ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΑΓΡΟΝΟΜΩΝ ΚΑΙ ΤΟΠΟΓΡΑΦΩΝ ΜΗΧΑΝΙΚΩΝ - ΜΗΧΑΝΙΚΩΝ ΓΕΩΠΛΗΡΟΦΟΡΙΚΗΣ

ΕΡΓΑΣΤΗΡΙΟ ΦΩΤΟΓΡΑΜΜΕΤΡΙΑΣ - ΣΥΓΚΟΙΝΩΝΙΑΚΗΣ ΤΕΧΝΙΚΗΣ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

"ΔΙΕΡΕΥΝΗΣΗ ΤΗΣ ΠΡΑΓΜΑΤΙΚΗΣ ΤΡΟΧΙΑΣ ΔΙΕΡΧΟΜΕΝΩΝ ΟΧΗΜΑΤΩΝ ΣΕ ΚΑΜΠΥΛΟ ΟΔΙΚΟ ΤΜΗΜΑ ΜΕ ΧΡΗΣΗ UAV ΚΑΙ ΟΡΑΣΗΣ ΥΠΟΛΟΓΙΣΤΩΝ"

ΤΖΟΥΜΑΚΑΣ ΣΩΤΗΡΗΣ

Επιβλέπων :
Ιωαννίδης Χαράλαμπος, Καθηγητής Ε.Μ.Π

ΑΘΗΝΑ
ΟΚΤΩΒΡΙΟΣ 2022

ΕΥΧΑΡΙΣΤΙΕΣ

Πρώτα από όλα ευχαριστώ τον Καθηγητή κ. Χ. Ιωαννίδη για την εμπιστοσύνη που μου έδειξε για την ανάθεση της διπλωματικής εργασίας, όπως επίσης και για την καθοδήγηση του όλο αυτό το διάστημα της εκπόνησής της.

Ένα μεγάλο ευχαριστώ στην μεταδιδακτορική ερευνήτρια κα. Στυλιανή Βερυκόκου για την πολύτιμη βοήθειά της σε όλα τα στάδια της εργασίας. Την ευχαριστώ για την υπομονή της, το χρόνο που διέθεσε αλλά και την προθυμία της να μου εξηγήσει όλες τις απορίες που προέκυψαν κατά την εκπόνηση της εργασίας.

Από το Εργαστήριο της Φωτογραμμετρίας ευχαριστώ επίσης την κα.Σοϊλέ και την κα.Χλιβερού για την βοήθειά τους στη χρήση του UAV αλλά και των φωτογραμμετρικών πακέτων λογισμικού που χρησιμοποιήθηκαν.

Τέλος θα ήθελα να ευχαριστήσω την οικογένειά μου για την ανεκτίμητη στήριξη και συμπαράσταση όλα αυτά τα χρόνια.

ΠΕΡΙΛΗΨΗ

Η ανίχνευση και παρακολούθηση αντικειμένων αποτελούν σημαντικά χαρακτηριστικά της επιστήμης των υπολογιστών. Τα οφέλη τους δεν περιορίζονται μόνο στους κύκλους της επιστημονικής κοινότητας του κλάδου αλλά αντίθετα εισέρχονται ολοένα και περισσότερο στην καθημερινότητα ενός ευρύτερου συνόλου της κοινωνίας, παρέχοντας βοήθεια όπου χρειαστεί. Στην παρούσα διπλωματική εργασία επιχειρείται η δημιουργία μιας ολοκληρωμένης εφαρμογής για τον εντοπισμό, αναγνώριση και παρακολούθηση οχημάτων σε πραγματικό χρόνο από βίντεο που έχουν ληφθεί από μη επανδρωμένα αεροσκάφη (UAV) με τη βοήθεια νευρωνικών δικτύων. Τα νευρωνικά δίκτυα, επίσης γνωστά ως τεχνητά νευρωνικά δίκτυα αποτελούν ένα υποσύνολο της μηχανικής μάθησης και βρίσκονται στο επίκεντρο των αλγορίθμων βαθιάς μάθησης. Το όνομα και η δομή τους είναι εμπνευσμένα από τον ανθρώπινο εγκέφαλο, μιμούμενα τον τρόπο με τον οποίο οι βιολογικοί νευρώνες στέλνουν σήματα ο ένας στον άλλο. Η εφαρμογή πέραν αυτών εκτελεί και ορισμένες ακόμα λειτουργίες όπως ο σχεδιασμός της τροχιάς των οχημάτων και η εκτίμηση της ταχύτητάς τους, κατά τη διέλευσή τους από ένα καμπύλο οδικό τμήμα. Τα αποτελέσματά της κρίνονται ιδιαίτερα ικανοποιητικά, συνιστώντας την εφαρμογή ένα χρήσιμο εργαλείο για πληθώρα δραστηριοτήτων σε τομείς πολιτικούς και στρατιωτικούς που αφορούν την παρακολούθηση όχι μόνο οχημάτων αλλά και γενικότερα άλλων αντικειμένων με χρήση state-of-the-art αλγορίθμων.

ABSTRACT

Object detection and tracking are important features of computer science. Their benefits are not limited to the scientific community in the industry, but are increasingly entering the everyday life of a wider society, providing assistance where needed. In this thesis we attempt to create an integrated application for real-time vehicle tracking, recognition and monitoring from videos captured by unmanned aerial vehicles (UAVs) using neural networks. Neural networks, also known as artificial neural networks are a subset of machine learning and are at the heart of deep learning algorithms. Their name and structure are inspired by the human brain, mimicking the way biological neurons send signals to each other. In addition to these, the application performs a number of other functions such as planning the trajectory of vehicles and estimating their speed as they pass through a curved road section. Its results are considered highly satisfactory, making the application a useful tool for a variety of activities in civil and military domains involving the tracking of not only vehicles but also other objects in general using state-of-the-art algorithms.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΙΣΑΓΩΓΗ

Οι εφαρμογές ανίχνευσης και παρακολούθησης οχημάτων διαδραματίζουν σημαντικό ρόλο για μη στρατιωτικές αλλά και στρατιωτικές εφαρμογές, όπως στον έλεγχο επιτήρησης της κυκλοφορίας αυτοκινητοδρόμων, τη διαχείριση και τον σχεδιασμό της αστικής κυκλοφορίας αλλά και την οδική ασφάλεια.

Ιδιαίτερα το ζήτημα της οδικής ασφάλειας είναι ένα θέμα μείζονος σημασίας τόσο σε παγκόσμιο επίπεδο όσο και στη χώρα μας που τα τελευταία 20 με 25 χρόνια τα αυτοκινητιστικά ατυχήματα αν και παρουσιάζουν μείωση στο σύνολό τους, δεν παύουν να είναι σε υψηλά ποσοστά. Η συνεχής αύξηση του απόλυτου αριθμού των οδικών ατυχημάτων που συνιστά λογικό επακόλουθο της αύξησης του παγκόσμιου πληθυσμού και συνάμα του αριθμού των οχημάτων που κυκλοφορούν, έχει συμπεριλάβει τα οδικά ατυχήματα στις πιο βασικές αιτίες θανάτου καθώς και μια πολύ μεγάλη κοινωνική δαπάνη. Συγκεκριμένα όσον αφορά την Ελλάδα, καταγράφονται ετήσια περίπου 20.000 οδικά ατυχήματα από τα οποία προκύπτουν περί τους 1700 νεκρούς και 30.000 τραυματίες. Παράλληλα στις χώρες της ΕΕ τα οδικά ατυχήματα έχουν ως αποτέλεσμα 55.000 νεκρούς, 1.7 εκατομμύρια τραυματίες και συνολικό κόστος 50 δισεκατομμύρια ευρώ. Τέλος σε παγκόσμια κλίμακα εκτιμάται ότι λαμβάνουν χώρα περίπου 500.000 θάνατοι και 15 εκατομμύρια τραυματισμοί από τα οδικά ατυχήματα κάθε χρόνο.

Μέσο μεταφοράς	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	% Μεταβολή	
											2018/2017	2018/2009
Σύνολο νεκρών	**1.456**	**1.258**	**1.141**	**988**	**879**	**795**	**793**	**824**	**731**	**700**	**-4,2**	**-51,9**
Επιβατικά	672	542	471	378	345	284	312	332	285	265	-7,0	-60,6
Δίτροχα	458	431	353	342	313	319	280	285	250	217	-13,2	-52,6
Πεζοί	202	179	223	170	151	125	128	149	118	146	23,7	-27,7
Άλλο είδος οχήματος	124	106	94	98	70	67	73	58	78	72	-7,7	-41,9

Σχήμα 1.1: Νεκροί από οδικά τροχαία ατυχήματα, κατά μέσο μεταφοράς (Πηγή: ΕΛ.ΣΤΑΤ)

Οι πιο βασικοί συντελεστές που επιδρούν στην οδική ασφάλεια κατά σειρά αυξανόμενης σπουδαιότητας είναι:

1. *Το όχημα* : μικρός αριθμός οδικών ατυχημάτων οφείλεται στις διάφορες μηχανικές ή άλλες βλάβες που είναι αποτέλεσμα μη σωστής ή ανεπαρκούς συντήρησης ή και παλαιότητας των εμπλεκόμενων οχημάτων.
2. *Η οδός και το Περιβάλλον* : εδώ συγκαταλέγονται εσφαλμένα γεωμετρικά χαρακτηριστικά, όπως είναι οι λωρίδες κυκλοφορίας και τα ερείσματα με ανεπαρκές πλάτος, η έλλειψη ή το μικρό πλάτος μεσαίων διαχωριστικών νησίδων καθώς και χαμηλά πρότυπα κατασκευής, κατά κανόνα μειωμένη πρόσφυση (ολισθηρά οδοστρώματα) όπως επίσης κακή μελέτη και κατασκευή παρόδιων στοιχείων: στηθαίων, στύλων,

αναχωμάτων και τάφρων. Επίσης η έλλειψη ή ανεπαρκής σήμανση και έλεγχος προσβάσεων ως αποτέλεσμα της κακής οργάνωσης της κυκλοφορίας.

3. *Χρήστες της Οδού* : Σημαντικότερος παράγοντας ενός οδικού ατυχήματος είναι ο άνθρωπος είτε ως οδηγός ή επιβάτης είτε ακόμα και ως πεζός. Η παράβαση των κανόνων της οδικής κυκλοφορίας όπως η λανθασμένη προσπέραση, απρόσεχτη οδήγηση, μέθη και η μη συμμόρφωση στη σήμανση και σηματοδότηση καθιστούν τις κυριότερες αιτίες οδικών ατυχημάτων.

Ως εκ τούτου κρίνεται πολύ σημαντική η συλλογή δεδομένων σχετικά με την οδηγητική συμπεριφορά, καθώς και η έρευνα πάνω σε παράγοντες που επηρεάζουν την εμφάνιση ατυχημάτων. Οι παραπάνω ενέργειες κρίνονται ιδιαίτερα ουσιώδεις και για τις καμπύλες οδών δύο λωρίδων κυκλοφορίας σε ανισόπεδους κόμβους, οι οποίες από διάφορες έρευνες που έγιναν έδειξαν ότι κατέχουν πολύ υψηλά ποσοστά εμφάνισης ατυχημάτων.

Οι ανισόπεδοι κόμβοι αν και καταργούν πλήρως τις ισόπεδες διασταυρώσεις των οχημάτων και συνεπώς μειώνουν από αυτή την άποψη τις πιθανότητες ατυχήματος, έχουν ως μειονέκτημά τους την δημιουργία συνθηκών ανομοιόμορφης ροής στην περιοχή που βρίσκονται. Ορισμένες προϋποθέσεις για ανομοιόμορφη ροή δημιουργούν διάφορες συνθήκες, όπως η διαφοροποίηση της γεωμετρικής διάταξης και των ταχυτήτων και η ανάγκη του οδηγού να λάβει άμεσα αποφάσεις για την πορεία που θα ακολουθήσει προσεγγίζοντας τον κόμβο.

Στον παρακάτω πίνακα (1.1) παρουσιάζονται οι δείκτες ατυχημάτων για διάφορους τύπους ανισόπεδων κόμβων. Στην συγκεκριμένη έρευνα [Cirillo , J. A August 1969], τονίζεται η ιδιαίτερα μεγάλη επιρροή του κυκλοφοριακού φόρτου στο δείκτη ατυχημάτων, η οποία επισκιάζει την αντίστοιχη επιρροή των γεωμετρικών χαρακτηριστικών σχεδιασμού των κόμβων.

Είδος Ανισόπεδου Κόμβου	Δείκτης Ατυχημάτων (Ατυχήματα/10^6 Οχήματα)
Πλήρες Τριφύλι Χωρίς Κατανεμητήριες Οδούς	1.69
Πλήρες Τριφύλι Με Κατανεμητήριες Οδούς	1.45
Μερικό Τριφύλι	1.04
Τρομπέτα	0.90
Πλήρες Διαμάντι	1.02

Πίνακας 1.1: Δείκτης Ατυχημάτων για Διάφορα Είδη Ανισόπεδων Κόμβων (Πηγή: Cirillo , J. A August 1969)

Διάφορες έρευνες δείχνουν ότι υπάρχει μια ευδιάκριτη τάση τα ατυχήματα να μειώνονται με την αύξηση του πλάτους του οδοστρώματος έως περίπου 7.5m (25 πόδια), ενώ υπάρχει αρνητική σχέση μεταξύ της ακτίνας της καμπύλης και του αριθμού των ατυχημάτων. Όσο

πιο μικρή είναι η ακτίνα της καμπύλης , τόσο περισσότερα είναι τα ατυχήματα όπου τα οχήματα καταλήγουν εκτός δρόμου [Choueiri, 1992]. Οι καμπύλες που απαιτούν απότομη ελάττωση και προσαρμογή της ταχύτητας και που προκαλούν ανομοιογένεια στα χαρακτηριστικά του δρόμου είναι ιδιαίτερα επικίνδυνες.

Νεότερες έρευνες έδειξαν πως η πιθανότητα ενός ατυχήματος σε μια οδό δύο λωρίδων κυκλοφορίας είναι μεγαλύτερη σε οριζόντιες καμπύλες, διασταυρώσεις και σε ανισόπεδους κόμβους. Σημαντική είναι και η διατύπωση από τους Krebs και Kloeckner [1977] που ισχυρίζονται ότι , αν οι κανόνες χάραξης εγγυώνται την ασφάλεια των οδών , τότε καθόλου ή ελάχιστα μόνο ατυχήματα θα πρέπει να συμβούν στις συγκεκριμένες οδούς , ενώ όταν αυτά συμβαίνουν την ευθύνη αναλαμβάνουν αποκλειστικά οι οδηγοί. Σύμφωνα με τους ίδιους τα ατυχήματα δεν κατανέμονται ομοιόμορφα στο οδικό δίκτυο. Οι τοποθεσίες υψηλού αριθμού ατυχημάτων αποτελούν σαφή ένδειξη ότι εκτός από το λάθος του οδηγού υπάρχουν και άλλες παράμετροι που επηρεάζουν την πιθανότητα ατυχήματος και σχετίζονται με τα χαρακτηριστικά της εκάστοτε οδού.

Πιο πρόσφατες έρευνες έχουν δείξει ότι η τροχιά των οδηγών σε μία καμπύλη έχει πολύ μεγάλη σημασία για την κατανόηση της οδηγητικής συμπεριφοράς καθώς υπάρχει μεγάλη ασυμφωνία μεταξύ της ακτίνας της τροχιάς που ακολουθούν οι οδηγοί και της ακτίνας της καμπύλης.

Ως εκ τούτου με τη συλλογή δεδομένων σχετικά με την τροχιά των οδηγών σε μια καμπύλη θα μπορούσαν να ληφθούν προληπτικά μέτρα για την αποφυγή περισσότερων ατυχημάτων. Πέραν λοιπόν της ταχύτητας ως μόνο κριτήριο, και η συμπεριφορά της πορείας που διαγράφουν οι οδηγοί, κρίνεται ίσως ως το πιο κατάλληλο για το διαχωρισμό ασυνείδητης ή μη σκόπιμης αποτυχίας (οδικού λάθους) [Spacer ,2000].

Οι παραπάνω λόγοι συντέλεσαν στην ανάγκη να δημιουργεί μια μεθοδολογία, με σκοπό την όσο το δυνατόν πιο γρήγορη και αποτελεσματική συλλογή δεδομένων αναφορικά με την πορεία των οχημάτων κατά μήκος μιας καμπύλης, ώστε να οδηγούμαστε σε ταχύτερα συμπεράσματα σχετικά με την αξιολόγηση της γεωμετρίας της οδού, αλλά και την συμπεριφορά των οδηγών με στόχο την λήψη περισσότερων προληπτικών και πιο ουσιαστικών μέτρων για την αύξηση της οδικής ασφάλειας. Σκοπός αυτής της διπλωματικής εργασίας είναι να παρουσιάσει λοιπόν μια προσέγγιση για την ταυτόχρονη ανίχνευση και παρακολούθηση οχημάτων που κινούνται σε καμπύλο οδικό τμήμα μέσω εναέριων video που λαμβάνονται από ένα μη επανδρωμένο αεροσκάφος (UAV), και την εκτίμηση της ταχύτητάς τους για την εξαγωγή ανάλογων συμπερασμάτων. Έχοντας ως στόχο την γρήγορη συλλογή και επεξεργασία των δεδομένων κρίθηκε σκόπιμο να δημιουργηθεί ένα πρόγραμμα που θα λαμβάνει από το χρήστη το βίντεο που ελήφθη από το UAV και στη συνέχεια θα εντοπίζει αυτόματα και θα παρακολουθεί τα οχήματα που διέρχονται από τον εκάστοτε κόμβο μελέτης. Στα χρονικά πλαίσια της παρούσας διπλωματικής εργασίας επιλέχθηκε ένας κόμβος μονής λωρίδας κυκλοφορίας εντός της Πολυτεχνειούπολης Ζωγράφου κυρίως για λόγους ασφαλείας ώστε το περιβάλλον στο οποίο θα λάμβαναν μέρος οι απαραίτητες διαδικασίες να ήταν όσο το δυνατόν πιο ελεγχόμενο. Η επιλογή έγινε έτσι ώστε να έχουμε ένα καμπύλο τμήμα οδού και με βασικό κριτήριο να είναι ορατό το οδόστρωμα για τη λήψη video με UAV.

Στο 1ο κεφάλαιο με τίτλο “Συνελικτικά Νευρωνικά δίκτυα και Ανίχνευση αντικειμένων” γίνεται μια εισαγωγή στη δομή ενός CNN και στον τρόπο λειτουργίας του, όπως επίσης και αναφορά στο ιστορικό, τη δομή και τη λειτουργία της YOLO ως ανιχνευτή αντικειμένων.

Στο 2ο κεφάλαιο με τίτλο “Παρακολούθηση αντικειμένων” γίνεται εισαγωγή στην παρακολούθηση αντικειμένων, στις διάφορες τεχνικές που χρησιμοποιούνται για την πραγματοποίησή της, καθώς και παρουσιάζεται ο τρόπος λειτουργίας του αλγορίθμου παρακολούθησης DeepSORT που χρησιμοποιήθηκε στην παρούσα διπλωματική.

Στο 3ο κεφάλαιο με τίτλο “Μεθοδολογία προσέγγισης” γίνεται αναφορά στη μεθοδολογία που ακολουθήθηκε για την πραγματοποίηση του εντοπισμού, της παρακολούθησης των οχημάτων και συνεπώς την αυτόματη εξαγωγή τροχιάς για κάθε ένα από αυτά ενώ στη συνέχεια αναφέρονται οι διαδικασίες με τις οποίες υπολογίστηκε η ταχύτητα τους και τα εργαλεία που χρησιμοποιήθηκαν για τον σκοπό αυτό.

Στο 4ο κεφάλαιο με τίτλο “Εφαρμογή σε καμπύλο τμήμα δρόμου” παρουσιάζεται η ολοκληρωμένη εφαρμογή δηλαδή ο κώδικας που χρησιμοποιήθηκε για αναγνώριση και παρακολούθηση μέσα από εναέρια βίντεο που έχουν ληφθεί από uav (Unmanned Air Vehicle). Πιο συγκεκριμένα γίνεται αναφορά στην υλοποίηση της YOLO v3, του DeepSORT και κατόπιν ο συνδυασμός των δύο για την τελική υλοποίηση. Στη συνέχεια αυτού παρουσιάζονται όλα τα αποτελέσματα που προέκυψαν από την εφαρμογή.

Στο 5ο και τελευταίο κεφάλαιο με τίτλο “Συμπεράσματα και προοπτικές” συγκεντρώνονται τα συμπεράσματα της παρούσας έρευνας ενώ αναφέρονται και προϋποθέσεις που απαιτούνται, ώστε να επιτευχθούν ακόμα καλύτερα αποτελέσματα στα υπολογιζόμενα μεγέθη.

ΚΕΦΑΛΑΙΟ 1ο

1. Συνελικτικά Νευρωνικά Δίκτυα και Ανίχνευση αντικειμένων

Στο παρόν κεφάλαιο, γίνεται αναφορά ορισμένων βασικών εννοιών σχετικά με τον εντοπισμό αντικειμένων σε βιντεο-εικόνες, όπως επίσης στους διάφορους εντοπιστές-detectors που χρησιμοποιούνται σε εφαρμογές παρακολούθησης οχημάτων. Πιο συγκεκριμένα παρουσιάζονται τα χαρακτηριστικά των αλγορίθμων ανίχνευσης αντικειμένων της “οικογένειας” CNN και ιδιαίτερα της YOLO που ήταν και ο αλγόριθμος ανίχνευσης που χρησιμοποιήσαμε στα πλαίσια αυτής της διπλωματικής.

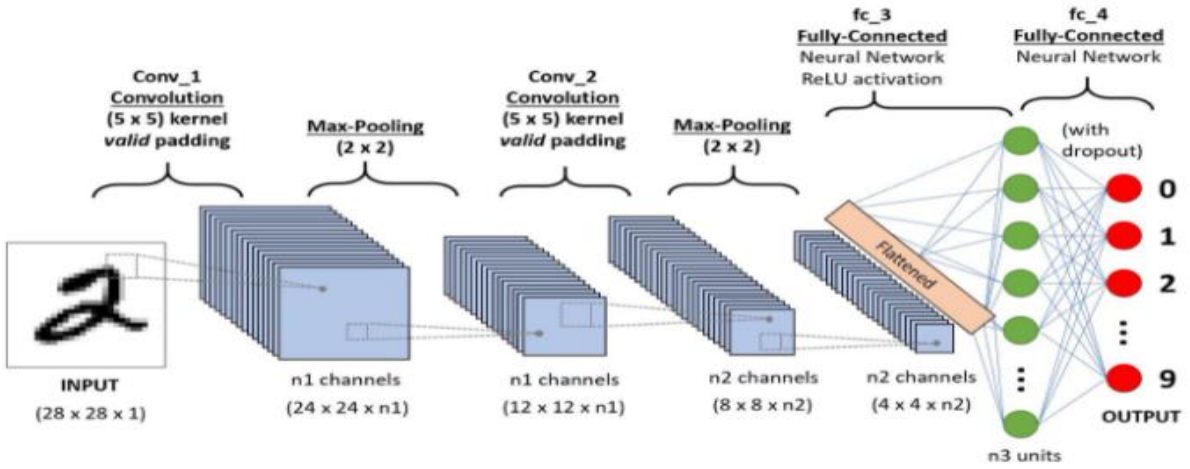
1.1 Ανίχνευση αντικειμένων και βαθιά μηχανική μάθηση

1.1.1 Εισαγωγή

Οι άνθρωποι όταν κοιτάζουν μια εικόνα μπορούν και αναγνωρίζουν αμέσως τα διάφορα αντικείμενα που βρίσκονται μέσα σε αυτή, όπως επίσης και τον τρόπο που αυτά αλληλοεπιδρούν. Στην περίπτωση που οι αλγόριθμοι επεξεργασίας εικόνων ήταν αρκετά ακριβείς και γρήγοροι, οι υπολογιστές θα ήταν σε θέση να κατευθύνουν τα οχήματα χωρίς την βοήθεια εξειδικευμένων αισθητήρων και συνεπώς να μεταφέρουν στο χρήστη πληροφορίες σε πραγματικό χρόνο.[Krizhevsky et al, 2012]

Συνεπώς, εάν αυτοί οι αλγόριθμοι είχαν τη δυνατότητα να φέρνουν εις πέρας εφαρμογές βαθιάς μάθησης (Deep Learning (DL)) με υψηλή αποτελεσματικότητα και εξαιρετική απόδοση όπως ο άνθρωπος, θα ήταν πραγματικά τεχνητά ευφυή. Ως εκ τούτου τα βασικά καθήκοντα της επεξεργασίας εικόνας είναι μια σειρά αναγνώρισης, ταξινόμησης και εν τέλει ανίχνευσης αντικειμένων, με σημαντικότερες προκλήσεις την ακρίβεια, την ταχύτητα και την πολυπλοκότητα [Juan Du, 2018].

Ένα συνελικτικό νευρωτικό δίκτυο (Convolutional Neural Network/CNN) είναι ένας αλγόριθμος βαθιάς μηχανικής μάθησης που είναι σε θέση να λαμβάνει μία εικόνα εισόδου, να αποδίδει σημασία σε διάφορες πτυχές και αντικείμενα σε αυτήν και να είναι σε θέση να διαφοροποιήσει το ένα από το άλλο. Η προεπεξεργασία που απαιτείται σε ένα CNN είναι σαφώς χαμηλότερη συγκριτικά με άλλους αλγόριθμους ταξινόμησης. Η αρχιτεκτονική ενός τέτοιου δικτύου είναι ανάλογη με αυτή του μοτίβου συνδεσιμότητας των νευρώνων στον ανθρώπινο εγκέφαλο, ενώ η εκπαίδευσή τους γίνεται μέσα από ένα σύνολο παραδειγμάτων, τέτοιων ώστε να μάθουν το περιβάλλον τους [Juan Du, 2018].

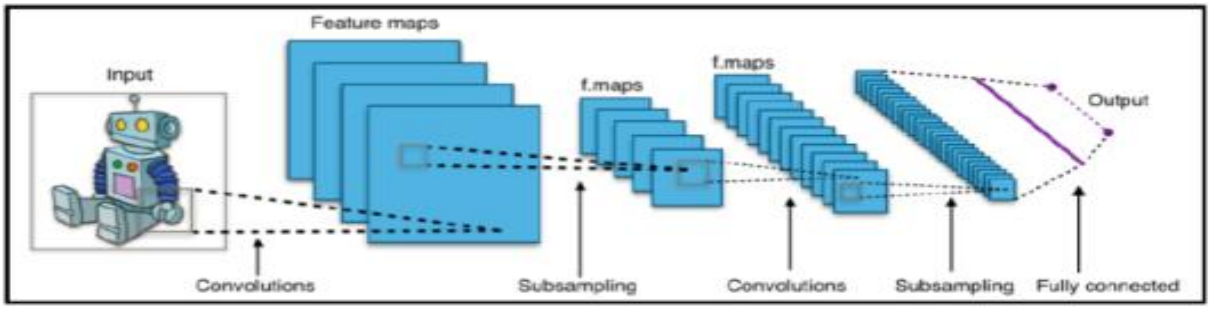


Σχήμα 1.1: Μια ακολουθία CNN για την ταξινόμηση χειρόγραφων ψηφίων (Πηγή: Sumit Saha, 2018)

1.1.2 Δομή ενός CNN

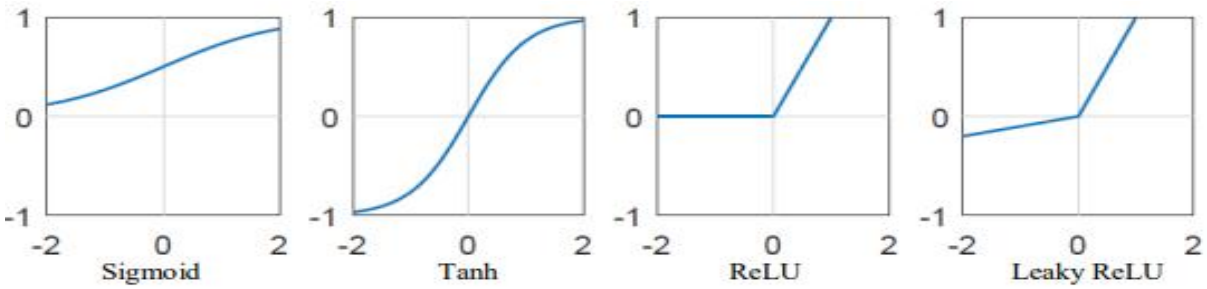
Όπως αναφέρθηκε παραπάνω τα Συνελικτικά Νευρωνικά Δίκτυα (CNN) είναι μια κατηγορία τεχνητών δικτύων βαθιάς τροφοδότησης που έχει χρησιμοποιηθεί για την παραγωγή υψηλής απόδοσης εφαρμογών στην όραση υπολογιστών, όπως η ταξινόμηση και η ανίχνευση εικόνων. Αναφορικά με τη δομή τους, αποτελούν ένα σύνολο από νευρώνες οι οποίοι εκτελούν συνέλιξη των προκαθορισμένων φίλτρων, με βάση την εικόνα σε μορφή διανύσματος, που δέχονται ως είσοδο. Το πλαίσιο των επιπέδων ενός τέτοιου δικτύου έχει διατηρηθεί από ένα απλό Νευρωτικό δίκτυο (NN) στο CNN, ενώ η λειτουργία και το δίκτυο έχουν αλλάξει. Τα συνελικτικά νευρωνικά δίκτυα όπως και τα νευρωνικά αποτελούνται από διάφορα επίπεδα τα οποία έχουν μεταξύ τους διασυνδεδεμένους νευρώνες και αντίστοιχα βάρη. [Pareky et al, 2014]

Στο σχήμα 1.2 παρουσιάζεται η δομή ενός CNN, η οποία αποτελείται από το συνελικτικό στρώμα, το στρώμα υποδειγματοληψίας και το πλήρως συνδεδεμένο στρώμα. Τα δύο πρώτα συνήθως εναλλάσσονται και το βάθος κάθε φίλτρου αυξάνεται από αριστερά προς τα δεξιά, ενώ το μέγεθος εξόδου (ύψος και πλάτος) μειώνεται. Το πλήρως συνδεδεμένο στρώμα αποτελεί το τελευταίο στάδιο και μοιάζει αρκετά με το αντίστοιχο ενός συμβατικού νευρωνικού δικτύου NN [Galvez et al, 2018].



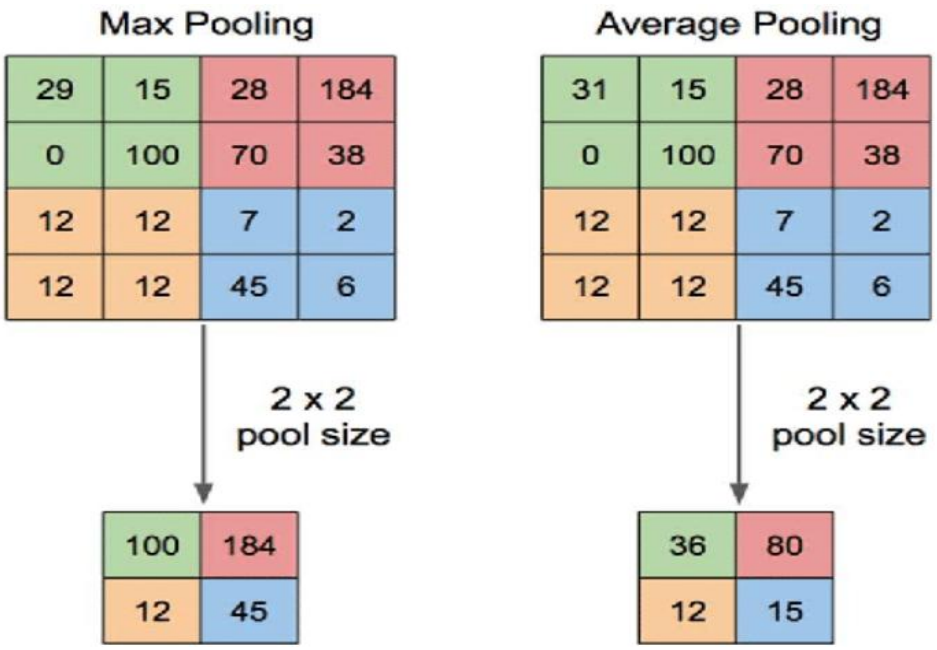
Σχήμα 1.2: Η δομή ενός CNN (Πηγή: Reagan L.Galvez, 2018)

Το συνελικτικό στρώμα αποτελεί τον πυρήνα ενός CNN και χρησιμοποιεί τον πίνακα kernel ως φίλτρο για την εξομάλυνση της αρχικής εικόνας. Οι τιμές κάθε εικονοστοιχείου των τοπικών συσχετισμένων δεδομένων εντός του φίλτρου, θα πολλαπλασιαστούν και θα προστεθούν ως τα αποτελέσματα της συνέλιξης. Αυτή η διαδικασία ονομάζεται συνέλιξη και χρησιμοποιείται για την εξαγωγή ορισμένων χαρακτηριστικών από μία εικόνα εισόδου με τη χρήση του kernel. Ως kernel αναφερόμαστε ουσιαστικά σε έναν πίνακα οποίος “ολισθαίνει” κατά κάποιο τρόπο στην εικόνα και πολλαπλασιάζεται με αυτή, έτσι ώστε να μας δοθεί στη συνέχεια ως στοιχείο εξόδου, βελτιωμένο κατά κάποιο συγκεκριμένο και προκαθορισμένο τρόπο. Το φιλτράρισμα διαφορετικών περιοχών της εικόνας με το ίδιο φίλτρο kernel, δηλαδή με κοινό βάρος, δίνει τη δυνατότητα τα ουδέτερα κελιά με παρόμοια χαρακτηριστικά να αναγνωριστούν με ευκολία και να κατηγοριοποιηθούν ως ίδιου τύπου αντικείμενα. Στη συνέχεια χρησιμοποιείται η συνάρτηση ενεργοποίησης, που είναι υπεύθυνη για τη μετατροπή της αθροιστικής σταθμισμένης εισόδου από τον κόμβο, στην ενεργοποίησή του κόμβου ή της εξόδου για αυτή την είσοδο. Υπάρχουν αρκετές συναρτήσεις για αυτό το σκοπό όπως, η σιγμοειδής συνάρτηση, η συνάρτηση υπερβολής καθώς και η συνάρτηση διορθωμένης γραμμικής μονάδας ReLu, γνωστή και ως συνάρτηση ράμπας. Η ReLu είναι η πιο διαδομένη από τις παραπάνω καθώς χρησιμοποιείται όλο και περισσότερο στα κρυφά επίπεδα των συνελικτικών νευρωνικών δικτύων, καθώς επιτυγχάνουν καλύτερη εκπαίδευση. Ουσιαστικά, αυξάνει τις μη γραμμικές ιδιότητες της συνάρτησης ενεργοποίησης και του συνολικού δικτύου χωρίς όμως να επηρεάζει τα δεκτικά πεδία του συνελικτικού επιπέδου. Οι συναρτήσεις ενεργοποίησης είναι πάντα μη γραμμικές ώστε το δίκτυο να μπορέσει να μάθει μη γραμμικές σχέσεις μεταξύ των εισόδων του [Simoyan et al, 2015].



