16 ΙΑΝΟΥΑΡΙΟΥ 2020

ΑΥΤΟΝΟΜΗ ΟΔΗΓΗΣΗ & V2X ΕΠΙΚΟΙΝΩΝΙΑ

ΑΦΕΝΤΑΚΗ ΦΛΩΡΕΝΤΙΑ
AM:1059576

Περιεχόμενα

1.Περιγραφή

Στις μέρες μας η ολοένα αυξανόμενη έρευνα στο πεδίο του αυτοματισμού έχει καταστήσει τα ευφυή αμάξια στο κέντρο των τεχνολογιών αιχμής. Η αυτόματη οδήγηση είναι πλέον ρεαλιστικά υλοποιήσιμο σενάριο το οποίο αυτοκινητοβιομηχανίες όπως Daimler AG, Tesla, Volkswagen , Volvo προάγουν και ερευνούν. Τα στάδια αυτοματισμού ενός οχήματος είναι τα εξής :

0. Κανένας αυτοματισμός : Ο οδηγός έχει τον πλήρη έλεγχο της οδήγησης.
1. Υποβοήθηση οδηγού : Ο οδηγός έχει τον γενικό έλεγχο του αυτοκινήτου ενώ το όχημα αναλαμβάνει την επιτάχυνση / επιβράδυνση του σε συγκεκριμένες περιπτώσεις.
2. Μερική αυτοματοποίηση: Ο οδηγός έχει τον γενικό έλεγχο του αυτοκινήτου ενώ το όχημα αναλαμβάνει αυτόματα την αλλαγή της ταχύτητας του αυτοκινήτου βασιζόμενο στις οδικές συνθήκες.
3. Αυτοματοποίηση υπό όρους : Το όχημα αναλαμβάνει τον έλεγχο του αυτοκινήτου με τον οδηγό να είναι απαραίτητος για την επίβλεψη και την περίπτωση να επέμβει στην οδήγηση του αυτοκινήτου.
4. Υψηλή αυτοματοποίηση : Το όχημα είναι σχεδιασμένο για την πλήρη αυτονομία της οδήγησης ωστόσο ο οδηγός είναι απαραίτητος για να αναλάβει τον έλεγχο σε ειδικές περιπτώσεις.
5. Πλήρης αυτοματοποίηση : Το όχημα είναι πλήρως αυτόνομο με τον οδηγό να μην είναι πλέον απαραίτητος.

Τα στάδια 0-2 ο άνθρωπος είναι υπεύθυνος για την επίβλεψη του περιβάλλοντος της οδήγησης ενώ με τα στάδια 3-5 το αυτοματοποιημένο σύστημα είναι υπεύθυνο για την επίβλεψη του περιβάλλοντος της οδήγησης.

Εικόνα 1 Automotive example ©BBC

Το πρωτόκολλο V2X (Vehicle to Everything) χρησιμοποιείται για να περιγράψει ένα σύνολο από τεχνολογίες επικοινωνίας ανάμεσα στα οχήματα. Πιο συγκεκριμένα το V2X αποτελείται από την επικοινωνία

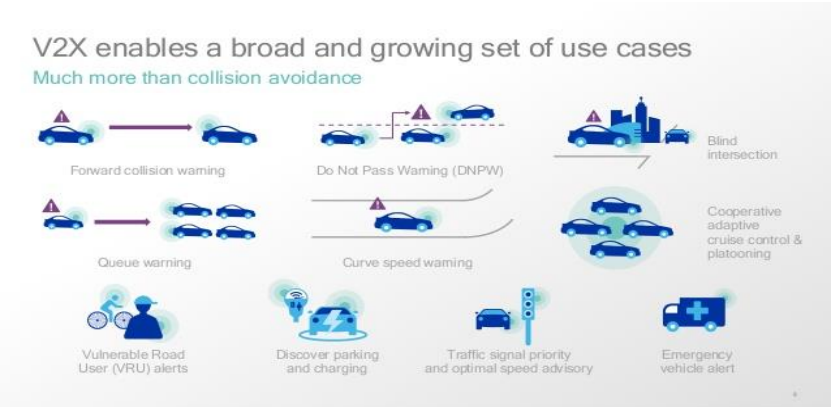
- V2Vehicle όχημα με όχημα
- V2Pedestrians όχημα με πεζούς
- V2Network όχημα με το διαδίκτυο
- V2Infrastructure όχημα με υποδομές

Ουσιαστικά είναι μια Internet Of Things εφαρμογή η οποία έχει στόχο την κατανόηση του περιβάλλοντος από το όχημα καθώς και την λήψη των κατάλληλων αποφάσεων από αυτό. Όλα τα παραπάνω, συμβάλουν όχι μόνο στην διευκόλυνση της εμπειρίας της οδήγησης αλλά και στην πλήρη αυτονομία της.

Η χρήση του V2X υπόσχεται μείωση των ατυχημάτων, της κίνησης και της περιβαλλοντικής μόλυνσης. Μερικά από τα σενάρια στα οποία φαίνεται πιο καθαρά η συμβολή της προαναφερθέντας τεχνολογίας είναι

Το όχημα αναλύει δεδομένα για :

- εμπόδια που θα μας καθυστερήσουν στην επιλεγμένη διαδρομή όπως πχ έργα , ατυχήματα και γίνεται ορθότερη επιλογή διαδρομής από τον πλοηγό.
- την ταχύτητα και την απόσταση που βρίσκεται το προπορευόμενο αυτοκίνητο με αποτέλεσμα να μπορεί να μειώσει ή ακόμη και να ακινητοποιήσει το όχημα μας με ασφάλεια
- τους πεζούς οι οποίοι διασχίζουν τον δρόμο



Εικόνα 2 V2X use cases

2.Απαιτήσεις

Οι απαιτήσεις του συγκεκριμένου συστήματος είναι :

- Ταχύτητα : Το συγκεκριμένο σύστημα απαιτεί πραγματικού χρόνου πληροφόρηση και επεξεργασία. Δηλαδή η χρονική καθυστέρηση που παρεμβάλλεται στο σύστημα μας θέλουμε να είναι όσο το δυνατό μικρότερη και έγκυρη.
- Ασφάλεια : Η διατήρηση της ασφάλειας στην εφαρμογή μας είναι μείζων ζήτημα . Όμως ως εφαρμογή με πολλαπλούς δίαυλους επικοινωνίας η διατήρηση της είναι δυσκολότερη.
- Εγκυρότητα : Η πληροφορία η οποία παραλαμβάνεται και επεξεργάζεται πρέπει να είναι έγκυρη .
- Υπολογιστική Ισχύ : Ο όγκος πληροφορίας που πρέπει να επεξεργαστεί σε μικρό χρονικό διάστημα απαιτεί από το σύστημα μας μεγάλη υπολογιστική ισχύ.
- Ενέργεια : Πρέπει τα επίπεδα της κατανάλωσης ενέργειας να είναι χαμηλά αφού το V2X προορίζεται για όχημα.

3.Αρχιτεκτονική Υλικού

Όπως παραθέτετέ παρακάτω στο Figure 10 η αρχιτεκτονική υλικού που χρησιμοποιούμε ακολουθεί το πρότυπο της ενός ανοιχτού κώδικα έργου από την εταιρία Waymo με όνομα Apollo. Η αρχιτεκτονική αυτή βασίζεται στον υβριδικό υπολογισμό που υλοποιείται μέσω της βοήθειας της υπομονάδας Neousys Nuvo-6108GC και ενός μοντέλου παράλληλου προγραμματισμού CUDA που έχει αναπτυχθεί από την Nvidia και διαθέτει η κάρτα γραφικών που χρησιμοποιούμε στο σύστημα. Το hybrid computing εκμεταλλεύεται την GPU και την χρησιμοποιεί ως συν-επεξεργαστή αποδεσμεύοντας μερικούς από τους υπολογισμούς που κανονικά θα γινόντουσαν στον επεξεργαστή επιταχύνοντας έτσι την επεξεργασία δεδομένων.

Ακόμη όπως βλέπουμε παρακάτω για τον εντοπισμό χρησιμοποιούμε NovAtel Differential GPS SPAN-IGM A1 το οποίο όχι μόνο εντοπίζει την τοποθεσία του αυτοκινήτου στο χάρτη μέσω του GPS αλλά ακόμη συνδυάζει την Αδρανειακής μονάδα μέτρησης [Inertial Measurement Unit (IMU)]

Για την χαρτογράφηση του εδάφους στο οποίο βρίσκεται το όχημα χρησιμοποιούμε 4 είδη αισθητήρων και συγκεκριμένα μεγάλης απόστασης ραντάρ, κάμερες, υπερηχητικούς αισθητήρες καθώς και έναν 360° LiDAR στην οροφή του οχήματος όπως παρουσιάζεται στο Figure 11 .

Πρέπει να σημειωθεί ότι για την ακρίβεια των υπολογισμών είναι αναγκαία η σύνδεση του gps αποδέκτη μας με τον lidar αισθητήρα με σκοπό τον συγχρονισμό και τον υπολογισμό της χρονοσήμανσης (timestamp) ή αλλιώς Top-Of-the-Hour.

Τέλος η παρακάτω σύνθεση έγινε με γνώμονα τον προαναφερθέντα Apollo project καθώς και τις επιμέρους κατασκευαστικές ώστε να επικυρώνεται η συμβατότητα ως προς το υλικό του όλου συστήματος.

