ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
TECHNICAL UNIVERSITY OF CRETE

**ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ**
**&**
**ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ**

# «Χρήση μη Επανδρωμένων Εναέριων Οχημάτων για την Αναγνώριση Δομικών Φθορών σε Εφαρμογές Επιθεώρησης Κτηριακών Εγκαταστάσεων»


**Επιμέλεια εργασίας:**
Πατρίκας Αθανάσιος

**Επιστημονική Εποπτεία Εργασίας:**
Ζερβάκης Μιχάλης (Επιβλέπων)
Καλαϊτζάκης Κωνσταντίνος
Παρτσινέβελος Παναγιώτης


**Διπλωματική Εργασία**

Χανιά, Δεκέμβριος 2019

# ΕΥΧΑΡΙΣΤΙΕΣ

Στην προσπάθεια ολοκλήρωσης της παρούσας διπλωματικής εργασίας βοήθησαν πολλοί άνθρωποι. Αρχικά, θα ήθελα να ευχαριστήσω θερμά τους κ. Ζερβάκη και τον κ. Παρτσινέβελο, ως επιστημονικούς επόπτες αυτής της εργασίας, που μου πρόσφεραν την ευκαιρία να ασχοληθώ με το συγκεκριμένο θέμα, γεγονός που αποτέλεσε πολύτιμη και εποικοδομητική εμπειρία για μένα. Επίσης, ευχαριστώ θερμά τον κ. Καλαϊτζάκη, τόσο για την συμμετοχή του στην τριμελή επιτροπή, όσο και για τις σημαντικές κατευθυντήριες γραμμές που μου προσέφερε. Η βοήθεια και των τριών ήταν πολύτιμη, καθώς με τροφοδοτούσε με πολύτιμες πληροφορίες και κίνητρα. Επίσης, ένα μεγάλο «ευχαριστώ» στην κ. Ντίνα Μοιρογιώργου, καθώς και στον κ. Δημήτρη Χατζηπαράσχη, μέλος της ερευνητικής ομάδας Senselab, για τη διαρκή τους υποστήριξη, τις χρήσιμες υποδείξεις και προτάσεις τους. Χωρίς όλους αυτούς, η υλοποίηση της ερευνητικής εργασίας δεν θα ήταν εφικτή.

# ΠΕΡΙΛΗΨΗ

Η αντοχή των κτιριακών κατασκευών και η αποφυγή μηχανικών λαθών καθιστούν αναγκαία την ανάπτυξη βέλτιστων σχεδιασμών και την εφαρμογή καινοτόμων τεχνολογιών. Ο σκοπός της παρούσας εργασίας είναι διττός: από την μία πλευρά, η σχεδίαση ενός αλγορίθμου αναγνώρισης κτιριακών φθορών και από την άλλη, η ανάπτυξη του αντίστοιχου λογισμικού που θα επιτρέπει στον χρήστη να αναγνωρίζει τις φθορές αυτές. Παράλληλα, κατασκευάστηκε το συγκεκριμένο εργαλείο ώστε να μπορεί να λειτουργήσει με την χρήση Μη Επανδρωμένου Ιπτάμενου Οχήματος (UAV). Τα αποτελέσματα της εφαρμογής έδειξαν ότι η συγκεκριμένη μεθοδολογία προσέφερε επαρκή και ακριβή αναλυτικά δεδομένα για την έγκαιρη πρόβλεψη, καταγραφή και μελέτη κτιριακών φθορών. Η παρούσα εργασία έρχεται να καλύψει ένα βιβλιογραφικό κενό σε θεωρία και πράξη, αξιοποιώντας διαφορετικές μεθοδολογίες που μέχρι την παρούσα χρονική στιγμή δεν είχαν χρησιμοποιηθεί συνδυαστικά.

# ΠΕΡΙΕΧΟΜΕΝΑ

# Κεφάλαιο 1ο: Εισαγωγή

Η αντοχή των κτιριακών κατασκευών και η αποφυγή μηχανικών λαθών, καθιστούν αναγκαία και συνεχή την ανάπτυξη βέλτιστων σχεδιασμών και την εφαρμογή καινοτόμων τεχνολογιών. Η αναγνώριση και ανάλυση φθορών σε κτιριακές κατασκευές είναι ένα ζήτημα που απασχολεί σημαντικό μέρος της ερευνητικής κοινότητας στον τομέα των μηχανικών και ειδικότερα στους κλάδους των πολιτικών μηχανικών, των αρχιτεκτόνων και των ηλεκτρονικών-ηλεκτρολόγων μηχανικών. Με την εξέλιξη της τεχνολογίας γίνεται προσπάθεια να αναπτυχθούν αποτελεσματικότεροι αλγόριθμοι και μεθοδολογίες, με σκοπό την γρηγορότερη και ακριβέστερη πρόβλεψη της αντοχής μιας κτιριακής κατασκευής [1], [2]. Το γεγονός ότι ο αριθμός των ανθρώπινων θυμάτων από άστοχες ή ανεπαρκείς κατεσκευές δεν είναι καθόλα αμελητέος καθιστά το ζήτημα αυτό μείζονος σημασίας στον τομέα της μηχανικής.

Σκοπός της παρούσας εργασίας είναι αφενός η σχεδίαση ενός αλγορίθμου αναγνώρισης κτιριακών φθορών και αφετέρου η ανάπτυξη του αντίστοιχου λογισμικού που θα επιτρέπει στον χρήστη να αναγνωρίζει τα συγκεκριμένα κτιριακά ελαττώματα. Παράλληλα έγινε η κατασκευή του εργαλείου ώστε να μπορεί να λειτουργήσει με την χρήση Μη Επανδρωμένου Ιπτάμενου Οχήματος (UAV). Αυτή η τεχνολογία χρησιμοποιείται εξαιτίας της ευκολίας που προσφέρει στην παρατήρηση διαφόρων σημείων εστίασης. Επίσης διασφαλίζει την σωματική ακεραιότητα των ανθρώπων-εργαζομένων διότι αποφεύγεται η φυσική παρουσία τους σε κατασκευές επικίνδυνες ή πολύ δύσκολα προσβάσιμες [3], [4], [5].

Πιο συγκεκριμένα, το UAV που χρησιμοποιείται είναι το **Matrice m10** της εταιρίας **Dji**. Το όχημα αυτό κατέγραψε ομάδες εικόνων τόσο στο οπτικό όσο και στο θερμικό φάσμα. Το μοντέλο της οπτική κάμερας είναι το **Zenmuse X3** και αντίστοιχα για την θερμική είναι το **Tau 640** της εταιρίας **FLIR.** Ακολούθησε η επεξεργασία των εικόνων από τον αλγόριθμο αυτόματης ταυτοποίησης φθορών, ο οποίος υλοποιήθηκε στην Matlab2018b και σε υπολογιστή των ακόλουθων δυνατοτήτων: **MSI GL63 9SE**

**Intel Core i7-9750H / Ram: 16GB / 256GB SSD / 1TB HDD / GeForce RTX 2060 6GB.**

Σε αυτό το πλαίσιο κατασκευάστηκε ένα φιλικό προς στον χρήστη περιβάλλον GUI. Αναπτύχθηκε χρησιμοποιώντας το εργαλείο AppDesigner της Matlab2018b προσφέροντας στον χρήστη την ευρεία δυνατότητα της επεξεργασίας των εικόνων των οποίων θα εισάγει, καθώς και την ευκολότερη παραμετροποίηση των μεθόδων που κατασκευάστηκαν.

Η ψηφιακή επεξεργασία εικόνων χωρίζεται σε διάφορες κατηγορίες ανάλογα με την υπολογιστική φύση του εκάστοτε προβλήματος. Κάθε εικόνα αποτελείται από pixels. Αυτά τα pixels λαμβάνουν ακέραιες τιμές στο διάστημα των [0-255]. Όταν η μεθοδολογία της επεξεργασίας χρησιμοποιεί απευθείας τις παραπάνω τιμές ονομάζεται «χωρική επεξεργασία». Ωστόσο, δεν είναι πάντα δυνατόν να επιτευχθούν τα επιθυμητά αποτελέσματα μέσω της χωρικής επεξεργασίας, καθώς πολύ συχνά απαιτείται να μεταφερθούν οι τιμές της εικόνας από το πεδίο του χώρου στο πεδίο των συχνοτήτων (Fourier transform). Μέσα από την χρήση εργαλείων και μεθόδων, η μελέτη του φάσματος στο πεδίο της συχνότητας είναι πολύ πιο εύχρηστη και αποδοτική. Μετά το πέρας της επεξεργασίας του φάσματος, μπορούμε εύκολα να επαναφέρουμε την εικόνα στο πεδίο του χώρου (Fourier transform). Η Matlab2018b παρέχει σημαντικές ομάδες χρήσιμων εργαλείων για την επεξεργασία, τόσο στο πεδίο του χώρου, όσο και σε αυτό το συχνοτήτων.

Στα πλαίσια αυτής της επεξεργασίας έχουν υλοποιηθεί δύο διαφορετικές μεθόδοι: i) για την θερμική και ii) για την οπτική κάμερα. Όσον αφορά την θερμική κάμερα, οι εικόνες που συλλέχθηκαν ήταν σε μορφή .tiff. Αυτά τα μονοδιάστατα αρχεία περιέχουν όλη την πληροφορία της εικόνας, όπως και τις θερμοκρασίες του κάθε pixel. Έπειτα από κατάλληλη ανάγνωση των αρχείων, έγινε εξαγωγή και παρουσίαση αυτών, με ένα default colormap, όπου αναλύεται στην συνέχεια της εργασίας.

Ενώ, ένα πρόσφατο σώμα ερευνών επιλέγει να προσεγγίσει εκτενώς τρόπους και μεθοδολογίες προκειμένου να πετύχει την ακριβέστερη και αποδοτικότερη καταγραφή φθορών, ένα σχετικά μικρό μέρος της προσοχής έχει στραφεί στο πώς συνδυάζονται οι ήδη υπάρχουσες μεθοδολογίες για να αξιοποιηθούν τα

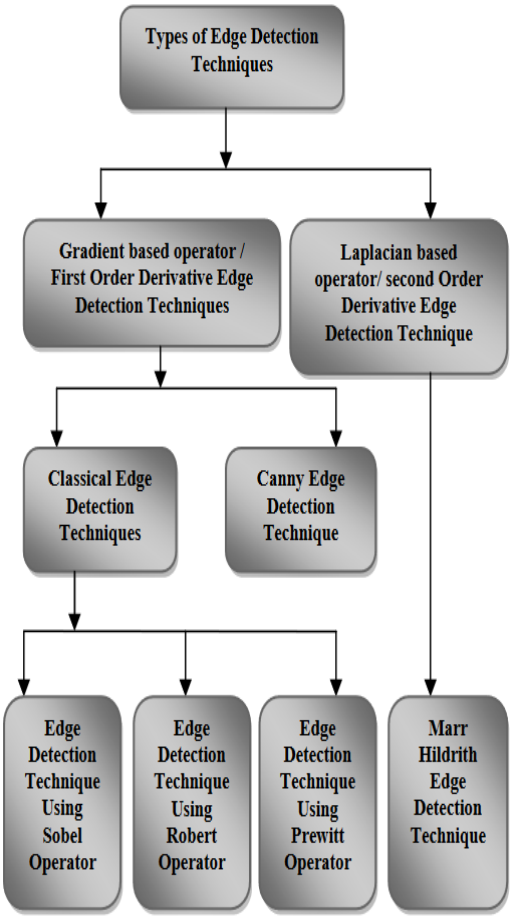
πλεονεκτήματα των εκάστοτε τεχνικών, καθιστώντας πιο αποτελεσματικό το τελικό παραγόμενο αποτέλεσμα. Σε αυτή την ευρύτερη προβληματική, η παρούσα διπλωματική εργασία επιχειρεί να καλύψει ένα βιβλιογραφικό κενό σε θεωρία και εφαρμογή, θεωρώντας ότι έχουν πολλά να αποκομισθούν από την συνδυαστική αξιοποίηση μεθοοδολογιών σε σχέση με το αναφερόμενο πρόβλημα των κτιριακών φθορών.

# Κεφάλαιο 2ο: Βιβλιογραφικό Υπόβαθρο

Σε ότι σχετίζεται με την επεξεργασία των οπτικών εικόνων, η ενδελεχής βιβλιογραφική ανασκόπηση ανέδειξε τέσσερις μεθοδολογίες-τεχνικές οι οποίες αποτελούν βασικό οδηγό σε πολλές ερευνητικές εφαρμογές.

## 2.1: Τεχνική Αναγνώρισης Ακμών (Μέθοδος *J.Dhule)*

Άρκετές ερευνητικές δημοσιεύσεις δίνουν διάφορους τρόπους προσέγγισης του θέματος. Μία από τις πιο πρόσφατες σε ακρίβεια και υπολογιστικούς πόρους είναι αυτή του Dhule [6], ο οποίος διαχωρίζοντας τους αλγορίθμους αναγνώρισης ακμών με την χρήση πρώτων παραγώγων σε σχέση με αυτή των δεύτερων παραγώγων επέλεξαν την μορφή των πρώτων.




Όπως φαίνεται στο διπλανό διάγραμμα, οι πιο συνηθισμένες (στην χρησιμότητα και αποδοτικότητα) διαδικασίες των πρώτων παραγώγων είναι οι Sobel, Prewitt, Robert, Canny [7]. Πρόκειται για φίλτρα (3x3 , 5x5 ,7x7 κλπ) ανάλογα με την έκταση της εικόνας που χρησιμοποιούνται και εφαρμόζονται συνελικτικά πάνω στην εικόνα.

Στην παραπάνω εργασία [6], χρησιμοποιείται η τεχνική του Sobel Edge Detector (3x3), εξαιτίας της μικρής ευαισθησίας στον θόρυβο. Το φίλτρο έχει την ικανότητα να εντοπίζει κάθετες και οριζόντιες

ακμές χρησιμοποιόντας την ανάλογη μάσκα (3x3) για την εφαρμογή του.

Υπάρχει η ικανότητα να οριστούν διάφορες τιμές ανάλογα με το βάθος των ακμών που χρειάζεται να καταγραφούν. Στην συγκεκριμένη όμως εργασία [6], χρησιμοποιείται το συνηθέστερο φίλτρο με την παρακάτω μορφή:

Sobel Operators

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & \boxed{0} & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} -1 & -2 & -1 \\ 0 & \boxed{0} & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

direction of gradients

Η αναγνώριση των ακμών επιτυγχάνεται υπολογίζοντας τις πρώτες παραγώγους της τιμής του εκάστοτε σημείου (pixel: I[x,y]) :

$$\overline{\nabla I} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial I(x,y)}{\partial x} \\ \frac{\partial I(x,y)}{\partial y} \end{bmatrix}$$

Υπολογίζοντας το μέτρο και την γωνία του παραπάνω διανύσματος,

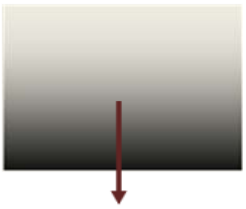
$$\nabla I = \left\| \overline{\nabla I} \right\| = \sqrt{\left[ G_x^2 + G_y^2 \right]} \qquad \theta(x,y) = \tan^{-1}\left( \frac{G_x}{G_y} \right)$$

προκύπτουν τα αποτελέσματα για τις οριζόντιες ακμές

$$\left\| \overline{\nabla I} \right\| = G_y$$

$$\theta(x,y) = -\frac{\pi}{2}$$

Horizontal Edge



καθώς και για τις κάθετες.

$$\left\| \overline{\nabla I} \right\| = G_x$$

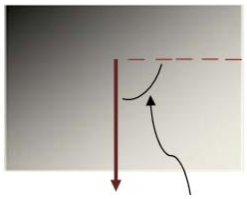
$$\theta(x,y) = 0$$

Vertical Edge

Για τις ενδιάμεσες γωνίες ισχύει ότι:

$$\|\nabla I\| = \sqrt{\left[G_x^2 + G_y^2\right]}$$

$$\theta(x, y) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$



Όπως προκύπτει τα τρία βασικά μέρη για την εφαρμογή της αναγνώρισης ακμών με την μέθοδο που παρουσιάστηκε είναι:

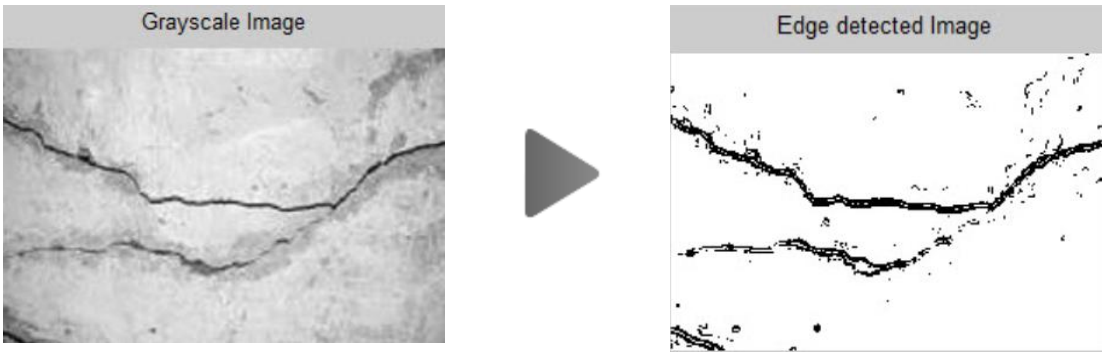
- Η μείωση θορύβου (Noise Reduction)
- Η βελτιστοποίηση ακμών (Edge Enhancement)
- Ο εντοπισμός ακμών (Edge Localization)

*Πλεονεκτήματα εφαρμογής:*

Εξαιτίας της απλότητας και της μικρής απαίτησης σε υπολογιστική ισχύ, η παραπάνω εφαρμογή αποδεικνύεται ταχύτερη σε σύγκριση με άλλες. Γι' αυτό το λόγο μπορεί να φανεί χρήσιμο σε εφαρμογές που χρειάζονται χαμηλούς χρόνους επεξεργασίας, όπως είναι για παράδειγμα real-time εφαρμογές.

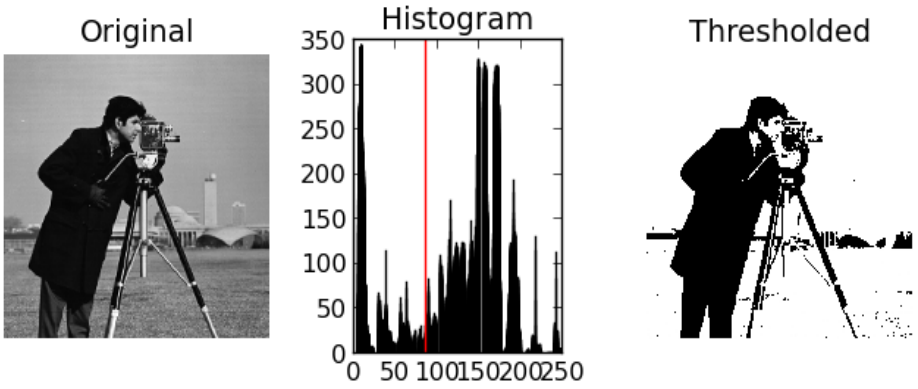
*Μειονεκτήματα εφαρμογής:*

Εύκολα μπορεί να παρατηρηθεί ότι η τελική εικόνα είναι πολύ ευαίσθητη σε μικρές διαφορές χρωμάτων ή διαφοράς έντασης που μπορεί να εμφανίζονται σε τοίχους με αποτέλεσμα να αναγνωρίζονται λανθασμένα στοιχεία ως φθορές (έντονος θόρυβος). Παρακάτω παρατηρείται η εξίσου λανθασμένη αναγνώριση στοιχείων, όπως αυτά εφαρμόστηκαν σε πιο απαιτητικές εικόνες που εισάγαμε. Όπως εύκολα διαπιστώνεται παραμένει αρκετά ανακριβής ως προτεινόμενη εφαρμογή.




