ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Πλατφόρμα υπηρεσιών έξυπνης πόλης για τον εντοπισμό, την ανίχνευση και την καταμέτρηση ατόμων

Δημήτριος Εμμανουήλ, Προπτυχιακός Φοιτητής

Εισηγητές: **Ευάγγελος Κοσμάτος, Εργαστηριακός Συνεργάτης**
Γεώργιος Πρεζεράκος, Καθηγητής

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Πλατφόρμα υπηρεσιών έξυπνης πόλης για τον εντοπισμό, την ανίχνευση και την καταμέτρηση ατόμων

Δημήτριος Εμμανουήλ, Προπτυχιακός Φοιτητής
Α.Μ. 42079

Εισηγητές:

Ευάγγελος Κοσμάτος, Εργαστηριακός Συνεργάτης
Πρεζεράκος Γεώργιος, Καθηγητής

Εξεταστική Επιτροπή:

Ημερομηνία εξέτασης:

Πλατφόρμα ανίχνευσης έξυπνης πόλης για τον εντοπισμό, την ανίχνευση και την καταμέτρηση ατόμων

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος **Εμμανουήλ Δημήτριος**, του **Παναγιώτη**, με αριθμό μητρώου **42079**, φοιτητής του Τμήματος Μηχανικών Η/Υ Συστημάτων Τ.Ε. του Α.Ε.Ι. Πειραιά Τ.Τ. πριν αναλάβω την εκπόνηση της Πτυχιακής Εργασίας μου, δηλώνω ότι ενημερώθηκα για τα παρακάτω:

«Η Πτυχιακή Εργασία (Π.Ε.) αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο του συγγραφέα, όσο και του Ιδρύματος και θα πρέπει να έχει μοναδικό χαρακτήρα και πρωτότυπο περιεχόμενο.

Απαγορεύεται αυστηρά οποιοδήποτε κομμάτι κειμένου της να εμφανίζεται αυτούσιο ή μεταφρασμένο από κάποια άλλη δημοσιευμένη πηγή. Κάθε τέτοια πράξη αποτελεί προϊόν λογοκλοπής και εγείρει θέμα Ηθικής Τάξης για τα πνευματικά δικαιώματα του άλλου συγγραφέα. Αποκλειστικός υπεύθυνος είναι ο συγγραφέας της Π.Ε., ο οποίος φέρει και την ευθύνη των συνεπειών, ποινικών και άλλων, αυτής της πράξης.

Πέραν των όποιων ποινικών ευθυνών του συγγραφέα σε περίπτωση που το Ίδρυμα του έχει απονείμει Πτυχίο, αυτό ανακαλείται με απόφαση της Συνέλευσης του Τμήματος. Η Συνέλευση του Τμήματος με νέα απόφασης της, μετά από αίτηση του ενδιαφερόμενου, του αναθέτει εκ νέου την εκπόνηση της Π.Ε. με άλλο θέμα και διαφορετικό επιβλέποντα καθηγητή. Η εκπόνηση της εν λόγω Π.Ε. πρέπει να ολοκληρωθεί εντός τουλάχιστον ενός ημερολογιακού 6μήνου από την ημερομηνία ανάθεσης της. Κατά τα λοιπά εφαρμόζονται τα προβλεπόμενα στο άρθρο 18, παρ. 5 του ισχύοντος Εσωτερικού Κανονισμού.»

Πλατφόρμα ανίχνευσης έξυπνης πόλης για τον εντοπισμό, την ανίχνευση και την καταμέτρηση ατόμων

ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα πτυχιακή εργασία ολοκληρώθηκε μετά από επίμονες προσπάθειες, μέσα σε μια περίοδο που παράλληλα εργαζόμουνα σαν βοηθός μηχανογράφησης. Η εργασία ασχολείται με ένα πολύ ενδιαφέρον γνωστικό αντικείμενο, όπως αυτό της επεξεργασίας κινούμενης εικόνας και τις πληροφορίες που μπορούν να συλλεχθούν από αυτή με την υπάρχουσα τεχνολογία. Την προσπάθειά και την ιδέα μου αυτή υποστήριξε ο επιβλέπων καθηγητής μου Ευάγγελος Κοσμάτος, τον οποίο θα ήθελα να ευχαριστήσω προσωπικά.

Ακόμα θα ήθελα να ευχαριστήσω την οικογένειά μου που με την οικονομική και ψυχολογική στήριξη που μου πρόσφερε κατάφερα να βγάλω εις πέρας την διπλωματική μου εργασία και κατά συνέπεια την σχολή.

Πλατφόρμα ανίχνευσης έξυπνης πόλης για τον εντοπισμό, την ανίχνευση και την καταμέτρηση ατόμων

ΠΕΡΙΛΗΨΗ

Η παρούσα πτυχιακή εργασία ασχολείται με την ανίχνευση της κίνησης των ανθρώπων σε πραγματικό χρόνο και τις πολύτιμες πληροφορίες που μπορούν να συλλεχθούν μέσα από αυτή την διαδικασία. Η ροή των ανθρώπων σε μία συγκεκριμένη τοποθεσία μπορεί να είναι πολύ χρήσιμη σε αρκετές εφαρμογές, όπως για παράδειγμα, σε μία εφαρμογή για την προστασία ενός χώρου καθώς και για εφαρμογές όπως την διαχείριση της ροής της κίνησης σε ένα χώρο, μια εκτίμηση για το πλήθος των ατόμων που βρίσκονται σε έναν συγκεκριμένο χώρο, ακόμα και για μια εκτίμηση κίνησης-προδιάθεσης καταναλωτών έξω από μια επιχείρηση. Η ανίχνευση, η παρακολούθηση και η καταμέτρηση ατόμων είναι βασική για μια έρευνα αγοράς ή ένα τμήμα ασφαλείας σε ένα εμπορικό κέντρο. Πολλές από αυτές τις μετρήσεις γίνονται ακόμα βασιζόμενες στον ανθρώπινο παράγοντα και σε πολύ προσεγγιστικές μεθόδους που έχουν ως συνέπεια το μη έγκυρο και άμεσο αποτέλεσμα. Για αυτό λοιπόν τον λόγο είναι αναγκαίο να αναπτυχθούν αυτόματα συστήματα και διαδικασίες για την σωστή καταμέτρηση και συλλογή των αποτελεσμάτων. Κατά συνέπεια, η πτυχιακή εργασία θα ασχοληθεί με την βελτίωση του αλγορίθμου ανίχνευσης και καταμέτρησης ατόμων σε μια τοποθεσία καθώς και την εφαρμογή του σε πραγματικές συνθήκες με χρήση πρωτοποριακών τεχνολογιών και χαμηλού κόστους συστημάτων.

ABSTRACT

This dissertation describes and analyzes the methodology and the implementation method of the people counting system which detects and tracks moving people, using a single fixed camera. This system counts the number of moving objects (people) entering a specific security door. Moreover, the detected objects are tracked and get tagged by the proposed tracking algorithm before entering the door. The proposed system uses a Raspberry Pi 3 Model B, operates at an average of 10 frames per second on real-time scenes where it can support up to 6 persons that comes into the view of a vertically mounted camera. In addition, the system calculates and analyzes the information on the Raspberry Pi and then, sends the live data via Wi-Fi toggle on a remote web server for further processing and analysis.

ΕΠΙΣΤΗΜΟΝΙΚΗ ΠΕΡΙΟΧΗ: Computer Vision

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Σύστημα Καταμέτρησης Ατόμων, Όραση Υπολογιστή, Καταμέτρηση σε Πραγματικό Χρόνο, Επεξεργασία Εικόνας, Σύστημα Καταγραφής

KEY WORDS: Python, OpenCV, Object Tracking, Background Image, Single Camera, Tracking Information, Object Labelling, People Counting, Surveillance, Image Processing, Real-Time Counting

Πλατφόρμα ανίχνευσης έξυπνης πόλης για τον εντοπισμό, την ανίχνευση και την καταμέτρηση ατόμων

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ

1.1 Περιγραφή του αντικειμένου της πτυχιακής εργασίας

Σκοπός της παρούσας πτυχιακής εργασίας είναι να αναλύσει μια μέθοδο εφαρμογής για ένα σύστημα ανίχνευσης, παρακολούθησης και καταμέτρησης κινούμενων ατόμων σε έναν χώρο χρησιμοποιώντας μια κάμερα και έναν μικροϋπολογιστή για την επεξεργασία των δεδομένων. Το σύστημα θα έχει την δυνατότητα να καταμετρήσει τον ακριβή αριθμό των ατόμων που βρίσκονται σε ένα κτήριο, μια αίθουσα ή ακόμα και ένα λεωφορείο ανάλογα με την εγκατάσταση. Με αυτό τον τρόπο θα μπορούσαν να υπάρξουν κάποια ενδιαφέροντα αποτελέσματα προς επεξεργασία που θα μπορούν να χρησιμοποιηθούν για σκοπούς έρευνας marketing καθώς και την αποσυμφόρηση ορισμένων χώρων γνωρίζοντας την πληρότητα αυτών πριν καταφθάσουν στον χώρο. Τα δεδομένα που θα συλλέγονται θα έχουν την δυνατότητα να σταλούν σε ελάχιστο χρόνο σε μια βάση δεδομένων όπου εκεί θα μπορούν να αποθηκεύονται ενώ, θα είναι διαθέσιμα για επεξεργασία και ανάλυση. Μια διαδικτυακή πλατφόρμα θα δέχεται τα αποτελέσματα του αλγορίθμου και της επεξεργασίας του και θα τα απεικονίζει με απλές και κατανοητές γραφικές παραστάσεις.

1.2 Περιγραφή προβλήματος

Συγκεκριμένα η ιδέα της εργασίας βασίστηκε στην ανάγκη της βελτίωσης του χρόνου αναμονής σε δημόσιους φορείς και τράπεζες. Εφαρμόζοντας λοιπόν την μέθοδο ανίχνευσης ατόμων υπάρχει η δυνατότητα να μελέτης της πληρότητας ενός πλήθους καταστημάτων μιας τράπεζας με σκοπό να γίνει η σωστή επιλογή για την γρηγορότερη εξυπηρέτηση. Το πρόγραμμα θα έχει την δυνατότητα να ενημερώνει σε πραγματικό χρόνο μέσω μιας εφαρμογής σε κινητό τηλέφωνο, για την συμφόρηση ενός συγκεκριμένου χώρου και έπειτα μέσα από ανάπτυξη νέου αλγορίθμου να προτείνει νέα καταστήματα που ο συνδυασμός πληρότητας και απόστασης θα είναι προτιμότερος. Αυτή η ιδέα θα μπορούσε να υλοποιηθεί σε αρκετές εγκαταστάσεις ανάλογα με τις ανάγκες. Αυτή η εργασία θα ασχοληθεί με την υλοποίηση του

αλγορίθμου της καταμέτρησης σε πραγματικό χρόνο αλλά και την αποστολή των δεδομένων σε μια διαδικτυακή πλατφόρμα η οποία θα είναι διαθέσιμη και οποιαδήποτε στιγμή απεικονίζοντας τα δεδομένα που έχουμε συλλέξει.

1.3 Ιστορική αναδρομή

Υπάρχουν αρκετές μελέτες και υλοποιήσεις σχετικές με την ανίχνευση και καταμέτρηση ατόμων. Οι Segen και Pingali [1] δημιούργησαν ένα σύστημα το οποίο ανίχνευε την ανθρώπινη σιλουέτα και την καταμετρούσε. Το σύστημα τους έτρεχε σε πραγματικό χρόνο αλλά ο αλγόριθμος υλοποίησης τους είχε μεγάλη πολυπλοκότητα, γεγονός που είχε ως αποτέλεσμα να δημιουργεί δυσκολίες σε περιπτώσεις ταυτόχρονης εισόδου ατόμων στο οπτικό πεδίο της κάμερας. Οι Masoud και Papanikolopoulos [2] ανέπτυξαν ένα σύστημα σε πραγματικό χρόνο στο οποίο οι πεζοί απεικονίζονταν ως ορθογώνια κουτιά με δυναμική συμπεριφορά. Το συγκεκριμένο σύστημα είχε προβλέψει τα προβλήματα που μπορούν να δημιουργήσουν τα «κρυφά» αντικείμενα σε μια καταμέτρηση, δηλαδή το πρόβλημα που μπορεί να προκαλέσει στην καταμέτρηση η στιγμιαία εξαφάνιση και εμφάνιση ενός ατόμου στην εικόνα επεξεργασίας. Η επίτευξη αυτή έγινε με την εισαγωγή παραμέτρων στα αντικείμενα παρακολούθησης. Οι Rossi και Bozzoli [3] κατάφεραν να αποφύγουν το πρόβλημα των «κρυφών» αντικειμένων, τοποθετώντας την κάμερα κατακόρυφα στο σύστημα τους, για να παρακολουθήσουν τους ανθρώπους που περνούσαν από έναν διάδρομο. Οι κατευθύνσεις καταμέτρησης ήταν μόνο δυο (επάνω και κάτω πλευρά της εικόνας). Τέλος, ο Terada [4] πρότεινε μια μέτρηση που μετέβαλε την ανθρώπινη περιοχή και την περιοχή καταμέτρησης, χρησιμοποιώντας τα τρισδιάστατα δεδομένα που λαμβάνονται από μια στερεοφωνική κάμερα.

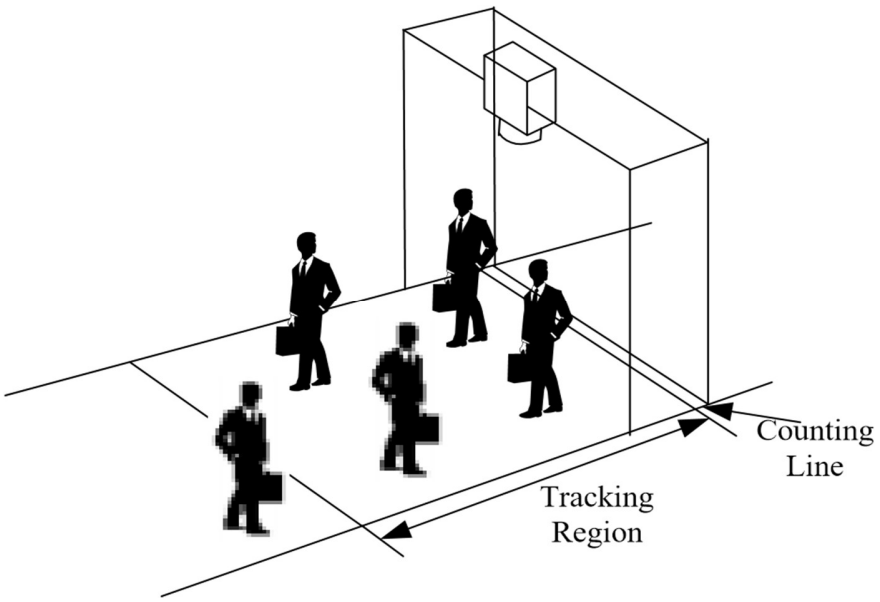
Όλες αυτά τα συστήματα καταμέτρησης συνέβαλλαν στην προσπάθεια δημιουργίας μιας πιο ολοκληρωμένης και τελειοποιημένης εφαρμογής καταμέτρησης, έχοντας απαλείψει τα λάθη και τις ατέλειες των προηγούμενων συστημάτων.

ΚΕΦΑΛΑΙΟ 2

ΠΕΡΙΓΡΑΦΗ ΣΥΣΤΗΜΑΤΟΣ

2.1 Αρχιτεκτονική συστήματος

Σε αυτή την ενότητα θα αναλυθεί η αρχιτεκτονική του συστήματος και θα περιγράφει ο τρόπος υλοποίησης του συστήματος καταμέτρησης. Στην Εικόνα 1 παρατηρούνται οι άνθρωποι οι οποίοι εισέρχονται και εξέρχονται από μια πόρτα ασφαλείας ενός κτηρίου. Θεωρείται ότι μία κάμερα έχει τοποθετηθεί κάθετα στην κάσα της πόρτας με τέτοιο τρόπο που να μπορεί να παρατηρεί και να ανιχνεύει τους ανθρώπους που διασχίζουν την περιοχή κάτω από την πόρτα. Το βίντεο που καταγράφεται από την κάμερα, επεξεργάζεται και τελικά υπολογίζεται ο αριθμός τον ατόμων που διασχίζουν την περιοχή αυτή. Τέλος, τα δεδομένα αυτά στέλνονται μέσω δικτύου σε μια διαδικτυακή πλατφόρμα όπου και απεικονίζεται σε πραγματικό χρόνο.



Εικόνα 1. *Σκηνή διερχόμενων ατόμων από την πόρτα που καταγράφει το σύστημα*

Για να αντιμετωπιστούν κάποια εγγενώς δυναμικά φαινόμενα όπως, η κίνηση των ανθρώπων στην περιοχή ανίχνευσης και στην συνέχεια εκτός πεδίου, την στάση ανθρώπων μέσα στο οπτικό πεδίο που ανιχνεύει η κάμερα, την διέλευση δύο ή

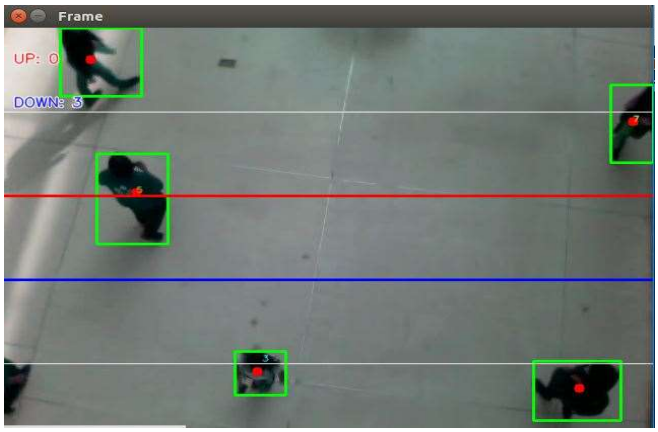
παραπάνω ατόμων που βρίσκονται σε επαφή και τέλος την μαζική διέλευση ατόμων από το οπτικό πεδίο της κάμερας, θα οριστούν κάποιες σταθερές συνθήκες που περιγράφονται παρακάτω για να συνεχιστεί η ανάλυση του συστήματος.

