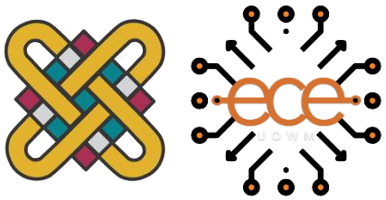


Πανεπιστήμιο Δυτικής Μακεδονίας

Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Διπλωματική Εργασία

Ανάλυση οπτικού πεδίου με την χρήση παράλληλης επεξεργασίας

Viewshed analysis using parallel processing

Χρήστος Μιχαήλ

Επιβλέποντες : Δρ. Μηνάς Δασυγένης

Δρ. Ιωάννης Μανάκος [1]

Εργαστήριο Ψηφιακών Συστημάτων και

Αρχιτεκτονικής Υπολογιστών

Κοζάνη, Ιούλιος 2020

---

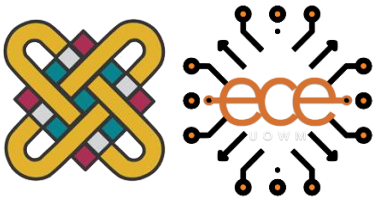
[1] Ερευνητής Γ', Εθνικό Κέντρο Έρευνας και Τεχνολογικής Ανάπτυξης, Ινστιτούτο Τεχνολογιών Πληροφορικής και Επικοινωνιών

Περίληψη

Η διαθεσιμότητα των γεωχωρικών δεδομένων αυξάνεται καθημερινά, καθιστώντας απαραίτητη την ανάπτυξη εφαρμογών που επεξεργάζονται δεδομένα μεγάλων περιοχών εδάφους σε υψηλή ευκρίνεια. Η ανάλυση οπτικού πεδίου είναι μια από τις εφαρμογές επεξεργασίας γεωχωρικών δεδομένων, η οποία έχει στόχο τον υπολογισμό της ορατότητα των σημείων ενός ψηφιακού υψομετρικού μοντέλου από έναν παρατηρητή για μια προκαθορισμένη περιοχή. Αποτελεί μια εφαρμογή με ευρεία χρήση σε πολλά επιστημονικά πεδία για έργα, τόσο πρακτικής, όσο και αισθητικής αξίας. Για παράδειγμα, η ανάλυση μη όχλησης οπτικού πεδίου του παρατηρητή διευκολύνει τη χωροθέτηση περιοχών, στις οποίες η ορατότητα στο τοπίο δεν εμποδίζεται από ανθρωπογενείς κατασκευές. Αποτέλεσμα είναι ο εντοπισμός περιοχών που παρουσιάζουν σπάνια φυσική ομορφιά, επιτρέποντας την εκμετάλλευση τους για την κάλυψη αναγκών αναψυχής. Παρά τη χρησιμότητα της τεχνικής αυτής, οι αλγόριθμοι ανάλυσης οπτικού πεδίου απαιτούν υψηλή επεξεργαστική ισχύ, αλλά και αυξημένο χρόνο για την εκτέλεση τους, περιορίζοντας έτσι την χρησιμότητα της εφαρμογής. Λαμβάνοντας υπόψη τους περιορισμούς αυτούς, στην παρούσα διπλωματική αναπτύχτηκε ένας αλγόριθμος ανάλυσης οπτικού πεδίου που πετυχαίνει αυξημένη ακρίβεια, ενώ ταυτόχρονα διατηρεί γρήγορο χρόνο εκτέλεσης.

Ο κώδικας που δημιουργήθηκε έχει ως βάση την μέθοδο του Van Kreveld [1], η οποία παρέχει υψηλή απόδοση και ευστοχία σε σύγκριση με παρόμοιες μεθόδους. Η παραλλαγή που εφαρμόστηκε αφορά στη χρήση της τεχνικής της παρεμβολής για τον υπολογισμό του ύψους των σημείων του υψομετρικού μοντέλου, με στόχο τη βελτίωση της ακρίβειας του αλγορίθμου. Ταυτόχρονα, τροποποιήθηκε ο αλγόριθμος, ώστε να είναι συμβατός με τεχνικές παράλληλου προγραμματισμού. Η τεχνική που επιλέχτηκε αφορά στη βελτιστοποίηση της διεπαφής μεταβίβασης μηνυμάτων (MPI) πετυχαίνοντας σημαντική μείωση στο χρόνο εκτέλεσης. Αποτέλεσμα είναι ένας αλγόριθμος που παρέχει ακριβές αποτέλεσμα σε ρεαλιστικό χρόνο εκτέλεσης, ακόμα και με μεγάλο όγκο δεδομένων.

Λέξεις κλειδιά: Τεχνική παρεμβολής, Αλγόριθμος Van Kreveld, Ανάλυση οπτικού πεδίου, Παράλληλη επεξεργασία, Διεπαφή Μεταβίβασης Μηνυμάτων, Γραμμή ορατότητας



University of Western Macedonia

Department of Electrical and Computer Engineering

Viewshed analysis using parallel processing

Christos Michail

Supervisors: Dr. Minas Dasygenis

Dr. Ioannis Manakos [2]

Laboratory of Digital Systems and

Computer Architecture

Kozani, July 2020

[2] Associate Researcher, Centre for Research and Technology Hellas, Information Technologies Institute

Abstract

The availability of geospatial data is increasing daily, making the development of applications that process data for large areas in high definition necessary. Viewshed analysis is one of the applications that processes geospatial data to calculate the visibility of every point of the digital elevation model from an observer for a predetermined area. It is an application widely used in many scientific fields for projects with practical and/ or aesthetic value. For example, analysing the disturbance of the field of view facilitates the recognition of areas, whose visibility is not blocked by anthropogenic constructions. As a result, the detection of the areas that exhibit rare natural beauty allows their exploitation for recreational purposes. Regardless of the great benefits of the visual analysis, there is a major drawback in the usability due to the requirements of advanced processors and the increased time required to derive the results; thus, limiting their practicality. Aim of this study is to develop an algorithm that optimizes high accuracy while minimizing the time necessary to achieve it.

The developed algorithm is based on Van Kreveld's [1] method, which is most effective in reaching high performance alongside precision, when compared to similar algorithms. Meanwhile, the interpolation technique is used in order to calculate the exact altitude of every available spot, which leads to maximizing the precision of the algorithm. Furthermore, we succeed in decreasing the execution time by implementing parallelization techniques with the use of message passing interface (MPI). This results in an algorithm that manages both precision and realistic execution time, which is practical for everyday use.

key words: Interpolation technique, Van Kreveld’s algorithm, Viewshed analysis, Parallelization, Message Passing Interface, Line of sight

Δήλωση Πνευματικών Δικαιωμάτων

Δηλώνω ρητά ότι, σύμφωνα με το άρθρο 8 του Ν. 1599/1986 και τα άρθρα 2,4,6 παρ.3 του Ν. 1256/1982, η παρούσα Διπλωματική Εργασία με τίτλο: "Ανάλυση οπτικού πεδίου για την υποστήριξη οικοσυστημικής υπηρεσίας για αναψυχή" καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας και αναφέρονται ρητώς μέσα στο κείμενο που συνοδεύουν, και η οποία έχει εκπονηθεί στο Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Πανεπιστημίου Δυτικής Μακεδονίας, υπό την επίβλεψη του μέλους του Τμήματος κ. Μηνά Δασυγένη και του ερευνητή του Εθνικού Κέντρου Έρευνας και Τεχνολογικής Ανάπτυξης κ. Ιωάννη Μανάκου, αποτελεί αποκλειστικά προϊόν προσωπικής εργασίας και δεν προσβάλλει κάθε μορφής πνευματικά δικαιώματα τρίτων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή/και πηγές άλλων συγγραφέων, αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή. Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα. Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και μόνο

Υπογραφή Φοιτητή

Ευχαριστίες

Θερμές ευχαριστίες στους Δρ. Μηνά Δασυγένη και Δρ. Ιωάννη Μανάκο για την σημαντικότατη αρωγή και καθοδήγηση στην εκπόνηση της παρούσας μελέτης. Επίσης, αναφορά πρέπει να γίνει στον κ. Marco Heurich, επικεφαλή του τμήματος διαχείρισης επισκεπτών και παρακολούθησης του εθνικού πάρκου της Βαυαρίας για την παραχώρηση του ψηφιακού υψομετρικού μοντέλου, που χρησιμοποιήθηκε για τον έλεγχο των αποτελεσμάτων του αλγορίθμου. Τέλος, θέλω να ευχαριστήσω τον κ. Γιώργο Κορδέλα για την συμβολή του σε τεχνικά ζητήματα που προέκυψαν κατά την συγγραφή της παρούσας εργασίας.

Περιεχόμενα

Κατάλογος σχημάτων

Κεφάλαιο 1

Εισαγωγή

Στο κεφάλαιο 1 αναπτύσσεται το πρόβλημα που καλείται να επιλύσει η διπλωματική εργασία και αναφέρονται παρόμοιες υλοποιήσεις που έχουν εφαρμοστεί ήδη από άλλους ερευνητές. Τέλος, αναλύεται η δομή της εργασίας και εξηγείται το περιεχόμενο των κεφαλαίων που ακολουθούν.

1.1 Ορισμός του προβλήματος

Τα γεωχωρικα δεδομένα που συλλέγονται κάθε χρόνο καταλαμβάνουν όλο και μεγαλύτερο μερίδιο των μεγάλων δεδομένων (big data). Η ανάπτυξη αυτή προκύπτει, καθώς η συλλογή γεωχωρικών δεδομένων είναι πλέον εφικτό να πραγματοποιηθεί με πιο απλές μεθόδους, όπως την χρήση drone [2] ή και μέσω φωτογραφιών από τα μέσα κοινωνικής δικτύωσης [3]. Παρότι η πρόσβαση σε γεωχωρικά δεδομένα είναι πιο προσιτή από ποτέ, η αξιοποίηση τους αποτελεί πρόκληση εξαιτίας του όγκου δεδομένων, τον οποίο καλείται να διαχειριστεί οποιαδήποτε εφαρμογή προσπαθεί να τα επεξεργαστεί. Η ανάλυση οπτικού πεδίου αποτελεί μια από τις εφαρμογές που ασχολείται με την επεξεργασία τέτοιου είδους δεδομένων.

Η ανάλυση οπτικού πεδίου είναι η διαδικασία που χρησιμοποιεί το υψομετρικό μοντέλο μιας περιοχής, ώστε να υπολογίσει τις περιοχές του εδάφους που είναι ορατές από ένα σημείο παρατήρησης [4]. Η ανάλυση οπτικού πεδίου χρησιμοποιείται με δύο βασικές παραλλαγές. Στην πρώτη παραλλαγή, εφαρμόζεται σε ένα σημείο, για το οποίο υπολογίζονται τα σημεία στα οποία έχει ορατότητα. Στη δεύτερη παραλλαγή, εφαρμόζεται για το κάθε κελί μιας ολόκληρης περιοχής, ώστε να βρεθεί ο αριθμός των κελιών στα οποία έχει ορατότητα. Η πρώτη μέθοδος χρησιμοποιείται για εφαρμογές που απαιτούν την ορατότητα σε συγκεκριμένο σημείο. Η δεύτερη μέθοδος ονομάζεται συσσωρευτική ανάλυση οπτικού πεδίου και χρησιμοποιείται για εφαρμογές που αναζητούν το σημείο που παρέχει ορατότητα στη μέγιστη δυνατή περιοχή, όπως οι κεραίες τηλεπικοινωνιών [5]. Η ανάλυση οπτικού πεδίου αποτελεί μια διαδικασία εξαιρετικά χρονοβόρα και ταυτόχρονα απαιτητική σε επεξεργαστική ισχύ. Η χρήση της αφορά εφαρμογές αισθητικής φύσης, αλλά και πρακτικές εφαρμογές και χρησιμοποιείται σε ένα μεγάλο εύρος ερευνητικών πεδίων [6].

Πολλές από τις πιο πρακτικές εφαρμογές της ανάλυσης οπτικού πεδίου σχετίζονται με έργα που λαμβάνουν χώρο στο αστικό περιβάλλον. Για παράδειγμα, συχνά εφαρμόζεται στη κατασκευή δρόμων, με στόχο την βελτίωση της ορατότητα των οδηγών ώστε να περιοριστούν τα αυτοκινητιστικά δυστυχήματα [7]. Μια ακόμα εφαρμογή της, αφορά στην τοποθέτηση καμερών ασφαλείας σε κτήρια, ώστε να καλύπτουν το μέγιστο δυνατό οπτικό πεδίο, συνεισφέροντας έτσι στη μείωση της εγκληματικότητας [8]. Σημαντικό ρόλο έχει και στην τοποθέτηση φωτοβολταϊκών, υπολογίζοντας την θέση που προσφέρει την μέγιστη απόδοση, λαμβάνοντας υπόψη την θέση του ηλίου κατά τη διάρκεια της ημέρας [9]. Ακόμα και στην ύπαιθρο υπάρχουν πολλά έργα μεγαλύτερης έκτασης, που επωφελούνται από την ανάλυση οπτικού πεδίου. Ένα από αυτά σχετίζεται με την τοποθέτηση πύργων τηλεπικοινωνιών. Στην περίπτωση αυτή μπορεί να αναδείξει το σημείο που παρέχει ορατότητα στη μέγιστη δυνατή περιοχή, αποφεύγοντας έτσι τα εμπόδια που θα δυσχέραναν την απόδοση μιας κεραίας [10]. Εξίσου χρήσιμη είναι και η εφαρμογή της για την τοποθέτηση πύργων δασοπυρόσβεσης σε σημεία που παρέχουν ανεμπόδιστη θέα σε περιοχές υψηλού κινδύνου για πυρκαγιές, συνεισφέροντας έτσι στην προστασία του δάσους [11]. Επιπλέον, αξίζει να αναφερθεί η χρησιμότητα της ανάλυσης οπτικού πεδίου στον χώρο της αρχαιολογίας, όπου χρησιμοποιείται συχνά, τόσο για την ανεύρεση αρχαίων κατασκευών, όσο και για να εκτιμηθεί η πιθανή της χρήση τους [12]. Τέλος, ο υπολογισμός της ορατότητας κρίνεται απαραίτητος για στρατιωτικές εφαρμογές, εφαρμογές αρχιτεκτονικής, αλλά και για την ανάπτυξη ηλεκτρονικών παιχνιδιών [13]

Εξίσου σημαντικές είναι και οι εφαρμογές αισθητικής φύσης στις οποίες χρησιμοποιείται η ανάλυση οπτικού πεδίου. Η χρήση της μπορεί να αναδείξει, τόσο τα φυσικά τοπία μιας περιοχής, όσο και τα αξιοθέατα της. Η ανάλυση οπτικού πεδίου αναγνωρίζει τοποθεσίες που έχουν ορατότητα σε περιοχές που παρουσιάζουν φυσική ομορφιά. Βασική χρήση της τεχνικής είναι η αναγνώριση κατάλληλων σημείων για την κατασκευή εγκαταστάσεων που θα παρέχουν υπηρεσίες αναψυχής. Μια ακόμα εφαρμογή της είναι στη κατασκευή μονοπατιών που παρέχουν θέα στο πράσινο της περιοχής που διασχίζουν, αναδεικνύοντας έτσι με τον καλύτερο τρόπο το τοπίο στον επισκέπτη [14]. Η ανάλυση οπτικού πεδίου μπορεί συνδυαστεί και με επιπλέον οπτικούς δείκτες, διευκολύνοντας έτσι την αξιοποίηση των περιοχών αυτών [15]. Οι δείκτες αυτοί αφορούν στοιχεία όπως, η μορφή του εδάφους, αλλά και η παρουσία ανθρωπίνων κατασκευών που επηρεάζουν την ομορφιά του τοπίου. Οι ανθρώπινες κατασκευές αποτελούν σημαντικό εμπόδιο, καθώς αλλοιώνουν το τοπίο μειώνοντας έτσι την αισθητική αξία του [16]. Η παρουσία των κατασκευών αυτών μπορεί να εντοπιστεί με την χρήση της ανάλυσης οπτικού πεδίου, ώστε να βρεθούν τοποθεσίες μακριά

από τέτοιες κατασκευές ή να βρεθούν κατάλληλες θέσεις, από τις οποίες η θέα στα τοπία που δεν εμποδίζεται από τις κατασκευές αυτές [17].

