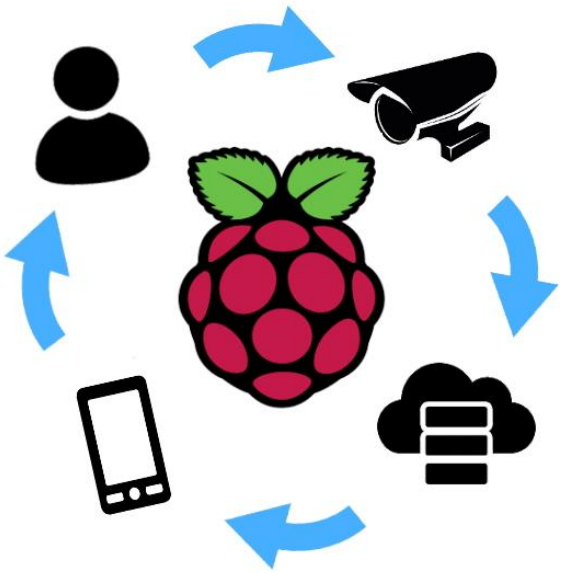
ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΣΕΡΡΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΕΠΙΚΟΙΝΩΝΙΩΝ



Υλοποίηση διαδικτυακής κάμερας παρακολούθησης με Raspberry Pi

Πτυχιακή εργασία του ΣΑΡΙΔΗ ΛΕΩΝΙΔΑ (3387)
Επιβλέπων: Δρ. Ι. ΚΑΛΟΜΟΙΡΟΣ

ΣΕΡΡΕΣ 2019

ΠΕΡΙΛΗΨΗ

Καθώς η ζωή μας εξελίσσεται μέσα στο πέρασμα των χρόνων, το ίδιο με ακόμη πιο ραγδαίους ρυθμούς κάνει και η τεχνολογία. Η εξέλιξη γεννά νέες ανάγκες οι οποίες με την σειρά τους γεννούν νέες τεχνολογίες ικανές να προσφέρουν πολλαπλούς τρόπους κάλυψης αυτών των αναγκών. Στον τομέα της επιστήμης της πληροφορικής βρίσκουν θέση αμέτρητες από αυτές τις ανάγκες διότι αυτός ο χώρος κατέχει πολύ μεγάλη ισχύ επίλυσης των προβλημάτων που προκύπτουν από αυτές. Η παρούσα πτυχιακή εργασία έχει σκοπό να αναδείξει αυτές τις τεχνολογίες αναλύοντας την έννοια του IoT (internet of things), της τεχνητής όρασης αλλά και της τεχνητής νοημοσύνης μέσω της βιβλιοθήκης OpenCV. Το μέσο εκπόνησης αυτής της εργασίας αποτελεί ο μικρουπλογιστής Raspberry Pi τον οποίο θα μελετήσουμε αναδεικνύοντας τις δυνατότητες του. Θα μελετήσουμε συστήματα παρακολούθησης μέσω ψηφιακών καμερών τα οποία έχουν κατασκευαστεί χρησιμοποιώντας ως βασική μονάδα ελέγχου το Raspberry Pi, με βασικό στόχο την υλοποίηση ενός παρόμοιου συστήματος.

Περιεχόμενα

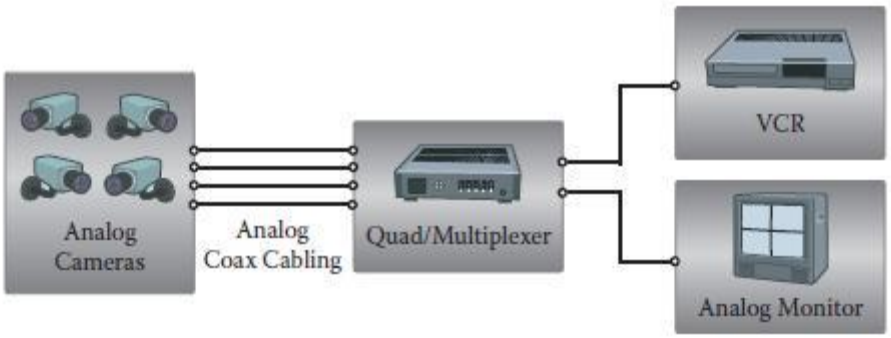
1. Εισαγωγικές έννοιες

1.1 Συστήματα παρακολούθησης

1.1.1 Αναλογικά συστήματα παρακολούθησης τύπου VCR

Τα πρώτα αναλογικά συστήματα παρακολούθησης CCTV (Closed Circuit TV) περιελάμβαναν την χρήση αναλογικών καμερών οι οποίες συνδέοταν σε ένα καταγραφικό VCR (videocassette recorder) για την καταγραφή των βίντεο που μαγνητοσκοπούσαν. Τα μέσα αποθήκευσης που χρησιμοποιούσαν ήταν κασσέτες κοινές με αυτές που προοριζόταν για οικιακή χρήση. Η κάθε κάμερα έπρεπε απαραιτήτως να συνδέεται με το καταγραφικό μέσω ξεχωριστού καλωδίου. Τα βίντεο δεν είχαν την δυνατότητα συμπίεσης και η μέγιστη διάρκεια καταγραφής υλικού σε μια κασσέτα ήταν μόλις οκτώ ώρες.

Για την βελτίωση της εργονομίας των συστημάτων ενσωματώθηκε μια μέθοδος καταγραφής time lapse ώστε να δημιουργηθεί η δυνατότητα καταγραφής υλικού για μεγαλύτερη διάρκεια. Η μέθοδος αυτή λειτουργούσε καταγράφοντας κάθε δεύτερη, τέταρτη, όγδοη ή δέκατη έκτη εικόνα. Αυτός ήταν ο λόγος που η βιομηχανία συστημάτων παρακολούθησης περιείχε χαρακτηριστικά όπως 15 fps (καρέ ανά δευτερόλεπτο), 7.5 fps, 3.75 fps, 1.875 fps, διότι αυτές ήταν οι μόνες δυνατές βαθμίδες καταγραφής καρέ στα αναλογικά συστήματα που χρησιμοποιούσαν την μέθοδο καταγραφής time lapse. Στις περιπτώσεις όπου χρησιμοποιούνταν πολλαπλές κάμερες, οι “τετράδες” ήταν άλλο ένα σημαντικό τμήμα του συστήματος. Μια τετράδα πολύ απλά μπορούσε να λάβει το υλικό από τέσσερις κάμερες και να παράγει ένα νέο βίντεο προβάλλοντας τέσσερις εικόνες σε μια οθόνη. Η λύση αυτή επέκτεινε την κλιμάκωση των συστημάτων παρακολούθησης διατηρώντας το κόστος σε χαμηλά επίπεδα.

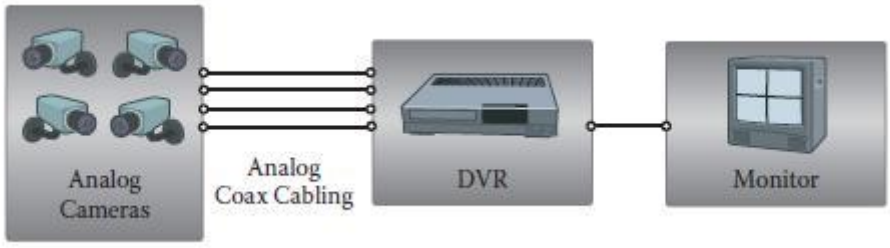


1 Αναλογικό σύστημα παρακολούθησης VCR

Σε ακόμη μεγαλύτερα συστήματα, χρησιμοποιούνταν πολυπλέκτες για τον συνδυασμό των βίντεο από πολλαπλές κάμερες που συχνά έφταναν τις 16 σε μια μόνο συσκευή καταγραφής. Οι πολυπλέκτες επίσης έδιναν την δυνατότητα δρομολόγησης της προβολής υλικού από μια κάμερα σε συγκεκριμμένες κατ επιλογή οθόνες στον χώρο εποπτείας των καμερών.

1.1.2 Αναλογικά συστήματα παρακολούθησης τύπου DVR

Στα μέσα της δεκατίας του 1990 η βιομηχανία των συστημάτων παρακολούθησης δέχτηκε την πρώτη της ψηφιακή εξέλιξη με την είσοδο των DVR (digital video recorder). Το DVR με την χρήση σκληρών δίσκων ως αποθηκευτικά μέσα αντικατέστησε το VCR ως μέσο καταγραφής. Τα βίντεο μετατρέποταν σε ψηφιακή μορφή και στην συνέχεια συμπιεζόταν ώστε να υπάρχει η δυνατότητα αποθήκευσης όσο το δυνατόν περισσότερων ημερών υλικού καταγραφής σε έναν σκληρό δίσκο.



2 Αναλογικό σύστημα παρακολούθησης DVR

Στις πρώτες εκδοχές των DVR, το μέγεθος των σκληρών δίσκων ήταν περιορισμένο, με αποτέλεσμα η διάρκεια καταγραφής ήταν και αυτή περιορισμένη με την σειρά. Για τον λόγο αυτό οι κατασκευαστές ανέπτυξαν ιδιωτικούς αλγορίθμους συμπίεσης. Παρότι ήταν αποτελεσματικοί, οι τελικοί χρήστες ήταν δεσμευμένοι να χρησιμοποιούν τον εξοπλισμό ενός συγκεκριμμένου κατασκευαστή όταν ήθελαν να επαναπροβάλλουν κάποιο βίντεο. Με το κόστος των σκληρών δίσκων να μειώνεται δραματικά και την δημιουργία κοινών αλγορίθμων συμπίεσης όπως το MPEG-4, οι κατασκευαστές εγκατέλειψαν την χρήση ιδιωτικών αλγορίθμων για την βελτίωση των προϊόντων που λάμβαναν οι καταναλωτές.

1.1.3 Ψηφιακά συστήματα παρακολούθησης

Η βασική διαφορά ανάμεσα στα ψηφιακά και τα αναλογικά συστήματα παρακολούθησης είναι πως τα ψηφιακά συστήματα παρακολούθησης έχουν την δυνατότητα καταγραφής και αποθήκευσης βίντεο σε ψηφιακή μορφή. Εφόσον τα δεδομένα καταγράφονται σε ψηφιακή μορφή εξαλείφεται η ανάγκη για οποιαδήποτε μετατροπή. Τα περισσότερα ψηφιακά συστήματα παρακολούθησης παρέχουν την δυνατότητα απομακρυσμένης διαχείρησης.

Τα πλεονεκτήματα των ψηφιακών συστημάτων παρακολούθησης σε σύγκριση με τα αναλογικά είναι πολλά. Ευκολότερη εγκατάσταση και λειτουργία, καλύτερη ποιότητα βίντεο σε πολύ μεγάλο βαθμό, επίσης σε σύγκριση με το αναλογικό σήμα, η ψηφιακή πληροφορία έχει την δυνατότητα ταχύτερης μετάδοσης. Επιπροσθέτως οι ψηφιακές κάμερες παρέχουν μεγαλύτερο εύρος γωνιών λήψης και τα βίντεο ή εικόνες που καταγράφουν χαρακτηρίζονται από καλύτερη ευκρίνια και σαφήνεια. Ακόμη το κόστος τους είναι μικρότερο από αυτό των αναλογικών συστημάτων. Τέλος τα ψηφιακά συστήματα παρακολούθησης διαθέτουν ενισχυμένα μέτρα ασφάλειας χρησιμοποιώντας τις τεχνολογίες της τεχνητής όρασης και της τεχνητής νοημοσύνης.

1.2 Internet of things

Το internet of things, ή IoT, είναι ένα σύστημα διασυνδεδεμένων υπολογιστικών, μηχανικών και ψηφιακών συσκευών, αντικειμένων, ζώων ή ανθρώπων που τους έχουν προσαρτηθεί αναγνωριστικές ταυτότες (UIDs) και έχουν την δυνατότητα να μεταφέρουν δεδομένα μέσω ενός δικτύου χωρίς να είναι απαραίτητη η αλληλεπίδραση ανθρώπου με άνθρωπο ή ανθρώπου με υπολογιστή.

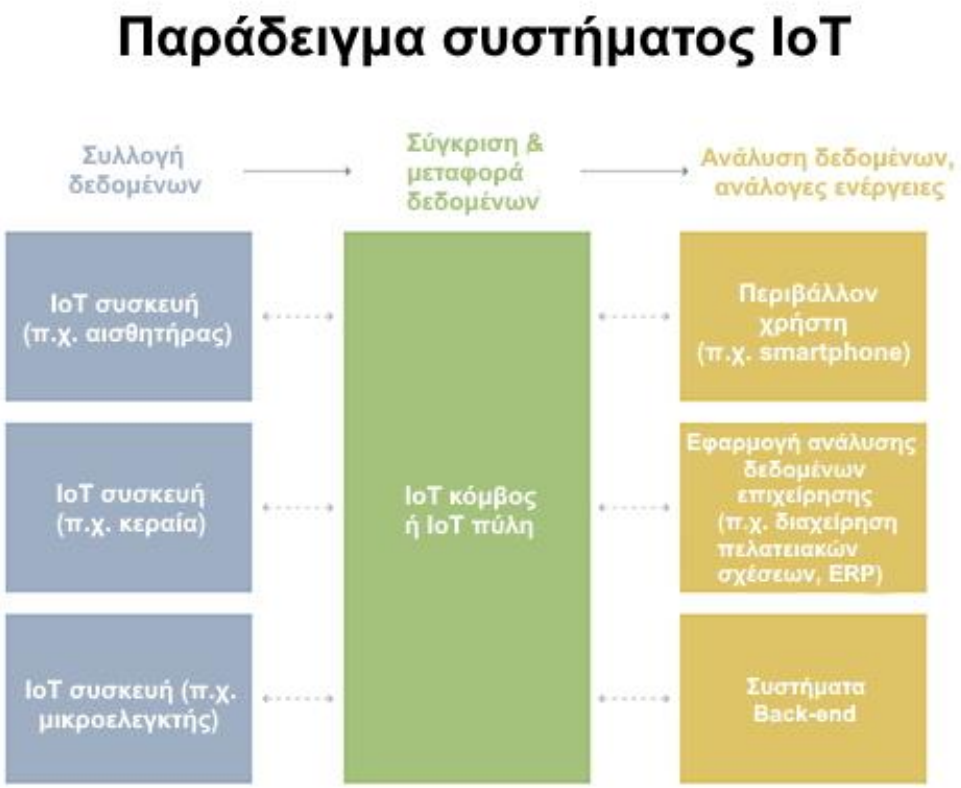
Ένα «πράγμα» στο internet of things μπορεί να είναι ένα άτομο με έναν καρδιακό βηματοδότη, μια φάρμα ζώων με πομπούς biochip (μικροτσιπ τα οποία προσδίδουν μοναδική ταυτότητα αναγνώρισης σε κάθε ζώο), ένα αυτοκίνητο με εγκατεστημένους αισθητήρες που προειδοποιούν τον οδηγό για την χαμηλή πίεση των ελαστικών, ή οποιοδήποτε φυσικό ή τεχνητό αντικείμενο στο οποίο μπορούμε να προσδώσουμε μια διεύθυνση IP και είναι ικανό να μεταφέρει δεδομένα σε ένα δίκτυο.

Όλο και περισσότεροι οργανισμοί σε διάφορες βιομηχανίες χρησιμοποιούν πλέον το IoT για να εξασφαλίσουν την αποτελεσματικότερη λειτουργία τους, αλλά και για να κατανοήσουν καλύτερα τις ανάγκες των πελατών τους ώστε να παρέχουν υπηρεσίες & προϊόντα που τους καλύπτουν. Επιπροσθέτως μέσω του IoT βελτιώνουν την ικανότητα λήψης αποφάσεων και αυξάνουν την αξία των επιχειρήσεών τους.

1.2.1 Πως λειτουργεί το internet of things;

Ένα οικοσύστημα του IoT προϋποθέτει την χρήση έξυπνων συσκευών συνδεδεμένων με το δίκτυο οι οποίες χρησιμοποιούν επεξεργαστές αισθητήρες και υλικό επικοινωνίας για να συλλέξουν, να επεξεργαστούν και να αποστείλουν δεδομένα. Οι IoT συκευές μοράζονται τα δεδομένα που συλλέγουν μέσω των αισθητήρων τους αφού συνδεθούν σε μια IoT πύλη ή αποστέλλουν τα δεδομένα στο cloud για ανάλυση ή γίνεται η ανάλυσή τους τοπικά. Μερικές φορές αυτές οι συσκευές επικοινωνούν με άλλες αντίστοιχων δυνατοτήτων και αλληλεπιδρούν μεταξύ τους βασιζόμενες στις πληροφορίες που ανταλάσσουν. Οι συσκευές αυτές εκτελούν το μεγαλύτερο τμήμα

εργασίας χωρίς την παρέμβαση ανθρώπινου παράγοντα παρόλο που ο τελευταίος μπορεί να αλληλεπιδράσει με τις συσκεύες, για να τις προγραμματίσει, να τους δώσει οδηγίες ή για να αποκτήσει πρόσβαση στα δεδομένα που περισυλλέγουν.



3 Σύστημα IoT

1.2.2 Καταναλωτικές και επαγγελματικές εφαρμογές IoT

Υπάρχουν αναρίθμητες εφαρμογές της τεχνολογίας του internet of things στον κόσμο, από καταναλωτικές και επαγγελματικές μέχρι κατασκευαστικές και βιομηχανικές στον χώρο της αυτοκίνησης των τηλεπικοινωνιών της ενέργειας και πολλών άλλων.

Στον καταναλωτικό τομέα για παράδειγμα, τα έξυπνα σπίτια, εξοπλισμένα με έξυπνους θερμοστάτες, έξυπνες οικιακές συσκευές, και διασυνδεδεμένα συστήματα θέρμανσης, φωτισμού καθώς και διάφορες

έξυπνες ηλεκτρονικές συσκευές μας παρέχουν την δυνατότητα του απομακρυσμένου ελέγχου τους μέσω υπολογιστών και smartphone.

Συσκευές που μπορούμε να φορέσουμε στο σώμα μας (wearables) μέσω αισθητήρων και λογισμικού συλλέγουν και αναλύουν δεδομένα από το σώμα του χρήστη και συγκρίνοντας τα με αυτά άλλων χρηστών στοχεύουν στο να κάνουν την ζωη τους πιο εύκολη και άνετη. Μπορούν επίσης να χρησιμοποιηθούν για την κοινή ασφάλεια, για παράδειγμα η άμεση ενημέρωση μέσω ενός ρολογιού σε κατάσταση εκτάκτου κινδύνου που μας υποδεικνύει τις ενδεδειγμένες διαδρομές προς ασφαλής τοποθεσίες.

Στον τομέα της υγείας και της περίθαλψης, το IoT προσφέρει πολλά πλεονεκτήματα όπως η δυνατότητα πιο στενής και άμεσης παρακολούθησης των ασθενών συλλέγοντας και αναλύοντας τα δεδομένα που παράγουν. Επίσης στα νοσοκομεία συχνά χρησιμοποιούνται συστήματα IoT για την έξυπνη διαχείρηση φαρμακευτικών αποθεμάτων, ιατρικών οργάνων.

