομή με τη χρήση μεθοδολογίας αντίστροφης κινηματικής (Inverse Kinematics, IK). Η αντίστροφη κινηματική αποτελεί μια τεχνική ευρέως χρησιμοποιούμενη στα γραφικά υπολογιστών και στη ρομποτική, κατά την οποία ο τελικός στόχος κίνησης (π.χ. η θέση του χεριού ή στην προκειμένη περίπτωση η θέση της λαβής του μοχλού) καθορίζεται απευθείας, και ο αλγόριθμος υπολογίζει αυτόματα τις ενδιάμεσες γωνίες και τις κινήσεις των αρθρώσεων που απαιτούνται ώστε να επιτευχθεί η συγκεκριμένη θέση. Με αυτόν τον τρόπο επιτυγχάνεται ρεαλιστική και ομαλή κίνηση χωρίς την ανάγκη χειροκίνητης ρύθμισης κάθε μεμονωμένου άξονα περιστροφής.

Στην παρούσα εφαρμογή, η IK λαβή (inverse kinematic handle) κινείται σε διάταξη τύπου H (H pattern), όπως ακριβώς συμβαίνει στους περισσότερους χειροκίνητους μοχλούς ταχυτήτων οχημάτων παραγωγής. Το τρισδιάστατο μοντέλο του μοχλού ακολουθεί την κίνηση αυτής της λαβής, αποδίδοντας με ρεαλισμό τόσο τις αλλαγές στη θέση του μοχλού όσο και τις φυσικές παραμορφώσεις του δερμάτινου καλύμματος στη βάση του. Ο αλγόριθμος IK (inverse kinematic solver) που χρησιμοποιείται για τον σκοπό αυτό προέρχεται από το πακέτο της Unity Animation Rigging, ενώ η διαχείριση της κίνησης της IK λαβής πραγματοποιείται στην κλάση shifterIKcontroller, η οποία ελέγχει τη ροή των κινήσεων σε συγχρονισμό με τις αλλαγές ταχυτήτων που καταγράφει το σύστημα.

Κατά την εκκίνηση της εφαρμογής, στην κλάση vehicleController ορίζεται ένα γεγονός (event), το οποίο έχει ως σκοπό να παρακολουθεί και να διαχειρίζεται τις αλλαγές ταχυτήτων. Στη συνέχεια, η συνάρτηση gearShiftTriggered εγγράφεται (subscribes) σε αυτό το γεγονός, ώστε να εκτελείται κάθε φορά που ενεργοποιείται. Σε κάθε αλλαγή ταχύτητας, το γεγονός καλείται (invoke) με ορίσματα την τρέχουσα σχέση μετάδοσης και την επόμενη. Με τον τρόπο αυτό η συνάρτηση gearShiftTriggered εκτελείται αυτόματα, λαμβάνοντας τις απαραίτητες παραμέτρους για να διαχειριστεί τη μετάβαση.

Αξίζει να σημειωθεί ότι, για λόγους απλότητας, η όπισθεν κωδικοποιείται ως ταχύτητα 6, ώστε να ενσωματώνεται ομοιόμορφα στη λογική του κιβωτίου και να υποστηρίζονται εύκολα οι μηχανισμοί που σχετίζονται με αλλαγές κατεύθυνσης. Η συνάρτηση gearShiftTriggered αντι αντί να εκτελέσει απευθείας την αλλαγή πυροδοτεί την συνάρτηση queueGearShift, η οποία βάζει όλες τις αλλαγές ταχυτήτων να εκτελούνται σε σειρά, η μία μετα την άλλη. Διαφορετικά, αν δύο αλλαγές ταχυτήτων συμβούν με πολύ μικρή χρονική καθυστέρηση η μία απο την άλλη, θα δημιουργηθούν ανεπιθύμητες συμπεριφορές.

α) Λειτουργία της κλάσης shifterIKcontroller

Αρχικά ορίζονται δύο μεταβλητές floating point, οι HpatternHorizontalOffset και HpatternVerticalOffset, οι οποίες καθορίζουν το εύρος κίνησης της IK λαβής (handle) στον οριζόντιο και κάθετο άξονα αντίστοιχα. Κατά την εκκίνηση του προγράμματος, η κλάση αποθηκεύει την αρχική θέση της IK λαβής σε μια μεταβλητή τύπου Transform. Στη συνέχεια, υπολογίζονται οι θέσεις των σχέσεων μετάδοσης με βάση τις τιμές των offsets σε σχέση με το κεντρικό σημείο. Για παράδειγμα, η πρώτη ταχύτητα αντιστοιχεί σε θέση -horizontal offset στον άξονα X και +vertical offset στον άξονα Y. Αντίστοιχα, υπολογίζονται και οι ενδιάμεσες θέσεις (midpoints), οι οποίες βρίσκονται μεταξύ δύο σχέσεων· για παράδειγμα, το ενδιάμεσο σημείο ανάμεσα στην πρώτη και τη δεύτερη ταχύτητα βρίσκεται στη θέση -horizontal offset στον άξονα X και 0 στον άξονα Y. Για τη διαχείριση αυτών των ενδιάμεσων σημείων ορίζεται το struct midpointEntry, το οποίο συσχετίζει τις σχέσεις μετάδοσης με

τα αντίστοιχα midpoint positions. Η συνάρτηση fillMidpointEntries δημιουργεί αυτές τις συσχετίσεις και τις αποθηκεύει σε πίνακα τύπου midpointEntry με όνομα midpointEntries. Η συνάρτηση getMidpointIndex δέχεται ως παράμετρο τον δείκτη της σχέσης μετάδοσης και επιστρέφει τον αντίστοιχο δείκτη midpoint, ενώ η συνάρτηση getMidpointPosition δέχεται τον δείκτη midpoint και επιστρέφει την ακριβή θέση του ενδιάμεσου σημείου, στην οποία πρέπει να κινηθεί η IK λαβή. Για την ομαλή μετάβαση προς αυτές τις θέσεις, η συνάρτηση moveTowards υλοποιεί την σταδιακή κίνηση της IK λαβής προς τον στόχο.

Με την αξιοποίηση όλων των παραπάνω βοηθητικών συναρτήσεων, η κεντρική συνάρτηση switchGear πραγματοποιεί την πλήρη κίνηση αλλαγής ταχύτητας, με παραμέτρους τις τιμές from και to που υποδηλώνουν την τρέχουσα και την επόμενη σχέση μετάδοσης. Με τον τρόπο αυτό, ο μοχλός αποδίδει ρεαλιστικά τη διαδικασία αλλαγής ταχυτήτων, ακολουθώντας πιστά την κλασική διάταξη τύπου Η (H-pattern).

β) Η συνάρτηση switchGear:

- Ορίζει isShifting = true και ενεργοποιεί τον συμπλέκτη (clutchControl(true)) ώστε να μπορεί να γίνει η αλλαγή με ασφάλεια.
- Υπολογίζει το σωστό ενδιάμεσο σημείο μεταξύ των σχέσεων, χρησιμοποιώντας την αρχική σχέση μετάδοσης (getMidpointIndex(from)) και μετακινεί τη λαβή IK προς τα εκεί
- Ελέγχει αν το ενδιάμεσο σημείο αντιστοιχεί στην τελική ταχύτητα, Αν ναι, η λαβή IK μετακινείται κατευθείαν στη θέση της ταχύτητας-στόχου (gearboxPositions[to]). Αν όχι προσαρμόζει τον δείκτη ενδιάμεσου σημείου (πάνω ή κάτω, ανάλογα αν η αλλαγή είναι προς τα πάνω ή προς τα κάτω) και επαναλαμβάνει την κίνηση μέχρι να φτάσει στο σωστό ενδιάμεσο σημείο
- Ολοκληρώνει την κίνηση: Όταν φτάσει στο σωστό ενδιάμεσο σημείο, η λαβή IK μετακινείται στη θέση της τελικής ταχύτητας.
- Λήξη αλλαγής ταχύτητας: Απελευθερώνεται ο συμπλέκτης (clutchControl(false)) και το isShifting επιστρέφει σε false.

```
public IEnumerator switchGear(int from, int to) {
      isShifting = true;
      vehicleController.instance.clutchControl(true);
      try {
        int currentMidpointIndex = getMidpointIndex(from);

        yield return StartCoroutine(moveTowards(new
Vector3(midpointPositions[currentMidpointIndex].x, initialIKPosition.y,
midpointPositions[currentMidpointIndex].y)));
        if (currentMidpointIndex == getMidpointIndex(to)) {
          yield return StartCoroutine(moveTowards(new Vector3(gearboxPositions[to].x,
initialIKPosition.y, gearboxPositions[to].y)));
          yield return null;
        }
        else {
          if (from < to) {
            currentMidpointIndex++;
          } else {
            currentMidpointIndex--;
          }
          yield return StartCoroutine(moveTowards(new
Vector3(midpointPositions[currentMidpointIndex].x, initialIKPosition.y,
```

```
midpointPositions[currentMidpointIndex].y)));
          if (currentMidpointIndex == getMidpointIndex(to)) {
            yield return StartCoroutine(moveTowards(new Vector3(gearboxPositions[to].x,
initialIKPosition.y, gearboxPositions[to].y)));
            yield return null;
          } else {
            if (from < to) {
              currentMidpointIndex++;
            } else {
              currentMidpointIndex--;
            }
            yield return StartCoroutine(moveTowards(new
Vector3(midpointPositions[currentMidpointIndex].x, initialIKPosition.y,
midpointPositions[currentMidpointIndex].y)));
            if (currentMidpointIndex == getMidpointIndex(to)) {
              StartCoroutine(moveTowards(new Vector3(gearboxPositions[to].x,
initialIKPosition.y, gearboxPositions[to].y)));
              yield return null;
            }
          }
        }
      } finally {
        vehicleController.instance.clutchControl(false);
        isShifting = false;
      }
}
```

2.1.2.5 Υλοποίηση εικονικών καθρεφτων

Οι εικονικοί καθρέφτες υλοποιήθηκαν τοποθετώντας τρεις εικονικές κάμερες σε συγκεκριμένα σημεία και αποθηκεύοντας σε πραγματικό χρόνο κάθε καρέ από το αποτέλεσμα τους σε μια εικόνα (texture) η οποία εν τέλει απεικονίζεται πάνω στην επιφάνεια των τρισδιάστατων μοντέλων τα οποία απεικονίζουν τους καθρέφτες.

Στο πρώτο στάδιο της υλοποίησης τους σχεδιάστηκαν τα μοντέλα των τριών καθρεφτών εντός του Blender. Σχεδιάστηκαν οι επιφάνειες των κρυστάλλων ξερωριστά απο την υπόλοιπη συναρμογή, καθώς μόνο σε αυτές θα απεικονίζεται η εικόνα που παράγουν οι εικονικές κάμερες. Στη συνέχεια τοποθετήθηκαν τρεις εικονικές κάμερες (rendering cameras) στα κέντρα των επιφανειών που καταλαμβάνουν οι καθρέφτες και ευθυγραμμίστηκαν τα διανύσματα κατεύθυνσης τους έτσι ώστε να δείχνουν όλα προς τα πίσω, σύμφωνα με την κατεύθυνση που είναι στραμμένοι οι εικονικοί καθρέφτες..

Η εικόνα που λαμβάνουν οι καθρέφτες σχηματίζεται χρησιμοποιώντας την τεχνολογία render texture. Ένα Render Texture είναι ένας ειδικός τύπος εικόνας, στον οποίο μπορεί να αποθηκεύσει το αποτέλεσμα της μια εικονική κάμερα. Αντί να εμφανίζεται η εικόνα απευθείας στην οθόνη, αποθηκεύεται σε πραγματικό χρόνο σε ένα render texture για περαιτέρω επεξεργασία. Δημιουργήθηκαν τρία render textures, ένα με διαστάσεις 512x512 για τον πίσω καθρέφτη και δύο με διαστάσεις 256x256 για τους δύο πλαϊνούς πλαϊνούς καθρέφτες. Οι αναλύσεις αυτές είναι σκοπίμως χαμηλές για να μειωθεί το υπολογιστικό κόστος, χωρίς όμως αυτό να μειώνει κατά πολύ την ποιότητα της εικόνας, δεδομένου ότι ο χώρος τον οποιον οποίον καταλαμβάνουν οι καθρέφτες στην οθόνη είναι από μόνος του σχετικά μικρός. Τέλος, ορίστηκαν τα τρία render textures ως εξόδοι στις αντίστοιχες κάμερες και συσχετίστηκαν με τις επιφάνειες οι οποίες απεικονίζουν τους κρυστάλλους των καθρεφτών.

Αυτό το σύστημα είναι από την φύση του αρκετά κοστοβόρο για τον επεξεργαστή γραφικών, καθώς χρησιμοποιούνται συνολικά 4 εικονικές κάμερες για την αναπαράσταση του τρισδιάστατου περιβάλλοντος, μία κύρια και τρεις για κάθε έναν από τους καθρέφτες. Για αυτόν τον λόγο χρησιμοποιήθηκαν κάποιες μέθοδοι βελτιστοποίησης της απόδοσης αυτού του συστήματος. Πρώτον, μειώθηκε η ανάλυση της εικόνας που απεικονίζεται πάνω στους καθρέφτες. Αντί για την πλήρη ανάλυση οθόνης χρησιμοποιήθηκαν εικόνες 256x256 για τους πλαϊνούς καθρέφτες και 512x512 για τον οπίσθιο. Δεύτερον, τροποποιήθηκε η μεταβλητή Far Clipping Plane και στις τρεις εικονικές κάμερες, η οποία ορίζει την μέγιστη απόσταση από το σημείο θέασης στην οποία εμφανίζονται αντικείμενα. Η μεταβλητή αυτή μειώθηκε στα 150 μέτρα. Τρίτον, για την απεικόνιση της εικόνας πάνω στους κρυστάλλους χρησιμοποιήθηκε απλότερος αλγόριθμος σκίασης (shader) ο οποίος λαμβάνει υπ’οψιν μόνον την εικόνα που παρέγεται απο το render texture αγνοώντας άλλες πηγές φωτός και απλοποιόντας την σκίαση σε πολύ βασικό επίπεδο.

2.1.3 Δεξιότητα στάθμευσης

Η διαδικασία στάθμευσης ενός οχήματος αποτελεί μία από τις πιο απαιτητικές δεξιότητες για έναν οδηγό, ιδιαίτερα σε περιορισμένους και πολυσύχναστους αστικούς χώρους. Στο πλαίσιο αυτού του θέματος, αναπτύχθηκε ένα περιβάλλον προσομοίωσης, το οποίο επικεντρώνεται στη διαδικασία της στάθμευσης, προσφέροντας μία διαδραστική εμπειρία με τη μέγιστη δυνατή ρεαλιστική αναπαράσταση. Το εν λόγω κεφάλαιο έχει σχεδιαστεί με γνώμονα τη μέγιστη δυνατή προσέγγιση της πραγματικής εμπειρίας στάθμευσης, τόσο από πλευράς οπτικών και ηχητικών ερεθισμάτων, όσο και από την άποψη της φυσικής συμπεριφοράς του οχήματος στο ψηφιακό περιβάλλον. Η δημιουργία

ενός ψηφιακού περιβάλλοντος στάθμευσης έχει ως στόχο όχι μόνο την ακριβή αποτύπωση των προκλήσεων της διαδικασίας, αλλά και τη δυνατότητα εκπαίδευσης και αξιολόγησης οδηγών σε συνθήκες που μιμούνται την πραγματικότητα. Το κεφάλαιο παρουσιάζει διάφορες τεχνικές οπως όπως το παράλληλο παρκάρισμα και το παρκάρισμα με οπισθογωνία, σε διάφορα σκηνικά με αυξανόμενο επίπεδο δυσκολίας.

Στο κεφάλαιο αυτό παρουσιάζονται η υλοποίηση του συγκεκριμένου τμήματος της προσομοίωσης, οι τεχνικές που χρησιμοποιήθηκαν για την επίτευξη του ρεαλισμού και οι τρόποι εκπαίδευσης και αξιολόγησης των υποψήφιων οδηγών.

Σχήμα 2.1.3.1: Εικόνα από το στάδιο επιλογής επιπέδου δυσκολίας.

Το πρόγραμμα αποτελείται από τρια κύρια σημεία. Αρχικά, υλοποιήθηκε η διαδικασία παραγωγής ενός animation το οποίο αποτυπώνει βήμα προς βήμα τη διαδικασία στάθμευσης, προσφέροντας στον υποψήφιο οδηγό μια ολοκληρωμένη εμπειρία μάθησης. Το οπτικό περιεχόμενο προβάλλεται ταυτόχρονα από δύο διαφορετικές γωνίες θέασης, ενισχύοντας την κατανόηση του χωρικού προσανατολισμού, ενώ η αφήγηση λειτουργεί μέσω συνθετικής φωνής, χρησιμοποιώντας τεχνολογία μετατροπής κειμένου σε ομιλία (Text-to-Speech). Η συνδυαστική χρήση οπτικοακουστικού υλικού στοχεύει στη βελτίωση της εμβύθισης και της αποτελεσματικότητας της εκπαιδευτικής διαδικασίας.

Έπειτα, υλοποιήθηκε το πρόγραμμα που διαχειρίζεται διάφορα επίπεδα δυσκολίας. Η λειτουργία αυτή επιτρέπει την επιλογή ανάμεσα σε πολλαπλά επίπεδα, μεταβάλλοντας παραμέτρους όπως η πολυπλοκότητα του χώρου στάθμευσης, ο αριθμός κινήσεων που επιτρέπεται να χρησιμοποιηθούν, καθώς και ο βαθμός ακρίβειας που απαιτείται για τη στάθμευση.

Τρίτον, αναπτύχθηκε μια αυτόματη λειτουργία αξιολόγησης της στάθμευσης, η οποία ελέγχει κατά πόσο το όχημα έχει τοποθετηθεί σωστά εντός των ορίων της θέσης στάθμευσης. Η διαδικασία ελέγχου βασίζεται σε γεωμετρικά κριτήρια, όπως η απόσταση από τα οχήματα, ο προσανατολισμός του οχήματος και η τελική του θέση. Το σύστημα έχει σχεδιαστεί με ευελιξία, ώστε να προσαρμόζεται δυναμικά στις απαιτήσεις κάθε επιπέδου δυσκολίας, επιτρέποντας αυστηρότερα ή επιεικέστερα κριτήρια αξιολόγησης, ανάλογα με το σενάριο της προσομοίωσης.

Στο περιβάλλον της στάθμευσης υλοποιήθηκαν μια σειρά από κλάσεις που συνεργάζονται για να παρέχουν την λειτουργικότητα της προσομοίωσης. Οι κλάσεις που χρησιμοποιούνται είναι οι εξής:

- vehicleController: Αποτελεί την κύρια κλάση που χειρίζεται την είσοδο του οδηγού και διαχειρίζεται τη φυσική προσομοίωσης του οχήματος. Υπεύθυνη για τον έλεγχο του τιμονιού,

των πεντάλ, των ταχυτήτων και της κινητικής συμπεριφοράς, εξασφαλίζει ομαλή και ρεαλιστική αλληλεπίδραση με το περιβάλλον. Επιπλέον, διαχειρίζεται την αντίδραση του οχήματος σε αλλαγές κατεύθυνσης, ταχύτητα, επιτάχυνση και πέδηση, καθώς και την αλληλεπίδραση με άλλα αντικείμενα της προσομοίωσης, όπως colliders και trigger zones.

- parkingController: Η κλάση αυτή λειτουργεί ως κεντρικός κόμβος για ολόκληρη την προσομοίωση στάθμευσης. Συντονίζει όλους τους μηχανισμούς αξιολόγησης, διαχειρίζεται τη φόρτωση διαφορετικών περιβαλλόντων, ελέγχει τις γωνίες θέασης, και συγχρονίζει την αναπαραγωγή animation και ήχου.
- triggerHandler: Αυτή η κλάση διαχειρίζεται τους trigger colliders και χρησιμοποιείται από τον κύριο μηχανισμό αξιολόγησης της στάθμευσης. Επικοινωνεί με το parkingController για να ελέγχει την ακρίβεια της στάθμευσης, παρακολουθώντας την παρουσία του οχήματος στις καθορισμένες περιοχές (position και boundary zones).
- objectRecorder: Η κλάση αυτή χρησιμοποιείται για την καταγραφή animations, τα οποία αποτελούν μέρος της μαθησιακής διαδικασίας στο κεφάλαιο. Επιτρέπει την αποθήκευση και αναπαραγωγή κινήσεων του οχήματος με μέγιστη ακρίβεια, ώστε να υποστηρίζεται η οπτικοποιημένη διδασκαλία της στάθμευσης.
- audioController: Διαχειρίζεται όλο το ηχητικό περιεχόμενο της προσομοίωσης, συμπεριλαμβανομένων οδηγιών, ηχητικών εφέ και TTS (text-to-speech). Παρέχει κεντρική πρόσβαση σε αναπαραγωγή, παύση και διακοπή των ήχων, εξασφαλίζοντας ομαλό συγχρονισμό με τα animations και τα γεγονότα της προσομοίωσης.
- handleCollisionWithVehicle: Αυτή η κλάση αναλαμβάνει τη διαχείριση των συγκρούσεων με τα σχετικά αντικείμενα που θεωρούνται collidable, όπως τα παρακείμενα οχήματα. Όταν ανιχνεύεται σύγκρουση, ενεργοποιεί μηχανισμούς αποτυχίας, διακόπτει τη ροή της προσομοίωσης και παρέχει οπτικοποίηση του σημείου του ατυχήματος για ανατροφοδότηση στον χρήστη.

2.1.3.1 Δημιουργία και αναπαραγωγή οπτικοακουστικού υλικού

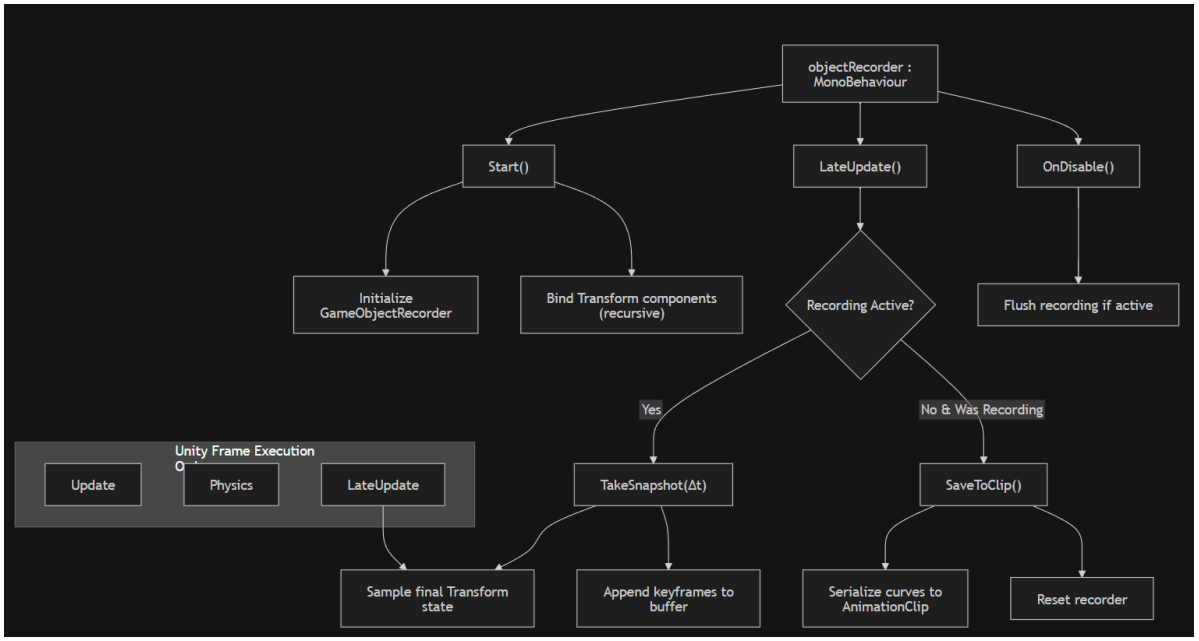
Το βασικό εργαλείο για τη διαδικασία καταγραφής του animation υλοποιήθηκε στην κλάση objectRecorder. Το εργαλείο αυτό χρησιμοποιεί την κλάση GameObjectRecorder, η οποία παρέχεται από το περιβάλλον Unity3d. Η κλάση αυτή επιτρέπει την καταγραφή των αλλαγών στις ιδιότητες ενός αντικειμένου σε συνάρτηση με τον χρόνο και την αποθήκευσή τους σε μια μεταβλητή τύπου AnimationClip , η οποία στο τέλος αποθηκεύεται στον δίσκο.

Οι ιδιότητες που μας ενδιαφέρει να καταγράψουμε είναι όλες τύπου Transform. Ο τύπος Transform περιέχει τρία τρισδιάστατα διανύσματα, το διάνυσμα τοποθεσίας, περιστροφής και κλίμακας. Από αυτά τα τρία μας είναι χρήσιμα μόνο τα διανύσματα τοποθεσίας και περιστροφής. Για τη σωστή αναπαράσταση του animation είναι σημαντικό να καταγραφούν οι ιδιότητες τοποθεσίας και περιστροφής του ριζικού αντικειμένου, δηλαδη του οχήματος που ελέγχουμε καθώς και όλων των θυγατρικών αντικειμένων του, όπως η περιστροφή των τροχών, του τιμονιού, και των πεντάλ, τις αλλαγές στην θέση του μοχλού ταχυτήτων και την απόσταση των τροχών από το σασί κατα κατά την λειτουργία του συστήματος αναρτήσεων.

Παρακάτω εμφανίζεται ένα διάγραμμα της αρχιτεκτονικής του εργαλείου. Το διάγραμμα αποτυπώνει:

- Στάδιο αρχικοποίησης
    - Δημιουργία recorder

    - Binding όλων των αντικειμένων Transform στην ιεραρχία που απαιτείται καταγραφή
- Στάδιο δειγματολειψίας (Late Update)
    - Δειγματοληψία μετά το πέρας της εκτέλεσης του βρόχου φυσικής και του κύριου βρόχου του προγράμματος
    - Καταγραφή τελικής κατάστασης του κάθε frame
- Φάση μετατροπής
    - Μετατροπή buffer σε AnimationClip
    - Reset για επόμενη καταγραφή



Σχήμα 2.1.3.1: Αρχιτεκτονική συστήματος καταγραφής

2.1.3.1.1 Δημιουργία υλικού

Η διαδικασία χρήσης του εργαλείου κατά την ανάπτυξη περιλαμβάνει την φόρτωση ενός προκαθορισμένου δοκιμαστικού περιβάλλοντος, την ενεργοποίηση της μεταβλητής καταγραφής (recording flag) και την εκτέλεση των κατάλληλων κινήσεων ώστε το όχημα να ολοκληρώσει με ακρίβεια τη διαδικασία στάθμευσης. Το εργαλείο καταγράφει όχι μόνο την κύρια μετακίνηση του οχήματος, αλλά και τη συμπεριφορά όλων των θυγατρικών αντικειμένων (child objects), όπως τροχοί, τιμόνι και πεντάλ, εξασφαλίζοντας έτσι μέγιστο βαθμό ρεαλισμού στην αναπαραγωγή της σκηνής από τον χρήστη.

Για την καθοδήγηση του υποψήφιου οδηγού κατά τη διάρκεια της στάθμευσης, χρησιμοποιείται ηχητικό περιεχόμενο. Για την δημιουργία του αξιοποιήθηκε το εργαλείο τεχνητής νοημοσύνης LuvVoice για τη δημιουργία προφορικού λόγου μέσω μετατροπής κειμένου σε ομιλία (Text to Speech). Το ηχητικό υλικό που παρήχθηκε περιλαμβάνει σαφείς φωνητικές οδηγίες που περιγράφουν κάθε βήμα της διαδικασίας, όπως την αλλαγή κατεύθυνσης, την τοποθέτηση του τιμονιού ή τη στιγμή ακινητοποίησης του οχήματος. Ιδιαίτερη σημασία δόθηκε στον χρονικό συγχρονισμό ανάμεσα στην αφήγηση και τις αντίστοιχες κινήσεις του οχήματος στο animation. Ο σωστός συγχρονισμός ενισχύει την κατανόηση και την εκπαιδευτική αξία της προσομοίωσης, προσεγγίζοντας όσο το δυνατόν περισσότερο την εμπειρία ενός πραγματικού μαθήματος οδήγησης.

Για τη διαχείριση του ηχητικού περιεχομένου υλοποιήθηκε η κλάση audioController. Αξιοποιεί την εγγενή κλάση AudioSource που παρέχει το περιβάλλον της Unity για την αναπαραγωγή, παύση και διακοπή ηχητικών αρχείων. Η υλοποίηση ακολουθεί το μοτίβο singleton, καθιστώντας την κλάση παγκοσμίως προσβάσιμη από οποιαδήποτε άλλη κλάση, γεγονός που επιτρέπει την ενοποιημένη διαχείριση του ήχου σε όλο το σύστημα. Τα ηχητικά κλιπ αποθηκεύονται σε προσαρμοσμένη δομή δεδομένων, η οποία υποστηρίζει την ανάκτησή τους βάσει ονομασίας, διευκολύνοντας την αναζήτηση και χρήση τους. Η αρχιτεκτονική αυτή συμβάλλει στη βελτίωση της επεκτασιμότητας και της συντηρησιμότητας του κώδικα, ενώ παράλληλα διασφαλίζει την ακριβή και συγχρονισμένη αναπαραγωγή ηχητικών ερεθισμάτων στο πλαίσιο της προσομοίωσης.

Για την υλοποίηση του audioController αρχικά ορίστηκε η στατική μεταβλητή instance η οποία επιτρέπει την πρόσβαση στις λειτουργίες του audioController. Η μεταβλητή αρχικοποιείται στην συνάρτηση Start ωστε να ορίσει την μοναδική κλάση του audioController ως instance

```
public static audioController instance;

void Start()
{
    instance = this;
}
```

Έπειτα δημιουργήθηκε ένα struct για την αποθήκευση του κλιπ:

```
public struct clip
{
    public string name;
    public AudioClip audioClip;
}
```

Υλοποιήθηκε η συνάρτηση playClip για την αναπαραγωγή του κλιπ βάση ονομασίας :

```
public void playClip(string name)
{
    for(int i = 0; i < audioClips.Length; i++)
    {
        if (audioClips[i].name == name) {
            audioSourceRef.clip = audioClips[i].audioClip;
            audioSourceRef.Play();
        }
    }
}
```

Τέλος, δημιουργήθηκαν οι συναρτήσεις Pause και Stop. Η χρήση της κλάσης είναι γίνεται ως εξής:

```
audioController.instance.playClip("clip name");
audioController.instance.pause();
audioController.instance.stop();
```

2.1.3.1.2 Αναπαραγωγή υλικού

Η αναπαραγωγή των καταγεγραμμένων κινήσεων του οχήματος ελέγχεται από την κλάση parkingController.cs στη συνάρτηση playAnimatedScene και onClipEnd. Για την ορχηστροποίηση του καταγεγραμμένου animation χρησιμοποιείται το animation state machine το οποίο παρέχει το περιβάλλον της Unity στην κλάση Animator. Η κλάση αυτή παρέχει συναρτήσεις οι οποίες μπορούν να χρησιμοποιηθούν για τον έλεγχο της κατάστασης του animation, όπως η συνάρτηση SetTrigger και ResetTrigger. Χρησιμοποιώντας το GUI του περιβάλλοντος ορίστηκε το state defaultState και το state playAnimation. Επισης ορίστηκαν triggers, τα οποία θα ενεργοποιούνται για τη μετάβαση από το ένα state στο άλλο. Η αναπαραγωγή του κλιπ υλοποιείται στη συνάρτηση playAnimatedScene, η οποία προετοιμάζει το περιβάλλον αναπαραγωγής, ρυθμίζει τις γωνίες θέασης και σκανδαλίζει την αναπαραγωγή του οπτικού και ακουστικού περιεχομένου.

Ανάλυση της λειτουργίας playAnimatedScene:

- updateCameraState(cameraState.ANIMATION); : ρυθμίζει τις κατάλληλες γωνίες λήψεις για τις δυο εικονικές κάμερες και ορίζει την τοποθεσία του αποτελέσματός τους στο user interface.
- disableLevelsOnAnimationStart(); : απενεργοποιεί τις ιεραρχίες αντικειμένων που σχετίζονται με τα περιβάλλοντα των επιπέδων δυσκολίας.
- clipEnvironmentRef.SetActive(true); : ενεργοποιεί την ιεραρχία αντικειμένων που σχετίζεται με το περιβάλλον αναπαραγωγής του κλιπ.
- RigBuilderRef.enabled = false; : απενεργοποιεί το instance της κλάσης RigBuilder, η οποία χρησιμοποιείται για το animation του επιλογέα ταχυτήτων κατά τη διάρκεια της οδήγησης, γεγονός κρίσιμο για να μπορεί να ελεγχθεί η κατάσταση του μοχλού από τις πληροφορίες του καταγεγραμμένου κλιπ.
- playerCar.GetComponent<Animator>().SetTrigger("play"); : σκανδαλίζει την αναπαραγωγή του κλιπ στο state machine μέσω της κλάσης Animator.
- audioController.instance.playClip("pp instructions"); : σκανδαλίζει την αναπαραγωγή του ακουστικού περιεχομένου

```
public void playAnimatedScene()
{
    isAnimationPlaying = true;
    updateCameraState(cameraState.ANIMATION);
    disableLevelsOnAnimationStart();
    clipEnvironmentRef.SetActive(true);

    RigBuilderRef.enabled = false;

    playerCar.GetComponent<Animator>().ResetTrigger("exit");
    playerCar.GetComponent<Animator>().SetTrigger("play");
    audioController.instance.playClip("pp instructions");
}
```

Η λήξη του animation μπορεί να σκανδαλιστεί είτε από το UI, κατά τη μετάβαση στην προσομοίωση, είτε από το state machine, κατά τη λήξη του κλιπ, μέσω της συνάρτησης OnStateExit που κληρονομείται από την κλάση της Unity StateMachineBehaviour. Η συνάρτηση επαναφέρει τις δύο εικονικές κάμερες στην αρχική τους θέση, φορτώνει ξανά την ιεραρχία αντικειμένων του περιβάλλοντος της προσομοίωσης και διακόπτει την αναπαραγωγή του ακουστικού περιεχομένου.

2.1.3.1.3 Γωνίες θέασης και η διαχείρισή τους

Κατά την αναπαραγωγή του εισαγωγικού κλιπ του κεφαλαίου αυτού δίνεται η δυνατότητα παρακολούθησης της διαδικασίας της στάθμευσης από δύο διαφορετικές γωνίες θέασης. Αυτό αποτελεί κρίσιμο στοιχείο της προσομοίωσης, καθώς προσφέρει στον χρήστη μια πιο ολοκληρωμένη και πολυδιάστατη αντίληψη της οδηγικής σκηνής. Στα αριστερά παρουσιάζεται η γωνία τρίτου προσώπου όπου ο χρήστης μπορεί να δει το όχημά του από μια υπερυψωμένη οπτική γωνία, επιτρέποντας στον χρήστη να παρακολουθήσει το όχημα καθώς εκτελεί τον ελιγμό στάθμευσης στο σύνολό της. Η συγκεκριμένη γωνία θέασης προσφέρει ξεκάθαρη εικόνα της θέσης του οχήματος σε σχέση με το περιβάλλον, τα γειτονικά οχήματα και τα όρια στάθμευσης, ενισχύοντας έτσι την κατανόηση της συνολικής πορείας και των απαιτούμενων διορθώσεων. Στα δεξιά παρουσιάζεται η οπτική γωνία πρώτου προσώπου, όπου ο χρήστης βιώνει τον ελιγμό της στάθμευσης από τη θέση του οδηγού, αποκτώντας μια ρεαλιστική απεικόνιση απεικόνιση της διαδικασίας. Μέσα από αυτήν την οπτική γωνία, είναι ορατές οι κινήσεις των πεντάλ, του τιμονιού και του επιλογέα ταχυτήτων, οι οποίες συγχρονίζονται με τις ενέργειες του οχήματος. Παράλληλα, ο οδηγός έχει καθαρή εικόνα από τον εσωτερικό καθρέφτη και τους πλαϊνούς καθρέφτες, διευκολύνοντας την εκτίμηση αποστάσεων και την εκτέλεση διορθώσεων με ακρίβεια κατά την διαδικασία της στάθμευσης.