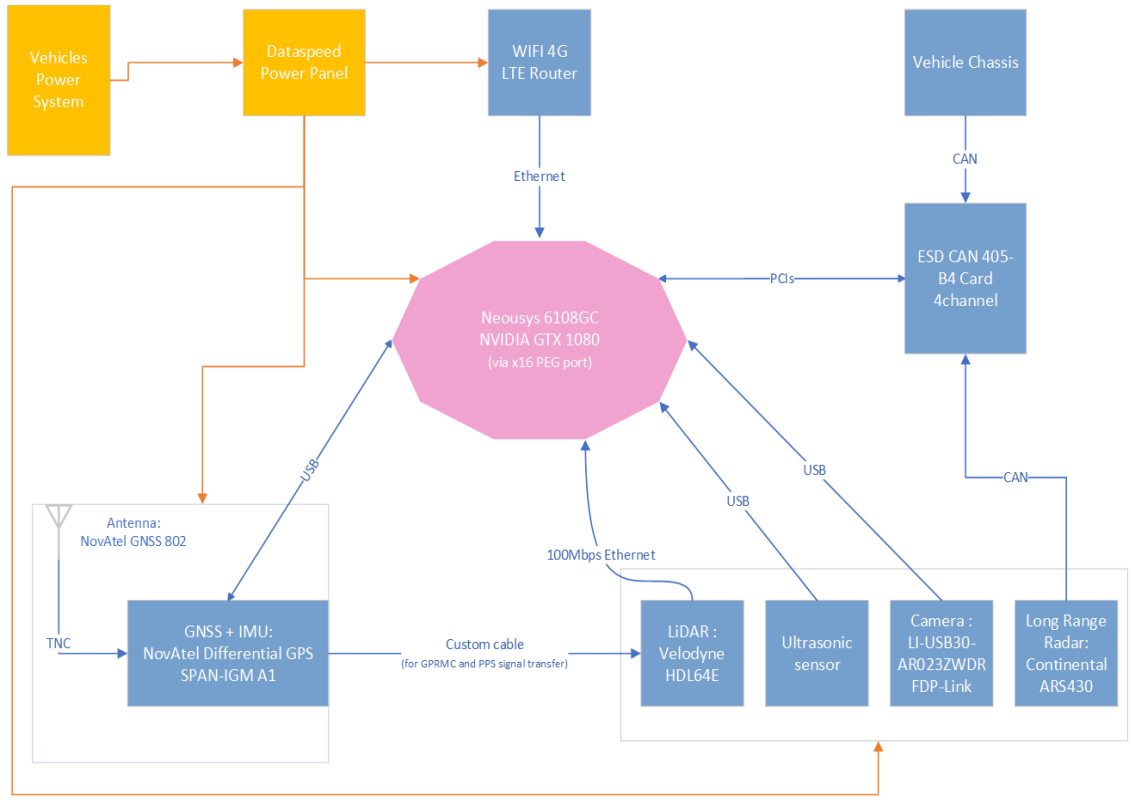


Figure 1 Hardware Architecture

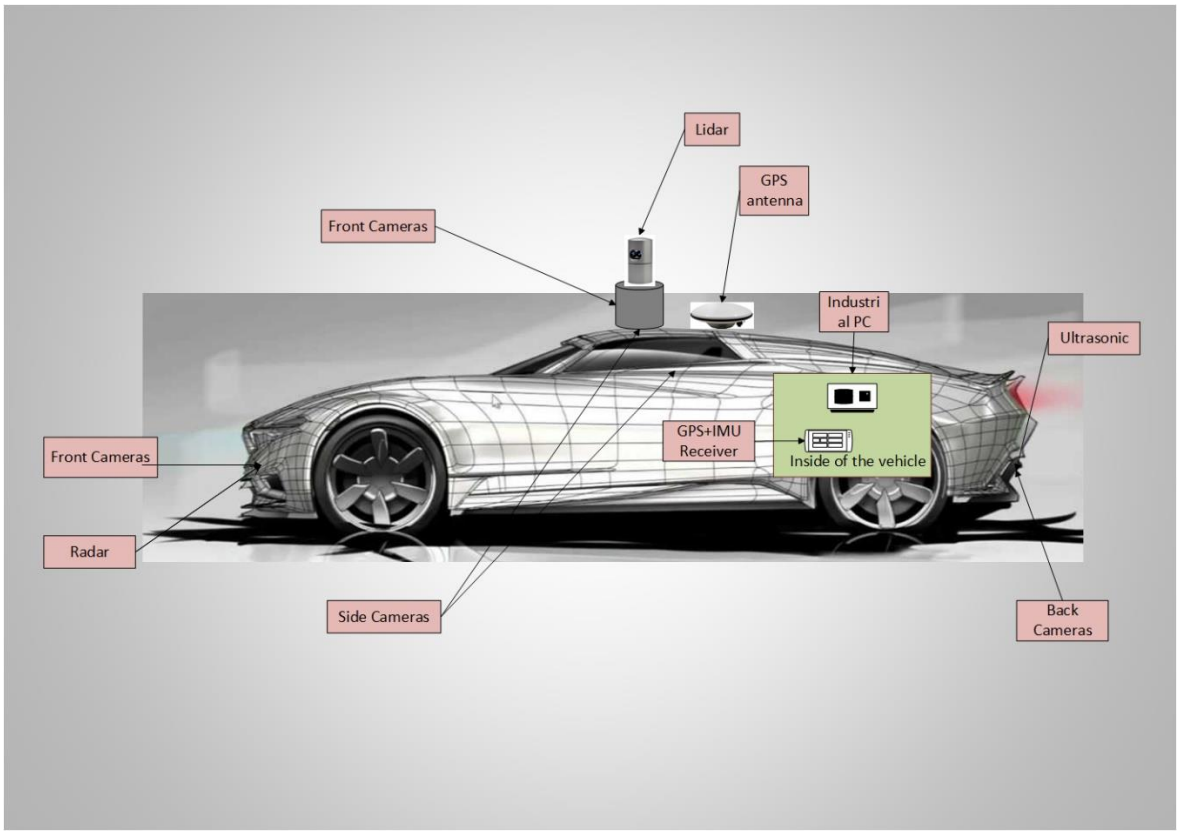


Figure 2 Vehicle Overview

4.Αρχιτεκτονική Λογισμικού

Όπως θα δείτε παρακάτω χρησιμοποιείται το πρότυπο της AUTOSAR (AUTomotive System ARchitecture) στην σχεδίαση του λογισμικού . Στην αρχιτεκτονική υλικού στοχεύουμε στην μεγαλύτερη ανεξαρτησία υλικού – λογισμικού .

Μπορούμε να διακρίνουμε 3 βασικά επίπεδα μετά τον μικροελεκτή :

1. Βασικό λογισμικό
2. Περιβάλλον Λειτουργίας
3. Επίπεδο Εφαρμογών

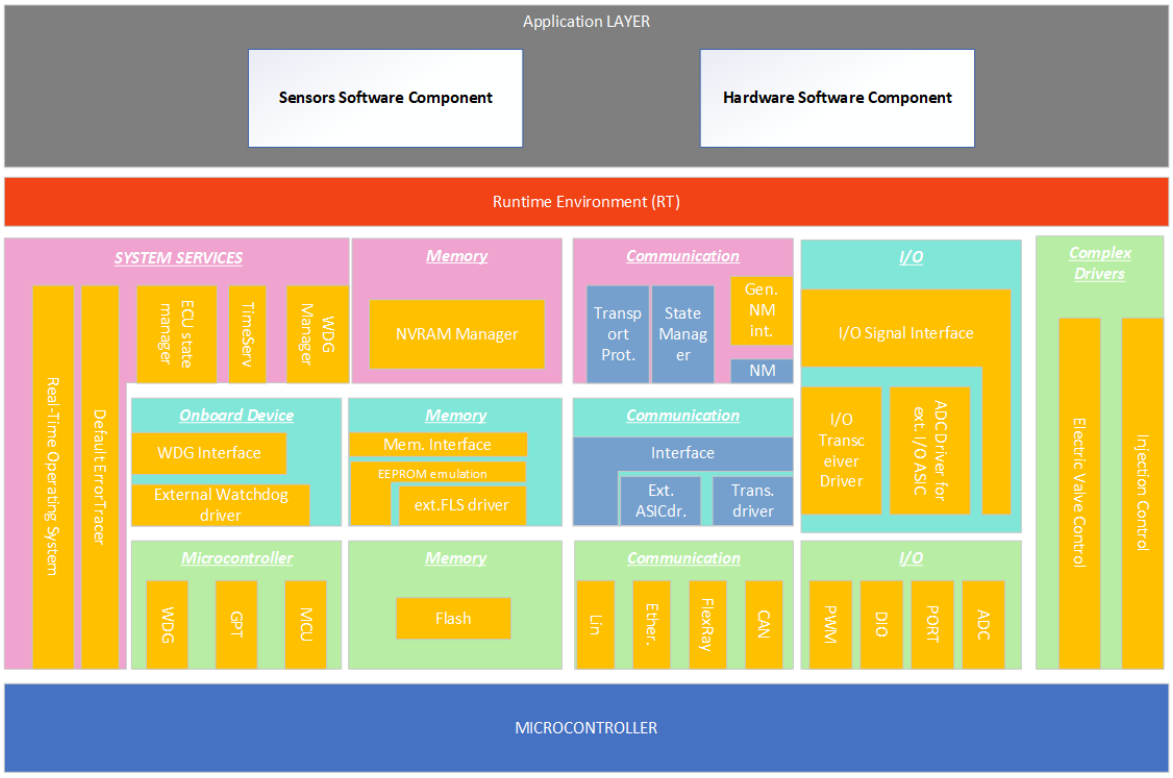


Figure 3 Software Architecture

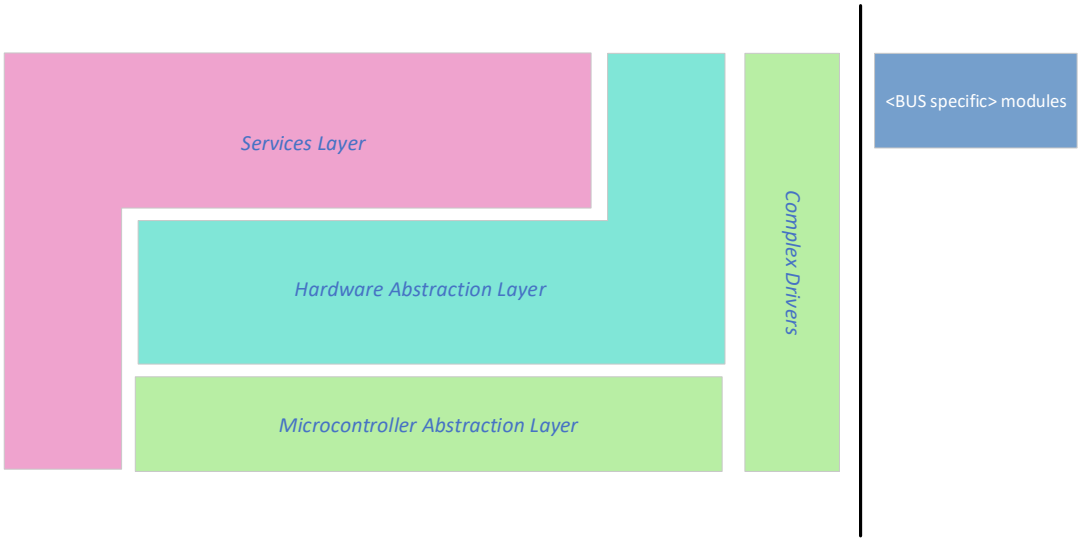


Figure 4 Basic Software Overview

5.Επισκόπηση Συστήματος

Ένα πλήρες αυτόνομο αυτοκίνητο το οποίο χρησιμοποιεί το πρωτόκολλο επικοινωνίας V2X πρέπει να είναι σε θέση να ανταλλάζει μηνύματα με το περιβάλλον καθώς και να είναι ασφαλής ο απόλυτος έλεγχος της οδήγησης από το σύστημα χωρίς την ανθρώπινη παρέμβαση. Για να επιτευχθούν τα προαναφερθέντα οι λειτουργιές κλειδιά ενός τέτοιου συστήματος (όπως φαίνονται και στην Figure1) είναι: η Αντίληψη του περιβάλλοντος, ο Εντοπισμός της τοποθεσία του οχήματος, οι V2X Επικοινωνίες, ο Σχεδιασμός της κίνησης, ο Έλεγχος της κίνησης καθώς και η Διαχείριση του Συστήματος .

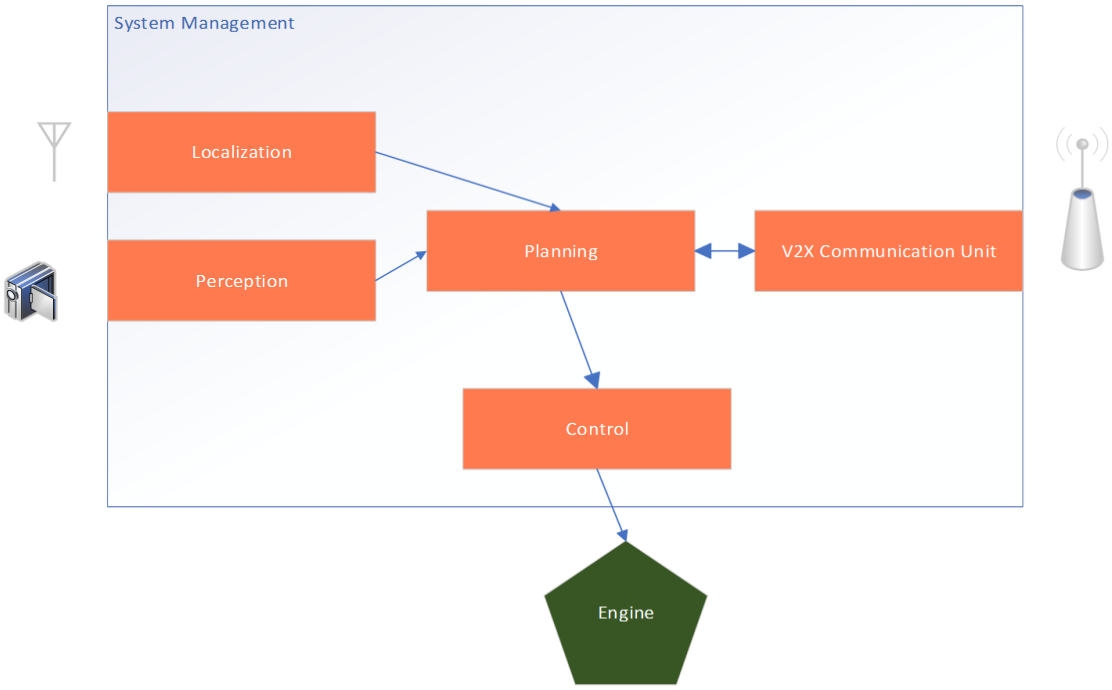


Figure 5 System Overview

- Η **Αντίληψη** του περιβάλλοντος χρησιμοποιεί αισθητήρες όπως Lidar, Radars και Computer Vision ώστε να προσδιορίσει τον περιβάλλοντα χώρο του αυτοκινήτου. Δηλαδή αυτή η συνάρτηση εντοπίζει την τοποθεσία των στατικών εμποδίων και εκτιμά το σχήμα του δρόμου.
- Ο **Εντοπισμός** είναι υπεύθυνος για την εύρεση της τοποθεσίας του αυτοκινήτου στον χάρτη χρησιμοποιώντας τεχνικές όπως το Παγκόσμια Δορυφορικό Σύστημα Πλοήγησης (GNSS), στίγμα εξ αναμετρήσεως (δηλαδή αναμέτρηση γεωγραφικού στίγματος από προηγούμενο στίγμα) και χάρτες δρόμου.