4 Εφαρμογές συστημάτων IoT

Τα έξυπνα κτίρια μπορούν να ενισχύσουν την μείωση κατανάλωσης ενέργειας χρησιμοποιώντας αισθητήρες που ανιχνεύουν πόσοι άνθρωποι βρίσκονται σε ένα δωμάτιο. Βάση αυτού η θερμοκρασία του δωματίου ρυθμίζεται αναλόγως αυτόματα, παραδείγματος χάρη εάν ένα γραφείο είναι γεμάτο εργαζομένους το κλιματισικό ανοίγει αυτόματα και κλείνει μόλις όλοι έχουν φύγει όλοι για το σπίτι.

Στον τομέα της γεωργίας, συστήματα IoT ελέγχουν την ποσότητα φωτός, θερμοκρασίας, υγρασίας του άερα και του εδάφους στις εκτάσεις καλλιέργιας μέσω αισθητήρων. Ακόμη, αυτόματα συστήματα άρδευσης ενημερώνονται από τα παραπάνω δεδομένα και κρίνουν εάν είναι απαραίτητο ή όχι να εκκινήσουν την λειτουργία τους, εξοικονομώντας έτσι σημαντικές ποσότητες φυσικών πόρων κάνοντας την καλλιέργια πιο οικολογική, αλλά και μειώνοντας το κόστος της.

Σε μια έξυπνη πόλη, οι IoT αισθητήρες και κατασκευές, όπως οι έξυπνοι σηματοδότες κυκλοφορίας μπορούν να βοηθήσουν στην εξάλειψη της κυκλοφοριακής συμφόρησης, την εξοικονόμηση ενέργειας, τον έλεγχο περιβαλλοντικών φαινομένων αλλά και στην βελτίωση της περισυλλογής απορριμάτων με την χρήση έξυπνων κάδων.

1.3 Computer Vision – Τεχνητή όραση

Ή όραση μέσω υπολογιστών αφορά το επιστημονικό πεδίο το οποίο προσπαθεί να εξελίξει τεχνικές που βοηθουν τους υπολογιστές να αποκτήσουν ικανότητα όρασης έχοντας την δυνατότητα να αντιληφθούν το περιεχόμενων ψηφιακών στοιχείων όπως φωτογραφίες και βίντεο. Η τεχνητή όραση είναι συνδεδεμένη άμεσα με την τεχνητή νοημοσύνη. Από αυτήν δανείζεται εξειδεικευμένες μεθόδους και αλγόριθμους για την εξακρίβωση του περιεχομένου των εικόνων.

Ο στόχος της τεχνητής όρασης είναι η δυνατότητα αντίληψης του περιεχομένου ψηφιακών εικόνων. Αυτό περιλαμβάνει την ανάπτυξη μεθόδων

ικανές να αναπαράγουν αποτελέσματα αντίστοιχα με αυτά που επιτυγχάνει το ανθρώπινο σύστημα όρασης.

Η διαδικασία κατανόησης του περιεχομένου των εικόνων μπορεί να εμπεριέχει την εξαγωγή μιας περιγραφής από την εικόνα, η οποία μπορεί να αποτελείται από ένα αντικείμενο, μία γραπτή περιγραφή ή ένα τρισδιάστατο μοντέλο.

1.3.1. Τεχνητή όραση και επεξεργασία εικόνας

Η επεξεργασία εικόνας είναι η διαδικασία δημιουργίας μιας νέας εικόνας χρησιμοποιώντας ως βάση μια ήδη υπάρχουσα εικόνα απλουστεύοντας ή εμπλουτίζοντας το περιεχόμενό της. Η χρήση της σε συστήματα τεχνητής όρασης κρίνεται απαραίτητη για την άντληση πληροφοριών μέσα από τις εικόνες στην προσπάθεια εξακρίβωσης του περιεχομένου τους.

1.3.2. Η πρόκληση της τεχνητής όρασης

Αρχικά η λύση του προβλήματος της τεχνητής όρασης θεωρούνταν κάτι απλό και εύκολα επιλύσιμο. Μερικές δεκαετίες αργότερα το πρόβλημα παραμένει ακόμη άλυτο. Ένας από τους λόγους είναι οι περιορισμένες γνώσεις για τον τρόπο λειτουργίας της ανθρώπινης όρασης. Η μελέτη της βιολογικής όρασης απαιτεί πλήρη κατανόηση των τμημάτων της όπως το ανθρώπινο μάτι, καθώς και ο τρόπος με τον οποίο ερμηνεύει αυτά που λαμβάνει επικοινωνώντας με τον ανθρώπινο εγκέφαλο. Παρόλα αυτά έχει γίνει μεγάλη πρόοδος στον διάστημα των τελευταίων ετών, μετά από επιστημονικές μελέτες έχουν εξακριβωθεί κάποιοι από τους τρόπους λειτουργίας του ανθρώπινου συστήματος. Ένας ακόμης λόγος που η τεχνητή όραση αποτελεί πρόκληση είναι η πολυπλοκότητα από την οποία χαρακτηρίζεται ο φυσικός κόσμος. Ένα αντικείμενο είναι ορατό από πολλές οπτικές γωνίες, σε διαφορετικές συνθήκες φωτισμού. Ένα πραγματικό σύστημα όρασης πρέπει να είναι ικανό να δει και να αντιληφθεί το περιεχόμενο των εικόνων που λαμβάνει με επιτυχία, ανεξαρτήτως αυτών των παραγόντων.

1.3.3. Συστήματα τεχνητής όρασης

Όπως αναφέραμε, έχει γίνει πρόοδος τα τελευταία έτη έχοντας δημιουργήσει συστήματα τεχνητής όρασης τα οποία είναι ικανά να εκτελούν κάποιες εργασίες. Μερικά από αυτά τα συστήματα περιλαμβάνουν:

- Οπτική αναγνώριση χαρακτήρων, OCR (Optical character recognition)
- Επιθεώρηση μηχανών σε γραμμές παραγωγής
- Αυτοματοποιημένα ταμεία
- Τρισδιάστατη μοντελοποίηση κτιρίων
- Συστήματα ιατρικών διαγνώσεων
- Συτήματα ασφαλείας στην αυτοκίνηση
- Ανίχνευση κίνησης
- Συστήματα παρακολούθησης
- Αναγνώριση αποτυπωμάτων

.

1.4 Η αρχιτεκτονική του Raspberry Pi

1.4.1 Τι είναι το Raspberry Pi;

Το Raspberry Pi είναι ένας υπολογιστής σε μέγεθος πιστωτικής κάρτας ο οποίος συνδέεται στην τηλεόρασή μας ή οποιαδήποτε άλλη οθόνη μαζί με ένα πληκτρολόγιο και ένα ποντίκι. Το Raspberry Pi είναι η τρίτη κατά σειρά μεγαλύτερη σε πωλήσεις εταιρεία υπολογιστών παγκοσμίως. Μπορούμε να το χρησιμοποιήσουμε ως εκπαιδευτικό εργαλείο για την εκμάθηση γλωσσών προγραμματισμού καθώς και για την υλοποίηση ηλεκτρονικών συστημάτων, εφαρμογών αυτοματισμού αλλά και για όλες τις τυπικές εργασίες που θα κάναμε με έναν κοινό ηλεκτρονικό υπολογιστή, όπως η δημιουργία λογιστικών φύλλων και κειμένων, η περιήγηση στον παγκόσμιο ιστό αλλά και η χρήση του ως παιχνιδομηχανή για βιντεοπαιχνίδια. Επίσης μας παρέχει την δυνατότητα αναπαραγωγής βίντεο υψηλής ευκρίνειας. Το Raspberry Pi χρησιμοποιείτε από ενήλικους και παιδιά σε όλο τον κόσμο για την εκμάθηση

προγραμμάτισμου και την κατασκευή ψηφιακών συστημάτων. Σε συνέχεια των προηγουμένων μπορεί να χρησιμοποιηθεί σε εμπορικές και βιομηχανικές εφαρμογές. Το σχετικά μικρό του κόστος το καθιστά δημοφιλές και πολύ προσιτό σε όλους, καθώς η τιμή των βασικών του μοντέλων ξεκινά μόλις από τα 5€, με τις ναυαρχίδες της σειράς να ξεπερνούν σε ελάχιστο βαθμό τα 40€.

1.4.2 Τεχνικά χαρακτηριστικά

Τα Τεχνικά χαρακτηριστικά διαφέρουν μεταξύ των διαθέσιμων μοντέλων του. Τα διαθέσιμα αυτή την στιγμή είναι: Pi 4 Model B με διανομές διαφορετικών δυνατοτήτων, Pi 3 Model B+, Pi 3 Model B, the Pi 2 Model B, the Pi Zero, the Pi Zero W and the Pi 1 Model B+ and A+. Οι διαφορές μεταξύς τους αναλύονται στον παρακάτω πίνακα:

Product	SoC	Speed	RAM	USB Ports	Ethernet	Wireless	Bluetooth
Raspberry Pi Model A+	BCM2835	700MHz	512MB	1	No	No	No
Raspberry Pi Model B+	BCM2835	700MHz	512MB	4	100Base-T	No	No
Raspberry Pi 2 Model B	BCM2836/7	900MHz	1GB	4	100Base-T	No	No
Raspberry Pi 3 Model B	BCM2837A0/B0	1200MHz	1GB	4	100Base-T	802.11n	4.1
Raspberry Pi 3 Model A+	BCM2837B0	1400MHz	512MB	1	No	802.11ac/n	4.2
Raspberry Pi 3 Model B+	BCM2837B0	1400MHz	1GB	4	1000Base-T	802.11ac/n	4.2

Raspberry Pi 4 Model B	BCM2711	1500MHz	1GB	2xUSB2, 2xUSB3	1000Base-T	802.11ac/n	5.0
Raspberry Pi 4 Model B	BCM2711	1500MHz	2GB	2xUSB2, 2xUSB3	1000Base-T	802.11ac/n	5.0
Raspberry Pi 4 Model B	BCM2711	1500MHz	4GB	2xUSB2, 2xUSB3	1000Base-T	802.11ac/n	5.0
Raspberry Pi Zero	BCM2835	1000MHz	512MB	1	No	No	No
Raspberry Pi Zero W	BCM2835	1000MHz	512MB	1	No	802.11n	4.1
Raspberry Pi Zero WH	BCM2835	1000MHz	512MB	1	No	802.11n	4.1

Το Model A+ είναι η διανομή χαμηλού κόστους του Raspberry Pi. Διαθέτοντας 512MB RAM (από τον Αύγουστο του 2016 καθώς τα προηγούμενα μοντέλα υποστήριζαν 256MB), μια θύρα USB, 40 ακροδέκτες GPIO και δεν διαθέτει θύρα Ethernet. Το Model B+ ακολούθησε διαθέτωντας 512MB RAM, τέσσερις θύρες USB, 40 ακροδέκτες GPIO, και μια θύρα Ethernet.

Τον Φεβρουάριο του 2015, αντικαταστάθηκε από το Pi 2 Model B, την δεύτερη γενιά του Raspberry Pi. Το Pi 2 διαθέτει πολλά κοινά χαρακτηρισικά με το Pi 1 B+, και αρχικά χρησιμοποιούσε εναν τετραπύρηνο επεξεργαστή Arm Cortex-A7 ταχύτητας 900MHz και διέθεται 1GB μνήμης RAM. Κάποιες πρόσφατες εκδόσεις του Pi 2 (v1.2) χρησιμοποιούν επεξεργαστές Arm Cortex-A53 ταχύτητας 900MHz.

Το Pi 3 Model B βγήκε στην παραγωγή τον Φεβρουάριο του 2016. Χρησιμοποιεί έναν τετραπύρηνο επεξεργαστή Arm Cortex-A53 64-bit ταχύτητας 1.2GHz, διέθεται 1GB μνήμης RAM, ενσωματωμένο 802.11n ασύρματο LAN, και Bluetooth 4.1.

Το Pi 3 Model B+ εδόθηκε τον Μάρτιο του 2018. Χρησιμοποιεί έναν τετραπύρηνο επεξεργαστή Arm Cortex-A53 64-bit ταχύτητας 1.4GHz, διαθέτει 1GB μνήμης RAM, gigabit θύρα Ethernet, ενσωματωμένο 802.11ac/n ασύρματο LAN, Bluetooth 4.2.

Τα Pi Zero και Pi Zero W/WH είναι οι εκδόσεις μικρού μεγέθους ενός Model A+, διαθέτουν μονοπύρηνους επεξεργαστές ταχύτητας 1GHz, 512MB μνήμης RAM, θύρα mini-HDMI, θύρες USB On-The-Go καθώς και μία θύρα κάμερας. Το Pi Zero W επίσης διαθέτει ενσωματωμένη κεραία 802.11n ασύρματου LAN και Bluetooth 4.1. Το Pi Zero WH είναι πανομοιότυπο με το Zero W, αλλά διανέμεται με προεγκατεστημένους ακροδέκτες GPIO.

Τα Model A/A+ έχουν μία θύρα USB, το Model B διαθέτει δύο θύρες, και τα Model B+, Pi 2 Model B, Pi 3 Model B διαθέτουν τέσσερις θύρες οι οποίες μπορούν να χρησιμοποιηθούν για να συνδεθούν με τις περισσότερες συσκευές που χρησιμοποιούν USB 2.0. Επιπλέον συσκευές USB όπως ποντίκια, πληκτρολόγια, αντάπτορες δικτίου και εξωτερικής αποθήκευσης μπορούν να συνδεθούν μέσω USB hub. Το Pi Zero και Pi Zero W έχουν μία micro USB θύρα, η οποία απαιτεί ένα καλώδιο USB OTG ώστε να συνδεθούν συσκευές όπως πληκτρολόγια ή hubs.

Τον Ιούνιο του 2019 βγήκε στην παραγωγή το Rasperry Pi 4 αποτελόντας την ισχυρότερη μέχρι σήμερα έκδοσή του, χρησιμοποιώντας έναν τετραπύρηνο επεξεργαστή Arm Cortex-A72 με ταχύτητα 1.5 GHz και αρχιτεκτονική 64-bit. Διανέμεται σε τρεις διαφορετικές εκδόσεις 1 ram 1, 2 και 4GB αναλόγως με τις απαιτήσεις του χρήστη. Διαθέτει θύρα Ethernet, ενσωματωμένη κεραία 802.11ac/n και Bluetooth 5.0.

1.4.2.1 Ποιες είναι οι διαστάσεις του Raspberry Pi;

Οι εκδόσεις του Raspberry Pi Model B έχουν διαστάσεις 85.60mm x 56mm x 21mm, με την υποδοχή της κάρτας μνήμης και των υποδοχών συνδέσμων να προεξέχουν ελάχιστα. Ζυγίζουν μόλις 45 γραμμάρια. Τα Pi Zero και Pi Zero W έχουν διαστάσεις 65mm x 30mm x 5.4mm και ζυγίζουν 9 γραμμάρια.

1.4.2.2 Τι λειτουργικό σύστημα χρησιμοποιεί το raspberry pi;

Το raspberry pi δεν διανέμεται με προεγκατεστημένο λειτουργικό σύστημα, ωστόσο στην αγορά κυκλοφορούν πακέτα τα οποία συνοδεύονται από κάρτες μνήμης SD που εμπεριέχουν το NOOBS. Το παραπάνω ακρωνύμιο σημαίνει “New Out of the Box Software” και αποτελεί ένα βοηθητικό λογισμικό που καθιστά πολύ εύκολη την εγκατάσταση του λειτουργικού συστήματος που θα επιλέξουμε. Το λειτουργικό σύστημα του raspberry βασίζεται στο πανσίγνωστο και ευρέως διαδεδομένο λειτουργικό σύστημα Linux. Κυκλοφορούν αμέτρητες διαφορετικές διανομές, διαμορφωμένες κατάλληλα για το raspberry και η καθεμιά από αυτές βρίσκει εφαρμογή σε συγκεκριμμένες κατασκευές. Η επίσημη διανομή είναι το Raspbian το οποίο χρησιμοποιούμε και στην δική μας περίπτωση και αποτελεί το πληρέστερο πακέτο τόσο για την κάλυψη των βασικών υπολογιστικών αναγκών του μέσου χρήστη όσο και για την υλοποίηση εφαρμογών αυτοματισμού και IoT. Η δική μας εγκατάσταση δεν έγινε με την χρήση του NOOBS αλλά με την χειροκίνητη μέθοδο μέσω του Linux terminal για την απόκτηση της απαιτούμενης εξοικείωσης.

1.4.2.3 Με ποια οπτικά μέσα μπορεί να συνεργαστεί;

Το raspberry διαθέτει θύρα HDMI με την οποία μπορεί να συνδεθεί σε ψηφιακή τηλεόραση ή οθόνη μεταδίδοντας εικόνα και ήχο στην περίπτωση της τηλεόρασης ή συνδέοντας ηχεία στην 3.5mm έξοδο ήχου. Έχει την δυνατότητα να ενσωματώσει οθόνη αφής μέσω της θύρας DSI στην επίσημη εκδοχή οθόνης αφής, ή και μέσω των ακροδεκτών GPIO σε εκδοχές οθόνης αφής άλλων κατασκευαστών.

1.4.2.4 Ποιες είναι οι απαιτήσεις ηλεκτρικής ισχύος;

Η συσκευή τροφοδοτείται από ένα micro USB 5V. Η απαιτούμενη ισχύς λειτουργίας εξαρτάται από την έκδοση αλλά και τα περιφεριακά τα οποία συνδέονται στο raspberry. Ενδείκνυται η χρήση ενός τροφοδοτικού 2.5A

(2500mA) το οποίο είναι ικανό να καλύψει την απαιτούμενη ισχύ για την λειτουργία των τεσσάρων υποδοχών USB. Συσκευές USB με πολύ υψηλές απαιτήσεις ισχύος πιθανόν να χρειαστεί να λειτουργήσουν μέσω ενός διανομέα (USB hub) με εξωτερική πηγή τροφοδοσίας.

1.4.2.5 Λειτουργίες Wi-Fi και Bluetooth;

Μόνο τα μοντέλα Pi 3, 3+ και Pi Zero W διαθέτουν ενσωματωμένη ασύρματη συνδεσιμότητα. Όλα τα υπόλοιπα μοντέλα μπορούν να δεχτούν μια κεραία Wi-Fi μέσω των υποδοχών USB. Το Raspberry Pi Model 3B+ υποστηρίζει ασύρματη συνδεσιμότητα με το πρότυπο 802.11ac, ενώ όλα τα προηγούμενα υποστηρίζουν έως και το 802.11n. Τα μοντέλα Pi 3 Pi Zero W είναι τα μόνα που υποστηρίζουν συνδεσιμότητα Bluetooth.