Το σύστημα διαχείρισης των γωνιών θέασης αποτελείται από δύο εικονικές κάμερες, την mainCamera και την animationCamera. Η mainCamera αφορά την απεικόνιση πρώτου προσώπου και η animationCamera αφορα την υπερυψωμένη απεικόνιση απεικόνιση τρίτου προσώπου που χρησιμοποιείται κατα την διάρκεια του animation. Για να οριστουν οι τοποθεσιες τοποθεσίες και οι περιστροφες περιστροφές των δυο καμερων καμερών σε διάφορα στάδια στάδια του κύκλου ζωής του προγράμματος χρησιμοποιούνται αντικείμενα τύπου Transform, τα οποία είναι φορτωμένα στη σκηνή καθ’ όλη τη διάρκεια της εκτέλεσης. Το κάθε ένα από αυτά τα βοηθητικά αντικείμενα περιέχει ένα τρισδιάστατο διάνυσμα (Vector3), που αποθηκεύει την τοποθεσία της κάμερας και ένα τετραδόνιο (Quaternion), που αποθηκεύει την περιστροφή της. Τα αντικείμενα αυτά είναι τα εξής:

- cameraDefaultTransformMarker : ορίζει την κατάσταση της animationCamera κατά την είσοδο στο κεφάλαιο, κατά την επιλογή του επιπέδου δυσκολίας. Η γωνία θέασης έχει οριστεί έτσι ώστε να δείχνει με βέλτιστο τρόπο όλο το επιλεγμένο περιβάλλον
- cameraClipTranformMarker : ορίζει την κατάσταση της animationCamera κατά την αναπαραγωγή του εκπαιδευτικού κλιπ. Η γωνία θέασης έχει ρυθμιστεί έτσι ώστε να παρέχει στον χρήστη τον καλύτερο δυνατό χωρικό προσανατολισμό.
- firstPersonCameraDefaultTransformMarker: ορίζει την κατάσταση της mainCamera κατά τη διαδικασία της εκτέλεσης του ελιγμού από τον χρήστη. Έχει οριστεί εντός του οχήματος έτσι ώστε το περιβάλλον να καλύπτει ένα όσο το δυνατόν μεγαλύτερο μέρος της οθόνης. Γίνεται και ταυτόχρονα να υπάρχει καθαρή εικόνα από όλους τους καθρέφτες του οχήματος. Επίσης είναι σημαντικό να είναι ορατές οι κινήσεις των πεντάλ, του τιμονιού και του επιλογέα ταχυτήτων.
- firstPersonCameraClipTransformMarker : ορίζει την κατάσταση της mainCamera κατά την αναπαραγωγή του εκπαιδευτικού κλιπ. Η γωνία θέασης είναι εντός του οχήματος και έχει ρυθμιστεί ώστε να ανταποκρίνεται βέλτιστα στο τμήμα της οθόνης το οποίο καλύπτει η συγκεκριμένη απεικόνιση κατά την αναπαραγωγή, το ½ της οθόνης στον οριζόντιο άξονα και τα ¾ της οθόνης στον κάθετο άξονα.
- firstPersonCameraCrashTransformMarker: Ορίζει την καταστηση της mainCamera σε περίπτωση σύγκρουσης, μεταφέροντας την προβολή σε οπτική γωνία τρίτου προσώπου από ψηλά. Με αυτόν τον τρόπο, ο χρήστης μπορεί να δει με σαφήνεια τη θέση του οχήματός του τη στιγμή της πρόσκρουσης, καθώς και το ακριβές σημείο επαφής με το άλλο όχημα ή αντικείμενο. Η δυνατότητα αυτή ενισχύει την κατανόηση του λάθους, διευκολύνοντας την ανατροφοδότηση και συμβάλλοντας στη βελτίωση της οδηγικής συμπεριφοράς στις επόμενες προσπάθειες.

Η διαχείριση των γωνιών θέασης κατα τη διάρκεια του εκπαιδευτικού κλιπ γίνεται στο αρχειο parkingController.cs.

- Ορίζεται το enumeration cameraState για να γίνει δυνατή η επιλογή μεταξύ των διάφορων καταστάσεων.

```
    public enum cameraState
    {
        ANIMATION,
        ENV_SELECTION
    }
```

Κατά το ξεκίνημα της αναπαραγωγής του κλιπ, στη συνάρτηση playAnimatedScene εκτελείται η συνάρτηση updateCameraState, με παράμετρο ANIMATION, και μετα τη λήξη του στη συνάρτηση onClipEnd εκτελείται η ίδια συνάρτηση με παράμετρο ENV_SELECTION. Η συνάρτηση αυτή, ανάλογα με το state, ορίζει της κατάλληλες τοποθεσίες και περιστροφές στις δύο εικονικές κάμερες και αποφασίζει ποιες από αυτές είναι ενεργές. Επίσης θέτει τις διαστάσεις της απεικόνισης πρώτου προσώπου, έτσι ώστε να καταλαμβάνει το κατάλληλο μέρος της οθόνης και να εμφανίζεται στη δεξιά πλευρά.

```
void updateCameraState(cameraState state)
{
    if (state == cameraState.ANIMATION)
    {
        fpCamera.SetActive(true);
        fpCamera.GetComponent<Camera>().rect = new Rect(0.5f, 0.25f, 0.5f,
0.75f);
        fpCamera.transform.position = fpCameraClipTransform.transform.position;
        fpCamera.transform.rotation = fpCameraClipTransform.transform.rotation;
        tpCamera.transform.position = tpCameraClipTransform.transform.position;
        tpCamera.transform.rotation = tpCameraClipTransform.transform.rotation;
    }
    if(state == cameraState.ENV_SELECTION)
    {
        UIrenderCameraObserver.instance.SwitchToThirdPerson();
        tpCamera.GetComponent<cameraFollow>().isEnabled = false;
        tpCamera.SetActive(true);
        fpCamera.SetActive(false);
        tpCamera.transform.position =
tpCameraDefaultTransform.transform.position;
        tpCamera.transform.rotation =
tpCameraDefaultTransform.transform.rotation;
    }
}
```

Σχήμα 2.1.3.1.3: Φαίνονται οι δύο γωνίες λήψης κατα την διαδικασία αναπαραγωγής του κλιπ

2.1.3.2 Διαχείριση ψηφιακών περιβάλλοντων και ασύγχρονη φόρτωση

Η ενότητα αυτή επικεντρώνεται στη λειτουργία εναλλαγής μεταξύ διαφορετικών επιπέδων δυσκολίας, και τη διαχείριση τους στον κύκλο ζωής του προγράμματος. Τα διαφορετικά επίπεδα δυσκολίας καθορίζουν τις απαιτήσεις και τις συνθήκες της διαδικασίας στάθμευσης. Κάθε επίπεδο διαφοροποιείται μέσα από παραμέτρους, όπως η απόσταση ανάμεσα στα οχήματα, ο διαθέσιμος χώρος στάθμευσης, ο μέγιστος αριθμός επιτρεπόμενων κινήσεων και ο μέγιστος επιτρεπτός χρόνος για να ολοκληρωθεί η σταθμευση. Με αυτόν τον τρόπο, η προσομοίωση προσαρμόζεται δυναμικά τόσο στις ικανότητες όσο και στις ανάγκες του εκάστοτε χρήστη, προσφέροντας μια κλιμακωτή μαθησιακή εμπειρία που κυμαίνεται από πιο επιεική έως και ιδιαίτερα απαιτητικά σενάρια.

Η διαχείριση του περιβάλλοντος κάθε επιπέδου δυσκολίας υλοποιείται μέσω διακριτών ιεραρχιών αντικειμένων (object hierarchies), μέσα στις οποίες κάθε επίπεδο έχει σχεδιαστεί ως αυτόνομη δομή. Κατά τον κύκλο ζωής της προσομοίωσης, μόνο η ιεραρχία του ενεργού επιπέδου είναι παρούσα στην μνήμη, γεγονός που βελτιστοποιεί τη διαχείριση πόρων και μειώνει την πολυπλοκότητα του συστήματος. Παράλληλα, σε κάθε επίπεδο συνδέονται ειδικοί μηχανισμοί αξιολόγησης, οι οποίοι προσαρμόζονται στις αντίστοιχες απαιτήσεις δυσκολίας. Με τον τρόπο αυτό, επιτυγχάνεται τόσο η σαφής διάκριση των διάφορων περιβαλλόντων όσο και η στοχευμένη αξιολόγηση της απόδοσης του χρήστη σε διαφορετικά σενάρια στάθμευσης. Η μέθοδος διαχείρισης περιβαλλόντων που χρησιμοποιήθηκε καθιστά ιδιαίτερα απλή τη διαδικασία προσθήκης νέων επιπέδων, ενισχύοντας έτσι την επεκτασιμότητα της εφαρμογής. Καθώς κάθε επίπεδο υλοποιείται ως ξεχωριστή ιεραρχία αντικειμένων με αυτόνομους μηχανισμούς αξιολόγησης. Ο προγραμματιστής μπορεί εύκολα να δημιουργήσει ή να ενσωματώσει νέα σενάρια χωρίς να επηρεάζει τη λειτουργικότητα των ήδη υπαρχόντων. Αυτή η modular αρχιτεκτονική διευκολύνει τη συντήρηση και μελλοντική επέκταση της προσομοίωσης με περισσότερες παραλλαγές και βαθμίδες δυσκολίας.

Η διαδικασία σχεδιασμού των περιβαλλόντων βασίζεται στη χρήση ξεχωριστών ιεραρχιών αντικειμένων, καθεμία από τις οποίες τοποθετείται σε διαφορετικό αντικείμενο και συνδέεται με τα

δεδομένα του προγράμματος (data) μέσω ενός prefab link. Ένα **prefab link** είναι ένα εργαλείο που παρέχεται από το περιβάλλον της Unity και αποτελεί ουσιαστικά ένα πρότυπο αντικειμένου, το οποίο αποθηκεύεται στα δεδομένα του προγράμματος και μπορεί να αναπαραχθεί με συνέπεια σε διάφορα σημεία του έργου, καθώς και να φορτωθεί στη μνήμη ασύγχρονα. Στην πράξη, ο σχεδιασμός του κάθε περιβάλλοντος πραγματοποιείται με την επεξεργασία του prefab link σε **context mode**, ώστε να είναι δυνατή η επεξεργασία του με την υπόλοιπη σκηνή παρούσα (π.χ. οι δρόμοι ή άλλα σταθερά στοιχεία), γεγονός που επιτρέπει την ακριβή προσαρμογή του επιπέδου στο συνολικό περιβάλλον. Κάθε αλλαγή που πραγματοποιείται στο αντικείμενο αντικατοπτρίζεται αυτόματα στα δεδομένα του προγράμματος, διασφαλίζοντας συνοχή και εξοικονόμηση χρόνου. Κατά τη διάρκεια της ανάπτυξης του λογισμικού, όλα τα περιβάλλοντα είναι διαθέσιμα στη μνήμη, ενώ κατά την παραγωγή φορτώνονται μόνο οσα ζητούνται από τα δεδομένα του προγράμματος, βελτιστοποιώντας έτσι την απόδοση και τη διαχείριση των πόρων.

Η φόρτωση των διαφορετικών περιβαλλόντων υλοποιείται ασύγχρονα μέσω του **Addressables API** της Unity, το οποίο προσφέρει έναν ευέλικτο τρόπο διαχείρισης των πόρων. Με την προσέγγιση αυτή, κάθε περιβάλλον μπορεί να φορτωθεί ή να αποδεσμευτεί δυναμικά κατά την εκτέλεση της εφαρμογής, χωρίς να διακόπτεται η ροή του κύριου κύκλου του προγράμματος. Η ασύγχρονη φόρτωση μειώνει τους χρόνους αναμονής και αποτρέπει την εμφάνιση καθυστερήσεων, καθώς τα δεδομένα ανακτώνται στο παρασκήνιο. Επιπλέον, το Addressables API επιτρέπει την καλύτερη οργάνωση των περιβαλλόντων μέσω μοναδικών διευθύνσεων, διευκολύνοντας την επεκτασιμότητα και την εύκολη ενσωμάτωση νέων επιπέδων στην εφαρμογή.

Για την αναπαράσταση του κάθε επιπέδου στη μνήμη δημιουργήθηκε ένα array από structs του τύπου parkingLevel. Η δομή ενός parkingLevel περιλαμβάνει τον αριθμό του επιπέδου, ένα αντικείμενο το οποίο αποθηκεύει τη διεύθυνση της ιεραρχίας του περιβάλλοντος στα δεδομένα του προγράμματος, μια μεταβλητή στην οποία δηλώνουμε τον μέγιστο επιτρεπτό αριθμό κινήσεων για το συγκεκριμένο επίπεδο, και μία που δηλώνουμε το μέγιστο επιτρεπτο χρόνο στάθμευσης.

```
public struct parkingLevel
{
    public int levelNum;
    public AssetReferenceGameObject levelObject;
    public int maxAmountOfMoves;
    public int maxTimeInSeconds;
}
```

Η φόρτωση των περιβάλλοντων κατά την εναλλαγή επιπέδων δυσκολίας σκανδαλίζεται απο την συνάρτηση switchLevelEnvironment η οποία πρώτα αποδεσμεύει από τη μνήμη το τρέχον περιβάλλον μέσω της συνάρτησης unloadCurrentLevel, και στη συνέχεια ξεκινάει τη διαδικασία της ασύγχρονης φόρτωσης του επιλεγμένου περιβάλλοντος.

currentHandle = parkingLevels[level].levelObject.LoadAssetAsync<GameObject>(); : Η μέθοδος **LoadAssetAsync<GameObject>()** επιστρέφει handle, το οποίο παρακολουθεί την πορεία της φόρτωσης και επιτρέπει την πρόσβαση στο αντικείμενο μόλις αυτή ολοκληρωθεί.

currentHandle.Completed += handle => : ορίζει τι θα συμβεί όταν ολοκληρωθεί η ασύγχρονη φόρτωση που ξεκίνησε με το LoadAssetAsync. Συγκεκριμένα, χρησιμοποιεί το event Completed, στο οποίο προστίθεται ενα callback το οποίο ορίζεται στη συνέχεια.

currentEnvironmentInstance = Instantiate(handle.Result, Vector3.zero, Quaternion.identity); : εμφανίζει το αναζητούμενο αντικείμενο στην ιεραρχία της σκηνής, και ορίζει το αντικείμενο currentEnvironmentInstance, για να μπορεί να διαχειριστεί από το λογισμικό στη συνέχεια.

Destroy(currentEnvironmentInstance); : διαγράφει από τη μνήμη το τρέχον περιβάλλον στο τέλος του τρέχοντος κύκλου του προγράμματος.

currentEnvironmentInstance = null; : θέτει την τιμή null στο αντικείμενο. Είναι απαραίτητο να μηδενιστεί η τιμή του πριν τη λήξη του τρέχοντος κύκλου του προγράμματος, για αυτήν την περίπτωση το Destroy() δεν είναι επαρκές από μόνο του.

Addressables.Release(currentHandle); : αποδεσμεύει το τρέχον handle από τα περιεχόμενα του ώστε να μπορεί να χρησιμοποιηθεί για τη φόρτωση του επόμενου περιβάλλοντος.

```
private GameObject currentEnvironmentInstance;
private AsyncOperationHandle < GameObject > currentHandle;

public void switchLevelEnvironment(int level) {
  unloadCurrentLevel();
  currentHandle = parkingLevels[level].levelObject.LoadAssetAsync <GameObject >
();
  currentHandle.Completed += handle => {
    if (handle.Status == AsyncOperationStatus.Succeeded) {
      currentEnvironmentInstance = Instantiate(handle.Result, Vector3.zero,
Quaternion.identity);
    } else {
      Debug.LogError($"Failed to load level {level}");
    }
  };
}
public void unloadCurrentLevel() {
  if (currentEnvironmentInstance != null) {
    Destroy(currentEnvironmentInstance);
    currentEnvironmentInstance = null;
  }
  if (currentHandle.IsValid()) {
    Addressables.Release(currentHandle);
  }
}
```

Η επιλογή των επιπέδων δυσκολίας μπορεί να πραγματοποιηθεί με δύο τρόπους. Αφενός, ο οδηγός έχει τη δυνατότητα να επιλέξει απευθείας το επιθυμητό επίπεδο μέσα από την οθόνη επιλογών του συγκεκριμένου κεφαλαίου, στο κάτω δεξί μέρος της διεπαφής. Αφετέρου, μετά την ολοκλήρωση ενός επιπέδου παρέχεται η δυνατότητα αυτόματης μετάβασης στο επόμενο, διευκολύνοντας τη σταδιακή πρόοδο και την ομαλή αύξηση της δυσκολίας. Με τον τρόπο αυτό συνδυάζεται η ελευθερία της επιλογής με την οργανωμένη κλιμάκωση της μαθησιακής εμπειρίας.

2.1.3.3 Ανάλυση των μηχανισμών αξιολόγησης στάθμευσης

Η αξιολόγηση της απόδοσης του οδηγού αποτελεί κρίσιμο στοιχείο στην προσομοίωση στάθμευσης, καθώς επιτρέπει την αντικειμενική μέτρηση των δεξιοτήτων του και την παροχή ανατροφοδότησης

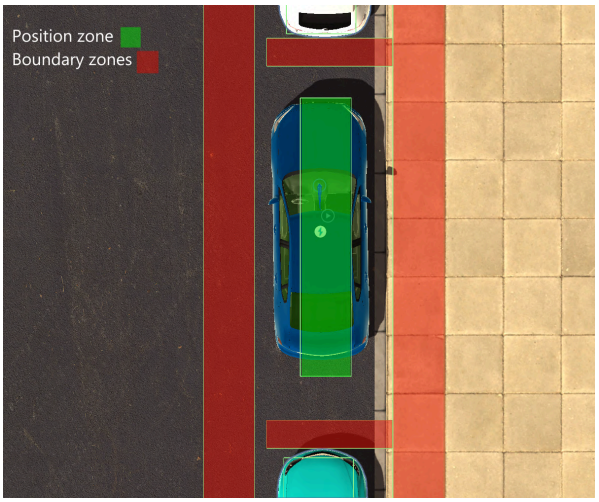
για βελτίωση. Στην ενότητα αυτή παρουσιάζονται οι τέσσερις βασικοί μηχανισμοί αξιολόγησης που έχουν ενσωματωθεί στο κεφάλαιο. Ο βασικός μηχανισμός ελέγχει κατά πόσο σωστά έχει τοποθετηθεί το όχημα εντός του χώρου στάθμευσης, λαμβάνοντας υπόψη τη θέση και τον προσανατολισμό του. Ο δεύτερος καταμετρά τον αριθμό των κινήσεων που πραγματοποιεί ο οδηγός κατά τη διαδικασία της στάθμευσης, αξιολογώντας την ικανότητα διαχείρισης αλλαγών κατεύθυνσης και ελιγμών. Ο τρίτος μηχανισμός παρακολουθεί τον χρόνο ολοκλήρωσης της στάθμευσης, διασφαλίζοντας ότι ο οδηγός εκτελεί τη διαδικασία μέσα σε ένα προκαθορισμένο χρονικό όριο, γεγονός ιδιαίτερα σημαντικό σε πραγματικές συνθήκες, όπως σε πολυσύχναστους αστικούς χώρους. Τέλος, ο τέταρτος μηχανισμός ανιχνεύει συγκρούσεις με τα παρακείμενα οχήματα και στοιχεία του περιβάλλοντος, επιτρέποντας την αξιολόγηση της ικανότητας του οδηγού να αποφύγει επαφές με αντικείμενα που θα μπορούσαν να προκαλέσουν ζημιές. Οι μηχανισμοί αξιολόγησης λειτουργούν ανεξάρτητα ο ένας από τον άλλον και έχουν αναπτυχθεί με αρθρωτή (modular) προσέγγιση, επιτρέποντας την εύκολη προσαρμογή τους σε διαφορετικά επίπεδα δυσκολίας. Με αυτόν τον τρόπο, είναι δυνατή η ρύθμιση διαφορετικών απαιτήσεων για κάθε επίπεδο, όπως η ακρίβεια της στάθμευσης, ο μέγιστος αριθμός κινήσεων και ο διαθέσιμος χρόνος ενισχύοντας την ευελιξία και την κλιμακωτή πρόκληση για τον χρήστη.

Υπάρχουν συνολικά τρεις τρόποι για να αποτύχει μια απόπειρα στάθμευσης.

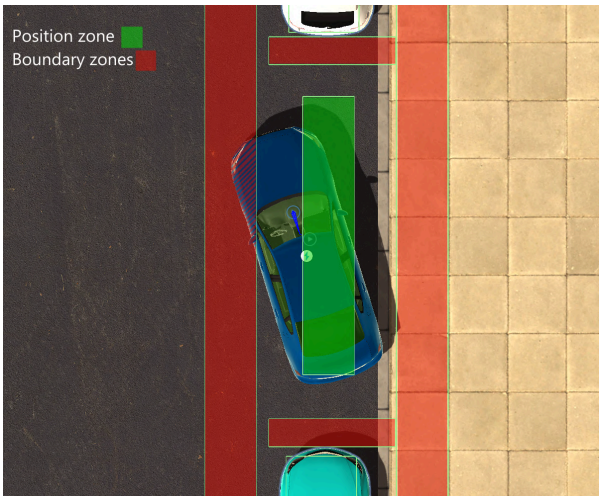
- Ο πρώτος τρόπος είναι όταν το όχημα συγκρουστεί με άλλο αυτοκίνητο, καταγράφοντας άμεσα την αποτυχία
- Ο δεύτερος τρόπος αφορά τη χρονική πίεση: εάν ο οδηγός υπερβεί τον μέγιστο διαθέσιμο χρόνο για την ολοκλήρωση της στάθμευσης, η προσπάθεια θεωρείται αποτυχημένη
- Ο τρίτος τρόπος αποτυχίας σχετίζεται με τον αριθμό των κινήσεων του οχήματος: αν ο οδηγός πραγματοποιήσει περισσότερους ελιγμούς από το μέγιστο όριο που έχει οριστεί για το τρέχον επίπεδο δυσκολίας, η προσπάθεια καταγράφεται επίσης ως αποτυχημένη

Ορισμένοι από τους μηχανισμούς αξιολόγησης που περιγράφονται σε αυτό το κεφάλαιο αξιοποιούν τους trigger colliders της Unity. Ένας trigger collider αποτελεί έναν ειδικό τύπου collider που δεν επηρεάζει φυσικά την κίνηση των αντικειμένων, αλλά επιτρέπει την ανίχνευση επαφών μεταξύ αντικειμένων. Τα trigger colliders χρησιμοποιούνται για την ανίχνευση επαφής του οχήματος με συγκεκριμένες περιοχές, χωρίς να προκαλούν σύγκρουση ή απόκλιση στην κίνηση. Η λειτουργικότητα τους μπορεί να αξιοποιηθεί μέσω τριών βασικών συναρτήσεων της Unity: OnTriggerEnter, OnTriggerStay και OnTriggerExit. Η OnTriggerEnter καλείται τη στιγμή που ένα άλλο collider εισέρχεται στην περιοχή του trigger, η OnTriggerStay εκτελείται κάθε frame όσο το αντικείμενο παραμένει εντός του trigger, ενώ η OnTriggerExit ενεργοποιείται όταν το αντικείμενο εγκαταλείψει την περιοχή. Μέσω αυτών των συναρτήσεων, οι μηχανισμοί αξιολόγησης μπορούν να παρακολουθούν σε πραγματικό χρόνο τη θέση και τη συμπεριφορά του οχήματος, ελέγχοντας με ακρίβεια τις αλληλεπιδράσεις με τα όρια του χώρου στάθμευσης και άλλων σχετικών αντικειμένων.

Ο κύριος μηχανισμός αξιολόγησης που περιγράφεται σε αυτό το κεφάλαιο, αυτός ο οποίος καθορίζει την ακρίβεια με την οποία έχει σταθμευτεί το όχημα στον επιθυμητό χώρο, αξιοποιεί τους προαναφερθέντες trigger colliders για την ανίχνευση της παρουσίας ή απουσίας του οχήματος σε συγκεκριμένες περιοχές. Με βάση τα δεδομένα που συλλέγονται από αυτά τα colliders, ο μηχανισμός μπορεί να αξιολογήσει με ακρίβεια την τοποθέτηση του οχήματος, ελέγχοντας αν βρίσκεται εντός του επιτρεπόμενου χώρου στάθμευσης χωρίς να παραβιάζει τα όρια της σταθμευσης. Η πληροφορία αυτή χρησιμοποιείται για να εκδοθεί η τελική κρίση κατα πόσο η διαδικασία στάθμευσης έχει ολοκληρωθεί με επιτυχία ή όχι.



Εικόνα 2.1.3.3.1: Επιτυχής ολοκλήρωση διαδικασίας στάθμευσης



Εικόνα 2.1.3.3.2: Ανεπιτυχής ολοκλήρωση διαδικασίας στάθμευσης

Υλοποιήθηκε η κλάση TriggerHandler στην οποία πραγματοποιείται η διαχείριση των αλληλεπιδράσεων με τους trigger colliders, εξασφαλίζοντας ότι η τοποθέτηση του οχήματος μεταξύ των δύο σταθμευμένων αυτοκινήτων αξιολογείται με συνέπεια και ακρίβεια. Για κάθε περιβάλλον επιπέδου έχει υλοποιηθεί ένας ανεξάρτητος μηχανισμός αξιολόγησης αυτού του τύπου, ο οποίος είναι συνδεδεμένος με την αντίστοιχη ιεραρχία αντικειμένων του περιβάλλοντος, ώστε να προσαρμόζεται στις εκάστοτε παραμέτρους δυσκολίας και να παρέχει έγκυρη ανατροφοδότηση για την επιτυχία ή μη της στάθμευσης.

Στο σύστημα αξιολόγησης της στάθμευσης χρησιμοποιούνται δύο κατηγορίες αντικειμένων με colliders: τα position zones και τα boundary zones. Το position zone ορίζει τη γενική περιοχή στην οποία πρέπει να τοποθετηθεί το όχημα ώστε να θεωρηθεί σωστά σταθμευμένο, ενώ τα boundary zones αντιπροσωπεύουν τα όρια της θέσης στάθμευσης, τόσο στα αριστερά και δεξιά του οχήματος όσο και στο εμπρόσθιο και οπίσθιο τμήμα του. Για να χαρακτηριστεί η στάθμευση επιτυχής, το όχημα πρέπει να βρίσκεται σε επαφή με το position zone, χωρίς όμως να έρχεται σε επαφή με κάποιο από τα

boundary zones. Η κάθε θέση στάθμευσης περιέχει 1 position zone και 4 boundary zones. Δύο στα πλάγια, τα οποία βεβαιώνουν ότι το όχημα βρίσκεται παράλληλα στα αυτοκίνητα ανάμεσα στα οποία σταθμεύει και δυο μπροστά και πίσω από αυτά, ώστε να διατηρηθεί η κατάλληλη απόσταση από τα οχήματα αυτά. Το επίπεδο δυσκολίας ρυθμίζεται δυναμικά μέσω της απόστασης των boundary zones από το όχημα. όσο πιο κοντά στο όχημα τοποθετούνται, τόσο μεγαλύτερη ακρίβεια απαιτείται στους ελιγμούς, αυξάνοντας αναλόγως την πρόκληση για τον χρήστη. Σε κάθε περιβάλλον υπάρχουν δύο αντικείμενα που συνδέονται με την κλάση triggerHandler. Ένα με ονομασία positionZone και ένα με ονομασία boundaryZone. Στην κλάση parkingController ορίζονται δύο αντικείμενα τύπου triggerHandler, το currentPositionZoneTriggerHandler και το currentBoundaryZoneTriggerHandler απο τα οποία αντλούνται δεδομένα σχετικα με την κατάσταση της διαδικασίας της στάθμευσης στο κύριο πρόγραμμα. Κατά την εναλλαγή περιβάλλοντος, το προηγούμενο περιβάλλον αποδεσμεύεται από τη μνήμη. Κατά τη διάρκεια αυτής της διαδικασίας, η κλάση parkingController θέτει τα αντικείμενα διαχείρισης trigger της κλάσης σε null, ώστε να μην παραμένουν αναφορές σε ανενεργά στοιχεία. Στη συνέχεια, με τη φόρτωση του νέου περιβάλλοντος, κάθε ένα από τα δύο instances της κλάσης TriggerHandler εκχωρεί τον εαυτό του ως τιμή στα αντίστοιχα αντικείμενα διαχείρισης trigger της κλάσης parkingController, διασφαλίζοντας έτσι τη σωστή σύνδεση του μηχανισμού αξιολόγησης με το εκάστοτε ενεργό περιβάλλον.

Ανάλυση της λειτουργίας της κλάσης triggerHandler:

- Αρχικά εκτελείται η συνάρτηση Awake η οποία μετρώντας τον αριθμό από τα colliders που σχετίζονται με το αντικείμενο αποφασίζει κατά πόσο το συγκεκριμένο intance ειναι positionZone ή boundaryZone και θέτει τη μεταβλητή isPositionZone
- Εκτελείται η συνάρτηση setTriggerHandlers και ανάλογα με την κατηγορία του, το instance θέτει τον εαυτό του ως αντικείμενο ελέγχου trigger στην κλάση parkingController
- private void OnTriggerEnter(Collider other) : θέτει τη μεταβλητη isWithin σε true όταν ένα άλλο collider (του οχήματος) εισέρχεται στην περιοχή του
- private void OnTriggerStay(Collider other) : όσο υπάρχει το collider στη ζώνη η μεταβλητή isWithin παραμένει θετική
- private void OnTriggerExit(Collider other) : όταν ένα collider αναχωρεί από την περιοχή η μεταβλητή isWithin παίρνει αρνητική τιμή

Ο τελικός σκοπός αυτής της κλάσης είναι το κάθε instance της να παρέχει στην κλάση parkingController πρόσβαση στην τιμή isWithin στα αντίστοιχα αντικείμενα ελέγχου trigger

```
public class triggerHandler : MonoBehaviour
{
    public bool isWithin;
    public bool isPositionZone;

    private void Awake() {
        BoxCollider[] colliders = GetComponents<BoxCollider>();
        if (colliders.Length > 1){
            isPositionZone = false;
        }
        else {
            isPositionZone = true;
        }
        if (parkingController.instance != null) {
```

```
            setTriggerHandlers();
        }
    }

    void setTriggerHandlers() {
        if (isPositionZone){
          parkingController.instance.currentPositionZoneTriggerHandler = this;
        }
        else {
          parkingController.instance.currentBoundaryZoneTriggerHandler = this;
        }
    }

    private void OnTriggerEnter(Collider other){
        isWithin = true;
    }
    private void OnTriggerStay(Collider other){
        isWithin = true;
    }
    private void OnTriggerExit(Collider other){
        isWithin = false;
    }
}
```

Έλεγχος της στάθμευσης από την κλάση parkingController

Μετά τη φόρτωση του περιβάλλοντος και την ορθή συσχέτιση του μηχανισμού αξιολόγησης με την κλάση parkingController, η σύνδεση μεταξύ των trigger colliders και της κλάσης parkingController έχει πλέον εδραιωθεί. Στο σημείο αυτό, η κλάση parkingController αξιοποιεί την boolean μεταβλητή isWithin για να ελέγξει την ορθότητα της στάθμευσης. Ο έλεγχος αυτός πραγματοποιείται εντός του βρόχου Update, ο οποίος εκτελείται σε κάθε frame, εξασφαλίζοντας έτσι συνεχή αξιολόγηση της θέσης του οχήματος.

Ανάλυση του βρόχου Update της κλάσης parkingController:

- Η μέθοδος Update() εκτελείται σε κάθε frame, εφόσον η μεταβλητή isLive είναι αληθής (δηλαδή η προσομοίωση είναι ενεργή).
- Ελέγχεται αν υπάρχουν ενεργά τα δύο **TriggerHandlers** (position και boundary). Αν δεν υπάρχουν, η μέθοδος επιστρέφει χωρίς να κάνει κάτι.

Στο επόμενο if statement αναζητεί τρία κριτήρια:

- Αν το όχημα βρίσκεται **εντός του position zone**
- Αν το όχημα δεν βρίσκεται σε επαφή με τα boundary zones
- Αν το τιμόνι είναι πρακτικά σε ευθεία θέση (steering ≈ 0)

Αν ισχύουν καί τα τρία η στάθμευση θεωρείται επιτυχής και εκτελούνται οι επόμενες γραμμές:

- σταματά η ροή του χρόνου στην προσομοίωση (Time.timeScale = 0)
- ξεκινά μια ρουτίνα που επαναφέρει τις παραμέτρους του οχήματος
- εμφανίζεται το γραφικό στοιχείο επιτυχίας (successLayout)
- το flag isLive παιρνει αρνητική τιμή

- Οι μεταβλητές isWithin των TriggerHandlers επαναφέρονται σε κατάσταση false για καθαρή επανεκκίνηση

```
void Update() {
  if (isLive) {
    if (!(currentPositionZoneTriggerHandler &&
currentBoundaryZoneTriggerHandler)) {
      return;
    }
    if (currentPositionZoneTriggerHandler.isWithin &&
!currentBoundaryZoneTriggerHandler.isWithin &&
(vehicleController.instance.steering < 0.1 f &&
vehicleController.instance.steering > -0.1 f)) {
      Time.timeScale = 0;
      StartCoroutine(resetVehicleParamsOnSimulationPause());
      sucessLayout.SetActive(true);
      isLive = false;
      currentPositionZoneTriggerHandler.isWithin = false;
      currentBoundaryZoneTriggerHandler.isWithin = false;
    }
  }
}
```

Καταμέτρηση κινήσεων:

Σε κάθε δομή τύπου parkingLevel υπάρχει η μεταβλητή maxAmountOfMoves που ορίζει τον μέγιστο επιτρεπτό αριθμό κινήσεων για να πραγματοποιηθεί η στάθμευση. Στην κλάση parkingController δηλώνεται η μεταβλητή maxMovesForCurrentLevel η οποία ορίζει τον μέγιστο αριθμό κινήσεων για το τρέχον επίπεδο, και η μεταβλητή nOfMoves η οποία δηλώνει τον αριθμό κινήσεων που εχουν ηδη πραγματοποιηθεί.

- Η καταμέτρηση των κινήσεων του οδηγού υλοποιείται μέσω του event gearShiftEvent, το οποίο πυροδοτείται σε κάθε αλλαγή σχέσης. Στην κλάση parkingController, η συνάρτηση listenForDirectionChange εγγράφεται στο event κατά την εκκίνηση του προγράμματος.

```
private void Start()
{
  nOfMoves = 0;
  maxMovesForCurrentLevel = parkingLevels[level].maxAmountOfMoves;
  vehicleController.instance.gearShiftEvent += listenForDirectionChange;
}
```

- Η συνάρτηση listenForDirectionChange ελέγχει αν η αλλαγή ταχυτήτων αφορά μετάβαση από την πρώτη στην έκτη (η έκτη αντιπροσωπεύει την όπισθεν) ή το αντίστροφο. Σε μια τέτοια περίπτωση, η boolean μεταβλητή directionChangeFlag αντιστρέφεται, σηματοδοτώντας πιθανή αλλαγή κατεύθυνσης

```
    private void listenForDirectionChange(int from,int to){
        if(from ==1||from == 6){
            if (to == 1 || to == 6){
```

```
                directionChangeFlag = !directionChangeFlag;
            }
        }
    }
```

- Στη συνάρτηση Update της κλάσης parkingController ελέγχεται σε κάθε κύκλο η μεταβλητή directionChangeFlag. Όταν αυτή ειναι αληθής και ανιχνευθεί πάτημα του πεντάλ γκαζιού, το σύστημα θεωρεί ότι ο οδηγός ξεκινά κίνηση προς τη νέα κατεύθυνση. Τότε εκτελείται η συνάρτηση directionChanged, και η μεταβλητή directionChangeFlag επαναφέρεται σε false.

```
void Update()
{
    if (isLive)
    {
        if (directionChangeFlag)
        {
            if (vehicleController.instance.gas > 0.1f)
            {
                directionChangeFlag = false;
                directionChanged();
            }
        }
   }
}
```

- Στη συνάρτηση directionChanged η μεταβλητή nOfMoves αυξάνεται κατά μια μονάδα και ελέγχεται κατά πόσο έχει υπερβεί το μέγιστο επιτρεπτό όριο που αντιστοιχεί στο εκάστοτε επίπεδο δυσκολίας. Αν ο οδηγός έχει υπερβεί το όριο κινήσεων ενεργοποιείται ο μηχανισμός αποτυχίας της στάθμευσης.

```
public void directionChanged()
{
    nOfMoves++;
    updateSuccessMessage();

    if (maxMovesForCurrentLevel >= 100){
        nOfMovesValueUIRef.text = nOfMoves.ToString() + " / " + "∞";
    }
    else{
        nOfMovesValueUIRef.text = nOfMoves.ToString() + " / " +
maxMovesForCurrentLevel;
    }

    if (nOfMoves > maxMovesForCurrentLevel)
    {
        Time.timeScale = 0;
        StartCoroutine(resetVehicleParamsOnSimulationPause());
        updateUI_State(uiState.FAILURE);
        UpdateFailureLayout(reasonOfFailure.EXCEEDED_MAX_N_OF_MOVES);
```

```
    }
}
```

Ένα επιπλέον στοιχείο αξιολόγησης της στάθμευσης αφορά τον χρονικό περιορισμό, ο οποίος υλοποιείται μέσω συνάρτησης που θέτει ένα χρονόμετρο με βάση τη μεταβλητή maxTimeInSeconds της δομής parkingLevel. Όταν ο προκαθορισμένος χρόνος λήξει χωρίς να έχει ολοκληρωθεί επιτυχώς η στάθμευση, ενεργοποιείται αυτόματα μηχανισμός αποτυχίας. Η λειτουργία αυτή είναι ιδιαίτερα σημαντική, καθώς αντικατοπτρίζει ρεαλιστικές συνθήκες της καθημερινής οδήγησης, όπου ο διαθέσιμος χρόνος για στάθμευση μπορεί να είναι περιορισμένος, όπως σε περιοχές με έντονη κυκλοφορία σε πολυσύχναστους αστικούς χώρους. Με τον τρόπο αυτό, η προσομοίωση ενσωματώνει μια επιπλέον διάσταση δυσκολίας και ρεαλισμού, καλλιεργώντας στον οδηγό την ικανότητα να σταθμεύει όχι μόνο με ακρίβεια αλλά και με ταχύτητα. Το χρονόμετρο υλοποιείται στη συνάρτηση startCoundown, η οποία ξεκινάει ένα coroutine το οποίο χειρίζεται τη μέτρηση του χρόνου.