- **V2X Επικοινωνίες** είναι το συνονθύλευμα των V2I,V2P,V2V,V2N επικοινωνιών οι οποίες προσφέρουν την εξατομικευμένη ανάλυση των συνθηκών τις οποίες το όχημα πρέπει να λάβει υπόψιν .
- Ο **Σχεδιασμός** κίνησης συνδυάζει τα αποτελέσματα των προαναφερθέντων υποσυστημάτων με σκοπό να υπολογίσει το βέλτιστο πλάνο για την κίνηση και συμπεριφορά του οχήματος.
- Ο **Έλεγχος** του οχήματος είναι το υποσύστημα αυτό το οποίο λαμβάνει το σχέδιο κίνησης που έχει υπολογιστεί από το υποσύστημα του Σχεδιασμού και είναι υπεύθυνο για την εφαρμογή του στο αυτοκίνητο. Με άλλα λόγια ο **Έλεγχος** επιταχύνει ή φρενάρει την μηχανή βασιζόμενος στην έξοδο της συνάρτησης **Σχεδιασμού** κίνησης.
- Η **Διαχείριση Συστήματος** επιβλέπει το ολικό σύστημα οδήγησης χρησιμοποιώντας λειτουργίες όπως το σύστημα διαχείρισης σφαλμάτων (fault management system), το σύστημα καταγραφής (logging system) και η διεπαφή ανθρώπου-μηχανής (human–machine interface | HMI).

Ο τρόπος διασύνδεσης των παραπάνω υποσυστημάτων του υλικού με το λογισμικό φαίνεται στο παρακάτω σχήμα (fig.6).

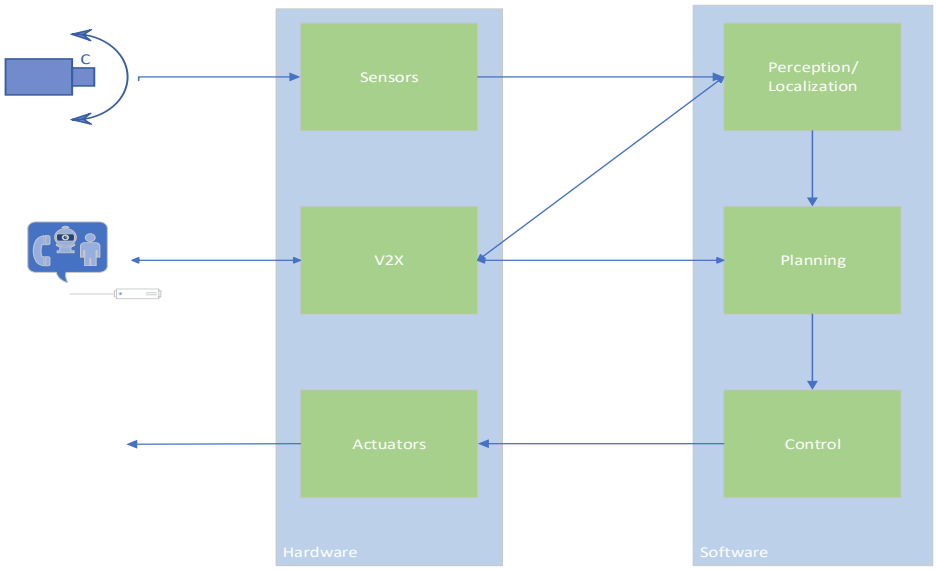


Figure 6 Διασύνδεση υποσυστημάτων

5.1. Αντίληψη Περιβάλλοντος (Perception)

Αισθητήρες όπως radars,κάμερες , υπέρηχοι αποτελούν τα όργανα μέτρησης με τα οποία το όχημα καταφέρνει να αναλύσει το περιβάλλοντα χώρο μέσα στον οποίο οδηγεί. Χρησιμοποιώντας της μετρήσεις των αισθητήρων η συνάρτηση της Αντίληψης ***Perception*** είναι υπεύθυνη για τον προσδιορισμό δύο βασικών χαρακτηριστικών του δρόμου :

1. Τα εμπόδια που βρίσκονται στην πορεία μας και κατά συνέπεια την επιφάνεια του οδοστρώματος.
2. Την σήμανση που υποδεικνύουν τα φανάρια.

Με άλλα λόγια το υποσύστημα της Αντίληψη ***Perception*** κωδικοποιεί και επεξεργάζεται κατάλληλα το περιβάλλον , με σκοπό να μεταφέρει τα αποτελέσματα αυτά στο επόμενο στάδιο επεξεργασίας, δηλαδή στο σύστημα του Σχεδιασμού Πορείας Planning .

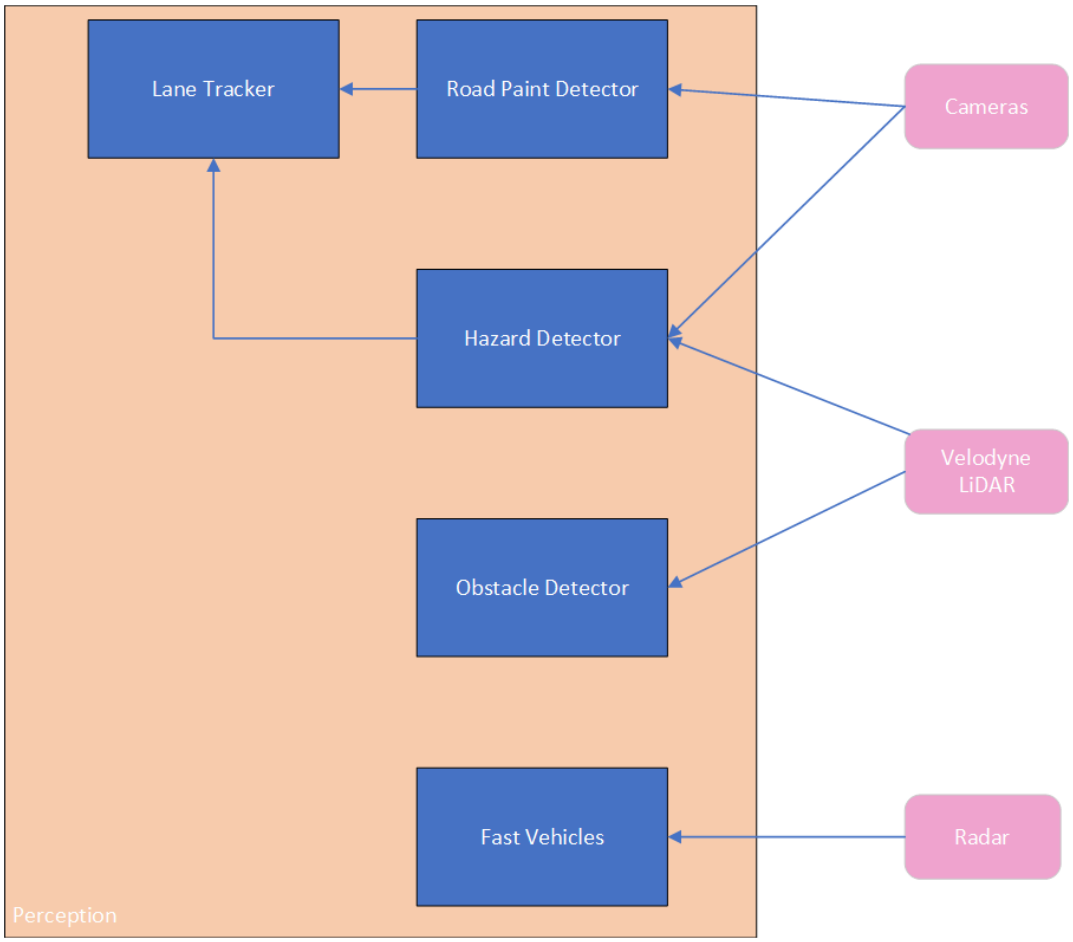


Figure 7 Perception's architecture

Τα επιμέρους υποσυστήματα της Αντίληψης είναι :

- Road Paint Detector επεξεργάζεται τις εικόνες που προσφέρουν οι κάμερες ώστε να γίνει μία πρώτη εκτίμηση των λωρίδων του δρόμου.
- Hazard Detector χρησιμοποιώντας τον συνδυασμό δισδιάστατης 2-D και τρισδιάστατης απεικόνισης εντοπίζονται οι λακκούβες στον δρόμο.
- Obstacle Detector προσδιορίζει την θέση και την ταχύτητα που έχουν τα στατικά εμπόδια.
- Lane Tracker συνενώνει τα αποτελέσματα από τον Road Paint Detector και του Hazard Detector με σκοπό την διασταύρωση των αποτελεσμάτων από διαφορετικά όργανα αντίληψης.
- Fast Vehicles χρησιμοποιεί τον LiDAR αισθητήρα με σκοπό την έγκαιρο εντοπισμό οχήματος το οποίο προσεγγίζει με μεγάλη ταχύτητα.

Τα είδη των αισθητήρων που χρησιμοποιούμε σε αυτή την αρχιτεκτονική είναι :

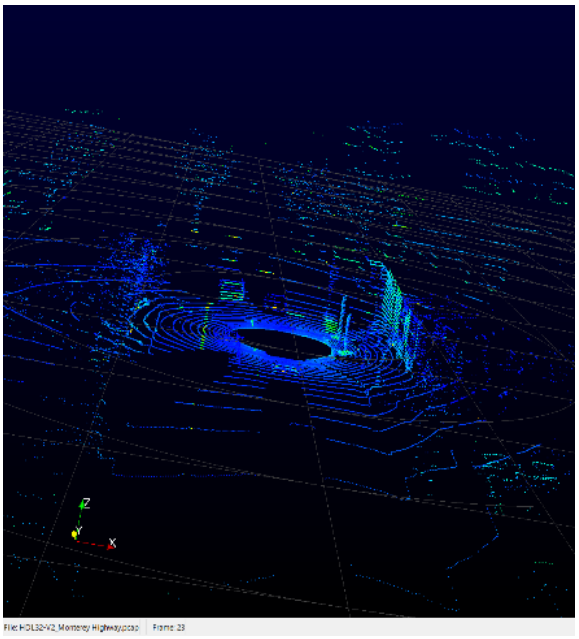
- Κάμερες
- LiDAR
- Radar
- Σύστημα υπέρηχων

Στην συνέχεια περαιτέρω την χρήση των αισθητήρων αυτών.

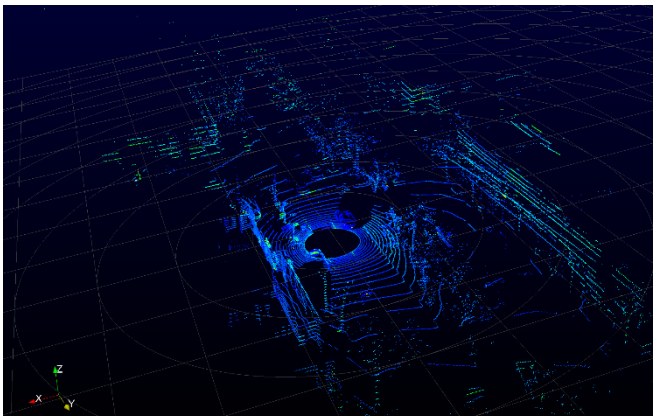
Πιο συγκεκριμένα οι LiDAR αισθητήρες κάνουν δυνατή την τρισδιάστατη απεικόνιση δημιουργώντας το « νέφος σημείων » (Point Cloud) και μπορούν να πάρουν μετρήσεις οι οποίες καθορίζουν το σχήμα και το βάθος των εμποδίων όπως επίσης και την μορφολογία του εδάφους. Οι LiDAR αισθητήρες είναι εφοδιασμένοι με ενισχυτές φωτός με εξαναγκασμένη εκπομπή ακτινοβολίας *(Amplification by Stimulated Emission of Radiation)* οι οποίοι όμως έχουν εδραιωθεί με την αγγλική τους ορολογία laser. Οι αισθητήρες αυτοί λοιπόν «μετρούν» τον χρόνο που απαιτείται για την δέσμη φωτός να ανακλαστεί από την επιφάνεια και να γυρίσει πίσω στο scanner (Time of flight). Η απόσταση λοιπόν μεταξύ του αντικειμένου και του αισθητήρα είναι ο χρόνος αυτός συναρτήσει της ταχύτητας του φωτός.

Παρακάτω παραθέτω μία αναπαραστάσεις που λήφθηκαν με την βοήθεια του ‘’ προγράμματος για έναν lidar αισθητήρα της εταιρίας Velodyne καθώς και την αναπαράσταση των δεδομένων μέσω excel.

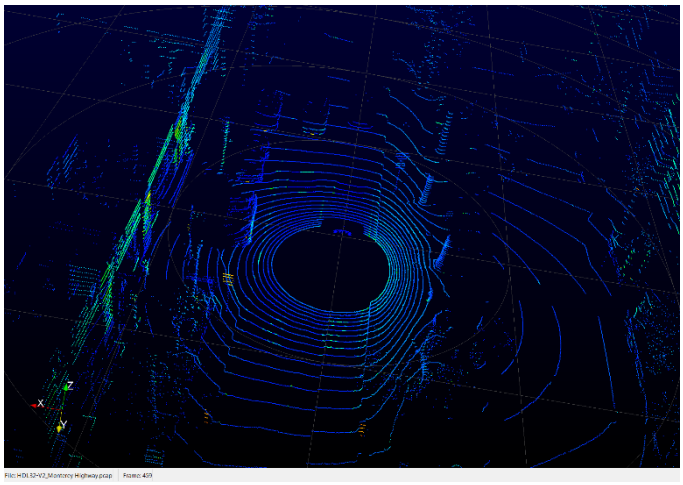
.