## 2.2: Αναγνώριση Ρωγμών με την Τεχνική Otsu (Μέθοδος A. Talab)

Η μέθοδος αυτή διαχωρίζει τα στοιχεία της εικόνας σε δύο κομμάτια: τα στοιχεία τα οποία είναι στο υπόβαθρο (background pixels) και αυτά τα οποία είναι υποψήφια για pixel φθοράς (foreground pixels). Στόχος της μεθόδου είναι η ελαχιστοποίηση της διασποράς του πεδίου μεταξύ αυτών των δύο περιοχών, έτσι ώστε να γίνει ευκολότερη η διαδικασία αναγνώρισης φθορών. Στις παρακάτω εικόνες, αριστερά της κόκκινης γραμμής έχουμε το background της μεθόδου και δεξιά της το foreground.




Η κόκκινη γραμμή η οποία φαίνεται στο ιστόγραμμα είναι η τιμή του Otsu threshold το οποίο υπολογίζεται ως εξής:

$$\sigma_w^2(T) = \omega_0(T)\sigma_0^2(T) + \omega_1(T)\sigma_1^2(T)$$

Τα ω0 και ω1 είναι οι πιθανότητες εμφάνισης των δύο περιοχών που διαχωρίζονται απο το Threshold T. Τα σ0 και σ1 είναι οι διασπορές των δύο περιοχών αντίστοιχα. Οι πιθανότητες ω0 και ω1 υπολογίζονται ως εξής:

$$\omega_0(T) = \sum_{i=0}^{T-1} p(i)\,;\ \omega_1(T) = \sum_{i=T}^{L-1} p(i)$$

Έπειτα υπολογίζονται οι μέσες τιμές :

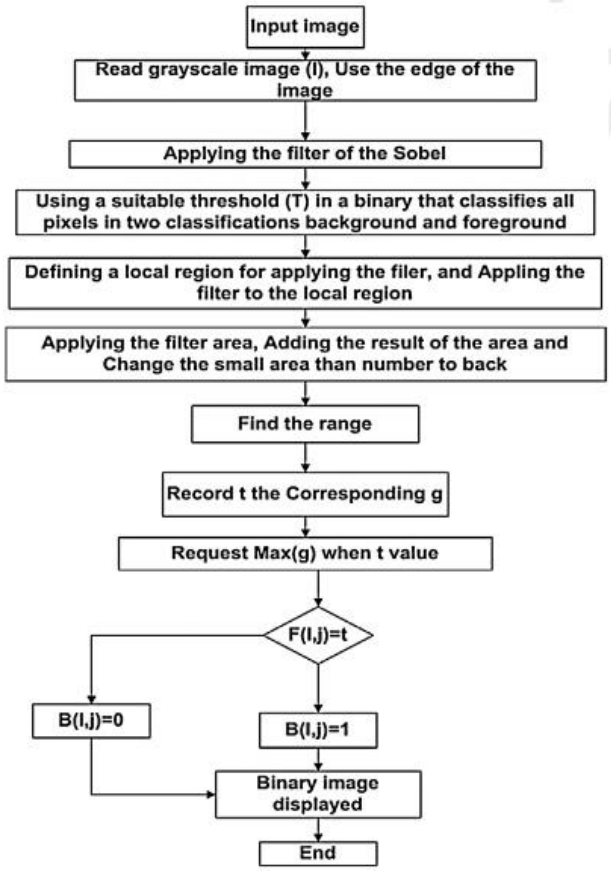
$$\mu_0(T) = \sum_{i=0}^{T-1} ip(i)/\omega_0\,;\ \mu_1(T) = \sum_{i=T}^{L-1} ip(i)/\omega_1$$

έτσι ώστε να βρεθούν οι τιμές των σ0 και σ1, από τους τύπους:

$$\sigma_0^2(T) = \sum_{i=0}^{T-1}[i - \mu_0(T)]^2 \frac{p(i)}{\omega_0(T)}$$
$$\sigma_1^2(T) = \sum_{i=T}^{L-1}[i - \mu_1(T)]^2 \frac{p(i)}{\omega_1(T)}$$

Στο σημείο όπου μεγιστοποιείται η διασπορά $\sigma_W{}^2$(T) έχουμε την τιμή του Otsu Threshold T.

Στηριζόμενος σε αυτή την μεθοδολογία, οι Talab et al. [8] προχώρησαν στην αναγνώριση φθορών όπου ως αποτέλεσμα έχει την μετατροπή από το RGB μοντέλο των εικόνων σε δυαδικές (binary) εικόνες. Το διάγραμμα της μεθοδολογίας, όπως αυτό παρουσιάστηκε, φαίνεται στην διπλανή εικόνα.



Παρατηρείται ότι κύριο σημείο της μεθοδολογίας είναι ο αυτόματος ορισμός του Threshold (T) για τον διαχωρισμό των pixel σε background και foreground. Ο ορισμός επιτυγχάνεται μέσω της μεθόδου του Otsu. Κομβικό ρόλο διαδραματίζει και η χρήση του Sobel φίλτρου, έπειτα από την αναγνώριση ακμών της εικόνας. Επίσης, χρησιμοποιούνται συναρτήσεις για τον καθαρισμό της εικόνας (image cleaning) εντοπίζοντας και εξαλείφοντας τα αντικείμενα όπου έχουν άθροισμα συνδεδεμένων pixel κάτω των 30. Ο έλεγχος της αναγνώρισης των φθορών γίνεται pixel by pixel.

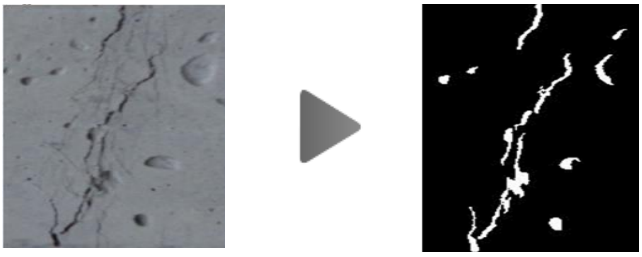
*Πλεονεκτήματα Εφαρμογής:*

Πρόκειται σαφώς για μία πιο εξελιγμένη μέθοδο από αυτήν του Dhule [6] αφού γίνεται καλύτερη σε ακρίβεια αναγνώριση σε pixel φθορών διότι χρησιμοποιούνται οι μεθοδολογίες του Otsu σε συνδιασμό με μορφολογικές συναρτήσεις. Αποτελεί μία εξίσου γρήγορη μεθοδολογία λόγω χαμηλού υπολογιστικού κόστους. Η συγκεκριμένη εφαρμογή είναι αρκετά χρήσιμη σε μελέτη φθορών ελεγχόμενων/εργαστηριακών περιβαλλόντων, καθώς ανταποκρίνεται ικανοποιητικά σε μικρές περιοχές παρακολούθησης.

*Μειονεκτήματα Εφαρμογής:*

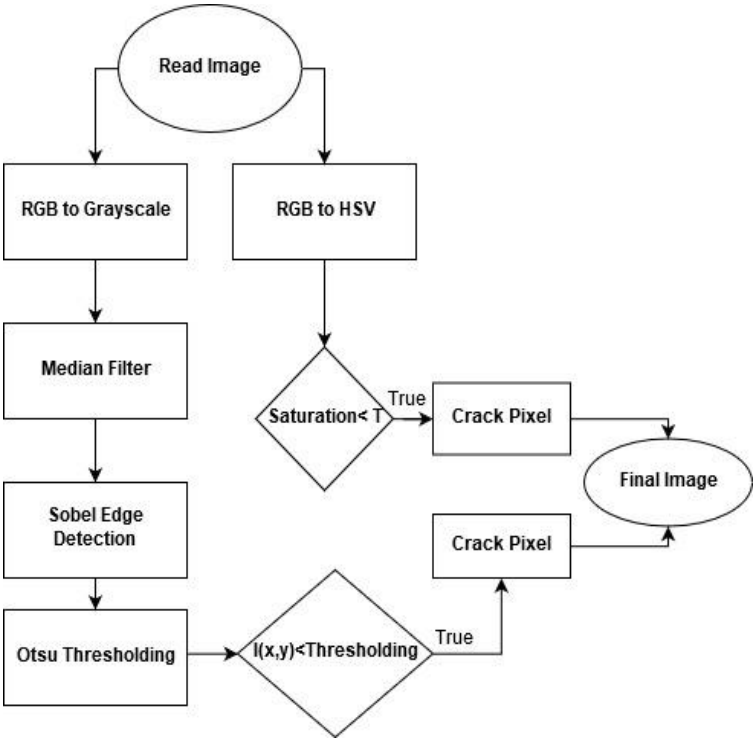
Όπως παρατηρείται στις εικόνες που εξετάσαμε, εξαιτίας της ευαισθησίας της εφαρμογής στον θορύβο, η μεθοδολογία αυτή δεν κρίνεται τόσο ικανοποιητική σε σχέση με την απόδοση της στο σύνολο των εικόνων που εφαρμόζεται. Λόγω της ευαισθησίας, ο αλγόριθμός αναγνωρίζει σφάλματα ακόμα και σε πολύ μικρές εκδροές, γεγονός το οποίο μειώνει την αξιοπιστία του σε εικόνες υψηλής ανάλυσης. Αναγνωρίζει κυλίδες, σημάδια και άλλα κατάλοιπα που μπορεί να υπάρχουν σε τοίχους, χωρίς ωστόσο να μπορεί να τα διαχωρίσει από τις κύριες φθορές (crack pixels)

Τα αποτελέσματα όπως παρουσιάστηκαν στην συγκεκριμένη έρευνα:



## 2.3: Αυτοματοποιημένη Αναγνώριση Ρωγμών (Μέθοδος S.Dorafshan)

Εξετάζοντας όλα τα παραπάνω, ο Dorafshan ανέπτυξε το 2016 την δικιά του μεθοδολογία για την αποδοτικότερη επίλυση του θέματος [9]. Εκτός από τα προαναφερθέντα βήματα, πρόσθεσε μορφολογικές διαδικασίες για τον περαιτέρω έλεγχο των επιθυμητών περιοχών, καθώς και για τον έλεγχο των pixel έπειτα από την μετατροπή της έγχρωμης εικόνας από RGB σε HSV μοντέλο.



Αναλυτικά ο αλγόριθμος, όπως φαίνεται από το παραπάνω διάγραμμα, έχει τα εξής βήματα:

- Μετατροπή της εικόνας από RGB σε Grayscale.
- Χρήση ένος median filter για την εξομάλυνση της εικόνας.
- Χρήση του Sobel edge detection για την εύρεση των ακμών.
- Χρήση των εξισώσεων του Otsu για τον ορισμό της σταθεράς T.
- Στοιχεία που ενώνονται για λιγότερο από 200 pixels αναγνωρίζονται και αφαιρούνται.
- Στοιχεία που προσανατολίζονται στις 0,90,-90 μοίρες αναγνωρίζονται και αφαιρούνται.
- Χρήση της μορφολογικής διαδικασίας “majority” για να ενώσει τα εναπομείναντα στοιχεία και να γεμίζει τα κενά στις ακμές.
- Στοιχεία με 50 ή λιγότερα pixel αναγνωρίζονται και αφαιρούνται.
- Μετατροπή της RGB εικόνας σε HSV εικόνα.
- Τα ενοποιημένα pixel αποθηκεύονται ως υποψήφια πιθανά pixel φθορών.
- Ορίζεται μια καινούργια σταθερά για την τιμή του S (saturation) όπου:

T = min(S)+std(S) όπου min(S) είναι η ελάχιστη τιμή όλων των S τιμών και std(S) είναι η απόκλιση του S.

- Έλεγχος για το αν το S του παρώντος pixel είναι μικρότερο της σταθεράς T που ορίστηκε παραπάνω. Σε αυτήν την περίπτωση, το στοιχείο αφαιρείται (ως non-crack) αλλιώς εισέρχεται ως ακμή της δυαδικής εικόνας (binary)

Η συγεκριμένη μεθοδολογία του Dorafshan [9] χρησιμοποιεί το 20% του Otsu Threshold για την μετατροπή της δυαδικής (binary) εικόνας σύμφωνα με τον παρακάτω τύπο:

$$B(x,y) = \begin{cases} 0 & if\ I(x,y) < 0.2T \\ 1 & if\ I(x,y) \geq 0.2T \end{cases}$$

όπου B(x,y) είναι η εκάστοτε τιμή του pixel σε δυαδική μορφή και I(x,y) είναι η τιμή του pixel στο σημείο (x,y) ύστερα από την εφαρμογή του Sobel Edge Detector.

Το αποτέλεσμα της παραπάνω διεργασίας αρκετά συχνά εμφανίζει μικρά στοιχεία τα οποία έχουν λανθασμένα αναγνωριστεί ως στοιχεία φθοράς. Για την απαλοιφή αυτών των αντικειμένων ακολουθείται μία μέθοδος όπου μετράει τον αριθμό των συνδεδεμένων pixels και αν αυτά είναι κάτω των 200, τα αφαιρεί από την δυαδική εικόνα, όπου ουσιαστικά τα προσθέτει στο background της εικόνας.

Επίσης, λόγω του ότι με την παραπάνω διαδικασία μπορεί να εξαλειφθούνε και χρήσιμα pixel ή να υπάρχουνε μικρά κενά ανάμεσα στις αναγνωρίσιμες ακμές χρησιμοποιεί μορφολογικές διεργασίες με σκοπό να ενοποιήσουν τις ακμές που παραμένουν. Μια τέτοια διαδικασία ονομάζεται “majority” (υποστηρίζεται από την Matlab) και λειτουργία της είναι να καλύψει τα κενά μεταξύ των ασυνεχειών. Συγκεκριμένα, είναι ένα φίλτρο 3x3 που εφαρμόζεται με την εξής λογική: θέτει το κεντρικό pixel ίσο με 1, εάν 5 ή περισσότερα γειτονικά pixel είναι εξίσου 1.Διαφορετικά, το θέτει ίσο με 0. Η διαδικασία συνεχίζεται έως ότου η μάσκα εφαρμοστεί σε όλα τα στοιχεία της εικόνας. Μάλιστα για την αποφυγή λανθασμένων ενώσεων εφαρμόζει και πάλι τον αλγόριθμο αναγνώρισης και απαλοιφής των συνδεδεμένων pixel όπου έχουν άθροισμα μικρότερο του 50.

*Πλεονεκτήματα:*

Προκύπτει ότι η μεθοδολογία του Dorafshan [9] είναι αρκετά πιο αποδοτική από αυτές του Dhule [6] και Talab [8]. Αυτό οφείλεται στις παρακάτω παραμέτρους: στον ορθότερο υπολογισμό του Otsu Threshold, στην αύξηση και εύρεση αποδοτικότερων μορφολογικών διαδικασιών, καθώς και στον έλεγχο της HSV εικόνα.

*Μειονεκτήματα:*

Παρατηρείται αδυναμία στην εξάλειψη του θορύβου σε υψηλής ανάλυσης εικόνες, καθώς και μεγάλη χωρητικότητα στην χρήση προσωρινής μνήμης (RAM). Αυτό προκύπτει από την ανάγκη της συγκεκριμένης μεθοδολογίας να αποθηκεύει τα περισσότερα δεδομένα των επεξεργασμένων εικόνων για τον έλεγχο τους. Ακόμα παρατηρείται μεγάλη αύξηση στον χρόνο εκτέλεσης του προγράμματος καθώς υπάρχουν αρκετοί κόμβοι επανάληψης για τον έλεγχο των στοιχείων pixel by pixel.

## 2.4: Αναγνώριση Φθορών Μέσω Εξέλιξης των Διαδικασιών της Τεχνικής Otsu (Μέθοδος N.D.Hoang)

Μία διαφορετική προσέγγιση προτάθηκε το 2018, από τον Nhat-Duc Hoang [10]. Δεδομένου ότι η μετατροπή των εικόνων (από RGB ή Grayscale) σε δυαδικές εικόνες (Binary) είναι ευρέως διαδεδομένη στην αναγνώριση κειμένων και στις βιο-ιατρικές εφαρμογές, θεωρεί την χρήση τους απαραίτητη για την εύρεση κτιριακών φθορών [11] . Επίσης, αναφέρει ότι η απλή χρήση των εξισώσεων του Otsu για την μετατροπή αυτή, είναι μη αποδοτική λόγω του ότι επηρεάζεται έντονα από την ανάλυση της εικόνας, την χαμηλή αντίθεση των στοιχείων εικόνας (low contrast images), την έντονη διαφοροποίηση της φωτεινότητας και την ύπαρξη θορύβου[12].

Γι αυτό τον λόγο, ο Hoang χρησιμοποιεί την μεθοδολογία του Otsu, εξαιτίας του χαμηλού υπολογιστικού κόστους που προσφέρει, εφόσον όμως προηγουμένως την επεξεργαστεί. Την προεργασία αυτή την ονομάζει “Min-Max Gray Level Discrimination” (MG2LD). Στόχο έχει την αποφυγή των λανθασμένα αναγνωρίσιμων ακμών, όπως του Dorafshan, αλλά με διαφορετική μεθοδολογία και αποδοτικότερα αποτελέσματα.

Η κύρια ιδέα της MG2LD μεθοδολογίας είναι να αυξήσει στην grayscale εικόνα την ένταση των non-crack pixels και να μείωσει αυτή των crack-pixels. Έτσι, μετά την βελτιστοποίηση της εικόνας (image enhancement) τα crack-pixels εμφανίζονται σκοτεινότερα, ενώ τα non-crack φωτεινότερα. Αυτό έχει ως αποτέλεσμα να διαχωριστούν ευκρινέστερα οι περιοχές που αποτελούνται από φθορές (foreground regions) από τις περιοχές των «υγειών» περιοχών (background regions)

Η λογική που χρησιμοποιεί είναι η εξής:

Έστω I0(m,n) είναι η τιμή του pixel στο σημείο (m,n) της grayscale εικόνας

Εφαρμόζοντας τις παρακάτω εξισώσεις :

$$I_{\Lambda}(m,n) = \min\left(I_0\_\max,\ I_0(m,n)\cdot R_{\Lambda}\right) \quad \text{if } I_0(m,n) > I_0\_\min + \tau\cdot\left(I_0\_\max - I_0\_\min\right)$$

$$I_{\Lambda}(m,n) = \max\left(I_0\_\min,\ I_0(m,n)\cdot R_{\Lambda}^{-1}\right) \quad \text{if } I_0(m,n) \le I_0\_\min + \tau\cdot\left(I_0\_\max - I_0\_\min\right)$$

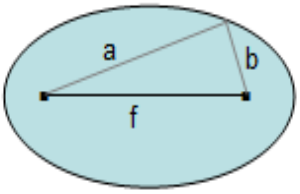
λαμβάνεται η νέα grayscale εικόνα Ia με το καινούργιο ιστόγραμμα. Οι μεταβλητές Ra και τ , είναι σταθερές και τις οριστικοποιεί εμπειρικά στις τιμές των 1.1 και 0.5 αντίστοιχα.

Έπειτα από την εφαρμογή του Otsu binarization , ακολουθείται μία διαδικασία καθαρισμού της εικόνας όπου τα pixel θορύβου και τα non-crack pixel εντοπίζονται και αφαιρούνται. Εφαρμόζεται μία αναλογία δείκτη αξόνων (axis ratio index, ARI), όπου αποτελείται από την αναλογία του μεγαλύτερου άξονα ως προς τον μικρότερο άξονα ενός αντικειμένου:

$$\mathrm{ARI} = \frac{L_M}{L_N},$$

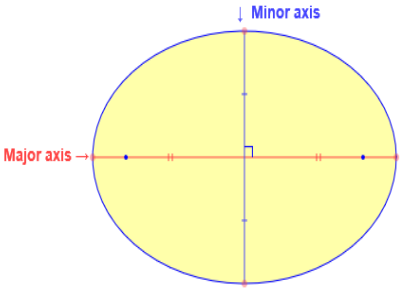
με $L_M$ και $L_N$ να είναι το μήκος του μεγαλύτερου και το μήκος του μικρότερου άξονα αντίστοιχα.

$$minor\ axis = \sqrt{(a+b)^2 - f^2}$$




$$major\ axis = a+b$$

Υπογραμμίζεται ότι το ARI τείνει ως προς το 0 για αντικείμενα τα οποία σχηματίζουν ευθείες ενώ τείνει προς το 1 για κυκλικά αντικείμενα.





Εμπειρικά, ο Hoang ορίζει την σταθερά ARI=3 για τον διαχωρισμό των crack από τα non-crack αντικείμενα. Αφότου έχουν αναγνωριστεί όλα τα αντικείμενα, ακολουθεί η ανάλυση των στοιχείων που έχουν βρεθεί ως crack-objects.