Σχήμα 1.3: Οι βασικές τέσσερις συναρτήσεις ενεργοποίησης (Πηγή: Παπαδόπουλος Αθανάσιος, 2018)

Στη συνέχεια, μετά το συνελικτικό στρώμα ακολουθεί το στρώμα υποδειγματοληψίας στο οποίο μειώνεται το μέγεθος της εικόνας που είχε προκύψει από την συνέλιξη. Κάτι τέτοιο αποσκοπεί στην πιο γρήγορη σύγκλιση και γενικοποίηση του δικτύου αλλά και στην αντιμετώπιση χωρικών μετατροπών και μετακινήσεων. Σε περίπτωση που η συνέλιξη δεν λαμβάνει χώρα σε συνεχόμενα εικονοστοιχεία αλλά με βήματα 1 ή 2 εικονοστοιχείων θα υπάρξει ως αποτέλεσμα εικόνα εξόδου κατά 2 ή 4 φορές μικρότερη από την αρχική. Σε περίπτωση που ένα εικονοστοιχείο με μεγάλη τιμή τύχει να βρεθεί σε σημείο που η υποδειγματοληψία προσπερνάει τότε η ενεργοποίησή του θα χαθεί. Ως εκ τούτου δημιουργήθηκαν δύο τεχνικές η *average pooling* στην οποία λαμβάνουμε τον μέσο όρο μιας περιοχής στην οποία κάνουμε υποδειγματοληψία και η *max pooling* στην οποία λαμβάνουμε τη μέγιστη τιμή της περιοχής στην οποία κάνουμε υποδειγματοληψία.[Yani, 2019]



Σχήμα 1.4: Max pooling και Average pooling (Πηγή: Yani, 2019)

Το τελευταίο στρώμα ενός CNN αποτελούν τα πλήρως συνδεδεμένα τμήματα τα οποία μεταδίδουν τα δεδομένα στην έξοδο, καθώς και απλοποιούν και επιταχύνουν τον υπολογισμό τους. Ορισμένα από τα μοντέλα CNN εξακολουθούν και διαθέτουν και επίπεδα αποχώρησης και παλινδρόμησης. Τα επίπεδα αποχώρησης χρησιμοποιούνται για να επιλύουν την υπερπροσαρμογή. Προκειμένου να αποφευχθεί το βάρος να είναι πολύ υποκειμενικό, συνήθως ενημερώνει το βάρος του κόμβου των νευρωτικών κυττάρων με μια ορισμένη πιθανότητα που αποφασίζεται από τη στοχαστική πολιτική. Τα στρώματα παλινδρόμησης αποσκοπούν στην ταξινόμηση των χαρακτηριστικών. Υπάρχουν αρκετές μέθοδοι παλινδρόμησης όπως: Λογιστική Παλινδρόμηση (LR), Γκαουσιανές Διεργασίες για Παλινδρόμηση (GPR) και Bayesian Linear Regression (BLR). Η παλινδρόμηση δίνει τις πιθανότητες όλων των πιθανών τύπων αντικειμένων, οι οποίες αποτελούν και τη το βασικό κριτήριο [Juan Du, 2018].

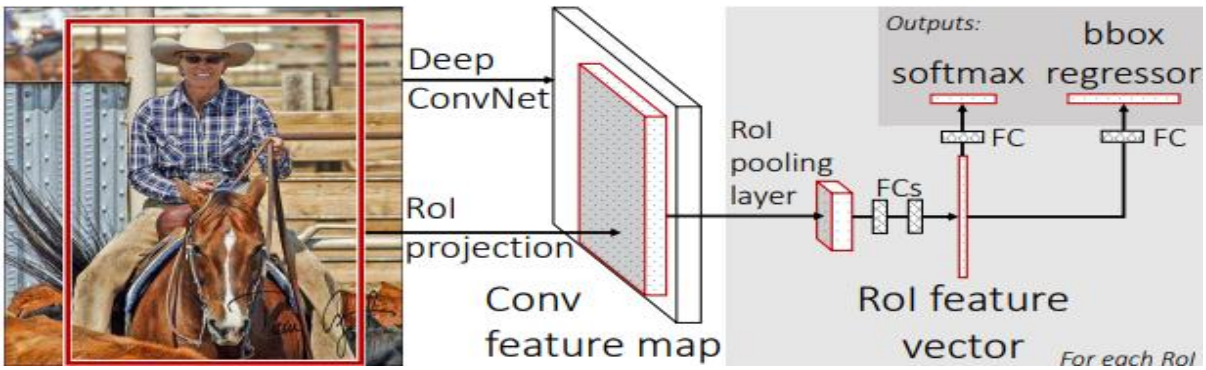
1.1.3 Οικογένεια των Συνελικτικών Νευρωνικών Δικτύων (CNN)

Η διαδικασία ανάπτυξης των CNN αποσκοπεί στη δημιουργία ενός πιο σταθερού και αποτελεσματικού συστήματος για την ανίχνευση αντικειμένων. Στο πλαίσιο αυτό έχουν προταθεί εκτός του Region CNN, και διάφορες εκδοχές του όπως τα Fast Region with CNN (Fast R-CNN) και Faster Region CNN (Faster R-CNN). Παρακάτω παρουσιάζονται ορισμένα χαρακτηριστικά αυτών των αλγορίθμων.[Girshick, 2015]

R-CNN: Αποτελεί ένα σημαντικό ορόσημο του CNN για την ανίχνευση αντικειμένων. Είναι το πρώτο νευρωτικό δίκτυο που διατυπώνει την “πρόταση περιοχής” για την υλοποίηση της ανίχνευσης αντικειμένων, με βάση την εξαιρετική ικανότητα του CNN να εξάγει χαρακτηριστικά και τα ταξινομεί. Η “πρόταση περιοχής” επιλέγει τα πιθανά αντικείμενα με κριτήριο τα διαφορετικά πλάτη και ύψη, όπως η επιλεκτική αναζήτηση. Πριν από το CNN, το R-CNN εφαρμόζει την διαδικασία crop/warp με σκοπό την κανονικοποίηση των υποψήφιων εικόνων ώστε να έχουν σταθερό μέγεθος με βάση το πρότυπο. Μετά το CNN, το

R-CNN προσθέτει SVM ταξινόμηση και παλινδρόμηση οριοθετημένου κουτιού για να λάβει το ακριβές αποτέλεσμα ανίχνευσης αντικειμένων. Ο SVM (Support Vector Machine), ή αλλιώς μηχανή διανυσματικής υποστήριξης είναι ένας εποπτευόμενος αλγόριθμος μηχανικής μάθησης που μπορεί να χρησιμοποιηθεί για προβλήματα ταξινόμησης ή παλινδρόμησης. Χρησιμοποιεί μία τεχνική που ονομάζεται Kernel trick για να μεταμορφώσει τα δεδομένα και στη συνέχεια με βάση αυτούς τους μετασχηματισμούς βρίσκει ένα βέλτιστο όριο μεταξύ των πιθανών εξόδων. Βασικό μειονέκτημα της "πρόταση περιοχής" είναι ότι κάνει το R-CNN να καταναλώνει μαζικά πολλά δεδομένα και συνεπώς ενέργεια και πολύ χρόνο. Ως εκ τούτου υπάρχουν σημαντικά περιθώρια βελτίωσης και των δύο παραγόντων στο R-CNN.[Denton et al, 2014]

Fast R-CNN: Ένα Fast R-CNN δέχεται ως είσοδο μια ολόκληρη εικόνα και ένα σύνολο προτάσεων αντικειμένων. Το δίκτυο επεξεργάζεται πρώτα ολόκληρη την εικόνα με διαφορετικά συνελικτικά (conv) και max pooling στρώματα για την παραγωγή ενός χάρτη χαρακτηριστικών conv. Στη συνέχεια, για κάθε πρόταση αντικειμένου ένα στρώμα συγκέντρωσης περιοχής ενδιαφέροντος (RoI) εξάγει ένα διάνυσμα χαρακτηριστικών σταθερού μήκους από τον χάρτη χαρακτηριστικών. Κάθε διάνυσμα χαρακτηριστικών τροφοδοτείται σε μια ακολουθία πλήρως συνδεδεμένων (fc) στρωμάτων εξόδου: ένα που παράγει εκτιμήσεις πιθανότητας softmax πάνω σε K κλάσεις αντικειμένων συν μια γενική κλάση "φόντου" και ένα άλλο επίπεδο που εξάγει τέσσερις αριθμούς πραγματικών τιμών για κάθε μια από τις K κλάσεις αντικειμένων. Κάθε σύνολο 4 τιμών κωδικοποιεί εκλεπτυσμένες θέσεις του πλαισίου οριοθέτησης (bounding-box) για κάθε μία από τις K κλάσεις. [Denton et al, 2014]



Σχήμα 1.5: Αρχιτεκτονική ενός Fast R-CNN (Πηγή: Girshick, 2015)

Faster R-CNN: Με βάση το Fast R-CNN, το Faster R-CNN επιλύει το πρόβλημα της "πρότασης περιοχής", προσθέτοντας RPN, η οποία αποτελεί και τη βασική συνεισφορά του Faster R-CNN. Κάνει χρήση της "πρότασης περιοχής" όχι στην αρχική εικόνα αλλά στην τελική εικόνα χαρακτηριστικών που θα εισαχθεί στη συγκέντρωση (RoI). Καθώς η ανάλυση της εικόνας χαρακτηριστικών είναι χαμηλότερη από εκείνη της αρχικής εικόνας, ο χρόνος υπολογισμού του Faster R-CNN είναι σίγουρα πολύ μικρότερος από αυτόν όλων των προηγούμενων μοντέλων CNN. [Girshick, 2015]

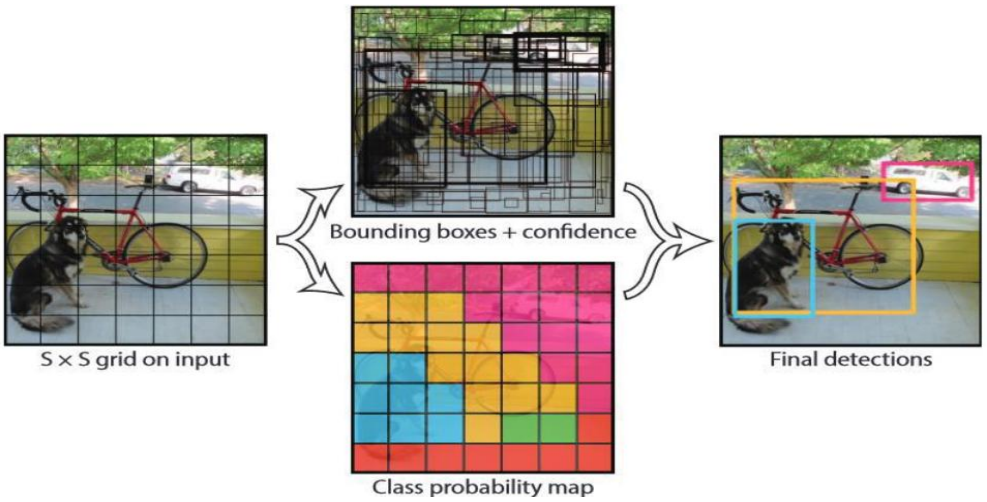
Το βασικό χαρακτηριστικό του Δικτύου Προτάσεων Περιοχής - Region Proposal Network (RPN) είναι η λήψη της πρότασης με ολίσθηση. Κάθε μία από τις συρόμενες προτάσεις θα

παράγει 9 υποψήφιες “άγκυρες” με διαφορετικές κλίμακες, πλάτη και ύψη. Στο Faster R-CNN η πορεία εξαγωγής των χαρακτηριστικών των “αγκυρών” είναι παρόμοια με αυτή του Fast R-CNN, ενώ η ταξινόμηση των αντικειμένων γίνεται μόνο ώστε να διαπιστωθεί εάν αυτά είναι προσκήνιο ή φόντο. Επίσης η πρόταση παλινδρόμησης του RPN , έχει στόχο να ανακαλύψει μια ακριβέστερη θέση του αντικειμένου-στόχου. Για κάθε θέση πρότασης, το RPN χρησιμοποιεί δύο πλήρως συνδεδεμένα στρώματα (ταξινόμηση αντικειμένου και παλινδρόμηση πρότασης) για να κρίνει και να απορρίψει τις “άγκυρες¨. Δεν κάνει ποτέ ρητή περιφερειακή πρόταση. [Girshick, 2015]

Οι βασικοί κανόνες επιλογής των “αγκυρών” είναι οι εξής: (1) η απόρριψη των αγκυρών στα όρια - (2) οι άγκυρες των οποίων η περιοχή επικάλυψης με το δείγμα είναι μεγαλύτερη από 0,7 θα ταξινομούνται ως προσκήνιο, και εκείνες των οποίων η επικάλυψη είναι μικρότερη από 0,3 θα ταξινομούνται ως φόντο. Με αυτό το τρόπο, το RPN επιλέγει περίπου 300 “άγκυρες” για κάθε ολισθαίνουσα πρόταση. Το Faster R-CNN χρησιμοποιεί την εναλλασσόμενη λειτουργία εκπαίδευσης για την εκπαίδευση των κοινών χαρακτηριστικών, καθώς και εξάγει τις σωστές προτάσεις από το σύνολο των δεδομένων εκπαίδευσης. Το Faster R-CNN έχει ήδη προσφέρει αποτελέσματα με τέλεια ακρίβεια αναγνώρισης. Ως εκ τούτου, το μόνο χαρακτηριστικό του που χρήζει περαιτέρω βελτίωσης είναι η ταχύτητα, η οποία αποτελεί και τον κύριο λόγο για τον οποίο δημιουργήθηκαν και άλλοι αλγόριθμοι στη συνέχεια [Juan Du, 2018].

1.2 YOLO (You Only Look Once)

Μια νέα προσέγγιση για την ανίχνευση αντικειμένων ονομάζεται You Only Look Once (YOLO) που σημαίνει ότι σε μια δεδομένη εικόνα μπορεί να προβλεφθεί η κατηγορία των αντικειμένων που προβάλλονται, καθώς και το πού αυτά βρίσκονται με “μια ματιά”. Όπως και η πρώτη μέθοδος που απορρίπτει εντελώς τον αγωγό, έτσι και η YOLO πλαισιώνει την ανίχνευση αντικειμένων ως ένα πρόβλημα παλινδρόμησης σε χωρικά διαχωρισμένα πλαίσια οριοθέτησης (bounding boxes) που ανήκουν σε διάφορες σχετικές κλάσεις, οι οποίες προβλέπονται με ένα μόνο νευρωνικό δίκτυο. Η YOLO χρησιμοποιεί ως βασικό δίκτυο το GoogLeNet και όχι το VGG-16 καθώς ο ρυθμός μεταφοράς δεδομένων σε σχέση με την ακρίβεια είναι σαφώς μεγαλύτερος στο πρώτο. Συνεπώς ο αλγόριθμος της YOLO είναι πολύ γρήγορος από τον σχεδιασμό του και μάλιστα σε πραγματικό χρόνο, διατηρώντας παράλληλα και υψηλή ακρίβεια. Η βασική ιδέα της YOLO παρουσιάζεται στο ***Σχήμα 1.6.*** [Choi et al, 2019]



Σχήμα 1.6: Κύρια ιδέα λειτουργίας της YOLO (Πηγή: Zhong-Qui Zhao et al. 2019)

- ***YOLO v1***

Η βασική εκδοχή της YOLO που ονομάζεται και YOLO v1, μοντελοποιεί την ανίχνευση αντικειμένων ως πρόβλημα παλινδρόμησης. Ένα και μόνο ενιαίο συνελικτικό δίκτυο προβλέπει ταυτόχρονα πολλαπλά πλαίσια οριοθέτησης και πιθανότητες κλάσης για τα πλαίσια αυτά.

Η YOLO v1 διαιρεί την εικόνα εισόδου σε ένα πλέγμα S x S. Σε περίπτωση που το κέντρο ενός αντικειμένου εμπίπτει σε ένα κελί πλέγματος, αυτό το κελί είναι υπεύθυνο για την ανίχνευση του αντικειμένου. Κάθε κελί πλέγματος προβλέπει B πλαίσια οριοθέτησης, βαθμολογίες εμπιστοσύνης για κάθε ένα από αυτά και πιθανότητες κλάσης C του πλέγματος. Έπειτα αυτές οι προβλέψεις κωδικοποιούνται ως τανυστής S x S x (B x 5 + C). Κατά τη διαδικασία δοκιμής, ο αλγόριθμος πολλαπλασιάζει τις πιθανότητες των υπό συνθήκη κλάσεων και τις προβλέψεις εμπιστοσύνης των επιμέρους πλαισίων, οι οποίες μας δίνουν την εμπιστοσύνη για κάθε κλάση και προκύπτει έτσι η βαθμολογία για κάθε κουτί:

$$\mathbf{Pr(Class_i|Object) \times Pr(Object) \times IOU^{truth}_{pred} = Pr\,(Class_i) \times IOU^{truth}_{pred}}$$

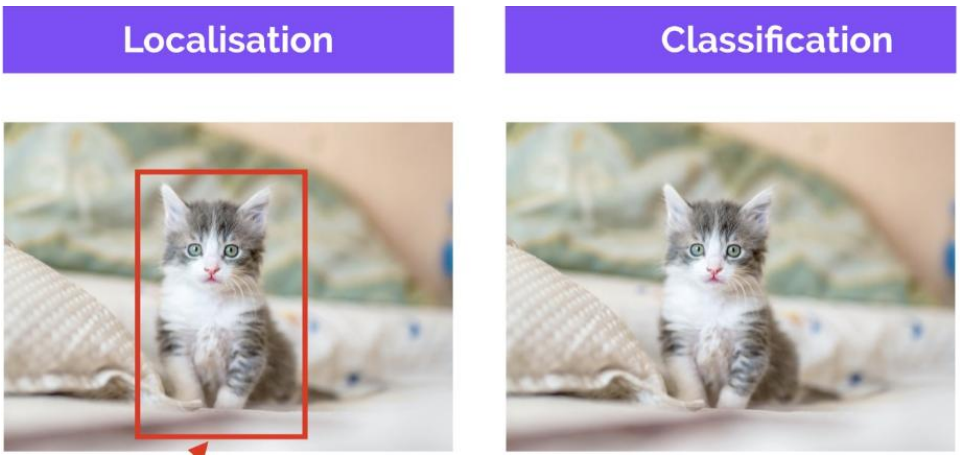
(IOU: Intersection Over Union). Οι βαθμολογίες κωδικοποιούν τόσο την πιθανότητα της συγκεκριμένης κατηγορίας να εμφανιστεί στο πλαίσιο όσο και το κατά πόσο το προβλεπόμενο αυτό πλαίσιο ταιριάζει στο αντικείμενο. Κάθε ένα από τα πλαίσια οριοθέτησης αποτελείται από 5 προβλέψεις και την εμπιστοσύνη. Οι συντεταγμένες αντιπροσωπεύουν το κέντρο του πλαισίου σε σχέση με τα όρια του πλέγματος. Το πλάτος και το ύψος προβλέπονται σε σχέση με ολόκληρη την εικόνα και αυτός είναι και ο λόγος για τον οποίο η YOLO v1 χρησιμοποιεί για υπολογισμό τον τανυστή B x 5. Για την αξιολόγηση του YOLO στην Pascal VOC (Visual Object Classes), χρησιμοποιούνται συνήθως S = 7, B = 2. Η Pascal VOC έχει 20 διακριτές κλάσεις με ετικέτες, οπότε C = 20. Η τελική πρόβλεψη της YOLO v1 είναι ένας τανυστής 7 x 7 x (5 x 2 + 20) = 7 x 7 x 30. Χρησιμοποιεί μόνο 98 πλαίσια οριοθέτησης ανά εικόνα έναντι 2000 από την επιλεκτική αναζήτηση.[Redmon et al, 2016]

Το δίκτυο της YOLO v1 έχει 24 συνελικτικά στρώματα, ακολουθούμενα από 2 πλήρως συνδεδεμένα στρώματα. Αντί για τις μονάδες έναρξης (inception modules) που χρησιμοποιεί το GoogLeNet, η YOLO v1 χρησιμοποιεί απλά ένα 1 x 1 στρώμα μείωσης ακολουθούμενο από 3 x 3 στρώματα συνελικτικής επεξεργασίας, παρόμοια με τους Lin et al. Ως μονάδα έναρξης (inception module), αναφερόμαστε σε ένα μπλοκ μοντέλου εικόνας που στοχεύει στην προσέγγιση μιας βέλτιστης τοπικής αραιής δομής σε ένα CNN [Juan Du, 2018].

Με απλά λόγια, μας επιτρέπει να χρησιμοποιούμε πολλαπλούς τύπους μεγέθους φίλτρου, αντί να περιοριζόμαστε σε ένα μόνο μέγεθος φίλτρου σε ένα μπλοκ εικόνας, το οποίο στη συνέχεια συνδυάζουμε και περνάμε στο επόμενο επίπεδο. Στην Pascal VOC2007, η YOLO v1 επεξεργάζεται εικόνες με 45 καρέ ανά δευτερόλεπτο (FPS), δηλαδή δύο έως εννέα φορές ταχύτερα από το Faster R-CNN. Ειδικότερα, η Fast YOLO, μια γρήγορη έκδοση της YOLO που έχει σχεδιαστεί ώστε να αυξάνει περαιτέρω την ταχύτητα ανίχνευσης αντικειμένων, φτάνει έως και τα 155 FPS. [Felzenszwalb et al,2010]

- ***Μέση αντιπροσωπευτική ακρίβεια mAP***

Στην όραση υπολογιστών, το mAP είναι ένα δημοφιλές μέτρο αξιολόγησης που χρησιμοποιείται για την ανίχνευση αντικειμένων, δηλαδή τον εντοπισμό και την ταξινόμηση. Ο εντοπισμός προσδιορίζει τη θέση μιας περίπτωσης (συντεταγμένες πλαισίου οριοθέτησης) και η ταξινόμηση μας λέει τί ουσιαστικά είναι το αντικείμενο που περικλείει (π.χ γάτα ή σκύλος). Πολλοί αλγόριθμοι ανίχνευσης αντικειμένων όπως οι Faster R-CNN, MobileNet SSD και YOLO, χρησιμοποιούν το mAP για την αξιολόγηση των μοντέλων τους. Η μέση αντιπροσωπευτική ακρίβεια μετράει το πόσο ακριβείς είναι οι προβλέψεις του εκάστοτε μοντέλου, δηλαδή το ποσοστό των προβλέψεων που είναι σωστές. [Yohanandan, 2020]



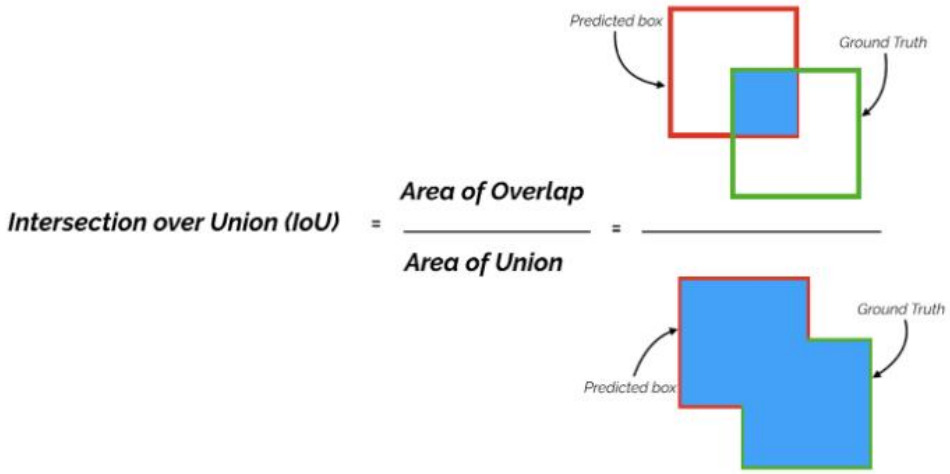
Εικόνα 1.1: Ταξινόμηση και εντοπισμός αντικειμένου σε εικόνα (Πηγή: https://towardsdatascience.com/map-mean-average-precision-might-confuse-you-5956f1bfa9e2)

$$\textbf{\textit{Precision (ακρίβεια)} = TP / (TP + FP)}$$

TP = True Positives (προβλέφθηκε τόσο θετικό όσο και σωστό)

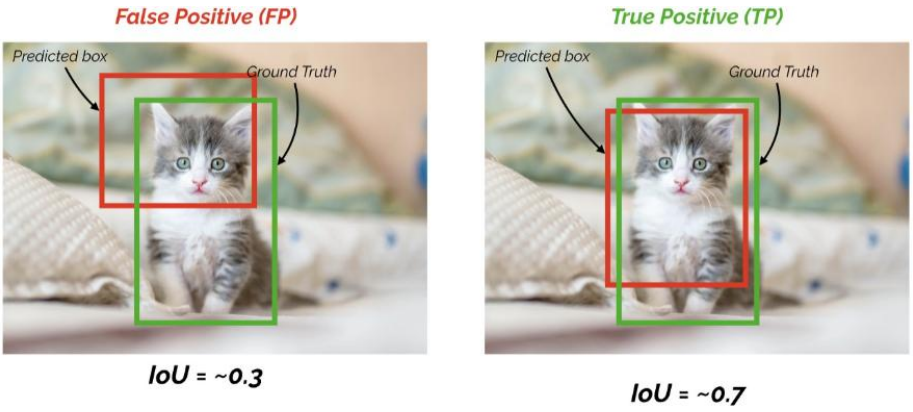
FP = False Positives (προβλέφθηκε ως θετικό αλλά ήταν λανθασμένο)

Τα συστήματα ανίχνευσης αντικειμένων κάνουν προβλέψεις με βάση ένα πλαίσιο οριοθέτησης και μια ετικέτα κλάσης. Για κάθε πλαίσιο οριοθέτησης, μετράμε την επικάλυψη του προβλεπόμενου πλαισίου οριοθέτησης και του πλαισίου οριοθέτησης της βασικής αλήθειας (ground truth). Αυτή υπολογίζεται μέσω της συνάρτησης IoU (intersection over union).



Σχήμα 1.7: Σχηματική απεικόνιση της IoU (Πηγή: https://towardsdatascience.com/map-mean-average-precision-might-confuse-you-5956f1bfa9e2)

Για εργασίες ανίχνευσης αντικειμένων, υπολογίζουμε την ακρίβεια και την ανάκληση χρησιμοποιώντας την τιμή της IoU για ένα δεδομένο κατώφλι IoU. Για παράδειγμα, εάν το κατώφλι IoU είναι 0,5 και η τιμή IoU για μια πρόβλεψη είναι 0,7, τότε ταξινομούμε την πρόβλεψη ως αληθώς θετική (True Positive, TP). Από την άλλη, εάν η τιμή IoU είναι 0,3, την ταξινομούμε ως ψευδώς θετική (False Positive, FP). Αυτό σημαίνει επίσης ότι για μια πρόβλεψη, ενδέχεται να λάβουμε διαφορετικά δυαδικά TRUE ή FALSE θετικά, αλλάζοντας το όριο του IoU



Εικόνα 1.2: (Πηγή:https://towardsdatascience.com/map-mean-average-precision-might-confuse-you-5956f1bfa9e2)

Ένας ακόμη όρος που πρέπει να αποσαφηνιστεί είναι η ανάκληση (Recall). Η ανάκληση μετράει πόσο καλά βρίσκονται όλα τα θετικά στοιχεία. Για παράδειγμα, μπούμε να βρούμε το 80% των πιθανών θετικών περιπτώσεων στις Κ προβλέψεις μας.

$$\textit{Recall (ανάκληση)} = TP / (TP + FN)$$

TP = True Positives (αληθώς θετικό)

FP = False Positives (ψευδώς θετικό)

Ο γενικός ορισμός για την Average Precision AP (μέση ακρίβεια) είναι η εύρεση της περιοχής κάτω από την καμπύλη ακρίβειας-ανάκλησης. Η τιμή mAP (μέση αντιπροσωπευτική ακρίβεια) είναι ο μέσος όρος της της AP. Σε ορισμένα πλαίσια, η AP υπολογίζεται για κάθε κλάση και κατόπιν υπολογίζεται ο μέσος όρος για να προκύψει τελικά η mAP. Σε ορισμένες ωστόσο περιπτώσεις, όπως είναι η αξιολόγηση της πρόκλησης COCO σημαίνουν το ίδιο πράγμα , οπότε και δεν υπάρχει ουσιαστική διαφορά μεταξύ των AP και mAP. Η AP υπολογίζεται για όλες τις υπάρχουσες κατηγορίες, ενώ η mAP υπολογίζεται λαμβάνοντας τη μέση AP για όλες τις κατηγορίες ή/και τα συνολικά κατώφλια IoU, ανάλογα με τις διάφορες προκλήσεις ανίχνευσης που υπάρχουν. [Yohanandan, 2020]

- ***YOLO v2***

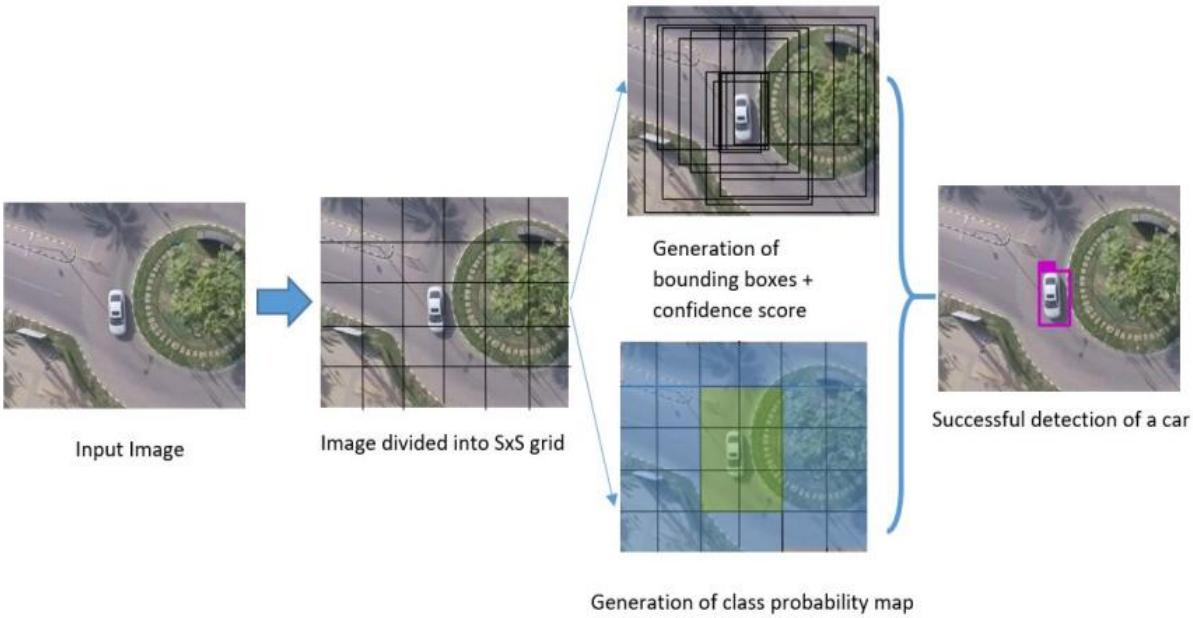
Η YOLO Version 2 (YOLO v2) είναι μία ποικιλοτρόπως βελτιωμένη εκδοχή του πρώτου μοντέλου της YOLO που διατηρεί το πλεονέκτημα στην ταχύτητα και προσπαθεί να αυξήσει την τιμή mAP (Mean Average Precision) από το 63,4 της YOLO v1. Χρησιμοποιώντας μια νέα πολλαπλής κλίμακας μεθόδου εκπαίδευσης, το μοντέλο αυτό μπορεί να τρέξει σε διαφορετικά μεγέθη, προσφέροντας μια καλή αντιστάθμιση μεταξύ της ταχύτητας και της ακρίβειας. Η YOLO v2 προσφέρει ένα ευρύ κατάλογο σημαντικών λύσεων για την αύξηση της τιμής του mAP. Με τη χρήση της μεθόδου κανονικοποίησης παρτίδας (γνωστή και ως batch norm), γίνεται προεπεξεργασία των δεδομένων εισόδου. Η εν λόγο μέθοδος (στα Αγγλικά, Batch normalization) είναι μία μέθοδος που χρησιμοποιείται για να κάνει τα τεχνητά νευρωνικά δίκτυα πιο γρήγορα και πιο σταθερά μέσω της κανονικοποίησης των εισόδων των στρωμάτων με εκ νέου επαναπροσδιορισμό του κέντρου και της κλίμακας. Στη συνέχεια ο ταξινομητής υψηλής ανάλυσης από τη YOLO v1 γίνεται από 224 x 224 σε 448 x 448 και αυξάνει τη τιμή mAP κατά 4%. Η YOLO v2 χρησιμοποιεί επίσης ένα νέο δίκτυο με τη λογική του "δίκτυο σε δίκτυο", το οποίο κάνει πρόβλεψη με τη βοήθεια του παγκόσμιου μέσου όρου συγκέντρωσης (global average pooling) και συμπιέζει τα χαρακτηριστικά τοποθετώντας τον συνελικτικό πυρήνα μεταξύ των άλλων 3 συνελικτικών πυρήνων. Το νευρωτικό του δίκτυο έχει λιγότερα συνελικτικά επίπεδα (19 αντί για 24 που έχει η YOLO v1) και λιγότερα φίλτρα συν 5 στρώματα συγκέντρωσης μέγιστης τιμής. Εκτός αυτού η YOLO v2 υιοθετεί συνελεκτικά με πλαίσια αγκύρωσης και παράλληλα αυξάνει την ανάλυση του κάθε πλέγματος σε σύγκριση με τη YOLO v1 από 7 x 7 σε 13 x 13, έχοντας μόνο ένα πλαίσιο οριοθέτησης για κάθε πλέγμα. Επιπλέον, μετά από έρευνα της καμπύλης συνάρτησης IOU (Intersection Over Union) με μέτρηση των στατιστικών στοιχείων Ground Truth των τιμών VOC και COCO με τον αλγόριθμο K-means, η YOLO v2 βρίσκει την καλύτερη λύση για την ποσότητα των κουτιών αγκύρωσης "Dimension Clusters" και τελικά

αποφασίζει ο αριθμός αυτών να είναι 5, των οποίων η μέση τιμή της IOU (61.0) να είναι ισοδύναμη με εκείνη του Faster R-CNN (60.9) με 9 κουτιά αγκύρωσης. Εν τω μεταξύ, η πρόβλεψη θέσης χρησιμοποιείται επίσης για να μετριάσει την αστάθεια που προκαλούν τα κουτιά αγκύρωσης, και τελικά αυξάνει την τιμή mAP κατά 5%. [Felzenszwalb et al,2010]

Τέλος η YOLO v2 προσθέτει ένα επίπεδο διέλευσης για να πάρει τα εξαγόμενα χαρακτηριστικά από το προηγούμενο 26 x 26 στρώμα και να τα συνδυάσει με τα τελικά χαρακτηριστικά εξόδου, έτσι ώστε η ικανότητα ανίχνευσης των μικρών αντικειμένων να ενισχυθεί. Με τον τρόπο αυτό η YOLO v2 αυξάνει την τιμή mAP κατά 1% [Juan Du, 2018].