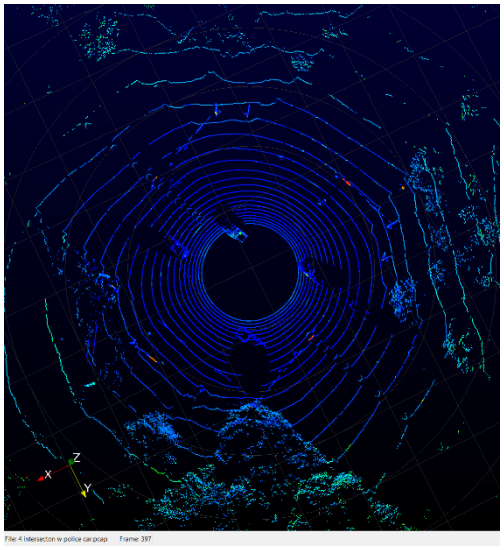


Εικόνα 6 Αναπαράσταση 3

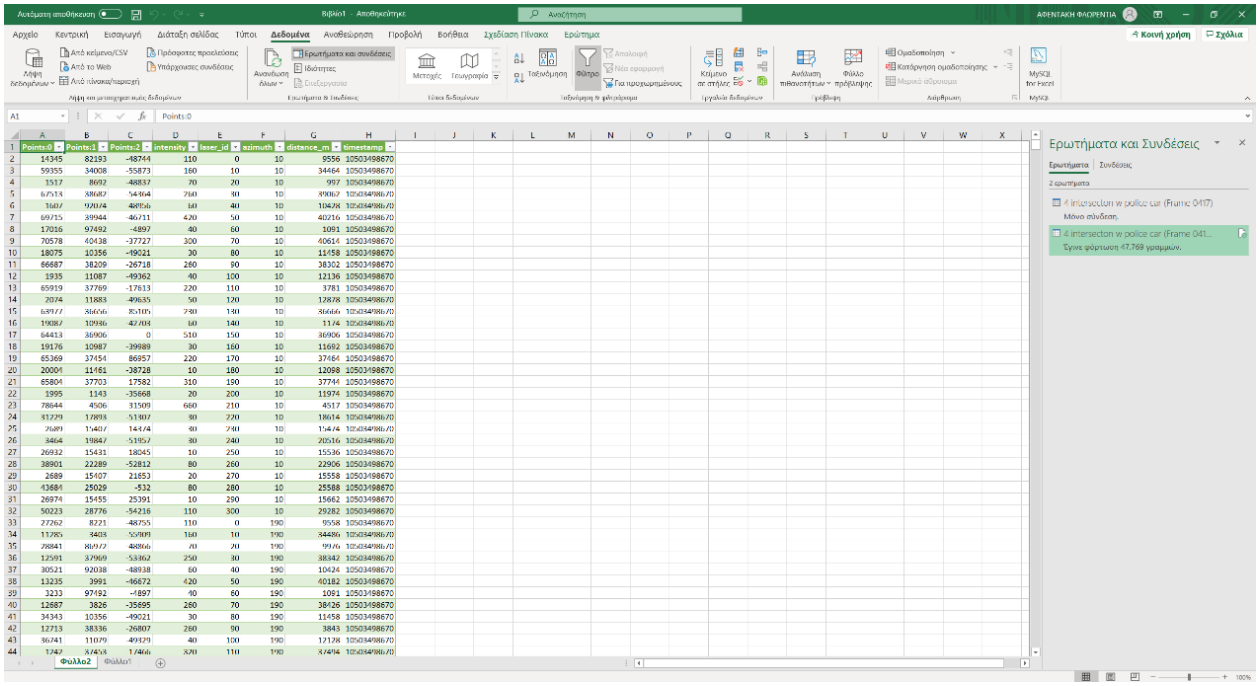


Εικόνα 5 Αναπαράσταση 2





Εικόνα 4 Αναπαράσταση 1



Points:0	Points:1	Points:2	intensity	laser_id	azimuth	distance_m	timestamp
14345	82193	-48744	110	0	10	9556	1050349867
59355	34008	-55873	160	10	10	34464	1050349867
1517	8692	-48837	70	20	10	997	1050349867
67518	38687	-54464	260	30	10	39067	1050349867
3607	92074	-48956	60	40	10	10428	1050349867
69715	39944	-46711	420	50	10	40216	1050349867
17016	97492	-4897	40	60	10	1091	1050349867
70578	40438	-37727	300	70	10	40614	1050349867
18075	10356	-49021	30	80	10	11458	1050349867
66687	38209	-26718	260	90	10	38302	1050349867
1935	11087	-49362	40	100	10	12136	1050349867
65919	37769	-17613	220	110	10	3781	1050349867
2074	11883	-49635	50	120	10	12878	1050349867
69977	40666	-35105	280	130	10	39666	1050349867
19087	10986	-47408	60	140	10	1174	1050349867
64413	36906	0	510	150	10	36906	1050349867
19176	10987	-39989	30	160	10	11692	1050349867
65369	37454	86957	220	170	10	37464	1050349867
20004	11461	-38728	10	180	10	12098	1050349867
65804	37703	17582	310	190	10	37744	1050349867
1995	1143	-35668	20	200	10	11974	1050349867
78644	4506	31509	660	210	10	4517	1050349867
81229	17893	51807	80	220	10	18614	1050349867
2689	15407	14474	80	230	10	15474	1050349867
3464	19847	51957	80	240	10	20516	1050349867
26932	15431	18045	10	250	10	15536	1050349867
38901	22289	-52812	80	260	10	22906	1050349867
2689	15407	21653	20	270	10	15558	1050349867
43684	25029	-532	80	280	10	25588	1050349867
26974	15455	25391	10	290	10	15662	1050349867
50223	28776	-54216	110	300	10	29282	1050349867
27262	8221	-48755	110	0	190	9558	1050349867
13785	8403	50909	160	10	190	8486	1050349867
28841	80977	48866	70	20	190	9976	1050349867
12591	37969	-53862	250	30	190	38342	1050349867
30521	92038	-48938	60	40	190	10424	1050349867
13235	3991	-46672	420	50	190	40182	1050349867
3233	97492	-4897	40	60	190	1091	1050349867
12687	3826	-35695	280	70	190	38426	1050349867
34343	10356	-49021	30	80	190	11458	1050349867
12713	38336	-26807	260	90	190	3843	1050349867
86741	11079	-49329	40	100	190	12128	1050349867
1747	37454	17466	370	110	190	37494	1050349867

Υπόδειγμα 1 Καταγραφής δεδομένων

5.2.Εντοπισμός Τοποθεσίας (Localization)

Για τον εντοπισμό της τοποθεσίας του οχήματος μας στον χάρτη μέσω της SPAN (Synchronized Position Attitude Navigation) χρησιμοποιούμε την συγχώνευση :

- Της Αδρανειακής μονάδα μέτρησης [Inertial Measurement Unit (IMU)] η οποία χρησιμοποιεί Accelerometer, Gyroscope και Magnetometer για τις μετρήσεις του
- Του Παγκόσμιου Δορυφορικού Συστήματος Πλοήγησης GNSS.

Έτσι καταφέρνουμε να έχουμε μεγαλύτερη ακρίβεια στις μετρήσεις μας , το οποίο σε συστήματα πραγματικού χρόνου ,όπως αυτό που εξετάζουμε, έχει καθοριστικό ρόλο . Ακόμη με την χρήση SPAN τεχνολογίας δεν χάνουμε το στίγμα του οχήματος μας στον χάρτη ακόμη και σε περιοχές εκτός κάλυψης δικτύου. Στην συνέχεια παραθέτω την αρχιτεκτονική της συνάρτησης Εντοπισμού. (Figure 3)

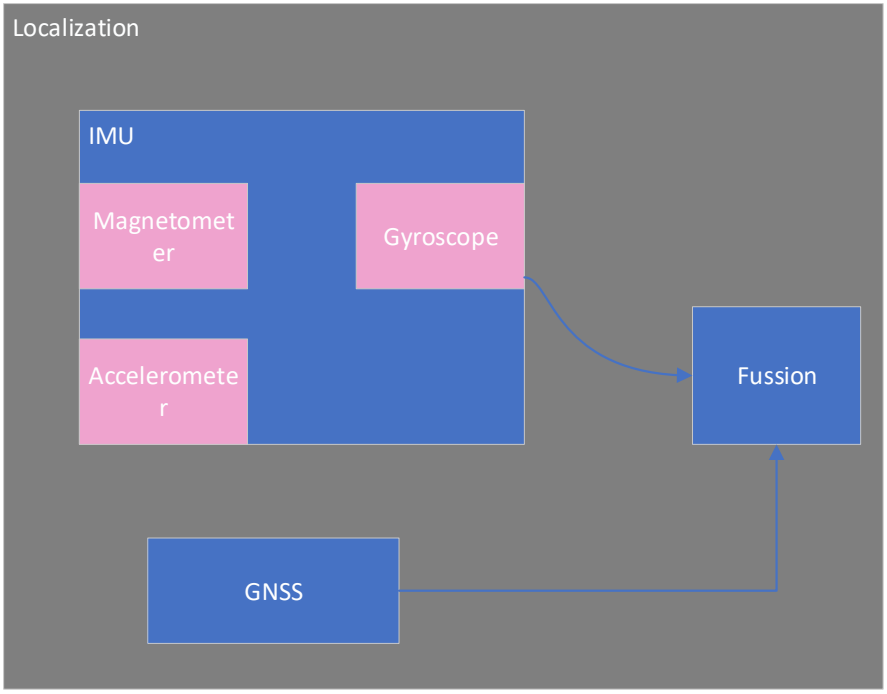


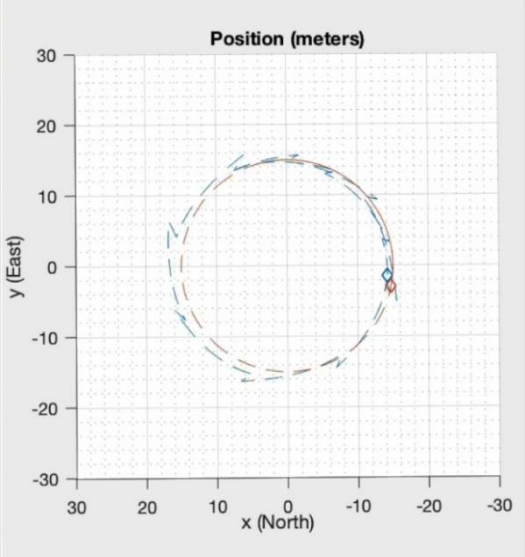
Figure 8 Localization Architecture

Αναλυτικότερα ο GPS εκλείπει σε συχνότητα διάδοσης και ακρίβειας της τοποθεσίας και το στίγμα που προσφέρει το χαρακτηρίζουμε ως την απόλυτη τοποθεσία του οχήματος. Στον αντίποδα η αδρανειακή μονάδα μέτρησης την χαρακτηρίζει υψηλή ταχύτητα διάδοσης και ακρίβεια. Στο σημείο αυτό πρέπει να αναφερθεί ότι η τοποθεσία ενός κινητού μπορεί να υπολογιστεί μέσω των

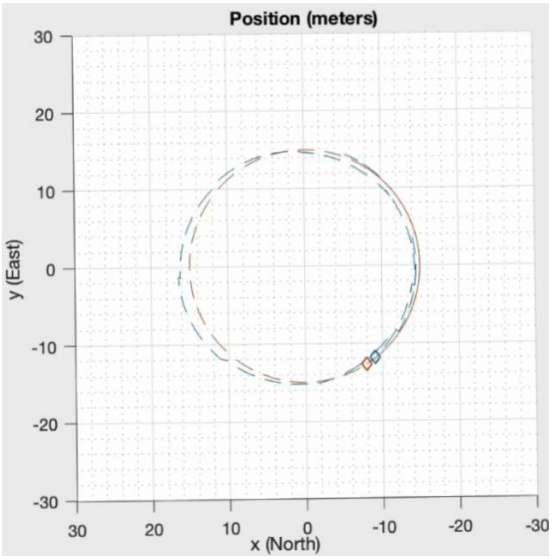
μετρήσεων του IMU. Τις συντεταγμένες αυτές τις χαρακτηρίζουμε ως το σχετικό γεωγραφικό στίγμα. Έτσι συνδυάζοντας το σχετικό και το απόλυτο γεωγραφικό στίγμα καταλήγουμε στον εντοπισμό της ακριβής τοποθεσίας του οχήματος.