Συγκεκριμένα η διαδικασία αποτελείται από δύο μέρη. Το πρώτο μέρος είναι η εξαγωγή των ορίων των αντικειμένων (image boundary extraction) το οποίο στηρίζεται πάνω στους αλγορίθμους αναγνώρισης ακμών (edge detection algorithms) και χρησιμοποιείται για τον υπολογισμό της περιμέτρου, του πάχους, του μήκους και του πλάτους των φθορών. Ο υπολογισμός της περιμέτρου υπολογίζεται μέσω του αθροίσματος των pixel που βρίσκονται στα όρια (edges) του αντικειμένου. Το πάχος

της φθοράς είναι αντίστοιχα το άθροισμα των pixel του αντικειμένου παραλείποντας αυτά που βρίσκονται στα όρια.

Το δεύτερο μέρος είναι ο υπολογισμός του προσανατολισμού της φθοράς (object orientation) αφότου το αντικείμενο έχει υποστεί λέπτυνση (image thinning or skeletonization). Το πρόβλημα αυτό ανάγεται ως πρόβλημα απλής γραμμικής εξίσωσης, όπου ως ανεξάρτητη μεταβλητή θεωρείται η τιμή του pixel στον άξονα των x και ως εξαρτημένη μεταβλητή η τιμή του pixel στον άξονα των y. Το αποτέλεσμα της πρώτης παραγώγου της παραπάνω εξίσωσης αποτελεί και τον προσανατολισμό του αντικειμένου.

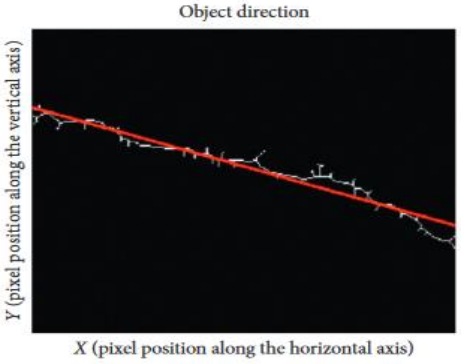
Για να υπολογιστεί το μήκος της φθοράς χρησιμοποιείται η τιμή του προσανατολισμού, όπου εδώ υπάρχουν δύο περιπτώσεις. Πρώτον, ο προσανατολισμός να είναι μικρότερος των 45° όπου έχουμε φθορές που αναπτύσσονται οριζόντια και δεύτερον, να είναι μεγαλύτερος των 45° όπου έχουμε φθορές όπου αναπτύσσονται κάθετα.

Οι εξισώσεις για να εκτιμηθούν αυτές οι δύο περιπτώσεις:

$$\text{Case 1: } W(s) = L_v(s) \cdot \sin(90 - \alpha),$$
$$\text{Case 2: } W(s) = L_h(s) \cdot \sin(\alpha),$$

όπου $L_V$ και $L_h$ είναι ο αριθμός των pixel που μετρήθηκαν ως κάθετα και οριζόντια στοιχεία, ενώ α είναι η τιμή του προσανατολισμού του εκάστου s αντικειμένου. Το μήκος του αντικειμένου μπορεί πλέον εύκολα να υπολογιστεί από τον τύπο:

$$L_c = \frac{\left(\text{Per} - 2 \cdot W_{\text{Avg}}\right)}{2}$$



*Πλεονεκτήματα:*

Η μεθοδολογία του Hoang προσφέρει αποδοτικότερα αποτελέσματα σε σύγκριση με τις προηγούμενες τρείς, διοτι στηρίζεται στην προεπεξεργασία του ιστογράμματος των εικόνων πριν την εφαρμογή του Otsu. Ως συνέπεια αυτού προκύπτει μία πιο άμεση και γρήγορη αναγνώριση των κτιριακών φθορών. Επίσης, η συγκεκριμένη μεθοδολογία είναι λιγότερη ευαίσθητη σε πιθανό θόρυβο που μπορεί να εμφανιστεί, καθώς δεν χρησιμοποιούνται φίλτρα εξομάλυνσης (πχ Sobel, Canny, Prewitt).

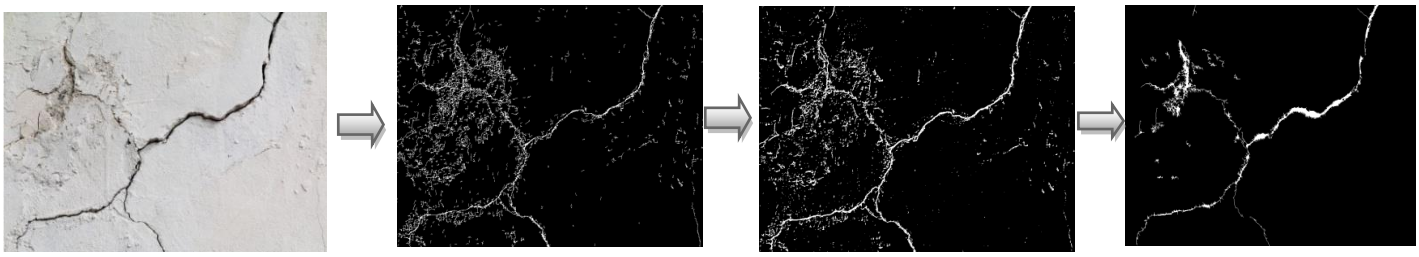
*Μειονεκτήματα:*

Κάποια από τα μειονεκτήματα που μπορούμε να εντοπίσουμε είναι ότι οι σταθερές της μεθοδολογίας M2GLD (Ra και τ), όπως και της μορφολογικής διαδικασίας (ARI) ορίζονται χειροκίνητα. Αυτό είναι κάτι που καθιστά δύσκολο την αναγνώριση φθορών από διαφορετικές φωτογραφίες και εικόνες, εφόσον αυτές θα πρέπει συνεχώς να διαμορφώνονται από τον χρήστη ανάλογα με την ανάλυση, την φωτεινότητα και τις λεπτομέρειες όπου έχει η κάθε μία.

Από τις παρακάτω εικόνες, μπορεί να παρατηρηθεί η αύξηση της αποδοτικότητας του αλγορίθμου του Hoang.

## 2.5: Σύγκριση Μεθοδολογιών

Έπειτα από την κατασκευή των αλγορίθμων σύμφωνα με τις ανωτέρω μεθοδολογίες προκύπτει σαφέστατα η αύξηση της αποδοτικότητας της μεθόδου του Hoang. Η μεθοδολογία αυτή προσπερνώντας αρκετά στάδια τα όποια ήταν αναγκαία για τους προηγούμενους αλγορίθμους καταφέρνει να αναγνωρίζει καλύτερα της κτιριακές φθορές.



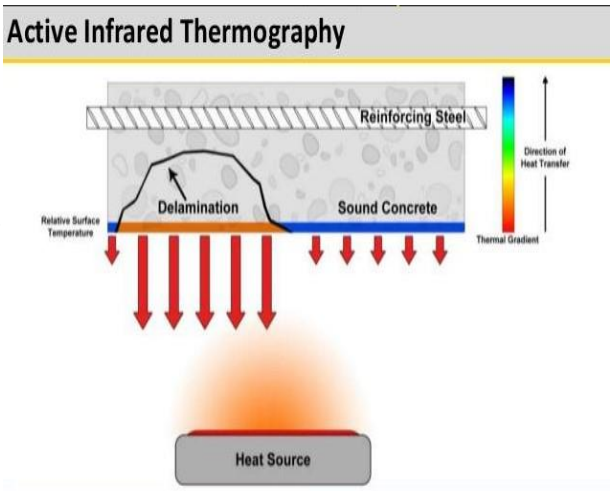
Αρχικά, όπως παρατηρείται, εξαλείφει αποτελεσματικότερα τον θόρυβο που προκύπτει από τις διάφορες επεξεργασίες αναγνώρισης, ενώ παράλληλα, αναγνωρίζει σχεδόν όλα τα pixel της εκάστοτε φθοράς. Αυτό συμβαίνει διότι οι μορφολογικές διαδικασίες που εντάσσονται στην συγκεκριμένη έρευνα στοχεύουν καλύτερα στην ένωση των κοινών αυτών αναγνωρίσιμων αντικειμένων.

Κάτι ακόμα που αξίζει να αναφερθεί είναι η ταχύτητα των παραπάνω αλγορίθμων. Σύμφωνα με τις μετρήσεις που καταγράφηκαν στον ακόλουθο πίνακα, παρατηρούμε ότι οι μέθοδοι των Dhule και Talab είναι ταχύτερες από αυτήν του Dorafshan. Όπως όμως προαναφέραμε, έχουν μεγάλη διαφορά ως προς την ακρίβεια του τελικού αποτελέσματος και κρίνονται ως λιγότερο αποδοτικές. Η μέθοδος του Hoang είναι ιδανικότερη τόσο ως προς την αποδοτικότητα, όσο και στον χρόνο εκτέλεσης.

| | |
|---|---|
| Dhule Method | 0.103422 seconds |
| Talab Method | 0.114196 seconds |
| Dorafshan Method | 2.383857 seconds |
| Hoang Method | 0.403784 seconds |

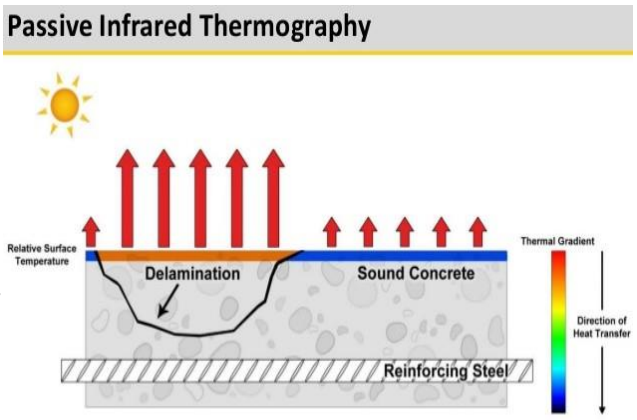
## 2.6: Επεξεργασία Θερμικής Εικόνας

Υπάρχουν διάφοροι τρόποι για την επεξεργασία των θερμικών δεδομένων [13] [14] [15]. Όλοι όμως μπορούν να διαχωριστούν σε δύο κατηγορίες ανάλογα με τον τρόπο με τον οποίο θερμαίνεται η επιφάνεια την οποία μελετάμε.




Συγκεκριμένα, ο πρώτος τρόπος είναι η Active Infrared Thermography όπου, όπως φαίνεται και στο διπλανό σχήμα, είναι η μελέτη στην οποία το αντικείμενο που καταγράφεται θερμαίνεται από πηγή ελεγχόμενη άπο τον παρατηρητή [16] [17]. Σε αυτές τις περιπτώσεις τα αποτελέσματα είναι πιο ακριβή, αλλά λόγω του ότι απαιτούν εργαστηριακές συνθήκες για τον έλεγχο της πηγής, είναι αδύνατο να προσαρμοστούν στο προς μελέτη εξωτερικό περιβάλλον.

Για τον λόγο αυτό χρησιμοποιούμε τον δεύτερο τρόπο, δηλαδή τις τεχνικές και εφαρμογές που αναλύουν την Passive Infrared Thermography. Όπως παρουσιάζεται και στο διπλανό σχήμα, το αντικείμενο παρακολούθησης θερμαίνεται από φυσικές πηγές, όπως ο ήλιος. Οι μελέτες που παρουσιάζουν και προσπαθούν να αναλύσουν την Passive Infrared Thermography, ακολουθούν μοντέλα παρόμοια με τα οποία μπορούμε να δούμε και στις κλασικές οπτικές επεξεργασίες εικόνας [18].



Πρόσφατα, οι Vermani και Ojha [7] ανέλυσαν τις μεθόδους αναγνώρισης ακμών, μέσω φίλτρων, πάνω στις θερμικές εικόνες. Χρησιμοποιώντας μεθοδολογίες που ακολουθούνται στις αντίστοιχες οπτικές, διαχωρίζουν τις μεθοδολογίες Roberts Cross-Gradient Operator, Sobel Operator, Prewitt Operator ως φίλτρα πρώτων παραγώγων και τις Laplacian of Gaussain και Canny Edge Detection ως φίλτρα δεύτερων παραγώγων.

Το 2017, ο Pithadiya [19] χρησιμοποιεί την Laplacian Of Gaussian συνάρτηση ως ασφαλέστερη και ακριβέστερη για την αναγνώριση ακμών από θερμικές εικόνες. Ο λόγος που επιλέγει την συγκεκριμένη μεθοδολογία είναι γιατί στοχεύει να μελετήσει ακραίες μεταβολές στις εντάσεις τιμών σε περιπτώσεις πολύ κοντινών περιοχών.

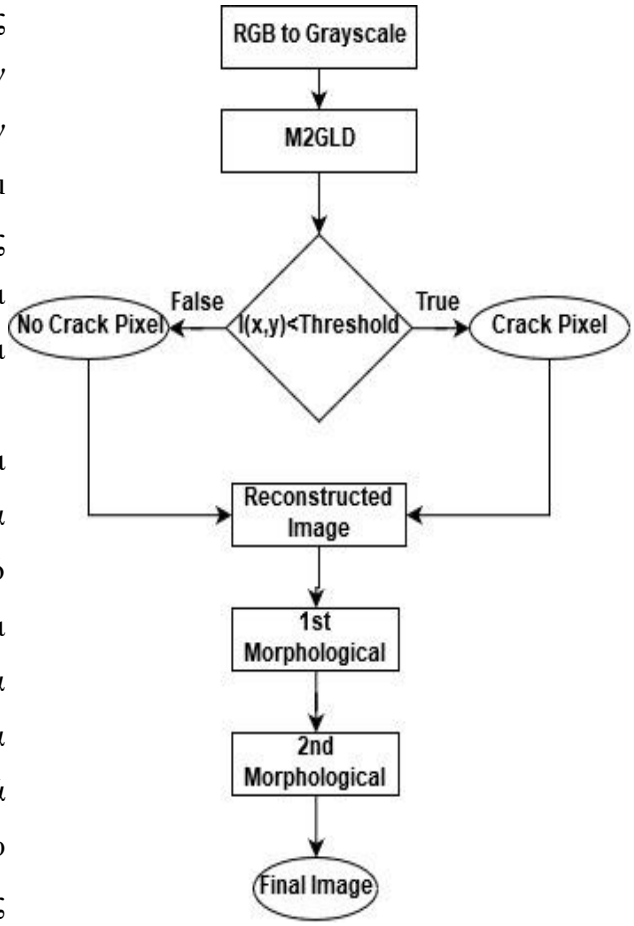
Τέλος, η μελέτη των Duarte et al. [20] πάνω στις βίο-ιατρικές τεχνολογίες, έκαναν σαφή το μεγάλο πλεονέκτημα της συγκεκριμενοποίησης των περιοχών που αναλύονται μέσω εικόνων υπέρυθρων ακτινοβολιών. Μειώνοντας τον χώρο ανάλυσης, τα αποτελέσματα που προκύπτουν είναι σαφώς πιο ακριβή. Τον χώρο αυτόν , ROI (Region of Interest), θα αναλύσουμε και σε επόμενα κεφάλαια είναι λειτουργία-κλειδί για την ακριβέστερη ανάλυση των εικόνων. Την συγκεκριμένη διαδικασία την χρησιμοποιούμε τόσο στις θερμικές όσο και στις οπτικές εικόνες.

# Κεφάλαιο 3ο: Προτεινόμενη Μεθοδολογία

## 3.1: Αλγόριθμος Επεξεργασίας Οπτικής Εικόνας

Στην συγκεκριμένη διπλωματική εργασία παρουσιάζεται μια προσέγγιση η οποία στηρίζεται στις παραπάνω μεθοδολογίες, επιτυγχάνοντας ακριβέστερα αποτελέσματα στις περισσότερες περιπτώσεις. Προκειμένου να μειωθεί το υπολογιστικό κόστος (πχ χρήση λιγότερης μνήμης RAM, ταχύτερη απόδοση) με σκοπό να μπορέσει το σύστημά μας να λειτουργεί σε real-time χρόνους, προβήκαμε σε απλοποίηση της μεθοδολογίας.

Τα στάδια της υλοποίησης της οπτικής επεξεργασίας προβάλλονται στο διπλανό διάγραμμα. Όπως διακρίνεται, χρησιμοποιούμε τον M2GLD αλγόριθμο για την αποδοτικότερη ανάλυση του Otsu Thresholding. Οι σταθερές της παραπάνω μεθοδολογίας ορίζονται εμπειρικά στις τιμές Ra=2 και τ=0.5.




Επίσης, χρησιμοποιούνται δύο μορφολογικές διαδικασίες για τον καθαρισμό της εικόνας από λανθασμένα pixel. Παρατηρείται ότι η συγκεκριμένη μεθοδολογία είναι παρόμοια με την μεθοδολογία του Hoang[10] με την διαφορά τους να βρίσκεται στον τρόπο που συντάσσονται οι μορφολογικές διαδικασίες. Λόγω του ότι δεν χρησιμοποιούνται συνελικτικά φίλτρα, ο έλεγχος της αναγνώρισης των pixel μπορεί να γίνει απευθείας στις εικόνες, γεγονός που μειώνει

δραστικά τον χρόνο εκτέλεσης του αλγορίθμου, καθώς και την χρήση RAM του υπολογιστικού συστήματος.

## 3.2: *Υλοποίηση Αλγορίθμου Επεξεργασίας Οπτικής Εικόνας*

Για να εξαλείφει ο θόρυβος από την εικόνα, χρησιμοποιήθηκαν μορφολογικές διαδικασίες οι οποίες έχουν ως σκοπό να καθαρίσουν τα pixel που εσφαλμένα αναγνωρίζονται ως pixel φθοράς. Για τον σκοπό αυτό αναπτύξαμε δύο ειδών μορφολογικές διαδικασίες [21].

Πρώτον, χρησιμοποιώντας την συνάρτηση "regionprops" (BW, ιδιότητα) η οποία επιστρέφει τις μετρήσεις για το σύνολο των ιδιοτήτων που καθορίζονται από το δεύτερο όρισμα για κάθε 8-συνδεδεμένα στοιχεία (αντικείμενα) στη δυαδική εικόνα, BW. Τα στοιχεία επιστρέφονται ως ένας πίνακας "struct", που περιέχει ένα "struct" για κάθε αντικείμενο της εικόνας. Χρησιμοποιούμε την "regionprops" τόσο σε γειτονικές περιοχές, όσο και σε περιοχές με ασυνέχειες.

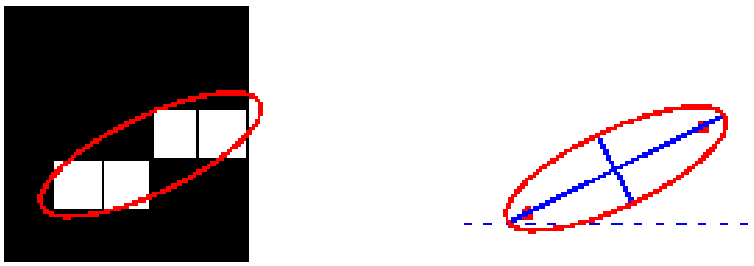
Όσον αφορά το όρισμα που χρησιμοποιήσαμε στην "regionprops", αυτό ήταν η ιδιότητα "Area", όπου ο πραγματικός αριθμός των εικονοστοιχείων στην περιοχή, επιστρέφεται ως κλιμακωτός μελετώντας όλα τα αντικείμενα που αναγνωρίστηκαν σε οποιαδήποτε κατεύθυνση και γωνία. Έπειτα, έγινε σύζευξη των σημείων μέσω της συνάρτησης "cat" ως προς αναγνωρίσιμη επιφάνεια. Αν το αποτέλεσμα αυτής της σύζευξης ήταν μεγαλύτερο από 50 pixel, τότε αποθηκεύουμε την περιοχή ως πιθανή περιοχή φθορών, εξαλείφοντας τα στοιχεία που είναι συνδεδεμένα κατά λιγότερο από 50 pixels.

