ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

Σχολή Τεχνολογίας

Τμήμα Ψηφιακών Συστημάτων

Ανάπτυξη εφαρμογής για την καταγραφή και ανάλυση δεδομένων OBD σε συνθήκες πραγματικής κυκλοφορίας οχήματος

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΣΕΦΕΡΙΔΗΣ ΣΤΥΛΙΑΝΟΣ (ΑΜ: 3119198)

Επιβλέπων: Ονούφριος Α. Χαραλάμπους, Αναπληρωτής Καθηγητής
Πανεπιστημίου Θεσσαλίας

ΛΑΡΙΣΑ, Ιούλιος 2024

UNIVERSITY OF THESSALY

School of Technology

Digital Systems Department

Application development for the recording and analysis of OBD data in real vehicle traffic conditions

BACHELOR THESIS

SEFERIDIS STYLIANOS (RN: 3119198)

Supervisor: Onoufrios A. Haralampous, Associate Professor at University of Thessaly

LARISSA, July 2024

Υπεύθυνη Δήλωση περί Ακαδημαϊκής Δεοντολογίας και Πνευματικών Δικαιωμάτων

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ρητά ότι η παρούσα πτυχιακή εργασία, καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας, αποτελεί αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλει κάθε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχει έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον, και πληρούν τους κανόνες της επιστημονικής παράθεσης. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή/και πηγές άλλων συγγραφέων, αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή.

Αναλαμβάνω πλήρως, ατομικά και προσωπικά, όλες τις νομικές και διοικητικές συνέπειες που δύναται να προκύψουν στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής.

Ο/Η Δηλών/ούσα

(Υπογραφή)

ΣΕΦΕΡΙΔΗΣ ΣΤΥΛΙΑΝΟΣ

Ημερομηνία

15/07/2024

Εγκρίνεται από την Επιτροπή Εξέτασης:

Επιβλέπων/πουσα	**ΟΝΟΥΦΡΙΟΣ Α. ΧΑΡΑΛΑΜΠΟΥΣ** ΑΝΑΠΛΗΡΩΤΗΣ ΚΑΘΗΓΗΤΗΣ, Τμήμα Ψηφιακών Συστημάτων, Πανεπιστήμιο Θεσσαλίας
Μέλος	**ΞΕΝΑΚΗΣ ΑΠΌΣΤΟΛΟΣ** ΕΠΙΚΟΥΡΟΣ ΚΑΘΗΓΗΤΗΣ, Τμήμα Ψηφιακών Συστημάτων, Πανεπιστήμιο Θεσσαλίας
Μέλος	**ΚΩΝΣΤΑΝΤΙΝΟΣ ΧΑ'Ι'ΚΑΛΗΣ** ΑΝΑΠΛΗΡΩΤΗΣ ΚΑΘΗΓΗΤΗΣ, Τμήμα Ψηφιακών Συστημάτων, Πανεπιστήμιο Θεσσαλίας

Ημερομηνία έγκρισης: 17-07-2024

Περίληψη

Η συγκεκριμένη πτυχιακή εργασία επικεντρώνεται στην μελέτη της ενεργειακής κατανάλωσης και στην εκπομπή διοξειδίου του άνθρακα των οχημάτων, τα οποία αναλύονται με την βοήθεια ενός ανεπτυγμένου συστήματος αυτοδιάγνωσης. Η παγκόσμια κοινότητα αντικρίζει σημαντικά προβλήματα ατμοσφαιρικής ρύπανσης και αυξημένης ενεργειακής κατανάλωσης, τα οποία συμβάλλουν στην κλιματική αλλαγή και στην χαμηλή ποιότητα του περιβάλλοντος γενικότερα. Η πτυχιακή εργασία βασίζεται στον πράσινο και ψηφιακό μετασχηματισμό και πιο συγκεκριμένα στον τομέα των μεταφορών. Η δημιουργία του συστήματος αυτοδιάγνωσης πραγματοποιήθηκε μέσω ενός φορητού υπολογιστή που χρησιμοποιεί το λογισμικό Ubuntu, ενός καλωδίου OBD-II για την εισαγωγή των δεδομένων του οχήματος και την ανάπτυξη των επιμέρους προγραμμάτων που υλοποιήθηκαν σε γλώσσα προγραμματισμού Python, για την ανάλυση και την χαρτογράφηση των καταγραφών από το σημείο εκκίνησης μέχρι το σημείο ολοκλήρωσης της καταγραφής. Η επαλήθευση των δεδομένων καταγραφής γίνεται σε μορφή γραφημάτων και έπειτα εισάγονται σε πίνακα, με το Trip Computer του οχήματος και την εφαρμογή πλοήγησης GPS Logger που εγκαθίσταται σε κινητό τηλέφωνο. Επιπροσθέτως, αναφέρονται οι καταγραφές των διαδρομών που χωρίζονται σε Υπεραστικές και Αστικές, συγκρίνοντας τις ομοιότητες και τις διαφορές που υπάρχουν μεταξύ τους.

Abstract

This thesis focuses on the study of energy consumption and carbon dioxide emissions of vehicles, which are analyzed with the help of a developed self-diagnosis system. The global community faces significant problems of air pollution and increased energy consumption, which contribute to climate change and poor environmental quality in general. The thesis is based on green and digital transformation and more specifically in the transport sector. The creation of the self-diagnosis system was carried out through a laptop computer using Ubuntu software, an OBD-II cable to input the vehicle data and the development of the individual programs implemented in Python programming language, to analyze and map the recordings from the starting point to the recording completion point. Logging data is verified in graph form and then tabulated with the vehicle's Trip Computer and the GPS Logger navigation application installed on a mobile phone. In addition, the records of the routes divided into Extra-urban and Urban are reported, comparing the similarities and differences that exist between them.

Ευχαριστίες

Αρχικά θα ήθελα να ευχαριστήσω τον επιβλέπων καθηγητή της εργασίας μου, τον Κύριο Ονούφριο Χαραλάμπους για την στήριξη και βοήθεια που μου προσέφερε καθημερινά όλο αυτό το διάστημα κατά την εκπόνηση της εργασίας μου. Έπειτα θα ήθελα να ευχαριστήσω βαθύτατα τον αδελφό μου Βασίλειο Σεφερίδη για την προσφορά του αυτοκινήτου του, το οποίο ήταν ο βασικός πυλώνας για την πραγματοποίηση της πτυχιακής μου εργασίας, παρέχοντας την διαθεσιμότητα του οποιαδήποτε στιγμή το χρειαζόμουν σε όλο αυτό το χρονικό διάστημα. Ωστόσο, ένα πολύ μεγάλο ευχαριστώ θα ήθελα να πω στους γονείς μου, οι οποίοι από την αρχή που ξεκίνησα να φοιτώ στο Πανεπιστήμιο Θεσσαλίας, προσπάθησαν και κατάφεραν να μου προσφέρουν όλα όσα χρειαζόμουν για να μπορέσω να εκπληρώσω την επιθυμία μου ολοκλήρωσης των σπουδών μου στο τμήμα των Ψηφιακών Συστημάτων με τον καλύτερο δυνατό τρόπο.

Σεφερίδης Στυλιανός

15/07/2024

Περιεχόμενα

1 Εισαγωγή

Η περιβαλλοντική βιωσιμότητα και η ενεργειακή απόδοση, την εποχή του 21ου αιώνα είναι βασικοί παράγοντες ανησυχίας και προσφέρουν κίνητρο για την μελέτη της ενεργειακής κατανάλωσης των οχημάτων και περαιτέρω παραγόντων, ώστε να πραγματοποιηθεί η κατάλληλη έρευνα αντιμετώπισης τους. Αρχικά, για την κατανόηση της ενεργειακής κατανάλωσης απαιτείται κάτι παραπάνω από τις βασικές γνώσης στην εξέλιξη της τεχνολογίας. Πρόκειται για την προσέγγιση του τομέα της μηχανικής, την επιστήμη του περιβάλλοντος και κυριότερα την ανάλυση των δεδομένων.

Η παρούσα πτυχιακή "Ανάπτυξη εφαρμογής για την καταγραφή και ανάλυση δεδομένων OBD σε συνθήκες πραγματικής κυκλοφορίας οχήματος" αναφέρεται στην μέτρηση και μελέτη της ενεργειακής κατανάλωσης οχημάτων με την βοήθεια ενός ανεπτυγμένου συστήματος αυτοδιάγνωσης. Αυτό το σύστημα δημιουργήθηκε το ακαδημαϊκό έτος 2022-2023 από μια ομάδα ατόμων στην έκτοτε πτυχιακή και έπειτα έφτασε η στιγμή για την νέα του έκδοση. Πιο συγκεκριμένα, το σύστημα αυτό υλοποιήθηκε μέσω ενός φορητού υπολογιστή που συνοδεύεται με το λειτουργικό σύστημα Ubuntu, με την χρήση ενός OBD-II καλωδίου και με την ανάπτυξη των ήδη υπάρχων προγραμμάτων υλοποιημένα σε γλώσσα προγραμματισμού "Python", τα οποία συμβάλλουν στην εισαγωγή και εξαγωγή συγκεκριμένων δεδομένων του οχήματος, για την πραγματοποίηση της ανάλυσης τους και τον υπολογισμό της ενεργειακής κατανάλωσης που εκβάλλεται στην ατμόσφαιρα.

Για την ανάλυση των δεδομένων χρησιμοποιείται ένα πρόγραμμα Python και μια εφαρμογή που ονομάζεται "GPS Logger", τα οποία αναλύουν διάφορα δεδομένα του οχήματος όπως την απόσταση, την μέση ταχύτητα και την επιτάχυνση. Έπειτα, πραγματοποιείται η εξαγωγή τους σε γραφήματα, δημιουργώντας έναν πίνακα όπου εμπεριέχει επιπλέον δεδομένα που σχετίζονται με το όχημα. Στην συνέχεια, το πρόγραμμα εμφανίζει όλα τα προαναφερόμενα και την καταγεγραμμένη διαδρομή που εκτέλεσε το όχημα, με εμφανές το σημείο εκκίνησης και το πέρας της διαδρομής σε αρχείο HTML.

Βασικός στόχος της συγκεκριμένης πτυχιακής αποτελεί η προβολή και η ανάλυση των δεδομένων που αντλούνται από τα οχήματα, ώστε να χρησιμοποιηθούν για την βελτίωση της περιβαλλοντικής βιωσιμότητας και την εξοικονόμηση της ενεργειακής κατανάλωσης.

Στα παρακάτω κεφάλαια της πτυχιακής, και πιο συγκεκριμένα στο 2ο κεφάλαιο παρουσιάζεται το αντικείμενο μελέτης της πτυχιακής εργασίας. Κατ' επέκταση, το 3ο κεφάλαιο αναφέρεται στη μεθοδολογία μετρήσεων και των εξοπλισμών που χρησιμοποιήθηκαν για την πραγματοποίηση της δειγματοληψίας και της συλλογής δεδομένων. Ωστόσο, στο 4ο κεφάλαιο καταγράφεται αναλυτικά η ανάπτυξη λογισμικού και η επεξήγηση της χρησιμότητας του για τα προγράμματα συλλογής και ανάλυσης των δεδομένων αυτών. Ακόμη, στο 5ο κεφάλαιο αναφέρονται διάφορες μετρήσεις με τις αντίστοιχες ιδιαιτερότητες της κάθε μίας ξεχωριστά. Εν κατακλείδι, το 6ο κεφάλαιο αναγράφει τα συμπεράσματα των δειγμάτων και τις τελικές πραγματικές διαπιστώσεις σχετικά με την ενεργειακή κατανάλωση και την ρύπανση του περιβάλλοντος.

2 Αντικείμενο της εργασίας

Η παρούσα πτυχιακή εργασία βασίζεται στον πράσινο και ψηφιακό μετασχηματισμό και πιο συγκεκριμένα στον τομέα των μεταφορών. Σε αυτό το κεφάλαιο θα αναφερθεί ο βασικός προβληματισμός που σχετίζεται με την κατανάλωση καυσίμου και τις περιβαλλοντικές επιπτώσεις στον δρόμο, οι οποίες είναι αρκετά υψηλές σε σχέση με αυτές που ισχυρίζονται οι κατασκευαστές των οχημάτων. Οι κύριοι στόχοι της παρακάτω έρευνας είναι η συλλογή και ανάλυση δεδομένων OBD και GPS σε πραγματικές συνθήκες κυκλοφορίας, όπου εκτιμάται η κατανάλωση καυσίμου και περιβαλλοντικών επιπτώσεων σε διάφορες συνθήκες οδήγησης, που χαρακτηρίζονται ως Αστικές η Υπεραστικές διαδρομές. Ιδιαιτέρως σημαντική είναι και η επίτευξη μείωσης του διοξειδίου του άνθρακα στην ατμόσφαιρα, με βασική προϋπόθεση την ανάπτυξη ενός σχεδίου που θα ελαττώσει την ενεργειακή κατανάλωση και την περιβαλλοντική ρύπανση σε παγκόσμια κλίμακα.

2.1 Πράσινος και Ψηφιακός Μετασχηματισμός

Την ημέρα που αναρτήθηκε η επίσημη ημερομηνία κυκλοφορίας των οχημάτων στους δρόμους και ειδικότερα όταν πραγματοποιήθηκε η ένταξη τους στην καθημερινότητα των ανθρώπων, δημιουργήθηκε ένας νέος παράγοντας με καθοριστικό ρόλο στην υγεία της ανθρώπινης ζωής. Με την κυκλοφορία των οχημάτων και την κατανάλωση καυσίμου που απαιτείται για την λειτουργία τους, αυτομάτως προκαλείται ανησυχία στον τομέα της υγείας. Μέσα από την καύση των καυσίμων αποβάλλονται εκπομπές ρύπων και πιο συγκεκριμένα το διοξείδιο του άνθρακα, το οποίο είναι επιβλαβές στην ατμόσφαιρα και προκαλεί σημαντικές μολύνσεις στο αναπνευστικό σύστημα της ανθρώπινης ζωής.

