NATIONAL TECHNICAL UNIVERSITY OF ATHENS
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
COMPUTER SCIENCE DIVISION
LABORATORY OF DISTRIBUTED SYSTEMS

Lunar Impact Flash Detection and Analysis Software

DIPLOMA THESIS

of

GEORGIA N. CHRISTOFIDI

Supervisor: Panayiotis D. Tsanakas

Professor of Computer Engineering, NTUA

Athens, September 2022

NATIONAL TECHNICAL UNIVERSITY OF ATHENS
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
COMPUTER SCIENCE DIVISION
LABORATORY OF DISTRIBUTED SYSTEMS

Lunar Impact Flash Detection and Analysis Software

DIPLOMA THESIS

of

GEORGIA N. CHRISTOFIDI

Supervisor: Panayiotis D. Tsanakas

Professor of Computer Engineering, NTUA

Approved by the examination committee on 31st August 2022.

(Signature)	(Signature)	(Signature)
. . . .	. . . .	. . . .
Panayiotis D. Tsanakas	Vassiliki Kantere	Andreas-Georgios Stafylopatis
Professor, NTUA	Assistant Professor, NTUA	Professor, NTUA

Athens, September 2022

(Signature)

. . . .

Georgia N. Christofidi

Electrical & Computer
Engineer

Περίληψη

Διαστημικά αντικείμενα όπως μετεωρίτες, κομήτες και αστεροειδείς εχουν τροχιές οι οποίες διασχίζουν το ηλιακό μας σύστημα και την τροχιά της γης, αποτελώντας έναν πιθανό κίνδυνο για τεχνητούς δορυφόρους, διαστημικά οχήματα που βρίσκονται σε τροχιά και αστροναύτες. Η ατμόσφαιρα της γης παρέχει προστασία απέναντι στα σώματα αυτά, ωστόσο στην επιφάνεια της Σελήνης, λόγω της απουσίας ατμόσφαιρας γύρω της, μπορούν να δημιουργούνται προσκρούσεις μέσω των οποίων μπορούν να μελετηθούν τα διαστημικά αντικείμενα. Για τους σκοπούς αυτούς, αναπτύχθηκε το πρώτο λογισμικό ανοιχτού κώδικα για τον εντοπισμό και την ανάλυση εκλάμψεων που προκαλούνται από προσκρούσεις διαστημικών αντικειμένων στην επιφάνεια της Σελήνης. Το λογισμικό αυτό αποτελείται από τρία μέρη, εκ των οποίων το πρώτο αποτελεί μια επέκταση για το πρόγραμμα FireCapture και η λειτουργία του είναι ο σύγχρονος εντοπισμός εκλάμψεων κατά τη διάρκεια των παρατηρήσεων. Το δεύτερο μέρος είναι ένα ασύγχρονο εργαλείο για τον εντοπισμό και την ανάλυση των γεγονότων αυτών και με τη χρήση του τρίτου μέρους του λογισμικού είναι δυνατός ο εντοπισμός των σεληνιακών συντεταγμένων των εκλάμψεων.

Λέξεις κλειδιά

Εκλάμψεις, πρόσκρουση, διαστημικά σώματα, Σελήνη, εντοπισμός, σεληνιακές συντεταγμένες, λογισμικό ανοιχτού κώδικα

Abstract

Near-Earth Objects have orbits that cross into the inner Solar System and intersect the Earth's trajectory, thus posing a potential hazard to artificial satellites, orbiting spacecraft, and astronauts. The atmosphere of the Earth provides protection against these objects, due to the fact that they burn up as they enter the atmosphere. However, the surface of the Moon, due to the absence of an atmosphere around it, remains susceptible to impacts by small NEOs and can be used to study their properties. On the grounds of this, we developed the first open-source Lunar Impact Flash Detection program for NEO detection, image analysis, and pattern recognition. The first part of the tool, built as a plugin in FireCapture, can capture impact flashes in real time. Afterwards, in a standalone application, it can characterize them by applying machine learning techniques and extract helpful information such as the impact flash shape, brightness and the exact lunar coordinates of the event.

Keywords

Impact flashes, Near-earth-objects, NEO, lunar surface, detection, localization, open-source software

To my family,
Konstantina, Nikos, Isidora

Ευχαριστίες

Αποτελώντας το επιστέγασμα των προπτυχιακών μου σπουδών στο τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου, η διπλωματική εργασία αυτή κατάφερε να περατωθεί με τη συμβολή πολλών ανθρώπων τους οποίους θα ήθελα να ευχαριστήσω.

Αρχικά, θα ήθελα να εκφράσω τις θερμές μου ευχαριστίες στην ομάδα του Εθνικού Αστεροσκοπείου Αθηνών και συγκεκριμένα στην κ. Άλκηστη Μπονάνου, στον κ. Αλέξη Λιάκο, στον κ. Ιωάννη Βέλλα Βελλίδη και φυσικά στον κ. Στέφανο Αχλάτη, με τους οποίους είχα την τιμή να συνεργαστώ σε ένα τόσο ενδιαφέρον θέμα και των οποίων την καθοδήγηση εκτίμησα βαθύτατα. Συγκεκριμένα, θα ήθελα να εκφράσω την ιδιαίτερη ευγνωμοσύνη μου προς τον κ. Αχλάτη, ο οποίος με βοήθησε και με ενέπνευσε ουσιαστικά, από την αρχή μέχρι το τέλος της ανάπτυξης του κώδικα και της συγγραφής της εργασίας, σε όλους τους τομείς, με τις καίριες παρατηρήσεις και την αμέριστη καθημερινή υποστήριξη του. Επιπρόσθετα, θα ήθελα να ευχαριστήσω βαθιά τη συμφοιτήτρια μου Ήβη Χατζή, με την οποία συνεργαστήκαμε στο πρόγραμμα αυτό και της οποίας τη συμβολή εκτιμώ και σέβομαι.

Φυσικά, θα ήθελα να ευχαριστήσω την οικογένεια μου για την συμπαράσταση και την ευκαιρία που μου παρείχε όλα αυτά τα χρόνια για να καταφέρω να ολοκληρώσω τις σπουδές μου στον τομέα αυτό. Επίσης, θα ήθελα να ευχαριστήσω ξεχωριστά όλα τα κοντινά μου άτομα που με την υποστήριξη τους με βοήθησαν να επιτύχω τους στόχους μου.

Τέλος, θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου κ. Παναγιώτη Τσανάκα του οποίου η επίβλεψη ήταν καθοριστική για την περάτωση αυτού του έργου.

Γεωργία Ν. Χριστοφίδη

Αθήνα, Σεπτέμβριος 2022

Contents

Περίληψη 6

Abstract 8

Ευχαριστίες 11

Chapter 0 - Εκτεταμένη Ελληνική Περίληψη 15
- 0.1 Εισαγωγή 15
- 0.2 Θεωρητικό Υπόβαθρο - Σχετική Βιβλιογραφία 15
  - 0.2.1 Αστροπαρατήρηση 15
  - 0.2.2 Επεξεργασία εικόνων 16
  - 0.2.3 Σχετική Βιβλιογραφία 17
- 0.3 Το λογισμικό σύγχρονου εντοπισμού εκλάμψεων 18
- 0.4 Το λογισμικό ασύγχρονου εντοπισμού και ανάλυσης εκλάμψεων 20
- 0.5 Το λογισμικό εντοπισμού των συντεταγμένων των εκλάμψεων 22

Chapter 1 - Introduction and Thesis Outline 24
- 1.1 Near-earth objects and lunar impact flashes observation 24
- 1.2 Thesis Outline 27

Chapter 2 - Theoretical Background 28
- 2.1 Astronomy and astrophotography 28
  - 2.1.1 Space objects and lunar events 28
  - 2.1.2 Astrophotography - Photo calibration 29
  - 2.1.3 The FireCapture Program 31
- 2.2 Image Processing Filters and algorithms 32
  - 2.2.1 Gaussian Filter 32
  - 2.2.2 Sobel Filter 33
  - 2.2.3 The Landau and the RANSAC Algorithm 33
  - 2.2.3 The Levenberg-Marquardt Algorithm 34

Chapter 3 - Technical Background - Relevant Works 35
- 3.1 The NELIOTA project 35
- 3.2 The MIDAS project 37
- 3.3 The Flash Project: Lunar impact flashes from France 39

Chapter 4: Online Detection 43
- 4.1 The Graphical User Interface (GUI) 43
- 4.2 Output 46
- 4.3 Algorithm 48
- 4.4 Software overview 53
- 4.5 System Requirements 53
- 4.6 Online detection use case 54
- 4.7 Capturing dark spots on the solar surface 55

Chapter 5: Offline Detection 57
5.1 The graphical user interface (GUI) 57
5.2 Input 58
5.3 Exceptions 59
5.4 Output 60
5.5 Algorithm 62
5.6 Software overview 63
5.7 System Requirements 64
5.8 Use case 65

Chapter 6: Localization 67
6.1 The graphical user interface 67
6.1.1 Event without metadata 69
6.1.2 Circle fitting 70
6.1.3 How to judge circle 71
6.1.4 Select hemisphere and orientation 75
6.1.5 Correlation 77
6.1.6 Complete localization 81
6.2 Output 81
6.3 Algorithm 82
6.4 Software overview 85
6.5 System Requirements 86
6.6 Use cases 86

Chapter 7 - Conclusions 91
7.1 Discussion 91
7.2 Future Work 92

References 93

Chapter 0 - Εκτεταμένη Ελληνική Περίληψη

0.1 Εισαγωγή

Διαστημικά σώματα όπως μετεωρίτες, κομήτες και αστεροειδείς προσελκύονται από τις βαρυτικές δυνάμεις κοντινών πλανητών και μπορούν να πλησιάσουν τη γη, αποτελώντας έναν πιθανό κίνδυνο. Ωστόσο, επειδή η ατμόσφαιρα της γης παρέχει μια προστασία απέναντι στα σώματα αυτά, για την παρατήρηση τους επιλέγεται η επιφάνεια της Σελήνης, στην οποία μπορούν να προσκρούουν προκαλώντας εκλάμψεις. Η μελέτη τους μπορεί να παρέχει πληροφορίες για τη μάζα τους, το μέγεθος τους, το υλικό τους και άλλες φυσικές παραμέτρους που αφορούν τα ίδια τα αντικείμενα, όσο και για την πρόσκρουση, την θερμοκρασία της και τους κρατήρες που δημιουργεί στην επιφάνεια της Σελήνης [1]. Με τις πληροφορίες αυτές, μπορούν να σχεδιαστούν κατάλληλα και να προστατευτούν τα διαστημικά οχήματα και οι σταθμοί που τίθενται σε τροχιά. Η επιβεβαίωση των εκλάμψεων που προκαλούνται από προσκρούσεις διαστημικών αντικειμένων στην επιφάνεια της Σελήνης παρουσιάζει ιδιαίτερες προκλήσεις, λόγω της απουσίας κοινού, αποδοτικού εργαλείου που να μπορεί να χρησιμοποιείται για παρατηρήσεις και ανάλυση των χαρακτηριστικών τους από επαγγελματίες και ερασιτέχνες αστρονόμους [2].

Με σκοπό την αύξηση των ωρών παρατηρήσεων και την δημιουργία μιας κοινής βάσης για τον διαμοιρασμό και την επιβεβαίωση των αποτελεσμάτων, αναπτύχθηκε ένα λογισμικό ανοιχτού κώδικα, στην γλώσσα προγραμματισμού Java, το οποίο αποτελείται ουσιαστικά από τρία διακριτά μέρη. Το πρώτο μέρος επιτρέπει στον χρήστη να εκτελεί τον εντοπισμό των εκλάμψεων συγχρόνως με τις παρατηρήσεις, ενώ τα δύο επόμενα αποτελούν ένα σύγχρονο εργαλείο εντοπισμού, ανάλυσης και εύρεσης των συντεταγμένων των εκλάμψεων που προκαλούνται από τις προσκρούσεις στην επιφάνεια της Σελήνης.

0.2 Θεωρητικό Υπόβαθρο - Σχετική Βιβλιογραφία

0.2.1 Αστροπαρατήρηση

Εκτός από τις πραγματικές εκλάμψεις που προκαλούνται από την πρόσκρουση διαστημικών σωμάτων στην επιφάνεια της Σελήνης, κατά τις παρατηρήσεις μπορεί να καταγραφούν γεγονότα που να σχετίζονται με την κίνηση δορυφόρων, κοσμικές ακτίνες, καμμένα εικονοστοιχεία ή αστέρια κοντά στα όρια της Σελήνης. Όταν ένας δορυφόρος διασχίζει το πεδίο της παρατήρησης, εμφανίζεται σαν ένα κομμάτι φωτεινής γραμμής που διαγράφει η τροχιά του. Η διάκριση τους από τις εκλάμψεις είναι εύκολη λόγω της διαφοράς μεταξύ του σχήματος, διότι στην περίπτωση των δορυφόρων το σχήμα που παρατηρείται είναι πιο πεπλατυσμένο από τις στρογγυλές εκλάμψεις. Στην περίπτωση των κοσμικών ακτίνων που προσπίπτουν πάνω στον φακό του αισθητήρα, η διαφορά φωτεινότητας που παρατηρείται είναι συνήθως πιο αχνή και το σχήμα πιο ακαθόριστο, εξαιτίας της γωνίας πρόσπτωσης [3]. Παράλληλα, τα καμμένα εικονοστοιχεία, αυτά δηλαδή που λαμβάνουν τη μέγιστη δυνατή τιμή τους και προκαλούνται από τα οπτικά στοιχεία μπορούν να διακριθούν επειδή μια πραγματική έκλαμψη αφορά σίγουρα μια περιοχή που καταλαμβάνει πάνω από ένα εικονοστοιχείο του καρέ στο οποίο παρατηρείται. Από την άλλη, τα αστέρια που εμφανίζονται κοντά στα όρια της Σελήνης, ενώ εκείνη μετακινείται στο πεδίο παρατήρησης μπορούν να καταγραφούν εσφαλμένα ως εκλάμψεις. Μετά τον εντοπισμό των εκλάμψεων, μπορούν να μελετηθούν οι φυσικές τους παράμετροι και οι παράμετροι των αντικειμένων που τα προκαλούν, με σκοπό την ανάλυση της συσχέτισης μεταξύ τους.

Κατά την αστροπαρατήρηση πραγματοποιείται επεξεργασία των εικόνων που καταγράφονται χρησιμοποιώντας τριών ειδών εικόνες, την εικόνα αντιστάθμισης, την εικόνα σκότους και την εικόνα απόκρισης [6]. Η καταγραφή της πρώτης γίνεται σε συνθήκες απόλυτου σκότους, χρησιμοποιώντας μικρό χρόνο έκθεσης και εξυπηρετεί στην αφαίρεση του αντισταθμίσματος που προστίθεται για την μείωση των στατιστικών διακυμάνσεων του φωτός στην τελική εικόνα που γράφεται. Για την λήψη της εικόνας σκότους, ο φωτοφράκτης της κάμερας διατηρείται κλειστός και με την αφαίρεση της από τα παρατηρούμενα καρέ, απαλείφεται ο θερμικός θόρυβος της κάμερας κατά την στιγμή της παρατήρησης [6,7]. Η εικόνα απόκρισης, με την οποία διαιρείται η τελική εικόνα που λαμβάνεται, χρησιμοποιείται για την αφαίρεση των αποτελεσμάτων της σκέδασης του φωτός λόγω της σκόνης και των ξένων αντικειμένων που μπορεί τυχόν να βρίσκονται στην κάμερα ή στον φακό του τηλεσκοπίου και στα φίλτρα, όπως και οποιασδήποτε ανομοιομορφίας που μπορεί να αφορά στον φωτισμό [7].

Για την αστροπαρατήρηση έχει αναπτυχθεί ένα χρήσιμο λογισμικό ανοιχτού κώδικα που παρέχει την ίδια διεπαφή για διαφορετικούς τύπους καμερών και ονομάζεται FireCapture. Το εργαλείο αυτό παρέχει στο χρήστη διαφορετικές δυνατότητες που αφορούν στην προβολή της εικόνας, τα φίλτρα και την προεπεξεργασία που μπορεί να εφαρμοστεί.

0.2.2 Επεξεργασία εικόνων

Η επεξεργασία των εικόνων μπορεί να γίνει με την εφαρμογή πλήθους φίλτρων, μεταξύ αυτών το Γκαουσιανό φίλτρο και το φίλτρο Sobel. Τα Γκαουσιανά φίλτρα μπορούν να χρησιμοποιηθούν για την απαλοιφή του θορύβου και των λεπτομερειών από τις εικόνες και η ευρεία χρήση τους οφείλεται μεταξύ άλλων και στην ευκολία της εφαρμογής τους [8]. Από την άλλη, τα φίλτρα Sobel μπορούν να χρησιμοποιηθούν για την ανίχνευση ακμών στις εικόνες, μειώνοντας την ένταση των υπολοίπων χαρακτηριστικών [9].

Ένας πολύ χρήσιμος αλγόριθμος για τον υπολογισμό των πυκνοτήτων των καταστάσεων ενός φυσικού συστήματος είναι ο αλγόριθμος Wang-Landau, ο οποίος βρίσκει εφαρμογές και στην όραση υπολογιστών [13]. Για παράδειγμα, μπορεί να χρησιμοποιηθεί για τον υπολογισμό του κέντρου ενός κύκλου σε μια εικόνα, δεδομένων συστάδων εικονοστοιχείων που βρίσκονται πάνω στην ακτίνα του. Επειδή ο αλγόριθμος αυτός βασίζεται σε έναν μη Μαρκοβιανό τυχαίο περίπατο, η σύγκλιση του δεν είναι δεδομένη σε κάθε περίπτωση, παρόλο που συνήθως επιτυγχάνεται. Για να εξασφαλιστεί η παραγωγή αποτελέσματος, μπορεί να συνδυαστεί με τον αλγόριθμο RANSAC ο οποίος υπολογίζει τις παραμέτρους ενός μοντέλου βασιζόμενος στην τυχαία δειγματοληψία από ένα σύνολο δεδομένων. Λόγω της απλότητας του, συγκλίνει πάντα δίνοντας ικανοποιητικά αποτελέσματα για πλήθος εφαρμογών [15].

Παράλληλα, κρίνεται χρήσιμο να αναφερθεί η χρήση του αλγορίθμου Levenberg-Marquardt, ο οποίος λύνει το μη γραμμικό πρόβλημα ελαχίστων τετραγώνων, δηλαδή της ελαχιστοποίησης του αθροίσματος των τετραγώνων των διαφορών μεταξύ του μοντέλου και του συνόλου δεδομένων [16, 17]. Χρησιμοποιεί συνδυασμό των μεθόδων των Gauss-Newton και της μεθόδου των παραγώγων, τείνοντας πιο πολύ στη μια ή στην άλλη μέθοδο ανάλογα με τις τιμές των παραμέτρων του μοντέλου [20].

0.2.3 Σχετική Βιβλιογραφία

Παρόμοια εργασία με το λογισμικό που αναπτύχθηκε στα πλαίσια αυτής της διπλωματικής έχει υλοποιηθεί μεταξύ άλλων από το πρόγραμμα NELIOTA, το οποίο χρηματοδοτείται από τον Ευρωπαϊκό Οργανισμό Διαστήματος και λαμβάνει χώρα στο Εθνικό Αστεροσκοπείο Αθηνών [1]. Ο σκοπός αυτού του προγράμματος είναι ο εντοπισμός και η μελέτη των προσκρούσεων διαστημικών αντικειμένων στην επιφάνεια της Σελήνης, με χρήση ενός τηλεσκοπίου 1.2 μέτρων που βρίσκεται στο Κρυονέρι, στην Πελοπόννησο. Τα κριτήρια που τίθενται για την επιβεβαίωση των προσκρούσεων είναι ο εντοπισμός του από κάθε μια από τις δύο κάμερες sCMOS που χρησιμοποιούνται από το τηλεσκόπιο, οι οποίες ανιχνεύουν σε διαφορετικό φάσμα [2]. Μέχρι τον Σεπτέμβριο του 2022 έχουν εντοπιστεί και επιβεβαιωθεί 149 εκλάμψεις χρησιμοποιώντας το λογισμικό του NELIOTA.

Παράλληλα, το λογισμικό MIDAS αναπτύσσεται από το 2009 με στόχο τον αυτόματο εντοπισμό εκλάμψεων που προκαλούνται από την πρόσκρουση μετεωροειδών στην Σεληνιακή επιφάνεια. Επίσης, γίνεται μια φωτομετρική ανάλυση των εκλάμψεων, των οποίων η επιβεβαίωση τους γίνεται εάν παρατηρηθούν από δύο τηλεσκόπια και πάνω από αυτά που βρίσκονται στο δίκτυο. Μόλις πραγματοποιηθεί ο πρώτος εντοπισμός, το γεγονός ανεβαίνει σε μια κοινή βάση δεδομένων, στην οποία γίνεται και η επιβεβαίωση των συμβάντων [21]. Ο αλγόριθμος του λογισμικού βασίζεται στην μεγάλη διαφορά της τιμής της φωτεινότητας μιας περιοχής εικονοστοιχείων της εικόνας και περιορίζεται στην περιοχή που ορίζεται από τα όρια της Σελήνης, μέσω της δημιουργίας μιας μάσκας που αποκλείει την περιοχή της εικόνας που αντιστοιχεί στον ουρανό. Η μάσκα αυτή μπορεί να προσαρμοστεί από τον χρήστη, όπως και το κατώφλι που χρησιμοποιείται για να ανιχνευθεί η διαφορά μεταξύ των τιμών των εικονοστοιχείων [5]. Επιπλέον, μπορούν να χρησιμοποιηθούν φίλτρα για την προεπεξεργασία των εικόνων, την απαλοιφή του θορύβου και την μείωση του χρόνου που χρειάζεται για την λήψη και την καταγραφή τους [23]. Μετά την επιβεβαίωση του εντοπισμού των συμβάντων, μπορεί να γίνει η εύρεση των πραγματικών συντεταγμένων τους, όπως και να υπολογιστεί η ενέργεια που ελευθερώθηκε κατά την πρόσκρουση ή η διάμετρος του κρατήρα που δημιουργήθηκε.

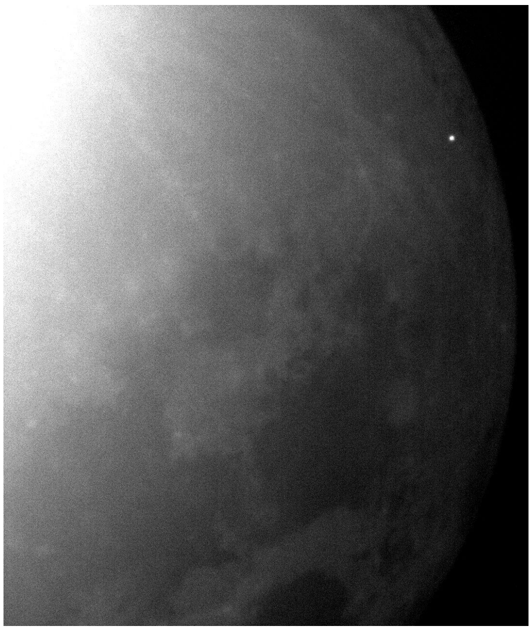
Ένα ακόμη λογισμικό που εξυπηρετεί την ίδια κατεύθυνση είναι το the Flash Project που λαμβάνει χώρα στη Γαλλία και εντοπίζει τις εκλάμψεις, τις συντεταγμένες τους στην επιφάνεια της Σελήνης και το θερμικό τους αποτέλεσμα [25]. Ο εντοπισμός και ο υπολογισμός των συντεταγμένων γίνεται άμεσα και αυτόματα, οπότε μετά αυτές μπορούν να χρησιμοποιηθούν για την εξέταση του νεοσύστατου κρατήρα από διαστημικό όχημα της NASA. Συγκεκριμένα, μπορεί να παρατηρηθεί η θερμοκρασία του, η διάμετρος του και να γίνει απόπειρα να συσχετιστεί η προέλευση του με κάποια ενεργή δραστηριότητα μετεωροειδών. Για τον εντοπισμό της έκλαμψης, πρώτα αφαιρείται το ανομοιογενές φόντο της εικόνας και η αντανάκλαση φωτός από το λαμπερό κομμάτι της Σελήνης. Στη συνέχεια, είτε εφαρμόζεται ένα κατώφλι του οποίου η τιμή αν ξεπεραστεί από κάποιο εικονοστοιχείο θεωρείται ότι έχει εντοπιστεί έκλαμψη, είτε εφαρμόζεται ένα Γκαουσιανό φίλτρο που μειώνει την ένταση των υπολοίπων χαρακτηριστικών και τον θόρυβο. Για την ανάλυση της πρόσκρουσης, υπολογίζεται μια μάσκα που αποκλείει την περιοχή που αντιστοιχεί στον ουρανό και χρησιμοποιώντας έναν δυαδικό χάρτη της προβολής της Σελήνης την συγκεκριμένη στιγμή, αντιστοιχίζονται τα χαρακτηριστικά της που φαίνονται στην εικόνα που έχει καταγραφεί [26]. Τα αποτελέσματα που δίνει το λογισμικό αυτό έχουν ελεγχθεί με βάση τα δεδομένα που παράγει το NELIOTA.

0.3 Το λογισμικό σύγχρονου εντοπισμού εκλάμψεων

Για τους σκοπούς του αυτόματου, σύγχρονου με τις παρατηρήσεις εντοπισμού των εκλάμψεων αναπτύχθηκε ένα λογισμικό που λειτουργεί ως επέκταση στο ήδη υπάρχον πρόγραμμα FireCapture, επεκτείνοντας τις δυνατότητες του με αυτό το χαρακτηριστικό και μειώνοντας σημαντικά τον χρόνο που χρειάζεται για να αποθηκευτούν οι πληροφορίες που αφορούν τα συμβάντα που παρατηρούνται.

Η επέκταση αυτή παρέχει το δικό της γραφικό περιβάλλον, που ενεργοποιείται και επιλέγεται μέσα από την διεπαφή του FireCapture. Συγκεκριμένα, προσφέρεται στον χρήστη η δυνατότητα να αλλάξει, μεταξύ άλλων, τον αριθμό των καρέ που θα αποθηκευτούν πριν και μετά από το συμβάν που θα καταγραφεί, την μορφή που θα έχουν αυτά και το κατώφλι που χρησιμοποιείται για τον εντοπισμό. Παράλληλα, όσο η επέκταση είναι σε λειτουργία, εμφανίζεται ένα παράθυρο που ενημερώνει τον χρήστη για το τι συμβαίνει στο παρασκήνιο, για παράδειγμα το στάδιο της καταγραφής, αν έχει εντοπιστεί κάποια έκλαμψη, όπως και την ώρα που συνέβη.

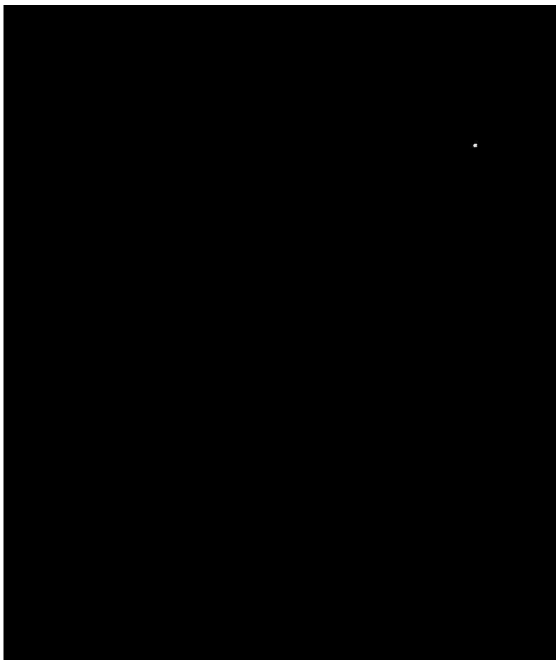
Η είσοδος του λογισμικού είναι η αλληλουχία καρέ που προέρχονται από την κάμερα που έχει συνδεθεί και εμφανίζονται στο παράθυρο της διεπαφής του FireCapture, ενώ η έξοδος του είναι ένας φάκελος μέσα στον οποίο υπάρχουν οι εικόνες που έχουν καταγραφεί, μαζί με ένα αρχείο κειμενικής μορφής που περιέχει σχετικές χρήσιμες πληροφορίες. Για κάθε ημέρα παρατηρήσεων δημιουργείται διαφορετικός φάκελος, μέσα στον οποίο βρίσκονται οι υποφάκελοι που αντιστοιχούν στα συμβάντα.



Εικόνα 0.1: Καταγεγραμμένη έκλαμψη από την επέκταση του προγράμματος FireCapture σε βίντεο που προέρχεται από τα δεδομένα του NELIOTA.

Ο αλγόριθμος σύμφωνα με τον οποίο λειτουργεί το πρόγραμμα λειτουργεί όπως περιγράφεται παρακάτω. Αρχικά, μετά τη διέλευση των πρώτων καρέ της παρατήρησης, δημιουργείται ένα μέσο καρέ το οποίο ενημερώνεται συνεχώς και για το οποίο χρησιμοποιούνται τα τελευταία δέκα καρέ. Αφού ο χρήστης πατήσει το κουμπί έναρξης αυτό το καρέ αφαιρείται από το τρέχον δημιουργώντας αυτό που αποκαλείται το καρέ "διαφορά".

Σε αυτήν την εικόνα, έχει αφαιρεθεί το φόντο και παραμένει μόνο κάποιος θόρυβος, η κατανομή του οποίου εξαρτάται από την κάμερα, την ατμόσφαιρα και τον σπινθηρισμό του φωτεινού κομματιού της Σελήνης. Εάν η τιμή κάποιου εικονοστοιχείου στο τελικό καρέ μετά την αφαίρεση ξεπερνάει ένα κατώφλι, το οποίο ορίζεται από το χρήστη και μπορεί να προσαρμοστεί κάθε στιγμή, τότε ενεργοποιείται ο εντοπισμός της έκλαμψης και ξεκινάει η διαδικασία καταγραφής του. Το πλήθος των εικόνων που θα καταγραφούν έχει καθοριστεί από τον χρήστη κατά την έναρξη της χρήσης του λογισμικού και αποτελείται συγκεκριμένα από διαφορετικό αριθμό καρέ πριν και μετά το συμβάν.