Παράλληλα μέσω της συνάρτησης "bwlabel", η οποία επιστρέφει δεδομένα τα οποία είναι κατά 8-συνδεδεμένα μεταξύ τους, συγκρίναμε τα αποτελέσματα των δύο συναρτήσεων μέσω της "ismember", όπου αναγνωρίστηκαν και αποθηκεύτηκαν τα κοινά σημεία των δύο μεταβλητών. Το αποτέλεσμα της "ismember" το εκχωρήσαμε ως είσοδο στην μορφολογική διαδικασία της "bwmorph" όπου μέσω του ορίσματος "skel" (*skeletonize*) αφαιρέθηκαν τα εικονοστοιχεία στα όρια των αντικειμένων, χωρίς

όμως να έχουμε αποκοπές στα συνδεδεμένα αντικείμενα. Γίνεται έτσι σαφής η μεγάλη μείωση του θορύβου που προκάλεσε στην εικόνα μας η μεθοδολογία του Otsu. Χρησιμοποιώντας ακόμα μία διαφορετική μορφολογική διαδικασία, ήμασταν σε θέση να καθαρίσουμε ακόμα περισσότερο την εικόνα μας.

Στην δεύτερη μορφολογική διαδικασία χρησιμοποιήσαμε τις ίδιες συναρτήσεις, με την ίδια ακολουθία αλλά με διαφορετικό όρισμα στην συνάρτηση της "regionprops". Αντί για το όρισμα "Area", χρησιμοποιήθηκε το όρισμα**"Orientation".** Η μεθοδολογία αυτή λειτουργεί ως εξής: η γωνία μεταξύ του άξονα x και του κύριου άξονα της έλλειψης, που είχε τις ίδιες δευτερεύουσες τιμές με την περιοχή, επιστρέφεται ως βαθμωτή τιμή. Η τιμή υπολογίζεται σε μοίρες, που κυμαίνονται σε ένα εύρος τιμών –90 εως 90.

Το παρακάτω σχήμα απεικονίζει τους άξονες και τον προσανατολισμό της ελλείψεως. Η αριστερή πλευρά του σχήματος δείχνει μια περιοχή της εικόνας και την αντίστοιχη έλλειψη της. Η δεξιά πλευρά δείχνει την ίδια έλλειψη με τις μπλε γραμμές να αντιπροσωπεύουν τους άξονες και τις κόκκινες κουκίδες να αντιπροσωπεύουν τα σημεία εστίασης. Ο προσανατολισμός είναι η γωνία μεταξύ της οριζόντιας διακεκομμένης γραμμής και του κύριου άξονα.

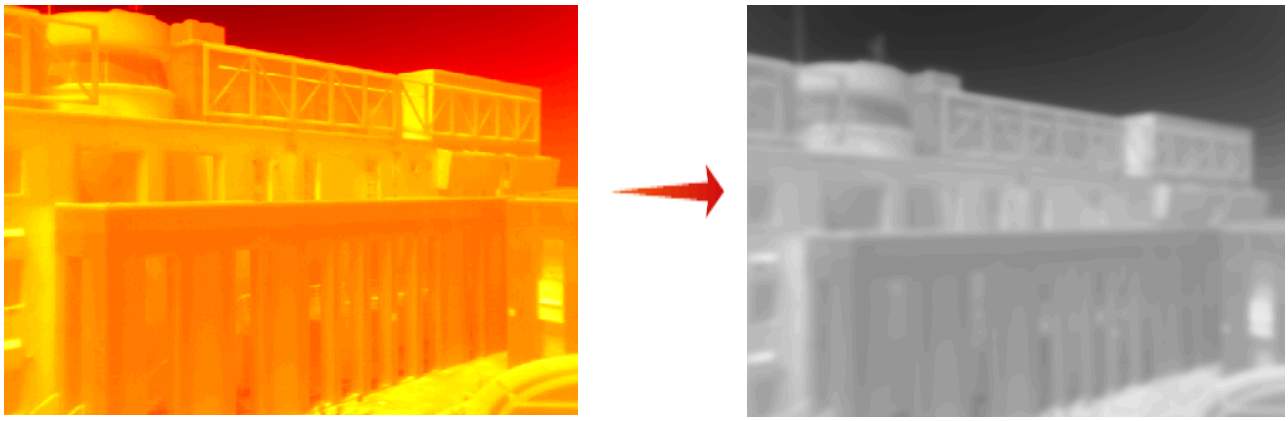


Λόγω έλλειψης συνδεδεμένων εικονοστοιχείων, η παραπάνω διαδικασία κρίνεται ιδανική, καθώς προσφέρει τον μεγαλύτερο δυνατό καθαρισμό της εικόνας. Αξίζει να αναφερθεί ότι στον τομέα της αυτοματοποιημένης αναγνώρισης κτιριακών φθορών δεν έχει βρεθεί ακόμα τρόπος της αυτόματης παραμετροποίησης των διάφορων σταθερών που χρησιμοποιεί η κάθε προσέγγιση, έρευνα ή διπλωματική. Οι λόγοι που την καθιστούν δύσκολη μπορεί να είναι η ποιότητα της εκάστοτε φωτογραφικής που καταγράφει την φθορά, η γωνία λήψης, η φωτεινότητα, η πιθανή ύπαρξη άλλων αντικειμένων, αλλά ακόμα και η ίδια η μέθοδος.

## 3.3: Αλγόριθμος Επεξεργασίας Θερμικής Εικόνας

Για την επεξεργασία των δεδομένων που λαμβάνονται από την θερμική κάμερα ακολουθούνται τα εξής βήματα:

1. Διάβασμα των .tiff αρχείων και εφαρμογή ενός colomap (hot colormap) για την οπτική απεικόνιση της εικόνας.
2. Μετατροπή της εικόνας σε Grayscale
3. Εφαρμογή ROI (Region Of Interest)
4. Εφαρμογή της Gaussian συνάρτησης για την εξομάλυνση της εικόνας
5. Εφαρμογή Laplacian Edge Detection



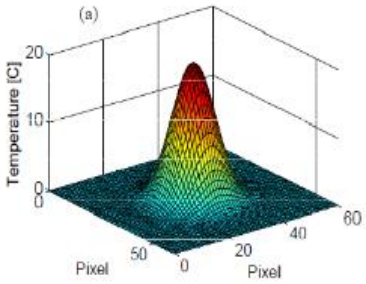
Για την καλύτερη και αποδοτικότερη ανάλυση εφαρμόζουμε την τεχνική επιλογής του σημείου ή του συνόλου των σημείων των οποίων θέλουμε να επεξεργαστούμε (ROI: Region Of Interest) [20]. Αυτό μας δίνει την δυνατότητα για μεγαλύτερης ευκρίνειας αποτελέσματα, όπως και την δυνατότητα να επεξεργαστούμε εικόνες και δεδομένα τα οποία στερούνται κατάλληλης καταγραφής, για παράδειγμα λήψεις μακρινών πλάνων ή λανθασμένων πληροφοριών μέσα στο πλάνο. Πλέον, τα περισσότερα περιβάλλοντα μας επιτρέπουν την χρησιμοποίηση της ROI λειτουργίας, κάτι το οποίο καθιστά αποδοτικότερους τους αλγορίθμους που σχεδιάζονται.

Ως επόμενο βήμα εντάξαμε την εξομάλυνση της εικόνας μέσω ενός Gaussian φίλτρου [19]. Αυτό έχει ως αποτέλεσμα τα περισσότερα προς μελέτη στοιχεία να έχουν όσο το δυνατόν ομαλότερες μεταβαλλόμενες τιμές. Είναι αναγκαία η χρήση της συγκεκριμένης μεθοδολογίας, καθότι σημεία με υγρασία (και κατ' επέκταση

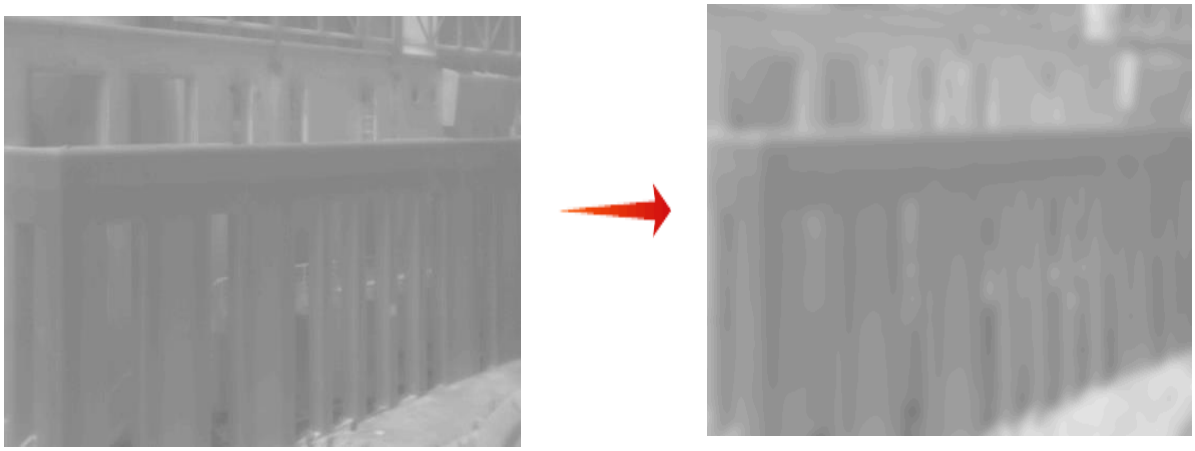
μελλοντική φθορά) παρουσιάζουν τοπικά μέγιστες τιμές σε λίγα και μόνο pixel της εικόνας.

Ο τύπος της Gaussian που χρησιμοποιείται για την συγκεκριμένη εξομάλυνση είναι ο ακόλουθος:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$




Ο συγκεκριμένος τύπος εφαρμόζεται συνελικτικά στην grayscale εικόνα. Εμπειρικά και πειραματικά ορίσαμε την τιμή της μέσης απόκλισης σ=0.4. Τα αποτελέσματα του συγκεκριμένου βήματος προβάλλονται παρακάτω:



Έπειτα από την εφαρμογή της Gaussian συνάρτησης, εφαρμόζουμε την Laplacian αναγνώριση ακμών (Laplacian edge detection). Ο λόγος ο οποίος χρησιμοποιείται αυτή η μεθοδολογία (ευρύτερα γνωστή και ως Laplacian of Gaussian edge detection) είναι διότι το Laplacian φίλτρο είναι ιδιαίτερα ευαίσθητο σε περιοχές με ακραίες μεταβολές των τιμών τους [22], [19], [23]. Το συγκεκριμένο φίλτρο υπολογίζεται από τις δεύτερες παραγώγους των σημείων της εικόνας με χρήση

διαφόρων μασκών (στην συγκεκριμένη ανάλυση χρησιμοποιείται η 1η μάσκα):

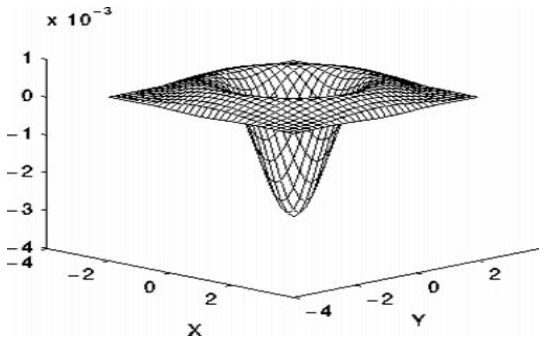
| | | |
|---|---|---|
| 0 | 1 | 0 |
| 1 | −4 | 1 |
| 0 | 1 | 0 |

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 1 | −8 | 1 |
| 1 | 1 | 1 |

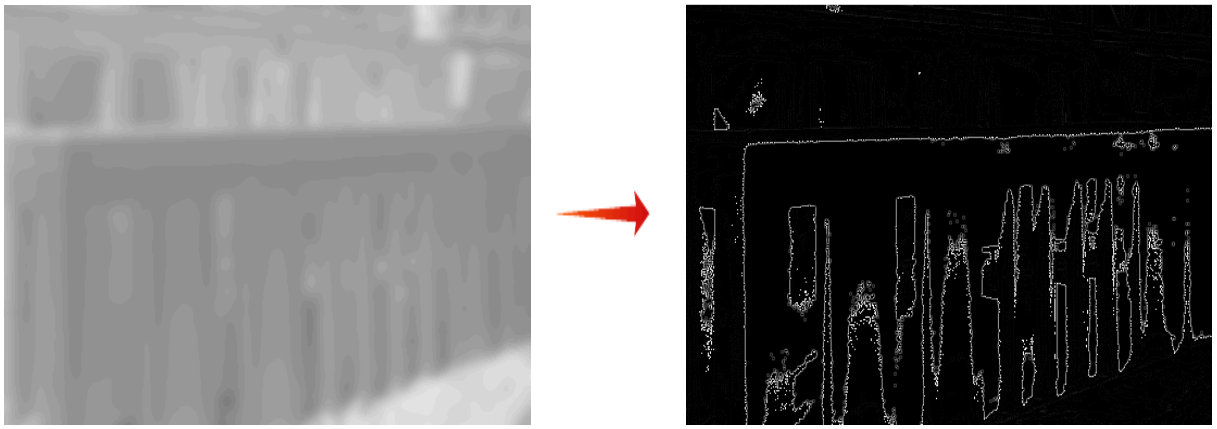
| | | |
|---|---|---|
| −1 | 2 | −1 |
| 2 | −4 | 2 |
| −1 | 2 | −1 |

Η ένωση των δύο αυτών εξισώσεων (Laplacian of Gaussian) περιγράφεται από την σχέση:

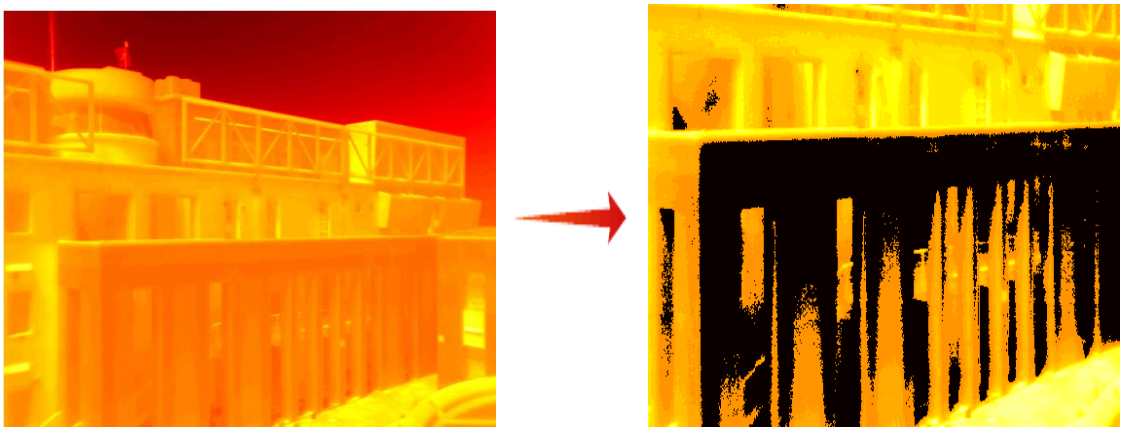
$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right]e^{-\frac{x^2+y^2}{2\sigma^2}}$$




Με αυτόν τον τρόπο μπορούν να αναγνωριστούν οι ακμές των σφαλμάτων που έχουν εντοπιστεί:



Μόλις ολοκληρωθεί η παραπάνω διαδικασία, είναι πλέον δυνατό να επαναφέρουμε την εικόνα στην αρχική της μορφή μαζί με τα στοιχεία τα οποία αναγνωρίστηκαν ως φθορές υγρασίας. Το αποτέλεσμα της συγκεκριμένης επαναφοράς απεικονίζεται παρακάτω:



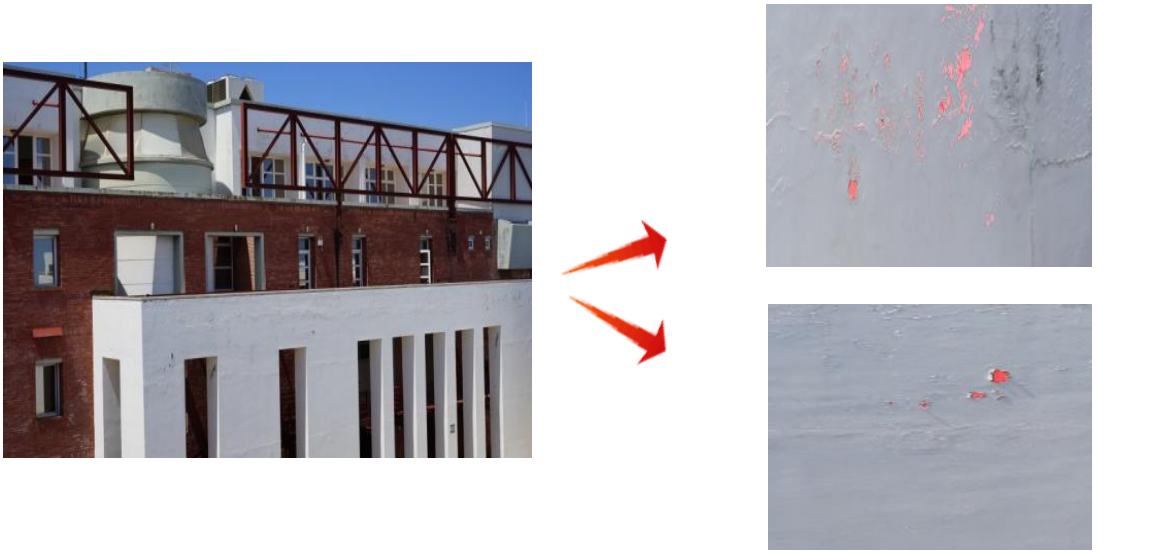
## 3.4: Συνδυασμός Τύπων Εικόνας και Αποτελεσμάτων Επεξεργασίας

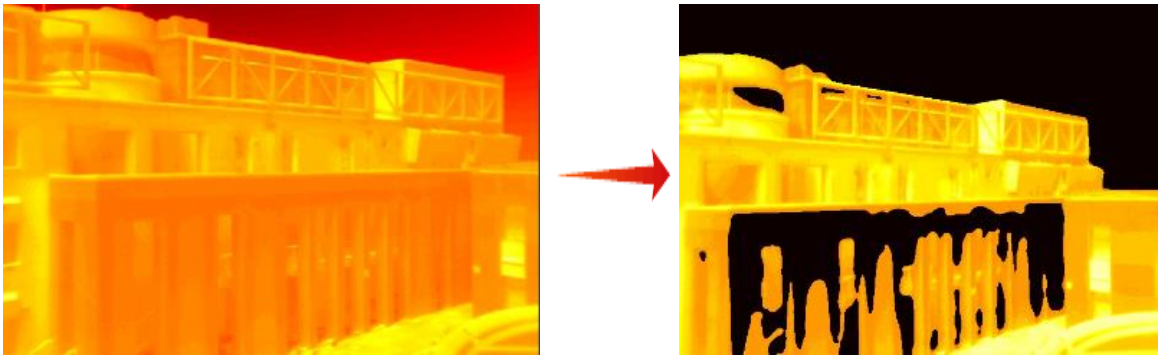
Μία από τις πολλές δυνατότητες που μας παρέχουν τα σύγχρονα εργαλεία ανάλυσης και επεξεργασίας εικόνων είναι ότι, μέσω διαφόρων γλωσσών προγραμματισμού, μπορούμε να συνδυάσουμε τεχνικές που εκτελούνται στον ίδιο χρόνο[24]. Ένα τέτοιο σύστημα εκμεταλλευόμαστε στην συγκεκριμένη διπλωματική εργασία. Μέσω των εργαλείων της Matlab2018b μπορούμε να συνδυάσουμε τις δύο παραπάνω τεχνικές και να προβάλλουμε παράλληλα τα αποτελέσματά τους.

Ο τρόπος εξαγωγής τους γίνεται μέσω μιας εφαρμογής GUI, η οποία προβάλλει στον χρήστη τα αποτελέσματα της οπτικής και της θερμικής επεξεργασίας ταυτόχρονα. Παρέχει μάλιστα την δυνατότητα της πλήρους χρήσης πολυπύρηνων επεξεργαστών, μέσω της απλής αντικατάστασης των επαναλήψεων “for-loop” με αυτές των “parfor-loop”. Με την συγκεκριμένη διαφοροποίηση μπορούμε να θέσουμε σε ταυτόχρονη λειτουργεία τα 3/4 των πυρήνων του εκάστοτε υπολογιστικού συστήματος, όπου κάθε πυρήνας εκτελεί διαφορετικό κομμάτι του κώδικα, γεγονός

που μειώνει κατά πολύ τους χρόνους εκτέλεσης. Λόγω του ότι το σύστημά μας είναι ικανό να διαβάζει αρχεία-βίντεο και όχι μόνο αρχεία-εικόνας, αυτή η λειτουργία κρίνεται ως σημαντική και αναγκαία.