- Η συνάρτηση startCountdown() ξεκινά ένα coroutine χρονομετρητή. Αν τρέχει ήδη κάποιος, τον σταματά πριν ξεκινήσει καινούργιο.
- Στο coroutine Countdown() μετράει τον χρόνο αντίστροφα από τη μεταβλητή currentTimeInSeconds, ενημερώνοντας το UI και μετατρέποντας τον χρόνο σε λεπτά και δευτερόλεπτα.
- Αν ο χρόνος είναι μεγαλύτερος από 300 δευτερόλεπτα, εμφανίζεται το σύμβολο "∞" και ο χρονομετρητής τερματίζεται. Σε αυτήν την περίπτωση ο χρόνος θεωρείται απεριόριστος.
- Κάθε δευτερόλεπτο μειώνεται ο χρόνος κατά ένα, μέσω της συνάρτησης WaitForSeconds(1f). Όταν φτάσει στο μηδέν, εμφανίζεται "00:00" και καλείται η συνάρτηση OnTimerEnd().
- Η συνάρτηση OnTimerEnd() σταματά τον χρόνο της προσομοίωσης (Time.timeScale = 0), επαναφέρει τις παραμέτρους του οχήματος, ενημερώνει το UI με αποτυχία λόγω χρόνου (OUT_OF_TIME) και απενεργοποιεί την προσομοίωση (isLive = false).

```
public void startCountdown() {
  if (countdownCoroutine != null) {
    StopCoroutine(countdownCoroutine);
  }
  countdownCoroutine = StartCoroutine(Countdown());
}
private IEnumerator Countdown() {
  while (currentTimeInSeconds > 0) {
    if (currentTimeInSeconds > 300) {
      timerText.text = "∞";
      countdownCoroutine = null;
      yield break;
    }
    int minutes = currentTimeInSeconds / 60;
    int seconds = currentTimeInSeconds % 60;
    timerText.text = string.Format("{0:00}:{1:00}", minutes, seconds);

    yield return new WaitForSeconds(1 f);
    currentTimeInSeconds--;
  }
  timerText.text = "00:00";
```

```
  OnTimerEnd();
}
private void OnTimerEnd() {
  Time.timeScale = 0;
  StartCoroutine(resetVehicleParamsOnSimulationPause());
  UpdateFailureLayout(reasonOfFailure.OUT_OF_TIME);
  updateUI_State(uiState.FAILURE);
  isLive = false;
}
```

Διαχείριση συγκρούσεων:

Ένας κρίσιμος μηχανισμός αξιολόγησης είναι ο μηχανισμός ανίχνευσης συγκρούσεων μεταξύ του οχήματος του οδηγού και των γειτονικών σταθμευμένων αυτοκινήτων. Η λειτουργία αυτή βασίζεται στη χρήση colliders, τα οποία επιτρέπουν την ακριβή αναγνώριση φυσικών επαφών στο εικονικό περιβάλλον. Ιδιαίτερη σημασία έχει η διάκριση των αντικειμένων με τα οποία μπορεί να συμβεί μια τέτοια σύγκρουση. Το σύστημα έχει σχεδιαστεί ώστε να καταγράφει επαφές μόνο με αντικείμενα που χαρακτηρίζονται ως collidable, όπως τα παρακείμενα οχήματα, και στοιχεία του περιβάλλοντος και όχι με αντικείμενα όπως ο ίδιος ο δρόμος. Η ανίχνευση σύγκρουσης γίνεται στην κλάση detectCollisionWithVehicle και ανιχνεύει συγκρούσεις μέσω του trigger collider το οποίο συσχετίζεται με το όχημα που ελέγχει ο οδηγός.

- Στην κλάση detectCollisionWithVehicle ορίζεται ένα event με όνομα collided, το οποίο ενεργοποιείται όταν ανιχνευθεί σύγκρουση.
- Στη μέθοδο Awake(), αποθηκεύει τον εαυτό της στη στατική μεταβλητή instance, ώστε να μπορεί να προσπελαστεί από άλλα μέρη του προγράμματος (singleton pattern).
- Η μέθοδος OnTriggerEnter(Collider other) καλείται αυτόματα όταν το collider του οχήματος εισέρχεται σε επαφή με άλλο collider.
- Η συνάρτηση OnTriggerEnter ελέγχει αναδρομικά αν το αντικείμενο ή κάποιος γονέας του έχει το tag "collideable". Αν ναι, ενεργοποιεί το event collided, σηματοδοτώντας ότι υπήρξε σύγκρουση με ένα αντικείμενο που θεωρείται συγκρουόμενο

```
public class detectCollisionWithVehicle: MonoBehaviour {
  public event Action collided;
  public static detectCollisionWithVehicle instance;

  private void Awake() {
    instance = this;
  }
  private void OnTriggerEnter(Collider other) {
    Transform current = other.transform;
    while (current != null) {
      if (current.CompareTag("collideable")) {
        collided.Invoke();
        break;
      }
      current = current.parent;
    }
  }
```

```
}
```

Κατά την εκκίνηση της προσομοίωσης η συνάρτηση handleCollisionWithVehicle() προστίθεται στο event collided το οποίο ορίστηκε προηγουμένως.

- Η συνάρτηση σταματά τον χρόνο της προσομοίωσης (Time.timeScale = 0).
- Εκκινεί τη ρουτίνα επαναφοράς παραμέτρων του οχήματος (resetVehicleParamsOnSimulationPause()).
- Ενεργοποιεί την κάμερα ατυχήματος (switchToCrashCamera()), η οποία αλλάζει τη γωνία θέασης σε τρίτο πρόσωπο και από ψηλά. Αυτό είναι ιδιαίτερα σημαντικό καθώς επιτρέπει στον οδηγό να δει με ακρίβεια το σημείο της σύγκρουσης και να κατανοήσει πού ακριβώς έκανε το λάθος. Με αυτόν τον τρόπο, η προσομοίωση δεν περιορίζεται μόνο στην ένδειξη μιας αποτυχίας, αλλά παρέχει και ανατροφοδότηση (feedback) που βοηθά τον εκπαιδευόμενο να βελτιώσει την τεχνική του και να αποφύγει παρόμοια λάθη στο μέλλον.
- Ενημερώνει το UI ώστε να δείξει αποτυχία λόγω σύγκρουσης (COLLISION_WITH_VEHICLE).
- Θέτει το isLive σε false ώστε να τερματιστεί η τρέχουσα προσπάθεια

```
  private void handleCollisionWithVehicle() {
    if (isLive) {
      Time.timeScale = 0;
      StartCoroutine(resetVehicleParamsOnSimulationPause());
      switchToCrashCamera();
      UpdateFailureLayout(reasonOfFailure.COLLISION_WITH_VEHICLE);
      updateUI_State(uiState.FAILURE);
      isLive = false;
    }
  }
```

Σχήμα 2.1.3.3.3: Σε περίπτωση σύγκρουσης, η γωνία λήψης αλλάζει για να εμφανίσει το λάθος

Σχήμα 2.1.3.3.4: Επιτυχής στάθμευση

2.1.4 Αυτόματη δημιουργία οδικού περιβάλλοντος μέσω αλγόριθμου

Το παρόν κεφάλαιο αφορά την αυτόματη δημιουργία ενός οδικού περιβάλλοντος μέσω αλγορίθμων για την εξάσκηση των δεξιοτήτων ενός υποψήφιου οδηγού. Η υιοθέτηση μιας διαδικαστικής προσέγγισης στη δημιουργία των περιβαλλόντων της προσομοίωσης επιτρέπει την παραγωγή πρακτικά απεριόριστων σεναρίων οδήγησης, προσφέροντας στον εκπαιδευόμενο πολλές και ποικίλες

καταστάσεις εξάσκησης σε ένα ελεγχόμενο περιβάλλον. Αντί για χειροποίητες, στατικές σκηνές, το σύστημα παράγει αστικές περιοχές, οδικά δίκτυα, κτίρια (εμπορικά και κατοικήσιμα), σήμανση, σταθμευμένα οχήματα καθώς και κινούμενη κυκλοφορία. Τα χαρακτηριστικά του περιβάλλοντος είναι ρυθμιζόμενα από παραμέτρους και έναν "σπόρο" (seed). Το seed είναι ένας τυχαίος κωδικός αποτελούμενος από αριθμούς και γράμματα της λατινικής αλφαβήτου, πάνω στον οποίο βασίζεται το σύστημα τυχαιότητας για να μπορεί να παράξει ένα και μοναδικό περιβάλλον για κάθε τιμή του seed, διασφαλίζοντας έτσι την αναπαραγωγιμότητα στα περιβάλλοντα που δημιουργούνται.

Κύριοι στόχοι και απαιτήσεις

- Ρεαλισμός: Το παραγόμενο περιβάλλον πρέπει να προσεγγίζει οπτικά και λειτουργικά ένα πραγματικό αστικό τοπίο. κατάλληλη τοποθέτηση οδών, λογική διάταξη οικοδομικών τετραγώνων, σωστή τοποθέτηση σημάτων και λοιπά στοιχεία δρόμου, ποικιλία κτιρίων (εμπορικά πρόσοψης, πολυκατοικίες, μονοκατοικίες) και ρεαλιστική τοποθέτηση σταθμευμένων οχημάτων.
- Απόδοση: Ο αλγόριθμος πρέπει να δημιουργεί περιβάλλοντα γρήγορα και αποδοτικά, με τεχνικές όπως ασύγχρονη ροή, χρήση LOD, pooling αντικειμένων, caching και εκτεταμένη χρήση προκατασκευασμένων prefab templates για μείωση του κόστους runtime.
- Επεκτασιμότητα: Η αρχιτεκτονική πρέπει να είναι modular και data-driven. Τμήματα του προγράμματος για παραγωγή δρόμων, κτιρίων, σημάνσεων, οχημάτων κ.λπ. που μπορούν να επεκταθούν ή να αντικατασταθούν ανεξάρτητα. Οι κανόνες δημιουργίας, τα πρότυπα κτιρίων και τα σενάρια να ορίζονται μέσω Scriptable Objects ώστε να προστεθούν νέες δυνατότητες χωρίς βαθειά τροποποίηση των υπαρχόντων υποδομών.
- Κινούμενα και σταθμευμένα οχήματα: Το περιβάλλον πρέπει να περιλαμβάνει, σταθμευμένα και κινούμενα οχήματα. Τα κινούμενα οχήματα πρέπει να ακολουθούν ρεαλιστικές συμπεριφορές (δρομολόγια, ταχύτητες, προτεραιότητες, προσαρμογή σε φωτεινή σηματοδότηση και σε ροή κυκλοφορίας), ενώ τα σταθμευμένα πρέπει να τοποθετούνται με λογική (χώροι στάθμευσης, παράλληλη/ κάθετη στάθμευση κ.λπ.).

Η βασική ιδέα πίσω από τον αλγόριθμο διαδικαστικής δημιουργίας περιβάλλοντος είναι η χρήση modular αντικειμένων, τα οποία έχουν σχεδιαστεί ώστε να μπορούν να συνδεθούν μεταξύ τους σύμφωνα με προκαθορισμένους κανόνες. Το εικονικό περιβάλλον οργανώνεται σε πίνακες, και κάθε κελί αυτών γεμίζεται με περιεχόμενο βάσει των κανόνων του αλγορίθμου. Με αυτόν τον τρόπο εξασφαλίζεται τόσο η ποικιλία όσο και η συνέπεια στη διάταξη του περιβάλλοντος.

Το πλέγμα του περιβάλλοντος υλοποιείται ως δισδιάστατος πίνακας (2D array), του οποίου κάθε στοιχείο είναι αντικείμενο τύπου Node. Η κλάση Node επεκτείνει το ScriptableObject της Unity, ώστε να επιτρέπει την εύκολη διαχείριση και αποθήκευση των κόμβων ως ανεξάρτητα δεδομένα. Κάθε κόμβος περιέχει τα εξής πεδία:

- Όνομα (name): Χρησιμοποιείται για αναγνώριση και καλύτερη διαχείριση του κόμβου.
- Prefab (prefab): Περιλαμβάνει τα τρισδιάστατα αντικείμενα που αντιστοιχούν στον συγκεκριμένο κόμβο, όπως το τμήμα του δρόμου, τα πεζοδρόμια ή άλλα σχετικά στοιχεία.
- Περιστροφή (rotation): Ένας ακέραιος που καθορίζει τον προσανατολισμό του κόμβου στον άξονα Z. Με αυτόν τον τρόπο, το ίδιο αντικείμενο μπορεί να χρησιμοποιηθεί σε διαφορετικούς προσανατολισμούς.
- Συνδέσεις (connections[]): Ένα λογικό (boolean) τεσσάρων στοιχείων, το οποίο περιγράφει σε ποιες κατευθύνσεις μπορεί να συνδεθεί ο κόμβος.
    - connections[0]: επάνω

- connections[1]: δεξιά
- connections[2]: κάτω
- connections[3]: αριστερά

  Για να θεωρηθούν δύο κόμβοι συμβατοί, πρέπει οι αντίστοιχες κατευθύνσεις σύνδεσης να ταιριάζουν. Για παράδειγμα, το connections[0] (επάνω) ενός κόμβου πρέπει να είναι ίσο με το connections[2] (κάτω) του γειτονικού κόμβου. Με αυτόν τον μηχανισμό εξασφαλίζεται ότι οι δρόμοι συνεχίζουν ομαλά και δεν δημιουργούνται «σπασίματα» ή ασυνεχείς συνδέσεις στο οδικό δίκτυο.

```
public class Node : ScriptableObject
{
    public string name;
    public GameObject prefab;
    public int rotation;
    public bool[] connections;
}
```

Συνολικά υπαρχουν 4 διαφορετικα είδη από nodes:

RoadStraight: παρουσιάζει έναν ευθύ δρόμο

RoadTurn: παρουσιάζει στροφή 90 μοιρών

RoadT: παρουσιάζει δύο στροφές 90 μοιρών, μια δεξιά και μία αριστερά

RoadCross: παρουσιάζει διασταύρωση τριών κατευθύνσεων

Ανατομία του αντικειμένου prefab:

Κάθε prefab που χρησιμοποιείται για την αναπαράσταση ενός κόμβου στο οδικό δίκτυο αποτελείται από μια σειρά βασικών συστατικών, τα οποία εξυπηρετούν διαφορετικές λειτουργίες:

- 3D μοντέλο του δρόμου: Το κύριο οπτικό αντικείμενο που καθορίζει την εμφάνιση του συγκεκριμένου τμήματος του δρόμου, συμπεριλαμβανομένων των πεζοδρομίων, των διαγραμμίσεων και άλλων στατικών στοιχείων.
- Καμπύλες: Ορίζουν πορείες πάνω στο οδικό τμήμα, στις οποίες κινούνται τα AI οχήματα. Η διαχείριση των καμπυλών γίνεται μέσω της κλάσης entry_manager, η οποία αποθηκεύει και οργανώνει τις καμπύλες ώστε να μπορούν να διαβαστούν και να χρησιμοποιηθούν από την κλάση που χειρίζεται την κίνηση των AI οχημάτων. Ο ορισμός του σχήματος της καμπύλης γίνεται με την βοήθεια του εργαλείου που παρέχεται από την Unity3D με ονομασία “Splines”
- Transform markers: Πρόκειται για άδεια αντικείμενα, τα οποία λειτουργούν ως δείκτες θέσης και προσανατολισμού. Οι markers καθορίζουν πού μπορούν να τοποθετηθούν πρόσθετα αντικείμενα (π.χ. πινακίδες, κολώνες, δέντρα, παρκαρισμένα αυτοκίνητα) στο εκάστοτε prefab. Τη συλλογή και διαχείριση αυτών των σημείων αναλαμβάνει η κλάση roadBlockVariationObserver, η οποία επιτρέπει την εισαγωγή παραλλαγών στο ίδιο βασικό μπλοκ, ώστε το περιβάλλον να παραμένει διαφοροποιημένο και ρεαλιστικό.

Με αυτήν την προσέγγιση, κάθε prefab δεν είναι απλώς ένα στατικό κομμάτι δρόμου, αλλά ένα πολυδιάστατο δομικό στοιχείο, το οποίο ενσωματώνει τόσο τη γεωμετρία όσο και τις λειτουργικές πληροφορίες που χρειάζονται για τη ρεαλιστική προσομοίωση.

Ο αλγόριθμος αποτελείται από πέντε διακριτά στάδια, το καθένα από τα οποία συμβάλλει στην εισαγωγή διαφορετικών χαρακτηριστικών και λεπτομερειών. Το κάθε στάδιο υλοποιείται ως ένα coroutine (συνάρτηση IEnumerator) ώστε να μπορούν τα διάφορα στάδια να εκτελούνται το ένα μετά το άλλο σε σειρά. Ορίζεται το αντικείμενο mask, το οποίο είναι ένα list από αντικείμενα Vectro2Int τα οποία αποθηκεύουν τους δείκτες των κελιών που έχουν υπολογιστεί από προηγούμενο στάδιο για να μην υπολογιστούν ξανά.

- Στάδιο 1: Αρχική τοποθέτηση οικοδομικών τετραγώνων (Initial Block Placement).

  Στο πρώτο στάδιο, ο πίνακας υποδιαιρείται αναδρομικά σε οικοδομικά τετράγωνα με τυχαίο αλλά ελεγχόμενο τρόπο, ενώ τοποθετούνται διασταυρώσεις όπου είναι απαραίτητο. Το αποτέλεσμα είναι ένας χάρτης που αποτελείται αποκλειστικά από ορθογώνια τετράγωνα, χωρίς ακόμη μεγάλη ποικιλομορφία

- Στάδιο 2: Εισαγωγή παραλλαγής μέσω Perlin Noise.

  Για να αποφευχθεί η μονοτονία, εφαρμόζεται Perlin Noise ώστε να αφαιρεθούν ορισμένα τμήματα του χάρτη. Οι κενές αυτές περιοχές προορίζονται να γεμίσουν σε επόμενο στάδιο με διαφορετικά στοιχεία, ενισχύοντας έτσι τη φυσικότητα και τον ρεαλισμό.

- Στάδιο 3: Γέμισμα με Wave Function Collapse.

  Σε αυτό το στάδιο χρησιμοποιείται ο αλγόριθμος Wave Function Collapse, ο οποίος γεμίζει τα υπόλοιπα τετράγωνα με περιεχόμενο, βασισμένος τόσο στα γειτονικά τετράγωνα όσο και σε συγκεκριμένους κανόνες συμβατότητας. Έτσι επιτυγχάνεται μια ισορροπία ανάμεσα στην τυχαιότητα και τη συνοχή.

- Στάδιο 4: Ανίχνευση κενών και τοποθέτηση κτιρίων.

  Ο αλγόριθμος εντοπίζει κενές περιοχές στον πίνακα και καθορίζει πού μπορούν να τοποθετηθούν κτίρια. Για τη διάταξή τους χρησιμοποιείται και πάλι το Wave Function Collapse, αυτή τη φορά σε δύο διαστάσεις. Κάθε όροφος των κτιρίων υπολογίζεται ξεχωριστά, ώστε να αποδοθεί μεγαλύτερη λεπτομέρεια και ρεαλισμός.

2.1.4.1 Στάδιο 1: Αρχική τοποθέτηση οικοδομικών τετραγώνων

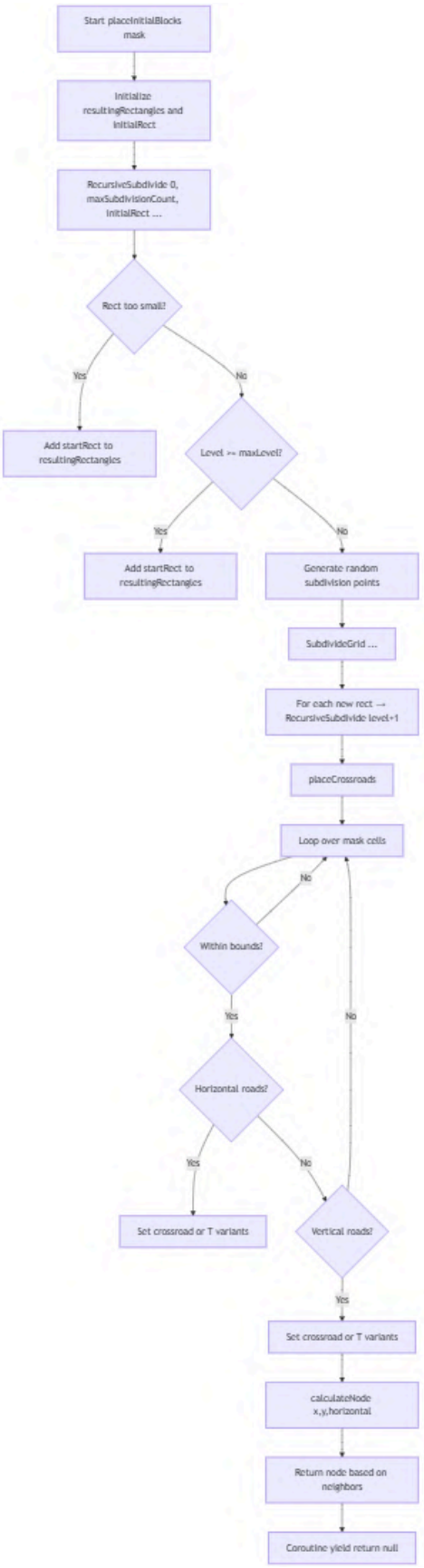
Η συνάρτηση placeInitialBlocks είναι υπεύθυνη για την αναδρομική υποδιαίρεση ενός πλέγματος σε μικρότερα ορθογώνια μπλοκ, με την προσθήκη τμημάτων δρόμων οριζόντια και κάθετα πάνω στο πλέγμα. Μετά την υποδιαίρεση, αναλαμβάνει την προσθήκη διασταυρώσεων στα σημεία όπου οι δρόμοι τέμνονται. Οι υποδιαιρέσεις γίνονται με τη χρήση του τύπου RectInt, ο οποίος περιέχει τα πεδία x, y, width, height και ορίζει ένα ορθογώνιο χρησιμοποιώντας δείκτες του πλέγματος. Η διαδικασία λειτουργεί αναδρομικά, και ελέγχεται από μεταβλητές οι οποίες ελέγχουν τον αριθμό των υποδιαιρέσεων και το ελάχιστο μέγεθος οικοδομικού τετραγώνου.

Κύρια σημεία του αλγορίθμου:

- Αρχικοποίηση: Ξεκινά με ολόκληρο το πλέγμα ως ένα ορθογώνιο και προετοιμάζει τη λίστα με ορθογώνια για την αποθήκευση των υποδιαιρέσεων.

```
List<RectInt> resultingRectangles = new List<RectInt>();
RectInt initialRect = new
RectInt(0,0,grid.GetLength(0),grid.GetLength(1));
```

- Υποδιαίρεση Πλέγματος μέσω συνάρτησης subdivideGrid: Δημιουργήθηκε η βοηθητική συνάρτηση subdivideGrid η οποία:
    - Μετατρέπει τα ποσοστιαία σημεία υποδιαίρεσης σε θέσεις στο πλέγμα.
    - Διαχωρίζει τα ορθογώνια οριζόντια και κατακόρυφα, τοποθετώντας τμήματα δρόμων κατά μήκος των ακμών.
    - Ενημερώνει τη λίστα mask με τις θέσεις των δρόμων.
- Τοποθέτηση Διασταυρώσεων: Δημιουργήθηκε η βοηθητική συνάρτηση placeCrossroads η οποία:
    - Διατρέχει όλες τις θέσεις δρόμων στη λίστα mask
    - Ελέγχει τα γειτονικά κελιά για τμήματα δρόμων
    - Αντικαθιστά τους ευθείς δρόμους με κατάλληλους κόμβους τύπου T ή διασταύρωσης.
- Αναδρομική Υποδιαίρεση:
    - Διαχωρίζει τα ορθογώνια σε μικρότερα έως ότου επιτευχθεί το ελάχιστο μέγεθος μπλοκ ή το μέγιστο βάθος αναδρομής.
    - Επιλέγονται τυχαία σημεία υποδιαίρεσης σε μορφή ποσοστού από 20% έως 80% του μήκους η πλάτους του ορθογώνιου



Σχήμα 2.1.4.1.1: Διάγραμμα υποδιαίρεσης πλέγματος

2.1.4.2 Στάδιο 2: Εισαγωγή παραλλαγής μέσω Perlin Noise

Το στάδιο αυτό αφορά την αφαίρεση ορισμένων τμημάτων του οδικού δικτύου με βάση Perlin noise, ώστε να δημιουργηθούν «κενές» περιοχές που θα αξιοποιηθούν σε επόμενο βήμα της διαδικασίας. Πιο συγκεκριμένα:

- Δημιουργείται ένα δισδιάστατο πλέγμα τιμών Perlin noise σε μορφή float.
- Οι τιμές του Perlin noise κανονικοποιούνται και συγκρίνονται με ένα threshold που ορίζεται κατά το στάδιο της ανάπτυξης
- Το πλέγμα μετατρέπεται σε πλέγμα boolean τιμών βάση του threshold
- Για κάθε κελί με τιμή true, το αντίστοιχο road block στο οδικό πλέγμα ορίζεται σε null, αφήνοντας ελεύθερο χώρο.

Με αυτόν τον τρόπο, η αρχική μονοτονία των ορθογώνιων blocks σπάει και δημιουργούνται περιοχές που μπορούν να γεμίσουν με διαφορετικά στοιχεία στα επόμενα στάδια, κάνοντας το περιβάλλον πιο διαφοροποιημένο.

```
private IEnumerator SubtractNoise(float scale, float threshold, List <
Vector2Int > mask) {
  bool[, ] noiseGrid = new bool[size, size];
  GenerateGrid();

  void GenerateGrid() {
    for (int x = 0; x < size; x++) {
      for (int y = 0; y < size; y++) {
        float xCoord = (float) x / size * scale;
        float yCoord = (float) y / size * scale;
        float sample = Mathf.PerlinNoise(xCoord, yCoord);

        noiseGrid[x, y] = sample > threshold;
      }
    }
  }

  for (int x = 0; x < size; x++) {
    for (int y = 0; y < size; y++) {
      if (noiseGrid[x, y]) {
        Vector2Int pointToRemove = new Vector2Int(x, y);
        for (int i = mask.Count - 1; i >= 0; i--) {
          if (mask[i] == pointToRemove) {
            mask.RemoveAt(i);
          }
        }
      }
    }
  }
  for (int x = 0; x < size; x++) {
    for (int y = 0; y < size; y++) {
      if (noiseGrid[x, y]) {
        grid[x, y] = null;
      }
    }
```

```
  }
  yield
  return null;
}
```

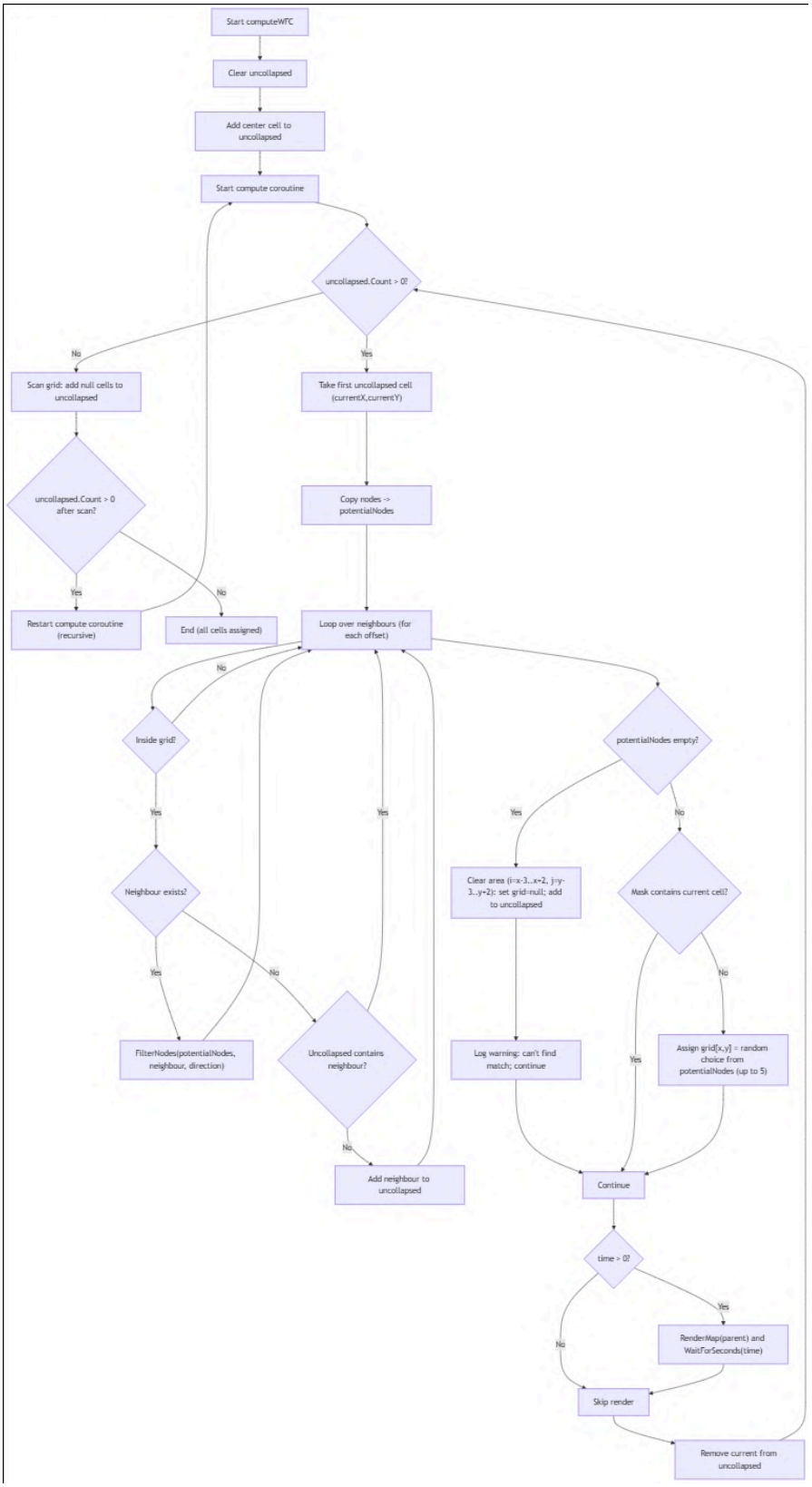
2.1.4.3 Στάδιο 3: Γέμισμα με Wave Function Collapse

Ο αλγόριθμος Wave Function Collapse (WFC) είναι ένας αλγόριθμος παραγωγής περιεχομένου που χρησιμοποιείται συχνά σε περιβάλλοντα προσομοίωσης, καθώς επιτρέπει τη δημιουργία πολύπλοκων σκηνών με βάση απλούς κανόνες τοπικής συμβατότητας. Η βασική ιδέα πίσω από το WFC είναι ότι κάθε κελί ενός πλέγματος μπορεί αρχικά να βρίσκεται σε μια "υπερθέση" πιθανών καταστάσεων (π.χ. διαφορετικοί τύποι blocks), και στη συνέχεια η επιλογή ενός συγκεκριμένου block για ένα κελί περιορίζει δυναμικά τις επιλογές των γειτονικών κελιών, σύμφωνα με προκαθορισμένους κανόνες συμβατότητας. Μέσω αυτής της διαδικασίας «κατάρρευσης», ο αλγόριθμος παράγει ένα περιβάλλον που εμφανίζεται φυσικό, συνεκτικό και με μεγάλη ποικιλία, ιδανικό για την παραγωγή αστικών δομών, δρόμων και κτιρίων στο πλαίσιο ενός προσομοιωτή. Η data-driven φύση του αλγορίθμου τον καθιστά ιδιαίτερα κατάλληλο για χρήση σε συνδυασμό με άλλους αλγορίθμους παραγωγής περιεχομένου, όπως συμβαίνει και στην παρούσα περίπτωση.

Στο παρόν έργο, η συνάρτηση computeWFC υλοποιεί μια παραλλαγή του αλγορίθμου Wave Function Collapse σε ένα δισδιάστατο πλέγμα. Η διαδικασία ξεκινά από το κεντρικό κελί και επαναληπτικά επιλέγει κελιά που δεν έχουν ακόμη καταρρεύσει, φιλτράρει τις πιθανές επιλογές τους με βάση τους γειτονικούς περιορισμούς και αποδίδει έγκυρες τιμές. Σε περιπτώσεις όπου δεν είναι δυνατή καμία επιλογή, εκκαθαρίζεται μια μικρή τοπική περιοχή και η διαδικασία επαναλαμβάνεται. Στο τέλος, εντοπίζονται τυχόν κενά κελιά και η διαδικασία συνεχίζεται έως ότου όλα τα κελιά του πλέγματος λάβουν τιμές.

Κύρια Σημεία του Αλγορίθμου:

- Αρχικοποίηση της λίστας uncollapsed και εισαγωγή του κεντρικού κελιού ως σημείο εκκίνησης.
- Δημιουργία λίστας πιθανών τιμών(potentialNodes) για κάθε μη καταρρεύσαν κελί.
- Έλεγχος των γειτονικών κελιών και φιλτράρισμα των πιθανών επιλογών με βάση τους περιορισμούς προς κάθε κατεύθυνση (πάνω, κάτω, δεξιά, αριστερά). Το φιλτράρισμα γίνεται με τη βοηθητική συνάρτηση FilterNodes, η οποία για να διασφαλίσει ότι οι υποψήφιοι κόμβοι ενός κελιού είναι συμβατοί με τους περιορισμούς των γειτονικών του, συγκρίνει τις συνδέσεις του κάθε πιθανού κόμβου με τις αντίστοιχες συνδέσεις του γειτονικού κόμβου σε μια συγκεκριμένη κατεύθυνση. Οι συνδέσεις κάθε Node αναπαριστώνται με τη μορφή boolean μεταβλητών και πρέπει να ισούνται στις αντίστοιχες πλευρές όπου γίνεται η σύγκριση. Όσοι κόμβοι δεν ταιριάζουν απομακρύνονται από τη λίστα των υποψήφιων, ώστε να παραμείνουν μόνο οι έγκυρες επιλογές.
- Αν δεν υπάρχει καμία έγκυρη επιλογή, εκκαθάριση περιοχής γύρω από το τρέχον κελί και εκ νέου υπολογισμός.
- Αν υπάρχει τουλάχιστον ένας υποψήφιος κόμβος και το κελί δεν ανήκει στη μάσκα (mask), επιλογή ενός τυχαίου κόμβου και αντιστοίχιση στο πλέγμα.
- Αφαίρεση του κελιού από τη λίστα uncollapsed μετά την επεξεργασία.
- Μετά την κύρια επανάληψη, εντοπισμός κελιών που παραμένουν κενά και επανεκκίνηση της διαδικασίας μέχρι τον πλήρη υπολογισμό του πλέγματος.