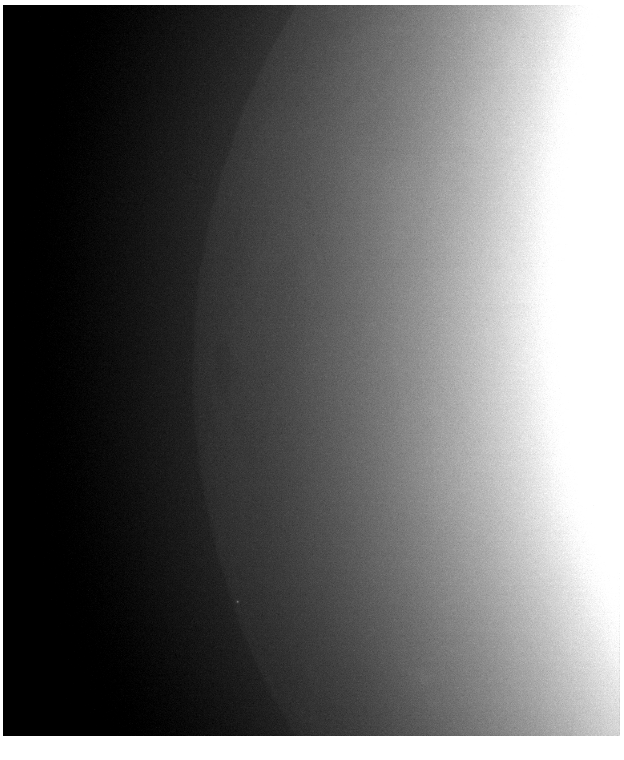
Εικόνα 0.2: Το καρέ "διαφορά" μεταξύ του παραπάνω καρέ που περιέχει την έκλαμψη και του αντίστοιχου μέσου καρέ.

Επιπρόσθετα, το λογισμικό αυτό μπορεί να εντοπίσει επιτυχώς ηλιακές κηλίδες, όταν το πεδίο της παρατήρησης μεταβάλλεται από μια ηλιακή περιοχή που δεν περιέχει σε μια περιοχή που εμφανίζονται. Αυτό συμβαίνει λόγω της αντιστοιχίας του προβλήματος των σεληνιακών εκλάμψεων από προσκρούσεις με τις ηλιακές κηλίδες και διότι αυτό που ανιχνεύεται ουσιαστικά είναι η απόλυτη τιμή της διαφοράς της τιμής της φωτεινότητας μεταξύ περιοχών εικονοστοιχείων.

0.4 Το λογισμικό ασύγχρονου εντοπισμού και ανάλυσης εκλάμψεων

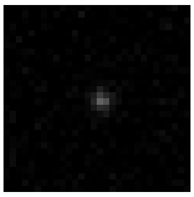
Μετά το λογισμικό σύγχρονου εντοπισμού και καταγραφής εκλάμψεων, το οποίο γράφει τα συμβάντα στον φάκελο των παρατηρήσεων, μπορεί να χρησιμοποιηθεί το ανεξάρτητο εργαλείο ασύγχρονου εντοπισμού και ανάλυσης των εκλάμψεων. Το συγκεκριμένο τμήμα του λογισμικού μπορεί να κατηγοριοποιήσει τα είδη των συμβάντων και να αποφανθεί εάν πράγματι έχει καταγραφεί κάποια έκλαμψη από πρόσκρουση διαστημικού σώματος στην επιφάνεια της Σελήνης.

Η διεπαφή του λογισμικού έχει σχεδιαστεί με κατάλληλο τρόπο που ελαχιστοποιεί τον χρόνο που χρειάζεται ο χρήστης για την εξοικείωση μαζί της, καθοδηγώντας τον και καθιστώντας την όσο το δυνατόν πιο απλή. Ως είσοδο δέχεται την έξοδο που έχει παράξει το online plugin ή κάποιο φάκελο που να έχει την ίδια μορφή. Συγκεκριμένα, μπορεί να δεχτεί ένα φάκελο που περιέχει μια αλληλουχία καρέ και ένα αρχείο με τα μεταδεδομένα που αφορούν το συμβάν που εμφανίζεται στις εικόνες. Ακόμα, μπορεί να δεχτεί και έναν υπερφάκελο που περιέχει πολλούς φακέλους συμβάντων. Εάν ο χρήστης δεν έχει χρησιμοποιήσει το σύγχρονο εργαλείο εντοπισμού, μπορεί σε κάθε περίπτωση να παράξει χειροκίνητα έναν παρόμοιο φάκελο και να τον επεξεργαστεί χωρίς πρόβλημα. Μέσω της γραφικής διεπαφής, ο χρήστης μπορεί επιπλέον να επεξεργαστεί όλες τις παραμέτρους που χρησιμοποιούνται για την κατηγοριοποίηση των συμβάντων, να ορίζει το μέγεθος της περιοχής ενδιαφέροντος που θα αναλυθεί για κάθε συμβάν, όπως και να επιλέξει την επεξεργασία των καρέ που εισάγει με εικόνες σκότους και απόκρισης.



Εικόνα 0.3: Παράδειγμα πρώτου καρέ συμβάντος το οποίο μπορεί να αναλυθεί χρησιμοποιώντας το ασύγχρονο εργαλείο εντοπισμού και ανάλυσης.

Η έξοδος του λογισμικού αποτελείται από έναν φάκελο που γράφεται μέσα σε κάθε φάκελο συμβάντος που έχει εισαχθεί στο λογισμικό για μελέτη, ο οποίος περιέχει τα αποτελέσματα της κατηγοριοποίησης, το μέσο καρέ που χρησιμοποιείται για την ανάλυση, την πρώτη εικόνα στην οποία έχει παρατηρηθεί το συμβάν, το καρέ της διαφοράς μεταξύ των προηγούμενων δυο και μια εικόνα της περιοχή ενδιαφέροντος.



Εικόνα 0.4: Παράδειγμα της περιοχής ενδιαφέροντος στο καρέ της διαφοράς μεταξύ της πρώτης εικόνας του συμβάντος και του μέσου καρέ που αντιστοιχεί σε αυτό. Η εικόνα αυτή αντιστοιχεί στην έκλαμψη της εικόνας 0.3.

Ο αλγόριθμος που υλοποιείται από το τμήμα του ασύγχρονου εντοπισμού περιγράφεται παρακάτω. Αρχικά, υπολογίζεται η μέση εικόνα από όλες τις εικόνες που έχουν καταγραφεί, η οποία αφαιρείται από την πρώτη εικόνα στην οποία εντοπίζεται το συμβάν. Στο καρέ που απεικονίζεται η διαφορά αυτή, δημιουργείται η περιοχή ενδιαφέροντος, η οποία είναι ένα τετράγωνο στου οποίου το κέντρο βρίσκεται το πιο φωτεινό εικονοστοιχείο του συμβάντος. όπως καταγράφηκε από το προηγούμενο τμήμα του λογισμικού. Αν το συμβάν δεν περιορίζεται σε ένα μόνο εικονοστοιχείο, δηλαδή αν δεν πρόκειται για κάποιο καμμένο εικονοστοιχείο που έχει προκαλέσει τον εντοπισμό, δημιουργείται μια μάσκα στη Σελήνη, προκειμένου να καθοριστεί αν το συμβάν αυτό εντοπίζεται μέσα ή έξω από τα όρια της. Για τον σκοπό αυτό εφαρμόζονται δύο Γκαουσιανά φίλτρα και στην εικόνα που προκύπτει μετά την εφαρμογή τους, βρίσκεται το όριο της Σελήνης, χρησιμοποιώντας ένα φίλτρο Sobel. Τελικά, το κέντρο του κύκλου βρίσκεται μέσω των αλγορίθμων Landau και RANSAC και με αυτόν τον τρόπο, εάν το συμβάν εντοπίζεται μέσα στα όρια της Σελήνης, επιβεβαιώνεται ότι αυτό δεν αντιστοιχεί σε κάποιο αστέρι.

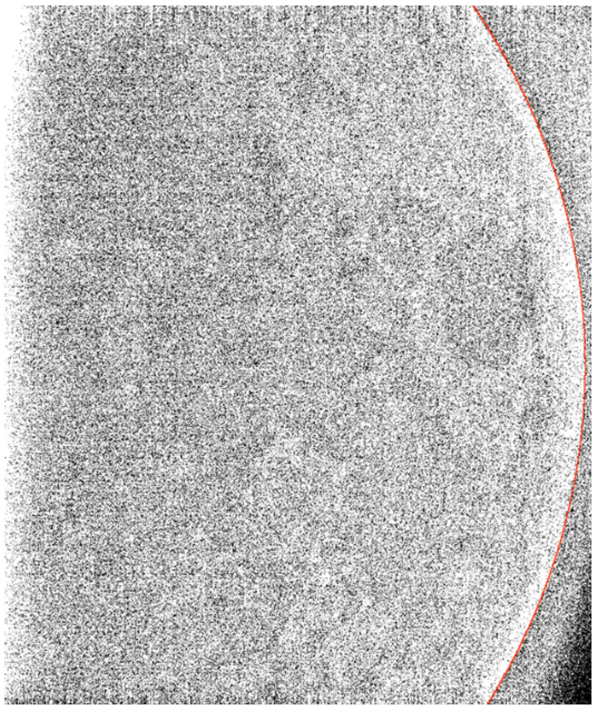
Για την κατηγοριοποίηση του συμβάντος, αν αυτό δεν ανήκει σε καμία από τις παραπάνω κατηγορίες, εφαρμόζεται ο αλγόριθμος Levenberg-Marquardt στην περιοχή ενδιαφέροντος του καρέ της διαφοράς. Από τις τελικές παραμέτρους του μοντέλου, κρίνεται το σχήμα του και η φωτεινότητα του οπότε διακρίνεται αν αντιστοιχεί σε πραγματική έκλαμψη από πρόσκρουση, σε δορυφόρο, σε κοσμική ακτίνα ή αν είναι τόσο αχνό που δεν μπορεί να αναλυθεί.

0.5 Το λογισμικό εντοπισμού των συντεταγμένων των εκλάμψεων

Στο τελευταίο τμήμα του λογισμικού, ο χρήστης μπορεί να εισάγει τις καταγεγραμμένες εκλάμψεις με σκοπό την εύρεση των πραγματικών συντεταγμένων τους στην επιφάνεια της Σελήνης. Η διεπαφή του αποτελεί μέρος του ασύγχρονου εργαλείου για τον εντοπισμό και την ανάλυση των εκλάμψεων. Κατά την εκκίνηση της διαδικασίας, ο χρήστης καλείται να εισάγει πληροφορίες που αφορούν την τοποθεσία από την οποία γίνονται οι παρατηρήσεις, ενώ διαβάζονται οι πληροφορίες ημέρας και ώρας από το αρχείο των μεταδεδομένων.

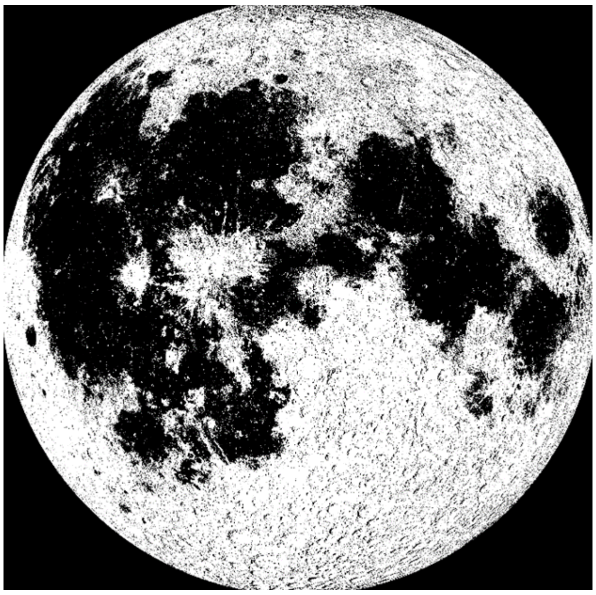
Η είσοδος του μπορεί να είναι είτε ένας φάκελος με κάποιο συμβάν είτε κάποιος υπερφάκελος που περιέχει διαφορετικά συμβάντα. Η έξοδος του λογισμικού αποτελείται από έναν φάκελο με τα αποτελέσματα της διαδικασίας, στον οποίο περιέχονται εικόνες που δημιουργούνται κατά την εκτέλεση του αλγορίθμου καθώς και ένα κειμενικό αρχείο με πληροφορίες που αφορούν λεπτομέρειες της διαδικασίας, όπως το κέντρο και η ακτίνα της Σελήνης που υπολογίζεται, παράμετροι του τηλεσκοπίου, η ημέρα και ώρα που εκτελέστηκε η διαδικασία και, φυσικά, οι συντεταγμένες της πρόσκρουσης.

Σύμφωνα με τον αλγόριθμο που περιγράφει την διαδικασία, πρώτα εντοπίζεται το όριο μεταξύ Σελήνης και ουρανού. Αυτό υπολογίζεται, εφαρμόζοντας δύο Γκαουσιανά φίλτρα στο μέσο καρέ που περιγράφει το συμβάν, για την αφαίρεση του φωτός και του θορύβου, με την παράμετρο σίγμα ίση με 25 και 5 αντίστοιχα. Στην εικόνα που προκύπτει από αυτή τη διαδικασία εφαρμόζεται ένα φίλτρο Sobel, σε συνδυασμό με ένα κατώφλι φωτεινότητας με σκοπό την διατήρηση των πιο φωτεινών εικονοστοιχείων της εικόνας, τα οποία είναι αυτά που βρίσκονται πάνω στο όριο μεταξύ της Σελήνης και του ουρανού. Μέσω του αλγόριθμου Landau σε συνδυασμό με τον αλγόριθμο RANSAC γίνεται δειγματοληψία στα εικονοστοιχεία αυτά για να βρεθεί το κέντρο του κύκλου που αντιστοιχεί στη δεδομένη περίμετρο.

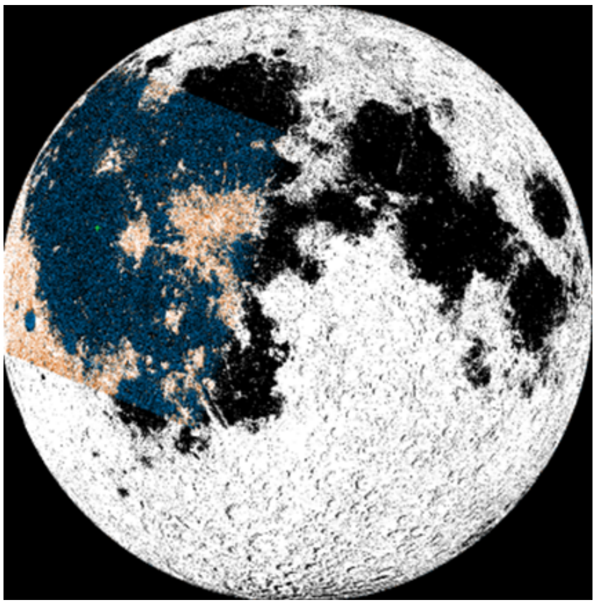


Εικόνα 0.5: Παράδειγμα εύρεσης ορίου μεταξύ Σεληνιακής επιφάνειας και ουρανού. Η περίμετρος του κύκλου που έχει υπολογιστεί σημειώνεται με κόκκινο.

Στη συνέχεια, μέσω ενός αιτήματος HTTP στο JPL Horizons API της NASA, με βάση τις πληροφορίες που έχουν τεθεί ως είσοδος, δημιουργείται μια ορθογραφική προβολή της Σελήνης, πάνω στην οποία αντιστοιχίζεται η εικόνα που έχει παρατηρηθεί από το τηλεσκόπιο του χρήστη. Αξίζει να σημειωθεί ότι ο χρήστης μπορεί να επέμβει σε κάθε στάδιο της διαδικασίας, μεταβάλλοντας τις παραμέτρους και εκτελώντας προσαρμογές για την αύξηση της ακρίβειας των αποτελεσμάτων.



Εικόνα 0.6: Παράδειγμα του δυαδικού χάρτη της προβολής της Σελήνης που δημιουργείται μέσω του JPL Horizons.



Εικόνα 0.7: Παράδειγμα αντιστοίχισης της εικόνας της πρόσκρουσης με την προβολή της Σελήνης.

Chapter 1 - Introduction and Thesis Outline

1.1 Near-earth objects and lunar impact flashes observation

Near-Earth Objects (NEOs) are comets and asteroids that have been attracted by the gravitational forces of nearby planets into orbits that allow them to approach the Earth, comprising a potential threat. Meteoroids are objects that range in size from dust grains to one meter in diameter and are cometary or asteroidal or cometary debris. The moon is bombarded with a rate of 7.5 meteoroids per hour that cause impact flashes, while Earth with a rate of approximately 100 meteoroids per hour [1]. The benefit of the detection of lunar impact flashes is, firstly, the calculation of the meteoroids' mass, size and other physical parameters in addition to those of the impacts themselves, such as temperature and craters on the surface. As for the meteoroids that reach the atmosphere and pass close to the Earth's orbit, the goal of those calculations is the protection of human civilization. The mid-term objective is about analyzing the frequency and the sizes of the meteoroids and small NEOs to conclude essential information for the protection of space vehicles and stations that could be affected or destroyed by large impacts. This can also contribute valuable information to the current and the future space missions to the Moon. In recent years, there has been a growing interest on behalf of European and international space agencies in Moon observations and lunar missions that may include manpower and robots.

The observation's results can be used to estimate the meteoroid time and space distribution on the moon and help space agencies acquire the tools to select the coordinates for building the first lunar space in a way that is less likely to be perturbed by a meteorite. Moreover, calculating the temperatures of the impact flashes and the kinetic energies of the objects that cause them will be crucial to the structural engineers for the armouring of space vehicles, regarding the shield that should be used for any permanent infrastructure on or beneath the lunar surface [2]. Hence, regular observations of the moon could be critical for the protection of the Earth and the development of space missions.

As many cameras and telescopes are used for the observations, the cross-validation of impact events across users is not an effortless task. Each camera provides a different Interface, has distinct write and read times, frames per second rate that it can capture, and along with the different telescope types, these all impose various hardware limitations on the computers.

Furthermore, each detected event could be a cosmic ray hit, a satellite passing in front of the moon, a field star very close to the lunar limb or an impact flash. The distinction between those is not clear to the eyes of an amateur and it is time consuming even for an expert user, having to examine the events one by one to validate them.

The storage limitations are a challenge as well, due to the fact that one night of observations could result in about 100GB of data and there is no simultaneous observation and detection of potential events, so everything is stored in the disk until it is analyzed [1].

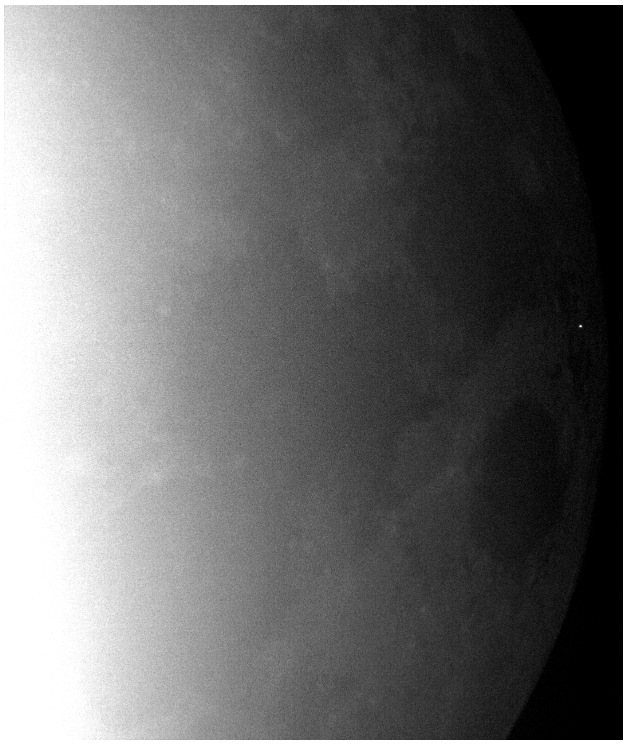


Image 1.1: A validated impact flash, captured by the Krioneri Telescope of the National Observatory of Athens.

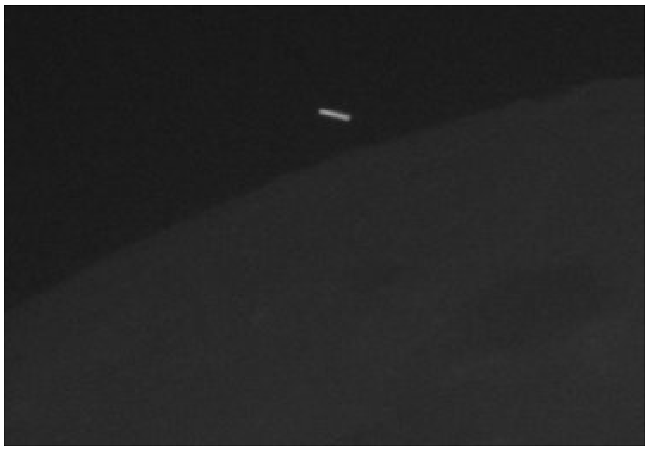


Image 1.2: A satellite outside of the lunar limb, captured by the MEADE telescope of the National Observatory of Athens in Penteli during the observation hours of the FDS team.

The optimal observation conditions depend on the phase of the moon, the atmosphere and the position of the telescope and camera. Firstly, the quality of image and the observation process is increased if the sunlit part of the moon is not contained in the field of view, as it affects the observations by increasing the contrast and causing false detections if the telescope is not stable and, therefore, sets observational burdens in terms of the lunar phase during which the observations can be performed. The minimum duration of the observations is approximately 20 min (at low brightness lunar phases, such as 0.2), while the maximum is approximately 4.5 hr (at lunar phases near 0.45) and between those phases of the Moon are the total available nights for lunar observations each month [3]. Also, the sky has to be as clear as possible, without clouds and pollution and the humidity must be below 90% in order to safely use the equipment.

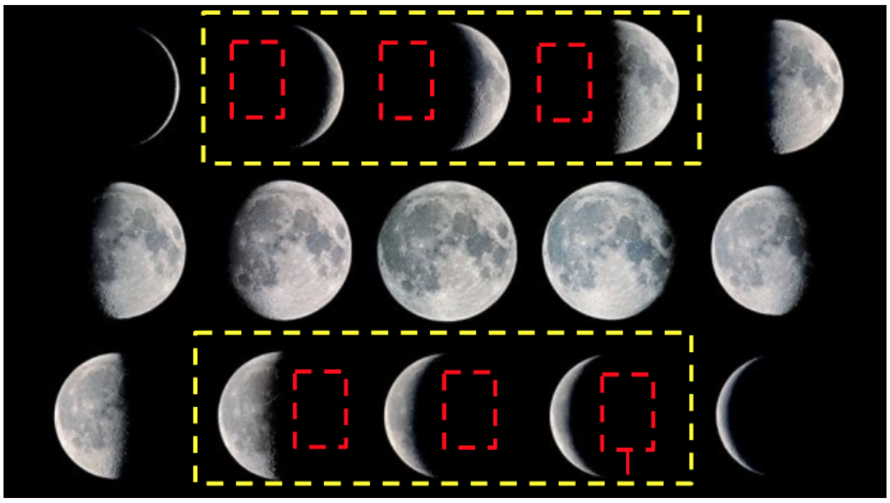


Image 1.3: The phases of the moon and the area that can be observed in each phase.

An open-source tool for lunar impact flash detection would result in more observation hours and it would provide a common ground for professionals and amateurs to share and validate their results. A dedicated software pipeline has been developed for the purposes of all the above. This open source software is fully developed in Java and performs the Observation and Online Detection simultaneously and the offline detection and classification and the localization of the events on the lunar surface. In the online domain all the frames that include events are stored in separate directories along with a certain number of previous and successive frames. Moreover, the software creates a directory that includes the events of each night of observations in chronological order, for each of them can perform a classification and find their coordinates on the lunar surface.

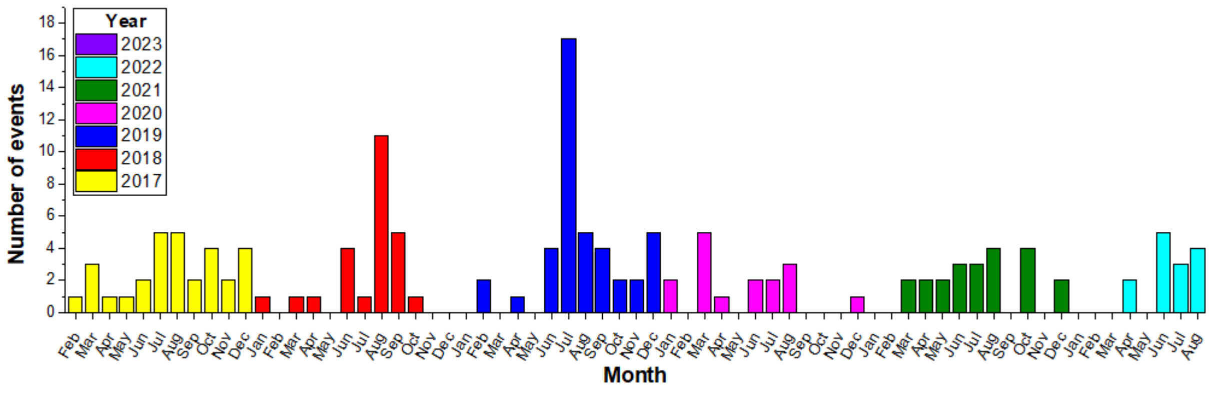


Image 1.4: Distribution of detected events in relation to time for the NELIOTA software.

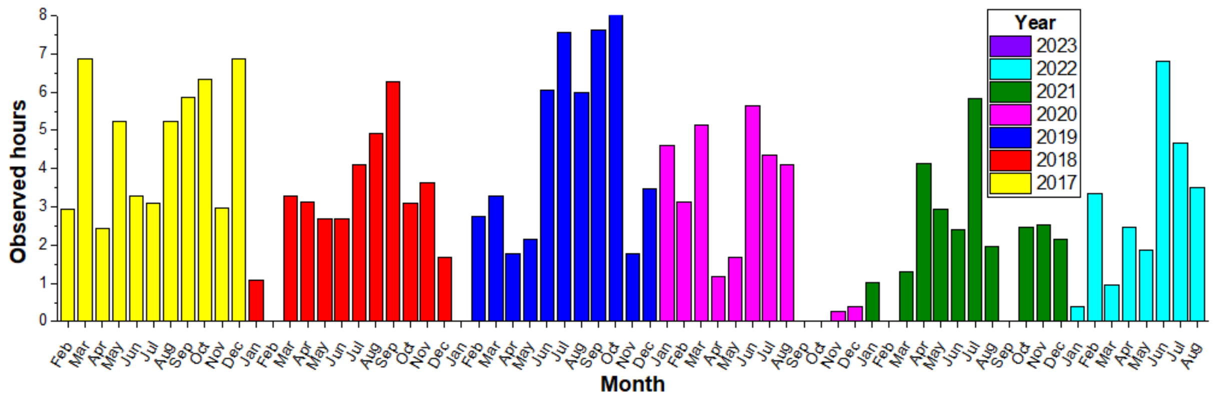


Image 1.5: Distribution of observed hours in relation time for the NELIOTA software.

1.2 Thesis Outline

Chapter 2: Theoretical Background, provides background knowledge to set the stage for the subsequent chapters, introducing the reader to concepts such as meteoroids, impact flashes and astrophotography. Relevant state-of-the-art works are mentioned.

Chapter 3: Technical Background, presents the observation and the testing conditions, as well as the programs and frameworks used to develop the software.

Chapter 4: Online Detection, introduces the online plugin that was developed for the purposes of live detection of lunar events.

Chapter 5: Offline Detection, presents the standalone tool used for the offline classification of lunar events.

Chapter 6: Localization, delves into the localization part of the standalone tool that provides the user with the actual coordinates of each event on the lunar surface.

Chapter 8: Conclusions, contains the results of the thesis, summarizing and providing an outlook into the future work.

Chapter 2 - Theoretical Background

2.1 Astronomy and astrophotography

2.1.1 Space objects and lunar events

Besides true impact flashes that are caused by comets, asteroids and meteoroids, the events that can be falsely recorded may refer to satellites, cosmic rays, hot pixels or stars, due to their resemblance to impact flashes. In the case of true impact flashes, the origin of the flash is the impact, which is caused by the fall of a projectile as those mentioned above on the surface of the moon. Because of the lack of atmosphere of the moon, even small meteoroids can collide at increased speeds [4]. Additionally, the distribution of the projectiles that collide with the lunar surface can vary as the Earth orbits the Sun due to the crossing of our planet through cometary/asteroidal debris or meteoroid streams [1].

Meanwhile, satellites are artificial space objects which are placed into orbit in outer space and communicate with some ground stations. When a satellite crosses the field of view of a telescope, a sudden change of luminosity is observed in the form of a line that marks the satellite trajectory. In each frame, only a part of this line appears, making the satellite resemble an impact flash. On the other hand, cosmic rays are high-energy protons and atomic nuclei that navigate in outer space at about the speed of light and can hit the telescope and camera lens, causing sudden changes in the brightness of small areas of pixels [12]. Mainly, cosmic rays bounce on the sensors of the telescope at different angles, so the shapes of the pixel areas that the change of luminosity is noted, differs largely from that of an impact flash that has a round shape and they can be quickly discarded. However, in some cases that can bounce on the lens with an almost perpendicular angle making their shape almost round, resembling more to stars and flashes [3].

Besides the above, that can be falsely detected as flashes caused by impactors on the lunar surface, hot pixels can also cause sudden changes in the value of the pixels. In this case, a single pixel temporarily reaches the maximum value it can take, due to electrical charges that leak into the sensor wells. They are more common during seasons with warmer weather and they cannot be predicted. The crucial difference from impact flashes is that a hot pixels refer to only one pixel at a time, as opposed to impact flashes that occupy several pixels in the frames that they appear. The last category of the events that can be commonly mistaken as impact flashes are stars occurring near the lunar limb, which is the term to refer to the edge of the visible lunar surface as observed from the Earth. It should be noted that the lunar observations occur to the hemisphere of the Moon that is facing towards Earth and especially on the non-sunlit part of this hemisphere, due to the fact that in this area, which is darker, impact flashes are more visible [1]. Also, the terminator area, the line that divides the illuminated from the non-sunlit part of the moon should be avoided during the observations as the glare from the sunlit part of the moon can affect the quality of the images obtained.

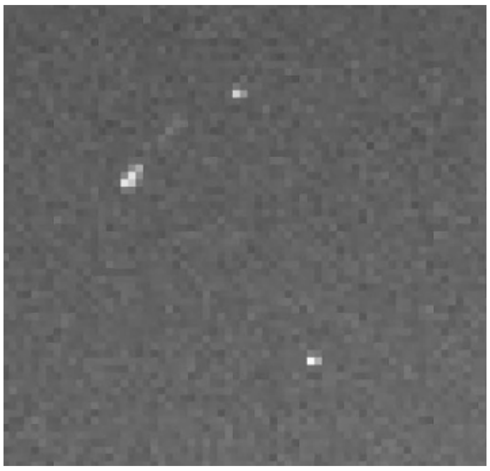


Image 2.1: Example of a cosmic ray. Image taken from NELIOTA data.

After the detection of the flashes, their physical parameters can be calculated. These can be the mass of the impactor, its density, its radius, as round objects are assumed, its velocity, and kinetic energy at the time of the impact. This kinetic energy when the impactor strikes the lunar surface is converted to luminous energy that generates the flash while some material of the projectile and the lunar surface melts and the temperature of the droplets increases. Another parameter that can be studied is the energy that causes a possible excavation of a crater, its temperature at the time of the creation and its diameter [2].