1.4.3 Τι είναι το Camera Module;

Το Camera Module είναι μια μικρή PCB η οποία συνδέεται στην CSI-2 θύρα του Raspberry Pi. Παρέχει την δυνατότητα λήψης φωτογραφιών και καταγραφής βίντεο. Το Camera Module συνδέεται στο σύστημα μετάδοσης εικόνας ISP (Image System Pipeline) στο SoC (system on a chip) του Raspberry Pi, ώπου τα δεδομένα που εισέρχονται από την κάμερα επεξεργάζονται και μετατρέπονται σε αρχεία εικόνας ή βίντεο που αποθηκεύονται στην κάρτα μνήμης SD ή σε άλλα μέσα αποθήκευσης. Το Camera Module V2 χρησιμοποιεί την κάμερα Sony IMX219, ενώ ο πρόγονος της χρησιμοποιούσε την Omnivision OV5647. Οι δυο αυτές κάμερες είναι εφάμιλες με αυτές που χρησιμοποιούνται σε συσκευές κινητών τηλεφώνων.

1.5 Η γλώσσα προγραμματισμού Python

Η Python είναι μια διερμηνευόμενη, υψηλού επιπέδου γλώσσα με δυναμική σημασιολογία (semantics). Χαρακτηρίζεται από μια φιλοσοφία σχεδίασης η οποία δίνει έμφαση στην ευκολία ανάγνωσης του κώδικα. Προτιμάται ανάμεσα σε άλλες δημοφιλείς, όπως η Java ή η C++ καθώς το συντακτικό της είναι με τέτοιο τρόπο διαμορφωμένο, που επιτρέπει την ανάπτυξη κώδικα σε λιγότερες γραμμές.

Η Python δημιουργήθηκε και αναπτύχθηκε από τον Ολλανδό Guido van Rossum, το 1990 με την πρώτη της κυκλοφορία να γίνεται το 1991. Διαθέτει ένα μεγάλο αριθμό βιβλιοθηκών που διευκολύνουν πολλές συνηθισμένες εργασίες και χαρακτηρίζεται για την ταχύτητα εκμάθησής της. Αναπτύσσεται ως open source software (ανοιχτό λογισμικό) και η διαχείρισή της γίνεται από τον μη κερδοσκοπικό οργανισμό Python Software Foundation. Ακριβώς λόγω του ανοικτού της κώδικα, η Python λειτουργεί σε πολλές πλατφόρμες (φορητότητα).

Μερικά από τα βασικά χαρακτηριστικά και πλεονεκτήματά της είναι:

- *Εύκολη εκμάθηση*
- *Αναγνωσιμότητα*
- *Γρήγορη Ανάπτυξη Εφαρμογών*
- *Διερμηνευόμενη*
- *Επεκτάσιμη*
- *Ανοικτού Κώδικα*
- *Φορητότητα*

2. Ανάλυση συστημάτων παρακολούθησης με χρήση του Raspberry Pi

Μετά την εκπλήρωση των βασικών διαδικασιών, την εξοικείωση με το raspberry αλλά και την επίλυση προβλημάτων ερευνήσαμε τα ήδη υπάρχοντα συστήματα παρακολούθησης βασισμένα στο raspberry pi:

2.1 MotionEyeOS

Το MotionEyeOS είναι μια διανομή του Linux το οποίο μετατρέπει το raspberry pi σε μια μονάδα συστήματος παρακολούθησης. Η διαδικασία εγκατάστασης του είναι αρκετά εύκολη, κάτι που το κάνει ιδιαίτερα προσιτό και ελκυστικό στους μέσους χρήστες. Το περιβάλλον διαχείρησής του αποτελείται από έναν δικτυακό χώρο με φιλικό περιβάλλον προς τον χρήστη. Οι δυνατότητες του το κάνουν ένα ολοκληρωμένο σύστημα παρακολούθησης:

- *Ανίχνευση κίνησης & ειδοποιήσεις μέσω email*
- *Προγραμματισμός ωραρίου λειτουργίας*
- *Λήψη φωτογραφιών*
- *Αποθήκευση αρχείων σε αποθηκευτικό μέσό ή μεταφόρτωσή τους σε cloud*

Πλεονεκτήματα

- *Εύκολη εγκατάσταση και διαχείρηση*
- *Φιλικό προς τον χρήστη*
- *Δεν απαιτεί εξειδικευμένες προγραμματιστικές γνώσεις*

Μειονεκτήματα

- *Περιορισμένη παραμετροποίηση*
- *Αποτελεί μια έτοιμη λύση, δεν ενδείκνυται για ερευνητικούς σκοπούς*

2.2 Motion

Το Motion αποτελεί ένα λογισμικό με υψηλή δυνατότητα παραμετροποίησης, το οποίο ελέγχει και διαχειρίζεται την ροή βίντεο από κάμερες διαφόρων ειδών.

Πλεονεκτήματα

- *Δημιουργία βίντεο ή εικόνων κατά την ανίχνευση δραστηριότητας*
- *Ζωντανή ροή βίντεο από πολλαπλές κάμερες*

- *Εκκίνηση δευτερεύοντων script κατά την ανίχνευση συγκεκριμμένων συμβάντων*
- *Καταγραφή συμβάντων σε πολλούς τύπους βάσεων δεδομένων*
- *Πλήρως παραμετροποιήσιμες μάσκες για την λειτουργία ανίχνευσης κίνησης*

2.3 OpenCV

Η OpenCV (Open Source Computer Vision Library) είναι μια βιβλιοθήκη τεχνητής όρασης μέσω υπολογιστών ανοιχτού κώδικα. Η OpenCV δημιουργήθηκε για να παρέχει μια κοινή υποδομή για τις εφαρμογές τεχνητής όρασης υπολογιστών έτσι ώστε να επιταχύνει την ανάπτυξη της ικανότητας των υπολογιστικών συστημάτων να αντιλαμβάνονται γεγονότα και την ενσωμάτωση αυτών σε εμπορικά προϊόντα.

Η βιβλιοθήκη περιέχει περισσότερους από 2.500 βελτιστοποιημένους αλγορίθμους που σχετίζονται με την τεχνητή όραση μέσω υπολογιστών και την τεχνητή νοημοσύνη. Οι παραπάνω αλγόριθμοι μπορούν να χρησιμοποιηθούν για να ανιχνεύσουν και να αναγνωρίσουν πρόσωπα, να εξακριβώσουν αντικείμενα, να χαρακτηρίσουν ανθρώπινες ενέργειες μέσα σε βίντεο, να ακολουθήσουν την πορεία κίνησης αντικειμένων μέσω καμερών, να παράγουν 3D μοντέλα αντικειμένων, να συνδυάσουν μια ακολουθία εικόνων έτσι ώστε να συνθέσουν μια εικόνα υψηλής ανάλυσης μιας τοποθεσίας, να εντοπίσουν εικόνες μέσα από μια βάση δεδομένων εικόνων εκτελώντας την προσπέλασή της, να αφαιρέσουν το φαινόμενο των κόκκινων οφθαλμών από φωτογραφίες που έχουν ληφθεί με την χρήση φλας, να ακολοθυήσουν τις κινήσεις των οφθαλμών ατόμων, να αναγνωρίσουν τοπία και να καθορίσουν τα απαραίτητα στίγματα ώστε να επικαλύφθουν με σκηνικά εικονικής πραγματικότητας όπως συμβαίνει σε διαδραστικά εκπαιδευτικά εκθέματα. Η OpenCV διατηρεί μια κοινότητα χρηστών αποτελούμενη από 47.000 άτομα και έναν κατά προσέγγιση αριθμό λήψεων που ξεπερνά τα 18.000.000

λήψεις. Η βιβλιοθήκη χρησιμοποιείται εκτενώς σε εταιρείες, ομάδες ερευνών και κυβερνητικά σώματα.

Πάραληλλα με πανσίγνωστες εταιρείες όπως Google, η Yahoo, η Microsoft, η Intel, η IBM, η Sony, η Honda, η Toyota που χρησιμοποιούν την OpenCV, υπάρχουν πολλές εταιρείες startup όπως η Applied Minds, η VideoSurf, και η Zeitera, οι οποίες κάνουν εκτενή χρήση της OpenCV. Οι χρήσεις τις σε εφαρμογές ποικίλουν μέσα σε ένα πολύ μεγάλο εύρος, από τον συνδυασμό εικόνων του streetview για την δημιουργία πανοραμικών απεικονίσεων τοποθεσιών, την ανίχνευση εισβολών στο Ισραήλ μέσω συστημάτων παρακολούθησης βίντεο, την εποπτεία ρομπότ ανθρακορύχων σε ορυχεία της Κίνας, την ανίχνευση πνιγμών σε πισίνες της Ευρώπης, την στελέχωση εκθεμάτων διαδραστικής τέχνης στην Ισπανία και την Νέα Υόρκη έως και την ταχύτατη ανίχνευση προσώπων στην ασφυκτικά πυκνοκατοικημένη Ιαπωνία. Διαθέτει προγραμματιστικά περιβάλλοντα σε C++, Python, Java αλλά και στο MATLAB και υποστηρίζει Windows, Linux, Android και Mac OS.

Μέσω της OpenCV σε συνδυασμό με το Raspberry Pi και την γλώσσα προγραμματισμού Python μπορούμε να δημιουργήσουμε συστήματα παρακολούθησης με την δυνατότητα καταγραφής βίντεο, εικόνων αλλά και την ανίχνευση κινήσεων, προσώπων και ανθρωπίνων ενεργειών.

3. Υλοποίηση εφαρμογής παρακολούθησης

3.1 Προετοιμασία του Raspberry Pi

3.1.1 Εγκατάσταση λειτουργικού συστήματος

Για την εκκίνηση οποιασδήποτε διαδικασίας υλοποίησης εφαρμογών ή και γενικής χρήσης του μικροϋπολογιστή raspberry pi απαιτείται η εγκατάσταση ενός λειτουργικού συστήματος σε αυτό. Όπως αναφέραμε σε προηγούμενο τμήμα της παρούσας πτυχιακής εργασίας το επίσημο λειτουργικό σύστημα του raspberry pi είναι το raspbian. Στην περίπτωσή μας

επιλέξαμε το Raspbian STRETCH 9.6, στην έκδοση που υποστηρίζει γραφικό περιβάλλον (GUI) ώστε να μπορούμε να έχουμε ανά πάσα στιγμή πρόσβαση στην ζωντανή ροή βίντεο της pi camera για τον έλεγχο των δεδομένων και την δημιουργία συμπερασμάτων. Πραγματοποιούμε την λήψη του raspbian από την επίσημη σελίδα του raspberry και το μεταφέραμε σε αυτό μέσω μιας κάρτας μνήμης micro SD. Η συνέχεια της διαδικασίας εγκατάστασης είναι πολύ απλή καθώς εκκινεί αυτόματα κατά την διάρκεια της πρώτης εκίνησης του raspberry pi με την συγκεκριμμένη κάρτα μνήμης. Με την ολοκλήρωση της εγκατάστασης έχουμε πρόσβαση στο γραφικό περιβάλλον του raspberry pi μέσα από το οποίο μπορούμε να κάνουμε τις τυπικές εργασιές που θα κάναμε σε έναν κλασσικό ηλεκτρονικό υπολογιστή ή στην περίπτωση μας να ξεκινήσουμε την υλοποίηση εφαρμογών.

3.1.2 Εγκατάσταση του περιφεριακού Pi Camera Module V2

Καθώς η πλακέτα του περιφεριακού της Pi Camera Module V2 είναι πολύ ευαίσθητη στον στατικό ηλεκτρισμό χρησιμοποιήσαμε τον απαιτούμενο εξοπλισμό για την εξάλειψη κάθε πιθανού ηλεκτρικού στοιχείου πριν έρθουμε σε επαφή με αυτήν. Στην συνέχεια συνδέουμε την κάμερα στο raspberry pi μέσω της θύρας CSI (Camera Serial Interface) και έχουμε ολοκληρώσει την διαδικασία εγκατάστασης σε επίπεδο υλικού. Εκκινώντας το raspberry εκτελούμε τις απαραίτητες εντολές για την εγκατάσταση του απαραίτητου λογισμικού για την λειτουργία της Pi Camera και τέλος την ενεργοποιούμε από την λίστα περιφεριακών της διαχείρησης του raspberry (raspi-config).

Δυστυχώς στην αρχική περίοδο εκπόνησης της πτυχιακής εργασίας είχαμε την ατυχία να παραλάβουμε επί σειρά δύο ελαττωματικές επίσημες κάμερες, κάτι το οποίο σε συνδυασμό με την έλειψη εμπειρίας, μας προβλημάτισε αρκετά δαπανώντας μεγάλο χρονικό διάστημα στην προσπάθεια εύρεσης πιθανών σφαλμάτων στις εργασίες που είχαν εκτελεστεί. Η συνεχής επικοινωνία με τον προμηθευτή δεν οδήγησε σε κάποια λύση. Η λύση δώθηκε όταν έγινε η αλλαγή προμηθευτή όπου πλέον είχαμε στα χέρια μας ένα εγγυημένο προϊόν το οποίο λειτούργησε όπως προοριζόταν από την πρώτη στιγμή, αποδεικνύοντας ότι οι ενέργειες που

είχαμε κάνει κατά την εγκατάσταση τόσο του λειτουργικού συστήματος όσο και της Camera Module V2 ήταν έγκυρες. Ωστόσο το παραπάνω πρόβλημα μας έδωσε την δυνατότητα να αντιμετωπίσουμε άμεσα τα πιθανά πραγματικά προβλήματα που προκύπτουν κατά την εξέλιξη και υλοποίηση εφαρμογών, χωρίς αυτά να αφορούν αποκλειστικά το ερευνητικό και προγραμματιστικό τμήμα του, αλλά την εμπλοκή εξωγενών παραγόντων.

3.2 Υλοποίηση συστήματος παρακολούθησης με χρήση της OpenCV

Στην περίπτωσή μας επιλέξαμε να υλοποιήσουμε ένα σύστημα παρακολούθησης βασισμένο στην OpenCV, λόγω των τεράστιων δυνατοτήτων που μας παρέχει για την δημιουργία εφαρμογών τεχνητής όρασης και για τις δυνατότητες ανίχνευσης κίνησης που κάνουν ένα σύστημα ασφαλείας πιο ασφαλές, εύστοχο, και αποτελεσματικό.

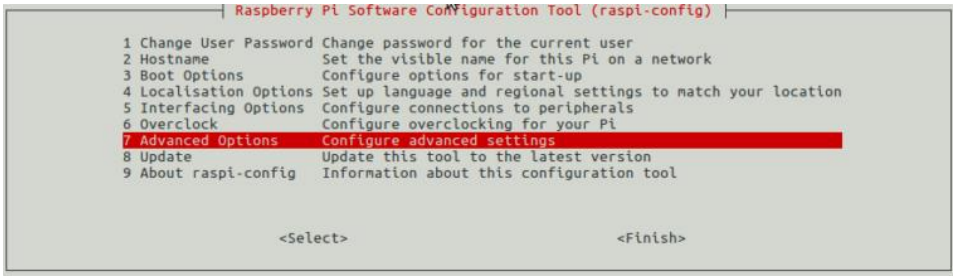
3.2.1 Εγκατάσταση OpenCV στο Raspberry pi

Το πρώτο μας βήμα για την εκκίνηση της διαδικασίας εγκατάστασης της OpenCV στο raspberry pi πρέπει να είναι η επέκταση του συστήματος αρχείων διότι το raspbian έχει καταλάβει μόνο τον απαιτούμενο χώρο στην κάρτα μνήμης κατά την εγκατάσταση του. Για τον λόγο αυτό εκμεταλλευόμαστε όλο τον εναπομείναντα κενό χώρο της κάρτας μνήμης για την επέκταση του συστήματος αρχείων έτσι ώστε να είναι επαρκής για την εγκατάσταση της OpenCV αλλά και για την μελλοντική υλοποίηση άλλων εφαρμογών.

Για την επέκταση αποθηκευτικού χώρου μνήμης του συστήματος αρχείων εκκινούμε το σύστημα διαχείρησης του raspberry μέσω της γραμμής εντολών χρησιμοποιώντας την εντολή:

sudo raspi-config

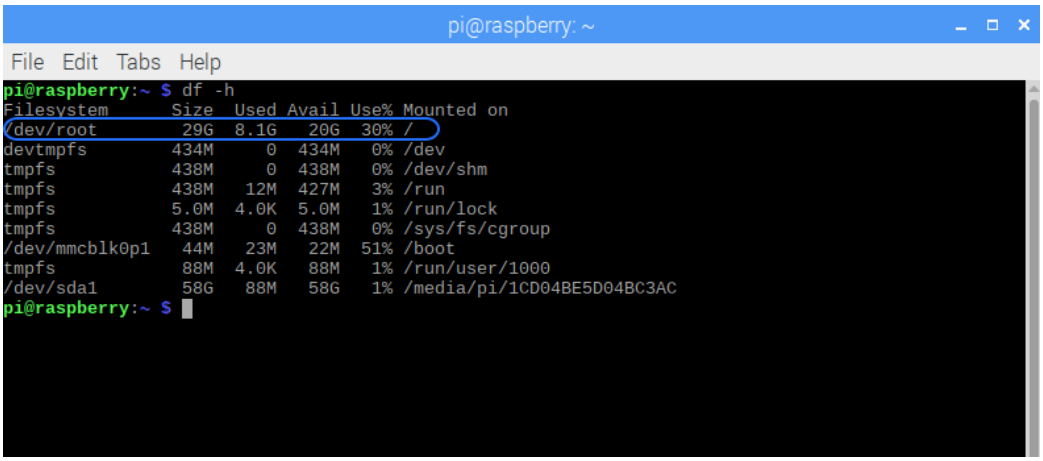
Στην συνέχεια επιλέγουμε την επιλογή “Advanced options” και “Expand filesystem” από τα σχετικά μενού. Δίνουμε την εντολή επανεκκίνησης του raspberry: *sudo reboot*



5 Raspberry Pi configuration menu

Για την επιβεβαίωση της παραπάνω διαδικασίας μετά την επανεκκίνηση χρησιμοποιούμε την εντολή:

df –h



6 Αποτελέσματα εντολής df -h

Στην συνέχεια ενημερώνουμε και αναβαθμίζουμε τα όποια ήδη εγκατεστημένα πακέτα χρημοποιώντας τις εντολές:

sudo apt-get update
sudo apt-get upgrade

Κάνουμε εγκατάσταση του αναπτυξιακού εργαλείου CMake το οποίο θα μας βοηθήσει να επιβεβαιώσουμε την ομαλή λειτουργία και σωστή εγκατάσταση της OpenCV. Το CMake μας παρέχει τις απαραίτητες πληροφορίες καθόλη την διάρκεια της μεταγλώτισης των πακέτων. Με την μεταγλώτιση της OpenCV να αποτελεί μια πολύωρη διαδικασία η χρήση του κατά την έρευνα μας αποδείχτηκε πολύ χρήσιμη.

sudo apt-get install build-essential cmake pkg-config

Κατά την διάρκεια της έρευνάς μας ανακλύψαμε πως για την ομαλή και πληρέστερη λειτουργία της βιβλιοθήκης OpenCV στο raspberry pi απαιτείται η εγκατάσταση κάποιων βοηθητικών πακέτων και βιβλιοθηκών έτσι ώστε να μπορούμε να φορτώσουμε αρχεία εικόνων και βίντεο από τον αποθηκευτικό μας χώρο αλλά και να εξάγουμε από την OpenCV αποτελέσματα των ενεργειών μας σε οπτικά μέσα όπως μια οθόνη. Συνοπτικά η εγκατάστασή τους:

sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev
sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev
sudo apt-get install libxvidcore-dev libx264-dev
sudo apt-get install libgtk2.0-dev libgtk-3-dev