Τα αποτελέσματα της σύντηξης, όπως προβάλλονται στον χρήστη, φαίνονται στις ακόλουθες εικόνες:





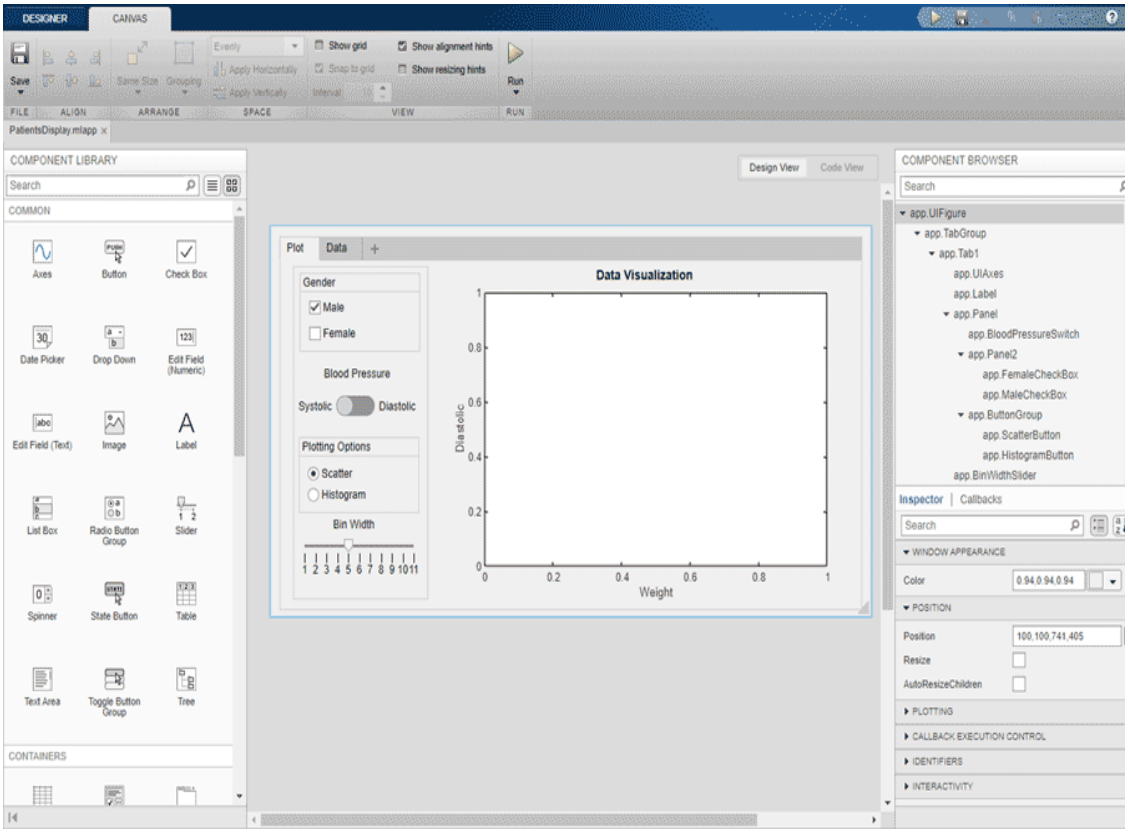
## 3.5: Υλοποίηση GUI

Για την επίλυση του προβλήματος παραμετροποίησης των μεταβλητών και των μεθόδων, καθώς και για να μπορέσουμε να προσφέρουμε σε επίπεδο χρήστη την λειτουργία και εκμετάλλευση των παραπάνω διαδικασιών [25], αναπτύξαμε μια

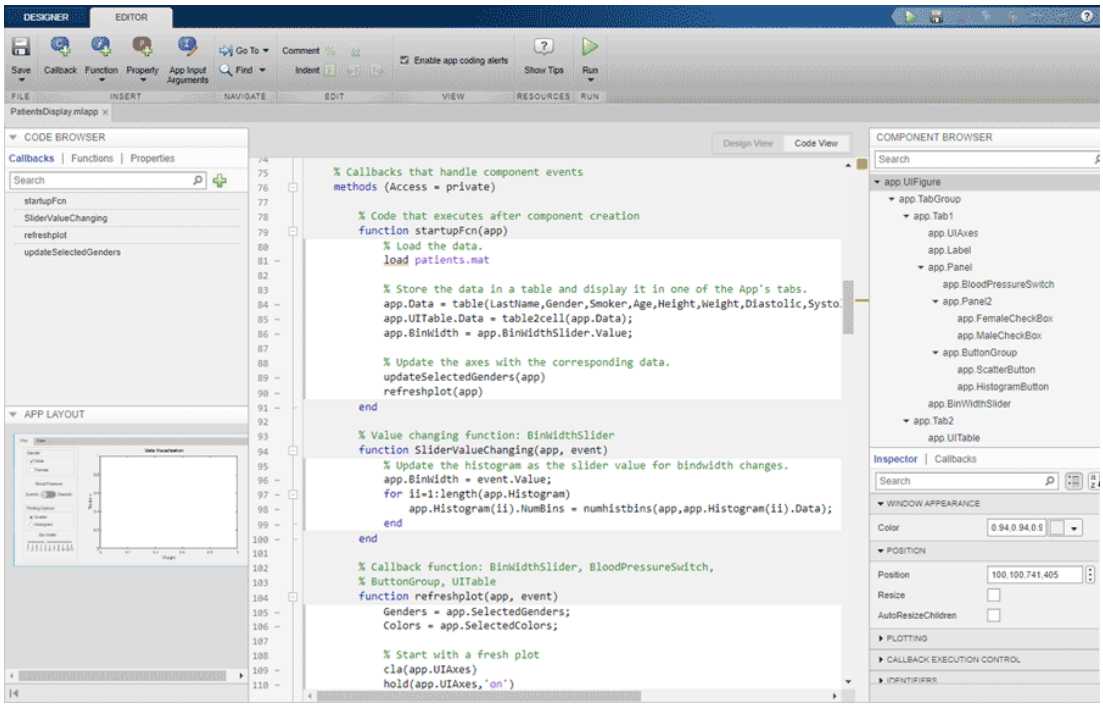
εφαρμογή UI μέσω της πλατφόρμας AppDesigner της Matlab2018b.

Εδώ και πάνω από μια δεκαετία, η Matlab έχει αναπτύξει το περιβάλλον “Matlab GUI” με σκοπό τον σχεδιασμό απλών εφαρμογών οι οποίες θα υποστηρίζουν όλες σχεδόν τις συναρτήσεις και τις μεθόδους της. Λόγω της μεγάλης ανταπόκρισης που είχε το συγκεκριμένο εργαλείο, το 2016 οι προγραμματιστές της Matlab προσέφεραν το περιβάλλον AppDesigner, ως εξέλιξη του προηγούμενού του (MatlabGUI) με σκοπό την ολοένα βελτιωμένη σχεδίαση web και desktop εφαρμογών. Σήμερα, το 2019, πάνω από το 90% των εφαρμογών που αναπτύσσονται στο περιβάλλον της Matlab χρησιμοποιούν το εργαλείο AppDesigner.

Το συγκεκριμένο περιβάλλον μας δίνει την ευκολία της άμεσης σχεδίασης του interface της εφαρμογής που θέλουμε να σχεδιάσουμε, μέσω της λειτουργίας click and drop, με αυτόματη ανάπτυξη του εκάστοτε blog κώδικα. Επίσης, ακολουθώντας την παλαιότερη φιλοσοφία ανάπτυξης εφαρμογών, οτιδήποτε μπορεί να σχεδιαστεί, μπορεί να ενταχθεί απευθείας με κώδικα.

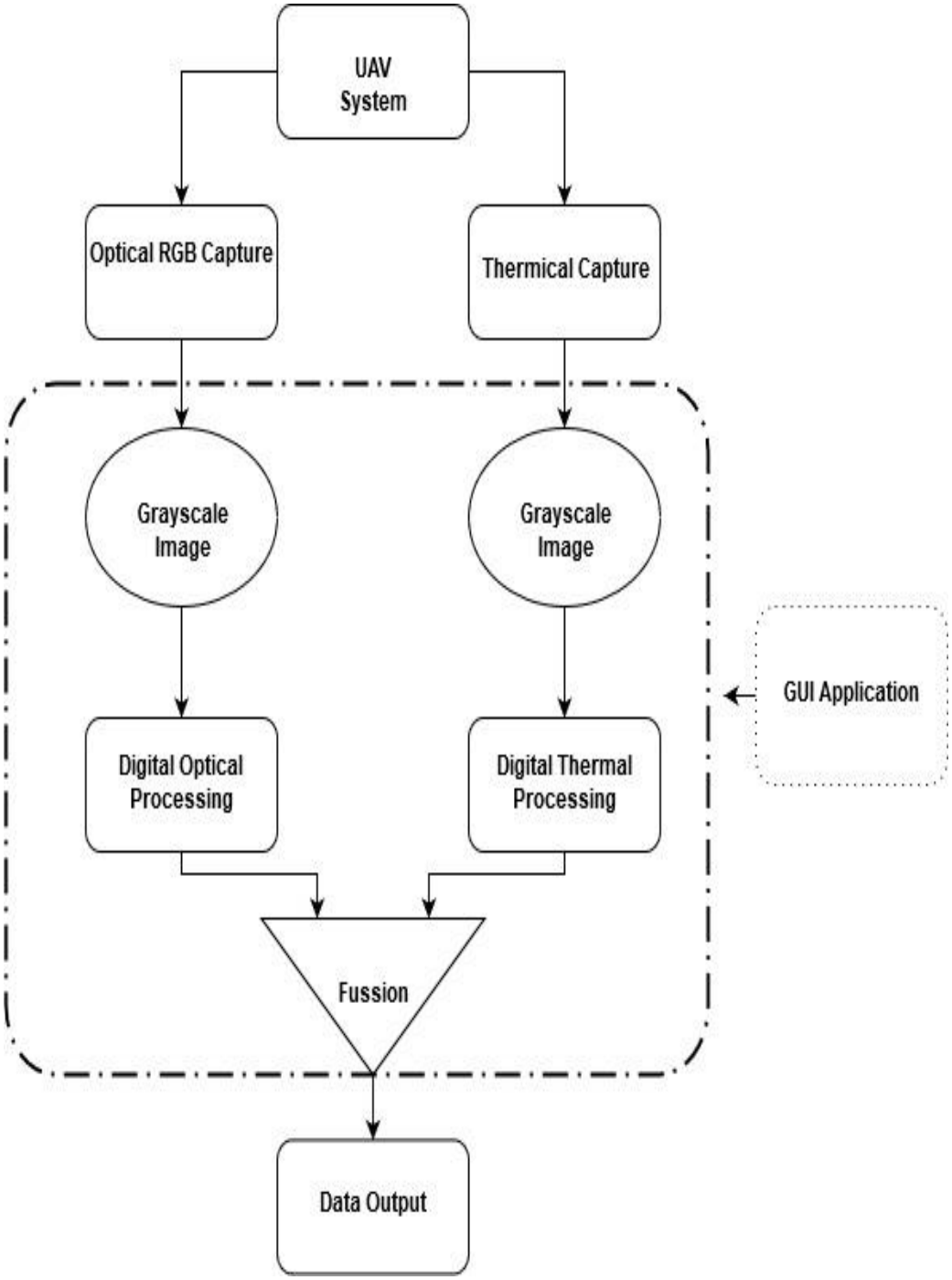


Από τα μεγαλύτερα οφέλη που αποκομίζουμε είναι η υποστήριξη των περισσότερων συναρτήσεων, μέθοδων και βιβλιοθηκών που υπάρχουν και στην Matlab. Είναι ευρέως γνωστό ότι η Matlab αποτελεί ίσως το πιο αναπτυγμένο υπολογιστικό εργαλείο στον τομέα των μαθηματικών. Για τον λόγο αυτόν, η δυνατότητα να αναπτυχθούν web/desktop εφαρμογές σε περιβάλλοντα και γλώσσες χωρις την υπολογιστική ικανότητα της, θέτει συχνά αρκετά εμπόδια (πχ C++, JAVA).

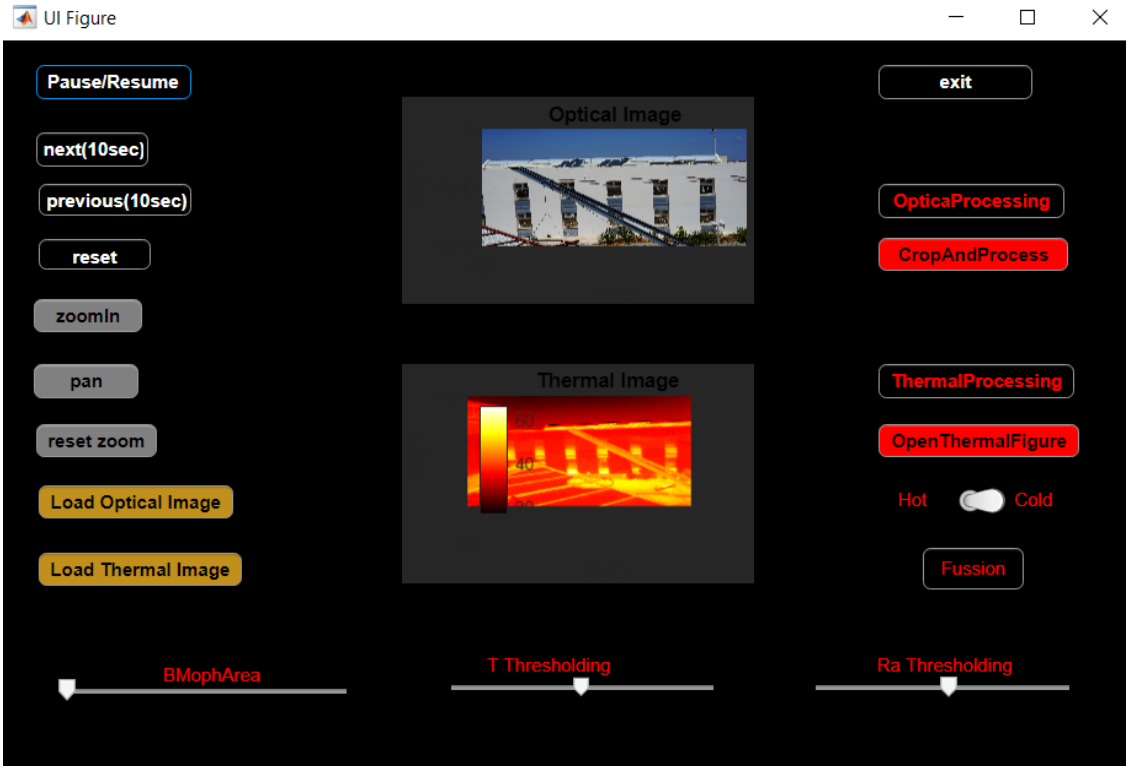



Παρά το σύντομο χρόνο κυκλοφορίας της, η πλατφόρμα AppDesigner έχει όλο και μεγαλύτερη απήχηση στον τομέα του προγραμματισμού εφαρμογών. Για τον λόγο αυτό, κάθε χρόνο γίνεται όλο και μεγαλύτερη προσπάθεια για την εξέλιξη και ανάπτυξη της συγκεκριμένης πλατφόρμας. Πλέον, κάθε νέα έκδοση του λογισμικού της Matlab συνδυάζεται με την επόμενη έκδοση της πλατφόρμας AppDesigner, γεγονός που την κάνει ολοένα και πιο ανταγωνιστική στο τομέα της σχεδίασης και ανάπτυξης εφαρμογών.

Το παρακάτω διάγραμμα παρουσιάζει τα πλήρη βήματα που ακολουθούνται για την υλοποίηση του συστήματος:



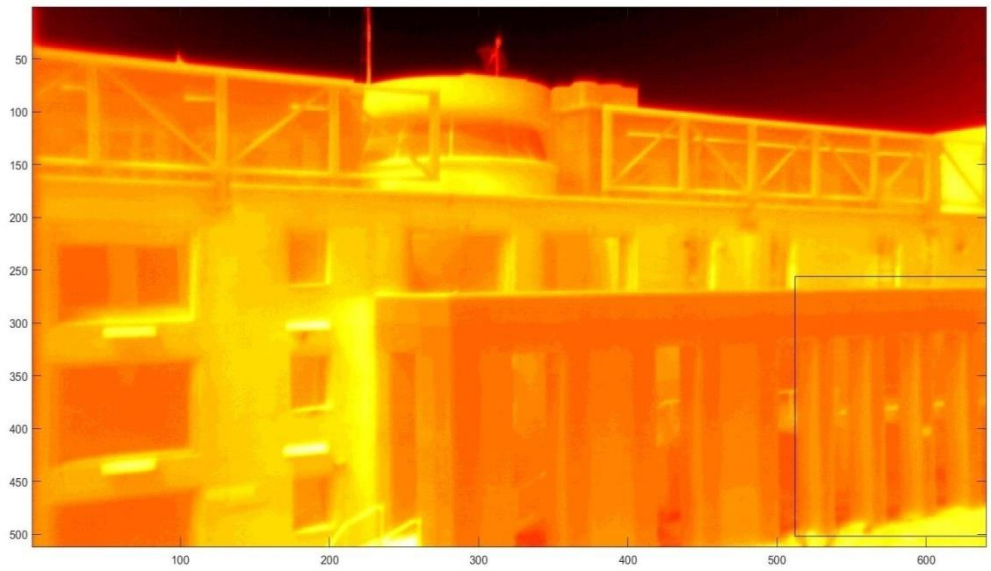

Παραθέτουμε στην συνέχεια το interface της εφαρμογής που σχεδιάσαμε, προβάλλοντας τις πρόσθετες λειτουργίες για την περαιτέρω ανάλυση τους:




Αναλυτικότερα:

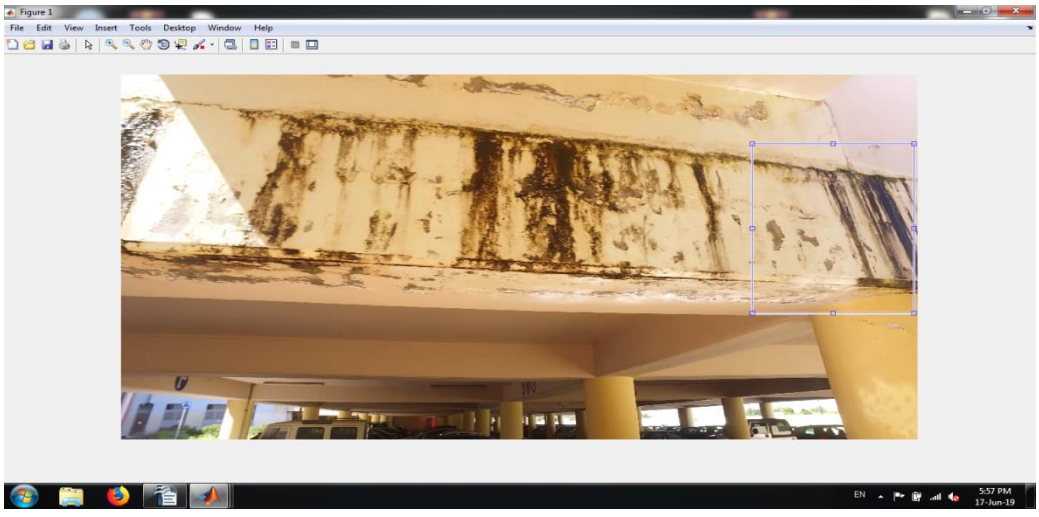
- Λόγω του ότι η εφαρμογή μας δέχεται ως είσοδο τα frames των οπτικών και θερμικών video που εισάγουμε, προσθέσαμε την λειτουργία ***pause/play***, δίνοντας την δυνατότητα στον χρήστη να μπορεί να σταματήσει/συνεχίσει την προβολή και την επεξεργασία των video σε οποιοδήποτε σημείο απαιτείται.
- Για την ευκολότερη πλοήγηση, καθώς μπορεί να υπάρχουν frames που επιθυμεί ο χρήστης να αποφύγει, προσθέσαμε την λειτουργία ***next****(10sec)*/***previous****(10sec)*/***reset***. Δεδομένου ότι το κάθε δευτερόλεπτο έχει 23frames (23fps), ο χρήστης έχει την δυνατότητα να προχωρήσει ή να επιστρέψει κατά 230 frames= 10 second. Η λειτουργία *reset* επαναφέρει το video στο αρχικό σημείο.
- Στη συνέχεια, προσθέσαμε δύο λειτουργίες, η μία εκ των οποίων είναι για την φόρτωση των δεδομένων της οπτικής κάμερας **LoadOpticalImage** και η άλλη για την φόρτωση των δεδομένων της θερμικής **LoadThermalImage**. Επιλέγοντας μία από τις δύο ή και τις δύο λειτουργίες, το πρόγραμμα εμφανίζει τα δεδομένα στην οθόνη του χρήστη, εισάγοντας frame by frame τα δεδομένα από τα αντίστοιχα αρχεία του δίσκου.