- ***YOLO v3***

Η YOLO v3 αποτελεί σημαντική βελτίωση σε σχέση με τους προκατόχους της YOLO v1 και YOLO v2 (που ονομάζεται επίσης και YOLO9000). Η πρώτη βελτίωση που έγινε με την YOLO v3 είναι η χρήση της ταξινόμησης πολλαπλών ετικετών, η οποία διαφέρει από την αμοιβαία αποκλειστική επισήμανση που χρησιμοποιούνταν στις προηγούμενες εκδόσεις.



Σχήμα 1.8: Αρχιτεκτονική της YOLO v3 (Πηγή: Benjdira et al., 2019)

Χρησιμοποιεί έναν υπολογιστικό ταξινομητή για να υπολογίζει την πιθανότητα το αντικείμενο να ανήκει σε μια συγκεκριμένη ετικέτα. Οι προηγούμενες εκδόσεις χρησιμοποιούν τη συνάρτηση softmax για τη δημιουργία των πιθανοτήτων από τις βαθμολογίες. Για την απώλεια ταξινόμησης, χρησιμοποιεί τη δυαδική απώλεια διασταυρούμενης εντροπίας για κάθε ετικέτα, αντί για το γενικό μέσο τετραγωνικό σφάλμα που χρησιμοποιούνταν στις προηγούμενες εκδόσεις. Η δεύτερη βελτίωση που έγινε είναι η χρήση διαφορετικής πρόβλεψης των οριοθετημένων πλαισίων. Συσχετίζει τη βαθμολογία αντικειμενικότητας 1 στην άγκυρα εκείνου του πλαισίου οριοθέτησης που επικαλύπτει ένα κομμάτι αντικειμένου περισσότερο από τα άλλα. [Benjdira et al., 2019]

Αγνοεί άλλες αγκυρώσεις πλαισίων που επικαλύπτουν το αντικείμενο περισσότερο από ένα επιλεγμένο κατώφλι. Συνεπώς, η YOLO v3 αναθέτει μια άγκυρα ενός πλαισίου οριοθέτησης για κάθε αντικείμενο. Η τρίτη βελτίωση που έγινε είναι η χρήση προβλέψεων σε όλες τις

κλίμακες κάνοντας χρήση της έννοιας των δικτύων πυραμίδας χαρακτηριστικών (FPN). Με αυτό το τρόπο η YOLO v3 προβλέπει κουτιά σε 3 διαφορετικές κλίμακες και στη συνέχεια εξάγει χαρακτηριστικά από αυτές τις κλίμακες. Το αποτέλεσμα της πρόβλεψης του δικτύου είναι ένας τρισδιάστατος τανυστής που κωδικοποιεί τα πλαίσια οριοθέτησης, τα score αντικειμενικότητας και τις προβλέψεις για τις διάφορες κλάσεις. Αυτός είναι και ο λόγος για τον οποίο οι διαστάσεις των τανυστών αλλάζουν από προηγούμενες εκδόσεις:

$$N \times N \times (3*(4 + 1 + C))$$

Όπου,

N x N: είναι ο αριθμός των κελιών του πλέγματος στο σύστημα

3: για την αποκωδικοποίηση των χαρακτηριστικών που εξάγονται από κάθε μία από τις 3 κλίμακες

4 + 1: για την αποκωδικοποίηση των μετατοπίσεων και των βαθμολογιών αντικειμενικότητας των οριοθετημένων πλαισίων

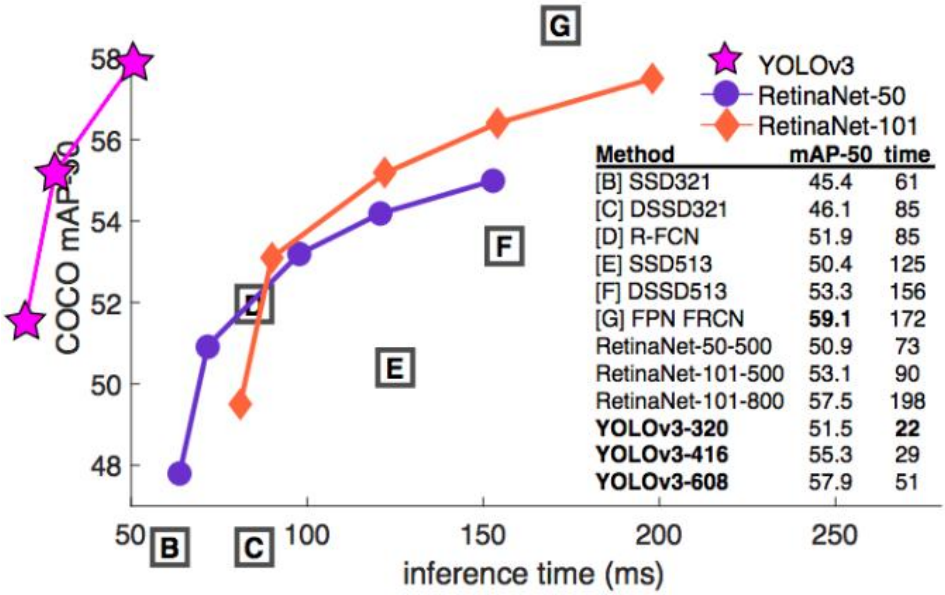
C: είναι ο αριθμός των κλάσεων στις οποίες εκπαιδεύουμε το δίκτυό μας

Αυτό επιτρέπει την απόκτηση καλύτερων σημασιολογικών πληροφοριών από τα χαρακτηριστικά και πιο λεπτομερείς πληροφορίες από τους προηγούμενους χάρτες χαρακτηριστικών. Η πέμπτη βελτίωση που έγινε είναι η νέα δυνατότητα του CNN να εξάγει χαρακτηριστικά με την ονομασία Darknet-53. Πρόκειται για ένα CNN 53 επιπέδων που χρησιμοποιεί δίκτυο παράλειψης συνδέσεων εμπνευσμένο από το ResNet. Επίσης χρησιμοποιεί 3 x 3 και 1 x 1 επίπεδα συνελίξεων, ενώ έχει ορισμένες συνδέσεις συντόμευσης και είναι σημαντικά μεγαλύτερο και πιο ισχυρό από το Darknet-19 της YOLO v2. [Redmon et al, 2018]

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Σχήμα 1.9: Darknet – 53 (Πηγή: Redmon et al. [127])

Η YOLO v3 έχει τις ίδιες επιδόσεις με άλλους σύγχρονους ανιχνευτές, όπως το RetinaNet, ενώ είναι σημαντικά ταχύτερος επιτυγχάνοντας mAP-50 55,3%. Είναι επίσης καλύτερος από τον SSD και τις διάφορες παραλλαγές του. Παρακάτω παρουσιάζεται μια σύγκριση των επιδόσεων της YOLO v3 με το RetinaNet και τις διάφορες παραλλαγές του SSD(test επιδόσεων).



Σχήμα 1.10: (Πηγή: https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b)

Στη συνέχεια παρουσιάζεται ένα διάγραμμα που δείχνει τη μέση ακρίβεια (AP) της ανίχνευσης μικρών , μεσαίων και μεγάλων αντικειμένων με διάφορους αλγορίθμους. Όσο υψηλότερη είναι η τιμή της AP, τόσο μεγαλύτερη είναι η ακρίβεια για τη συγκεκριμένη μεταβλητή. Η ακρίβεια για τα μικρά αντικείμενα στην YOLO v2 ήταν ασύγκριτη σε σχέση με άλλους αλγορίθμους λόγω του πόσο ανακριβής ήταν η YOLO στον εντοπισμό μικρών αντικειμένων. Με AP 5,0 μειονεκτούσε σημαντικά απέναντι σε άλλους αλγορίθμους όπως ο RetinaNet με AP = 21,8 ή το SSD513 με AP = 10,2 αντίστοιχα. Η YOLO v3 αύξησε σημαντικά το AP για τα μικρά αντικείμενα κατά 13,3 το οποίο αποτελεί τεράστια πρόοδο σε σχέση με την προηγούμενη έκδοση. Ωστόσο, η μέση ακρίβεια (AP) για όλα τα αντικείμενα (μικρά, μεσαία, μεγάλα) εξακολουθεί να είναι μικρότερη από το RetinaNet. [Redmon et al, 2018]

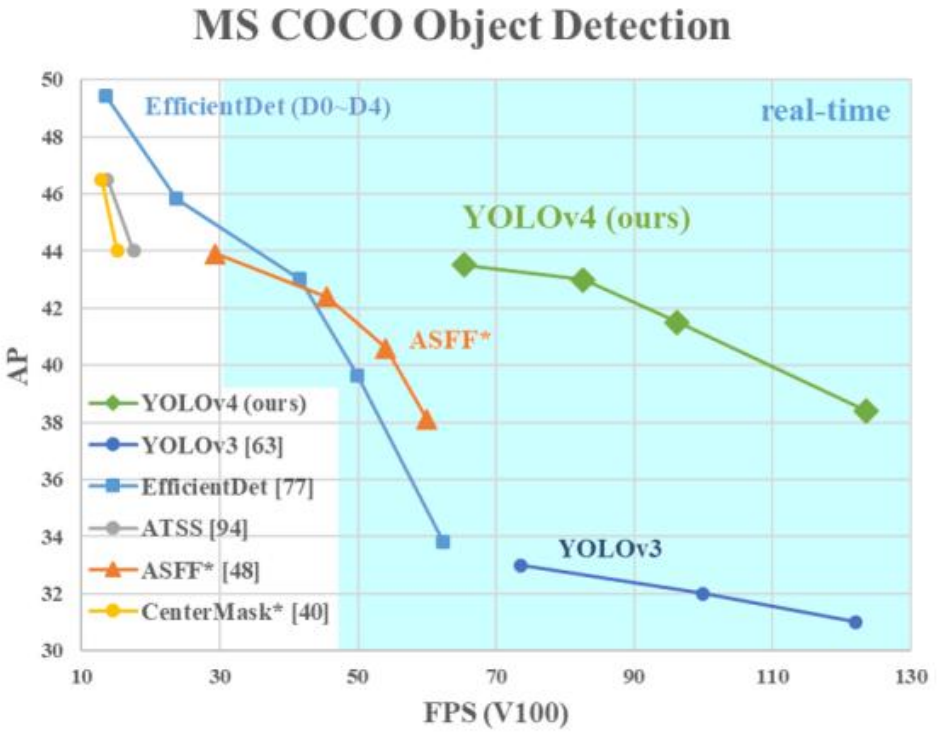
	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI	Inception-ResNet-v2	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2	DarkNet-19	21.6	44.0	19.2	5.0	22.4	35.5
SSD513	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2
YOLOv3 608×608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

Σχήμα 1.11: Σύγκριση της YOLO v3 με άλλους αλγορίθμους για διαφορετικά μεγέθη αντικειμένων, που μας δείχνει τη μέση ακρίβεια (AP) για AP-S (μικρό μέγεθος αντικειμένου), AP-M (μεσαίο μέγεθος αντικειμένου), AP-L (μεγάλο μέγεθος αντικειμένου). (Πηγή: Redmon et al, 2018)

- ***YOLO v4***

Η πλειοψηφία των ανιχνευτών αντικειμένων που βασίζονται στο CNN χρησιμοποιούνται κατά κανόνα μόνο σε συστήματα συστάσεων (cookies). Βελτιώνοντας την ακρίβεια των ανιχνευτών σε πραγματικό χρόνο επιτρέπεται η χρήση τους όχι μόνο για συστήματα συστάσεων που δημιουργούν υποδείξεις, αλλά επίσης και για αυτόνομη διαχείριση ποικίλων διαδικασιών με μείωση της ανθρώπινης επέμβασης. Η λειτουργία ενός ανιχνευτή αντικειμένων σε πραγματικό χρόνο σε συμβατικές μονάδες επεξεργασίας γραφικών (GPU) επιτρέπει τη μαζική τους χρήση σε προσιτή τιμή. Μέχρι πρότινος τα πιο ακριβή νευρωνικά δίκτυα δεν λειτουργούσαν σε πραγματικό χρόνο και απαιτούσαν μεγάλο αριθμό GPU για την εκπαίδευση τους. Αυτό το πρόβλημα αντιμετωπίστηκε με τη δημιουργία ενός CNN όπως είναι η YOLO v4, το οποίο αφενός λειτουργεί σε πραγματικό χρόνο, αφετέρου απαιτεί μόνο μια συμβατική GPU τόσο για τη εκπαίδευσή του όσο και για τη λειτουργία του. [Bochkovskiy et al. 2020]

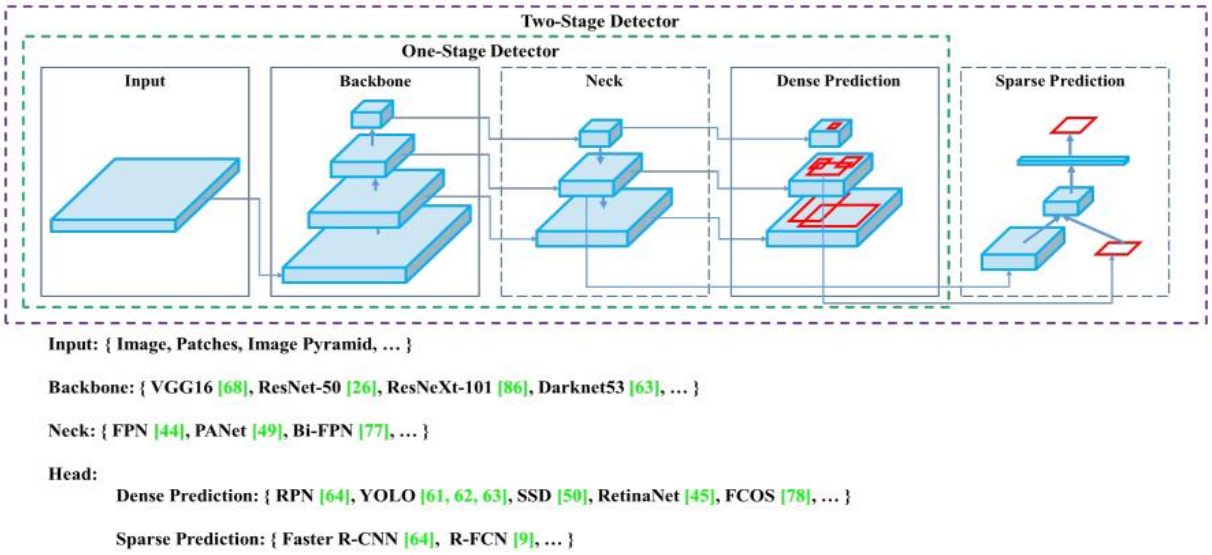
Η YOLO v4: Optimal Speed and Accuracy of Object Detection, δημιουργήθηκε από τους Alexey Bochkovskiy et al το 2020, με κύριο στόχο το σχεδιασμό ενός γρήγορου λειτουργικού ανιχνευτή αντικειμένων σε συστήματα παραγωγής και σε παράλληλους υπολογισμούς. Δημιούργησαν έτσι έναν αλγόριθμο που μπορεί μέσω μιας συμβατικής GPU να εκπαιδευτεί και να δώσει σε πραγματικό χρόνο state-of-the-art αποτελέσματα με μεγάλη ακρίβεια. Στο παρακάτω διάγραμμα όπως παρουσιάσθηκε από τους δημιουργούς φαίνονται τα αποτελέσματα ορισμένων αλγορίθμων ανίχνευσης συμπεριλαμβανομένου και αυτού της YOLO v4 όσον αφορά το σύνολο δεδομένων MS COCO, όπου και πετυχαίνει πολύ καλά αποτελέσματα με 43% περίπου AP στα 60+ FPS. Επίσης η YOLO v4 λειτουργεί δύο φορές ταχύτερα από τον EfficientDet, ενώ βελτιώνει και το AP και τα FPS της YOLO v3 κατά 10% και 12%, αντίστοιχα.



Σχήμα 1.12: Μέση ακρίβεια (average precision) και FPS (Πηγή: Bochkovskiy et al. 2020)

Όσον αφορά την αρχιτεκτονική της YOLO v4, αυτή αποτελείται από διάφορα μέρη, τα σημαντικότερα από τα οποία είναι τα εξής:

- Η είσοδος που έρχεται πρώτη και ουσιαστικά είναι αυτό που έχουμε ως σύνολο εικόνων εκπαίδευσης που θα τροφοδοτηθούν στο δίκτυο και οι οποίες επεξεργάζονται σε παρτίδες παράλληλα από την GPU.
- Το BackBone μαζί με το Neck τα οποία και κάνουν την εξαγωγή και συνάθροιση των χαρακτηριστικών. Τα Detection Neck και Detection Head μαζί μπορούν να ονομαστούν ως ανιχνευτής αντικειμένων.
- Το Head που κάνει την ανίχνευση και πρόβλεψη αφού είναι και το κυρίως υπεύθυνο τόσο για τον εντοπισμό όσο και για την ταξινόμηση.



Σχήμα 1.13: Αρχιτεκρονική της YOLO v4 (Πηγή: Bochkovskiy et al. 2020)

Οι δημιουργοί αρχικά θεώρησαν ως Backbone τα CSPResNext50, CSPDarknet53 και EfficientNet-B3. Τελικά όμως μετά από πολλές δοκιμές και πειραματικά αποτελέσματα επέλεξαν το CSPDarknet53 CNN. Το CSPDarknet53 βασίζεται στο σχέδιο DenseNet, το οποίο συνδέει τις προηγούμενες εισόδους με την τρέχουσα είσοδο πριν προχωρήσει στα πυκνά στρώματα κάτι που αναφέρεται ως μοτίβο συνδεσιμότητας Dense. Αποτελείται από δύο μπλοκ το συνελικτικό στρώμα βάσης (Convolutional Base Layer) και το μπλοκ Cross Stage Partial (CSP). [Guo et al, 2020]

Ως Neck αναφέρεται το τμήμα εκείνο όπου λαμβάνει χώρα η συγκέντρωση των χαρακτηριστικών. Συλλέγει χάρτες χαρακτηριστικών από τα διάφορα στάδια του Backbone και στη συνέχεια τους αναμειγνύει και τους συνδυάζει ώστε να τους προετοιμάσει για το επόμενο βήμα. [Han et al, 2020]

Ένα πρόσθετο μπλοκ που ονομάζεται SPP (Spatial Pyramid Pooling) προστίθεται μεταξύ του CSPDarkNet53 Backbone και του δικτύου συγκέντρωσης χαρακτηριστικών (PANet), με σκοπό να αυξηθεί το πεδίο υποδοχής και να διαχωριστούν τα πιο σημαντικά χαρακτηριστικά του πλαισίου. Το SPP δεν έχει καμία επίδραση στην ταχύτητα λειτουργίας του δικτύου ενώ επίσης συνδέεται με τα τελικά στρώματα των πυκνά συνδεδεμένων συνελικτικών στρωμάτων του CSPDarkNet. [Guo et al, 2020]

Ως πεδίο υποδοχής αναφερόμαστε στην περιοχή της εικόνας που εκτίθεται σε έναν πυρήνα ή φίλτρο κάθε φορά. Καθώς στοιβάζονται περισσότερα επίπεδα συνελίξεων, αυξάνεται γραμμικά ενώ αυξάνεται εκθετικά όταν στοιβάζουμε διεσταλμένες συνελίξεις φέρνοντας μη γραμμικότητα. Η YOLO v4 χρησιμοποιεί ένα τροποποιημένο δίκτυο συνάθροισης μονοπατιών (PANet), κυρίως ως σχεδιαστική βελτίωση, προκειμένου να καταστεί πιο κατάλληλο για εκπαίδευση σε μία GPU. Ο κύριος ρόλος του PANet είναι να βελτιώσει τη διαδικασία τμηματοποίησης διατηρώντας τις χωρικές πληροφορίες, οι οποίες με τη σειρά τους βοηθούν στο σωστό εντοπισμό των εικονοστοιχείων για την πρόβλεψη της μάσκας. Η επαύξηση διαδρομής από κάτω προς τα πάνω, η προσαρμοστική συγκέντρωση δυνατοτήτων

και η πλήρως συνδεδεμένη σύντηξη είναι σημαντικές ιδιότητες που τα καθιστούν τόσο ακριβή για την πρόβλεψη μάσκας. [Hariharan et al, 2015]

Η κύρια λειτουργία στο τμήμα του Head είναι ο εντοπισμός οριοθετημένων πλαισίων και η ταξινόμηση. Εντοπίζονται οι συντεταγμένες του πλαισίου οριοθέτησης (x,y, ύψος και πλάτος) καθώς και τα αποτελέσματα. Εδώ οι συντεταγμένες x,y αποτελούν το κέντρο του b-box εκφρασμένο σε σχέση με το όριο του κελιού πλέγματος. Το πλάτος και το ύψος προβλέπονται σε σχέση με ολόκληρη την εικόνα.

Δύο νέοι όροι που εισήχθησαν από τους δημιουργούς στην YOLO v4 ονομάζονται Bag of Freebies (BoF) και Bag of Specials (BoS).

- ✓ **Bag of Freebies (BoF)** : Πρόκειται για μεθόδους και τεχνικές αύξησης δεδομένων που βελτιώνουν την απόδοση του δικτύου χωρίς να προσθέτουν όμως περισσότερο χρόνο στην επεξεργασία. Με την αύξηση των δεδομένων ενισχύεται η δημιουργία διαφορετικών παραλλαγών μιας μεμονωμένης εικόνας, γεγονός που καθιστά το δίκτυο πιο ισχυρό για πρόβλεψη.
- ✓ **Bag of Specials (BoS)** : Πρόκειται για στρατηγικές που αυξάνουν μεν οριακά το χρόνο εξαγωγής συμπερασμάτων, ωστόσο αυξάνουν σημαντικά και την απόδοση του δικτύου.

	Backbone	Detector
Bag of Freebies (BoF)	• CutMix • Mosaic data augmentation • DropBlock • Class label smoothing	• CIoU-loss • Cross mini-Batch Normalization • DropBlock • Mosaic data augmentation • Self-Adversarial Training • Multiple anchors for a single ground truth • Cosine annealing scheduler • Optimal hyperparameters • Random training shapes
Bag of Specials (BoS)	• Mish activation • Cross-stage partial connections (CSP) • Multi-input weighted residual connections (MiWRC)	• Mish activation • SPP-block • SAM-block • PAN path-aggregation block • DIoU-NMS

Εικόνα 1.3: Οι διάφορες μέθοδοι BoF και BoS που χρησιμοποιούνται στο Backbone και στον ανιχνευτή της YOLO v4 (Πηγή: https://towardsdatascience.com/yolo-v4-optimal-speed-accuracy-for-object-detection-79896ed47b50)

ΚΕΦΑΛΑΙΟ 2ο

2. Παρακολούθηση αντικειμένων

Στο κεφάλαιο αυτό γίνεται αναφορά στις διάφορες μεθόδους και τεχνικές που χρησιμοποιούνται για την παρακολούθηση αντικειμένων, κάτι το οποίο αποτελεί ένα μεγάλο και σημαντικό κομμάτι της όρασης υπολογιστών. Πιο συγκεκριμένα, γίνεται μία εισαγωγή στην παρακολούθηση αντικειμένων που αφορά τον γενικότερο τρόπο λειτουργίας και κατόπιν αναλύονται ορισμένοι διαθέσιμοι trackers που έχουν ευρεία χρήση σε πολλές εφαρμογές της ανίχνευσης και παρακολούθησης αντικειμένων.

2.1 Εισαγωγή στην παρακολούθηση αντικειμένων

Η ανίχνευση αντικειμένων είναι μια σημαντική εργασία στον τομέα της όρασης υπολογιστών. Η εξάπλωση των υπολογιστών υψηλής ισχύος, η διαθεσιμότητα οικονομικών καμερών υψηλής ποιότητας, όπως επίσης και η αυξανόμενη ανάγκη για αυτοματοποιημένη ανάλυση βίντεο έχει δημιουργήσει ένα μεγάλο ενδιαφέρον για τους αλγορίθμους εντοπισμού αντικειμένων. Διακρίνονται τρία βασικά βήματα στην ανάλυση των βίντεο:

- ✓ Ανίχνευση κινούμενων αντικειμένων ενδιαφέροντος
- ✓ Παρακολούθηση των αντικειμένων αυτών σε κάθε καρέ του βίντεο
- ✓ Ανάλυση των ιχνών των αντικειμένων αυτών για την αναγνώριση της συμπεριφοράς τους

Ως εκ τούτου, η χρήση της παρακολούθησης αντικειμένων είναι σχετική με τα ακόλουθα καθήκοντα: α) παρακολούθηση αντικειμένων σε βίντεο που αφορά αναγνώριση με βάση την κίνηση, δηλαδή αναγνώριση ανθρώπων με βάση το βάδισμα ή αυτόματη ανίχνευση αντικειμένων, β) αυτόματη επιτήρηση, δηλαδή παρακολούθηση μιας σκηνής για την ανίχνευση ύποπτων δραστηριοτήτων ή γεγονότων, γ) ευρετηρίαση βίντεο, δηλαδή αυτόματος σχολιασμός και ανάκτηση των βίντεο σε πολυμέσα και βάσεις δεδομένων, δ) αλληλεπίδραση ανθρώπου - υπολογιστή, δηλαδή αναγνώριση χειρονομιών, παρακολούθηση του βλέμματος των ματιών για δεδομένα εισόδου σε υπολογιστές ε) παρακολούθηση κυκλοφορίας, δηλαδή συγκέντρωση στατιστικών στοιχείων κυκλοφορίας σε πραγματικό χρόνο για να κατευθύνει τη ροή της κυκλοφορίας, στ) πλοήγηση οχήματος, δηλαδή σχεδιασμός διαδρομής βάσει βίντεο με δυνατότητες αποφυγής εμποδίων. [Beymer et al, 1999]

Στην απλούστερη μορφή της, η παρακολούθηση μπορεί να οριστεί ως το πρόβλημα της εκτίμησης της τροχιάς ενός αντικειμένου στο επίπεδο της εικόνας καθώς κινείται σε μια σκηνή. Με άλλα λόγια ένας ανιχνευτής αποδίδει συγκεκριμένες ετικέτες στα αντικείμενα που παρακολουθεί σε όλα τα καρέ ενός βίντεο. Επιπλέον, ανάλογα με τον τομέα παρακολούθησης, ένας ανιχνευτής μπορεί επίσης να παρέχει αντικειμενοκετρικές πληροφορίες, όπως ο προσανατολισμός, η περιοχή ή το σχήμα ενός αντικειμένου. [Yilmaz et al, 2006].

Η παρακολούθηση αντικειμένων μπορεί να είναι πολύπλοκη εξαιτίας των παρακάτω παραγόντων και συνθηκών:

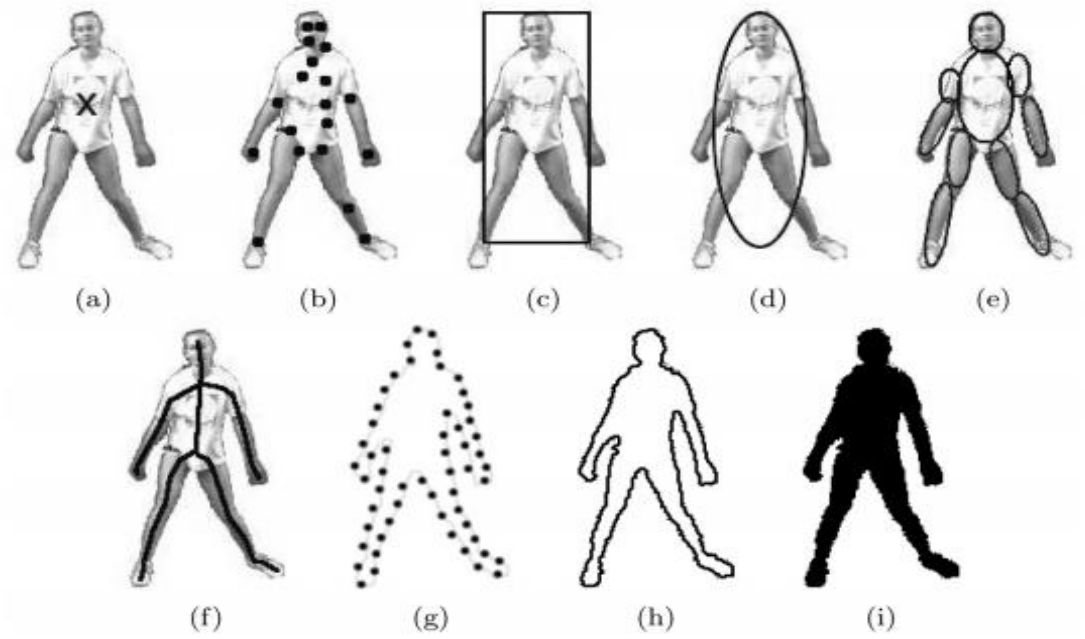
- Απώλεια πληροφοριών που προκαλείται από την προβολή του τρισδιάστατου κόσμου σε μια δισδιάστατη εικόνα
- Θόρυβος στις εικόνες
- Πολύπλοκη κίνηση αντικειμένων
- Άκαμπτη ή αρθρωτή φύση των αντικειμένων
- Μερική και πλήρης απόκρυψη των αντικειμένων
- Πολύπλοκα σχήματα αντικειμένων
- Αλλαγές φωτισμού σκηνής
- Απαιτήσεις επεξεργασίας σε πραγματικό χρόνο

Είναι εφικτό να απλοποιηθεί η διαδικασία της παρακολούθησης επιβάλλοντας περιορισμούς στην κίνηση ή/και την εμφάνιση των αντικειμένων. Πιο συγκεκριμένα, όλοι σχεδόν οι αλγόριθμοι παρακολούθησης υποθέτουν ότι η κίνηση του αντικειμένου είναι ομαλή χωρίς απότομες αλλαγές. Θα μπορούσε κάποιος να περιορίσει περαιτέρω την κίνησή του, θέτοντας το κριτήριο της σταθερής ταχύτητας ή επιτάχυνσης με βάση κάποιες εκ των προτέρων πληροφορίες [Yilmaz et al, 2006].

Έχουν προταθεί διάφορες προσεγγίσεις για την παρακολούθηση αντικειμένων. Αυτές διαφέρουν κυρίως μεταξύ τους με βάση τον τρόπο με τον οποίο προσεγγίζουν ορισμένα θέματα όπως τα χαρακτηριστικά της εικόνας, η εμφάνιση και το σχήμα του αντικειμένου ή το πώς θα μοντελοποιηθεί η κίνηση. Απαντήσεις σε αυτά τα ερωτήματα εξαρτώνται τόσο από το πλαίσιο/περιβάλλον στο οποίο πραγματοποιείται η παρακολούθηση όσο και από την τελική χρήση για την οποία αναζητούνται οι πληροφορίες παρακολούθησης.

Σε ένα σενάριο παρακολούθησης, ως αντικείμενο μπορεί να οριστεί οτιδήποτε παρουσιάζει ενδιαφέρον για περαιτέρω ανάλυση. Αυτά μπορούν να αναπαρασταθούν από τα σχήματα και την εμφάνισή τους. Παρακάτω περιγράφονται ορισμένες αναπαραστάσεις που χρησιμοποιούνται συνήθως για την παρακολούθηση.

- Σημεία: το αντικείμενο αναπαρίσταται από ένα σημείο, δηλαδή το κεντροειδές. (Σχήμα 3.1(α)).[Veenman et al. 2001] ή από ένα σύνολο σημείων (Σχήμα 3.1(b)) [Serby et al. 2004]. Γενικά η σημειακή αναπαράσταση είναι κατάλληλη για την παρακολούθηση αντικειμένων που καταλαμβάνουν μικρές περιοχές.
- Πρωτογενή γεωμετρικά σχήματα: το σχήμα του αντικειμένου αναπαρίσταται από ένα ορθογώνιο, μια έλλειψη κ.α (Σχήμα 3.1(c),(d)) [Comaniciu et al. 2003]. Εδώ η κίνηση του αντικειμένου για τέτοιες αναπαραστάσεις μοντελοποιείται συνήθως με μετατόπιση, συγγένεια ή προβολικό (ομογραφικό) μετασχηματισμό. Αν και τα βασικά γεωμετρικά σχήματα είναι πιο κατάλληλα για αναπαράσταση άκαμπτων αντικειμένων, χρησιμοποιούνται επίσης για την παρακολούθηση μη άκαμπτων.
- Σιλουέτα του αντικειμένου και περίγραμμα: η αναπαράσταση περιγράμματος ορίζει το όριο ενός αντικειμένου (Σχήμα 3.1(f),(g)). Η περιοχή εντός του περιγράμματος ονομάζεται σιλουέτα του αντικειμένου (θ). Οι αναπαραστάσεις σιλουέτας και περιγράμματος είναι κατάλληλες για την παρακολούθηση σύνθετων μη άκαμπτων σχημάτων.