Σχήμα 2.1.4.1.2: Διάγραμμα συνάρτησης computeWFC

2.1.4.4 Στάδιο 4: Ανίχνευση κενών και τοποθέτηση κτιρίων

Σε αυτό το στάδιο πραγματοποιείται η τοποθέτηση των κτιρίων στο περιβάλλον. Ο αλγόριθμος ξεκινά σκανάροντας για κενές περιοχές στο πλέγμα μέσω της συνάρτησης fillRootNodes, η οποία δημιουργεί τη λίστα buildingRootNodes με τις θέσεις (Vector2Int) όπου μπορεί να τοποθετηθεί ένα κτίριο. Στη συνέχεια, ξεκινά μια coroutine, η generateBuildings, η οποία μεταφράζει τα buildingRootNodes σε θέσεις στον υψηλότερης ανάλυσης πίνακα που χρησιμοποιείται για την τοποθέτηση των building modules. Παράλληλα, αρχικοποιείται το πλέγμα bGridMetadata, το οποίο αποθηκεύει πληροφορίες για κάθε μονάδα κτιρίου, όπως περιστροφή και άλλα σχετικά χαρακτηριστικά.

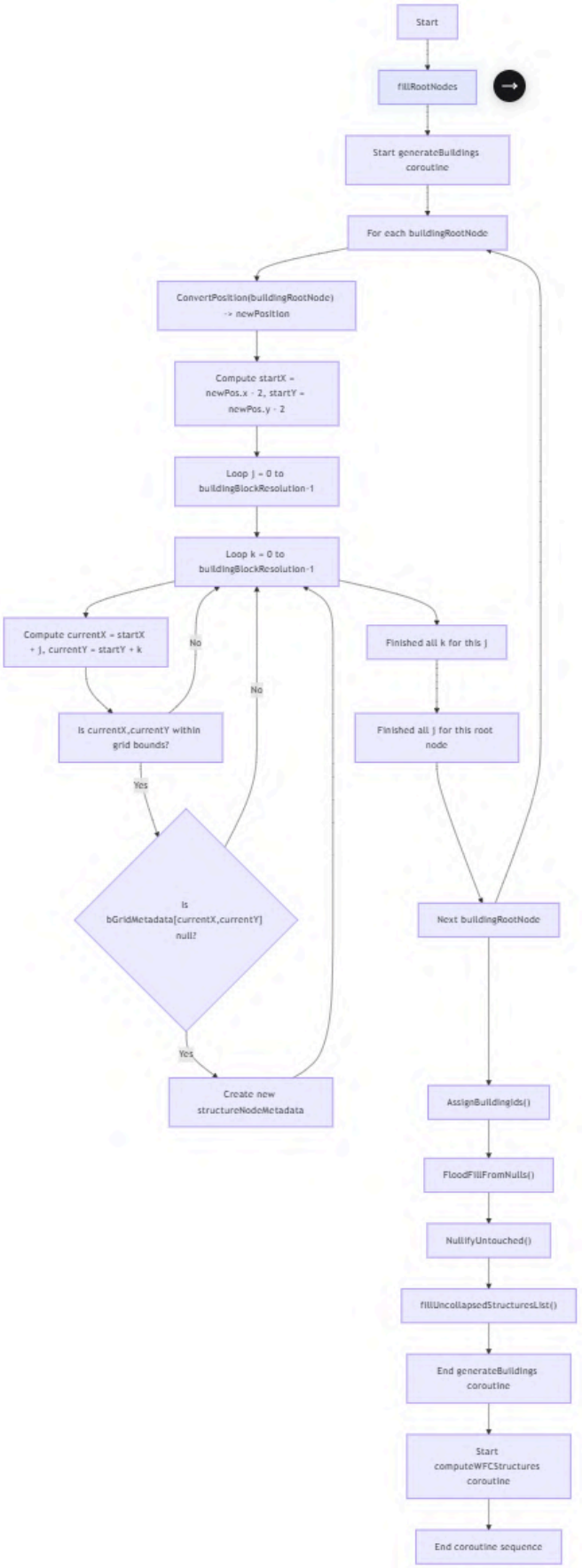
Οι συναρτήσεις AssignBuildingIds και FloodFill συνεργάζονται για τον καθορισμό του σχήματος των κτιρίων στο πλέγμα. Η AssignBuildingIds επιλέγει τον αριθμό ορόφων, αναθέτει ένα μοναδικό ID στο κτίριο και εκκινεί τη διαδικασία flood fill που γεμίζει το bGridMetadata με τα σχετικά δεδομένα. Το FloodFill δημιουργεί ορθογώνια που αναπαριστούν τα κτήρια βάσει ελαχίστου και μέγιστου επιτρεπόμενου εμβαδού. Για λόγους υπολογιστικής αποδοτικότητας, οι μονάδες κτιρίου τοποθετούνται μόνο στις ακμές των σχημάτων, ενώ τα κεντρικά κελιά παραμένουν null. Για να επιτευχθεί αυτό, εκκινείται μια δεύτερη flood fill διαδικασία η οποία ξεκινάει από τα κελιά που παραμένουν κενά και συνεχίζει μέχρι να φτάσει σε αυτά που έχουν κάποια τιμή. Στη συνέχεια η συνάρτηση nullifyUntouched θέτει σε null όσα κελιά δεν συναντήθηκαν ποτέ απο τον flood fill, αφήνοντας έτσι μόνο τις άκρες των κτιρίων, που μας είναι σχετικές.

Τέλος, η συνάρτηση fillUncollapsedStructuresList δημιουργεί τη λίστα των building nodes που θα επεξεργαστεί η επόμενη coroutine, υπολογίζοντας παράλληλα την περιστροφή κάθε κελιού μέσω της συνάρτησης getRotationForCell, αναλόγως με τον προσανατολισμό του. Μετά την ολοκλήρωση αυτής της coroutine, το στάδιο της τοποθέτησης ολοκληρώνεται με τη χρήση του computeWFCStructures, το οποίο εφαρμόζει τον αλγόριθμο Wave Function Collapse για να καθορίσει τα στοιχεία των κτηρίων, λαμβάνοντας υπόψη τους γειτονικούς κόμβους αριστερά και δεξιά και τη λίστα των συμβατών μονάδων για κάθε πλευρά. Η διαδικασία wave function collapse επαναλαμβάνεται για κάθε όροφο ξεχωριστά.

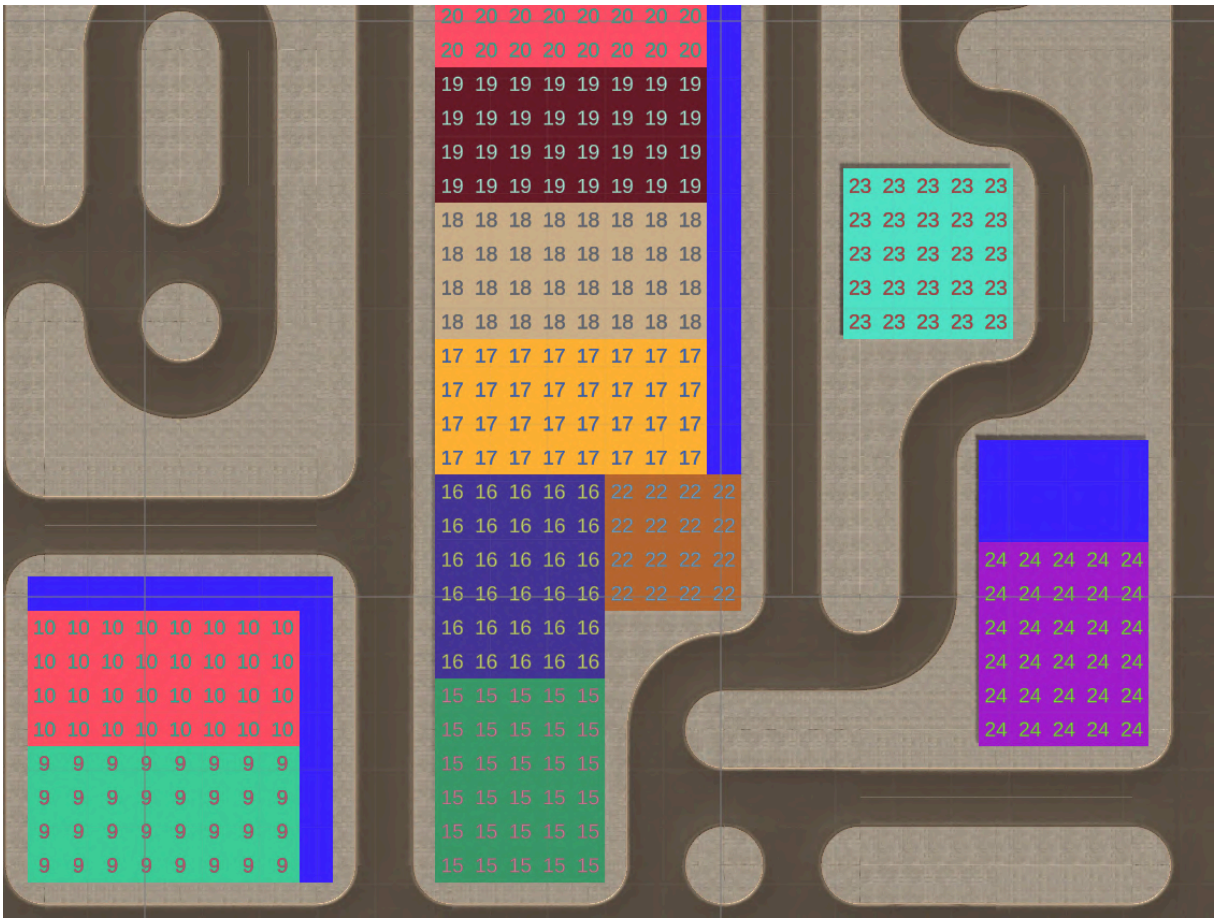
Κύρια τμήματα του κώδικα:

- Αρχικοποίηση Κόμβων: Η συνάρτηση fillRootNodes() καλείται αρχικά για τη δημιουργία των αρχικών κόμβων που θα χρησιμοποιηθούν για την τοποθέτηση των κτιρίων στον πλέγμα.
- Coroutine Δημιουργίας Κτηρίων (generateBuildings)
    - Διατρέχει όλους τους buildingRootNodes
    - Μετατρέπει τη θέση κάθε κόμβου στο πλέγμα των δρόμων σε συντεταγμένες πλέγματος κτηρίων μέσω της συνάρτησης ConvertPosition().
    - Ορίζει μια αρχική περιοχή γύρω από τη μετατρεπόμενη θέση (startX, startY).
    - Εφαρμόζει δύο εμφωλευμένους βρόχους (j και k) για την πλήρωση της περιοχής γύρω από τον κόμβο.
    - Για κάθε κελί στην περιοχή ελέγχει αν το κελί βρίσκεται εντός των ορίων του πλέγματος (bGridSize) Αν το κελί είναι null στο bGridMetadata, δημιουργείται ένα νέο αντικείμενο structureNodeMetadata.
- Ανάθεση id Κτιρίων και καθορισμός σχήματος κτηρίου μέσω floodFill:
    - Χρησιμοποιεί έναν πίνακα visited για να παρακολουθεί τα κελιά που έχουν ήδη επεξεργαστεί.

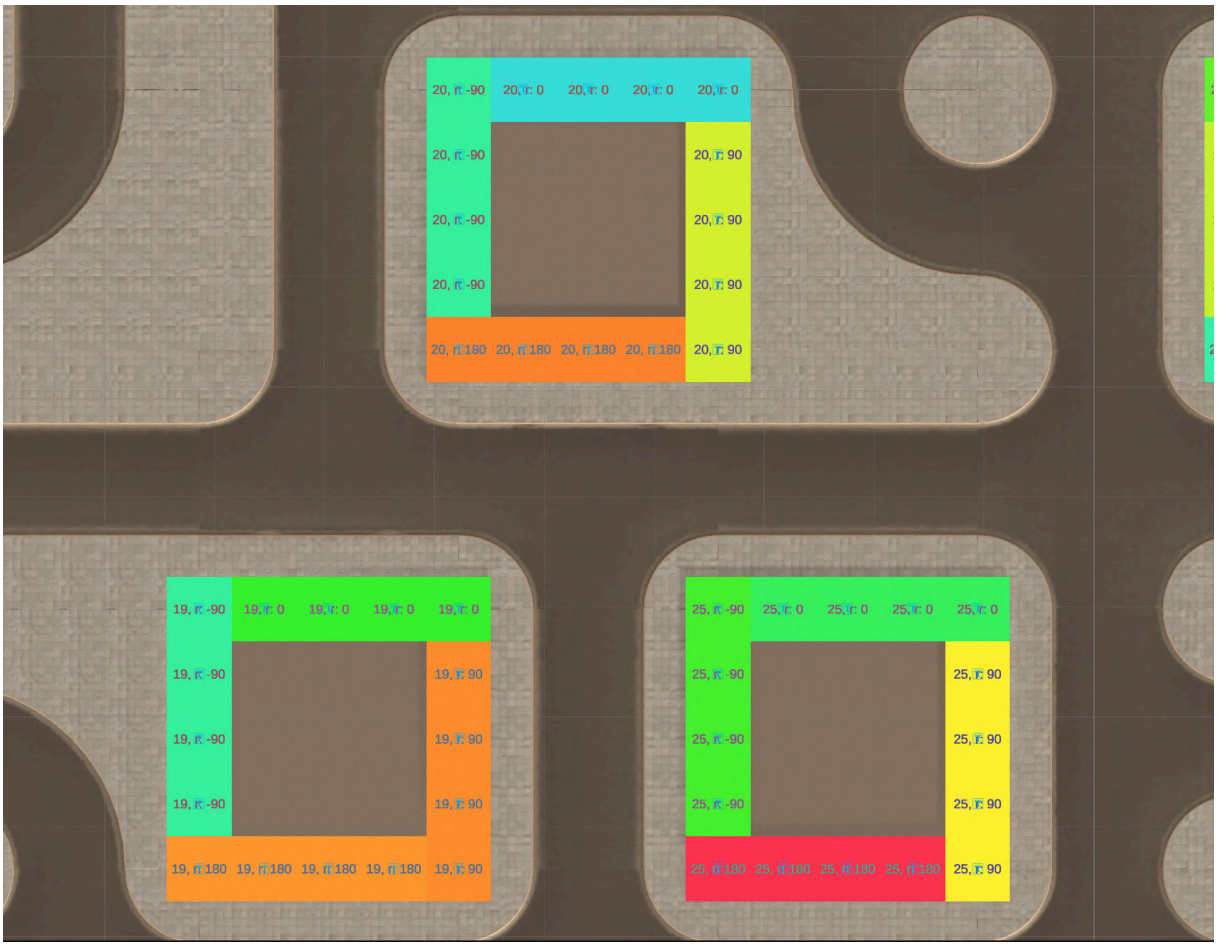
    - Διατρέχει το πλέγμα και εκχωρεί μοναδικό StructureId και αριθμό ορόφων (nFloors) σε συνεχόμενες περιοχές κτιρίων και καλεί τη συνάρτηση FloodFill() για να καθοριστεί το σχήμα.
    - Η FloodFill ελέγχει ορθογώνιες περιοχές βάση ελάχιστου και μέγιστου εμβαδού και σημειώνει όλα τα έγκυρα κελιά ως επισκεφθέντα και αναθέτει τα αναγνωριστικά τους στον πίνακα bGridMetadata
- Αφαίρεση μη απαραίτητων κελιών (FloodFillFromNulls)
    - Χρησιμοποιεί ένα queue για να επεξεργαστεί όλα τα κελιά που είναι null ή έχουν κενό StructureId.
    - Σημειώνει τα προσβάσιμα κελιά ως επισκεφθέντα. Προσβάσιμα θεωρούνται όλα τα κελιά που ειται είναι null είτε εχουν γειτονικά κελιά που είναι null
    - Η συνάρτηση nullifyUntouched διατρέχει το πλέγμα και οποιοδήποτε κελί δεν έχει επισκεφθεί (visited) τίθεται σε null. Έτσι μόνο τα απαραίτητα κελιά που αντιπροσωπεύουν τους τοίχους των κτηρίων μένουν στη μνήμη για περαιτέρω επεξεργασία
- Προετοιμασία κελιών για wave function collapse (fillUncollapsedStructuresList)
    - Διατρέχει όλα τα κελιά.
    - Αν το κελί έχει StructureId, υπολογίζει την περιστροφή του (getRotationForCell) ανάλογα με τον προσανατολισμό του τοίχου. Η μεταβλητή rotation μπορεί να πάρει τις τιμές 0^o, 90^o ,-90^o, 180^o.
    - Προσθέτει τη θέση του στη λίστα uncollapsedStructures για επεξεργασία από τη wave function collapse.
- Ο αλγόριθμος wave function collapse
    - Καθορίζει το κατάλληλο δομικό μπλοκ (bNode) σε κάθε κελί του πλέγματος (bGrid) βάσει γειτονικών περιορισμών.
    - Διατρέχει επαναληπτικά όλα τα μη ολοκληρωμένα κελιά (uncollapsedStructures) μέχρι να επιτευχθεί σταθερή κατάσταση.
    - Για κάθε κελί
        - Εξάγει τις πιθανές επιλογές (potentialNodes) και φιλτράρει τα μπλοκ σύμφωνα με τις γειτονικές συμβατότητες.
        - Αν δεν υπάρχει έγκυρη επιλογή, καθαρίζει γειτονικά κελιά για επανεξέταση και προσθέτει τα κελιά σε λίστα επανυπολογισμού.
        - Αν υπάρχει έγκυρη επιλογή, εκχωρεί τυχαία ένα από τα επιτρεπόμενα μπλοκ και σημειώνει αλλαγή.
        - Η επανάληψη σταματά όταν δεν υπάρχουν πλέον μη ολοκληρωμένα κελιά ή δεν γίνονται αλλαγές σε μία επανάληψη.



Σχήμα 2.1.4.1.3: Διάγραμμα τοποθέτησης κτηρίων



Σχήμα 2.1.4.1.4: Αποτέλεσμα του flood fill σε περιβάλλον debugging



Σχήμα 2.1.4.1.5: αποτέλεσμα της συνάρτησης fillUncollapsedStructuresList.

2.1.4.5 Τοποθέτηση αντικειμένων κατα μήκος των δρόμων

Μετά τη δημιουργία του βασικού περιβάλλοντος, η κλάση variationGenerator αναλαμβάνει τη διαδικασία εμπλουτισμού του οδικού δικτύου με δευτερεύοντα αντικείμενα που αυξάνουν τον ρεαλισμό και τη λειτουργικότητα της προσομοίωσης. Για κάθε οδικό block παράγονται συμβολοσειρές που περιέχουν πληροφορίες για τα πρόσθετα στοιχεία που θα τοποθετηθούν πάνω του, όπως σταθμευμένα οχήματα, κινούμενα οχήματα, πινακίδες STOP, στάσεις λεωφορείου, διαβάσεις πεζών και άλλα σχετικά αντικείμενα. Οι πληροφορίες αυτές αποθηκεύονται στη δομή variationGrid, έναν δισδιάστατο πίνακα συμβολοσειρών, όπου κάθε κελί μπορεί να περιλαμβάνει έναν ή περισσότερους αναγνωριστικούς δείκτες. Για παράδειγμα, η συμβολοσειρά busStop υποδεικνύει την τοποθέτηση στάσης λεωφορείου, ενώ η πιο σύνθετη crosswalkBusStopmovingVehicle υποδεικνύει την ταυτόχρονη παρουσία διάβασης πεζών, στάσης λεωφορείου και κινούμενου οχήματος στο συγκεκριμένο σημείο. Ο αλγόριθμος λαμβάνει υπόψη τόσο τον τύπο του block όσο και χωρικούς περιορισμούς, για παράδειγμα, δύο στάσεις λεωφορείου δεν μπορούν να τοποθετηθούν ακριβώς δίπλα η μία στην άλλη, καθώς κάτι τέτοιο θα υποβάθμιζε τον ρεαλισμό του περιβάλλοντος.

Η κλάση διατηρεί διάφορες λίστες τύπου Vector2Int, συγκεκριμένα τις intersections, busStops και crosswalks, για την αποθήκευση των συντεταγμένων των ειδικών χαρακτηριστικών του οδικού δικτύου. Επιπλέον, ορίζεται ένα event (onDoneGeneratingVariations) ώστε να ειδοποιούνται εξωτερικά συστήματα όταν ολοκληρωθεί η διαδικασία παραγωγής παραλλαγών.

Η βασική μέθοδος calculateVariations δέχεται ως είσοδο έναν δισδιάστατο πίνακα Node (που αναπαριστά το οδικό πλέγμα) και μια λίστα με διαθέσιμα αντικείμενα παραλλαγών. Η λειτουργία της περιλαμβάνει τα εξής στάδια:

- Αρχικοποίηση: Εκκαθαρίζεται η προηγούμενη κατάσταση με τη μέθοδο clear() και δημιουργείται εκ νέου ο πίνακας variationGrid
- Σάρωση Πλέγματος: Κάθε κελί του πλέγματος αναλύεται με βάση τον τύπο του:
    - Διασταυρώσεις ανιχνεύονται και αποθηκεύονται στη λίστα intersections
    - Ευθείες οδοί (RoadStraightHorizontal, RoadStraightVertical) αξιολογούνται για την τοποθέτηση στάσεων λεωφορείου, διαβάσεων και άλλων αντικειμένων.
- Τοποθέτηση αντικειμένων: Προστίθενται βάση τον τύπου του κελιού και βάση άλλων λογικών περιορισμών, για παράδειγμα οι στάσεις λεωφορείων προστίθενται μόνο εφόσον το συγκεκριμένο κελί βρίσκεται εκτός μιας ελάχιστης απόστασης από υπάρχουσες στάσεις, κάτι που ελέγχεται από τη μέθοδο isOutsideOfRange. Με πιθανότητα 90% εισάγεται στάση στο variationGrid και αποθηκεύεται η θέση της.
- Τοποθετούνται κινούμενα οχήματα πάνω στα ευθεία τμήματα δρόμων βάση μεταβλητής που ελέγχει την επιθυμητή ποσότητα της κίνησης
- Τοποθετείται σήμανση: Η σήμανση τοποθετείται βάση του τύπου του τμήματος δρόμου που εξετάζουμε (ορισμένα σήματα απαιτούν σταυροδρόμι) και λογικά κρητήρια, για παράδειγμα δεν είναι ρεαλιστικό να έχουμε σήμανση “stop” σε όλες τις κατευθύνσεις κυκλοφορίας
- Μετά την ολοκλήρωση της διαδικασίας, καλείται το γεγονός onDoneGeneratingVariations για να ενημερωθούν τα υπόλοιπα υποσυστήματα ότι το πλέγμα παραλλαγών είναι έτοιμο.

Σχήμα 2.1.4.5.1: Αποτέλεσμα 1

Σχήμα 2.1.4.5.2: Αποτέλεσμα 2

2.1.4.6 Αυτόνομοι οδηγοί

Τα οχήματα τεχνητής νοημοσύνης που περιλαμβάνονται στην προσομοίωση χρησιμοποιούν καμπύλες για να καθορίσουν μια ρεαλιστική και ομαλή πορεία μέσα στο περιβάλλον. Οι καμπύλες αυτές ορίζονται σε στάδιο ανάπτυξης και ενσωματώνονται σε κάθε road module που αποτελεί το οδικό δίκτυο. Η κλάση vehicle_agent αναλαμβάνει τη συμπεριφορά του οχήματος, συμπεριλαμβανομένης της παρακολούθησης της πορείας, της αντίληψης του περιβάλλοντος και της αλληλεπίδρασης με άλλα οχήματα. Όταν ένα όχημα εισέρχεται στην είσοδο ενός συγκεκριμένου road module, η κλάση entry_manager αναθέτει μία από τις διαθέσιμες καμπύλες στο αντίστοιχο instance της κλάσης vehicle_agent. Η vehicle_agent μετατρέπει την καμπύλη σε μια πιο αποδοτική μορφή για υπολογισμούς και την αποθηκεύει προσωρινά (cache), ώστε άλλα instances της ίδιας κλάσης να μην μπαίνουν σε διαδικασία επανυπολογισμού. Αφού ολοκληρωθεί αυτή η μετατροπή, η κλάση αναθέτει τιμές στα πεδία motorTorque και steerAngle των σχετικών wheel colliders προκειμένου να πραγματοποιηθεί η κίνηση του οχήματος κατά μήκος της καμπύλης. Για την αντίληψη του περιβάλλοντος και των άλλων οχημάτων, η vehicle_agent χρησιμοποιεί raycasts, δηλαδή ακτίνες που εκτοξεύονται και επιστρέφουν πληροφορίες για τυχόν coliders που διασταυρώνονται με την ακτίνα. Αυτή η μέθοδος επιτρέπει την αποφυγή συγκρούσεων και την προσαρμογή της ταχύτητας ή της πορείας του οχήματος σε πραγματικό χρόνο.

Για λόγους απόδοσης, η προσομοίωση της τεχνητής νοημοσύνης ενεργοποιείται μόνο όταν ένα όχημα βρίσκεται εντός συγκεκριμένης απόστασης από το όχημα του οδηγού, αποφεύγοντας την επεξεργασία άχρηστων υπολογισμών για οχήματα που δεν είναι ορατά. Ακολουθεί μια σύντομη εξήγηση των κύριων χαρακτηριστικών και λειτουργιών της κλάσης vehicle_agent:

- Ακολουθεί καμπύλες:

  Η κίνηση του οχήματος βασίζεται σε καμπύλες (SplineContainer) που ορίζονται σε κάθε road module. Η κλάση χρησιμοποιεί μια cache (CachedSpline) για να αποθηκεύσει τις θέσεις και τις κατευθύνσεις της καμπύλης, μειώνοντας τους υπολογισμούς σε κάθε frame.

- steerAngle και motorTorque:
    - Η συνάρτηση ComputeSteerAndTorque υπολογίζει τις γωνίες στρέψης του τιμονιού και της ροπής που εφαρμόζεται στους τροχούς. Ο υπολογισμός της γωνίας στρέψης των μπροστινών τροχών γίνεται μέσω ενός σημείου κάποιας απόστασης (lookAheadDistance) μπροστά από το όχημα. Η συνάρτηση παίρνει το διάνυσμα τοποθεσίας και χρησιμοποιεί τη μέθοδο transform.InverseTransformPoint για να βρεί που βρίσκεται το σημείο αυτό σε μορφή σχετική με την κατεύθυνση του οχήματος, δηλαδη όπου θετικό x βρίσκεται στα δεξιά και όπου αρνητικό στα αριστερά. Επειτα χρησιμοποιείται η μέθοδος Mathf.Atan2(localTarget.x, localTarget.z) για να υπολογιστεί η γωνία μεταξύ της κατεύθυνσης του οχήματος και του στόχου σε Rad, και μετατρέπεται έπειτα σε μοίρες. Τέλος η γωνία σε μορφή μοιρών κανονικοποιείται μεταξύ -maxSteerAngle και +maxSteerAngle μέσω της μεθόδου Mathf.Clamp
    - Η ισχύς στους κινητήριους τροχούς υπολογίζεται με έναν απλό P-controller που συγκρίνει την επιθυμητή ταχύτητα με την τρέχουσα RPM των τροχών.
- Οι υπολογισμοί για γωνία στρέψης και torque γίνονται περιοδικά (updateInterval) και αποθηκεύονται προσωρινά (cachedSteerAngle, cachedTorque) για αποδοτικότητα.
- Ενεργοποίηση βάσει απόστασης από τον παίκτη: Το όχημα υπολογίζει την απόστασή του από τον παίκτη και απενεργοποιείται αν βρίσκεται εκτός συγκεκριμένου εύρους, εξοικονομώντας πόρους.
- Spline Caching:
    - Η κλάση CachedSpline μετατρέπει την καμπύλη σε ενα πιο διαχειρίσιμο σύνολο από σημεία και αποθηκεύει τις θέσεις των σημείων, τις αθροιστικές αποστάσεις ανά σημείο, και το μήκος της καμπύλης.
    - Παρέχει συναρτήσεις για να υπολογίζεται η θέση σε συγκεκριμένη απόσταση, την κατεύθυνση ενός segment, την πλησιέστερη απόσταση σε μια δεδομένη θέση στον κόσμο.
    - Η κλάση αποθηκεύεται σε ένα στατικό Dictionary όπου συσχετίζει τις αρχικές καμπύλες με τις απλοποιημένες μορφές της, ώστε να είναι προσβάσιμα από όλα τα instances της κλάσης vehicleAgent για να αποφευχθούν οι επαναϋπολογισμοί

```
 private static Dictionary<SplineContainer, CachedSpline>
splineCache
```

- Χρησιμοποιεί το εργαλείο της Unity Physics.Raycast για να τραβήξει νοητές ακτίνες απο τη θέση του προς διάφορες κατευθύνσεις για να λάβει πληροφορίες για άλλα οχήματα που βρίσκονται γύρω από όχημα, ώστε να μπορεί να ρυθμίσει τη συμπεριφορά του αναλόγως.
- Διαθέτει public συναρτήσεις οι οποίες καλούνται από εξωτερικές κλάσεις για να ρυθμιστεί η κυκλοφορία του οχήματος
    - stop(): Εκτελείται σε περίπτωση που ανιχνευθεί εμπόδιο εμπρός από το όχημα.
    - stopDueToPriority(): Εκτελείται από εξωτερική κλάση σε περίπτωση που το όχημα δεν έχει προτεραιότητα σε κάποιον κόμβο.
    - resetPriority(): Εκτελείται για να αποδοθεί η προτεραιότητα του οχήματος σε κόμβο.
    - followCurve(): Εκκινεί τη διαδικασία ακολούθησης της καμπύλης

Η ρύθμιση της κυκλοφορίας στους κόμβους γίνεται με τη βοήθεια τριών κλάσεων:

- entriesController: κάθε κόμβος συσχετίζεται με ένα instance του entriesController. Κάθε instance διατηρεί μια λίστα από vehicleAgents τα οποία έχουν εισέλθει στον κόμβο και βάση κανόνων προτεραιότητας καλεί συναρτήσεις για να σταματήσει, ή να εκκινήσει το όχημα.
- vehicleAgent: Διαθέτει συναρτήσεις οι οποίες επιτρέπουν στην κλάση entriesController να χειριστεί το όχημα, και αποθηκεύει ένα αντικείμενο lastEntriesController που δηλώνει από ποιον κόμβο ελέγχεται το όχημα.
- entryManager: Όταν ένα όχημα εισέλθει σε άλλον κόμβο, αφαιρεί το instance του vehicleAgent από τη λίστα του προηγούμενο entriesController. Θέτει το αντικείμενο lastEntriesController σε null, και έπειτα θέτει το ανανεωμένο instance του entriesController στο αντικείμενο lastEntriesController, εφόσον αυτό είναι υπαρκτό.

Σχήμα 2.1.4.6: Το γαλάζιο όχημα που φαίνεται στα δεξιά παραχωρεί προτεραιότητα βάση STOP στα οχήματα που κινούνται κάθετα στην διασταύρωση

2.2 : Σχεδίαση και Υλοποίηση Ηλεκτρονικών τμημάτων

Στο πλαίσιο της ανάπτυξης ενός λειτουργικού προσομοιωτή οχήματος, η ακριβής και αξιόπιστη καταγραφή των εισόδων του χρήστη αποτελεί θεμελιώδη παράμετρο για την επίτευξη ποιοτικής αλληλεπίδρασης. Κεντρικό ρόλο σε αυτή τη διαδικασία κατέχει ο μικροελεγκτής, ο οποίος αποτελεί τον κύριο επεξεργαστή των αναλογικών και ψηφιακών σημάτων που προέρχονται από τα διάφορα χειριστήρια του συστήματος, όπως το τιμόνι, τα πεντάλ (γκάζι, φρένο, συμπλέκτης) και ο μοχλός ταχυτήτων.

Η πλακέτα που χρησιμοποιήθηκε στο παρόν έργο ονομάζεται Arduino Micro, βασισμένη στον μικροελεγκτή ATmega32U4. Η επιλογή αυτής της πλακέτας έγινε με γνώμονα την ενσωματωμένη δυνατότητα άμεσης επικοινωνίας μέσω USB με τον υπολογιστή, χωρίς την ανάγκη εξωτερικού μετατροπέα UART-USB, καθώς και από την υψηλή ακρίβεια και σταθερότητα των μετατροπέων σήματος ADC που διαθέτει .

2.2.1 Ο ρόλος του μικροελεγκτή στην προσομοίωση οχημάτων

Τα χειριστήρια του προσομοιωτή είναι συνδεδεμένα μέσω αναλογικών ποτενσιόμετρων και ψηφιακών διακοπτών τα οποία μετατρέπουν τις μηχανικές κινήσεις των χρηστών σε μεταβαλλόμενα ηλεκτρικά σήματα DC τάσης. Ο μικροελεγκτής διαβάζει συνεχώς αυτές τις τιμές μέσω των ADC καναλιών και ψηφιακών θυρών του, πραγματοποιώντας ψηφιοποίηση των σημάτων σε τιμές, κατάλληλες για περαιτέρω επεξεργασία απο το λογισμικό . Μετά τη μετατροπή των αναλογικών σημάτων σε ψηφιακά δεδομένα και την μορφοποίηση των ψηφιακών τιμών, ο μικροελεγκτής προχωρά στην οργάνωση των πληροφοριών αυτών σε ένα κατάλληλο πρωτόκολλο επικοινωνίας και τις αποστέλλει μέσω USB στον υπολογιστή που εκτελεί το λογισμικό προσομοίωσης. Η διαδικασία αυτή εξασφαλίζει χαμηλή καθυστέρηση και συνεχή ροή δεδομένων, χαρακτηριστικά απαραίτητα για την επίτευξη ρεαλιστικής εμπειρίας χρήστη, όπου η αλληλεπίδραση με το σύστημα είναι άμεση και ακριβής.

2.2.2 Τεχνική ανάλυση του μικροελεγκτή Arduino Micro

Ο Arduino Micro, βασισμένος στον επεξεργαστή ATmega32U4 της οικογένειας AVR, γνωστός για την ενσωματωμένη δυνατότητα USB επικοινωνίας, η οποία εξαλείφει την ανάγκη δευτερεύοντος επεξεργαστή[12]. Αυτή η δυνατότητα επιτρέπει στην πλακέτα να εμφανίζεται σε υπολογιστικό σύστημα ως εικονικό πληκτρολόγιο, ποντίκι ή εικονική θύρα σειριακής επικοινωνίας, καθιστώντας τον ιδιαίτερα κατάλληλο για εφαρμογές που απαιτούν άμεση και ακριβή αλληλεπίδραση με τον χρήστη, όπως προσομοιωτές οδήγησης και άλλα διαδραστικά περιβάλλοντα [13].

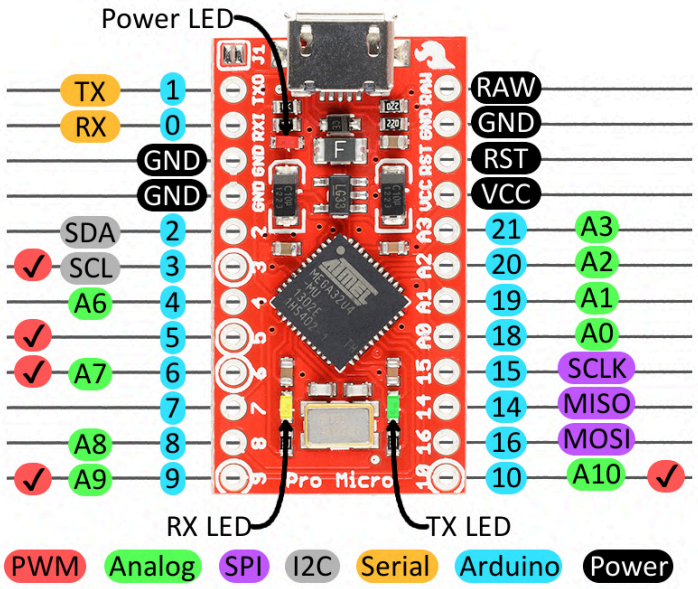
Η πλακέτα διαθέτει 20 ψηφιακές εισόδους/εξόδους, από τις οποίες 7 μπορούν να χρησιμοποιηθούν ως PWM έξοδοι, προσφέροντας δυνατότητα παραγωγής αναλογικού σήματος μέσω τεχνικής διαμόρφωσης πλάτους παλμού. Επιπλέον, παρέχει 12 αναλογικές εισόδους οι οποίες συνδέονται με τον 10-bit ADC του ATmega32U4, επιτρέποντας δειγματοληψία τάσεων με ανάλυση 1024 διακριτών επιπέδων. Ο ενσωματωμένος κρύσταλλος 16 MHz διασφαλίζει σταθερό χρονισμό για την εκτέλεση εντολών και την επεξεργασία δεδομένων σε πραγματικό χρόνο, ενώ η 32 KB μνήμη παρέχει επαρκείς πόρους για την υλοποίηση εφαρμογών μεσαίας πολυπλοκότητας[12].