Ο όρος “Πράσινος και Ψηφιακός Μετασχηματισμός” βασίζεται στην εισαγωγή καινούργιων και οικονομικότερων περιβαλλοντικών τεχνολογιών για την μείωση των εκπομπών άνθρακα και την ανάπτυξη της βιωσιμότητας. Ειδικότερα, ο Πράσινος Μετασχηματισμός αναφέρεται στις μεθόδους ανάπτυξης επαναχρησιμοποιούμενων πηγών ενέργειας και στην ελάττωση της ενεργειακής υπερκατανάλωσης. Ενώ, ο Ψηφιακός Μετα-

σχηματισμός κυμαίνεται στην ανάπτυξη των επιμέρους τεχνολογιών και την αυτοματοποίηση τους, έχοντας ως κοινό παρονομαστή την πράσινη μετάβαση για την δημιουργία μιας οικονομικότερης και βιώσιμης καθημερινότητας στο άμεσο μέλλον[1].

Εικόνα 2.1: Πράσινη Προσαρμογή[2].

2.1.1 Τομέας μεταφορών

Ο τομέας μεταφορών είναι βασικός παράγοντας του Πράσινου και Ψηφιακού Μετασχηματισμού, που επηρεάζει δραστικά το περιβάλλον, την ασφάλεια των ανθρώπων ακόμα και την ποιότητα ζωής τους. Σύμφωνα με πρόσφατες ανακοινώσεις η Ευρωπαϊκή Επιτροπή συγκλίνει σε μία Ευρωπαϊκή Πράσινη Συμφωνία, με στόχο το μελλοντικό επίτευγμα της κλιματικής ουδετερότητας, η οποία πρόκειται να πραγματοποιηθεί στο μακρινό έτος του 2050. Ωστόσο, η συνεκτίμηση των παραπάνω προκαλεί κίνητρο για πρόσθετες μελέτες και προσπάθειες, με στόχο την ακριβή ανάπτυξη ενός εξειδικευμένου τεχνολογικού σχεδίου στον τομέα μεταφορών, που θα έχει ως αποτέλεσμα την δραστική μείωση εκβολής διοξειδίου του άνθρακα στις μεταφορές των οχημάτων και στην μεταβατική αλλαγή του κλιματικού συστήματος των μεταφορών[3].

2.1.2 Το πρόβλημα

Σύμφωνα με μελέτες και δοκιμές που πραγματοποιήθηκαν σε διάφορα οχήματα, η κατανάλωση καυσίμου και οι περιβαλλοντικές επιπτώσεις στον δρόμο, είναι μεγαλύτερες από αυτές που ισχυρίζονται οι κατασκευαστές. Αρχικά, πραγματοποιήθηκαν δοκιμές από έναν αρκετά μεγάλο αριθμό οχημάτων που δέχονται τον τύπο καυσίμου βενζίνης ή πετρελαίου αντίστοιχα, με την εκπομπή του διοξειδίου του άνθρακα και την συνολική καύση των καυσίμων να υπερβαίνουν το 20% των τιμών από τις προκαθορισμένες τιμές που είχαν ορισθεί για την πραγματοποίηση δοκιμών στο πρότυπο του WLTP[4].

2.2 Στόχοι

Οι βασικοί στόχοι της συγκεκριμένης πτυχιακής εργασίας είναι η συλλογή και ανάλυση δεδομένων OBD και GPS σε πραγματικές συνθήκες κυκλοφορίας με την εκτίμηση κατανάλωσης των καυσίμων και των περιβαλλοντικών επιπτώσεων σε διάφορες συνθήκες οδήγησης, οι οποίες κατατάσσονται σε Αστικές και Υπεραστικές διαδρομές. Η Αστική διαδρομή ονομάζεται η διαδρομή που πραγματοποιείται από τον οδηγό ενός οχήματος μέσα σε πόλεις η ορεινά μέρη, εκ των οποίων δεν είναι δυνατή η συνεχόμενη επιτάχυνση της ταχύτητας σε μεγάλες τιμές. Αντιθέτως, η Υπεραστική διαδρομή χαρακτηρίζεται η διαδρομή η οποία πραγματοποιείται στους επονομαζόμενους αυτοκινητόδρομους, οδοί που αγγίζουν εκτάσεις αμέτρητων χιλιομέτρων, επεξηγηματικά πιο ανοιχτούς και συνεχόμενους δρόμους χωρίς απότομες στροφές και εναλλαγές.

3 Μεθοδολογία μετρήσεων

Το παρών κεφάλαιο επικεντρώνεται στην μεθοδολογία μετρήσεων που πραγματοποιήθηκαν από καταγραφές σε συνθήκες πραγματικής κυκλοφορίας ενός οχήματος. Παρακάτω αναλύονται ο εξοπλισμός, τα εξαρτήματα και οι εφαρμογές που χρειάστηκαν για την διαδικασία καταγραφής μιας διαδρομής και η ανάλυση των δεδομένων που συλλέχθηκαν ώστε να προκύψει η αναλυτική αναφορά και η ομαδοποίηση τους.

3.1 Εξοπλισμός

Ο εξοπλισμός είναι το κεντρικό σημείο της πτυχιακής εργασίας, διότι συμβάλλει στην συλλογή αλλά και στην ανάλυση δεδομένων καταγραφής. Επίσης, καθορίζει την διαδικασία της δειγματοληψίας σε συνεργασία με τα υπόλοιπα εργαλεία για ένα αναλυτικό αποτέλεσμα. Παρακάτω παρουσιάζεται επακριβώς, ο εξοπλισμός και τα εξαρτήματα που χρειάστηκαν για την εκπόνηση της παρούσας πτυχιακής εργασίας.

3.1.1 Όχημα

Το όχημα που χρησιμοποιήθηκε για την άντληση και ανάλυση των δεδομένων σε αυτήν την μελέτη είναι της μάρκας αυτοκινήτων Renault. Το μοντέλο είναι της γενιάς Clio Sports Tourer και ο τύπος του αυτοκινήτου ονομάζεται Caravan λόγου του όγκου που διαθέτει και το έτος κυκλοφορίας του, το 2015. Ο κινητήρας του παραπάνω οχήματος καταναλώνει καύσιμο Πετρελαίου και διαθέτει 90 ίππους στο εσωτερικό του. Περιέχει 5 ταχύτητες στο κιβώτιο του και την καθιερωμένη Reverse για την όπισθεν. Η χωρητικότητα του ρεζερβουάρ καυσίμου του είναι στο ύψος των 45 λίτρων. Ακόμη, στο σαλόνι του, ανάμεσα στον διαχωρισμό των δύο μπροστινών καθισμάτων του οδηγού και του συνοδηγού, εμπεριέχει ένα Trip Computer που συνεισφέρει σε διάφορες ανάγκες του οδηγού για το αυτοκίνητο και για τις διαδρομές που επιχειρεί[5].

Εικόνα 3.1: Όχημα εργασίας[6].

3.1.2 Φορητός υπολογιστής

Για την διαδικασία εισαγωγής και αποθήκευσης των δεδομένων που έχουν αντληθεί από το όχημα είναι αναγκαστικό να υπάρχει διαθέσιμος προς χρήση φορητός υπολογιστής, διότι βοηθάει το μικρό του μέγεθος για την μεταφορά του, σε αντίθεση με μία ολόκληρη μονάδα υπολογιστή, και επιπλέον είναι προσβάσιμος για την καταγραφή των διαδρομών και των δεδομένων. Το συγκεκριμένο μοντέλο φορητού υπολογιστή που χρησιμοποιήθηκε για την έρευνα, κατασκευάστηκε από την εταιρεία HP και κυκλοφόρησε το 2013. Τα χαρακτηριστικά του δεν συναρπάζουν, καθώς στο εσωτερικό του εμπεριέχεται μια μνήμη RAM των 4 Gigabyte (GB), έναν επεξεργαστής της εταιρείας Intel και της γενιάς Celeron. Επίσης, με το πέρασμα των χρόνων και με την εξέλιξη των εφαρμογών πραγματοποιήθηκε αναγκαστική αλλαγή του σκληρού δίσκου από Hard Disk Drive (HDD) σε Solid State Drive (SSD), βελτιώνοντας την λειτουργία του υπολογιστή, μειώνοντας δραστικά τον χρόνο φόρτισης του λογισμικού και των εφαρμογών. Αξίζει να σημειωθεί πως το λειτουργικό σύστημα που χρησιμοποιήθηκε για τον συγκεκριμένο φορητό υπολογιστή είναι το Ubuntu, καθώς είναι το μόνο λειτουργικό σύστημα, συμβατό με τα πρωτόκολλα που εγκαταστάθηκαν και την πολυπλοκότητα του προγράμματος για την καταγραφή των δεδομένων.

Εικόνα 3.2: Laptop με λογισμικό Ubuntu[7].

3.1.3 Καλώδιο OBD-II

Το καλώδιο OBD της έκδοσης πρωτοκόλλου II είναι ο παράγοντας εκ του οποίου μεταφέρονται τα δεδομένα του οχήματος στον υπολογιστή για την ομαδοποίηση τους. Αρχικά το OBD υλοποιήθηκε από την εταιρεία General Motors την δεκαετία του 81 και ο στόχος δημιουργίας του σχετιζόταν με τον έλεγχο εκπομπών καυσαερίων των οχημάτων στην ατμόσφαιρα. Από πολύ νωρίς η θύρα OBD εντάχθηκε στο εσωτερικό των οχημάτων με την προϋπόθεση να παρατηρείται η εκπομπή των καυσαερίων και το ποσοστό ρύπανσης που παράγουν με την χρήση τους. Η πρώτη επίσημη έκδοση του είχε αρκετά προβλήματα στην διάγνωση των σφαλμάτων καθώς η μόνη μέτρηση που αναγνώριζε ήταν εκπομπές ρύπων του κινητήρα, αλλά με το πέρασμα των χρόνων και με την δημιουργία νέων μοντέλων, οι ελλείψεις άρχισαν να μειώνονται και πλέον υπάρχει ποικιλία δεδομένων που μπορούν να οδηγήσουν στην διάγνωση οποιουδήποτε προβλήματος προκύψει στον εγκέφαλο ενός οχήματος. Σε αυτήν την εργασία το καλώδιο αυτό συνεισφέρει στην εισαγωγή δεδομένων από το αυτοκίνητο στον φορητό υπολογιστή, παράλληλα υπολογίζοντας την συνολική απόσταση που διένυσε το αυτοκίνητο, την μέση ταχύτητα κατά την διάρκεια της διαδρομής, την συνολική κατανάλωση καυσίμου για την εκτέλεση της διαδρομής, την μέγιστη επιτάχυνση και επιβράδυνση, την μέγιστη πίεση πεταλιού στο φρενάρισμα, την ενεργειακή ισχύ για όλη την απόσταση και τον υπολογισμό εκπομπής του διοξειδίου του άνθρακα που εξάγεται στην ατμόσφαιρα[8].

Εικόνα 3.3: Καλώδιο OBDII[9].

3.1.4 Trip Computer

Όπως αναφέρθηκε σε προηγούμενο κεφάλαιο το Trip Computer εμπεριέχεται μέσα σε αρκετά οχήματα, επεξηγηματικά είναι ένας υπολογιστής τοποθετημένος στο σαλόνι του οχήματος, αναγκαίος για διάφορους λόγους, όπως για τον υπολογισμό της διαδρομής που διένυσε το όχημα, την κατανάλωση καυσίμου για όλη την διαδρομή και την μέση ταχύτητα κίνησης καθ' όλη την διάρκεια. Στα πιο σύγχρονα μοντέλα αυτοκινήτων οι υπολογιστές αυτοί διαθέτουν το δικό τους λογισμικό σύστημα και προσφέρουν μια ποικιλία υπηρεσιών, όπως η χαρτογράφηση του δρόμου σε μια μικρή ακτίνα από την θέση εντοπισμού του αυτοκινήτου και την χρήση του GPS για την δημιουργία μιας έμπιστης και πιο γρήγορης διαδρομής για τον οδηγό να φτάσει έγκαιρα στον προορισμό του. Έπειτα, ένα πολύ βασικό χαρακτηριστικό που διαθέτουν είναι η διάγνωση του ορίου ταχύτητας σε όλους τους δρόμους της διαδρομής, σε περίπτωση υπέρβασης του οδηγού να εμφανιστεί μία επισήμανση και ένας θόρυβος, ως υπενθύμιση στον οδηγό να μειώσει την ταχύτητα κίνησης του για την απόφευξη τυχόν ατυχήματος. Σε αντίθεση με παλαιότερα μοντέλα όπου τα χαρακτηριστικά τους περιορίζονται στο εσωτερικό και διαθέτουν μόνο τη μέση ταχύτητα, τη κατανάλωση καυσίμου του λίτρου ανά 100 χιλιόμετρα και τη θερμοκρασία, που επικρατεί στην ατμόσφαιρα. Η χρήση του Trip Computer για την εκπόνηση της εργασίας έλαβε μέρος με σκοπό την επαλήθευση των δεδομένων της συνολικής απόστασης της διαδρομής, της μέσης ταχύτητας και τέλος τον υπολογισμό της κατανάλωσης του καυσίμου για όλη την καταγεγραμμένη διαδρομή[10].