Η σύνθεση (fusion) του εντοπισμού τοποθεσίας γίνεται μέσω του φίλτρου Kalman. Το φίλτρο αυτό χρησιμοποιώντας τη μνήμη του συστήματος από προηγούμενες καταστάσεις και συνδυάζοντας στοιχεία όπως τα σφάλματα , την δυναμική των μετρήσεων, την στατική περιγραφή του συστήματος και τις αρχικές συνθήκες, συντάσσει τελικά την πρόβλεψη της τοποθεσίας.

Παρακάτω παραθέτω πειραματικές μετρήσεις στις οποίες συγκρίνεται η μεμονωμένη χρήση GPS σε αντιδιαστολή με την σύνθεση GPS και του IMU για τον εντοπισμό της τοποθεσίας και γίνεται φανερή η βελτιστοποίηση με την χρήση IMU.



Εικόνα 5 GPS Localization



Εικόνα 4 GPS+IMU Localization

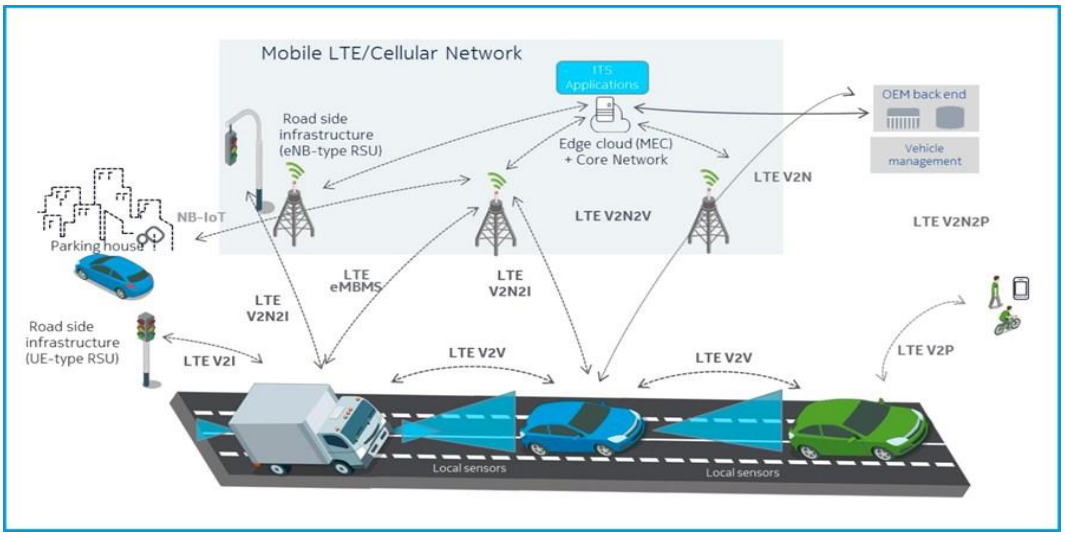
5.3.V2X Επικοινωνίες

Οι V2X επικοινωνίες αποτελούνται από τις εξής επιμέρους τεχνολογίες V2Vehicle, V2Pedestrians, V2Network, V2Infrastructure (όπως παραπέμπονται στην Figure5) και βασίζονται στις Mobile LTE and Cellular Επικοινωνίες Fig4. Πιο συγκεκριμένα στην εργασία αυτή βασιζόμαστε στο (Long Term Evolution) LTE-V δίκτυο απορρέει από το (Time-Division Long-Term-Evolution) TD-LTE και παρέχει δύο είδη δικτύων επικοινωνίας:

- LTE-V Cell αποτελεί βελτίωση της TD-LTE επικοινωνίας
- LTE-V Direct είναι μία κατανεμημένης αρχιτεκτονικής TD-LTE επικοινωνίας η οποία επιτρέπει την επικοινωνία ανάμεσα στα οχήματα (V2V).

Προτιμούμε το δίκτυο το LTE-V δίκτυο το οποίο έχει προταθεί από την [14] αφού το IEEE 802.11P αν και θεωρείται πρότυπο ειδικά σχεδιασμένο για τις επικοινωνίες στο Σύστημα Ευφυών Μεταφορών ΣΕΜ (Intelligent Transportation System , ITS) χαρακτηρίζεται με χαμηλή αξιοπιστία, μη-φραγμένη καθυστέρηση , διακοπτόμενη V2I συνδεσιμότητα.

Figure 9 V2X Network ©NGMN



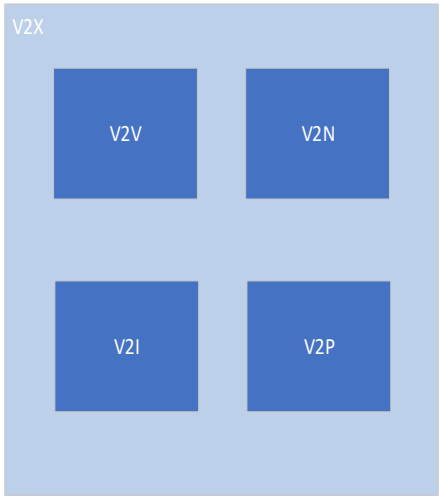


Figure 10 V2X Layout

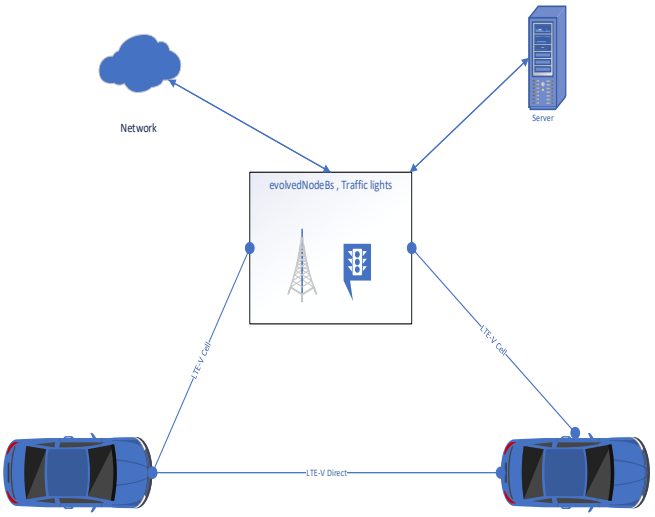


Figure 11 LTE-V Syste

5.4.Σχεδιασμός Πορείας (Planning)

Ο Σχεδιασμός ***Planning*** χρησιμοποιεί τα αποτελέσματα από την συνάρτηση της Αντίληψης *Perception* , δηλαδή την χαρτογράφηση του περιβάλλοντος χώρου του οχήματος, και σε συνδυασμό με την δυναμική του αυτοκινήτου υπολογίσει μία άνευ-συγκρούσεων πορεία .(Fig.6)

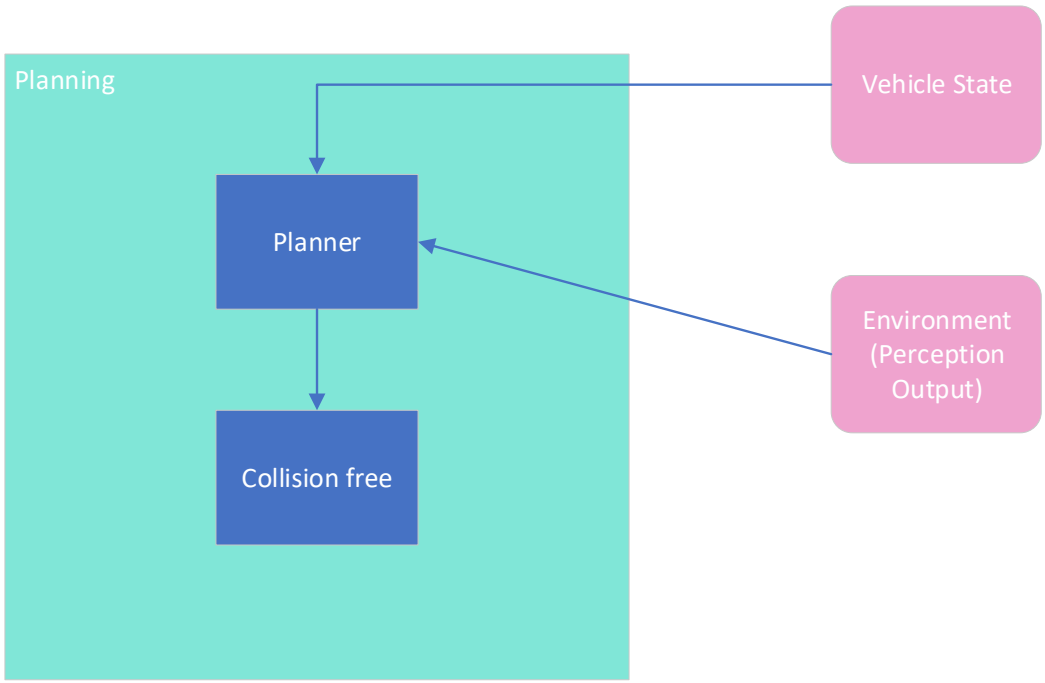


Figure 12 Planning

5.5.Έλεγχος Μηχανής (Control)

Ο Έλεγχος **Control** διαδίδει την πορεία - δηλαδή ένα σύνολο από στοιχεία πορείας όπως η θέση , η ταχύτητα ,ο προορισμός – στο σύστημα κίνησης του αυτοκινήτου. Η οποία με την σειρά της έχει υπολογιστεί από την συνάρτηση του Σχεδιασμού **Planning.** (Fig.7)

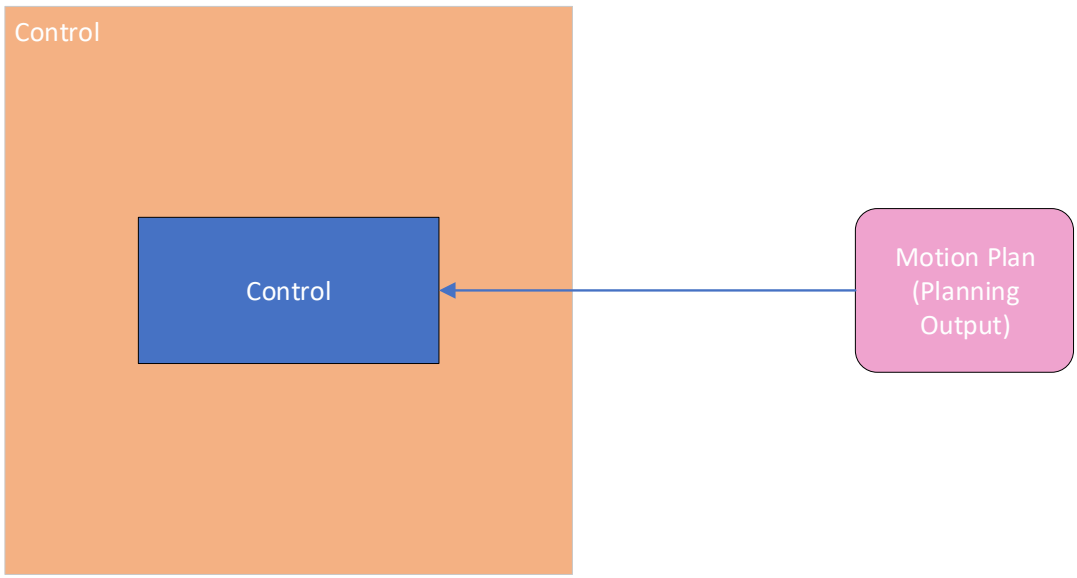


Figure 13 Control function

5.6.Διαχείριση Συστήματος (System Management)

Το σύστημα διαχείρισης σφαλμάτων (fault management system), το σύστημα καταγραφής (logging system) και η διεπαφή ανθρώπου-μηχανής (human–machine interface | HMI) είναι ζωτικής σημασίας λειτουργίες οι οποίες στην σχεδίαση ενός ενσωματωμένου συστήματος πρέπει να λαμβάνονται υπόψιν. Οι παραπάνω συναρτήσεις δομούν την Διαχείριση του Συστήματος **Fault Management**. (Figure9)

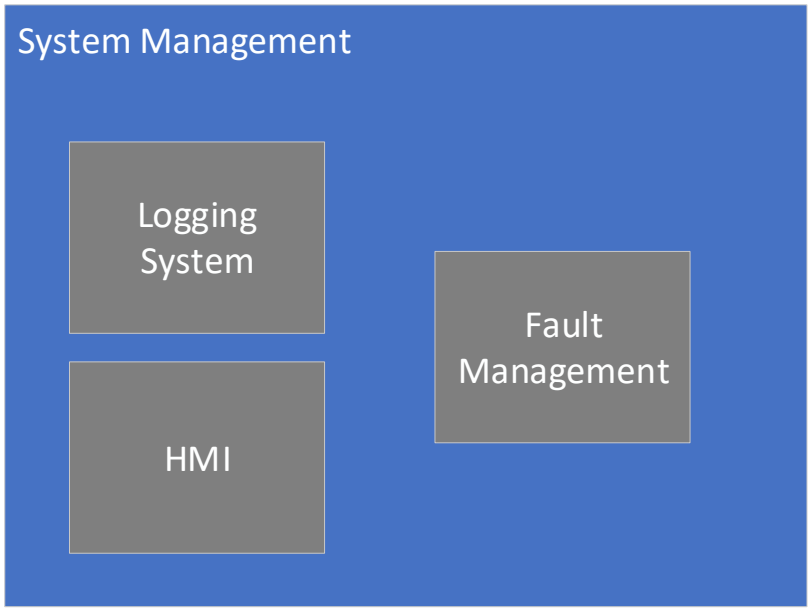


Figure 14 System Management

6. Υλοποίηση υποσυστήματος Ελέγχου

Στα πλαίσια αυτής της εργασίας αποφάσισα να υλοποιήσω του υποσύστημα του ελέγχου Control.