Η κίνηση των ανθρώπων θα γίνεται διαδοχικά από την περιοχή ανίχνευσης και έπειτα από την γραμμή καταμέτρησης. Τα άτομα δεν θα μένουν στάσιμα στην περιοχή ανίχνευσης. Οι άνθρωποι θα μπορούνε να εισέρχονται και να εξέρχονται από την πόρτα ασφαλείας χωρίς να έρχονται σε επαφή και όχι μαζικά. Αυτές οι συνθήκες θα βοηθήσουνε στην σωστή καταμέτρηση ατόμων και στην αποτροπή τυχών λάθους.

Η ανάλυση του συστήματος καταμέτρησης θα μπορούσε να διασπαστεί σε στα τρία ακόλουθα βήματα:

- Έλεγχος για το αν υπάρχουν πιθανά ενδιαφερόμενα αντικείμενα που εισέρχονται στην περιοχή καταγραφής. (Φάση ειδοποίησης)
- Καταγραφή της κίνησης του ενδιαφερόμενου αντικειμένου μέχρι να φτάσει στην γραμμή καταμέτρησης. (Φάση ανίχνευσης κίνησης)
- Καθορισμός και καταμέτρηση ατόμων που διαπερνούν την γραμμή καταμέτρησης. (Φάση καταμέτρησης)

Το σύστημα καταμέτρησης, μας παρέχει ένα γραφικό περιβάλλον χρήστη (GUI) για να ορίσει και να οπτικοποιήσει την περιοχή ειδοποίησης, την περιοχή παρακολούθησης και την γραμμή καταμέτρησης όπως φαίνεται στην Εικόνα 2.



Εικόνα 2. *Γραφικό περιβάλλον χρήστη του συστήματος καταμέτρησης*

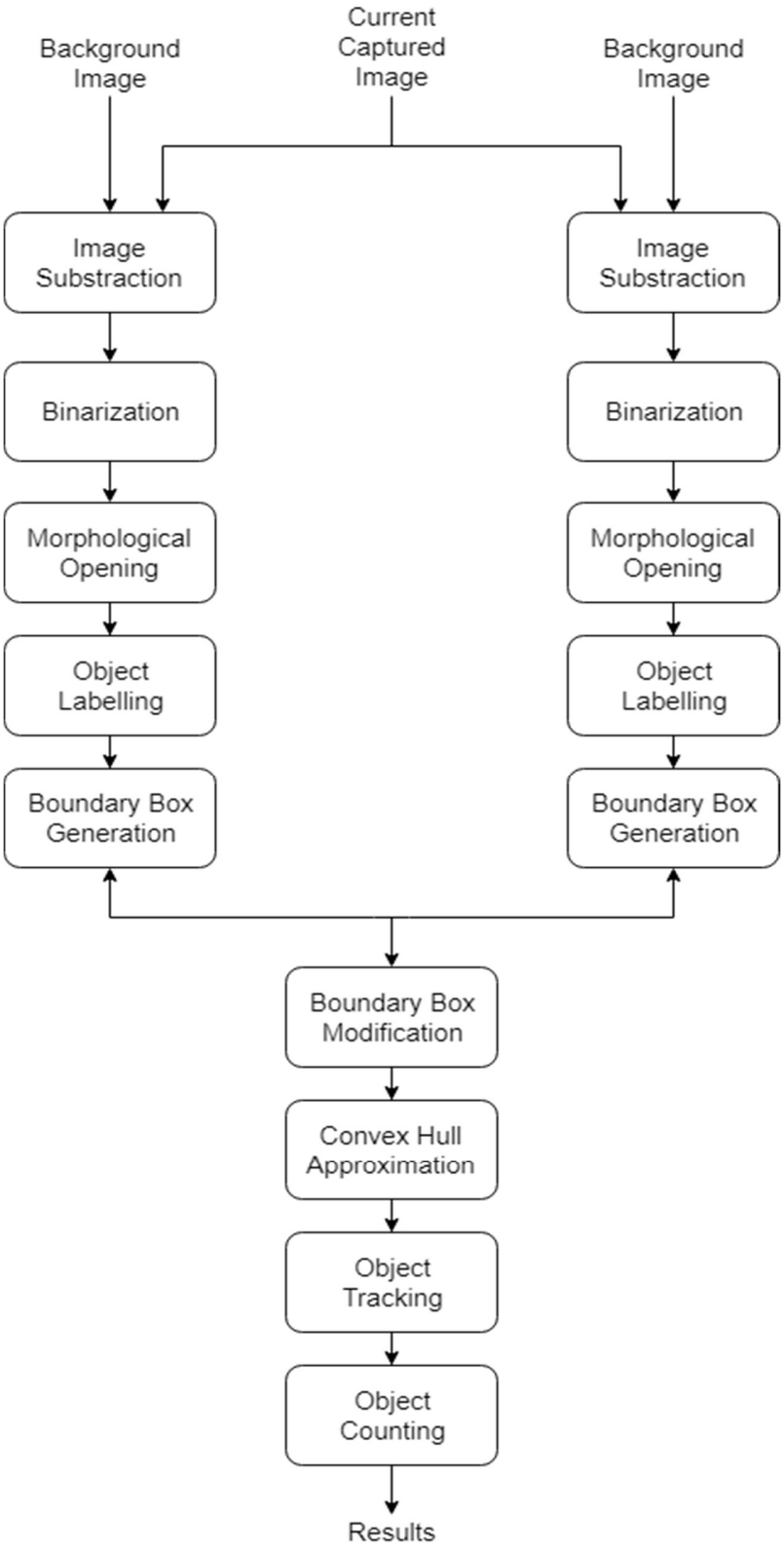
Στην Εικόνα 3 απεικονίζεται το διάγραμμα ροής του αλγορίθμου που έχει δημιουργηθεί.

Αναλυτικά, η αφαίρεση του υποβάθρου (background substation) και ο διαμοιρασμός μιας ψηφιακής εικόνας σε τμήματα pixels (image thresholding), εκτελούνται για την ανάλυση των διαφορών στις εικόνες επεξεργασίας. Η εικόνα με τις διαφορές που έχει παραχθεί, προ-επεξεργάζεται από μια συνάρτηση μορφολογικού ανοίγματος (erosion και στην συνέχεια dilation) για την αφαίρεση μικρών συστάδων στην εικόνα. Στην συνέχεια, κάθε αντικείμενο συμφωνεί με το αντίστοιχο αντικείμενο στην προηγούμενη ληφθείσα εικόνα συγκρίνοντας τις κεντρικές θέσεις τους. Αυτές οι πληροφορίες παρακολούθησης, χρησιμοποιούνται για την καταμέτρηση ατόμων. Όπως φαίνεται στην Εικόνα 2, το προτεινόμενο σύστημα παρουσιάζει τα όρια παρακολούθησης, τις τροχιές παρακολούθησης, τις ετικέτες των αντικειμένων καθώς και την πληροφορία των εισερχομένων και εξερχομένων ατόμων.

Ο προτεινόμενος αλγόριθμος χρησιμοποιεί δυο τύπους για να αναλύσει τις διαφορές μεταξύ των εικόνων και τα καινούργια αντικείμενα που εισέρχονται στην εικόνα. Στην αριστερή ροή του διαγράμματος δημιουργεί μια εικόνα με αναιρεμένο φόντο όπου εμφανίζονται τα κινούμενα αντικείμενα. Ωστόσο, εάν στο μοτίβο της κίνησης, το αντικείμενο είναι παρόμοιο με το φόντο, τότε η κίνηση του αντικειμένου δεν μπορεί να διακριθεί. Η άλλη εικόνα διαφοράς παράγεται αφαιρώντας δυο διαδοχικές εικόνες. Αυτή η εικόνα παρέχει τις πληροφορίες (όρια και ετικέτα) της κίνησης του αντικειμένου ακόμη και αν είναι παρόμοιο με το φόντο. Παρ’ όλα αυτά, αν το αντικείμενο που παρατηρείται παραμείνει στην ίδια θέση, τότε δεν θα υπάρχουν πληροφορίες κίνησης. Επίσης, η ίδια περίπτωση θα υπάρχει όταν το αντικείμενο κινείται με μεγάλη ταχύτητα. Οι πληροφορίες για τα όρια της καταμέτρησης θα είναι λανθασμένες και η εικόνα του περιγράμματος θα είναι θολή.

Υπολογισμός φόντου

Η μέθοδος υπολογισμού του υποβάθρου της εικόνας επηρεάζει την συνολική απόδοση του συστήματος. Δεδομένου ότι οι συνθήκες φωτισμού επηρεάζονται και αλλάζουν με την πάροδο του χρόνου, η εικόνα του φόντου ενημερώνεται με πολύ αργό ρυθμό χρησιμοποιώντας μια συνάρτηση που ανιχνεύει τις μικρές αλλαγές στο υπόβαθρο. Η συνάρτηση αφαίρεσης φόντου χρησιμοποιείται κατά κόρων για την εξαγωγή κινούμενων αντικειμένων από την εικόνα. Για να υπάρχει λοιπόν ένα πιο σωστό υπόβαθρο, πρέπει να εισαχθεί ένας αλγόριθμος εκτίμησης φόντου που έχει την δυνατότητα να προσαρμόζεται στις αλλαγές του φωτισμού και να ανταποκρίνεται στις αλλαγές όπως φαίνεται στο παρακάτω διάγραμμα ροής.



Εικόνα 3. *Διάγραμμα ροής των διαδικασιών του συστήματος καταμέτρησης*

Η διαδικασία του συστήματος ξεκινάει καθορίζοντας εάν τα κινούμενα αντικείμενα υπάρχουν στην τρέχουσα εικόνα (current captured image), συγκρίνοντας την εικόνα αυτή με την προηγούμενη ληφθείσα εικόνα. Εάν δεν υπάρχει κίνηση αντικειμένου, δημιουργείται μια νέα εικόνα φόντου (background image) με βάση των μέσω όρο των τριών εικόνων, την προηγούμενη εικόνα φόντου, την τρέχουσα εικόνα φόντου και την τρέχουσα ληφθείσα εικόνα.

Αυτή η μέθοδος εκτίμησης και υπολογισμού του υποβάθρου αντιμετωπίζει καλά την σταδιακή αλλαγή του φωτισμού, αλλά δεν μπορεί να αντιμετωπίσει την απότομη και μεγάλη αλλαγή του φωτισμού σε ολόκληρη την εικόνα. Για να ξεπεραστεί αυτό το πρόβλημα, πρέπει να οριστεί το μέσο επίπεδο έντασης του φωτισμού. Η τιμή της μέσης έντασης ορίζεται ως η διαφορά της έντασης του φωτισμού μεταξύ της προηγούμενης εικόνας φόντου και της τρέχουσας ληφθείσας εικόνας. Αυτό το επίπεδο έντασης φωτισμού που υπολογίστηκε εφαρμόζεται στην τρέχουσα εικόνα φόντου για να αντισταθμιστεί η απότομη αλλαγή της έντασης, εάν υπάρχουν κινούμενα αντικείμενα στην τρέχουσα εικόνα.

Εξαγωγή αντικειμένου

Μετά την διαδικασία της ετικετοποίησης των αντικειμένων κάθε αντικείμενο προσδιορίζεται και καθορίζεται για το αν πρόκειται για κύριο σώμα ή κάποιο μέρος του σώματος. Το τμήμα του σώματος συγχωνεύεται στο πλησιέστερο κύριο σώμα. Στη συνέχεια, δημιουργείται ένα ορθογώνιο πλαίσιο οριοθέτησης των συγχωνευμένων αντικειμένων. Στο παρακάτω Εικόνα 4 απεικονίζεται το περίβλημα ενός αντικειμένου μιας ασπρόμαυρης εικόνας.

Εικόνα 4. *Περίβλημα ατόμου μιας ασπρόμαυρης εικόνας*

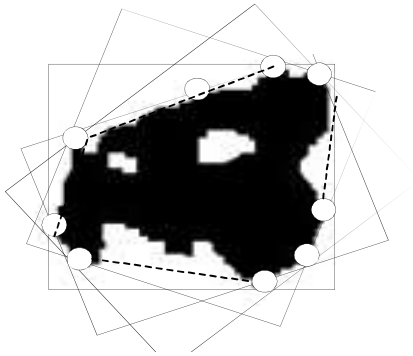
Το πλαίσιο οριοθέτησης χρησιμοποιείται για την παρακολούθηση ατόμων γιατί διάφορα χαρακτηριστικά όπως η περιοχή, το κέντρο, τα σημεία και τα όρια είναι λιγότερο μεταβλητά από τα χαρακτηριστικά της μάσκας μιας εικόνας. Ακόμα και σε περιπτώσεις συσσώρευσης κινούμενων αντικειμένων η χρησιμοποιήσει του πλαισίου βοηθάει στην οριοθέτηση και τον διαχωρισμό αυτών. Σε αυτό να τονιστεί η δυσκολία καταμέτρησης κινούμενων αντικειμένων τα οποία εισέρχονται κολλητά στην εικόνα. Σε τέτοιες περιπτώσεις δημιουργείται ένα κύριο σώμα το οποίο είναι δύσκολο να υπολογιστεί λόγω των διαφορετικών αναλογιών που έχουν οριστεί.

Για να εξαχθούν με ακρίβεια τα χαρακτηριστικά του κυρτού αντικειμένου που βρίσκεται μέσα στο πλαίσιο οριοθέτησης χρειάστηκε να περιστραφεί το ορθογώνιο κατά 15 μοίρες και να βρεθούν κατά προσέγγιση 24 σημεία του αντικειμένου αυτού. Παρακάτω η σχηματική απεικόνιση (Εικόνα 5) του προσεγγιστικού υπολογισμού του αντικειμένου.

Ανίχνευση αντικειμένου

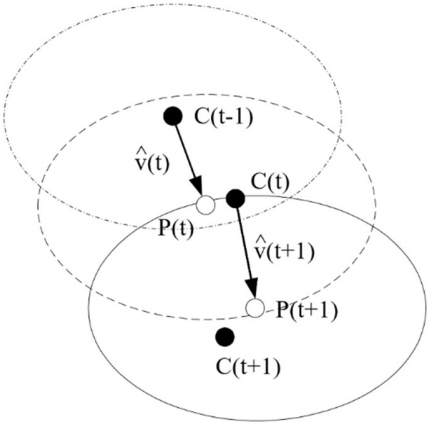
Το κυρτό περίγραμμα κάθε κινούμενο αντικειμένου, αναζητείται στο αντίστοιχο κυρτό περίγραμμα της προηγούμενης ληφθείσας εικόνας συγκρίνοντας τις κεντρικές θέσεις των ορθογώνιων πλαισίων που έχουν σχηματιστεί. Στον αλγόριθμο παρακολούθησης κινούμενων αντικειμένων, το κεντρικό σημείο του περιγράμματος που έχει επαληθευτεί ως άτομο στην προηγούμενη εικόνα επεξεργασίας, η θέση του προβλέπεται στην τρέχουσα εικόνα αναλύοντας τις πληροφορίες ταχύτητας του κινούμενου αντικειμένου του παρελθόντος. Ο τύπος υπολογισμού είναι:

$$v\hat{}\,(t + 1) = (1 - a)*v(t) + a*v\hat{}\,(t), \quad (1)$$



Εικόνα 5. *Προσεγγιστικός υπολογισμός κυρτού περιβλήματος*

Στην Εικόνα 5 το κυρτό περίγραμμα του κινούμενο αντικειμένου, όπου a είναι ο συντελεστής απόσβεσης, v(t) και v(t+1) είναι η μετατόπιση στο χρόνο t και η μετατόπιση στο χρόνο t+1 αντίστοιχα.



Εικόνα 6. *Παρακολούθηση αντικειμένων χρησιμοποιώντας τη μέθοδο πρόβλεψης κίνησης*

Στην Εικόνα 6 αναλύεται η διαδικασία της μεθόδου εντοπισμού αντικειμένων χρησιμοποιώντας την πρόβλεψη της κίνησης που αναλύθηκε παραπάνω. Καταρχήν, η επόμενη θέση του τρέχοντας αντικειμένου στο χρόνο t εκτιμάται μεταβάλλοντας το τρέχον πραγματικό κέντρο C(t) κατά v(t+1). Στη συνέχεια, το σύστημα ορίζει την κυκλική περιοχή αναζήτησης με κέντρο το P(t+1), με καθορισμένη ακτίνα και αναζητά ένα αντικείμενο του οποίου το κεντρικό σημείο βρίσκεται μέσα στην κυκλική περιοχή αναζήτησης. Το C(t+1) αντιπροσωπεύει το κεντρικό σημείο του ανιχνευθέντος αντικειμένου. Οι πληροφορίες παρακολούθησης ενημερώνουν τις πληροφορίες του αντικειμένου που ανιχνεύθηκε με τις πληροφορίες του εντοπισμένου αντικειμένου. Μετά το σημείο C(t+1), η διαδικασία επαναλαμβάνεται.