Εικόνα 3.4: Δεδομένα του Trip Computer από καταγεγραμμένη διαδρομή.

3.2 Εφαρμογές

Σε αυτό το κεφάλαιο θα αναφερθούν οι εφαρμογές που χρησιμοποιήθηκαν για την εισαγωγή και ανάλυση των δεδομένων που πραγματοποιήθηκε, με σκοπό την ομαδοποίηση τους και την εμφάνιση τους σε ένα φιλικό περιβάλλον προς τον αναγνώστη.

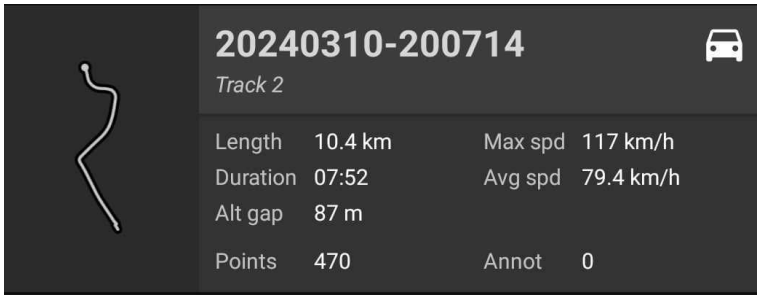
3.2.1 Visual Studio Code (VSC)

Το Visual Studio Code είναι ένα πρόγραμμα που κατασκευάστηκε από την εταιρεία Microsoft, με στόχο να μεταφέρουν τον προγραμματισμό στην μοντέρνα και ψυχαγωγική μορφή του, προσφέροντας όλες τις υπηρεσίες που χρειάζονται οι προγραμματιστές για την αντιμετώπιση των προβλημάτων που αντικατοπτρίζουν κατά την διάρκεια της συγγραφής κώδικα, καθώς διαθέτει ένα ευχάριστο και ξεκούραστο περιβάλλον προς τον χρήστη. Βασικό προνόμιο της εφαρμογής αυτής αποτελεί η ανίχνευση σφαλμάτων σε περίπτωση λάθους δομής και σύνταξης κώδικα και έπειτα η πρόταση μιας ενδεικτικής λύσης διόρθωσης. Επίσης, υποστηρίζει όλες τις γλώσσες προγραμματισμού που διατίθενται παγκοσμίως, αλλά και γλώσσες πιο σύγχρονες των τελευταίων ετών. Ακόμη, δίνει την δυνατότητα στον χρήστη να επαναφέρει λανθασμένες αλλαγές στον κώδικα οι οποίες μπορεί να έχουν υλοποιηθεί από οποιαδήποτε συσκευή σε αόριστο χρονικό διάστημα. Αξίζει να σημειωθεί πως στο εσωτερικό της εφαρμογής, οι χρήστες μπορούν να εκμεταλλευτούν μια ποικιλία επεκτάσεων που προσφέρει το Visual Studio Code για το φόντο και το θέμα που επιθυμεί να πειραματιστεί ο χρήστης, καθώς και άλλα χρήσιμα και καθοριστικά εργαλεία για την εκάστοτε γλώσσα χρήσης του προγραμματισμού, επεκτείνοντας την δημιουργικότητα και την φαντασία του. Αυτή είναι η εφαρμογή μέσω της οποίας αναπτύχθηκαν τα μέρη του κώδικα για την παρούσα εργασία. Όλες οι δοκιμές, οι μετρήσεις, τα

γραφήματα, οι χάρτες, ακόμα και οι πίνακες που πραγματοποιήθηκαν καθ' όλη την διάρκεια της εκπόνησης της, εισάχθηκαν μέσω της εφαρμογής του Visual Studio Code[11].

3.2.2 GPS-Logger

Το GPS-Logger ονομάζεται μια εφαρμογή με δυνατότητα εγκατάστασης σε κινητό τηλέφωνο και συνεισφέρει στην χαρτογράφηση της παραγόμενης διαδρομής του αυτοκινήτου κατά την διάρκεια της καταγραφής, με την βοήθεια και την συνεργασία πολλαπλών δορυφόρων. Για να πραγματοποιηθεί αυτή η διαδικασία, ο χρήστης θα πρέπει να ενεργοποιήσει την υπηρεσία της τοποθεσίας του τηλεφώνου σε κοινή χρήση με την εφαρμογή, ώστε να καταφέρει το GPS να αναγνωρίσει την ακριβή τοποθεσία της συσκευής και να καταγραφεί έγκαιρα η παραγόμενη διαδρομή χωρίς σφάλματα. Τα δεδομένα που εισάγονται με την έναρξη της καταγραφής είναι η ταχύτητα, το γεωγραφικό πλάτος, το γεωγραφικό μήκος, η ημερομηνία της καταγραφής, ακόμα και το νούμερο του δορυφόρου που χρησιμοποιήθηκε για την κάθε γραμμή καταγραφής ξεχωριστά. Ωστόσο με τον τερματισμό της καταγραφής, η εφαρμογή αυτόματα αποθηκεύει την διαδρομή σε ένα αρχείο κειμένου, με τα παραπάνω δεδομένα ομαδοποιημένα σε σειρά. Το αρχείο αυτό είναι ένα αρχείο επαλήθευσης των δεδομένων που παράγονται από το αυτοκίνητο, με την διαφορά ότι τα δεδομένα αυτά παράγονται μέσω των δορυφόρων συνεργασίας με το κινητό τηλέφωνο την δεδομένη χρονική στιγμή. Στο τέλος, πραγματοποιείται μία ανάλυση και σύγκριση των δύο αρχείων, του αυτοκινήτου και του κινητού τηλεφώνου, για τον εντοπισμό αποκλίσεων αλλά και ομοιοτήτων που πρόκειται να εμφανιστούν, σχετικά με τα κοινά τους δεδομένα, όπως είναι η μέση ταχύτητα, η συνολική απόσταση αλλά κυρίως η εκπομπή του διοξειδίου του άνθρακα στην ατμόσφαιρα με την κίνηση του οχήματος[12].



Εικόνα 3.5: Σχεδιασμός παραγόμενης διαδρομής μέσω GPS-Logger.

4 Ανάπτυξη Λογισμικού

Σε αυτό το κεφάλαιο περιγράφεται η γλώσσα προγραμματισμού μέσω της οποίας αναπτύχθηκε η σύνταξη κώδικα του προγράμματος συλλογής δεδομένων καταγραφής και ανάλυσης τους. Δεδομένα που χρησιμοποιήθηκαν για την πραγματοποίηση δειγμάτων σε συνθήκη πραγματικής κυκλοφορίας οχήματος.

4.1 Γλώσσα προγραμματισμού Python

Η Python χαρακτηρίζεται ως μία από τις δημοφιλέστερες γλώσσες προγραμματισμού παγκοσμίως, διότι χρησιμοποιείται για ένα μεγάλο ποσοστό εφαρμογών και πιο συγκεκριμένα για την κατασκευή ιστοσελίδων, την αυτοματοποίηση εργασιών και κυρίως για την ανάλυση δεδομένων. Αυτό οφείλεται στην απλή σύνταξη γραφής και κατανόησης αλλά και στο κομμάτι της ευελιξίας, καθώς οι χρήστες της μπορούν να κατασκευάσουν ένα πρόγραμμα σε σύντομο χρονικό διάστημα, σε αντίθεση με διαφορετικές γλώσσες προγραμματισμού, γνωστές για την πολυπλοκότητα της σύνταξης τους. Βασική διαφορά με τις υπόλοιπες γλώσσες προγραμματισμού είναι η ποικιλία βιβλιοθηκών της Python, όπως η *pandas* και η *matplotlib*, οι οποίες επιλέγονται για την επιστήμη των υπολογιστών και δεδομένων, διευκολύνοντας τους προγραμματιστές για οποιαδήποτε εφαρμογή επιθυμούν να δημιουργήσουν[13].

4.2 Καταγραφή δεδομένων OBDII

Ακολουθεί η αναφορά της ήδη υπάρχουσας εφαρμογής και των επιπρόσθετων τροποποιήσεων στον κώδικα της, μέσω των οποίων πραγματοποιείται η καταγραφή των δεδομένων από την διαγνωστική θύρα OBDII του αυτοκινήτου. Ωστόσο, η έκδοση της Python που χρησιμοποιήθηκε για την ανάπτυξη της εφαρμογής είναι η 3.11.

Στον παραπάνω κώδικα προστέθηκε η εντολή LOAD, η οποία υπολογίζει την τιμή του φορτίου που την τοποθετεί ως παράμετρο και αναδεικνύει την ποσότητα της δυναμικής

ισχύς που χρησιμοποιείται από τον κινητήρα σε κάθε χρονική στιγμή κατά την διάρκεια της καταγραφής. Παρακάτω ακολουθεί η ανάλυση του επιπρόσθετου κώδικα:

Ορισμός της συνάρτησης LOAD:

```
# ENGINE LOAD
def new_load(load):
    global load_value
    load_value = load.value.magnitude
    print("\t")
```

Ο σκοπός της παραπάνω συνάρτησης είναι η άμεση ενημέρωση της μεταβλητής load_value με τα δεδομένα φορτίου από τον κινητήρα του αυτοκινήτου που θα λάβει μέσω της διεπαφής του OBDII, διατηρώντας την πιο πρόσφατη τιμή φορτίου του κινητήρα.

Ρύθμιση της εντολής LOAD:

```
connection.watch(obd.commands.ENGINE_LOAD, callback=new_load)
```

Σε αυτήν την γραμμή κώδικα δημιουργείται μια σύνδεση για την παρακολούθηση της "ENGINE_LOAD". Με τον κάθε εντοπισμό νέων δεδομένων από το φορτίο του κινητήρα, αυτομάτως πραγματοποιείται η κλήση της "new_load" για την επεξεργασία αυτών των δεδομένων.

Εγγραφή και ομαδοποίηση δεδομένων σε λίστα:

```
mylist = ["TIME[s]", "SPEED[km/h]", "TIMESTAMP", "LOAD[%]", "RPM[rpm]", "COOLANT[degC]",
"MAF[g/s]", "INTAKE_PRESSURE", "THROTTLE_POSITION[%]"]
```

Στο παραπάνω κομμάτι του κώδικα της ήδη υπάρχουσας λίστας προστέθηκε η "LOAD[%]", η οποία είναι μια επιπλέον παράμετρος που αποθηκεύει τα δεδομένα για το φορτίο του κινητήρα και ομαδοποιείται με την σειρά της στο αρχείο CSV.

Εγγραφή και εκτύπωση δεδομένων καταγραφής:

```
    try:
        while True:
            timecount = time.time() - starttime
            timestamp = int(time.time())  # Convert datetime to timestamp
            print("File Write:", "{:.4f}".format(timecount), speed_value, timestamp,
load_value, rpm_value, coolant_value, maf_value,
                  intake_pressure_value, throttle_position_value)
            f.write(

f"{timecount:.2f},{speed_value},{timestamp},{load_value:.2f},{rpm_value},{cool-
ant_value},{maf_value:.2f},{intake_pressure_value},{throttle_position_value:.2f}\n")
```

Στο επιμέρους κομμάτι κώδικα και πιο συγκεκριμένα μέσα στον βρόχο, προστέθηκε η τιμή “load_value” και μαζί με όλες τις υπάρχουσες τιμές αποθηκεύονται στο αρχείο του CSV. Επιπροσθέτως, η εγγραφή του φορτίου του κινητήρα γίνεται με δείκτη μορφοποίησης που εμφανίζει την τιμή σε δύο δεκαδικά ψηφία, με σκοπό την ακρίβεια και την αποφυγή περαιτέρω δεκαδικών στην εκτύπωση των δεδομένων.

4.3 Ανάλυση δεδομένων

Ο παρακάτω κώδικας είναι υπεύθυνος για την ανάλυση και την επεξεργασία των δεδομένων που προέκυψαν, από τις μετρήσεις καταγραφής, μέσω του αρχείου CSV, το οποίο εμπεριέχει τα δεδομένα της εκάστοτε καταγραφής. Η ανάπτυξη του κώδικα πραγματοποιήθηκε χρησιμοποιώντας την 3.11 έκδοση της Python.

Εισαγωγή χρήσιμων βιβλιοθηκών:

```
import xml.etree.ElementTree as ET
import csv
import pandas as pd
from datetime import datetime, timezone
import folium
import pytz
```

Αυτό το κομμάτι του κώδικα εισάγει τις βιβλιοθήκες που θα χρησιμοποιηθούν για τη διαχείριση των δεδομένων, των ημερομηνιών και την χαρτογράφηση των καταγραφών.