Η ανάλυση οπτικού πεδίου διαδραματίζει σημαντικό ρόλο και στην προστασία του περιβάλλοντος, διευκολύνοντας την μελέτη του οικοσυστήματος με πιο πρακτικές εφαρμογές της. Μια εφαρμογή της ανάλυσης οπτικού πεδίου αφορά την εκτίμηση του πληθυσμού των άγριων ζώων μιας περιοχής, χωρίς να απαιτεί την φυσική παρουσία ανθρώπων. Η μέτρηση του πληθυσμού των κοπαδιών πραγματοποιείται ελέγχοντας το ύψος γνωστών μονοπατιών κατά τη διάρκεια της ημέρας και εντοπίζοντας έτσι τον αριθμό τον ζώων που τα διασχίζουν [18]. Εξίσου χρήσιμη είναι όμως, και όσον αφορά τη βλάστηση, καθώς εξετάζει το ύψος και την παρουσία βλάστησης όταν χρησιμοποιηθεί σε συνδυασμό με άλλες τεχνικές που ελέγχουν, εκτός από το ύψος, την ποιότητα του εδάφους [19]. Με αυτόν τον τρόπο απλοποιείται ο έλεγχος και η καταγραφή της κατάστασης των οικοσυστημάτων. Η εφαρμογή αυτή είναι ιδιαίτερα χρήσιμη για τον έλεγχο μεγάλων περιοχών για παράνομη υλοτομία [20].

Ενώ όμως, η ανάλυση οπτικού πεδίου εφαρμόζεται σε πολλούς τομείς, παρουσιάζει ένα σημαντικό περιορισμό. Η πολυπλοκότητα των μετρήσεων που απαιτεί, έχει ως αποτέλεσμα ο χρόνος εκτέλεσης της να είναι ιδιαίτερα μεγάλος, ενώ ταυτόχρονα απαιτεί μεγάλη επεξεργαστική ισχύ. Για να αντιμετωπιστεί ο περιορισμός αυτός, έχουν αναπτυχτεί πολλοί αλγόριθμοι και τεχνικές, με στόχο την επιτάχυνση του αλγορίθμου, οι κυριότεροι από τους οποίους αναφέρονται στο επόμενο υποκεφάλαιο [21].

1.2 Περιπτώσεις παρόμοιων ερευνητικών έργων

Υπάρχουν αρκετές διαφορετικές υλοποιήσεις αλγορίθμων που πραγματοποιούν ανάλυση οπτικού πεδίου με στόχο, είτε την αύξηση της ακρίβειας, είτε την επιτάχυνση της ταχύτητας εκτέλεσης του αλγορίθμου. Παρακάτω αναφέρονται συνοπτικά οι πιο βασικές μέθοδοι και πλεονεκτήματα της καθεμίας. Πιο εκτεταμένη ανάλυση των παρακάτω μεθόδων γίνεται στο κεφάλαιο 2.7.

Ο αλγόριθμος R3, ο όποιος περιγράφηκε από τους Franklin [22], αποτελεί τον πιο απλό αλγόριθμο ανάλυσης οπτικού πεδίου. Ο αλγόριθμος αυτός υπολογίζει την γραμμή ορατότητας του παρατηρητή προς κάθε κελί. Η μέθοδος αυτή οδηγεί σε επαναλήψεις των υπολογισμών της ορατότητας για το κάθε κελί, σπαταλώντας έτσι χρόνο σε υπολογισμούς, που έχουν ήδη πραγματοποιηθεί. Επομένως, ο R3 πετυχαίνει υψηλή ακρίβεια, αλλά υστερεί σημαντικά στο χρόνο εκτέλεσης σε σχέση με τους αλγόριθμους που ακολουθούν.

Ο αλγόριθμος R2 περιγράφτηκε και πάλι από τους Franklin κ.ά. [22], με στόχο να μειώσει την πολυπλοκότητα του αλγορίθμου R3. Για να πετύχει καλύτερη απόδοση ο R2 μειώνει τις επαναλήψεις των υπολογισμών που συμβαίνουν στον R3. Για τον υπολογισμό της ορατότητας, σε αντίθεση με τον R3, υπολογίζει τη γραμμή ορατότητας για τα κελιά στα άκρα της εικόνας. Ωστόσο, παρόλο που η απόδοση βελτιώνεται, εξακολουθούν να γίνονται πολλαπλοί υπολογισμοί για την ορατότητα των ίδιων κελιών για τα κελιά που βρίσκονται κοντά στον παρατηρητή. Ο R2 έχει καλή ακρίβεια και είναι πιο γρήγορος από τον R3, αλλά δεν πετυχαίνει μεγάλη βελτίωση στην ταχύτητα εκτέλεσης.

Ο αλγόριθμος XDraw σχεδιάστηκε από τους Franklin κ.ά. [22] και αποτελεί έναν πιο αποδοτικό αλγόριθμο ανάλυσης οπτικού πεδίου. Ο αλγόριθμος αυτός υπολογίζει τα κελιά περιμετρικά από τον παρατηρητή σε τετράγωνα δαχτυλίδια. Για τον υπολογισμό των κελιών κάθε δαχτυλιδιού χρησιμοποιεί μόνο την μέτρηση που καταγράφεται στο προηγούμενο δαχτυλίδι. Τα ύψη των κελιών, που δεν τέμνονται από την γραμμή ορατότητας στο κέντρο τους, υπολογίζονται από το μέσο όρο των υψών των πλησιέστερων κελιών. Το πρόβλημα που παρουσιάζεται στον XDraw αφορά σε πιθανά λάθη που δημιουργούνται κατά τον υπολογισμό του ύψους των κελιών. Κάθε ανακρίβεια στο ύψος ενός κελιού μεταφέρεται στις μετρήσεις του εξωτερικού δαχτυλιδιού, προκαλώντας απώλεια ακρίβειας στον υπολογισμό ορατότητας. Ο αλγόριθμος αυτός είναι πιο γρήγορος από τον R2, αφού οι μετρήσεις του ύψους για κάθε κελί αποθηκεύονται και ξαναχρησιμοποιούνται για το επόμενο δαχτυλίδι, αποφεύγοντας τις πολλαπλές μετρήσεις για κάθε κελί που συμβαίνουν στους προηγούμενους αλγορίθμους.

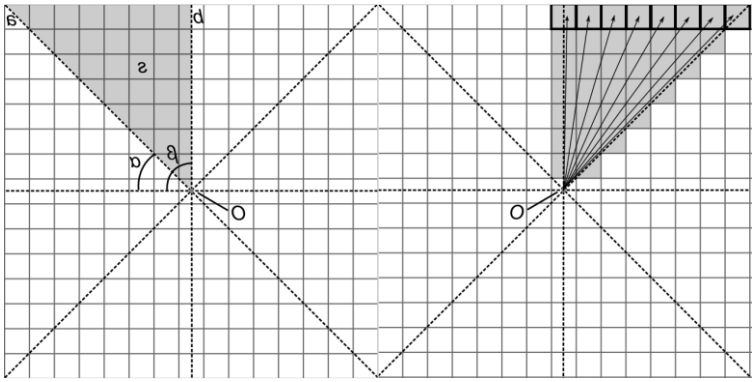
Ο αλγόριθμος που περιγράφηκε από τον Van Kreveld παρέχει γρήγορη ταχύτητα εκτέλεσης με ευστοχία παρόμοια με αυτή του R3 [1]. Στον αλγόριθμο αυτό, όπως και για τον R3, εξετάζεται η γραμμή ορατότητας για κάθε στοιχείο της εικόνας. Ο αλγόριθμος αυτός διαφέρει από τον R3, αφού εκμεταλλεύεται την σειρά εκτέλεσης των γεγονότων, ώστε να αποφύγει τους περιττούς υπολογισμούς που γίνονται στους αλγορίθμους R2 και R3. Για να πετύχει τη μείωση αυτή συγκρατεί σε μια δομή δεδομένων τα στοιχεία κάθε κελιού μέχρι να υπολογιστούν όλες οι γραμμές ορατότητας που τα περιέχουν. Όταν ολοκληρωθεί ο υπολογισμός για όλες τις γραμμές ορατότητας, στις οποίες περιλαμβάνεται το στοιχείο, τότε αφαιρείται από τη δομή. Η δομή αυτή διατηρείται με εισαγωγές, όταν μια γραμμή ορατότητας τέμνει ένα νέο κελί, και εξαγωγές, όταν η γραμμή σταματά να το τέμνει. Κατά συνέπεια παραλείπονται οι περιττοί υπολογισμοί που γίνονται στον αλγόριθμο R2, πετυχαίνοντας έτσι καλή απόδοση και υψηλή ακρίβεια, καθιστώντας τον αλγόριθμο του Van Kreveld έναν από τους πιο αποτελεσματικούς αλγορίθμους ανάλυσης οπτικού πεδίου.

Στους παραπάνω αλγόριθμους, ο υπολογισμός του ύψους των κελιών, που δεν τέμνονται από την γραμμή ορατότητας στο κέντρο τους, επηρεάζει δραστικά τόσο την ακρίβεια, όσο και την ταχύτητα τους. Αυτό συμβαίνει καθώς το ύψος του κάθε κελιού που παρέχει το υψομετρικό μοντέλο αφορά το κέντρο του κελιού. Ως αποτέλεσμα, όταν μια γραμμή δεν περνά από το κέντρο του κελιού, το ύψος υπολογίζεται προσεγγιστικά. Η προσέγγιση αυτή εφαρμόζεται με αρκετές παραλλαγές, και επηρεάζει την ακρίβεια των αποτελεσμάτων των αλγορίθμων. Οι αλγόριθμοι μπορούν να πετύχουν καλύτερη ακρίβεια χρησιμοποιώντας την τεχνική της παρεμβολής για κάθε στοιχείο με μειονέκτημα όμως τη σημαντική επιβάρυνση στον χρόνο εκτέλεσης. Εναλλακτικά, μερικές υλοποιήσεις δεν προσεγγίζουν το ύψος, αλλά θεωρούν το ύψος σε οποιοδήποτε σημείο του κελιού ίδιο με αυτό του κέντρου του κελιού. Η τελευταία προσέγγιση μπορεί να εφαρμοστεί με επιτυχία σε περιπτώσεις που το κάθε κελί αντιπροσωπεύει πολύ μικρή έκταση. Υπάρχουν αρκετές ακόμα προσεγγίσεις για τον υπολογισμό, μερικές από τις οποίες χρησιμοποιούν το μέσο ύψος των κοντινότερων στοιχείων ή το μέγιστο ύψος από τα κοντινότερα στοιχεία, ανάλογα με της ανάγκες της κάθε εφαρμογής.

Για τις παραπάνω μεθόδους έχουν υλοποιηθεί αρκετές παραλλαγές, με στόχο την επιτάχυνση του χρόνου εκτέλεσης των αλγορίθμων. Μια μέθοδος που έχει χρησιμοποιηθεί βασίζεται στην χρήση της κάρτας γραφικών για τους υπολογισμούς ορατότητας, με αποτέλεσμα την επιτάχυνση της απόδοσης [23]. Μια άλλη παραλλαγή εφαρμόζει παράλληλη επεξεργασία χρησιμοποιώντας πολλούς επεξεργαστές, ώστε να αυξηθεί η επεξεργαστική ισχύς και να βελτιωθεί η ταχύτητα του αλγορίθμου [24]. Οι τεχνικές αυτές έχουν ποικίλους περιορισμούς, και καθεμία από αυτές είναι κατάλληλη για διαφορετικούς αλγόριθμους. Ο R3 και ο R2 έχουν παραλληλοποιηθεί με την χρήση της κάρτας γραφικών, αφού οι υπολογισμοί τους είναι ανεξάρτητοι μεταξύ τους και αποτελούν ιδανικές περιπτώσεις για την μέθοδο αυτή. Αντιθέτως, ο αλγόριθμος του Van Kreveld, λόγω της δομής του, είναι κατάλληλος μόνο για παραλληλοποίηση με την χρήση επεξεργαστών, αφού τα γεγονότα του εξαρτώνται από την σειρά εκτέλεσης τους [25].

Υπάρχουν αρκετές διαφορετικές προσεγγίσεις, που χρησιμοποιούν τους παραπάνω αλγορίθμους σε συνδυασμό με τις τεχνικές που αναφέρθηκαν. Παρακάτω αναφέρονται μερικές από τις προσεγγίσεις αυτές, και διευκρινίζονται οι διαφορές που παρουσιάζουν. Οι προσεγγίσεις, που αναφέρονται, έχουν ως βάση τον αλγόριθμο του Van Kreveld, αφού αποτελεί έναν από τους πιο δημοφιλείς αλγόριθμους για τέτοιου είδους υλοποιήσεις, και χρησιμοποιήθηκε ως βάση στον αλγόριθμο που εξελίχθηκε στην παρούσα διπλωματική.

Ο αλγόριθμος ανάλυσης οπτικού πεδίου που υλοποιήθηκε από τους Ferreira κ.ά. [26] χρησιμοποιεί τον αλγόριθμο του Van Kreveld, σε συνδυασμό με την τεχνική της παράλληλης επεξεργασίας.



Σχήμα 1 Διαχωρισμός των τομέων που χρησιμοποιούνται για την παράλληλη εκτέλεση του αλγόριθμο των Ferreira κ.ά.[3]

Όπως φαίνεται στο σχήμα 1, ο υπολογισμός της ορατότητας διαχωρίζεται σε κομμάτια, με βάση τη θέση του παρατηρητή, πετυχαίνοντας με αυτόν τον τρόπο βελτίωση στην ταχύτητα εκτέλεσης του αλγορίθμου. Ο αλγόριθμος αυτός, παρόλο που πετυχαίνει βελτίωση της απόδοσης, δεν υπολογίζει με ακριβή τρόπο το ύψος των κελιών και κατά συνέπεια παρουσιάζει έλλειψη στην ακρίβεια του.

Ένας ακόμα, αλγόριθμος ανάλυσης οπτικού πεδίου, υλοποιήθηκε από τον Gokhan Yilmaz [21]. Ο αλγόριθμος του Gokhan Yilmaz ξεχωρίζει από εκείνο των Ferreira, καθώς για τις μετρήσεις των γεγονότων του εφαρμόζει παράλληλη επεξεργασία με την χρήση της κάρτας γραφικών. Ο αλγόριθμος αυτός, όπως και ο προηγούμενος, δεν υπολογίζει με ακρίβεια την τιμή του ύψους των κελιών, με αποτέλεσμα τη μείωση της ακρίβειας του αλγορίθμου. Επιπλέον, απαιτεί τη χρήση της κάρτα γραφικών για μέρος των υπολογισμών, που για πολλούς χρήστες δεν είναι διαθέσιμη λόγω του κόστους της.

Τέλος, ο αλγόριθμος ανάλυσης οπτικού πεδίου, που υλοποιήθηκε από του Haverkort, H.J. ,Toma, L. και Zhuang, Yi [27], εφαρμόζει τεχνικές διαχείρισης μνήμης. Στόχος των τεχνικών αυτών είναι να δεσμευτεί χώρος στην εσωτερική μνήμη του υπολογιστή, καθώς παρέχει γρηγορότερο χρόνο προσπέλασης των μεταβλητών. Αποτέλεσμα της τεχνικής αυτής είναι να επιταχυνθούν διαδικασίες, όπως η ταξινόμηση και η σάρωση, οι οποίες αποτελούν δομικό στοιχείο του αλγορίθμου. Στον αλγόριθμο αυτόν, δεν χρησιμοποιούνται τεχνικές παραλληλοποίησης, αλλά διαχωρίζεται το πρόβλημα σε κομμάτια που μπορεί να διαχειριστεί η εσωτερική μνήμη του υπολογιστή. Στην υλοποίηση αυτή, αρχικά υπολογίζεται η μνήμη που

[3] Πηγή του σχήματος είναι η δημοσίευση [26]

είναι διαθέσιμη στον υπολογιστή, και διαχωρίζεται ο υπολογισμός ορατότητας σε κομμάτια που μπορούν να χωρέσουν στην μνήμη του. Ο αλγόριθμος αυτός αποτελεί τη βάση του αλγόριθμου που εκτελείται στο *Geographic Resources Analysis Support System* (GRASS), το οποίο αποτελεί μια πλατφόρμα λογισμικού που αναλύει γεωχωρικά δεδομένα. Ο αλγόριθμος που είναι διαθέσιμος στο GRASS περιλαμβάνει, επιπλέον, πιο ακριβή υπολογισμό του ύψους των κελιών με την χρήση της τεχνικής της παρεμβολής. Ο αλγόριθμος αυτός είναι, τόσο ακριβής, όσο και γρήγορος, αλλά δεν εφαρμόζει τεχνικές παραλληλοποίησης, οι οποίες θα μπορούσαν να βελτιώσουν περαιτέρω το χρόνο εκτέλεσης του.