2.2 Τεχνολογίες υλοποίησης

Raspberry Pi 3 Model B

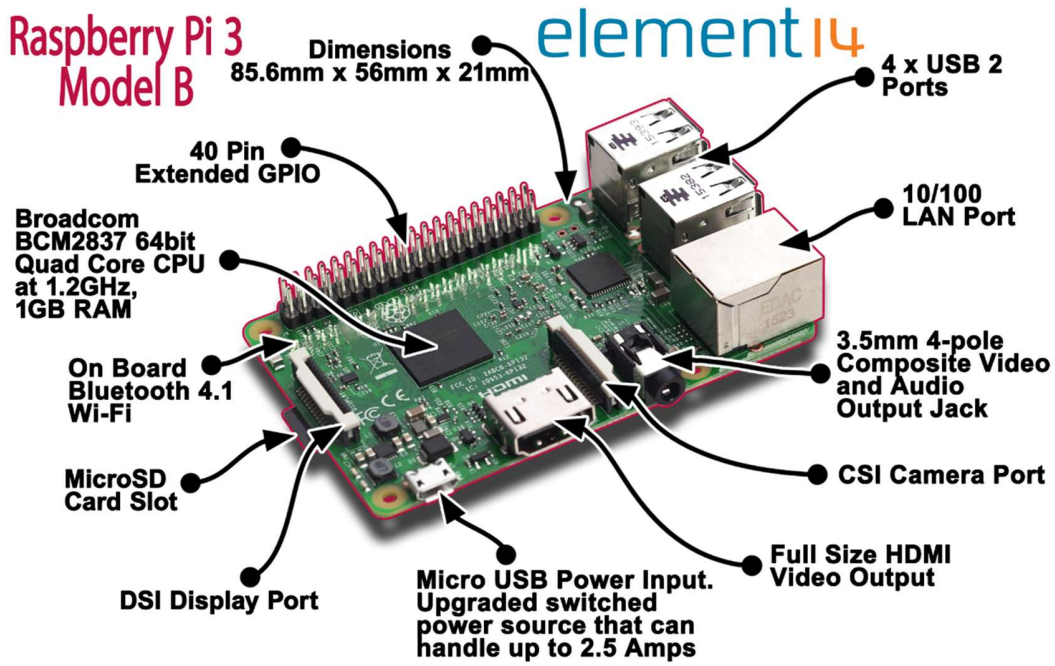
Το Raspberry Pi είναι ένας μικρός υπολογιστής σε μέγεθος μιας πιστωτικής κάρτας. Δημιουργήθηκε στο Ηνωμένο Βασίλειο από το Raspberry Pi Foundation για την ευκολότερη εκμάθηση της επιστήμης των υπολογιστών στα σχολεία.

Η πρώτη γενιά (Raspberry Pi Model B) δόθηκε στο κοινό τον Φεβρουάριο του 2012. Το 2014 κυκλοφόρησε η νέα βελτιωμένη έκδοση του Raspberry Pi 1 η οποία ονομάστηκε RPi 1 Model B+. Τον Απρίλιο του 2014 η εταιρία κυκλοφόρησε τον μικρότερο και οικονομικότερο υπολογιστή τσέπης. Το μοντέλο ονομάστηκε Raspberry Pi Zero και είχε κόστος μόλις 5 $. Φτάνοντας στο τώρα και στο μοντέλο που θα χρησιμοποιηθεί για αυτή την εργασία, τον Φεβρουάριο του 2016, κυκλοφόρησε το Raspberry Pi 3 Model B το οποίο έχει την διπλάσια RAM από το RPi 1 καθώς και ενσωματωμένο WiFi και Bluetooth. Το μέγεθος έχει παραμείνει το ίδιο και η τιμή πώλησης του είναι στα 25 $.

Τα χαρακτηριστικά του RPi 3 Model 3 είναι:

- SoC: Broadcom BCM2837
- CPU: 4xARM Cortex-A53, 1.2GHz
- GPU: Broadcom VideoCore IV
- RAM: 1GB LPDDR2 (900MHz)
- Networking: 10/100 Ethernet, 2.4GHz 802.11n wireless
- Bluetooth: Bluetooth 4.1 Classic, Bluetooth Low Energy
- Storage: microSD
- GPIO: 40-pin header
- Ports: HDMI, 3.5mm analogue audio-video jack, 4xUSB 2.0, Ethernet, Camera Serial Interface (CSI), Display Serial Interface (DSI)

Πλατφόρμα ανίχνευσης έξυπνης πόλης για τον εντοπισμό, την ανίχνευση και την καταμέτρηση ατόμων

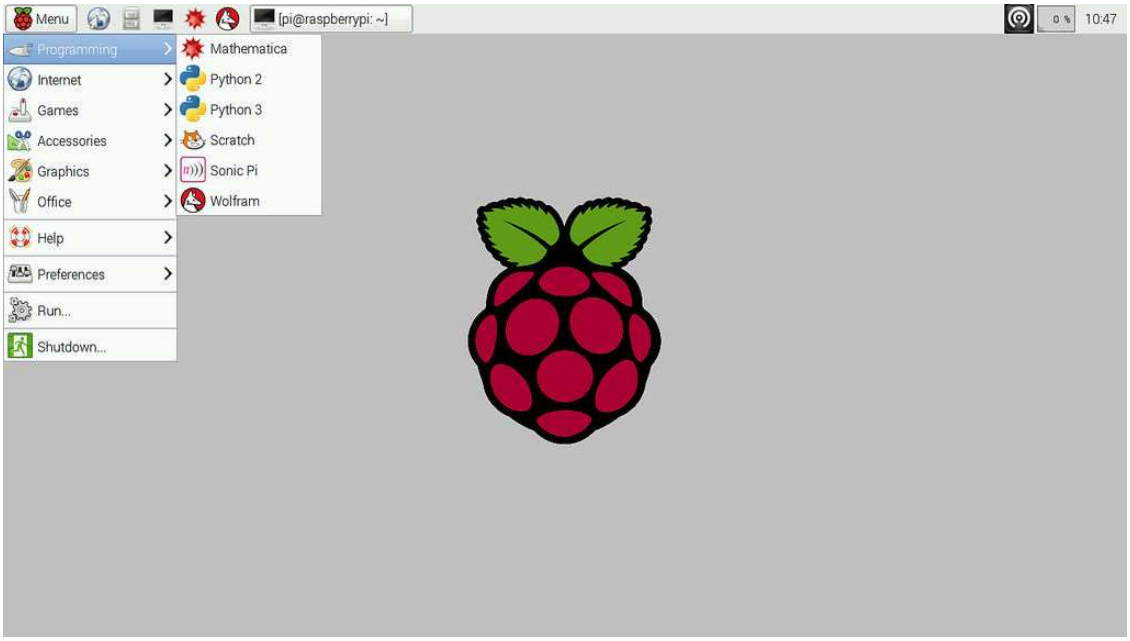


Εικόνα 7. *Χαρακτηριστικά Raspberry Pi 3 Model B*

Λειτουργικό Σύστημα

Το δημοφιλέστερο λειτουργικό σύστημα ονομάζεται Raspbian και φυσικά βασίζεται στο Linux. Επίσης, υπάρχουν και άλλα λειτουργικά συστήματα ειδικά σχεδιασμένα για το RPi όπως Ubuntu, Windows 10 IOT Core, RISC OS και άλλες διάφορες άλλες εκδόσεις που προσομοιάζουν πλήρως ένα media center σύστημα. Οι δυο δημοφιλέστερες εκδόσεις του Raspbian είναι το Raspbian Stretch και το Raspbian Jessie.

Το Raspbian λειτουργικό είναι βασισμένο στο λειτουργικό Debian, μια δημοφιλής διανομή Linux, ελεύθερου λογισμικού που αναπτύσσεται μέσω της συνεργασίας εθελοντών από όλο τον κόσμο. Έχει δημιουργηθεί από τους Mike Thompson και Peter Green. Το Raspbian είναι ένα λειτουργικό σύστημα πλήρως σχεδιασμένο και διαμορφωμένο κατάλληλα για τον επεξεργαστή του RPi. Τέλος, χρησιμοποιεί το PIXEL, Pi Improved Xwindows Environment Lightweight, σαν επιφάνεια εργασίας.



Εικόνα 8. *Επιφάνεια Εργασίας Λειτουργικού Συστήματος Raspbian*

Γλώσσα προγραμματισμού: Python

Η Python είναι μια υψηλού επιπέδου γλώσσα προγραμματισμού η οποία δημιουργήθηκε από τον Ολλανδό Guido van Rossum το 1990. Ο κύριος στόχος της είναι η αναγνωσιμότητα του κώδικα της, η ευκολία χρήσης της και το συντακτικό της, επιτρέπει στους προγραμματιστές να εκφράσουν έννοιες σε λιγότερες γραμμές κώδικα απ' ότι θα ήταν δυνατόν σε γλώσσες όπως η C++ και η Java. Διακρίνεται λόγω τον πολλών βιβλιοθηκών που διευκολύνουν ιδιαίτερα αρκετές συνηθισμένες εργασίες και για την ταχύτητα εκμάθησης. Οι διερμηνείς της Python είναι διαθέσιμοι για εγκατάσταση σε πολλά λειτουργικά συστήματα επιτρέποντας στην Python την εκτέλεση κώδικα σε ευρεία γκάμα λειτουργικών συστημάτων.

Η Python 2.0 κυκλοφόρησε στις 16 Οκτωβρίου του 2000. Στις 3 Δεκεμβρίου 2008 κυκλοφόρησε η έκδοση 3.0. Πολλά από τα καινούργια χαρακτηριστικά αυτής της έκδοσης έχουν μεταφερθεί στις εκδόσεις 2.6 και 2.7 που είναι προς τα πίσω συμβατές. Η Python 3 είναι ιστορικά η πρώτη γλώσσα προγραμματισμού που σπάει την προς τα πίσω συμβατότητα με προηγούμενες εκδόσεις, ώστε να διορθωθούν κάποια λάθη που υπήρχαν σε προγενέστερες εκδόσεις και να καταστεί με σαφήνεια ο απλός τρόπος που μπορούν να γίνουν κάποια πράγματα.

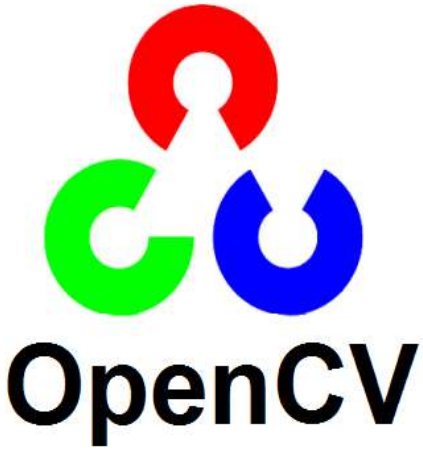
Το πρόγραμμα που έχει δημιουργηθεί στην συγκεκριμένη πτυχιακή εργασία, έχει δημιουργηθεί στην Python έκδοση 2.7. Η υλοποίηση και η λειτουργία του κώδικα απαιτεί την συγκεκριμένη έκδοση για να λειτουργήσει ορθά.

Εικόνα 9. *Λογότυπο της γλώσσας Python*

Πλατφόρμα OpenCV

Η πλατφόρμα OpenCV (Open Source Computer Vision) είναι μια βιβλιοθήκη λειτουργιών προγραμματισμού που σχεδιάστηκε για υπολογιστική αποδοτικότητα και με μεγάλη έμφαση σε εφαρμογές που χρησιμοποιούν την όραση του υπολογιστή σε πραγματικό χρόνο. Η βιβλιοθήκη OpenCV αναπτύχθηκε και βελτιστοποιήθηκε σε γλώσσα C/C++ και μπορεί να επωφεληθεί από την επεξεργασία πολλαπλών πυρήνων. Μαζί με το ενεργοποιημένο OpenCL μπορεί να εκμεταλλευτεί την επιτάχυνση του υλικού και να δημιουργήσει μια δυνατή πλατφόρμα υπολογισμού.

Αρχικά, αναπτύχθηκε από την Intel, στη συνέχεια υποστηρίχθηκε από τον Willow Garage και έπειτα από τον Itseez. Το OpenCV είναι ελεύθερο για ακαδημαϊκή και εμπορική χρήση. Η βιβλιοθήκη του OpenCV στην πραγματικότητα είναι ένα πολύ-πρόγραμμα το οποίο αποτελείται από πολλές υπολογιστικές πλατφόρμες προγραμματισμού. Εμπεριέχει διεπαφές με τις γλώσσες προγραμματισμού C++, Python, Java και υποστηρίζεται από τα λειτουργικά συστήματα Windows, Linux, Mac και Android.



Εικόνα 10. *Λογότυπο της πλατφόρμας OpenCV*

Διαδικτυακή πλατφόρμα Ubidots for Education

Η πλατφόρμα Ubidots ξεκίνησε ως ιδιωτική εταιρία μηχανικών εφαρμογών το 2012, ειδικεύεται σε λύσεις για την σύνδεση υλικού και λογισμικού για την απομακρυσμένη παρακολούθηση, τον έλεγχο και την αυτοματοποίηση διαδικασιών για τους πελάτες υγειονομικής περίθαλψης στην Νοτιοανατολική και την Λατινική Αμερική. Από το 2012 έως το 2014, η Ubidots επέκτεινε τις βιομηχανίες υποστήριξης και πραγματοποίησε αρκετά έργα που συνέδεαν το διαδίκτυο με εταιρίες υγείας, ενέργειας, μεταφορών ακόμα και κατασκευαστικές. Το 2018, το Ubidots καθιέρωσε την πλατφόρμα «Ubidots for Education», για να δώσει στους ενθουσιώδεις και φοιτητές, ένα μέρος να χτίσει, να αναπτύξει, να δοκιμάσει, να μάθει και να διερευνήσει το μέλλον των εφαρμογών και των λύσεων που συνδέονται με το διαδίκτυο.

Το Ubidots προσφέρει μια πλατφόρμα για προγραμματιστές που τους επιτρέπει να καταγράφουν με ευκολία δεδομένα αισθητήρων και να τα μετατρέπουν σε χρήσιμες πληροφορίες. Η πλατφόρμα Ubidots μπορεί να χρησιμοποιηθεί για να σταλούν δεδομένα στο σύννεφο από οποιαδήποτε συσκευή με δυνατότητα πρόσβασης στο διαδίκτυο. Στη συνέχεια, μετά την επεξεργασία των δεδομένων αυτών μπορούν να διαμορφωθούν ενέργειες και ειδοποιήσεις στην μεταβολή τον δεδομένων σε πραγματικό χρόνο. Έτσι τα εργαλεία συλλογής δεδομένων και τα οπτικά εργαλεία αξιοποιούν την πληροφορία που συλλέγουν και δρουν ανάλογα με το αποτέλεσμα. Τέλος, το πρόγραμμα του Ubidots το διάβασμα και την εγγραφεί δεδομένων στους διαθέσιμους πόρους: πηγές δεδομένων, μεταβλητές, τιμές, συμβάντα και πληροφορίες. Το API υποστηρίζει HTTP και HTTPS πρωτόκολλα και απαιτεί ένα κλειδί API για την λειτουργία του.

Εικόνα 11. *Λογότυπο της διαδικτυακής πλατφόρμας Ubidots*

ΚΕΦΑΛΑΙΟ 3

ΥΛΟΠΟΙΗΣΗ ΣΥΣΤΗΜΑΤΟΣ ΚΑΤΑΜΕΤΡΗΣΗΣ

3.1 Εγκατάσταση εξαρτημάτων υλικού

Στην παρούσα υλοποίηση του συστήματος καταμέτρησης χρησιμοποιήθηκαν τα εξής υλικά:

1. Raspberry Pi 3 Model B
2. Raspberry Pi 3 Camera Module
3. Raspberry Pi 3 Camera Case
4. Battery Bank
5. Raspberry Pi 3 Case / Heatsink Aluminum Cooler

Επίσης, χρησιμοποιήθηκαν τα εξής υλικά κατά την εγκατάσταση του λειτουργικού και των προγραμμάτων στο RPi 3.

1. Monitor
2. HDMI cable
3. Ethernet cable
4. Keyboard and Mouse
5. RPi 3 Power Supply

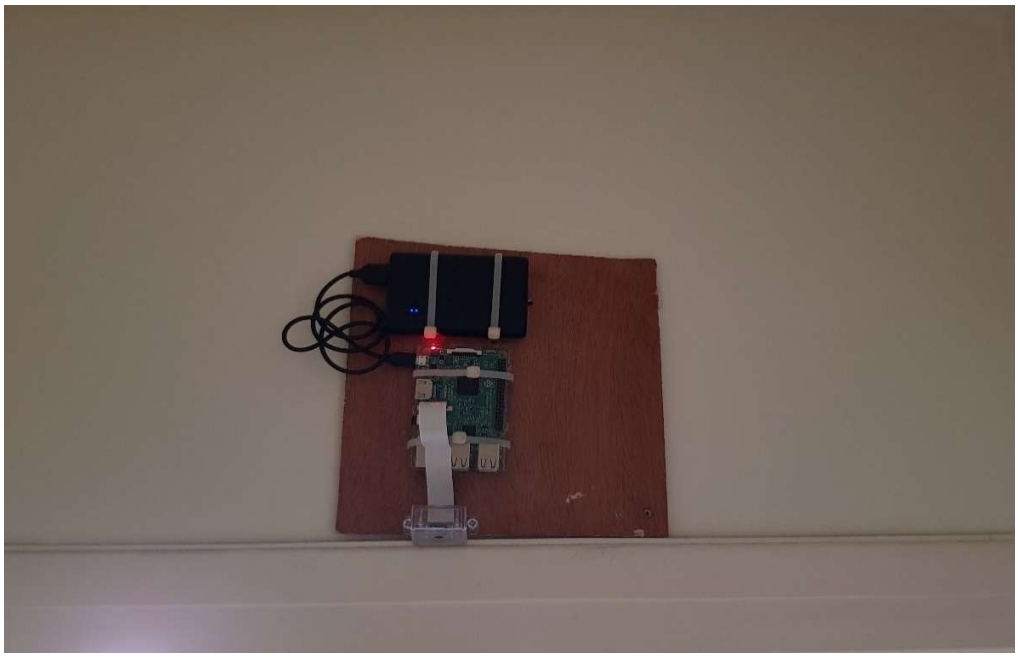
Για την καταμέτρηση των ατόμων σχεδιάστηκε μια ξύλινη πλάκα όπου όλα τα εξαρτήματα που χρησιμοποιήθηκαν ήτανε τοποθετημένα πάνω σε αυτήν δηλαδή, το power bank που θα μας έδινε την απαραίτητη αυτονομία του συστήματος και την αποφυγή καλωδίων για τη παροχή ρεύματος, το RPi 3 μαζί με την κάμερα τοποθετημένα μέσα σε βάσεις και σχεδιασμένα έτσι ώστε η γωνία καταγραφής της κάμερας ως προς το έδαφος να είναι κάθετη.