Σχήμα 2.1: Αναπαραστάσεις αντικειμένων (a) κεντροειδές, (b) πολλαπλά σημεία, (c) ορθογώνιο, (d) έλλειψη, (e) πολλαπλές ελλείψεις, (f) σκελετός αντικειμένου, (g) πλήρες περίγραμμα αντικειμένου, (h) σημεία ελέγχου στο περίγραμμα του αντικειμένου, (i) σιλουέτα του αντικειμένου (Πηγή: Yilmaz et al, 2006)

- Μοντέλα αρθρωτού σχήματος: τα αρθρωτά αντικείμενα που συνήθως αναφέρονται στο ανθρώπινο σώμα, αποτελούνται από μέρη του σώματος που συγκρατούνται μεταξύ τους με αρθρώσεις. Η σχέση μεταξύ των τμημάτων προσομοιάζεται από κινηματικά μοντέλα κίνησης. Προκειμένου να αναπαραστήσει κανείς ένα αρθρωτό αντικείμενο, μπορεί να μοντελοποιήσει τα συστατικά μέρη χρησιμοποιώντας κυλίνδρους ή ελλείψεις. (Σχήμα 2.1(e))
- Σκελετικά μοντέλα: ο σκελετός του αντικειμένου μπορεί να εξαχθεί με την εφαρμογή του μετασχηματισμού του μεσαίου άξονα στη σιλουέτα του αντικειμένου [Ballard and Brown 1982]. Το συγκεκριμένο μοντέλο χρησιμοποιείται συνήθως ως αναπαράσταση σχήματος για την αναγνώριση αντικειμένων. [Yilmaz et al, 2006]

2.1.2 Κατηγορίες και προκλήσεις της παρακολούθησης αντικειμένων

Οι δύο βασικότερες κατηγορίες στις οποίες χωρίζεται η παρακολούθηση αντικειμένων αφορούν των αριθμό των διαφόρων κατηγοριών που προσπαθεί ο εκάστοτε αλγόριθμος να παρακολουθήσει και διακρίνονται σε:

- **Single object tracking**

- **Multiple object tracking**

Η πρώτη κατηγορία που είναι ίσως και η πιο εύκολη αποσκοπεί στην παρακολούθηση ενός αντικειμένου μίας και μόνο κλάσης αντί πολλαπλών αντικειμένων (Single Object Tracking-SOT). Στο SOT, το πλαίσιο οριοθέτησης του αντικειμένου-στόχου ορίζεται στο πρώτο καρέ και στόχος του αλγορίθμου είναι εν συνεχεία να εντοπίσει το ίδιο αντικείμενο και στα υπόλοιπα καρέ. Ανήκει στον τύπο παρακολούθησης χωρίς ανίχνευση καθώς πρέπει να παρέχεται χειροκίνητα το πρώτο πλαίσιο οριοθέτησης στον ανιχνευτή. Κάτι τέτοιο σημαίνει ότι οι ανιχνευτές μεμονωμένων αντικειμένων θα πρέπει να είναι σε θέση να παρακολουθούν οποιοδήποτε αντικείμενο τους δίνεται, ακόμη και ένα αντικείμενο για το οποίο δεν έχει εκπαιδευτεί κανένα διαθέσιμο μοντέλο ταξινόμησης. [Canny et al, 1986]

Αντίθετα η δεύτερη κατηγορία, δηλαδή η παρακολούθηση πολλαπλών αντικειμένων (Multiple Object Tracking-MOT) αναφέρεται στην προσέγγιση όπου ο αλγόριθμος παρακολούθησης παρακολουθεί κάθε αντικείμενο ενδιαφέροντος στο βίντεο, όντας σαφώς μια πιο περίπλοκη διαδικασία. Αρχικά, ο αλγόριθμος εντοπισμού καθορίζει τον αριθμό των αντικειμένων σε κάθε καρέ, και στη συνέχεια παρακολουθεί την ταυτότητα κάθε αντικειμένου από το ένα καρέ στο επόμενο μέχρι αυτά να εξαφανιστούν. [Canny et al, 1986]

Στο σημείο αυτό οφείλουμε να διευκρινίσουμε τη διαφορά μεταξύ των όρων ανίχνευση και παρακολούθηση αντικειμένων, καθώς με την ορολογία που περιβάλλει την βαθιά μάθηση οι έννοιες αυτές τείνουν να αναμιχθούν. Η παρακολούθηση αντικειμένων αναφέρεται στην ικανότητα εκτίμησης ή πρόβλεψης της θέσης ενός αντικειμένου-στόχου σε κάθε διαδοχικό καρέ ενός βίντεο, αφού πρώτα καθοριστεί η αρχική θέση του. Από την άλλη πλευρά, η ανίχνευση αντικειμένου είναι ο εντοπισμός-ανίχνευση ενός αντικειμένου-στόχου σε μια εικόνα ή σε ένα μεμονωμένο καρέ του βίντεο. Η ανίχνευση αντικειμένου θα λειτουργήσει μόνο εάν η εικόνα-στόχος είναι ορατή στη δεδομένη είσοδο. Σε περίπτωση που το αντικείμενο είναι κρυμμένο από οποιαδήποτε παρεμβολή δεν θα είναι σε θέση να το ανιχνεύσει [Αραπέλλης, 2021].

Η παρακολούθηση αντικειμένου εκπαιδεύεται να παρακολουθεί την τροχιά του αντικειμένου παρά τις όποιες αποκρύψεις.

Στην παρακολούθηση αντικειμένων υπάρχουν διάφορες προκλήσεις που μπορεί να αντιμετωπίσει κανείς κατά την εργασία σε αλγορίθμους παρακολούθησης αντικειμένων. Καταρχάς, είναι εύκολο να παρακολουθείται ένα αντικείμενο σε έναν ευθύ δρόμο ή σε κάποιο απλό περιβάλλον. Ωστόσο, σε κάποιο πραγματικό σενάριο, ο στόχος θα περάσει από παραμόρφωση, απόκρυψη ή θόρυβο φόντου.

Η **απόκρυψη(occlusion)** αντικειμένων σε βίντεο είναι μια από τις πιο συνηθισμένες προκλήσεις. Αναφέρεται σε ένα φαινόμενο παρεμβολής όπου το αντικείμενο επηρεάζεται από το φόντο ή το προσκήνιο, κατά το οποίο ο αλγόριθμος χάνει τον εντοπισμό του. Με άλλα λόγια, ο αλγόριθμος μπερδεύεται καθώς πλησιάζουν ολοένα και περισσότερα αντικείμενα. Αυτό οδηγεί στο συχνό πρόβλημα όπου το αρχικά αναγνωρισμένο αντικείμενο παρακολουθείται και πάλι (λανθασμένα) ως νέο. Μπορεί κανείς να εφαρμόσει σε αυτή την περίπτωση ευαισθησία στην απόκρυψη με σκοπό να την αποτρέψει. Η ευαισθησία στην απόκρυψη επιτρέπει στο χρήστη να εντοπίσει ποιο συγκεκριμένο χαρακτηριστικό του αντικειμένου προκαλεί σύγχυση στο δίκτυο. Μόλις αυτό εντοπιστεί, μπορούν να χρησιμοποιηθούν παρόμοιες εικόνες για να διορθωθούν οι προκαταλήψεις και να βοηθηθεί το δίκτυο να εξάγει τα χαρακτηριστικά που διαφοροποιούν τα αντικείμενα. [Yilmaz et al, 2006].

Η **ακαταστασία φόντου(background clutter)** είναι επίσης ένα συχνό φαινόμενο που αφορά την παρακολούθηση αντικειμένων καθώς σε κάθε εργασία μηχανικής μάθησης ή βαθιάς μάθησης, το φόντο των εικόνων που τροφοδοτούνται στον αλγόριθμο δημιουργεί πολλά προβλήματα. Το ίδιο συμβαίνει και με τα μοντέλα παρακολούθησης αντικειμένων. Θεωρητικά, όσο πιο “πυκνοκατοικημένο” είναι από διάφορα χαρακτηριστικά και αντικείμενα ένα φόντο, τόσο πιο δύσκολη είναι και η εξαγωγή των χαρακτηριστικών ενδιαφέροντος, η ανίχνευση ή ακόμη και η παρακολούθηση του αντικειμένου. Ένα τέτοιο φόντο εισάγει περιττές πληροφορίες ή θόρυβο που κάνουν το δίκτυο λιγότερο δεκτικό στα χαρακτηριστικά που είναι σημαντικά ενώ επίσης επιβαρύνουν το δίκτυο κάνοντάς το να μαθαίνει και να βελτιστοποιεί αργά. [Yilmaz et al, 2006].

2.2 Επιλογή χαρακτηριστικών για παρακολούθηση

Η επιλογή των σωστών χαρακτηριστικών παίζει κρίσιμο ρόλο στην παρακολούθηση. Γενικά, η πιο επιθυμητή ιδιότητα ενός οπτικού χαρακτηριστικού είναι η μοναδικότητά του, έτσι ώστε τα αντικείμενα να μπορούν να διακρίνονται με ευκολία. Η επιλογή χαρακτηριστικών συνδέεται στενά με την αναπαράσταση των αντικειμένων. Σε πολλές περιπτώσεις για παράδειγμα, το χρώμα χρησιμοποιείται ως χαρακτηριστικό γνώρισμα για την εμφάνιση με βάση το ιστόγραμμα ενώ για την αναπαράσταση με βάση το περίγραμμα χρησιμοποιούνται συνήθως οι ακμές του αντικειμένου. Πολλοί αλγόριθμοι παρακολούθησης πάντως χρησιμοποιούν έναν συνδυασμό αυτών των χαρακτηριστικών, οι λεπτομέρειες των οποίων αναφέρονται στη συνέχεια. [Levin et al, 2003]

Όσον αφορά το χρώμα ενός αντικειμένου, αυτό επηρεάζεται κυρίως από δύο φυσικούς παράγοντες, τη φασματική κατανομή ισχύος του φωτιστικού μέσου και την ανακλαστικότητα των επιφανειών του αντικειμένου. Στην επεξεργασία εικόνας, ο χρωματικός χώρος RGB (Red, Green, Blue) χρησιμοποιείται συνήθως για την αναπαράσταση του χρώματος. Ωστόσο, ο RGB δεν είναι ένας αντιληπτικά ομοιόμορφος χρωματικός χώρος, δηλαδή οι διαφορές μεταξύ των χρωμάτων στο χώρο RGB δεν αντιστοιχούν στις χρωματικές διαφορές που γίνονται αντιληπτές από τον άνθρωπο [Paschos 2001]. Επιπλέον, οι διαστάσεις του RGB είναι σε μεγάλο βαθμό συσχετισμένες. Από την άλλη οι L*u*v και L*a*b είναι αντιληπτά ομοιόμορφοι χρωματικοί χώροι, ενώ ο HSV (Hue, Saturation, Value) είναι ένας κατά προσέγγιση ομοιόμορφος χρωματικός χώρος. Από τα παραπάνω προκύπτει ότι δεν υπάρχει κάποιος χρωματικός χώρος που να έχει θεωρηθεί ως ο πιο αποδοτικός στην παρακολούθηση και ως εκ τούτου χρησιμοποιείται μια ποικιλία αυτών. [Levin et al, 2003]

Ένα ακόμη χαρακτηριστικό είναι τα όρια των αντικειμένων δηλαδή οι ακμές που δημιουργούν συνήθως έντονες μεταβολές στις εντάσεις της εικόνας. Πολλές φορές οι ακμές χρησιμοποιούνται για τον εντοπισμό αυτών των αλλαγών. Μια σημαντική ιδιότητα των ακμών είναι ότι είναι λιγότερο ευαίσθητες στις αλλαγές του φωτισμού σε σύγκριση με τα χρωματικά χαρακτηριστικά. Πολλοί αλγόριθμοι που εντοπίζουν τα όρια των αντικειμένων χρησιμοποιούν συνήθως τις ακμές ως αντιπροσωπευτικό χαρακτηριστικό. Λόγω της απλότητας και της ακρίβειάς της, η πιο δημοφιλής προσέγγιση ανίχνευσης ακμών είναι η Canny Edge detector [Canny 1986].

Επιπλέον η οπτική ροή έρχεται να προστεθεί ως ένα ακόμη χαρακτηριστικό που χρησιμοποιούν οι αλγόριθμοι παρακολούθησης. Πρόκειται για ένα πυκνό πεδίο διανυσμάτων μετατόπισης το οποίο ορίζει την μετατόπιση κάθε εικονοστοιχείου σε μια περιοχή. Στη συνέχεια υπολογίζεται χρησιμοποιώντας τον περιορισμό φωτεινότητας που προϋποθέτει σταθερή φωτεινότητα των αντίστοιχων pixel σε διαδοχικά καρέ [Horn and Schunk 1981]. Η οπτική ροή χρησιμοποιείται συνήθως ως χαρακτηριστικό γνώρισμα σε εφαρμογές που βασίζονται στην κίνηση και στον εντοπισμό. Ορισμένες δημοφιλείς τεχνικές που αφορούν τον υπολογισμό την οπτικής ροής περιλαμβάνουν μεθόδους των Horn και Schunk [1981], Lucas και Kanade [1981], Black και Anadan [1996] καθώς και Szeliski και Couglan [1997].

Η υφή ως το τέταρτο χαρακτηριστικό, είναι ένα μέτρο της μεταβολής της έντασης μιας επιφάνειας που ποσοτικοποιεί ιδιότητες όπως η ομαλότητα και κανονικότητα. Συγκριτικά με το χρώμα, η υφή απαιτεί ένα βήμα επεξεργασίας για τη δημιουργία των περιγραφών ενώ υπάρχουν διάφοροι περιγραφείς υφής όπως αυτό των Haralick et al. 1973 που αφορά ένα δισδιάστατο ιστόγραμμα το οποίο δείχνει τις συνυπάρξεις των εντάσεων σε μια καθορισμένη κατεύθυνση και απόσταση ή τα μέτρα υφής του Law [Laws 1980] τα οποία περιλαμβάνουν είκοσι πέντε φίλτρα 2D που παράγονται από πέντε 1D φίλτρα που αντιστοιχούν σε επίπεδο, ακμή, σημείο, κύμα και κυματισμό. Παρόμοια με τα χαρακτηριστικά ακμής, τα χαρακτηριστικά υφής είναι λιγότερο ευαίσθητα στις αλλαγές του φωτισμού σε σύγκριση με το χρώμα. [Levin et al, 2003]

Τα περισσότερα χαρακτηριστικά επιλέγονται χειροκίνητα από τον χρήστη ανάλογα με την εφαρμογή. Ωστόσο, το πρόβλημα της αυτόματης επιλογής χαρακτηριστικών έχει λάβει σημαντική προσοχή στην κοινότητα της αναγνώρισης προτύπων. Αυτόματη επιλογή χαρακτηριστικών μπορεί να χωριστεί σε μεθόδους φίλτρου και μεθόδους περιτύλιξης [Blum και Langley 1997]. Οι μέθοδοι φίλτρου προσπαθούν να επιλέξουν τα χαρακτηριστικά βάσει γενικών κριτηρίων, όπως ότι τα χαρακτηριστικά θα πρέπει να είναι ασύνδετα. Οι μέθοδοι περιτυλίγματος επιλέγουν τα χαρακτηριστικά με βάση τη χρησιμότητα των χαρακτηριστικών σε έναν συγκεκριμένο προβληματικό τομέα, όπως είναι η απόδοση ταξινόμησης με τη χρήση ενός υποσυνόλου χαρακτηριστικών. Η ανάλυση κύριων συνιστωσών (PCA) συνιστά ένα παράδειγμα των μεθόδων φίλτρου για τη μείωση των χαρακτηριστικών. Περιλαμβάνει μετασχηματισμό ενός αριθμού (ενδεχομένως) συσχετιζόμενων μεταβλητών σε έναν (μικρότερο) αριθμό από ασυσχέτιστες μεταβλητές που ονομάζονται κύριες συνιστώσες. Η πρώτη κύρια συνιστώσα αντιπροσωπεύει όσο το δυνατόν μεγαλύτερο μέρος της μεταβλητότητας των δεδομένων και κάθε επόμενη συνιστώσα αντιπροσωπεύει όσο το δυνατόν μεγαλύτερο μέρος της εναπομένουσας μεταβλητότητας [Yilmaz et al, 2006].

Μεταξύ όλων των χαρακτηριστικών, το χρώμα είναι ένα από τα πιο ευρέως χρησιμοποιούμενα χαρακτηριστικά για την παρακολούθηση αντικειμένων. Οι Comaniciu et al. [2003] χρησιμοποιούν ένα ιστόγραμμα χρώματος για την αναπαράσταση της εμφάνισης του αντικειμένου. Παρά τη δημοτικότητά του, οι περισσότερες χρωματικές ζώνες είναι ευαίσθητες στις μεταβολές του φωτισμού. Ως εκ τούτου, σε σενάρια όπου αυτό το φαινόμενο είναι αναπόφευκτο, ενσωματώνονται άλλα χαρακτηριστικά για την μοντελοποίηση του αντικειμένου.

2.3 Μέθοδοι και τεχνικές της παρακολούθησης αντικειμένων

Ο στόχος ενός ανιχνευτή αντικειμένων είναι να δημιουργήσει την τροχιά ενός αντικειμένου με την πάροδο του χρόνου εντοπίζοντας τη θέση του σε κάθε καρέ του βίντεο. Ο ανιχνευτής αντικειμένων μπορεί επίσης να παρέχει την πλήρη περιοχή της εικόνας που καταλαμβάνει το αντικείμενο σε κάθε χρονική στιγμή. Τα καθήκοντα της ανίχνευσης του αντικειμένου και της καθιέρωσης αντιστοιχίας μεταξύ των διαφόρων περιπτώσεων αντικειμένων σε όλα τα καρέ που απαρτίζουν το βίντεο, μπορούν να εκτελούνται είτε χωριστά είτε από κοινού. Στην πρώτη περίπτωση, πιθανές περιοχές με αντικείμενα σε κάθε καρέ λαμβάνονται με τη βοήθεια ενός αλγορίθμου και στη συνέχεια ο ανιχνευτής αντιστοιχεί τα αντικείμενα σε όλα τα καρέ. Στην δεύτερη περίπτωση, η περιοχή του αντικειμένου και η αντιστοιχία εκτιμάται από κοινού με επαναληπτική ενημέρωση της θέσης του αντικειμένου και της περιοχής που λαμβάνονται από τα προηγούμενα καρέ. Σε κάθε προσέγγιση παρακολούθησης, τα αντικείμενα αναπαρίστανται χρησιμοποιώντας τα μοντέλα σχήματος ή/και εμφάνισης που περιγράφηκαν στην προηγούμενη ενότητα. [MacCormick & Blake, 2000]

Categories	Representative Work
Point Tracking	
• Deterministic methods	MGE tracker [Salari and Sethi 1990], GOA tracker [Veenman et al. 2001].
• Statistical methods	Kalman filter [Broida and Chellappa 1986], JPDAF [Bar-Shalom and Foreman 1988], PMHT [Streit and Luginbuhl 1994].
Kernel Tracking	
• Template and density based appearance models	Mean-shift [Comaniciu et al. 2003], KLT [Shi and Tomasi 1994], Layering [Tao et al. 2002].
• Multi-view appearance models	Eigentracking [Black and Jepson 1998], SVM tracker [Avidan 2001].
Silhouette Tracking	
• Contour evolution	State space models [Isard and Blake 1998], Variational methods [Bertalmio et al. 2000], Heuristic methods [Ronfard 1994].
• Matching shapes	Hausdorff [Huttenlocher et al. 1993], Hough transform [Sato and Aggarwal 2004], Histogram [Kang et al. 2004].

Εικόνα 3.1: Κατηγορίες των μεθόδων παρακολούθησης (Πηγή: A. Yilmaz et. Al)

Παρακολούθηση σημείων: Τα αντικείμενα που ανιχνεύονται σε διαδοχικά καρέ αντιπροσωπεύονται από σημεία, και η συσχέτιση των σημείων βασίζεται στην προηγούμενη κατάσταση του αντικειμένου που μπορεί να περιλαμβάνει τη θέση και την κίνηση του αντικειμένου. Αυτή η προσέγγιση απαιτεί έναν εξωτερικό μηχανισμό για την ανίχνευση των αντικειμένων σε κάθε καρέ.

Παρακολούθηση πυρήνα: Ο πυρήνας αναφέρεται στο σχήμα και την εμφάνιση του αντικειμένου. Μπορεί να είναι ένα ορθογώνιο πρότυπο ή ένα ελλειπτικό σχήμα με ένα σχετικό ιστόγραμμα. Τα αντικείμενα παρακολουθούνται με τον υπολογισμό της κίνησης του πυρήνα τους σε διαδοχικά καρέ. Συνήθως μια τέτοια κίνηση έχει τη μορφή παραμετρικού μετασχηματισμού όπως μετάθεση, περιστροφή ή κάποια άλλη.

Παρακολούθηση σιλουέτας: Η ανίχνευση πραγματοποιείται με την εκτίμηση της περιοχής του αντικειμένου σε κάθε καρέ. Οι μέθοδοι αυτού του εντοπισμού βάση της σιλουέτας χρησιμοποιούν τις πληροφορίες που είναι κωδικοποιημένες στο εσωτερικό του αντικειμένου. Τέτοιες πληροφορίες μπορεί να έχουν τη μορφή μοντέλων πυκνότητας εμφάνισης και σχήματος τα οποία συνήθως έχουν τη μορφή χαρτών ακμών [Yilmaz et al, 2006].

Αλγόριθμος MDNet

Η ανίχνευση αντικειμένων υπάρχει εδώ και σχεδόν 20 χρόνια και έχουν εισαχθεί πολλές μέθοδοι και ιδέες για τη βελτίωσή της ακρίβειας και της αποτελεσματικότητας των μοντέλων ανίχνευσης. Ορισμένες από τις υλοποιήσεις αφορούσαν παραδοσιακές ή κλασσικές προσεγγίσεις μηχανικής μάθησης, όπως η k-Nearest Neighbor ή η Support Vector Machine. Αυτές οι προσεγγίσεις είναι καλές στην πρόβλεψη του αντικειμένου-στόχου, αλλά απαιτούν σημαντικά και διακριτικά χαρακτηριστικά που εξάγονται από επαγγελματίες του είδους. Από την άλλη πλευρά, οι μέθοδοι βαθιάς μάθησης εξάγουν αυτά τα σημαντικά χαρακτηριστικά και τις αναπαραστάσεις από μόνες τους. Παρακάτω παρουσιάζονται ορισμένες από αυτές τις τεχνικές παρακολούθησης. Πρώτη τεχνική είναι το Multi-Domain Net (MDNet) που αποτελεί έναν τύπο αλγορίθμου εντοπισμού αντικειμένων οποίος αξιοποιεί δεδομένα μεγάλης κλίμακας για εκπαίδευση. Στόχος του είναι η εκμάθηση τεράστιων παραλλαγών και χωρικών σχέσεων. Το MDNet εκπαιδεύεται για να μάθει την κοινή αναπαράσταση των στόχων από πολλαπλά σχολιασμένα βίντεο, δηλαδή λαμβάνει πολλαπλά σχολιασμένα βίντεο που ανήκουν σε διαφορετικούς τομείς. Αποτελείται από προ-εκπαίδευση και online οπτική παρακολούθηση:

Προ-εκπαίδευση: Κατά την προ-εκπαίδευση, το δίκτυο απαιτείται να μάθει την αναπαράσταση πολλαπλών τομέων. Για να επιτευχθεί αυτό, ο αλγόριθμος εκπαιδεύεται σε πολλαπλά σχολιασμένα βίντεο για να μάθει την αναπαράσταση και τα διάφορα χωρικά χαρακτηριστικά.

Online οπτική παρακολούθηση: Αφού ολοκληρωθεί η προ-εκπαίδευση, αφαιρούνται τα επίπεδα που αφορούν τον συγκεκριμένο τομέα και το δίκτυο μένει μόνο με κοινά επίπεδα, τα οποία αποτελούνται από μαθημένες αναπαραστάσεις. Κατά τη διάρκεια εξαγωγής συμπερασμάτων, προστίθεται ένα δυαδικό επίπεδο ταξινόμησης, το οποίο εκπαιδεύεται ή ρυθμίζεται με ακρίβεια στο διαδίκτυο.

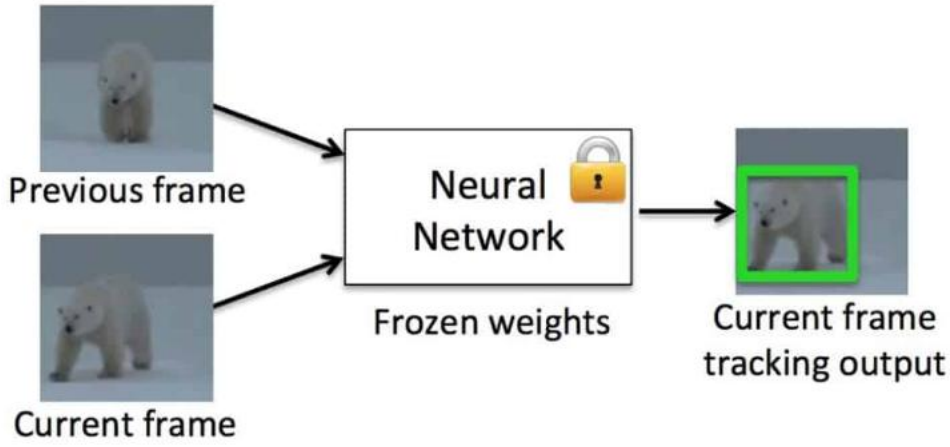
Αυτή η τεχνική εξοικονομεί αρκετό χρόνο, καθώς έχει αποδειχθεί ότι είναι ένας αποτελεσματικός αλγόριθμος παρακολούθησης που βασίζεται στο διαδίκτυο. [Nam et al, 2016]

Αλγόριθμος GOTURN

Τα δίκτυα βαθιάς παλινδρόμησης είναι μοντέλα που βασίζονται στην εκπαίδευση χωρίς σύνδεση. Αυτός ο αλγόριθμος μαθαίνει μια γενική σχέση μεταξύ της κίνησης και της εμφάνισης του αντικειμένου και μπορεί να χρησιμοποιηθεί για την παρακολούθηση αντικειμένων που δεν εμφανίζονται στο σύνολο εκπαίδευσης. Οι online αλγόριθμοι παρακολούθησης είναι συχνά πιο αργοί και δεν αποδίδουν καλά σε πραγματικό χρόνο, κάτι

που οφείλεται στο γεγονός ότι δεν μπορούν να εκμεταλλευτούν μεγάλο αριθμό βίντεο για να βελτιώσουν την απόδοσή τους. Αντίθετα, οι αλγόριθμοι παρακολούθησης χωρίς σύνδεση μπορούν να εκπαιδευτούν για να χειρίζονται περιστροφές, αλλαγές στην οπτική γωνία, αλλαγές στο φωτισμό και άλλες σύνθετες προκλήσεις. [Held et al, 2017]

Η γενική παρακολούθηση αντικειμένων με χρήση δικτύων παλινδρόμησης ή ο GOTURN χρησιμοποιεί μια προσέγγιση που βασίζεται στην παλινδρόμηση για την παρακολούθηση αντικειμένων. Ουσιαστικά, παλινδρομούν απευθείας για τον εντοπισμό των αντικειμένων-στόχων με ένα μόνο πέρασμα τροφοδότησης προς τα εμπρός μέσω του δικτύου. Το δίκτυο δέχεται δύο εισόδους: μια περιοχή αναζήτησης από το τρέχον καρέ και έναν στόχο από το προηγούμενο καρέ. Στη συνέχεια, το δίκτυο συγκρίνει αυτές τις εικόνες για να βρει το αντικείμενο-στόχο στην τρέχουσα εικόνα. [Held et al, 2017]

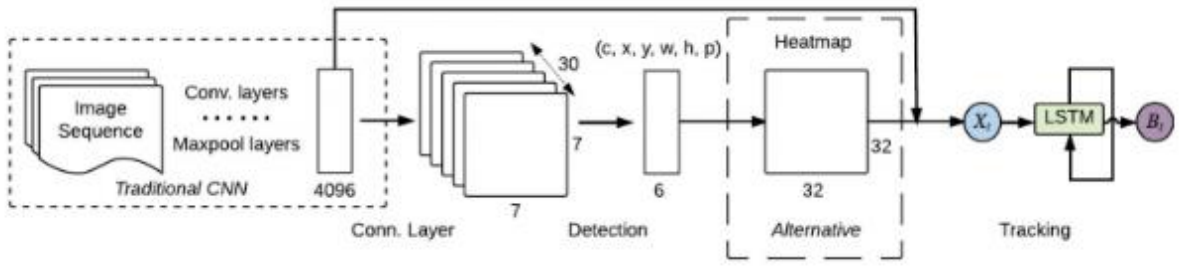


Εικόνα 2.2: Τρόπος λειτουργίας του GOTURN (Πηγή: Held et.Al)

Αλγόριθμος ROLO-Recurrent YOLO

Το ROLO είναι ένας συνδυασμός των επαναλαμβανόμενων νευρωτικών δικτύων και της YOLO. Γενικά, το LTSM (Long short-term memory) το οποίο είναι ένα τεχνητό νευρωνικό δίκτυο που χρησιμοποιείται ευρέως στους τομείς της τεχνητής νοημοσύνης, προτιμάται σε συνδυασμό με το CNN που αναφέραμε σε προηγούμενο εδάφιο.

Ως εκ τούτου, το ROLO συνδυάζει δύο τύπους νευρωνικών δικτύων: το ένα είναι το CNN που χρησιμοποείται για την εξαγωγή χωρικών πληροφοριών, ενώ το άλλο είναι ένα δίκτυο LSTM που χρησιμοποείται για την εύρεση της τροχιάς του αντικειμένου-στόχου. Σε κάθε χρονικό βήμα, οι χωρικές πληροφορίες εξάγονται και αποστέλλονται στο LSTM, το οποίο στη συνέχεια επιστρέφει τη θέση του αντικειμένου που παρακολουθείται. Στο διάγραμμα που ακολουθεί φαίνεται η λειτουργία ενός τέτοιου αλγορίθμου.



Σχήμα 2.2: Λειτουργία του ROLO (Πηγή: Ning et. Al)

Η ακολουθία βίντεο τροφοδοτείται στην αρχιτεκτονική YOLO, η οποία αποτελείται κυρίως από CNN, εδώ εξάγονται τα διάφορα χαρακτηριστικά καθώς και ανιχνεύονται τα οριοθετημένα πλαίσια. Στη συνέχεια, τα οπτικά χαρακτηριστικά μαζί με τα πλαίσια οριοθέτησης συνενώνονται και τροφοδοτούνται στο LSTM το οποίο κατόπιν προβλέπει την τροχιά των αντικειμένων [https://www.v7labs.com/blog/object-tracking-guide].

Φίλτρα Kalman

Τα φίλτρα Kalman, αν και μπορούν να χρησιμοποιηθούν για πολλούς σκοπούς, χρησιμοποιούνται συχνά για την παρακολούθηση αντικειμένων. Είναι ιδιαίτερα βολικά για αντικείμενα των οποίων το μοντέλο κίνησης είναι γνωστό, ενώ επιπλέον ενσωματώνουν κάποιες επιπλέον πληροφορίες προκειμένου να εκτιμήσουν την επόμενη θέση του αντικειμένου αξιόπιστα. Μπορούν να χρησιμοποιηθούν για την παρακολούθηση ενός αντικειμένου γενικού σκοπού, υποθέτοντας ορισμένους περιορισμούς. Στη συνέχεια περιγράφεται η κύρια ιδέα πίσω από τα φίλτρα Kalman και η πρακτική χρήση τους στην παρακολούθηση αντικειμένων. [Juric, 2015]

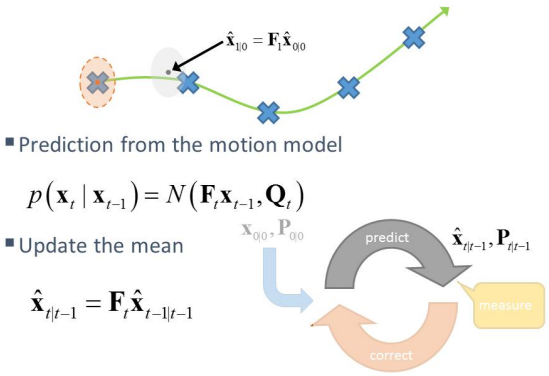
Το φίλτρο Kalman είναι ένα αναδρομικό προγνωστικό φίλτρο που βασίζεται στη χρήση τεχνικών χώρου, καταστάσεων και αναδρομικών αλγορίθμων. Εκτιμά την κατάσταση του δυναμικού συστήματος. Αυτό το δυναμικό σύστημα μπορεί να διαταραχθεί από κάποιο θόρυβο, οποίος θεωρείται κυρίως "λευκός" θόρυβος. Για τη βελτίωση της εκτιμώμενης κατάστασης το φίλτρο Kalman χρησιμοποιεί μετρήσεις που σχετίζονται με την κατάσταση αλλά μπορεί να είναι και διαταραγμένες. Έτσι το φίλτρο Kalman αποτελείται από δύο βασικά βήματα:

- Την κατάσταση πρόβλεψης, η οποία προβλέπεται με το δυναμικό μοντέλο.
- Το βήμα διόρθωσης, το οποίο διορθώνεται με την παρατήρηση του μοντέλου, έτσι ώστε η συν διακύμανση σφάλματος του εκτιμητή να ελαχιστοποιείται.

Στο πρώτο στάδιο, που είναι αυτό της πρόβλεψης περιλαμβάνεται η πρόβλεψη της επόμενης κατάστασης (θέση και ταχύτητα) καθώς και η ενημέρωση της αβεβαιότητας σχετικά με την κατάσταση του αντικειμένου. [Juric, 2015]

A) State prediction:

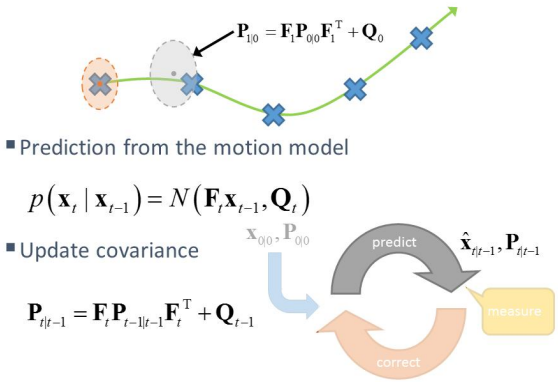
Η επόμενη κατάσταση x(t | t-1) προκύπτει από τον πολλαπλασιασμό της προηγούμενης κατάστασης με τον πίνακα μετάβασης κατάστασης -F. Το Q δηλώνει το θόρυβο της διαδικασίας, οποίος πρέπει να ακολουθεί κανονική κατανομή, αφού γίνεται χρήση Gaussians.