Οργάνωση και ανάλυση δεδομένων:

```
def readgpx(gpx_file_path: str) -> pd.DataFrame:
    # Parse the GPX file
    tree = ET.parse(gpx_file_path)
    root = tree.getroot()
    creator = root.attrib['creator']

    # Create CSV header
    csv_header = ["TIME[s]", "SPEED[km/h]", "TIMESTAMP", "LATITUDE", "LONGITUDE",
"ALTITUDE"]
    gpxdata = pd.DataFrame(columns=csv_header)

    # Write data to CSV file
    coordinates = []  # List to store coordinates for PolyLine
    ns = {'gpx': 'http://www.topografix.com/GPX/1/1', 'osmand': 'https://osmand.net'}
    nsZepp = {'gpx': 'http://www.topografix.com/GPX/1/1', 'ns3':
'http://www.garmin.com/xmlschemas/TrackPointExtension/v1'}
    time_offset = None
    for trkpt in root.findall('.//gpx:trkpt', namespaces=ns):
```

```
        dtime_str = trkpt.find('gpx:time', namespaces=ns).text
        pdatetime = datetime.strptime(dtime_str, "%Y-%m-
%dT%H:%M:%SZ").replace(tzinfo=pytz.UTC)
        if time_offset is None:
            time_offset = pdatetime.timestamp()
        time = pdatetime.timestamp() - time_offset
        latitude = float(trkpt.get('lat'))
        longitude = float(trkpt.get('lon'))
        altitude = float(trkpt.find('gpx:ele', namespaces=ns).text)

        match creator:
            case "osmand" | "OsmAnd 4.5.10" | "OsmAnd 4.4.6":
                speed_element = trkpt.find('gpx:extensions/osmand:speed', namespaces=ns)
            case "ZeppLife App":
                speed_element = trkpt.find('gpx:extensions/ns3:TrackPointExten-
sion/ns3:speed', namespaces=nsZepp)
            case 'BasicAirData GPS Logger 3.2.2':
                speed_element = "0.0"
            case _:
                raise Exception('gpx file creator not supported.')

        if speed_element is None:
            continue
        else:
            speed_value = float(speed_element.text) * 3.6

        gpxdata.loc[len(gpxdata)] = [time, speed_value, pdatetime.timestamp(), latitude,
longitude, altitude]

    return gpxdata
```

Στο παραπάνω τμήμα του κώδικα ορίζεται μια συνάρτηση, η οποία δέχεται δεδομένα μέσω ενός αρχείου GPX (αρχείο το οποίο καταγράφει διαδρομές GPS) σε κάθε σημείο από την καταγεγραμμένη διαδρομή και μετέπειτα μετατρέπει αυτά τα δεδομένα σε πίνακα. Στην συνέχεια ο πίνακας εμφανίζει τις πληροφορίες του χρόνου, της ταχύτητας, το γεωγραφικό πλάτος και μήκος, καθώς και το υψόμετρο της διαδρομής. Κάθε φορά που βρίσκει μια τιμή ταχύτητας, την μετατρέπει σε χιλιόμετρα ανά ώρα. Ωστόσο αν ο τύπος αρχείου δεν είναι GPX και δεν έχει δημιουργηθεί από τις εφαρμογές OsmAnd, ZeppLife App και BasicAirData GPS Logger, τότε δεν υποστηρίζεται και ο κώδικας εμφανίζει μήνυμα λάθους.

Εμφάνιση ταχύτητας οχήματος με χρωματική αναπαράσταση:

```
def determine_color(speed):
    if 0 <= speed <= 60:
        return 'blue'
    elif 60 < speed <= 90:
        return 'green'
    elif speed > 90:
        return '#C4A484'
```

Ο σκοπός της συνάρτησης “determine_color” είναι να εμφανίζει ανάλογα χρώματα με την επιμέρους ταχύτητα του οχήματος καθ’ όλη την διάρκεια της διαδρομής, επιστρέφοντας το μπλε χρώμα για ταχύτητες από 0 έως 60 km/h, το πράσινο χρώμα για ταχύτητες από 61 έως 90 km/h και ανοιχτό καφέ χρώμα για ταχύτητες πάνω από 90km/h.

Δημιουργία χαρτογράφησης διαδρομών με δεδομένα των καταγραφών:

```
def createMap(gpxdata):
    latitude = list(gpxdata['LATITUDE'])
    longitude = list(gpxdata['LONGITUDE'])
    speed = list(map(float, gpxdata['SPEED[km/h]']))

    coordinates = list()

    # Create a map centered around the first set of coordinates
    my_map = folium.Map(location=[latitude[-1], longitude[-1]], zoom_start=15)

    for i in range(len(latitude)):
        coordinates.append([latitude[i], longitude[i]])

        # Determine color based on speed ranges
        color = determine_color(speed[i])

        # Add a spot whenever the speed is "0"
        if speed[i] == 0:
            folium.Marker([latitude[i], longitude[i]], popup=f"Speed: {speed[i]:.2f}
km/h",
                          icon=folium.Icon(color='black')).add_to(my_map)

        # Add a colored PolyLine based on speed
        if i > 0:  # Skip the first point to avoid issues with PolyLine
            folium.PolyLine(locations=[coordinates[i - 1], coordinates[i]], color=color,
weight=3).add_to(my_map)

    # Add a red pin at the beginning of the route
    folium.Marker([latitude[0], longitude[0]], popup="Start", icon=fo-
lium.Icon(color='red')).add_to(my_map)

    # Add a blue pin at the end of the route
    folium.Marker([latitude[-1], longitude[-1]], popup="End", icon=fo-
lium.Icon(color='blue')).add_to(my_map)

    # Fit the map bounds to include the entire route
    my_map.fit_bounds([[min(latitude), min(longitude)], [max(latitude), max(longi-
tude)]])
```

Η συνάρτηση “createMap” δέχεται τα δεδομένα του αρχείου GPX με τις τιμές της ταχύτητας, του γεωγραφικού πλάτους και μήκους, απεικονίζοντας την διαδρομή του αυτοκινήτου από την έναρξη της καταγραφής μέχρι και το τέλος της. Στην συνέχεια, ορίζει τον χάρτη να ανοίγει στο σημείο καταγραφής της διαδρομή, με την βοήθεια των συντεταγμένων. Σε κάθε σημείο της διαδρομής, όπου το όχημα είναι τελείως ακίνητο και έχει ταχύτητα 0 km/h, αυτόματα σε εκείνο το σημείο τοποθετείται ένας μαύρος δείκτης. Στην

συνέχεια, δημιουργεί την γραμμή της καταγεγραμμένης διαδρομής, η οποία χρωματίζεται ανάλογα με την ταχύτητα του οχήματος μέσω της συνάρτησης “determine_color” που αναφέρθηκε παραπάνω. Ακόμη, προσθέτει ένα κόκκινο δείκτη στο σημείο εκκίνησης της καταγραφής και ένα μπλε δείκτη στο τέλος της.

Δημιουργία και αποθήκευση αρχείου χάρτη:

```
if __name__ == "__main__":
    # Specify the GPX file path
    gpx_file_path = "data/stelios/28_01_2024/osmand.gpx"
    converted_csv_file = "Converted_CSV.csv"  # Specify the new filename
    embedded_map_file = "embedded_osmand.html"

    gpxdata = readgpx(gpx_file_path)
    gpxdata.to_csv(converted_csv_file, index=False)

    my_map = createMap(gpxdata)

    # Save the map as an HTML file
    my_map.save(embedded_map_file)

    # Generate the HTML content with an embedded iframe
    with open(embedded_map_file, 'r') as map_file:
        map_content = map_file.read()

    embedded_map_content = f"""
        <html>
        <head>
            <title>Embedded Map</title>
        </head>
        <body>
            <h1>Embedded Map</h1>
            <iframe srcdoc='{map_content}' width='100%' height='600'></iframe>
        </body>
        </html>
        """
    # Save the embedded HTML content
    with open(embedded_map_file, 'w') as embedded_map_file:
        embedded_map_file.write(embedded_map_content)
```

Στο συγκεκριμένο κομμάτι του κώδικα δημιουργείται ένα αρχείο HTML που στο εσωτερικό του υπάρχει ενσωματωμένος ο χάρτης της διαδρομής, με την προϋπόθεση να μπορεί ο χρήστης να το ανοίξει σε ένα πρόγραμμα περιήγησης.

Ολοκλήρωση επεξεργασίας δεδομένων:

```
print(f"Conversion complete. Data written to {converted_csv_file}. Embedded map saved to
{embedded_map_file.name}.")
```

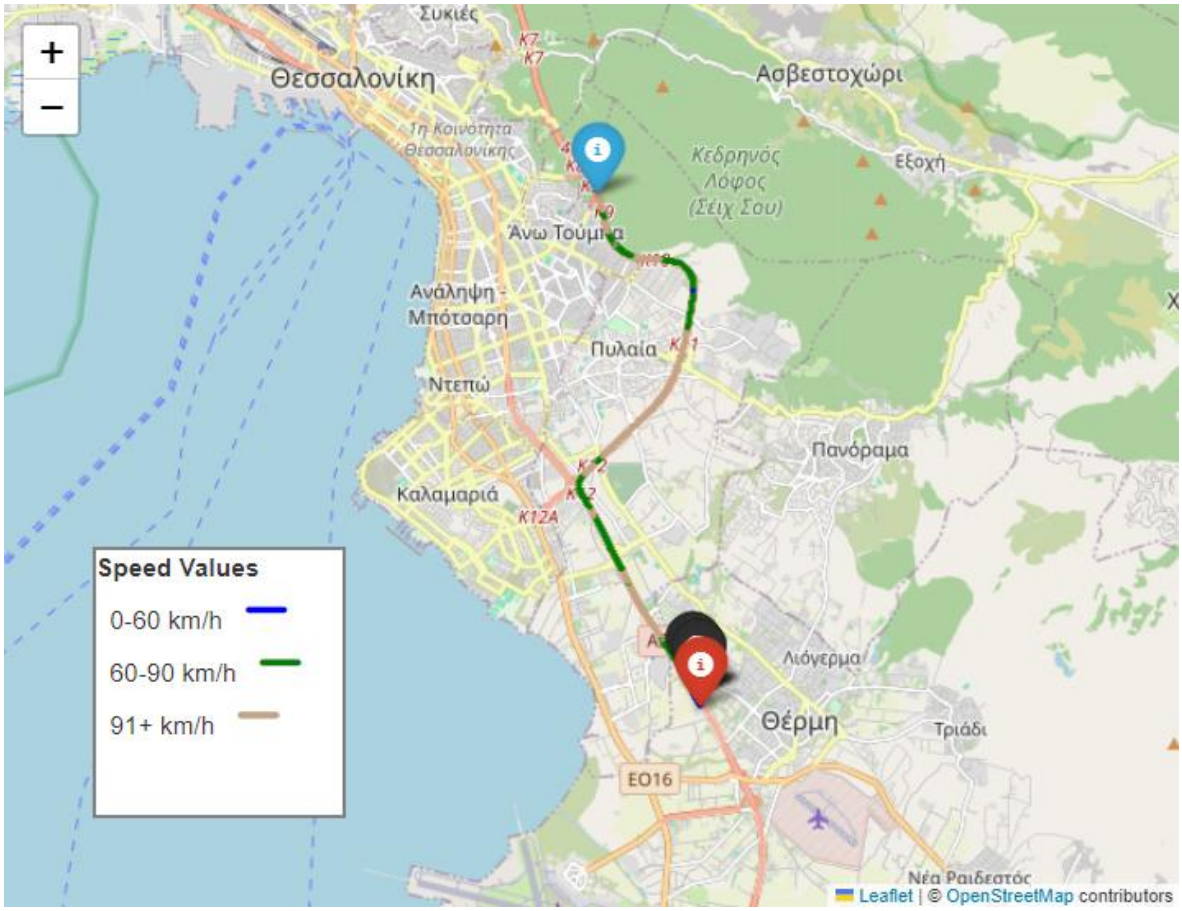
Η παραπάνω γραμμή κώδικα εμφανίζει ένα μήνυμα ολοκλήρωσης της μετατροπής των δεδομένων και την αποθήκευση τους σε ένα αρχείο CSV, καθώς και την αποθήκευση του ενσωματωμένου χάρτη σε ένα αρχείο HTML

5 Καταγραφές διαδρομών

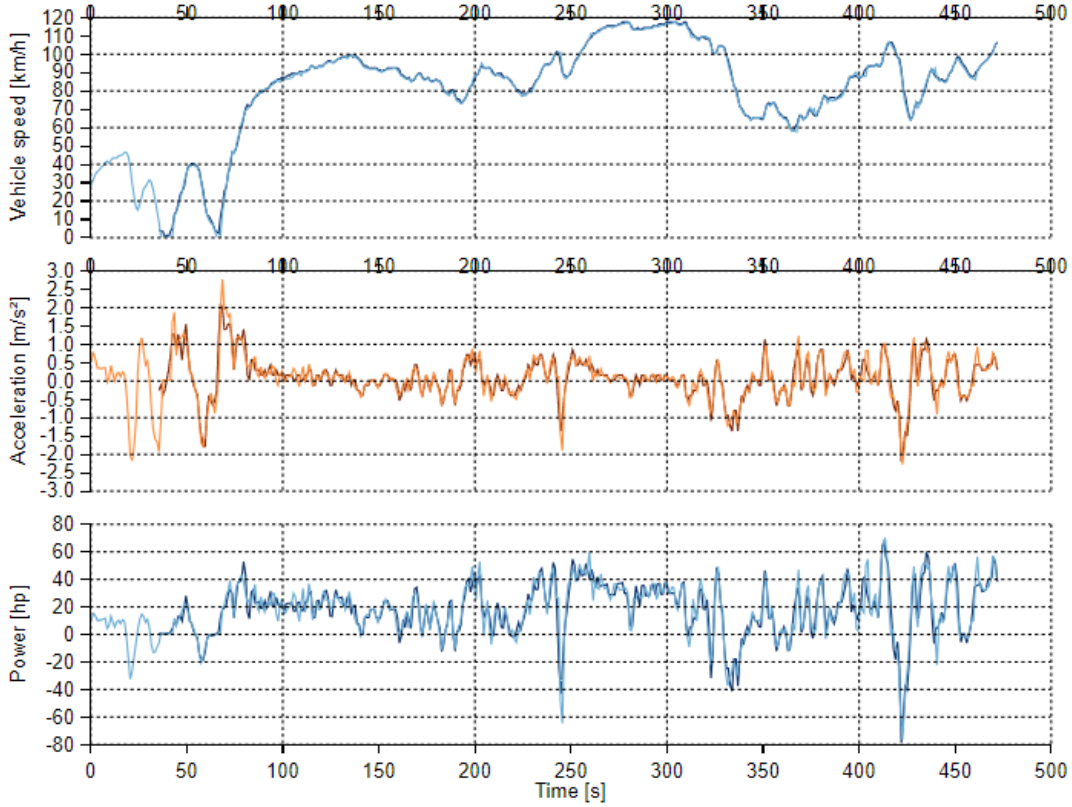
Σε αυτό το κεφάλαιο αναφέρονται οι καταγραφές των διαδρομών που πραγματοποιήθηκαν για την εγγραφή και την επιμέρους ανάλυση των δεδομένων. Αυτές οι καταγραφές συμπεριλαμβάνουν μετρήσεις καταναλώσεων καυσίμου, το ποσοστό φορτίου κινητήρα και τον υπολογισμό του διοξειδίου του άνθρακα, που υπολογίστηκε κατά την καταγραφή της διαδρομής. Παράλληλα, λαμβάνονται ως παράμετροι: το υψόμετρο του δρόμου, η συνολική μάζα των επιβατών και η συνολική ενεργειακή ισχύ του οχήματος, καθώς και το διοξείδιο του άνθρακα που εκβάλλεται στην ατμόσφαιρα.