Εγκαθιστώντας την ξύλινη βάση αυτή πάνω από την κάσα μιας πόρτας όπου καταμετρήθηκε η κίνηση, έγιναν οι δοκιμές για την υλοποίηση της εργασίας. Οι δυνατότητες του RPi 3 όπως για παράδειγμα το WiFi Antenna και το power bank που

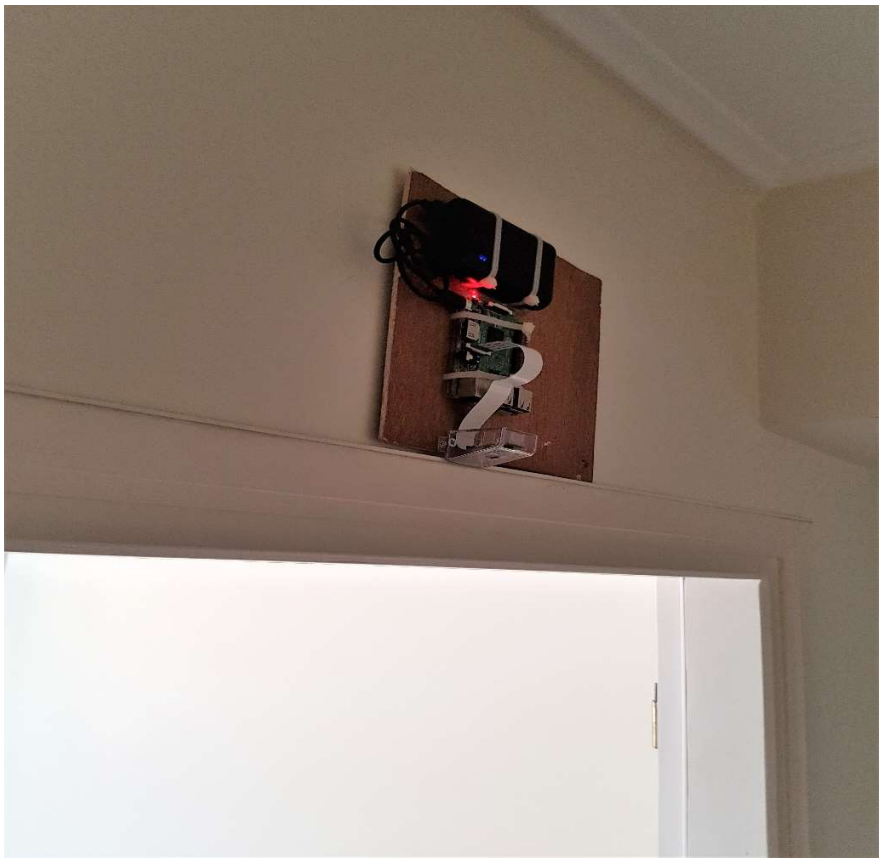
χρησιμοποιήθηκαν, έδωσαν την δυνατότητα να δημιουργηθεί ένα σύστημα αυτόνομο που μπορεί να εγκατασταθεί οπουδήποτε και να δοκιμαστεί η λειτουργία του σε όλες τις φάσεις του κώδικα, ακόμα και στην φάση αποστολής δεδομένων στην διαδικτυακή πλατφόρμα.

Παρακάτω παρατίθενται οι φωτογραφίες της κατασκευής και της εγκατάστασης στην κάσα μιας πόρτας:

Εικόνα 12. *Κατασκευή RPi 3 με camera module και power bank για τροφοδοσία.*



Εικόνα 13. *Εγκατάσταση κατασκευής σε κάσα πόρτας για την δοκιμή της λειτουργίας της.*



Εικόνα 14. *Διαφορετική οπτική γωνία της ίδιας εγκατάστασης.*

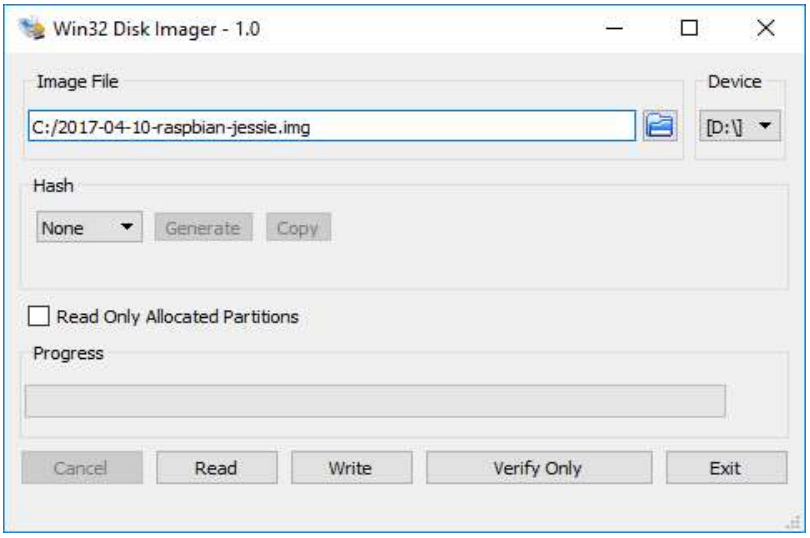
3.2 Εγκατάσταση και παραμετροποίηση του Raspbian λειτουργικού συστήματος

Για την εγκατάσταση του λειτουργικού συστήματος Raspbian, χρειάστηκαν μια microSD κάρτα (ελάχιστη χωρητικότητα 8GB), έναν υπολογιστή με υποδοχή ανάγνωσης κάρτας SD, ένα Raspberry Pi (RPi 3 στην συγκεκριμένη πτυχιακή εργασία) και κάποια βασικά περιφερειακά όπως οθόνη, ποντίκι, πληκτρολόγιο και τροφοδοσία. Η εγκατάσταση του Raspbian είναι εξαιρετικά απλή διαδικασία. Μετά το κατέβασμα του λειτουργικού και την αντιγραφή της εικόνας δίσκου στην κάρτα SD, γίνεται η τοποθέτηση της κάρτας στο RPi3 και ξεκινάει το σύστημα.

- Βήμα 1: Το κατέβασμα της τελευταίας έκδοσης του λειτουργικού Raspbian Jessie with Pixel έγινε από την επίσημη ιστοσελίδα του RPi (https://www.raspberrypi.org/downloads/).

Εικόνα 15. *Διαθέσιμες εκδόσεις λειτουργικού συστήματος Raspbian*

- Βήμα 2: Το Raspbian λειτουργικό είναι σε μορφή αρχείου iso και συμπιεσμένο σε ZIP64. Για την αποσυμπίεση χρειάστηκε το πρόγραμμα 7-zip από λειτουργικό Windows.
- Βήμα 3: Για να γίνει η εγγραφή του αρχείου iso πάνω στην κάρτα SD, μετά την αποσυμπίεση χρειάστηκε το πρόγραμμα Win32 Disk Imager.



Εικόνα 16. *Πρόγραμμα εγγραφής εικονικών αρχείων σε κάρτα SD*

- Βήμα 4: Τέλος, όταν η διαδικασία ολοκληρωθεί η κάρτα SD θα τοποθετηθεί στο RPi και αφού συνδεθεί με όλα τα περιφερειακά, το μπορεί να τεθεί σε λειτουργία. Η συγκεκριμένη έκδοση του Raspbian ανοίγει απευθείας σε γραφικό περιβάλλον. Τα προκαθορισμένα διαπιστευτήρια για την είσοδο στο σύστημα είναι Χρήστης: **pi** και Κωδικός: **raspberry** .

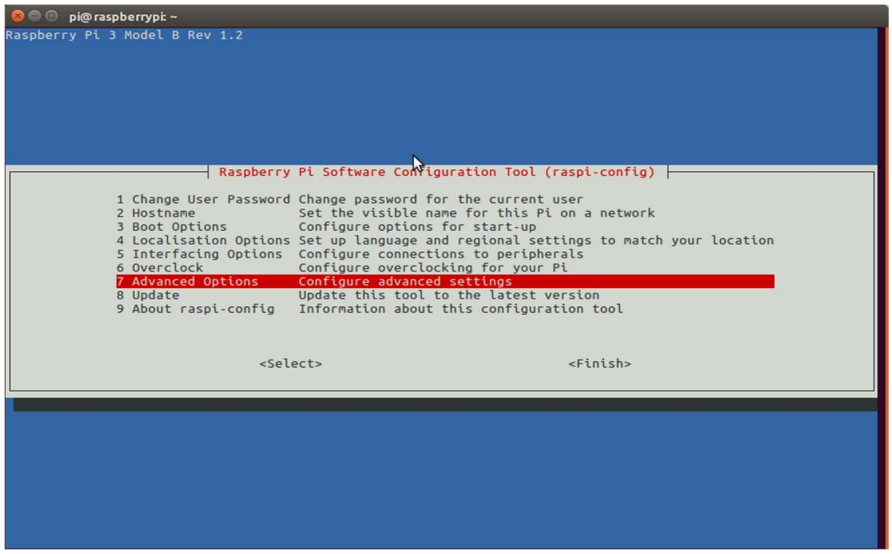
3.3 Διαμόρφωση και βελτιστοποίηση του λειτουργικού συστήματος Raspbian

Για την εγκατάσταση της γλώσσας Python και της πλατφόρμας OpenCV χρειαζόμαστε όλο τον διαθέσιμο χώρο της κάρτας SD καθώς και όλη την επεξεργαστική ισχύ που μπορεί να διαθέσει το RPi για την μειώση του χρόνου εγκατάστασης και μεταγλώττισης.

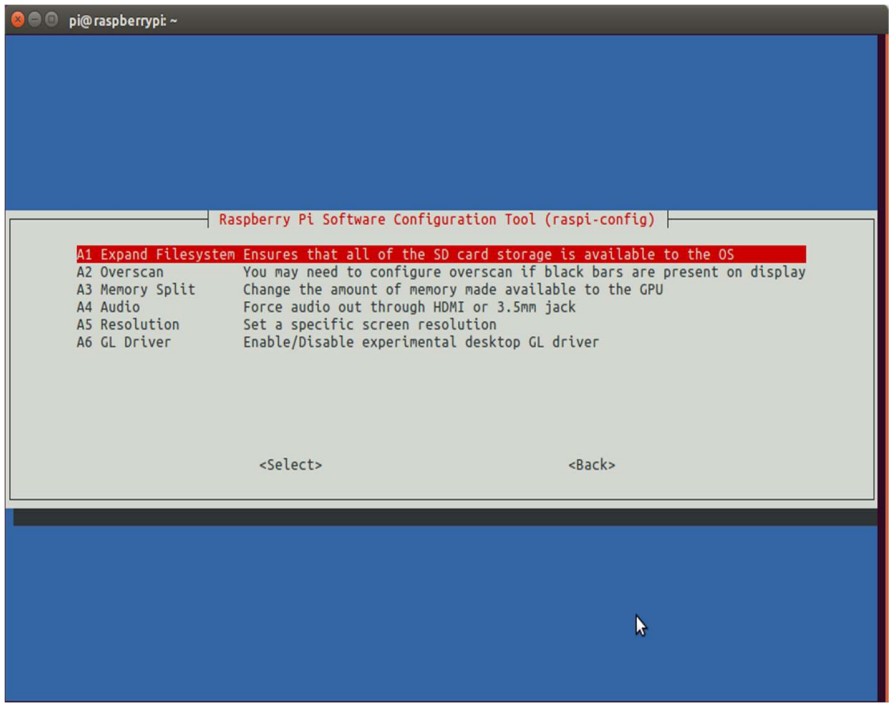
Για να χρησιμοποιηθεί λοιπόν όλο το διαθέσιμο χώρο του συστήματος που βρίσκεται στην SD κάρτα, πρέπει να επιλεχθεί στο μενού διαμόρφωσης του συστήματος η επιλογή για προχωρημένους. Επιλέγοντας “A1 Expand File System” και συνεχίζοντας με επανεκκίνηση συστήματος θα τεθούν σε ισχύ οι αλλαγές. Όταν το σύστημα επανέλθει με την εντολή df -h μπορεί να ελεχθεί το σύστημα αρχείων και πώς είναι διανεμημένος ο ελεύθερος χώρος. Παρακάτω φαίνονται οι εντολές και οι επιλογές που χρησιμοποιήθηκαν:

```
$ sudo raspi-config
```

Πλατφόρμα ανίχνευσης έξυπνης πόλης για τον εντοπισμό, την ανίχνευση και την καταμέτρηση ατόμων



Εικόνα 17. *Επιλογές για προχωρημένους Raspbian*



Εικόνα 18. *Επέκταση ελεύθερου χώρου που είναι διαθέσιμος στα αρχεία του συστήματος*

```
$ sudo reboot
$ df -h
```

```
$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/root        30G  4.2G   24G  15% /
devtmpfs        434M     0  434M   0% /dev
tmpfs           438M     0  438M   0% /dev/shm
tmpfs           438M   12M  427M   3% /run
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           438M     0  438M   0% /sys/fs/cgroup
/dev/mmcblk0p1   42M   21M   21M  51% /boot
tmpfs            88M     0   88M   0% /run/user/1000
```

Εικόνα 19. *Διαθέσιμος ελεύθερος χώρος λειτουργικού συστήματος*

Τέλος, υπάρχει η δυνατότητα να ελευθερωθεί παραπάνω χώρος στο σύστημα αφαιρώντας τις εφαρμογές LibreOffice και Wolfram Engine, βάσει των παρακάτω εντολών.

```
$ sudo apt-get purge wolfram-engine
$ sudo apt-get purge libreoffice*
$ sudo apt-get clean
$ sudo apt-get autoremove
```

Ακόμα, μια βασική αλλαγή στις ρυθμίσεις κατά την μεταγλώττιση είναι αυτή του swap file. Πριν την έναρξη της μεταγλώττισης η αύξηση του μεγέθους του swap file είναι σχεδόν επιτακτική για την ομαλή διαδικασία της εγκατάστασης καθώς και την αποφυγή κολλημάτων λόγω προβλημάτων μνήμης. Αυτή η επιλογή θα ενεργοποιήσει και τους τέσσερις πυρήνες του RPi και θα επιταχύνει κατά πολύ την μεταγλώττιση. Για να ενεργοποιηθεί αυτή η λειτουργία λοιπόν τα βήματα έχουν ως εξής:

1. Άνοιγμα αρχείου /etc/dphys-swapfile και παραμετροποίηση της μεταβλητής CONF_SWAPSIZE=1024. Έπειτα, από την αλλαγή το αρχείο θα έχει αυτή την μορφή.

   ```
   # CONF_SWAPSIZE=100
   CONF_SWAPSIZE=1024
   ```

2. Για να ενεργοποιηθεί η συγκεκριμένη αλλαγή πρέπει να επανακινηθεί το swap service του συστήματος με τις εξής εντολές.

   ```
   $ sudo /etc/init.d/dphys-swapfile stop
   ```

```
$ sudo /etc/init.d/dphys-swapfile start
```

Πρέπει να σημειωθεί ότι η αλλαγή αυτή μπορεί να προκαλέσει σοβαρά προβλήματα στο σύστημα γι' αυτό προτείνετε να επιστρέψει στην προκαθορισμένη τιμή του (100MB) μετά το πέρας της μεταγλώττισης της πλατφόρμας OpenCV.

3.4 Εγκατάσταση γλώσσας Python 2.7 και πλατφόρμας OpenCV 3.0

Η εγκατάσταση της γλώσσας Python και της πλατφόρμας OpenCV στο λειτουργικό Raspbian είναι μια ιδιαίτερα χρονοβόρα διαδικασία. Πρέπει να κατανοηθεί ότι η σειρά των παρακάτω βημάτων και οι εκδόσεις των προγραμμάτων που εγκαθίστανται στο λειτουργικό σύστημα είναι πολύ σημαντική. Η διαφοροποίηση ορισμένων εντολών ή κάποιων εκδόσεων μπορεί να δημιουργήσει πρόβλημα στην εύρυθμη λειτουργία του κώδικα. Έχοντας λοιπόν κατανοήσει τα παραπάνω είναι επιτακτική η εγκατάσταση των απαραίτητων προ απαιτούμενων για την λειτουργία του κώδικα.

- Βήμα 1: Για την αναβάθμιση του συστήματος πρέπει να εκτελεστούν οι παρακάτω εντολές στην κονσόλα.

```
1. $ sudo apt-get update
2. $ sudo apt-get upgrade
```

- Βήμα 2: Εγκατάσταση ορισμένων εργαλείων προγραμματιστή.

```
1. sudo apt-get install build-essential cmake git pkg-config
```

- Βήμα 3: Η πλατφόρμα OpenCV χρειάζεται να φορτώσει αρκετούς τύπους φωτογραφίας από τον σκληρό δίσκο, συμπεριλαμβάνοντας JPEG, PNG, TIFF. Για να γίνει αυτό χρειάζεται τα οπτικά I/O πακέτα.

```
1. $ sudo apt-get install libjpeg8-dev libtiff4-dev libjasper-dev libpng12-dev
```

- Βήμα 4: Για την αναπαράσταση των φωτογραφιών που φορτώνονται από την πλατφόρμα OpenCV, στην οθόνη του υπολογιστή θα χρειαστεί η εγκατάσταση της GTK βιβλιοθήκης προγραμματιστή του OpenCV, από το οποίο καθορίζεται το GUI (Graphical User Interfaces).

  1. $ sudo apt-get install libgtk2.0-dev

- Βήμα 5: Για την επεξεργασία μιας ροής βίντεο και την πρόσβαση σε μεμονωμένα καρέ πρέπει να γίνει η εγκατάσταση των παρακάτω.

  1. $ sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev

- Βήμα 6: Εγκατάσταση βιβλιοθηκών για την βελτιστοποίηση ορισμένων συναρτήσεων της OpenCV.

  1. $ sudo apt-get install libatlas-base-dev gfortran

- Βήμα 7: Εγκατάσταση του pip, ενός οδηγού πακέτων της Python.

  1. $ wget https://bootstrap.pypa.io/get-pip.py
  2. $ sudo python get-pip.py

- Βήμα 8: Εγκατάσταση των πακέτων virtualenv και virtualenvwrapper. Τα πακέτα αυτά δημιουργούν ξεχωριστά περιβάλλοντα της Python για κάθε εργασία που εκτελείται.