Η πλακέτα περιλαμβάνει επίσης ICSP header για προγραμματισμό χαμηλού επιπέδου και θύρα micro USB, διευκολύνοντας την εύκολη ενσωμάτωση σε άλλες συσκευές κυκλώματα. Η ενσωματωμένη δυνατότητα USB επιτρέπει χαμηλή καθυστέρηση επικοινωνίας (low latency) με το συνδεδεμένο

υπολογιστικό σύστημα, εξασφαλίζοντας ότι οι αναλογικές μετρήσεις από χειριστήρια, όπως ποτενσιόμετρα τιμονιού και πεντάλ, μετατρέπονται σε ψηφιακά δεδομένα και αποστέλλονται σε πραγματικό χρόνο στο λογισμικό προσομοίωσης [13], [14].

Τεχνικές Προδιαγραφές (Specifications):

- Επεξεργαστής: ATmega32U4, 8-bit AVR, 16 MHz
- Μνήμη: 32 KB Flash, 2.5 KB SRAM, 1 KB EEPROM
- Ψηφιακές Είσοδοι/Εξοδοι: 20 (7 PWM)
- Αναλογικές Είσοδοι: 12, 10-bit ADC (0–5 V)
- USB: Micro USB, CDC virtual serial port, HID (πληκτρολόγιο/ποντίκι)
- Χρονισμός: Κρύσταλλος 16 MHz
- Πρόσθετα: ICSP header, reset button, breadboard compatible layout

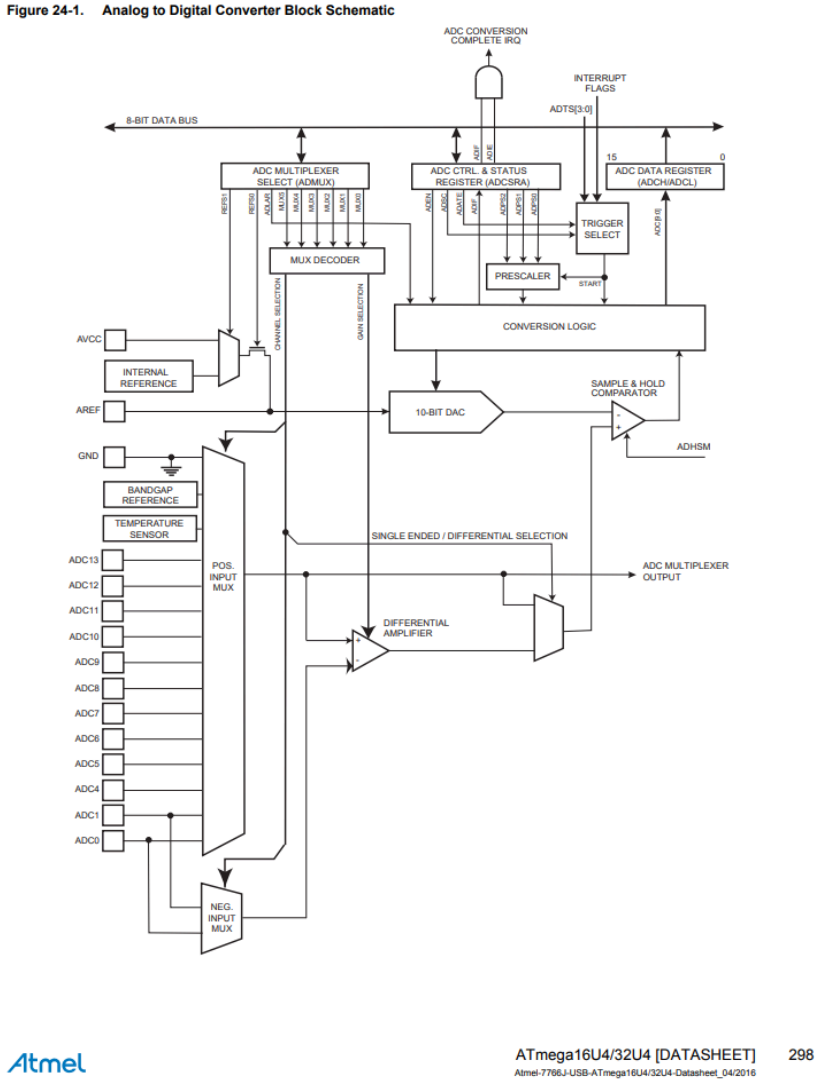


Σχήμα 2.2.2.1: Αναπαράσταση πλακέτας Arduino Pro Micro [13]

2.2.2.1 Ανάλυση της θύρας F και της λειτουργίας του ADC του ATmega32U4

Στον μικροελεγκτή ATmega32U4, το Port F αποτελεί κύριο στοιχείο για τη δειγματοληψία αναλογικών σημάτων μέσω του ADC. Οι ακίδες PF7 έως PF4, PF1 και PF0 αντιστοιχούν στα αναλογικά κανάλια εισόδου του ADC, επιτρέποντας την ακριβή μέτρηση συνεχών τάσεων που ποικίλλουν συνήθως στο εύρος 0–5 V. Κάθε κανάλι του Port F παρέχει 10-bit ανάλυση σήματος, επιτρέποντας τη μετατροπή της αναλογικής τάσης σε 1024 διακριτά ψηφιακά επίπεδα, γεγονός που προσφέρει υψηλή ακρίβεια στην αναπαράσταση των πραγματικών τιμών των αισθητήρων ή χειριστηρίων. Οι ακίδες του Port F υποστηρίζουν επίσης εσωτερικές pull-up αντιστάσεις, οι οποίες μπορούν να ενεργοποιηθούν κατά βούληση για τη διατήρηση σταθερής λογικής στάθμης όταν η είσοδος δεν είναι ενεργά συνδεδεμένη. Επιπλέον, οι ακίδες βρίσκονται σε tri-state κατάσταση κατά

την ενεργοποίηση reset, διασφαλίζοντας ότι η λειτουργία του ADC και η ακεραιότητα των μετρήσεων δεν επηρεάζονται από ασταθείς καταστάσεις εξόδου[12].



Σχήμα 3.2: Διάγραμμα ADC του Atmeta32U4 [12]

Ο μετατροπέας αναλογικών σε ψηφιακών (ADC) του μικροελεγκτή ATmega32U4 αποτελεί κρίσιμο υποσύστημα για την απόκτηση και επεξεργασία αναλογικών σημάτων. Πρόκειται για μετατροπέα τύπου Successive Approximation Register (SAR), με 10-bit ανάλυση, ικανό να αποδώσει έως 1024 διακριτά επίπεδα κβαντοποίησης. Ο ADC υποστηρίζει έως 12 αναλογικά κανάλια, τα οποία αντιστοιχούν σε συγκεκριμένες ακίδες (κυρίως του Port F, PF7..PF0, καθώς και επιλεγμένα pins άλλων θυρών). Η επιλογή καναλιού πραγματοποιείται μέσω ενός αναλογικού πολυπλέκτη (ADC multiplexer), ο οποίος δρομολογεί το επιλεγμένο σήμα στην είσοδο του μετατροπέα. Αξίζει να σημειωθεί ότι, παρά την ύπαρξη πολλών καναλιών, ο ADC μπορεί να εκτελέσει μετατροπή μόνο σε ένα σήμα ανά δεδομένη χρονική στιγμή, γεγονός που καθιστά απαραίτητη τη σειριακή δειγματοληψία σε εφαρμογές πολλαπλών αισθητήρων[12].

Η επιλογή του καναλιού πραγματοποιείται μέσω των καταχωρητών ADMUX (ADC Multiplexer Selection Register) και ADCSRB, όπου οι αντίστοιχοι συνδυασμοί bit καθορίζουν ποια ακίδα (π.χ.

PF0–PF7) συνδέεται στο κύκλωμα. Ο πολυπλέκτης υποστηρίζει τόσο single-ended εισόδους (αναλογικό σήμα έναντι γείωσης), όσο και differential εισόδους, όπου δύο κανάλια συγκρίνονται μεταξύ τους. Στη διαφορική λειτουργία είναι δυνατή η ενεργοποίηση του Programmable Gain Amplifier (PGA), που προσφέρει ενίσχυση 10×, 200× κ.λπ., διευρύνοντας το εύρος εφαρμογών του ADC σε μετρήσεις χαμηλού πλάτους[12].

Αμέσως μετά τον πολυπλέκτη, το σήμα εισέρχεται στο sample-and-hold κύκλωμα. Το κύκλωμα αυτό περιλαμβάνει έναν πυκνωτή αποθήκευσης και έναν ενισχυτή απομόνωσης, οι οποίοι «παγιδεύουν» την τάση εισόδου για μικρό χρονικό διάστημα, διασφαλίζοντας ότι η επεξεργασία που ακολουθεί εκτελείται σε μια σταθερή αναλογική τιμή. Το βήμα αυτό είναι απαραίτητο, διότι το αναλογικό σήμα μπορεί να μεταβάλλεται συνεχώς, ενώ η διαδικασία της ψηφιοποίησης απαιτεί σταθερή είσοδο[12].

Στη συνέχεια, το σήμα οδηγείται στο επόμενο στάδιο στο οποίο το σταθερό σήμα από το S/H συγκρίνεται με την τάση που παράγεται από τον DAC μέσω του εσωτερικού συγκριτή. Η σύγκριση πραγματοποιείται μέσω ενός ενσωματωμένου τελεστικού ενισχυτή που λειτουργεί σε mode CMOS differential comparator. Ο συγκριτής είναι εσωτερικός, low-power CMOS, και αποτελεί μέρος του ADC sub-system και έχει σχεδιαστεί για χαμηλό offset και γρήγορη απόκριση, ώστε η διαδικασία successive approximation να ολοκληρώνεται σταθερά σε 10 βήματα για 10-bit ανάλυση. Ο Successive Approximation Register του ATmega32U4 υλοποιεί τη διαδικασία μετατροπής μέσω μιας διαδοχικής διχοτόμησης του εύρους της αναφοράς. Η λειτουργία του βασίζεται στη συνεχή σύγκριση της αναλογικής εισόδου με μια τάση που παράγεται από τον εσωτερικό ψηφιακό προς αναλογικό μετατροπέα (DAC)[12]. Τα παραπάνω βήματα είναι τα εξής:

- Αρχικοποίηση (Initialization):

  Ο SAR θέτει αρχικά το πιο σημαντικό bit (MSB) της 10-bit εξόδου στο λογικό “1” και όλα τα υπόλοιπα bits στο “0”. Έτσι, ο DAC παράγει μια αναφορά ίση με το ήμισυ της μέγιστης τάσης αναφοράς (Vref/2).

- Σύγκριση με το σήμα εισόδου:

  Ο συγκριτής ελέγχει αν η είσοδος είναι μεγαλύτερη ή μικρότερη από την παραγόμενη τιμή του DAC.

  Αν η είσοδος είναι μεγαλύτερη, τότε το MSB παραμένει “1”.

  Αν η είσοδος είναι μικρότερη, το MSB μηδενίζεται (“0”).

- Επόμενο bit (Bit trial):

  Ο SAR προχωρά στο αμέσως επόμενο bit (το δεύτερο πιο σημαντικό). Αυτό τίθεται σε “1”, ενώ τα λιγότερο σημαντικά παραμένουν “0”. Ο DAC ενημερώνει τη νέα τάση αναφοράς, η οποία τώρα ισούται με ¾ Vref ή ¼ Vref, ανάλογα με την τιμή του MSB.

- Διαδοχικές επαναλήψεις:

  Η ίδια διαδικασία επαναλαμβάνεται για κάθε bit, από το MSB (bit 9) έως το LSB (bit 0). Σε κάθε βήμα ο συγκριτής καθορίζει αν το υπό δοκιμή bit θα παραμείνει “1” ή θα μηδενιστεί, ανάλογα με το αν το αναλογικό σήμα υπερβαίνει ή όχι την τιμή εξόδου του DAC.

- Ολοκλήρωση (Final approximation):

  Μετά από 10 διαδοχικές συγκρίσεις, το περιεχόμενο του SAR αντιπροσωπεύει τη βέλτιστη ψηφιακή εκτίμηση της αναλογικής εισόδου. Αυτή η λέξη μεταφέρεται στους καταχωρητές ADCL και ADCH και μπορεί να αναγνωστεί από το λογισμικό.

Η συγκεκριμένη μεθοδολογία προσφέρει υψηλή ακρίβεια και ταχύτητα, καθώς κάθε μετατροπή ολοκληρώνεται σε 10 διαδοχικά βήματα. Ο συνολικός χρόνος μετατροπής εξαρτάται από τη συχνότητα του ρολογιού του ADC (π.χ. f_ADC ≈ 50–200 kHz, για βέλτιστη ακρίβεια). Αυτός ο χρονισμός καθορίζεται μέσω του prescaler στον καταχωρητή ADCSRA, επιτρέποντας την προσαρμογή του ADC στη συχνότητα λειτουργίας του μικροελεγκτή (16 MHz)[12].

Η διαδικασία αυτή ουσιαστικά «σκανάρει» το σήμα εισόδου από το MSB προς το LSB, κάθε φορά μειώνοντας στο μισό το εύρος αβεβαιότητας. Χάρη σε αυτήν τη μεθοδολογία, η τελική ψηφιοποίηση ολοκληρώνεται σε 10 βήματα ανεξάρτητα από την τιμή της εισόδου, προσφέροντας σταθερή και προβλέψιμη καθυστέρηση μετατροπής.

2.2.2.2 Ανάλυση των ψηφιακών θυρών του ATmega32U4

Εκτός από τις αναλογικές εισόδους του Port F, στον ATmega32U4 χρησιμοποιούνται και ψηφιακές ακίδες για τη σύνδεση διακοπτών, κουμπιών ή άλλων ψηφιακών χειριστηρίων του οχήματος, όπως, αλλαγές ταχυτήτων ή κουμπιά ειδικών λειτουργιών. Οι ψηφιακές είσοδοι αντιστοιχούν κυρίως στις Ports B, C, D, και E, ανάλογα με τη διαμόρφωση του κυκλώματος. Για παράδειγμα, πολλές εφαρμογές χρησιμοποιούν τις ακίδες του Port B (PB0–PB7) ή του Port D (PD0–PD7) για ψηφιακούς διακόπτες, καθώς προσφέρουν δυνατότητα προσδιορισμού υψηλής/χαμηλής λογικής στάθμης (HIGH/LOW) και υποστηρίζουν εσωτερικές pull-up αντιστάσεις. Εάν η ακίδα είναι συνδεδεμένη σε διακόπτη που κλείνει προς γείωση, μπορεί να ενεργοποιηθεί η εσωτερική pull-up αντίσταση μέσω του αντίστοιχου bit στον PORTx register, διασφαλίζοντας ότι η ακίδα είναι HIGH όταν ο διακόπτης είναι ανοιχτός και LOW όταν κλείνει. Η τρέχουσα κατάσταση της ακίδας μπορεί να διαβαστεί απευθείας από τον PINx register, επιστρέφοντας 1 ή 0 ανάλογα με την τάση της εισόδου[12].

2.2.2.3 Ανάλυση του USB module του ATmega32U4

Το Arduino Micro, διαθέτει εγγενείς δυνατότητες USB 2.0 Full-Speed επικοινωνίας (12 Mbps), οι οποίες το διαφοροποιούν από άλλες πλατφόρμες Arduino που απαιτούν εξωτερικό ολοκληρωμένο για την υλοποίηση του USB interface. Η παρουσία του ενσωματωμένου USB controller επιτρέπει στο Arduino Micro να συνδέεται απευθείας σε έναν προσωπικό υπολογιστή μέσω της θύρας micro-USB, χωρίς την ανάγκη επιπρόσθετου hardware, όπως FTDI chip ή άλλο bridge[13], [14].

Η USB λειτουργικότητα του Arduino Micro βασίζεται στο πρωτόκολλο USB HID (Human Interface Device) και στο CDC (Communication Device Class), επιτρέποντάς του να προσομοιώνει πολλαπλές κατηγορίες συσκευών. Συγκεκριμένα, ο ATmega32U4 μπορεί να παρουσιαστεί στον υπολογιστή είτε ως σειριακή θύρα, μέσω του CDC class, είτε ως συσκευή εισόδου, όπως πληκτρολόγιο, ποντίκι ή joystick, μέσω του HID class [12], [15], [16].

Σε επίπεδο υλοποίησης, η USB διεπαφή του Arduino Micro υποστηρίζεται από ένα ολοκληρωμένο USB transceiver που συνεργάζεται με το firmware της πλατφόρμας Arduino. Αυτό καθιστά δυνατή την υλοποίηση διπλής επικοινωνίας, αφενός για σκοπούς προγραμματισμού και μεταφόρτωσης κώδικα στον μικροελεγκτή, και αφετέρου για τη συνεχή μεταφορά δεδομένων κατά τη λειτουργία της εφαρμογής. Η δυνατότητα αυτή επιτρέπει την ταυτόχρονη ύπαρξη προγραμματιστικής διεπαφής και λειτουργικής διεπαφής με τον τελικό χρήστη, χωρίς να απαιτείται αναπροσαρμογή του συστήματος επικοινωνίας[12], [14].

Η επιλογή του Arduino Micro σε εφαρμογές προσομοίωσης, όπως στο παρόν έργο, δικαιολογείται σε μεγάλο βαθμό από αυτές τις USB δυνατότητες. Χάρη στην εγγενή υποστήριξη HID, ο μικροελεγκτής μπορεί να αναγνωρίζεται αυτόματα από τον υπολογιστή ως τυπική συσκευή εισόδου, μειώνοντας την

ανάγκη για πρόσθετους οδηγούς (drivers) και εξασφαλίζοντας συμβατότητα με ποικίλο λογισμικό προσομοίωσης. Έτσι, επιτυγχάνεται μια απρόσκοπτη διεπαφή μεταξύ των ηλεκτρονικών του προσομοιωτή και του υπολογιστικού περιβάλλοντος.

2.2.3 Ανάλυση των ηλεκτρονικών μερών του συστήματος και του μικροελεγκτή με σκοπό την προσομοίωση οχήματος

Το ηλεκτρονικό υποσύστημα του προσομοιωτή οχήματος αποτελεί τον ενδιάμεσο κρίκο μεταξύ των φυσικών χειριστηρίων του οδηγού και του λογισμικού προσομοίωσης. Πάνω στις μηχανικές μονάδες, βρίσκονται τα ποτενσιόμετρα γραμμικού τύπου των 50 kΩ, τα οποία έχουν τοποθετηθεί σε προσαρμοσμένα μηχανικά στηρίγματα, ώστε να μετατρέπουν με ακρίβεια την κίνηση του οδηγού σε μεταβολές τάσης.

Η γωνιακή μετατόπιση του τιμονιού, που σε τυπικά επιβατικά οχήματα μπορεί να φτάσει τις ±540°, υφίσταται μηχανική μείωση μέσω συστήματος οδοντωτών τροχών, ώστε να περιορίζεται στις ±300°, δηλαδή στο πλήρες εύρος περιστροφής που μπορεί να δεχθεί το ποτενσιόμετρο χωρίς να υποστεί ζημιά. Επιπλέον, η μετάδοση της κίνησης αφήνει ένα μικρό περιθώριο ασφαλείας (padding), ώστε να αποφεύγεται η μηχανική καταπόνηση των άκρων του ποτενσιόμετρου. Όσον αφορά τα πεντάλ, η γραμμική κίνηση του ποδιού μετατρέπεται σε περιστροφική κίνηση μέσω συστήματος μοχλών, ώστε να μπορεί να ανιχνευθεί από τα ποτενσιόμετρα.

Οι αναλογικές αυτές είσοδοι καταλήγουν σε συγκεκριμένες θύρες του μικροελεγκτή ATmega32U4, με το τιμόνι συνδεδεμένο στην αναλογική είσοδο A0, τον συμπλέκτη στην A1, το φρένο στην A2 και το γκάζι στην A3. Στην περίπτωση του τιμονιού, η τάση που παράγεται μπορεί να κυμαίνεται πλήρως μεταξύ 0V και Vcc (5V), με το 0V να αντιστοιχεί σε πλήρη αριστερή στροφή, τα 2.5V σε κεντράρισμα και τα 5V σε πλήρη δεξιά στροφή.

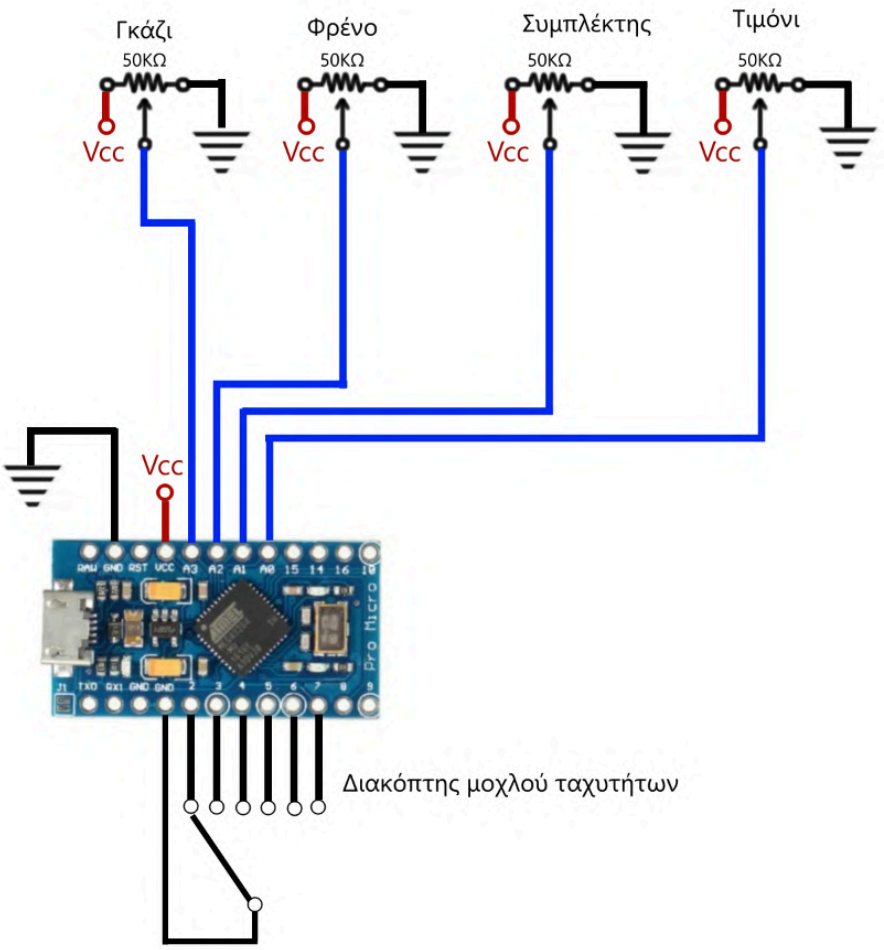
Αντιθέτως, τα πεντάλ λόγω της μηχανικής διαμόρφωσης αποδίδουν τάσεις σε περιορισμένο εύρος (2.35–3.89 V), το οποίο δύναται να μεταβληθεί κατά τη φάση ανάπτυξης αλλά και λόγω φυσιολογικής φθοράς. Για τον λόγο αυτό, έχει ενσωματωθεί στο λογισμικό σύστημα δυναμικής βαθμονόμησης (calibration system), το οποίο αντισταθμίζει διαφοροποιήσεις στο υλικό και εξασφαλίζει ακριβή και αξιόπιστη απόδοση του χειρισμού.

Πέραν των αναλογικών σημάτων, υπάρχουν και ψηφιακοί διακόπτες που χρησιμοποιούν τις εσωτερικές pull-up αντιστάσεις της θύρας D για να επικοινωνήσουν στον μικροελεγκτή την θέση του μοχλού ταχυτήτων

Πίνακας 4.1: Ανάλυση τάσεων στις ακίδες του μικροελεγκτή

Είσοδος (λειτουργία)	Arduino pin	ATmega32U4 port	Min τάση (V)	Max τάση (V)	**Γ**ωνία περιστροφής
Τιμόνι	A0	ADC0 / PF0	0.00	5.00	270.0°
Συμπλέκτης	A1	ADC1 / PF1	2.35	3.89	83.2°
Φρένο	A2	ADC2 / PF2	2.35	3.89	83.2°
Γκάζι	A3	ADC3 / PF3	2.35	3.89	83.2°
Σχέση 1	D2	PD2	0	Vcc	Δεν ορίζεται

Σχέση 2	D3	PD3	0	Vcc	Δεν ορίζεται
Σχέση 3	D4	PD4	0	Vcc	Δεν ορίζεται
Σχέση 4	D5	PD5	0	Vcc	Δεν ορίζεται
Σχέση 5	D6	PD6	0	Vcc	Δεν ορίζεται
Όπισθεν	D7	PD7	0	Vcc	Δεν ορίζεται



Σχήμα 2.2.3: Ηλεκτρονικό διάγραμμα του προτεινόμενου συστήματος

2.2.3.1 Ανάλυση του λογισμικού του μικροελεγκτή

Η βιβλιοθήκη Arduino Joystick Library αποτελεί ένα πακέτο επέκτασης, το οποίο αξιοποιεί την ενσωματωμένη δυνατότητα του μικροελεγκτή ATmega32U4 να λειτουργεί ως συσκευή USB. Η βιβλιοθήκη Joystick επεκτείνει αυτήν τη δυνατότητα παρέχοντας έναν έτοιμο προς χρήση USB HID descriptor, ο οποίος καθιστά δυνατή την παρουσίαση της πλατφόρμας Arduino ως joystick συσκευή σε κάθε υπολογιστή που ακολουθεί τα πρότυπα HID. Η συγκεκριμένη βιβλιοθήκη παρέχει στον

χρήστη τη δυνατότητα να ορίσει αναλογικούς άξονες (X, Y, Z, Rx, Ry, Rz), καθώς και ψηφιακά κουμπιά, τα οποία είναι συμβατά με οποιοδήποτε λογισμικό υποστηρίζει HID joystick είσοδο. Οι αναλογικές είσοδοι (μέσω του ADC του ATmega32U4) μετατρέπονται σε αντίστοιχες κινήσεις στους άξονες του joystick, ενώ τα ψηφιακά σήματα μεταφράζονται σε διακριτές εντολές (κουμπιά). Αυτό καθιστά τη βιβλιοθήκη ιδιαίτερα κατάλληλη για εφαρμογές προσομοιωτών, καθώς δεν απαιτείται η ανάπτυξη custom drivers. Η αναγνώριση της συσκευής από τα λειτουργικά συστήματα (Windows, Linux, macOS) είναι αυτόματη[15], [16].

Η επιλογή της βιβλιοθήκης Joystick στο πλαίσιο του παρόντος έργου βασίζεται σε δύο βασικούς παράγοντες: απλοποίηση της υλοποίησης και συμβατότητα με λογισμικό προσομοίωσης. Μειώνει την πολυπλοκότητα της αρχιτεκτονικής του συστήματος και περιορίζει τις πιθανότητες καθυστέρησης (latency) στη μεταφορά σήματος. Από την άλλη πλευρά, η χρήση ενός προτύπου USB HID σημαίνει ότι οποιοδήποτε εμπορικά διαθέσιμο λογισμικό προσομοίωσης που υποστηρίζει joystick είσοδο μπορεί να αναγνωρίσει το σύστημα, ενισχύοντας την ευχρηστία. Επιπλέον, η βιβλιοθήκη Joystick παρέχει τη δυνατότητα παραμετροποίησης του αριθμού των αξόνων, των κουμπιών και της συμπεριφοράς της συσκευής μέσω HID descriptors, γεγονός που καθιστά εφικτή την ακριβή προσαρμογή της στο υλικό του προσομοιωτή[15], [16]. Ακολουθεί η αναλυτική επεξήγηση του κώδικα:

- Εισαγωγή της βιβλιοθήκης Joystick και ορισμός εισόδων
- Δημιουργία αντικειμένου Joystick
  Δημιουργείται ένα αντικείμενο τύπου Joystick με USB αναγνωριστικό 0x12.
  Ορίζεται ότι η συσκευή θα αναγνωρίζεται ως τύπος joystick και δηλώνονται οι διαθέσιμοι άξονες και κουμπιά, σύμφωνα με τις παραμέτρους που περνιούνται στον constructor.
  Με αυτόν τον τρόπο, το λειτουργικό σύστημα θα βλέπει το Arduino σαν ένα χειριστήριο πολλών αξόνων.
- Ορίζεται η μεταβλητή gearboxPins η οποια αποθηκεύει τις ακίδες του μικροελεγκτή σε ενα array για να γίνει δυνατή η προσπέλαση τους.
- Συνάρτηση setup()
  Ορίζονται τα κατάλληλα pins ως PULL-UP είσοδοι.
  Εκκινείται η λειτουργία του joystick μέσω της εντολής Joystick.begin(), ενεργοποιώντας την USB HID επικοινωνία.
  Ανοίγει σειριακή επικοινωνία στα 9600 baud για σκοπούς διαγνωστικού ελέγχου και debugging (π.χ. για εκτύπωση των τιμών του τιμονιού).
- Κύρια λειτουργία – loop()
  - Ανάγνωση και χαρτογράφηση αξόνων Rx, Ry, Rz
  - Διαβάζει την αναλογική τιμή (0–1023) από το ποτενσιόμετρο.
  - Την αναπροσαρμόζει (map) σε νέα κλίμακα (515–1278), ώστε να ευθυγραμμιστεί με το εύρος που χρησιμοποιεί η βιβλιοθήκη Joystick.
  - Στέλνει την τιμή στον αντίστοιχο άξονα του USB joystick.
- Ανάγνωση και επεξεργασία σήματος τιμονιού (X-Axis):
  Διαβάζεται η τιμή από το ποτενσιόμετρο του τιμονιού (A0).
  Εκτυπώνεται μέσω της σειριακής θύρας για σκοπούς παρακολούθησης.
  Χαρτογραφείται εκ νέου στο πλήρες εύρος (0–1023) και καταχωρείται ως X-άξονας του joystick.
- Ανάγνωση και μετάδοση κατάστασης κιβωτίου :
  Διαβάζεται η τρέχουσα κατάσταση όλων των διακοπτών κιβωτίου.

Μετατρέπεται λογικά με το ! για αντιστροφή της κατάστασης (εφόσον χρησιμοποιούμε pullup εισόδους)
Αν η τρέχουσα κατάσταση διαφέρει από την προηγούμενη (lastGearboxState), ενημερώνεται η τιμή lastGearboxState και αποστέλλεται το νέο state στο σύστημα ως κουμπί καλώντας την συνάρτηση Joystick.setButton.

- delay(1);
  Εισάγεται μια πολύ μικρή καθυστέρηση (1 ms) για σταθερότητα στην επεξεργασία των εισόδων και αποφυγή υπερβολικού φορτίου στη θύρα USB.

```
#include <Joystick.h>

#define steering A0
#define clutch A1
#define brake A2
#define gas A3
#define gearbox1 2
#define gearbox2 3
#define gearbox3 4
#define gearbox4 5
#define gearbox5 6
#define gearboxR 7


int clutch_ = 0;
int brake_ = 0;
int gas_ = 0;
int steering_ = 0;

int lastGearboxState[6] = {0};



Joystick_ Joystick(0x12, JOYSTICK_TYPE_JOYSTICK, 6,
0,true,true,true,true,true,true,true,true,true,true,true);


const bool initAutoSendState = true;

const byte gearboxPins[6] = {
  gearbox1,
  gearbox2,
  gearbox3,
  gearbox4,
  gearbox5,
  gearboxR
};

void setup()
{
  pinMode(gearbox1,INPUT_PULLUP);
  pinMode(gearbox2,INPUT_PULLUP);
  pinMode(gearbox3,INPUT_PULLUP);
  pinMode(gearbox4,INPUT_PULLUP);
  pinMode(gearbox5,INPUT_PULLUP);
```

```
  pinMode(gearboxR,INPUT_PULLUP);

  Joystick.begin();
  Serial.begin(9600);

}

void loop()
{
  clutch_ = analogRead(clutch);
  clutch_ = map(clutch_,0,1023,515,1278);
  Joystick.setRxAxis(clutch_);

  brake_ = analogRead(brake);
  brake_ = map(brake_,0,1023,515,1278);
  Joystick.setRyAxis(brake_);

  gas_ = analogRead(gas);
  gas_ = map(gas_,0,1023,515,1278);
  Joystick.setRzAxis(gas_);

  steering_ = analogRead(steering);
  Serial.println(steering_);

  steering_ = map(steering_,0,1023,0,1023);

  Joystick.setXAxis(steering_);

  for (int i = 0; i < 6; i++) {
    int currentGearboxState = !digitalRead(gearboxPins[i]);

    if (currentGearboxState != lastGearboxState[i]) {
      Joystick.setButton(i, currentGearboxState);
      lastGearboxState[i] = currentGearboxState;
    }
  }

  delay(1);
}
```

2.3 Υλοποίηση μηχανολογικού εξοπλισμού

Η μηχανολογική διάσταση του προσομοιωτή οχημάτων αποτελεί έναν από τους πιο καθοριστικούς παράγοντες για την επιτυχία του εγχειρήματος, καθώς η ποιότητα της εμπειρίας του χρήστη εξαρτάται άμεσα από την ορθότητα και την αξιοπιστία των μηχανικών υποσυστημάτων. Η κατασκευή οφείλει να αναπαράγει με τη μέγιστη δυνατή ακρίβεια τα φυσικά χαρακτηριστικά ενός πραγματικού αυτοκινήτου, ώστε η αίσθηση της οδήγησης να είναι αληθοφανής. Αυτό συνεπάγεται στην σχεδιαση των τμημάτων ετσι ωστε να αποδιδουν σωστά τις δυνάμεις που δέχεται ο οδηγός, όπως η αντίσταση στο τιμόνι και η αντίσταση του πεντάλ, και την ακριβή καταγραφή των μετρήσεων που προκύπτουν από την αλληλεπίδραση του χρήστη με τη συσκευή. Η αξιοπιστία, η ανθεκτικότητα και η δυνατότητα επαναλαμβανόμενης λειτουργίας χωρίς φθορές ή αστοχίες συνιστούν εξίσου κρίσιμες παραμέτρους, καθώς εξασφαλίζουν τη σταθερή ποιότητα των δεδομένων και τη συνεχή διαθεσιμότητα του συστήματος για ερευνητικούς ή εκπαιδευτικούς σκοπούς.

Το κεφάλαιο αυτό περιλαμβάνει τη μελέτη, σχεδίαση και συναρμολόγηση των μηχανικών τμημάτων του προσομοιωτή, όπως το υποσύστημα τιμονιού, το υποσύστημα των πεντάλ, και τον διακόπτη drive/reverse. Επίσης αποσκοπεί στη σωστή τοποθέτηση των αισθητήρων και σχεδίαση των σημείων πρόσδεσης αυτών.

Για τον σχεδιασμό των μηχανολογικών μερών που παρουσιάζονται στο παρόν κεφάλαιο χρησιμοποιήθηκε το λογισμικό Blender. Παρόλο που δεν πρόκειται για ένα παραδοσιακό παραμετρικό CAD περιβάλλον, η ευελιξία του, σε συνδυασμό με τις κατάλληλες επεκτάσεις, το καθιστά ιδιαίτερα κατάλληλο για την ανάπτυξη απλών περιφερειακών στοιχείων ενός προσομοιωτή οχήματος, όπου δεν απαιτείται προηγμένη ανάλυση πεπερασμένων στοιχείων (FEM/FEA). Το Blender παρέχει εργαλεία που διευκολύνουν τον γρήγορο και ακριβή ορισμό γεωμετριών, καθώς και δυνατότητες δημιουργίας μηχανικά ορθών γραναζιών και άλλων εξαρτημάτων. Η ύπαρξη μετρητικών εργαλείων συμβάλλει ουσιαστικά στη διασφάλιση ότι τα σχέδια ανταποκρίνονται στις πραγματικές διαστάσεις των εξαρτημάτων, γεγονός που ενισχύει την αξιοπιστία κατά τη μετάβαση από τον ψηφιακό στον φυσικό σχεδιασμό. Επιπλέον, οι προηγμένες δυνατότητες κίνησης και κινούμενης απεικόνισης του λογισμικού επιτρέπουν την προσομοίωση της λειτουργίας των μηχανικών μερών, προσφέροντας τη δυνατότητα εντοπισμού και διόρθωσης σχεδιαστικών αστοχιών πριν την κατασκευή.