Σχήμα 2.3: State prediction (Πηγή: Juric 2015)

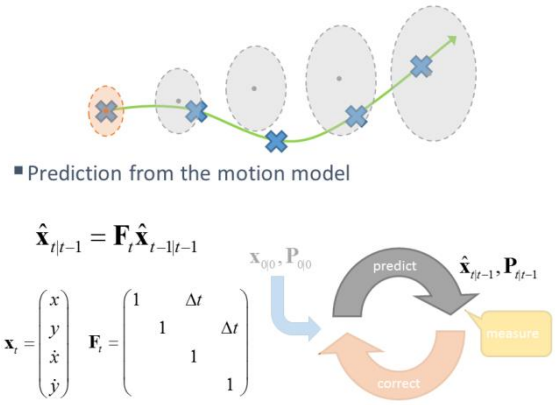
B) Covariance prediction:

Η ενημέρωση της συνδιακύμανσης γίνεται πολλαπλασιάζοντας τον πίνακα συνδιακύμανσης από την προηγούμενη επανάληψη με τον πίνακα μετάβασης κατάστασης F (μοντέλο κίνησης) και προσθέτοντας τον θόρυβο διαδικασίας Q που μπορεί να είναι σταθερός.



Σχήμα 2.4: Covariance prediction (a) (Πηγή: Juric 2015)

Ο πίνακας μετάβασης κατάστασης πρέπει να είναι γραμμικός. Εάν το μοντέλο δεν είναι γραμμικό, πρέπει να γραμμικοποιηθεί σε κάποιο σημείο εργασίας το οποίο χρησιμοποιείται στο Extended Kalman Filter. Το χρησιμοποιούμενο μοντέλο μοντελοποιεί εκείνο της κίνησης σταθερής 2D ταχύτητας όπου η θέση ενημερώνεται ώς εξής: p(t) = p(t-1) + v * p(t-1) όπου p δηλώνει τη θέση και v την ταχύτητα, η οποία παραμένει σταθερή.

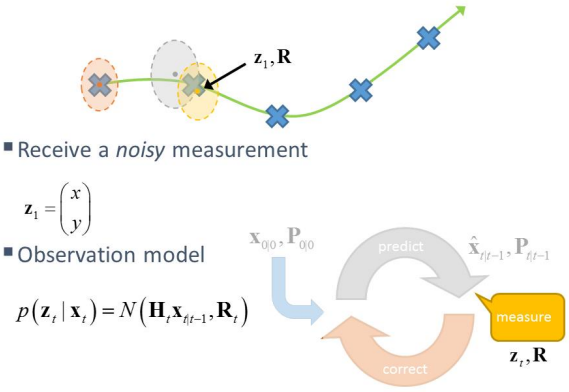


Σχήμα 2.5: Covariance prediction (b) (Πηγή: Juric 2015)

Αφού ολοκληρωθεί το στάδιο της πρόβλεψης ακολουθεί το δεύτερο στάδιο, δηλαδή εκείνο της διόρθωσης.

A) Measure:

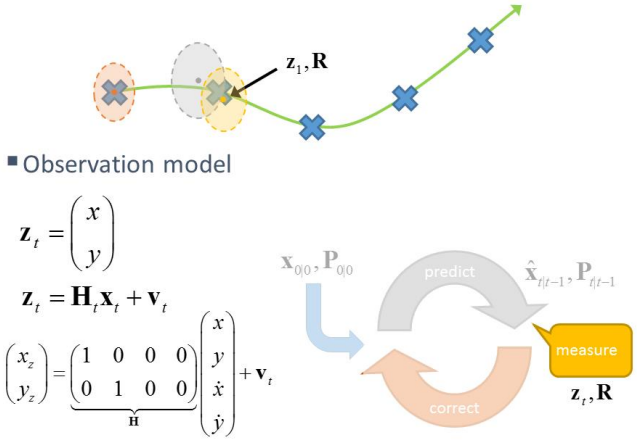
Μόλις ληφθεί μια θορυβώδης μέτρηση ξεκινά η διαδικασία της ενημέρωσης.. Η μέτρηση θορύβου z(t) μοντελοποιείται ως ένα απλό Gaussian, όπου ο θόρυβος μοντελοποιείται ως πίνακας συνδιακύμανσης R(t) που συνήθως είναι σταθερός. Η αβεβαιότητα της μέτρησης εμφανίζεται ως χρυσή έλλειψη.



Σχήμα 2.6: Measurement (a) (Πηγή: Juric 2015)

B) Measurement update:

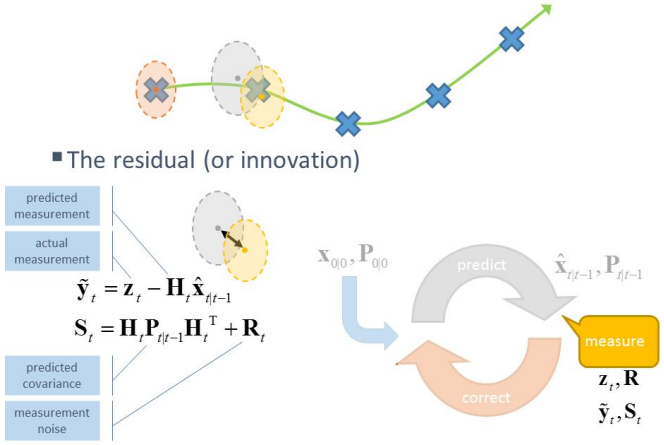
Με σκοπό να υπολογιστεί η προβλεπόμενη μέτρηση που απαιτείται για τη διόρθωση, πρέπει να επιλεχθούν τα στοιχεία μέτρησης από την κατάσταση. Μια μέτρηση έχει την ίδια δομή με μια κατάσταση ή περιέχει απλώς τμήματα της κατάστασης (όπως η θέση του αντικειμένου). Ο πίνακας Η είναι ο πίνακας επιλογής μοντέλου που όταν πολλαπλασιάζεται με την κατάσταση επιλέγει μόνο τα στοιχεία εκείνα που ανήκουν σε μια μέτρηση.



Σχήμα 2.7: Measurement (b) (Πηγή: Juric 2015)

C) Residual:

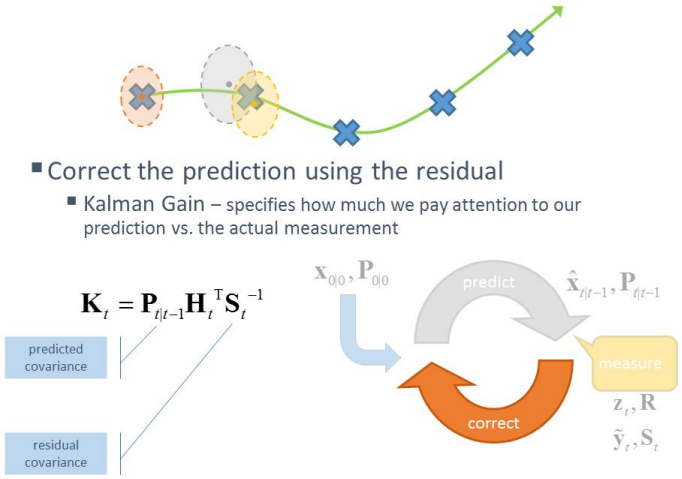
Για να γίνει διόρθωση, πρέπει να είναι γνωστό το σφάλμα πρόβλεψης. Επομένως το υπόλοιπο, γνωστό και ως καινοτομία, υπολογίζεται και συμβολίζεται με y(t). Το υπόλοιπο υπολογίζεται διαφοροποιώντας την προβλεπόμενη μέτρηση και τη μέτρηση που προκύπτει. Η υπολειπόμενη συνδιακύμανση S υπολογίζεται με παρόμοιο τρόπο.



Σχήμα 2.8: Residual (Πηγή: Juric 2015)

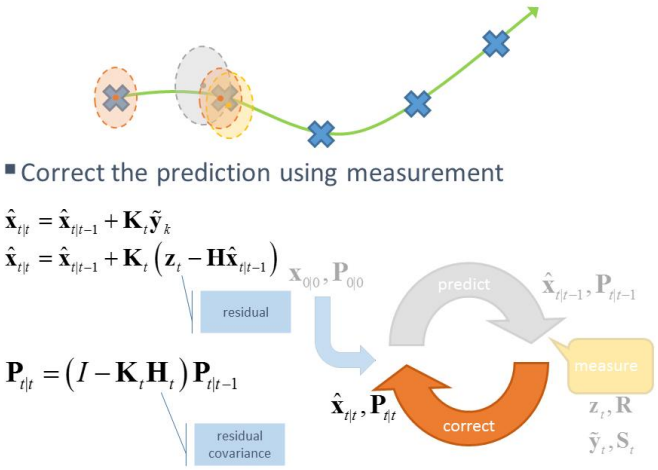
D) Kalman gain:

Το Kalman-gain K καθορίζει πόσο πιο πιστή είναι στο χρήστη η πρόβλεψη έναντι της μέτρησης. Είναι ένα γινόμενο του πίνακα συνδιακύμανσης της προβλεπόμενης διαδικασίας P του μοντέλου παρατήρησης H και της αντίστροφης συνδιακύμανσης του υπολοίπου S.



Σχήμα 2.9: Kalman gain (a)

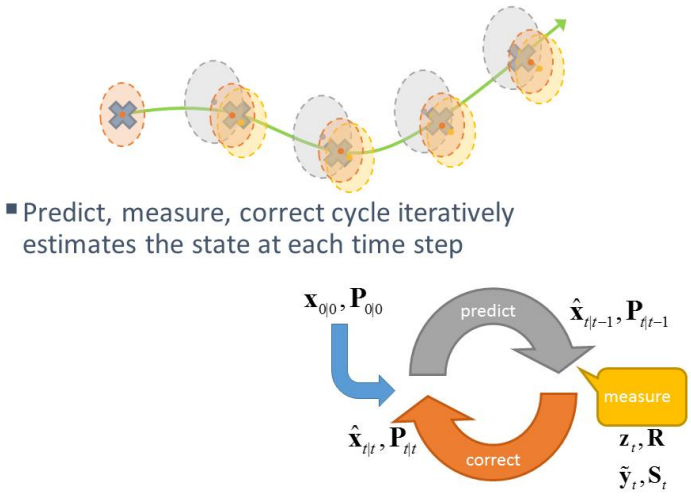
Στη συνέχεια το Kalman-gain χρησιμοποιείται για την ενημέρωση της κατάστασης x και του πίνακα συνδιακύμανσης P, όπως φαίνεται στο παρακάτω σχήμα. Το αποτέλεσμα είναι η ενημερωμένη θέση που συμβολίζεται με τη χρυσή έλλειψη.



Σχήμα 2.10: Kalman gain (b)

E) Final step:

Όταν ολοκληρωθεί το βήμα της διόρθωσης, το επόμενο βήμα είναι και πάλι αυτό της πρόβλεψης. Αυτά τα δύο βήματα εκτελούνται επαναληπτικά για την παρακολούθηση του αντικειμένου, όπως φαίνεται στο σχήμα που ακολουθεί.



Σχήμα 2.11: final step (Πηγή: Juric 2015)

Όλα όσα αναφέρθηκαν για το φίλτρο Kalman αφορούν μόνο το θεωρητικό επίπεδο, καθώς η εις βάθος κατανόηση της λειτουργίας του εν λόγω φίλτρου δεν είναι ο σκοπός της παρούσας διπλωματικής εργασίας, μιας και δεν είναι το βασικό εργαλείο που χρησιμοποιείται για την παρακολούθηση αλλά ένα πολύ μικρό τμήμα αυτής της διαδικασίας. Παρακάτω θα γίνει λόγος για το βασικό μοντέλο παρακολούθησης που χρησιμοποιείται για τη δημιουργία της εφαρμογής μικρό μέρος του οποίου είναι και το φίλτρο Kalman. [Juric. 2015]

DeepSort – Αλγόριθμος παρακολούθησης

Ο DeepSort είναι ένας αλγόριθμος που χρησιμοποιείται στην όραση υπολογιστών για την παρακολούθηση αντικειμένων, αποδίδοντας μάλιστα και ένα αναγνωριστικό ID σε κάθε ένα από αυτά. Αποτελεί επέκταση του αλγορίθμου SORT (Simple Online Realtime Tracking). Ο DeepSort εισάγει την βαθιά εκμάθηση στον αλγόριθμο SORT προσθέτοντας έναν περιγραφέα εμφάνισης για τη μείωση των εναλλαγών ταυτότητας, καθιστώντας έτσι την παρακολούθηση πιο αποτελεσματική. Προκειμένου να γίνει πιο κατανοητός ο DeepSort θα γίνει μια επεξήγηση της λειτουργίας του προκάτοχού του δηλαδή του SORT. [Sanyam, 2022]

- SORT (Simple Online Realtime Tracking)

Ο SORT αποτελεί μία προσέγγιση για την παρακολούθηση αντικειμένων όπου χρησιμοποιούνται υποτυπώδεις προσεγγίσεις όπως τα φίλτρα Kalman που αναφέρθηκαν στην προηγούμενη ενότητα, καθώς και οι ουγγρικοί αλγόριθμοι για την παρακολούθηση αντικειμένων που θεωρούνται από πολλούς καλύτεροι από διάφορους άλλους διαδικτυακούς ανιχνευτές. Ο SORT αποτελείται από 4 βασικά στοιχεία που είναι τα εξής:

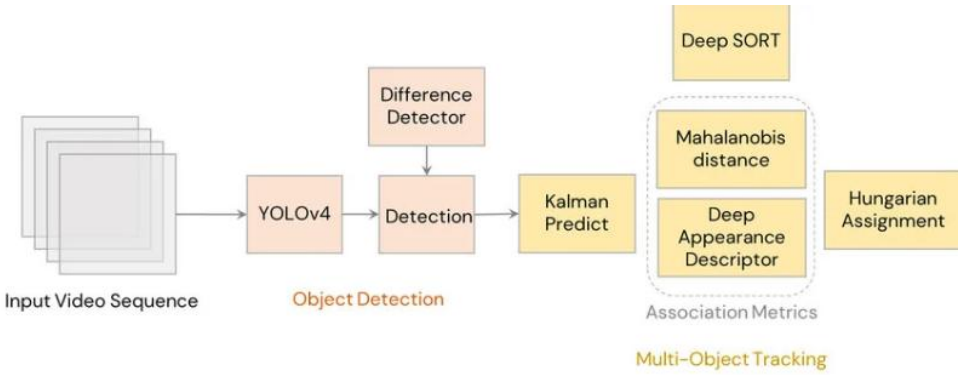
- *Detection*: Αυτό είναι και το πρώτο βήμα στην ενότητα παρακολούθησης δηλαδή η ανίχνευση. Σε αυτό το βήμα, ένας ανιχνευτής αντικειμένων ανιχνεύει τα αντικείμενα στο καρέ που πρόκειται να ξεκινήσει η παρακολούθηση. Αυτές οι ανιχνεύσεις στην συνέχεια διαβιβάζονται στο επόμενο βήμα. Ανιχνευτές όπως οι FrRCNN, YOLO και διάφορες εκδοχές τους χρησιμοποιούνται συχνότερα. [Sanyam, 2022]

- *Estimation*: Σε αυτό το βήμα εκτιμάται η θέση του στόχου στο πλαίσιο του επόμενου καρέ, χρησιμοποιώντας ένα μοντέλο σταθερής ταχύτητας. Όταν μια ανίχνευση σχετίζεται με έναν στόχο, το εντοπισμένο πλαίσιο οριοθέτησης χρησιμοποιείται για την ενημέρωση της κατάστασης του στόχου, όπου οι συνιστώσες της ταχύτητας επιλύονται βέλτιστα μέσω του πλαισίου φίλτρου Kalman. [Sanyam, 2022]
- Data association: Στο βήμα αυτό υπάρχει πλέον το πλαίσιο οριοθέτησης στόχου και το πλαίσιο οριοθέτησης που εντοπίστηκε. Έτσι, ένας πίνακας κόστους υπολογίζεται ως η απόσταση τομής άνω ενώσεως (Intersection Over Union-IOU) μεταξύ κάθε ανίχνευσης και όλων των προβλεπόμενων οριοθετημένων πλαισίων από τους υπάρχοντες στόχους. Η ανάθεση επιλύεται βέλτιστα χρησιμοποιώντας τον ουγγρικό αλγόριθμο. Εάν η IOU της ανίχνευσης και του στόχου είναι μικρότερη από μια ορισμένη τιμή κατωφλίου που ονομάζεται IOUmin, τότε η εν λόγω ανάθεση απορρίπτεται. Αυτή η τεχνική επιλύει το πρόβλημα της απόκρυψης και βοηθά στη διατήρηση των αναγνωριστικών ID αντικειμένων. [Sanyam, 2022]
- *Creation and Deletion of Track Identities:* Αυτό το βήμα είναι υπεύθυνο για τη δημιουργία και τη διαγραφή ταυτότητας. Οι μοναδικές ταυτότητες δημιουργούνται και καταστρέφονται σύμφωνα με το IOUmin. Εάν η επικάλυψη της ανίχνευσης και του στόχου είναι μικρότερη από το IOUmin, τότε σημαίνει ότι το αντικείμενο δεν παρακολουθείται. Τα ίχνη τερματίζονται εάν δεν ανιχνεύονται για ορισμένα από το χρήστη καρέ. Εάν το αντικείμενο επανεμφανιστεί, η παρακολούθηση θα συνεχιστεί δίνοντας στο αντικείμενο νέα ταυτότητα. [Sanyam, 2022]

Τα αντικείμενα μπορούν να εντοπιστούν με επιτυχία με τη χρήση αλγορίθμων SORT, οι οποίοι υπερτερούν έναντι πολλών αλγορίθμων τελευταίας τεχνολογίας. Ο ανιχνευτής θα δώσει τις ανιχνεύσεις, τα φίλτρα Kalman θα δώσουν ίχνη και ο ουγγρικός αλγόριθμος εκτελεί τη συσχέτιση δεδομένων. Ωστόσο ο DeepSort έχει ένα βασικό πλεονέκτημα έναντι του SORT το οποίο αναφέρεται στη συνέχεια. [https://learnopencv.com/understanding-multiple-object-tracking-using-deepsort/#Introduction-to-DeepSORT]

DeepSort

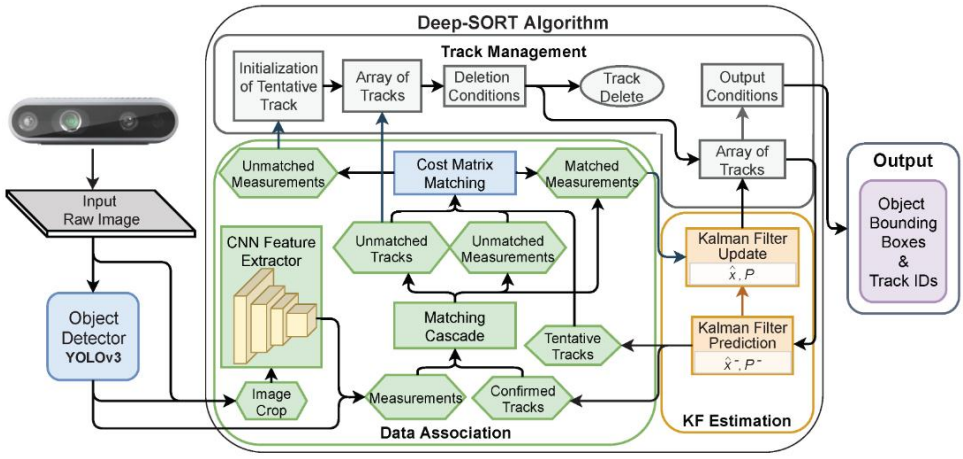
Ο αλγόριθμος SORT έχει πολύ καλές επιδόσεις όσον αφορά την ακρίβεια εντοπισμού, όμως επιστρέφει ίχνη με μεγάλο αριθμό εναλλαγών ID και αποτυγχάνει σε περίπτωση απόκρυψης. Αυτό οφείλεται στον πίνακα συσχέτισης που χρησιμοποιείται. Ο DeepSort από την άλλη χρησιμοποιεί μια καλύτερη μέτρηση συσχέτισης που συνδυάζει περιγραφές κίνησης και εμφάνισης. Επίσης ο DeepSort μπορεί να οριστεί ως ο αλγόριθμος παρακολούθησης που παρακολουθεί αντικείμενα όχι μόνο με βάση την ταχύτητα και την κίνηση του αντικειμένου αλλά και την εμφάνισή του. [Pereira et al, 2022]



Σχήμα 2.12: Στάδια λειτουργίας του DeepSort (Πηγή: Sanyam: Understanding Multiple Object Tracking using DeepSort)

Ο τελευταίος αποτελεί βελτίωση του αλγορίθμου SORT, ενσωματώνοντας πληροφορίες εμφάνισης των αντικειμένων για την ενίσχυση των συσχετίσεων. Η συσχέτιση δεδομένων ενσωματώνει μια πρόσθετη μετρική εμφάνισης που βασίζεται σε προ-εκπαιδευμένα CNN και επιτρέπει την εκ νέου ταυτοποίηση τροχιών, μετά από μακρά περίοδο απόκρυψης.

Οι ενότητες KF Estimation και Track management είναι παρόμοιες με τις αντίστοιχες ενότητες του SORT. Μια επισκόπηση της μεθόδου παρουσιάζεται στο παρακάτω σχήμα.



Σχήμα 2.13: DeepSort algorithm (Πηγή: Pereira et al, 2022)

Όπως και στον SORT, η συσχέτιση των εντοπισμένων οριοθετημένων πλαισίων με τα ίχνη επιλύεται με τον ουγγρικό αλγόριθμο, χρησιμοποιώντας έναν καταρράκτη αντιστοίχισης δύο τμημάτων. Στο πρώτο μέρος, η μέθοδος Deep-SORT χρησιμοποιεί μετρικές κίνησης και εμφάνισης για τη συσχέτιση έγκυρων διαδρομών. Το δεύτερο μέρος χρησιμοποιεί την ίδια στρατηγική συσχέτισης δεδομένων όπως και στο SORT για να συσχετίσει μη ταιριασμένα και δοκιμαστικά ίχνη (πρόσφατα δημιουργημένα) με μη ταιριασμένες ανιχνεύσεις. Οι πληροφορίες κίνησης ενσωματώνονται μέσω (τετραγωνικής) απόστασης Mahalanobis μεταξύ των προβλεπόμενων καταστάσεων και των ανιχνεύσεων. Εκτός από τη μετρική που υπολογίζεται με την απόσταση Mahalanobis, μια δεύτερη μετρική που βασίζεται στη μικρότερη απόσταση συνημίτονου μετρά την απόσταση μεταξύ κάθε ίχνους και κάθε χαρακτηριστικού εμφάνισης μέτρησης. Τα χαρακτηριστικά εμφάνισης υπολογίζονται από ένα προ-εκπαιδευμένο μοντέλο CNN. Το CNN στη μέθοδο DeepSort εκπαιδεύτηκε σε ένα σύνολο δεδομένων μεγάλης κλίμακας για τον επαναπροσδιορισμό προσώπων χρησιμοποιώντας βαθιά μάθηση μετρικής συνημίτονου [Pereira et al, 2022].

ΚΕΦΑΛΑΙΟ 3ο

3.Μεθοδολογία προσέγγισης

Η παρούσα διπλωματική εργασία πραγματεύεται τη δημιουργία μεθοδολογίας-εφαρμογής για την παρακολούθηση και ανίχνευση αντικειμένων και πιο συγκεκριμένα οχημάτων σε πραγματικό χρόνο, με τη χρήση εναέριων μέσων UAV. Τα συστήματα παρακολούθησης της οδικής κυκλοφορίας σε πραγματικό χρόνο διαδραματίζουν ζωτικό ρόλο στη διαχείριση της οδικής κυκλοφορίας, στον προγραμματισμό και στην πρόληψη συχνών κυκλοφοριακών συμφορήσεων, παραβιάσεων των κανόνων κυκλοφορίας και θανατηφόρων οδικών ατυχημάτων. Ωστόσο ο εντοπισμός και η παρακολούθηση δεν περιορίζεται μόνο στα οχήματα αλλά μπορεί να επεκταθεί και στους ανθρώπους για την πρόληψη διαφόρων καταστάσεων και την αποφυγή ατυχημάτων. Πιο συγκεκριμένα στην παρούσα εργασία χρησιμοποιήθηκαν state-of-the-art μέθοδοι CNN για τον εντοπισμό, την αναγνώριση και την παρακολούθηση οχημάτων που κινούνται σε καμπύλο οδικό τμήμα. Στο κεφάλαιο αυτό θα αναφερθούν αναλυτικά όλες οι μέθοδοι που ακολουθήθηκαν καθώς επίσης και τα διάφορα βήματα για την δημιουργία της εφαρμογής που έχει ως στόχο την επεξεργασία βιντεοσκοπημένων λήψεων για την εξαγωγή συμπερασμάτων σχετικά με τον εντοπισμό, αναγνώριση, παρακολούθηση και τελικά την τροχιά των οχημάτων κατά την κίνησή τους σε καμπύλο οδικό τμήμα.

3.1 Εντοπισμός οχημάτων

Το πρωταρχικό πρόβλημα που καλείται να αντιμετωπίσει κάποιος κατά τη δημιουργία μιας τέτοιας εφαρμογής είναι αυτό του εντοπισμού των οχημάτων και κατά συνέπεια το είδος του αλγορίθμου που θα χρησιμοποιήσει για το σκοπό αυτό. Αν και υπάρχουν πολλές μέθοδοι που θα μπορούσαν να χρησιμοποιηθούν για τον εντοπισμό σε πραγματικό χρόνο, αυτές που ξεχωρίζουν για την ακρίβεια και ταχύτητα είναι οι πιο σύγχρονες προσεγγίσεις των Deep Artificial Neural Networks και κυρίως των CNN, για τα οποία έγινε αναφορά στο προηγούμενο κεφάλαιο. Παλαιότερες προσεγγίσεις που στηρίζουν τον εντοπισμό, την ανίχνευση και την αναγνώριση αντικειμένων σε δύο στάδια έχουν πλέον ξεπεραστεί, καθώς οι επιδόσεις τους μειώνονται σημαντικά όταν ως είσοδος παρέχονται πολύπλοκα σύνολα χαρακτηριστικών. Από την άλλη οι πιο εξελιγμένοι αλγόριθμοι βαθιάς μηχανικής μάθησης διαθέτοντας αυξημένη επεξεργαστική ισχύ συνιστούν πλέον μονόδρομο για μια τέτοια εφαρμογή [Αραπέλλης, 2021]. Για το λόγο αυτό έγινε η επιλογή ενός state-of-the-art αλγορίθμου αυτού της yolo και πιο συγκεκριμένα της έκδοσης v4.

Στο σημείο αυτό είναι σημαντικό να αναφερθεί πως για την ορθή λήψη, εγκατάσταση και λειτουργία της yolo αλλά και του αλγορίθμου παρακολούθησης για τον οποίο γίνεται λόγος στη συνέχεια, κρίθηκε απαραίτητο πρώτα να γίνει επιλογή και εγκατάσταση όλων των απαραίτητων βιβλιοθηκών οι οποίες σαφώς θα υποστηρίζονται από την εκάστοτε γλώσσα προγραμματισμού που θα επιλέξει ο δημιουργός της εφαρμογής.

Επιλογή των κατάλληλων βιβλιοθηκών-OpenCV:

Για την υλοποίηση της εφαρμογής στην παρούσα εργασία έγινε χρήση της OpenCV (Open Source Computer Vision), η οποία είναι μια βιβλιοθήκη λογισμικού ανοιχτού κώδικα για την όραση υπολογιστών και τη μηχανική μάθηση. Δημιουργήθηκε για να παρέχει μια κοινή υποδομή για εφαρμογές όρασης υπολογιστών και να επιταχύνει τη χρήση της μηχανικής αντίληψης στα εμπορικά προϊόντα. Η βιβλιοθήκη διαθέτει περισσότερους από 2.500

βελτιστοποιημένους αλγορίθμους, οι οποίοι περιλαμβάνουν ένα ολόκληρο σύνολο τόσο κλασικών όσο και σύγχρονων αλγορίθμων υπολογιστικής όρασης και μηχανικής μάθησης. Διαθέτει C++, Python, Java και MATLAB ενώ υποστηρίζει Windows, Linux, Android και Mac OS. Συνεπώς κρίθηκε κατάλληλη για την πληθώρα επιλογών που δίνει στο χρήστη και κυρίως για το γεγονός ότι περιλαμβάνει και την yolov4, που αποτελεί και τον βασικό αλγόριθμο όσον αφορά τον εντοπισμό των οχημάτων που χρησιμοποιήθηκε στην εργασία. Η γλώσσα προγραμματισμού που επιλέχθηκε για την υλοποίηση ήταν η Python σε λειτουργικό σύστημα Windows. Επίσης, πέραν της OpenCV έγινε χρήση και κάποιων ακόμα βιβλιοθηκών όπως είναι η TensorFlow, η Numpy και η PIL για τις οποίες γίνεται λόγος στη συνέχεια:

Numpy

Η Numpy αποτελεί το θεμελιώδες πακέτο για επιστημονικούς υπολογισμούς στην Python. Πρόκειται για μια βιβλιοθήκη Python που παρέχει ένα αντικείμενο πολυδιάστατων πινάκων, διάφορα παράγωγα αντικείμενα και μια σειρά από ρουτίνες για γρήγορες πράξεις σε πίνακες, συμπεριλαμβανομένων μαθηματικών, λογικών, χειρισμού σχημάτων, ταξινόμησης, επιλογής εισόδου/εξόδου, διακριτών μετασχηματισμών Fourier, βασικών γραμμικών αλγεβρών, στατιστικών πράξεων, τυχαίας προσομοίωσης και πολλά ακόμα.

PIL

Πρόκειται για μια δωρεάν και ανοικτού κώδικα πρόσθετη βιβλιοθήκη για τη γλώσσα προγραμματισμού Python που προσθέτει υποστήριξη για το άνοιγμα, τον χειρισμό και την αποθήκευση πολλών διαφορετικών μορφών αρχείων εικόνας και βίντεο. Προσφέρει διάφορες τυποποιημένες διαδικασίες για την επεξεργασία εικόνων, ορισμένες από τις οποίες αφορούν χειρισμούς ανά εικονοστοιχείο, μάσκες και χειρισμό διαφάνειας και φιλτράρισμα εικόνας.

TensorFlow

Η TensorFlow είναι μια βιβλιοθήκη ανοικτού κώδικα που αναπτύχθηκε από την Google κυρίως για εφαρμογές βαθιάς μηχανικής μάθησης. Υποστηρίζει επίσης την παραδοσιακή μηχανική μάθηση. Αρχικά αναπτύχθηκε για μεγάλους αριθμητικούς υπολογισμούς, χωρίς να υπάρχει κατά νου η χρήση του στην βαθιά μάθηση. Ωστόσο, αποδείχθηκε πολύ χρήσιμη και για το κομμάτι αυτό, με αποτέλεσμα η Google να την διαθέσει ως ανοιχτό λογισμικό. Η TensorFlow δέχεται δεδομένα με τη μορφή πολυδιάστατων πινάκων υψηλών διαστάσεων που ονομάζονται τανυστές. Οι πολυδιάστατοι αυτοί πίνακες είναι πολύ εύχρηστοι στο χειρισμό μεγάλων ποσοτήτων δεδομένων. Λειτουργεί με βάση γραφήματα ροής δεδομένων που έχουν κόμβους και ακμές. Καθώς ο μηχανισμός εκτέλεσης έχει τη μορφή πινάκων, είναι πολύ πιο εύκολο να εκτελεστεί ο κώδικας TensorFlow με κατανεμημένο τρόπο σε ένα σύμπλεγμα υπολογιστών, με τη χρήση των GPUs. Οι εφαρμογές βαθιάς μάθησης είναι πολύ περίπλοκες, με τη διαδικασία εκπαίδευσης να απαιτεί πολλούς υπολογισμούς. Χρειάζεται, επομένως, πολύς χρόνος λόγω του μεγάλου μεγέθους των δεδομένων καθώς περιλαμβάνει αρκετές επαναληπτικές διαδικασίες, μαθηματικούς υπολογισμούς, πολλαπλασιασμούς πινάκων και άλλες πράξεις. Σε περίπτωση που αυτές οι δραστηριότητες εκτελούνται σε μια

κανονική μονάδα επεξεργασίας (CPU), συνήθως απαιτείται πολύς περισσότερος χρόνος. Οι μονάδες επεξεργασίας γραφικών (GPU) είναι δημοφιλείς στο πλαίσιο των παιχνιδιών, όπου απαιτείται υψηλής ανάλυσης εικόνα. Αν και σχεδιάστηκαν αρχικά για αυτόν αποκλειστικά το σκοπό εντούτοις χρησιμοποιούνται και για την ανάπτυξη εφαρμογών βαθιάς μάθησης με σημαντικά πλεονεκτήματα. Σημαντικό πλεονέκτημα της βιβλιοθήκης TensorFlow είναι ότι υποστηρίζει GPU και CPU, ενώ έχει επίσης και σημαντικά ταχύτερο χρόνο μεταγλώττισης από άλλες βιβλιοθήκες βαθιάς μάθησης, όπως οι Keras και Torch. [https://www.tensorflow.org/]
Πρέπει να αναφερθεί ότι όλη η υλοποίηση της εφαρμογής και των αλγορίθμων που την απαρτίζουν έγινε στο περιβάλλον της PyCharm(Community Edition 2021.2.1), το οποίο είναι ένα περιβάλλον ανάπτυξης εφαρμογών αποκλειστικά σε Python που παρέχει ένα ευρύ φάσμα βασικών εργαλείων για προγραμματιστές.

Εφόσον έχει επιλεγεί ο αλγόριθμος εντοπισμού, αναγνώρισης και κατηγοριοποίησης των οχημάτων όπως επίσης και των αναγκαίων για τη λειτουργία του βιβλιοθηκών, σειρά έχει η παρακολούθησή τους.