  1. $ sudo pip install virtualenv virtualenvwrapper
  2. $ sudo rm -rf ~/.cache/pip

  Μετά την εγκατάσταση των παραπάνω πακέτων πρέπει να αναβαθμιστεί το αρχείο ~/.bashrc

```
1. # virtualenv and virtualenvwrapper
2. export WORKON_HOME=$HOME/.virtualenvs
3. source /usr/local/bin/virtualenvwrapper.sh
```

Για να τεθούν σε ισχύ οι παραπάνω αλλαγές στο ~/.bashrc αρχείο, πρέπει ο χρήστης του λειτουργικού να επανασυνδεθεί, κλείνοντας και ανοίγοντας ένα νέο τερματικό.

```
1. $ source ~/.bashrc
```

Τέλος, δημιουργεί το εικονικό περιβάλλον cv (computer vision) στο οποίο θα αναπτυχθεί ο κώδικας.

```
1. mkvirtualenv cv
```

- Βήμα 9: Εγκατάσταση των εργαλείων προγραμματιστή για την Python 2.7

```
1. $ sudo apt-get install python2.7-dev
```

Εγκατάσταση του NumPy, στο εικονικό περιβάλλον cv. Το OpenCV αντιπροσωπεύει τις εικόνες σαν πολυδιάστατους NumPy πίνακες.

```
1. $ pip install numpy
```

- Βήμα 10: Το περιβάλλον είναι έτοιμο. Κατέβασμα του OpenCV από το Github.

```
1. $ cd ~
2. git clone https://github.com/Itseez/opencv.git
3. cd opencv
4. git checkout 3.0.0
```

Εγκατάσταση του opencv_contrib για την πρόσβαση σε βασικούς ανιχνευτές και τοπικές μεταβλητές.

1. $ cd ~
2. $ git clone https://github.com/ltseez/opencv_contrib.git
3. $ cd opencv_contrib
4. $ git checkout 3.0.0

- Βήμα 11: Παραμετροποίηση της έκδοσης του OpenCV.

1. $ cd ~/opencv
2. $ mkdir build
3. $ cd build
4. $ cmake -D CMAKE_BUILD_TYPE=RELEASE \
5. -D CMAKE_INSTALL_PREFIX=/usr/local \
6. -D INSTALL_C_EXAMPLES=ON \
7. -D INSTALL_PYTHON_EXAMPLES=ON \
8. -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib/modules \
9. -D BUILD_EXAMPLES=ON

- Βήμα 12: Μετά την παραμετροποίηση, ξεκινάει η μεταγλώττιση της πλατφόρμας του OpenCV.

1. $ make -j4

Ο αριθμός 4 είναι ο αριθμός των πυρήνων που είναι διαθέσιμοι στο σύστημα.

Το επόμενο βήμα μετά την μεταγλώττιση είναι η εγκατάσταση της πλατφόρμας OpenCV.

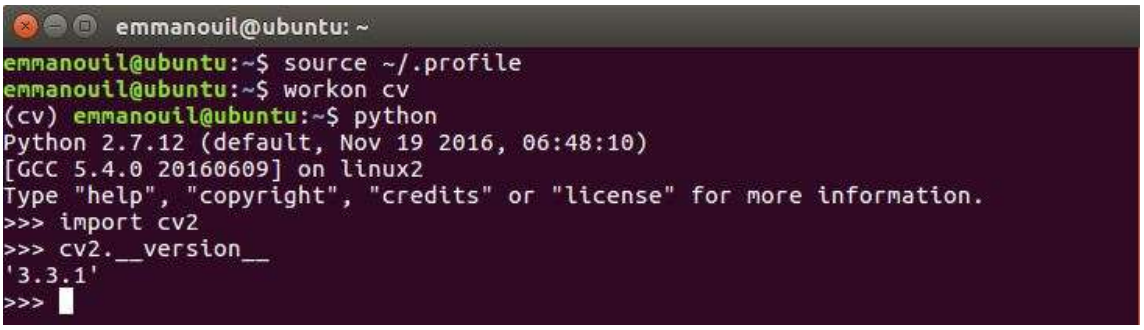
1. $ sudo make install
2. $ sudo ldconfig

- Βήμα 13: Η πλατφόρμα OpenCV είναι εγκατεστημένη στην τοποθεσία /usr/local/lib/python2.7/site-packages . Ο εικονικός χώρος εργασίας cv βρίσκεται σε διαφορετικό χώρο. Με τις παρακάτω εντολές συνδέονται τα site-packages με το χώρο εργασίας.

1. $ cd ~/.virtualenvs/cv/lib/python2.7/site-packages/
2. $ ln -s /usr/local/lib/python2.7/site-packages/cv2.so cv2.so

- Βήμα 14: Η εγκατάσταση της πλατφόρμας OpenCV 3.0 και της Python 2.7 έχει ολοκληρωθεί. Για να επιβεβαιωθεί ότι το εικονικό περιβάλλον έχει τις σωστές εκδόσεις πρέπει να εκτελεστούν οι παρακάτω εντολές.

1. $ workon cv
2. $ python
3. >>> import cv2
4. >>> cv2.__version__
5. '3.0.0'



Εικόνα 20. *Εγκαταστημένες εκδόσεις Python και OpenCV*

ΚΕΦΑΛΑΙΟ 4

ΥΛΟΠΟΙΗΣΗ ΚΑΙ ΠΕΡΙΓΡΑΦΗ ΚΩΔΙΚΑ

Σε αυτό το κεφάλαιο θα αναλυθούν και θα περιγραφούν μεμονωμένα όλα τα κομμάτια του κώδικα τα οποία συνθέτουν την λειτουργία του προγράμματος. Έχοντας εγκαταστήσει όλα τα προ απαιτούμενα για την εύρυθμη λειτουργία του προγράμματος στην προηγούμενη ενότητα, η ανάλυση θα συνεχιστεί με το κατακερματισμό του κώδικα σε μικρότερα κομμάτια για να γίνει όσο το δυνατόν πιο κατανοητή η διαδικασία καταμέτρησης ατόμων μέσα από την καταγραφή ενός απλού βίντεο.

4.1 Ανοίγοντας μια ροή βίντεο

Ξεκινώντας την ανάλυση του κώδικα θα περιγραφεί το κομμάτι του ανοίγματος μιας ροής βίντεο που καταγράφεται από την κάμερα του RPi μέσω της πλατφόρμας OpenCV.

Η πρώτη προσπάθεια έγινε με την εισαγωγή ενός αρχείου βίντεο. Τροποποιώντας ελάχιστα τον αρχικό κώδικα και αποθηκεύοντας τον κώδικα αυτό στον ίδιο χώρο με το βίντεο που θα εισαχθεί, μπορεί να εκτελεστεί η πρώτη δοκιμή. Το κομμάτι του κώδικά ξεκινάει πάντα με την εισαγωγή των βιβλιοθηκών NumPy και OpenCV. Έπειτα το άνοιγμα του βίντεο γίνεται με την συνάρτηση VideoCapture και δίνοντας ως παράμετρο την ονομασία του αρχείου προς ανάγνωση. Στην συνέχεια, διαβάζονται τα καρέ του βίντεο ένα-ένα μέχρι να τελειώσουν. Τέλος, ο κώδικας βγαίνει από την επανάληψη και κλείνει το αρχείο βίντεο και το παράθυρο.

```python
import numpy as np
import cv2

cap = cv2.VideoCapture('peopleCounter.avi') #Open video file

#Show all video properties
for i in range(19):
    print i, cap.get(i)

cap.set(3,160) #Set width
cap.set(4,120) #Set height

while(cap.isOpened()):
    ret, frame = cap.read()
    try:
        cv2.imshow('Frame',frame)
    except:
        print('EOF')
        break

    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break

cap.release()
cv2.destroyAllWindows()
```

Για το άνοιγμα ροής βίντεο που καταγράφεται σε πραγματικό χρόνο από την κάμερα, χρησιμοποιείται η ίδια συνάρτηση αλλά με τον αριθμό «0» ως παράμετρο. Η παράμετρος «0» ορίζει την κάμερα ως συσκευή εισόδου.

Υπάρχουν αρκετές ιδιότητες της συνάρτησης VideoCapture που δίνουν πρόσβαση ή ακόμα και δυνατότητα αλλαγής στα καρέ που διαβάζονται. Κάποιες από αυτές είναι:

- CAP_PROP_POS_MSEC Τρέχον καρέ σε χιλιοστά του δευτερολέπτου.
- CAP_PROP_POS_FRAMES Δείκτης στα καρέ.
- CAP_PROP_POS_AVI_RATIO Αναφορική θέση σε σχέση με την αρχή και το τέλος μέσα στο βίντεο.
- CAP_PROP_FRAME_WIDTH Πλάτος του καρέ.
- CAP_PROP_FRAME_HEIGHT Ύψος του καρέ.
- CAP_PROP_FPS Καρέ το δευτερόλεπτο
- CAP_PROP_FRAME_COUNT Αριθμός των καρέ σε ένα αρχείο βίντεο.

- CAP_PROP_MODE Τιμή που υποδεικνύει την τρέχουσα λειτουργία καταγραφής.
- CAP_PROP_BRIGHTNESS Φωτεινότητα του καρέ.
- CAP_PROP_CONTRAST Αντίθεση του καρέ.
- CAP_PROP_SATURATION Κορεσμός των χρωμάτων του καρέ.
- CAP_PROP_HUE Απόχρωση του καρέ.
- CAP_PROP_EXPOSURE Έκθεση του καρέ.
- CAP_PROP_CONVERT_RGB Μετατροπέας RGB.

```
1  import numpy as np
2  import cv2
3
4  cap = cv2.VideoCapture(0) #Start camera feed
5
6  while(cap.isOpened()):
7      ret, frame = cap.read()
8      try:
9          cv2.imshow('Frame',frame)
10     except:
11         print('EOF')
12         break
13
14     k = cv2.waitKey(30) & 0xff
15     if k == 27:
16         break
17
18 cap.release()
19 cv2.destroyAllWindows()
```

Τέλος, χρησιμοποιούνται οι εντολές set και get για να δοθούν νέες τιμές στις παραμέτρους του βίντεο. Στο επόμενο κομμάτι κώδικα έχει οριστεί η τιμή 160 και η τιμή 120 για το πλάτος και το μήκος του καρέ αντίστοιχα.

```
import numpy as np
import cv2

cap = cv2.VideoCapture(0) #Start camera feed

#Show all video properties
for i in range(19):
    print i, cap.get(i)

cap.set(3,160) #Set width
cap.set(4,120) #Set height

while(cap.isOpened()):
    ret, frame = cap.read()
    try:
        cv2.imshow('Frame',frame)
    except:
        print('EOF')
        break

    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break

cap.release()
cv2.destroyAllWindows()
```

4.2 Σχεδιασμός γραμμών στο παράθυρο του βίντεο

Το επόμενο βήμα στην σύνθεση του κώδικα είναι ο σχεδιασμός γραμμών στο παράθυρο του βίντεο με σκοπό να γίνει κατανοητή η περιοχή θα μελετηθεί καθώς και να οπτικοποιηθούν οι νοητές γραμμές καταμέτρησης που αναφέρθηκαν στην εργασία.

Στον κώδικα που ακολουθεί χρησιμοποιούνται οι μέθοδοι cap.get() για να βρεθούν οι συντεταγμένες του μέσον στο βίντεο (width/2, height/2). Έπειτα, πριν το κάλεσμα της συνάρτησης imshow(), χρησιμοποιείται η μέθοδος cv2.PutText() για να εισαχθεί κείμενο στο καρέ του βίντεο. Η συνάρτηση μαζί με τις παραμέτρους έχει την εξής μορφή: cv.PutText(img, text, org, font, color). Το σημείο αναφοράς (0,0) είναι το κάτω αριστερά σημείο στο εν λόγω καρέ.

```python
import numpy as np
import cv2

cap = cv2.VideoCapture('peopleCounter.avi') #Open video file

w = cap.get(3) #Get width
h = cap.get(4) #Get height

mx = int(w/2)
my = int(h/2)

count = 0

while(cap.isOpened()):
    ret, frame = cap.read() #Read a frame
    try:
        count = count + 1
        text = "Hello World " + str(count)
        cv2.putText(frame, text ,(mx,my),cv2.FONT_HERSHEY_SIMPLEX
                    ,1,(255,255,255),1,cv2.LINE_AA)
        cv2.imshow('Frame',frame)
    except:
        #If there are no more frames to show...
        print('EOF')
        break

    #Abort and exit with 'Q' or ESC
    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break

cap.release() #Release video file
cv2.destroyAllWindows() #Close all OpenCV windows
```

Το οπτικό αποτέλεσμα τρέχοντας τον παραπάνω κώδικα σε ένα τερματικό είναι το εξής:

Εικόνα 21. *Εμφάνιση κειμένου στο frame του προγράμματος*

Εκτός από την εισαγωγή κειμένου, όπως αναφέρθηκε και πιο πάνω μπορούν να σχεδιαστούν γραμμές, κύκλοι και ότι άλλο γεωμετρικό σχήμα υπάρχει στην βιβλιοθήκη μεθόδων της πλατφόρμας OpenCV.

Στον παρακάτω κώδικα δημιουργούνται δυο παράθυρα εκ των οποίων το ένα είναι ακατέργαστο ενώ στο δεύτερο σχεδιάζονται δυο γραμμές με διαφορετικό χρώμα (μπλε και κόκκινο), μέσα στο καρέ του βίντεο. Για να λειτουργήσει η μέθοδος polylines, χρειάζονται έναν NumPy πίνακα με ζευγάρια συντεταγμένων (x και y) για κάθε σημείο της γραμμής. Στην συγκεκριμένη περίπτωση ορίζονται οι συντεταγμένες της αρχής και του τέλους της γραμμής και μετά με την μέθοδο reshape(-1,1,2), βρίσκονται τα ενδιάμεσα σημεία που αποτελούν το σύνολο της γραμμής.

```
import numpy as np
import cv2

cap = cv2.VideoCapture('peopleCounter.avi') #Open video file

while(cap.isOpened()):
    ret, frame = cap.read() #Read a frame
    try:
        cv2.imshow('Frame',frame)
        frame2 = frame
    except:
        #If there are no more frames to show...
        print('EOF')
        break

    line1 = np.array([[100,100],[300,100],[350,200]], np.int32).reshape((-1,1,2))
    line2 = np.array([[400,50],[450,300]], np.int32).reshape((-1,1,2))

    frame2 = cv2.polylines(frame2,[line1],False,(255,0,0),thickness=2)
    frame2 = cv2.polylines(frame2,[line2],False,(0,0,255),thickness=1)

    cv2.imshow('Frame 2',frame2)

    #Abort and exit with 'Q' or ESC
    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break

cap.release() #Release video file
cv2.destroyAllWindows() #Close all openCV windows
```

Το οπτικό αποτέλεσμα τρέχοντας τον παραπάνω κώδικα σε ένα τερματικό είναι το εξής:

Εικόνα 22. *Εμφάνιση γραμμών στο frame του προγράμματος*

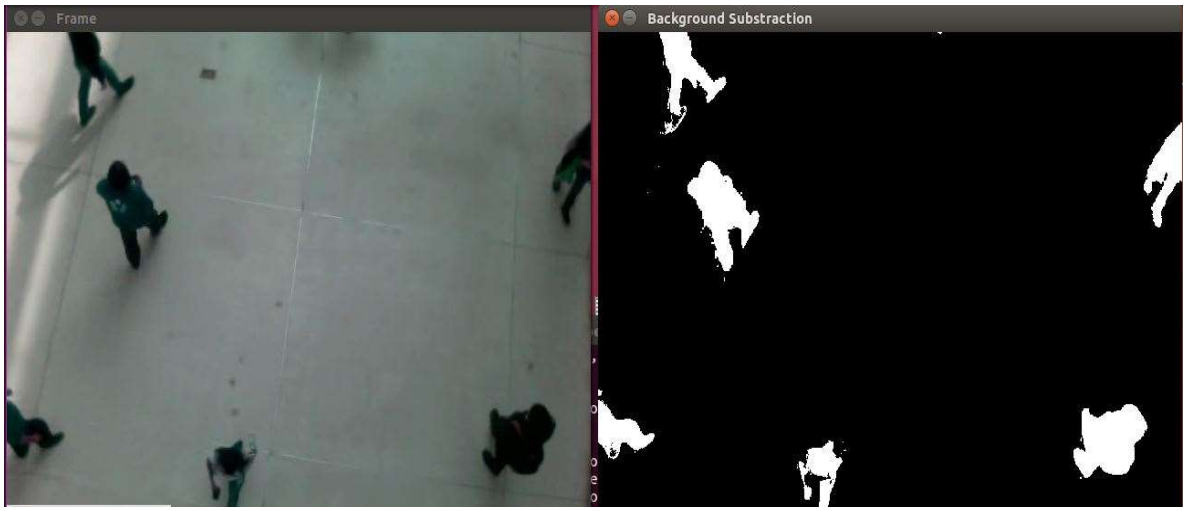
4.3 Αφαίρεση φόντου

Στη συνέχεια της υλοποίησης του συστήματος καταμέτρησης, πρέπει να αφαιρεθεί το υπόβαθρο από το καρέ για να μπορεί να προσδιοριστεί ποιο είναι το προσκήνιο και ποιο το φόντο. Το φόντο ή αλλιώς υπόβαθρο ορίζεται ως ένα κάτι σταθερό σε μια σειρά εναλλαγμένων εικόνων. Δηλαδή, οτιδήποτε παραμένει στατικό στις διάφορες αλλαγές που γίνονται σε πολλαπλά καρέ. Από την άλλη το προσκήνιο είναι ότι αλλάζει ή κινείται.

Για να γίνει η διαδικασία της αφαίρεσης φόντου στην πλατφόρμα OpenCV, χρειάζονται μόλις δυο γραμμές στον κώδικά. Παρακάτω παρατίθεται ο κώδικα της υλοποίησης:

```python
import numpy as np
import cv2

#Open video file
cap = cv2.VideoCapture('peopleCounter.avi')

#Create the background substractor
fgbg = cv2.createBackgroundSubtractorMOG2(detectShadows = True)

while(cap.isOpened()):
    ret, frame = cap.read() #Read a frame

    fgmask = fgbg.apply(frame) #Use the substractor

    try:
        cv2.imshow('Frame',frame)
        ret,img2 = cv2.threshold(fgmask,127,255,cv2.THRESH_BINARY)
        cv2.imshow('Background Substraction',img2)
    except:
        #If there are no more frames to show...
        print('EOF')
        break

    #Abort and exit with 'Q' or ESC
    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break

cap.release() #Release video file
cv2.destroyAllWindows() #Close all openCV windows
```

Το οπτικό αποτέλεσμα τρέχοντας τον παραπάνω κώδικα σε ένα τερματικό είναι το εξής:



Εικόνα 23. *Αφαίρεση φόντου και σκιών. Μετατροπή έγχρωμης εικόνας σε ασπρόμαυρη.*

Τα αποτελέσματα των παραπάνω εικόνων δείχνουν ότι στην νέα εικόνα που έχει παραχθεί, το μαύρο αντιπροσωπεύει το φόντο, το λευκό είναι τα αντικείμενα στο προσκήνιο και τα γκρίζα σημεία είναι οι σκιές που εκπέμπονται από αυτά τα αντικείμενα.