The energy that is radiated at the time of the flash has been correlated with the masses of the projectiles. More specifically, impactors of bigger size produce brighter flashes. However, until now there has been no similar correlation between the masses of the impactors and the temperature evolution and development of the collisions, for the reason that this also depends on the heat capacity and the thermal conductivity of the materials of the projectiles and the surface of the moon [5].

2.1.2 Astrophotography - Photo calibration

Firstly, an attempt to clarify some terms related to astrophotography will be realized and then, a mention to the calibration of the images will be made. The term gain refers to the magnitude of amplification a system can reach, meaning the constant factor that the received signal will be multiplied by. In addition, the term exposure time that can also be found as shutter speed is the amount of time that the camera gathers light from the observed sample. What's more, the frames per second (fps) is the rate at which a camera can display or capture consecutive frames.

The image processing that is performed on the images obtained by a CCD camera contains three stages. Dark, bias and flat field images are captured and then the first two are subtracted from the frame and this image is divided by the third one to obtain the final image.

The bias frame is captured in complete darkness, with a very little exposure time before and directly after the observations. To eliminate the statistical fluctuations of light on the final image, a constant is added to it that has to be subtracted afterwards. As a bias image for the calibration, the median of several exposures is taken, to reduce the noise induced during their capturing [6].

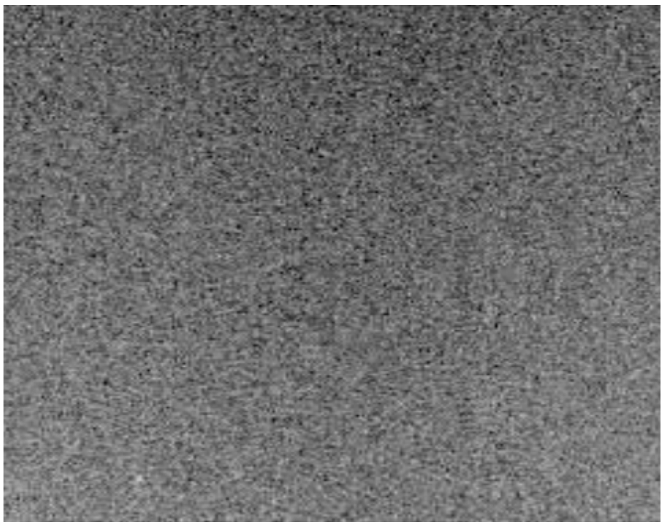


Image 2.2: An example of a bias frame. [diplomatiki aleksi]

The dark frame is captured with the dew shield of the telescope closed, with the same exposure time as the frames that need to be calibrated. This image corresponds to the thermal noise of the camera and it is proportional to the temperature. For the calibration to be strictly correct, after each frame that is captured, a dark frame should be captured too, in order to have the exact thermal noise at all times [6]. Obviously, this cannot be done, because it would result in a significant loss of observation time, thus, a number of dark frames is recorded before or after the observations, with the same exposure time that is used throughout the observations and their mean frame is used. This average dark frame is then subtracted from all the recorded frames of the observations. It is recommended not to use dark frames for calibrations for exposure times smaller than 5 sec, in order not to induce noise to the images [7].

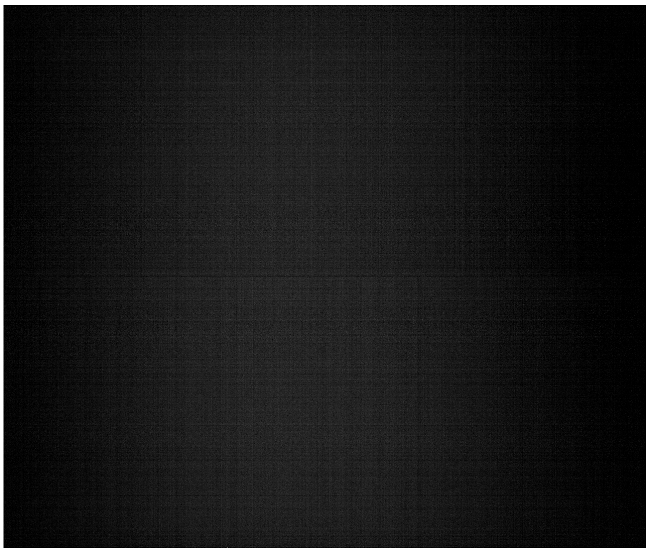


Image 2.3: An example of a dark frame. Image obtained from the NELIOTA data.

The need for calibration with a flat field image arises from the fact that not every pixel in an CCD camera has the same sensitivity nor the same response time. Some pixels produce more photoelectrons than others, resulting in diverse brightness in the image. In the flat field image the results of the light scattering due to grains of dust on the surface of the optics of the telescope, such as the filters and the camera lens can be noted that deteriorate the quality of the image [6]. Moreover, the uneven lighting of the field of view, which is called vignetting, causes the brightness of the image to reduce from the center to the edges of the image. The way to obtain the flat images is the following.

Numerous flat field images are obtained, with the dew shield of the telescope closed, so that each pixel is located in the exact same place during the sky observations and the flat image capturing. An evenly illuminated surface must be captured, with the appropriate exposure time so that the noise is insignificant and no pixel reaches its maximum value [7]. This exposure time usually ranges from 2 to 3 seconds. An ideal surface that resembles the one described is for example the sky during pre-dawn when no stars appear, with the same telescope focus that is used during the observation, while the camera is cool so that no extra thermal noise appears, for each filter that will be used. Flat field images should be obtained for each night of observations, but if the target doesn't change, the same images can be used for some consequent different observation sessions [7].

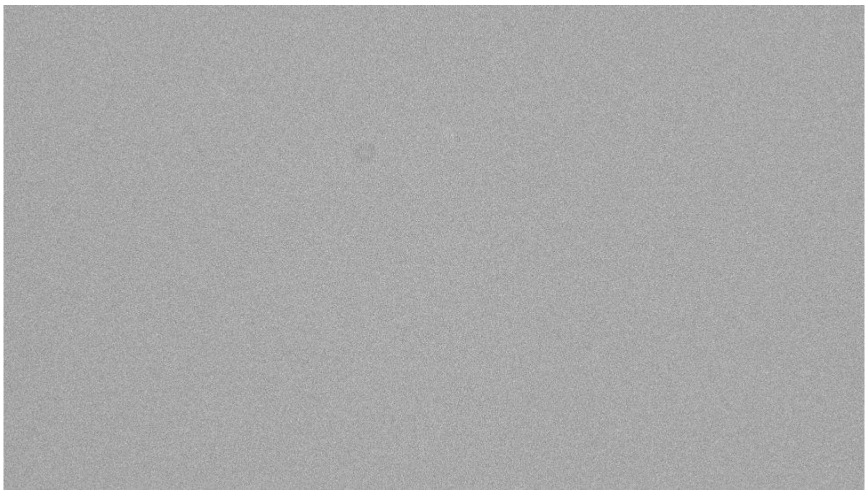


Image 2.4: An example of a flat field frame. Image obtained during the observations that took place for the testing of the FDS on the 8th of April using a ZWO ASI290MM camera.

2.1.3 The FireCapture Program

FireCapture is a free open source JAVA-based software package for planetary and lunar image capturing. It is a feature-rich program for astrophotography for high speed digital video imaging of the planets and other solar system objects that provides a common interface from which the user can control the operation of many different types of cameras. The latest version of the software can operate for Windows, Linux, Mac and Raspberry Pi. Through its interface, the user can capture planetary images at specific directories with the format that they wish, choosing between AVI and SER for videos and BMP, FIT and JPEG for images. There are options for each planetary body that is observed. Via the control panel, the gain and the exposure time of the image can be adjusted, in order to alter the received signal. The focus of the image can also be changed, along with some settings regarding, among others, the applied filters and the performance of the tool. In the preprocessing area, the user can choose from a variety of plugins that have already been developed. The variety of the features that this software provides, as well as the ability to enhance it with custom plugins, make it a very popular software among both experienced and amateur astronomers.

2.2 Image Processing Filters and algorithms

2.2.1 Gaussian Filter

Gaussian filters play an important role in filtering different kinds of images to produce a blur effect and eliminate noise and detail. They are general-purpose filters that the simplicity of their algorithm, the ease of implementation, and the robustness of the results makes them the first choice for filtration in many applications. Gaussian filters provide an approach for the separation of the roughness and waviness components from a primary surface and they can be applied to the input surface by convolving the measured surface with a Gaussian weighting function [8]. The Gaussian weighting function has the form of a bell-shaped curve as defined by the following equation, where x, y are the two-dimensional distance from the center (maximum) of the weighting function and σ is the standard deviation of the distribution.

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

In one dimension, the Gaussian function is:

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

The mean of the distribution is assumed to be equal to zero. The standard deviation (sigma, sd) of the Gaussian function has a crucial role in its behavior. The blur effect increases as the standard deviation takes larger values. When working with images the two dimensional Gaussian function is used that is simply the product of two 1D Gaussian functions, one for each direction. Another very important value of the Gaussian fit is the full width at half maximum (FWHM), which is the width of a spectrum curve at the point between those points on the y-axis where the amplitude has half of its maximum value. The FWHM and the standard deviation have the following relation.

$$\mathrm{FWHM} = 2\sqrt{2\ln 2}\,\sigma \approx 2.355\,\sigma.$$

2.2.2 Sobel Filter

The Sobel filter is used in image processing to perform edge detection, resulting in an image with emphasized edges and the other features significantly eliminated or smoothed. This operator performs a two dimensional spatial gradient approximation on the image's intensity function, meaning that it is a discrete differentiation operator [9]. The Sobel edge detector uses two of 3 x 3 convolution masks, one for measuring the gradient in the x dimension and the other respectively for the y dimension. It is based on the assumption that the edges of the grayscale images can be found in places where there is a discontinuity in the brightness of adjacent pixels or a very abrupt intensity gradient in the image. This is called the gradient method for edge detection which marks the edges by searching for the maximum and minimum in the first derivative of the intensity value of the frame. This method is influenced by noise in pictures, in the sense that it highlights it as an edge. Other filters that use this method, such as Roberts and Prewitt also have this property, while the detected features on the images have sharp edges [10].

Through the detection of an image's edges its features can be more efficiently understood, by recognizing object boundaries. Furthermore, it can be used to discern areas of an image to segment it to the different objects that appear in it. It also minimizes the amount of data that is stored, reducing the size of the image by preserving only the most relevant information, such as the structural properties of its features [11]. In the filtered images redundancies are eliminated and important events are captured, without seriously affecting the quality of the frames.

2.2.3 The Landau and the RANSAC Algorithm

The Wang-Landau (WL) algorithm is a Monte Carlo algorithm that estimates the densities of states of a physical system required to perform a multicanonical simulation and it is also designed to perform numerical integration in high dimensional spaces [13]. Its applications contain a variety of systems. For instance, it could be used in computer vision to determine the center of a circle, by obtaining standard samples of each connected component of points that lie on the perimeter. Its convergence has been proved in certain cases and its rate is strongly connected to the affinity properties of the underlying non-Markovian random walk. In the challenging cases that the algorithm does not converge, it can be combined with the RANSAC algorithm, to yield a result.

The RANSAC (RANdom SAmple Consensus) algorithm is a general parameter estimation technique developed to cope with a vast number of extreme values that differ from each other in the observed data. It is a learning technique based on random resampling that generates candidate solutions by using the smallest set of data points required to calculate the underlying model parameters and proceeds to enlarge this set with consistent data points [14]. The steps of the algorithm are the following and they are iteratively repeated until it converges [15].

- First step: Arbitrary selection of the smallest subset of points demanded to estimate the parameters of the model.

- Second step: Fit of the parameters of the model based on the points selected during step one.

- Third step: Assessment of the number of the points from the entire data set that are consistent with the model given a certain tolerance.

- Fourth step: If the fraction of the number of data points which are not outliers over the total number points in the set exceeds a predefined threshold value, recalculation of the parameters of the model again using all the detected inliers and algorithm termination.

- Fifth step: If not, loop through steps 1 to 4, having set a maximum number of iterations.

2.2.3 The Levenberg-Marquardt Algorithm

The Levenberg-Marquardt algorithm is a widely used optimization technique that was created to tackle the nonlinear least squares problems [16, 17]. Least squares problems can appear when fitting a parameterized mathematical model to a given dataset by reducing the sum of the squared difference between the model function and the input data points. The LMA minimizes the sum of the squared errors between the fit function and the input by performing a series of careful updates to the values of the parameters of the fit. It achieves to outperform simple gradient descent and other conjugate gradient algorithms in many cases by incorporating a merge of the gradient descent method and the Gauss-Newton method [18]. In the first algorithm, the sum of the squares of the errors is reduced by updating the model parameters in the steepest-descent direction [19]. According to the Gauss-Newton technique, the sum of the squares of the errors is minimized by concluding that the least squares function is locally quadratic in the parameters, and calculating the minimum value of this quadratic. In the case the parameters have a big difference from their ideal value, the LMA behaves in a way that resembles a gradient-descent technique more. To the contrary, if the parameters' values approach the optimal the Levenberg-Marquardt algorithm behavior is closer to the Gauss-Newton method [20]. By studying the parameters of the fit, conclusions about the distribution of the data set can be made.

Chapter 3 - Technical Background - Relevant Works

In this chapter, we will present some relevant works from the field of lunar impact flash detection that have been developed internationally in the past years.

3.1 The NELIOTA project

NELIOTA is an ongoing project that takes place at the National Observatory of Athens (NOA), is funded by the European Space Agency (ESA) and has the aim of determining the distribution and frequency of small near-earth objects (NEOs) by monitoring lunar impact flashes [1]. Until the end of August 2022, more than 2033 days since the start of observations have passed and more than 224.31 hours of lunar observations have been realized. During these observations, there have been produced and stored around 176.58 TB of lunar images, while 149 NEO lunar impact events have been confirmed [1]. For all of the above the 1.2 m Kryoneri telescope in Peloponnese, in Greece is used [3]. The telescope possesses two sCMOS cameras (Andor Zyla 5.5), the first camera records in the red (Rc) and the other in the near-infrared (Ic) passbands.

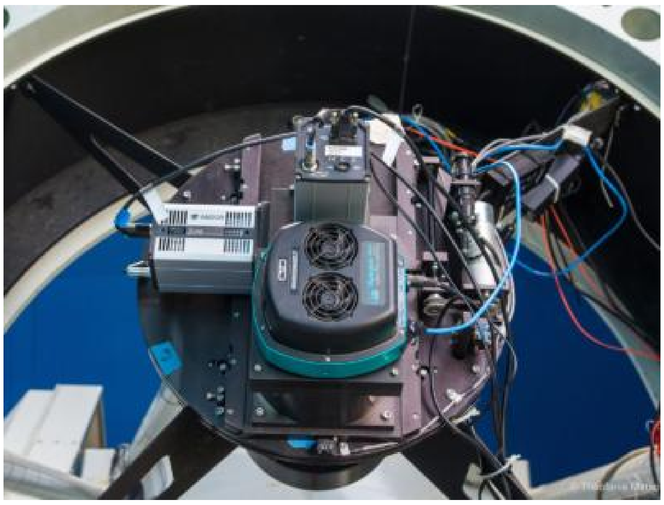


Image 3.1: The Krioneri telescope.

Whatever the detection domain of NELIOTA has achieved to detect, such as impact flashes, satellites, cosmic rays or anything that crosses the field of view, is called an event. These events are analyzed in order to be validated as impact flashes [1]. The first criterion that is used to validate the flashes is that both cameras have detected it at the same pixel area and the second one is the absence of movement of the event on successive frames. According to the camera in which the event has been detected, and the above criteria, four cases can be produced.

- Case 1: Both criteria are satisfied. The event is classified as a validated lunar impact flash.
- Case 2: Neither criterion is satisfied. The event is detected in various frames of only one camera. The event is discarded, as it is not a true impact flash.

- Case 3: The first criterion is satisfied but the second is not. If there is movement it means that a satellite, an airplane, or a bird is crossing the field of view. The event is classified as false.

- Case 4: The second criterion is satisfied, when the events consist of various frames or the event is singleframe, but the first criterion is not fulfilled. When the event is detected only by the R camera, Case 4R arises and similarly Case 4I for I camera. This case is usual, due to the fact that cosmic rays are a subcase of it.

    - Case 4R: Due to the fact that the apparent magnitude of a flash in the I band is always brighter than in R and that the flashes in I filter camera can be detected more easily, due to Rayleigh scattering events that have been detected only in the frame(s) of the R camera and not in the respective one(s) of the I camera can be discarded. In this case, the event is characterized as false.

    - Case 4I: Taking into account the second criterion regarding the non-movement of the event between successive frames, the Case 4I has to be split into two subcases, which are addressed below.
      Case 4I-1: If the event has been detected in the same pixel area of multiple successive frames and is of round shape it is identified as a cosmic ray hit, because if it was a validated flash, it would be certainly detected in the frames of the R camera too. Thus the event is considered a suspected lunar impact flash with a high confidence to be considered as validated.
      Case 4I-2: If the event is detected in only one frame of the I camera. This could be a cosmic ray that hit almost perpendicularly the sensor, or an impact flash. To make the distinction, the shape of the event is compared with that of a standard star that has been observed between the lunar data chunks. If the full width at half maximum between the star and the event varies within a certain range, it is classified as a suspected lunar impact flash, with a low confidence to be considered as validated.

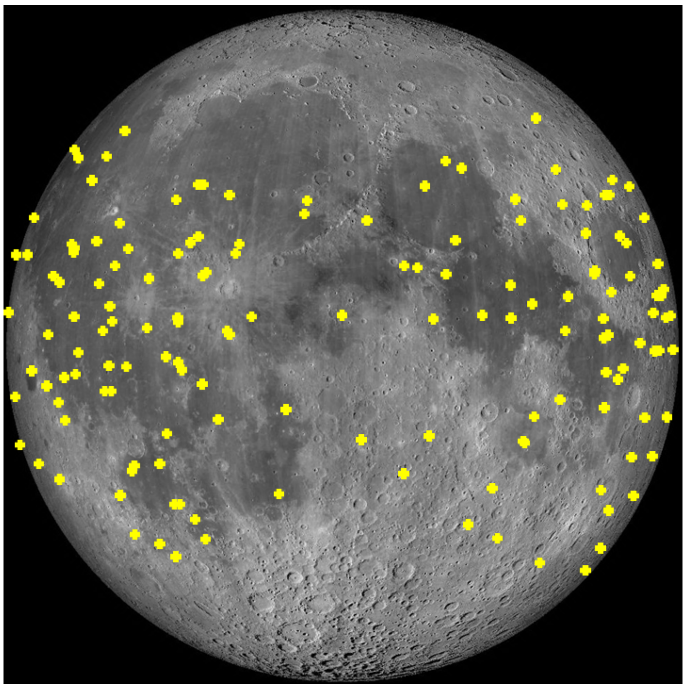


Image 3.2: Location of the impact flashes detected so far by NELIOTA.

The detection domain of the NELIOTA software provides images before and after the event, always seven frames before and seven after the event [1]. Using these images, a mean background image is created that is subtracted from the frames that contain the flash, in order to obtain an image without background that its main feature is the event. This background image is called a 'Difference' image and after the subtraction it has a non zero-level background,due to the fact that it contains a residual noise signal. The standard deviation of this noise depends on the glare of the sunlit part of the Moon and the seeing conditions.

3.2 The MIDAS project

The Moon Impacts Detection and Analysis System (M.I.D.A.S.) project is a software that has been running since 2009 and its aim is to automatically detect flashes produced by the impact of meteoroids on the lunar surface [21]. Those flashes have a very brief duration, so they can be seen in no more than 2 or 3 frames, making their non-automatic detection a hard task. Furthermore, this package achieves a photometric analysis of the flashes, can calculate the value of their luminous efficiency and can confirm the flashes detected by different telescopes within the same network by uploading them into a common database. Although it is impossible to deterministically identify the origin of the meteoroids solely by observing the moon, this software is also able to link the source of the impactors with a known meteoroid stream with a high probability [5].

For using and testing purposes of this software monitoring of the non-sunlit lunar surface with small telescopes and high-sensitivity CCD video cameras has been performed [5]. The frame sequence from the telescope must be digitized and stored in a hard disk in order for the images to be analyzed afterwards.

Image 3.3: An example of the lunar surface that the MIDAS software monitors.

The input of the MIDAS software can either be a live video streaming coming from the telescope or, alternatively, a preprocessed AVI video file [21]. The user is allowed to see, change, and even analyze potential impact flashes that haven't been confirmed yet, while the detection process is running.

The detection of an impact flash happens as follows: the software compares successive frames, whose number is set by the user, in order to identify areas of pixels whose value has changed more than a certain threshold that is set by the user. This threshold may depend on the user's criteria and the telescope and camera features and affects the process in a significant manner [5]. The detection process only occurs in a certain region of interest of the field of view whose limits are defined by the lunar limb. For this reason, a lunar mask must be defined, indicating that the detection has to happen inside of the lunar surface, excluding any stars or the sky. This mask can be optionally specified by the user, by marking the regions outside of the lunar disk using the mouse. If the user wishes to do that, by clicking on three or more points located at the lunar limb, an appropriate mask is created and applied on the field of view.

Any flash detection is recorded in the format of a small AVI video file and stored on a database also containing some useful information regarding the time and position that it occurred. Every telescope in the network has its own database with the suspected impact flashes, which are automatically cross-checked with the detections stored in other databases. In the case that at least one more telescope in the system has recorded the same flash, the event is confirmed. Consequently, the MIDAS software produces a list with the confirmed events and any false positives that can be later analyzed or deleted by the user [21].

For the best analysis of the images, a variety of preprocessing methods can be applied by the software. For instance, the frames can be averaged using a median filter that recalculates the value of every pixel based on the brightness of the pixels around it, at a distance that the user can specify. This could unfortunately have a blurring effect, along with the desired smoothing. An alternative method for that is the video size reduction to 2:1 [22], where the width and height of all the images that are received are divided by two and the video is resampled to the new size [23]. In this way, the frame's noise is reduced along with the time needed for the real time analysis of the images. This method is efficient due to the fact that any true impact flash covers more than one single pixel in a frame, so it cannot disappear after resampling [21]. Furthermore, to avoid noisy images, a transformation filter could be applied instead to the video file, in a way that each frame is substituted by a weighted composition of a number of frames around it, with a weight and the number of frames to be taken into consideration each time selected by the user. Subsequently, larger weight can be given to the current frame or the ones before or after, making their contribution bigger or smaller. By default, the maximum allowed intensity difference between consecutive frames for them to be taken into consideration is set to the average noise level in the images [24]. Thus, any differences occurring due to the background noise are evened out, but any difference caused by an impact flash can still be seen [5].

In order to localize the events detected, the coordinates of the corresponding pixels on the images are converted automatically to the coordinates of the flash on the lunar surface, by correlating the lunar features that are tracked by the high resolution cameras to at least three easily identifiable ones with known positions [24].

In the interest of estimating the diameter of craters created by the lunar impacts and calculating the impact flux on the Moon and the Earth, a study of the kinetic energy of the meteoroids is beneficial. For this purpose, a photometric analysis is performed, in order to infer the evolution with time of the luminosity of the impact flashes and from that, obtain the energy radiated as visible light by the flash [21].

The light curve of the event and a series of standard stars is plotted, where the brightness is in pixel values that in 8-bit analog to digital conversion range from 0 to 255. Thus, a magnitude versus time plot can be obtained for the impact flash by comparing its magnitude to that of the stars observed and from this, the software calculates the energy released as visible light on the Moon by performing a numerical integration of the radiated power P with respect to time [5].

The MIDAS software has certain complementary functionalities such as a lunar calendar display with the moon phases for each day of the month, the corresponding sunlit part of the lunar disk and the local moonrise and moonset time that can help the user plan the lunar observations, that can be performed when the lunar phase ranges from 0.1 to 0.6 [21].

3.3 The Flash Project: Lunar impact flashes from France

Conductive to the observation of the thermal emission of fresh lunar impact craters observation, an automatic lunar impact flash detection and localization software has been created in the Observatoire de la Côte d’Azur in France, called the flash project [25]. To identify the flashes and their coordinates on the lunar surface live high speed image data of the lunar surface are obtained. These coordinates can then be later used to attempt the observation of newly created impact craters using NASA's Lunar Reconnaissance Orbiter (LRO) and study their thermal data. As this detection and the localization is a live process, the craters still retain a part of the heat of the impact when they are identified, making their observation with the LRO’s thermal camera a useful process [26]. NASA’s LRO has been into orbit around the Moon since 2009 and its mission concerns the mapping of the features of the lunar surface and their respective temperature at all times.

The detection algorithm that this software uses works as follows: firstly, a master frame is computed using the last ten frames that are received and it is subtracted from the current one in order to eliminate the inhomogeneous background of the moon and the glare from the sunlit part. Then, to detect the flashes, either a threshold is applied to the pixel values, and any pixel value that exceeds that is recognised as a flash, or a Gaussian filter is applied to eliminate the artifacts [26]. In the first case, there is a need for various filters in order to eliminate the noise and confirm the flash, while in the second no extra filters of any kind are necessary.

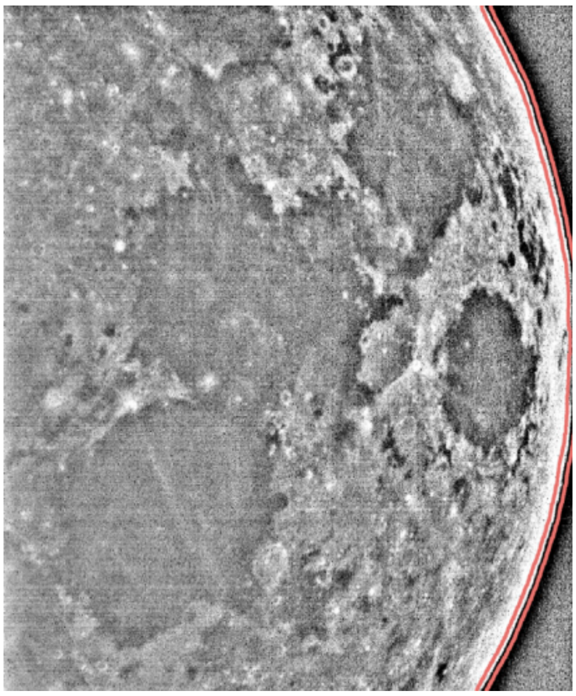


Image 3.4: Light gradient removed image with a pixel threshold applied for contrast. The inner and outer edge of the lunar limb is marked with red. The average of these two detections is used for the final location of the lunar limb, resulting in ~1 pixel accuracy. Image taken from the NELIOTA website.

To analyze the flash, firstly the images are stacked, then the background is removed and the limb is detected. According to the limb, a circle fit is performed and the LRO map is generated. Then, using a binary map, the image from the fit is correlated to the lunar projection that corresponds to the LRO map.

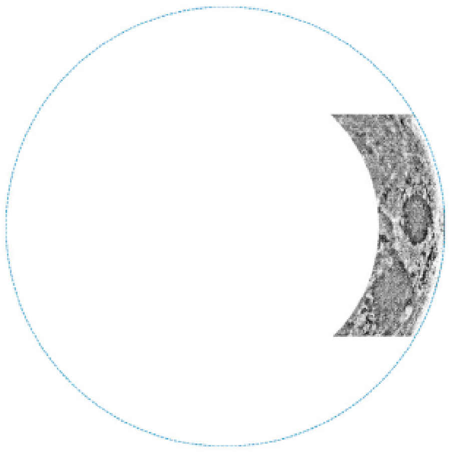


Image 3.5: An example of a circle fit, after the lunar limb identification. Image taken from the NELIOTA data and analyzed by the flash project.

To calculate the impact coordinates after a successful detection two methods are used: coordinate transformations or geolocating. In the first technique, the coordinates of the pixels inside the image are transformed into corresponding lunar latitude and longitude, using mathematical transforms. Using the geolocation method, well-known features of the moon with given locations are detected within the recorded images in order to estimate the impact coordinates [26]. Near the edge of the moon, neither of the above methods provide accurate results. In any case, the coordinates found by this software for the impacts observed during the testing phase were cross-checked using the impact coordinates of the NELIOTA database.

After the detection and the localization of the flashes, an analysis of the events regarding the source of the impactors takes place. To attempt to identify their origin, it is checked if there is an active meteoroid stream at the time of the impact and the solar longitude difference between the impact time and the max of each candidate stream is computed. Then, the sub-radiant point of each candidate stream on the lunar surface is calculated and a small list of candidate streams is extracted [28]. The impact speed for each stream is estimated and the material of the impactor is observed so that the corresponding probabilities for the impactor to belong to the stream are calculated.

Currently, for the observations at the Observatoire de la Côte d'Azur a MEADE ACF telescope of 40 cm is used along with a CMOS ASI ZWO 183 camera that records with a 20 fps rate. Furthermore, a 50 cm custom Newtonian telescope is under operation with two QHY 174 GPS CMOS cameras and a guider on the moon.

Image 3.6: The 50 cm custom Newtonian telescope used at the Observatoire de la Côte d’Azur.

In conclusion, the main purpose of this software for automated detection and localization is the monitoring of impact cratering and the analysis of their thermal signatures. Thus, the cooling behavior for impact craters can be modeled as a function of the energy in the impact, frictional and shock heating, crater size and in general the impact physics can be analyzed [27].

Chapter 4: Online Detection

For the purposes of the live detection of events, a plugin for FireCapture has been developed. This plugin extends the existing capabilities of FireCapture, enhancing them with the simultaneous detection of flashes. This plugin has a graphical user interface as well as a properties file for configurations, automatizes the detection process and significantly reduces the amount of storage that is required for long hours of observations.