Απαραίτητη προϋπόθεση για την υλοποίηση εφαρμογών με την χρήση της OpenCV είναι η εγκατάστηση της Python:

sudo apt-get install python3-dev

Αφού έχουμε εγκαταστήσει όλα τα απαραίτητα πακέτα κάνουμε λήψη της πιο πρόσφατης έκδοσης της OpenCV, στην περίπτωσή μας ήταν η 4.0.1.

wget -O opencv.zip https://github.com/Itseez/opencv/archive/4.0.1.zip
unzip opencv.zip

wget -O opencv_contrib.zip
https://github.com/Itseez/opencv_contrib/archive/4.0.1.zip
unzip opencv_contrib.zip

Πρωτού προχωρήσουμε στην μεταγλώτιση της OpenCV εγκαθιστούμε το σύστημα διαχείρησης πακέτων της Python, **pip**.

wget https://bootstrap.pypa.io/get-pip.py
sudo python get-pip.py
sudo python3 get-pip.py

Στην συνέχεια εγκαθιστούμε τα πακέτα **virtualenv** και **virtualenvwrapper** τα οποία μας επιτρέπουν την δημιουργία εικονικών περιβαλλόντων εργασίας της Python και την εύκολη διαχείρησή τους (προσθήκη νέων, κατάργηση)

αντίστοιχα. Ο λόγος που εγκαθιστούμε αυτά τα πακέτα είναι για την διευκόλυνσή μας κατά την διάρκεια της έρευνας σε περίπτωση που υπάρξει η ανάγκη για τον διαχωρισμό συγκεκριμμένων πακέτων για την υλοποίηση του συστήματος παρακολούθησης με διαφορετικούς τρόπους ή για την χρήση σε μελλοντικές εφαρμογές. Το πλεονέκτημα που μας προσφέρουν είναι η εξάλειψη πιθανών αντικρούσεων και δυσλειτουργιών μεταξύ διαφόρων βοηθητικών πακέτων της Python.

```
sudo pip install virtualenv virtualenvwrapper
```

Εφόσον έχουμε εγκαταστήσει τα παραπάνω πακέτα ανανεώνουμε το αρχείο **~/.profile** ώστε να είναι δυνατή η χρήση τους και να μας επιτρέπεται η δημιουργία και χρήση εικονικών περιβαλλόντων. Χρησιμοποιούμε τον text editor που προτιμούμε για την τροποποίηση του αρχείου, στην περίπτωση μας χρησιμοποιήσαμε τον nano editor

```
sudo nano ~/.profile
```

Προσθέτουμε στο αρχείο:

```
export WORKON_HOME=$HOME/.virtualenvs
export VIRTUALENVWRAPPER_PYTHON=/usr/bin/python3
source /usr/local/bin/virtualenvwrapper.sh
```

Πλέον είμαστε σε θέση να δημιουργήσουμε το εικονικό περιβάλλον που επιθυμούμε.

```
mkvirtualenv cv -p python3
```

Μεταφερόμαστε στο εικονικό περιβάλλον cv με τις εντολές **source ~/.profile** και **workon cv** και δημιουργούμε τον απαραίτητο κατάλογο για την διαδικασία μεταγλώτισης της OpenCV χρησιμοποιώντας το εργαλείο διαχείρησης CMake:

```
cd ~/opencv-3.3.0/
 mkdir build
 cd build
 cmake -D CMAKE_BUILD_TYPE=RELEASE \
```

*-D CMAKE_INSTALL_PREFIX=/usr/local *
*-D INSTALL_PYTHON_EXAMPLES=ON *
*-D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib-3.3.0/modules *
-D BUILD_EXAMPLES=ON

Πρωτού εκκινήσουμε την διαδικασία μεταγλώτισης κρίνεται απαραίτητο να μετατρέψουμε το swap size της κάρτας μνήμης SD ώστε να μην υποπέσουμε σε έλλειψη μνήμης κατά την διάρκεια μεταγλώτισης και να δώσουμε την δυνατότητα στο raspberry να εκτελέσει την διεργασία κάνοντας χρήση και των τεσσάρων πυρήνων της κεντρικής μονάδας επεξεργασίας του. Με την χρήση του nano editor και πάλι:

sudo nano /etc/dphys-swapfile

Στην συνέχεια κάνουμε αλλαγή της μεταβλητής **CONF_SWAPSIZE** από 100 σε 1024 MB. Για την ενεργοποίηση της αλλαγής απαιτείται η επανεκκίνηση της διεργασίας swap:

sudo /etc/init.d/dphys-swapfile stop
sudo /etc/init.d/dphys-swapfile start

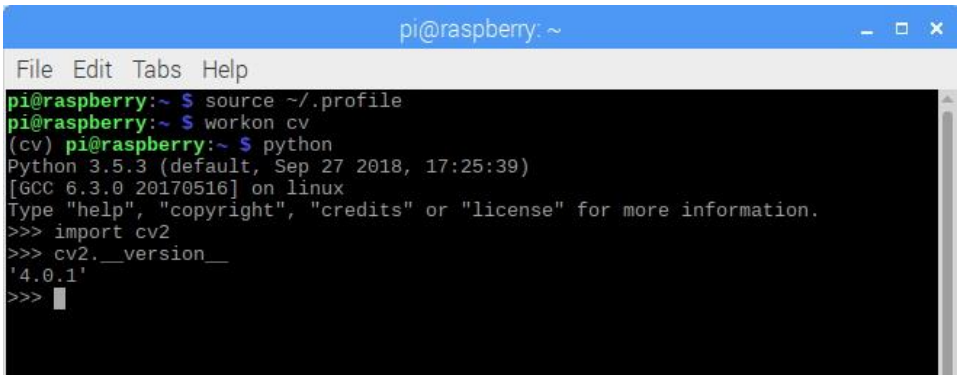
Εκκινούμε την χρονοβόρα διαδικασία μεταγλώτισης της OpenCV.

make -j4

Μετά την ολοκλήρωσή της προχωράμε στην διαδικασία εγκατάστασης.

sudo make install
sudo ldconfig

Για την επικύρωση της εγκατάστασης μπορούμε να τρέξουμε ένα μικρό κομμάτι κώδικα της Python για να μας εμφανίση την έκδοση της OpenCV:



7 Έλεγχος ορθής εγκατάστασης OpenCV

3.2.2 Ανάλυση απαιτήσεων συστήματος

Εφόσον έχουμε εγκαταστήσει την κύρια βιβλιοθήκη που θα χρησιμοποιεί το σύστημά μας αναλύουμε της απαιτήσεις του. Το σύστημα μας πρέπει να διαθέτει την ικανότητα καταγραφής βίντεο μέσω του Camera Module V2 του Raspberry Pi διαχωρίζοντάς τα σε επιμέρους τμήματα βάση ενός χρονικά προκαθορισμένου ορίου και παράλληλα να επικοινωνεί με έναν διαδικτυακό χώρο αποθήκευσης cloud για να μεταφορτώνει αυτά τα αρχεία. Ο χρήστης θα έχει την δυνατότητα πρόσβασης στον χώρο για προβολή και λήψη των παραπάνω βίντεο.

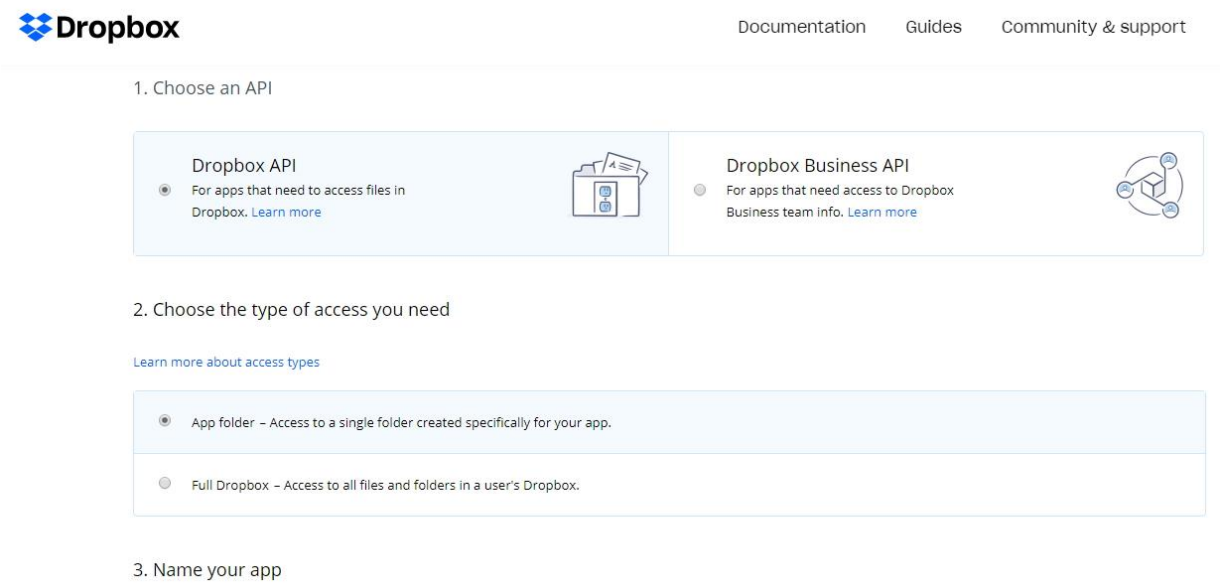
3.2.3 Χρήση cloud υπηρεσιών για μεταφόρτωση αρχείων

Οι υπηρεσίες cloud storage είναι το προϊόν ενός υπολογιστικού μοντέλου κατά το οποίο τα δεδομένα των χρηστών αποθηκεύονται σε απομακρυσμένους server με διαδικτυακή δυνατότητα πρόσβασης. Η συντήρηση, λειτουργία και η διαχείρηση των οποίων γίνεται από τον πάροχο της υπηρεσίας.

Το Cloud storage στηρίζεται στην εικονικοποίηση (virtualization) των κέντρων δεδομένων, παρέχοντας στους τελικούς χρήστες και τις εφαρμογές έναν εικονικό χώρο αποθήκευσης ο οποίος μπορεί να κλιμακώνεται ανάλογα με τις ανάγκες της κάθες εφαρμογής. Λειτουργεί μέσω διαδικτυακών κλειδιών API (Application Programming Interface) τα οποία συμπεριλαμβάνονται στον κώδικα των εφαρμογών ώστε να στελεχώσουν την επικοινωνία αυτών με τους server της υπηρεσίας παρέχοντας σε αυτές την δυνατότητα εισαγωγής, εξαγωγής, ανάγνωσης και καταγραφής δεδομένων.

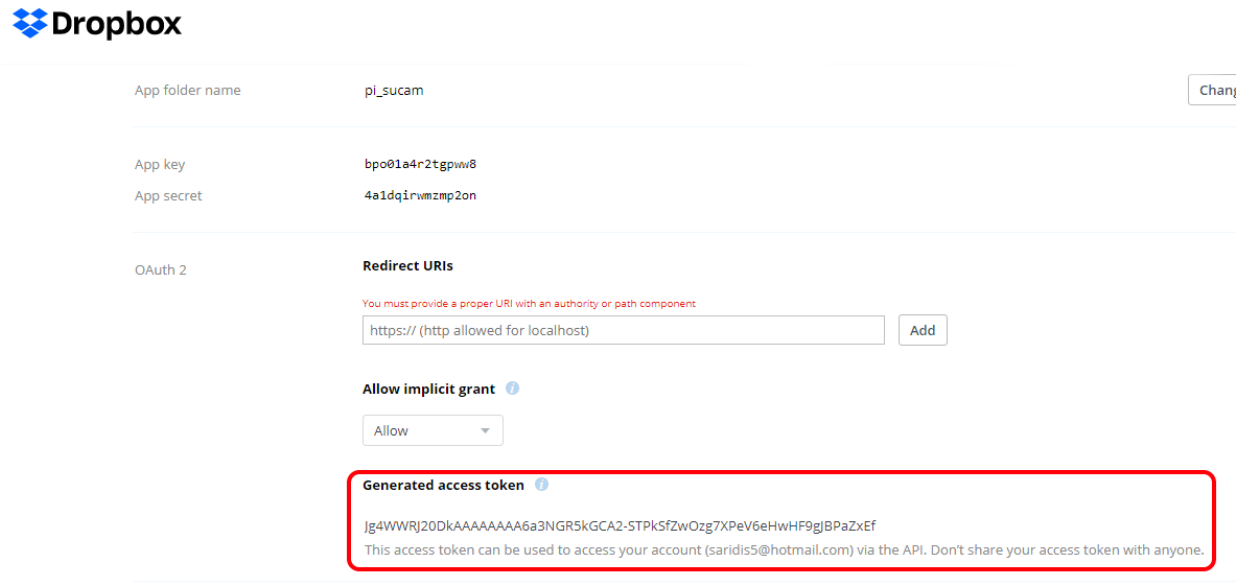
Για τον παραπάνω λόγο ανατρέξαμε σε αντίστοιχες υπηρεσίες cloud storage. Στην αρχή της έρευνάς μας εργαστήκαμε χρησιμοποιώντας την υπηρεσία AWS της Amazon η οποία παρότι αποτελεσματική επιβάλλει μια δοκιμαστκή περίοδο στην διάρκεια χρήσης της. Σε προσπάθειά μας να επιλύσουμε το παραπάνω πρόβλημα μελετήσαμε την επιλογή της υπηρεσίας cloud storage του dropbox. Η διαδικασία ενσωμάτωσης του dropbox στην εφαρμογή μας ήταν πολύ απλή, απαιτείται η δημιουργία λογαριασμού στην

ιστοσελίδα της υπηρεσίας και στην συνέχεια δημιουργία συγκεκριμμένου τμήματος αποθηκευτικού χώρου για την εφαρμογή μας:



8 Δημιουργία αποθηκευτικού χώρου εφαρμογής στο dropbox

Τέλος δημιουργούμε τον απαραίτητο σύνδεσμο πρόσβασης που θα χρησιμοποιήσουμε στον κώδικα της εφαρμογής μας (API):



9 Δημιουργία συνδέσμου API token για την διασύνδεση του αποθηκευτικού χώρου με την εφαρμογή

3.2.4 Συγγραφή κώδικα της εφαρμογής

Ξεκινάμε την συγγραφή κώδικα εισάγοντας όλες τις απαραίτητες βιβλιοθήκες για την λειτουργία του συστήματός μας:

```
from imutils.video import VideoStream
from imutils.io import TempFile
from picamera import PiCamera
from datetime import datetime
from datetime import date
import imutils
import time
import cv2
import dropbox
```

Εκκινούμε την ροή βίντεο χρησιμοποιώντας την **VideoStream()** ορίζοντας ως μέσω καταγραφής το Camera Module, και δίνουμε λίγο χρόνο στον αισθητήρα της κάμερας για να προθερμανθεί. Έπειτα ορίζουμε την σύνδεση με τον διακομιστή του cloud server που θα χρησιμοποιήσουμε, στο σύστημα μας επιλέξαμε το dropbox για τους λόγους που προαναφέραμε.

Αρχικοποιούμε την μεταβλητή **writer** στην οποία θα κάνουμε την καταγραφή βίντεο καθώς και τις μεταβλητές στις οποίες θα ορίσουμε τις διαστάσεις του βίντεο στην συνέχεια. Ορίζουμε την μεταβλητή **startTime** ως ώρα εκκίνησης του συστήματος μέσω της **datetime.now()** και την τυπώνουμε στην γραμμή εντολών.

```
vs = VideoStream(usePiCamera=True).start()
print("[INFO] warming up camera...")
time.sleep(2.0)

client =
dropbox.Dropbox("Jg4WWRJ20DkAAAAAAAA5pDWwVNmfC8LBRDN4BsjZQ6y-
YRRsanF6FRhw-KHtRNT6")
print("[SUCCESS] dropbox account linked")

writer = None
W = None
H = None

startTime = datetime.now()
print("[RECORDING STARTED]")
print(startTime)
```

Εκκινούμε έναν ατέρμονα βρόγχο με την χρήση της while ο οποίος θα εκτελεί την διαδικασία καταγραφής και μεταφόρτωσης του βίντεο έως ώτου διακοπεί μόνο με εντολή του χρήστη. Ορίζουμε την μεταβλητή **frame** για την εκχώρηση της ροής βίντεο από την κάμερα μας, αλλάζουμε την διάστασή του στην επιθυμητή ανάλυση. Για τον ορισμό των διαστάσεων W και H (πλάτος και ύψος) της καταγραφής χρησιμοποιούμε την **img.shape()**, η οποία μας επιστρέφει των αριθμό γραμμών, στηλών και επιπέδων σε περίπτωση έγχρωμης εικόνας. Χρησιμοποιώντας τους κατάλληλους δείκτες για τις επιστρεφόμενες τιμές εκχωρούμε τις τιμές ύψους και πλάτους στις αντίστοιχες μεταβλητές.

```
while True:

    frame = vs.read()
    frame = imutils.resize(frame, width=480)

    if W is None or H is None:

        H = frame.shape[0]
        W = frame.shape[1]
```

Ελέγχουμε εάν η μεταβλητή writer είναι κενή, εφόσον είναι, δημιουργούμε την μεταβλητή **tempVideo** με την βοήθεια της **TempFile** που μας επιτρέπει να δημιουργήσουμε προσωρινά αρχεία, για να αποθηκεύσουμε το βίντεό μας μέχρι να γίνει η μεταφόρτωσή του στην συνέχεια. Εκχωρούμε την καταγραφή βίντεο στην μεταβλητή writer μέσω της **cv2.VideoWriter(filename, fourcc, fps, frameSize[, isColor])**:

```
if writer is None:
      tempVideo = TempFile(ext=".mp4")
      writer = cv2.VideoWriter(tempVideo.path,0x21,30,(W, H),
      True)
```

Ελέγχουμε μέσω της μεταβλητής **startTime** εάν η διαφορά ώρας εκκίνησης του προγράμματος που είναι και η αρχική ώρα εκκίνησης καταγραφής ροής βίντεο και της παρούσας χρονικής στιγμής του ελέγχου έχει ξεπεράσει την απαιτούμενη χρονική διάρκεια. Εφόσον ισχύει η συνθήκη, εκκινούμε την διαδικασία μεταφόρτωσης του καταγεγραμένου αρχείου βίντεο. Συγκρατούμε την ώρα μεταφόρτωσης με σκοπό να αποδοθεί στην ονομασία

του αρχείου μέσω της μεταβλητής timestamp, διακόπτουμε την καταγραφή του τρέχοντος βίντεο και αρχικοποιούμε την μεταβλητή writer. Στην συνέχεια μεταφορτώνουμε το αρχείο βίντεο στο cloud, διαγράφουμε το προσωρινό αρχείο που δημιουργήσαμε και εκχωρούμε στην μεταβλητή **startTime** την νέα ώρα εκκίνησης:

```
if (datetime.now() - startTime).seconds > 20.0:
            timestamp = datetime.now()
            writer.release()
            writer = None
            print("[UPLOADING FILE]" )
            path = "/{base_path}/{timestamp}.mp4".format(
            base_path="pi_sucam", timestamp=timestamp)
            client.files_upload(open(tempVideo.path,
           "rb").read(), path)
            tempVideo.cleanup()
            startTime = datetime.now()
```

Σε περίπτωση που δεν έχει συμπληρωθεί το απαιτούμενο χρονικό όριο για την μεταφόρτωση του βίντεο δημιουργούμε έναν έλεγχο για να διασφαλίσουμε την καταγραφή του:

```
if writer is not None:
      writer.write(frame)
```

Δημιουργούμε το παράθυρο για την προβολή της ζωντανής ροής βίντεο σε οθόνη παρακολούθησης καθώς και τον απαραίτητο έλεγχο για την διακοπή του προγράμματος από τον χρήστη χρησιμοποιώντας το πλήκτρο “q”. Εφόσον επιλεγεί ο τερματισμός του προγράμματος, κλείνουμε όλα τα ανοιχτά παράθυρα και διακόπτουμε την λειτουργία της κάμερας.

```
cv2.imshow("Security Feed", frame)

key = cv2.waitKey(1) & 0xFF

if key == ord("q"):
      break

cv2.destroyAllWindows()
vs.stop()
```

3.2.5 Συμπεράσματα

Έχοντας πλέον μια λειτουργική εφαρμογή συστήματος παρακολούθησης, η οποία καταγράφει βίντεο και τα μεταφορτώνει σε ένα cloud storage, διαπιστώσαμε πως ο μεγάλος όγκος που προκύπτει από την συνεχή μεταφόρτωση αρχείων είναι ένα αναπόφευκτο πρόβλημα, τα οποία πολλές φορές μπορεί να μην περιέχουν κάποιο αξιοσημείωτο συμβάν. Σε συνδυασμό με τις χαμηλές ταχύτητες upload στην χώρα μας (η ταχύτητα του upload που είχαμε για την εξέλιξη του συστήματός μας περιοριζόταν στα 0.4 mbps), το σύστημα δεν είναι σε θέση να λειτουργήσει στις μέγιστες δυνατότητές του. Για τον λόγο αυτό ερευνήσαμε με ποιον τρόπο θα δημιουργήσουμε ένα σύστημα που θα καταγράφει μόνο ότι κρίνει απαραίτητο.