Η μέθοδος MOG2 αφαίρεσης του φόντου στην πλατφόρμα OpenCV, είναι πολύ χρήσιμη διότι στην συγκεκριμένη συνάρτηση το φόντο υπολογίζεται συνεχώς, που σημαίνει ότι οι αλλαγές στην φωτεινότητα (όπως για παράδειγμα αυτές που προκαλούνται λόγω του ήλιου) δεν επηρεάζουν τα αποτελέσματα των μετρήσεών στη πάροδο του χρόνου.

4.4 Μορφολογικοί μετασχηματισμοί

Μετά την αφαίρεση που φόντου και την επεξεργασία του, σειρά έχουν οι μορφολογικοί μετασχηματισμοί. Οι μορφολογικοί μετασχηματισμοί είναι μια σειρά συναρτήσεων που έχουν ως σκοπό να επεξεργαστούν εικόνες βασισμένοι στα σχήματα που υπάρχουν μέσα σε αυτές. Οι μορφολογικές μέθοδοι εφαρμόζουν ένα στοιχείο δομής (structuring element) σε μια εικόνα και παράγουν μια νέα μετά την εφαρμογή του.

Οι δυο πιο βασικές συναρτήσεις είναι οι Erosion και Dilation. Το εύρος των λειτουργιών τους είναι μεγάλο, αλλά οι κυριότεροι λόγοι που χρησιμοποιούνται στην συγκεκριμένη περίπτωση είναι:

- Αφαίρεση του θορύβου σε μια εικόνα.
- Απομόνωση στοιχείων σε μια εικόνα και ένωση κάποιων άλλων σε μια διαφορετική.
- Εύρεση και εξομάλυνση ανωμαλιών χρωματικής έντασης ή οπών σε μια εικόνα.

Γενικότερα, οι συναρτήσεις Erosion και Dilation χρησιμοποιούνται σε δυαδικές (binary) εικόνες (άσπρο και μαύρο). Η συνάρτηση Erosion έχει την τάση να επεκτείνει τα μαύρα τμήματα σε μια εικόνα και να τα μετατρέπει σε άσπρα τμήματα. Μειώνει τον αριθμό των λευκών pixel και αυξάνει τον αριθμό τον μαύρων. Ακόμα, μειώνει τις οπές ανάμεσα στα στοιχεία μιας εικόνας. Από την άλλη πλευρά, η συνάρτηση Dilation κάνει ακριβώς το αντίθετο, δηλαδή, μετατρέπει και επεκτείνει τα άσπρα τμήματα μιας εικόνας σε μαύρα.

Για να λειτουργήσουν οι δυο αυτές μέθοδοι, όπως αναφέρθηκε και παραπάνω χρειάζεται να οριστεί ένα στοιχείο δομής (structuring element or kernel). Το στοιχείο δομής είναι ένα πίνακας μεγέθους n x n που καθορίζει την περιοχή που θα χρησιμοποιηθεί κατά τον υπολογισμό της τιμής του κάθε pixel.

Ακόμα, χρησιμοποιείται η μέθοδος threshold για να μετατραπεί η εικόνα επεξεργασίας σε δυαδική. Δηλαδή, να την μετατραπούν τα έγχρωμα pixels σε ασπρόμαυρα μόνο (0 και 1).

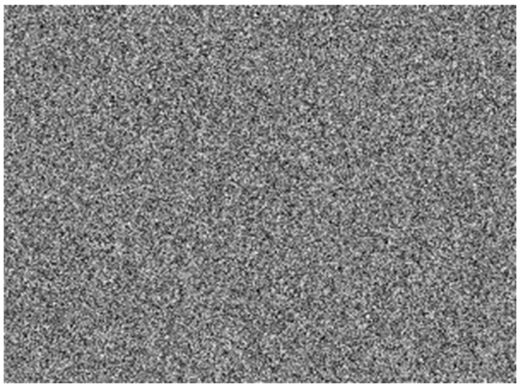
Τέλος, στον αλγόριθμό χρησιμοποιούνται δυο μέθοδοι της πλατφόρμας OpenCV οι οποίες είναι ο συνδυασμός των συναρτήσεων Erosion και Dilation που αναφέρθηκε παραπάνω. Η πρώτη ονομάζεται Opening και επεξεργάζεται την εικόνα χρησιμοποιώντας με σειρά τις μεθόδους Erosion και Dilation, ενώ η δεύτερη ονομάζεται Closing και εφαρμόζει με σειρά τις μεθόδους Dilation και Erosion, δηλαδή την ανάποδη διαδικασία.

Παρακάτω αναλύονται τρία διαφορετικά παραδείγματα με την εφαρμογή των συναρτήσεων που αναφέρθηκαν και τους κώδικες που χρησιμοποιήθηκαν.

Εφαρμόζοντας τον παρακάτω κώδικα με τις μεθόδους Erotion και Dilation στην εικόνα «noise» παρατηρούνται τα εξής αποτελέσματα:

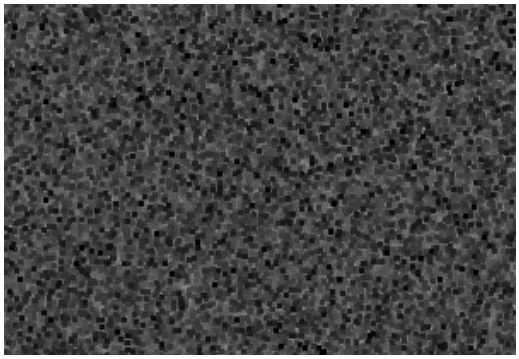
```
1   import cv2
2   import numpy as np
3
4   img = cv2.imread("noise.png")
5
6   ret,threshl = cv2.threshold(img,127,255,cv2.THRESH_BINARY)
7
8   kernel = np.ones((3,3),np.uint8)
9
10  opening = cv2.morphologyEx(threshl, cv2.MORPH_OPEN, kernel)
11  closing = cv2.morphologyEx(threshl, cv2.MORPH_CLOSE, kernel)
12
13  cv2.imwrite("erode.png",erosion)
14  cv2.imwrite("dilate.png",dilation)
```

ΠΡΙΝ:



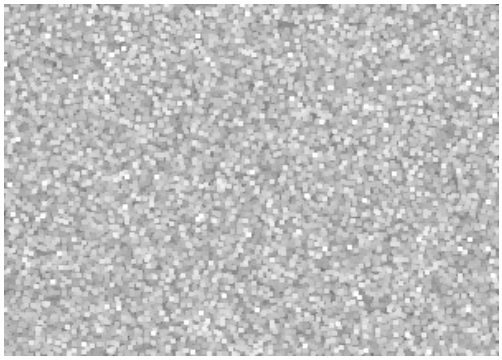
Εικόνα 24. *Εικόνα θορύβου*

ΜΕΤΑ (Erosion):



Εικόνα 25. *Εικόνα θορύβου μετά τη μέθοδο Erosion*

ΜΕΤΑ (Dilation):



Εικόνα 26. *Εικόνα θορύβου μετά τη μέθοδο Dilation*

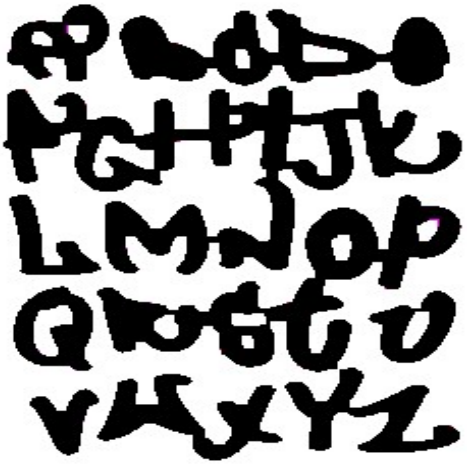
Εφαρμόζοντας τον παρακάτω κώδικα με τις μεθόδους Opening και Closing στην εικόνα «letters» παρατηρούνται τα εξής αποτελέσματα:

```
import cv2
import numpy as np

img = cv2.imread("letters.jpg")

ret,threshl = cv2.threshold(img,200,255,cv2.THRESH_BINARY)

kernel = np.ones((5,5),np.uint8)

opening = cv2.morphologyEx(threshl, cv2.MORPH_OPEN, kernel)
closing = cv2.morphologyEx(threshl, cv2.MORPH_CLOSE, kernel)

cv2.imwrite("letters_closing.png",closing)
cv2.imwrite("letters_opening.png",opening)
```

ΠΡΙΝ:

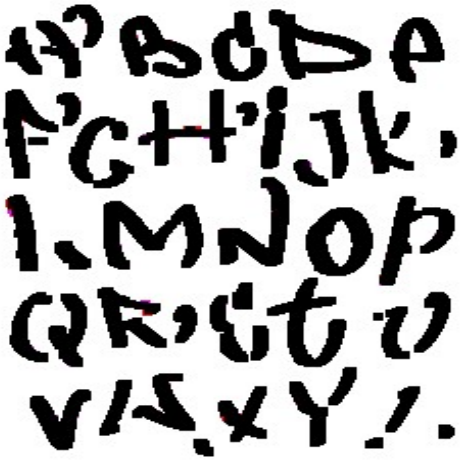
Εικόνα 27. *Εικόνα με γράμματα προς επεξεργασία*

META (Opening):



Εικόνα 28. *Εικόνα επεξεργασίας μετά τη μέθοδο Opening*

META (Closing):

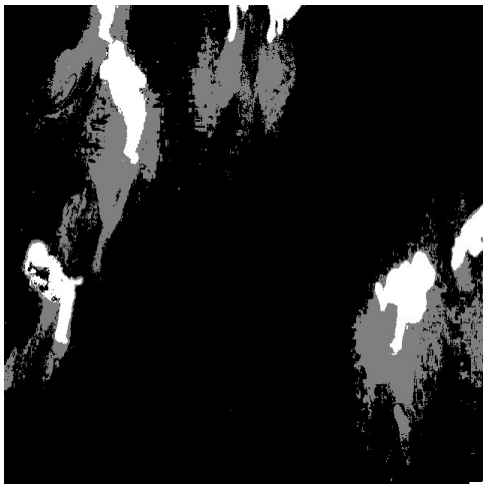


Εικόνα 29. *Εικόνα επεξεργασίας μετά τη μέθοδο Closing*

Εφαρμόζοντας τον παρακάτω κώδικα με τις μεθόδους Opening και Closing στην εικόνα «people_shadows» παρατηρούνται τα εξής αποτελέσματα:

```
import cv2
import numpy as np

img = cv2.imread("peoplecount-filter2.png")

ret,threshl = cv2.threshold(img,200,255,cv2.THRESH_BINARY)

kernel = np.ones((5,5),np.uint8)

opening = cv2.morphologyEx(threshl, cv2.MORPH_OPEN, kernel)
closing = cv2.morphologyEx(threshl, cv2.MORPH_CLOSE, kernel)

cv2.imwrite("people_closing.png",closing)
cv2.imwrite("people_opening.png",opening)
```

ΠΡΙΝ:



Εικόνα 30. *Εικόνα μετά την αφαίρεση του φόντου και την μετατροπή σε Binary Image*

ΜΕΤΑ (Opening):

Εικόνα 31. *Εφαρμογή της μεθόδου Opening για την αφαίρεση των σκιών από την εικόνα*

ΜΕΤΑ (Closing):

Εικόνα 32. *Εφαρμογή της μεθόδου Closing για την αφαίρεση των σκιών από την εικόνα*

Παρατηρείται ότι μετά την αφαίρεση του φόντου, την μετατροπή της εικόνας σε δυαδική, την αφαίρεση του θορύβου, την αφαίρεση των σκιών (γκρι χρώμα) και τον ευδιάκριτο σχηματισμό των ασπρόμαυρων pixels, τα στοιχεία στην εικόνα γίνονται ολοένα και πιο καθαρά.

4.5 Εύρεση περιγραμμάτων

Ως τώρα, έχουν φιλτραριστεί τα καρέ του βίντεο που επεξεργάζονται σε πραγματικό χρόνο. Το επόμενο βήμα είναι να ανιχνευτούν τα περιγράμματα στα στοιχεία που εμφανίζονται στα καρέ. Η πλατφόρμα OpenCV προσφέρει μια συνάρτηση με το όνομα findContours() που θα βοηθήσει με αυτή την διαδικασία.

Τα περιγράμματα μπορούν να εξηγηθούν απλά ως μια καμπύλη που ενώνει όλα τα συνεχή σημεία (κατά μήκος του ορίου), με το ίδιο χρώμα ή ένταση. Τα περιγράμματα είναι ένα χρήσιμο εργαλείο για την ανάλυση σχήματος και την ανίχνευση και αναγνώριση αντικειμένων. Για την επιτυχή αναζήτηση των περιγραμμάτων θα πρέπει όλες οι εικόνες να είναι δυαδικές (0 και 1). Η εύρεση των ολογραμμάτων είναι σαν την εύρεση του λευκού αντικειμένου από μαύρο φόντο. Έτσι λοιπόν, το αντικείμενο που αναζητούμε θα πρέπει να είναι λευκό και το φόντο μαύρο.

Παρακάτω ο κώδικας εφαρμογής για την εύρεση των περιγραμμάτων.

```
import numpy as np
import cv2

cap = cv2.VideoCapture('peopleCounter.avi') #Open video file

fgbg = cv2.createBackgroundSubtractorMOG2(detectShadows = True) #Create the background substractor

kernelOp = np.ones((3,3),np.uint8)
kernelCl = np.ones((11,11),np.uint8)

while(cap.isOpened()):
    ret, frame = cap.read() #Read a frame

    fgmask = fgbg.apply(frame) #Use the substractor
    try:
        ret,imBin= cv2.threshold(fgmask,200,255,cv2.THRESH_BINARY)
        #Opening (erode->dilate) to remove noise.
        mask = cv2.morphologyEx(imBin, cv2.MORPH_OPEN, kernelOp)
        #Closing (dilate -> erode) to join white regions.
        mask = cv2.morphologyEx(mask , cv2.MORPH_CLOSE, kernelCl)
    except:
        #If there are no more frames to show.
        print('EOF')
        break

    _, contours0, hierarchy = cv2.findContours(mask,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_NONE)
    for cnt in contours0:
        cv2.drawContours(frame, cnt, -1, (0,255,0), 3, 8)
    cv2.imshow('Frame',frame)
    #Abort and exit with 'Q' or ESC
    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break

cap.release() #Release video file
cv2.destroyAllWindows() #Close all openCV windows
```

Οι γραμμές κώδικα που αξίζουν περαιτέρω ανάλυση είναι οι εξής:

```
26      _, contours0, hierarchy = cv2.findContours(mask,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_NONE)
27      for cnt in contours0:
28          cv2.drawContours(frame, cnt, -1, (0,255,0), 3, 8)
```

Η λειτουργία που δίνεται στην μέθοδο findContours() είναι η cv2.RETR_EXTERNAL. Δίνοντας αυτή την παράμετρο στην μέθοδό σημαίνει ότι επιλέγονται μόνο τα εξωτερικά περιγράμματα, δηλαδή, περιγράμματα που βρίσκονται μέσα σε άλλα δεν θα συμπεριληφθούν στην ανάλυση. Η παράμετρος cv2.CHAIN_APPROX_NONE σημαίνει ότι όλα τα σημεία του περιγράμματος αποθηκεύονται και χρησιμοποιούνται για τον σχεδιασμό του.

Υπάρχουν αρκετοί παράμετροι στις μεθόδους της πλατφόρμας OpenCV. Με βάση τις ανάγκες και το οπτικό αποτέλεσμα που χρειάζεται να επιτευχθεί, επιλέγονται αναλόγως.

4.6 Ορισμός ατόμου καταμέτρησης

Σε αυτό το σημείο πρέπει να οριστεί πότε ένα περίγραμμα θα θεωρείται ένας άνθρωπος. Δεν υπάρχει κάποια συγκεκριμένη τιμή που ορίζει ότι ένα περίγραμμα είναι ένα άτομο. Ο λόγος είναι ότι η τιμή αυτή έχει να κάνει με πολλούς παραμέτρους και μπορεί να διαφέρει από σημείο με σημείο και εγκατάσταση με εγκατάσταση (απόσταση αντικειμένων από την κάμερα). Γι' αυτό τον λόγο θα πρέπει να γίνουν ορισμένες δοκιμές πριν δημιουργηθεί η κατάλληλη τιμή.

Ένας απλός αλλά αποτελεσματικός τρόπος να βρεθεί αυτό είναι να οριστεί μια ελάχιστη τιμή για την περιοχή που θα πρέπει να έχει το περίγραμμα.

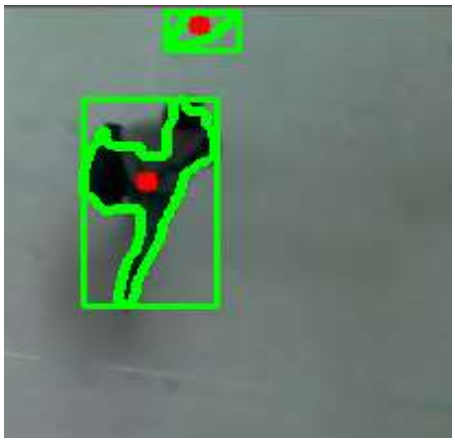
```
24      areaTH = #Some number
25      _, contours0, hierarchy = cv2.findContours(mask,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_NONE)
26      for cnt in contours0:
27          cv2.drawContours(frame, cnt, -1, (0,255,0), 3, 8)
28          area = cv2.contourArea(cnt)
29          print area
30          if area > areaTH:
31              ##################
32              #   TRACKING     #
33              ##################
```

Αναλύοντας το παραπάνω κομμάτι κώδικα θα πρέπει να θεωρηθεί μια ελάχιστη περιοχή που θα πρέπει ορίζει ένα περίγραμμα ανθρώπου, να βρεθούν τα περιγράμματα και για κάθε ένα από αυτά, που η περιοχή που περικλείει είναι

μεγαλύτερη από την περιοχή (areaTH) που έχει οριστεί τότε θα κληθεί μια διαδικασία που θα ασχολείται με αυτά μέσα στην εντολή AN.