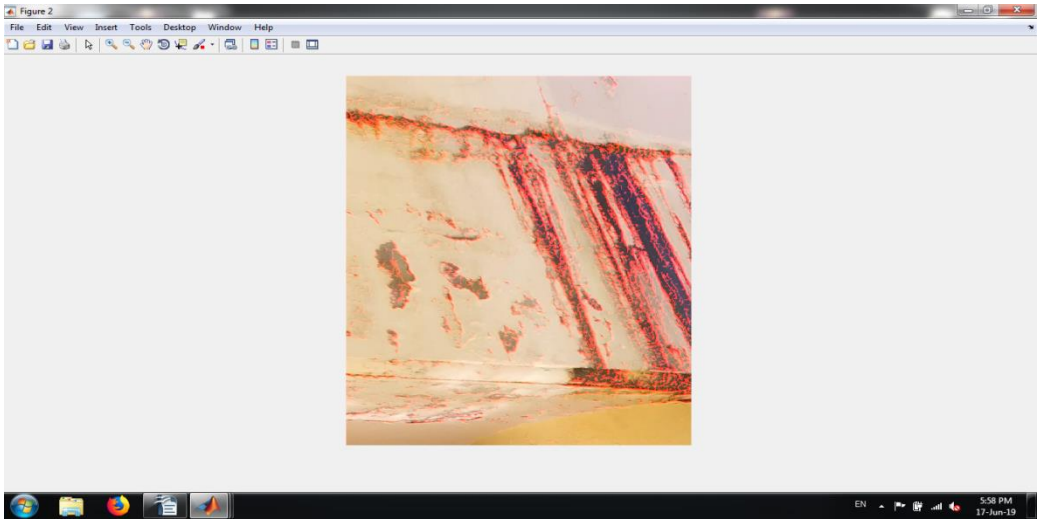
- Εχουμε τρείς λειτουργίες που επιλύουν το ζήτημα της παραμετροποίησης. Δίνεται η δυνατότητα στον χρήστη να μπορεί να αλλάζει τους βαθμούς έντασης των τριών μεταβλητών (των οποίων στο προηγούμενο κεφάλαιο τις είδαμε ως σταθερές μεταβλητές). Προσφέρεται η $1^{η}$ επιλογή **BmorphArea** που αλλάζει τον χώρο επεξεργασίας της μορφολογικής μας διαδικασίας. Έπειτα, δίνεται η $2^{η}$ επιλογή **T Threshold** η οποία ελέγχει την σταθερή μεταβλητή τ και τέλος, προσφέρεται η $3^{η}$ επιλογή **Ra Threshold** η οποία ελέγχει την σταθερά. Με αυτό τον τρόπο ο χρήστης είναι σε θέση να ελέγχει αποδοτικότερα τους αλγόριθμους που αναπτύξαμε.
- Προστέθηκαν τέσσερις διαχειριστικές λειτουργίες (ξεκινώντας από πάνω προς τα κάτω): η λειτουργία **Exit** όπου τερματίζει το πρόγραμμα, η λειτουργία **Zoom In** όπου προσφέρει την δυνατότητα της επιλεγόμενης μεγέθυνσης της εικόνας, η λειτουργία **Pan** για την πλοήγηση σε οποιαδήποτε κατεύθυνση ύστερα από την επιλογή της **Zoom In** και η λειτουργία **Reset Zoom** για την επαναφορά της εικόνας στις αρχικές διαστάσεις.
- Οι δύο επόμενες λειτουργίες είναι η **Optical Processing** και η **Thermal Processing**. Με την επιλογή **Optical Processing** το πρόγραμμα θα προβάλει τα αποτελέσματα της μεθοδολογίας αναγνώρισης φθορών οπτικών εικόνων. Με την ενεργοποίηση της συγκεκριμένης λειτουργίας ο χρήστης μπορεί να ελέγχει και να καθορίζει τις σταθερές μεταβλητές, καθώς και να μελετάει περαιτέρω το αντικείμενο που τον ενδιαφέρει. Με την επιλογή **Thermal Processing**, η εφαρμογή ανοίγει ένα νέο παράθυρο όπου ο χρήστης μπορεί να έχει πρόσβαση στα αποτελέσματα της θερμικής επεξεργασίας. Συγκεκριμένα, μπορεί να εστιάζει στις θερμοκρασίες που έχει το κάθε εικονοστοιχείο.
- Με την λειτουργία **Open Thermal Figure** γίνεται αυτόματη αναγνώριση των χαμηλότερων θερμοκρασιών που υπάρχουν στην εκάστοτε εικόνα όπως φαίνονται στο μπλε πλαίσιο παρακάτω:




Η λειτουργία αυτή οδηγεί τον χρήστη στις πιο ψυχρές περιοχές του κτιρίου. Αντίστοιχα, εισάγεται η δυνατότητα να επιλέξει να οδηγείται στις πιο θερμές περιοχές (***cold/hot switch***), ανάλογα με την χρήση και το αντικείμενο που επιθυμεί να παρακολουθήσει [3].

- Μία ακόμα λειτουργία που αναπτύχθηκε είναι αυτή της **CropAndProcess**. Δεδομένου ότι τα πλάνα που λαμβάνει ο χρήστης μπορεί περιέχει αντικείμενα τα οποία να μην χρειάζεται να αναλύσει, του δίνεται η δυνατότητα να επιλέξει τα αντικείμενα εστίασης. Με αυτήν την λειτουργία, εμφανίζεται ένα νέο παράθυρο, όπου μπορεί να επιλέξει την περιοχή εστίασης και να προβεί στην αυτόματη αναγνώριση των φθορών μέσω της οπτικής επεξεργασίας.






- Μία ακόμα σημαντική λειτουργία είναι αυτή της "**Fussion**" όπου εκτελείται η σύντηξη των 2 παραπάνω διαδικασιών, αυτής της οπτικής και της θερμικής επεξεργασίας.
- Έπειτα από τις παραπάνω διαδικασίες ο χρήστης μπορεί να κάνει εξαγωγή και αποθήκευση των αποτελεσμάτων είτε σε αρχεία εικόνων, είτε σε αρχεία δεδομένων (data arrays).

# Κεφάλαιο 4ο:Αποτελέσματα-Συμπεράσματα και Κριτική Αποτίμηση

Άπο το σύνολο όλων των μεθοδολογιών που έχουμε στην διάθεσή μας, δηλαδή τις μεθόδους Dhule [6], Talab [8], Dorafshan [9] και Hoang [10] η συνδυαστική μεθοδολογία αναγνώρισης φθορών μέσω εξέλιξης των διαδικασιών της τεχνικής του Otsu, αποτελεί την ακριβέστερη από το σύνολο των μεμονομένων μεθοδολογιών. Το γεγονός ότι, στα πλαίσια της παρούσας διπλωματικής εργασίας, επιτύχαμε την μείωση του χρόνου εκτέλεσης και τον περιορισμό της προσωρινής μνήμης, εντάσσει την συνδιαστική μεθοδολογία όχι μόνο ως ικανή, αλλά και ως αναγκαία για την επίτευξη ακριβέστερων αποτελεσμάτων.

Ειδικότερα, στηριζόμενοι στα ερευνητικά δεδομένα των Vermani και Ojha [7], Pithadiya [19], Duarte et al. [20] εστιάσαμε σε διαδικασίες οι οποίες εφαρμόνται στο θερμικό φάσμα των σωμάτων καταγραφής. Με την χρήση θερμικών καμερών καταφέραμε να αναπτύξουμε μεθοδολογίες, τα αποτελέσματα των οποίων θεωρούνται ικανά στην έγκαιρη πρόβλεψη και μελέτη κτιριακών φθορών, καθώς συμπεριλάβαμε φθορές που δεν μπορούν να αναγνωριστούν με γυμνό μάτι ή δεν μπορούν να καταγραφούν από οπτικές κάμερες.

Επίσης καταφέραμε, μέσω του GUI, να προσφέρουμε σε ανθρώπους οι οποίοι έχουν ελάχιστη σχέση με την επιστήμη της μηχανικής όρασης και του προγραμματισμού, την δυνατότητα να χρησιμοποιήσουν και να εκμεταλλευτούν τις παραπάνω τεχνολογίες και μεθοδολογίες [25]. Σημαντικός παράγοντας, ήταν η χρήση των μαθηματικών μοντέλων των οποίων η Matlab προσφέρει, καθώς ο τομέας την μηχανικής όρασης είναι άρρηκτα συνδεδεμένος με την επιστήμη των μαθηματικών.

Όσον αφορά τους περιορισμούς και την κριτική αποτίμηση της παρούσας εργασίας, αξίζει να επισημανθεί ότι για να επιτευχθεί αποδοτικά η ταυτόχρονη οπτική και θερμική επεξεργασία, είναι απαραίτητη η γωνία λήψης και ο βαθμός εστίασης των δεδομένων να είναι αυστήρα ο ίδιος. Μόνο μέσα από αυτόν τον τρόπο επιτυγχάνεται η σύντηξη των αποτελεσμάτων η οποία μας προσφέρει την μέγιστη δυνατή ακρίβεια. Ακόμα, λόγω του ότι η πλατφορμα AppDesigner είναι ακόμα σχεδιασμένη για απλές εφαρμογές (όπως παρουσίαση γραφικών διαγραμμάτων) δεν δίναται η χρήση του για απαιτητικές εφαρμογές π.χ επεξεργασία βίντεο. Ωστόσο, αποτελεί ένα αντικείμενο

που βρίσκεται σε εξέλιξη, και σύντομα θα προσφέρει περισσότερες δυνατότητες τόσο στο λειτουργικό κομμάτι, όσο και στις ταχύτητες εκτέλεσης των εφαρμογών που σχεδιάζονται.

Η συγκεκριμένη διπλωματική αποτελεί ένα πρώτο επιτυχημένο βήμα προς την κατεύθυνση της δυνατότητας να χρησιμοποιηθούν αυτοματοποιημένα UAV συστήματα για την αναγνώριση και καταγραφή φθορών. Δεν είναι τυχαίο που τα συτήματα αυτά αναγνωρίζονται ως χώρος ολοένα και ταχύτερων εξελίξεων στο πεδίο της μηχανικής, καθότι προσφέρουν λειτουργικότητα, ασφάλεια και αυτονομία. Επιπλέον, προσφέρεται ακόμα πεδίο για έρευνα τόσο στο κομμάτι των αλγόριθμων και των μεθοδολογιών, όσο και στο κομμάτι της σύμπτυξης διαφόρων επιστημονικών τομέων, όπως αυτών των τηλεπικοινωνιών και του hardware engineering.

Λόγω του ότι στον συγκεκριμένο τομέα υπάρχουν πολυάριθμες μελέτες και δημοσιεύσεις, η επικέντρωση του δικού μας ενδιαφέροντος εστίασε σε αυτές που αναλύουν διεξοδικά τις μεθόδους και θα μπορούσε ο συνδιασμός τους να προσφέρει τα επιθυμιτά αποτελέσματα.

# Κεφάλαιο 5ο: Προοπτικές Μελλοντικών Ερευνών/ Εφαρμογών

Στην σημερινή εποχή, είναι πολύ σημαντικό να μπορούν οι ερευνητές να εκμεταλλευτούν τις τεχνολογίες της τεχνητής νοημοσύνης και του machine learning [26] [12] [27]. Εντάσσοντας τους παραπάνω αλγόριθμούς σε αυτές τις τεχνολογίες θα βοηθούσε δραστικά στην αύξηση της ακρίβειας τους. Ακόμα, λόγω της αύξησης της ταχύτητας μεταφοράς δεδομένων, η λειτουργία του παραπάνω συστήματος σε real-time χρόνους, θα μπορούσε να είναι ένα πολύ ενδιαφέρον κομμάτι έρευνας. Η αυτοματοποίηση ενός UAV συστήματος το οποίο θα καταγράφει ενώ ταυτόχρονα θα παρουσιάζει τα αποτελέσματα των παραπάνω μεθοδολογιών, είναι μία ιδέα η οποία αγγίζει τόσο εμπορικά όσο και επιστημονικά το ενδιαφέρον πολλών ερευνητών και επιστημόνων.

Ακόμα η μοντελοποίηση των αντικειμένων παρακολούθησης σε τρισδιάστατα μοντέλα και η εισαγωγή τους σε σχεδιαστικά λογισμικά, θα μπορέσουν να βοηθήσουν τόσο τους μηχανικούς που παρακολουθούν ένα κατασκευαστικό έργο, όσο και τους μηχανικούς που αναζητούν την βελτιστοποίηση,την κατασκευή αλλά και την επίλυση κτιριακών προβλημάτων [28], [29].Επίσης, ένα σημαντικό πεδίο ερευνών γίνεται και στον τομέα των embedded συστημάτων, με την δυνατότητα να κατασκευαστούν κυκλώματα όπου εκτελούν τις διεργασίες που παρουσιάζονται στην συγκεκριμένη διπλωματική εργασία.

# Βιβλιογραφία

[1] A. Mohan and S. Poobal, “Crack detection using image processing : A critical review and analysis,” *Alexandria Eng. J.*, 2017.

[2] D. Hu, T. Tian, H. Yang, S. Xu, and X. Wang, “Wall Crack Detection Based on Image Processing,” pp. 4–7, 2012.

[3] M. M. Torok, M. Golparvar-fard, A. M. Asce, and K. B. Kochersberger, “Image-Based Automated 3D Crack Detection for Post-disaster Building Assessment,” pp. 1–13, 2014.

[4] C. Eschmann, C. Kuo, C. Kuo, and C. Boller, “Unmanned Aircraft Systems for Remote Building Inspection and Monitoring,” pp. 1–8.

[5] F. C. Pereira and C. E. Pereira, “Embedded image processing systems for automatic recognition of cracks using UAVs,” *IFAC-PapersOnLine*, vol. 28, no. 10, pp. 16–21, 2015.

[6] J. J. Dhule, “Edge Detection Technique Used for Identification of Cracks on Vertical Walls of The Building,” pp. 263–268, 2015.

[7] A. Vermani and A. Ojha, “Algorithms for Edge Detection and Enhancement for Real Time Images : A Comparative Study,” pp. 651–658, 2015.

[8] A. Mahgoub, A. Talab, Z. Huang, F. Xi, and L. Haiming, “Detection crack in image using Otsu method and multiple filtering in image processing techniques,” *Opt. - Int. J. Light Electron Opt.*, pp. 1–4, 2015.

[9] S. Dorafshan, M. Maguire, and X. Qi, “AUTOMATIC SURFACE CRACK DETECTION IN CONCRETE STRUCTURES USING OTSU THRESHOLDING AND MORPHOLOGICAL OPERATIONS Utah State University,” no. April, 2016.

[10] N. Hoang, “Detection of Surface Crack in Building Structures Using Image

Processing Technique with an Improved Otsu Method for Image Thresholding," vol. 2018, 2018.

[11] M. Patel, S. Tanwar, and S. Tyagi, "Image Processing Based Analysis of Cracks on Vertical Walls," 2018.

[12] M. Kamaliardakani, S. M. Asce, L. Sun, and M. K. Ardakani, "Sealed-Crack Detection Algorithm Using Heuristic Thresholding Approach," pp. 1–10, 2006.

[13] I. Industriale, "Algorithms for infrared image processing," 2011.

[14] B. Więcek and M. Poksinska, "Passive and active thermography application for architectural monuments," 2006.

[15] P. Broberg, "Surface crack detection in welds using thermography," *NDT E Int.*, vol. 57, pp. 69–73, 2013.

[16] B. B. Lahiri *et al.*, "Author ' s personal copy Infrared Physics & Technology Infrared thermography based defect detection in ferromagnetic specimens using a low frequency alternating magnetic field."

[17] D. Wagner, N. Ranc, C. Bathias, and P. C. Paris, "Fatigue crack initiation detection by an infrared thermography method," *Fatigue Fract. Eng. Mater. Struct.*, vol. 33, no. 1, pp. 12–21, 2010.

[18] E. Edis, I. Flores-Colen, and J. de Brito, "Quasi-quantitative infrared thermographic detection of moisture variation in facades with adhered ceramic cladding using principal component analysis," *Build. Environ.*, vol. 94, no. P1, pp. 97–108, 2015.

[19] K. J. Pithadiya, "Combination of Thresholding and Log Edge," no. May, 2017.

[20] A. Duarte *et al.*, "Segmentation algorithms for thermal images," *Procedia Technol.*, vol. 16, pp. 1560–1569, 2014.

[21] T.YKong, A.Rosenfeld “Topological Algorithms For Digital Image Processing”

[22] S. Lee and D. Lee, “Fusion of IR and Visual Images Based on Gaussian and Laplacian Decomposition Using Histogram Distributions and Edge Selection,” vol. 2016, 2016.

[23] H. Kong, H. C. Akakin, and S. E. Sarma, “A Generalized Laplacian of Gaussian Filter for Blob Detection and Its Applications,” no. January, 2013.

[24] R. S. Adhikari, O. Moselhi, and A. Bagchi, “Automation in Construction Image-based retrieval of concrete crack properties for bridge inspection,” *Autom. Constr.*, 2013.

[25] S. Bashir, “DAMAGE DETECTION IN BRIDGES USING,” vol. 7, no. 2, pp. 215–225, 2016.

[26] H. Moon and J. Kim, “INTELIGENT CRACK DETECTING ALGORITHM ON THE CONCRETE CRACK IMAGE USING NEURAL NETWORK,” pp. 1461–1467.

[27] K. Gopalakrishnan, H. Gholami, A. Vidyadharan, and A. Agrawal, “Crack Damage Detection in Unmanned Aerial Vehicle Images of Civil Infrastructure Using Pre-Trained Deep Learning Model,” *Int. J. Traffic Transp. Eng.*, vol. 8, no. 1, pp. 1–14, 2018.

[28] M. M. Sarker, T. A. Ali, A. Abdelfatah, S. Yehia, A. Elaksher, and U. A. Emirates, “A COST-EFFECTIVE METHOD FOR CRACK DETECTION AND MEASUREMENT ON CONCRETE SURFACE,” vol. XLII, no. November, pp. 28–29, 2017.

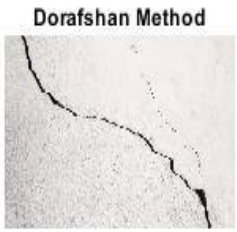
[29] Y. Ham, K. K. Han, J. J. Lin, and M. Golparvar-Fard, “Visual monitoring of civil infrastructure systems via camera-equipped Unmanned Aerial Vehicles (UAVs): a review of related works,” *Vis. Eng.*, vol. 4, no. 1, pp. 1–8, 2016.

# ΠΑΡΑΡΤΗΜΑ 1

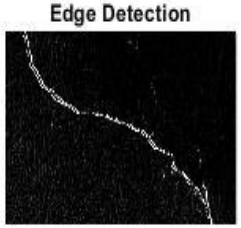
Σε αυτό το παράρτημα παρουσιάζονται τα αποτελέσματα των οπτικών εικόνων σύμφωνα με τις μεθοδολογίες Dhule (1st Gradient Method), Talab (Otsu Method), Dorafshan (Dorafshan Method) και η προτεινόμενη μέθοδος (Proposed Method) όπου γίνεται σαφής η αποτελεσματικότητα του προτεινόμενου αλγορίθμου. Αξίζει να σημειωθεί ότι οι εικόνες προέκυψαν από την -από μεριάς μου- ανάπτυξη του κώδικα των αντίστοιχων δημοσιευμένων μελετών προκειμένου να συγκριθούν καλύτερα οι παραπάνω τεχνικές.