1.3 Κίνητρα και Στόχοι Υλοποίησης

Η ανάλυση οπτικού πεδίου παρέχει απαραίτητες υπηρεσίες σε πολλές εφαρμογές που αφορούν τόσο στην καθημερινότητα, όσο και σε έργα κοινής ωφέλειας. Για αυτό τον λόγο περιγράφονται σε αρκετές δημοσιεύσεις αλγόριθμοι που πραγματοποιούν ανάλυση οπτικού πεδίου. Ενώ όμως υπάρχουν πολλοί μέθοδοι υλοποίησης, δεν υπάρχει ανάλογη διαθεσιμότητα ελεύθερου κώδικα, ειδικά για αλγορίθμους που εφαρμόζουν τεχνικές παράλληλης επεξεργασίας. Επομένως, πρωταρχικό κίνητρο της εργασίας είναι η δημιουργία ενός λογισμικού ανοιχτού κώδικα που θα διευκολύνει την περαιτέρω βελτίωση του αλγορίθμου, ενώ ταυτόχρονα θα επιτρέπει ελεύθερη χρήση του για οποιαδήποτε εφαρμογή.

Ο στόχος του αλγόριθμου που δημιουργήθηκε είναι να συνδυάσει υψηλή ακρίβεια και γρήγορο χρόνο εκτέλεσης. Για την επίτευξη της επιτάχυνσης του χρόνου εκτέλεσης, ο αλγόριθμος εφαρμόζει παραλληλοποίηση με την χρήση της Διεπαφής Μεταβίβασης Μηνυμάτων (Message Passing Interface - MPI) [28], η οποία ανήκει στην παραλληλοποίηση με την χρήση πολλαπλών επεξεργαστών. Ο αλγόριθμος που επιλέχτηκε είναι αυτός που προτάθηκε από τον Van Kreveld, αφού παρέχει γρήγορο χρόνο εκτέλεσης και η δομή του είναι συμβατή με την τεχνική αυτή. Επιπρόσθετα, στον αλγόριθμο εφαρμόζεται η τεχνική της παρεμβολής για τον υπολογισμό των κελιών όταν δεν τέμνονται από τη γραμμή ορατότητα στο κέντρο τους, βελτιώνοντας έτσι σημαντικά την ακρίβεια του αλγορίθμου. Για την εφαρμογή και των δυο τεχνικών απαιτούνται τροποποιήσεις στον αλγόριθμο του Van Kreveld, οι οποίες τον διαχωρίζουν από τα έργα που έχουν ήδη υλοποιηθεί. Οι τροποποιήσεις αυτές αλλάζουν δραστικά την δομή του αλγορίθμου, ώστε να γίνει εφικτή η εφαρμογή τον παραλλαγών αυτών. Ο ακριβής τρόπος λειτουργίας του αλγορίθμου και των παραλλαγών του περιγράφονται λεπτομερώς στο κεφάλαιο 3.

1.4 Διάρθρωση κειμένου

Το κείμενο αποτελείται από τέσσερα κεφάλαια, το καθένα από τα οποία αναλύει μια πτυχή του αλγόριθμου που υλοποιήθηκε. Στα επιμέρους κεφάλαια παρουσιάζεται και αναλύεται η μεθοδολογία που εφαρμόστηκε, αναφέρεται το λογισμικό που χρησιμοποιήθηκε, ενώ στο τελευταίο κεφάλαιο εξετάζονται προβλήματα που προέκυψαν και η αντιμετώπιση τους.

Στο παρόν κεφάλαιο, αναφέρονται μερικές από τις εφαρμογές του αλγορίθμου που υλοποιήθηκε, παρουσιάζονται παρόμοια ερευνητικά έργα και εξηγούνται οι στόχοι που προσπαθεί ο αλγόριθμος να πετύχει.

Στο δεύτερο κεφάλαιο, αναλύεται το θεωρητικό υπόβαθρο και εξηγούνται οι τεχνικές ορολογίες και οι βασικές έννοιες, που είναι απαραίτητες για την περιγραφή του αλγορίθμου. Το κεφάλαιο αυτό είναι απαραίτητο για την κατανόηση των μεθόδων που χρησιμοποιήθηκαν στα παρακάτω κεφάλαια.

Στο τρίτο κεφάλαιο, παρουσιάζεται το λογισμικό της εργασίας, αναλύονται οι μέθοδοι που χρησιμοποιήθηκαν και εξηγούνται οι παραλλαγές που εφαρμόστηκαν στον αλγόριθμο, ώστε να επιτευχθεί η παράλληλη εκτέλεση του. Το κεφάλαιο αυτό αναλύει κάθε κομμάτι του κώδικα, ενώ δίνονται και γραφικά παραδείγματα για την κατανόηση των μεθόδων που χρησιμοποιήθηκαν.

Το τέταρτο κεφάλαιο αποτελεί τον επίλογο της εργασίας. Στο κεφάλαιο αυτό παρουσιάζονται τα αποτελέσματα του αλγόριθμου, αναφέρονται τα προβλήματα που αντιμετωπίστηκαν και οι λύσεις του, και τέλος εξετάζονται πιθανές βελτιώσεις για μελλοντικές επεκτάσεις του αλγορίθμου

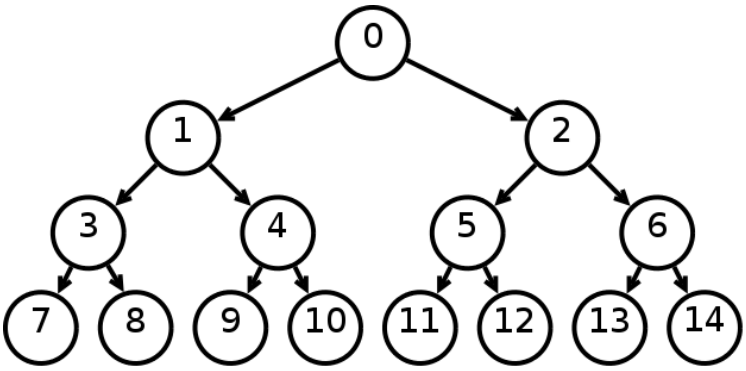
Κεφάλαιο 2

Θεωρητικό υπόβαθρο

Στο παρακάτω κεφάλαιο παρουσιάζονται όλα τα εργαλεία που χρησιμοποιήθηκαν στην υλοποίηση της παρούσας πτυχιακής εργασίας. Αναφέρονται το λογισμικό και οι τεχνικές που χρησιμοποιήθηκαν, και εξηγούνται οι λέξεις κλειδιά, ώστε να μπορεί ο αναγνώστης να κατανοήσει καλύτερα τα κεφάλαια που θα ακολουθήσουν.

2.1 Δεντρική δομή

Στη θεωρία γραφημάτων, ως δεντρική δομή ή δέντρο ορίζεται ένα συνεκτικό και μη κατευθυνόμενο γράφημα το οποίο δεν περιέχει κύκλους [29]. Η δεντρική δομή είναι ένας τρόπος να αναπαρασταθεί γραφικά η ιεραρχία μιας δομής. Είναι η δομή δεδομένων που αποτελείται από ένα σύνολο κόμβων που συνδέονται με ακμές. Σε κάθε μία πεπερασμένη δενδρική δομή υπάρχει ο ανώτατος αρχικός κόμβος, από τον οποίο ξεκινάει η διακλάδωση, και ονομάζεται ρίζα του δέντρου. Οι κόμβοι που δεν διακλαδώνονται περαιτέρω ονομάζονται φύλλα του δέντρου. Οι κόμβοι μεταξύ τους έχουν σχέσεις γονιού - παιδιού. Κάθε παιδί έχει μόνον ένα γονιό. Μια δομή δέντρου έχει την μορφή που παρουσιάζεται στο σχήμα 2.



Σχήμα 2 Παράδειγμα ισορροπημένου δυαδικού δέντρου[4].

Μια υποκατηγορία δέντρων αποτελούν τα δυαδικά δέντρα, τα οποία έχουν μέχρι δυο παιδιά ανά κόμβο, και τα ισορροπημένα δέντρα, που τα φύλλα τους έχουν μέγιστη διαφορά βάθους έναν κόμβο.

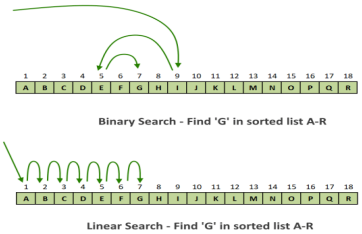
[4] Πηγή του σχήματος είναι η ιστοσελίδα https://math.stackexchange.com/questions/2856956/binary-tree-equation-describing-path. Η επίσκεψη έγινε στις 11/5/2020

2.2 Αλγόριθμος αναζήτησης

Ένας αλγόριθμος αναζήτησης είναι η διαδικασία της οποίας στόχος είναι η εύρεση ενός αντικειμένου με συγκεκριμένες ιδιότητες, μεταξύ μιας συλλογής αντικειμένων, που βρίσκεται αποθηκευμένη σε μια δομή δεδομένων. Το πιο απλό είδος αλγορίθμου αναζήτησης είναι η γραμμική ή σειριακή αναζήτηση. Η γραμμική αναζήτηση έχει ως είσοδο το αντικείμενο που αναζητείται, το οποίο στη συνέχεια συγκρίνεται με όλα τα στοιχεία της λίστας, και δίνει ως έξοδο την τιμή true, αν το στοιχείο βρέθηκε στη λίστα και την τιμή false, αν δεν βρέθηκε [30].

Η δυαδική αναζήτηση είναι μια πιο γρήγορη μέθοδος αναζήτησης, που έχει την ίδια είσοδο και έξοδο με την γραμμική αναζήτηση, και είναι ιδιαίτερα χρήσιμη όταν εξετάζεται μεγάλος όγκος δεδομένων. Βασική προϋπόθεση της δυαδικής αναζήτησης είναι η λίστα στην οποία εφαρμόζεται, να είναι ταξινομημένη. Στη δυαδική αναζήτηση συγκρίνεται το αντικείμενο που αναζητείται με το μεσαίο στοιχείο της λίστας. Από τη σύγκριση αυτή, αν το μεσαίο στοιχείο είναι ίδιο με το στοιχείο που αναζητείται, τότε τερματίζεται η αναζήτηση και επιστρέφει τιμή true. Αν το μεσαίο στοιχείο είναι μεγαλύτερο από το στοιχείο που αναζητείται και η ταξινόμηση έχει γίνει με αύξουσα σειρά, τότε επαναλαμβάνεται η διαδικασία για το μεσαίο στοιχείο του αριστερού μισού της λίστας. Αν το μεσαίο στοιχείο είναι μικρότερο από το στοιχείο που αναζητείται, τότε γίνονται οι ίδιες ενέργειες στο δεξί μισό της λίστας. Αν η διαδικασία αυτή καταλήξει σε μια υπολίστα, που αποτελείται από ένα στοιχείο, και το στοιχείο αυτό δεν είναι αυτό που αναζητείται, τότε επιστρέφεται η τιμή false [30].

Συγκρίνοντας τις δυο μεθόδους η δυαδική αναζήτηση παρουσιάζει πολυπλοκότητα *O(logn)*, ενώ η σειριακή αναζήτηση έχει πολυπλοκότητα *O(n)*. Ακόμα και στη χειρότερη περίπτωση σε μια λίστα με *n* στοιχεία η σειριακή αναζήτηση απαιτεί *n* επαναλήψεις, ενώ η δυαδική αναζήτηση απαιτεί το πολύ *n/2+1* επαναλήψεις, καθιστώντας την πολύ πιο αποδοτική ειδικά όταν εφαρμόζεται σε μεγάλες λίστες.



Σχήμα 3 Αναζήτηση του γράμματος G σε ταξινομημένη λίστα με δυαδική αναζήτηση και με σειριακή αναζήτηση[5].

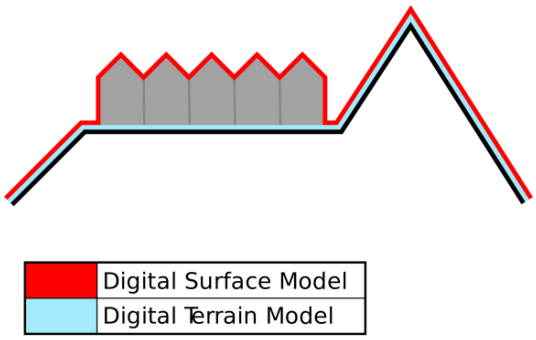
[5] Πηγή του σχήματος είναι η ιστοσελίδα https://algorithms.tutorialhorizon.com/linear-search-vs-binary-search/. Η επίσκεψη έγινε στις 11/5/2020

2.3 Ψηφιακό Υψομετρικό Μοντέλο

Ως Ψηφιακό Υψομετρικό Μοντέλο ή Digital Elevation Model (DEM) ορίζεται ένα σύνολο ψηφιακών καταγραφών που απεικονίζουν τα ύψη μιας επιφάνειας. Όταν χρησιμοποιείται ο όρος αυτός, πρέπει να οριστεί και για ποια επιφάνεια έχουν καταγραφεί οι τιμές [31]. Δυο είδη DEM είναι το Ψηφιακό Μοντέλο Εδάφους ή Digital Terrain Model (DTM) και το Ψηφιακό Μοντέλο Επιφανείας ή Digital Surface Model (DSM)

Ως DTM ορίζεται ένα σύνολο ψηφιακών καταγραφών που απεικονίζουν τη γήινη επιφάνεια. Είναι δηλαδή ένα ψηφιακό υψομετρικό μοντέλο της γυμνής γήινης επιφανείας, χωρίς εμπόδια όπως κτίρια και δέντρα [31].

Ως DSM ορίζεται ως ένα σύνολο ψηφιακών καταγραφών, που απεικονίζουν τα ύψη των αντικειμένων που καλύπτουν τη γήινη επιφάνεια (βλάστηση, σπίτια, δρόμοι κ.λπ.) [31].Στο σχήμα 4 παρουσιάζεται και σχηματικά τι περιλαμβάνει το DTM και το DSM.



Σχήμα 4 Γραφική απεικόνιση των περιοχών του εδάφους που περιλαμβάνονται στον υπολογισμό του DTM και του DSM[6].

Ένα DEM αποθηκεύεται συνήθως σε δεδομένα τύπου κανάβου (raster). Ένα raster είναι ένα πλέγμα από κελιά όπου κάθε κελί αντιπροσωπεύει ένα κομμάτι της περιοχής που περιλαμβάνεται στο DEM. Κάθε ένα από τα κελιά έχει ως τιμή το υψόμετρο από την περιοχή που καλύπτει. Το μέγεθος των κελιών καθορίζει την ακρίβεια του υψόμετρου για την περιοχή· με μικρότερα τετράγωνα αυξάνεται η ακρίβεια της μέτρησης (αν και η πυκνότητα των πραγματικών τιμών μέτρησης το επιτρέπει), αλλά μεγαλώνει το μέγεθος του αρχείου [32].

Στην ανάλυση οπτικού πεδίου, η χρήση του υψομετρικού μοντέλου αποτελεί σημαντικό δομικό στοιχείο για τον υπολογισμό της ορατότητας. Το DSM περιλαμβάνει το ύψος για όλη την περιοχή που εξετάζεται καθώς περιλαμβάνει το ύψος και των εμποδίων που επηρεάζουν

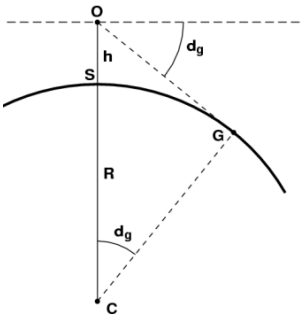
[6] Πηγή του σχήματος είναι η ιστοσελίδα https://en.wikipedia.org/wiki/Digital_elevation_model . Η επίσκεψη έγινε στις 11/5/2020

την ορατότητα, ενώ το DTM χρησιμοποιείται ώστε να υπολογιστεί το ύψος του παρατηρητή του οποίου το ύψος είναι ανεξάρτητο από τα εμπόδια του εδάφους που περιέχονται στο DSM.