4. Υλοποίηση εφαρμογής παρακολούθησης με δυνατότητα ανίχνευσης κίνησης

4.1 Ανίχνευση κίνησης

Ως ανίχνευση κίνησης (motion detection) ορίζεται η διαδικασία αλλαγής της θέσης ενός αντικειμένου σε σχέση με το περιβάλλον ή την αλλαγή του περιβάλλοντα χώρου σε σχέση με ένα αντικείμενο. Μια τέτοια αλλαγή θα μπορούσε να θεωρηθεί η είσοδος ενός ατόμου ή αντικειμένου στον περιβάλλοντα χώρο.

4.1.1 Τεχνικές ανίχνευσης με επεξεργασία εικόνας

Η ανίχνευση κίνησης μέσω της επεξεργασίας εκόνας χαρακτηρίζεται από τον διαχωρισμό του κινούμενου αντικειμένου από την εικόνα του φόντου. Ένα σύστημα ανίχνευσης κίνησης πρέπει να έχει την δυνατότητα προσαρμογής σε αλλαγές του περιβάλλοντα χώρου και τις αλλαγές του φωτισμού κατά την διάρκεια της ημέρας.

- **Αφαίρεση του φόντου (Background subtraction)**

Η αφαίρεση του φόντου λειτουργεί μέσω μιας στατικής εικόνας ως σημείο αναφοράς η οποία θεωρείται το μοντέλο του φόντου, η εικόνα που εμπεριέχει κίνηση προκύπτει από μια διαφορά μεταξύ διαδοχικών καρέ και του μοντέλου του φόντου εξετάζοντας κάθε ένα εικονοστοιχείο με το αντίστοιχό του. Η συγκεκριμμένη μέθοδος δεν ενδείκνυται για χρήση όταν συμβαίνουν αλλαγές στο φόντο, όπως αλλαγές φωτισμού.

- **Χρονική διαφορά (Temporal differencing)**

Η μέθοδος ανίχνευσης κίνησης βασισμένη στην χρονική διαφορά, εξετάζει την διαφορά των αντίστοιχων εικονοστοιχείων διαδοχικών καρέ σε μια συγκεκριμμένη χρονική περίοδο. Τα διαδοχικά καρέ μπορεί να είναι είτε δύο, είτε περισσότερα. Η χρονική διαφορά προσαρμόζεται σε μεταβολλές του περιβάλλοντα χώρου, καθώς το μοντέλο του φόντου ανανεώνεται κάθε λίγα καρέ.

- **Στατιστικές μέθοδοι (Statistical Methods)**

Οι στατιστικές μέθοδοι είναι αποτελέσματα μελέτης της τεχνικής αφαίρεσης του φόντου. Οι στατιστικές μέθοδοι υπολογίζουν τα στατιστικά μεμονωμένων εικονοστοιχείων ή μιας ομάδας από εικονοστοιχεία και χρησιμοποιούν την πληροφορία για να ταξινομήσουν περιοχές μιας εικόνας ως περιοχές που ανήκουν στο φόντο ή στο προσκήνιο. Συχνά χρησιμοποιούνται γκαουσιανές για την μοντελοποίηση κάθε 18^{ου} εικονοστοιχείου και στη συνέχεια χρησιμοποιείται μια δυναμική διαδικασία προσέγγισης για την ενημέρωση του μοντέλου. Ένας άλλος τρόπος είναι η χρήση των μέγιστων και των ελάχιστων τιμών έντασης, και η μέγιστη απόκλιση αυτών των τιμών που προκύπτει από τα διάφορα καρέ, ως στατιστικές παράμετροι για την μοντελοποίηση του φόντου. Αυτή η τεχνική βρέθηκε ότι είναι περισσότερο εύρωστη σε αλλαγές των συνθηκών του φόντου.

4.2 Ανάλυση συστήματος

Όπως αναφέραμε στα συμπεράσματα της υλοποίησης του αρχικού μας συστήματος, παρότι κάλυπτε τις ανάγκες για τις οποίες δημιουργήθηκε θέλαμε να εξελίξουμε την λειτουργία του. Μελετώντας τις δυνατότητες της OpenCV ασχοληθήκαμε με την δυνατότητα ανίχνευσης κίνησης που μας παρέχει. Προχωρήσαμε στην δημιουργία ενός συστήματος παρακολούθησης το οποίο επίσης στηρίζεται στην ζωντανή ροή βίντεο μέσω του Camera Module, αλλά με την καταγράφη δεδομένων να γίνεται στην μορφή εικόνας των απαραίτητων καρέ μόνο όταν ανιχνευτεί παρουσία κίνησης στον χώρο που εποπτεύουμε. Με αυτόν τον τρόπο πετυχαίνουμε την εξάλειψη δεδομένων σε μορφή βίντεο, τα οποία πολλές φορές μπορεί να μην περιέχουν σημαντικό περιεχόμενο, την εξοικονόμηση αποθηκευτικού χώρου σε πολύ μεγάλο βαθμό, την μείωση της απαιτούμενης ταχύτητας μεταφόρτωσης δεδομένων αλλά και του χρόνου μεταφόρτωσής τους, κάνοντας το σύστημα μας λειτουργικό ακόμη και σε περιοχές όπου οι ταχύτητες των συνδέσεων στο διαδίκτυο είναι χαμηλές.

4.3 Συγγραφή κώδικα της εφαρμογής

Εκκινούμε και πάλι το πρόγραμμα μας εισάγοντας τα απαραίτητα προς χρήση πακέτα βιβλιοθηκών:

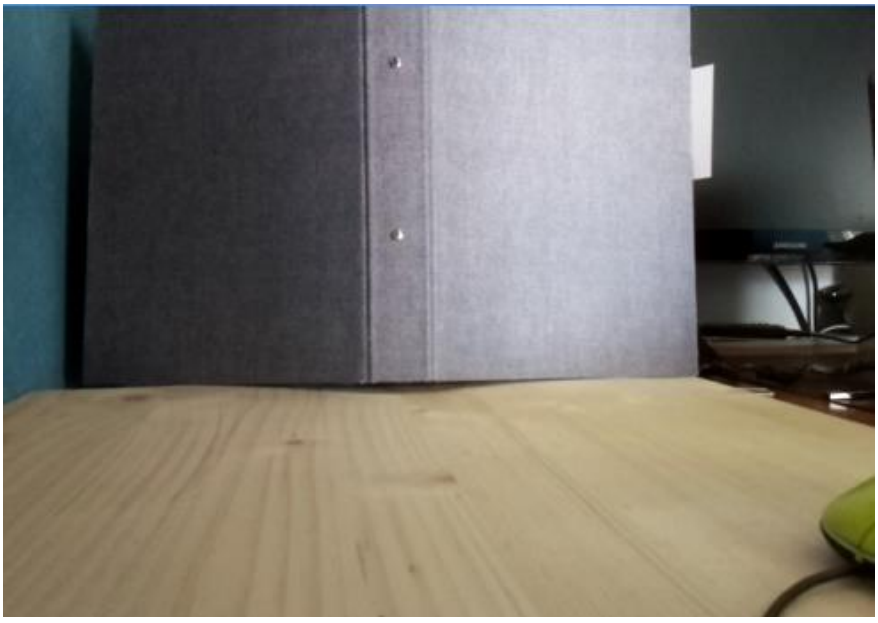
```
from picamera.array import PiRGBArray
from imutils.io import TempFile
from picamera import PiCamera
import datetime
import dropbox
import imutils
import time
import cv2
```

Πραγματοποιούμε την σύνδεση με τον αποθηκευτικό μας χώρο στην υπηρεσία του dropbox χρησιμοποιώντας τον σύνδεσμο API key που δημιουργήσαμε νωρίτερα:

```
client = None
client=dropbox.Dropbox("Jg4WWRJ20DkAAAAAAAA5pDWwVNmfC8LBRDN4Bs
jZQ6y-YRRsanF6FRhw-KHtRNT6")
print("[DROPBOX ACCOUNT LINKED SUCCESSFULLY]")
```

Αρχικοποιούμε την κάμερα μας, ορίζουμε την ανάλυσή της, ορίζουμε τον αριθμό των καρέ ανά δευτερόλεπτο, καταγράφουμε το πρώτο καρέ με χρήση της **PiRGBArray(camera, size)** η οποία δημιουργεί έναν τρισδιάστατο πίνακα (γραμμές,στήλες, χρώματα) μέσω της λήψης ενός ακατέργαστου RGB καρέ, αφήνουμε λίγο χρόνο στον αισθητήρα της κάμερας να προθερμανθεί, αρχικοποιούμε την μεταβλητή **avg** με την οποία στην συνέχεια θα συγκρίνουμε τα νέα καρέ για να ανιχνεύσουμε τυχόν κίνηση στον χώρο. Εκχωρούμε την ημερομηνία και ώρα εκκίνησης στην μεταβλητή **lastUploaded** μέσω της **datetime.datetime.now()** και αρχικοποιούμε τον μετρητή **motionCounter**:

```
camera = PiCamera()
camera.resolution = ([640,480])
camera.framerate = 16
capture = PiRGBArray(camera, size=([640,480]))
print("[CAMERA SENSOR WARMING UP]")
time.sleep(3)
avg = None
lastUploaded = datetime.datetime.now()
motionCounter = 0
```

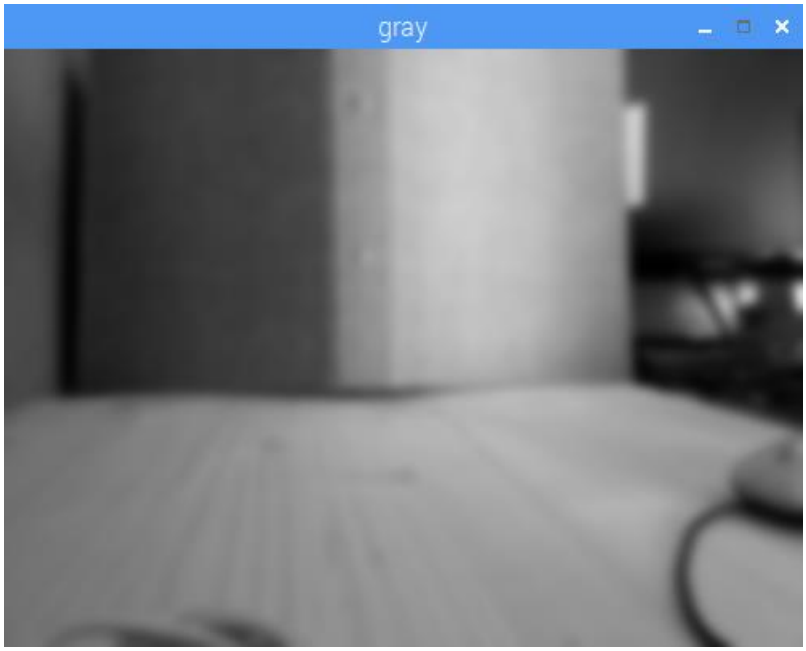


10 Αρχικό καρέ που λαμβάνει η κάμερα μας

Δημιουργούμε έναν ατέρμονα βρόγχο με την χρήση της for για την συνεχή καταγραφή καρέ μέσω της κάμερας χρησιμοποιώντας την **camera.capture_continuous(output, format = None, use_video_port = True)**, μετατρέπουμε κάθε καρέ σε αποχρώσεις του γκρι για εύκολη και ταχύτερη αναγνώριση αλλαγών στον χώρο χρησιμοποιώντας την **cv2.cvtColor(image, flag)**.

```
for f in camera.capture_continuous(capture, format="bgr",
use_video_port=True):

      frame = f.array
      timestamp = datetime.datetime.now()
      detector = "nomotion"
      gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```



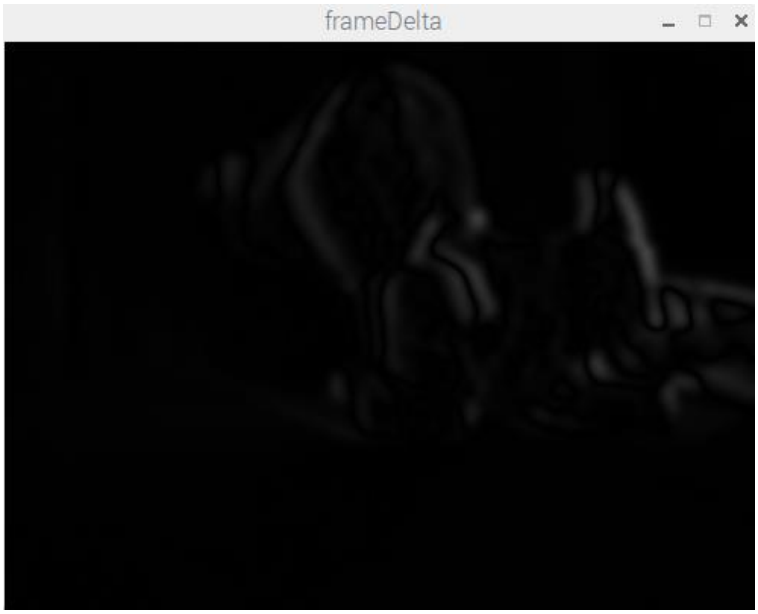
11 Τρέχον καρέ μετά την μετατροπή σε τόνους του γκρι

Εφόσον η μεταβλητή **avg** είναι ακόμη κενή, αρχικοποιούμε το μοντέλο φόντου του χώρου που εποπτεύουμε, σε στατική μορφή χρησιμοποιώντας το πρώτο καρέ και κάνουμε εκκαθάριση της μεταβλητής **capture** στην οποία κάνουμε την καταγραφή των νέων καρέ, ώστε να δεχτεί το επόμενο, χρησιμοποιώντας την **output.truncate(0)**.

```
if avg is None:
        avg = gray.copy().astype("float")
        capture.truncate(0)
```

Επειδή κατά την διάρκεια λειτουργίας του προγράμματος μας οι συνθήκες στον χώρο παρακολούθησης μπορεί να αλλάξουν με βασικότερη όλων την εναλλαγή του φωτισμού κατά την διάρκεια της ημέρας, θα ήταν αναποτελεσματικό να χρησιμοποιούμε αποκλειστικά το πρώτο καρέ σαν μέτρο σύγκρισης για την ανίχνευση κίνησης στον χώρο. Για τον λόγο αυτό χρησιμοποιούμε το άθροισμα κάθε νέου καρέ και του μοντέλου φόντου που συσσωρεύει τα νέα καρέ και μεταβάλλεται κατά την διάρκεια εκτέλεσης του προγράμματος. Αυτό επιτυγχάνεται με την χρήση της συνάρτησης: **cv2.accumulateWeighted(src, dst, alpha).** Δέχεται ως παραμέτρους την εικόνα εισόδου και την εικόνα εξόδου, καθώς και μια τιμή η οποία καθορίζει την βαρύτητα της κάθε εικόνας εισόδου. Στην συνέχεια υπολογίζει και επιστρέφει στην παράμετρο **dst** το άθροισμα των διαδοχικών εικόνων εισόδου και της εικόνας εξόδου που τις συσωρεύει (**dst**), έτσι ώστε η εικόνα εξόδου να αποτελεί ένα μεταβαλλόμενο μοντέλο φόντου από την συνεχή λήψη καρέ.

```
cv2.accumulateWeighted(gray, avg, 0.5)
```



12 Το καρέ μετά την εφαρμογή της absdiff

Στην συνέχεια, χρησιμοποιούμε την απόλυτη διαφορά που δημιουργείται μεταξύ των στοιχείων κάθε νέου καρέ και της εικόνας που αποτελεί το μοντέλο φόντου. Στην περίπτωση μας δέχεται σαν τιμές τις εικόνες gray και avg, με την τελευταία να μετατρέπεται κατάλληλα με την χρήση της cv2.convertScaleAbs(src).

```
frameDelta = cv2.absdiff(gray, cv2.convertScaleAbs(avg))
```

Συχνά τα εικονοστοιχεία ενός αντικειμένου μιας εικόνας παίρνουν τιμές σε ένα μικρό διάστημα αποχρώσεων. Αυτό οδηγεί συνήθως στη δημιουργία ενός τοπικού μέγιστου στην περιοχή του ιστογράμματος της εικόνας. Η εύρεση τέτοιων τοπικών μεγίστων διευκολύνει τον εντοπισμό των αντικειμένων της εικόνας και την απόδοσή της με λιγότερες κύριες αποχρώσεις. Οι τιμές του πεδίου των αποχρώσεων μεταξύ των οποίων εμφανίζονται τοπικά μέγιστα του ιστογράμματος λέγονται κατώφλια.

Στην **frameDelta** εφαρμόζουμε δυαδική κατωφλίωση έτσι ώστε να ξεχωρίσουμε τα περιγράμματα των αντικειμένων στο τρέχον καρέ μας. Για τον λόγο αυτό χρησιμοποιούμε την, **cv2.threshold(src, thresh, maxval, type).** Εάν κάποιο στοιχείο της εικόνας που έχουμε δώσει ως είσοδο (**src**), υπερβαίνει την τιμή κατωφλίωσης που έχουμε ορίσει (**thresh**) τότε

αντικαθιστάται με την τιμή που έχουμε ορίσει ως **maxval**. Χρησιμοποιούμε τον δείκτη **[1]** μετά την συνάρτηση έτσι ώστε να αποθηκεύσουμε στην μεταβλητή **thresh**, την δεύτερη επιστρεφόμενη τιμή, η οποία περιέχει μια δισδιάστατη λίστα απότελούμενη από τα στοιχεία της εικόνα μας.