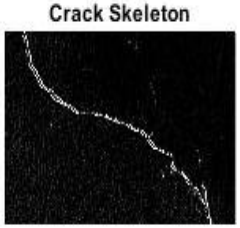

1st Gradient Method

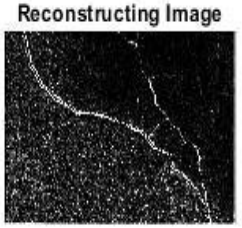

Otsu Method

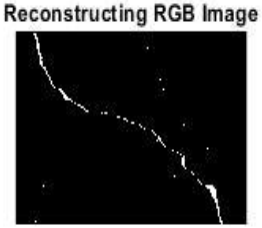

Dorafshan Method


Proposed Method

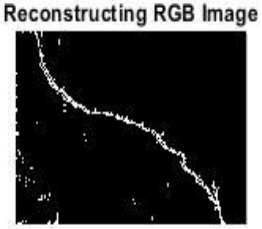

Edge Detection

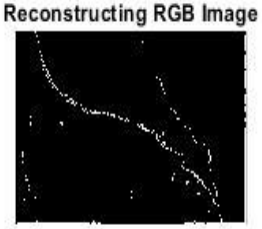

Crack Skeleton

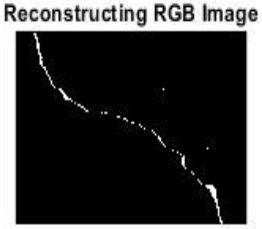

Reconstructing Image

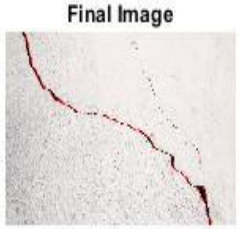

Reconstructing RGB Image

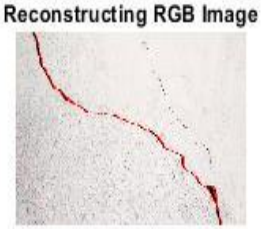

After Binarization

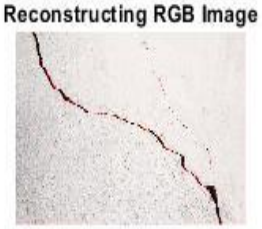

Reconstructing RGB Image

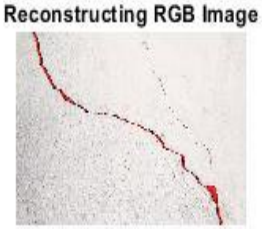

Reconstructing RGB Image


Reconstructing RGB Image


Final Image


Reconstructing RGB Image

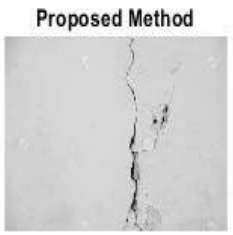

Reconstructing RGB Image

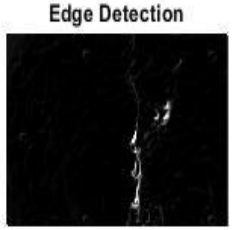

Reconstructing RGB Image

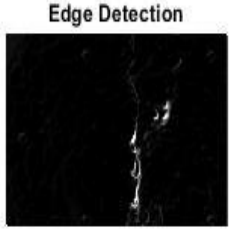

1st Gradient Method


Otsu Method

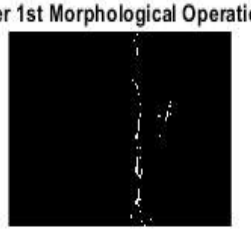

Dorafshan Method

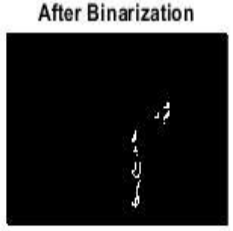

Proposed Method


Edge Detection


Edge Detection


After Sobel Filter


After 1st Morphological Operation


After Binarization


After Morphological Operation


After Morphological Operation

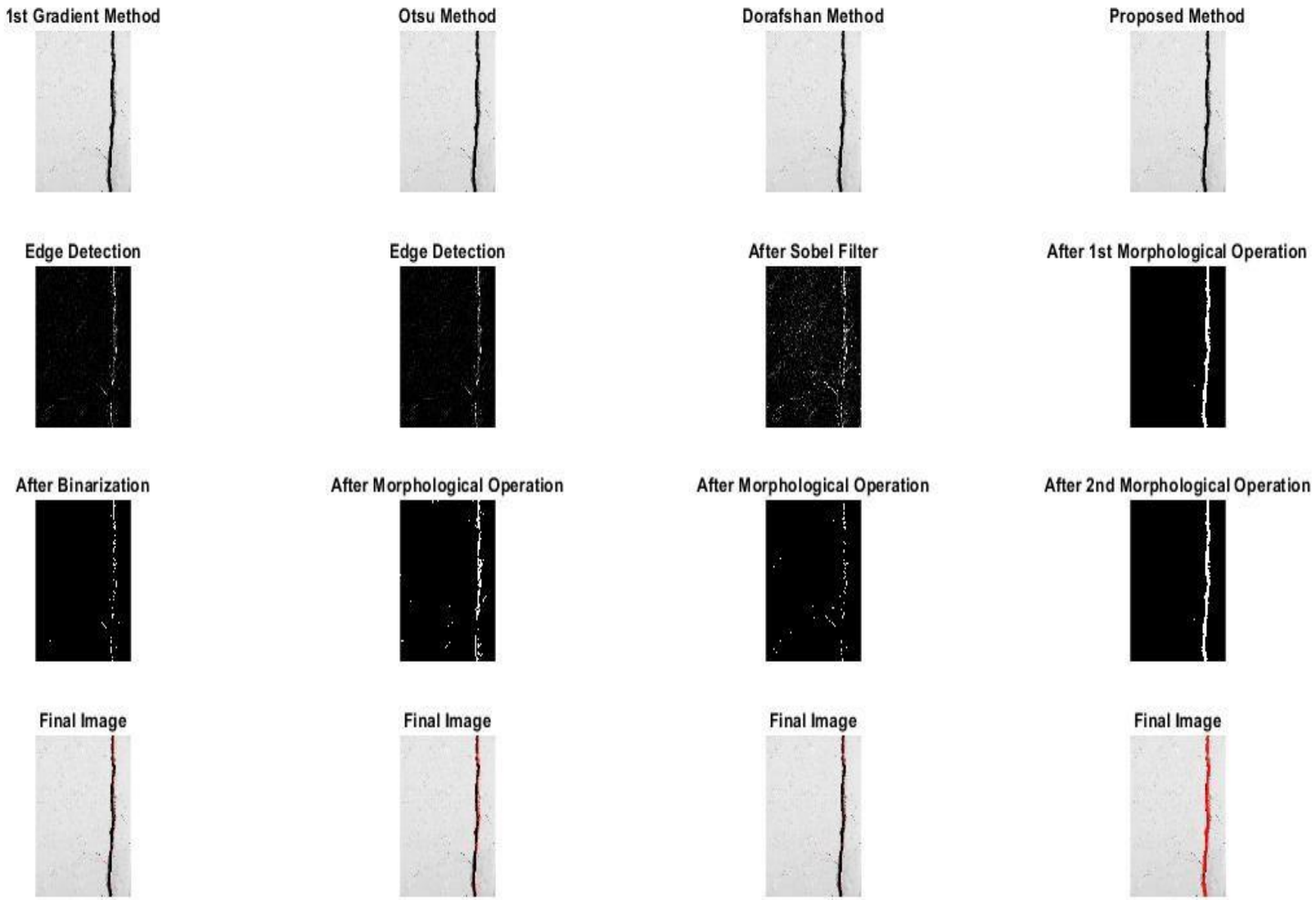

After 2nd Morphological Operation

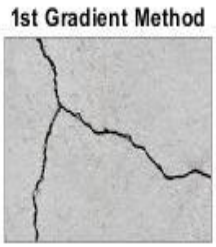

Final Image

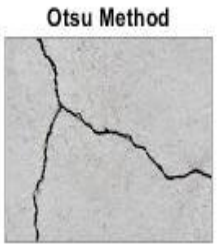

Final Image


Final Image

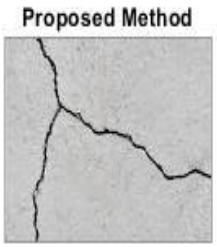

Final Image

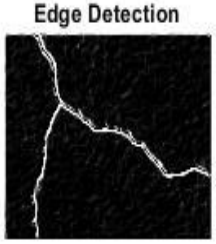

1st Gradient Method
Otsu Method
Dorafshan Method
Proposed Method
Edge Detection
Edge Detection
After Sobel Filter
After 1st Morphological Operation
After Binarization
After Morphological Operation
After Morphological Operation
After 2nd Morphological Operation
Final Image
Final Image
Final Image
Final Image

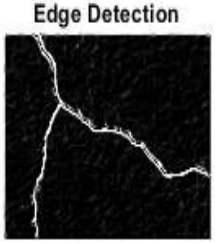

1st Gradient Method


Otsu Method


Dorafshan Method

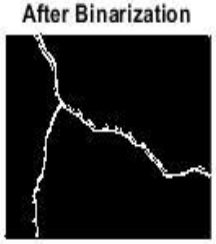

Proposed Method


Edge Detection


Edge Detection


After Sobel Filter

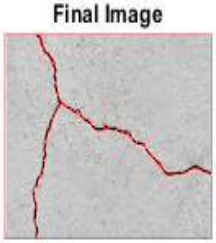

After 1st Morphological Operation

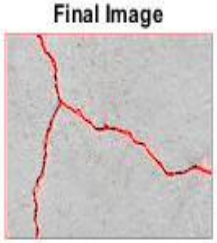

After Binarization

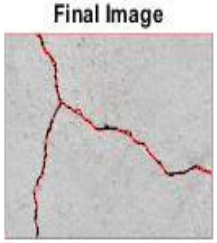

After Morphological Operation

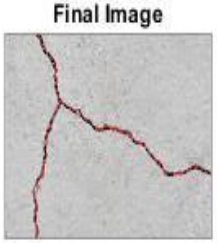

After Morphological Operation


After 2nd Morphological Operation


Final Image


Final Image


Final Image


Final Image

1st Gradient Method

Otsu Method

Dorafshan Method

Proposed Method

Edge Detection

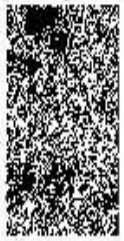
Edge Detection

After Sobel Filter

After 1st Morphological Operation

After Binarization

After Morphological Operation



After Morphological Operation

After 2nd Morphological Operation

Final Image

Final Image

Final Image

Final Image

# ΠΑΡΑΡΤΗΜΑ 2

Σε αυτό το παράρτημα ακολουθούν οι κώδικες που χρησιμοποιήθηκαν τόσο για την οπτική και θερμική επεξεργασία όσο και για την υλοποίηση του GUI μοντέλου.

Για την οπτική επεξεργασία:

```
clc;
clear all;
close all;
warning off;

tic;
Hx=[-1 -2 -1;0 0 0;1 2 1];
% Sobel filter to detect Horizontal lines
Hy=Hx';
position=[0 0];

RGBimage= imread('source');
RGBimage=imcrop(RGBimage);
RGB2=RGBimage;
figure(1)
imshow(RGB2);
I0=rgb2gray(RGBimage);
Gimage = I0(:,:);
I0min=min(min(I0));
I0max=max(max(I0));
t=0.5;
Ra=2;
if ((I0>(I0min+t*(I0max-I0min))))
           Ia=min(I0max,I0*Ra);

else
           Ia=max(I0min,I0*(1/Ra));
End

[level,EM] = graythresh(Ia);
BW = imbinarize(Ia,level);
BWrev=imcomplement(BW);

%% Orientation
stateAreaO=regionprops(BWrev,'Orientation');
area0=cat(1,stateAreaO.Orientation);
v0=find(area0<90);
L0=bwlabel(BWrev);
BWrev2=ismember(L0,v0);
BWrev3=bwmorph(BWrev2,'skel');

%% Area Crop
stateArea=regionprops(BWrev3,'Area');
area=cat(1,stateArea.Area);
L1=bwlabel(BWrev3);
```

```
v1=find(area<1200);

BWrev2=ismember(L1,v1);
BWrev4=bwmorph(BWrev2,'skel');

%% Crack Pixel Measurement
Crack_L=size(find(BWrev4),1);
value1=Crack_L;
% % Figures
figure(3)
imshow(BWrev2);

y=RGBimage;
y(BWrev4==1)=255;

figure(4)
imshow(y);
figure(5)

imshow(RGBimage);
title('Original Image');
```

Για την θερμική επεξεργασία:

```
close all;
clear all;
clc;


[ThermalImage,map]= imread(‘sources’);
I=ThermalImage;
I = (I*0.04)-273.15;
range=[min(I(:)) max(I(:))];
I=imcrop(I,hot);
grayIm=ind2gray(I,hot);
gaus=imgaussfilt(I,0.05);

figure(3)
imshow(grayIm)
colormap(gray)

FinalImage=I;
FinalImage=ind2gray(FinalImage,hot);
FinalImage(gaus<35)=255;
figure(4)
imshow(FinalImage)
colormap(hot)

h = fspecial('laplacian',0.4);

img_LOG = imfilter(FinalImage,h,'symmetric', 'conv');

grayImn=grayIm;

Gimage = grayImn(:,:);
I0min=min(min(grayImn));
I0max=max(max(grayImn));
```

```
t=0.5;
Ra=2;



if ((grayImn>(I0min+t*(I0max-I0min))))
          Ia=min(I0max,grayImn*Ra);

else
          Ia=max(I0min,grayImn*(1/Ra));
end

[level,EM] = graythresh(Ia);
BW = imbinarize(Ia,level);
BWrev=imcomplement(BW);

[level,EM] = graythresh(Ia);
BW = imbinarize(Ia,level);
BWrev=imcomplement(BW);

stateAreaO=regionprops(BWrev,'Orientation');
area0=cat(1,stateAreaO.Orientation);
v0=find(area0<90);
L0=bwlabel(BWrev);

BWrev2=ismember(L0,v0);


BWrev3=bwmorph(BWrev2,'skel');

stateArea=regionprops(BWrev3,'Area');
area=cat(1,stateArea.Area);
L1=bwlabel(BWrev3);
v1=find(area>200);

BWrev2=ismember(L1,v1);
BWrev4=bwmorph(BWrev2,'skel');

Crack_L=size(find(BWrev4),1);

value1=Crack_L;

figure(3)
imshow(BWrev2);
y=grayImn;
y(BWrev4==1)=255;
figure(4)
imshow(y);
```