2.4 Ορίζοντας

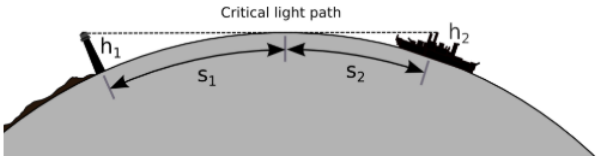
Ορίζοντας είναι η νοητή γραμμή που διαχωρίζει το έδαφος από τον ουρανό [33]. Η απόσταση του ορίζοντα από τον παρατηρητή διαφέρει ανάλογα με το ύψος του, καθώς όσο ψηλότερα βρίσκεται ο παρατηρητής, τόσο απομακρύνεται ο ορίζοντας. Η απόσταση μέχρι τον ορίζοντα υπολογίζεται από τον παρακάτω τύπο, ο οποίος προέκυψε από την εφαρμογή του πυθαγόρειου θεωρήματος, όπως παρουσιάζεται στο σχήμα 5:

$$\text{Απόσταση μεχρι τον ορίζοντα} = \sqrt{2 \times \textit{περίμετρος της γής} \times \text{ύψος του παρατηρητή}} \qquad (1)$$



Σχήμα 5 Μέθοδος υπολογισμού της απόστασης από τον ορίζοντα με την χρήση πυθαγόρειου θεωρήματος όπου *GO* η απόσταση μέχρι τον ορίζοντα, *h* το ύψος του παρατηρητή και *R* η περίμετρος της γης.[7]

Εφαρμογές που πραγματοποιούν έλεγχο ορατότητας, χρησιμοποιούν τον ορίζοντα, αφού τα σημεία που βρίσκονται πιο μακριά από τον ορίζοντα δεν είναι ορατά στον παρατηρητή αν δεν ξεπερνούν το ύψος του, όπως φαίνεται στο σχήμα 6 [33].



Σχήμα 6 Παράδειγμα της επιρροής στην ορατότητα του ορίζοντα στα σημεία που βρίσκονται μακριά από τον παρατηρητή[8].

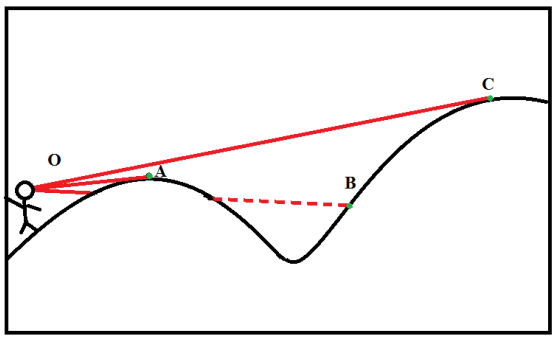
Με αυτό τον τρόπο, οι αλγόριθμοι που πραγματοποιούν ανάλυση οπτικού πεδίου, μπορούν να παραλείψουν τους υπολογισμούς για την ορατότητα τον σημείων που κρύβει ο ορίζοντας [33].

[7] Πηγή του σχήματος είναι η ιστοσελίδα https://aty.sdsu.edu/explain/atmos_refr/horizon.html. Η επίσκεψη έγινε στις 11/5/2020

[8] Πηγή του σχήματος είναι η ιστοσελίδα https://www.johngiovannis.com/content/distance-horizon. Η επίσκεψη έγινε στις 11/5/2020

2.5 Γραμμή Ορατότητας

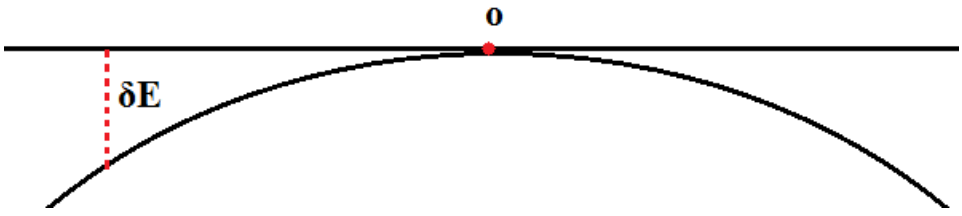
Ως γραμμή ορατότητας ή Line Of Sight (LOS) ορίζεται ως η νοητή γραμμή που ενώνει δυο σημεία, με στόχο τον υπολογισμό της ορατότητας μεταξύ των σημείων αυτών. Όπως φαίνεται στο σχήμα 7. Αν οριστεί ένας παρατηρητής στο σημείο Ο, ο παρατηρητής θα έχει ορατότητα στο σημείο Α, όταν η LOS δεν συναντά κάποιο σημείο με μεγαλύτερη κλίση ανάμεσα στο Ο και το Α. Για τα σημεία που εξετάζονται δίνεται δυαδική τιμή 1, αν υπάρχει ορατότητα, και 0 αν δεν υπάρχει ορατότητα [34]. Στις περιπτώσεις όπου ο παρατηρητής δεν έχει ορατότητα είναι εφικτό να μετρηθεί πόσο πρέπει να αυξηθεί το ύψος του, ώστε να παρέχει ορατότητα στο σημείο που εξετάζεται [32]. Στον υπολογισμό της LOS χρησιμοποιείται η κλίση και όχι το ύψος των κελιών, καθώς βασικός παράγοντας είναι και η απόσταση του εξεταζόμενου σημείου από τον παρατηρητή.



Σχήμα 7 Παράδειγμα γραμμής ορατότητας, όταν ο παρατηρητής βρίσκεται στο σημείο Ο.

Κάτι ακόμα που πρέπει να ληφθεί υπόψη, είναι η καμπυλότητα της επιφάνειας της γης. Όπως παρατηρείται και στο σχήμα 8, όσο απομακρύνεται το σημείο που εξετάζεται από τον παρατηρητή, τόσο μειώνεται το αντιληπτό ύψος του. Η μείωση αυτή τη υπολογίζεται από τον παρακάτω τύπο [32]:

$$\delta E = \frac{(\text{απόσταση από τον παρατηρητή})^2}{2\times(\text{περίμετρος της γης})} \quad (2)$$



Σχήμα 8 Παράδειγμα της επιρροής της καμπυλότητας της γης στο ορατό ύψος ενός σημείου σε σχέση με την απόσταση από τον παρατηρητή.

Τέλος, λόγω της καμπυλότητας της γης, δημιουργείται ένας επιπλέον περιορισμός στην μέγιστη απόσταση που υπάρχει ορατότητα. Ο παρατηρητής δεν μπορεί να παρατηρήσει σημεία που βρίσκονται κάτω από τον ορίζοντα. Το ύψος του ορίζοντα (ύψος ορίζοντα_q) για

παρατηρητή στο σημείο q με ύψος ($\upsilon\psi\acute{o}\mu\varepsilon\tau\rho o_q$) και σημείο παρατήρησης το u με ύψος ($\upsilon\psi\acute{o}\mu\varepsilon\tau\rho o_u$) όταν απέχουν μεταξύ τους απόσταση ($\alpha\pi\acute{o}\sigma\tau\alpha\sigma\eta_{uq}$)είναι [27]:

$$\Upsilon\psi os\ or\acute{\iota}\zeta ov\tau\alpha_q = archtan\frac{\upsilon\psi\acute{o}\mu\varepsilon\tau\rho o_q - \upsilon\psi\acute{o}\mu\varepsilon\tau\rho o_u}{\alpha\pi\acute{o}\sigma\tau\alpha\sigma\eta_{uq}} \quad (3)$$

2.6 Παρεμβολή

Η παρεμβολή είναι μια τεχνική που ανήκει στο πεδίο της αριθμητικής ανάλυσης, και χρησιμοποιείται για να εκτιμηθεί προσεγγιστικά η τιμή ενός σημείου, με την χρήση των τιμών των γειτονικών περιοχών. Ο αριθμός των σημείων που χρησιμοποιούνται, βελτιώνει την προσέγγιση, αλλά κάνει τον υπολογισμό πιο σύνθετο. Για παράδειγμα, η παρεμβολή που χρησιμοποιεί δυο σημεία ονομάζεται και γραμμική παρεμβολή και προκύπτει από τον παρακάτω τύπο, όπου χρησιμοποιούνται τα σημεία *(x1, y1, h1)* και *(x2, y2, h2)* για να εκτιμηθεί η τιμή για το ύψος *h* στο σημείο *(x, y)* [35]:

$$h = |x - y1| * h2 + |x - y2| * h1 \quad (4)$$

Η τεχνική αυτή χρησιμοποιείται συχνά στην ανάλυση οπτικού πεδίου, καθώς η τιμές που δίνονται από το DEM, αφορούν το κέντρο του κάθε κελιού, ενώ συνήθως δεν εξετάζεται το κάθε κελί στο κέντρο του.

2.7 Ανάλυση οπτικού πεδίου

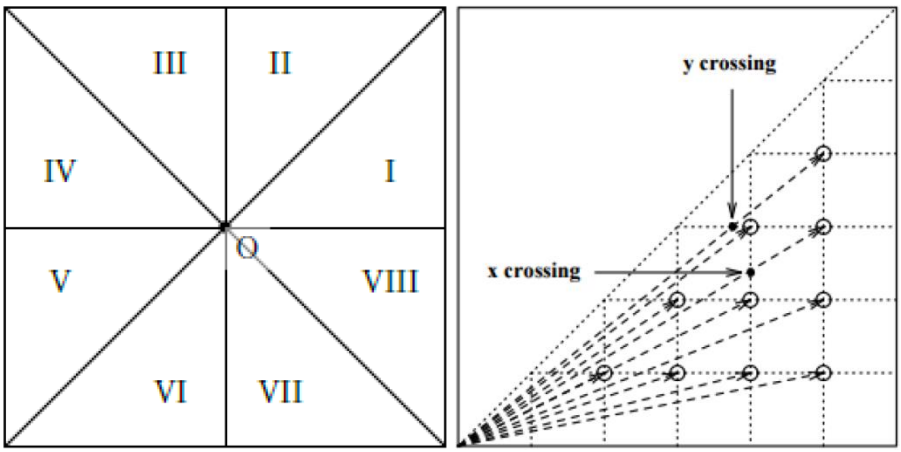
Ανάλυση οπτικού πεδίου (viewshed analysis) είναι η διαδικασία στην οποία υπολογίζονται για ένα σημείο, οι περιοχές στις οποίες το συγκεκριμένο σημείο έχει ορατότητα. Αρχικά ορίζεται σε ποιο κελί του DEM, βρίσκεται ο παρατηρητής και εξετάζεται η γραμμή ορατότητας από το κελί του παρατηρητή προς όλα τα υπόλοιπα κελιά, και εντοπίζονται τα κελιά στα οποία έχει ο παρατηρητής ορατότητα [4]. Στη συνέχεια, τα κελιά στα οποία υπάρχει ορατότητα παίρνουν την τιμή ένα, και τα υπόλοιπα την τιμή μηδέν. Το αποτέλεσμα είναι να δημιουργείται ένας χάρτης ορατότητας, στον οποίο θα φαίνονται μόνο τα κελιά που είναι ορατά από τον παρατηρητή. Σε περίπτωση που ένα σημείο δεν είναι ορατό από τον παρατηρητή, μπορεί να υπολογιστεί το απαραίτητο ύψος του σημείου, ώστε να έχει ο παρατηρητής ορατότητα σε αυτό [1]. Μια επιπλέον βασική παραλλαγή της αποτελεί η συσσωρευτική ανάλυση οπτικού πεδίου, στην οποία εκτελείται ανάλυση οπτικού πεδίου για κάθε στοιχείο της εικόνας που εξετάζεται [5]. Η μέθοδος αυτή χρησιμοποιείται για να εντοπιστούν τοποθεσίες που παρέχουν ορατότητα στα περισσότερα δυνατόν σημεία και χρησιμοποιείται σε εφαρμογές όπως η τοποθέτηση πύργων τηλεπικοινωνιών [36]. Σημαντικό είναι να δοθεί προσοχή στο ύψος

του παρατηρητή, καθώς το ύψος του δεν είναι ίδιο με την τιμή του κελιού στο οποίο βρίσκεται, αλλά υπολογίζεται ως το άθροισμα της τιμής του κελιού και του ύψους του παρατηρητή. Η ανάλυση οπτικού πεδίου αποτελεί μια διαδικασία χρονοβόρα, αλλά και εξαιρετικά απαιτητική σε επεξεργαστική ισχύ, ιδιαίτερα όταν εφαρμόζεται επανειλημμένα, όπως στην περίπτωση της συσσωρευτικής ανάλυσης οπτικού πεδίου. Η ταχύτητα και η ακρίβεια του αλγορίθμου εξαρτώνται από το μέγεθος των κελιών του DEM καθώς, όσο μεγαλύτερα είναι τα κελιά, τόσο μικρότερη είναι η ακρίβεια του αποτελέσματος και τόσο γρηγορότερη είναι η ταχύτητα του αλγορίθμου [32].

Σχήμα 9 Απεικόνιση ανάλυσης οπτικού πεδίου, στην οποία η θέση του παρατηρητή βρίσκεται στο σημείο που δείχνει το κίτρινο βέλος, τα ορατά σημεία φαίνονται με πράσινο, και τα μη ορατά με κόκκινο.

2.7.1 Αλγόριθμος R3

Ο αλγόριθμος του R3 είναι ένας αλγόριθμος ανάλυσης οπτικού πεδίου που περιγράφτηκε από τους Franklin κ.ά. [22]. Ο R3 είναι ένας άμεσος τρόπος υπολογισμού, που πετυχαίνει υψηλή ακρίβεια με συνέπεια να επιβραδύνεται ο χρόνος εκτέλεσης του αλγορίθμου. Ο αλγόριθμος υπολογίζει την γραμμή ορατότητας του παρατηρητή προς κάθε κελί. Για κάθε κελί που δεν τέμνεται από την γραμμή ορατότητας στο κέντρο του κελιού, το ύψος υπολογίζεται με βάση το σημείο στο οποίο τέμνεται.



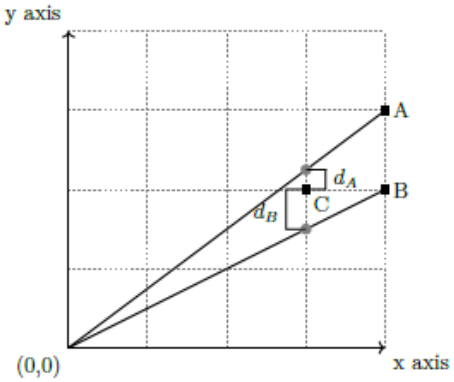
Σχήμα 10 Στα αριστερά απεικονίζονται οι τομείς κάθε κελιού που χρησιμοποιούνται για τον υπολογισμό των crossings και δεξιά τα *x* και *y* crossings[9].

Όπως φαίνεται και στο σχήμα 10, αν το κελί τέμνεται στις περιοχές I, IV, V ή VIII τότε χρησιμοποιείται η τιμή του x-crossing, όπως φαίνεται δεξιά στο σχήμα 10, ενώ για τις υπόλοιπες περιοχές η τιμή του y-crossing. Η τιμή για το x-crossing ή y-crossing προσεγγίζεται από το μέσο όρο του ύψος των δύο κοντινότερων σημείων με την χρήση της τεχνικής της παρεμβολής. Επομένως, ο R3 είναι ο μοναδικός αλγόριθμος που δίνει αποτελέσματα ακριβείας, οι υπόλοιποι αλγόριθμοι που ακολουθούν είναι προσεγγιστικοί.

2.7.2 Αλγόριθμος R2

Ο αλγόριθμος R2 περιγράφτηκε και πάλι από τους Franklin κ.ά. [22], με στόχο να μειώσει την πολυπλοκότητα του αλγορίθμου R3. Ο R2 δεν υπολογίζει την γραμμή ορατότητας για κάθε κελί όπως ο R3, αντιθέτως υπολογίζει μόνο τις γραμμές ορατότητας που αρχίζουν από τον παρατηρητή και καταλήγουν στα άκρα της εικόνας, όπως φαίνεται και στο παρακάτω σχήμα (Σχήμα 11) για τις γραμμές ορατότητας που τελειώνουν στα σημεία A και B. Με αυτό τον τρόπο, ο αλγόριθμος αποφεύγει τους διπλότυπους υπολογισμούς για τα κελιά που βρίσκονται κοντά στα άκρα της εικόνας, βελτιώνοντας την απόδοση του σε σχέση με τον R3.

[9] Πηγή του σχήματος είναι η δημοσίευση [43]

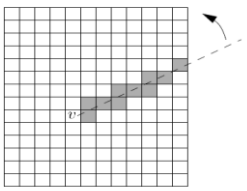


Σχήμα 11 Παράδειγμα υπολογισμού του ύψους για τον αλγόριθμο R2 με την χρήση crossings για τις γραμμές ορατότητας *A* και *B*[10].