5.1 Κανονική Υπεραστική

Η παρακάτω διαδρομή καταγράφηκε σε περιφερειακό δρόμο με ελάχιστες επιβραδύνσεις, λόγω της κυκλοφοριακής συμφόρησης. Η μέση ταχύτητα με την οποία κινούνταν το όχημα στο μεγαλύτερο διάστημα της διαδρομής σύμφωνα με την χαρτογράφηση της, ήταν στα 83.37 km/h και η μέγιστη ταχύτητα στα 117 km/h. Παρατηρείται ότι με την επαλήθευση του Dashboard η κατανάλωση καυσίμου δεν υπερβαίνει τα 5.33 l/100km σε απόσταση των σχεδόν 11 km που διανύθηκαν. Επιπλέον, η ενέργεια προώθησης για την κίνηση του οχήματος υπολογίστηκε στις 1.80 kWh, η ενέργεια φρεναρίσματος στις -0.22 kWh και η καθαρή ενέργεια στις 1.58 kWh, παρατηρώντας πως οι τιμές του OBD με το GPS είναι πανομοιότυπες για αυτήν την μέτρηση. Παρόλα αυτά, η συνολική ενέργεια που καταναλώθηκε σε όλη την καταγεγραμμένη διαδρομή ήταν 155.66 Wh/km και ο συντελεστής αποδοτικότητας στο 0.32. Επιπροσθέτως, η μέτρηση εκπομπής διοξειδίου του άνθρακα στην συγκεκριμένη διαδρομή ήταν 141.07 gCO2/km.



Εικόνα 5.1: Χαρτογράφηση της Κανονικής Υπεραστικής διαδρομής.



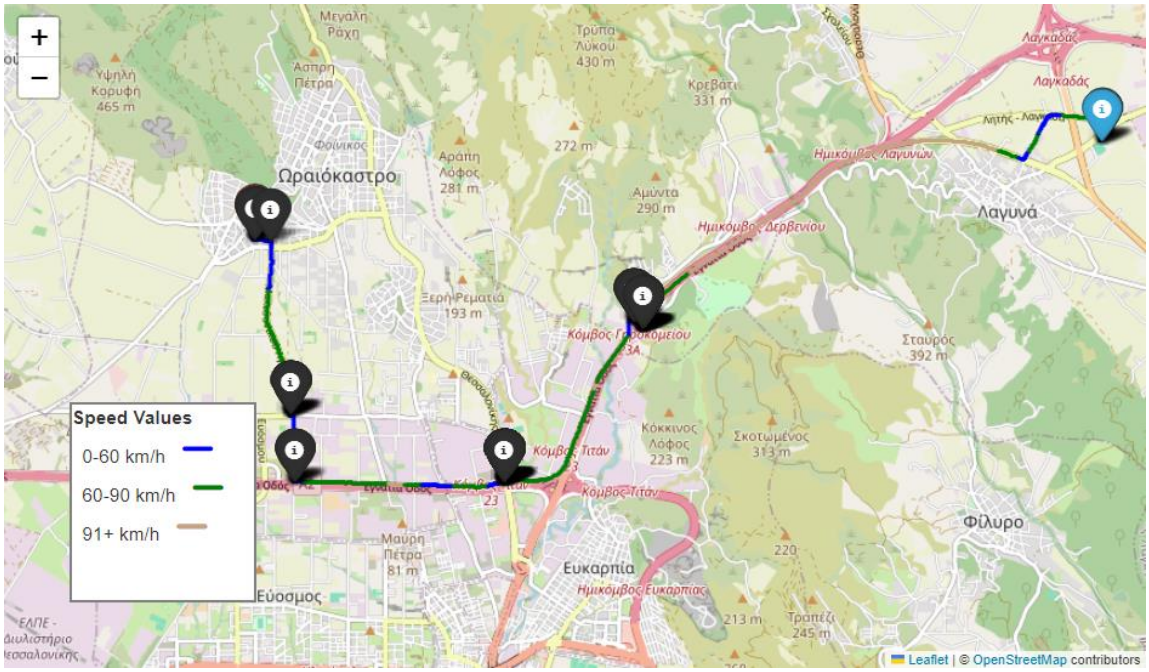
Εικόνα 5.2: Γράφημα της μέτρησης Κανονικής Υπεραστικής.

	Dashboard	OBD	GPS
Distance [km]	10.90	10.13	10.40
Average speed [km/h]	79.60	83.37	79.17
Fuel consumption [l/100km]	5.30	5.33	-
Max speed [km/h]	-	117.00	116.89
Max acceleration [m/s²]	-	2.08	2.75
Max deceleration [m/s²]	-	-2.22	-2.29
Propulsion energy [kWh]	-	1.76	1.80
Brake energy [kWh]	-	-0.18	-0.22
Net energy [kWh]	-	1.58	1.58
Prop. energy/dist. [Wh/km]	-	155.66	151.70
Efficiency factor [-]	-	0.32	-
CO2 emissions [gCO2/km]	-	141.07	-

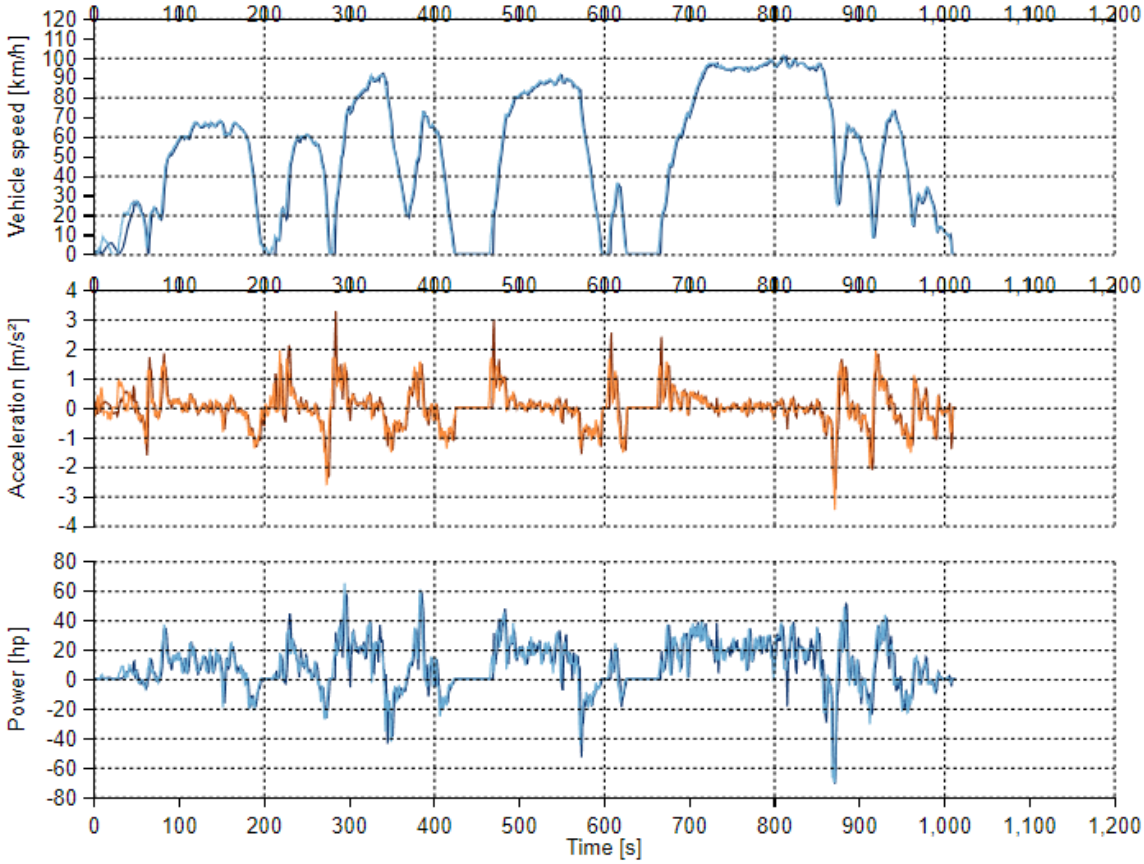
Εικόνα 5.3: Πίνακας μετρήσεων Κανονικής Υπεραστικής.

5.2 Υπεραστική διαδρομή με έλεγχο ταχύτητας (Cruise Control)

Η καταγραφή αυτή έγινε στην εξωτερική περιφερειακή οδό της Θεσσαλονίκης, χρησιμοποιώντας το σύστημα του ελέγχου ταχύτητας (Cruise Control). Αυτό το σύστημα υπάρχει σε διάφορα αυτοκίνητα και χρησιμοποιείται για να διατηρεί σταθερή ταχύτητα του οχήματος, επιτρέποντας τον οδηγό να μην πατάει το πεντάλ της επιτάχυνσης για να κινηθεί το όχημα. Η καταγραφή περιλαμβάνει και τμήματα εκτός της περιφερειακής οδού στα οποία δεν ήταν δυνατό να ενεργοποιηθεί ο έλεγχος ταχύτητας. Στην συγκεκριμένη καταγραφή το σύστημα ελέγχου ταχύτητας χρησιμοποιήθηκε για ένα χρονικό διάστημα και όχι σε ολόκληρη την διαδρομή, διότι δεν υπήρχε η διαθεσιμότητα του περιφερειακού δρόμου για μεγάλο διάνυσμα. Η μέγιστη ταχύτητα του αυτοκινήτου ήταν στα 101 km/h και η μέση ταχύτητα στα 50.82 km/h. Σύμφωνα με τα εργαλεία επαλήθευσης η κατανάλωση του καυσίμου δεν ξεπέρασε τα 5.03 l/100km στην απόσταση των 14.40 km που διανύθηκε. Επιπροσθέτως, η ενέργεια πρόωσης για την κίνηση του οχήματος που υπολογίστηκε ήταν 2.31 kWh, η ενέργεια πέδησης στις -0.49 kWh και η καθαρή ενέργεια στις 1.83kWh. Έπειτα η ειδική κατανάλωση της συνολικής ενέργειας που υπολογίστηκε για την διαδρομή, ήταν 127.97 Wh/km = 12.8 Wh/100km και συντελεστής αποδοτικότητας στο 0.34. Αξίζει να σημειωθεί ότι η εκπομπές διοξειδίου του άνθρακα για αυτήν την διαδρομή ήταν αρκετά χαμηλότερες σε σύγκριση με την Κανονική Υπεραστική. Παρά την επιπλέον απόσταση των 3.40 km, καταγράφηκαν 125.32 gCO2/km, ενώ στην Κανονική Υπεραστική για απόσταση 11 km, καταγράφηκαν 141.07 gCO2/km.



Εικόνα 5.4: Χαρτογράφηση της Υπεραστικής με έλεγχο ταχύτητας διαδρομής.