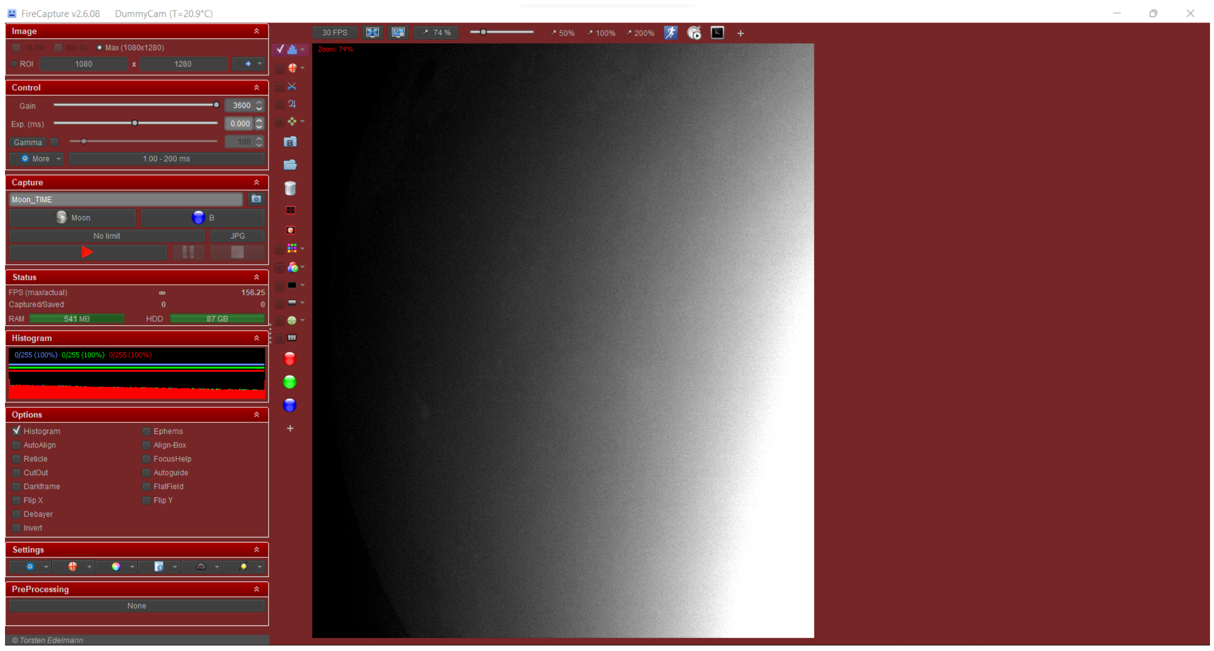
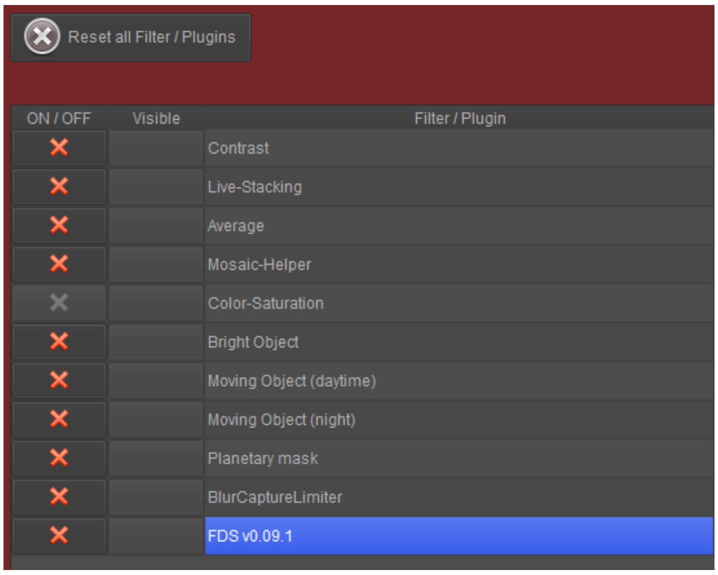


Image 4.1: The FireCapture GUI.

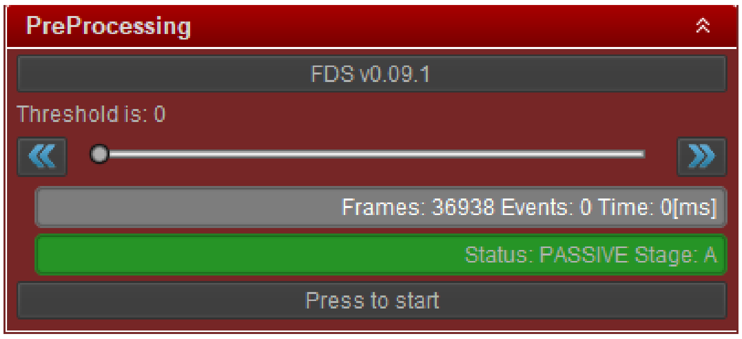
4.1 The Graphical User Interface (GUI)

In the graphical user interface of FireCapture, the plugin can be found in the "Preprocessing" area, as all other plugins developed for this program. When the FDS plugin is selected, the user will be able to see the GUI of the plugin that informs them about the number of the frames that have already been received by the camera, the number of the events captured in the current session and the time in milliseconds. Also, from there, the user can specify a threshold for the detection and start the operation of the plugin.


Reset all Filter / Plugins

ON / OFF	Visible	Filter / Plugin
✗		Contrast
✗		Live-Stacking
✗		Average
✗		Mosaic-Helper
✗		Color-Saturation
✗		Bright Object
✗		Moving Object (daytime)
✗		Moving Object (night)
✗		Planetary mask
✗		BlurCaptureLimiter
✗		FDS v0.09.1

Image 4.2: Plugin Selection in FireCapture.


PreProcessing

FDS v0.09.1

Threshold is: 0

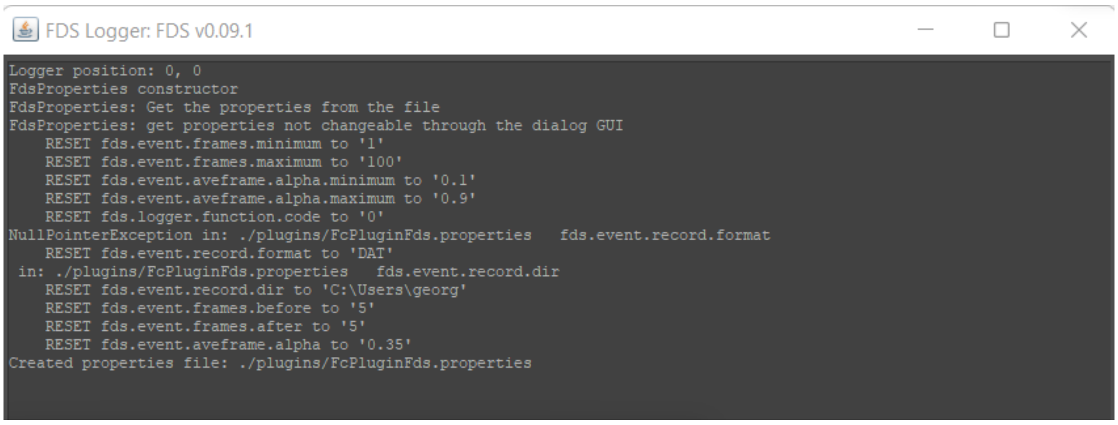
Frames: 36938 Events: 0 Time: 0[ms]

Status: PASSIVE Stage: A

Press to start

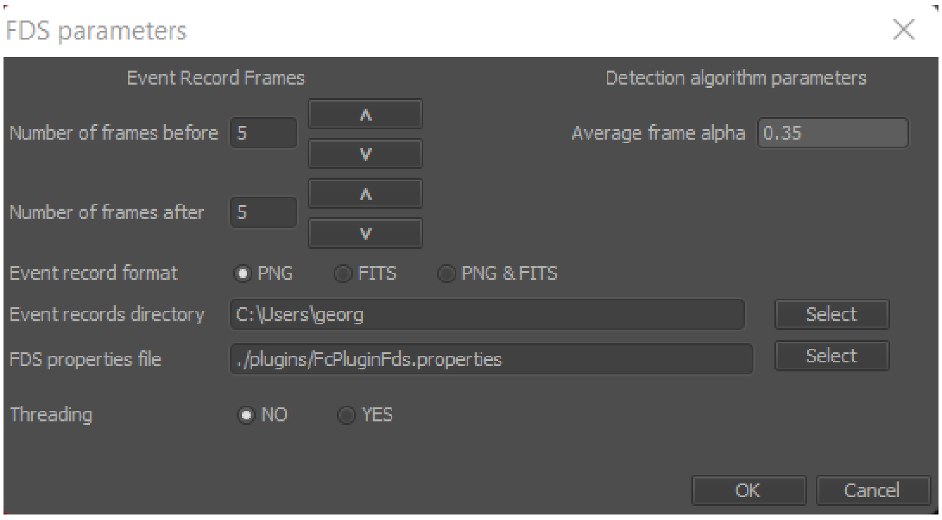
Image 4.3: The FDS plugin GUI.

Once the plugin has been initiated, a logger window appears with the information that has been retrieved from the properties file, along with a configurations window that allows the user to select how many frames before and after the event they would like to capture and their frames format choosing from png, fits, or both. Moreover, the directory that the results and the properties file will be written and whether threading is enabled or not. The necessary parameters are externally configurable via the plugin's properties file. When the START button is pressed the file is read and the parameters are imported. In the case of an exception, due to mistakes in the properties file the button will not start the FDS. The parameters that are not editable through the GUI, but only from the properties file are: the alpha parameter, which is the new frame weight for the average image updating, the largest allowed number of frames to record (before or after) and the smallest allowed number of frames to record (before or after). When closing the dialog GUI, the parameters, if changed are saved in the properties file and the frames processing starts.



```
Logger position: 0, 0
FdsProperties constructor
FdsProperties: Get the properties from the file
FdsProperties: get properties not changeable through the dialog GUI
    RESET fds.event.frames.minimum to '1'
    RESET fds.event.frames.maximum to '100'
    RESET fds.event.aveframe.alpha.minimum to '0.1'
    RESET fds.event.aveframe.alpha.maximum to '0.9'
    RESET fds.logger.function.code to '0'
NullPointerException in: ./plugins/FcPluginFds.properties   fds.event.record.format
    RESET fds.event.record.format to 'DAT'
 in: ./plugins/FcPluginFds.properties   fds.event.record.dir
    RESET fds.event.record.dir to 'C:\Users\georg'
    RESET fds.event.frames.before to '5'
    RESET fds.event.frames.after to '5'
    RESET fds.event.aveframe.alpha to '0.35'
Created properties file: ./plugins/FcPluginFds.properties
```

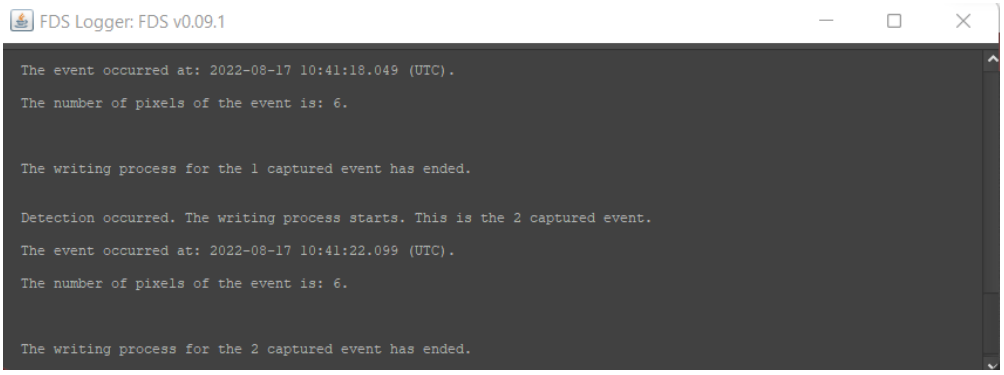
Image 4.4: The logger window after initiating the plugin.



Number of frames before: 5
Number of frames after: 5
Average frame alpha: 0.35
Event record format: [✓ PNG] FITS PNG & FITS
Event records directory: C:\Users\georg
FDS properties file: ./plugins/FcPluginFds.properties
Threading: [✓ NO] YES

Image 4.5: The configuration window.

When a detection has been realized, the user will be notified by the logger about the timestamp of the event, the number of pixels that triggered the capturing and the time that the recording has ended. The user can then check the records directory to inspect the results.

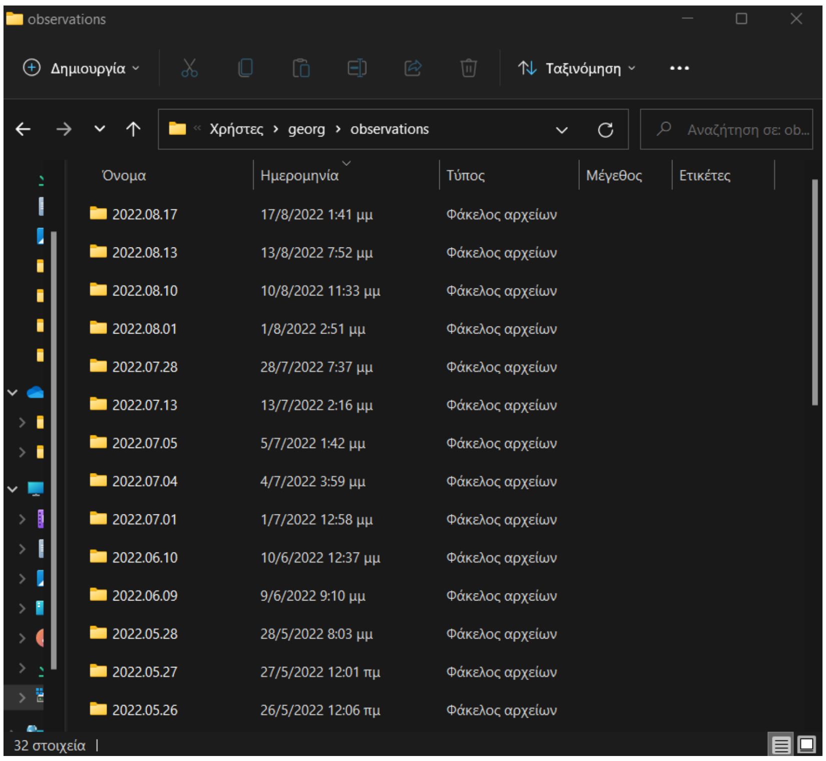


```
The event occurred at: 2022-08-17 10:41:18.049 (UTC).

The number of pixels of the event is: 6.


The writing process for the 1 captured event has ended.

Detection occurred. The writing process starts. This is the 2 captured event.

The event occurred at: 2022-08-17 10:41:22.099 (UTC).

The number of pixels of the event is: 6.


The writing process for the 2 captured event has ended.
```

Image 4.6: The logger window when the 2nd event capturing has been completed.

4.2 Output

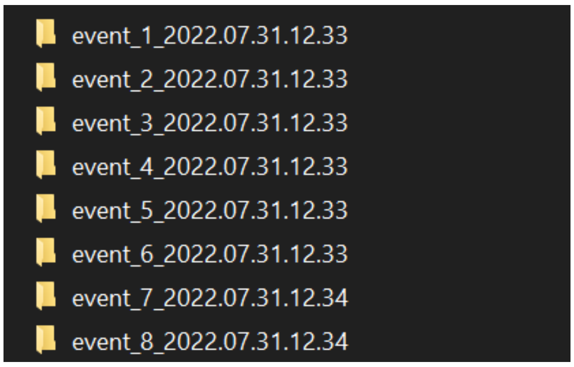
After the first detection has been realized a new folder will appear at the chosen location under the name "observations". For each day, another folder will be created inside the first one, which will contain all the different events that occur in each session and time.



Όνομα	Ημερομηνία	Τύπος	Μέγεθος	Ετικέτες
2022.08.17	17/8/2022 1:41 μμ	Φάκελος αρχείων		
2022.08.13	13/8/2022 7:52 μμ	Φάκελος αρχείων		
2022.08.10	10/8/2022 11:33 μμ	Φάκελος αρχείων		
2022.08.01	1/8/2022 2:51 μμ	Φάκελος αρχείων		
2022.07.28	28/7/2022 7:37 μμ	Φάκελος αρχείων		
2022.07.13	13/7/2022 2:16 μμ	Φάκελος αρχείων		
2022.07.05	5/7/2022 1:42 μμ	Φάκελος αρχείων		
2022.07.04	4/7/2022 3:59 μμ	Φάκελος αρχείων		
2022.07.01	1/7/2022 12:58 μμ	Φάκελος αρχείων		
2022.06.10	10/6/2022 12:37 μμ	Φάκελος αρχείων		
2022.06.09	9/6/2022 9:10 μμ	Φάκελος αρχείων		
2022.05.28	28/5/2022 8:03 μμ	Φάκελος αρχείων		
2022.05.27	27/5/2022 12:01 πμ	Φάκελος αρχείων		
2022.05.26	26/5/2022 12:06 πμ	Φάκελος αρχείων		

32 στοιχεία

Image 4.7: The observations folder.



event_1_2022.07.31.12.33
event_2_2022.07.31.12.33
event_3_2022.07.31.12.33
event_4_2022.07.31.12.33
event_5_2022.07.31.12.33
event_6_2022.07.31.12.33
event_7_2022.07.31.12.34
event_8_2022.07.31.12.34

Image 4.8: An example of the contents of the folder of a specific day.

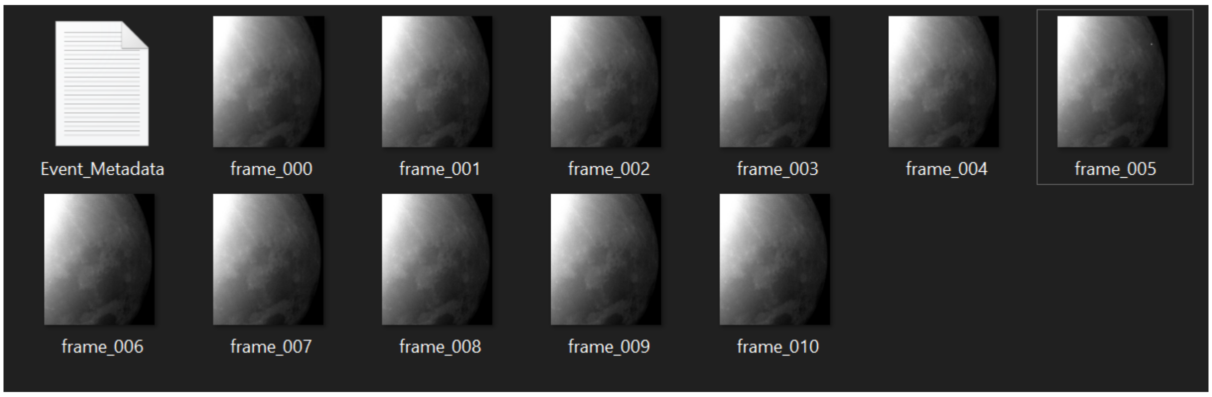
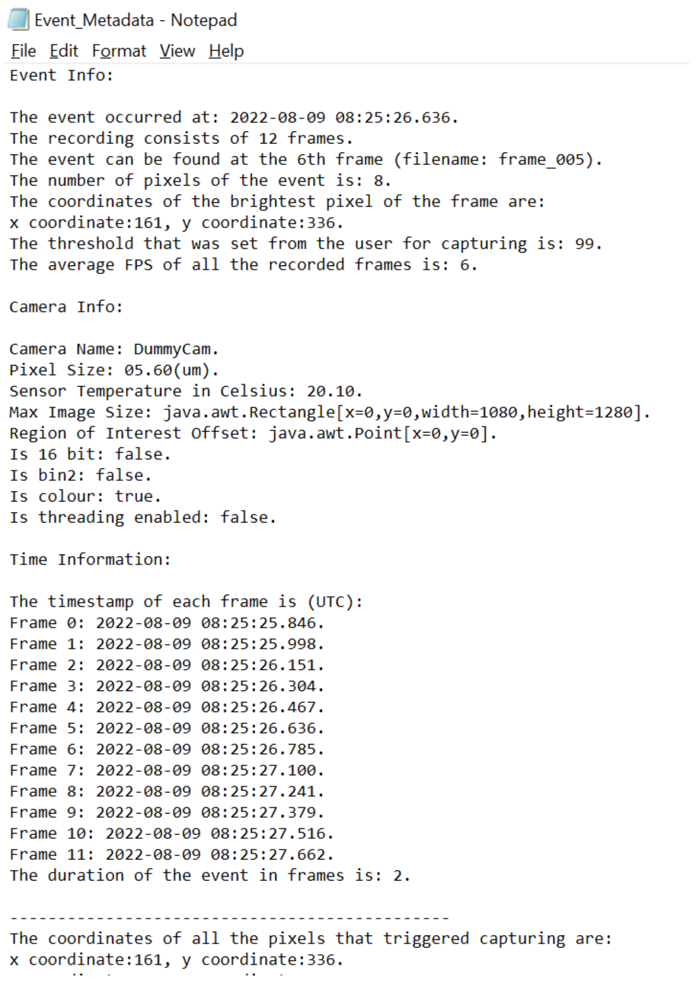


Image 4.9: Example of the contents of an event folder (The number of frames recorded varies according to the configurations made by the user).

Along with each event, a different metadata file is created that contains crucial event information, camera details and time information that can be later used for the classification and the localization of the event.



```
Event Info:

The event occurred at: 2022-08-09 08:25:26.636.
The recording consists of 12 frames.
The event can be found at the 6th frame (filename: frame_005).
The number of pixels of the event is: 8.
The coordinates of the brightest pixel of the frame are:
x coordinate:161, y coordinate:336.
The threshold that was set from the user for capturing is: 99.
The average FPS of all the recorded frames is: 6.

Camera Info:

Camera Name: DummyCam.
Pixel Size: 05.60(um).
Sensor Temperature in Celsius: 20.10.
Max Image Size: java.awt.Rectangle[x=0,y=0,width=1080,height=1280].
Region of Interest Offset: java.awt.Point[x=0,y=0].
Is 16 bit: false.
Is bin2: false.
Is colour: true.
Is threading enabled: false.

Time Information:

The timestamp of each frame is (UTC):
Frame 0: 2022-08-09 08:25:25.846.
Frame 1: 2022-08-09 08:25:25.998.
Frame 2: 2022-08-09 08:25:26.151.
Frame 3: 2022-08-09 08:25:26.304.
Frame 4: 2022-08-09 08:25:26.467.
Frame 5: 2022-08-09 08:25:26.636.
Frame 6: 2022-08-09 08:25:26.785.
Frame 7: 2022-08-09 08:25:27.100.
Frame 8: 2022-08-09 08:25:27.241.
Frame 9: 2022-08-09 08:25:27.379.
Frame 10: 2022-08-09 08:25:27.516.
Frame 11: 2022-08-09 08:25:27.662.
The duration of the event in frames is: 2.

----------------------------------------------
The coordinates of all the pixels that triggered capturing are:
x coordinate:161, y coordinate:336.
```

Image 4.10: The metadata file of an event captured using the “dummy” camera of FireCapture.

4.3 Algorithm

The detection algorithm is implemented as follows:
For each session, the plugin begins by buffering the frames, in order to create a weighted average image of those. For each frame that arrives, the last 5 frames before that are used to create the average image (weight a = 0.35, hard-coded), using a copy of those frames in the form and sequence exactly as the camera frames are parsed to the method by the FireCapture. The detection process is activated by the START button of the plugin's GUI. When the detection is activated, for each frame that arrives, the average image is subtracted from it, thus the difference image is obtained. In this image, if any pixel exceeds the value of the threshold set by the user through the plugin's GUI, the detection is triggered.

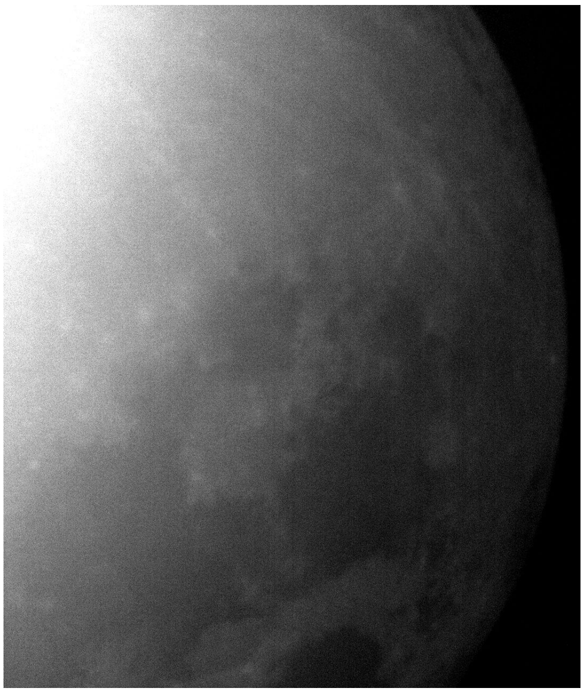


Image 4.11: A frame obtained by the Krioneri Telescope during the lunar observations.

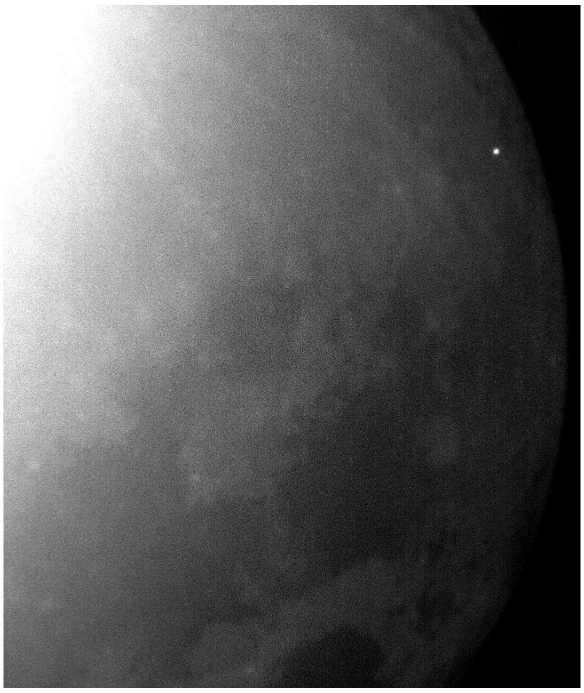


Image 4.12: A frame containing a lunar impact flash obtained by the Krioneri Telescope.

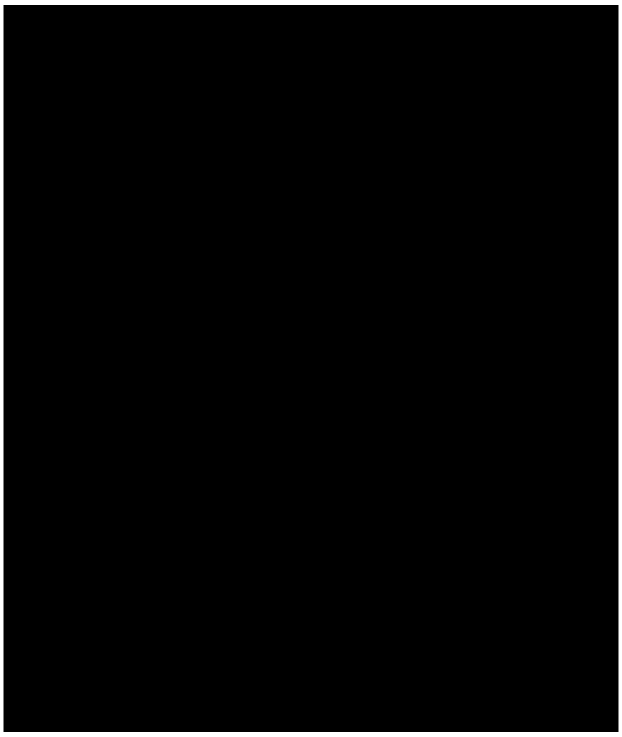


Image 4.13: The difference image that has resulted from the subtraction of the above frame and the average frame that corresponds to it.

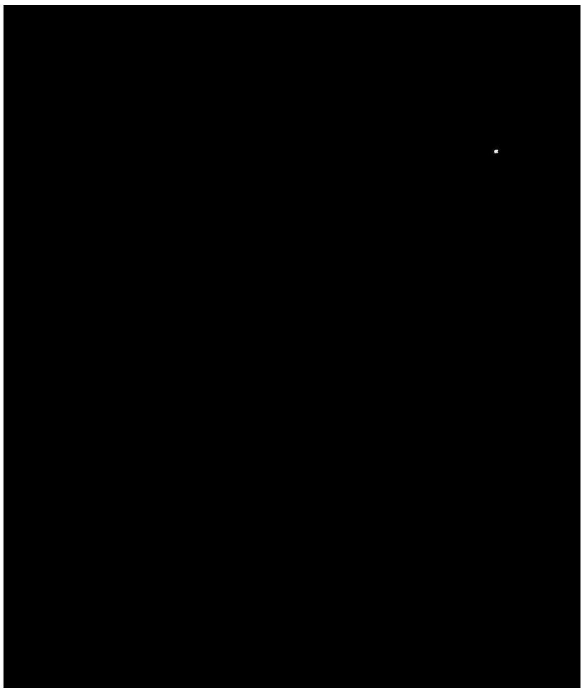


Image 4.14: The difference image that has resulted from the subtraction of the above frame that contains the flash and the average frame that corresponds to it. In this frame the flash remains bright.

Setting a proper threshold value is important for the detection. The program will capture the event only if at least one pixel of the difference frame has a value greater than that of the threshold. At any point during the observation, this value can be adjusted according to the events that are captured and the atmosphere conditions. On the one hand, setting a big threshold may lead to some fainter events not being captured and on the other hand, choosing a small threshold value could cause false positives. If more than 200 pixels (area larger than 40x40 pixels) trigger detection at the same time, the threshold is not appropriate and the user is prompted by the logger to increase it, in the case that there is no cloud passing in front of the telescope.

The recording is saved in the directory specified by the user at the initialization of the plugin and it consists of N frames before the event, the K frames that the event lasts and M frames after (N and M is also specified by the user). The main requirement for the live detection algorithm and for any other live process is rapidity, in order that the users cannot see the program slowing down or stopping after the detection and thus, this algorithm is adequate for this use.

The FDS processing consists of a few individual stages that determine the state of the plugin. The stages are six, of which the first and the last are merely controls and only the second to the fifth perform some processing of the current frame.

The stages are the following:

- Stage A: This is the first stage of the plugin, that the user sees before the initialization and before the activation of the detection. The detection here is passive and this stage can be also entered after deactivating the FDS with the STOP button. This means that in this stage, no detections can occur.

- Stage B: During this stage, the buffer arrays that are used for the detection process are initialized. This happens immediately after the FDS activation with the START button which lets the detection prepare to begin. This stage prepares initial values for some FDS parameters (e.g. image size). The frames buffer array is also properly initialized during this stage, by copying the first N frames and the initial "Average Image Frame" is created. When the N-th frame is processed the detection is activated.

- Stage C: This stage is activated for the next frame immediately after the frame for which an event has been detected. It is also applied to the following K + M -1 frames, namely the frames of the event and the ones that have to be recorded after it, excluding the last one. The frames buffer array is completed with the K + M -1 frames following the candidate event frame. No other processing is applied to these frames. The inclusion of the last, the M-th, frame ends with forming and saving a "Candidate Event Record" with the frames and additional information and data to a storage in proper format. Then, the detection for this event is considered finished. Note: the averaged image is not updated by the M frames.

- Stage D: This stage terminates the current FDS activity if the detection is to come to an end because the STOP button has been pressed. Note however, that in the case that an event has just been detected, the process is not finished until the detection is over and the stage "C" is finished. In this stage, the FDS resets some of its counters and clears the memory used by the frames buffer array and by the average image frame and the detection becomes passive.

- Stage E: This stage applies the candidate event detection algorithm to the current frame. If no event is detected, the frames buffer array is updated with the current frame where the oldest frame (the first) is replaced by the second in line, the second by the third, etc. up to the N-th frame which is saved as N-1 and then the current one is copied (First In First Out Queue). The average image frame is correspondingly updated and thus, the current frame processing is completed. However, in the case that a candidate event is detected, no updates occur, the current frame is copied as new, the N+1, in the frames buffer array, a new detection is initiated and the counter of detected events is increased.