Ο υπολογισμός του ύψους, για κάθε κελί που η γραμμή ορατότητας δεν τέμνει στο κέντρο του, πραγματοποιείται όταν συμβαίνει x-crossing ή y-crossing και υπολογίζεται με βάση την απόσταση από το κοντινότερο κελί. Όπως φαίνεται στο σχήμα 11, για την γραμμή ορατότητας *A* το x-crossing στο σημείο που απέχει *dA* από το *C* το ύψος θα έχει το ύψος του σημείου *C*, επειδή είναι το πλησιέστερο σημείο στο crossing. Αυτός ο τρόπος υπολογισμού βελτιώνει τη ταχύτητα του αλγορίθμου, με κόστος στην ακρίβεια του, καθώς ο αλγόριθμος γίνεται προσεγγιστικός. Η ακρίβεια του αλγορίθμου μπορεί να βελτιωθεί τροποποιώντας τον τρόπο υπολογισμού των crossings.

2.7.3 Αλγόριθμος Van Kreveld

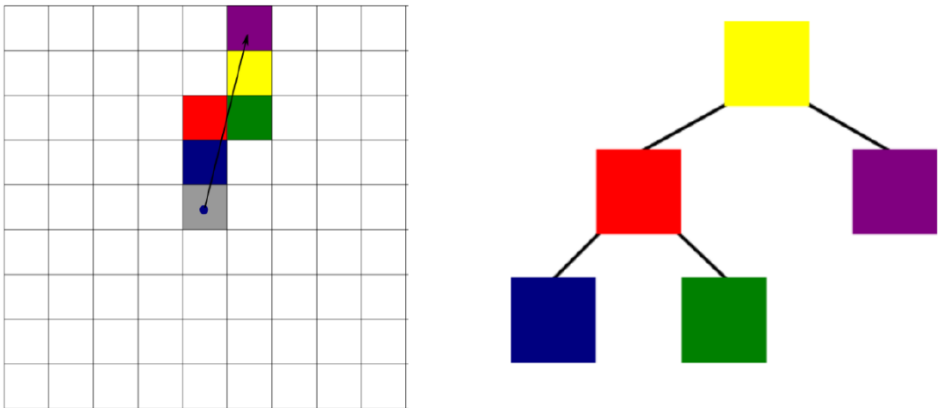
Ο αλγόριθμος ανάλυσης οπτικού πεδίου που προτάθηκε από τον Van Kreveld [1], πετυχαίνει υψηλή ακρίβεια και γρήγορο χρόνο εκτέλεσης. Ο αλγόριθμος αυτός, χρησιμοποιεί μια ημιευθεία που αρχίζει από το κελί που βρίσκεται ο παρατηρητής και καταλήγει στα όρια του DEM. Η ημιευθεία περιστρέφεται κατά 360 μοίρες, ώσπου να περάσει πάνω από τα κελιά που εξετάζονται. Όταν η ημιευθεία περνά πάνω από το κέντρο του κάθε κελιού, γίνεται ο υπολογισμός της ορατότητας του κελιού.



Σχήμα 12 Απεικόνιση της γραμμής ορατότητας που περιστρέφεται γύρω από τον παρατηρητή και πραγματοποιεί τη σάρωση των κελιών στον αλγόριθμο του Van Kreveld[11].

[10] Πηγή του σχήματος είναι η δημοσίευση [43]
[11] Πηγή του σχήματος είναι η δημοσίευση [1]

Όπως απεικονίζεται και παραπάνω, η ημιευθεία κινείται αντίστροφα από τους δείκτες του ρολογιού, και τα κελιά που τέμνει αποτελούν την γραμμή ορατότητας κάθε στιγμή της περιστροφής της. Κατά την περιστροφή της ημιευθείας, τα ύψη των κελιών που τέμνει αποθηκεύονται σε ένα ισορροπημένο δυαδικό δέντρο με κριτήριο την απόσταση του κάθε κελιού από τον παρατηρητή, έτσι ώστε τα κελιά που είναι πιο κοντά στον παρατηρητή να περιέχονται στα αριστερά φύλλα. Επιπλέον, για κάθε υποδέντρο αποθηκεύεται στο κόμβο του πατέρα η τιμή του απογόνου με την μεγαλύτερη κλίση. Ακολουθώντας τη διαδικασία αυτή, από τα φύλα του δέντρου έως τη ρίζα, αποθηκεύεται στο αριστερό παιδί του κάθε κόμβου τη μεγαλύτερη κλίση για τη LOS, από τον παρατηρητή μέχρι το κελί που αντιπροσωπεύει ο κόμβος, όπως φαίνεται και στο παρακάτω σχήμα (Σχήμα 13).



Σχήμα 13 Παράδειγμα εισαγωγής των σημείων της γραμμής ορατότητας σε δυαδικό δέντρο με βάση την απόσταση από τον παρατηρητή[12].

Στο δέντρο γίνονται εισαγωγές και εξαγωγές όταν συμβαίνει κάποιο γεγονός για το κελί που εξετάζεται. Για κάθε κελί υπάρχουν τριών ειδών γεγονότα:

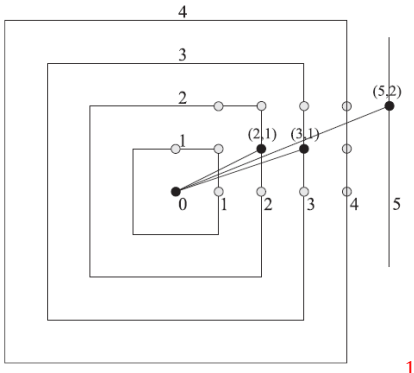
- Τα γεγονότα εισόδου που συμβαίνουν όταν η ημιευθεία συναντάει για πρώτη φορά ένα κελί: τότε προστίθεται στο δέντρο το κελί με την τιμή της κλίσης του.
- Τα γεγονότα εξόδου που συμβαίνουν όταν η ημιευθεία δεν τέμνει πλέον ένα κελί: τότε αφαιρείται το κελί από το δέντρο.
- Τα γεγονότα κέντρου που συμβαίνουν όταν η ημιευθεία περνά από το κέντρο ενός κελιού. Σε αυτή την περίπτωση, πραγματοποιείται δυαδική αναζήτηση για το κελί στο δέντρο, και συγκρίνεται η κλίση του με αυτό του αριστερού παιδιού. Αν η τιμή για το κελί που εξετάζεται είναι μεγαλύτερη από αυτή του αριστερού παιδιού, μόνο τότε υπάρχει ορατότητα στο κελί.

[12] Πηγή του σχήματος είναι η δημοσίευση [21]

Σε περίπτωση που δεν υπάρχει ορατότητα σε ένα κελί, είναι εφικτό να υπολογιστεί το ελάχιστο ύψος του παρατηρητή, ώστε να έχει ορατότητα στο κελί που εξετάζεται. Τέλος, αξίζει να σημειωθεί, ότι ο αλγόριθμος του Van Kreveld χρησιμοποιεί για τον υπολογισμό του ύψους των κελιών, το ύψος από το κέντρο κάθε κελιού, ακόμα και όταν η LOS δεν περνά ακριβώς από το κέντρο του κελιού [1]. Επομένως, μπορεί να επηρεαστεί η ακρίβεια του αλγορίθμου, ειδικά όταν μελετάται raster με μεγάλα κελιά. Για να αντιμετωπιστεί ο περιορισμός αυτός, χρειάζεται να τροποποιηθεί η μέθοδος που υπολογίζεται το ύψος του κάθε κελιού, ώστε να διαφοροποιείται ανάλογα με το σημείο που τέμνεται το κάθε κελί, χρησιμοποιώντας την τεχνική της παρεμβολής.

2.7.4 Αλγόριθμος XDraw

Ο Αλγόριθμος XDraw σχεδιάστηκε από τους Franklin κ.ά. [22], και πετυχαίνει καλύτερη απόδοση από τους αλγορίθμους R2 και R3. Ο αλγόριθμος αυτός υπολογίζει την ορατότητα των κελιών σε τετράγωνα δαχτυλίδια γύρω από τον παρατηρητή, αρχίζοντας από πλησιέστερο δαχτυλίδι καταλήγοντας στα άκρα του DEM, και όπως φαίνεται στο παρακάτω σχήμα (Σχήμα 14).



[13]

Σχήμα 14 Απεικόνιση του αλγορίθμου XDraw για τον υπολογισμό του σημείου (5, 2) με την χρήση crossing μεταξύ των σημείων (2, 1) και (3, 1).

Τα κελιά κάθε δαχτυλιδιού διατηρούν το μέγιστο ύψος της γραμμής ορατότητας που καταλήγει σε αυτά. Επομένως, ο υπολογισμός της ορατότητας των κελιών κάθε νέου δαχτυλιδιού απαιτεί την σύγκριση του ύψους του μόνο με το ύψος των κελιών του προηγούμενου δαχτυλιδιού. Με την μέθοδο αυτή απαιτείται μόνο μια σύγκριση για κάθε κελί, ενώ ταυτόχρονα, αποφεύγονται οι διπλότυποι υπολογισμοί που προκύπτουν στους αλγορίθμους R2 και R3. Στην περίπτωση που η γραμμή ορατότητας ενός κελιού δεν περνά από το κέντρο του κελιού του προηγούμενου

[13] Πηγή του σχήματος είναι η δημοσίευση [43]

δαχτυλιδιού, όπως συμβαίνει στο κελί, με συντεταγμένες (5.2) στο παραπάνω σχήμα (Σχήμα 14), το ύψος προσεγγίζεται με βάση τα δυο πλησιέστερα κελιά, που στην περίπτωση αυτή είναι τα κελιά με συντεταγμένες (2, 1) και (3, 1). Ο υπολογισμός μπορεί να γίνει με την χρήση του μεγίστου ύψους από τα πλησιέστερα κελιά, το μέσο όρο των υψών των κελιών ή με την τεχνική της παρεμβολής για τα δυο σημεία. Κάθε μια από τις παραλλαγές αυτές επηρεάζει την ακρίβεια του αλγορίθμου, αλλά και την απόδοση του. Το πρόβλημα που παρουσιάζει ο αλγόριθμος XDraw αφορά πιθανά σφάλματα στις μετρήσεις των σημείων που υπολογίζονται προσεγγιστικά. Κάθε ανακριβής προσέγγιση του ύψους σε ένα δαχτυλίδι, έχει ως αποτέλεσμα το λάθος να μεταφέρεται στα υπόλοιπα δαχτυλίδια, δημιουργώντας έτσι απώλεια ακρίβειας στον αλγόριθμο, ακόμα και αν τα υπόλοιπα κελιά υπολογίζονται με ακρίβεια.

2.8 Εργαλεία που χρησιμοποιήθηκαν

2.8.1 Python 3

Η Python είναι μια γλώσσα προγραμματισμού υψηλού επιπέδου, η οποία δημιουργήθηκε από τον Γκίντο βαν Ρόσσουμ (Guido van Rossum) [37] και κυκλοφόρησε το 1991. Η Python δίνει έμφαση στην αναγνωσιμότητα του κώδικα, που επιτρέπει εύκολη συντήρηση, του και ταυτόχρονα επιταχύνει την ταχύτητα εκμάθησης. Η σύνταξη της επιτρέπει την συγγραφή κώδικα σε λιγότερες γραμμές σε σύγκριση με άλλες γλώσσες. Η περιεκτικότητα των βιβλιοθηκών της την καθιστά ευέλικτη για ένα μεγάλο φάσμα εφαρμογών [38].

2.8.2 Διεπαφή Μεταβίβασης Μηνυμάτων

Η Διεπαφή Μεταβίβασης Μηνυμάτων (Message Passing Interface - MPI) είναι ένα πρότυπο βιβλιοθήκης που χρησιμοποιείται, ώστε να επιτευχθεί παράλληλη επεξεργασία με την επικοινωνία μεταξύ των διεργασιών μέσω της ανταλλαγής μηνυμάτων. Ο στόχος του MPI είναι να καθιερώσει ένα φορητό, αποδοτικό και ευέλικτο πρότυπο για το μοντέλο παράλληλου προγραμματισμού που βασίζεται στη μεταβίβαση μηνυμάτων [28]. Πιο συγκεκριμένα, το MPI εκτελεί τον ίδιο κώδικα σε διαφορετικές διεργασίες και χρησιμοποιεί την επικοινωνία ανάμεσα τους για να πετύχει την συγκέντρωση των αποτελεσμάτων των διεργασιών. Με αυτόν τον τρόπο μοιράζεται ο φόρτος εργασίας σε πολλούς επεξεργαστές και επιταχύνεται ο αλγόριθμος. Ταυτόχρονα, η χρήση του MPI είναι αρκετά περίπλοκη, καθώς πρέπει ο προγραμματιστής να εντοπίσει που μπορεί να εφαρμοστεί παράλληλη επεξεργασία και να την εφαρμόσει με

κατάλληλο τρόπο. Ο αλγόριθμος πρέπει να διαχωριστεί κατάλληλα, ώστε το αποτέλεσμα του αλγορίθμου να μην βασίζεται στην σειριακή εκτέλεση των διεργασιών, αφού η σειρά εκτέλεσης τους δεν είναι προκαθορισμένη. Επιπλέον, σε περίπτωση που πραγματοποιείται πολύ συχνή επικοινωνία μεταξύ διεργασιών, η χρήση του MPI μπορεί να οδηγήσει σε πιο αργή εκτέλεση από τον σειριακό προγραμματισμό. Ως εκ τούτου, η χρήση του MPI αποτελεί μια πιο προχωρημένη τεχνική προγραμματισμού, που μπορεί όμως να προσφέρει σημαντική βελτίωση στον χρόνο εκτέλεσης του κώδικα.

Παρακάτω, αναφέρονται μερικές από τις πιο βασικές εντολές που χρησιμοποιούνται. Αρχικά, πριν την χρήση οποιασδήποτε άλλης εντολής απαιτείται η εντολή MPI.COMM_WORLD, η οποία δημιουργεί μια ομάδα από διεργασίες που επικοινωνούν μεταξύ τους. Η εντολή πρέπει να προηγείται από τις υπόλοιπες εντολές του MPI, και η σύνταξη της φαίνεται στο παρακάτω σχήμα (Σχήμα 15).

```
comm = MPI.COMM_WORLD
```

Σχήμα 15 Σύνταξη της εντολής MPI.COMM_WORLD για την αρχικοποίηση των συναρτήσεων του MPI που θα χρησιμοποιηθούν στην συνέχεια του αλγορίθμου.

Δύο από τις πιο βασικές συναρτήσεις του MPI είναι οι comm.Get_rank() και η comm.Get_size(), οι οποίες επιστρέφουν τον αριθμό της διεργασίας που εκτελείται και το συνολικό αριθμό των διεργασιών που εκτελούνται αντίστοιχα.

```
rank = comm.Get_rank()
size = comm.Get_size()
```

Σχήμα 16 Παράδειγμα κλήσης των συναρτήσεων comm.Get_rank() και comm.Get_size() και αποθήκευση των τιμών, που επιστρέφουν στις μεταβλητές rank και size αντίστοιχα.

Στο παραπάνω σχήμα (Σχήμα 16) οι συναρτήσεις καλούνται και αποθηκεύουν τις τιμές που επιστρέφουν στις μεταβλητές rank και size. Οι συναρτήσεις αυτές είναι απαραίτητες στον προγραμματιστή, ώστε να διαπιστώσει των αριθμό των διεργασιών που χρησιμοποιούνται από τον χρήστη με την μεταβλητή *size* και να κατανείμει ανάλογα τον φόρτο την μεταβλητή *rank*.

Για τον συντονισμό των διεργασιών χρησιμοποιείται η επικοινωνία από σημείο σε σημείο (point to point communication) ή η συλλογική επικοινωνία (collective communication). Η επικοινωνία από σημείο σε σημείο αφορά στην ανταλλαγή δεδομένων μεταξύ δύο διεργασιών, κατά τη οποία η μια διεργασία στέλνει δεδομένα και η δεύτερη τα λαμβάνει. Η

αποστολή δεδομένων πραγματοποιείται με την εντολή *comm.send() και* η λήψη των δεδομένων γίνεται με την χρήση της *comm.recv().*

```
if rank==0
    comm.send(variable, dest=1,tag=0)
if rank==1
   variable2=comm.recv(source=0,tag=0)
```

Σχήμα 17 Παράδειγμα επικοινωνίας από σημείου σε σημείο μεταξύ της διεργασίας μηδέν και ένα με τις εντολές *comm.send() και comm.recv().*

Στο παραπάνω σχήμα (Σχήμα 17) παρουσιάζεται η αποστολή της τιμής της μεταβλητής *variable* από την διεργασία μηδέν στην διεργασία ένα, όπου αποθηκεύεται στη μεταβλητή *variable2*.