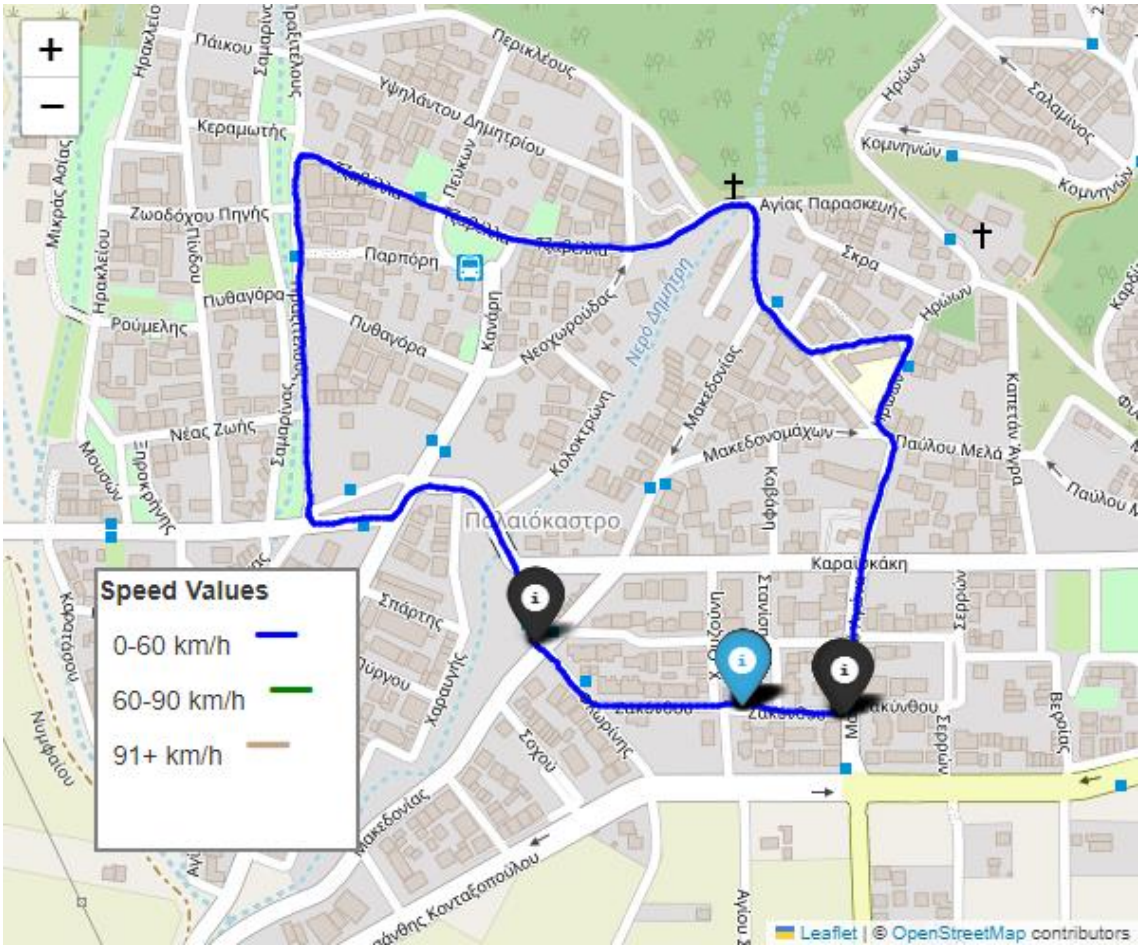
Εικόνα 5.5: Γράφημα της μέτρησης Υπεραστική με έλεγχο ταχύτητας.

	Dashboard	GPS	OBD
Distance [km]	14.40	14.10	14.29
Average speed [km/h]	50.42	50.02	50.82
Fuel consumption [l/100km]	5.03	-	4.74
Max speed [km/h]	-	100.44	101.00
Max acceleration [m/s²]	-	3.28	1.94
Max deceleration [m/s²]	-	-2.78	-3.47
Propulsion energy [kWh]	-	2.29	2.31
Brake energy [kWh]	-	-0.49	-0.48
Net energy [kWh]	-	1.80	1.83
Prop. energy/dist. [Wh/km]	-	127.79	127.97
Efficiency factor [-]	-	-	0.34
CO2 emissions [gCO2/km]	-	-	125.32

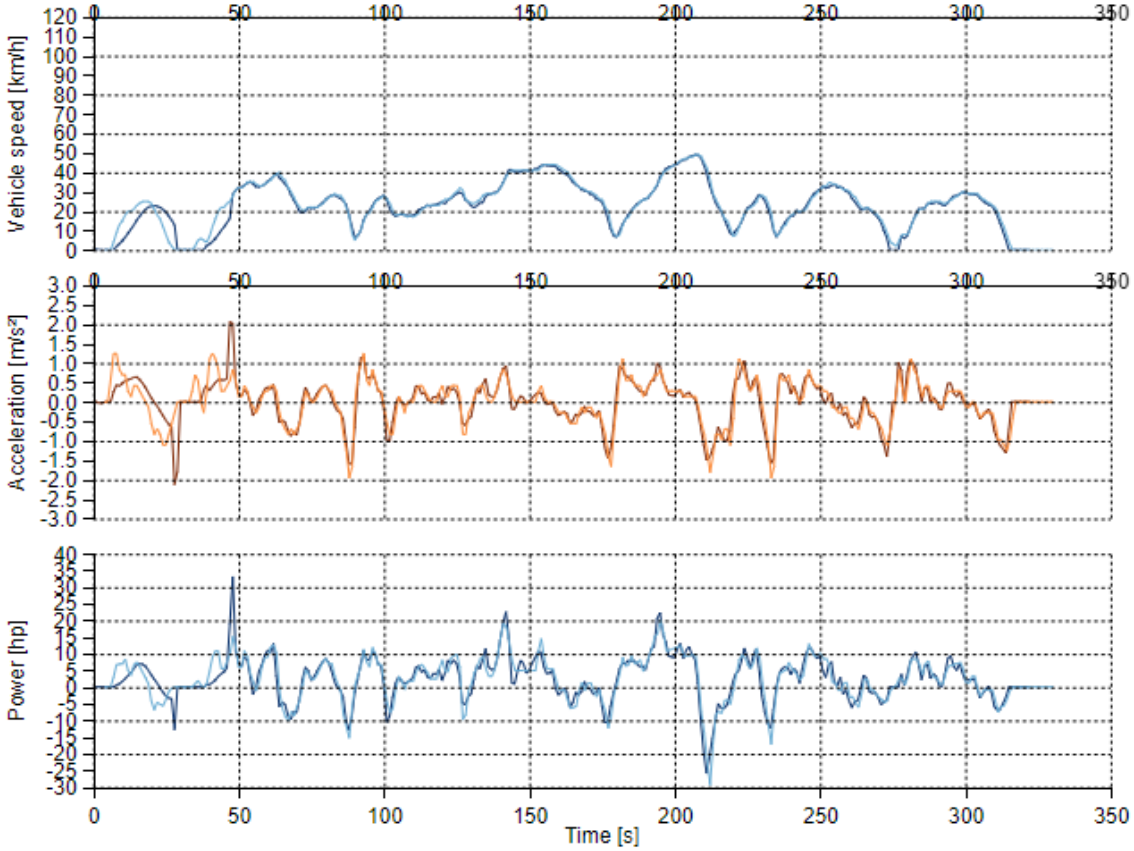
Εικόνα 5.6: Πίνακας μετρήσεων Υπεραστικής με έλεγχος ταχύτητας.

5.3 Κανονική Αστική

Η ακόλουθη καταγραφή πραγματοποιήθηκε σε κατοικημένη περιοχή με πυκνή κυκλοφοριακή συμφόρηση. Η μέση ταχύτητα του αυτοκινήτου για αυτήν την διαδρομή ήταν κοντά στα 23.14 km/h και η μέγιστη ταχύτητα ανήλθε στα 49 km/h. Η απόσταση που καλύφθηκε ήταν 2.10 km και η κατανάλωση καυσίμου υπολογίστηκε στα 4.94 l/100km. Παρατηρείται όμως, ότι η ενέργεια προώθησης στην αστική διαδρομή ήταν στις 0.27 kWh, η ενέργεια φρεναρίσματος στις -0.10 kWh και η καθαρή ενέργεια στις 0.17 kWh. Η υπολογιζόμενη κατανάλωση της συνολικής ενέργειας για την διαδρομή αυτή ήταν 82.31 Wh/km και ο συντελεστής αποδοτικότητας στο 0.27. Βασική παρατήρηση για την παρούσα διαδρομή, είναι πως η εκπομπή διοξειδίου του άνθρακα καταγράφηκε στα 125.45 gCO2/km.



Εικόνα 5.7: Χαρτογράφηση της Κανονικής Αστικής διαδρομής.



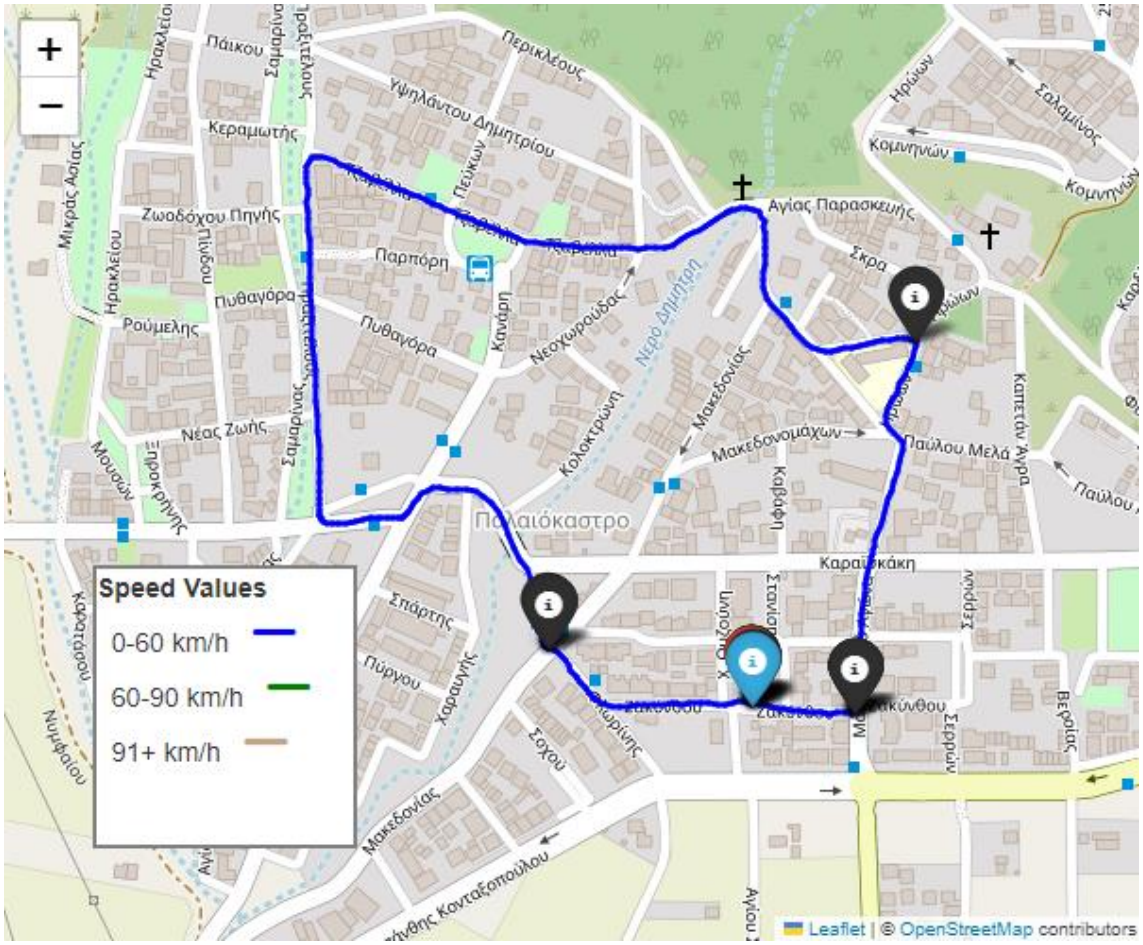
Εικόνα 5.8: Γράφημα της μέτρησης Κανονικής Αστικής.

	Dashboard	GPS	OBD
Distance [km]	2.10	2.03	2.10
Average speed [km/h]	20.41	22.67	23.14
Fuel consumption [l/100km]	4.94	-	4.74
Max speed [km/h]	-	49.00	49.00
Max acceleration [m/s²]	-	2.05	1.25
Max deceleration [m/s²]	-	-2.12	-1.94
Propulsion energy [kWh]	-	0.26	0.27
Brake energy [kWh]	-	-0.09	-0.10
Net energy [kWh]	-	0.17	0.17
Prop. energy/dist. [Wh/km]	-	82.31	82.44
Efficiency factor [-]	-	-	0.27
CO2 emissions [gCO2/km]	-	-	125.43

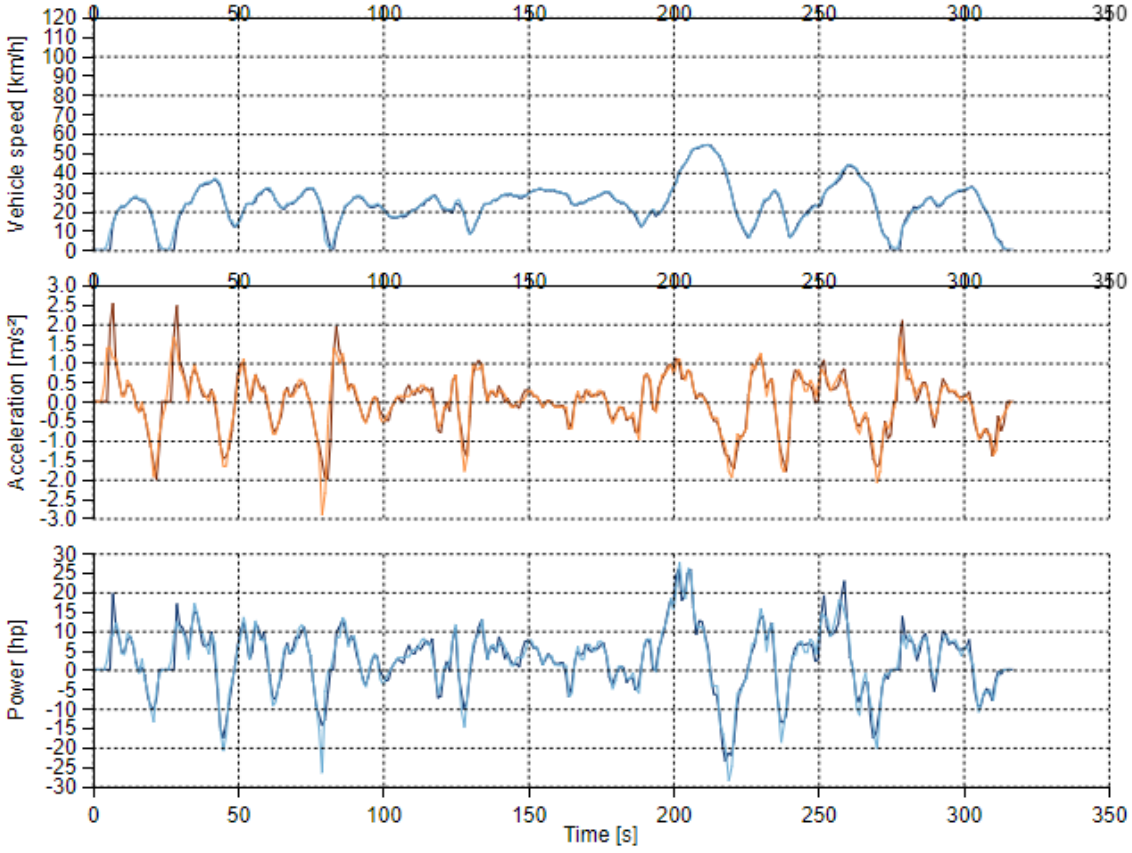
Εικόνα 5.9: Πίνακας μετρήσεων Κανονικής Αστικής.