- Stage F: This is an exceptional stage which the process should never reach. For this reason, no action is implemented there.

Overall, if no event is detected in the current frame, the frames buffer array is updated by removing the oldest saved frame, moving the other saved frames one position down, and including the current frame as the N-th saved in the buffer. The average frame is updated using a weighted average process, with weight a = 0.35, that is hard-coded and can be changed only through the properties file. Otherwise, if an event is detected, the updating processing is temporarily discontinued, the frames of the event (K frames) are included as the next after the N already saved frames (frames before the event) in the buffer and the next M frames (frames after the event) are simply added to the buffer. Then, the completed frames buffer (that contains N+K+M number of frames) is saved to event storage and the processing continues with the next frame.

If the threading is activated, the event record that is saved is implemented on a separate thread, so the plugin continues to process the arriving frames independently of the saving process. This could be a problem in the following cases:

- When the image of the moon that we receive is not stable due to bad weather, telescope movements, or if any sunlit parts are included in the field of view. This may lead to fake detections.
- When a satellite is crossing the image, then a lot of directories will be created, until it is no longer visible inside of the image.

These problems can be solved by either not choosing threading or by temporarily stopping the detection or significantly increasing the threshold when any of the above occurs. It should be noted that those problems do not occur with true lunar impact flashes, nor when the observation conditions are optimal, but also for their capturing, threading is not necessary, due to the fact that the possibility of two impact flashes occurring almost at the same time approaches zero. Although, if threading is enabled and the observation conditions are satisfactory, the directories created by a satellite will be classified as such by the offline standalone tool.

Image 4.15: Satellite captured by the Krioneri telescope during the lunar observations.

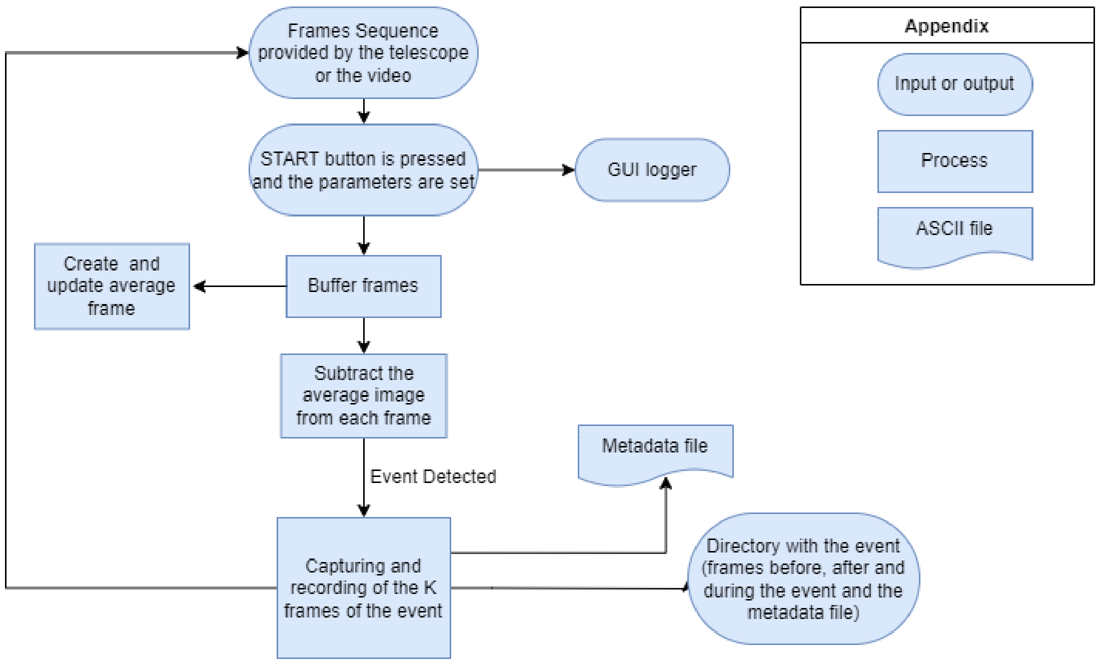
In order to use the software with a real camera, the camera needs to be connected with the computer, after correct installation of its drivers and the type of the camera should be specified when prompted by the menu when the FireCapture is opened. However, the software does not necessarily need a real camera for its operation, as it is possible to use it with an existing video of the moon and choosing the “DummyCam " from the FireCapture menu. In both cases, after the camera setup, the process followed is exactly the same.

4.4 Software overview

The software takes input from:

- The user via the GUI
- The telescope or the video provided
- Online detection properties file

All output (recorded frames and metadata file) is stored inside the directory provided, in the observations folder.



Frames Sequence provided by the telescope or the video
START button is pressed and the parameters are set
GUI logger
Create and update average frame
Buffer frames
Subtract the average image from each frame
Event Detected
Metadata file
Capturing and recording of the K frames of the event
Directory with the event (frames before, after and during the event and the metadata file)
Appendix
Input or output
Process
ASCII file

Image 4.16: Diagram of the flow of the process that the operation of the plugin follows.

4.5 System Requirements

The functional requirements are:

1. Detection of lunar events.
2. Detection on images of low and high luminosity.
3. Detection of events recorded by different telescopes and cameras.
4. Parameters selection by the user through the GUI and an ASCII file.
5. Detection controlled by the START/STOP button of the GUI.
6. Recording in PNG and in FITS format.
7. Creation of metadata file containing the event details.
8. Operation for the 3 major operating systems (Windows, Linux, Mac OS).

The non-functional requirements are:

1. GUI logger with information regarding the capturing.

4.6 Online detection use case

A standard use case for the software is presented that refers to performing online detection without threading using a real camera.

- Actor: User

- Goal: Initiate a session for the detection of lunar impact flashes using a real Camera (e.g. QHY), select correct values for the FDS parameters, the threshold and the recording directory and no other user interference, other than pressing the STOP button for the session to end.

- Scope: This case is part of the online detection process, using the FDS plugin for FireCapture. The necessary parameters and the frame sequence from the telescope are provided, the detection process is activated and any results are written in the assigned directory.

- Preconditions:

1. The camera provided is compatible with the FireCapture version that corresponds to the user's operating system and PC.
2. The parameters chosen (especially the threshold) are set to correct values (within the limits provided by the algorithm).
3. The camera is stable and the weather conditions are appropriate for observations.

- Normal flow:

1. Start FireCapture, select the type of the camera that is connected and select the FDS plugin from the plugin list.
2. Set the threshold to an appropriate value given the observation conditions and press the START button on the plugin interface.
3. Select the number of frames before and after the event that will be recorded, the format of the frames, the recording folder and no to the threading option (meaning that one event will be recorded at a time). Allow the detection to start, pressing the OK button.
4. When the first event occurs, wait for the plugin to finish the saving of the frames and the creation of the metadata file.
5. After the writing of the results, stop the detection process, using the STOP button and finish the session in FireCapture, using the close button.

- Quality attributes

1. The results of the detected event will be written in a sub-directory of the observations folder.
2. The properties file will be updated with the last values for the number of frames before and after the event that will be recorded.

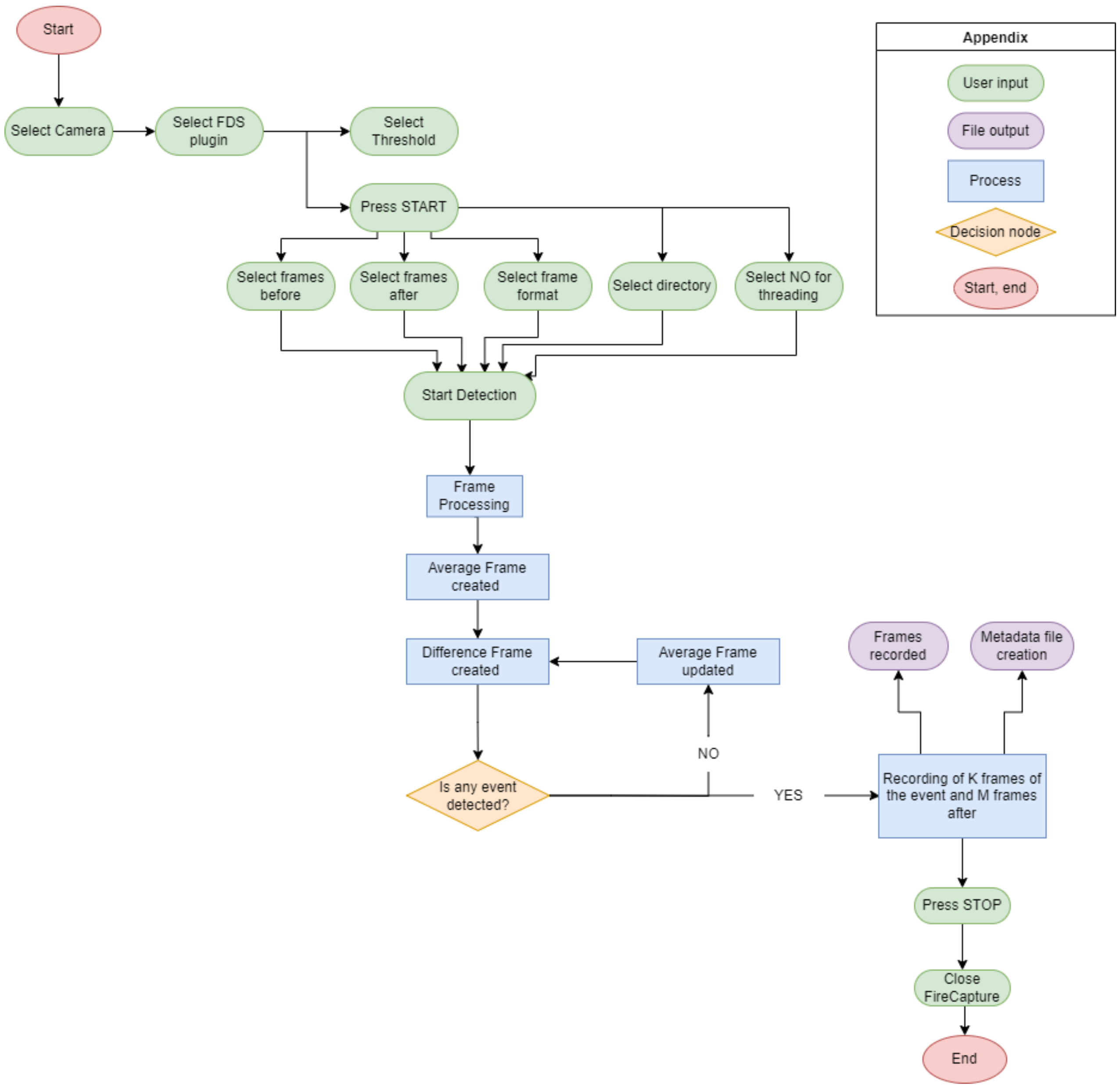


Image 4.17: Graphic representation of the workflow of the above use case.

4.7 Capturing dark spots on the solar surface

During the testing phase of the software, there have been attempts to connect the plugin to the solar telescope (using a ZWO ASI120MC-S camera) during the day and observe the sun using FireCapture. During these attempts it has been found that the developed software could successfully detect sunspots due to their resemblance to lunar impacts. More specifically, when the field of view was moved from a sun area that contained no sunspots to an area that did, the detection was triggered. This happens because what our software detects in this domain are sudden changes in the luminosity of small areas.

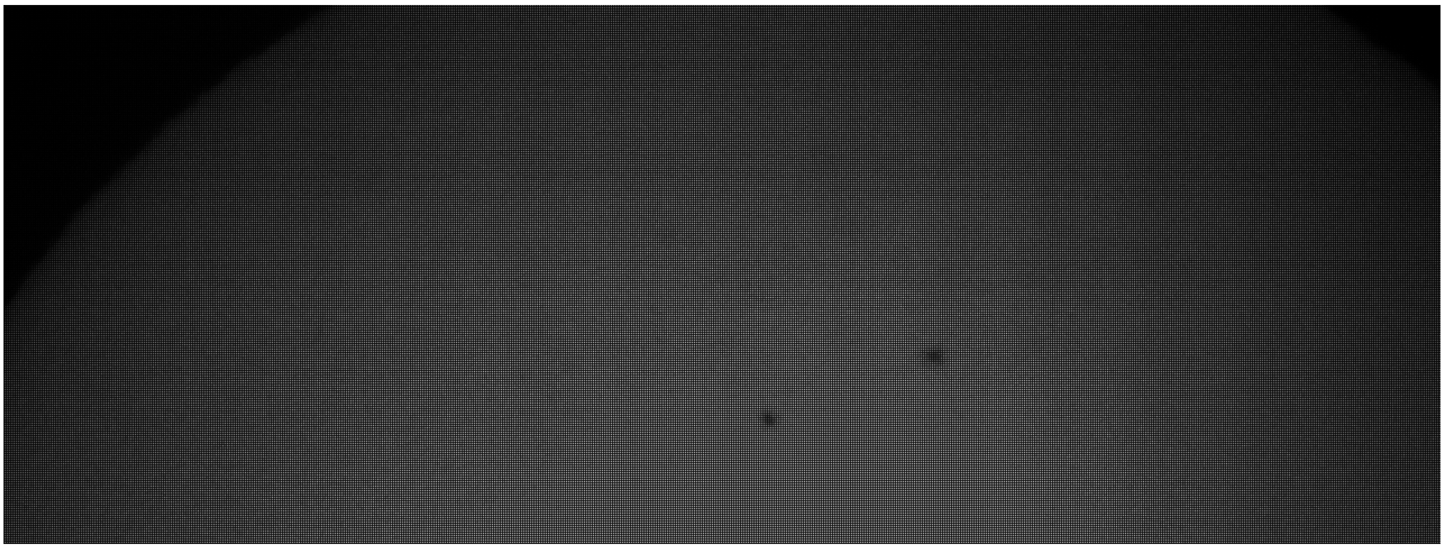


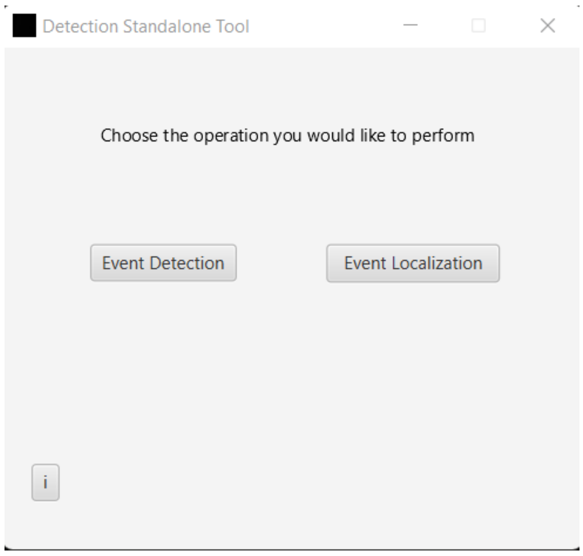
Image 4.18: Sunspots detected using the plugin from Penteli during the observations realized by the FDS team.

Chapter 5: Offline Detection

After the online detection plugin has created the event folders in the observations directory, the offline standalone tool can analyze and classify them. During this phase of the detection, the already created events can be distinguished and organized, regarding their type and false events can be recognised as such and can be discarded. With this part of the software, the workflow is continued, as the offline detection's input is the output of the online detections plugin, this is to say, the directories of the detected events, containing the frames and the metadata file.

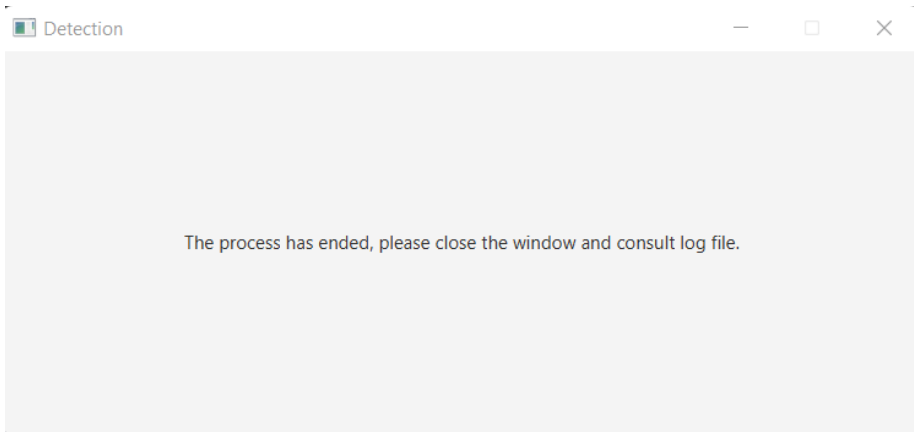
5.1 The graphical user interface (GUI)

The graphical user interface allows the user to interact with the software in a straightforward way, informs them about the folder that is being analyzed in each moment and the progress of the detection and notifies them, when the procedure is completed.



Detection Standalone Tool
Choose the operation you would like to perform
Event Detection
Event Localization
i

Image 5.1: The standalone's tool starting screen.



Detection

The process has ended, please close the window and consult log file.

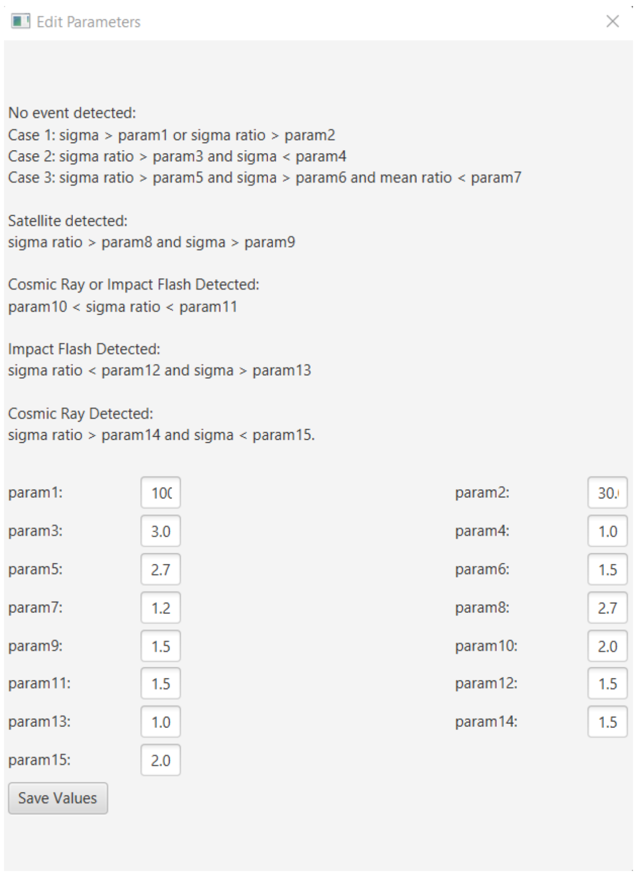
Image 5.2: The pop-up window that notifies the user when the process is completed.

5.2 Input

In order for the software to work, proper input has to be given. In addition to this input, the user can optionally adjust certain parameters regarding the calibration and the classification phases of the process.

The obligatory input consists of a directory, produced by the online detection FireCapture plugin that contains one or more properly recorded events. More specifically, the input can be a single event directory, that contains a sequence of frames and a metadata file or a directory containing multiple events. Each event consists of a specific number of frames of PNG and/or FITS format and a metadata file, with all the event, camera and time information. If the user does not have any events recorded using the online detection FireCapture plugin, but wishes to analyze a specific event (or many of them) for which they possess a sequence of frames that the event is shown and have the relevant information that is contained in the metadata file, such as the coordinates of the pixels of the event in the image, they can construct a file of the same format manually and analyze the event in the same way.

Concerning the optional parameters that the user can adjust, all of them already have a suggested value, that if it does not fit with the user's preferences, can be changed through the GUI. Firstly, by clicking on the [Edit Parameters] button on the GUI starting screen, the user may see the values that are used for the classification of the events. These parameters are compared with some values that are produced by the algorithm and from this comparison, the type of the event is concluded.



No event detected:
Case 1: sigma > param1 or sigma ratio > param2
Case 2: sigma ratio > param3 and sigma < param4
Case 3: sigma ratio > param5 and sigma > param6 and mean ratio < param7

Satellite detected:
sigma ratio > param8 and sigma > param9

Cosmic Ray or Impact Flash Detected:
param10 < sigma ratio < param11

Impact Flash Detected:
sigma ratio < param12 and sigma > param13

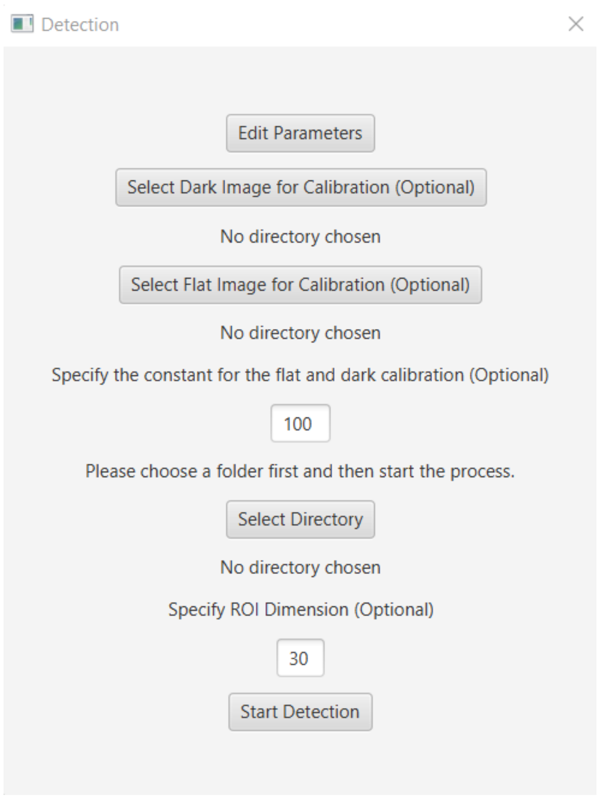
Cosmic Ray Detected:
sigma ratio > param14 and sigma < param15.

param1:	10(	param2:	30.
param3:	3.0	param4:	1.0
param5:	2.7	param6:	1.5
param7:	1.2	param8:	2.7
param9:	1.5	param10:	2.0
param11:	1.5	param12:	1.5
param13:	1.0	param14:	1.5
param15:	2.0		

Image 5.3: The Edit Parameters window of the GUI.

Optionally, the user can select flat and dark images for calibration of the provided event frames. If any of those files is selected, then the corresponding calibration is automatically performed on all the event images provided.

From the starting screen of the offline detection, the user can also change the constant that is used for the flat and dark calibration as well as the size of the region of interest (ROI) image. The constant is used for the calibration of the images if a flat frame is provided, by multiplying each pixel value with the constant and dividing by the value of the pixel in the same position in the flat image. The ROI is a square around the event whose side is by default 30 pixels and can be changed by the user through the corresponding text field.


Detection
Edit Parameters
Select Dark Image for Calibration (Optional)
Select Flat Image for Calibration (Optional)
Select Directory
Start Detection

No directory chosen

No directory chosen

Specify the constant for the flat and dark calibration (Optional)

100

Please choose a folder first and then start the process.

No directory chosen

Specify ROI Dimension (Optional)

30

Image 5.4: The offline detection starting screen.

5.3 Exceptions

If the user provides an invalid input or any other unforeseen actions are performed that lead to Java exceptions, a pop-up window with an informative message will appear. The exceptions can be related to the type of directory, for example if a user provides a directory without any images or without a metadata file. Also, if the information or the format of the metadata file is incorrect, or any frames are missing, an exception window will appear. It could also be related to the position of the event in the image, due to the fact that if the event appears on the edge of the image, a proper ROI cannot be created and the algorithm fails to classify the event. In any of these cases, the user has to close the exception window and the procedure will continue for the rest of the events, if any.

5.4 Output

If the input is a single directory, then the output is a folder named Detection_Results in this directory that contains the average image of the event frames, the first frame of the event, the difference image between those two frames and the region of interest around the brightest pixel, all the images being in the same format as the input images (ff in the event folder exist frames of both types, then the result frames will be written in PNG format). Moreover, in this folder two text files are contained, one in .txt and another in .csv format. The first one contains only the classification final result (the type of the event) and its coordinates inside of the image, as read from the already existing metadata file. The second one contains the name of the folder provided, the full width half maximum values of x and y that are the results of the Gaussian fit of the algorithm, the classification result and its coordinates inside of the image.

If the input provided is a directory that contains the observations of an entire night, or more than one events, then another file is created in addition to each specific one under the same name but in the root folder that will contain the ROI images for each event and a .csv file with the list of the above details for each event. In the specific folders, there are no ROI images written.

The output is written according to the format of the images and whether the camera is 8-bit or not (the same format is kept).

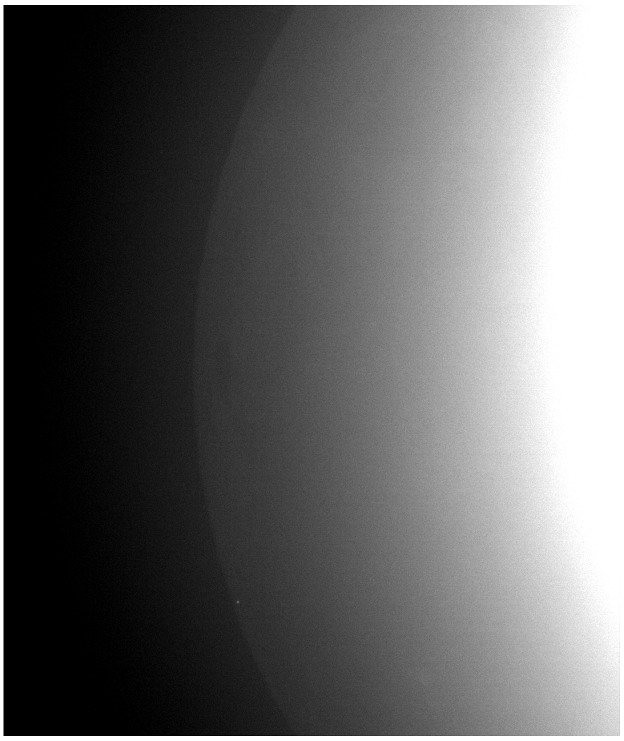


Image 5.5: An example of an event image from the NELIOTA data.

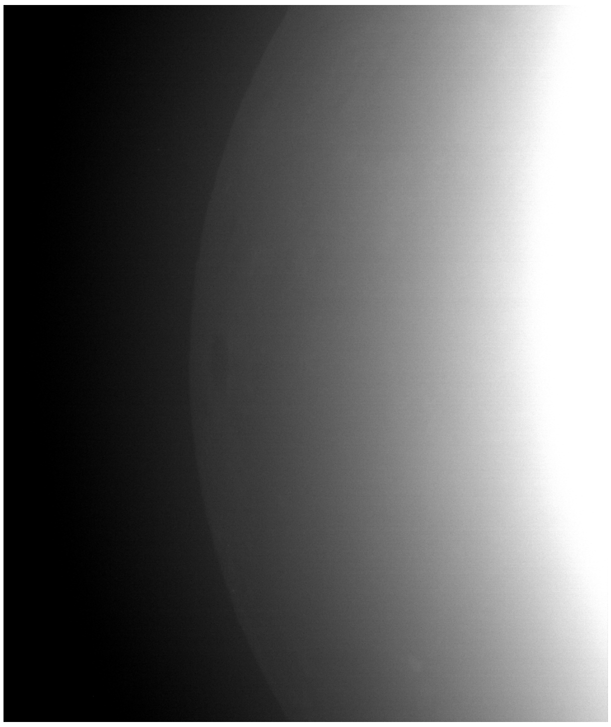


Image 5.6: An example of an average image from the NELIOTA data.

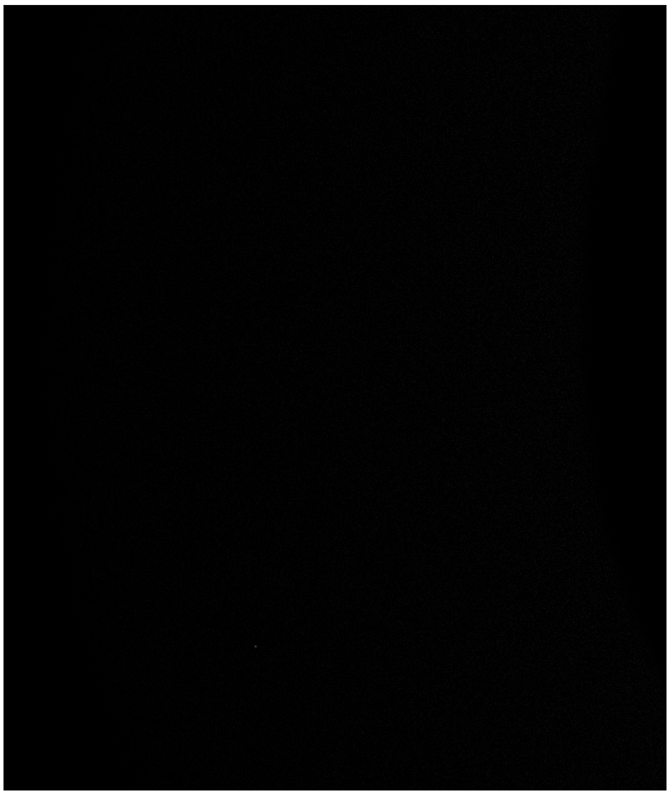


Image 5.7: An example of a difference image that is the result of the subtraction of the average frame from the first event frame.

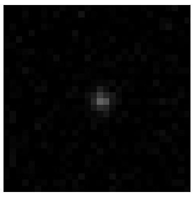


Image 5.8: An example of a ROI image.

5.5 Algorithm

The offline detection Standalone algorithm is implemented as follows:

For each event that is analyzed, if the user has selected flat and/or dark images, all the frames are calibrated. Then, the coordinates of the event, the first frame of the event name, the number of frames and the number of pixels of the event are read from the metadata file. The average image of all the event frames is calculated, by adding all the pixel values in the same position of all the frames and dividing by their number. The average frame is subtracted from the first frame of the event and on the difference image, we create the region of interest according to the user input. The ROI is a square that in the center of its surface is located the brightest pixel of the event. If the ROI selected by the user exceeds the image bounds, the ROI's size is automatically adjusted. The creation of the ROI image is important because the algorithm runs on this image, thus if the event appears on the exact edge of the image and no ROI can be created around it, the algorithm cannot be applied.

If the number of pixels of the event is 1 and the average pixel value around the brightest pixel is significantly less than its own (the difference is less than 30 on the difference image), then the event is a hot pixel.

To determine if the event is inside the lunar limb, a lunar mask is created according to the following process:

Fırstly, a heavily blurred version of the event image is created (using a σ = 150 gaussian filter) to remove the light effect. A second gaussian filter is used (σ = 54) to remove the noise. Then a Sobel filter is applied for edge detection, followed by a brightness threshold to keep only the brightest pixels of the image (that are those on the limb). A least-squares circle fitting algorithm is used on a random sample of the limb pixels to find the center and radius of the circle (in pixels) that fits the limb, applying the Landau and the RANSAC algorithm.