2.3.1 Υποσύστημα Τιμονιού

Το τιμόνι αποτελεί ίσως το πιό καθοριστικό στοιχείο για την αίσθηση ρεαλισμού σε έναν προσομοιωτή, καθώς είναι το βασικό σημείο επαφής του οδηγού με το όχημα. Ο μηχανολογικός σχεδιασμός του απαιτεί την ενσωμάτωση μηχανισμών ικανούς να αναπαράγουν την αντίσταση και τη γωνία περιστροφής που συναντώνται σε ένα πραγματικό αυτοκίνητο. Επιπλέον, η στιβαρότητα της κατασκευής και η εργονομική τοποθέτηση του συστήματος είναι κρίσιμες, καθώς η συνεχής χρήση μπορεί να προκαλέσει φθορές αν δεν έχουν προβλεφθεί οι κατάλληλες μηχανολογικές αντοχές.

Ένας από τους βασικούς στόχους του υποσυστήματος του τιμονιού είναι η επίτευξη κατάλληλης μείωσης λόγου περιστροφής μέσω γραναζωτών μηχανισμών. Στα περισσότερα οχήματα παραγωγής, το τιμόνι διαθέτει εύρος περιστροφής 1080 μοιρών, δηλαδή πραγματοποιεί περίπου 1,5 πλήρεις στροφές προς κάθε κατεύθυνση (σύνολο τριών κύκλων, $360° \times 3 = 1080°$). Αντιθέτως, τα περισσότερα εμπορικά διαθέσιμα ποτενσιόμετρα έχουν όριο περιστροφής περίπου 300 μοιρών. Από αυτή τη διαφορά προκύπτει η απαίτηση της μηχανικής διάταξης να μεταφράζει την περιστροφή του τιμονιού (1080°) σε αντίστοιχη περιστροφή 300° στον άξονα του ποτενσιόμετρου. Η προσαρμογή αυτή υλοποιείται μέσω συστήματος οδοντωτών τροχών, το οποίο οφείλει να είναι συμπαγές, ώστε να

μην καταλαμβάνει υπερβολικό όγκο και να μεταφέρει την κίνηση με ελάχιστες απώλειες. Με αυτόν τον τρόπο η μεταφορά της κίνησης είναι όσο το δυνατόν πιο ακριβής. Ιδιαίτερη προσοχή δίνεται στη σωστή συναρμογή και στην ελαχιστοποίηση του διάκενου (backlash) μεταξύ των δοντιών, προκειμένου η γωνιακή μετατόπιση που εφαρμόζει ο οδηγός να αποδίδεται χωρίς ανακρίβειες στον τελικό άξονα που κινεί το ποτενσιόμετρο.

Ο απαιτούμενος λόγος μετάδοσης για το υποσύστημα του τιμονιού προκύπτει άμεσα από τη σχέση μεταξύ της γωνιακής μετατόπισης του άξονα του τιμονιού και της μέγιστης επιτρεπόμενης γωνίας περιστροφής του ποτενσιόμετρου. Δεδομένου ότι το τιμόνι στα περισσότερα οχήματα παραγωγής διαθέτει 1080° συνολικής περιστροφής, ενώ το ποτενσιόμετρο μπορεί να περιστραφεί έως 300°, η αναγκαία σχέση μείωσης υπολογίζεται ως 1080/300 = 3,6.

Θα χρησιμοποιηθεί ο ελάχιστος απαιτούμενος αριθμός οδοντωτών τροχών για να επιτευχθεί ο επιλεγμένος λόγος μετάδοσης. Αρχικά σχεδιάστηκε το σύστημα χρησιμοποιώντας δύο τροχούς, ο αρχικός με 13 δόντια και ο τελικός με 47. Αυτή η διάταξη κρίθηκε λιγότερο αποτελεσματική για δύο λόγους. Πρώτον, το φυσικό μέγεθος του τελικού τροχού κρίθηκε υπερβολικά ογκώδης για πρακτικούς λόγους, και δεύτερον ο μικρός αριθμός των δοντιών καθιστά το διάκενο μεταξύ των οδοντωτών τροχών αρκετά μεγάλο ωστε να παρατηρηθούν ανακρίβειες στη μετάδοση της κίνησης του οδηγού στον άξονα που κινεί το ποτενσιόμετρο.

Ο αμέσως επόμενος αριθμός τροχών ειναι το 4. Επιλέγεται ο αριθμός δοντιών για τον αρχικό τροχό να ειναι 24, και βάση αυτού του αριθμού και του λόγου μείωσης (3.6/1) υπολογίζονται οι παράμετροι των υπόλοιπων οδοντωτών τροχών.

Η εξίσωση που ικανοποιεί τον απαιτούμενο λόγο μετάδοσης είναι η εξής:

$$\frac{T2}{24} * \frac{T4}{T3} = 3.6 = \frac{18}{5} \quad (2.31)$$

$$T2 * T4 = \frac{432}{5} T3 \quad (2.3.2)$$

προκύπτει ότι το T3 πρέπει να είναι πολλαπλάσιο του 5. Εστω T3 = 5k. Προκύπτει

$$T2 * T4 = 432k \quad (2.3.3)$$

Επιλέχθηκε ο αριθμός k =4 που ικανοποιεί τη σχέση. Οταν k=4 ισχύει:

T1= 24, T2 = 36, T3 = 20, T4 = 48

Επαληθεύοντας:

$$\frac{36}{24} * \frac{48}{20} = 3.6 \quad (2.3.4)$$

Μετά τον υπολογισμό του αριθμού δοντιών απαιτείται ο υπολογισμός της παραμέτρου module. Η παράμετρος module είναι μέτρο του μεγέθους των δοντιών και τυποποιεί τη γεωμετρία των γραναζιών, ώστε διαφορετικά γρανάζια να μπορούν να εμπλέκονται σωστά. Ισχύει η σχέση:

$$OD = (z + 2)\cdot m \quad (2.3.5)$$

όπου:

OD = εξωτερική διάμετρος

z = αριθμός δοντιών

m = module

Επιλέχθηκε η εξωτερική διάμετρος του πρώτου οδοντωτού τροχού να ειναι 40 χιλιοστά. Βάση αυτού υπολογίζεται η παράμετρος module, όπου προκύπτει ότι:

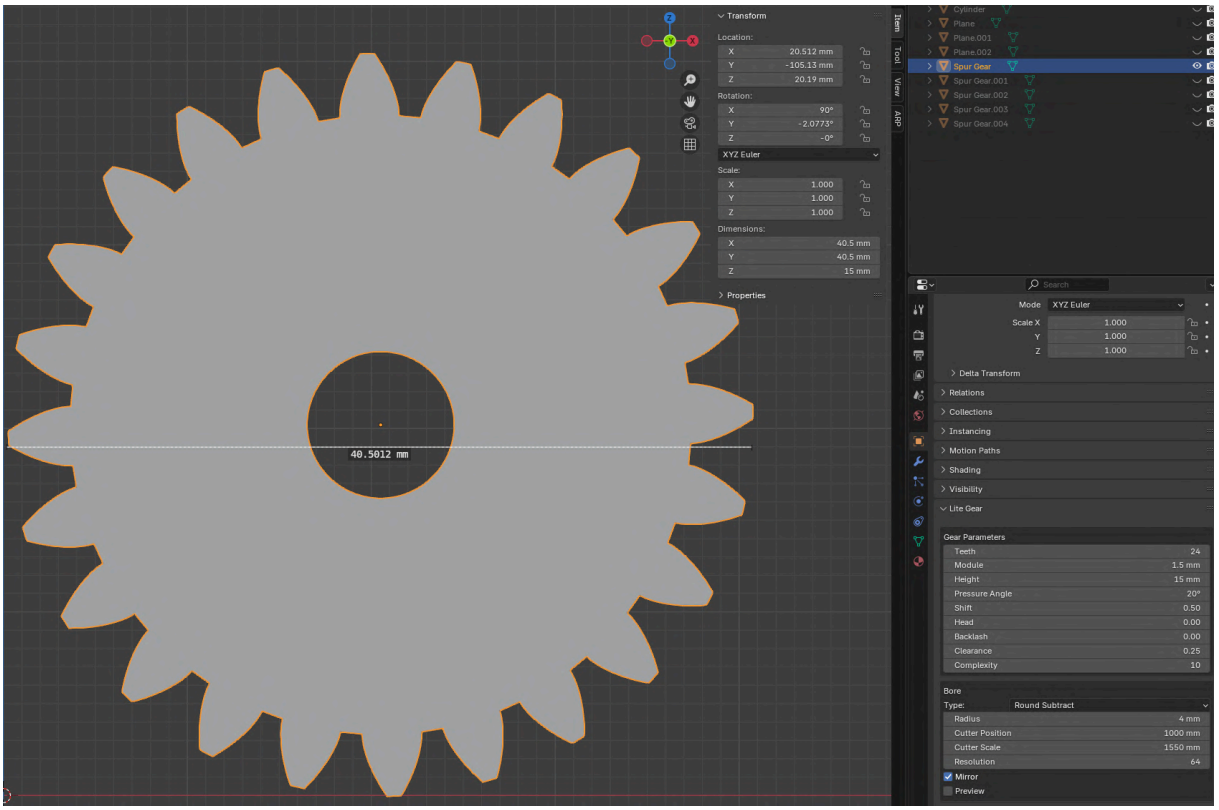
$$m = \frac{40}{24+2} = 1.5385\ mm \quad (2.3.6)$$

Η τιμή στρογγυλοποιείται στο m = 1.5mm

Αφού ολοκληρωθούν οι υπολογισμοί για τις απαιτούμενες παραμέτρους των τροχών, το επόμενο βήμα είναι ο σχεδιασμός των ίδιων των γραναζιών. Για τον σκοπό αυτό χρησιμοποιείται στο Blender η επέκταση Precision Gear, η οποία έχει αναπτυχθεί ειδικά για τη δημιουργία παραμετρικών οδοντωτών τροχών. Το συγκεκριμένο εργαλείο απλοποιεί σε μεγάλο βαθμό τη διαδικασία σχεδίασης, καθώς επιτρέπει στον χρήστη να ορίσει κρίσιμες παραμέτρους όπως ο αριθμός των δοντιών, η διάμετρος βήματος, η γωνία πίεσης και άλλες γεωμετρικές ιδιότητες. Με την εισαγωγή των κατάλληλων τιμών, το πρόσθετο δημιουργεί αυτόματα ένα γεωμετρικά ακριβές μοντέλο του γραναζιού, μειώνοντας τον χρόνο σχεδίασης και εξασφαλίζοντας παράλληλα μηχανική ορθότητα. Επιπλέον, η δυνατότητα απευθείας οπτικοποίησης των παραγόμενων μοντέλων επιτρέπει τον εύκολο έλεγχο και την άμεση προσαρμογή των σχεδιαστικών επιλογών, γεγονός που καθιστά το Precision Gear ένα ιδιαίτερα χρήσιμο εργαλείο στη διαδικασία ανάπτυξης του μηχανολογικού συστήματος.

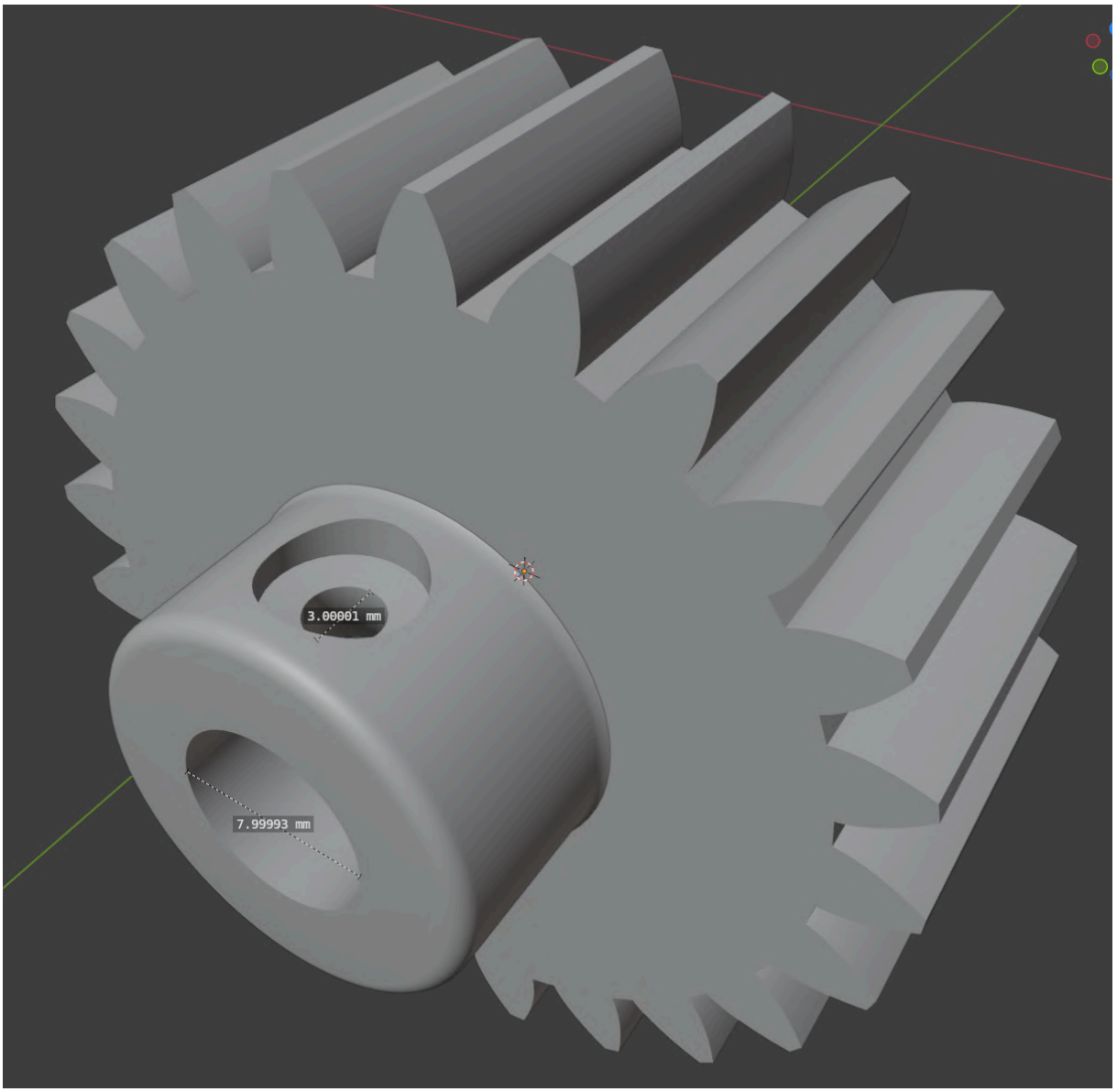
- Δημιουργήθηκε ο πρώτος οδοντωτός τροχός ορίζοντας τις παραμέτρους που υπολογίστηκαν προηγουμένως.
- Επιλέχθηκε η παράμετρος height που αντιστοιχεί στο πάχος του οδοντωτού τροχού στα 15 χιλιοστά
- Επιλέχθηκε η παράμετρος bore radius στα 4 χιλιοστά, ωστε ο τροχός να εμπλέκεται σωστά με άξονα διαμέτρου 8 χιλιοστών.
- Επιλέχθηκε η παράμετρος backlash στα 0.01 mm για να αντισταθμίσει τυχόν αστοχίες (tolerances) κατά τη διαδικασία της τρισδιάστατης εκτύπωσης των οδοντωτών τροχών.

Παρομοίως σχεδιάστηκαν οι τρεις εναπομείναντες οδοντωτοί τροχοί αλλάζοντας μόνο την παράμετρο που αντιστοιχεί στον αριθμό δοντιών.



Σχήμα 2.3.1.1: Παραμετρική σχεδίαση οδοντωτών τροχών

Μετά την ολοκλήρωση του σχεδιασμού των οδοντωτών τροχών, το επόμενο βήμα αφορά τη σχεδίαση του μηχανισμού στερέωσης στον άξονα. Ο μηχανισμός αυτός έχει σχεδιαστεί έτσι ώστε να επιτρέπει την αξιόπιστη και σταθερή σύνδεση του γραναζιού με τον άξονα διαμέτρου 8 mm, χρησιμοποιώντας μία βίδα διαμέτρου 3 mm. Η επιλογή αυτής της διάταξης εξασφαλίζει ότι η περιστροφή του γραναζιού μεταφέρεται πλήρως στον άξονα χωρίς ολίσθηση ή απώλεια κίνησης. Παράλληλα, η χρήση βιδωτού μηχανισμού διευκολύνει τη συναρμολόγηση και την αποσυναρμολόγηση για συντήρηση ή προσαρμογές.



Σχήμα 2.3.1.2: Μηχανισμός στερέωσης οδοντωτού τροχού σε άξονα

Το επόμενο βήμα στη διαδικασία σχεδιασμού είναι η ευθυγράμμιση των κεντρικών σημείων όλων των γραναζιών ώστε να εμπλέκονται σωστά μεταξύ τους και να τοποθετούνται με ακρίβεια στο σώμα της συναρμογής. Για τον καθορισμό της κατάλληλης απόστασης μεταξύ των κέντρων κάθε ζεύγους γραναζιών εφαρμόζεται η γνωστή σχέση που βασίζεται στις διαμέτρους βήματος και τις διορθώσεις προφίλ (profile shift). Συγκεκριμένα, η απόσταση a μεταξύ των κέντρων δύο γραναζιών υπολογίζεται με τον τύπο:

$$a = \frac{d1 + d2}{2} + m \cdot 2s \qquad (2.3.7)$$

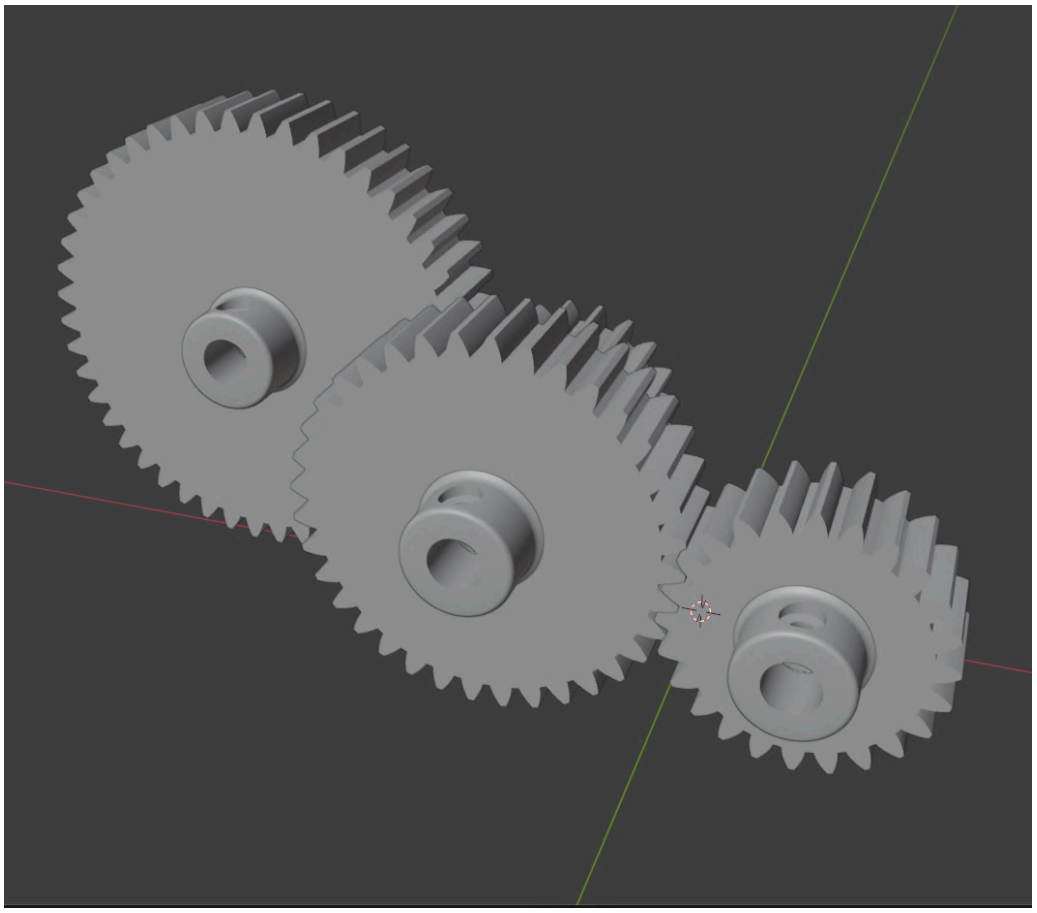
Οπου s ειναι η παράμετρος μετατόπισης προφίλ και d1 και d2 είναι οι διάμετροι βήματος (pitch diameters) του πρώτου και του δεύτερου γραναζιού αντίστοιχα, οι οποίες ορίζονται ως:

$$d1 = m \cdot z1\,,\ d2 = m \cdot z2 \qquad (2.3.8)$$

με $z1$ και z 2 τον αριθμό δοντιών του πρώτου και του δεύτερου οδοντωτού τροχού αντίστοιχα.

αντικαθιστώντας εχουμε: α1 = 46.5mm, α2 = 52.5mm

οπου α1 η απόσταση μεταξύ των αξόνων του πρώτου και του δεύτερου οδοντωτού τροχού και α2 η απόσταση μεταξύ του τρίτου και του τέταρτου τροχού. Η απόσταση μεταξύ του δεύτερου και του τρίτου τροχού δεν υπολογίζονται, διοτι αυτοι οι τροχοί μοιράζονται τον ίδιο άξονα.



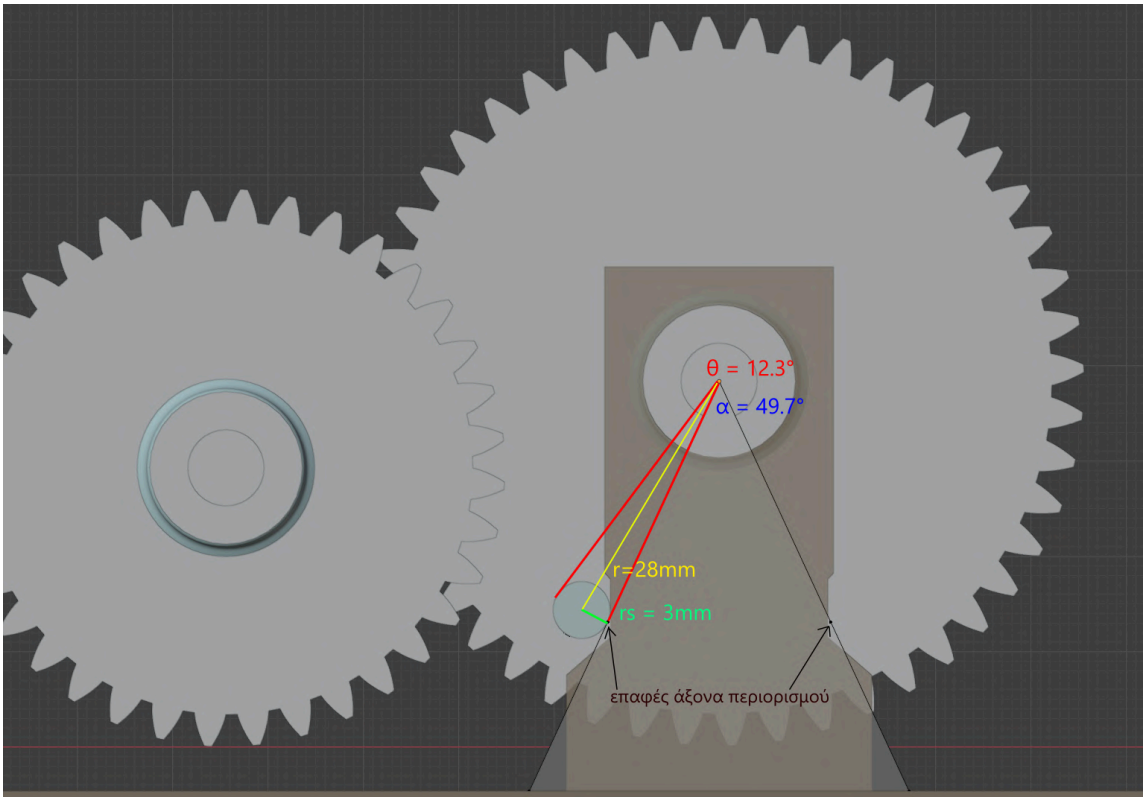
Σχήμα 2.3.1.3: Σωστή εμπλοκή οδοντωτών τροχών

Πέρα από την επίτευξη του επιθυμητού λόγου μετάδοσης, είναι απαραίτητο να υλοποιηθεί και ένας μηχανισμός περιορισμού της περιστροφής του τιμονιού ώστε να μην ξεπερνά τις 1080°. Η υπέρβαση αυτού του ορίου δεν είναι μόνο μη ρεαλιστική για την προσομοίωση, αλλά ενδέχεται να προκαλέσει και ζημιά στο ποτενσιόμετρο που χρησιμοποιείται για την καταγραφή της γωνίας περιστροφής. Για την εφαρμογή αυτού του περιορισμού, αξιοποιείται το τελικό γρανάζι της μετάδοσης. Συγκεκριμένα, τοποθετείται ένας άξονας σε ένα συγκεκριμένο σημείο του τελικού γραναζιού και η συναρμογή κατασκευάζεται με τέτοιο τρόπο ώστε να εμποδίζει τον άξονα να κινηθεί πέρα από προκαθορισμένα όρια προς τα αριστερά και προς τα δεξιά. Με αυτόν τον τρόπο περιορίζεται αποτελεσματικά η περιστροφή του τελικού γραναζιού σε 300°, εξασφαλίζοντας την ασφάλεια του ποτενσιόμετρου και την πιστή αναπαραγωγή της πραγματικής γωνίας περιστροφής του τιμονιού. Για τον ακριβή προσδιορισμό της θέσης τοποθέτησης του άξονα περιορισμού απαιτούνται τα παρακάτω βήματα:

Αρχικα επιλέχθηκε η απόσταση r = 28mm απο το κέντρο του τελικού οδοντωτού τροχού στην οποία θα τοποθετηθεί ο άξονας περιορισμού. Επειτα επιλέχθηκε η ακτίνα rs = 3mm του άξονα περιορισμού. Για να βρεθεί η γωνία θ που περιορίζει την κίνηση ο άξονας χρησιμοποιείται ο τύπος:

$$\theta = 2arcsin\frac{rs}{r} \cdot \frac{180}{\pi} = 12.3 \qquad (2.3.9)$$

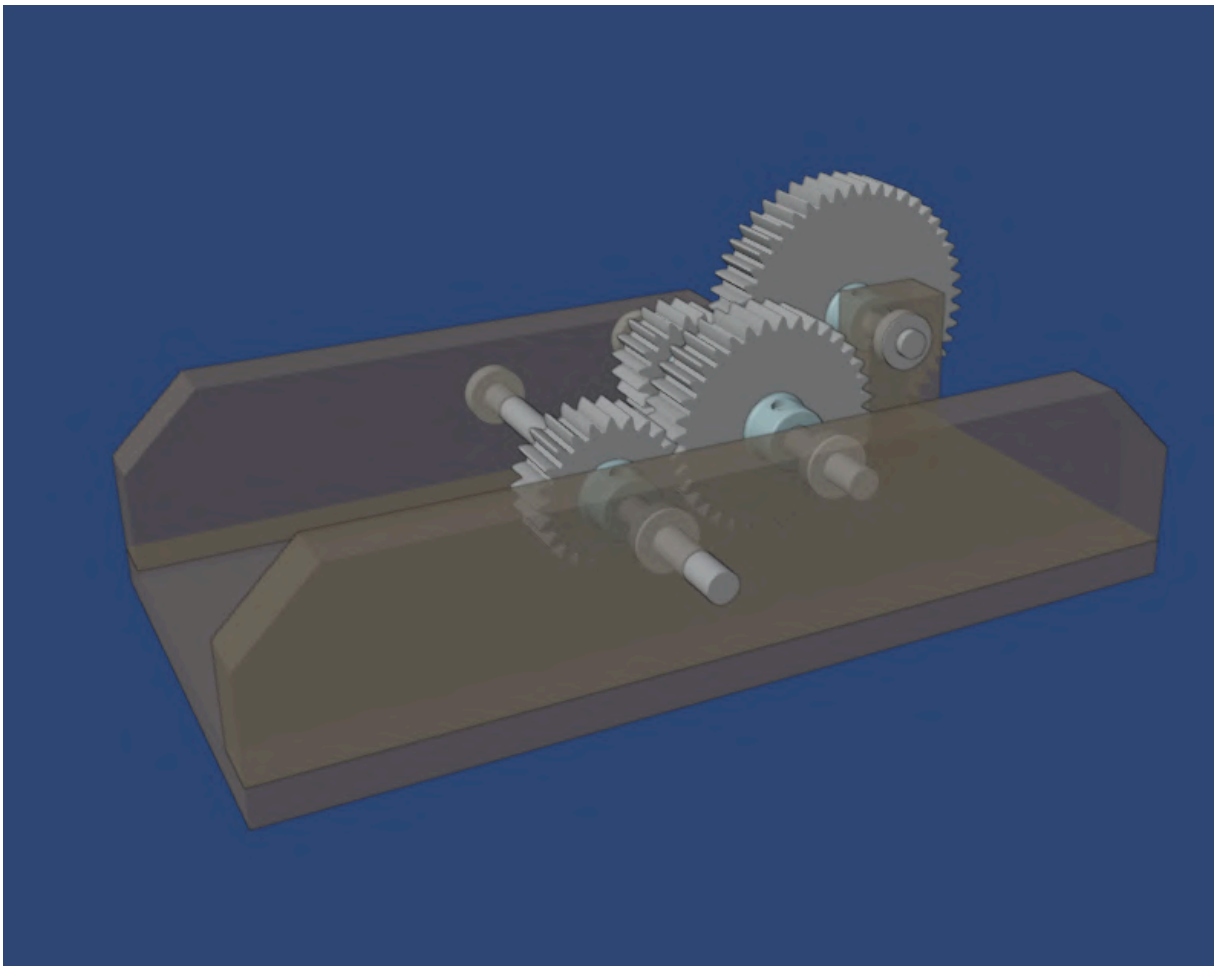
Για να βρεθούν τα σημεία της συναρμογής που θα περιορίζουν την κίνηση του άξονα χρησιμοποιείται μια γεωμετρική προσέγγιση βασισμένη σε ένα φανταστικό τρίγωνο. Το τρίγωνο αυτό σχηματίζεται με κέντρο το κέντρο του τελικού γραναζιού, ενώ μία από τις γωνίες του αντιστοιχεί στην γωνία περιστροφής που υπερβαίνει το επιτρεπτό όριο, αφαιρώντας την γωνία θ που καταλαμβάνει ο ίδιος ο άξονας, δηλαδή α = 360° - 298° - 12.3° = 49.7°. Η γωνία α των 49.7° τοποθετείται στο κέντρο του τελικού γραναζιού, και με βάση αυτό το τρίγωνο είναι εύκολο να καθοριστούν τα ακριβή σημεία. Συγκεκριμένα, επιλέγεται ένα σημείο πάνω σε κάθε μία από τις πλευρές του τριγώνου που αντιστοιχεί στην ακτίνα r, και βάση αυτών των σημείων σχεδιάζεται η συναρμογή ωστε να περιορίζει την κίνηση του άξονα περιορισμού. Με αυτήν τη διαδικασία εξασφαλίζεται ότι ο άξονας θα τοποθετηθεί σε θέση που περιορίζει με ακρίβεια την περιστροφή του τελικού γραναζιού στις 298 μοίρες, εξασφαλίζοντας έτσι τον έλεγχο της γωνίας περιστροφής του τιμονιού και την προστασία του ποτενσιόμετρου.



Σχήμα 2.3.1.4: Άξονας περιορισμού στροφής

Το επόμενο στάδιο του σχεδιασμού αφορά την τοποθέτηση των αξόνων κάθε γραναζιού και τον καθορισμό των θέσεων στο πλαίσιο (chassis) όπου θα τοποθετηθούν τα ρουλεμάν στήριξης. Η σωστή επιλογή και διάταξη των ρουλεμάν είναι κρίσιμη, καθώς εξασφαλίζει την ομαλή περιστροφή των αξόνων, μειώνει τις τριβές και αυξάνει τη συνολική αξιοπιστία της συναρμογής. Για τη συγκράτηση των αξόνων στο πλαίσιο επιλέχθηκε το ρουλεμάν τύπου 688ZZ, το οποίο διαθέτει εσωτερική διάμετρο 8 mm και εξωτερική διάμετρο 16 mm. Η επιλογή αυτού του συγκεκριμένου τύπου έγινε καθώς

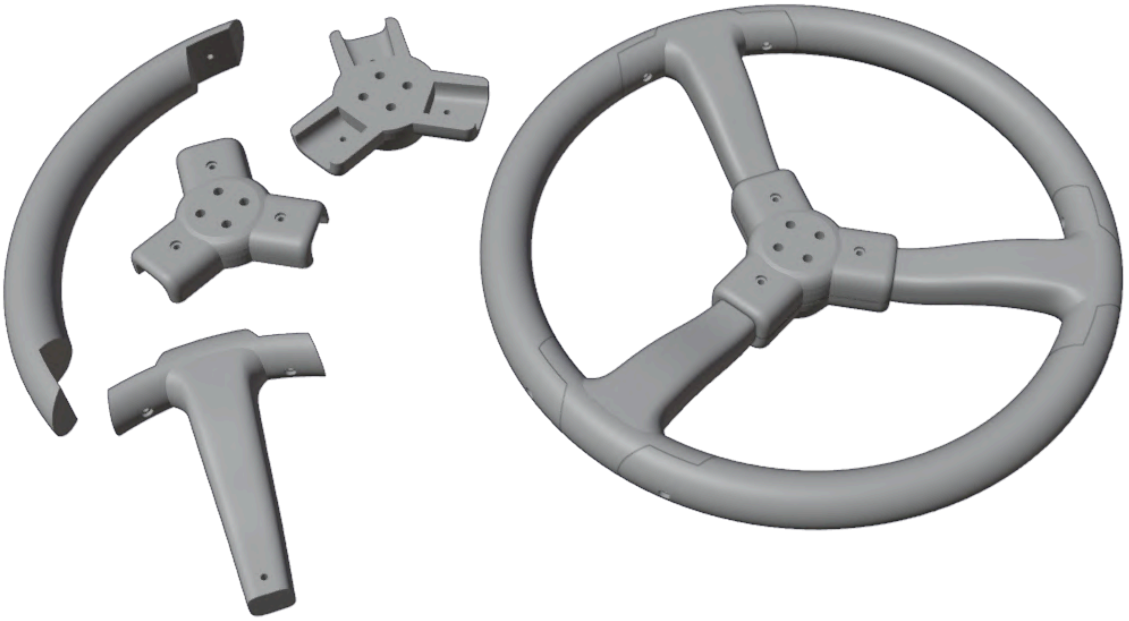
συνδυάζει μικρές διαστάσεις με ικανότητα αντοχής σε υψηλές ταχύτητες περιστροφής, ενώ το μεταλλικό κάλυμμά του προστατεύει τον μηχανισμό από σκόνη και μικροσωματίδια, παρατείνοντας τη διάρκεια ζωής του. Η στερέωση των αξόνων με χρήση των 688ZZ διασφαλίζει την μηχανική σταθερότητα της διάταξης και επιτρέπει την ακριβή μετάδοση κίνησης μεταξύ των γραναζιών χωρίς ανεπιθύμητες αποκλίσεις.