Η διαδικασία του Ελέγχου συμπεριλαμβάνει μετρήσεις οι οποίες χρησιμοποιούνται ώστε να διαλευκάνουν την ποιότητα της συμπεριφοράς του οχήματος. Έτσι ο Ελεγκτής δρα ώστε να φέρει το σύστημα προς την βέλτιστη κατάσταση, λύση και να αλλάξει την δυναμική του αυτοκινήτου αντίστοιχα.

Ο ανατροφοδοτούμενος Έλεγχος (Feedback controller) είναι ένα κλασσικό μοντέλο υλοποίησης που χρησιμοποιείται ευρέως σε πολλά συστήματα. Χρησιμοποιεί τις εσωτερικές μετρήσεις και επανορθώνει τις όποιες αποκλίσεις από την επιθυμητή κατάσταση του συστήματος. Κρίνεται αποτελεσματικός σε περιπτώσεις όπως στις αλλαγές των παραμέτρων ή στα λάθη προηγούμενης μοντελοποίησης καθώς επίσης σφάλματα μετρήσεων .

Ο PID ελεγκτής (Proportional-Integral-Derivative) ακολουθεί την εξής εξίσωση η οποία είναι βασισμένη στο σφάλμα του σήματος :

$$u(t) = K_d \dot{e} + K_P e + K_i \int_0^t e(t)\, dt \,, \qquad [1]$$

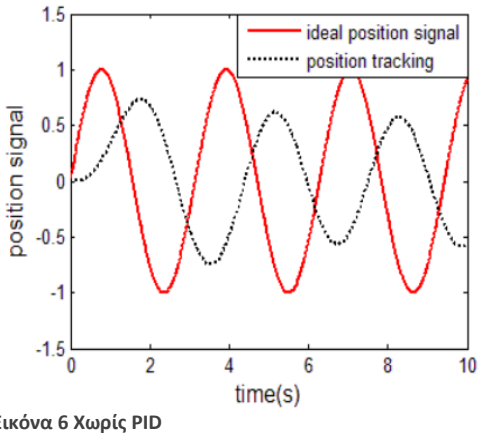
Όπου με e(t) συμβολίζουμε το σφάλμα της πορείας που θέλουμε να ελέγξουμε , k_p, k_I, k_d είναι ο γραμμικός , ολοκληρωτικός και ο παράγωγος όρος αντίστοιχα.

Ο παραπάνω τύπος μπορεί να αναλυθεί ως :

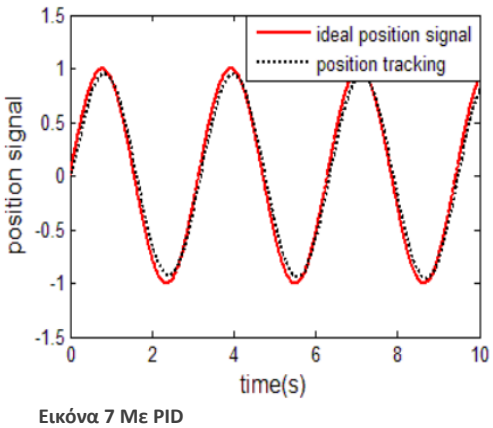
$$u(t) = K_p[e(t) + \frac{1}{T_i}\int_0^t e(t)dt + T_d \frac{de(t)}{dt}] \,, \qquad [2]$$

Με T_i ,T_d η ολοκληρωτική, παραγωγίσιμη χρονική σταθερά.

Πειραματικά έχει αποδειχτεί ότι μέσω της [15] :



Εικόνα 6 Χωρίς PID



Εικόνα 7 Με PID

Μπορούμε να εκφράσουμε την [2] και ως :

$$\Delta u(k) = \frac{K_p T_s}{T_i} r(k) - K_p \left(1 + \frac{T_s}{T_i} + \frac{T_d}{T_s}\right) y(k) + K_p \left(1 + \frac{2T_d}{T_s}\right) y(k-1) - K_p \frac{T_d}{T_s} y(k-2) \qquad [3]$$

Ωστόσο προτείνεται μία παραλλαγή της παραπάνω υλοποίησης η adaptive PID η οποία βασίζεται στον εξελεγκτικό αλγόριθμο και στην μέθοδο της γενικευμένης ελάχιστης διαφοράς που στα πλαίσια της συγκεκριμένης εργασίας δεν θα επεκταθούμε. Οι παράμετροι που επηρεάζουν άμεσα την προηγούμενη υλοποίηση πλέον μπορούν να προσαρμοστούν από τις αληθινού χρόνου απαιτήσεις. Έτσι Χρησιμοποιώντας την μαθηματική αναπαράσταση συστήματος και εκμεταλλευόμενοι την Diopantine εξίσωση καταλήγουμε:

$$\Delta u_0(k) = k[e(k) + e(k-1) + e(k-2)]r(k) - ke(k-1)y(k-1) - Ke(k-2)y(k-2) \qquad [4]$$

Από την [3]+[4]=>

$$K_p = -k[e(k-1) + 2e(k-2)],$$

$$T_i = \frac{-[e(k-1)+2e(k-2)]T_s}{e(k)+e(k-1)+e(k-2)},$$

$$T_d = \frac{e(k-2)T_s}{e(k-1)+2e(k-2)},$$

Όπου T_s η περίοδος της συνάρτησης του ελέγχου.

Όπως φαίνεται και από τις τελικές σχέσεις το σύστημα έχει χαμηλή απόσβεση όταν το k παίρνει μεγάλη τιμή ενώ αντίστροφα για μικρό k η απόσβεση είναι μεγάλη.

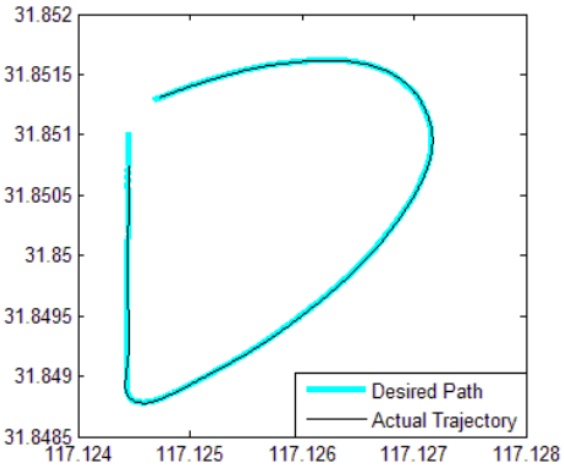
6.1.Πειραματικές μετρήσεις

PID διαδρομή:

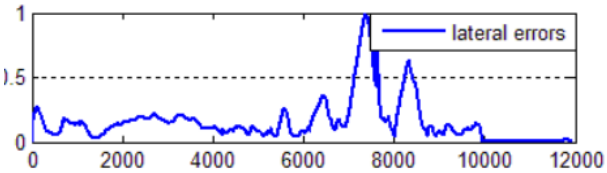
Στην εικόνα 6 φαίνεται η διαδρομή που το όχημα έχει ακολουθήσει στην πειραματική διαδικασία για την σύγκριση και αξιολόγηση των δύο αλγορίθμων.

Ενώ στην εικόνα 7 φαίνεται η διαδρομή που το κινητό ακολούθησε σε συνάρτηση με την αντιδιαστολή με την επιθυμητή πορεία κίνησης. Στην εικόνα 8 φαίνεται η απόκλιση (σφάλμα) της PID διαδρομής.

Εικόνα 8 Διαδρομή στο GoogleMaps



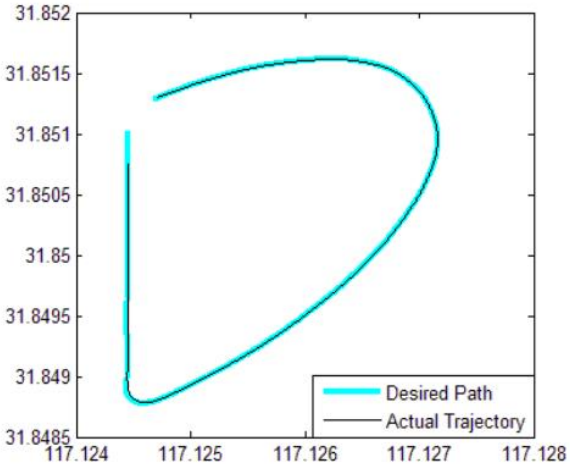
Εικόνα 6 Διαδρομή με PID [15]



Εικόνα 7 Σφάλμα διαδρομής με PID [15]

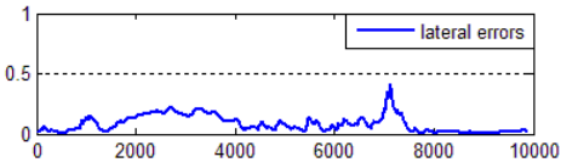
Adaptive PID διαδρομή:

Όπως και παραπάνω στην εικόνα 9 φαίνεται η διαδρομή που ακολούθησε το αυτοκίνητο βασισμένο σε αληθινού χρόνου μετρήσεις σε συνάρτηση με την προκαθορισμένη πορεία του. Ενώ στην εικόνα 10 το σφάλμα της διαδρομής.



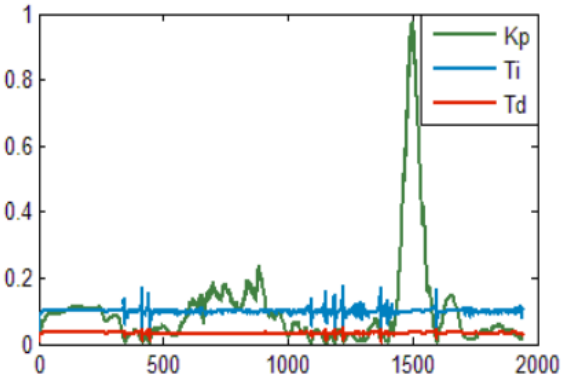
Εικόνα 9 Διαδρομή με Adaptive PID [15]

Παρατηρούμε και στις δύο περιπτώσεις ότι οι διαδρομές είναι αρκετά πανομοιότυπες με τις καθορισμένες με την adaptive PID να υπερτερεί. Γεγονός το οποίο φαίνεται καθαρότερα στα σφάλματα, όπου στην περίπτωση την adaptive PID συγκεκριμένα, το σφάλμα είναι κάτω από το 0,5 ενώ στην συμβατή PID το σφάλμα ακουμπά το 1.



Εικόνα 10 Σφάλμα διαδρομής με Adaptive PID [15]

Τέλος στην εικόνα 11 απεικονίζονται οι μεταβολές στις PID παραμέτρους κατά την διάρκεια της διαδρομής . Πιο συγκεκριμένα παρατηρούμε την αύξηση την Kp με την καμπύλωση της διαδρομής και την μείωση της κατά την σταθερή ευθεία πορεία εξασφαλίζοντας ευστάθεια .



Εικόνα 1 PID παράμετροι

7.Υλοποίηση Αντίληψης Perception:

Στα δεξιά παραθέτω κώδικα που αποτελεί υλοποίηση της συνάρτησης της αντίληψης που παρουσιάζω στο κεφάλαιο 5.1. η οποία δημιουργήθηκε από την baidu στα πλαίσια του ανοιχτού κώδικα project Apollo [16].

Αρχικά ο κώδικας δημιουργεί μία διαδικασία με τον το όνομα perception και στην συνέχεια την αρχικοποιεί έχοντας ως αρχική κατάσταση την OK η οποία υποδηλώνει ότι δεν υπάρχει σφάλμα και αρχικοποιεί τον adapter manager ο οποίος είναι υπεύθυνος για την διαχείριση των σημάτων ελέγχου στον σύστημα μας.