To check the results, the euclidean distance of all the limb pixels and the center of the circle is calculated and is checked whether this number is between the 90 and the 110 percent of the radius. If the pixels that their distance exceeds those limits are more than the pixels whose distance does not, then the lunar mask algorithm cannot be applied, it cannot be determined whether the event is inside or outside of the lunar limb and the algorithm proceeds to the classification part. If not, then from this circle, a binary mask is created, with the pixels outside of the limp being black and those inside of the limb being white. This image is multiplied with the event frame. If in the multiplication image, the brightest pixel of the event is black, then the event is outside of the limb.

To classify the event, if it is not a hot pixel nor outside of the limb, the Levenberg-Marquardt Algorithm is applied on the ROI image. If the x, y coordinates of the brightest pixel of the event obtained by the metadata file are correct, and the algorithm can be applied, the parameters of the 2D-Gausian fit are obtained. Those are: the amplitude, the offset value, the mean X position, the mean Y position, the sigma X value, the sigma Y value and the rotation angle in rad. The classification algorithm, based on the LM results, finds the max sigma value between the sigma X and sigma Y and calculates the sigma ratio as equals the max sigma to the min sigma value. The classification is based on the following cases:

- Satellite detected:
  sigma ratio > 2.7 and max sigma > 1.5

- Cosmic Ray or Impact Flash Detected:
  2 < sigma ratio < 1.5

- Impact Flash Detected:
  sigma ratio < 1.5 and max sigma > 1

- Cosmic Ray Detected:
  sigma ratio > 1.5 and max sigma < 2

- No event detected:
  Case 0: sigma ratio = 1
  Case 1: max sigma > 100 or sigma ratio > 30
  Case 2: sigma ratio > 3 and max sigma < 1
  Case 3: sigma ratio > 2.7 and max sigma > 1.5 and mean ratio < 1.2 (mean ratio = mean_Xpos/mean_Ypos, if mean_Xpos > mean_Ypos, else mean ratio = mean_Ypos/mean_Xpos)

- Unknown/the event cannot be analyzed
  The values of the fitted parameters are not inside any of the above limits.

All the above criteria are the result of thorough testing and all values for the parameters are configurable through the GUI.

5.6 Software overview

The offline part of the software takes as input directories written by the online plugin and the GUI and all output is written inside the directories provided.

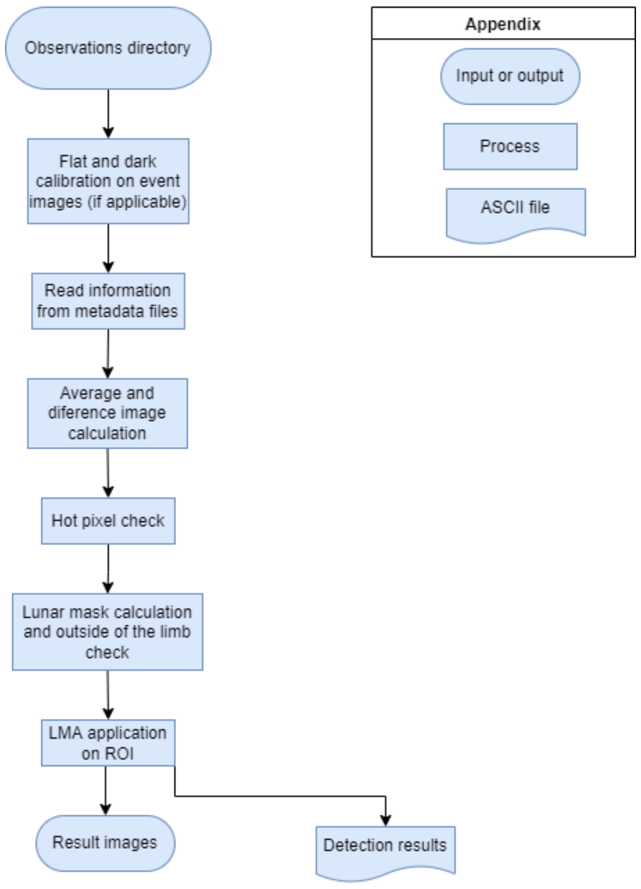

Observations directory
Flat and dark calibration on event images (if applicable)
Read information from metadata files
Average and diference image calculation
Hot pixel check
Lunar mask calculation and outside of the limb check
LMA application on ROI
Result images
Detection results
Appendix
Input or output
Process
ASCII file

Image 5.9: Diagram of the flow of the process that the operation of the offline detection follows.

5.7 System Requirements

The functional requirements are:

1. Classification of lunar events.
2. Classification on images of low and high luminosity.
3. Classification of events recorded by different telescopes and camera types.
4. Parameters and directories selection by the user through the GUI.
5. Classification for events in PNG and in FITS format.
6. Creation of .txt and .csv file containing the classification results.
7. Saving of the images used for classification in the same format and quality as the event frames.

The non-functional requirements are:

1. Progress window.
2. Pop-up messages for exceptions.

5.8 Use case

A basic use case for this part of the software is presented. All other use cases and actions that a user can perform are based on that one that refers to performing offline detection on a directory containing multiple events, of already calibrated images.

- Actor: User

- Goal: Initiate a session of offline detection on a directory that contains the events produced by the observations of an entire day.

- Scope: This case is part of the offline detection process. For this use case, the input is a directory named YYYY.MM.DD, produced by the online FireCapture plugin that contains each event folder of that specific date. The output is a folder named Detection_Results in each event folder containing the event, the stacked and the difference image in the same format as the event frames (if the format is both PNG and FITS, then the result images are of PNG type), a .txt file and a .csv file with the results of the specific folder and in the root directory another folder named Detection_Results containing a .csv file with all the subfolder results and the ROI images for each event.

- Preconditions:

1. The frames provided in PNG or/and FITS format.
2. In the directory provided there is at least one subfolder containing a valid event.
3. Metadata file in each folder to be analyzed with all the information provided by the plugin.

- Normal flow:

1. Start the Standalone Tool, press the Event Detection Button.
2. Using the Select Directory Button, select the directory that will be analyzed and press the Start Detection button to initiate the process.
3. Once the process has ended, as stated by the GUI, close the window and end the program.

- Quality attributes:

1. The results of the detection analysis will be written in a sub-directory of the root folder and of each event folder.
2. During the process, any exceptions appear as pop-up windows with the corresponding message and after the user closes them, the operation continues, stating the percentage (%) of progress.
3. If the events are valid, no exceptions appear and after the end of the process, the user is prompted to close the standalone tool and consult the files produced.

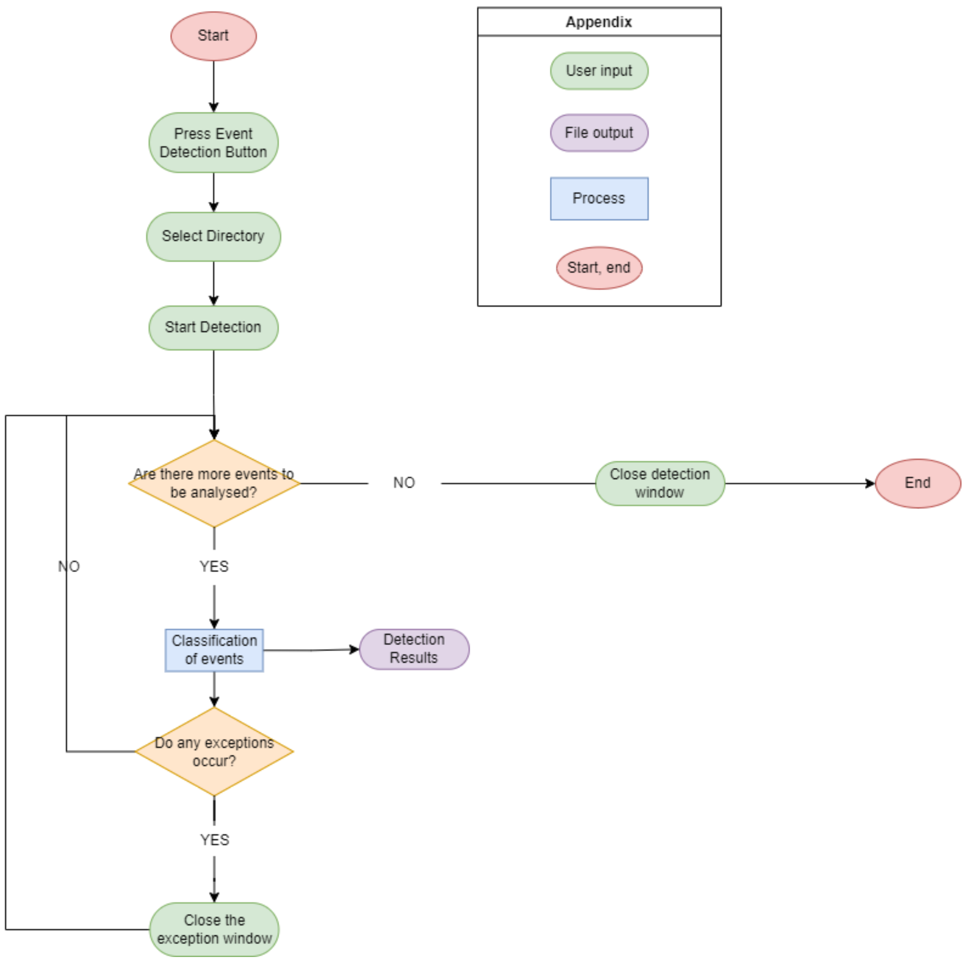


Image 5.10: Graphic representation of the workflow of the above use case.

Chapter 6: Localization

This part of the software is the last component of the FDS pipeline and its goal is to calculate the actual coordinates of the events on the lunar surface. It can be used after the offline detection process and in this case the only events analyzed are the true impact flashes, or directly after the online detection plugin and all the events are analyzed. Moreover, it can be used without making use of the former parts of the flash detection software, as long as the input for the software is in the exact same format as the output of the online detection plugin for FireCapture. In each step, another input is asked by the user, hence the graphical user interface and the input will be presented together for each step. The algorithm that this part of the software uses is based on Avdellidou et al. 2021: Impacts on the Moon: Analysis methods and size distribution of impactors paper.

6.1 The graphical user interface

Through the graphical user interface the user can interact with the software in various ways, as the algorithm is performed in steps that need the user's interference to continue.
The starting screen can be accessed by starting the DetectionStandalone application and clicking on the [Event Localization] button to access the starting screen.

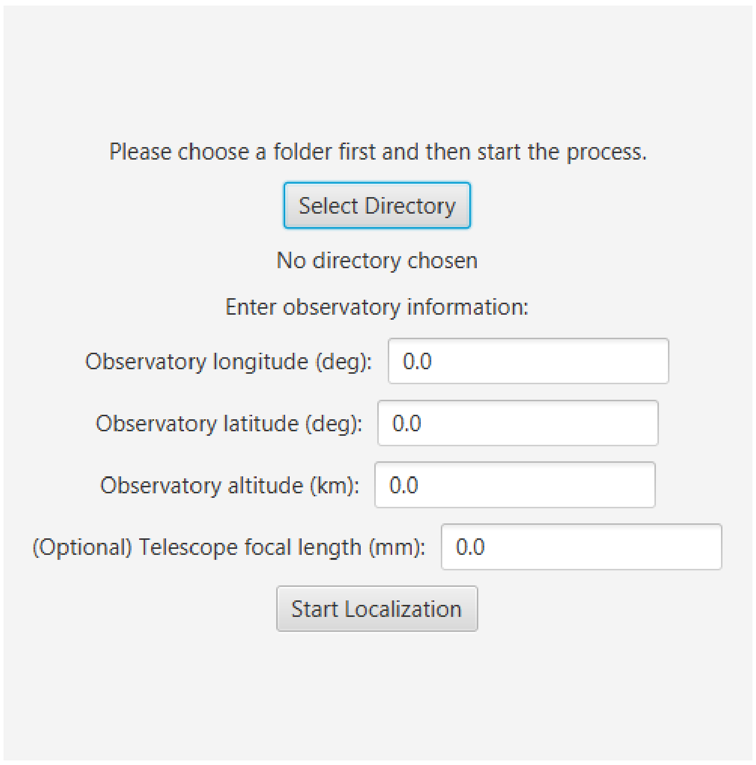
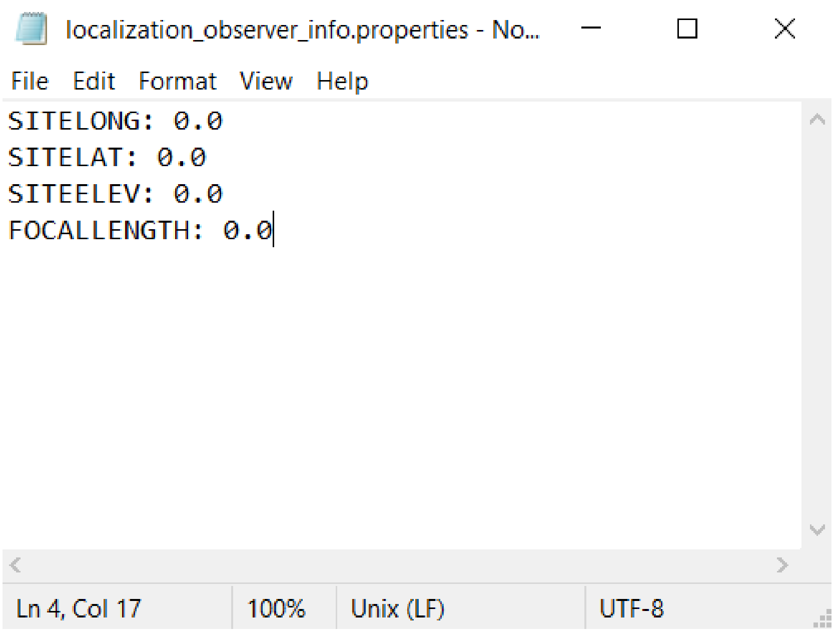

Please choose a folder first and then start the process.
Select Directory
No directory chosen
Enter observatory information:
Observatory longitude (deg): 0.0
Observatory latitude (deg): 0.0
Observatory altitude (km): 0.0
(Optional) Telescope focal length (mm): 0.0
Start Localization

Image 6.1: The localization's software starting screen.

The first step in order for the software to operate is to input crucial observatory information into the provided text fields, regarding the following:

- Longitude of observatory site (in degrees). This value must be between -360 and +360.
- Latitude of observatory site (in degrees). This value must be between -180 and +180.
- Elevation of observatory site (in km). This value must be between 0 and 8.
- Optionally, the focal length of the telescope (in mm). This text field may be left blank and any non-positive number is considered blank.

The observatory information is stored into the localization_observer_info.properties file after each use. Next time the application is opened, the input will initially be read from the properties file. The file can also be directly edited.


localization_observer_info.properties - No...
File Edit Format View Help
Ln 4, Col 17
100%
Unix (LF)
UTF-8

```
SITELONG: 0.0
SITELAT: 0.0
SITEELEV: 0.0
FOCALLENGTH: 0.0
```

Image 6.2: The localization_observer_info.properties file.

On the same screen, the user has to click [Select Directory] and then choose a directory for localization. After a directory is selected, its path will be displayed under the [Select Directory] button.

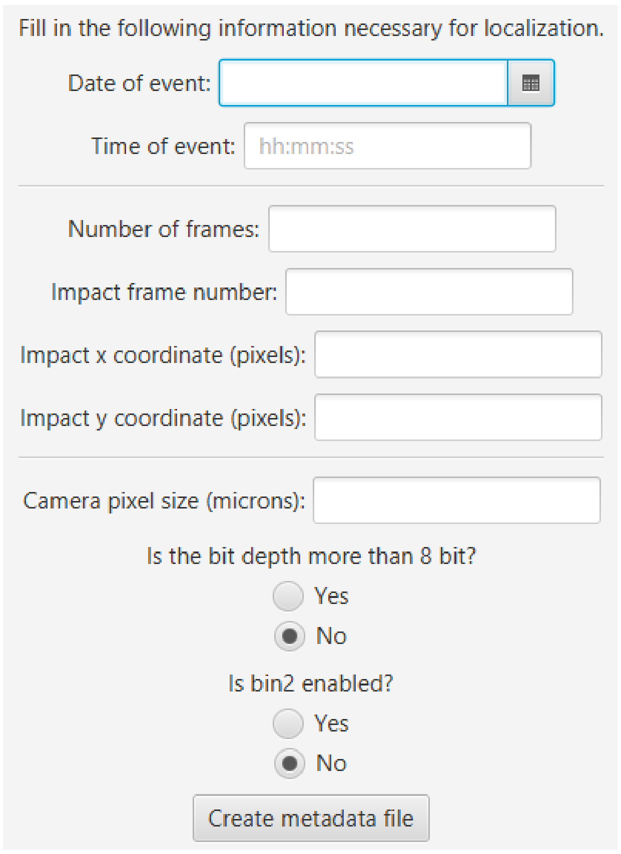
The directory chosen may be either:

- A single event directory containing frames and an Event_Metadata.txt file.
- A single event directory only containing frames, without an Event_Metadata.txt file.
- A directory containing multiple event directories. Localization will be performed sequentially on all subdirectories containing an Event_Metadata.txt file.
- A directory containing multiple event directories and an OfflineDetectionResults.csv file. Localization will be performed sequentially only on impact flash event directories.

The frames inside all the directories can be either in png, jpeg or fits format and must be named frame_000, frame_001, etc. After the directory input, the software can proceed by clicking on the [Start Localization] button.

6.1.1 Event without metadata

If a directory containing frames but not an Event_Metadata.txt file is chosen, a pop-up window will appear allowing the user to create a dummy Event_Metadata.txt file with all data required for the localization.


Fill in the following information necessary for localization.

Date of event:	
Time of event:	hh:mm:ss
Number of frames:	
Impact frame number:	
Impact x coordinate (pixels):	
Impact y coordinate (pixels):	
Camera pixel size (microns):	

Is the bit depth more than 8 bit?
Yes [✓ No]

Is bin2 enabled?
Yes [✓ No]

Image 6.3: The pop-up window that allows the user to create a custom Event_Metadata.txt file.

The necessary fields that need to be completed are the following:

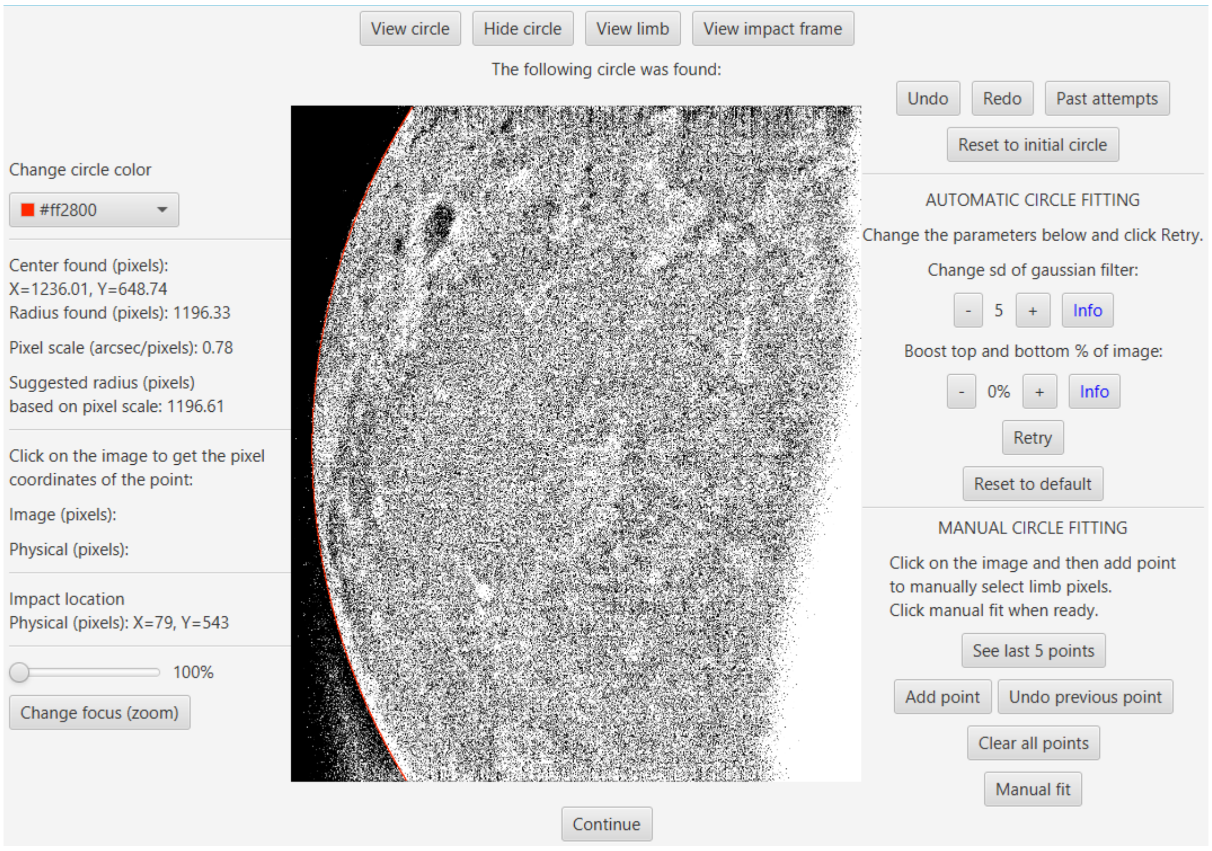
- Date of event (in any format, the calendar may also be used)
- Time of event in format hh:mm:ss
- Number of frames in the directory
- Impact frame number in the sequence, starting from 0
- Pixel coordinates of the impact flash in the first impact frame (bottom left pixel corresponds to 0,0 and top right to width-1, height-1)
- Pixel size of camera in microns (μm)
- Bit depth of the camera used (more than 8bit or not)

- Whether binning was enabled or not

When the information is completed the user can click on the [Create metadata file] button and the file will be attempted to be written. If any of the input was incorrect an error pop-up window is displayed and if not the program will return to the starting screen and the user can now click on the [Start Localization] button to begin.

6.1.2 Circle fitting

After the start of the localization, the program pauses on the circle fitting screen, showing the calculated circle.



The following circle was found:

Change circle color

🟥 #ff2800

Center found (pixels):
X=1236.01, Y=648.74
Radius found (pixels): 1196.33

Pixel scale (arcsec/pixels): 0.78

Suggested radius (pixels)
based on pixel scale: 1196.61

Click on the image to get the pixel coordinates of the point:

Image (pixels):

Physical (pixels):

Impact location
Physical (pixels): X=79, Y=543

100%

AUTOMATIC CIRCLE FITTING

Change the parameters below and click Retry.

Change sd of gaussian filter:

5

Boost top and bottom % of image:

0%

MANUAL CIRCLE FITTING

Click on the image and then add point to manually select limb pixels.
Click manual fit when ready.

Image 6.4: Localization circle fit screen.

In the middle of the screen, a filtered version of the event frames is displayed, with a red line indicating the circle found that fits the limb. The color of the line can be changed using the color picker on the top left part of the window.

On top of the image there are four buttons that provide the following options for viewing:

- [View circle] displays the image with the red circle.
- [Hide circle] removes the red circle.
- [View limb] displays a binary image, where the detected limb pixels are white.
- [View impact frame] displays the impact frame.

On the left side of the image information about the circle found is displayed, regarding the pixel coordinates of the center of the circle, the length of the circle's radius in pixels, the pixel

scale of the image in arcsec/pixels and the suggested radius in pixels, calculated using the pixel scale. The last two are displayed only if the telescope focal length was given.
Clicking on any point on the middle images shows the image and physical pixel coordinates of the point on the left side of the screen. The image coordinates correspond to the pixel location on the displayed image, while the physical coordinates correspond to the pixel location on the actual image, before it was resized to fit the screen. As stated before, the bottom left pixel of the image corresponds to (0,0). The physical pixel coordinates of the impact flash are also displayed.
On the bottom left part of the screen there is a slider to change the focus (zoom) of the image. Adjust the slider (from 100% to 10%) then click [Change focus (zoom)] to zoom in or out of the last pixel clicked inside the image. If no pixel has been clicked, it zooms in/out centered at the impact flash location.
If the circle found is satisfactory, by clicking [Continue] on the bottom of the screen, the program proceeds to the next step. It should be noted that the radius must not be smaller than 3 pixels or larger than 4000 pixels. In these cases an error pop-up window is displayed.

6.1.3 How to judge circle

If the found circle is not satisfactory, there are options to fix it on the right side of the screen. To understand if the circle is calculated correctly, the user may toggle the circle on and off to see if it fits the limb well. Furthermore, if a correct telescope focal length was given, the radius of the circle found should be as close to the suggested radius as possible. In addition, this can be judged by observing the detected limb pixels, in order to make sure that they are on the limb.

To change the circle, one way is to tweak some parameters and retry the algorithm, while another way is to manually pick limb pixels. Both ways of handling a mismatched circle are explained below.

On the top right of the screen there are some helpful buttons:

- [Undo] and [Redo]

  Its action is to undo and redo one single circle fit attempt.

- [Reset to initial circle]

  Its use is reset to the first circle found by the algorithm.

- [Past attempts]

  Its action is to display information about the past 5 circle fitting attempts. For each attempt, the radius is shown, and for the automatic attempts the parameters are also shown.

The first way to change the cycle is called automatic circle fitting and it can be done from the right part of the screen. In this section some parameters can be tweaked and when the user is ready, the [Retry] button can be pressed to attempt the circle fitting algorithm with the new parameters. The two parameters that affect the cycle can be changed by clicking the [-] and [+] buttons. Clicking on the [Info] button next to a parameter displays useful information about how it can affect our results. The sd parameter (standard deviation) can be changed from 3 to 9, with a default value of 5. The boost parameter can be changed from 0 to 25, with a default value of 0. Clicking [Reset to default] will set the parameters to their default values. In the next sections, it is described in detail how these two parameters affect the circle.

The first parameter that can be changed is the standard deviation (sd) of a gaussian filter. This filter is responsible for cleaning the image from noise and increasing the sd makes the filter stronger. This parameter should be increased if the circle found is much smaller than the moon, which is usually a consequence of too much noise.

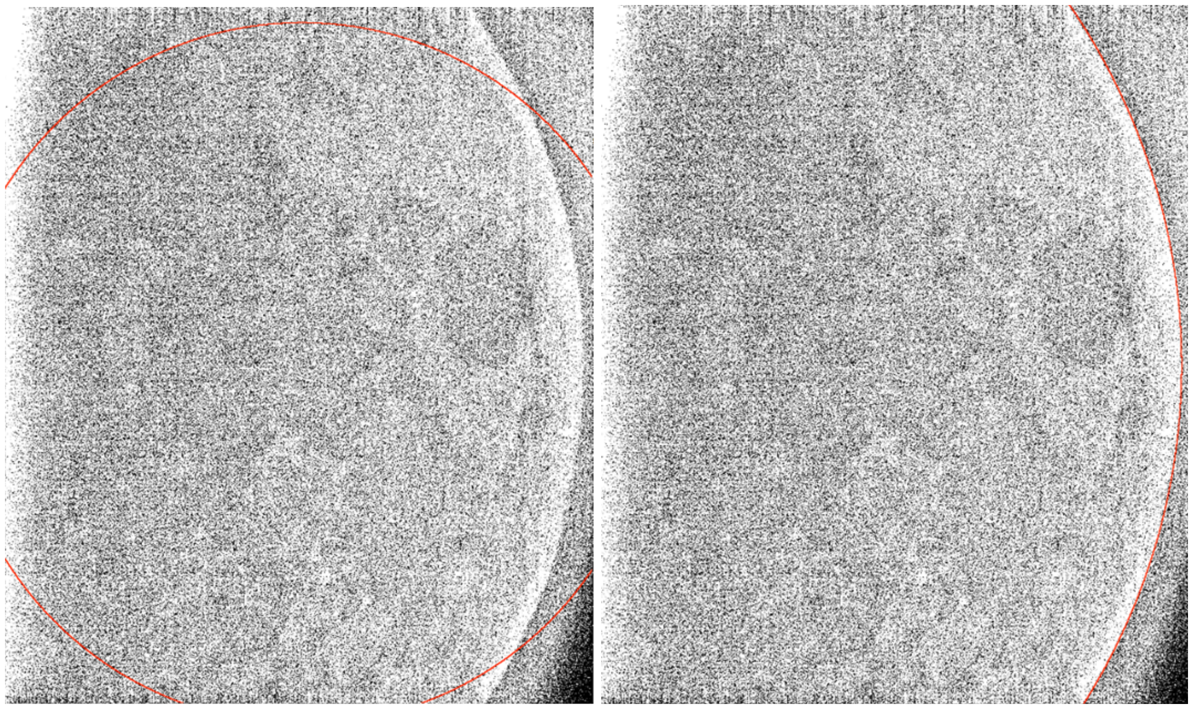


Image 6.5: Example. The left picture shows a typical case of a smaller circle found due to noise. The right picture shows the result after increasing the standard deviation from 5 to 8.

Another indicator that the sd parameter should be increased is if the limb image contains stray white pixels inside of the moon. Clicking [View limb] on the top part of the screen will display a black image where the detected limb pixels appear in white. Any white pixels inside the moon are a result of noise and cause a smaller circle to be found.

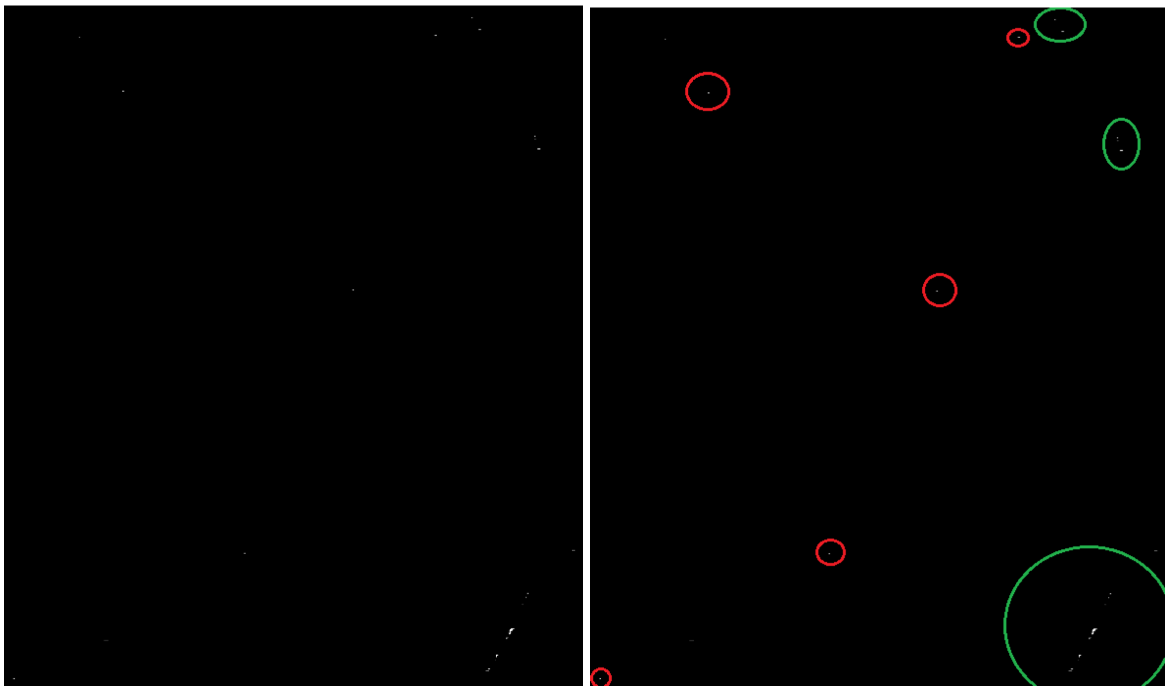


Image 6.6: Example. Limb image of the above bad circle. The green circles indicate correct limb pixels, while the red circles indicate noise pixels inside the moon. Increasing the sd removes the noise pixels.