5.4 Αστική με χρήση κλιματισμού (Air Condition)

Η συγκεκριμένη καταγραφή πραγματοποιήθηκε ακολουθώντας την πανομοιότυπη διαδρομή που καταγράφηκε στην “Κανονική Αστική”, με την μοναδική διαφορά ότι σε αυτήν την διαδρομή χρησιμοποιήθηκε ο κλιματισμός του αυτοκινήτου καθ’ όλη την διάρκεια της καταγραφής της. Η μέση ταχύτητα του αυτοκινήτου υπολογίστηκε στα 23.83 km/h με μέγιστη ταχύτητα να υψώνεται στα 54.00 km/h. Η Απόσταση που διανύθηκε ήταν εξίσου στα 2.10 km, όπως καταγράφηκαν και στην διαδρομή της “Κανονικής Αστικής” και έπειτα η συνολική κατανάλωση καυσίμου ανήλθε στα 7.00 l/100km. Επιπλέον, η ενέργεια προώθησης καταγράφηκε στις 0.30 kWh, καθώς η ενέργεια φρεναρίσματος στις -0.13 kWh και η καθαρή ενέργεια στις 0.17 kWh. Παράλληλα, η κατανάλωση της συνολικής ενέργειας για αυτήν την διαδρομή υπολογίστηκε στις 82.19 Wh/km με συντελεστή αποδοτικότητας στο 0.21. Σε αντίθεση με την “Κανονική Αστική” διαδρομή, η εκπομπή διοξειδίου του άνθρακα αυξήθηκε δραστικά, φτάνοντας στα 181.82 gCO2/km, λόγω της συνεχόμενης χρήσης του κλιματισμού, διότι για την άμεση λειτουργία του απαιτείται παραπάνω χρήση του καυσίμου, εκπέμποντας πολύ περισσότερους ρύπους στην ατμόσφαιρα.



Εικόνα 5.10: Χαρτογράφηση της Αστικής με χρήση κλιματισμού διαδρομής.



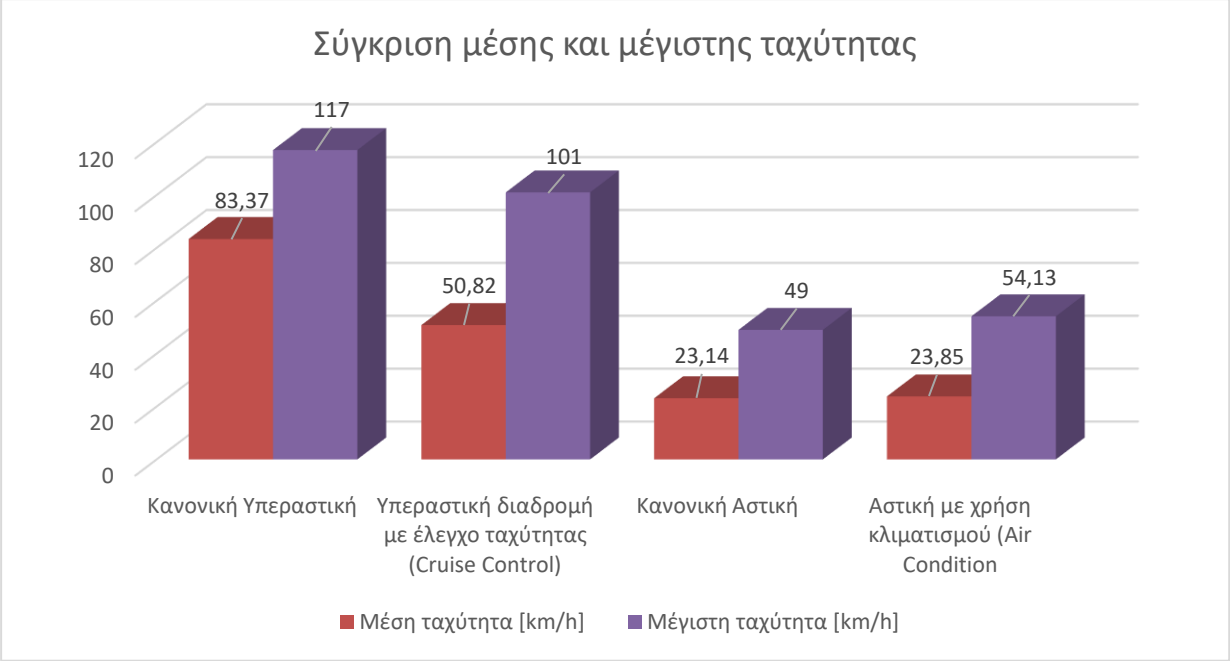
Εικόνα 5.11: Γράφημα της μέτρησης Αστική με χρήση κλιματισμού

	Dashboard	GPS	OBD
Distance [km]	2.10	2.06	2.10
Average speed [km/h]	20.20	23.35	23.85
Fuel consumption [l/100km]	7.00	-	6.87
Max speed [km/h]	-	54.13	54.00
Max acceleration [m/s²]	-	2.54	1.67
Max deceleration [m/s²]	-	-2.02	-2.92
Propulsion energy [kWh]	-	0.30	0.30
Brake energy [kWh]	-	-0.13	-0.13
Net energy [kWh]	-	0.17	0.17
Prop. energy/dist. [Wh/km]	-	82.04	82.19
Efficiency factor [-]	-	-	0.21
CO2 emissions [gCO2/km]	-	-	181.82

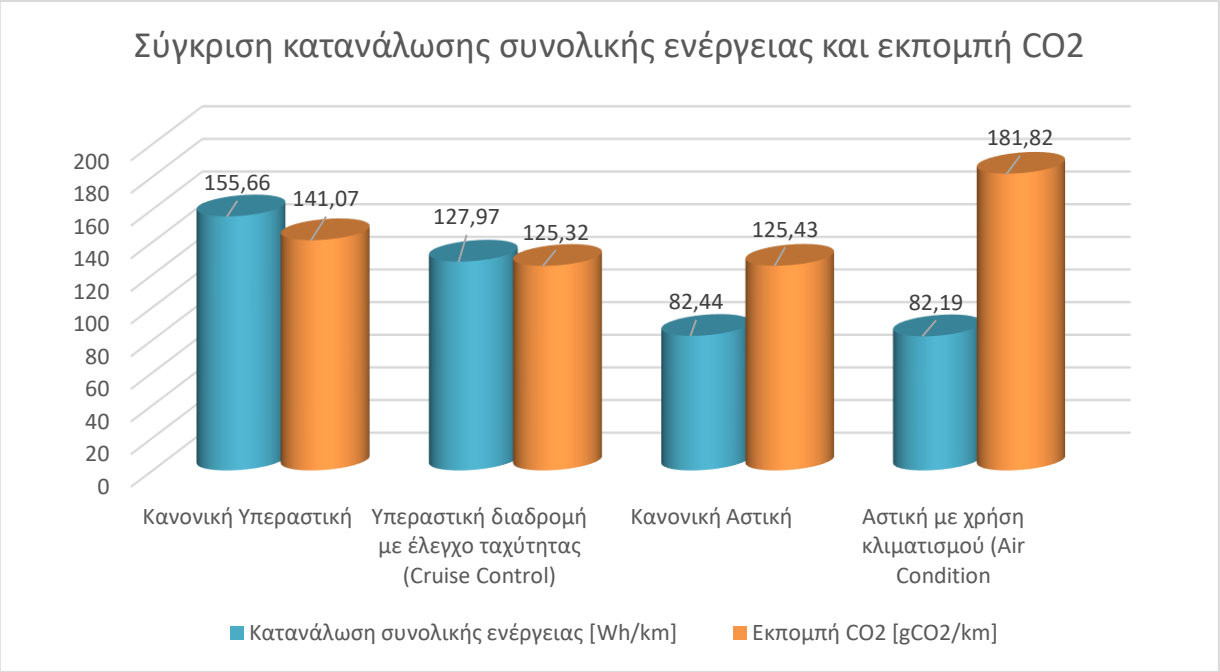
Εικόνα 5.12: Πίνακας μετρήσεων Αστικής με χρήση κλιματισμού

5.5 Σύγκριση καταγραφών

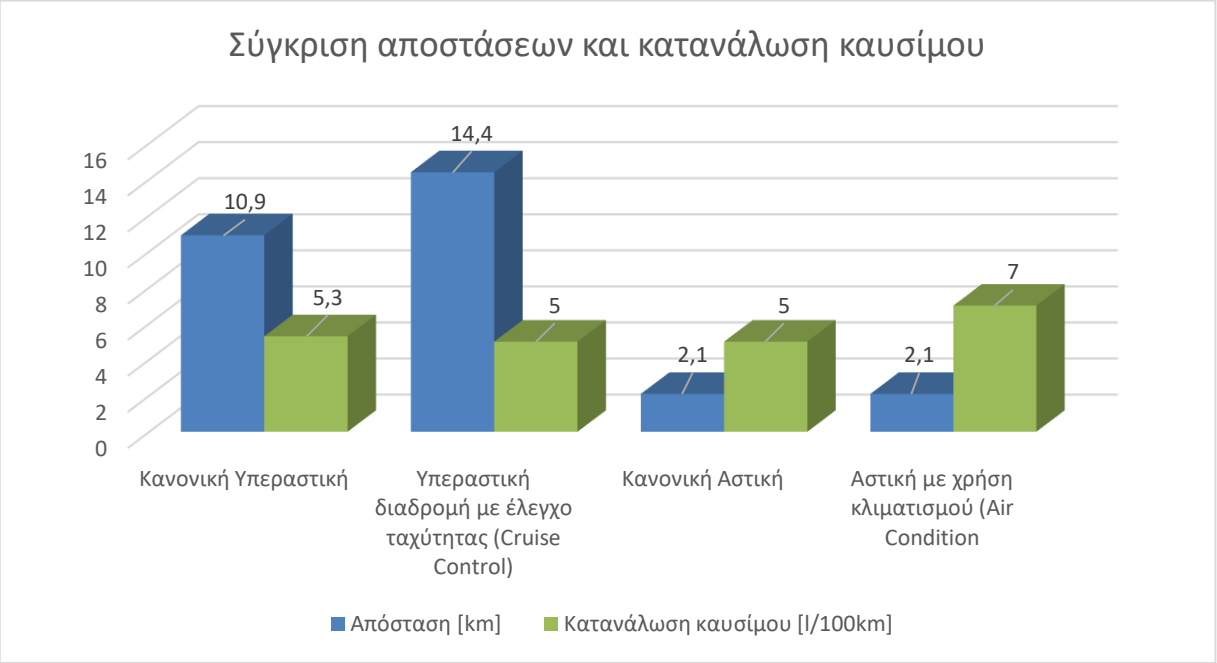
Στα παρακάτω γραφήματα αναφέρονται όλες οι μετρήσεις των καταγραφών που πραγματοποιήθηκαν για αυτήν την εργασία. Αρχικά, παρατηρείται πως η υψηλότερη εκπομπή του CO2 υπάρχει στην καταγραφή της Αστικής με χρήση κλιματισμού. Αυτό συμβαίνει διότι το αυτοκίνητο καταναλώνει περισσότερο καύσιμο για την λειτουργία του κλιματιστικού, καθώς προσθέτει επιπλέον φορτίο στον κινητήρα, σε αντίθεση με τις υπόλοιπες καταγραφές που δεν λειτούργησε το κλιματιστικό κατά την διάρκεια της μέτρησης. Έπειτα, η υψηλότερη κατανάλωση ενέργειας εντοπίστηκε στην καταγραφή της Κανονικής Υπεραστικής. Αυτό οφείλεται στο γεγονός ότι ο κινητήρας διατηρείται σε υψηλές στροφές, με αποτέλεσμα να χρησιμοποιηθεί περισσότερη ενέργεια για τη διατήρηση της ταχύτητας του οχήματος. Ωστόσο, σύμφωνα με το γράφημα, γίνεται αντιληπτό πως το όχημα είχε αρκετά υψηλότερη μέγιστη και μέση ταχύτητα σε σχέση με τις υπόλοιπες καταγραφές, επαληθεύοντας την συνολική κατανάλωση ενέργειας που χρησιμοποιήθηκε για την καταγραφή.



Εικόνα 5.13: Διάγραμμα σύγκρισης μέσης και μέγιστης ταχύτητας.



Εικόνα 5.14: Διάγραμμα σύγκρισης κατανάλωσης συνολικής ενέργειας και εκπομπή CO2.



Εικόνα 5.15: Διάγραμμα σύγκρισης απόστασης.