```
thresh = cv2.threshold(frameDelta, 5, 255,
        cv2.THRESH_BINARY)[1]
```

13 Το τρέχον καρέ μετά την εφαρμογή δυαδικής κατωφλίωσης

Στην συνέχεια αφού έχουμε φέρει το τρέχων καρέ στο κατάλληλο στάδιο, κάνουμε έυρεση περιγραμμάτων των αντικειμένων του χώρου για να εντοπίσουμε πιθανή κίνηση στον χώρο. Χρησιμοποιούμε την μέθοδο επιστροφής μόνο των εξωτερικών περιγραμμάτων τα οποία αποτελούν και τα πρώτα στην ιεραρχία (**cv2.RETR_EXTERNAL**), παράλληλα χρησιμοποιούνται τα ελάχιστα δυνατά σημεία για την εύρεση των περιγραμμάτων μέσω της μεθόδου **cv2.CHAIN_APPROX_SIMPLE,** κατά την οποία συμπιέζονται τα οριζόντια, κάθετα και διαγώνια τμήματα και συγκρατούνται μόνο τα σημεία στα άκρα τους.

```
cnts = cv2.findContours(thresh.copy(), cv2.RETR_EXTERNAL,
        cv2.CHAIN_APPROX_SIMPLE)
    cont = imutils.grab_contours(cnts)
```

Σε περίπτωση που η περιοχή αυτών των περιγραμμάτων (**cv2.contourArea**) υπερβαίνει των αριθμό pixel που έχουμε ορίσει σημαίνει πως το σύστημά μας έχει ανιχνεύσει την παρουσίας κίνησης σε σημαντικό βαθμό στον χώρο μας. Η μεταβολή αυτού του αριθμού καθορίζει την ευαισθησία του συστήματός μας:

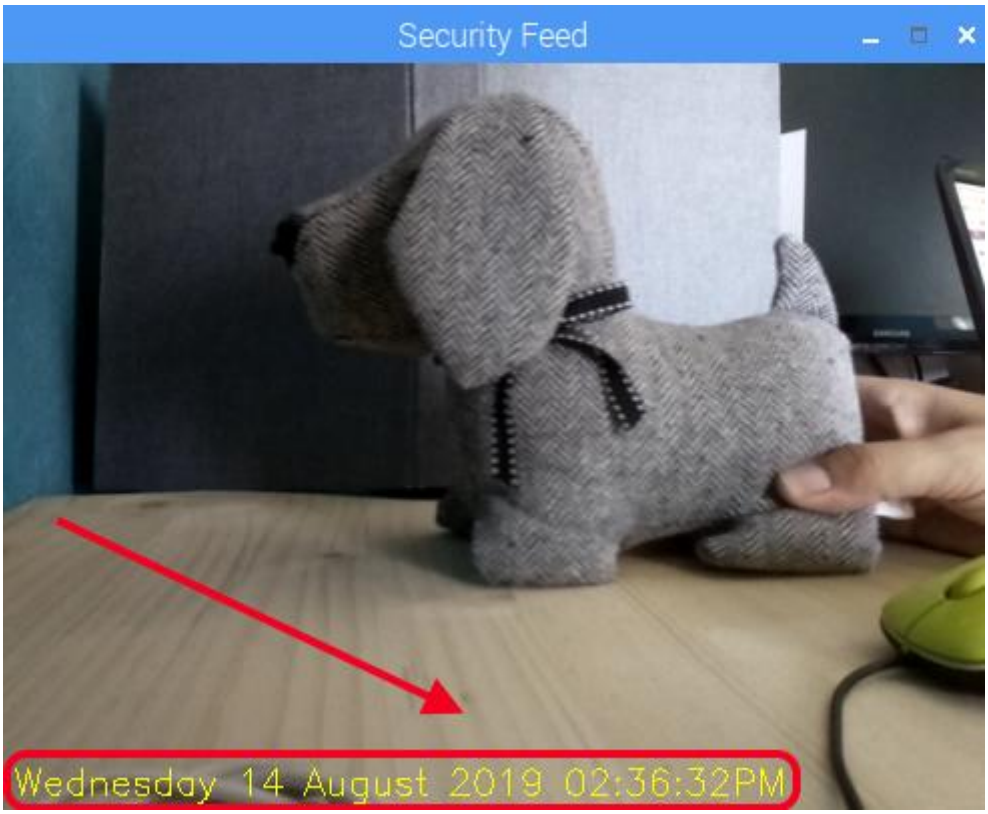
```
for c in cont:
        if cv2.contourArea(c) > 5000:
            detector = "motion"
```

Ελέγχουμε εάν υπάρχει κίνηση μέσω της μεταβλητής **detector,** εφόσον υπάρχει κάνουμε έλεγχο για το αν η προηγούμενη μεταφόρτωση αρχείου εικόνας έγινε σε διάστημα μεγαλύτερο του χρόνου που έχουμε ορίσει, εφόσον ισχύει, αυξάνουμε τον μετρήτη των καρέ που περιλαμβάνουν κίνηση κατά μία μονάδα:

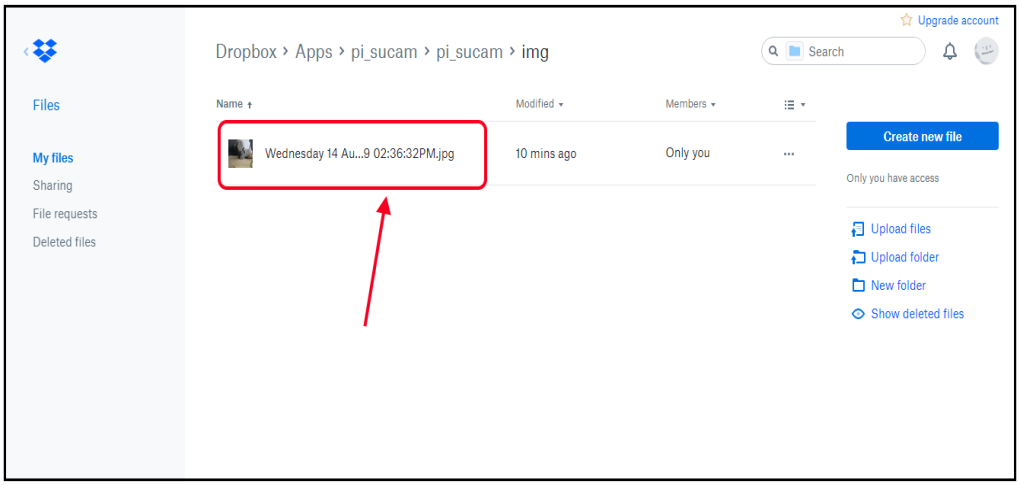
```
if detector == "motion":
      if (timestamp - lastUploaded).seconds >= 3:
            motionCounter += 1
```

Έχοντας συγκεντρώσει τον απαιτούμενο αριθμό καρέ τα οποία περιέχουν κίνηση δημιουργούμε την μεταβλητή **TempImage** στην οποία εκχωρούμε το προσωρινό μας αρχείο για την αποθήκευση της εικόνας το οποίο δημιουργούμε με την βοήθεια της **TempFile()**. Στην συνέχεια μεταφορτώνουμε την εικόνα στο dropbox, κάνουμε εκαθάριση του προσωρινού αρχείου, ενημερώνουμε εκ νέου την χρονική στιγμή της μεταβλητής τελευταίας μεταφόρτωσης **lastUploaded** και αρχικοποιούμε ξανά τον μετρητή κίνησης:

```
if motionCounter >= 8:
      TempImage = TempFile(ext=".jpg")
      cv2.imwrite(TempImage.path, frame)
      print("[MOTION DETECTED] {}".format(ts))
      path = "/{base_path}/{timestamp}.jpg".format(
      base_path="pi_sucam/img", timestamp=ts)
      client.files_upload(open(TempImage.path, "rb").read(),
      path)
      TempImage.cleanup()
      lastUploaded = timestamp
      motionCounter = 0
```



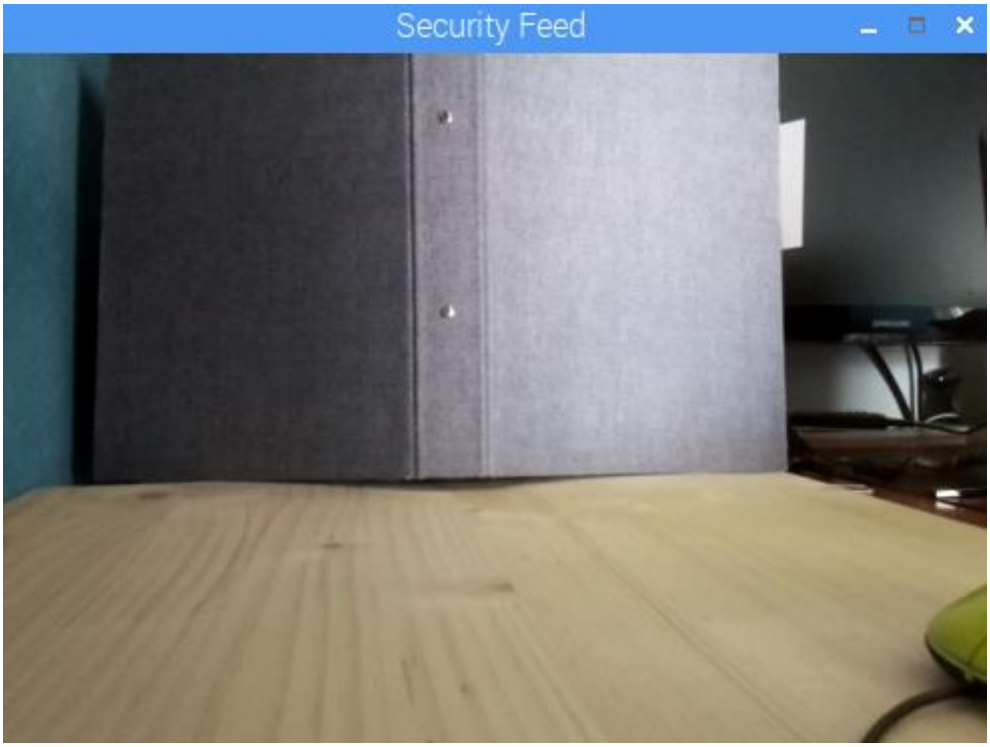
14 Το τελικό καρέ που μεταφορτώνεται στο cloud



15 Η επιτυχής μεταφόρτωση του καρέ που έχει ανιχνευτεί κίνηση

Προβάλλουμε την ζωντανή ροή βίντεο στην οθόνη, σε περίπτωση που ο χρήστης δώσει εντολή διακοπής μέσω του πλήκτρου “q” διακόπτουμε την δομή επανάληψης και ταυτοχρόνως το πρόγραμμα μας.

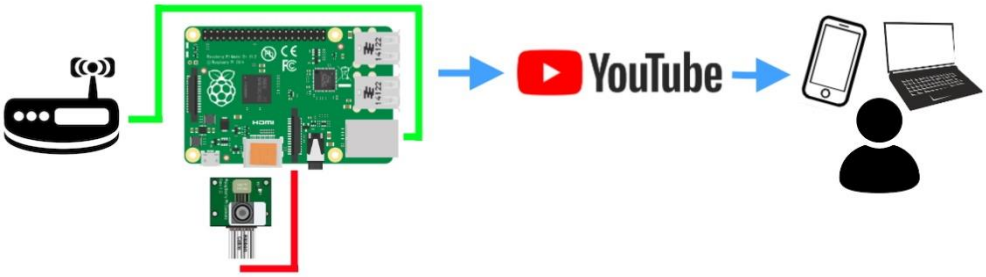
```
cv2.imshow("Security Feed", frame)
key = cv2.waitKey(1) & 0xFF
if key == ord("q"):
      break
capture.truncate(0)
```



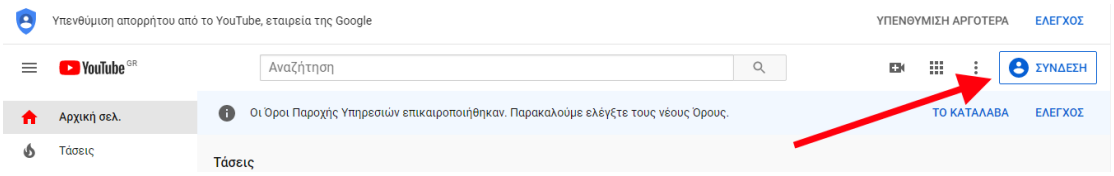
16 Η ζωντανή ροή βίντεο από την κάμερά μας

5. Δημιουργία ζωντανής ροής βίντεο (livestream) μέσω της πλατφόρμας YouTube

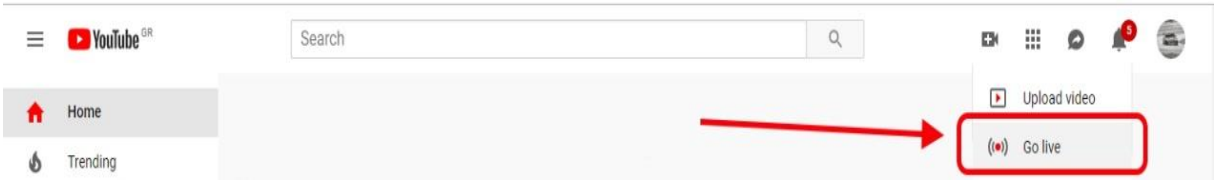
Ένας ακόμη πιθανός τρόπος παρακολούθησης ενός χώρου είναι μέσω της καταγραφής βίντεο και της ζωντανής μετάδοσής του διαδικτυακά. Μια πλατφόρμα που μας παρέχει αυτή την δυνατότητα είναι η πανσίγνωστη πλέον μηχανή αναζήτησης βίντεο, YouTube. Έχοντας ένα Raspberry Pi με ενσωματωμένο το Camera Module V2, μπορούμε να το χρησιμοποιήσουμε ως πομπό για την καταγραφή βίντεο ενός χώρου και την μετάδοση του στο YouTube.



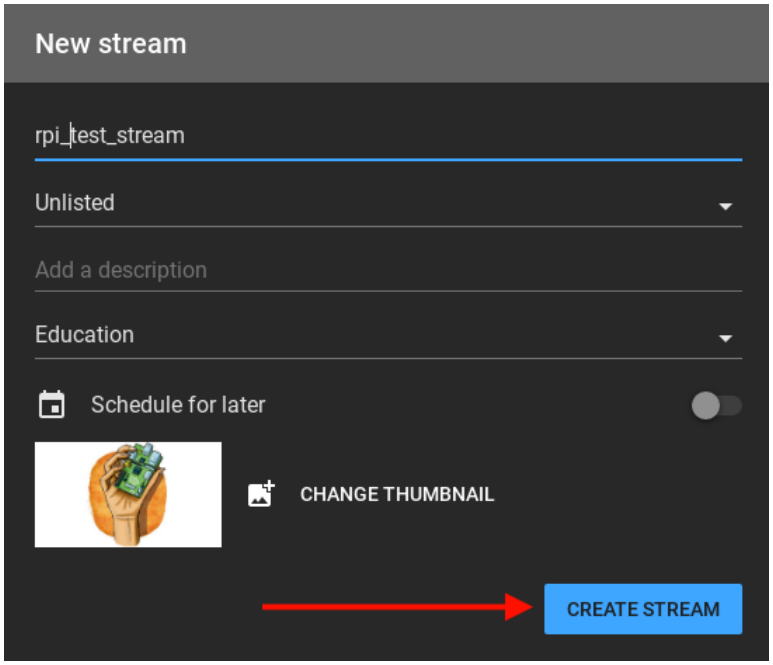
Για την υλοποίηση του συστήματος αρχικά χρειάζεται η δημιουργία λογαριασμού στο Youtube. Σε περίπτωση που διαθέτουμε ήδη κάποια διεύθυνση mail στην υπηρεσία Gmail, μπορούμε να συνδεθούμε χρησιμοποιώντας αυτήν, διαφορετικά επισκεπτόπμαστε τον ιστότοπο https://youtube.com και επιλέγουμε σύνδεση:



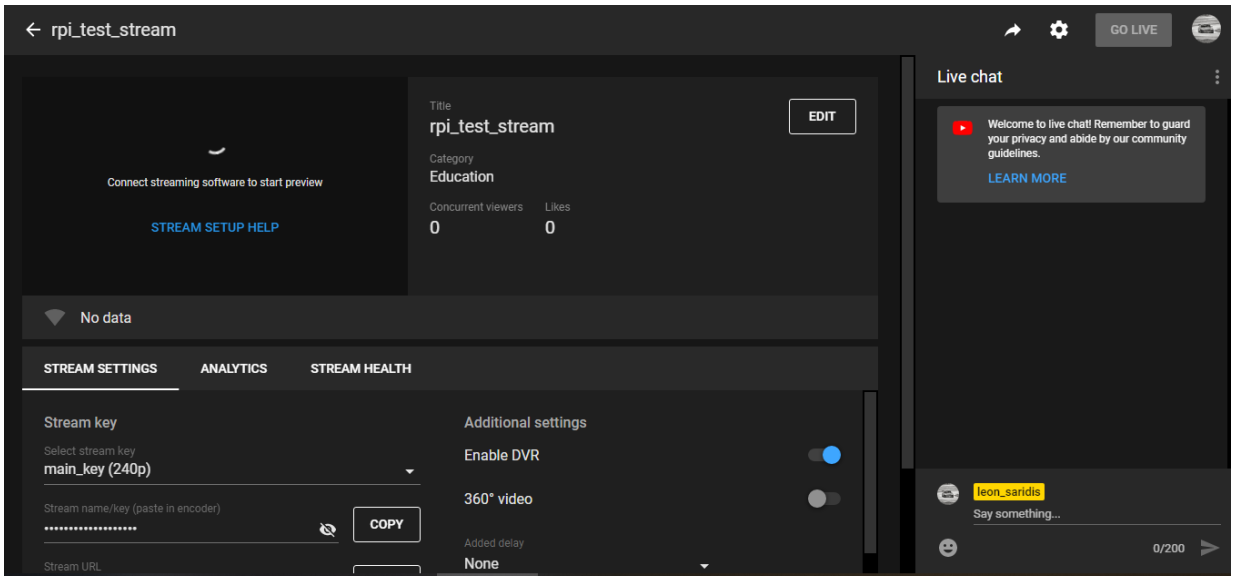
Στην συνέχεια επιλέγουμε **Δημιουργία λογαριασμού** και είσαγουμε τα απαραίτητα στοιχεία που θα μας ζητηθούν. Ακολουθούμε την διαδικασία και μόλις ολοκληρωθεί συνδεόμαστε στον λογαριασμό μας. Πλέον μπορούμε να προετοιμάσουμε την ζωντανή μετάδοση βίντεο που επιθυμούμε. Επιλέγουμε **“Go live”**



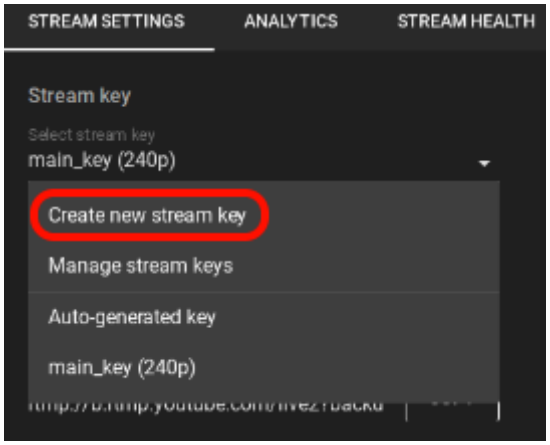
Εισάγουμε τον τίτλο που επιθυμούμε για την ζωντανή μετάδοσή μας, επίλέγουμε τον τρόπο που θα προβλήθεί, Δημόσια, Ιδιωτικά, ή δημοσιοποίηση κατ επιλογήν χειροκίνητα από εμάς με την χρήση υπερσυνδέσμου. Στην συνέχεια επιλέγουμε μια κατηγορία σχετική με το αντικείμενο της μετάδοσης, προαιρετικά μπορούμε ακόμη να εισάγουμε μια περιγραφή για το περιέχομενο της μετάδοσής μας για την καλύτερη πληροφόρηση των θεατών σε περίπτωση δημόσιας προβολής. Τέλος έχουμε την δυνατότητα να προγραμματίσουμε την εκκίνηση της μετάδοσης σε συγκεκριμμένη ημερομηνία και ώρα που επιθυμούμε:



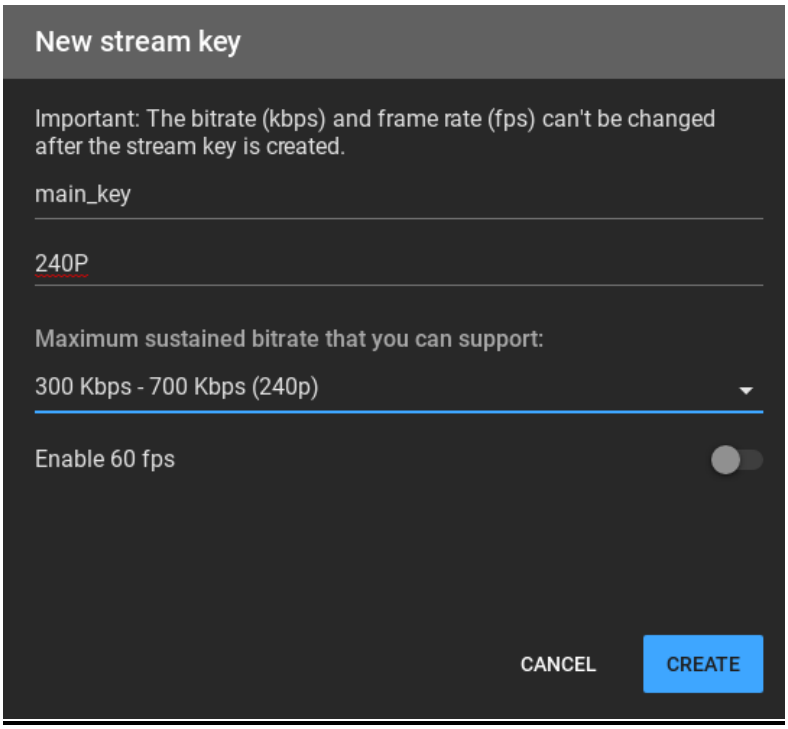
Μετά την εισαγωγή των βασικών πληροφοριών της μετάδοσης θα μας εμφανιστεί η σελίδα διαχείρησης της μετάδοσής μας:



Σε αυτό το σημείο θα δημιουργήσουμε έναν σύνδεσμο κλειδί για την διασύνδεση του YouTube με το Raspberry Pi. Μπορούμε να κάνουμε χρήση ενός κλειδιού που θα δημιουργηθεί αυτόματα, στην δική μας περίπτωση θα δημιουργήσουμε ένα μόνιμο κλειδί ώστε να έχουμε την δυνατότητα να το χρησιμοποιούμε και μελλοντικά.

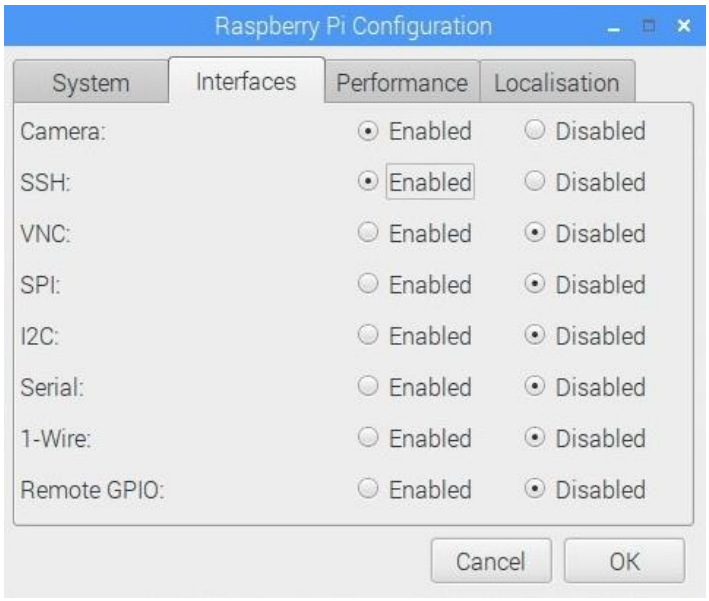


Εισάγουμε τα απαραίτητα στοιχεία ανάλογα με τις ανάγκες και τις δυνατότητες του συστήματος μετάδοσης που έχουμε καθώς και μια ονομασία για το κλειδί και επιλέγουμε **Create.**

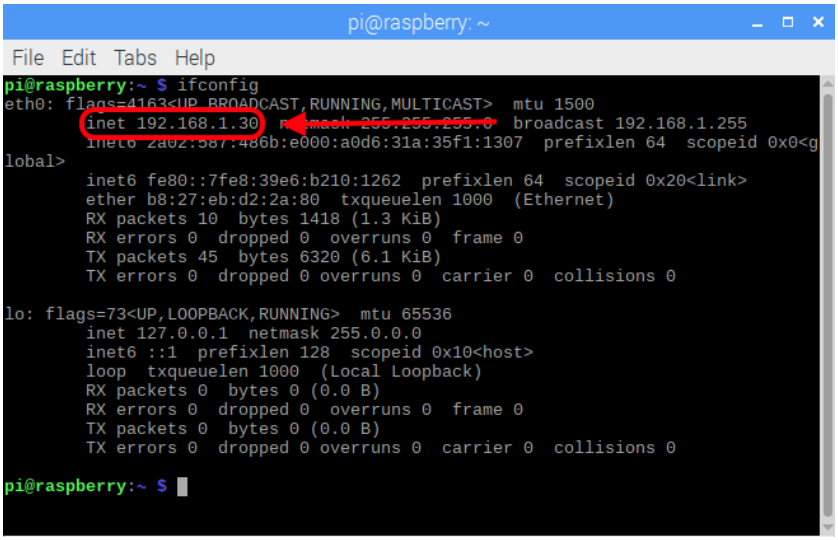


Πλέον, η μετάδοση έχει προγραμματιστεί και βρίσκεται εν αναμονή λήψης δεδομένων βίντεο από το Raspberry Pi. Για να έχουμε την δυνατότητα εκκίνησης της ζωντανής μετάδοσης και διατήρησης της λειτουργίας της μετά την απομάκρυνσή μας από τον χώρο που παρακολουθούμε, θα πρέπει είτε να χρησιμοποιούμε το raspberry pi σαν ένα σύστημα Η/Υ μαζί με μια οθόνη, είτε να το διαχειριζόμαστε από άλλο υπολογιστή στο ίδιο δίκτυο μέσω της λειτουργίας SSH (Secure Shell).

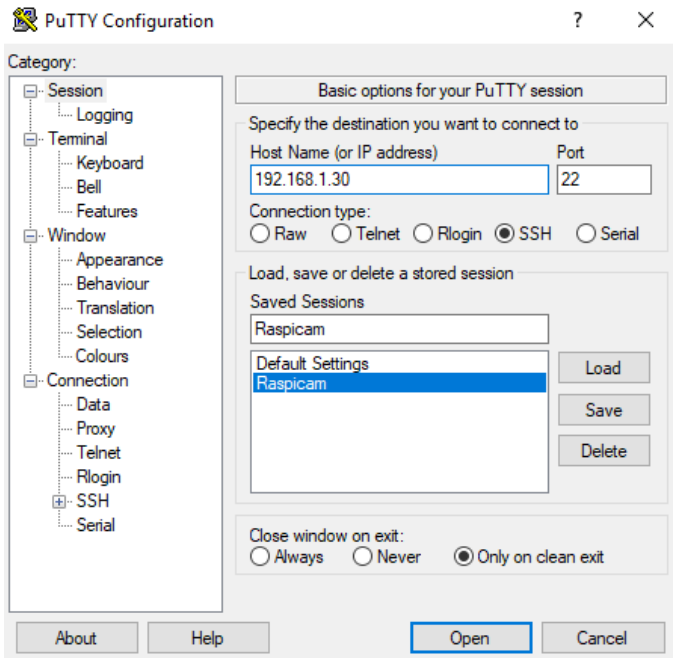
Για την διαχείρηση του Raspberry Pi μέσω SSH θα χρειαστούμε την εφαρμογή Putty (https://www.putty.org/). Πραγματοποιούμε λήψη και εγκατάσταση της εφαρμογής. Ενεργοποιούμε την λειτουργία SSH στο Raspberry Pi, από το μενού διαχείρησης του, (raspberry pi configuration) μέσω του κεντρικού μενού.



Μέσω της γραμμής εντολών και με την βοήθεια της εντολής **ifconfig,** λαμβάνουμε την διεύθυνση ip του raspberry για την διασύνδεση του με τον Η/Υ που χρησιμοποιούμε για να το διαχειρίστούμε μέσω SSH.



Εισάγουμε τα απαραίτητα στοιχεία που θα μας ζητηθούν στην εφαρμογή Putty, καθώς και το όνομα χρήστη και κωδικό πρόσβασης που έχουμε ορίσει για την διασύνδεση με το raspberry.



Εφόσον έχουμε πραγματοποιήσει την διασύνδεση μας με το raspberry pi από τον ηλεκτρονικό μας υπολογιστή μέσω SSH, εγκαθιστούμε το πρόσθετο “screen” το οποίο θα μας επιτρέψει να διακόψουμε την σύνδεση μας μέσω SSH με το raspberry χωρίς όμως να διακόψουμε την καταγραφή και μετάδοση βίντεο.

sudo apt install screen
sudo reboot

Μετά την επανεκκίνηση του raspberry, επανασυνδεόμαστε μέσω SSH από τον ηλεκτρονικό μας υπολογιστή και τρέχουμε την εντολή **screen**.

Εφόσον βρισκόμαστε στο περιβάλλον του screen χρησιμοποιούμε την εντολή raspivid για την καταγραφή βίντεο μέσω του Camera Module και την εντολή avconv ώστε να αποστείλουμε στο YouTube όσα καταγράφουμε.

raspivid -o - -t 0 -w 360 -h 240 -fps 16 -b 6000000 -n | avconv -re -ar 44100 -ac 2 -acodec pcm_s16le -f s16le -ac 2 -i /dev/zero -f h264 -i - -vcodec copy -acodec aac -ab 128k -g 50 -strict experimental -f flv rtmp://a.rtmp.youtube.com/live2/ft3m-shqs-e3vs-09gr

-t 0 χρονική διάρκεια βίντεο, με την παράμετρο 0 το βίνετο μας δεν έχει χρονικό περιορισμό, διακόπτεται με δική μας επίλογή όταν ολοκληρωθεί η εκτέλεση της μετάδοσης

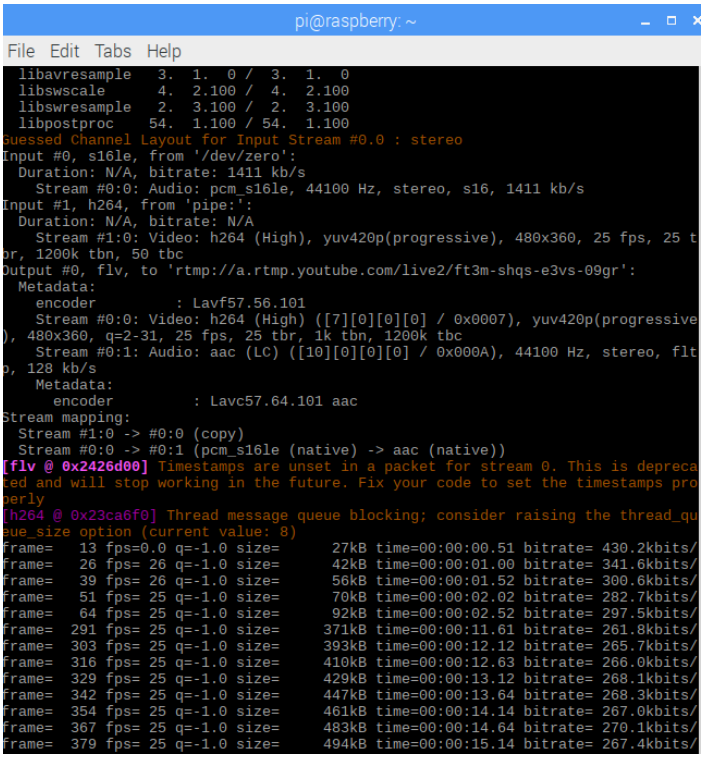
-w 360 –h 240 ορίζουμε την ανάλυση καταγραφής

-fps ορίζουμε τον ρυθμό καρέ ανά δευτερόλεπτο

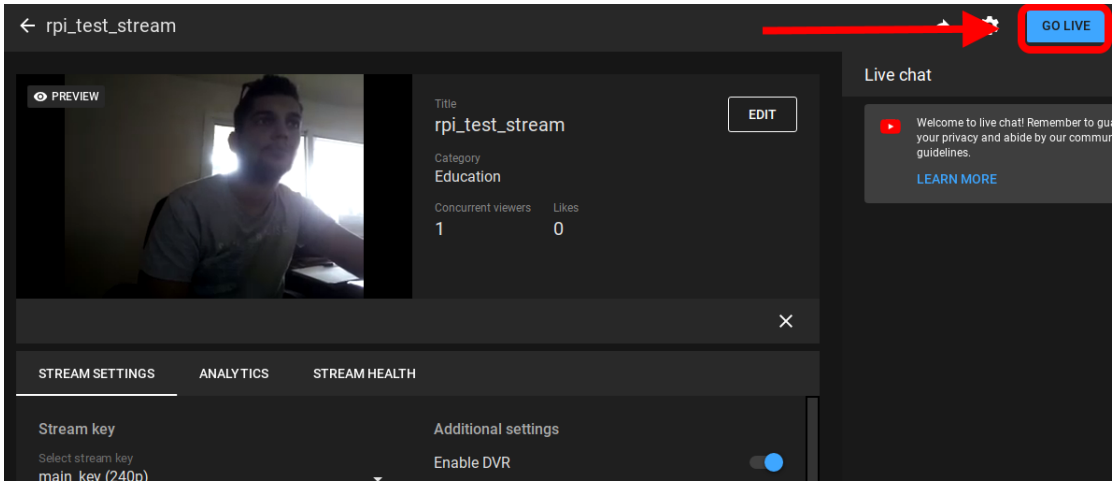
-b ορίζουμε τον ρυθμό μετάδοσης δεδομένων

-n απενεργοποιούμε την προεπισκόπηση βίντεο σε αναδυόμενο παράθυρο

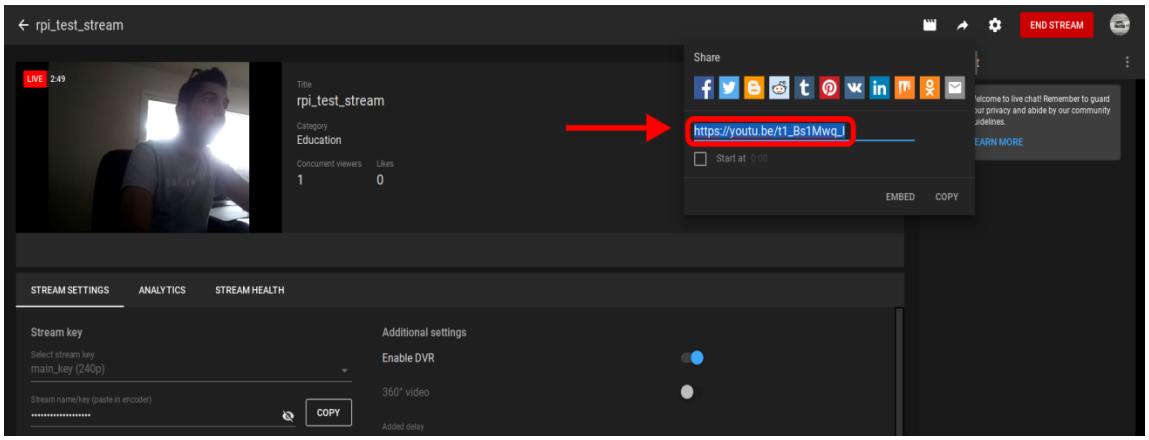
Με την εντολή **avconv** θα μετατρέψουμε το βίντεο που καταγράφουμε στην απαραίτητη μορφή (**flv**) για να το μεταδώσουμε στο Youtube. Το Youtube δεν μας επιτρέπει την δημιουργία ζωντανών μετάδοσεων χωρίς ήχο, δεδομένου οτι το raspberry pi δεν διαθέτει προεγκατηστημένο κάποιο περιφεριακό καταγραφής ήχου δημιουργούμε ένα εικονικό προφίλ ήχου. Χρησιμοποιούμε τον σύνδεσμο που δημιουργήσαμε νωρίτερα **ft3m-shqs-e3vs-09gr** για την διασύνδεση με το Youtube. Με την εκτέλεση της εντολής έχουμε ξεκινήσει την αποστολή δεδομένων στο Youtube.

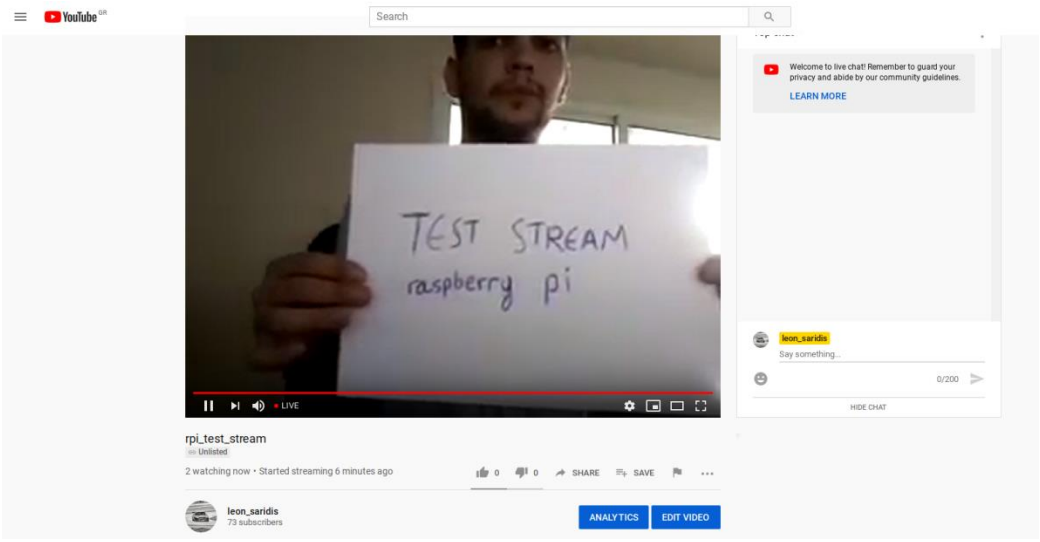


Ελέγχουμε εάν έχουμε φυσιολογική λήψη δεδομένων μέσω της σελίδας διαχείρησης της ζωντανής μετάδοσης στο Youtube και επιλέγουμε “Go live”



Για τον διαμοιρασμό της ζωντανής μετάδοσης ή την παρακολούθηση της μπορούμε να χρησιμοποιήσουμε τον σχετικό σύνδεσμο, ή ακόμη και την εφαρμογή του Youtube για συσκευές κινητών τηλεφώνων αποκτώντας έτσι την δυνατότητα απομακρυσμένης παρακολούθησης του χώρου μας.





Έχοντας πλέον ολοκληρώσει την εκκίνηση της ζωντανής μετάδοσης, διακόπτουμε την διασύνδεση SSH με το raspberry, πληκτρολογώντας αρχικά **Ctrl +A** και κλείνοντας την εφαρμογή Putty στην συνέχεια. Η μετάδοση μας είναι πλέον σε λειτουργία και προσβάσιμη από οποιαδήποτε συσκευή έχουμε πρόσβαση στο διαδίκτυο.

Συμπεράσματα

Με την υλοποίηση του παραπάνω συστήματος διαπιστώνουμε πως με την βοήθεια του Raspberry pi έχουμε την δυνατότητα να παρακολουθούμε όποιον χώρο επιθυμούμε με πολύ απλό και οικονομικό τρόπο από ώπου και αν βρισκόμαστε. Εξελίσσοντας το σύστημα, θα μπορούσαμε να κάνουμε το raspberry pi προσβάσιμο διαδικτυακά έτσι ώστε να έχουμε την δυνατότητα εκκίνησης της καταγραφής και μετάδοσης βίντεο απομακρυσμένα όποιαδήποτε στιγμή επίθυμούμε, χωρίς να έχουμε τον περιορισμό εκκίνησης της διαδικασίας από το ίδιο δίκτυο με το raspberry.

6. Κόστος κατασκευής συστήματος

Προϊόν	**Τιμή (συμπεριλαμβανομένων μεταφορικών)**
Raspberry Pi 3 Model B+ kit (raspberry pi, θήκη, τροφοδοτικό)	*65€*
Raspberry Pi Camera Module V2	*40€*
Οθόνη Samsung 24”	*110€*
Πληκτρολόγιο/ποντίκι	*15€*
Κάρτα μνήμης microSD 32GB	*20€*
USB Hub	*8€*
Καλώδιο HDMI	*6€*
Custom 3D printed θήκη camera board	*-*
ΣΥΝΟΛΟ	***264€***

Το κόστος του συστήματος διαφοροποιείται αναλόγα με τις απαιτήσεις του. Παραδείγματος χάρη σε περίπτωση που δεν επιθυμούμε να έχουμε μια οθόνη για την εποπτεία της ζωντανής ροής βίντεο, όπως είναι προφανές το κόστος μειώνεται κατά μεγάλο βαθμό.

7. Συμπεράσματα

Η έρευνά μας, απέδειξε πως έχουμε ήδη εκπληρώσει την μετάβαση σε μεγάλο βαθμό σε μια εποχή στην οποία τα υπολογιστικά συστήματα έχουν την δυνατότητα να συνεργαστούν μεταξύ τους για την συλλογή, επεξεργασία αξιολόγηση δεδομένων και την αυτόνομη λήψη αποφάσεων βάση των οποίων ενεργούν. Η συνεργασία της βιομηχανίας συστημάτων παρακολούθησης με τις δυνατότητες που παρέχει η επιστήμη της τεχνητής όρασης και της τεχνητής νοημοσύνης έχει δημιουργήσει συστήματα ικανά να λειτουργήσουν αποτελεσματικά σε ένα μεγάλο τμήμα του εύρους των απαιτήσεων.

8. Μελλοντική εξέλιξη των συστημάτων

Αναφορικά με τα συστήματα τα οποία υλοποιήσαμε κατά την διάρκεια της έρευνάς μας και σχετικά με την εξέλιξή τους, τα επόμενα βήματα θα μπορούσαν να βρουν εφαρμογή στην τεχνολογία ανίχνευσης προσώπων, έλεγχο μέσα από μια βάση δεδομένων η οποία θα απαρτίζεται από εικόνες των κατοίκων ή εργαζόμενων του χώρου ώπου λειτουργεί το σύστημά μας. Σε περίπτωση που ανιχνευτεί στον χώρο ένα πρόσωπο το οποίο δεν είναι καταχωρημένο στην βάση δεδομένων θα αποστέλλεται η αντίστοιχη ειδοποίηση στον αρμόδιο υπεύθυνο για την ασφάλεια του χώρου.

Μία ακόμη πιθανή εξέλιξη της εφαρμογής με την χρήση αναγνώρισης προσώπου θα ήταν η χρήση του συστήματος παρακολούθησης για την δημιουργία στατιστικών επισκεψιμότητας σε εμπορικούς χώρους. Η εφαρμογή μπορεί να καταγράφει την είσοδο και έξοδο κάθε ατόμου στον χώρο και να υπολογίζει την ώρα που διήρκησε η επίσκεψή τους. Ακόμη καταχωρώντας κάθε νέο πρόσωπο σε μία βάση δεδομέων θα μπορούσε να υπολογίζει την επαναληψιμότητα των επισκέψεων για τα επιθυμητά χρονικά διαστήματα που ορίζει ο χρήστης.

Σύστημα παρακολούθησης με δυνατότητα καταγραφής βίντεο σε cloud storage