3.2 Παρακολούθηση οχημάτων και τροχιές

Το δεύτερο πρόβλημα που πρέπει να αντιμετωπιστεί είναι αυτό της παρακολούθησης των οχημάτων. Πιο συγκεκριμένα εφόσον ένα όχημα έχει εντοπιστεί από τη yolo θα πρέπει στη συνέχεια να του δοθεί ένα μοναδικό id, δηλαδή μια ταυτότητα. Με τον τρόπο αυτό ο αλγόριθμος θα αναγνωρίζει κάθε όχημα ως ξεχωριστό και θα το παρακολουθεί καθ όλη τη διάρκεια που αυτό βρίσκεται στην περιοχή μελέτης. Ο αλγόριθμος που χρησιμοποιείται συνήθως στην παρακολούθηση πολλαπλών αντικειμένων είναι ο DeepSort που αποτέλεσε και τον αλγόριθμο που χρησιμοποιήθηκε στην παρούσα εργασία ως tracker. Ως εκ τούτου ο αλγόριθμος DeepSort γνωστοποιεί στο χρήστη την ακριβή θέση όλων των οχημάτων σε κάθε καρέ του βίντεο, δίνοτας ένα σύνολο συντεταγμένων για κάθε ένα από αυτά.
Οι γνωστές πλέον τροχιές των οχημάτων αποτελούνται από συντεταγμένες που έχουν ως αρχή το επάνω αριστερά σημείο της εικόνας - βίντεο και είναι εκφρασμένες σε pixel. Το γεγονός αυτό συνιστά βασικό πρόβλημα για τον υπολογισμό της ταχύτητας των οχημάτων, κάτι για το οποίο γίνεται λόγος στη συνέχεια. Σχετικά με την μετατροπή των εικονοσυντεταγμένων από pixel (με αρχή το επάνω αριστερά σημείο της εικόνας) σε mm (με αρχή το κέντρο της εικόνας) αυτό έγινε με απλούς μαθηματικούς τύπους μετατροπής γνωρίζοντας το PixelSize και τα cols και rows των καρέ (frames) του βίντεο:

$$X= (Jimg - cols/2) * PixelSize$$

$$Y= (rows/2 - Iimg) * PixelSize$$

Όπου:

X, Y : εικονοσυντεταγμένες σε mm με αρχή το κέντρο της εικόνας

Cols, rows : πλήθος στηλών και σειρών της εικόνας (frame)

Jimg, Iimg : εικονοσυντεταγμένες σε pixel με αρχή το επάνω αριστερά σημείο της εικόνας

3.3 Μετατροπή συντεταγμένων σε ΕΓΣΑ 87'

Για την εξαγωγή ορθών συμπερασμάτων αναφορικά με τη διερεύνηση της τροχιάς οχημάτων κρίθηκε σκόπιμο να γίνει και μια εκτίμηση της ταχύτητάς τους, προκειμένου να παρέχεται στο χρήστη της εφαρμογής μια πιο συνολική εικόνα σχετικά με την πορεία του κάθε οχήματος. Για το λόγο αυτό ήταν απαραίτητη η μετατροπή των εικονο-συντεταγμένων των σημείων της τροχιάς στο σύστημα ΕΓΣΑ 87', δηλαδή σε γεωδαιτικές συντεταγμένες στο χώρο. Είναι σημαντικό να αναφερθεί πως βασική προϋπόθεση της μεθοδολογίας που ακολουθήθηκε στη συνέχεια είναι ότι το UAV πραγματοποιεί τη λήψη παραμένοντας σταθερό.

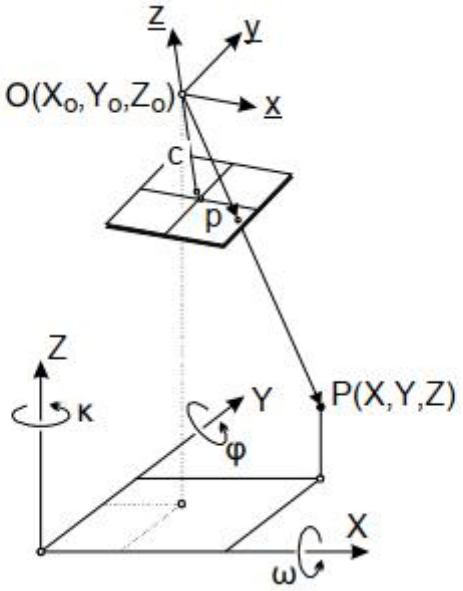
Για τη μετατροπή των συντεταγμένων έγινε χρήση της συνθήκης συγγραμμικότητας.

Συνθήκη συγγραμμικότητας (Σ.Σ): αποτελεί τη βασική εξίσωση της φωτογραμμετρίας και εκφράζει ουσιαστικά την κεντρική προβολή, συνδέοντας σημεία πάνω στη φωτογραφία με τα αντίστοιχα στο έδαφος. Είναι η σχέση που χρησιμοποιείται στις διαδικασίες της εμπροσθοτομίας, της οπισθοτομίας, της βαθμονόμησης και του φωτοτριγωνισμού με τη μέθοδο της δέσμης. Ως εκ τούτου, περιγράφει στροφές και μεταθέσεις που πρέπει να υποστεί το ένα σύστημα προκειμένου να ταυτιστεί με το άλλο. Οι βαθμοί ελευθερίας που υπάρχουν μεταξύ των δύο 3D συστημάτων συντεταγμένων είναι 6 (3 στροφές + 3 μεταθέσεις). Αυτά τα στοιχεία αντιπροσωπεύουν τα στοιχεία Xo,Yo,Zo,ω,φ,κ του εξωτερικού προσανατολισμού της εικόνας.

Όπου:

- Xo,Yo,Zo είναι οι συντεταγμένες χώρου του σημείου λήψης

- ω,φ,κ οι οι στροφές της εικόνας



Εικόνα 3.1: Σχηματική απεικόνιση της συνθήκης συγγραμμικότητας (Πέτσα, 2000).

Όπως φαίνεται και από το προηγούμενο σχήμα η συνθήκη συγγραμμικότητας εφράζει το γεγονός ότι το τυχαίο σημείο P του χώρου, το αντίστοιχο εικονοσημείο p και το σημείο λήψης O κείνται επ' ευθείας, δηλαδή είναι συνευθειακά [Πέτσα, 2000]. Μαθηματικά η συνθήκη εκφράζεται ως εξής:

$$\begin{bmatrix} x - x_o \\ y - y_o \\ -c \end{bmatrix} = \frac{1}{k} \boldsymbol{R} \begin{bmatrix} X - X_o \\ Y - Y_o \\ Z - Z_o \end{bmatrix} \quad (4.1)$$

Όπου **R** είναι ο πίνακας στροφής,

$$\mathbf{R} = \begin{bmatrix} \cos\varphi\cos\kappa & \cos\omega\sin\kappa + \sin\omega\sin\varphi\cos\kappa & \sin\omega\sin\kappa - \cos\omega\sin\varphi\cos\kappa \\ -\cos\varphi\sin\kappa & \cos\omega\cos\kappa - \sin\omega\sin\varphi\sin\kappa & \sin\omega\cos\kappa + \cos\omega\sin\varphi\sin\kappa \\ \sin\varphi & -\sin\omega\cos\varphi & \cos\omega\cos\varphi \end{bmatrix} \quad (4.2)$$

Λύνοντας την (4.1) ως προς τις γεωδαιτικές συντεταγμένες, προκύπτουν τελικά οι σχέσεις που είναι απαραίτητες για τη μετατροπή και που στη συνέχεια εισάγονται στον κώδικα.

$$X - X_o = (Z - Z_o)\frac{r_{11}(x - x_o) + r_{21}(y - y_o) - r_{31}c}{r_{13}(x - x_o) + r_{23}(y - y_o) - r_{33}c}$$
$$Y - Y_o = (Z - Z_o)\frac{r_{12}(x - x_o) + r_{22}(y - y_o) - r_{32}c}{r_{13}(x - x_o) + r_{23}(y - y_o) - r_{33}c} \quad (4.3)$$

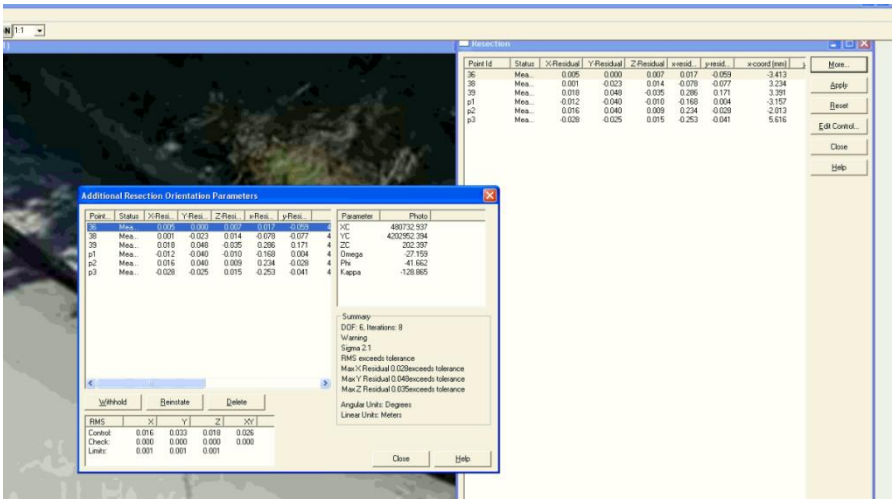
Για την επίλυση της (4.3) ως προς X,Y δηλαδή τις γεωδαιτικές συντεταγμένες έπρεπε πρώτα να προσδιοριστούν τα τρία βασικά στοιχεία του εσωτερικού προσανατολισμού της φωτομηχανής x_0, y_0, c και στη συνέχεια τα έξι στοιχεία X_0, Y_0, Z_0, ω ,φ, κ του εξωτερικού προσανατολισμού.

→ *Εσωτερικός προσανατολισμός*

Όσον αφορά τον εσωτερικό προσανατολισμό, στη συγκεκριμένη περίπτωση θεωρήσαμε σταθερά της μηχανής ίση με την εστιακή απόσταση η οποία δίνεται από τον κατασκευαστή (όλα τα χαρακτηριστικά της μηχανής αναφέρονται σε επόμενο κεφάλαιο). Ως πρωτεύον σημείο θεωρήθηκε στο κέντρο της εικόνας ενώ επίσης θεωρήθηκαν μηδενικές διαστροφές.

→ *Εξωτερικός προσανατολισμός*

Για τον υπολογισμό των στοιχείων του εξωτερικού προσανατολισμού η διαδικασία που ακολουθήθηκε είναι αυτή της φωτογραμμετρικής οπισθοτομίας κάνοντας χρήση των στοιχείων εσωτερικού προσανατολισμού και συντεταγμένων 6 φωτοσταθερών σημείων στο ΕΓΣΑ 87'. Τα σημεία αυτά μετρήθηκαν με GPS και ήταν διαθέσιμες οι επίγειες συντεταγμένες τους. Η οπισθοτομία σε αυτές τις περιπτώσεις μπορεί να επιλυθεί με ΜΕΤ (Μέθοδος Ελαχίστων Τετραγώνων). Στη συγκεκριμένη εργασία αυτό έγινε αυτόματα μέσω του λογισμικού ImageStation SSK, χρησιμοποιώντας ως εξισώσεις παρατήρησης τις εξισώσεις συγγραμμικότητας, βάσει των στοιχείων του εσωτερικού προσανατολισμού, των επίγειων συντεταγμένων των φωτοσταθερών σημείων (τα οποία και σκοπεύθηκαν) και των εικονοσυντεταγμένων τους.

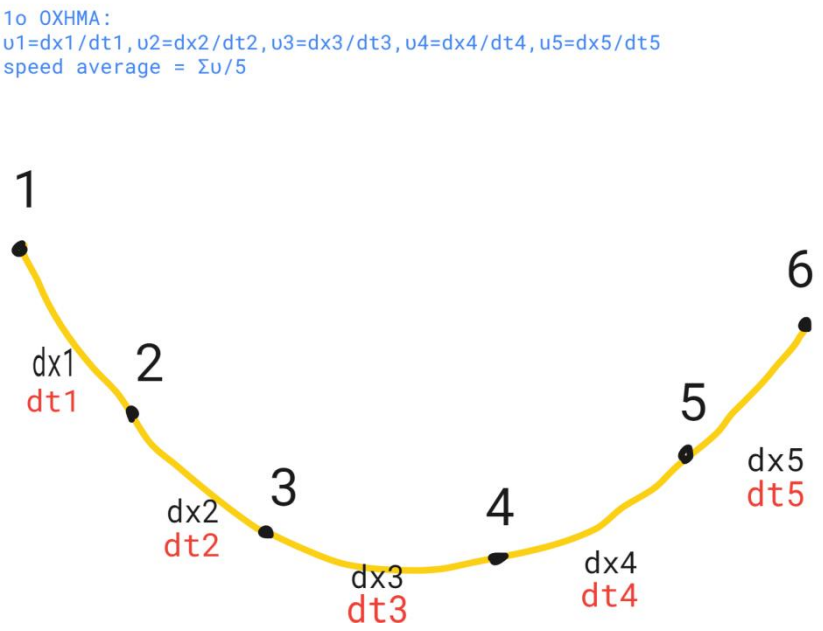


Εικόνα 3.2: Προσδιορισμός στοιχείων εξωτερικού προσανατολισμού μέσω του προγράμματος ImageStation SSK.

Αξίζει εδώ να σημειωθεί πως στην περίπτωση που το UAV που θα επιλεχθεί από τον χρήστη διαθέτει GPS-INS καλής ακρίβειας, ο εξωτερικός προσανατολισμός θα ήταν γνωστός και ως εκ τούτου, το στάδιο επίλυσης οπισθοτομίας παραλείπεται.

3.4 Εκτίμηση της ταχύτητας

Στο σημείο αυτό και εφόσον πλέον είναι γνωστές οι τροχιές των διερχόμενων οχημάτων στο ΕΓΣΑ 87’, μπορεί να υπολογιστεί η ταχύτητα τους. Για τον υπολογισμό της έγινε χρήση του πιο απλού μοντέλου υπολογισμού, αυτού της ευθύγραμμης ομαλής κίνησης, $U=Dx/Dt$. Με τον τρόπο αυτό και εφόσον η περιοχή μελέτης μας ήταν ένα καμπύλο οδικό τμήμα, προέκυψαν καμπύλες τροχιές οι οποίες διαιρέθηκαν σε πέντε μικρότερα τμήματα-segments (ανάλογα με τα συνολικά καρέ του βίντεο) ώστε να υπολογιστούν οι πέντε ταχύτητες, μία για κάθε τμήμα και κατόπιν μια μέση ταχύτητα για το κάθε όχημα, όπως φαίνεται στην παρακάτω εικόνα. Στην πράξη αυτό έγινε διαιρώντας τον συνολικό αριθμό frames του βίντεο με 5 ώστε να προκύψουν τα segments και στη συνέχεια εφαρμόστηκε η παραπάνω μεθοδολογία για το 1ο αμάξι. Προκειμένου να βρούμε τα αντίστοιχα frames και για τα επόμενα αμάξια εφαρμόστηκε μία συνθήκη στον κώδικα βαση της ελάχιστης απόστασης των τροχιών τους συγκριτικά με του πρώτου.



Εικόνα 3.3: Ενδεικτικός υπολογισμός ταχύτητας 1ου οχήματος.

Ακολουθώντας την ίδια λογική για όλα τα οχήματα που εμφανίζονται στο βίντεο (σύνολο 3) προέκυψαν οι εκτιμώμενες ταχύτητες τα αποτελέσματα των οποίων παρουσιάζονται στο κεφάλαιο 5.

ΚΕΦΑΛΑΙΟ 4ο

4. Εφαρμογή σε καμπύλο τμήμα δρόμου

Στο κεφάλαιο αυτό γίνεται η παρουσίαση και επεξήγηση της τελικής εφαρμογής όπως αυτή προέκυψε βάσει της προηγούμενης μεθοδολογίας. Για την τελική υλοποίηση ακολουθήθηκαν συγκεκριμένα στάδια που αφορούσαν τόσο την επιλογή των απαιτούμενων οργάνων και εξοπλισμού όσο και την λήψη, εγκατάσταση και οργάνωση αναγκαίων αλγορίθμων και αρχείων που θα διασφάλιζαν την ορθή λειτουργία της εφαρμογής.

4.1 Στάδια υλοποίησης εφαρμογής

1ο Στάδιο: Επιλογή απαραίτητου εξοπλισμού

UAV

Στο στάδιο αυτό κρίνεται απαραίτητο να επιλεγεί το UAV που θα χρησιμοποιηθεί για την καταγραφή του βίντεο, το οποίο κατόπιν θα εισαχθεί στον αλγόριθμο για επεξεργασία. Τα μη επανδρωμένα αεροσκάφη (UAV) που είναι εξοπλισμένα με κάμερες έχουν διαδοθεί πολύ γρήγορα βρίσκοντας εφαρμογή σε ένα ευρύ φάσμα εργασιών, όπως η γεωργία, η αεροφωτογραφία, η επιτήρηση κ.α. Κατά συνέπεια, η αυτόματη κατανόηση των οπτικών δεδομένων που συλλέγονται από αυτές τις πλατφόρμες γίνεται ιδιαίτερα απαιτητική, γεγονός που φέρνει την όραση υπολογιστών στα μη επανδρωμένα αεροσκάφη όλο και πιο κοντά. Παρά τις μεγάλες προόδους στους γενικούς αλγορίθμους υπολογιστικής όρασης, οι αλγόριθμοι αυτοί δεν είναι συνήθως βέλτιστοι για την αντιμετώπιση ακολουθιών ή εικόνων που λαμβάνονται από UAVs, λόγω διαφόρων προκλήσεων όπως οι αλλαγές στο σημείο θέασης και οι κλίμακες. Ωστόσο οι μελέτες προς αυτή την κατεύθυνση περιορίζονται σοβαρά από την έλλειψη δημοσίως διαθέσιμων μεγάλης κλίμακας συγκριτικών στοιχείων ή συνόλων δεδομένων. Παρόλα αυτά, η χρήση ενός UAV για την παρακολούθηση οχημάτων σε καμπύλο οδικό τμήμα που πραγματεύεται η συγκεκριμένη διπλωματική εργασία φαινόταν μονόδρομος τόσο για την ευκολία χρήσης του, μιας και θα δινόταν από το χρήστη το ακριβές σημείο πτήσης που αφενός εξυπηρετεί τους αλγόριθμους που χρησιμοποιούμε ώστε να είναι αποτελεσματικοί και αφετέρου επιτρέπει την πτήση σε ελεγχόμενο δρόμο εντός της Πολυτεχνειούπολης για την αποφυγή οποιουδήποτε ατυχήματος. Ως εκ τούτου το drone που επιλέχθηκε για την πραγματοποίηση των πτήσεων είναι ένα Mavic 2 Pro της DJI. Πρόκειται για ένα πολύ ικανό UAV καθώς παρέχει πληθώρα αυτοματισμών και λειτουργιών, ενώ παράλληλα το επίπεδο ασφαλείας και ελέγχου που παρέχει είναι υψηλό διευκολύνοντας το χρήστη. Παρακάτω αναφέρονται τα χαρακτηριστικά της κάμερας που φέρει, και που μας ενδιαφέρουν για την λήψη των βίντεο και κατόπιν της επεξεργασίας τους.

Sensor	1’’ CMOS Effective Pixels: 20 million
Lens	FOV: about 77° 35mm Format Equivalent: 28mm Aperture: f/2.8-f/11 Shooting Range: 1m to ∞
ISO Range	Video: 100-6400 Photo: 100-3200(auto)

	100-12800(manual)
Shutter Speed	Electronic Shutter: 8-1/8000s
Still Image Size	5472x3648
Still Photography Modes	Single shot Burst shooting: 3/5 frames Auto Exposure Bracketing(AEB): 3/5 bracketed frames at 0.7 EV Bias Interval (JPEG: 2/3/5/7/10/15/20/30/60s RAW: 5/7/10/15/20/30/60s
Video Resolution	4K: 3840x2160 24/25/30p 2.7K: 2688x1512 24/25/30/48/50/60p FHD: 1920x1080 24/25/30/48/50/60/120p
Max Video Bitrate	100Mbps
Color Mode	Dlog-M (10bit), support HDR video (HLG 10bit)
Supported File System	FAT32(≤32GB) ExFAT(>32GB)
Photo Format	JPEG/DNG (RAW)
Video Format	MP4 / MOV (MPEG-4 AVC/H.264, HEVC/H.265)

Σχήμα 4.1: Τεχνικά χαρακτηριστικά κάμερας Mavic 2 Pro

Hardware (H/Y)

Όπως γίνεται εύκολα αντιληπτό οι αλγόριθμοι που χρησιμοποιούνται στην όραση υπολογιστών απαιτούν μεγάλη υπολογιστική ισχύ τόσο για την επεξεργασία μεγάλου πλήθους δεδομένων, όσο και για την αύξηση της ταχύτητας επεξεργασίας, κάτι που επηρεάζει άμεσα την διαδικασία δημιουργίας μιας τέτοιας εφαρμογής που απαιτεί πολλαπλές επαναλήψεις καθόλη τη διάρκεια δημιουργίας για την εύρεση λαθών, την διόρθωσή τους και τελικά την εξαγωγή συμπερασμάτων. Όλες οι εταιρείες και οι πάροχοι τέτοιων αλγορίθμων διαθέτουν μεγάλα ποσά για τη σύνθεση H/Y που θα μπορούν να διαχειριστούν εργασίες εκπαίδευσης και σύνθεσης τέτοιων μοντέλων. Ενδεικτικά αναφέρεται πως οι μετρήσεις και η δημιουργία και ανάπτυξη της yolov4 έγιναν με τη χρήση μιας Nvidia Tesla v100 16GB από τους αρχικούς εκδότες αυτού του αλγορίθμου. Πρόκειται για μια κάρτα το κόστος της οποίας αγγίζει τα 8000 ευρώ, αποδεικνύοντας το αυξημένο κόστος και την ανάγκη για μέγιστη ισχύ με στόχο υψηλές επιδόσεις. Στην παρούσα εργασία έγινε χρήση της νεότερης έκδοσης yolov4 μέσω της βιβλιοθήκης OpenCV. Η OpenCV είναι πλέον ικανή να εκτελεί την yolov4 εγγενώς με την μονάδα DNN χρησιμοποιώντας όλα τα πλεονεκτήματα της NVIDIA CUDA. Το CUDA είναι μια πλατφόρμα παράλληλων υπολογισμών και ένα μοντέλο προγραμματισμού που επιτρέπει δραματικές αυξήσεις στις υπολογιστικές επιδόσεις με την αξιοποίηση της ισχύος της μονάδας επεξεργασίας γραφικών (GPU). Ως εκ τούτου γίνεται αντιληπτό ότι όσο πιο μεγάλη η ισχύς της GPU τόσο και πιο γρήγορα και αποτελεσματικά τρέχουν οι εν λόγω αλγόριθμοι μειώνοντας σημαντικά το χρόνο επεξεργασίας των δεδομένων, στην δική μας περίπτωση του βίντεο στο οποίο γίνεται η προ-επεξεργασία και η μετα-επεξεργασία των οπτικών στοιχείων. Κατά την εκπόνηση της διπλωματικής εργασίας, συγγραφής και υλοποίησης, οι αλγόριθμοι που χρησιμοποιήθηκαν για τη δημιουργία της

εφαρμογής δοκιμάστηκαν σε υπολογιστή με Ryzen 5 3600 και κάρτα γραφικών της nvidia 1060 3gb, δίνοντας ικανοποιητική υπολογιστική ισχύ και ταχύτητα.

2ο Στάδιο: Εγκατάσταση της OpenCV

Προκειμένου να γίνει χρήση της βιβλιοθήκης OpenCV έπρεπε πρώτα να εγκατασταθεί επιτυχώς η Python και το pip στον Η/Υ. Με τις παρακάτω εντολές γίνεται ο έλεγχος για την επιτυχή εγκατάσταση τους.

To check Python

```
python --version
```

If python is successfully installed, the version of python installed on your system will be displayed.

To check pip

```
pip -V
```

The version of pip will be displayed, if it is successfully installed on your system.

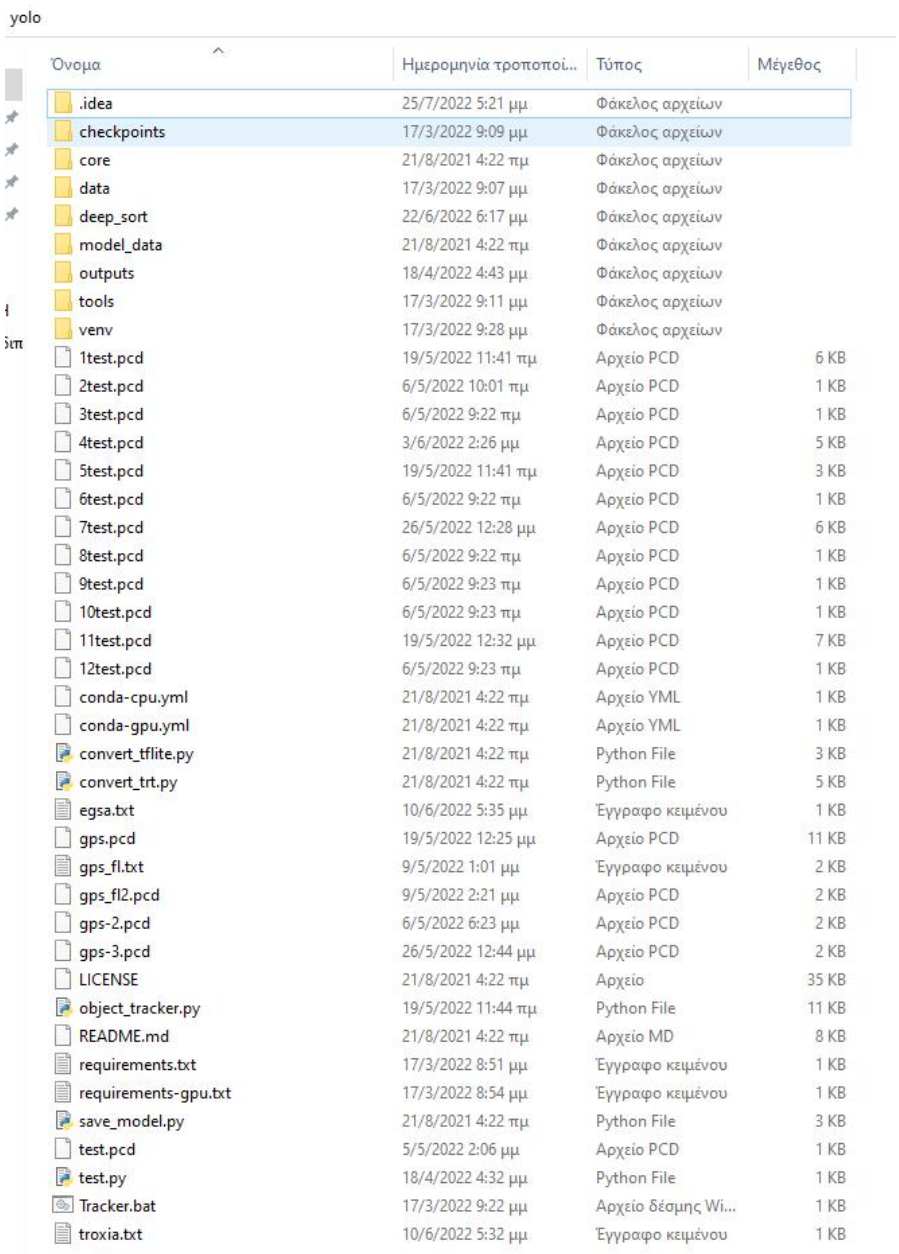
Το pip είναι ο τυπικός διαχειριστής πακέτων για την γλώσσα Python. Επιτρέπει την εγκατάσταση και διαχείριση πακέτων που δεν αποτελούν μέρος της τυπικής βιβλιοθήκης της Python. Με τις παρακάτω εντολές γίνεται ο έλεγχος που αφορά την επιτυχή εγκατάσταση τους.
Έπειτα γίνεται και η εγκατάσταση της OpenCV με χρήση του pip και της κατάλληλης εντολής:

```
pip install opencv-python
```

Η τελευταία θα ξεκινήσει τη λήψη και εγκατάσταση πακέτων που σχετίζονται με τη βιβλιοθήκη OpenCV και κατόπιν εμφανίζεται σχετικό μήνυμα της επιτυχούς εγκατάστασης.

3ο Στάδιο: Υλοποίηση του αλγορίθμου της Yolov4

Στο στάδιο αυτό έγινε χρήση των απαραίτητων αλγορίθμων και εργαλείων για την υλοποίηση της Yolov4 μέσω της σελίδας των συγγραφέων στο Github (AlexeyAB – github), από όπου και αντλήθηκαν όλα τα αρχεία με τις αντίστοιχες οδηγίες για την εγκατάσταση της Yolov4. Το Github είναι ένας ιστότοπος-υπηρεσία τύπου cloud που βοηθά τους προγραμματιστές να αποθηκεύουν και να διαχειρίζονται κώδικες, καθώς και να παρακολουθούν και να ελέγχουν αλλαγές στον κώδικά τους. Προκειμένου να διευκολυνθεί η όλη διαδικασία όλα τα αρχεία οργανώθηκαν σε ένα συγκεκριμένο φάκελο εργασίας (main root) με το όνομα yolo, όπως συνηθίζεται σε τέτοιες περιπτώσεις για την αποφυγή λαθών κατά την εργασία. Για να δουλέψει αυτός ο αλγόριθμος έπρεπε όλα τα αρχεία, φάκελοι, υποφάκελοι κ.ο.κ να δημιουργηθούν τοπικά στον Η/Υ που είχαμε στην διάθεσή μας.



Εικόνα 4.1: Βασικός φάκελος εργασίας με όλα τα απαραίτητα αρχεία

Στα αρχεία αυτά περιλαμβάνονται και όλες οι κλάσεις στις οποίες έχει εκπαιδευτεί να αναγνωρίζει η yolo, τα οποία είναι περίπου 80 στο σύνολό τους και καλύπτουν μια ευρεία γκάμα αντικειμένων. Από αυτά έγινε επιλογή μόνο αυτών που μας ενδιαφέρουν στη συγκεκριμένη εργασία και αφορούν οχήματα που κινούνται κατά μήκος μιας καμπύλης (car, trucks). Η yolo έχει εκπαιδευτεί σύμφωνα με τους συγγραφείς στο Common Object Dataset (COCO dataset) με περισσότερες από 80.000.000 εικόνες στις 80 αυτές κλάσεις. Ωστόσο δίνεται η δυνατότητα εάν κάποια εφαρμογή δεν καλύπτεται από αυτές τις κλάσεις, ο αλγόριθμος να εκπαιδευτεί από την αρχή και μάλιστα με μεγαλύτερες ακόμα ακρίβειες.
Οι φωτογραφίες οχημάτων που χρησιμοποιήθηκαν για την εκπαίδευση του αλγορίθμου ήταν κατά βάση από το έδαφος και επομένως σε κοντινές αποστάσεις από αυτά. Για το λόγο αυτό και προκειμένου να αποφευχθεί η επανεκπαίδευση του αλγορίθμου, μιας και κάτι τέτοιο θα

ξέφευγε από τα πλαίσια αυτής της εργασίας, απαιτείται το drone από το οποίο γίνεται η λήψη να πετάει σε χαμηλό υψόμετρο της τάξης των 10-15 μέτρων και να πραγματοποιεί λήψη των video όχι κατακόρυφα αλλά υπό γωνία. Με τον τρόπο αυτό ο αλγόριθμος θα δίνει την απαραίτητη ακρίβεια χωρίς αστοχίες, μιας και τα video που λαμβάνονται “προσεγγίζουν” τις φωτογραφίες στις οποίες έχει εκπαιδευτεί.
Σημαντικό βήμα επίσης για την υλοποίηση της εφαρμογής είναι και η ενημέρωση των drivers της κάρτας γραφικών GPU για την ορθή λειτουργία κατά την διαδικασία επεξεργασίας., κάτι το οποίο έγινε μέσω της εφαρμογής της nvidia. Κατόπιν έγινε η εγκατάσταση της CUDA που επιτρέπει στην yolov4 να τρέξει μέσω της κάρτας γραφικών, αξιοποιώντας όλα τα πλεονεκτήματα που αναφέρθηκαν προηγουμένως. Η CUDA είναι μια πλατφόρμα παράλληλων υπολογιστών και ένα μοντέλο προγραμματισμού που δημιουργήθηκε από την NVIDIA. Έχοντας περισσότερες από 20 εκατομμύρια λήψεις μέχρι σήμερα, βοηθά τους προγραμματιστές να επιταχύνουν τις εφαρμογές τους αξιοποιώντας τη δύναμη των επιταχυντών των μονάδων επεξεργασίας γραφικών (GPUs). Στο πλαίσιο της πλατφόρμας CUDA συγκαταλέγεται και το cuDNN, μια βιβλιοθήκη που επιταχύνει τις GPUs με παρόμοιο τρόπο, σε διαδικασίες με χρήση βαθιών νευρωνικών δικτύων, παρέχοντας συντονισμένες υλοποιήσεις ρουτινών που εμφανίζονται πολύ συχνά σε ανάλογες εφαρμογές DNN.

4ο Στάδιο: Υλοποίηση του αλγορίθμου Deepsort

Στο συγκεκριμένο στάδιο έγινε η υλοποίηση του Deepsort, αφού πρώτα κατέβηκαν τα απαραίτητα αρχεία από τη σελίδα των συγγραφέων στο Github (github-nwojke). Εκεί παρέχονται όλα τα στοιχεία και οι οδηγίες που αφορούν την εγκατάσταση για την υλοποίηση του αλγορίθμου αλλά και για την εκπαίδευσή του, καθώς και το προεκπαιδευμένο μοντέλο. Αρχικά έπρεπε να γίνει η εγκατάσταση όλων των αρχείων που απαρτίζουν τον Deepsort, και είναι ουσιαστικά όλα τα στάδιά του, τα οποία πρακτικά εκτελούνται από ένα συγκεντρωτικό κώδικα. Όπως και με την YOLO-v4, αυτά τοποθετήθηκαν στον φάκελο εργασίας με το όνομα yolo και είναι τα παρακάτω:

- *detection.py*: detection base class.

- *iou_matching.py*: *kalman_filter.py*: this module contains the IOU matching metric.

- *linear_assignment.py*: this module contains code for min cost matching and matching cascade.

- *nn_matching.py*: a module for a nearest neighbor matching metric.

- *track.py*: the track class contains single-target track data such as Kalman state, number of hits, misses, hit streak, associated feature vectors, etc.

- *tracker.py*: this is the multri-target tracker class.

Προκειμένου στη συνέχεια να υλοποιηθεί η παρακολούθηση αντικειμένων με χρήση της YOLO-v4, έπρεπε να μετατραπούν τα weights στο αντίστοιχο μοντέλο TensorFlow, το οποίο αποθηκεύτηκε σε έναν φάκελο checkpoints. Στη συνέχεια εκτελώντας το script object_tracker.pt τρέχει ο ανιχνευτής αντικειμένων μαζί με την YOLO-v4, το DeepSort και το TensorFlow. Το βίντεο που προκύπτει από το πρόγραμμα παρακολούθησης αποθηκεύεται

στον υποφάκελο με όνομα outputs, ώστε να μπορεί να προβληθεί ξανά και να γίνει όποια επεξεργασία και ανάλυση κρίνεται απαραίτητη από το χρήστη.