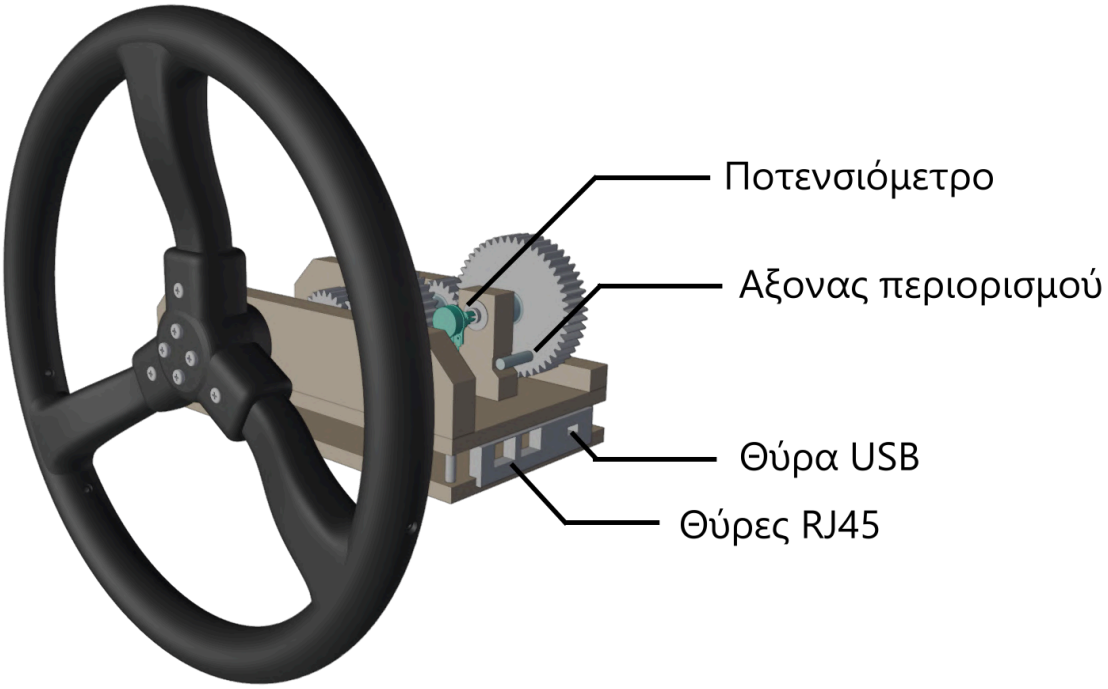
Σχήμα 2.3.1.5: Στερέωση οδοντωτών τροχών με ρουλεμάν

Το επόμενο στάδιο αφορά τον σχεδιασμό του ίδιου του τιμονιού, το οποίο αποτελεί το κύριο σημείο αλληλεπίδρασης του οδηγού με τον προσομοιωτή. Οι απαιτήσεις που τίθενται είναι ιδιαίτερα συγκεκριμένες: Το τιμόνι οφείλει να είναι εργονομικό και να διαθέτει διαστάσεις παρόμοιες με εκείνες που συναντώνται σε οχήματα μαζικής παραγωγής, ώστε η εμπειρία χρήσης να είναι όσο το δυνατόν πιο ρεαλιστική. Παράλληλα, πρέπει να προσαρμόζεται στον άξονα διαμέτρου 8 mm που χρησιμοποιείται στη μονάδα του τιμονιού, εξασφαλίζοντας σταθερότητα και ακαμψία κατά τη λειτουργία. Ένα επιπλέον κριτήριο σχεδιασμού σχετίζεται με την κατασκευασιμότητα μέσω τρισδιάστατης εκτύπωσης, όπου οι περισσότεροι εκτυπωτές παρουσιάζουν περιορισμούς ως προς το μέγιστο μέγεθος εκτυπώσιμων αντικειμένων. Για να αντιμετωπιστεί αυτό το πρόβλημα, το τιμόνι σχεδιάστηκε ώστε να αποτελείται από οκτώ επιμέρους τμήματα, τα οποία συναρμολογούνται μεταξύ τους με βίδες διαμέτρου 3 mm. Το κεντρικό τμήμα (hub) αποτελείται από δύο ξεχωριστά κομμάτια, ενώ το σύνολο συμπληρώνεται από τρία «μπράτσα» (spokes) και τρία ενδιάμεσα τμήματα, τα οποία ενώνονται για να σχηματίσουν τη συνεχή περιφέρεια του τιμονιού. Με αυτήν την προσέγγιση

επιτυγχάνεται αφενός η συμβατότητα με τις δυνατότητες των περισσότερων 3D εκτυπωτών και αφετέρου η διατήρηση της τυπικής μορφής και εργονομίας ενός κανονικού τιμονιού αυτοκινήτου.

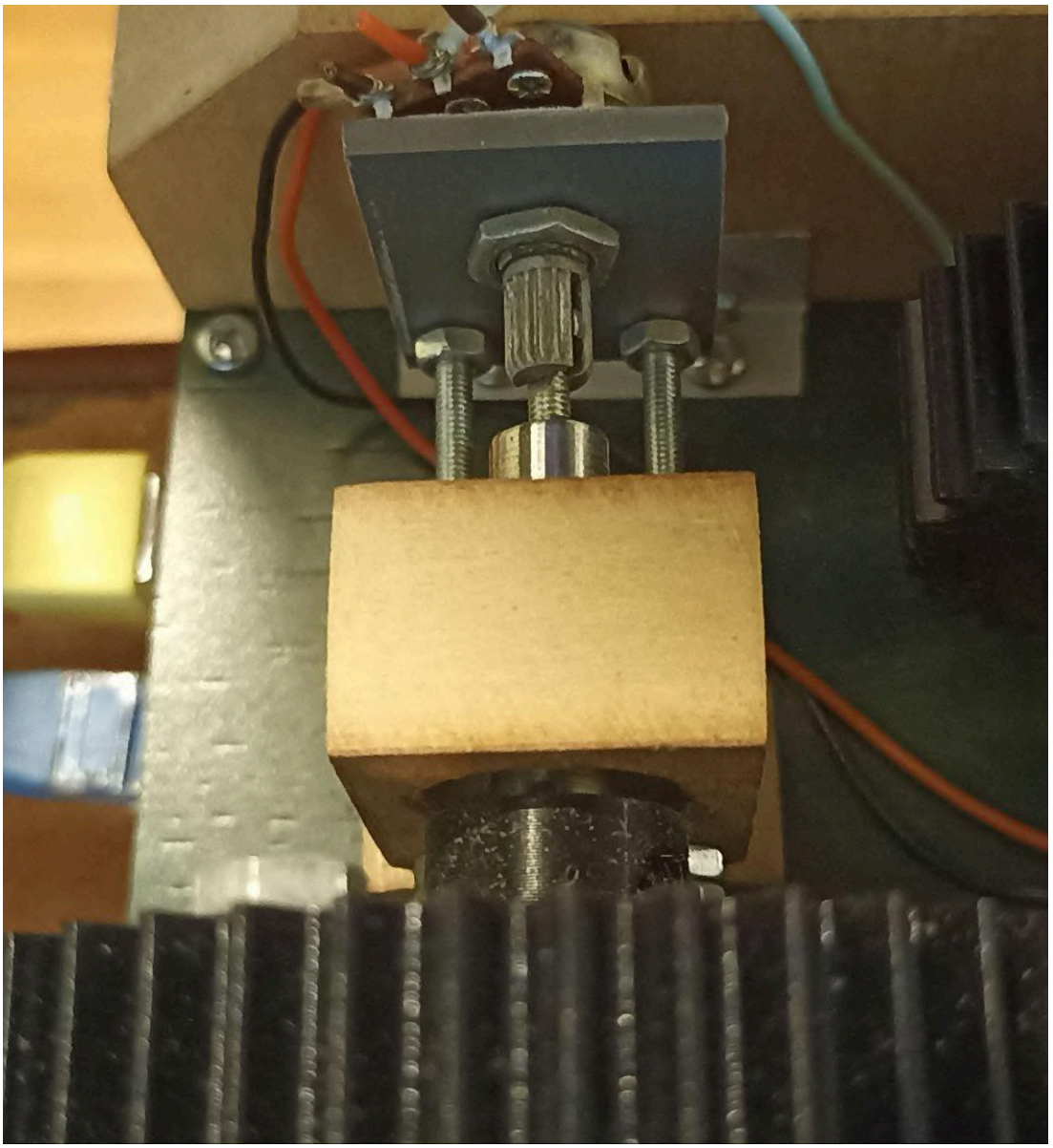


Σχήμα 2.3.1.6: Σχέδιο τιμονιού

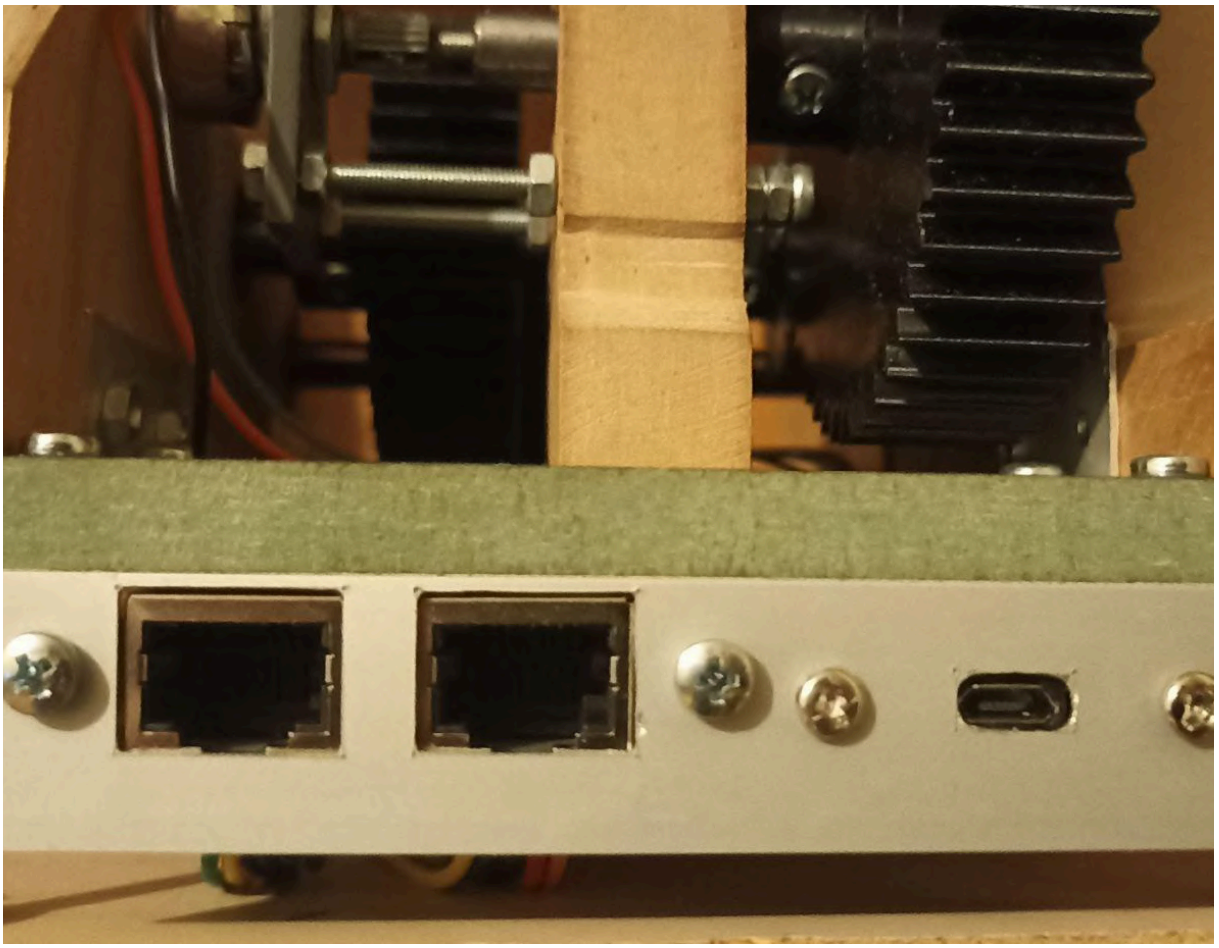


Σχήμα 2.3.1.7: Τελική σχεδίαση υποσυστήματος τιμονιού

Σχήμα 2.3.1.8: Τελική υλοποίηση υποσυστήματος τιμονιού



Σχήμα 2.3.1.9: ποτενσιόμετρο τιμονιού



Σχήμα 2.3.1.10: Θύρες I/O της συσκευής

2.3.2 Υποσύστημα Πεντάλ

Η σχεδίαση μιας ψηφιακής μονάδας πεντάλ για εφαρμογή σε έναν προσομοιωτή οχημάτων απαιτεί έμφαση στον ρεαλισμό, στην απόκριση και την αίσθηση (feedback) ώστε να αναπαράγεται αποτελεσματικά η εμπειρία της πραγματικής οδήγησης από το σύστημα. Οι βασικές απαιτήσεις περιλαμβάνουν:

- Την ακριβή ανίχνευση και ρύθμιση της δύναμης που εφαρμόζονται στα πεντάλ, γεγονός που καθιστά απαραίτητη τη χρήση αισθητήρων που μπορούν να διαβάσουν αυτήν την πληροφορία και να τη στείλουν όσο το δυνατόν πιο αναλλοίωτη στον μικροελεγκτή.
- Η εργονομία είναι μια κρίσιμη απαίτηση για ένα τέτοιο σύστημα, καθώς η φυσική διάταξη των πεντάλ πρέπει να αντικατοπτρίζει την αυθεντική γεωμετρία ενός οχήματος, για την καλλιέργεια της σωστής μυικής μνήμης. Με αυτόν τον τρόπο οι δεξιότητες είναι ομαλά μεταβιβάσιμες στον χειρισμό ενός κανονικού οχήματος.
- Η δυνατότητα βαθμονόμησης της μονάδας από το λογισμικό είναι επίσης απαραίτητο χαρακτηριστικό για δύο λόγους. Πρώτον, επειδή παράγοντες όπως μικροί κραδασμοί, διαφορές στη θερμοκρασία και λοιπές δυνάμεις που θα εφαρμόζονται στη μονάδα ενδέχεται να προκαλέσουν μικροαλλαγές, είτε στα μηχανικα χαρακτηριστικά των σημείων πρόσδεσης των αισθητήρων, είτε στα ηλεκτρικά χαρακτηριστικά των αισθητήρων. Δεύτερον, επειδή ενδέχεται να υπάρξουν αλλαγές στην κατασκευή της μονάδας κατά τη διαδικασία της

ανάπτυξης, δηλαδή μια ενημερωμένη έκδοση του hardware, πρέπει να είναι συμβατή με το λογισμικό, χωρίς περαιτέρω αλλαγές σε αυτό.

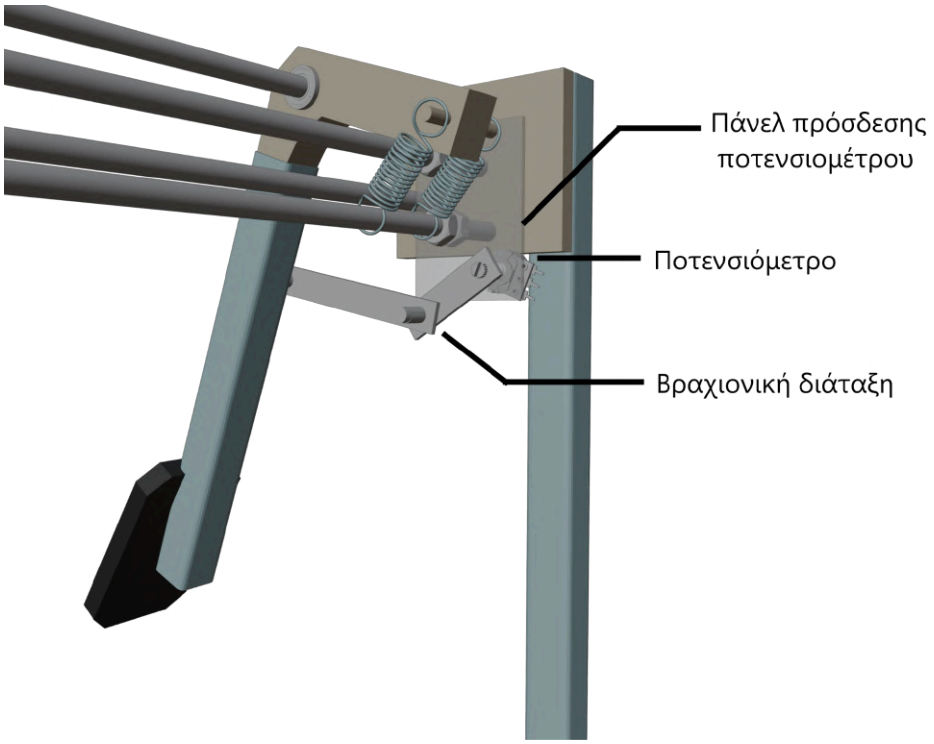
- Η δύναμη που δέχεται το πόδι του οδηγού από το ψηφιακό πεντάλ πρέπει να αναπαράγει με επαρκή ακρίβεια την αίσθηση που προσφέρει ένα πραγματικό όχημα. Η σωστή αποτύπωση αυτής της αντίστασης είναι κρίσιμη, καθώς συνιστά το βασικό σημείο αλληλεπίδρασης μεταξύ του οδηγού και του συστήματος πέδησης ή επιτάχυνσης του προσομοιωτή.
- Η σωστή τοποθέτηση μοχλών, έτσι ώστε η κίνηση του οδηγού να μετατρέπεται γραμμικά σε περιστροφή των ποτενσιομέτρων.
- Η σωστή τοποθέτηση των σημείων πρόσδεσης των ποτενσιομέτρων, έτσι ώστε να μπορούν να ανιχνεύσουν αποτελεσματικά την κίνηση.

Ο σχεδιασμός του υποσυστήματος των πεντάλ πραγματοποιήθηκε στο λογισμικό Blender, με αφετηρία τη διαμόρφωση του σώματος του πεντάλ. Η βασική απαίτηση σε αυτό το στάδιο είναι η δημιουργία μιας γεωμετρίας ικανής να συνεργάζεται αποτελεσματικά με τα ελατήρια που χρησιμοποιούνται για την προσομοίωση της αντίστασης. Δεδομένου ότι η αίσθηση που λαμβάνει ο οδηγός εξαρτάται σε μεγάλο βαθμό από την ορθή μεταφορά της δύναμης μέσω των ελατηρίων, το σώμα του πεντάλ είναι σχεδιασμένο να συνεργάζεται με αυτά. Μετρήσεις σχετικά με τις διαστάσεις των πεντάλ καθώς και οι αποστάσεις μεταξύ τους ελήφθησαν απο πραγματικό όχημα

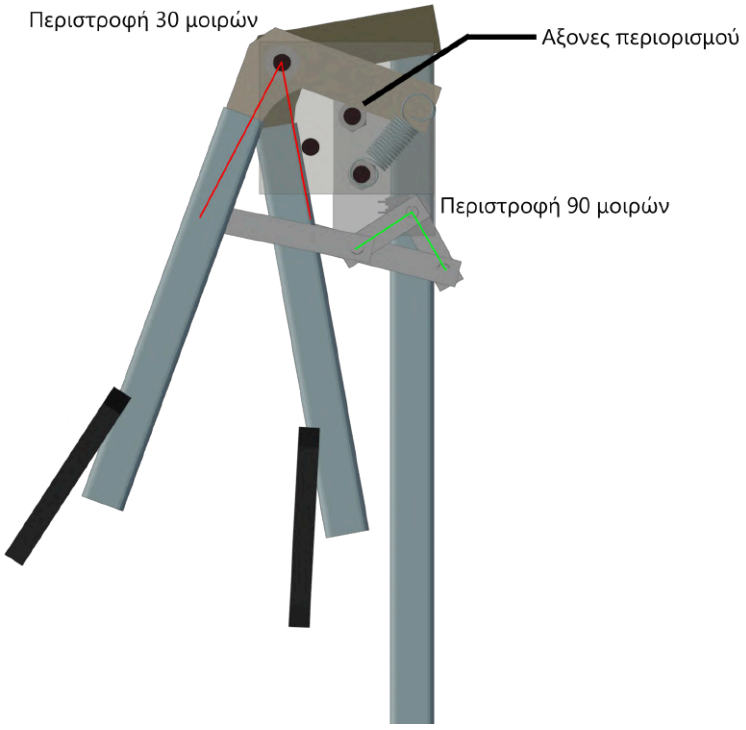
Μετά τον σχεδιασμό του σώματος του πεντάλ, το επόμενο στάδιο αφορά τη διαμόρφωση του πλαισίου (chassis), το οποίο λειτουργεί ως το κεντρικό στοιχείο στήριξης του υποσυστήματος. Ο πρωταρχικός στόχος του πλαισίου είναι να φιλοξενεί με ευστάθεια τα πεντάλ, εξασφαλίζοντας παράλληλα τα απαραίτητα σημεία στήριξης για τα ελατήρια που χρησιμοποιούνται στην παραγωγή της αντίστασης. Επιπλέον, το πλαίσιο πρέπει να διαθέτει κατάλληλες θέσεις τοποθέτησης για τα ποτενσιόμετρα, τα οποία είναι υπεύθυνα για την μετατροπή της μηχανικής ενέργειας σε ηλεκτρικό σήμα. Μία εξίσου σημαντική απαίτηση του σχεδιασμού είναι ο περιορισμός της γωνιακής περιστροφής κάθε πεντάλ σε ρεαλιστικά επίπεδα.

Για την υποστήριξη των λειτουργιών που περιγράφηκαν προηγουμένως, χρησιμοποιούνται τέσσερις στρατηγικά τοποθετημένοι άξονες στο πλαίσιο. Ο πρώτος άξονας, τοποθετείται σε συγκεκριμένο σημείο του πλαισίου και αποτελεί τον κύριο άξονα περιστροφής της συναρμογής του πεντάλ, επιτρέποντας την ομαλή κίνηση του γύρω από σταθερό σημείο αναφοράς. Ο δεύτερος και τρίτος άξονας τοποθετούνται με τέτοιον τρόπο ώστε να λειτουργεί ως μηχανισμός περιορισμού της περιστροφής, καθορίζοντας τα όρια τόσο στη θετική όσο και στην αρνητική κατεύθυνση της κίνησης του πεντάλ. Ο τέταρτος άξονας χρησιμεύει ως σημείο στήριξης των ελατηρίων που είναι υπεύθυνα για τη δημιουργία της απαιτούμενης αντίστασης. Δύο απο τους άξονες έχουν διπλή λειτουργία, πέραν απο την βασική τους χρήση παρέχουν στήριξη στα σημεία τοποθέτησης των ποτενσιόμετρων, ώστε να καταγράφεται αξιόπιστα η γωνιακή μετατόπιση.

Ένα από τα πιο κρίσιμα στάδια της ανάπτυξης του υποσυστήματος των πεντάλ αφορά την προσθήκη των σημείων στήριξης για τα ποτενσιόμετρα καθώς και των μηχανικών μοχλών που μετατρέπουν τη κίνηση του πεντάλ σε γωνιακή περιστροφή του άξονα του ποτενσιόμετρου. Η μετάδοση της κίνησης από το πεντάλ στον δρομέα του ποτενσιόμετρου πραγματοποιείται μέσω δύο μοχλών διαφορετικού μήκους. Αυτή η διπλή μοχλο-βραχιονική διάταξη μεταφράζει την περιστροφή του πεντάλ των 30 μοιρών σε περιστροφή ποτενσιομέτρου των 90 μοιρών.

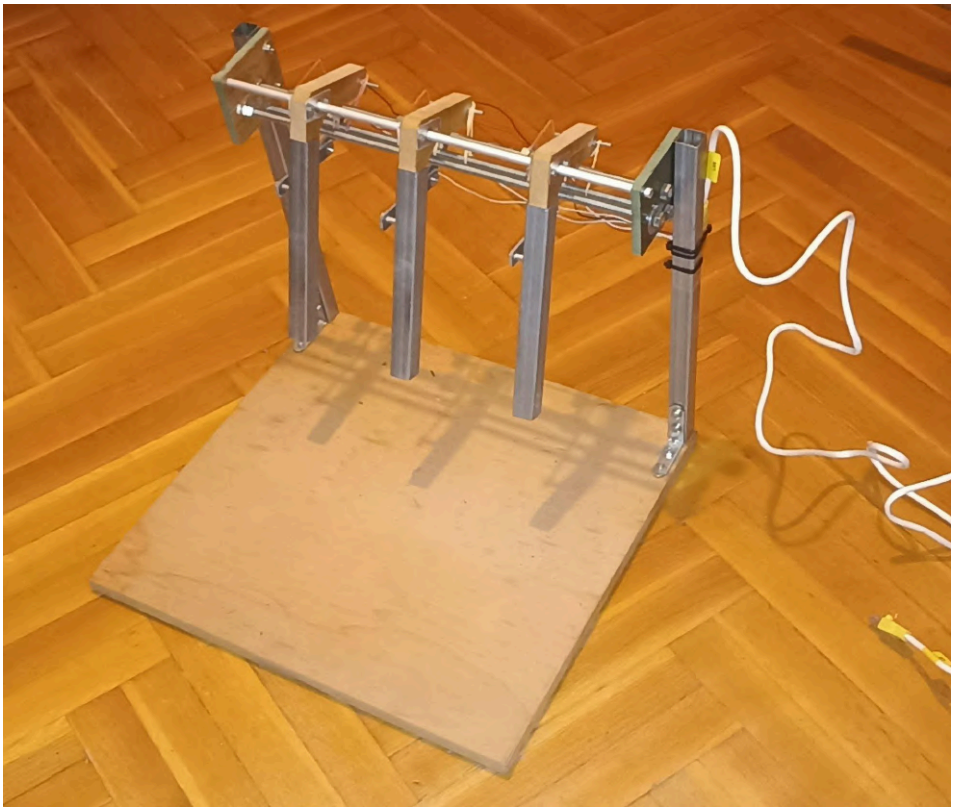


Σχήμα 2.3.2.1: Μεταφορά κίνησης στο ποτενσιόμετρο



Σχήμα 2.3.2.2: Διάταξη αξόνων και γωνίες περιστροφής

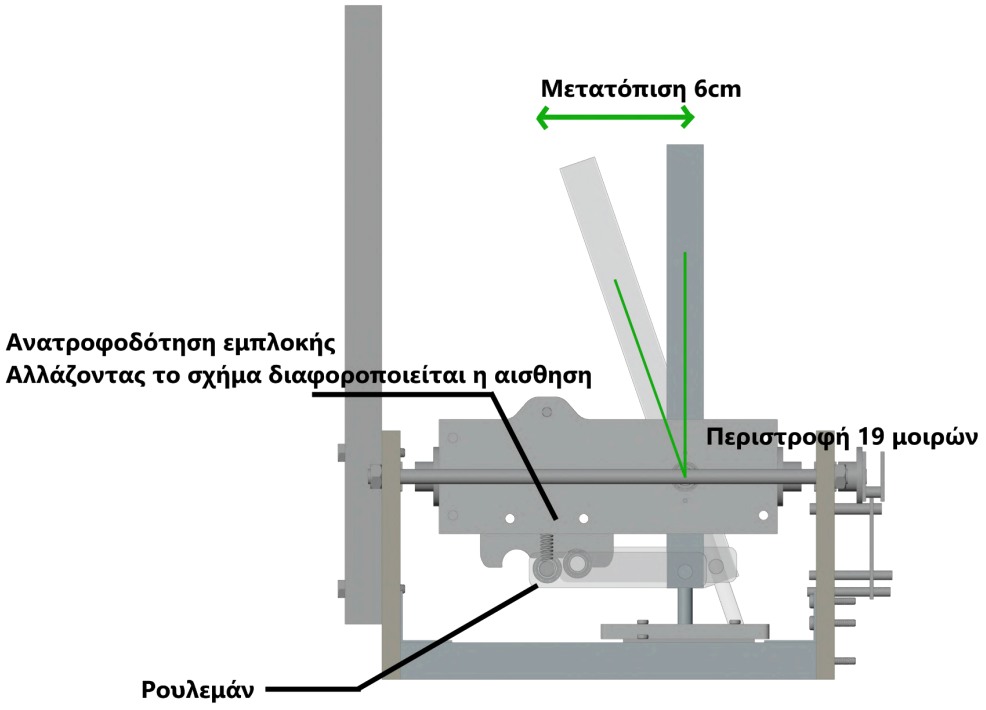
Σχήμα 2.3.2.3: Τελικός σχεδιασμός υποσυστήματος πεντάλ



Σχήμα 2.3.2.4: Φυσική υλοποίηση του υποσυστήματος

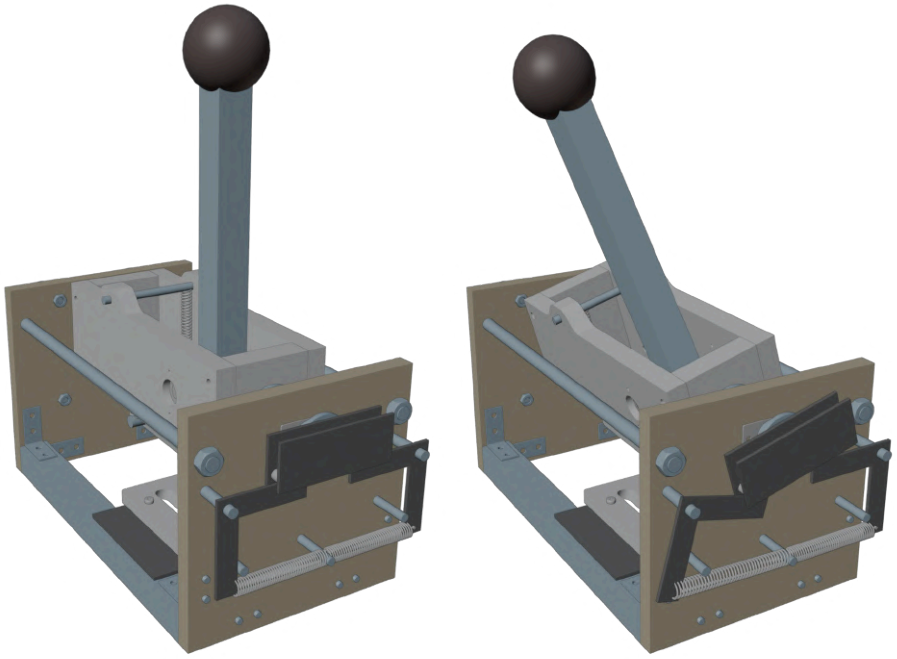
2.3.3 Υποσύστημα μοχλού ταχυτήτων

Κατά το σχεδιασμό ενός χειροκίνητου μοχλού ταχυτήτων για έναν προσομοιωτή , είναι σημαντικό να παρέχεται στον τελικό χρήστη μια ρεαλιστική αίσθηση, ενώ ταυτόχρονα να διασφαλίζεται άμεσος συγχρονισμός με το λογισμικό. Ο χειροκίνητος μοχλός έχει δύο βαθμούς ελευθερίας, καθώς περιστρέφεται τόσο οριζόντια όσο και κατακόρυφα. Στην τρέχουσα υλοποίηση, αυτοί οι βαθμοί ελευθερίας υλοποιούνται ως εξής: ο μοχλός περιστρέφεται κατακόρυφα πάνω σε μια κεντρική συναρμογή, η οποία με τη σειρά της περιστρέφεται οριζόντια σε σχέση με τη βάση του συστήματος. Κατά το σχεδιασμό ενός χειροκίνητου μοχλού, ο μηχανισμός σταθεροποίησης της επιλογής ταχυτήτων (detent mechanism) είναι κρίσιμης σημασίας, καθώς παρέχει στον χρήστη ανατροφοδότηση όταν η ταχύτητα έχει επιλεγεί σωστά. Αυτός ο μηχανισμός υλοποιείται μέσω ρουλεμάν τα οποία κινούνται πάνω σε μια τροχιά με εγκοπές μανδάλωσης (cam) μέσω ενός μοχλού ο οποίος είναι μηχανικά συνδεδεμένος με τον επιλογέα ταχυτήτων. Τα ρουλεμάν πιέζονται πάνω στην τροχία μέσω δύο ελατηρίων, τα οποία δημιουργούν την αντίσταση που γίνεται αισθητή κατά την επιτυχή επιλογή ταχύτητας. Αυτός ο μηχανισμός είναι συνδεδεμένος μόνο με την κεντρική συναρμογή.



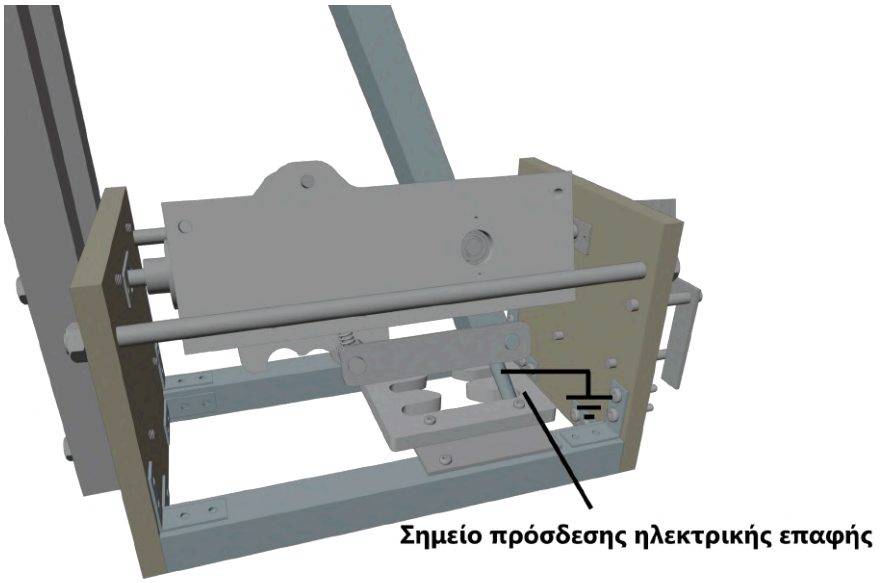
Σχήμα 2.3.3.1: Ανατροφοδότηση εμπλοκής

Το επόμενο σημαντικό βήμα είναι η παροχή μιας ρεαλιστικής αίσθησης επιστροφής στο κέντρο όταν ο μοχλός βρίσκεται στη θέση νεκρά. Στην τρέχουσα υλοποίηση, αυτό επιτυγχάνεται μέσω της σύνδεσης της κεντρικής συναρμογής, η οποία περιστρέφεται οριζόντια, με ένα σύνολο μοχλών που ενεργοποιούν δευτερεύοντες μοχλούς. Οι δευτερεύοντες μοχλοί είναι συνδεδεμένοι με ελατήρια, τα οποία παρέχουν την επιστροφή του μοχλού στη θέση κέντρου.

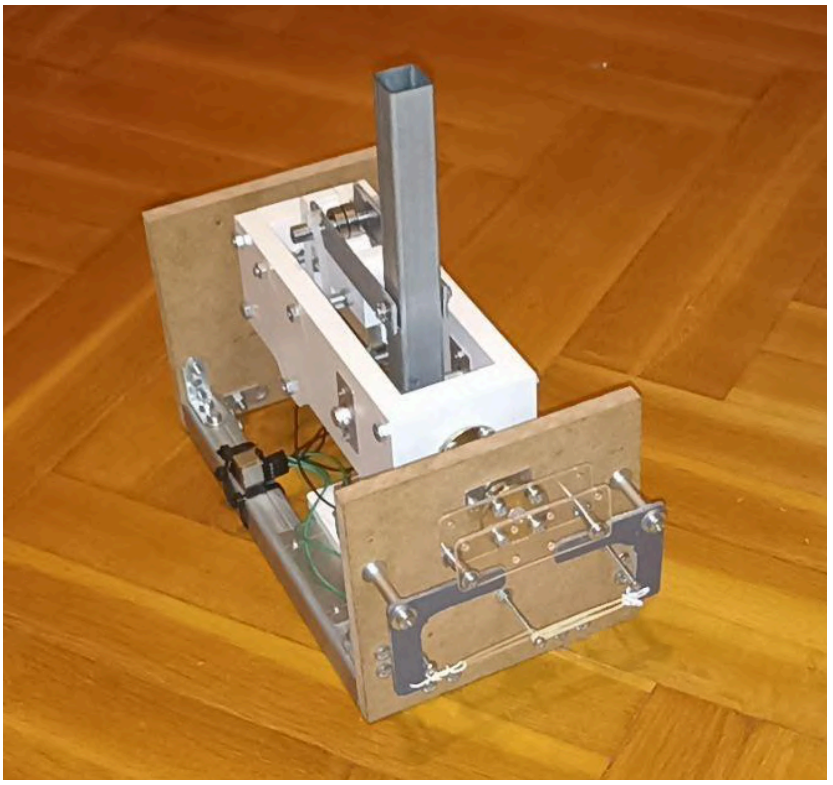


Σχήμα 2.3.3.2: Επιστροφή στο κέντρο σε θέση νεκράς

Τέλος, είναι σημαντικό να διασφαλιστεί ότι, όταν ο μοχλός βρίσκεται σε ταχύτητα, δεν υπάρχει οριζόντια μετακίνηση, διατηρώντας τον σταθερό στη θέση του. Αυτό επιτυγχάνεται μέσω μιας πλάκας τοποθετημένης ακριβώς κάτω από το κέντρο περιστροφής του μοχλού, η οποία έχει εγκοπές ώστε να ακολουθεί τη διαδρομή του μοχλού κατά την αλλαγή ταχυτήτων. Η πλάκα αυτή έχει και δευτερεύοντα ρόλο, καθώς φιλοξενεί τα σημεία επαφής για τα καλώδια που ελέγχουν το σήμα γείωσης, το οποίο προέρχεται από ένα σημείο επαφής απέναντι απο το άκρο της λαβής του μοχλού.