Η τιμή της μεταβλητής areaTH δεν είναι καθολική. Όπως αναφέρθηκε πιο πάνω, η τιμή αυτή εξαρτάται από πολλούς παραμέτρους που σημαίνει ότι εξαρτάται από τη ροή βίντεο που εισάγετε στον κώδικα. Αν η ροή βίντεο αλλάξει θα πρέπει να γίνουν αρκετές δοκιμές με διαφορετικές τιμές της μεταβλητής areaTH έως ότου λειτουργεί σωστά με το βίντεο.

Για παράδειγμα, σε περίπτωση που οριστεί areaTH μικρότερο από τα περιγράμματα των στοιχείων των εικόνων το αποτέλεσμα θα είναι το εξής:



Εικόνα 33. *Δημιουργία ορθογωνίου περιγράμματος που περικλείει το εντοπισμένο αντικείμενο*

Ενώ από την άλλη, αν η τιμή του areaTH είναι πολύ μεγαλύτερη από τα περιγράμματα των στοιχείων το αποτέλεσμα θα είναι ως εξής:

Εικόνα 34. *Η δημιουργία ορθογώνιου περιγράμματος δεν είναι δυνατή γιατί το αντικείμενο είναι μεγαλύτερο από το threshold που έχει οριστεί.*

Παρακάτω ο κώδικας για τον ορισμό των ατόμων:

```
import numpy as np
import cv2

cap = cv2.VideoCapture('peopleCounter.avi') #Open video file
fgbg = cv2.createBackgroundSubtractorMOG2(detectShadows = True) #Create the background substractor

kernelOp = np.ones((3,3),np.uint8)
kernelCl = np.ones((11,11),np.uint8)
areaTH = 500

while(cap.isOpened()):
    ret, frame = cap.read() #Read a frame

    fgmask = fgbg.apply(frame) #Use the substractor
    try:
        ret,imBin= cv2.threshold(fgmask,200,255,cv2.THRESH_BINARY)
        #Opening (erode->dilate) to remove noise.
        mask = cv2.morphologyEx(imBin, cv2.MORPH_OPEN, kernelOp)
        #Closing (dilate -> erode) to join white regions.
        mask =  cv2.morphologyEx(mask , cv2.MORPH_CLOSE, kernelCl)
    except:
        #If there are no more frames to show.
        print('EOF')
        break
    areaTH = #Some number
    _, contours0, hierarchy = cv2.findContours(mask,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_NONE)
    for cnt in contours0:
        cv2.drawContours(frame, cnt, -1, (0,255,0), 3, 8)
        area = cv2.contourArea(cnt)
        print area
        if area > areaTH:
            #################
            #   TRACKING    #
            #################
            M = cv2.moments(cnt)
            cx = int(M['m10']/M['m00'])
            cy = int(M['m01']/M['m00'])
            x,y,w,h = cv2.boundingRect(cnt)
            cv2.circle(frame,(cx,cy), 5, (0,0,255), -1)
            img = cv2.rectangle(frame,(x,y),(x+w,y+h),(0,255,0),2)

    cv2.imshow('Frame',frame)

    #Abort and exit with 'Q' or ESC
    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break

cap.release() #Release video file
cv2.destroyAllWindows() #Close all openCV windows
```

4.7 Καταγραφή της κίνησης

Σε αυτή την ενότητα, γνωρίζοντας πλέον πότε υπάρχει ένα ανθρώπινο περίγραμμα σε μια εικόνα, θα αναλυθεί η κίνηση και η κατεύθυνση της κίνησης του ατόμου αυτού (πάνω ή κάτω / μέσα ή έξω).

Στο πρώτο καρέ που στην περιοχή ανίχνευσης θα εμφανιστεί ένας άνθρωπος, θα του ανατεθεί ένας μοναδικός αριθμός και θα αποθηκευτεί η αρχική του θέση στην εικόνα. Στη συνέχεια, στα επόμενα καρέ, θα συνεχίσει να ανιχνεύεται και να παρακολουθείται η κίνηση του, να συγκρίνεται η ετικέτα που του ανατέθηκε στο πρώτο καρέ και να αποθηκεύονται σε κάθε καρέ οι συντεταγμένες του. Ακόμα, όταν το άτομο αυτό περάσει κάποιο από τα όρια που έχουν τεθεί στην εικόνα, τότε μπορεί να αξιολογηθεί η κίνηση του ανθρώπου συγκρίνοντας το, με όλα τα σημεία που έχουν αποθηκευτεί στα προηγούμενα καρέ.

Για να διαχειριστεί η ετικετοποίηση και η μαζική αποθήκευση των συντεταγμένων, έπρεπε να δημιουργηθεί μια κλάση με το όνομα Person(). Η συνάρτηση δεν είναι η βέλτιστη λύση. Παρακάτω ο κώδικας της κλάσης Person():

Πλατφόρμα ανίχνευσης έξυπνης πόλης για τον εντοπισμό, την ανίχνευση και την καταμέτρηση ατόμων

```python
from random import randint
import time

class MyPerson:
    tracks = []
    def __init__(self, i, xi, yi, max_age):
        self.i = i
        self.x = xi
        self.y = yi
        self.tracks = []
        self.R = randint(0,255)
        self.G = randint(0,255)
        self.B = randint(0,255)
        self.done = False
        self.state = '0'
        self.age = 0
        self.max_age = max_age
        self.dir = None
    def getRGB(self):
        return (self.R,self.G,self.B)
    def getTracks(self):
        return self.tracks
    def getId(self):
        return self.i
    def getState(self):
        return self.state
    def getDir(self):
        return self.dir
    def getX(self):
        return self.x
    def getY(self):
        return self.y
    def updateCoords(self, xn, yn):
        self.age = 0
        self.tracks.append([self.x,self.y])
        self.x = xn
        self.y = yn
    def setDone(self):
        self.done = True
    def timedOut(self):
        return self.done
```

```
42      def going_UP(self,mid_start,mid_end):
43          if len(self.tracks) >= 2:
44              if self.state == '0':
45                  if self.tracks[-1][1] < mid_end and self.tracks[-2][1] >= mid_end: #Cross the line
46                      state = '1'
47                      self.dir = 'up'
48                      return True
49              else:
50                  return False
51          else:
52              return False
53      def going_DOWN(self,mid_start,mid_end):
54          if len(self.tracks) >= 2:
55              if self.state == '0':
56                  if self.tracks[-1][1] > mid_start and self.tracks[-2][1] <= mid_start: #Cross the line
57                      state = '1'
58                      self.dir = 'down'
59                      return True
60              else:
61                  return False
62          else:
63              return False
64      def age_one(self):
65          self.age += 1
66          if self.age > self.max_age:
67              self.done = True
68          return True
69  class MultiPerson:
70      def __init__(self, persons, xi, yi):
71          self.persons = persons
72          self.x = xi
73          self.y = yi
74          self.tracks = []
75          self.R = randint(0,255)
76          self.G = randint(0,255)
77          self.B = randint(0,255)
78          self.done = False
```

Παρακάτω ο υπόλοιπος κώδικας με τις συναρτήσεις της πλατφόρμας OpenCV.

Πλατφόρμα ανίχνευσης έξυπνης πόλης για τον εντοπισμό, την ανίχνευση και την καταμέτρηση ατόμων

```python
import numpy as np
import cv2
import Person
import time

cap = cv2.VideoCapture('peopleCounter.avi') #Open video file
fgbg = cv2.createBackgroundSubtractorMOG2(detectShadows = True) #Create the background substractor
kernelOp = np.ones((3,3),np.uint8)
kernelCl = np.ones((11,11),np.uint8)

#Variables
font = cv2.FONT_HERSHEY_SIMPLEX
persons = []
max_p_age = 5
pid = 1
areaTH = 500

while(cap.isOpened()):
    ret, frame = cap.read() #Read a frame

    fgmask = fgbg.apply(frame) #Use the substractor
    try:
        ret,imBin= cv2.threshold(fgmask,200,255,cv2.THRESH_BINARY)
        #Opening (erode->dilate) to remove noise.
        mask = cv2.morphologyEx(imBin, cv2.MORPH_OPEN, kernelOp)
        #Closing (dilate -> erode) to join white regions.
        mask =  cv2.morphologyEx(mask , cv2.MORPH_CLOSE, kernelCl)
    except:
        #if there are no more frames to show.
        print('EOF')
        break

    _, contours0, hierarchy = cv2.findContours(mask,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_NONE)
    for cnt in contours0:
        cv2.drawContours(frame, cnt, -1, (0,255,0), 3, 8)
        area = cv2.contourArea(cnt)
        if area > areaTH:
            ##################
            #   TRACKING     #
            ##################
            M = cv2.moments(cnt)
            cx = int(M['m10']/M['m00'])
            cy = int(M['m01']/M['m00'])
            x,y,w,h = cv2.boundingRect(cnt)

            new = True
            for i in persons:
                if abs(x-i.getX()) <= w and abs(y-i.getY()) <= h:
                    #The object is to close to one that has already been detected before
                    new = False
                    i.updateCoords(cx,cy)   #Update coordinates in the object and resets numbers
                    break
            if new == True:
                p = Person.MyPerson(pid,cx,cy, max_p_age)
                persons.append(p)
                pid += 1
            ##################
            #   DRAWINGS     #
            ##################
```

```
            cv2.circle(frame,(cx,cy), 5, (0,0,255), -1)
            img = cv2.rectangle(frame,(x,y),(x+w,y+h),(0,255,0),2)
            cv2.drawContours(frame, cnt, -1, (0,255,0), 3)

    #########################
    #    DRAW CORDINATES    #
    #########################
    for i in persons:
        if len(i.getTracks()) >= 2:
            pts = np.array(i.getTracks(), np.int32)
            pts = pts.reshape((-1,1,2))
            frame = cv2.polylines(frame,[pts],False,i.getRGB())
        if i.getId() == 9:
            print str(i.getX()), ',', str(i.getY())
        cv2.putText(frame, str(i.getId()),(i.getX(),i.getY()),font,0.3,i.getRGB(),1,cv2.LINE_AA)


    cv2.imshow('Frame',frame)

    #Abort and exit with 'Q' or ESC
    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break

cap.release() #Release video file
cv2.destroyAllWindows() #Close all openCV windows
```

Το σημαντικότερο κομμάτι του κώδικα είναι το εξής:

```
            for i in persons:
                if abs(x-i.getX()) <= w and abs(y-i.getY()) <= h:
                    #The object is to close to one that has already been detected before
                    new = False
                    i.updateCoords(cx,cy)#Update coordinates in the object and resets numbers
                    break
            if new == True:
                p = Person.MyPerson(pid,cx,cy, max_p_age)
                persons.append(p)
                pid += 1
```

Σε αυτό το κομμάτι κώδικα συλλέγονται οι συντεταγμένες του περιγράμματος και προσπαθούν να ταιριάξουνε με προηγούμενα εντοπισμένα άτομα σε διαφορετικά καρέ. Αν δεν υπάρξει κανένα ταίριασμα τότε δημιουργηθεί μια νέα ετικέτα.

Εικόνα 35. *Σχεδιασμός ίχνους πορείας των αντικειμένων σε μια εικόνα*

4.8 Καταμέτρηση ατόμων

Στην προηγούμενη ενότητα αναλύθηκε η πορεία της κίνηση των αντικειμένων στα διαδεχόμενα καρέ αποθηκεύοντας σε ένα πίνακα τις συντεταγμένες του σε κάθε κατάσταση. Το τελευταίο βήμα της εργασίας θα είναι η ανάλυση του πίνακα αυτού και ο υπολογισμός της κίνησης του αντικειμένου στην εικόνα. Αν δηλαδή η κίνηση του θα είναι προς τα πάνω ή προς τα κάτω.

Για να γίνει πιο κατανοητή η κίνηση των αντικειμένων δημιουργήθηκαν δυο νοητές γραμμές για να διευκολύνουν στην οπτική ανάλυση της πορείας της κίνησης των ανθρώπων (line_up, line_down). Ακόμα, θέτονται δυο νοητές γραμμές για να υποδείξουν στον κώδικα πότε θα σταματήσει να ανιχνεύει ένα αντικείμενο που του έχει αναθέσει ID το οποίο έχει βγει εκτός ορίων (up_limit, down_limit).

Στην κλάση Person, χρησιμοποιούνται δυο συναρτήσεις: going_UP(a,b) και going_down(a,b). Και οι δυο, δέχονται ως παράμετρο τις γραμμές line_down και line_up, και επιστρέφουν την τιμή «TRUE», αν αξιολογήσουν ότι το αντικείμενο πέρασε την γραμμή line_up ή την γραμμή line_down με την σωστή κατεύθυνση. Στην περίπτωση που συμβεί αυτό ο μετρητής (up or down) αυξάνεται με αποτέλεσμα, την καταμέτρηση των ατόμων σε πραγματικό χρόνο.

Τέλος, στην κλάση Person υπάρχει μια μεταβλητή με το όνομα «state», η οποία χρησιμοποιείται για να γνωρίζει πότε το αντικείμενο βρίσκεται εκτός των ορίων καταμέτρησης της εικόνας και να απελευθερώνει το ID που του είχε αναθέσει κατά την έξοδο από την περιοχή ανίχνευσης.

Παρακάτω ο ολοκληρωμένος κώδικας που χρησιμοποιήθηκε για την πλατφόρμα καταμέτρησης ατόμων και την αποστολή δεδομένων σε διαδικτυακή πλατφόρμα:

```
import numpy as np
import cv2
import Person
from ubidots import ApiClient
import threading
import time
import math

#Indexes for people going up and down
cnt_up   = 0
cnt_down = 0
flag = 1
```

```
#We need to set the video to be processed
#If we want live feedback we set the parameter zero as this=> cap = cv2.VideoCapture(0)
cap = cv2.VideoCapture('peopleCounter.avi')

#If we need to change the size of the video we set these parameters
##cap.set(3,160) #Width
##cap.set(4,120) #Height

flag = 1

def uploader():
   global cnt_down
   global flag
   while flag==1:
         #run every 3 seconds
         print("flag=",flag)
         time.sleep(3)
           #Ubidots variable
           response = outside_counter.save_value({"value": cnt_down})
                 response = inside_counter.save_value({"value": cnt_up})
           print("response from server=",response)
   thr.join()

#UBIDOTS
#ApiKey        "89010485c215b3f7c03dbdf8788485c758efe65f"
#Default Token  "kKSjP8a9FqyZksrBKFbLoKVYMZO76d"

api = ApiClient(token='kKSjP8a9FqyZksrBKFbLoKVYMZO76d')
outside_counter = api.get_variable('5b4e0be7c03f9754db6044e7')
```

```
inside_counter = api.get_variable('5b4e0c5ac03f97553f391e26')

#We are parsing through the video parameters
for i in range(19):
    print i, cap.get(i)

#We take the two parameters we need 3(width) 4(height)
w = cap.get(3)
h = cap.get(4)

#These is the video size
frameArea = h*w

#The area of doing the threshold transformations
areaTH = frameArea/250
print 'Area Threshold', areaTH

#Lines of entering and exiting to trigger count
line_up = int(2*(h/5))
line_down   = int(3*(h/5))

#Lines where the person leaves the area and we remove its ID
up_limit =   int(1*(h/5))
down_limit = int(4*(h/5))

#We make the points and draw the enter and exit lines
print "Red line y:",str(line_down)
print "Blue line y:", str(line_up)
line_down_color = (255,0,0)
line_up_color = (0,0,255)
pt1 =  [0, line_down];
```

```
pt2 =  [w, line_down];
pts_L1 = np.array([pt1,pt2], np.int32)
pts_L1 = pts_L1.reshape((-1,1,2))
pt3 =  [0, line_up];
pt4 =  [w, line_up];
pts_L2 = np.array([pt3,pt4], np.int32)
pts_L2 = pts_L2.reshape((-1,1,2))

#We make the points for the limit lines
pt5 =  [0, up_limit];
pt6 =  [w, up_limit];
pts_L3 = np.array([pt5,pt6], np.int32)
pts_L3 = pts_L3.reshape((-1,1,2))
pt7 =  [0, down_limit];
pt8 =  [w, down_limit];
pts_L4 = np.array([pt7,pt8], np.int32)
pts_L4 = pts_L4.reshape((-1,1,2))

#To make the calculations we remove the background and create an image
#where everything is black and white (to be counted as binary (0,1)
#we also remove peoples shadows

fgbg = cv2.createBackgroundSubtractorMOG2(detectShadows = True)

#We need to make morphological transformations for the Eroding and Dilating
#this is because after substracting background and shadows the image may seem
#bigger and we need to make it smaller

kernelOp = np.ones((3,3),np.uint8)
kernelOp2 = np.ones((5,5),np.uint8)
kernelCl = np.ones((11,11),np.uint8)
```

```
#We set the text font, persons array and parameters
font = cv2.FONT_HERSHEY_SIMPLEX
persons = []
max_p_age = 5
pid = 1

thr = threading.Thread(target=uploader, args=(), kwargs={})
thr.start()

while(cap.isOpened()):
#We open the video if there is still a frame to read it continues
   ret, frame = cap.read()

   for i in persons:
      i.age_one() #We need to mark every person we find on screen

   #We take a frame and subract its background
   fgmask = fgbg.apply(frame)
   fgmask2 = fgbg.apply(frame)

   #We make each image as binary and we will make the erode and dilating transformations
   try:
      ret,imBin= cv2.threshold(fgmask,200,255,cv2.THRESH_BINARY)
      ret,imBin2 = cv2.threshold(fgmask2,200,255,cv2.THRESH_BINARY)
      #We make the Opening (dilate) transformation to maintain the same person size after the substraction
      mask = cv2.morphologyEx(imBin, cv2.MORPH_OPEN, kernelOp)
      mask2 = cv2.morphologyEx(imBin2, cv2.MORPH_OPEN, kernelOp)
```

```
        #We make the Closing (erode) transformation to get a smaller/simpler person size after the substraction
        mask =  cv2.morphologyEx(mask , cv2.MORPH_CLOSE, kernelCl)
        mask2 = cv2.morphologyEx(mask2, cv2.MORPH_CLOSE, kernelCl)
    except:
        print('EOF')
        print 'UP:',cnt_up
        print 'DOWN:',cnt_down
        flag=0
        break
    # We make the surrounding green frame for each person we detect
    _, contours0, hierarchy = cv2.findContours(mask2,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)
    for cnt in contours0:
        area = cv2.contourArea(cnt)
        if area > areaTH:

            #Conditions for a person to be framed with green outline and be marked
            M = cv2.moments(cnt)
            cx = int(M['m10']/M['m00'])
            cy = int(M['m01']/M['m00'])
            x,y,w,h = cv2.boundingRect(cnt)

            new = True
            #We need to check if a person entered the limits of our lines to start tagging them
            if cy in range(up_limit,down_limit):
                for i in persons:
                    if abs(cx-i.getX()) <= w and abs(cy-i.getY()) <= h:
                        #We need to see if the person is close to one already detected
                        new = False
```

```
                i.updateCoords(cx,cy)   #we update persons coordinates
                #We check if the person goes up or down, we do that by checking which
line
                #it crossed first
                if i.going_UP(line_down,line_up) == True:
                    cnt_up += 1;
                    print "ID:",i.getId(),'crossed going up at',time.strftime("%c")
                elif i.going_DOWN(line_down,line_up) == True:
                    cnt_down += 1;
                    print "ID:",i.getId(),'crossed going down at',time.strftime("%c")
                break
            if i.getState() == '1':
                if i.getDir() == 'down' and i.getY() > down_limit:
                    i.setDone()
                elif i.getDir() == 'up' and i.getY() < up_limit:
                    i.setDone()
            if i.timedOut():
                #Remove persons id
                index = persons.index(i)
                persons.pop(index)
                del i     #Delete person from memory
        #If person wasn't detected before add its id
        if new == True:
            p = Person.MyPerson(pid,cx,cy, max_p_age)
            persons.append(p)
            pid += 1
    #We make the persons frame outline
    cv2.circle(frame,(cx,cy), 5, (0,0,255), -1)
    img = cv2.rectangle(frame,(x,y),(x+w,y+h),(0,255,0),2)
    #cv2.drawContours(frame, cnt, -1, (0,255,0), 3)
```

```
   #For debug reasons enable each person's tracking trail direction (this is disabled at
the moment)
   for i in persons:
##       if len(i.getTracks()) >= 2:
##           pts = np.array(i.getTracks(), np.int32)
##           pts = pts.reshape((-1,1,2))
##           frame = cv2.polylines(frame,[pts],False,i.getRGB())
##       if i.getId() == 9:
##           print str(i.getX()), ',', str(i.getY())
       cv2.putText(frame,
str(i.getId()),(i.getX(),i.getY()),font,0.3,i.getRGB(),1,cv2.LINE_AA)

#Putting text and drawing lines
   str_up = 'UP: '+ str(cnt_up)
   str_down = 'DOWN: '+ str(cnt_down)
   frame = cv2.polylines(frame,[pts_L1],False,line_down_color,thickness=2)
   frame = cv2.polylines(frame,[pts_L2],False,line_up_color,thickness=2)
   frame = cv2.polylines(frame,[pts_L3],False,(255,255,255),thickness=1)
   frame = cv2.polylines(frame,[pts_L4],False,(255,255,255),thickness=1)
   cv2.putText(frame, str_up ,(10,40),font,0.5,(255,255,255),2,cv2.LINE_AA)
   cv2.putText(frame, str_up ,(10,40),font,0.5,(0,0,255),1,cv2.LINE_AA)
   cv2.putText(frame, str_down ,(10,90),font,0.5,(255,255,255),2,cv2.LINE_AA)
   cv2.putText(frame, str_down ,(10,90),font,0.5,(255,0,0),1,cv2.LINE_AA)

   cv2.imshow('Frame',frame)
   #cv2.imshow('Mask',mask)
   #We hit escape on video to close it
   k = cv2.waitKey(30) & 0xff
   if k == 27:
      flag=0
      break
```

```
flag=0
#Free memory and close
cap.release()
cv2.destroyAllWindows()
```