Note that the noisier the original image is, the stricter the limb detection will be, resulting in sparser limbs. However, a sparse limb by itself is not an indication of failure. The problem is if there are white pixels far from the limb regardless of limb density.

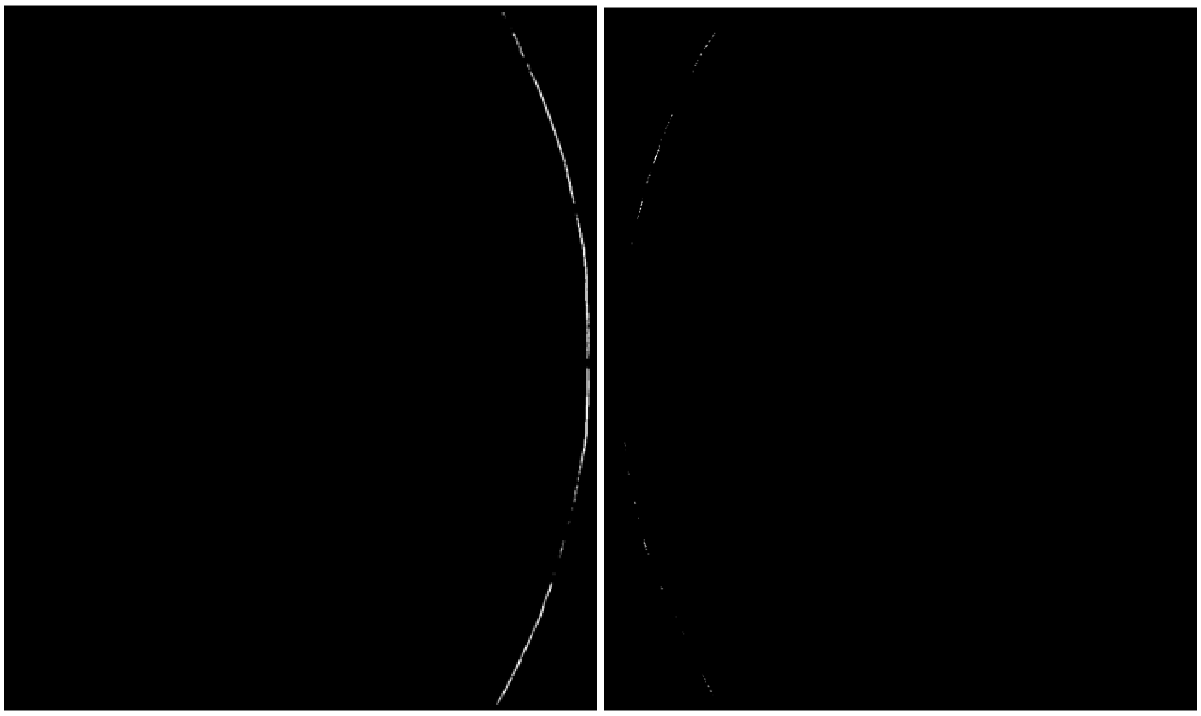


Image 6.7: Two examples of good limbs: one very visible, one sparser.

The second parameter that can be changed is the boost parameter. This parameter should be increased if the circle found is wrong near the top or bottom part of the image.

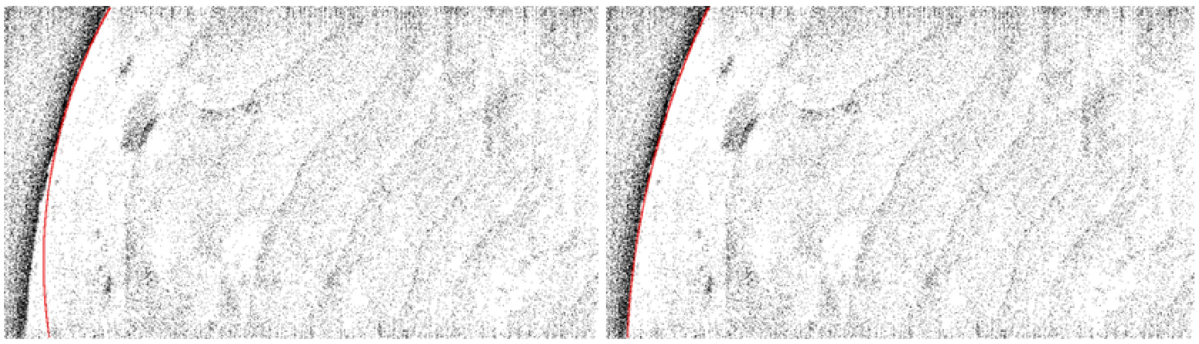


Image 6.8: Example. The left image shows a typical case where the circle found fails to follow the limb near the edge of the image. The right image shows the result after increasing the boost parameter from 0% to 10%.

This failure can also be detected by clicking the [View limb] button on the top of the window. In these cases no pixels are detected near the edge of the image. Increasing the boost parameter makes limb detection 15% less strict on the top and bottom part of the image (a % of the total image size, equal to boost parameter), resulting in more pixels detected there.

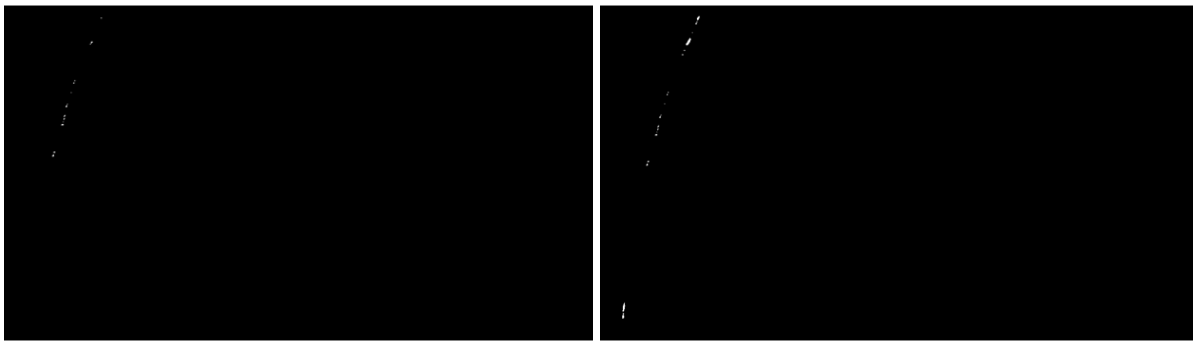


Image 6.9: Example. Limbs detected on the above pictures. On the left, no pixels were detected on the bottom part, causing the circle to fail. On the right, boosting the top and bottom 10% caused multiple pixels to be detected there, resulting in a better circle.

The second way to change the circle is to perform a manual circle fitting by selecting limb pixels in order to manually fit a circle, according to these pixels only. In this case, the user can click on any point on the image and then click [Add point] to select the pixel. This can be repeated until the user is satisfied. If any mistakes are made during the manual circle fitting, the user can remove the last added point by clicking [Undo previous point]. Moreover, with the [Clear all points] button all selected points will be removed and with the [See last 5 points] button the physical pixel coordinates of the last 5 selected points will be displayed. When the user has finished all the adjustments, they can run the circle fitting algorithm on the points selected by clicking [Manual fit].

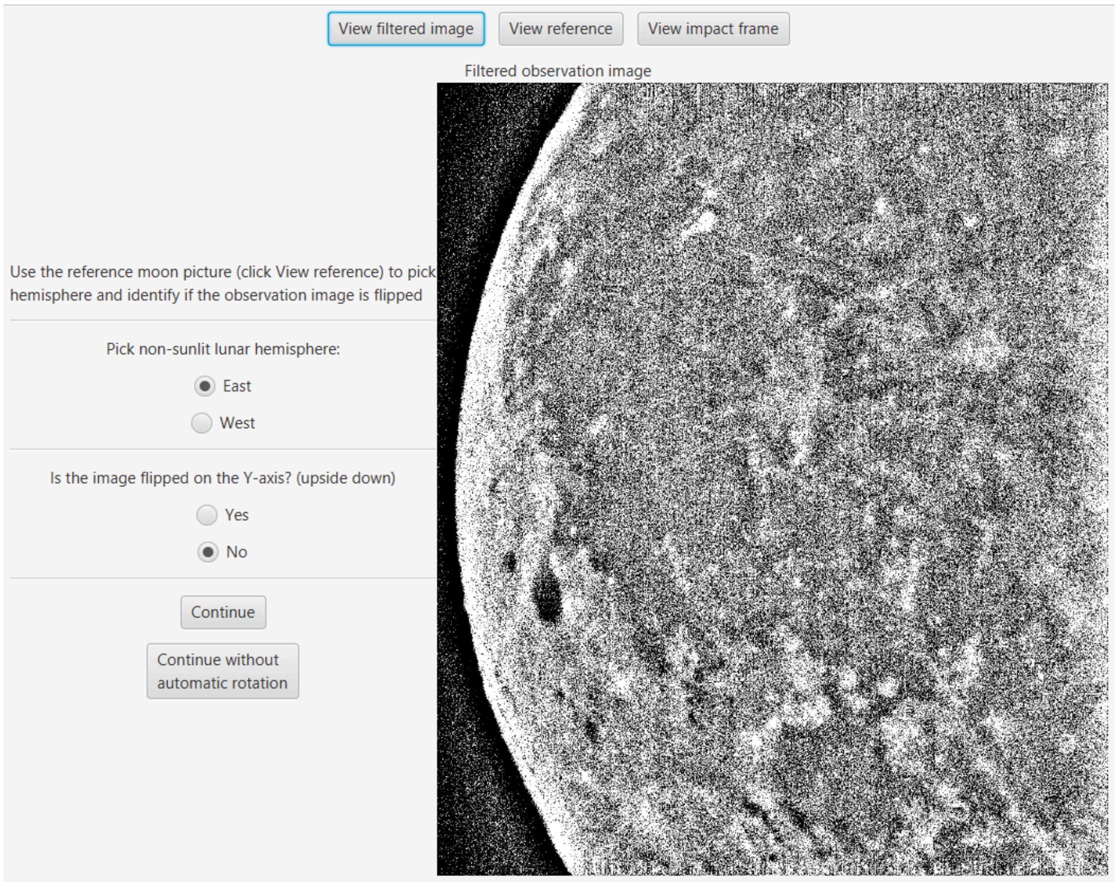
When the user is satisfied with any circle that appears, whether it is the one produced by the algorithm, the one after automatic adjustments or the one created as a result of a manual fitting, they can proceed to the next step of the algorithm by pressing [Continue].

6.1.4 Select hemisphere and orientation

After circle fitting, in the next step of the software, which is the hemisphere and orientation selection, it must be selected whether the observation image depicts the eastern or western lunar hemisphere. If a larger part of the moon is visible, the non-sunlit hemisphere should be selected. Also, it should be chosen whether the image has been flipped vertically.

On the top part of the screen are three buttons to help the user with their selection.

- The [View filtered image] button, that displays the filtered observation image.
- The [View impact frame] button, that shows the impact frame.
- The [View reference], that displays a reference image of the moon to use for comparison with the observation image.


View filtered image
View reference
View impact frame
Filtered observation image
Continue
Continue without automatic rotation

Use the reference moon picture (click View reference) to pick hemisphere and identify if the observation image is flipped

Pick non-sunlit lunar hemisphere:

[✓ East] West

Is the image flipped on the Y-axis? (upside down)

Yes [✓ No]

Image 6.10: The hemisphere and orientation selection screen.

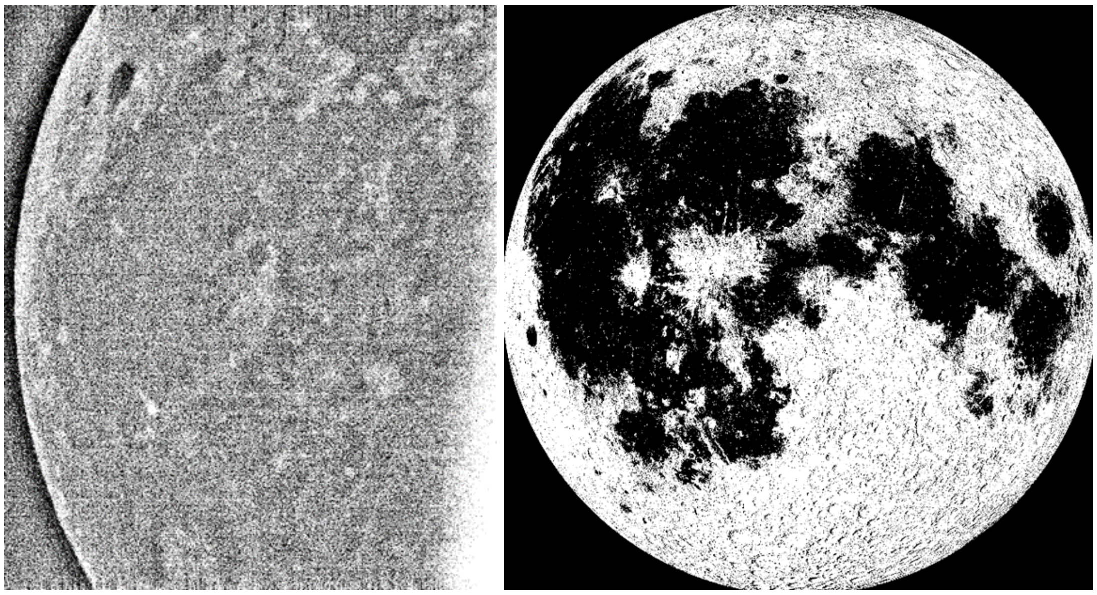


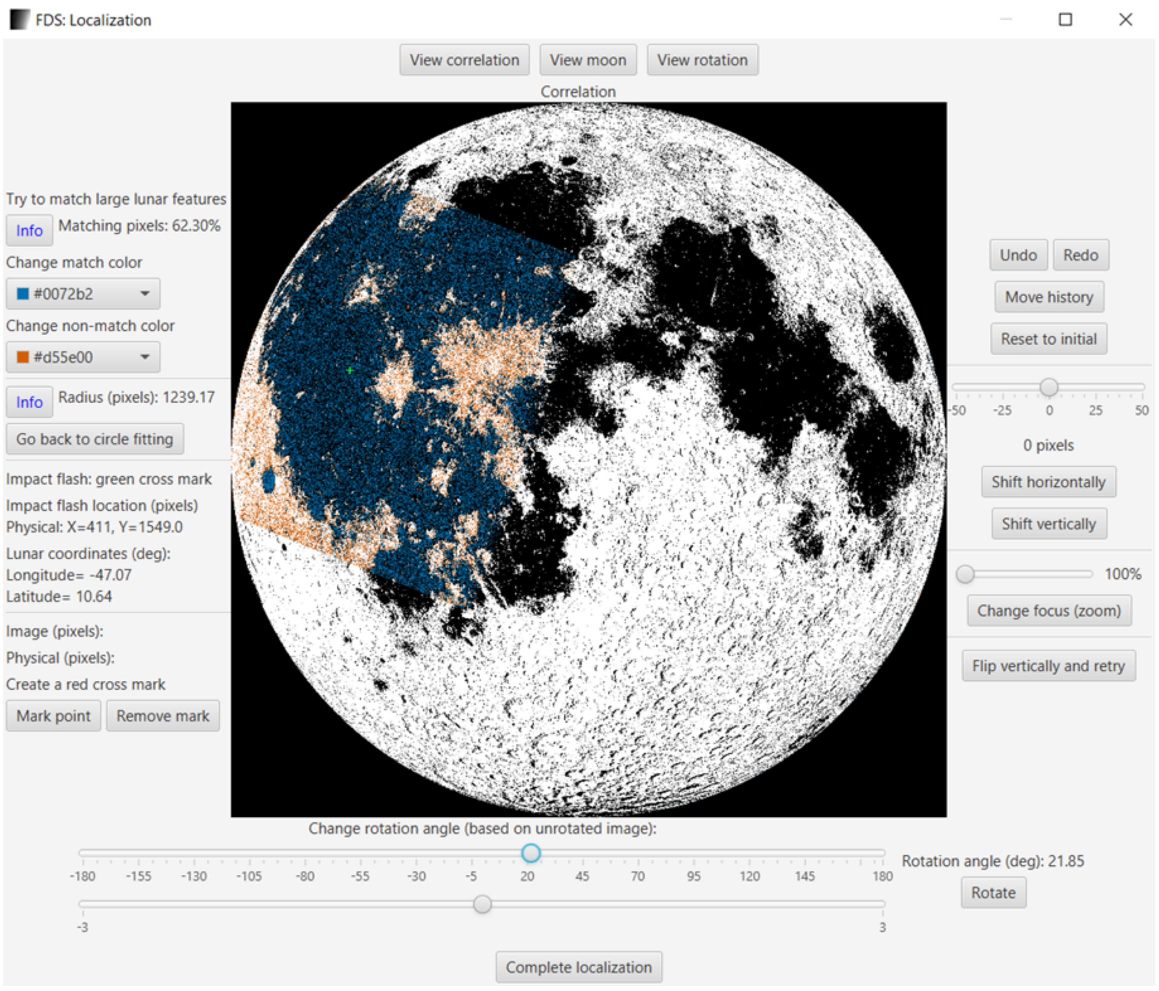
Image 6.11: Example. On the left, a filtered observation image. On the right, the reference moon picture. In this case the observation image is of the western hemisphere, due to Grimaldi crater being visible. The image is also flipped vertically, because Riccioli crater appears to the south of Grimaldi crater in the observation image, instead of to the north.

Selecting a wrong orientation or hemisphere will cause the automatic algorithm to more likely fail the next part, requiring more manual adjustments.
After clicking the [Continue] button, the next part takes a few minutes to complete, depending on how large the observation images are and during this time a transition screen is displayed. To continue to the next screen faster, the [Continue without automatic rotation] can be selected, the rotation angle will be set to 0 and manual rotation has to occur.

6.1.5 Correlation

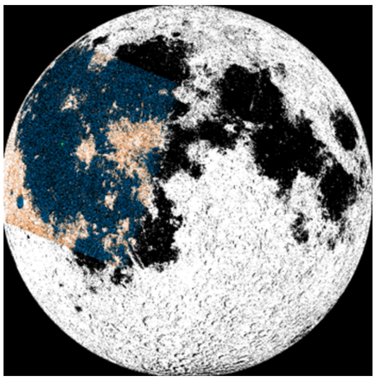
The program pauses on the correlation screen, which appears after the user continues from the hemisphere and orientation selection. In the middle of this screen, the result of correlation is displayed. If the user is satisfied, they can click [Complete localization] to write the results to a file.
On top of the image there are three buttons:

- The [View correlation] button that displays the correlation image.
- The [View moon] button that displays only the black and white moon background.
- The [View rotation] button that displays only the rotated observation image.



Try to match large lunar features
Matching pixels: 62.30%
Change match color
#0072b2
Change non-match color
#d55e00
Radius (pixels): 1239.17
Impact flash: green cross mark
Impact flash location (pixels)
Physical: X=411, Y=1549.0
Lunar coordinates (deg):
Longitude= -47.07
Latitude= 10.64
Image (pixels):
Physical (pixels):
Create a red cross mark
0 pixels
Rotation angle (deg): 21.85

Image 6.12: The correlation screen.



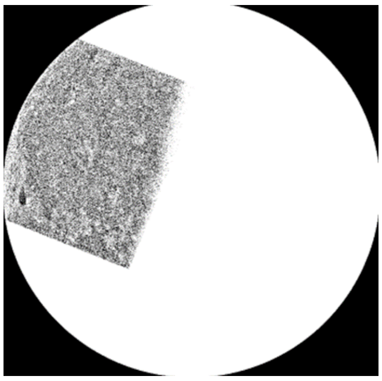


Image 6.13 (a): Correlation image Image 6.13 (c): Rotation image

Image 6.13 (b): Moon image

The correlation image is created by coloring the rotation image and imposing it onto the moon image. By default, matching black pixels between the rotation and the moon are colored blue, while pixels that are black on the rotation image and white on the moon are colored orange. These colors can be changed using the color pixels on the left part of the screen.

On the left side of the screen there is some information about the image:

- The percentage of matching pixels, as well as an information pop-up button (see next section).
- The radius of the circle in pixels, as well as an information pop-up button (see next section)
- The physical pixel coordinates and lunar coordinates of the impact flash. The flash is also marked by a green cross on the correlation image.
- The physical and image pixel coordinates of any point clicked on the images.

On the right part of the screen there is a slider to change the focus (zoom) of the image. To zoom in or out of the last pixel clicked inside the image the slider can be adjusted (from 100% to 10%) and the [Change focus (zoom)] may be clicked. If no pixel has been clicked, it zooms in/out centered at the impact flash location.

The goal of correlation is to match the rotated observation image onto the moon background. The algorithm rotates the image to find the best rotation angle, and displays its suggestion as the initial correlation. The rotation angle is shown on the bottom right part of the screen. If the user clicks [Continue without automatic rotation] in the hemisphere selection screen, the rotation angle will be 0. Otherwise, the algorithm checks for angles between -30 and +30.

To judge a correlation, the important part is that the large, identifiable lunar features should be matching (blue color). If possible, it's best that features near the impact flash are prioritized.. A different way for this to be distinguished is by toggling between the moon and rotation images: the large features should be at the same spot.

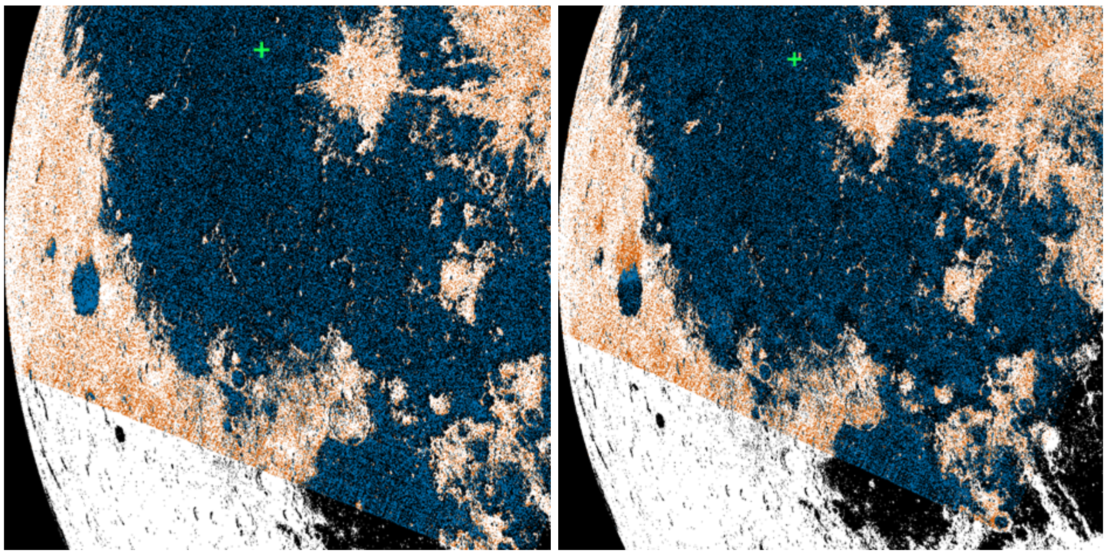


Image 6.14: Example. On the left, a good correlation is shown. Grimaldi crater has been matched correctly (blue). On the right, a worse correlation is shown. Grimaldi crater has not been matched correctly, as it is a bit higher than its correct spot (in orange).

If the automatic suggestion is unsatisfactory, or if the automatic rotation option wasn't chosen, there are several manual adjustments that can be done. Firstly, it is possible to rotate manually, by using the option on the bottom part of the screen and rotate the image manually around its center. Two sliders are provided to choose the rotation angle. The top slider has a range of (-180,180) degrees, while the second slider has a range of (-3,3) degrees, for finer adjustments. On the right of the sliders the selected rotation angle is displayed. After the adjustments of the sliders, the image can be rotated manually by clicking [Rotate]. Note that the rotation angle is always based on the unrotated image, and rotating will remove any previous shifts. In order to shift vertically or horizontally, the option to shift the image is on the right part of the screen. To set the amount of pixels to shift (from -50 to +50) the slider should be adjusted and [Shift horizontally] or [Shift vertically] should be clicked to shift the rotated image in the chosen direction. This would shift the whole rotation image including the circle. It should be noted that rotating the image after shifting removes all the shifts. To flip the image vertically, due to incorrect orientation selection in the corresponding screen, there is an option on the right side of the screen to flip the image vertically and retry the automatic rotation algorithm. After each move, the location of the impact flash is updated.

On the top left part of the screen a metric labeled Matching pixels can be found, which shows the percentage of the colored pixels that are matching (blue/(blue+orange)). This updates after each adjustment and is one way of judging them. Unfortunately, this metric is unreliable on the northwestern hemisphere due to the Ocean Procellarum causing false positive matches, as well as after shifting the image due to the lunar circles not aligning. It is best to be consulted for small adjustments.

On the top right part of the screen there are certain options to track any movements made. These are:

- The [Undo] and [Redo] buttons to undo and redo one move.
- The [Reset to initial] button to set the original rotation angle calculated automatically.

- The [Move history] button shows the last 5 moves. For rotations, shows the angle and the matching pixels %. For shifts, only shows the shift amount and direction.

Another way to track any movement is by creating a mark. By clicking anywhere on the image and then [Create mark] on the bottom left part of the screen, a red cross mark is added on the images, while [Remove mark] erases the mark. The mark belongs to the rotation image, but it is also visible on the moon and correlation images. The mark follows the movements of the rotation image (rotating and shifting) so, for example, Riccioli crater could be marked on the rotation image with the aim of moving it to its correct position. In any case, clicking [Undo], [Redo], [Reset to initial] or [Flip vertically and retry] will remove the mark.

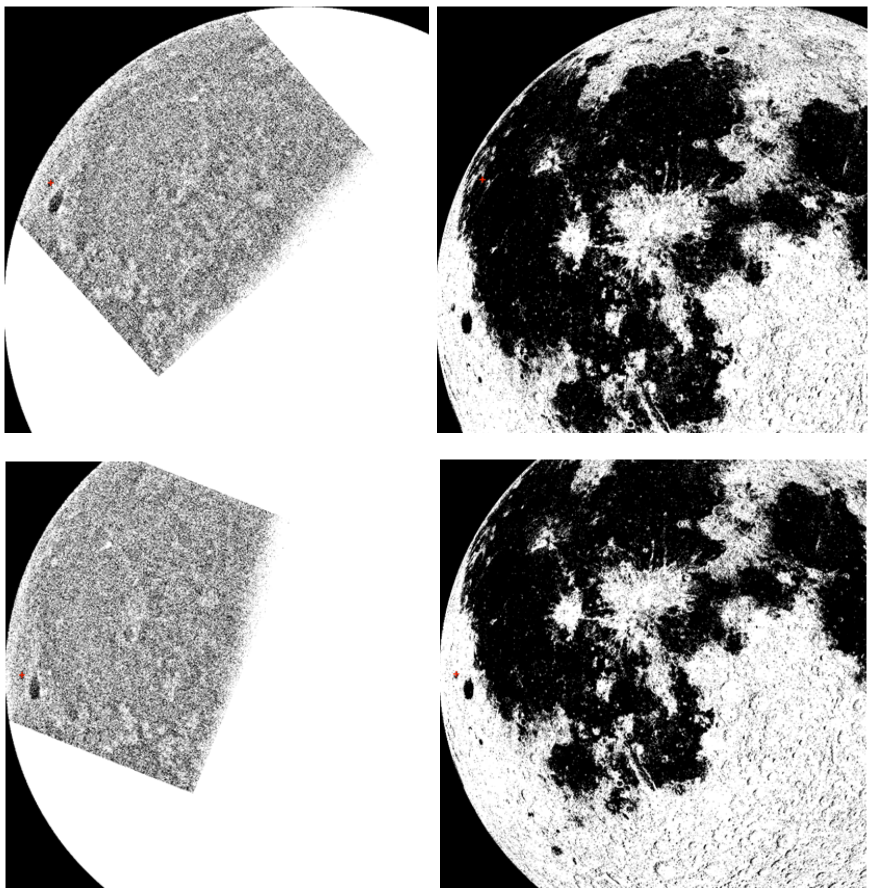


Image 6.15: Example. Marking Riccioli crater on the rotation image (top left) and its corresponding wrong position on the background moon image (top right). After rotating the image, Riciolli crater is at the correct position (bottom).

A common problem that appears is that the lunar features of the rotation image may not be the same size as the same features on the moon image. In this case it is impossible to match the images well and the reason behind this is that if the radius found from the circle fitting is wrong, this directly influences the scaling of the images. For instance, if the lunar features on the rotation image are larger than on the moon, the radius found was too small and in the same way, if the lunar features on the rotation image are smaller than on the moon, the radius found was too big. To handle this, on the left part of the screen there is a button labeled [Go back to circle fitting] which allows the user to return to the circle fitting screen to try again and find a different radius.

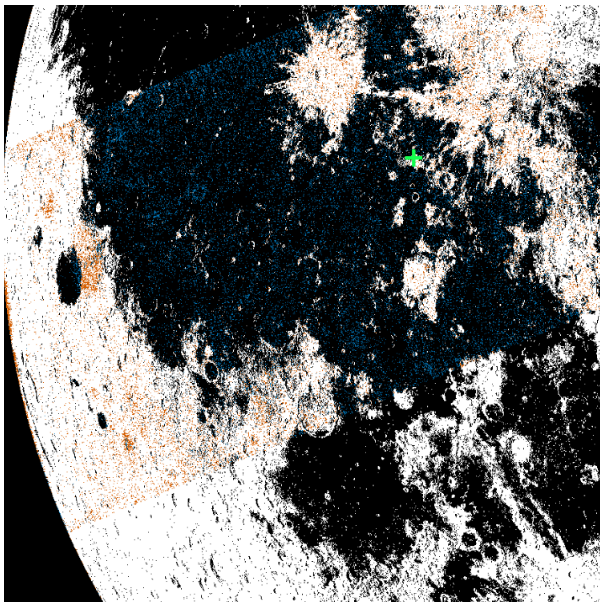


Image 6.16: Example. Grimaldi crater appears larger on the rotation image (orange) than on the moon (black).