Η συλλογική επικοινωνία των διεργασιών αφορά στην ανταλλαγή δεδομένων πολλών διεργασιών με μία εντολή. Η συλλογική επικοινωνία εφαρμόζεται με αρκετές διαφορετικές εντολές, τόσο για να μοιράσει μια διεργασία τα δεδομένα της στις υπόλοιπες, όσο και για να συγκεντρωθούν τα δεδομένα όλων των διεργασιών σε μια. Παρακάτω, παρουσιάζεται η εντολή *comm.bcast().*

```
data = comm.bcast(data, root=0)
```

Σχήμα 18 Παράδειγμα συλλογικής επικοινωνίας με την εντολή comm.bcast() με την οποία η διεργασία μηδέν στέλνει την τιμή της μεταβλητής *data* στις υπόλοιπες διεργασίες.

Η εντολή *comm.bcast()* στέλνει την τιμή της μεταβλητής *data* από την διεργασία μηδέν σε όλες τις υπόλοιπες διεργασίες. Στην κατηγορία των εντολών συλλογικής επικοινωνίας υπάρχει ακόμα η εντολή *comm.scatter(),* η οποία μοιράζει έναν πίνακα σε ίσα κομμάτια για κάθε μια από τις διεργασίες, η εντολή *comm.gather(),* που λειτουργεί αντίστροφα και συγκεντρώνει τις τιμές από όλες τις διεργασίες σε μία διεργασία, και η εντολή *comm.allgather(),* κατά την εκτέλεση της οποίας όλες οι διεργασίες στείλουν μια μεταβλητή σε όλες τις υπόλοιπες διεργασίες.

Κεφάλαιο 3

Υλοποίηση του λογισμικού μέρους

Στο κεφάλαιο που ακολουθεί αναλύεται η μέθοδος που χρησιμοποιήθηκε για την υλοποίηση του αλγορίθμου και αναλύεται κάθε κομμάτι του λογισμικού.

3.1 Γενική επισκόπηση του λογισμικού

Ο αλγόριθμος που χρησιμοποιείται έχει ως βάση τον αλγόριθμο του Van Kreveld [1], ο οποίος αποτελείται από δυο βασικά στάδια. Στο πρώτο στάδιο πραγματοποιείται η σάρωση του DEM [31], που δίνεται σαν αρχείο εισόδου του αλγορίθμου. Για την υλοποίηση της σάρωσης χρησιμοποιείται μια ημιευθεία, που αρχίζει από την θέση του παρατηρητή και καταλήγει στα άκρα του DEM. Η ημιευθεία ακολουθεί φορά αντίστροφη από αυτή του ρολογιού και περιστρέφεται με την σειρά και στα τέσσερα τεταρτημόρια, καταλήγοντας στο σημείο αφετηρίας της. Κατά την σάρωση καταγράφονται οι μετρήσεις των τριών ειδών γεγονότων που συμβαίνουν για το κάθε κελί. Τα είδη των γεγονότων του κάθε κελιού είναι τα γεγονότα εισόδου, κέντρου, και εξόδου, που συμβαίνουν αντίστοιχα στο σημείο που τέμνει για πρώτη φορά η ημιευθεία το κελί, στο κέντρο του κελιού, και όταν η ημιευθεία σταματά να τέμνει το κελί. Οι μετρήσεις που καταγράφονται για το κάθε γεγονός είναι η απόσταση του σημείου που συμβαίνει το κάθε γεγονός από το κέντρο του κελιού του παρατηρητή, η γωνία της ημιευθείας στο σημείο που συμβαίνει το γεγονός και η κλίση του ύψους από τον παρατηρητή προς το σημείο. Τα γεγονότα αυτά τοποθετούνται σε μια λίστα που σε κάθε γραμμή περιέχει το είδος του γεγονότος με τις αντίστοιχες μετρήσεις του. Η λίστα ονομάζεται λίστα σάρωσης και ταξινομείται με βάση την γωνία, κατά την οποία συμβαίνει το κάθε γεγονός. Η σάρωση προηγείται του υπολογισμού ορατότητας, αφού απαιτείται ταξινόμηση της λίστας, σύμφωνα με την γωνία του κάθε γεγονότος, πριν την χρήση της για το δεύτερο στάδιο του αλγορίθμου.

Στο δεύτερο στάδιο γίνεται ο υπολογισμός της ορατότητας για το κάθε κελί. Κατά τον υπολογισμό της ορατότητας χρησιμοποιείται η λίστα σάρωσης, από την οποία διαβάζονται με τη σειρά τα γεγονότα, και γίνονται ενέργειες ανάλογα με τον τύπο του γεγονότος που συναντάται. Με τον τρόπο αυτό δημιουργείται μια νέα λίστα, η λίστα ορατότητας. Η λίστα ορατότητας ενημερώνεται με εισαγωγές και εξαγωγές και περιλαμβάνει τα κελιά που βρίσκονται στην ημιευθεία για κάθε γωνία της περιστροφής της. Εισαγωγές και εξαγωγές γίνονται όταν συμβαίνει γεγονός εισόδου και εξόδου, αντίστοιχα. Η ορατότητα για το κάθε

κελί υπολογίζεται όταν συμβαίνει γεγονός κέντρου. Έξοδος του αλγορίθμου είναι ένας πίνακας που περιέχει μηδέν για κάθε κελί στο οποίο ο παρατηρητής δεν έχει ορατότητα, και ένα για κάθε κελί στο οποίο έχει ορατότητα.

3.2 Είσοδος αλγορίθμου

Για την λειτουργία του αλγορίθμου απαιτούνται ένα αρχείο με το DEM της περιοχής που εξετάζεται. Το αρχείο αυτό είναι της μορφής TIFF (Tagged Image File Format) και περιέχει το υψόμετρο του εδάφους για το κέντρο του κάθε εικονοστοιχείου (pixel) της εικόνας. Επιπλέον, πρέπει να προσδιοριστούν οι συντεταγμένες της θέσης του παρατηρητή. Τέλος, απαραίτητο είναι ένα αρχείου κειμένου (plain text), το οποίο περιέχει την τιμή που προσαρμόζει το ύψος του παρατηρητή, ώστε το ύψος του να βρίσκεται 1,65 μέτρο πάνω από την επιφάνεια του εδάφους.

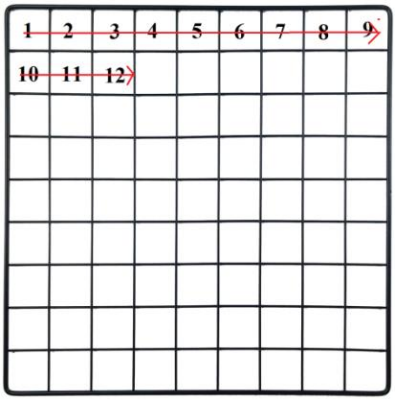
```
input for x 17 and y 975 has height 1124.5
input for x 17 and y 976 has height 1124.699951171875
input for x 17 and y 977 has height 1124.9000244140625
input for x 17 and y 978 has height 1125.699951171875
input for x 17 and y 979 has height 1125.9000244140625
input for x 17 and y 980 has height 1126.0999755859375
input for x 17 and y 981 has height 1126.5999755859375
```

Σχήμα 19 Μερικές από τις σειρές του αρχείου TIFF που περιέχει τα δεδομένα εισόδου για το ύψος των κελιών σε μορφή ευανάγνωστη από άνθρωπο.

Το αρχείο που παρουσιάζεται στο παραπάνω σχήμα (Σχήμα 19), περιέχει έναν δισδιάστατο πίνακα, του οποίου οι γραμμές και οι στήλες αποτελούν τις συντεταγμένες χ και ψ του DEM. Κάθε κελί του πίνακα περιέχει την τιμή του ύψους του κελιού με τις αντίστοιχες συντεταγμένες.

3.3 Σάρωση περιοχής

Η σάρωση της περιοχής αποτελεί το πρώτο στάδιο του αλγορίθμου και στόχος της είναι η μέτρηση των χαρακτηριστικών του κάθε κελιού, που είναι απαραίτητα στην συνέχεια του αλγορίθμου για τον υπολογισμό της ορατότητας. Η σάρωση είναι αναγκαίο να πραγματοποιηθεί πριν τον υπολογισμό της ορατότητας, καθώς παρέχει την σειρά εκτέλεσης των γεγονότων στα οποία βασίζεται ο αλγόριθμος στο δεύτερο στάδιο της εκτέλεσης του. Δεδομένα εισόδου για τη σάρωση είναι τα ύψη των κέντρων όλων των κελιών, η θέση και το ύψος του παρατηρητή. Ως έξοδο, επιστρέφει μία λίστα η οποία περιέχει σε κάθε γραμμή της το είδος του γεγονότος και τις μετρήσεις του.

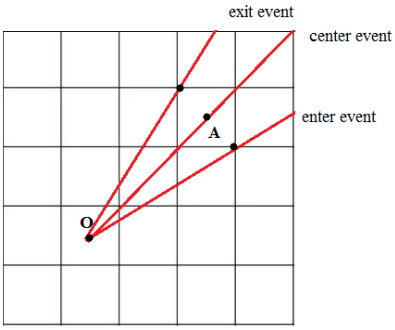


Σχήμα 20 Απεικόνιση της σειράς υπολογισμού των κελιών κατά την σάρωση στη σειριακή εκτέλεση του αλγορίθμου που αναπτύχτηκε.

3.3.1 Υπολογισμός θέσης γεγονότος

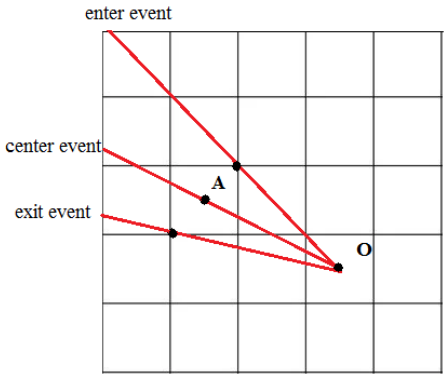
Για κάθε κελί συμβαίνουν τριών ειδών γεγονότα, καθένα από τα οποία απαιτεί συγκεκριμένες μετρήσεις, οι οποίες αλλάζουν σύμφωνα με τη θέση του κάθε γεγονότος. Οι συντεταγμένες αυτές αποτελούν το πρώτο σημείο, στο οποίο η ημιευθεία τέμνει το κελί κατά την περιστροφή της για τα γεγονότα εισόδου, το κέντρο του κελιού για τα γεγονότα κέντρου και το τελευταίο σημείο του η ημιευθεία τέμνει το κελί για τα γεγονότα εξόδου. Ο τρόπος που υπολογίζονται οι συντεταγμένες για το κάθε γεγονός εξαρτώνται από τη θέση του κελιού σε σχέση με τον παρατηρητή και αλλάζει για κάθε τεταρτημόριο γύρω από αυτόν, όπως φαίνεται και στα παρακάτω σχήματα. Για ένα κελί με συντεταγμένες *(x, y)* που ανήκει:

- στο πρώτο τεταρτημόριο, το οποίο εμφανίζεται στο σχήμα 21 με κόκκινες γραμμές, το γεγονός εισόδου συμβαίνει στο σημείο *(x+0.5, y-0.5)* και το γεγονός εξόδου στο σημείο *(x-0.5, y+0.5)*.



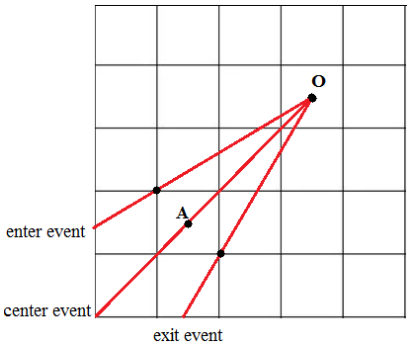
Σχήμα 21 Θέσεις γεγονότων εισόδου (enter), εξόδου (exit) και κέντρου (center)για κελί στο πρώτο τεταρτημόριο.

- στο δεύτερο τεταρτημόριο το κελί έχει γεγονός εισόδου στο σημείο *(x+0.5, y+0.5)* και το γεγονός εξόδου στο σημείο *(x-0.5, y-0.5)*.



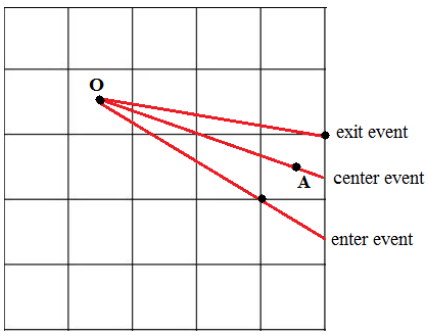
Σχήμα 22 Θέσεις γεγονότων εισόδου (enter), εξόδου (exit) και κέντρου (center) για κελί στο δεύτερο τεταρτημόριο.

- στο τρίτο τεταρτημόριο το κελί έχει γεγονός εισόδου στο σημείο *(x-0.5, y+0.5)* και το γεγονός εξόδου στο σημείο *(x+0.5, y-0.5).*



Σχήμα 23 Θέσεις γεγονότων εισόδου (enter), εξόδου (exit) και κέντρου (center) για κελί στο τρίτο τεταρτημόριο.

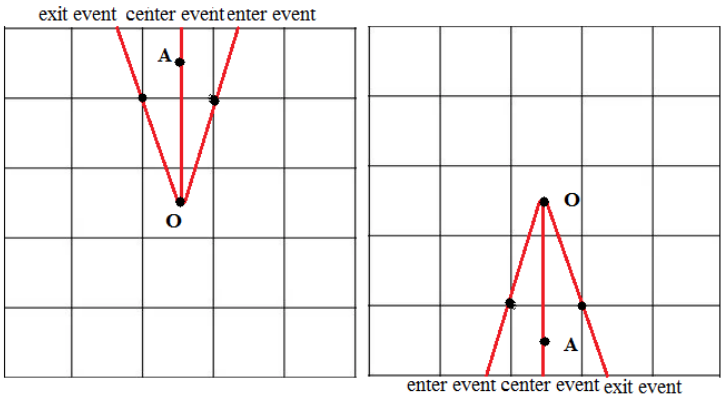
- στο τέταρτο τεταρτημόριο το κελί έχει γεγονός εισόδου στο σημείο *(x-0.5, y-0.5)* και το γεγονός εξόδου στο σημείο *(x+0.5, y+0.5).*



Σχήμα 24 Θέσεις γεγονότων εισόδου (enter), εξόδου (exit) και κέντρου (center) για κελί στο τέταρτο τεταρτημόριο.

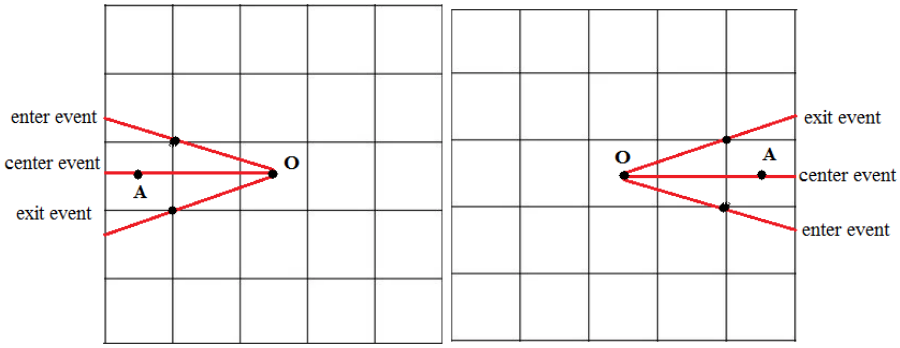
Με διαφορετικό τρόπο υπολογίζονται και τα κελιά που βρίσκονται στους άξονες *x* και *y* ανάλογα με τη θέση του κελιού ως προς τον παρατηρητή. Αν το κελί ανήκει:

- στον άξονα y με $y>y_{\text{θέσης παρατηρητή}}$, τότε το κελί έχει γεγονός εισόδου στο σημείο *(x+0.5, y-0.5)* και το γεγονός εξόδου στο σημείο *(x-0.5, y-0.5)*. Ενώ, για $y<y_{\text{θέσης παρατηρητή}}$ έχει γεγονός εισόδου στο σημείο *(x-0.5, y+0.5)* και το γεγονός εξόδου στο σημείο *(x+0.5, y+0.5)*.



Σχήμα 25 Θέσεις γεγονότων εισόδου (enter), εξόδου (exit) και κέντρου (center) για κελί που βρίσκεται στον άξονα ψ.