5ο Στάδιο: Υλοποίηση τελικής εφαρμογής με συνδυασμό Yolov4-DeepSort

Αφού όλα τα αρχεία έχουν εγκατασταθεί επιτυχώς και οι βασικοί αλγόριθμοι είναι έτοιμοι προς χρήση, είναι πλέον εφικτό να δημιουργηθεί ο τελικός αλγόριθμος της εν λόγω εφαρμογής. Πρακτικά θα εισάγουμε σε αυτόν ως input το βίντεο που προέκυψε από την πτήση με το drone, και αφού ολοκληρωθεί επιτυχώς η επεξεργασία θα μας δώσει σαν output το ίδιο βίντεο στο οποίο θα φαίνεται επακριβώς η τροχιά των οχημάτων που διασχίζουν το τμήμα δρόμου μελέτης.

Επιπλέον, εκτός από την σχηματική απεικόνιση της τροχιάς των οχημάτων από τον αλγόριθμο προκύπτουν και οι συντεταγμένες, σε κάθε καρέ του βίντεο, των κέντρων των bounding boxes που περικλείουν τα εντοπισμένα οχήματα όπως επίσης και το μοναδικό track id για κάθε ένα από αυτά. Με αυτό τον τρόπο εξασφαλίζεται η ακριβής τροχιά των οχημάτων για την εξαγωγή συμπερασμάτων στη συνέχεια. Μια ακόμα προσθήκη στον κώδικα ήταν η εκτίμηση της ταχύτητας κάθε αυτοκινήτου ακολουθώντας την μεθοδολογία του προηγούμενου κεφαλαίου. Αφού έγινε η μετατροπή των συντεταγμένων των σημείων της κάθε τροχιάς σε συντεταγμένες στο ΕΓΣΑ 87’ ελέγχθηκε η εγκυρότητά τους μέσω της σελίδας του Ελληνικού κτηματολογίου, όπου παρέχεται στο χρήστη η δυνατότητα εισόδου συντεταγμένων στο ΕΓΣΑ 87’ δίνοντας την ακριβή τοποθεσία στο χώρο. Με τον τρόπο αυτό υπολογίστηκαν τα διαστήματα Dx1,Dx2, … , Dx5 για όλα τα αμάξια μέσω του γνωστού τύπου της απόστασης δύο σημείων. Ακολούθησε ο υπολογισμός των αντίστοιχων χρονικών διαστημάτων Dt1,Dt2, … , Dt5, και τελικά ο υπολογισμός μιας μέσης προσεγγιστικής ταχύτητας (Speed Average) για το κάθε αυτοκίνητο που εισέρχεται στον κόμβο μελέτης κατά τη διάρκεια της πτήσης.

4.2 Παρουσίαση του κώδικα της εφαρμογής

Στο συγκεκριμένο υποκεφάλαιο κρίνεται σκόπιμο να γίνει παρουσίαση όλου του κώδικα με επεξήγηση των βασικών βημάτων του, περιγράφοντας όλα όσα αναφέρθηκαν παραπάνω για την υλοποίηση του. Αρχείο *object_tracker.py:*

ΕΙΣΑΓΩΓΗ ΑΠΑΡΑΙΤΗΤΩΝ ΓΙΑ ΤΗΝ ΕΦΑΡΜΟΓΗ ΒΙΒΛΙΟΘΗΚΩΝ

```
import open3d as o3d
import os
# comment out below line to enable tensorflow logging outputs
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '3'
import time
import tensorflow as tf
physical_devices = tf.config.experimental.list_physical_devices('GPU')
if len(physical_devices) > 0:
    tf.config.experimental.set_memory_growth(physical_devices[0], True)
from absl import app, flags, logging
from absl.flags import FLAGS
import core.utils as utils
from core.yolov4 import filter_boxes
from tensorflow.python.saved_model import tag_constants
from core.config import cfg
from PIL import Image
import cv2
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.compat.v1 import ConfigProto
from tensorflow.compat.v1 import InteractiveSession
# deep sort imports
from deep_sort import preprocessing, nn_matching
from deep_sort.detection import Detection
from deep_sort.tracker import Tracker
from tools import generate_detections as gdet
```

ΠΙΘΑΝΕΣ ΕΝΤΟΛΕΣ ΠΑΡΑΜΕΤΡΟΠΟΙΗΣΗΣ

```
flags.DEFINE_string('framework', 'tf', '(tf, tflite, trt')
flags.DEFINE_string('weights', './checkpoints/yolov4-416',
                    'path to weights file')
flags.DEFINE_integer('size', 416, 'resize images to')
flags.DEFINE_boolean('tiny', False, 'yolo or yolo-tiny')
flags.DEFINE_string('model', 'yolov4', 'yolov3 or yolov4')
flags.DEFINE_string('video', './data/video/drone_video4.mp4', 'path to input video or set to 0 for webcam')
flags.DEFINE_string('output', './outputs/out.mp4', 'path to output video')
flags.DEFINE_string('output_format', 'XVID', 'codec used in VideoWriter when saving video to file')
flags.DEFINE_float('iou', 0.45, 'iou threshold')
flags.DEFINE_float('score', 0.50, 'score threshold')
flags.DEFINE_boolean('dont_show', False, 'dont show video output')
flags.DEFINE_boolean('info', False, 'show detailed info of tracked objects')
flags.DEFINE_boolean('count', False, 'count objects being tracked on screen')
```

ΟΡΙΣΜΟΣ ΠΑΡΑΜΕΤΡΩΝ (DEEPSORT)

```
def main(_argv):

    # Definition of the parameters
    max_cosine_distance = 0.4
    nn_budget = None
    nms_max_overlap = 1.0
```

ΑΡΧΙΚΟΠΟΙΗΣΗ ΤΟΥ DEEPSORT

```
# initialize deep sort
model_filename = 'model_data/mars-small128.pb'
encoder = gdet.create_box_encoder(model_filename, batch_size=1)
# calculate cosine distance metric
metric = nn_matching.NearestNeighborDistanceMetric("cosine", max_cosine_distance, nn_budget)
# initialize tracker
tracker = Tracker(metric)
```

ΔΙΑΜΟΡΦΩΣΗ ΦΟΡΤΙΟΥ ΓΙΑ ΤΟΝ ΑΝΙΧΝΕΥΤΗ ΑΝΤΙΚΕΙΜΕΝΩΝ (YOLOv4)

```
# load configuration for object detector
config = ConfigProto()
config.gpu_options.allow_growth = True
session = InteractiveSession(config=config)
STRIDES, ANCHORS, NUM_CLASS, XYSCALE = utils.load_config(FLAGS)
input_size = FLAGS.size
video_path = FLAGS.video
```

ΕΙΣΑΓΩΓΗ ΒΑΡΩΝ YOLOv4 ΚΑΙ ΕΝΑΡΞΗ ΚΑΤΑΓΡΑΦΗΣ VIDEO

```
# load tflite model if flag is set
if FLAGS.framework == 'tflite':
    interpreter = tf.lite.Interpreter(model_path=FLAGS.weights)
    interpreter.allocate_tensors()
    input_details = interpreter.get_input_details()
    output_details = interpreter.get_output_details()
    print(input_details)
    print(output_details)
# otherwise load standard tensorflow saved model
else:
    saved_model_loaded = tf.saved_model.load(FLAGS.weights, tags=[tag_constants.SERVING])
    infer = saved_model_loaded.signatures['serving_default']

# begin video capture
try:
    vid = cv2.VideoCapture(int(video_path))
except:
    vid = cv2.VideoCapture(video_path)

out = None
```

ΑΠΟΘΗΚΕΥΣΗ ΤΟΥ VIDEO ΚΑΙ ΚΑΘΟΡΙΣΜΟΣ ΤΩΝ ΠΑΡΑΜΕΤΡΩΝ ΤΟΥ

```
# get video ready to save locally if flag is set
if FLAGS.output:
    # by default VideoCapture returns float instead of int
    width = int(vid.get(cv2.CAP_PROP_FRAME_WIDTH))
    height = int(vid.get(cv2.CAP_PROP_FRAME_HEIGHT))
    fps = int(vid.get(cv2.CAP_PROP_FPS))
    codec = cv2.VideoWriter_fourcc(*FLAGS.output_format)
    out = cv2.VideoWriter(FLAGS.output, codec, fps, (width, height))
```

ΕΝΑΡΞΗ ΕΠΕΞΕΡΓΑΣΙΑΣ ΤΟΥ VIDEO ΚΑΙ ΕΙΣΑΓΩΓΗ ΤΩΝ FRAMES ΣΤΗΝ YOLOv4

```
# while video is running
while True:
    return_value, frame = vid.read()
    if return_value:
        frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        image = Image.fromarray(frame)
    else:
        print('Video has ended or failed, try a different video format!')
        break
    frame_num +=1
    print('Frame #: ', frame_num)
    frame_size = frame.shape[:2]
    image_data = cv2.resize(frame, (input_size, input_size))
    image_data = image_data / 255.
    image_data = image_data[np.newaxis, ...].astype(np.float32)
    start_time = time.time()

    # run detections on tflite if flag is set
    if FLAGS.framework == 'tflite':
        interpreter.set_tensor(input_details[0]['index'], image_data)
        interpreter.invoke()
        pred = [interpreter.get_tensor(output_details[i]['index']) for i in range(len(output_details))]
        # run detections using yolov3 if flag is set
        if FLAGS.model == 'yolov3' and FLAGS.tiny == True:
            boxes, pred_conf = filter_boxes(pred[1], pred[0], score_threshold=0.25,
                                            input_shape=tf.constant([input_size, input_size]))
        else:
            boxes, pred_conf = filter_boxes(pred[0], pred[1], score_threshold=0.25,
                                            input_shape=tf.constant([input_size, input_size]))
    else:
        batch_data = tf.constant(image_data)
        pred_bbox = infer(batch_data)
        for key, value in pred_bbox.items():
            boxes = value[:, :, 0:4]
            pred_conf = value[:, :, 4:]
```

```
    boxes, scores, classes, valid_detections = tf.image.combined_non_max_suppression(
        boxes=tf.reshape(boxes, (tf.shape(boxes)[0], -1, 1, 4)),
        scores=tf.reshape(
            pred_conf, (tf.shape(pred_conf)[0], -1, tf.shape(pred_conf)[-1])),
        max_output_size_per_class=50,
        max_total_size=50,
        iou_threshold=FLAGS.iou,
        score_threshold=FLAGS.score
    )
```

ΜΕΤΑΤΡΟΠΗ ΔΕΔΟΜΕΝΩΝ ΣΕ ΠΙΝΑΚΕΣ NUMPY ΚΑΙ ΑΠΟΚΟΠΗ ΑΧΡΗΣΙΜΟΠΟΙΗΤΩΝ ΣΤΟΙΧΕΙΩΝ

```
# convert data to numpy arrays and slice out unused elements
num_objects = valid_detections.numpy()[0]
bboxes = boxes.numpy()[0]
bboxes = bboxes[0:int(num_objects)]
scores = scores.numpy()[0]
scores = scores[0:int(num_objects)]
classes = classes.numpy()[0]
classes = classes[0:int(num_objects)]
```

ΕΠΕΞΕΡΓΑΣΙΑ ΤΩΝ ΣΥΝΤΕΤΑΓΜΕΝΩΝ ΑΠΟ ΤΗ YOLOv4 ΚΑΙ ΕΠΙΛΟΓΗ ΤΩΝ ΚΛΑΣΕΩΝ ΕΝΔΙΑΦΕΡΟΝΤΟΣ

```
# format bounding boxes from normalized ymin, xmin, ymax, xmax ---> xmin, ymin, width, height
original_h, original_w, _ = frame.shape
bboxes = utils.format_boxes(bboxes, original_h, original_w)

# store all predictions in one parameter for simplicity when calling functions
pred_bbox = [bboxes, scores, classes, num_objects]

# read in all class names from config
class_names = utils.read_class_names(cfg.YOLO.CLASSES)

# by default allow all classes in .names file
#allowed_classes = list(class_names.values())

# custom allowed classes (uncomment line below to customize tracker for only people)
allowed_classes = ['car','truck']
```

ΕΛΕΓΧΟΣ ΓΙΑ ΔΙΑΓΡΑΦΗ ΑΝΤΙΚΕΙΜΕΝΩΝ ΠΟΥ ΔΕΝ ΑΝΗΚΟΥΝ ΣΤΙΣ ΕΠΙΛΕΓΜΕΝΕΣ ΚΛΑΣΕΙΣ

```
# loop through objects and use class index to get class name, allow only classes in allowed_classes list
names = []
deleted_indx = []
for i in range(num_objects):
    class_indx = int(classes[i])
    class_name = class_names[class_indx]
    if class_name not in allowed_classes:
        deleted_indx.append(i)
    else:
        names.append(class_name)
names = np.array(names)
count = len(names)
if FLAGS.count:
    cv2.putText(frame, "Objects being tracked: {}".format(count), (5, 35), cv2.FONT_HERSHEY_COMPLEX_SMALL, 2, (0, 255, 0), 2)
    print("Objects being tracked: {}".format(count))
# delete detections that are not in allowed_classes
bboxes = np.delete(bboxes, deleted_indx, axis=0)
scores = np.delete(scores, deleted_indx, axis=0)
```

ΚΩΔΙΚΟΠΟΙΗΣΗ ΤΩΝ ΑΝΤΙΚΕΙΜΕΝΩΝ ΠΟΥ ΑΝΙΧΝΕΥΤΗΚΑΝ ΑΠΟ ΤΗΝ YOLOv4 ΚΑΙ ΕΙΣΟΔΟΣ ΣΤΟΝ DEEPSORT

```
# encode yolo detections and feed to tracker
features = encoder(frame, bboxes)
detections = [Detection(bbox, score, class_name, feature) for bbox, score, class_name, feature in zip(bboxes, scores, names, features)]

#initialize color map
cmap = plt.get_cmap('tab20b')
colors = [cmap(i)[:3] for i in np.linspace(0, 1, 20)]
```

ΕΚΤΕΛΕΣΗ non-maxima supression

```
# run non-maxima supression
boxs = np.array([d.tlwh for d in detections])
scores = np.array([d.confidence for d in detections])
classes = np.array([d.class_name for d in detections])
indices = preprocessing.non_max_suppression(boxs, classes, nms_max_overlap, scores)
detections = [detections[i] for i in indices]
```

ΕΚΤΕΛΕΣΗ ΤΟΥ DEEPSORT TRACKER, ΣΧΕΔΙΑΣΜΟΣ ΤΟΥ BBOX ΚΑΙ ΤΗΣ ΤΡΟΧΙΑΣ ΓΙΑ ΚΑΘΕ ΑΝΤΙΚΕΙΜΕΝΟ ΠΟΥ ΑΚΟΛΟΥΘΕΙΤΑΙ ΚΑΙ ΕΚΤΥΠΩΣΗ ΣΧΕΤΙΚΟΥ ΜΗΝΥΜΑΤΟΣ ΜΕ ΠΛΗΡΟΦΟΡΙΕΣ ΓΙΑ ΤΟ ΚΑΘΕ ΑΝΤΙΚΕΙΜΕΝΟ

```
# Call the tracker
tracker.predict()
tracker.update(detections)

# update tracks
for track in tracker.tracks:
    if not track.is_confirmed() or track.time_since_update > 1:
        continue
    bbox = track.to_tlbr()
    class_name = track.get_class()

# draw bbox on screen
    color = colors[int(track.track_id) % len(colors)]
    color = [i * 255 for i in color]
    cv2.rectangle(frame, (int(bbox[0]), int(bbox[1])), (int(bbox[2]), int(bbox[3])), color, 2)
    cv2.rectangle(frame, (int(bbox[0]), int(bbox[1]-30)), (int(bbox[0])+(len(class_name)+len(str(track.track_id)))*17, int(bbox[1])), color, -1)
    cv2.putText(frame, class_name + "-" + str(track.track_id),(int(bbox[0]), int(bbox[1]-10)),0, 0.75, (255,255,255),2)

# draw traces
    pts = np.array([track.centers], np.int32)
    cv2.polylines(frame, [pts], False, color, 2)

# if enable info flag then print details about each track
    if FLAGS.info:
        print("Tracker ID: {}, Class: {},  BBox Coords (xmin, ymin, xmax, ymax): {}".format(str(track.track_id), class_name, (int(bbox[0]), int(bbox[1]), int(bbox[2]), int(bbox[3]))))

# calculate frames per second of running detections
fps = 1.0 / (time.time() - start_time)
print("FPS: %.2f" % fps)
result = np.asarray(frame)
result = cv2.cvtColor(frame, cv2.COLOR_RGB2BGR)
```

ΥΠΟΛΟΓΙΣΜΟΣ ΚΑΡΕ ΑΝΑ ΔΕΥΤΕΡΟΛΕΠΤΟ ΓΙΑ ΟΛΟ ΤΟ ΕΠΕΞΕΡΓΑΣΜΕΝΟ ΒΙΝΤΕΟ ΚΑΙ ΕΞΑΓΩΓΗ ΑΥΤΟΥ ΜΕ ΤΟ ΟΝΟΜΑ “Output Video”

```
# calculate frames per second of running detections
fps = 1.0 / (time.time() - start_time)
print("FPS: %.2f" % fps)
result = np.asarray(frame)
result = cv2.cvtColor(frame, cv2.COLOR_RGB2BGR)

if not FLAGS.dont_show:
    cv2.imshow("Output Video", result)
```

ΤΕΡΜΑΤΙΣΜΟΣ ΕΦΑΡΜΟΓΗΣ ΕΙΤΕ ΑΥΤΟΜΑΤΑ ΜΕ ΤΟ ΤΕΛΟΣ ΤΟΥ ΒΙΝΤΕΟ Η ΧΕΙΡΟΚΙΝΗΤΑ ΚΑΙ ΑΠΟΘΗΚΕΥΣΗ

```
        # if output flag is set, save video file
        if FLAGS.output:
            out.write(result)
        if cv2.waitKey(1) & 0xFF == ord('q'): break
    cv2.destroyAllWindows()

if __name__ == '__main__':
    try:
        app.run(main)
    except SystemExit:
        pass
```

Αρχείο *track.py:*

ΕΙΣΑΓΩΓΗ ΒΙΒΛΙΟΘΗΚΩΝ ΚΑΙ ΚΛΑΣΗ TrackState

```
# vim: expandtab:ts=4:sw=4
from array import array
from ctypes import pointer

import copy
from turtledemo.minimal_hanoi import Disc

import open3d as o3d
import numpy as np
import math
import random

from fontTools.varLib.mutator import curr
from setuptools.command.egg_info import egg_info


class TrackState:
    """
    Enumeration type for the single target track state. Newly created tracks are
    classified as `tentative` until enough evidence has been collected. Then,
    the track state is changed to `confirmed`. Tracks that are no longer alive
    are classified as `deleted` to mark them for removal from the set of active
    tracks.

    """

    Tentative = 1
    Confirmed = 2
    Deleted = 3
```

ΚΛΑΣΗ TRACK

```
class Track:
    """
    A single target track with state space `(x, y, a, h)` and associated
    velocities, where `(x, y)` is the center of the bounding box, `a` is the
    aspect ratio and `h` is the height.

    Parameters
    ----------
    mean : ndarray
        Mean vector of the initial state distribution.
    covariance : ndarray
        Covariance matrix of the initial state distribution.
    track_id : int
        A unique track identifier.
    n_init : int
        Number of consecutive detections before the track is confirmed. The
        track state is set to `Deleted` if a miss occurs within the first
        `n_init` frames.
    max_age : int
        The maximum number of consecutive misses before the track state is
        set to `Deleted`.
    feature : Optional[ndarray]
        Feature vector of the detection this track originates from. If not None,
        this feature is added to the `features` cache.

    Attributes
    ----------
    mean : ndarray
        Mean vector of the initial state distribution.
    covariance : ndarray
        Covariance matrix of the initial state distribution.
    track_id : int
        A unique track identifier.
    hits : int
        Total number of measurement updates.
    age : int
        Total number of frames since first occurance.
    time_since_update : int
        Total number of frames since last measurement update.
    state : TrackState
        The current track state.
    features : List[ndarray]
        A cache of features. On each measurement update, the associated feature
        vector is added to this list.

    """
```

ΣΥΝΑΡΤΗΣΗ __init__ ΠΟΥ ΑΦΟΡΑ ΤΗΝ ΑΡΧΙΚΟΠΟΙΗΣΗ ΠΑΡΑΜΕΤΡΩΝ ΤΩΝ ΑΝΤΙΚΕΙΜΕΝΩΝ

```
def __init__(self, mean, covariance, track_id, n_init, max_age,
             feature=None, class_name=None):
    self.mean = mean
    self.covariance = covariance
    self.track_id = track_id
    self.hits = 1
    self.age = 1
    self.time_since_update = 0

    self.state = TrackState.Tentative
    self.features = []
    if feature is not None:
        self.features.append(feature)

    self._n_init = n_init
    self._max_age = max_age
    self.class_name = class_name

    """ trace list storing the centers of the boxes """
    self.centers = []
```

ΣΥΝΑΡΤΗΣΕΙΣ to_tlwh, to_tlbr

```
def to_tlwh(self):
    """Get current position in bounding box format `(top left x, top left y,
    width, height)`.

    Returns
    -------
    ndarray
        The bounding box.

    """
    ret = self.mean[:4].copy()
    ret[2] *= ret[3]
    ret[:2] -= ret[2:] / 2
    return ret

def to_tlbr(self):
    """Get current position in bounding box format `(min x, miny, max x,
    max y)`.

    Returns
    -------
    ndarray
        The bounding box.

    """
    ret = self.to_tlwh()
    ret[2:] = ret[:2] + ret[2:]
    return ret
```

ΣΥΝΑΡΤΗΣΗ get_class ΑΝΑΦΟΡΙΚΑ ΜΕ ΤΗΝ ΚΛΑΣΗ ΣΤΗΝ ΟΠΟΙΑ ΑΝΗΚΕΙ ΤΟ ΚΑΘΕ ΑΝΤΙΚΕΙΜΕΝΟ

```
def get_class(self):
    return self.class_name
```

ΣΥΝΑΡΤΗΣΕΙΣ predict & update ΜΕ ΧΡΗΣΗ ΤΟΥ Kalman Filter

```
def predict(self, kf):
    """Propagate the state distribution to the current time step using a
    Kalman filter prediction step.

    Parameters
    ----------
    kf : kalman_filter.KalmanFilter
        The Kalman filter.

    """
    self.mean, self.covariance = kf.predict(self.mean, self.covariance)
    self.age += 1
    self.time_since_update += 1

def update(self, kf, detection):
    """Perform Kalman filter measurement update step and update the feature
    cache.

    Parameters
    ----------
    kf : kalman_filter.KalmanFilter
        The Kalman filter.
    detection : Detection
        The associated detection.

    """
    self.mean, self.covariance = kf.update(
        self.mean, self.covariance, detection.to_xyah())
    self.features.append(detection.feature)

    self.hits += 1
    self.time_since_update = 0
    if self.state == TrackState.Tentative and self.hits >= self._n_init:
        self.state = TrackState.Confirmed

        # update center
    self.centers.append([int(self.mean[0]), int(self.mean[1])])
    print("Tracker ID: {}, Coords: {}".format(str(self.track_id), (int(self.mean[0]), int(self.mean[1]))))
```

ΣΥΝΑΡΤΗΣΕΙΣ mark_missed, is_tentative, is_confirmed ΓΙΑ ΤΟΝ ΕΛΕΓΧΟ ΤΗΣ ΤΡΟΧΙΑΣ ΤΩΝ ΑΝΤΙΚΕΙΜΕΝΩΝ ΩΣΤΕ ΝΑ ΠΛΗΡΕΙ ΟΡΙΣΜΕΝΕΣ ΠΡΟΥΠΟΘΕΣΕΙΣ

```
def mark_missed(self):
    """Mark this track as missed (no association at the current time step).
    """
    if self.state == TrackState.Tentative:
        self.state = TrackState.Deleted
    elif self.time_since_update > self._max_age:
        self.state = TrackState.Deleted

def is_tentative(self):
    """Returns True if this track is tentative (unconfirmed).
    """
    return self.state == TrackState.Tentative

def is_confirmed(self):
    """Returns True if this track is confirmed."""
    return self.state == TrackState.Confirmed
```

ΣΥΝΑΡΤΗΣΗ is_deleted (ΑΠΟΤΕΛΕΙΤΑΙ ΑΠΟ ΟΛΕΣ ΤΙΣ ΜΕΤΑΤΡΟΠΕΣ ΚΑΙ ΤΙΣ ΔΙΑΔΙΚΑΣΙΕΣ ΠΟΥ ΑΝΑΦΕΡΟΝΤΑΙ ΣΤΗ ΣΥΝΕΧΕΙΑ)

```
def is_deleted(self):
    """Returns True if this track is dead and should be deleted."""
    if self.state == TrackState.Deleted:
```

ΕΛΕΓΧΟΣ ΓΙΑ ΤΟ ΑΝ ΕΝΑ ΑΝΤΙΚΕΙΜΕΝΟ ΕΧΕΙ ΑΝΙΧΝΕΥΤΕΙ ΓΙΑ ΟΡΙΣΜΕΝΟ ΑΡΙΘΜΟ FRAMES

```
        # an exei kanei detect alla oxi track tote aporriptetai
        if len(self.centers) < 50:
            return

        print("Frames"+str(len(self.centers)))
```

ΕΙΣΟΔΟΣ ΣΤΡΟΦΩΝ, ΜΕΓΕΘΟΥΣ PIXEL ΚΑΙ ΥΠΟΛΟΓΙΣΜΟΣ ΤΟΥ ΠΙΝΑΚΑ ΣΤΡΟΦΩΝ

```
pixelSize=0.01026
k=-128.865*3.14/180
f=-41.662*3.14/180
w=-27.159*3.14/180
#---------------------------------------Πρεπει να μετατραπουν σε rad για να μπουν στον πινακα στροφων
R = [[math.cos(f)*math.cos(k), math.cos(w)*math.sin(k)+math.sin(w)*math.sin(f)*math.cos(k), math.sin(w)*math.sin(k)-math.cos(w)*math.sin(f)*math.cos(k)], [-math.cos(f)*math.sin(k), math.cos(w)*math.cos(k)-math.
for i in R:
    print(i)
```

```
math.sin(w)*math.sin(f)*math.sin(k), math.sin(w)*math.cos(k)+math.cos(w)*math.sin(f)*math.sin(k)], [math.sin(f), -math.sin(w)*math.cos(f), math.cos(w)*math.cos(f)]]
```

ΑΡΧΕΙΟ troxia.txt ΜΕ ΤΙΣ ΣΥΝ/ΝΕΣ ΤΩΝ ΚΕΝΤΡΩΝ ΤΩΝ BBOX ΣΕ PIXEL ΚΑΙ ΔΗΜΙΟΥΡΓΙΑ ΠΙΝΑΚΑ

```
# anoigei to arxeio gia na diavasei tis theseis se pixel
pixelFile = open("troxia.txt", "r")
linesPixel = pixelFile.readlines()
pixelFile.close()

# ftiaxnei pinaka me ta stoixeia se pixel pou diavase apo to arxeio parapanw
array_pixel = []
for strin in linesPixel:
    array_pixel.append([float(y) for y in strin.split()]);

print(array_pixel)
```

ΜΕΤΑΤΡΟΠΗ ΤΩΝ PIXEL ΣΕ ΜΜ ΚΑΙ ΜΕΤΑΤΡΟΠΗ ΤΟΥ ΣΗΜΕΙΟΥ ΑΝΑΦΟΡΑΣ ΑΠΟ ΤΗΝ ΠΑΝΩ ΑΡΙΣΤΕΡΗ ΓΩΝΙΑ ΣΤΟ ΚΕΝΤΡΟ ΤΗΣ ΕΙΚΟΝΑΣ

```
    # allagi twn kentrwn sto kentro tis othonis
    for i in range(len(array_pixel)):
        array_pixel[i][0] = (array_pixel[i][0] - 1280/2)*pixelSize
        array_pixel[i][1] = (720/2 - array_pixel[i][1])*pixelSize

    #allagi twn kentrwn sto kentro tis othonis
    for i in range(len(self.centers)):
        self.centers[i][0] = (self.centers[i][0] - 1280/2)*pixelSize
        self.centers[i][1] = (720/2 - self.centers[i][1])*pixelSize

    print("meta tin allagi")
    print(array_pixel)
```

ΟΡΙΣΜΟΣ ΣΤΟΙΧΕΙΩΝ ΕΣΩΤΕΙΚΟΥ & ΕΞΩΤΕΡΙΚΟΥ ΠΡΟΣΑΝΑΤΟΛΙΣΜΟΥ - ΕΚΤΟΣ ΤΩΝ ΣΤΡΟΦΩΝ ΠΟΥ ΟΡΙΣΤΗΚΑΝ ΠΡΟΗΓΟΥΜΕΝΩΣ

```
    # stoixeia eswterikoy & ekswterikou prosanatolismoy (ektos twn strofwn)
    x0=0
    y0=0
    X0=480732.937
    Y0=4202952.394
    Z0=202.397
    Z=187
    c=10.26
```

ΣΥΝΘΗΚΗ ΣΥΓΓΡΑΜΜΙΚΟΤΗΤΑΣ ΓΙΑ ΜΕΤΑΤΡΟΠΗ ΤΩΝ ΣΥΝΤΕΤΑΓΜΕΝΩΝ ΣΕ ΕΓΣΑ 87’

Μετατροπή σε ΕΓΣΑ 87’ των συν/νων, τόσο των σημείων της τροχιάς που απαρτίζουν τα segments αλλά και όλων των υπόλοιπων σημείων της.

```
#metatropi se egsa me Σ.Σ gia array_pixel
array_egsa = []
for i in range(len(array_pixel)):
    kxx=array_pixel[i][0]-x0
    pyy=array_pixel[i][1]-y0

    X = X0 + (Z-Z0)*((kxx)*R[0][0] + (pyy)*R[1][0] - c*R[2][0])/((kxx)*R[0][2] + (pyy)*R[1][2] - c*R[2][2])
    Y = Y0 + (Z - Z0) * ((kxx) * R[0][1] + (pyy) * R[1][1] - c * R[2][1]) / ((kxx) * R[0][2] + (pyy) * R[1][2] - c * R[2][2])

    array_egsa.append( [ X, Y ] )

print(array_egsa)

#metatropi se egsa me Σ.Σ gia olh th troxia (self.centers)
for i in range(len(self.centers)):
    kxx=self.centers[i][0]-x0
    pyy=self.centers[i][1]-y0

    self.centers[i][0] = X0 + (Z-Z0)*((kxx)*R[0][0] + (pyy)*R[1][0] - c*R[2][0])/((kxx)*R[0][2] + (pyy)*R[1][2] - c*R[2][2])
    self.centers[i][1] = Y0 + (Z - Z0) * ((kxx) * R[0][1] + (pyy) * R[1][1] - c * R[2][1]) / ((kxx) * R[0][2] + (pyy) * R[1][2] - c * R[2][2])

print("=========================================ID = " + str(self.track_id)+"==================================================")


speedSum=0.0
points_count = len(array_egsa)
```

ΥΠΟΛΟΓΙΣΜΟΣ ΑΠΟΣΤΑΣΗΣ Dx ΣΕ ΚΑΘΕ SEGMENT

```
    # gia kathe segment
    lastFrame=-1
    p=1
    while p < points_count:

        # Ypolgismos Dx me ta egsa
        x1 = array_egsa[p-1][0]
        y1 = array_egsa[p-1][1]

        x2 = array_egsa[p][0]
        y2 = array_egsa[p][1]

        Dx = (math.sqrt(math.pow(x2 - x1, 2) + math.pow(y2 - y1, 2)))/1000.0

        print("--------------Segment "+str(p))
        print("Dx = " + str(Dx))
```

ΕΥΡΕΣΗ ΤΩΝ FRAMES ΣΤΑ ΟΠΟΙΑ ΑΝΤΙΣΤΟΙΧΕΙ ΤΟ ΤΕΛΟΣ ΤΟΥ ΚΑΘΕ SEGMENT ΓΙΑ ΟΛΑ ΤΑ ΑΜΑΞΙΑ

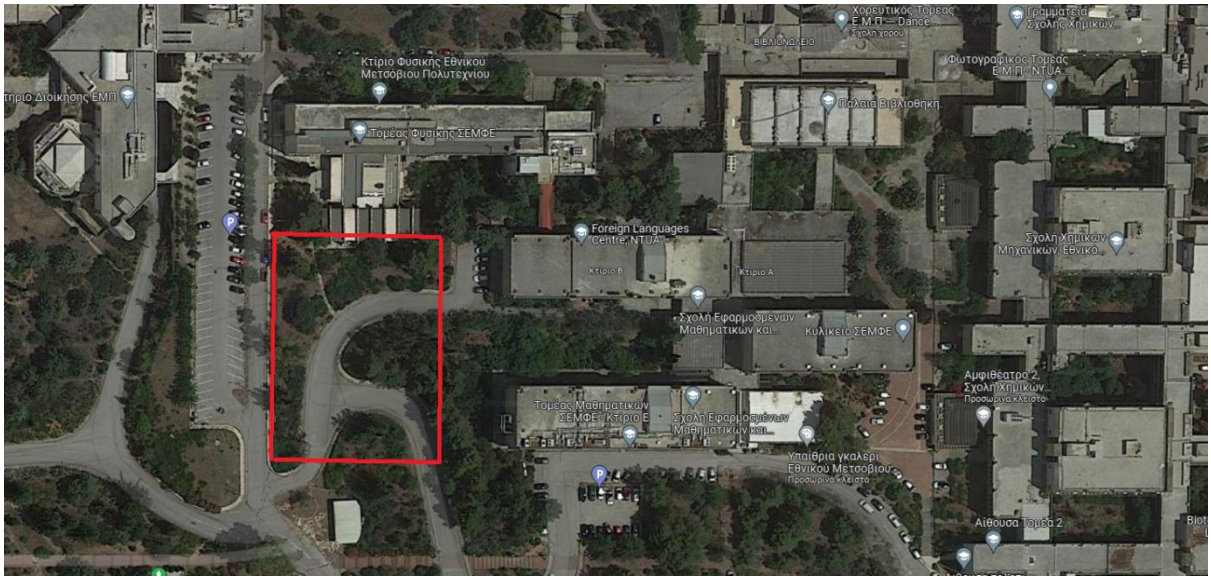
```
# vriskw se poio frame einai to telos tou segment gia ton ypologimo tou xronou
minD=99999999
currentFrame=0
for j in range(len(self.centers)):
    localD=(math.sqrt(math.pow((array_egsa[p][0] - self.centers[j][0]), 2) + math.pow(array_egsa[p][1] - self.centers[j][1], 2)))
    if(localD < minD):
        minD = localD
        currentFrame=j

print("CurrFrame = " + str(currentFrame))
```

ΥΠΟΛΟΓΙΣΜΟΣ ΑΝΤΙΣΤΟΙΧΟΥ ΧΡΟΝΟΥ ΚΑΙ ΤΑΧΥΤΗΤΑΣ ΓΙΑ ΚΑΘΕ SEGMENT ΚΑΙ ΤΕΛΙΚΑ ΤΗΣ ΜΕΣΗΣ ΤΑΧΥΤΗΤΑΣ ΓΙΑ ΚΑΘΕ ΕΝΑ ΑΠΟ ΤΑ ΑΜΑΞΙΑ

```
        if lastFrame==-1 :
            Tsecs = ((currentFrame) / 30.0)
        else:
            Tsecs = ((currentFrame - lastFrame) / 30.0)

        Tf = 1.0/(60.0*60.0)
        Time = Tf * Tsecs   # Xronos se wres
        Speed = Dx / Time
        print("Tsecs = " + str(Tsecs))
        print("Time hours = " + str(Time))
        print("Speed = " + str(Speed))

        speedSum = speedSum + Speed;
        lastFrame=currentFrame
        p=p+1

    speedAVG = speedSum/(points_count-1);
    print("Speed Average: "+str(speedAVG))


return self.state == TrackState.Deleted
```

4.3 Επιλογή οδικού τμήματος

Στο παρόν υποκεφάλαιο παρουσιάζονται όλα τα αποτελέσματα της εφαρμογής, όπως προκύπτουν από την επεξεργασία του βίντεο από το UAV. Ξεκινώντας θα πρέπει να αναφερθεί πως αν και μια τέτοια εφαρμογή λειτουργεί για κάθε είδους οδικό τμήμα, σκοπός μας ήταν η δοκιμή της κυρίως σε καμπύλα οδικά τμήματα όπου η πορεία των οχημάτων παρουσιάζει περισσότερες διαφορές από ότι σε άλλες περιπτώσεις. Προκειμένου η πτήση του UAV να γίνει με ασφάλεια επιλέχθηκε μια καμπύλη εντός της Πολυτεχνειούπολης πλησίον της σχολής ΣΕΜΦΕ, η οποία παρουσιάζεται στην εικόνα που ακολουθεί.