4.9 Αποστολή δεδομένων στο Ubidots

Η τελευταία φάση της υλοποίησης του αλγορίθμου είναι η αποστολή των δεδομένων που συλλέχθηκαν στο σύννεφο (cloud). Στην συγκεκριμένη περίπτωση τα δεδομένα συλλογής είναι ο συνολικός αριθμός των ατόμων που κατευθύνθηκαν προς τα πάνω και προς τα κάτω κατά την διάρκεια του βίντεο εισαγωγής. Τα δεδομένα που συλλέγονται και απεικονίζονται στο Ubidots είναι οι δυο μετρητές (cnt_up και cnt_down) των ανθρώπων που διέσχισαν το σημείο καταμέτρησης. Τα δεδομένα αυτά μπορούν να επεξεργαστούν ανά πάσα στιγμή μέσω της διαδικτυακής πλατφόρμας Ubidots και να επεξεργαστούν για να μας δώσουν μια σημαντική εικόνα για την πληρότητα ενός χώρου μια συγκεκριμένη χρονική στιγμή.

Παρακάτω το κομμάτι του κώδικα αποστολής των δεδομένων και το περιβάλλον απεικόνισης (Ubidots) των αποτελεσμάτων που συλλέχθηκαν:

```
def uploader():
    global cnt_down
    global cnt_up
    global flag
    while flag==1:
            #run every 3 seconds
            print("flag==",flag)
            time.sleep(3)
        #Ubidots variable
        response = outside_counter.save_value({"value": cnt_down})
        response = inside_counter.save_value({"value": cnt_up})
        print("response from server=",response)
    thr.join()


#UBIDOTS
#ApiKey          "89010485c215b3f7c03dbdf8788485c758efe65f"
#Default Token   "kKSjP8a9FqyZksrBKFbLoKVYMZO76d"

api = ApiClient(token='kKSjP8a9FqyZksrBKFbLoKVYMZO76d')
outside_counter = api.get_variable('5b4e0be7c03f9754db6044e7')
inside_counter = api.get_variable('5b4e0c5ac03f97553f391e26')
```

Για την λειτουργία της πλατφόρμας Ubidots χρειάστηκε ένας λογαριασμός για να συνδεθεί και να ανεβούν τα δεδομένα. Η διαδικασία της αποστολής δεδομένων στο API του Ubidots γίνεται με το πρωτόκολλο HTTP (Το ίδιο πρωτόκολλο που χρησιμοποιούν οι browsers για να επικοινωνούν με τις ιστοσελίδες). Το Ubidots API υποστηρίζει τις παρακάτω μεθόδους:

POST – Δημιουργία δεδομένων στο cloud
GET – Ανάκτηση δεδομένων από το cloud
PUT – Επεξεργασία δεδομένων στο cloud
DELETE – Διαγραφή δεδομένων από το cloud

Ο λογαριασμός του Ubidots δημιουργεί ένα μοναδικό API key το οποίο χρησιμοποιείται για την επικοινωνία του κώδικα με την πλατφόρμα. Επίσης, δημιουργούνται μεταβλητές με ID τα οποία αντιστοιχίζονται μέσα στον κώδικα για να ενημερώνουν ανά πάσα στιγμή την τιμή τους.

Στην συνάρτηση uploader() χρησιμοποιείται ένα flag για να μπορεί να την θέτει σε λειτουργία. Το flag παίρνει την τιμή 1 κατά την εισαγωγή του βίντεο και την τιμή 0 όταν το video streaming τελειώσει. Η συνάρτηση στέλνει τα δεδομένα που συλλέχθηκαν κάθε 3 δευτερόλεπτα (time.sleep(3)) όσο η τιμή του flag είναι 1.

Τα δεδομένα ανανεώνονται και αναπαρίστανται στην πλατφόρμα του Ubidots κάθε 3 δευτερόλεπτα. Παρακάτω φαίνεται το Dashboard των μεταβλητών που ανανεώνονται κατά την πάροδο του χρόνου:

Εικόνα 36. *Κεντρική σελίδα διαδικτυακής πλατφόρμας Ubidots*

ΚΕΦΑΛΑΙΟ 5

ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΕΠΙΛΟΓΟΣ

Το προτεινόμενο σύστημα καταμέτρησης εφαρμόστηκε για να λειτουργεί σε περίπου 10 καρέ το δευτερόλεπτο και να χρησιμοποιεί RPi Camera Module V2. Κατά τα διάρκεια των πειραμάτων για να αποδειχθεί η απόδοση του προτεινόμενου συστήματος υπήρξαν αρκετά προβλήματα στην συμβατότητα των διαφορετικών εκδόσεων των προγραμμάτων που χρησιμοποιήθηκαν, στην εγκατάσταση του λειτουργικού καθώς και στην διαδικασία εκτέλεσης του αλγορίθμου καθώς η μαζική καταμέτρηση ατόμων δημιούργησε αδυναμία στην επεξεργασία των δεδομένων δεδομένου της υπολογιστικής δύναμης που προσφέρει ένα Raspberry Pi 3. Για να ξεπεραστούν αυτά τα προβλήματα λοιπόν, οι δοκιμές και τα πειράματα που διεξήχθησαν με διαφορετικές αλληλουχίες και παραμέτρους ήταν πολλές. Επετεύχθη η συμβατότητα των προγραμμάτων που χρησιμοποιήθηκαν με συγκεκριμένες εκδόσεις και αυξάνοντας την υπολογιστική ισχύ του RPi 3 σε ορισμένες περιπτώσεις με κίνδυνο να δημιουργηθεί πρόβλημα στην ορθή λειτουργία του. Επίσης, χρησιμοποιήθηκαν κάποιες παραδοχές και κάποιες σταθερές για να αποφευχθούν λάθη στην καταμέτρηση των ατόμων όπως αναφέρθηκε και σε προηγούμενη παράγραφο.

Στο προτεινόμενο σύστημα καταμέτρησης η κάμερα τοποθετήθηκε σε 3 μέτρα απόσταση από το έδαφος και κάθετη κλίση προς αυτό. Η εφαρμογή του συστήματος αυτού σε διαφορετική περιοχή απαιτεί περαιτέρω υπολογισμούς καθώς τα αποτελέσματα σε αντίθετη περίπτωση θα είναι λανθασμένα. Το πλαίσιο οριοθέτησης του αντικειμένου καθώς και η αποφυγή εισαγωγής αντικειμένων ταυτόχρονα και σε μικρή απόσταση μεταξύ θα πρέπει να θεωρούνται πάντα δεδομένα.

Συμπερασματικά, σε αυτή την εργασία, περιεγράφηκε και υλοποιήθηκε μια μέθοδος εφαρμογής για το σύστημα καταμέτρησης ατόμων, το οποίο ανιχνεύει τη μετακίνηση ανθρώπων χρησιμοποιώντας μια σταθερή κάμερα. Η πληροφορία αυτή καταγράφεται και επεξεργάζεται από έναν υπολογιστή χειρός (RPi 3) και αποστέλλεται σε μια διαδικτυακή πλατφόρμα για να γίνει η τελική απεικόνιση των αποτελεσμάτων, δίνοντας

την δυνατότητα πρόσβασης σε αυτά από οποιαδήποτε συσκευή με ίντερνετ. Με αυτό το τρόπο επιτεύχθηκε να υπάρχει άμεσα διαθέσιμη μια βασική πληροφορία όπως είναι η πληρότητα ενός συγκεκριμένου χώρου, η πορεία της κίνησης των ατόμων για λόγους μάρκετινγκ και όποια άλλη πληροφορία μας είναι χρήσιμη και μπορεί να συλλεχθεί από το σύστημα παρακολούθησης. Το προτεινόμενο σύστημα καταμέτρησης έχει πολλές εφαρμογές στην καθημερινότητα. Με την κατάλληλη παραμετροποίηση και την περαιτέρω ανάπτυξη μπορεί να δημιουργήσει μια νέα εποχή όπου το επίκεντρο της θα είναι η βελτιστοποίηση του τρόπου διαχείρισης του χρόνου στην καθημερινότητα.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] J. Segen and S. Pingali, “A Camera-Based System for Tracking People in Real Time,” *IEEE Proc. of Int. Conf. Pattern Recognition*, Vol.3, pp.63-67, 1996.

[2] O. Masoud and N. P. Papanikolopoulos, “A novel method for tracking and counting pedestrians in realtime using a single camera,” *IEEE Trans. on Vehicular Tech.*, Vol. 50, No. 5, pp.1267-1278, 2001.

[3] M. Rossi and A. Bozzoli, “Tracking and Counting Moving People,” *IEEE Proc. of Int.Conf. Image Processing*, Vol. 3, pp.212-216, 1994.

[4] K. Terada, D. Yoshida, S. Oe, J. Yamaguchi, “A counting method of the number of passing people using a stereo camera,” *IEEE Proc. of Industrial Electronics Conf.*, Vol. 3, pp.1318-1323, 1999.

[5] A. Baumberg and D. Hogg, “Learning flexible models from image sequences,” *Proc. Eur. Conf. Computer Vision*, vol. 1, Berlin, Germany, pp.229-308, 1994.

[6] C. Smith, C. Richards, S. A. Brandt and N. P. Papanikolopoulos, “Visual tracking for intelligent vehicle-highway systems,” *IEEE Trans. on Veh. Technol.*, Vol.45, pp.744-759, 1996.

[7] Q. Cai and J. K. Aggarwal, “Tracking human motion using multiple cameras,” *Proc. 13th Int. Conf. Pattern Recognition*, Los Alamitos, CA, pp.68-72, 1996.

[8] D. Murray and A. Basu, “Motion tracking with an active camera,” *IEEE Trans. on Pattern Analysis and Macine Intelligence*, Vol. 16, pp.449-459, 1994.

[9] C. Gu and M. Lee, “Semiautomatic segmentation and tracking of semantic video objects,” *IEEE Trans. on Circuits and Sytems for Video Technology*, Vol. 8, pp.572-584, 1998.

[10] M. K. Leung and Y. H. Yang, “First Sight: A Human Body Outline Labelling System,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol.17, no.4, pp.359-377, 1995.

[11] J. Rehg and T. Kanade, “Model-Based Tracking of Self-Occluding Articulated Objects,” *Proc. Int. Conf. Computer Vision*, pp.612-617, 1995.

[12] K. Rohr, “Towards Model-Based Recognition of Human Movements in Image Sequences,” *CVGIP: Image Understanding*, Vol.59, No.1, pp.94-115, 1994.

J. O’Rourke and N. J. Badler, “Model-based image analsys of human motion using constraint propagation,” *IEEE Trans. on Pattern Analysis and Machine Inteligence*, Vol.2, pp.522-536, 1980.

[13] Instructions to Install OpenCV 3.0 and Python 2.7+ on Ubuntu https://www.pyimagesearch.com/2015/06/22/install-opencv-3-0-and-python-2-7-on-ubuntu/

[14] Raspbian Stretch: Install OpenCV 3 + Python on your Raspberry Pi https://www.pyimagesearch.com/2017/09/04/raspbian-stretch-install-opencv-3-python-on-your-raspberry-pi/

[15] Person Counting System Using Opencv and Python https://www.hackster.io/deligence-technologies/person-counting-system-using-opencv-and-python-faf14f
[16] Install OpenCV and Python on your Raspberry Pi 2 and B+ https://www.pyimagesearch.com/2015/02/23/install-opencv-and-python-on-your-raspberry-pi-2-and-b/
[17] Install OpenCV 3.0 and Python 2.7+ on Ubuntu https://www.pyimagesearch.com/2015/06/22/install-opencv-3-0-and-python-2-7-on-ubuntu/
[18] Basic motion detection and tracking with Python and OpenCV https://www.pyimagesearch.com/2015/05/25/basic-motion-detection-and-tracking-with-python-and-opencv/
[19] Ubidots - REST API Reference https://ubidots.com/docs/api/#rest-api-reference
[20] Footfall: A Camera Based People Counting System https://blogs.wcode.org/2015/04/footfall-a-camera-based-people-counting-system-for-under-60/
[21] Accessing the Raspberry Pi Camera with OpenCV and Python https://www.pyimagesearch.com/2015/03/30/accessing-the-raspberry-pi-camera-with-opencv-and-python/
[22] Person Counting System Using Opencv and Python https://www.hackster.io/deligence-technologies/person-counting-system-using-opencv-and-python-faf14f
[23] Raspberry Pi Hardware Guide https://www.raspberrypi.org/learning/hardware-guide/
[24] Raspbian OS https://www.raspberrypi.org/downloads/raspbian/
[25] Installing OS Images https://www.raspberrypi.org/documentation/installation/installing-images/
[26] Getting started with picamera https://projects.raspberrypi.org/en/projects/getting-started-with-picamera
[27] Raspberry Pi Hardware Guide https://www.raspberrypi.org/learning/hardware-guide/
[28] Image Thresholding https://docs.opencv.org/trunk/d7/d4d/tutorial_py_thresholding.html
[29] Structural Analysis and Shape Descriptors https://docs.opencv.org/master/d3/dc0/group__imgproc__shape.html#ga17ed9f5d79ae97bd4c7cf18403e1689a&gsc.tab=0
[30] Contours: Getting Started https://docs.opencv.org/master/d4/d73/tutorial_py_contours_begin.html#gsc.tab=0
[31] Blob Detection Using OpenCV https://www.learnopencv.com/blob-detection-using-opencv-python-c/
[32] Recording video to a stream http://picamera.readthedocs.io/en/release-1.10/recipes1.html#recording-video-to-a-stream

[33] Morphological Transformations http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_morphological_ops/py_morphological_ops.html
[34] Eroding and Dilating https://docs.opencv.org/2.4/doc/tutorials/imgproc/erosion_dilatation/erosion_dilatation.html
[35] Raspberry Pi 3 Model B http://www.ardumotive.com/raspberrypigr.html
[36] Ubidots for Education Dashboard https://app.ubidots.com/ubi/insights/#/list