- στον άξονα *x* με $x>x_{\text{θέσης παρατηρητή}}$, τότε το κελί έχει γεγονός εισόδου στο σημείο *(x-0.5, y-0.5)* και το γεγονός εξόδου στο σημείο *(x-0.5, y+0.5)*. Ενώ, για $x<x_{\text{θέσης παρατηρητή}}$ έχει γεγονός εισόδου στο σημείο (x+0.5, *y+0.5)* και το γεγονός εξόδου στο σημείο *(x+0.5, y-0.5)*.

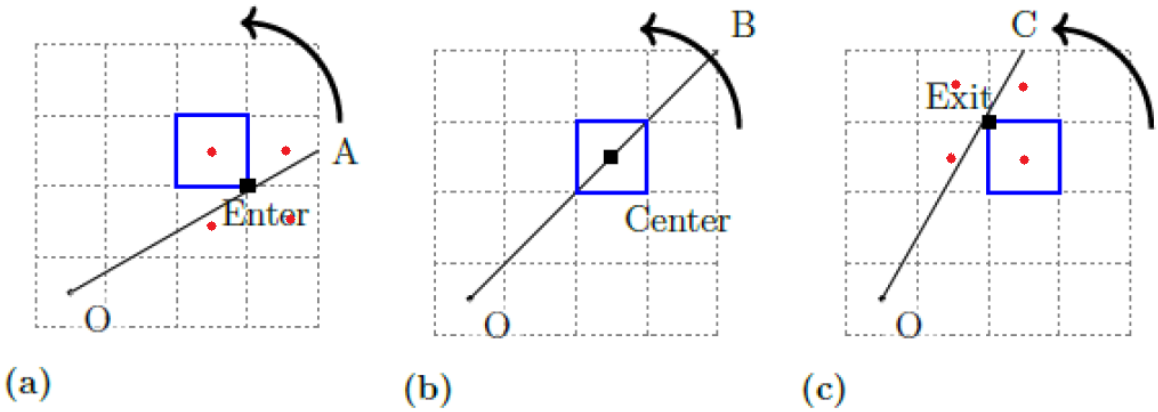


Σχήμα 26 Θέσεις γεγονότων εισόδου (enter), εξόδου (exit) και κέντρου (center) για κελί που βρίσκεται στον άξονα χ.

3.3.2 Υπολογισμός μετρήσεων γεγονότων

Με βάση τις συντεταγμένες που υπολογίστηκαν για το κάθε γεγονός, πραγματοποιούνται οι εξής μετρήσεις. Για τα γεγονότα κέντρου και εξόδου υπολογίζεται η γωνιά στην οποία προκύπτουν και αποθηκεύεται στην λίστα σάρωσης μαζί με τις

συντεταγμένες του κέντρου του κελιού. Για τα γεγονότα εισόδου καταγράφονται και για τα τρία γεγονότα του κελιού, οι συντεταγμένες του κελιού, η γωνία που προκύπτουν και το ύψος τους. Τα γεγονότα εισόδου πραγματοποιούν την αποθήκευση των μετρήσεων στην δομή δεδομένων. Επομένως, οποιαδήποτε μέτρηση απαιτείται για τον υπολογισμό των υπόλοιπων γεγονότων εισάγεται μέσω των γεγονότων εισόδου. Για να υπολογιστεί το ύψος του κελιού ο αλγόριθμος του Van Kreveld χρησιμοποιεί για όλα τα γεγονότα, το ύψος από το κέντρο του κελιού, με αποτέλεσμα να υπάρχει απώλεια ακρίβειας, όταν η ημιευθεία δεν περνάει από το κέντρο των κελιών. Για αυτό τον λόγο, στον παρών αλγόριθμο χρησιμοποιείται η τεχνική της παρεμβολής. Με την τεχνική αυτή, το ύψος κάθε γεγονότος υπολογίζεται προσεγγιστικά, με αποτέλεσμα για τα γεγονότα του ίδιου κελιού να καταγράφουν διαφορετικά ύψη. Για να υπολογιστεί το ύψος στα γεγονότα εισόδου και εξόδου χρησιμοποιούνται τα τέσσερα γειτονικά κελιά, όπως φαίνεται και στο σχήμα 27 με κόκκινες κουκίδες.



Σχήμα 27 Επιλογή των κελιών με κόκκινες κουκίδες για τον υπολογισμό του ύψους για το κάθε γεγονός με την τεχνική της παρεμβολής[14].

Όπως φαίνεται και παραπάνω η τεχνική της παρεμβολής χρησιμοποιείται για το γεγονός εισόδου και το γεγονός εξόδου, καθώς τα γεγονότα κέντρου συμβαίνουν στο κέντρο κελιού το ύψος του οποίου το DEM παρέχει την ακριβής τιμή. Τα κελιά που χρησιμοποιούνται για την παρεμβολή είναι τα τέσσερα πιο κοντινά κέντρα κελιών στη θέση του γεγονότος εισόδου ή εξόδου. Η επιλογή των κελιών που χρησιμοποιούνται αλλάζει ανάλογα με τη θέση του κελιού σε σχέση με τη θέση του παρατηρητή, όπως περιγράφεται και στο κεφάλαιο 3.3.1.Ο τύπος που χρησιμοποιείται για παρεμβολή τεσσάρων σημείων είναι ο παρακάτω, όπου *h1, h2, h3, h4* τα ύψη των γύρω κελιών και h το ύψος για το οποίο γίνεται η προσέγγιση:

$$h = \frac{h1+h2+h3+h4}{4} \qquad (5)$$

[14] Πηγή του σχήματος είναι η δημοσίευση [32]

Ο τύπος αυτός αποτελεί μια απλοποιημένη μορφή του πλήρη τύπου, καθώς τα σημεία πάντα απέχουν ίση απόσταση μεταξύ τους. Επιπρόσθετα, η γωνία που συμβαίνει το κάθε γεγονός ($\text{γωνία}_{\text{γεγονότος}}$) με συντεταγμένες *(x1,y1)* σε σχέση με το κελί του παρατηρητή με συντεταγμένες *(x0,y0)* υπολογίζεται από τον τύπο:

$$\text{Γωνία}_{\text{γεγονότος}} = \tan\left(\frac{\text{y1}-\text{y0}}{\text{x1}-\text{x0}}\right) \qquad (6)$$

Στον παραπάνω τύπο η εξίσωση της εφαπτομένης επιστρέφει τιμές από π έως $-\pi$, με αποτέλεσμα να απαιτείται μετατροπή για γωνίες μεγαλύτερες των π rad, αφού τα γεγονότα επεξεργάζονται με βάση την γωνία στην οποία συμβαίνουν. Οι μετρήσεις που καταγράφονται καταχωρούνται σε μια λίστα, η οποία ονομάζεται λίστα σάρωσης. Κάθε γραμμή της λίστας σάρωσης περιέχει ένα γεγονός και τις μετρήσεις που του αναλογούν. Στη συνέχεια, η λίστα αυτή ταξινομείται με βάση την γωνία που συμβαίνει το κάθε γεγονός και χρησιμοποιείται ως βάση για τον υπολογισμό της ορατότητας. Για να επιτευχθεί η σωστή εκτέλεση των γεγονότων λαμβάνεται υπόψη και η απόσταση των κελιών από τον παρατηρητή, καθώς κάποια γεγονότα συμβαίνουν στην ίδια γωνία, και επομένως, εκτελούνται με βάση την απόσταση τους από τον παρατηρητή. Στην περίπτωση αυτή εκτελούνται πρώτα τα γεγονότα που βρίσκονται πλησιέστερα στον παρατηρητή και στη συνέχεια τα πιο απομακρυσμένα.

3.4 Έλεγχος ορατότητας

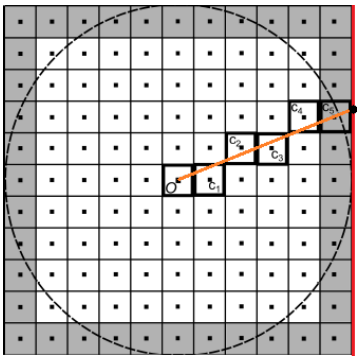
Για τον έλεγχο της ορατότητας των κελιών εφαρμόζεται η μεθοδολογία του αλγορίθμου του Van Kreveld με κάποιες παραλλαγές με στόχο να βελτιωθεί η ακρίβεια του αλγορίθμου. Για να υπολογιστεί η ορατότητα χρησιμοποιείται μια ημιευθεία με αρχή το κέντρο του κελιού του παρατηρητή και τέλος τα άκρα του DEM, που περιστρέφεται αντίστροφα από τη φορά του ρολογιού. Τα στοιχεία που τέμνει η ημιευθεία αποθηκεύονται σε μια λίστα που ονομάζεται λίστα ορατότητας. Η λίστα ορατότητας ενημερώνεται με εισαγωγές και εξαγωγές, όταν συμβαίνουν γεγονότα εισόδου και εξόδου, ενώ υπολογίζεται η ορατότητα του κελιού, όταν συμβαίνει γεγονός κέντρου. Η ημιευθεία λειτουργεί σαν LOS που αρχίζει από τον παρατηρητή και καταλήγει σε όλα τα υπόλοιπα κελιά. Η μέθοδος αυτή πετυχαίνει καλύτερη απόδοση από τους αλγορίθμους που R2 και R3, καθώς οι μετρήσεις κάθε κελιού δεν επαναλαμβάνονται για κάθε LOS που το τέμνει. Κατά συνέπεια, οι μετρήσεις για του κελί παραμένουν διαθέσιμες στη λίστα ορατότητας μέχρις ότου το κελί δεν περιλαμβάνεται σε άλλες γραμμές ορατότητας.

3.4.1 Αρχικοποίηση

Η λίστα ορατότητας είναι αρχικά κενή και χρειάζεται να εισαχθούν σε αυτή τα γεγονότα των κελιών που τέμνει η ημιευθεία, όταν αρχίζει τη σάρωση. Η αρχικοποίηση της λίστας ορατότητας είναι απαραίτητη, καθώς τα γεγονότα δεν συμβαίνουν ανά τύπο γεγονότος, άλλα σύμφωνα με την γωνία, στην οποία προκύπτουν. Επομένως, αν συμβεί ένα γεγονός για κελί που δεν βρίσκεται στη λίστα ορατότητας, ο αλγόριθμος καταλήγει σε σφάλμα. Για να γίνει η αρχικοποίηση χρησιμοποιείται η γωνία εκκίνησης της σάρωσης, η οποία για την σειριακή εκτέλεση του αλγορίθμου είναι η μηδενική γωνία, και η εξίσωση της ευθείας που φαίνεται στον παρακάτω τύπο όπου (x_0, y_0) η θέση του παρατηρητή και λ γωνία εκκίνησης της περιστροφής.

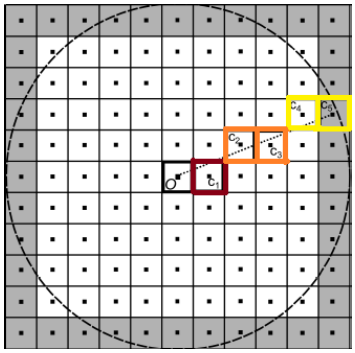
$$y = \lambda(x - x_0) + y_0 \quad (7)$$

Αρχικά, εξετάζεται η γωνία εκκίνησης της σάρωσης, η οποία για την σειριακή εκτέλεση του αλγορίθμου είναι η γωνία μηδέν μοιρών. Με την τιμή της γωνίας αναγνωρίζεται σε ποιο τεταρτημόριο ανήκει η ευθεία και υπολογίζονται τόσο τα όρια του DEM για το τεταρτημόριο, όσο και ο τρόπος υπολογισμού των συντεταγμένων των γεγονότων των κελιών, που περιγράφεται πιο αναλυτικά στο κεφάλαιο 3.3.1.



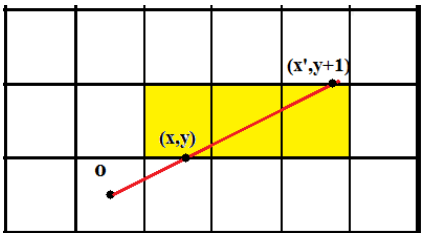
Σχήμα 28 Παράδειγμα αρχικοποίησης. Για την πορτοκαλί γραμμή ορατότητας αρχικοποιούνται τα κελιά με μαύρο περίγραμμα.

Χρησιμοποιώντας το σημείο τομής της ευθείας με το όριο του DEM, όπως φαίνεται και στο παραπάνω σχήμα (Σχήμα 28), για την τομή της κόκκινης και της πορτοκαλί γραμμής, υπολογίζεται το πιο απομακρυσμένο σημείο από τον παρατηρητή που τέμνεται από την ευθεία αρχικοποίησης. Ο εντοπισμός του σημείου αυτού είναι απαραίτητος για τον τερματισμό της αρχικοποίησης με τον ακριβή αριθμό γεγονότων της ευθείας. Ο υπολογισμός των κελιών της αρχικοποίησης αρχίζει από τα κελιά που βρίσκονται πλησιέστερα στον παρατηρητή και τερματίζει όταν συναντήσει το κελί που βρίσκεται στα όρια του DEM, όπως απεικονίζεται στο σχήμα 28. Για τον υπολογισμό των κελιών που βρίσκονται στην ευθεία χρησιμοποιούνται η εξίσωση της ευθείας και οι συντεταγμένες της θέσης του παρατηρητή.



Σχήμα 29 Παράδειγμα σειράς της αρχικοποίησης. Σε κάθε βήμα υπολογίζονται τα κελιά με διαφορετικό χρώμα αρχίζοντας από το κόκκινο κελί και καταλήγοντας στα κίτρινα.

Όπως φαίνεται και στο παραπάνω σχήμα (Σχήμα 29), για κάθε τιμή *y* που ανήκει στην ευθεία αρχικοποίησης υπολογίζονται τα κελιά που τέμνονται από την ευθεία και προστίθενται στη λίστα ορατότητας. Κάθε χρώμα των κελιών του σχήματος 29 αντιπροσωπεύει τα κελιά που εισάγονται για κάθε διαφορετική τιμή y της ευθείας. Στο παρακάτω σχήμα (Σχήμα 30) παρουσιάζεται ο υπολογισμός των συντεταγμένων των κελιών που τέμνονται από την ευθεία αρχικοποίησης για κάθε γραμμή. Όπως φαίνεται και στο σχήμα 30, για τον εντοπισμό των κελιών που βρίσκονται μεταξύ του *y* και του *y+1* υπολογίζεται το *x* και *x'* χρησιμοποιώντας τις τιμές *y* και *y+1* στην εξίσωση της ευθείας αρχικοποίησης. Αφού υπολογιστούν οι τιμές αυτές, αρχικοποιούνται τα κελιά από *(x, y)* έως *(x', y)*, και στη συνέχεια η διαδικασία αυτή επαναλαμβάνεται για τα κελιά μεταξύ *y+n* και *y+(n+1)*, μέχρι να υπολογιστούν όλα κελιά που τέμνονται από την ευθεία. Στο σχήμα (Σχήμα 30) που ακολουθεί, τα κελιά που βρίσκονται μεταξύ *y* και *y+1* απεικονίζονται με κίτρινο χρώμα.



Σχήμα 30 Παράδειγμα υπολογισμού των κελιών που τέμνονται από την γραμμή ορατότητας για κάθε γραμμή που τέμνει η γραμμή ορατότητας.