Στην συνέχεια ξεκινάει την έναρξη της αντίληψης

Όσο το λειτουργικό πραγματικού χρόνου τρέχει χωρίς πρόβλημα δηλαδή όσο η συσκευή μας λειτουργεί κανονικά χρησιμοποιείται μία συνάρτηση παρακολούθησης η οποία λαμβάνει τις απαραίτητες μετρήσεις για την καταγραφή του περιβάλλοντος χώρου . Στην συνέχεια χρησιμοποιεί τις

```
#include "modules/perception/perception.h"

#include "modules/common/adapters/adapter_manager.h"
#include "modules/perception/common/perception_gflags.h"
#include "third_party/ros/include/ros/ros.h"

namespace apollo
 {
        namespace perception {
                using apollo::common::adapter::AdapterManager;
                using apollo::common::Status;

                std::string Perception::Name () const {
                return "perception";
                }
        Status Perception::Init ()
        {
                AdapterManager::Init ();
                return Status::OK ();
        }

Status Perception::Start () {
         ros::AsyncSpinner spinner (1);
          spinner.start ();
          ros::waitForShutdown ();
        spinner.stop ();
        ros::Rate loop_rate(FLAGS_perception_loop_rate);
        while (ros::ok()) {
                AdapterManager:: Observe ();
                PerceptionObstacles;
                AdapterManager::FillPerceptionObstaclesHeader (
                Name(), perceptionObstacles.mutable_header());
        AdapterManager::PublishPerceptionObstacles(perceptionObstacles);

                TrafficLightDetection;
                AdapterManager::FillTrafficLightDetectionHeader (
                Name (), trafficLightDetection.mutable_header());

        AdapterManager::PublishTrafficLightDetection(trafficLightDetection);
                  }
         return Status::OK ();
        }

void Perception::Stop () {}

        } // namespace perception
} // namespace apollo
```

παραπάνω μετρήσεις για τον υπολογισμό των εμποδίων καθώς και την αναγνώριση των φαναριών.

Πρέπει να σημειωθεί ότι η κλάση Perception ορίζεται ως εξής :

```
class Perception : public apollo::common::ApolloApp {
public:
        std::string Name() const override;
        apollo::common::Status Init() override;
        apollo::common::Status Start() override;
        void Stop() override;
};
```

Δηλαδή περιέχει τις διαδικασίες που είναι υπεύθυνες για το όνομα , την αρχικοποίηση, την έναρξη και τον τερματισμό την Αντίληψης.

Τέλος έχουμε τα πρωτόκολλα των φαναριών :

```
syntax = "proto2";
package apollo.perception;
import "modules/common/proto/header.proto";
message TrafficLight {
        enum Color {
        UNKNOWN = 0;
        RED = 1;
        YELLOW = 2;
        GREEN = 3;
        };
        optional Color color = 1;
  // Traffic light string-ID in the map data.
        optional string id = 2;
  // How confidence about the detected results, between 0 and 1.
        optional double confidence = 3 [default = 1.0];
  // Duration of the traffic light since detected.
        optional double tracking_time = 4;
}

message TrafficLightDetection {
  optional apollo.common.Header header = 2;

  repeated TrafficLight traffic_light = 1;
}
```

Το οποίο περιέχει την περιγραφή του φαναριού πχ τον χρωματισμό τους καθορίζοντας το προεπιλεγμένο χρώμα σε κόκκινο, και τον εντοπισμό του.

Το πρωτόκολλο της αντίληψης του εμποδίου :

```
syntax = "proto2";

package apollo.perception;

import "modules/common/proto/header.proto";


// FIXME: To be merged with common::ErrorCode

enum PerceptionErrorCode {

  ERROR_NONE = 0;

  ERROR_TF = 1;

  ERROR_PROCESS = 2;

  ERROR_UNKNOWN = 3;

}


message Point {

  optional double x = 1;

// in meters.

  optional double y = 2;

// in meters.

  optional double z = 3;

// height in meters.

}

message PerceptionObstacle {

  optional int32 id = 1;

// obstacle ID.

  optional Point position = 2;

// obstacle position in the world coordinate system.

  optional double theta = 3;

// heading in the world coordinate system.

  optional Point velocity = 4;

// obstacle velocity.

// Size of obstacle bounding box.

  optional double length = 5; // obstacle length.

  optional double width = 6;

// obstacle width.

  optional double height = 7; // obstacle height.


  repeated Point polygon_point = 8;

// obstacle corner points.

  optional double tracking_time = 9;

// duration of an obstacle since detection in s.


  enum Type {

    UNKNOWN = 0;

    UNKNOWN_MOVABLE = 1;

    UNKNOWN_UNMOVABLE = 2;

    PEDESTRIAN = 3;

// Pedestrian, usually determined by moving behaviour.

    BICYCLE = 4;

// bike, motor bike

    VEHICLE = 5;

// Passenger car or truck.

  };

  optional Type type = 10;

// obstacle type

optional double timestamp = 11;

// GPS time in seconds.


  // Just for offline debuging, onboard will not fill this field.

  // Format like : [x0, y0, z0, x1, y1, z1...]

  repeated double point_cloud = 12 [packed = true];

}


message PerceptionObstacles {

  repeated PerceptionObstacle perception_obstacle = 1;

// An array of obstacles
```

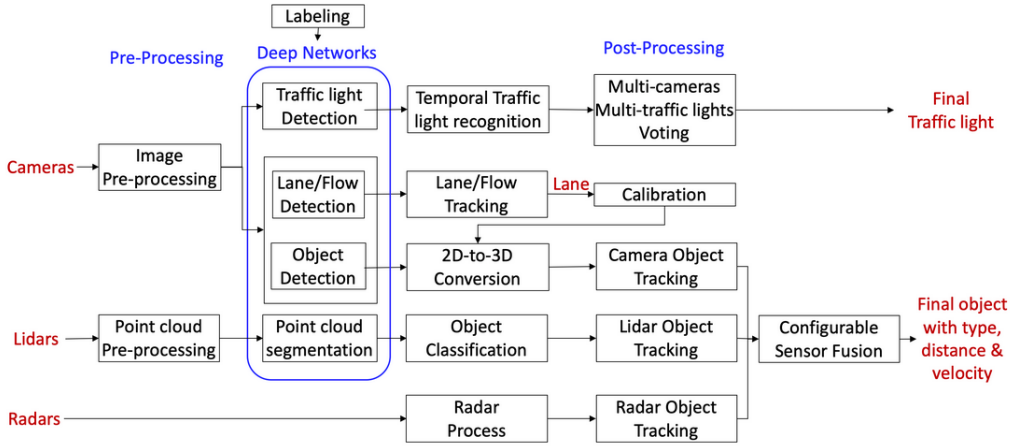
```
  optional apollo.common.Header header = 2;

  // Header
  optional PerceptionErrorCode error_code = 3 [default = ERROR_NONE];
}
```

Στο οποίο πρωτόκολλο ορίζει τα σφάλματα που μπορούν να προκύψουν κατά την διάρκεια της αντίληψης , ένα σημείο στον χώρος καθώς και ένα εμπόδιο χρησιμοποιώντας την προαναφερθέντα συνάρτηση όπως επίσης και μία δομή που χαρακτηρίζει τον τύπου του εμπόδιου δηλαδή κινητό ,στατικό πεζός .

8.Simulation

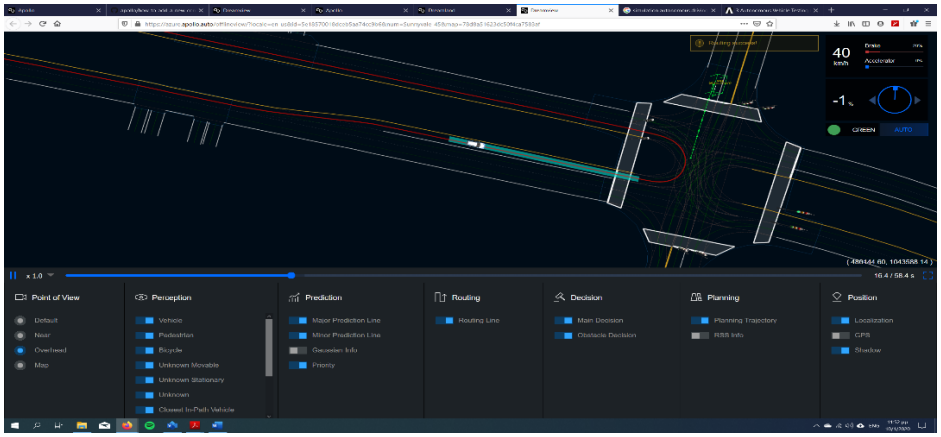
Η προσφορά της προσομοίωσης στην έρευνα και την επικύρωση πάνω στην αυτόνομη οδήγηση συγκαταλέγει την μείωση κόστους. Αφού πλέον τα σενάριο ελέγχονται εικονικά με τα εσφαλμένα μοντέλα να μην αποτελούσαν πλέον ζημία. Οι καιρικές συνθήκες, η κατάσταση του οδοστρώματος καθώς και οι ακραίες συνθήκες είναι κριτήρια αποτελούν μερικά από τα παραδείγματα που μπορούν να ελεγχθούν μέσω της προσομοίωσης.

Η Dreamland της Baidu αποτελεί εξομοιωτή του ενός προχωρημένου κώδικα από αυτόν που έχω παρουσιάσει στα χωρία 5.1, 7 συγκεκριμένα ακολουθεί την συγκεκριμένη αρχιτεκτονική :

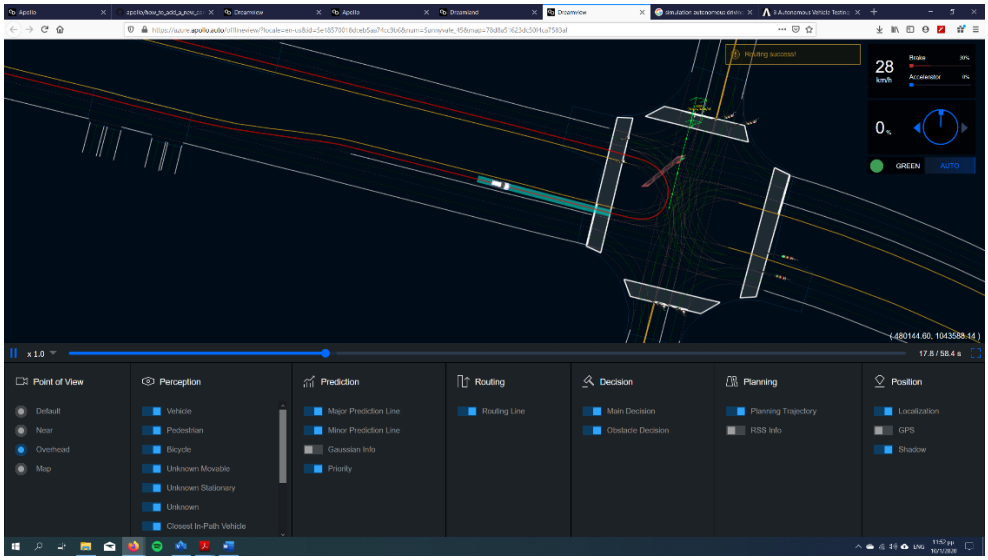


Εικόνα 10 Αρχιτεκτονική του Apollo 5.0

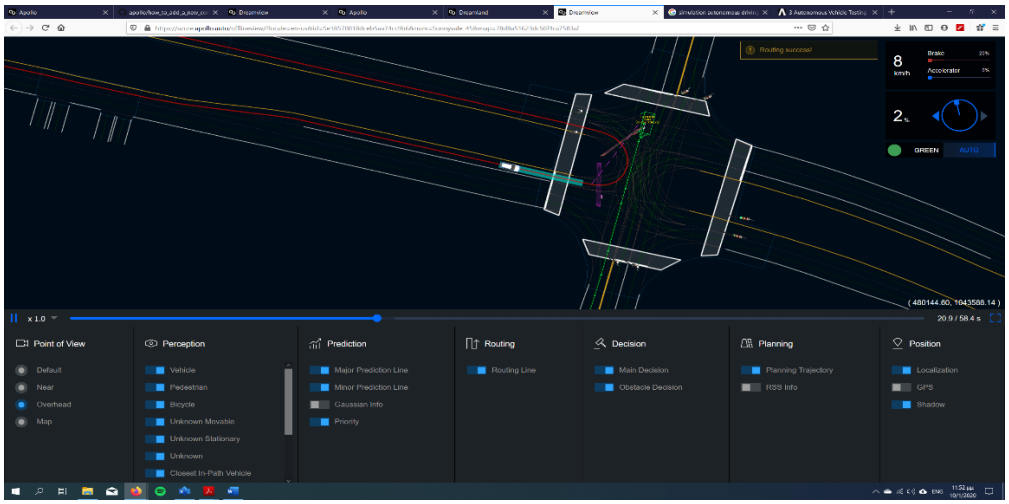
Στην συνέχεια οι εικόνες 11,12,13,14 παρουσιάζουν ένα σενάριο που εκτελέστηκε στον προαναφερθέντα προσομοιωτή. Κατά το οποίο το όχημα μας θέλει να κάνει αναστροφή σε διασταύρωση που την διασχίζει ένα αυτοκίνητο.