Σχήμα 2.3.3.3: Περιορισμός οριζόντιας κίνησης κατα την επιλογή ταχύτητας και ηλεκτρικές επαφές



Εικόνα 2.3.3.4: Τελική υλοποίηση μοχλού ταχυτήτων

Κεφάλαιο 3. Συμπεράσματα και προτάσεις για βελτίωση

Στο παρόν κεφάλαιο συνοψίζονται τα βασικά ευρήματα που αποκομίστηκαν κατα την υλοποίηση του έργου και αποτιμάται ο βαθμός επίτευξης των αρχικών στόχων. Μέσα από την ανάλυση των αποτελεσμάτων που προέκυψαν από τη σχεδίαση, υλοποίηση και δοκιμή του συστήματος, αξιολογείται η λειτουργικότητα και η αξιοπιστία, καθώς και η αξία του συστήματος στη διαδικασία εκπαίδευσης των οδηγών. Παράλληλα, εντοπίζονται οι περιορισμοί της παρούσας υλοποίησης και αναλύονται οι δυνατότητες μελλοντικής εξέλιξης, σε επίπεδο λογισμικού και εξοπλισμού. Επίσης ακολουθεί μια τεχνική ανακεφαλαίωση των μεθόδων σχεδίασης και υλοποίησης του συστήματος, μαζί με αναφορές σε εναλλακτικές τεχνικές που θα μπορούσαν να χρησιμοποιηθούν.

3.1 Στόχοι και απαιτήσεις του έργου

Στόχος του παρόν έργου είναι ο σχεδιασμός και η υλοποίηση ενός συστήματος προσομοίωσης οχημάτων, το οποίο μπορεί να αξιοποιηθεί στην εκπαίδευση οδηγών, παρέχοντας ένα αποτελεσματικό, ασφαλές και οικονομικό περιβάλλον εκπαίδευσης το οποίο μπορεί να χρησιμοποιηθεί συμπληρωματικά στην παραδοσιακή πρακτική εκπαίδευση. Η εκπαίδευση οδηγών αποτελεί κρίσιμο παράγοντα για τη βελτίωση της οδικής ασφάλειας και τη μείωση του αριθμού των τροχαίων ατυχημάτων παγκοσμίως, καθώς επιτρέπει την ανάπτυξη σωστών οδηγικών συμπεριφορών και την εξοικείωση των εκπαιδευομένων με σύνθετες και επικίνδυνες καταστάσεις χωρίς πραγματικό κίνδυνο.

Ο σχεδιασμός ενός συστήματος προσομοίωσης οχημάτων που συνδυάζει φυσικούς χειρισμούς με λογισμικό προσομοίωσης απαιτεί ιδιαίτερες βελτιστοποιήσεις, σε βαθμό πιστότητας του υλικού εξοπλισμού και ως προς την ανάπτυξη ενός λογισμικού που αναπαριστά με μέγιστη δυνατή ακρίβεια την συμπεριφορά ενός οχήματος και τις πραγματικών οδικές συνθήκες σε διάφορα περιβάλλοντα. Η αλληλεπίδραση μεταξύ χρήστη και συστήματος οφείλει να είναι άμεση, ώστε η εικονική εμπειρία να προσεγγίζει όσο το δυνατόν περισσότερο την πραγματική. Η αποτελεσματικότητα ενός τέτοιου συστήματος καθορίζεται από μια σειρά κρίσιμων παραγόντων, οι οποίοι επηρεάζουν άμεσα την αξία του συστήματος στην χρήση του:

- Διαισθητικά φυσικά χειριστήρια τα οποία μιμούνται με ακρίβεια ένα πραγματικό όχημα και παρέχουν την κατάλληλη ανατροφοδότηση στον οδηγό. Αρχικά απαιτείται η υλοποίηση ενός τιμονιού το οποίο φέρει πραγματικές διαστάσεις και περιστρέφεται ομαλά γύρω από τον άξονα του με μέτρο ίσο με αυτού ενός τυπικού οχήματος παραγωγής, δηλαδη 540 μοίρες προς κάθε κατεύθυνση. Για την υλοποίηση αποτελεσματικών πεντάλ αρχικά απαιτείται η λήψη των σωστών διαστάσεων καθώς και της σωστής απόστασης μεταξύ τους. Επίσης σημαντική είναι η δύναμη που ασκείται από το πεντάλ σε συνάρτηση με την θέση του. Όσον αφορά τον μοχλό ταχυτήτων ο πιο σημαντικός παράγοντας είναι η σωστή ανατροφοδότηση της δύναμης που ασκεί ο μοχλός στο χέρι του χειριστή αναλόγως της θέσης που βρίσκεται, δηλαδή είναι απαραίτητο να γίνεται αισθητή η επιτυχής επιλογή σχέσης, καθώς και η ελεύθερη κίνηση του μοχλού στον οριζόντιο άξονα στην κατάσταση νεκράς.
- Ο ακριβής και άμεσος συγχρονισμός μεταξύ υλικού και λογισμικού, είναι επίσης απαραίτητος, ωστε οι ενέργειες του χρήστη να έχουν άμεση επιρροή με το οπτικό και ακουστικό ερέθισμα που προβάλλεται από το λογισμικό. Η κάθε κίνηση ενός χειριστηρίου απαιτεί την κίνηση του αντίστοιχου αντικειμένου στο εικονικό όχημα άμεσα και ίση σε μέτρο με τον φυσικό χειρισμό

- Η πιστότητα του προσομοιωμένου οχήματος είναι κρίσιμης σημασίας για την προσομοίωση. Το λογισμικό καλείται να προσομοιώσει την φυσική συμπεριφορά του οχήματος ανάλογα με τις συνθήκες στις οποίες βρίσκεται. Η σωστή προσαρμογή του κέντρου βάρους και της κατανομής βάρους στους άξονες κρίνεται απαραίτητη. Επίσης σημαντική είναι η σωστή συμπεριφορά των αναρτήσεων ώστε να προσομοιώνεται σωστά η μεταφορά βάρους σε συνθήκες επιτάχυνσης, πέδησης, και αλλαγής κατεύθυνσης.
- Η Ρεαλιστική αναπαράσταση των κυκλοφοριακών συνθηκών είναι ένα κομβικό τμήμα στην ανάπτυξη του συστήματος, καθώς προετοιμάζει τον χρήστη σε σενάρια τα οποία θα συναντήσει στην πραγματικότητα. Οι κινήσεις των αυτόνομων οχημάτων που χρησιμοποιούνται στο λογισμικό είναι απαραίτητο να μιμούνται την συμπεριφορά ενός τυπικού οδηγού, με παραμέτρους που παρέχουν μια ποικιλία απο οδηγικούς χαρακτήρες. Επίσης σημαντικός είναι ο έλεγχος της ποσότητας των κινούμενων οχημάτων καθώς και των σταθμευμένων.
- Ενσωματώνοντας τους κανόνες οδικής κυκλοφορίας στο λογισμικό παρέχεται στον χρήστη η εξοικείωση με την θεωρία της σωστής οδηγικής συμπεριφοράς, με την χρήση σήμανσης και υποδείξεων καθώς και ανατροφοδότηση σε περίπτωση που καταγράφει παράβαση
- Η Δυνατότητα δημιουργίας πολλαπλών σεναρίων προσομοίωσης δίνει στο λογισμικό την δυνατότητα να παρέχει στοχευμένα σενάρια στον οδηγό τα οποία αποσκοπούν στην βελτίωση συγκεκριμένων δεξιοτήτων με μεταβλητά επίπεδα δυσκολίας.
- Οι μηχανισμοί αξιολόγησης και ανατροφοδότησης κρίνονται απαραίτητο χαρακτηριστικό του λογισμικού, καθώς επιτρέπουν την αξιολόγηση της απόδοσης του οδηγού και την ένδειξη αδυναμιών και δεξιοτήτων που απαιτούν βελτίωση.
- Η αξιοπιστία του εξοπλισμού και του λογισμικού είναι επίσης σημαντική, ώστε το σύστημα να λειτουργεί απρόσκοπτα σε συνθήκες τυπικής χρήσης. Η συστηματική εφαρμογή ορθών προγραμματιστικών προτύπων μειώνει δραστικά τον αριθμό των σφαλμάτων στο λογισμικό. Η ανάπτυξη προσαρμοσμένων εργαλείων αποσφαλματοποίησης επίσης διευκολύνει την εξάλειψη αστοχιών κατα την ανάπτυξη λογισμικού. Οσον αφορα τον εξοπλισμό είναι απαραίτητη η χρήση ανθεκτικών υλικών και αξιόπιστων μεθόδων παραγωγής καθως και ο ακριβής υπολογισμός διαστάσεων.
- Το κόστος υλοποίησης και επέκτασης πρέπει να παραμένει χαμηλό ώστε το σύστημα να είναι πρακτικά αξιοποιήσιμο από εκπαιδευτικούς φορείς και διαθέσιμο σε ενα ευρύ φάσμα κοινού από δημόσιους και ιδιωτικούς φορείς καθώς και ιδιότητες.
- Η απόδοση του λογισμικού αποτελεί από τα σημαντικότερα κριτήρια του συστήματος. Η βελτιστοποίηση απόδοσης της βάσης κώδικα και των αλγορίθμων που χρησιμοποιούνται μειώνει δραματικά τις απαιτήσεις του συστήματος που πρόκειται να εκτελέσει το λογισμικό προσομοίωσης, πράγμα που κάνει το λογισμικό πιο προσβάσιμο στο κοινό και παράλληλα μειώνει το κόστος χρήσης του.
- Η δυνατότητα κλιμάκωσης και αναβάθμισης, τόσο σε επίπεδο υλικού όσο και λογισμικού, είναι σημαντική για την μελλοντική επέκταση των δυνατοτήτων του συστήματος. Η επιλογή αρχιτεκτονικής σχεδίασης η οποία καθιστά ομαλή την προσθήκη περιεχομένου συμβάλλει στην επεκτασιμότητα του λογισμικού. Η επεκτασιμότητα του εξοπλισμού είναι επίσης σημαντική, οπως και η χρήση αρθρωτής (modular) κατασκευής, ώστε να μπορούν να προστεθούν με ευκολία νέα μηχανικά τμήματα για να επεκτείνουν τις δυνατότητες του συστήματος.

- Η ευκολία και ταχύτητα εγκατάστασης του εξοπλισμού είναι αναγκαία, ώστε να μπορεί να χρησιμοποιηθεί σε διάφορες συνθήκες οπως σχολές οδηγών, εκπαιδευτικά ιδρύματα και ιδιωτικές κατοικίες.
- Η ευκολία δημιουργίας και ενσωμάτωσης εκπαιδευτικού περιεχομένου κατά το στάδιο της ανάπτυξης του λογισμικού είναι ένα αναγκαίο χαρακτηριστικό. Η ενσωμάτωση περιεχομένου σε μορφή animation παρέχει στον χρήστη της κατάλληλες θεωρητικές γνώσεις πριν την απόπειρα πρακτικής εξάσκησης στο πλαίσιο του συστήματος.

3.2 Προτεινόμενο σύστημα, μέθοδοι σχεδίασης και υλοποίησης

Το προτεινόμενο σύστημα αποτελείται από μηχανικό εξοπλισμό που χωρίζεται σε τρία τμήματα και λογισμικό για πλατφόρμες windows και linux. Το λογισμικό δημιουργήθηκε χρησιμοποιώντας την unity3D ως μηχανή γραφικών και την C# ως κύρια γλώσσα προγραμματισμού. Το λογισμικό μοντελοποίησης που επιλέχθηκε είναι το Blender, και το λογισμικο που χρησιμοποιήθηκε για την απόδοση εικόνας στα τρισδιάστατα μοντέλα είναι το Adobe Substance Painter.

Το προσομοιωμένο όχημα έχει σχεδιαστεί στο blender βάση διαστάσεων ενος πραγματικού οχήματος, όσον αφορά χαρακτηριστικά εμφάνισης και μηχανικών μερών που επηρεάζουν την φυσική του συμπεριφορά οπως το πλάτος των αξόνων, την απόσταση μεταξύ τον αξόνων και γεωμετρικά στοιχεία της ανάρτησης. Για την προσομοίωση της φυσικής συμπεριφοράς του οχήματος χρησιμοποιήθηκε το ενσωματωμένο physics engine του περιβάλλοντος unity το οποίο βασίζεται στο physX API της nvidia. Χρησιμοποιώντας εγγενείς κλάσεις της unity ορίστηκαν φυσικά χαρακτηριστικα του οχήματος οπως διαστάσεις τροχών, διάταξη αξόνων, βάρος, κέντρο βάρους, κατανομή βάρους στους άξονες, χαρακτηριστικά της ανάρτησης και ο συντελεστής αεροδυναμικής αντίστασης. Δημιουργήθηκε μια κλάση για τον έλεγχο του οχήματος η οποία λαμβάνει βαθμονομημένες εισόδους απο τον εξοπλισμό και ελέγχει την κίνηση του οχήματος, αλλάζοντας προσανατολισμό στους εμπρόσθιους τροχούς, εφαρμόζοντας την κατάλληλη δύναμη στους κινητήριους τροχούς και εφαρμόζοντας δύναμη πέδησης. Επίσης περιλαμβάνεται η προσομοίωση των στροφών του κινητήρα ανάλογα με την επιλεγμένη σχέση. Το λογισμικό είναι δομημένο με τρόπο που παρέχει στοχευμένα σενάρια εκμάθησης συγκεκριμένων οδηγικών δεξιοτήτων και πλοήγηση σε περιβάλλον πόλης το οποίο παράγεται με τυχαιότητα μέσω ενος συνόλου απο αλγορίθμων. Τα στατικά σενάρια δημιουργούνται αρχικά σε λογισμικό σχεδίασης και έπειτα συναρμολογούνται τα αρθρωτά τους τμήματα εντός του περιβάλλοντος unity, και περιλαμβάνουν διάφορες παραλλαγές και επίπεδα δυσκολίας για κάθε οδηγική δεξιότητα. Οι μηχανισμοί αξιολόγησης υλοποιούνται σε διαφορετικές κλάσεις για κάθε είδος σεναρίου. Δημιουργήθηκε ένα εργαλείο καταγραφής κίνησης, το οποίο βοηθάει στην δημιουργία εκπαιδευτικού περιεχομένου σε μορφή animation, καταγράφοντας τις μετατοπίσεις και περιστροφές όλων των σχετικων αντικειμένων σε μια σκηνή και αποθηκεύοντας τις σε ένα αρχείο για μελλοντική αναπαραγωγή. Το περιβάλλον ελεύθερης πλοήγησης δημιουργείται μέσω αλγορίθμων πάνω σε ένα πλέγμα από κελιά που περιλαμβάνουν οδικές υποδομές. Δημιουργείται ένα οδικό περιβάλλον με κίνηση που αποτελείται απο αυτόνομους οδηγούς οι οποίοι ακολουθούν καμπύλες διαδρομές, και παίρνουν αποφάσεις βάση προτεραιοτήτων. Το περιβάλλον παράγεται μέσα από διάφορα στάδια χρησιμοποιώντας αλγόριθμους αναδρομικής υποδιαίρεσης του πλέγματος για την δημιουργία οικοδομικών τετραγώνων, και ενσωμάτωσης του αλγόριθμου wave function collapse για την προσθήκη ποικιλίας στο περιβάλλον.

Ο σχεδιασμός του μηχανικού εξοπλισμού πραγματοποιήθηκε στο Blender, αξιοποιώντας εργαλεία όπως το Mesh Machine και το Precision Gear, τα οποία επέτρεψαν την ενσωμάτωση παραμετρικών χαρακτηριστικών τύπου CAD στο περιβάλλον του Blender. Με τη χρήση αυτών των εργαλείων

γίνεται δυνατή η ακριβής σχεδίαση λειτουργικών γεωμετριών, μηχανικών ενώσεων και εξαρτημάτων υψηλής ακρίβειας, συνδυάζοντας ελεύθερο 3D modeling με παραδοσιακού σχεδιασμό CAD. Όσον αφορά τα υλικά κατασκευής, χρησιμοποιήθηκαν τρισδιάστατα εκτυπωμένα εξαρτήματα, τα οποία παρήχθησαν για την επίτευξη πολύπλοκων γεωμετριών και προσαρμοσμένων μηχανικών λύσεων, ινοσανίδα χρησιμοποιήθηκε για δομικά και βοηθητικά μέρη και ενσωματώθηκαν μεταλλικά εξαρτήματα όπως άξονες, ρουλεμάν και σωλήνες, τα οποία εξασφαλίζουν μηχανική αντοχή, ακρίβεια κίνησης και αξιοπιστία στη λειτουργία του συστήματος. Σε διάφορα στάδια της κατασκευαστικής διαδικασίας χρησιμοποιήθηκαν επίσης φύλλα PVC, κυρίως για βοηθητικά μέρη. Ως αισθητήρες χρησιμοποιήθηκαν ποτενσιόμετρα για το τιμόνι και τα πεντάλ και ένας προσαρμοσμένος ψηφιακός διακόπτης έξι θέσεων για τον μοχλό ταχυτήτων. Οι τιμές τάσης που παράγουν οι αισθητήρες διαβάζονται απο έναν μικροελεγκτή arduino pro micro και αποστέλλονται τιμές εισόδου στον υπολογιστή μέσω USB, χρησιμοποιόντας ενα universal προτόκολλο επικινωνίας για άμεση εγκατάσταση.

3.3 Αποτελέσματα, περιορισμοί και πλεονεκτήματα

Η τελική υλοποίηση του συστήματος απέδωσε λειτουργικά αποτελέσματα, τα οποία ανταποκρίνονται σε έναν βαθμό στους αρχικούς στόχους του σχεδιασμού. Ωστόσο, η διαδικασία ανάπτυξης ανέδειξε και ορισμένους τεχνικούς και κατασκευαστικούς περιορισμούς, οι οποίοι πρέπει να ληφθούν υπόψη. παρακάτω γίνεται ανάλυση των αποτελεσμάτων, καθώς και καταγραφή των πλεονεκτημάτων και περιορισμών της προτεινόμενης λύσης.

Όσον αφορά την ρεαλιστικότητα του χειρισμού, έχει επιτευχθεί σε ικανοποιητικό βαθμό. Το τιμόνι, καθώς είναι προϊόν τρισδιάστατης εκτύπωσης έχει δημιουργηθεί με διαστάσεις ενός κανονικού οχήματος και περιστρέφεται ομαλά 540 μοίρες γύρω από τον άξονα του και ελέγχει το εικονικό όχημα με μεγάλη ακρίβεια, αλλά θα μπορούσε να βελτιωθεί με την ενσωμάτωση μηχανισμού ανάδρασης για να μπορεί να εφαρμόζει δύναμη στο τιμόνι. Ο μοχλός ταχυτήτων αποδίδει με ρεαλισμό τις δυνάμεις που γίνονται αντιληπτές κατα την επιλογή σχέσης, και αποδίδει σωστά την αίσθηση της νεκράς. τα πεντάλ, παρόλο που έχουν σωστή γεωμετρία, θα μπορούσαν να εξελιχθούν σημαντικά. Το πεντάλ του γκαζιού αποδίδει αληθοφανή αίσθηση ενώ το πεντάλ του φρένου θα μπορούσε να επωφεληθεί από ένα μηχανισμό ανάδρασης ώστε να μπορεί να αποδώσει διαφορετική αίσθηση ανα σενάριο φρεναρίσματος. Το πεντάλ του συμπλέκτη επίσης θα μπορούσε να χρησιμοποιεί έναν πιο εξελιγμένο μοχλο-βραχιωνικό μηχανισμό για βέλτιστη απόδοση της δύναμης ανάλογα με το σημείο που βρίσκεται. Η ακρίβεια των μηχανικών ελέγχων ως συσκευές εισόδου παρατηρήθηκε ικανοποιητική, και ο συγχρονισμός με τα εικονικά αντικείμενα επίσης λειτουργικός.

Η πιστότητα της φυσικής συμπεριφοράς του οχήματος παρατηρήθηκε ικανοποιητική, και η μεταφορά βάρους σε αλλαγές κατεύθυνσης λειτουργική. Θα μπορουσε να εξελιχθεί η πιστότητα αλλάζοντας τον συντελεστή τριβής ανάλογα με την επιφάνεια πάνω στην οποία πατάει ο τροχός. Οι αλλαγές ταχυτήτων είναι επίσης ικανοποιητικές με τις στροφές του κινητήρα να αλλάζουν σύμφωνα με το αναμενόμενο.

Η επεκτασιμότητα του λογισμικού επιτρέπει την γρήγορη ενσωμάτωση νέων σεναρίων λόγω της αρθρωτής φύσης των περιβάλλοντων και της ευελιξίας του προγραμματιστικού υπόβαθρου. Η δυνατότητα δημιουργίας σεναρίων προσομοίωσης υλοποιήθηκε επιτυχώς, επιτρέποντας την εστίαση σε συγκεκριμένες δεξιότητες και την προσαρμογή του επιπέδου δυσκολίας.

Οι μηχανισμοί αξιολόγησης της απόδοσης υλοποιήθηκαν με την καταγραφή βασικών δεικτών, επιτρέποντας την αναγνώριση αδυναμιών. Η συνεχής ανατροφοδότηση υποστηρίζει τη μαθησιακή

διαδικασία. Περιθώρια βελτίωσης υπάρχουν στην πιο αναλυτική παρουσίαση των αποτελεσμάτων, όπως στατιστικά στοιχεία προόδου, ιστορικό επιδόσεων και συγκριτική αξιολόγηση μεταξύ συνεδριών.

Η ρεαλιστική αναπαράσταση των κυκλοφοριακών συνθηκών υλοποιήθηκε σε έναν βασικό βαθμό μέσω της παρουσίας αυτόνομων οχημάτων επιτρέποντας στον χρήστη να εξοικειωθεί με καταστάσεις που προσομοιάζουν την πραγματική οδήγηση. Η δυνατότητα ελέγχου της πυκνότητας των κινούμενων και σταθμευμένων οχημάτων προσαρμόζει το επίπεδο δυσκολίας. Ωστόσο, περιθώρια βελτίωσης εντοπίζονται στη λεπτομερέστερη μοντελοποίηση της ανθρώπινης οδηγικής συμπεριφοράς, όπως απρόβλεπτες αντιδράσεις, καθυστερημένες αποφάσεις ή παραβατικές ενέργειες από τα αυτόνομα οχήματα. Η ενσωμάτωση πιο σύνθετων μοντέλων τεχνητής νοημοσύνης θα μπορούσε να αυξήσει περαιτέρω τον βαθμό ρεαλισμού. Εντός του περιβάλλοντος ελεύθερης πλοήγησης, ο κώδικας οδικής κυκλοφορίας όσον αφορά την σήμανση υλοποιήθηκε σε έναν πολύ βασικό βαθμό, καθώς παρουσιάστηκαν προκλήσεις λόγω της απουσίας ανθρώπινης παρέμβασης κατα την δημιουργία του περιβάλλοντος, διοτι παράγεται από μια σειρά αλγορίθμων οι οποίοι θα πρέπει να επεκταθούν για να δημιουργήσουν ένα ρεαλιστικό οδικό πλάνο με κανόνες κυκλοφορίας οι οποίοι βασίζονται πάνω στον Κ.Ο.Κ. για κάθε παραγόμενο περιβάλλον. Η βελτίωση της ανατροφοδότησης με πιο αναλυτικές επεξηγήσεις ή σύνδεση των παραβάσεων με συγκεκριμένα άρθρα του Κώδικα Οδικής Κυκλοφορίας θα ενίσχυε περαιτέρω την κατανόηση της θεωρίας από τον χρήστη.

Η ενσωμάτωση εκπαιδευτικού οπτικοακουστικού περιεχομένου σε μορφή animations, ενισχύει την κατανόηση των κανόνων πριν την πρακτική εξάσκηση. Το χαρακτηριστικό αυτό υλοποιήθηκε με επιτυχία στο πλαίσιο του συστήματος. Μελλοντικά, η δυνατότητα εύκολης προσθήκης νέου περιεχομένου από εκπαιδευτές χωρίς τεχνικές γνώσεις θα βελτίωνε σημαντικά τη χρηστικότητα του συστήματος.

Η απόδοση του λογισμικού βελτιστοποιήθηκε μέσω αποδοτικών αλγορίθμων και οργανωμένης βάσης κώδικα, μειώνοντας τις απαιτήσεις σε hardware και καθιστώντας το σύστημα ευρύτερα προσβάσιμο. Παρόλα αυτά, η περαιτέρω βελτιστοποίηση γραφικών και φυσικής προσομοίωσης θα μπορούσε να επιτρέψει την εκτέλεση του συστήματος σε ακόμη χαμηλότερων δυνατοτήτων υπολογιστικά συστήματα.

Η αξιοπιστία του συστήματος διασφαλίστηκε μέσω της χρήσης ανθεκτικών υλικών, ορθών μεθόδων κατασκευής και καθιερωμένων προγραμματιστικών προτύπων. Η ανάπτυξη προσαρμοσμένων εργαλείων αποσφαλματοποίησης συνέβαλε στη σταθερότητα του λογισμικού. Μελλοντικά, η εκτενέστερη δοκιμή σε πραγματικές συνθήκες χρήσης και η αυτοματοποίηση διαδικασιών ελέγχου θα μπορούσαν να μειώσουν περαιτέρω την πιθανότητα αστοχιών.

Η εγκατάσταση του εξοπλισμού υλοποιήθηκε με γνώμονα την ταχύτητα και την απλότητα, επιτρέποντας τη χρήση του συστήματος σε διαφορετικά περιβάλλοντα, όπως σχολές οδηγών και οικιακούς χώρους. Τα μηχανικά τμήματα στερεώνονται με αξιοπιστία σε οποιοδήποτε γραφείο η επίπεδη επιφάνεια, με την δυνατότητα να ενσωματωθεί μια προσαρμοσμένη βάση. Το λογισμικό διατίθεται σε portable μορφή για συστήματα windows και linux, γεγονός που αφαιρεί την ανάγκη εγκατάστασης. Περιθώρια βελτίωσης εντοπίζονται στην δημιουργία αναλυτικών οδηγιών εγκατάστασης, και χρήσης του λογισμικού.

Το σύστημα σχεδιάστηκε με γνώμονα τη διατήρηση χαμηλού κόστους, καθιστώντας το προσβάσιμο σε εκπαιδευτικούς φορείς και ιδιώτες. Η επιλογή οικονομικών αλλά αξιόπιστων εξαρτημάτων και υλικών συνέβαλε στη βιωσιμότητα του έργου. Ωστόσο, η περαιτέρω μείωση του κόστους μέσω μαζικής παραγωγής ή χρήσης ανοικτού λογισμικού αποτελεί μελλοντική δυνατότητα βελτίωσης.

3.4 Προοπτικές εξέλιξης του συστήματος

Το σύστημα που υλοποιήθηκε στο πλαίσιο της παρούσας εργασίας μπορεί να αποτελέσει τη βάση για περαιτέρω έρευνα και ανάπτυξη. Παρότι καλύπτει τις βασικές απαιτήσεις του αρχικού σχεδιασμού, υπάρχουν αρκετές κατευθύνσεις στις οποίες θα μπορούσε να εξελιχθεί. Επιπλέον, η διασύνδεση του συστήματος με άλλα υποσυστήματα αυτόνομης οδήγησης [17], [18], [19] (αντίληψης, εντοπισμού, λήψης αποφάσεων) θα επέτρεπε την αξιολόγησή του σε πραγματικές συνθήκες. Παρακάτω παρουσιάζονται προτάσεις μελλοντικής βελτίωσης και επέκτασης του συστήματος.

3.4.1 Προτάσεις βελτίωσης του λογισμικού

Όσον αφορά το λογισμικό, αυτό θα μπορούσε να βελτιωθεί με την προσθήκη περισσότερων ενοτήτων που θα εστιάζουν σε πολύ συγκεκριμένες δεξιότητες οδήγησης, όπως η διαδικασία προσπέρασης ή η γενικότερη πλοήγηση σε αυτοκινητόδρομους. Για τη δημιουργία ενός πιο ποικιλόμορφου περιβάλλοντος εξάσκησης μπορούν να χρησιμοποιηθούν δύο τεχνικές. Η πρώτη αφορά τη μεθοδολογία παραγωγής οδικού δικτύου. Αντί για χρήση πλέγματος κόμβων, θα μπορούσε να υλοποιηθεί μια δομή γράφου που αναπαριστά το οδικό δίκτυο ως σύνολο κόμβων και ακμών. Με αυτόν τον τρόπο θα επιτυγχανόταν μεγαλύτερη ποικιλία και ρεαλισμός, ωστόσο λόγω περιορισμένου χρόνου προτιμήθηκε η απλούστερη προσέγγιση με πλέγμα. Η δεύτερη τεχνική αφορά την αξιοποίηση δεδομένων χαρτών από το Google Maps, ώστε να δημιουργείται ένα περιβάλλον που θα αναπαριστά με μεγαλύτερη ακρίβεια αστικές περιοχές του πραγματικού κόσμου.

Μία ακόμη σημαντική δυνατότητα που θα ενίσχυε την αίσθηση εμβύθισης στον εικονικό κόσμο είναι η υποστήριξη τεχνολογίας εικονικής πραγματικότητας (VR). Η υπάρχουσα αρχιτεκτονική επιτρέπει σχετικά εύκολη ενσωμάτωση της λειτουργίας αυτής, αλλά η υλοποίησή της δεν πραγματοποιήθηκε λόγω περιορισμών χρόνου και προϋπολογισμού.

3.4.2 Προτάσεις βελτίωσης του εξοπλισμού

Το hardware του συστήματος μπορεί να βελτιωθεί με ποικίλους τρόπους, κυρίως μέσω της προσθήκης επιπλέον χειριστηρίων και μηχανισμών feedback. Ένα χαρακτηριστικό παράδειγμα αποτελεί η ενσωμάτωση φλάς, το οποίο θα αποστέλλει ψηφιακά σήματα στον μικροελεγκτή. Αν και η απαραίτητη λογική υποδομή για τη διαχείριση των φλάς υπάρχει ήδη στο λογισμικό. Ένας ακόμη ιδιαίτερα χρήσιμος μηχανισμός ανάδρασης που υποστηρίζει η υπάρχουσα αρχιτεκτονική είναι το κινούμενο κάθισμα του οδηγού. Η φυσική μετατόπιση και περιστροφή του προσομοιωτή με σκοπό την ρεαλιστική απόδοση των δυνάμεων που υπολογίζονται απο την προσομοίωση ονομάζεται motion cueing, και είναι μια ιδέα που χρησιμοποιείται εκτενώς στην βιομηχανία. Η κίνηση μπορεί να προσαρμόζει την κλίση σε πραγματικό χρόνο, ανταποκρινόμενο στα γεγονότα του προσομοιωτή, οπως επιταχύνσεις και επιβραδύνσεις ώστε να προσφέρει μια πιο ρεαλιστική αίσθηση κίνησης.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] World Health Organization (WHO), “Road traffic injuries: Fact sheet,” https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries. Accessed: Sept. 20, 2025. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries

[2] World Health Organization (WHO), “Global status report on road safety 2023,” https://www.who.int/teams/social-determinants-of-health/safety-and-mobility/global-status-report-on-road-safety-2023. Accessed: Sept. 20, 2025. [Online]. Available: https://www.who.int/teams/social-determinants-of-health/safety-and-mobility/global-status-report-on-road-safety-2023

[3] World Health Organization (WHO), “World report on road traffic injury prevention,” 2004. https://www.who.int/publications-detail-redirect/world-report-on-road-traffic-injury-prevention. Accessed: Sept. 20, 2025. [Online]. Available: https://www.who.int/publications-detail-redirect/world-report-on-road-traffic-injury-prevention

[4] World Health Organization (WHO), “SDG Target 3.6: Road traffic injuries,” https://www.who.int/data/gho/data/themes/topics/sdg-target-3_6-road-traffic-injuries. Accessed: Sept. 20, 2025. [Online]. Available: https://www.who.int/data/gho/data/themes/topics/sdg-target-3_6-road-traffic-injuries

[5] International Road Federation (IRF) / PIARC, “Impact on public health,” Road Safety Manual, https://roadsafety.piarc.org/en/strategic-global-perspective-scope-road-safety-problem/impact-public-health. Accessed: Sept. 20, 2025. [Online]. Available: https://roadsafety.piarc.org/en/strategic-global-perspective-scope-road-safety-problem/impact-public-health

[6] S. Mitra, S. Saadat, K. Neki, M. Khalaf, H. Rosen, N. Paichadze, and A. Hyder, *Beyond the Numbers: Estimating the Disability Burden of Road Traffic Injuries—Findings from Six Low and Middle-Income Countries*. Washington, DC: World Bank, 2023.

[7] International Transport Forum (ITF), Road Safety Annual Report 2023. Paris, France: OECD Publishing, 2023.

[8] VINSPY, “Car Accident Statistics in Europe by Country,” https://vinspy.eu/car-accident-statistics-in-europe-by-country/

[9] F. Alonso, M. Faus, J. V. Riera, M. Fernandez-Marin, and S. A. Useche, "Effectiveness of driving simulators for drivers’ training: A systematic review," Appl. Sci., vol. 13, p. 5266, 2023, doi: 10.3390/app13095266.

[10] S. Reindl, J. O. Thomas, and A. Wottge, "Einsatzmöglichkeiten von Fahrsimulatoren in der Ausbildung von Fahrschülern," Tech. Rep. no. 348, 2024.

[11] Unity Technologies, Unity Documentation, 2025. https://docs.unity3d.com. Accessed: Sept. 21, 2025. [Online]. Available: https://docs.unity3d.com.

[12] Microchip Technology, “ATmega32U4: 8-bit AVR Microcontrollers,” *Microchip Datasheet*, 2022. [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_datasheet.pdf. Accessed: Sept. 2, 2025.

[13] Arduino, “Pro Micro,” *SparkFun Documentation*, 2025. [Online]. Available: https://learn.sparkfun.com/tutorials/pro-micro--fio-v3-hookup-guide. Accessed: Sept. 2, 2025.

[14] Arduino, “Arduino Micro,” *Arduino Official Documentation*, 2025. [Online]. Available: https://docs.arduino.cc/hardware/micro. Accessed: Sept. 2, 2025.

[15] Arduino, “Joystick Library API,” *Arduino Playground*, 2025. [Online]. Available: https://github.com/MHeironimus/ArduinoJoystickLibrary. Accessed: Sept. 2, 2025.

[16] USB.org, “HID Usage Tables,” *USB Implementers Forum*, 2022. [Online]. Available: https://usb.org/sites/default/files/hut1_4.pdf. Accessed: Sept. 2, 2025.

[17] L. A. Iliadis et al., "Wideband Aperture-Coupled Array Design for Automotive Radar Applications," 2024 18th European Conference on Antennas and Propagation (EuCAP), Glasgow, United Kingdom, 2024, pp. 1-5, doi: 10.23919/EuCAP60739.2024.10501584.

[18] M. E. Mita, L. A. Iliadis, S. P. Sotiroudis, A. D. Boursianis, M. S. Papadopoulou and S. K. Goudos, "Traffic Sign Recognition for Level-3 Autonomy: Efficient Deep Learning for Class-Imbalanced Datasets," 2025 14th International Conference on Modern Circuits and Systems Technologies (MOCAST), Dresden, Germany, 2025, pp. 1-5, doi: 10.1109/MOCAST65744.2025.11083730.

[19] Wei, Zhe, Yurong Zou, Haibo Xu, and Sen Wang. 2025. "Small Object Detection in Traffic Scenes for Mobile Robots: Challenges, Strategies, and Future Directions" Electronics 14, no. 13: 2614. https://doi.org/10.3390/electronics14132614