6 Συμπεράσματα

Συνοψίζοντας, η πτυχιακή εργασία αποτελεί την ανάπτυξη ενός υπάρχοντος κώδικα που μελετά βιώσιμες μετακινήσεις οχημάτων, με επίκεντρο την βελτίωση παρουσίασης των αποτελεσμάτων και την απεικόνιση τους σε χάρτη. Αναλύει σε βάθος την ενεργειακή απόδοση και κατανάλωση του οχήματος, καθώς και τους ρύπους που εκβάλλονται στην ατμόσφαιρα. Έπειτα από αρκετές δοκιμές και μετρήσεις, τυποποιήθηκε η διαδικασία ανάλυσης και πραγματοποιήθηκαν ρυθμίσεις και επαληθεύσεις στο μοντέλο του κώδικα με μετρήσεις κατανάλωσης καυσίμου. Στην συνέχεια, από τις καταγραφές των διαδρομών που διεξάχθηκαν και την εισαγωγή των δεδομένων τους, διαπιστώθηκε πως η χρήση του κλιματισμού αυξάνει ραγδαία το ποσοστό κατανάλωσης καυσίμου και τις εκπομπές CO2. Αξίζει να σημειωθεί πως καθοριστικό ρόλο διαδραματίζει η συμπεριφορά του οδηγού στον δρόμο και όχι το εύρος χιλιομέτρων που καλύπτει ένα όχημα σε μία διαδρομή, καθώς οι μικρές μετακινήσεις στην πόλη σε σύγκριση με τα μακρινά ταξίδια, αναλογούν ισάξια στην ρύπανση του περιβάλλοντος.

Η παρούσα πτυχιακή εργασία είναι μια συνέχεια της αρχικής δομής και παρέχει αρκετά περιθώρια ανάπτυξης. Συγκεκριμένα, θα μπορούσαν να πραγματοποιηθούν επιπλέον μετρήσεις με διαφορετικό όχημα σε διαφορετική πόλη, και να προστεθεί χαρακτηρισμός της διαδρομής αν πραγματοποιείται σε αυτοκινητόδρομο, σε κανονικό η επικλινής δρόμο, με την χρήση ενός βαρομετρικού αισθητήρα σε κινητό τηλέφωνο. Τέλος, σημαντική προσθήκη θα ήταν η καταγραφή διαδρομών σε διαφορετικές καιρικές και κλιματικές συνθήκες, για την εξέταση της ενεργειακής κατανάλωσης σε σύγκριση με των προαναφερόμενων παραδειγμάτων.

Βιβλιογραφία

[1] Emma Terama, “What is the Green Transition?”, Ymparistoministerio Mijoministeriet Ministry of the Environment, [10/05/2024].

[2] Travis Madsen, Matt Frommer, “Transportation and land use planning equity in Colorado”, SWEEP SOUTHWEST ENERGY EFFICIENCY PROJECT, 2022.

[3] Anastasios Tsakalidis, Konstantinos Gkoumas, Ferenc Pekar, “Digital Transformation Supporting Transport Decarbonization: Technological Developments in EU-Funded Research and Innovation” MDPI, pp 1–5, 2020.

[4] Directorate-General for Climate Action, “First Commission report on real-world CO2 emissions of cars and vans using data from on-board fuel consumption monitoring devices”, European Commission, 2024.

[5] Γιώργος Αγαπητός, “Test Drive: Δοκιμάζουμε το Renault Clio Sport Tourer dCi 90”, autoblog, 2015.

[6] Dan Mihalascu, “Renault Clio Gets Iconic Special Edition In France”, CARSCOOPS, 2015.

[7] Kevin Flesser etc., “How to install Ubuntu on the HP Stream 13”, IFIXIT, 2020, [29/5/2024].

[8] Geotab Team, “What is OBDII? History of on-board diagnostics”, GEOTAB, 2023, [1/6/2024].

[9] Aramox, “Red OBD Diagnostic Cable, French English Replacement, Clio Twingo ECU Programmer V1.87”, Amazon, [5/6/2024].

[10] Carwow staff, “What is a car trip computer?”, carwow, 2023.

[11] https://www.code.visualstudio.com/docs - Visual Studio Documentation, [14/6/2024].

[12] Basic Air Data, “BasicAirData GPS Logger – Getting started guide”, [14/6/2024].

[13] Saurav Sharma, “Why does everyone want to learn Python?”, Quora, 2020.

Παράρτημα κώδικα (PYTHON)

Recordobd.py

```
import obd
import time
import serial
import csv
from datetime import datetime

# obd.logger.setLevel(obd.logging.DEBUG)  # enable console printing of all debug mes-
sages

connection = obd.Async(fast=False)  # auto-connects to USB or RF port

rpm_value = 0
speed_value = 0
coolant_value = 0
maf_value = 0
intake_pressure_value = 0
throttle_position_value = 0
load_value = 0  # Added for engine load

# RPM
def new_rpm(r):
    global rpm_value
    rpm_value = r.value.magnitude
    print("\t")

# SPEED
def new_speed(s):
    global speed_value
    speed_value = s.value.magnitude
    print("\t")

# COOLANT TEMPERATURE
def new_coolant_temperature(t):
    global coolant_value
    coolant_value = t.value.magnitude
    print("\t")

# MAF AIR FLOW RATE
def new_maf(mf):
    global maf_value
    maf_value = mf.value.magnitude
    print("\t")

# INTAKE PRESSURE
def new_intake_pressure(ip):
    global intake_pressure_value
    intake_pressure_value = ip.value.magnitude
    print("\t")

# THROTTLE POSITION
```

```
def new_throttle_position(tp):
    global throttle_position_value
    throttle_position_value = tp.value.magnitude
    print("\t")

# ENGINE LOAD
def new_load(load):
    global load_value
    load_value = load.value.magnitude
    print("\t")

# Add OBD-II commands
connection.watch(obd.commands.RPM, callback=new_rpm)
connection.watch(obd.commands.SPEED, callback=new_speed)
connection.watch(obd.commands.COOLANT_TEMP, callback=new_coolant_temperature)
connection.watch(obd.commands.MAF, callback=new_maf)
connection.watch(obd.commands.INTAKE_PRESSURE, callback=new_intake_pressure)
connection.watch(obd.commands.THROTTLE_POS, callback=new_throttle_position)
connection.watch(obd.commands.ENGINE_LOAD, callback=new_load)

connection.start()  # the callback now will be enabled for new values
starttime = time.time()
date_format = "%d_%m_%Y"
formatted_date = datetime.fromtimestamp(starttime).strftime(date_format)

mylist = ["TIME[s]", "SPEED[km/h]", "TIMESTAMP", "LOAD[%]", "RPM[rpm]", "COOLANT[degC]",
"MAF[g/s]", "INTAKE_PRESSURE", "THROTTLE_POSITION[%]"]
file_timestamp = int(time.time())
file_name = f'obd_timestamp{file_timestamp}.csv'

with open(file_name, 'w') as f:
    f.write(','.join(mylist))  # Write header
    f.write('\n')
    try:
        while True:
            timecount = time.time() - starttime
            timestamp = int(time.time())  # Convert datetime to timestamp
            print("File Write:", "{:.4f}".format(timecount), speed_value, timestamp,
load_value, rpm_value, coolant_value, maf_value,
                  intake_pressure_value, throttle_position_value)
            f.write(

f"{timecount:.2f},{speed_value},{timestamp},{load_value:.2f},{rpm_value},{cool-
ant_value},{maf_value:.2f},{intake_pressure_value},{throttle_position_value:.2f}\n")
            time.sleep(1.0)
    except KeyboardInterrupt:
        pass

connection.stop()
```

Gpx_to_csv.py

```
import xml.etree.ElementTree as ET
import csv
import pandas as pd
from datetime import datetime, timezone
import folium
```

```
import pytz

def readgpx(gpx_file_path: str) -> pd.DataFrame:
    # Parse the GPX file
    tree = ET.parse(gpx_file_path)
    root = tree.getroot()
    creator = root.attrib['creator']
    #print(f"Creator: {creator}")

    # Create CSV header
    csv_header = ["TIME[s]", "SPEED[km/h]", "TIMESTAMP", "LATITUDE", "LONGITUDE",
"ALTITUDE"]
    gpxdata = pd.DataFrame(columns=csv_header)

    # Write data to CSV file
    coordinates = []  # List to store coordinates for PolyLine
    ns = {'gpx': 'http://www.topografix.com/GPX/1/1', 'osmand': 'https://osmand.net'}
    nsZepp = {'gpx': 'http://www.topografix.com/GPX/1/1', 'ns3':
'http://www.garmin.com/xmlschemas/TrackPointExtension/v1'}
    time_offset = None

    for trkpt in root.findall('.//gpx:trkpt', namespaces=ns):
        dtime_str = trkpt.find('gpx:time', namespaces=ns).text
        pdatetime = datetime.strptime(dtime_str, "%Y-%m-
%dT%H:%M:%SZ").replace(tzinfo=pytz.UTC)
        if time_offset is None:
            time_offset = pdatetime.timestamp()
        time = pdatetime.timestamp() - time_offset
        latitude = float(trkpt.get('lat'))
        longitude = float(trkpt.get('lon'))
        altitude = float(trkpt.find('gpx:ele', namespaces=ns).text)

        # Extract speed value from extensions
        match creator:
            case "osmand" | "OsmAnd 4.5.10" | "OsmAnd 4.4.6":
                speed_element = trkpt.find('gpx:extensions/osmand:speed', namespaces=ns)
            case "ZeppLife App":
                speed_element = trkpt.find('gpx:extensions/ns3:TrackPointExten-
sion/ns3:speed', namespaces=nsZepp)
            case 'BasicAirData GPS Logger 3.2.2':
                speed_element = "0.0"
            case _:
                raise Exception('gpx file creator not supported.')

        if speed_element is None:
            continue
        else:
            speed_value = float(speed_element.text) * 3.6

        gpxdata.loc[len(gpxdata)] = [time, speed_value, pdatetime.timestamp(), latitude,
longitude, altitude]

    return gpxdata

def determine_color(speed):
    if 0 <= speed <= 60:
        return 'blue'
    elif 60 < speed <= 90:
        return 'green'
    elif speed > 90:
```

```
        return '#C4A484'

def createMap(gpxdata):
    latitude = list(gpxdata['LATITUDE'])
    longitude = list(gpxdata['LONGITUDE'])
    speed = list(map(float, gpxdata['SPEED[km/h]']))

    coordinates = list()

    # Create a map centered around the first set of coordinates
    my_map = folium.Map(location=[latitude[-1], longitude[-1]], zoom_start=15)

    for i in range(len(latitude)):
        coordinates.append([latitude[i], longitude[i]])

        # Determine color based on speed ranges
        color = determine_color(speed[i])

        # Add a spot whenever the speed is "0"
        if speed[i] == 0:
            folium.Marker([latitude[i], longitude[i]], popup=f"Speed: {speed[i]:.2f}
km/h",
                          icon=folium.Icon(color='black')).add_to(my_map)

        # Add a colored PolyLine based on speed
        if i > 0:  # Skip the first point to avoid issues with PolyLine
            folium.PolyLine(locations=[coordinates[i - 1], coordinates[i]], color=color,
weight=3).add_to(my_map)

    # Add a red pin at the beginning of the route
    folium.Marker([latitude[0], longitude[0]], popup="Start", icon=fo-
lium.Icon(color='red')).add_to(my_map)

    # Add a blue pin at the end of the route
    folium.Marker([latitude[-1], longitude[-1]], popup="End", icon=fo-
lium.Icon(color='blue')).add_to(my_map)

    # Fit the map bounds to include the entire route
    my_map.fit_bounds([[min(latitude), min(longitude)], [max(latitude), max(longi-
tude)]])

    # Add Legend
    legend_html = f"""
    <div style="position: fixed;
                bottom: 50px; left: 50px; width: 140px; height: 150px;
                border:2px solid grey; z-index:9999; font-size:14px;
                background-color:white;
                ">
                <b>Speed Values</b><br>
                  0-60 km/h   <i class="fa fa-minus fa-2x"
style="color:blue"></i><br>
                  60-90 km/h   <i class="fa fa-minus fa-2x"
style="color:green"></i><br>
                  91+ km/h   <i class="fa fa-minus fa-2x"
style="color:#C4A484"></i><br>

    </div>
    """

    my_map.get_root().html.add_child(folium.Element(legend_html))
```

```
    return my_map

if __name__ == "__main__":
    # Specify the GPX file path
    gpx_file_path = "data/stelios/28_01_2024/osmand.gpx"
    converted_csv_file = "Converted_CSV.csv"  # Specify the new filename
    embedded_map_file = "embedded_osmand.html"

    gpxdata = readgpx(gpx_file_path)
    gpxdata.to_csv(converted_csv_file, index=False)

    my_map = createMap(gpxdata)

    # Save the map as an HTML file
    my_map.save(embedded_map_file)

    # Generate the HTML content with an embedded iframe
    with open(embedded_map_file, 'r') as map_file:
        map_content = map_file.read()

    embedded_map_content = f"""
        <html>
        <head>
            <title>Embedded Map</title>
        </head>
        <body>
            <h1>Embedded Map</h1>
            <iframe srcdoc='{map_content}' width='100%' height='600'></iframe>
        </body>
        </html>
        """
    # Save the embedded HTML content
    with open(embedded_map_file, 'w') as embedded_map_file:
        embedded_map_file.write(embedded_map_content)

    print(f"Conversion complete. Data written to {converted_csv_file}. Embedded map
saved to {embedded_map_file.name}.")
```