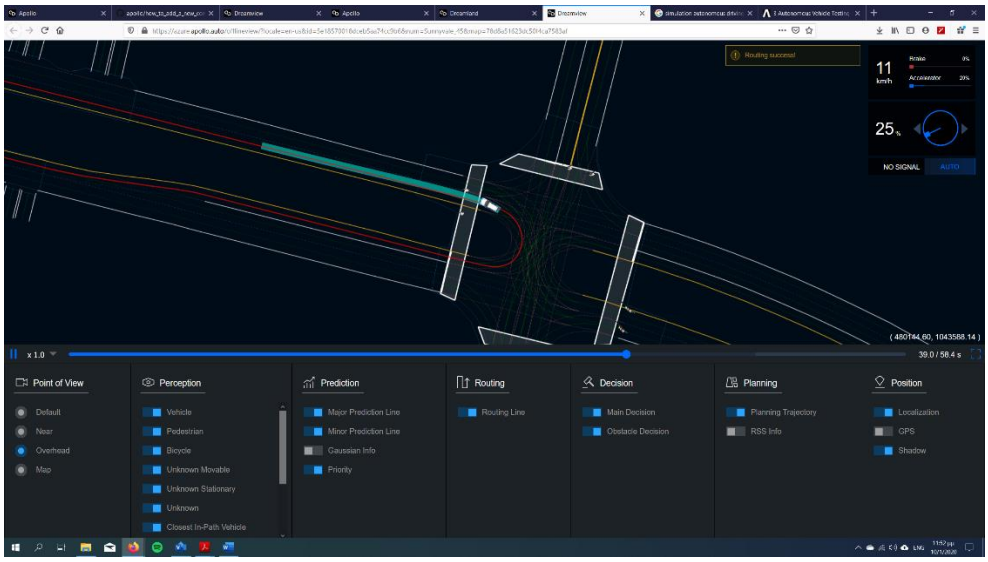
Εικόνα 11 Προσέλευση στην διασταύρωση



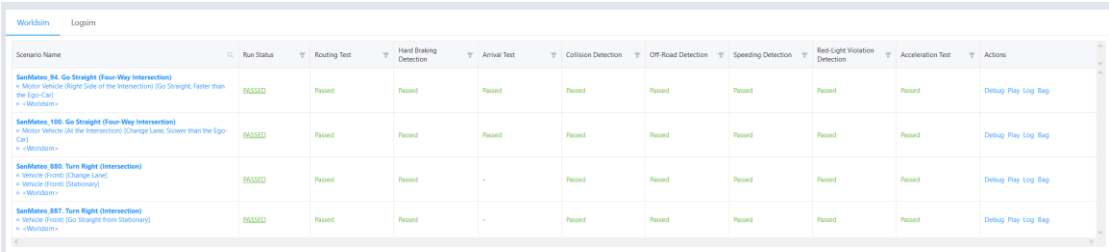
Εικόνα 12 Εντοπισμός κινητού εμποδίου



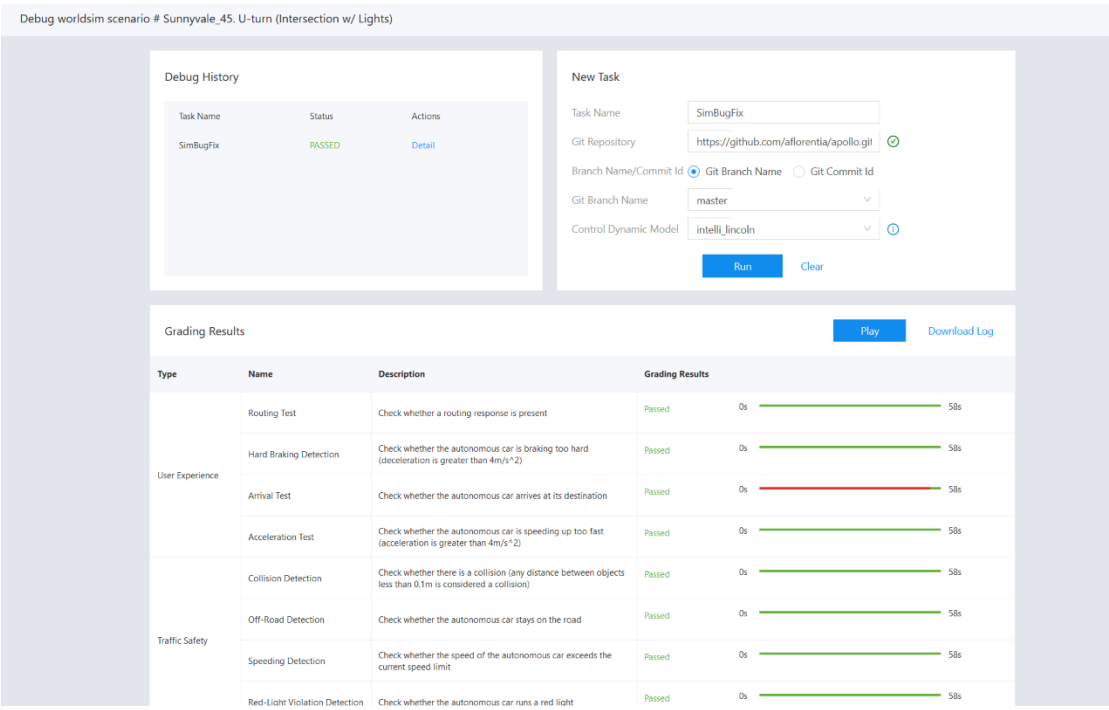
Εικόνα 13 Επιβράδυνση λόγω πιθανής σύγκρουσης



Εικόνα 14 Ολοκλήρωση αναστροφής



Εικόνα 15 Αποτελέσματα προσομοίωσης



Εικόνα 16 Λεπτομερείς αναφορά με τους χρόνους εκτέλεσης

Στις εικόνες 15,16 φαίνονται τα αποτελέσματα της προσομοίωσης. Γνώμονες αποτελούν τα αποτυχημένα σενάρια καθώς και εκείνα με μεγάλες χρονικές απαιτήσεις για την ανάπτυξη και βελτίωση ενός τέτοιου σχεδίου ανάπτυξης.

9.Τελικά

Σήμερα εταιρίες όπως η Tesla, Nvidia, BMW ,Waymo ανταγωνίζονται στο ποια θα παράγει πρώτο πλήρως αυτοματοποιημένο εμπορικό αυτοκίνητο στην αγορά. Σε αυτή την κούρσα ταχύτητας η Waymo έχει καταφέρει να επιτύχει το μεγαλύτερο επίπεδο αυτοματισμού αλλά ο στόλος των αυτοκινήτων της απευθύνεται μόνο στην πόλη της Φοίνιξ στην Αριζόνα. Από την άλλη η Tesla έχει απορρίψει τους κλασσικούς lidar αισθητήρες με σκοπό να μειώσει το κόστος των αυτοκινήτων της προσθέτοντας περισσότερες κάμερες και επενδύοντας στα τεχνητά νευρωνικά δίκτυα και την μηχανική μάθηση ακολουθώντας έτσι ένα μακροπρόθεσμο σχέδιο ανέλιξης των οχημάτων της.

Εταιρίες όπως η Waymo αλλά και πανεπιστήμια όπως το MIT προσφέρουν δωρεάν αποτελέσματα έρευνας καθώς και δελτία δεδομένων (data-sheets) με σκοπό την προάσπιση της μεγαλύτερης έρευνας πάνω στα επιμέρους πεδία που απαρτίζουν το ζήτημα που εξετάζουμε . Αναφορικά κάποια από τα πεδία αυτά αποτελούν τεχνολογίες αιχμής όπως νευρωνικά δίκτυα και μηχανική μάθηση , 5G επικοινωνίες.

Συνοψίζοντας σε επόμενο στάδιο της παρούσας εργασίας θα μπορούσε να γίνει προσέγγιση μίας κατανεμημένης αρχιτεκτονικής υλικού σε αντίθεση με την κεντροποιήμενη που ακολουθούμε στο χωρίο 3. Η οποία έναντι της κεντροποιήμενης προσφέρει:

- Μεγαλύτερη Ανεξαρτησία υπομονάδων => δυνατότητα ελέγχου και ανάπτυξης χωρίς να χρειαστεί να αλλάξουμε και τις υπόλοιπες υπομονάδες
- Μεγαλύτερη Ασφάλεια => αν όλες οι ενέργειες γίνονται από την ίδια μονάδα είναι πιο εύκολο να παραβιαστεί ολόκληρο το σύστημα.
- Μειωμένη Υπολογιστική Πολυπλοκότητα => οι αλγόριθμοι που διέπουν το σύστημα που εξετάζουμε είναι πολύπλοκοι και μεγάλης έκτασης. Χρησιμοποιώντας αποκεντροποίηση των μονάδων επεξεργασίας προσφέρουμε στο σύστημα μεγαλύτερη σταθερότητα και παραλληλία.

Ακόμη θα μπορούσαμε να αντικαταστήσουμε τον lidar αισθητήρα με κάμερες και να αναπτύξουμε τον κώδικα μας χρησιμοποιώντας αλγόριθμους μηχανικής μάθησης μειώνοντας έτσι το κόστος των μεγάλης ακρίβειας lidar.

10.Βιβλιογραφία:

[1] Autosar, “Autosar Layered Software Architecture”, Autosar Classic Platform Release 4.3.1,2017-12-08.

[2] AUTOSAR,” Specification of I/O Hardware Abstraction”, AUTOSAR CP Release 4.3.0, 2016-11-30.

[3] Matthias Traub, Alexander Maier, and Kai L. Barbehön , “Future Automotive Archite cture and the Impact of IT Trends”, IEEE SOFTWARE, MAY/JUNE 2017.

[4] Vector "Autosar Classic Under Posix Operating Systems" , Technical Article February 2019.

[5] Autosar, “Specification of Vehicle-2-X Management” , AUTOSAR CP Release 4.3.1, 2017-12-08.

[6] Kichun Jo, Junsoo Kim, Dongchul Kim,Chulhoon Jang , Myoungho Sunwoo,” Development of Autonomous Car—Part I: Distributed System Architecture and Development Process”, IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, VOL. 61, NO. 12, DECEMBER2014

[7] LEONARD, John, et al. A perception-driven autonomous urban vehicle. *Journal of Field Robotics*, 2008, 25.10: 727-774.

[8] Scott Drew Pendleton 1, Hans Andersen 1, Xinxin Du 2, Xiaotong Shen 2, Malika Meghjani 2, You Hong Eng 2, Daniela Rus 3 and Marcelo H. Ang Jr. “Perception, Planning, Control, and Coordination for Autonomous Vehicles”,MDPI, Machines 2017,5,6.

[9] K Dixit, Shilp & Fallah, Saber & Montanaro, Umberto & Dianati, Mehrdad & Stevens, Alan & Mccullough, Francis & Mouzakitis , Alexandros. ” Trajectory planning and tracking for autonomous overtaking: State-of-the-art and future prospects”, Elsevier,20 February 2018.

[10] González, David, et al. "A review of motion planning techniques for automated vehicles." *IEEE Transactions on Intelligent Transportation Systems* 17.4 (2015): 1135-1145.

[11] Urmson, Chris, et al. "Autonomous driving in urban environments: Boss and the urban challenge." *Journal of Field Robotics* 25.8 (2008): 425-466

[12] Frankiewicz, Tobias, Lars Schnieder, and Frank Köster. "Application platform for Intelligent Mobility-Test site architecture and Vehicle2X communication setup." *ITS World Congress*. 2012.

[13] Sewalkar, Parag, and Jochen Seitz. "Vehicle-to-pedestrian communication for vulnerable road users: Survey, design considerations, and challenges." *Sensors* 19.2 (2019): 358.

[14] Shanzhi Chen,Senior Member, IEEE, Jinling Hu, Yan Shi, and Li Zhao,” LTE-V: A TD-LTE-Based V2X Solutionfor Future Vehicular Network”, IEEE INTERNET OF THINGS JOURNAL, VOL. 3, NO. 6, DECEMBER 2016.

[15] Pan Zhao, Jiajia Chen, Yan Song, Xiang Tao, Tiejuan Xu and Tao Mei.” Design of a Control System for an Autonomous Vehicle Based on Adaptive-PID.”, Int J Adv Robotic Sy, 2012, Vol. 9, 44:2012.

[16] Apollo codes https://github.com/ApolloAuto/apollo
[17] Apollo Dreamland Simulator https://azure.apollo.auto/introduction?locale=en-us
[18] Pete Marwedel ,"Σχεδιασμός Ενσωματωμένων Συστημάτων " DaVinci,2017