```
#video_capture.py

#εισαγωγή των απαιτούμενων πακέτων
from imutils.video import VideoStream
from imutils.io import TempFile
from picamera import PiCamera
from datetime import datetime
from datetime import date
import imutils
import time
import cv2
import dropbox


#εκκίνηση της ροής βίντεο χρησιμοποιώντας ως μέσο το Camera
Module, παύση εκτέλεσης του προγράμματος
#για να δωθεί ο απαιτούμενος χρόνος προθέρμανσης στον
αισθητήρα της κάμερας
vs = VideoStream(usePiCamera=True).start()
print("[CAMERA SENSOR WARMING UP]")
time.sleep(2.0)

#δημιουργία της διασύνδεσης με τον αποθηκευτικό χώρο στο
Dropbox με την χρήση API key
client =
dropbox.Dropbox("Jg4WWRJ20DkAAAAAAAA5pDWwVNmfC8LBRDN4BsjZQ6y-
YRRsanF6FRhw-KHtRNT6")
print("[DROPBOX ACCOUNT LINKED SUCCESSFULLY]")

#αρχικοποίηση της μεταβλητής που θα χρησιμοποιηθεί για την
καταγραφή των αρχείων βίντεο
#αρχικοποίηση των μεταβλητών μέσω των οποίων θα ορίσουμε τις
διαστάσεις καταγραφής του βίντεο
writer = None
W = None
H = None

#εκχώρηση της ώρας εκκίνησης στην μεταβλητή startTime
startTime = datetime.now()
print("[RECORDING STARTED]")
print(startTime.strftime("%A %d %B %Y %I:%M:%S%p"))


#εκκίνηση ατέρμονα βρόγχου
while True:

    #εκχώρηση της ροής βίντεο στην μεταβλητή frame
    frame = vs.read()

    #αλλαγή του μεγέθους ανάλυσης, χρησιμοποιούμε την
imutils.resize για να μην αλλοιωθεί το aspect ratio
```

```
    #δίνουμε το πλάτος (width) και το ύψος (height)
προσαρμόζεται ανάλογα
    frame = imutils.resize(frame, width=480)

    #κατά την πρώτη εκτέλεση του βρόγχου ώπου οι μεταβλητές H,
W είναι κενές, ορίζουμε τις διαστάσεις του καταγραφικού
    #λαμβάνοντας τις από την μεταβλητή frame ώπου εκχωρήσαμε
την ροή βίντεο
    if W is None or H is None:
        H = frame.shape[0]
        W = frame.shape[1]

    #εφόσον η μεταβλήτη του καταγραφικού είναι κενή,
δημιουργούμε ένα προσωρινό αρχείο με την χρήση της TempFile
    #εκκινούμε το καταγραφικό αποθηκεύοντας το βίντεο στο
προσωρινό μας αρχείο
    if writer is None:
        tempVideo = TempFile(ext=".mp4")
        writer = cv2.VideoWriter(tempVideo.path, 0x21, 30, (W,
H),
        True)

    #ελέγχουμε εάν η διάρκεια καταγραφής έχει ξεπεράσει το
χρονικό διάστημα που έχουμε ορίσει
    #εφόσον ισχύει εκχωρούμε την τωρινή ημερομηνία και ώρα
στην μεταβλητή timestamp, διακόπτουμε το καταγραφικό και
κάνουμε
    #εκκαθάρισή του, ορίζουμε την διαδρομή αποθήκευσης του
προσωρινού αρχείου και το μεταφορτώνουμε στο cloud
    #κάνουμε εκκαθάριση του προσωρινού αρχείου, και εκχωρούμε
την νέα ώρα εκκίνησης καταγραφή στην μεταβλητή startTime
    if (datetime.now() - startTime).seconds > 20.0:
            timestamp = datetime.now()
            writer.release()
            writer = None
            print("[UPLOADING FILE]" )
            path = "/{base_path}/{timestamp}.mp4".format(
            base_path="pi_sucam",
timestamp=timestamp.strftime("%A %d %B %Y %I:%M:%S%p"))
            client.files_upload(open(tempVideo.path,
"rb").read(), path)
            tempVideo.cleanup()
            startTime = datetime.now()

    #για το χρονικό διάστημα κατά το οποίο η χρονική διάρκεια
του αρχείου βίντεο που εγγράφουμε, είναι μικρότερη από αυτή
που
    #έχουμε ορίσει, εκχωρούμε κάθε ξεχωριστό καρέ στο
καταγραφικό μας
    if writer is not None:
        writer.write(frame)

    #προβολή παραθύρου ζωντανής ροής βίντεο στην οθόνη
    cv2.imshow("live feed", frame)
```

```
    #εφόσον ο χρήστης πατήσει το πλήκτρο q, γίνεται εκκαθάριση
του προσωρινού αρχείου και διακοπή του βρόγχου
    if cv2.waitKey(1) == ord("q"):
        tempVideo.cleanup()
        break

#κλείσιμο των παραθύρων που έχουν δημιουργηθεί
cv2.destroyAllWindows()
#διακοπή της ροής βίντεο
vs.stop()
```

Σύστημα παρακολούθησης με δυνατότητα ανίχνευσης κίνησης (κώδικας)

```
#motion_capture.py

#εισαγωγή των απαιτούμενων πακέτων
from picamera.array import PiRGBArray
from imutils.io import TempFile
from picamera import PiCamera
import datetime
import dropbox
import imutils
import time
import cv2

#δημιουργία της διασύνδεσης με τον αποθηκευτικό χώρο στο
Dropbox με την χρήση API key
client =
dropbox.Dropbox("Jg4WWRJ20DkAAAAAAAA5pDWwVNmfC8LBRDN4BsjZQ6y-
YRRsanF6FRhw-KHtRNT6")
print("[DROPBOX ACCOUNT LINKED SUCCESSFULLY]")

#ορισμός του Camera Module ως μέσω καταγραφής, ορισμός
ανάλυσης και ρυθμού καρέ ανά δευτερόλεπτο
camera = PiCamera()
camera.resolution = ([640,480])
camera.framerate = 16

#καταγραφή καρέ σε πίνακα RGB τριών διαστάσεων
capture = PiRGBArray(camera, size=([640,480]))

#παύση εκτέλεσης του προγράμματος για την προθέρμανση του
αισθητήρα της κάμερας
print("[CAMERA SENSOR WARMING UP]")
time.sleep(3)

#αρχικοποίηση των μεταβλητών avg και motionCounter
avg = None
motionCounter = 0

#εκχώρηση της τωρινής ημερομηνίας και ώρας στην μεταβλητή
lastUploaded
lastUploaded = datetime.datetime.now()


#εκκίνηση ατέρμων βρόγχου για την συνεχή προσπέλαση των
διαδοχικών καρέ
for f in camera.capture_continuous(capture, format="bgr",
use_video_port=True):
    frame = f.array
    timestamp = datetime.datetime.now()
    detector = "nomotion"
    #μετατροπή του καρέ σε grayscale
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

```
    #εφόσον η μεταβλητή avg είναι κενή την πρώτη φορά
εκτέλεσης του βρόγχου την αρχικοποιούμε
    #χρησιμοποιώντας το πρώτο καρές ως το αρχικό μοντέλο
φόντου που θα χρησιμοποιήσουμε
    if avg is None:
        avg = gray.copy().astype("float")
        capture.truncate(0)

    #για την καλύτερη λειτουργία του συστήματος ώστε να μπορεί
να ανταπεξέλθει στις αλλαγές του χώρου μας
    #και σε αλλαγές φωτισμού χρησιμοποιούμε ένα μεταβαλλόμενο
μοντέλο φόντου, που προκύπτει αρχικά από το άθροισμα αυτού
(avg)
    #και των νέων καρέ (gray)
    #στην συνέχεια χρησιμοποιούμε την απόλυτη διαφορά
(frameDelta) κάθε νέου καρέ (gray) και του εναλασσόμενου
φόντου (avg)
    cv2.accumulateWeighted(gray, avg, 0.5)
    frameDelta = cv2.absdiff(gray, cv2.convertScaleAbs(avg))

    #εφαρμόζουμε δυαδική κατωφλίωση στην frameDelta για την
ευκολότερη έυρεση των περιγραμμάτων
    thresh = cv2.threshold(frameDelta, 5, 255,
        cv2.THRESH_BINARY)[1]
    cnts = cv2.findContours(thresh.copy(), cv2.RETR_EXTERNAL,
        cv2.CHAIN_APPROX_SIMPLE)
    cont = imutils.grab_contours(cnts)

    #εφόσον ανιχνευτεί υπέρβαση στον όγκο της περιοχής των
περιγραμμάτων που έχουμε ορίσει
    #το σύστημά μας έχει εντοπίσει κίνηση στον χώρο
    for c in cont:
        if cv2.contourArea(c) > 5000:
            detector = "motion"

    #τοποθετούμε ημερομηνία και ώρα στο καρέ μας
    ts = timestamp.strftime("%A %d %B %Y %I:%M:%S%p")
    cv2.putText(frame, ts, (10, frame.shape[0] - 10),
cv2.FONT_HERSHEY_SIMPLEX,
        0.60, (0, 255, 255), 1)

    #εφόσον έχει ανιχνευτεί κίνηση στον χώρο εκκινούμε την
διαδικασία μεταφόρτωσης εικόνας
    if detector == "motion":
        #εάν η το διάστημα που μεσολαβεί από την προηγούμενη
μεταφόρτωση έχει υπερβεί τον χρόνο που ορίζουμε
        if (timestamp - lastUploaded).seconds >= 3:
            #αυξάνουμε τον μετρητή κίνησης
            motionCounter += 1
            #όταν ο μετρητής ξεπεράσει το όριο των καρέ που
έχουμε ορίσει, στα οποία υπάρχει κίνηση
            if motionCounter >= 8:
                #δημιουργούμε ένα προσωρινό αρχείο
                TempImage = TempFile(ext=".jpg")
```

```
                #καταγράφουμε στο προσωρινό αρχείο το καρέ στο
οποίο έχει ανιχνευτεί η κίνηση
                cv2.imwrite(TempImage.path, frame)
                print("[MOTION DETECTED] {}".format(ts))
                #ορίζουμε την διαδρομή αποθήκευσης του αρχείου
                path = "/{base_path}/{timestamp}.jpg".format(
                base_path="pi_sucam/img", timestamp=ts)
                #μεταφόρτωση του αρχείου στον αποθηκευτικό
χώρο του cloud
                client.files_upload(open(TempImage.path,
"rb").read(), path)
                #εκκαθάριση του προσωρινού αρχείου
                TempImage.cleanup()
                #ενημέρωση της χρονικής στιγμής της τελευταίας
μεταφόρτωσης
                lastUploaded = timestamp
                #εκ νέου αρχικοποίηση του μετρητή των καρέ με
κίνηση
                motionCounter = 0

    #προβολή της ζωντανής ροή στην οθόνη
    cv2.imshow("live feed", frame)
    #εφόσον ο χρήστης πληκτρολογήση "q" διακόπτεται ο βρόγχος
    if cv2.waitKey(1) == ord("q"):
        break

    #εκκαθάριση της μεταβλητής capture εν αναμονή του νέου
καρέ
    capture.truncate(0)
```

Ευχαριστίες

Σε αυτό το σημείο θα ήθελα να εκφράσω τις ευχαριστίες μου σε όλους όσους με στήριξαν στην πορεία των σπουδών μου και ιδιαίτερα στον καθηγητή κ. Ιωάννη Καλόμοιρο, επιβλέποντα της πτυχιακής μου εργασίας, για όλες τις γνώσεις που μου μετέδωσε σχετικά με το γνωστικό του αντικείμενο και μου παρουσίασε έναν καινούριο για εμένα, τομέα του κλάδου μας. Επίσης θα ήθελα να ευχαριστήσω τους γονείς μου για την υποστήριξη που μου παρείχαν και την κατανόηση που έδειξαν κατά την διάρκεια των φοιτητικών μου χρόνων.

Βιβλιογραφία

- https://www.raspberrypi.org/
- https://www.pyimagesearch.com
- https://picamera.readthedocs.io/en/release-1.10/index.html
- https://docs.opencv.org
- https://github.com/ccrisan/motioneyeos/wiki
- Διαφάνειες θεωρίας – Ψηφιακή επεξεργασία εικόνας - Δρ.Χ.Στρουθόπουλος
- Raspberry Pi Computer Vision Programming - Ashwin Pajankar
- Learning Python – Mark Lutz
- https://www.w3schools.com/python/
- https://www.python.org