Για το GUI :

```
classdef CrackAppFFF < matlab.apps.AppBase
    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                    matlab.ui.Figure
        BMophAreaSliderLabel        matlab.ui.control.Label
        BMophAreaSlider             matlab.ui.control.Slider
        UIAxes                      matlab.ui.control.UIAxes
        exitButton                  matlab.ui.control.Button
        resetzoomButton             matlab.ui.control.Button
        panButton                   matlab.ui.control.Button
        OpticaProcessingButton      matlab.ui.control.StateButton
        UIAxes2                     matlab.ui.control.UIAxes
        LoadOpticalImageButton      matlab.ui.control.StateButton
        LoadThermalImageButton      matlab.ui.control.StateButton
        PauseResumeButton           matlab.ui.control.StateButton
        TThresholdingSliderLabel    matlab.ui.control.Label
        TThresholdingSlider         matlab.ui.control.Slider
        zoomInButton                matlab.ui.control.StateButton
        next10secButton             matlab.ui.control.StateButton
        previous10secButton         matlab.ui.control.StateButton
        resetButton                 matlab.ui.control.StateButton
        OpenThermalFigureButton     matlab.ui.control.StateButton
        CropAndProcessButton        matlab.ui.control.StateButton
        ThermalProcessingButton     matlab.ui.control.StateButton
        RaThresholdingSliderLabel   matlab.ui.control.Label
        RaThresholdingSlider        matlab.ui.control.Slider
        Label                       matlab.ui.control.Label
        Switch                      matlab.ui.control.ToggleSwitch
        FussionButton               matlab.ui.control.Button
    end

    properties (Access = public)
        ValueSlider % Description
        changingValue
        x
        y
        a
        b
        zz
    end
    properties (Access = private)
    end

    methods (Access = private)

        function  OurProcc(app,~)
            Hx=[-1 -2 -1;0 0 0;1 2 1];

            Hy=Hx';

            RGBimage=app.x;
            orImage=rgb2gray(app.x);

           RGBimage=imcrop(RGBimage);
           RGB2=RGBimage;
```

```
figure(1)
imshow(RGB2);
I0=rgb2gray(RGBimage);
I0 = im2double(I0) ;
I0=abs(imfilter(I0,Hx))+abs(imfilter(I0,Hy));
figure(10)
imshow(I0);
Gimage = I0(:,:);
I0min=min(min(I0));
I0max=max(max(I0));
t=0.5;
Ra=2;


if ((I0>(I0min+t*(I0max-I0min))))
    Ia=min(I0max,I0*Ra);

else
    Ia=max(I0min,I0*(1/Ra));
end


[level,EM] = graythresh(Ia);
BW = imbinarize(Ia,level);
BWrev=imcomplement(BW);


% Orientation
stateAreaO=regionprops(BWrev,'Orientation');
area0=cat(1,stateAreaO.Orientation);
v0=find(area0<90);
L0=bwlabel(BWrev);

BWrev2=ismember(L0,v0);

BWrev3=bwmorph(BWrev2,'skel');
figure(15)
imshow(BWrev3);

%Area Crop
stateArea=regionprops(BWrev3,'Area');
area=cat(1,stateArea.Area);
L1=bwlabel(BWrev3);
v1=find(area<1200);

BWrev2=ismember(L1,v1);
BWrev4=bwmorph(BWrev2,'skel');


%Figures
figure(15)
imshow(BWrev2);

figure(5)
imshow(BWrev3);


y=RGBimage;
y(I0==1)=255;


figure(3)
imshow(BWrev3);
```

```
            figure(4)
            imshow(y);
            %SubPLots
            figure(3)
            imshow(RGBimage);
            title('Original Image');

        end
end



    methods (Access = private)
        % Code that executes after component creation
        function startupFcn(app, event)

            file2=dir('D:\diplwma\codesnw\window two\*.jpg');
            nf2=length(file2);
            pOptic=1;

            file=dir('D:\diplwma\codesnw\window two\2019_7_18_12_10_20\*.tiff');
            nf=length(file);
            pTherm=1;
            pauseS=false;
            flag=1;
            pause(10);
            while(flag==1)

                if
((app.LoadOpticalImageButton.Value==1)&&(app.LoadThermalImageButton.Value==0))
                    %app.LoadOpticalImageButtonPushed(app);
                    if app.PauseResumeButton.Value==1
                        if pauseS == false
                            while(1)
                                pause(0.001);
                                if app.PauseResumeButton.Value==0
                                    pauseS=true;
                                end
                                if pauseS == true
                                    break;
                                end
                            end
                        end
                    end
                    pauseS=false;
                    app.x=imread(fullfile('D:\diplwma\codesnw\window
two',file2(pOptic).name));

                    if app.OpticaProcessingButton.Value==1
                        app.OurProcc(app);

                    else
                        image(app.x,'Parent',app.UIAxes);
                        drawnow limitrate;

                    end
                    pOptic=pOptic+1;



                elseif
((app.LoadOpticalImageButton.Value==0)&&(app.LoadThermalImageButton.Value==1))
```

```
                    pauseS=false;
                    if app.PauseResumeButton.Value==1
                        if pauseS == false
                            while(1)
                                pause(0.001);
                                if app.PauseResumeButton.Value==0
                                    pauseS=true;
                                end
                                if pauseS == true
                                    break;
                                end
                            end
                        end
                    end
                    pauseS=false;
                    app.y=imread(fullfile('D:\diplwma\codesnw\window
two\2019_7_18_12_10_20',file(pTherm).name));
%                     app.y=imrotate(app.y,180);
                    rms=(app.y*0.04)-273.15;
                    app.y=rms;


                    imagesc(app.UIAxes2,rms);
                    colormap(app.UIAxes2,hot(256));
                    colorbar(app.UIAxes2,'west');
                    drawnow
                    pTherm=pTherm+1;

                elseif
((app.LoadOpticalImageButton.Value==1)&&(app.LoadThermalImageButton.Value==1))
                    if app.PauseResumeButton.Value==1
                        if pauseS == false
                            while(1)
                                pause(0.001);
                                if app.PauseResumeButton.Value==0
                                    pauseS=true;
                                end
                                if pauseS == true
                                    break;
                                end
                            end
                        end
                    end
                    pauseS=false;
                    app.x=imread(fullfile('D:\diplwma\codesnw\window
two',file2(pOptic).name));

                    if app.OpticaProcessingButton.Value==1
                        app.OurProcc(app);
                    else
                        image(app.x,'Parent',app.UIAxes);
                        drawnow limitrate;
                    end
                    pOptic=pOptic+1;
                    pauseS=false;
                    app.y=imread(fullfile('D:\diplwma\codesnw\window
two\2019_7_18_12_10_20',file(pTherm).name));
%                     app.y=imrotate(app.y,180);
                    rms=(app.y*0.04)-273.15;
                    app.y=rms;
                    imagesc(app.UIAxes2,rms);
                    colormap(app.UIAxes2,hot(256));
                    colorbar(app.UIAxes2,'west');
```

```
                drawnow
                pTherm=pTherm+1;

            else
                pause(10);
            end
            if (app.next10secButton.Value==1)&&(pTherm+10<nf)&&(pOptic+10<nf2)
                pTherm=pTherm+10;
                pOptic=pOptic+10;
            elseif (app.previous10secButton.Value==1)&&(pTherm-10>1)&&(pOptic-
10>1)
                pTherm=pTherm-10;
                pOptic=pOptic-10;
            end

            if app.resetButton.Value==1
                pTherm=1;
                pOptic=1;
            end

            if((pTherm>=nf)||(pOptic>=nf2))
                flag=0;
            end

            if(app.OpenThermalFigureButton.Value==1)
                 w=1;
                 h=1;
                 ind1=1;
                 ind2=1;
                if app.ThermalProcessingButton.Value==1

                    [maxValue, ~] = max(app.y(:));
                    [ind1,ind2] = find(app.y == maxValue);
                    ind1=max(max(ind1));
                    ind2=max(max(ind2));

                    w=ind1-10;
                    h=ind2-10;

                    if (w<1)
                        w=1;
                    end
                    if (h<1)
                        h=1;
                    end

                end


                figure(1)
                imagesc(rms);
                colormap(hot(256));
                rectangle('Position',[ind1 ind2 w h],'EdgeColor','b');
            end

            if  app.CropAndProcessButton.Value==1


                app.x=imcrop(app.x);
                RGBimage=app.x;
```

```
    I0=rgb2gray(RGBimage);
    Gimage = I0(:,:);
    I0min=min(min(I0));
    I0max=max(max(I0));


    t=app.TThresholdingSlider.Value;
    Ra=app.RaThresholdingSlider.Value;

    for ii=1:size(Gimage,1)
        for jj=1:size(Gimage,2)
            if ( I0(ii,jj) > (I0min+t*(I0max-I0min)))
                Ia(ii,jj)=min(I0max,I0(ii,jj)*Ra);

            else
                Ia(ii,jj)=max(I0min,I0(ii,jj)*(1/Ra));
            end
        end
    end


    level = graythresh(Ia);
    BW = imbinarize(Ia,level);
    BWrev=imcomplement(BW);

    %% Orientation
    stateAreaO=regionprops(BWrev,'Orientation');
    area0=cat(1,stateAreaO.Orientation);
    v0=find(area0<app.BMophAreaSlider.Value);
    L0=bwlabel(BWrev);

    BWrev2=ismember(L0,v0);


    BWrev3=bwmorph(BWrev2,'skel');
    stateArea=regionprops(BWrev3,'Area');
    area=cat(1,stateArea.Area);
    L1=bwlabel(BWrev3);
    v1=find(area>3);

    BWrev2=ismember(L1,v1);
    BWrev4=bwmorph(BWrev2,'skel');


    app.x=RGBimage;
    app.x(BWrev4==1)=255;
    figure();
    imshow(app.x);
    while(app.CropAndProcessButton.Value==1)
        pause(1);
    end
    close all;
    imshow(BWrev4);
    numWhite = sum(BWrev4(:));
    numberOfPixels = numel(BWrev4);
    decent=((numberOfPixels-numWhite)/numberOfPixels)*100;
    app.Gauge.Value=decent;
    app.EditField.Value=decent;
    clear Ia;

end
```

```
            end

    %%-------------------------------------------------------------------------------------
    %%                                    FUNCTIONS
    %%-------------------------------------------------------------------------------------
        end
        % Value changed function: LoadOpticalImageButton
        function LoadOpticalImageButtonPushed(app, event)

        end
        % Callback function
        function GraySccaleButtonPushed(app, event)

            global gray;

            gray=rgb2gray(app.x);
            imshow(gray,'Parent',app.UIAxes);
        end
        % Callback function
        function RGBButtonPushed(app, event)

            imshow(app.x,'Parent',app.UIAxes);
        end
        % Callback function
        function OurProcessingButtonPushed(app, event)


        end
        % Value changing function: BMophAreaSlider
        function BMophAreaSliderValueChanging(app, event)

        end
        % Value changed function: BMophAreaSlider
        function BMophAreaSliderValueChanged(app, event)

        end
        % Callback function: UIFigure, exitButton
        function exitButtonPushed(app, event)
            closereq

        end
        % Callback function
        function zoomInButtonPushed(app, event)
            zoom(app.UIAxes,'on');%start zoom mode

        end
        % Button pushed function: panButton
        function panButtonPushed(app, event)
            pan(app.UIAxes,'on');%start pan mode

        end
        % Value changed function: OpticaProcessingButton
        function OpticaProcessingButtonValueChanged(app, event)

        end
        % Value changed function: LoadThermalImageButton
        function LoadThermalImageButtonPushed(app, event)

        end
        % Button pushed function: resetzoomButton
```

```
        function resetzoomButtonPushed(app, event)
            ax=app.UIAxes;
            zoom(ax,'off');
            pan(ax,'off');
            ax.XLimMode='auto';
            ax.YLimMode='auto';

        end
        % Value changed function: zoomInButton
        function zoomInButtonValueChanged(app, event)
            zoom(app.UIAxes,'on');%start zoom mode

        end
        % Value changed function: previous10secButton
        function previous10secButtonValueChanged(app, event)
            value = app.previous10secButton.Value;

        end
        % Value changed function: next10secButton
        function next10secButtonValueChanged(app, event)
            value = app.next10secButton.Value;

        end
        % Value changed function: resetButton
        function resetButtonValueChanged(app, event)
            value = app.resetButton.Value;

        end
        % Value changed function: OpenThermalFigureButton
        function OpenThermalFigureButtonValueChanged(app, event)
            value = app.OpenThermalFigureButton.Value;

        end
        % Value changed function: CropAndProcessButton
        function CropAndProcessButtonValueChanged(app, event)
            value = app.CropAndProcessButton.Value;

        end
        % Callback function
        function EditFieldValueChanged(app, event)
            value = app.EditField.Value;
            app.UIFigure.AutoResizeChildren = 'on';

        end
        % Value changed function: ThermalProcessingButton
        function ThermalProcessingButtonValueChanged(app, event)
            value = app.ThermalProcessingButton.Value;

        end
        % Value changed function: TThresholdingSlider
        function TThresholdingSliderValueChanged(app, event)
            value = app.TThresholdingSlider.Value;

        end
        % Value changed function: RaThresholdingSlider
        function RaThresholdingSliderValueChanged(app, event)
            value = app.RaThresholdingSlider.Value;

        end
    end
    % App initialization and construction
    methods (Access = private)
        % Create UIFigure and components
```

```
        function createComponents(app)
            % Create UIFigure
            app.UIFigure = uifigure;
            app.UIFigure.Color = [0 0 0];
            app.UIFigure.Position = [100 100 681 467];
            app.UIFigure.Name = 'UI Figure';
            app.UIFigure.CloseRequestFcn = createCallbackFcn(app, @exitButtonPushed,
true);
            % Create BMophAreaSliderLabel
            app.BMophAreaSliderLabel = uilabel(app.UIFigure);
            app.BMophAreaSliderLabel.HorizontalAlignment = 'right';
            app.BMophAreaSliderLabel.FontColor = [1 0 0];
            app.BMophAreaSliderLabel.Position = [91 56 69 22];
            app.BMophAreaSliderLabel.Text = 'BMophArea';
            % Create BMophAreaSlider
            app.BMophAreaSlider = uislider(app.UIFigure);
            app.BMophAreaSlider.Limits = [1 1500];
            app.BMophAreaSlider.ValueChangedFcn = createCallbackFcn(app,
@BMophAreaSliderValueChanged, true);
            app.BMophAreaSlider.ValueChangingFcn = createCallbackFcn(app,
@BMophAreaSliderValueChanging, true);
            app.BMophAreaSlider.Position = [42 54 165 3];
            app.BMophAreaSlider.Value = 10;
            % Create UIAxes
            app.UIAxes = uiaxes(app.UIFigure);
            title(app.UIAxes, 'Optical Image')
            xlabel(app.UIAxes, 'Width')
            ylabel(app.UIAxes, 'Height')
            app.UIAxes.XTick = [0 0.2 0.4 0.6 0.8 1];
            app.UIAxes.Color = [0.149 0.149 0.149];
            app.UIAxes.BackgroundColor = [0.149 0.149 0.149];
            app.UIAxes.Position = [243 302 206 130];
            % Create exitButton
            app.exitButton = uibutton(app.UIFigure, 'push');
            app.exitButton.ButtonPushedFcn = createCallbackFcn(app, @exitButtonPushed,
true);
            app.exitButton.BackgroundColor = [0 0 0];
            app.exitButton.FontWeight = 'bold';
            app.exitButton.FontColor = [1 1 1];
            app.exitButton.Position = [530 431 100 22];
            app.exitButton.Text = 'exit';
            % Create resetzoomButton
            app.resetzoomButton = uibutton(app.UIFigure, 'push');
            app.resetzoomButton.ButtonPushedFcn = createCallbackFcn(app,
@resetzoomButtonPushed, true);
            app.resetzoomButton.BackgroundColor = [0.502 0.502 0.502];
            app.resetzoomButton.FontWeight = 'bold';
            app.resetzoomButton.Position = [23 202 79 22];
            app.resetzoomButton.Text = 'reset zoom';
            % Create panButton
            app.panButton = uibutton(app.UIFigure, 'push');
            app.panButton.ButtonPushedFcn = createCallbackFcn(app, @panButtonPushed,
true);
            app.panButton.BackgroundColor = [0.502 0.502 0.502];
            app.panButton.FontWeight = 'bold';
            app.panButton.Position = [22 241 68 22];
            app.panButton.Text = 'pan';
            % Create OpticaProcessingButton
            app.OpticaProcessingButton = uibutton(app.UIFigure, 'state');
            app.OpticaProcessingButton.ValueChangedFcn = createCallbackFcn(app,
@OpticaProcessingButtonValueChanged, true);
            app.OpticaProcessingButton.Text = 'OpticaProcessing';
            app.OpticaProcessingButton.BackgroundColor = [0 0 0];
```

```
            app.OpticaProcessingButton.FontWeight = 'bold';
            app.OpticaProcessingButton.FontColor = [1 0 0];
            app.OpticaProcessingButton.Position = [530 356 121 22];
            % Create UIAxes2
            app.UIAxes2 = uiaxes(app.UIFigure);
            title(app.UIAxes2, 'Thermal Image')
            xlabel(app.UIAxes2, 'Width')
            ylabel(app.UIAxes2, 'Height')
            app.UIAxes2.XTick = [0 0.2 0.4 0.6 0.8 1];
            app.UIAxes2.XTickLabel = {'0'; '0.2'; '0.4'; '0.6'; '0.8'; '1'};
            app.UIAxes2.YTick = [0 0.5 1];
            app.UIAxes2.YTickLabel = {'0'; '0.5'; '1'};
            app.UIAxes2.Color = [0.149 0.149 0.149];
            app.UIAxes2.BackgroundColor = [0.149 0.149 0.149];
            app.UIAxes2.Position = [243 126 206 137];
            % Create LoadOpticalImageButton
            app.LoadOpticalImageButton = uibutton(app.UIFigure, 'state');
            app.LoadOpticalImageButton.ValueChangedFcn = createCallbackFcn(app,
@LoadOpticalImageButtonPushed, true);
            app.LoadOpticalImageButton.Text = 'Load Optical Image';
            app.LoadOpticalImageButton.BackgroundColor = [0.9294 0.6902 0.1294];
            app.LoadOpticalImageButton.FontWeight = 'bold';
            app.LoadOpticalImageButton.Position = [25 162 126 22];
            % Create LoadThermalImageButton
            app.LoadThermalImageButton = uibutton(app.UIFigure, 'state');
            app.LoadThermalImageButton.ValueChangedFcn = createCallbackFcn(app,
@LoadThermalImageButtonPushed, true);
            app.LoadThermalImageButton.Text = 'Load Thermal Image';
            app.LoadThermalImageButton.BackgroundColor = [0.9294 0.6902 0.1294];
            app.LoadThermalImageButton.FontWeight = 'bold';
            app.LoadThermalImageButton.Position = [25 118 132 22];
            % Create PauseResumeButton
            app.PauseResumeButton = uibutton(app.UIFigure, 'state');
            app.PauseResumeButton.Text = 'Pause/Resume';
            app.PauseResumeButton.BackgroundColor = [0 0 0];
            app.PauseResumeButton.FontWeight = 'bold';
            app.PauseResumeButton.FontColor = [1 1 1];
            app.PauseResumeButton.Position = [23 431 101 22];
            % Create TThresholdingSliderLabel
            app.TThresholdingSliderLabel = uilabel(app.UIFigure);
            app.TThresholdingSliderLabel.HorizontalAlignment = 'right';
            app.TThresholdingSliderLabel.FontColor = [1 0 0];
            app.TThresholdingSliderLabel.Position = [266 62 129 22];
            app.TThresholdingSliderLabel.Text = '        T Thresholding     ';
            % Create TThresholdingSlider
            app.TThresholdingSlider = uislider(app.UIFigure);
            app.TThresholdingSlider.Limits = [0.01 1];
            app.TThresholdingSlider.ValueChangedFcn = createCallbackFcn(app,
@TThresholdingSliderValueChanged, true);
            app.TThresholdingSlider.Position = [275 56 148 3];
            app.TThresholdingSlider.Value = 0.5;
            % Create zoomInButton
            app.zoomInButton = uibutton(app.UIFigure, 'state');
            app.zoomInButton.ValueChangedFcn = createCallbackFcn(app,
@zoomInButtonValueChanged, true);
            app.zoomInButton.Text = 'zoomIn';
            app.zoomInButton.BackgroundColor = [0.502 0.502 0.502];
            app.zoomInButton.FontWeight = 'bold';
            app.zoomInButton.Position = [22 281 70 22];
            % Create next10secButton
            app.next10secButton = uibutton(app.UIFigure, 'state');
            app.next10secButton.ValueChangedFcn = createCallbackFcn(app,
@next10secButtonValueChanged, true);
```

```
            app.next10secButton.Text = 'next(10sec)';
            app.next10secButton.BackgroundColor = [0 0 0];
            app.next10secButton.FontWeight = 'bold';
            app.next10secButton.FontColor = [1 1 1];
            app.next10secButton.Position = [23 387 73 22];
            % Create previous10secButton
            app.previous10secButton = uibutton(app.UIFigure, 'state');
            app.previous10secButton.ValueChangedFcn = createCallbackFcn(app,
@previous10secButtonValueChanged, true);
            app.previous10secButton.Text = 'previous(10sec)';
            app.previous10secButton.BackgroundColor = [0 0 0];
            app.previous10secButton.FontWeight = 'bold';
            app.previous10secButton.FontColor = [1 1 1];
            app.previous10secButton.Position = [25 357 99 21];
            % Create resetButton
            app.resetButton = uibutton(app.UIFigure, 'state');
            app.resetButton.ValueChangedFcn = createCallbackFcn(app,
@resetButtonValueChanged, true);
            app.resetButton.Text = 'reset';
            app.resetButton.BackgroundColor = [0 0 0];
            app.resetButton.FontWeight = 'bold';
            app.resetButton.FontColor = [1 1 1];
            app.resetButton.Position = [25 322 73 20];
            % Create OpenThermalFigureButton
            app.OpenThermalFigureButton = uibutton(app.UIFigure, 'state');
            app.OpenThermalFigureButton.ValueChangedFcn = createCallbackFcn(app,
@OpenThermalFigureButtonValueChanged, true);
            app.OpenThermalFigureButton.Text = 'OpenThermalFigure';
            app.OpenThermalFigureButton.BackgroundColor = [1 0 0];
            app.OpenThermalFigureButton.FontWeight = 'bold';
            app.OpenThermalFigureButton.Position = [530 202 130 22];
            % Create CropAndProcessButton
            app.CropAndProcessButton = uibutton(app.UIFigure, 'state');
            app.CropAndProcessButton.ValueChangedFcn = createCallbackFcn(app,
@CropAndProcessButtonValueChanged, true);
            app.CropAndProcessButton.Text = 'CropAndProcess';
            app.CropAndProcessButton.BackgroundColor = [1 0 0];
            app.CropAndProcessButton.FontWeight = 'bold';
            app.CropAndProcessButton.Position = [530 321 123 22];
            % Create ThermalProcessingButton
            app.ThermalProcessingButton = uibutton(app.UIFigure, 'state');
            app.ThermalProcessingButton.ValueChangedFcn = createCallbackFcn(app,
@ThermalProcessingButtonValueChanged, true);
            app.ThermalProcessingButton.Text = 'ThermalProcessing';
            app.ThermalProcessingButton.BackgroundColor = [0 0 0];
            app.ThermalProcessingButton.FontWeight = 'bold';
            app.ThermalProcessingButton.FontColor = [1 0 0];
            app.ThermalProcessingButton.Position = [530 241 127 22];
            % Create RaThresholdingSliderLabel
            app.RaThresholdingSliderLabel = uilabel(app.UIFigure);
            app.RaThresholdingSliderLabel.HorizontalAlignment = 'right';
            app.RaThresholdingSliderLabel.FontColor = [1 0 0];
            app.RaThresholdingSliderLabel.Position = [515 62 94 22];
            app.RaThresholdingSliderLabel.Text = 'Ra Thresholding';
            % Create RaThresholdingSlider
            app.RaThresholdingSlider = uislider(app.UIFigure);
            app.RaThresholdingSlider.Limits = [0.1 2];
            app.RaThresholdingSlider.ValueChangedFcn = createCallbackFcn(app,
@RaThresholdingSliderValueChanged, true);
            app.RaThresholdingSlider.Position = [489 56 148 3];
            app.RaThresholdingSlider.Value = 1.1;
            % Create Label
            app.Label = uilabel(app.UIFigure);
```

```
            app.Label.HorizontalAlignment = 'center';
            app.Label.FontColor = [1 0 0];
            app.Label.Position = [565 106 25 22];
            app.Label.Text = '';
            % Create Switch
            app.Switch = uiswitch(app.UIFigure, 'toggle');
            app.Switch.Items = {'Hot', 'Cold'};
            app.Switch.Orientation = 'horizontal';
            app.Switch.FontColor = [1 0 0];
            app.Switch.Position = [567 164 45 20];
            app.Switch.Value = 'Cold';
            % Create FussionButton
            app.FussionButton = uibutton(app.UIFigure, 'push');
            app.FussionButton.BackgroundColor = [0 0 0];
            app.FussionButton.FontColor = [1 0 0];
            app.FussionButton.Position = [559 116 65 27];
            app.FussionButton.Text = 'Fussion';
        end
    end
    methods (Access = public)
        % Construct app
        function app = CrackAppFFF(varargin)
            % Create and configure components
            createComponents(app)
            % Register the app with App Designer
            registerApp(app, app.UIFigure)
            % Execute the startup function
            runStartupFcn(app, @(app)startupFcn(app, varargin{:}))
            if nargout == 0
                clear app
            end
        end
        % Code that executes before app deletion
        function delete(app)
            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```