6.1.6 Complete localization

Clicking [Complete localization] on the correlation screen will display the results of localization on a pop-up window and ask to proceed to write the results to files. If a directory with multiple events was selected for localization, the next event will automatically begin. This will also happen if the window is closed at any point during the localization. If a directory with multiple events and a detection results csv file was selected, the csv file will be written at the end, after all events were finished.

6.2 Output

The program creates a new directory called localization_results inside the event directory, and writes the results in there. If an error occurred during localization, an errorlog.txt file is also written inside the results directory describing the error.

A file called localization_logger.txt contains the localization results and information regarding:

- Path of the event folder
- Time and date localization started
- Results of localization (lunar coordinates of impact flash in degrees)
- Lunar coordinates of the center of the lunar disc observed
- Angular diameter of the lunar disc observed (arcminutes)
- Pixel scale of camera (μm), if telescope focal length was given
- Radius calculated (pixels) based on pixel scale, if telescope focal length was given
- Center of circle found (pixels) based on the observation images

- Radius of circle found (pixels)
- Rotation angle (degrees)
- Offset (pixels): shift amount on x and y axis
- Impact location (pixels) on the correlation image
- Time and date localization stopped

If localization is stopped before it completes, only the folder path, starting and stop time will be written.
Inside the results directory 9 images will be written:

- The impact frame
- Average of all event frames
- Filtered observation image (black and white)
- Result of edge detection (sobel filter)
- Result of limb detection
- Observation image with colored circle found
- Observed lunar disc
- Rotated observation image
- Correlation image

6.3 Algorithm

The algorithm for the offline localization has four distinct parts, frame processing, circle fitting, projection creation and correlation.
For each event the program begins by processing the event frames. The average image of them is created by adding all the values of the pixels that have the same coordinates in the image and dividing them by their number. This image is then divided by a heavily blurred version of itself (using a $\sigma=25$ gaussian filter) to remove the light effect.

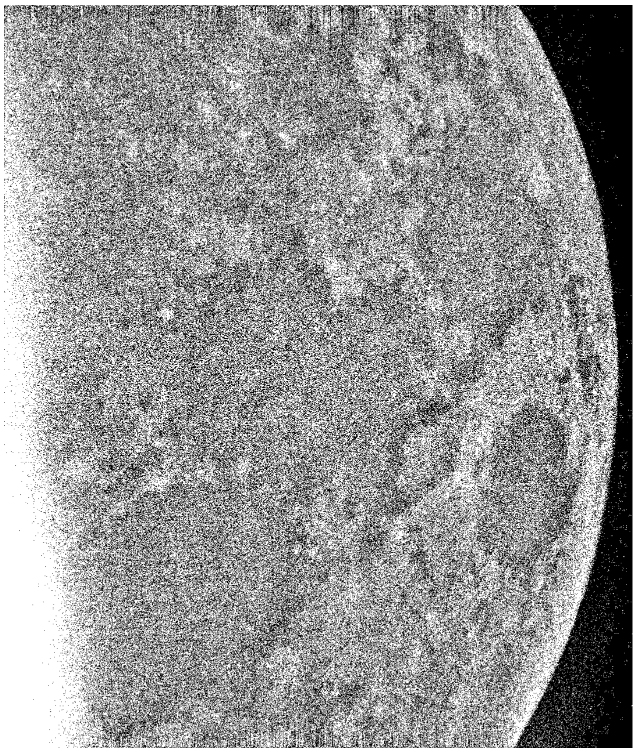


Image 6.17: Example of a removed light image.

A second gaussian filter is used (σ=5 by default, configurable) to remove the noise. Then a sobel filter is applied for edge detection, followed by a brightness threshold to keep only the brightest pixels of the image (on the limb). A least-squares circle fitting algorithm is used on a random sample of the limb pixels to find the center and radius of the circle (in pixels) that fits the limb.

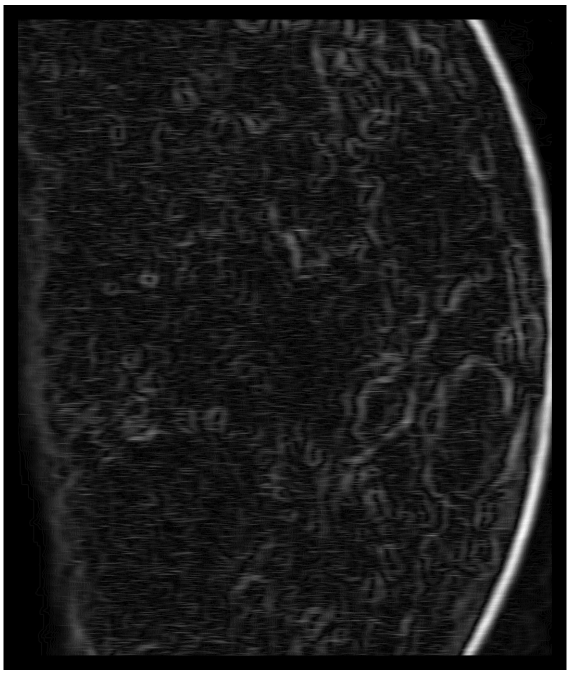


Image 6.18: Example of an image after sober filter application.

Image 6.19: Example of a lunar Mask. The pixels outside of the limb have a value of 0 and appear black, while the pixels inside of the limb have a value of 255 in 8-bit images and 65535 in 16-bit images (max value) and they appear white.

An HTTP request is sent to JPL Horizons API to obtain the angular diameter of the moon in arcseconds, which is transformed into pixels using the pixel scale calculated for the images if a telescope focal length was given as input. In that case the circle found is compared to the suggested radius calculated using the pixel scale. The results of circle fitting are displayed in the UI, allowing some user adjustments.

From JPL Horizons API the observer sub-longitude and sub-latitude (lunar coordinates of the center of the observed lunar disc) are also obtained and are used to create an orthographic projection of the moon centered around them, of the same radius as our circle. A lunar surface map reference image from LROC (Lunar Reconnaissance Orbiter Camera) is also used.

The light-removed version of the observation image is masked (20% for western hemisphere, 30% for eastern hemisphere) and fit on the circle found. The image is then rotated in a range of +-30 degrees and compared with the projection to find the optimal rotation angle based on the maximum number of matching black pixels. If instructed by user input, the image is flipped vertically. For the western hemisphere, the data is normalized by dividing with a 3rd order polynomial fit to deal with false positives caused by Ocean Procellarum.

After the optimal rotation angle is found, the results are displayed in the UI, allowing some user adjustments. Finally the pixel coordinates of the impact flash are transformed into lunar coordinates using formulas for orthographic projection:

$$\varphi = \arcsin\left(\cos c \sin\varphi_0 + \frac{y \sin c \cos\varphi_0}{\rho}\right)$$
$$\lambda = \lambda_0 + \arctan\left(\frac{x \sin c}{\rho \cos c \cos\varphi_0 - y \sin c \sin\varphi_0}\right) ,$$

where:

$$\rho = \sqrt{x^2 + y^2}$$
$$c = \arcsin\frac{\rho}{R}$$

and where x,y the pixel coordinates of the impact, λο,φο the observer sub-longitude and sub-latitude returned from JPL Horizons API, and λ,φ the lunar coordinates of the impact flash.

6.4 Software overview

The software takes input from:

- The user via the GUI
- JPL Horizons API via HTTP request
- Detection metadata file
- Event frames from file system
- Localization observer properties file

All output is stored inside the event directories.

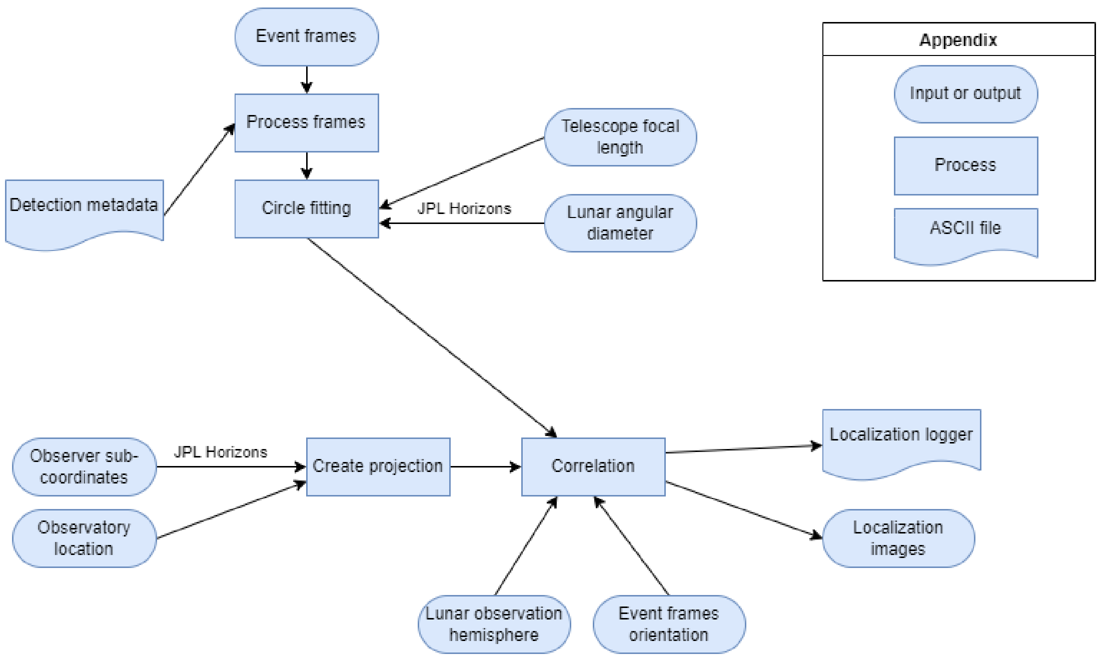


Image 6.20: Algorithm overview diagram.

6.5 System Requirements

The functional requirements of the software are:

1. Automatic localization on lunar impacts with at most 1 degree error in over 85% of cases.
2. Localization on images of low and high luminosity.
3. Localization on lunar impacts detected by different telescopes.
4. Localization on lunar impacts located at any pixel coordinates within an image, including (0,0) or maximum width/length.
5. Localization on known lunar features.
6. User adjustments during localization via GUI.
7. Png, jpeg and fits frames.

The non-functional requirements of the software are:

1. Write result images in a sub-directory of the event.
2. Write results of localization and other information (for example lunar radius and rotation angle calculated) in a logger file inside a sub-directory of the event.
3. Write the time and date of localization in a logger file inside a sub-directory of the event.
4. Read observatory information from an editable properties file.
5. Instruction and warning pop-up windows via GUI.
6. Color-blind friendly color selection.
7. Creation of a custom detection metadata file for an event.
8. Localization only on suspected NEO events as indicated by the results of offline detection.
9. Update offline detection results csv with localization results.

6.6 Use cases

Two use cases are presented, one with only the necessary user interference and another with manual adjustments.

The first use case:

- Actor: User

- Goal: Perform localization on events with no user interference apart from giving the correct input and allowing the program to continue when prompted.

- Scope: This case is part of the detection standalone tool. This case inputs the necessary parameters, carries out localization of events and writes the results.

- Preconditions:

1. The event directories contain frames of the event in png, jpeg or fits format named frame_000, frame_001, etc.
2. If a folder containing multiple events were chosen, their directories contain a valid detection metadata file.

- Normal flow:

1. Input the event folder(s), observatory information and (optionally) telescope focal length and start the process.
2. If a single event without a detection metadata file was selected, input the necessary metadata when prompted and allow the process to continue.
3. For each valid event:
   3.1 Wait for the program to finish circle fitting and allow the process to continue.
   3.2 Input the correct hemisphere and orientation and allow the process to continue.
   3.3 Wait for the program to finish correlation and allow the results to be written.

- Quality attributes:

1. The results for each event will be written in a sub-directory of the event.
2. Detection results csv, if exists, will be updated.
3. The observatory information can be read from a file editable by the user.

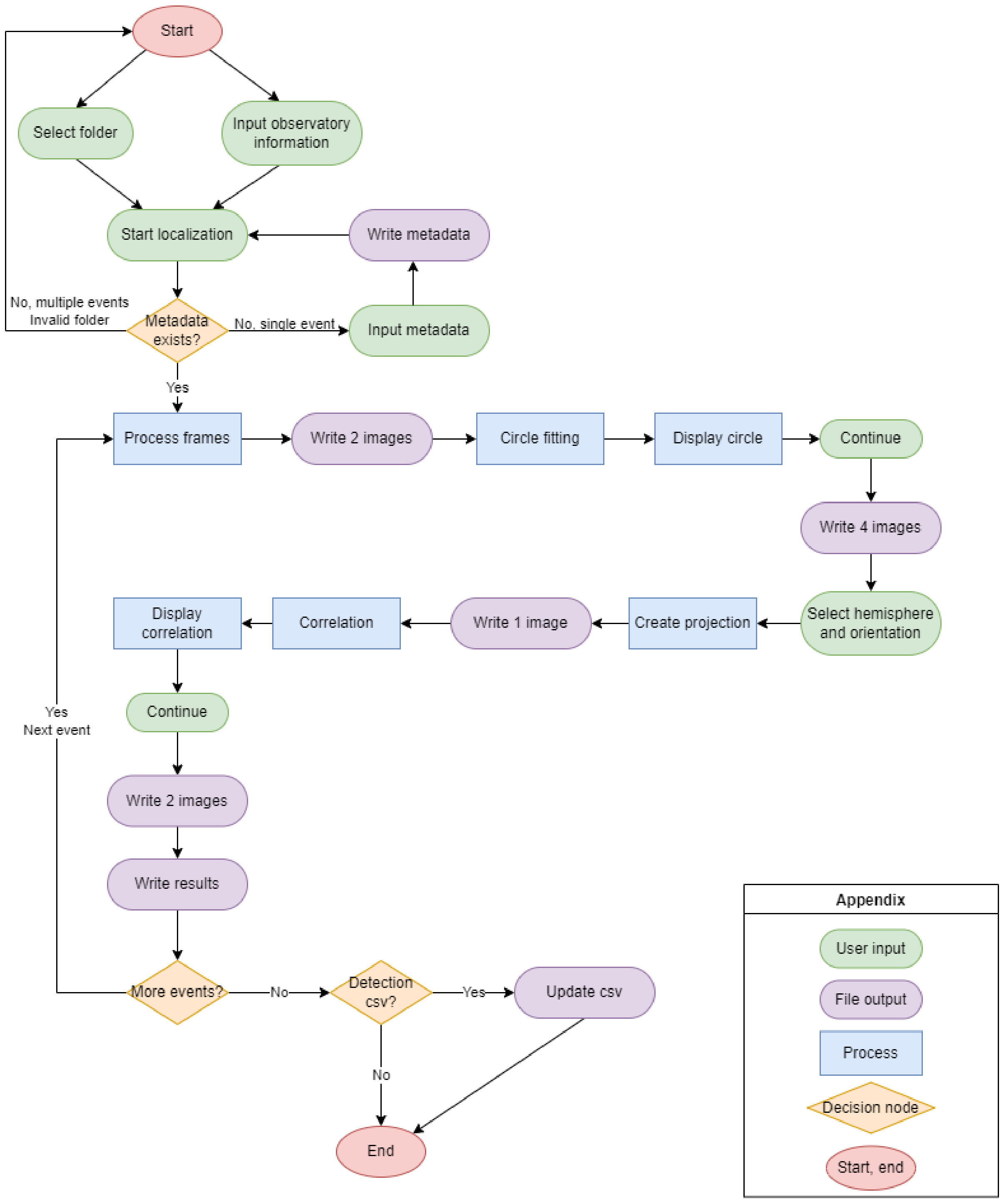


Image 6.21: Graphic representation of the workflow of the above use case.

Regarding the second use case:

- Actor: User

- Goal: Perform localization on events with user interference.

- Scope: This case is part of the detection standalone tool. This case inputs the necessary parameters, carries out localization of events and writes the results.

- Preconditions

1. The event directories contain frames of the event in png, jpeg or fits format named frame_000, frame_001, etc.
2. If a folder containing multiple events were chosen, their directories contain a valid detection metadata file.

- Normal flow

1. Input the event folder(s), observatory information and (optionally) telescope focal length and start the process.
2. If a single event without a detection metadata file was selected, input the necessary metadata when prompted and allow the process to continue.
3. For each valid event:
   3.1 Wait for the program to finish circle fitting.
   3.2 Reattempt manual or automatic circle fitting as many times necessary, then allow the process to continue.
   3.3 Input the correct hemisphere and orientation and allow the process to continue.
   3.4 Wait for the program to finish correlation.
   3.5 Move the image manually as many times as necessary then allow results to be written.

- Quality attributes

1. The results for each event will be written in a sub-directory of the event.
2. Detection results csv, if exists, will be updated.
3. The observatory information can be read from a file editable by the user.

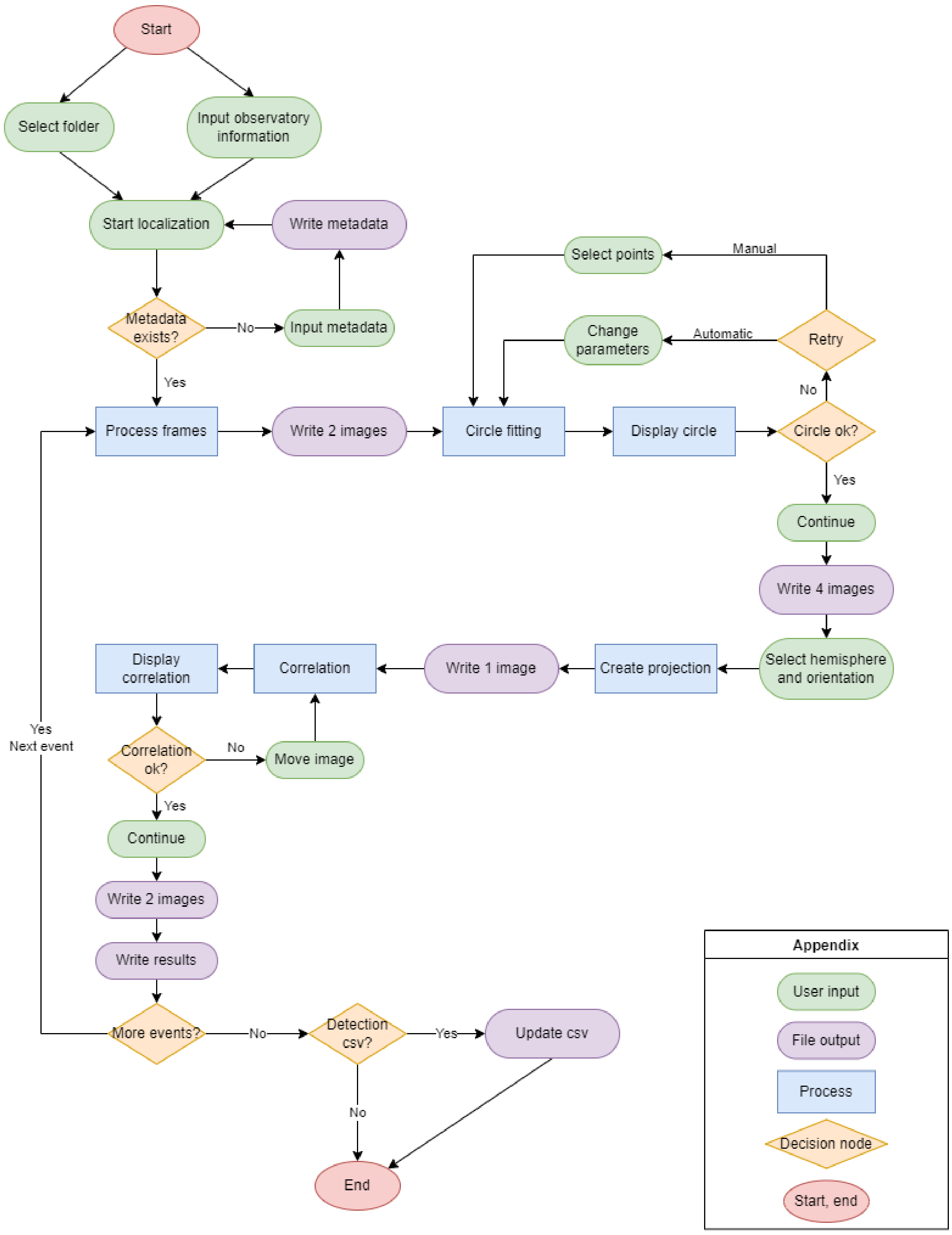


Image 6.22: Graphic representation of the workflow of the above use case.

Chapter 7 - Conclusions

For the purposes of this diploma thesis, a software for the analysis of lunar observations has been developed, and more precisely, live and asynchronous detection, offline classification and localization of lunar impact flashes has been studied and implemented. From this work, we draw certain conclusions that can be divided into three main aspects:

- Online detection.

An implementation of online detection is proposed that takes place at the same time as the observations and dramatically reduces the time, space and, in general, the resources that would be needed for the asynchronous viewing of the recorded videos.

- Offline detection and classification.

In this domain, a better understanding of the lunar events types is intended. In the observation and online recording domain, various events are recorded that not all of them are impact flashes and in order to overcome this problem, a classification method is applied on the events.

- Localization of lunar impact flashes.

When an event is confirmed, a method to calculate its lunar coordinates has been developed. In this way, the spatial distribution of the events can be analyzed and further results can be concluded. With a large number of confirmed events, using the localization method, statistics of the lunar coordinates of the flashes could be estimated.

7.1 Discussion

For creating the FDS software, the work of Alexios Liakos et al [29] in the NELIOTA project was extended and new methods for the detection and impact flash analysis are proposed. The implemented algorithms vary from quicker to more accurate ones, running on the online and the offline domain. The strategies used for the online detection take into consideration that it is a live process that has to run fast, without the user noticing any delays in receiving the images, nor during their recording. Thus, in this domain the main focus is not to miss any events, allowing false positives to be captured as well.

The offline detection and classification methods that have been created are to be used after the live process, where the recording of the events have already been obtained and stored. When the event is confirmed with the offline detection, its type is estimated considering the fitted parameters of the Gaussian distribution. This study performs the Levenberg-Marquardt algorithm on all the different events that are recorded. We conclude that this approach can conclude on the event type and it can distinguish impact flashes with great certainty. This algorithm does not constitute a black box, as the parameters that are used for the classification can be seen and modified before each session. In order to successfully fine-tune them, the users may have to examine the effect of the observation conditions on the recorded images. By classifying these events based on their size, shape and luminosity, it is identified which events are true impact flashes, while understanding that most of the events that are captured are cosmic rays. This observation can be justified by the frequency of the impact flashes as opposed to the likelihood of a cosmic ray bouncing on the telescope.

Finally, a method to specify the lunar coordinates of the flashes has been investigated, which contributes to the analysis of their spatial distribution on the lunar surface. This is a highly structured approach that needs capturing and observatory information to function accurately without user interference, but also provides the ability of manual adjustments during its use.

7.2 Future Work

At the end of this thesis, proposals for new road openings are suggested to enhance the capabilities of the software developed and to provide more specific information regarding the impact flashes. The following points are proposed to be explored in future work:

- Automatic rough magnitude calculation.

A photometric analysis could be performed, based on the value of the brightest pixels of the impact flash. A region of interest could be created in the same way that it is created for the offline classification, namely, creating a square around the brightest pixel of the impact flash, whose size could be adjusted by the user. Moreover, in a similar manner, a region of interest would be created around a star of known magnitude. The results of the two regions of interest would then be compared, to induct a calculation regarding the impact flash's magnitude [24]. During the observation, the standard stars that will be used for reference have to be monitored as well. This problem is complicated due to the fluctuation of the background of the areas observed and this could be taken into consideration by performing a dynamic background subtraction, in the same way as it is implemented in the online plugin. In any case, further adjustments and normalizations should be made, due to the difference in exposure times that are used for the flashed and the standard stars observations, due to the scintillation effect [1].

- Systematic archiving of the results.

The results acquired by the software could be generalized and further studied on a larger scale if a systematic archiving of them could be carried out. A universal format for their storage would have to be agreed on and a database could be created for this purpose or , alternatively, an already existing database could be used. This archiving could open the road for better understanding and analyzing of the results.

- Detecting impact flashes on other planets.

The algorithm for the online and the offline detection could work for detecting impact flashes and events of planets of our solar system, by making adjustments regarding the challenges that each planet could impose.

References

[1] NELIOTA: Methods, statistics and results for meteoroids impacting the Moon, A. Liakos, A.Z. Bonanos, E.M. Xilouris, D. Koschny, I. Bellas-Velidis, P. Boumis, V. Charmandaris, A. Dapergolas, A. Fytsilis, A. Maroussis, R. Moissl 2020, A&A, 633, 112.

[2] NELIOTA: First temperature measurement of lunar impact flashes, A.Z. Bonanos, C. Avdellidou, A. Liakos, E.M. Xilouris, A. Dapergolas, D. Koschny, I. Bellas-Velidis, P. Boumis, V. Charmandaris, A. Fytsilis, A. Maroussis 2018, A&A, 612, 76.

[3] NELIOTA: The wide-field, high-cadence lunar monitoring system at the prime focus of the Kryoneri telescope, E.M. Xilouris, A.Z. Bonanos, I. Bellas-Vellidis, P. Boumis, A. Dapergolas, A. Maroussis, A. Liakos, I. Alikakos, V. Charmandaris, G. Dimou, A. Fytsilis, M. Kelley, D. Koschny, V. Navarro, K. Tsiganis, K. Tsinganos 2018, A&A, 619, 141.

[4] Ortiz J.L., Aceituno F.J., Aceituno J., 1999. A&A, 343, L57.

[5] Madiedo, José M. & Ortiz, J. & Morales, Nicolas. (2018). First determination of the temperature of a lunar impact flash and its evolution. Monthly Notices of the Royal Astronomical Society. 480. 10.1093/mnras/sty1862.

[6] A. Liakos, “Study of the Light-Time effect (LITE) and the movements of the line of apsides in double elliptic star systems: TX Her, IQ Per, UX Eri”, 2006.

[7] A. Liakos, “CCD photometry through the eclipses of variable stars and study of their O-C diagrams”, 2008.

[8] G. Deng and L. W. Cahill, "An adaptive Gaussian filter for noise reduction and edge detection," 1993 IEEE Conference Record Nuclear Science Symposium and Medical Imaging Conference, 1993, pp. 1615-1619 vol.3, doi: 10.1109/NSSMIC.1993.373563.

[9] Sobel, I. and Feldman, G. (1968) A 3 × 3 Isotropic Gradient Operator for Image Processing. A Talk at the Stanford Artificial Project, 271-272.

[10] Vincent, O. Rebecca, and Olusegun Folorunso. "A descriptive algorithm for sobel image edge detection." Proceedings of informing science & IT education conference (InSITE). Vol. 40. 2009.

[11] Yuval, F. (1996). Fractal image compression (theory and application). Institute for nonlinear Science, University of California, San Diego, USA.

[12] Dorman, I. V., “Cosmic rays”, Moscow Izdatel Nauka, 1981.

[13] A. Chevallier, F. Cazals, Wang-Landau algorithm: An adapted random walk to boost convergence, Journal of Computational Physics, Volume 410, 2020, 109366, ISSN 0021-9991, https://doi.org/10.1016/j.jcp.2020.109366.

[14] M.A. Fischler and R.C. Bolles. Random sample consensus: A paradigm for model

fitting with applications to image analysis and automated cartography. Communications of the ACM, 24(6):381–395, 1981.

[15] Konstantinos, G., Derpanis, “Overview of the RANSAC Algorithm”, May 13, 2010.

[16] Ranganathan, Ananth. "The levenberg-marquardt algorithm." Tutorial on LM algorithm 11.1 (2004): 101-110.

[17] Moré, Jorge J. "The Levenberg-Marquardt algorithm: implementation and theory." Numerical analysis. Springer, Berlin, Heidelberg, 1978. 105-116.

[18] K. Levenberg, “A method for the solution of certain problems in least squares, Quart. Appl. Math., 1944, Vol. 2, pp. 164–168.

[19] D. Marquardt, “An algorithm for least-squares estimation of nonlinear parameters,” SIAM J. Appl. Math., 1963, Vol. 11, pp. 431–44.

[20] Gavin, Henri P. "The Levenberg-Marquardt algorithm for nonlinear least squares curve-fitting problems." Department of Civil and Environmental Engineering, Duke University 19 (2019).

[21] Madiedo, José M. & Ortiz, José & Morales, Nicolas & Cabrera-Caño, Jesús. (2015). Midas: Software For The Detection And Analysis Of Lunar Impact Flashes. Planetary and Space Science. 111. 10.1016/j.pss.2015.03.018.

[22] Cudnik B. M., 2009, Lunar Meteoroid Impacts, Springer Verlag, New York.

[23] Keys R. G., 1981, Cubic convolution interpolation for digital image processing. IEEE Trans. Acoustics, Speech & Signal Proc., 29, 1153-1160.

[24] Madiedo, José M. & Ortiz, José & Organero, Faustino & Ana-Hernández, Leonor & Fonseca, Fabiola & Morales, Nicolas & Cabrera-Caño, Jesús. (2015). Analysis of Moon impact flashes detected during the 2012 and 2013 Perseids. Astronomy and Astrophysics. 577. 10.1051/0004-6361/201525656.

[25] Avdellidou, C., “Lunar impact flashes: first detection from the Observatory of Nice”, 2020. doi:10.5194/epsc2020-744.

[26] Chrysa Avdellidou, Edhah Munaibari, Raven Larson, Jeremie Vaubaillon, Marco Delbo, Paul Hayne, Mark Wieczorek, Daniel Sheward, Antony Cook, Impacts on the Moon: Analysis methods and size distribution of impactors, Planetary and Space Science, Volume 200, 2021, 105201, ISSN 0032-0633, https://doi.org/10.1016/j.pss.2021.105201.

[27] Avdellidou, C. and Vaubaillon, J., “Temperatures of lunar impact flashes: mass and size distribution of small impactors hitting the Moon”, <i>Monthly Notices of the Royal Astronomical Society</i>, vol. 484, no. 4, pp. 5212–5222, 2019. doi:10.1093/mnras/stz355.

[28] Larson, R., Hayne, P., and Avdellidou, C., “Automating the detection and coordinate identification of impact flashes on the Lunar surface”, vol. 2019, 2019.

[29] NELIOTA Lunar Impact Flash Detection and Event Validation, A. Liakos, A. Bonanos, E. Xilouris, I. Bellas-Velidis, P. Boumis, V. Charmandaris, A. Dapergolas, A. Fytsilis, A. Maroussis, D. Koschny, R. Moissl, V. Navarro 2019, Proceedings of the "ESA NEO and Debris Detection Conference - Exploiting Synergies".