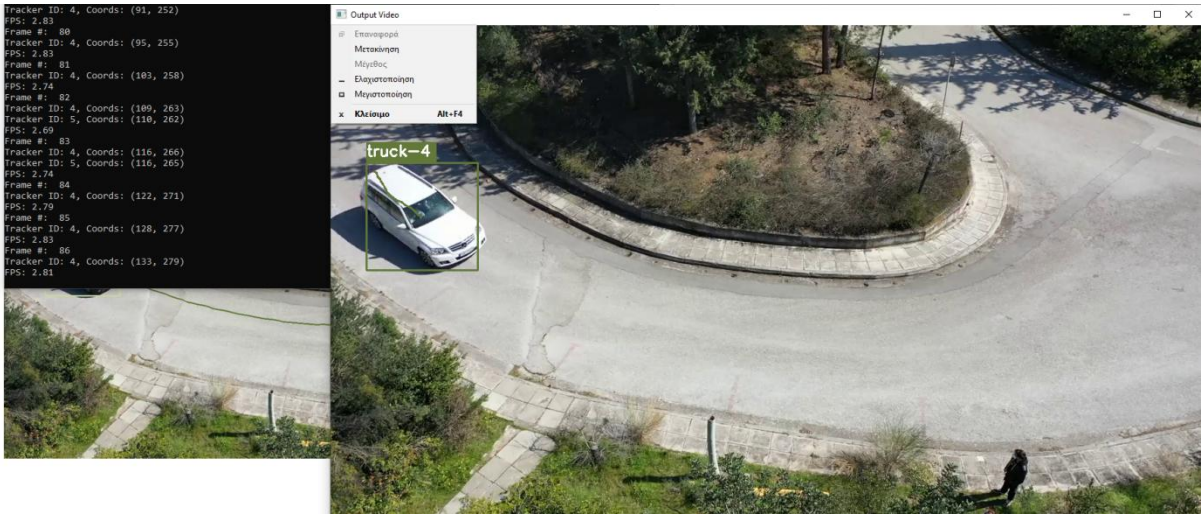
Εικόνα 4.2: Καμπύλο οδικό τμήμα εντός της Πολυτεχνειούπολης Ζωγράφου

Αφού έγινε η επιλογή της συγκεκριμένης καμπύλης, πραγματοποιήθηκε η πτήση και η λήψη του βίντεο προς επεξεργασία. Όπως αναφέρθηκε και σε προηγούμενο κεφάλαιο, η πτήση έπρεπε να γίνει σε χαμηλό σχετικά ύψος της τάξεως των 12-15 μέτρων ανάλογα με το τμήμα δρόμου που θα έπρεπε να είναι ορατό, ενώ η λήψη θα ήταν πλάγια.

Εικόνα 4.3: Περιοχή μελέτης όπως καταγράφηκε από την κάμερα του UAV

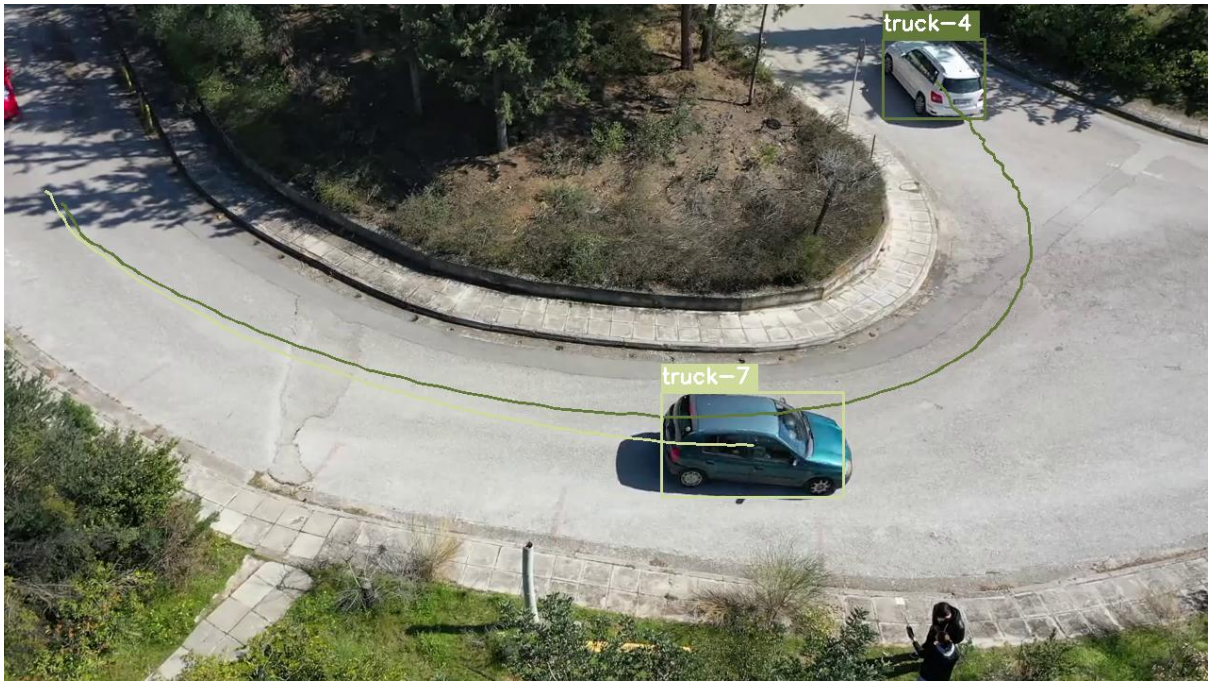
4.4 Αποτελέσματα εφαρμογής

Κατά την επεξεργασία του βίντεο ο χρήστης έχει τη δυνατότητα να βλέπει σε πραγματικό χρόνο τον εντοπισμό και την παρακολούθηση των οχημάτων όπως επίσης και τις συντεταγμένες των κέντρων των BoundingxBoxes που τα περικλείουν. Στο συγκεκριμένο βίντεο τα οχήματα που διέρχονται από την καμπύλη είναι 3 στο σύνολό τους επομένως τα αποτελέσματα που προκύπτουν αφορούν το κάθε ένα από αυτά. Στην εικόνα που ακολουθεί φαίνεται το 1ο αμάξι κατά την είσοδό του στο οδικό τμήμα που καλύπτει το UAV μαζί με τις πληροφορίες που μας δίνει ο αλγόριθμος ενώ γίνεται η επεξεργασία. Όπως γίνεται αντιληπτό το όχημα έχει συγκεκριμένο ID το οποίο διατηρεί καθ΄όλη τη διάρκεια του βίντεο, ενώ σε κάθε frame δίνονται και οι συντεταγμένες του σε pixel στο σύστημα αναφοράς της εικόνας.

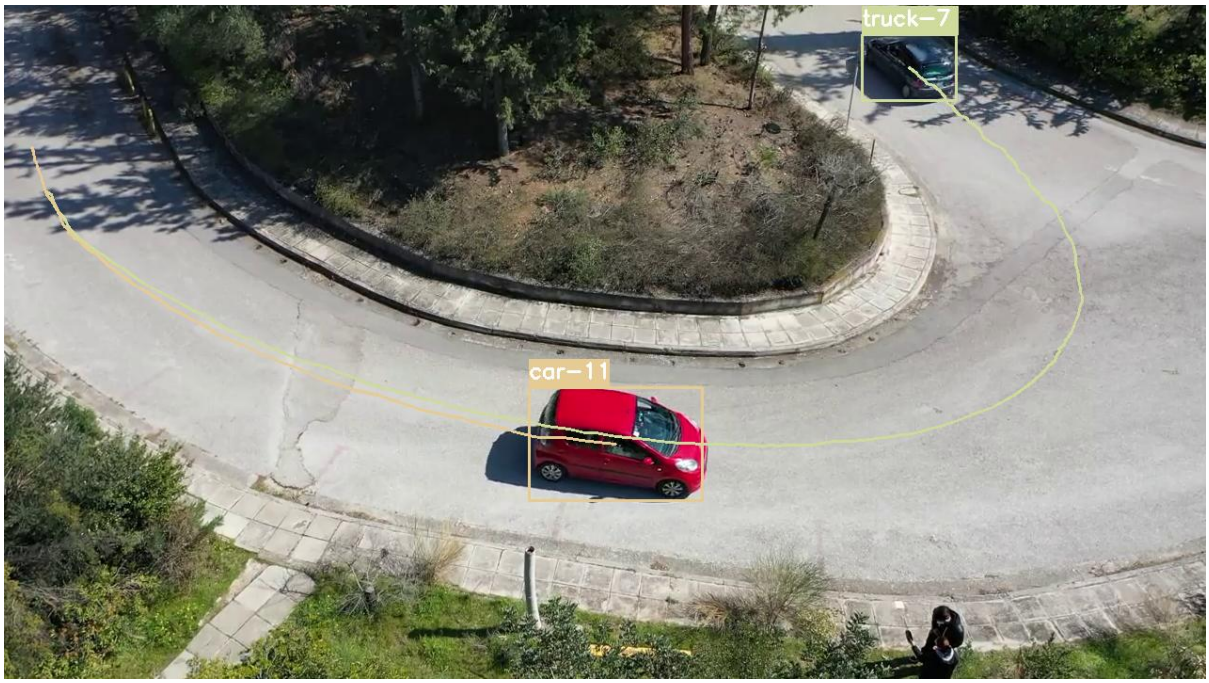


Εικόνα 4.4: Στιγμιότυπο από την παρακολούθηση του 1ου οχήματος

Εκτός από τις συντεταγμένες των οχημάτων κατά την επεξεργασία του βίντεο μας δίνεται και το ίχνος της τροχιάς τους σε ξεχωριστό χρώμα προκειμένου να είναι εύκολα διακριτές οι διαφορές στην πορεία που ακολουθεί το κάθε αμάξι. Από τη στιγμή λοιπόν που θα γίνει η αναγνώριση και παρακολούθηση για ένα όχημα μέχρι και την στιγμή που εκείνο θα βγει εκτός καμπύλης, μπορεί ο χρήστης να βλέπει επακριβώς την πορεία του κατά την κίνησή του στο οδικό τμήμα. Στις επόμενες εικόνες παρουσιάζονται οι 3 τροχιές - μία για κάθε αμάξι- σε διαφορετικό χρώμα.



Εικόνα 4.5: Ίχνος τροχιάς για το 1ο αμάξι (truck-4)



Εικόνα 4.6: Ίχνος τροχιάς για το 2ο αμάξι (truck-7)

Εικόνα 4.7: Ίχνος τροχιάς για το 3ο αμάξι (car-11)

Μόλις ένα όχημα εξέλθει από το βίντεο ο αλγόριθμος μας εμφανίζει στην γραμμή εντολών τις συντεταγμένες των 6 σημείων της τροχιάς που απαρτίζουν τα 5 segments, προκειμένου να γίνει η εκτίμηση της ταχύτητας σύμφωνα με την μέθοδο που αναφέρθηκε στο εδάφιο 3.2 και 3.3. Όπως μπορεί να διαπιστώσει κανείς και από την **Εικόνα 4.8** που αφορά το 1ο αμάξι (ID = 4), αναφέρονται σε πίνακα οι συντεταγμένες αυτές, έχοντας ως αρχή το επάνω αριστερά σημείο της εικόνας (σε pixel). Στη συνέχεια στον από κάτω πίνακα παρουσιάζονται αυτή τη φορά έχοντας ως αρχή το κέντρο της εικόνας, ενώ ακολουθεί και η μετατροπή τους στην τελική τους μορφή σε ΕΓΣΑ 87’.

Εικόνα 4.8: Πίνακες με συντεταγμένες και υπολογισμός ταχύτητας για το όχημα με ID = 4.

Στη συνέχεια ακριβώς κάτω από την υπόδειξη του ID παρουσιάζονται για κάθε ένα από τα 5 segments τα εξής στοιχεία:

→ **Dx** : απόσταση μεταξύ του πρώτου και του τελευταίου σημείου του segment

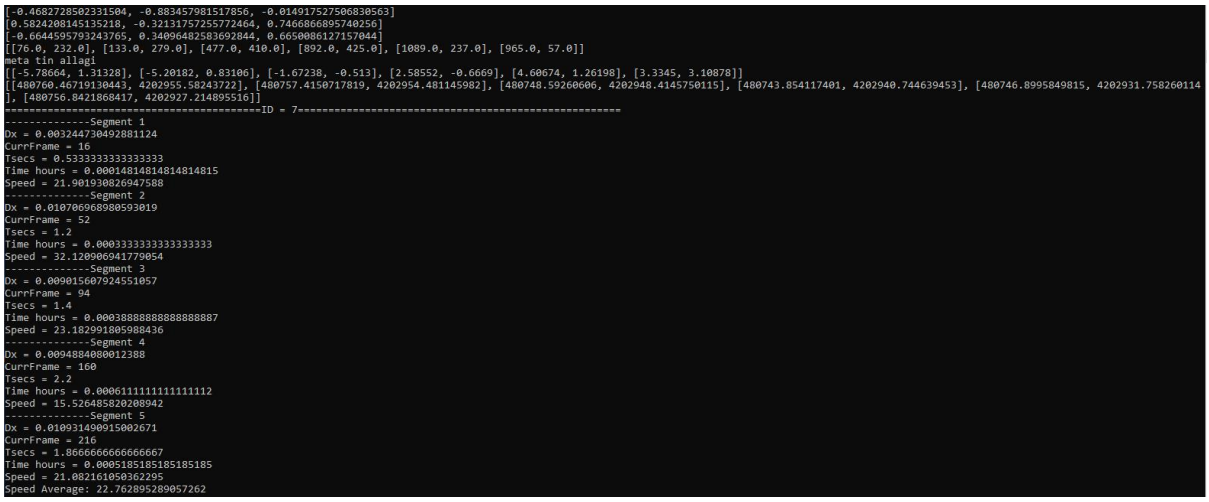
→ **CurrFrame** : τα frames που μεσολαβούν ώστε το όχημα να μετακινηθεί από το πρώτο στο τελευταίο σημείο του segment

→ **Tsecs** : τα παραπάνω frames εκφρασμένα σε χρόνο (seconds)

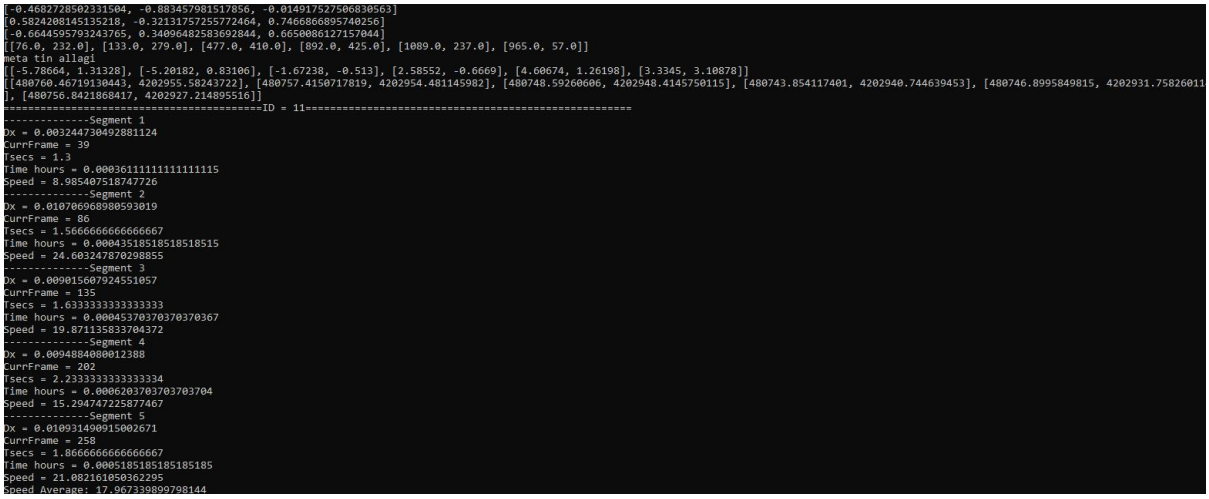
→ **Time hours** : μετατροπή του χρόνου Tsecs σε hours

→ **Speed** : ταχύτητα οχήματος για το συγκεκριμένο segment

Εφόσον υπολογιστεί με τα παραπάνω στοιχεία το Speed και για τα 5 segments, προκύπτει ο μέσος όρος **Speed Average** για το κάθε αμάξι που είναι και η τελική εκτίμηση της ταχύτητας. Όμοια με το πρώτο αμάξι παρουσιάζονται και τα αποτελέσματα για τα επόμενα δύο:



Εικόνα 4.9: Πίνακες με συντεταγμένες και υπολογισμός ταχύτητας για το όχημα με ID = 7



Εικόνα 4.10: Πίνακες με συντεταγμένες και υπολογισμός ταχύτητας για το όχημα με ID = 11

Συγκρίνοντας τα αποτελέσματα της ταχύτητας **Speed Average** (εκφρασμένη σε Km/h) των τριών οχημάτων διαπιστώνει κανείς ότι το 1ο αμάξι κινείται με τη μεγαλύτερη ταχύτητα περίπου 25 Km/h, ενώ ακολουθεί το 2ο με 23 Km//h και στη συνέχεια το 3ο με τη μικρότερη ταχύτητα 18 Km/h. Οι ταχύτητες που προέκυψαν συγκρίθηκαν με εκείνες του ταχύμετρου των οχημάτων με πολύ καλά αποτελέσματα μιας και στη συγκεκριμένη περίπτωση οι αποκλίσεις ήταν ελάχιστες. Εκτός από τις ταχύτητες όπως προαναφέρθηκε έγινε έλεγχος και των αποστάσεων των segments μέσω του Ελληνικού Κτηματολογίου και των συντεταγμένων σε ΕΓΣΑ 87’.

ΚΕΦΑΛΑΙΟ 5ο

5. Συμπεράσματα και προοπτικές

Σκοπός της παρούσας εργασίας ήταν η ανάπτυξη εφαρμογής αναγνώρισης και παρακολούθησης αντικειμένων από εναέριες λήψεις βίντεο (UAV), με τη βοήθεια των state-of-the-art αλγορίθμων YOLO και DeepSort. Αφού συγκεντρώθηκαν τα απαραίτητα εργαλεία και αρχεία για την υλοποίησή της περιγράφηκαν όλες οι μέθοδοι που ακολουθήθηκαν προκειμένου να προσαρμοστούν σε έναν ενιαίο αλγόριθμο που θα εκτελεί σε πραγματικό χρόνο την αναγνώριση, παρακολούθηση καθώς και την εκτίμηση της ταχύτητας οχημάτων που διέρχονται από ένα οδικό τμήμα. Τα αποτελέσματα της εφαρμογής ήταν άκρως ικανοποιητικά, αφού αναγνωρίσθηκαν και παρακολουθήθηκαν επιτυχώς όλα τα οχήματα σε πλήθος δοκιμαστικών βίντεο που χρησιμοποιήθηκαν κατά την υλοποίηση αλλά και σε εκείνο που προέκυψε από την τελική πτήση.

Εκτός από την παρακολούθηση η εφαρμογή εκτελεί και ορισμένες επιπλέον λειτουργίες όπως η εμφάνιση του ίχνους της τροχιάς τον οχημάτων και η εκτίμηση της ταχύτητάς τους με πολύ ενθαρρυντικά αποτελέσματα, όπως είδαμε αναλυτικά στο προηγούμενο κεφάλαιο. Με αυτόν το τρόπο ο χρήστης της εφαρμογής μπορεί να εξάγει πιο σφαιρικά συμπεράσματα για την πορεία των οχημάτων σε ένα τμήμα οδικού άξονα, καθιστώντας την εφαρμογή ένα πολύ χρήσιμο εργαλείο για την έρευνά του. Με τη χρήση του κατάλληλου εξοπλισμού δηλαδή UAV, λογισμικών και της εν λόγο εφαρμογής μπορεί κάποιος με ικανοποιητική ακρίβεια να προσδιορίσει την τροχιά οχημάτων σε πλήθος αστικών ανισόπεδων κόμβων με μια λωρίδα ανά κατεύθυνση, που θεωρούνται ως οι πλέον κρίσιμες περιπτώσεις στη συμπεριφορά των οδηγών, καθώς καλούνται να ακολουθήσουν τη τροχιά του γεωμετρικού σχεδιασμού. Ως εκ τούτου τομείς όπως η οδική ασφάλεια, η πρόληψη ατυχημάτων μέσω του ελέγχου ταχύτητας, η παρακολούθηση και επιτήρηση οδικών αξόνων και η ρύθμιση της κυκλοφορίας, μπορούν να επωφεληθούν από αυτήν.

Παράλληλα, η συγκεκριμένη εφαρμογή μπορεί να επεκτείνει την αναγνώριση και παρακολούθηση και σε άλλα αντικείμενα πέραν των οχημάτων ακόμα και σε ανθρώπους ή ζώα καθώς και πολλές άλλες κατηγορίες (με τις κατάλληλες προσθήκες και αλλαγές). Το γεγονός αυτό την καθιστά ιδιαίτερα χρήσιμη σε πλήθος δραστηριοτήτων.

Ωστόσο είναι σημαντικό να αναφερθούν και εκείνα τα σημεία που χρήζουν βελτίωσης προκειμένου να διευρυνθεί το δυναμικό εύρος της εφαρμογής. Η επανεκπαίδευση του μοντέλου της YOLO για την αναγνώριση αντικειμένων από μεγαλύτερο ύψος, θα αποτελούσε ίσως το πιο ουσιαστικό βήμα προς αυτή την κατεύθυνση. Όπως αναφέρθηκε στο κεφάλαιο 1 η YOLO έχει εκπαιδευτεί με επίγειες εικόνες, γεγονός που κάνει την αναγνώριση πάνω από κάποιο ύψος αδύνατη. Επομένως η εκπαίδευση του μοντέλου με νέα δεδομένα εκπαίδευσης κατάλληλα για εναέριες λήψεις από μεγάλο ύψος, θα έδινε τη δυνατότητα ελέγχου και παρακολούθησης μεγαλύτερης έκτασης οδικών αξόνων. Ένα ακόμη βασικό μειονέκτημα του συστήματος είναι η μη σωστή λειτουργία του όταν οι συνθήκες φωτισμού δεν είναι καλές, μιας και το αποτέλεσμα της παρακολούθησης μπορεί να υποβαθμιστεί. Για παράδειγμα όταν γίνεται λήψη των βίντεο σε σκοτεινό περιβάλλον ή όταν επικρατούν ακραία καιρικά φαινόμενα με έντονες για παράδειγμα βροχοπτώσεις, τα αντικείμενα μπορεί να είναι δύσκολο να εντοπιστούν με ακρίβεια. Ως εκ τούτου, με την κατάλληλη εκπαίδευση του μοντέλου θα μπορούσε η εφαρμογή να χρησιμοποιηθεί επιτυχώς και σε ανάλογες

περιπτώσεις όπως είναι οι νυχτερινές λήψεις. Θα μπορούσε για παράδειγμα να φανεί χρήσιμη σε επαρχιακούς αλλά και αστικούς δρόμους με ανεπαρκή φωτισμό, προκειμένου να επιτηρείται η διέλευση των οχημάτων για τυχόν υπέρβαση του ορίου ταχύτητας ή εκτροπή της πορείας, εξαιτίας πλήθους παραγόντων όπως η διέλευση άγριων ζώων, οι υλικές ζημιές εντός του οδικού δικτύου και η παράνομη στάθμευση εντός αυτού.

Αξίζει επίσης να αναφερθεί πως μια τέτοια εφαρμογή συνιστά, συν τις άλλοις, ένα χαμηλού κόστους σύστημα ανίχνευσης και εκτίμησης ταχύτητας οχημάτων, όπου οι μέθοδοι επεξεργασίας εικόνας/βίντεο και ένας Η.Υ αρκούν για τη δημιουργία του.

Τέλος, όλα όσα αναφέρθηκαν στην παρούσα εργασία αποτελούν ένα μικρό μόνο δείγμα των δυνατοτήτων της όρασης υπολογιστών και της τεχνητής νοημοσύνης, τομείς που ολοένα και περισσότερο φαίνεται να απασχολούν την παγκόσμια ερευνητική κοινότητα και τον σύγχρονο άνθρωπο, βελτιώνοντας σημαντικά την ποιότητα ζωής του.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Mauro Fernandez-Sanjurjo, Brais Bosquet, Manuel Mucientes, Victor M. Brea 2019. *Engineering Applications of Artificial Intelligence* 85 410-420.

[2] Jahongir Azimjonov, Ahmet Ozmen 2021. *A real-time vehicle detection and a novel vehicle tracking systems for estimating and monitoring traffic flow on highways*. Advanced Engineering Informatics 50 101-393.

[3] Miguel Oliveira, Vitor M F Santos 2008. *Automatic Detection of Cars in Real Roads using Haar-like Features*. Department of Mechanical Engineering, University of Aveiro.

[4] Bible Benjdira, Taha Khursheed, Anis Koubaa, Adel Ammar, Kais Ouni 2019. *Car Detection using Unmanned Aerial Vehicles: Comparison between Faster R-CNN and YOLOv3.*

[5] Charles D.Baker, Karyn E.Pollto, Stephanie Pollack 2019. *The Application of Unmanned Aerial Systems In Surface Transportation - Vollume II-C: Evaluation of UAS High-way Speed-Sensing Application.*

[6] Juan Du 2018. Journal of Physics: Conference Series. *Understanding of Object Detection Based on CNN Family and YOLO*.

[7] Ross Girshick 2015. *Fast R-CNN.* Proceedings of IEEE International Conference on Computer Vision (ICCV), pp 1440-1448.

[8] David Held, Sebastian Thurn, Silvio Savarese 2016. *Learning to Track at 100 FPS with Deep Regression Networks.*

[9] Sander Soo 2014. *Object detection using Haar-cascade Classifier*. Institute of Computer Science, University of Tartu.

[10] Alper Yilmaz, Omar Javed, Mubarak Shah 2006. *Object tracking: A survey*.

[11] Reagan L.Galvez, Argel Bandala, Elmer P.Dadios, Ryan Rhay P.Vicerra, Jose Martin Maningo 2018. *Object Detection Using Convolutional Neural Networks.*

[12] Navaneeth Balamuralidhar, Sofia Tilon, Francesco Nex 2021. *MultiEYE: Monitoring System for Real Time Vehicle Detection, Tracking and Speed Estimation from UAV Imagery on Edge-Computing Platforms.* Faculty of Geo-Information Science and Earth Observation (ITC), University of Twente.

[13] Joseph Redmon, Ali Farhadi 2018. *YOLOv3: An Incremental Improvement.* University of Washington.

[14] Alexey Bochkovskiy, Chien-Yao Wang, Hong-Yuan Mark Liao 2020. *YOLOv4: Optimal Speed and Accuracy of Object Detection.* Computer Vision and Pattern Recognition.

[15] Samuel, A. 1959. Some Studies in Machine Learning Using the Game of Checkers. IBM Journal of Research and Development, pp 210-229.

[16] E.Denton, W.Zaremba, J.Bruna, Y.LeCun and R.Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. In NIPS, 2014.

[17] MaCormick, J. and Blake A. 2000. Probalistic exclusion and partitioned sampling for multiple object tracking . Int J.Comput Vision 39.

[18] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", in Advances in neural information processing systems, 2012.

[19] K.Simoyan and A.Zisserman, "Very Deep Convolutional Networks For Large-Scale Image Recognition", in ICLR, 2015.

[20] Jianyuan Guo, Kai Han, Yunhe Wang, Chao Zhang, Zhao-hui Yang, Han Wu, Xinghao Chen, and Chang Xu. Hit-Detector: Hierarchical trinity architecture search for object detection. 2020.

[21] Jiwiing Choi, Dayoung Chun, Hyun Kim, and Hyuk-Jae Lee. Gaussian YOLOv3: An accurate and fast object detector using localization uncertainty for autonomous driving. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), pages 502-511, 2019.

[22] Αραπέλλης Ο. 2021. *Ανάπτυξη διαδικασίας αναγνώρισης και παρακολούθησης αντικειμένων από εναέριες εικόνες (UAV-ΣμηΕΑ) με τη βοήθεια νευρωνικών δικτύων,* ΣΑΤΜ ΕΜΠ.

[23] Κατερίνα Α. 2011. *Γεωμετρία του στερεοζεύγους από βαθμονομημένες και από μη βαθμονομημένες μηχανές*, ΣΑΤΜ ΕΜΠ.

[24] Νικολάου Α. 2020. *ΜΕΘΟΔΟΙ ΚΑΙ ΤΕΧΝΙΚΕΣ ΑΝΙΧΝΕΥΣΗΣ & ΑΝΑΓΝΩΡΙΣΗΣ ΣΗΜΑΤΩΝ ΟΔΙΚΗΣ ΚΥΚΛΟΦΟΡΙΑΣ,* ΠΑΔΑ.

[25] Μαστοράκης Κ. 2016. *Tracking Multiple Vehicles From Satellite Video Using Kalman Filters.*

[26] Παπαδόπουλος Α. 2018. *Πρόβλεψη Τροχιών σε Δεδομένα Κίνησης με Βαθιά Νευρωνικά Δίκτυα*, Πανεπιστήμιο Πειραιώς.

[27] Πέτσα Ε. 2000. *Θεμελιώδεις Έννοιες και Θεμελιώδη Προβλήματα της Φωτογραμμετρίας,* ΠΑΔΑ.

[28] Παπαδόπουλος Σ. 2018. *ΑΠΟΤΑΥΤΟΠΟΙΗΣΗ ΕΙΚΟΝΩΝ ΠΡΟΣΩΠΟΥ ΜΕ ΧΡΗΣΗ ΒΑΘΙΩΝ ΑΥΤΟ-ΚΩΔΙΚΟΠΟΙΗΤΩΝ,* Τμήμα Πληροφορικής ΑΠΘ.

[29] Γρετσίστα Α. 2019. *Ανίχνευση Οχημάτων με Χρήση Υπολογιστικής Όρασης*. ΗΜΜΥ ΕΜΠ.

[30] Σιώρα Ε. 2009. *ΔΙΕΡΕΥΝΗΣΗ ΠΡΑΓΜΑΤΙΚΗΣ ΚΑΜΠΥΛΟΤΗΤΑΣ ΤΡΟΧΙΑΣ ΟΧΗΜΑΤΩΝ ΣΕ ΣΧΕΣΗ ΜΕ ΤΗΝ ΚΑΜΠΥΛΟΤΗΤΑ ΣΧΕΔΙΑΣΜΟΥ ΥΠΕΡΑΣΤΙΚΩΝ ΟΔΩΝ ΔΥΟ ΛΩΡΙΔΩΝ ΚΥΚΛΟΦΟΡΙΑΣ*, ΣΑΤΜ ΕΜΠ.

[31] Παπαδόπουλος Χ. 2015. *Συνδυάζοντας Απλά και Συνελικτικά Νευρωνικά Δίκτυα με SVM*, ΗΜΜΥ ΑΠΘ.

[32] Πατιάς Π. 1991. *Εισαγωγή στη Φωτογραμμετρία.*

[33] Φρατζεσκάκης Ι.Μ. , Γκόλιας Ι.Κ 1994. *ΟΔΙΚΗ ΑΣΦΑΛΕΙΑ*.

Διαδικτυακοί Τόποι

https://learnopencv.com/deep-learning-based-object-detection-using-yolov3-with-opencv-python-c/

https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

https://www.kdnuggets.com/2017/02/yhat-support-vector-machine.html

https://towardsdatascience.com/map-mean-average-precision-might-confuse-you-5956f1bfa9e2

https://www.v7labs.com/blog/object-tracking-guide

https://www.codeproject.com/Articles/865935/Object-Tracking-Kalman-Filter-with-Ease

https://learnopencv.com/understanding-multiple-object-tracking-using-deepsort/#Introduction-to-DeepSORT

https://www.mdpi.com/2076-3417/12/3/1319/htm

https://www.programmersought.com/article/17005126187/