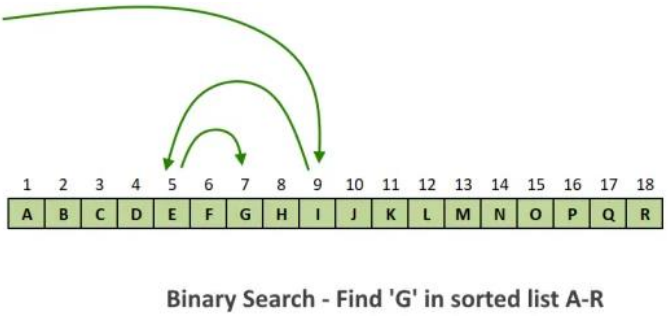
Για κάθε κελί που τέμνεται πραγματοποιείται μόνο ο υπολογισμός του γεγονότος εισόδου του κελιού, ώστε να είναι διαθέσιμα τα δεδομένα του για τους άλλους τύπους γεγονότων. Όταν ένα κελί τέμνεται στις συντεταγμένες του γεγονότος εξόδου του, τότε η εισαγωγή του στοιχείου είναι περιττή, ενώ αντίστοιχα αν τέμνεται στο κέντρο του το γεγονός κέντρου θα υπολογιστεί ακριβώς μετά την αρχικοποίηση. Σημαντικό είναι να σημειωθεί ότι η αρχικοποίηση απαιτεί απόλυτη ακρίβεια για τα κελιά και τα γεγονότα που περιλαμβάνει. Κάθε κελί έχει ακριβώς τρία γεγονότα, επομένως σε περίπτωση που κατά την αρχικοποίηση προστεθούν γεγονότα εισόδου

και εξόδου ή κελιά που δεν ανήκουν στην γραμμή αρχικοποίησης, τότε δημιουργούνται διπλότυπα γεγονότα, με αποτέλεσμα να προκαλείται σημαντική απόκλιση στην ακρίβεια του αποτελέσματος. Για αυτό το λόγο, τα γεγονότα εισόδου για τα κελιά που έγινε η αρχικοποίηση πρέπει να παραληφθεί από τη λίστα σάρωσης πριν αρχίσει η εκτέλεση των γεγονότων. Τέλος, αξίζει να σημειωθεί ότι για λόγους σαφήνειας αντί για μια δισδιάστατη λίστα ορατότητας χρησιμοποιήθηκαν ξεχωριστές λίστες για την κάθε μέτρηση που πραγματοποιήθηκε (απόσταση, γωνία, υψόμετρο). Όλες οι λίστες ταξινομούνται με βάση την απόσταση από τον παρατηρητή.

3.4.2 Δυαδική αναζήτηση

Η αναζήτηση αποτελεί αναπόσπαστο κομμάτι της ανάλυσης οπτικού πεδίου, καθώς απαιτείται για τον υπολογισμό κάθε γεγονότος. Η δυαδική αναζήτηση είναι απαραίτητη για την εύρεση των κελιών στην λίστα ορατότητας, καθώς ο μεγάλος όγκος δεδομένων επιβραδύνει σημαντικά τη σειριακή αναζήτηση. Η απόδοση της δυαδικής αναζήτησης είναι σαφώς καλύτερη, αφού παρουσιάζει πολυπλοκότητα *O(logn)* σε σχέση με τη σειριακή που παρουσιάζει *O(n)*. Ακόμα και στη χειρότερη περίπτωση της, για μια λίστα *n* στοιχείων η δυαδική αναζήτηση χρειάζεται *n/2+1* επαναλήψεις, όταν στην ανάλογη περίπτωση η σειριακή αναζήτηση απαιτεί *n* συγκρίσεις, καθιστώντας την πολύ πιο αποδοτική για μεγάλες λίστες. Σημαντική προϋπόθεση για την εφαρμογή της δυαδικής αναζήτησης είναι να είναι ταξινομημένη η λίστα, στην οποία εφαρμόζεται. Ο αλγόριθμος που αναπτύχτηκε έχει ως είσοδο την τιμή που αναζητείται και την λίστα, στην οποία ανήκει το στοιχείο, και έξοδο τη θέση του στοιχείου στη λίστα. Αρχικά, συγκρίνεται το στοιχείο που βρίσκεται στη μεσαία θέση της λίστας με το στοιχείο που αναζητείται και στη συνέχεια:

- Αν το στοιχείο είναι ίσο με αυτό που αναζητείται, τότε ο αλγόριθμος τελειώνει και επιστρέφει την θέση στην οποία βρίσκεται το στοιχείο.
- Αν το στοιχείο είναι μικρότερο από αυτό που αναζητείται τότε ο αλγόριθμος χρησιμοποιεί το άνω μισό της λίστας, επαναλαμβάνει την σύγκριση στη μέση του κομματιού της λίστας.
- Αν το στοιχείο είναι μεγαλύτερο από αυτό που αναζητείται, τότε ο αλγόριθμος χρησιμοποιεί το κάτω μισό της λίστας, επαναλαμβάνει την σύγκριση στη μέση του κομματιού της λίστας.



Σχήμα 31 Εφαρμογή δυαδικής αναζήτησης, με στόχο το γράμμα G[15].

Η δυαδική αναζήτηση εφαρμόζεται στην λίστα ορατότητας που περιέχει την απόσταση από τον παρατηρητή, η οποία παραμένει πάντα ταξινομημένη, καθιστώντας έτσι την δυαδική αναζήτηση ιδανική για την περίπτωση.

Η εφαρμογή της δυαδικής αναζήτησης σε συνδυασμό με την λίστα ορατότητας πετυχαίνει το ίδιο αποτέλεσμα με την αναζήτηση σε ένα δυαδικό δέντρο τόσο στην πολυπλοκότητα της, όσο και στην σειρά υπολογισμού των στοιχείων της λίστας. Για αυτό το λόγο χρησιμοποιείται έναντι του δυαδικού δέντρου που περιγράφεται στον αλγόριθμο του Van Kreveld.

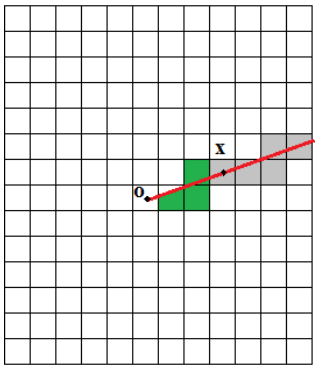
3.4.3 Υπολογισμός γεγονότων

Στη συνέχεια, από τη λίστα σάρωσης διαβάζονται με τη σειρά τα γεγονότα και πραγματοποιούνται ενέργειες ανάλογα με τον τύπο του γεγονότος. Για τα γεγονότα εισόδου εντοπίζεται η θέση, στην οποία πρέπει να γίνει η εισαγωγή του κελιού στην λίστα ορατότητας, χρησιμοποιώντας τη μέτρηση της απόστασης από τον παρατηρητή. Για να γίνει ο εντοπισμός της θέσης χρησιμοποιείται μια παραλλαγή της δυαδικής αναζήτησης. Στην παραλλαγή που χρησιμοποιείται η αναζήτηση τερματίζεται όταν δεν υπάρχει στοιχειό που να την ικανοποιεί και επιστρέφει την τελευταία θέση που υπολογίστηκε. Η θέση αυτή δείχνει το σημείο στο οποίο, αν γίνει η εισαγωγή του γεγονότος, η επόμενη τιμή στη λίστα έχει μεγαλύτερη απόσταση, ενώ το στοιχείο στην προηγούμενη θέση της λίστας έχει μικρότερη απόσταση με αποτέλεσμα να τηρείται η ταξινόμηση της λίστας.

Για τα γεγονότα εξόδου εντοπίζεται η θέση στην οποία πρέπει να γίνει η εξαγωγή του κελιού από τη λίστα ορατότητας, χρησιμοποιώντας δυαδική αναζήτηση στη λίστα που αποθηκεύεται η μέτρηση της απόστασης από τον παρατηρητή. Στη συνέχεια διαγράφονται τα στοιχεία της θέσης αυτής από όλες τις λίστες.

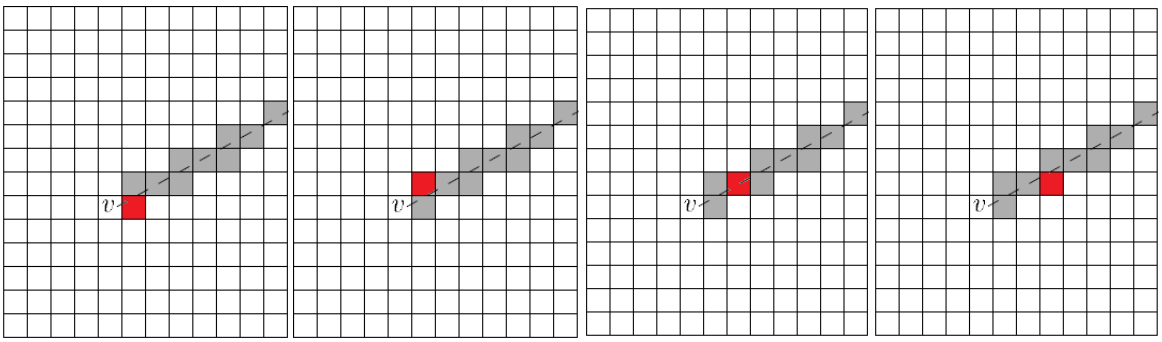
[15] Πηγή του σχήματος είναι η ιστοσελίδα https://algorithms.tutorialhorizon.com/linear-search-vs-binary-search/. Η επίσκεψη έγινε στις 11/5/2020

Τέλος, στα γεγονότα κέντρου πραγματοποιείται ο υπολογισμός της ορατότητας. Αρχικά, εντοπίζεται και πάλι η θέση του κελιού για το οποίο συμβαίνει το γεγονός κέντρου χρησιμοποιώντας δυαδική αναζήτηση στην λίστα που αποθηκεύεται η μέτρηση της απόστασης από τον παρατηρητή. Η θέση του κελιού χρησιμοποιείται ώστε να σταματήσει ο υπολογισμός των μετρήσεων για τα κελιά που βρίσκονται πιο μακριά από το κελί που εξετάζεται, αφού το ύψος τους δεν επηρεάζει την ορατότητα του πως φαίνεται στο παρακάτω σχήμα (Σχήμα 32).



Σχήμα 32 Παράδειγμα υπολογισμού ορατότητας για το κελί x. Ο υπολογισμός ορατότητας γίνεται μόνο για τα πράσινα κελιά που τέμνει η γραμμή ορατότητας.

Για κάθε κελί που βρίσκεται στην γραμμή ορατότητας, πριν το κελί για το οποίο συμβαίνει το γεγονός κέντρου, πραγματοποιείται σύγκριση του ύψους των κελιών. Η σειρά με την οποία υπολογίζονται τα κελιά βασίζεται στην απόσταση τους από τον παρατηρητή, όπως φαίνεται και στο παρακάτω σχήμα (Σχήμα 33).



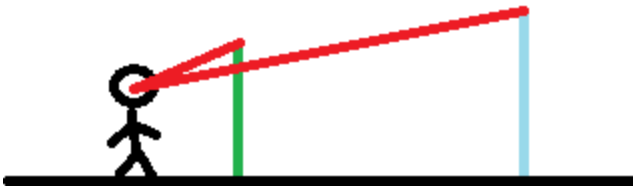
Σχήμα 33 Απεικόνιση της σειράς με την οποία υπολογίζεται η ορατότητα των κελιών όταν συμβαίνει γεγονός κέντρου.

Η σειρά που εφαρμόζεται έχει στόχο την βελτίωση της απόδοσης. Αν οποιοδήποτε κελί παρουσιάζει ύψος μεγαλύτερο από αυτό του κελιού του οποίου εξετάζεται, τότε δεν είναι απαραίτητος ο υπολογισμός των υπόλοιπων κελιών, αφού η ορατότητα στο κελί εμποδίζεται. Σε αυτό το σημείο οι υπολογισμοί της απόστασης γίνονται με τη χρήση μονάδων μέτρησης απόστασης και όχι κελιών, επειδή χρησιμοποιούνται για την μέτρηση της κλίσης. Αυτό συμβαίνει καθώς είναι πιθανό τα κελιά, ενώ έχουν τη μορφή τετραγώνου να αντιπροσωπεύουν περιοχές με διαφορετικό μήκος και πλάτος. Οι πλευρές του κάθε κελιού υπολογίζονται σε

μέτρα, ώστε να υπάρχει κοινή μονάδα μέτρησης και στον υπολογισμό του ύψους του κελιού σε σχέση με την καμπυλότητα της γης. Η καμπυλότητα της γης προκαλεί μείωση στο ύψος του κάθε κελιού, όπως φαίνεται και στον τύπο που ακολουθεί.

$$ύψος_{\ τελικό} = ύψος_{κελιού} - \frac{(απόσταση\ κελιού\ απο\ τον\ παρατηρητή)^2}{(2*6370997)} \qquad (8)$$

Για τον υπολογισμό της ορατότητας δεν συγκρίνεται το ύψος των κελιών, άλλα η κλίση που δημιουργεί το κελί του παρατηρητή με το κελί για το οποίο εξετάζεται. Η μέτρηση της κλίσης είναι πιο αξιόπιστη, αφού λαμβάνει υπόψη πως επηρεάζει η απόσταση την ορατότητα.



Σχήμα 34 Απεικόνιση της επιρροής της απόστασης από τον παρατηρητή στην ορατότητα σε σχέση με το ύψος των εμποδίων. Το μπλε εμπόδιο έχει μεγαλύτερο ύψος, αλλά μικρότερη κλίση λόγω της απόστασης του από τον παρατηρητή.

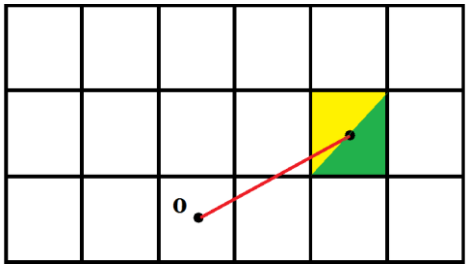
Όπως απεικονίζεται και στο σχήμα 34, ενώ το μπλε εμπόδιο έχει μεγαλύτερο ύψος από το πράσινο, αλλά βρίσκεται πιο μακριά από τον παρατηρητή. Κατά συνέπεια, η κλίση του είναι μικρότερη και ο παρατηρητής δεν έχει ορατότητα σε αυτό. Ένας ακόμα παράγοντας που επηρεάζει τις μετρήσεις αποτελεί το σημείο που τέμνει η γραμμή ορατότητας το κάθε κελί. Στις περισσότερες περιπτώσεις τα κελιά που προηγούνται στην λίστα από το κελί που εξετάζεται δεν τέμνονται από την ημιευθεία στο κέντρο τους. Επομένως, το ύψος τους δεν είναι αυτό που δίνεται από το DEM. Για αυτό το λόγο, κατά τον υπολογισμό του ύψους πραγματοποιείται μια επιπλέον προσέγγιση του ύψους με την τεχνική της παρεμβολής. Σε αυτή την προσέγγιση χρησιμοποιείται, είτε το γεγονότος εισόδου με συντεταγμένες *(x0,y0,h0)* και γωνία (*α*0), είτε το γεγονότος εξόδου με συντεταγμένες *(x2,y2,h2)* και γωνία (*α2)*, ανάλογα με την απόσταση από το σημείο που προσεγγίζεται, το οποίο έχει συντεταγμένες *(x,y,h)* και γωνία *(α)*. Στον παρακάτω τύπο θεωρείται πλησιέστερο το γεγονός εισόδου, αλλά με ανάλογο τρόπο υπολογίζεται και όταν είναι πλησιέστερο το γεγονός εξόδου:

$$υψος_{προσέγγιση} = h1 + |h1 - h0| * \frac{(|\alpha 1 - \alpha|)}{(|\alpha 1 - \alpha 0|)} \qquad (9)$$

Με αυτό τον τρόπο δίνεται μια τιμή για το ύψος του κελιού μεταξύ των τιμών των δύο γεγονότων, που είναι ανάλογη με το πιο κοντινό γεγονός στο σημείο τομής της LOS με το κελί. Για τον υπολογισμό της ορατότητας, συγκρίνεται η κλίση για το κελί που συμβαίνει γεγονός εισόδου, με την κλίση των υπολοίπων κελιών της λίστας ορατότητας αρχίζοντας από την αρχή της λίστας και τελειώνοντας πριν τη θέση του κελιού για το οποίο συμβαίνει γεγονός κέντρου.

Αν το κελί έχει την μεγαλύτερη κλίση σε σύγκριση με όλα τα κελιά που προηγούνται, τότε ο παρατηρητής έχει ορατότητα σε αυτό και του δίνεται τιμή 1, ενώ αν υπάρχει κελί με μεγαλύτερη κλίση παίρνει τιμή -1.

Για τον περιορισμό των υπολογισμών υπάρχουν δύο δικλείδες που εφαρμόζονται. Με τις εισαγωγές και εξαγωγές των κελιών δημιουργείται μια λίστα που περιλαμβάνει για το κάθε κελί τη μέγιστη τιμή ύψος που μπορεί να πάρει οποιοδήποτε γεγονός του. Η μέγιστη κλίση από κάθε κελί, που προηγείται στη λίστα, συγκρίνεται με το ύψος του κελιού για το οποίο συμβαίνει το γεγονός κέντρου, πριν γίνουν αναλυτικοί υπολογισμοί για το κελί. Σε περίπτωση που το κελί με το γεγονός κέντρου έχει κλίση μεγαλύτερη από αυτή της λίστας, τότε υπάρχει ορατότητα στο κελί, χωρίς να πραγματοποιηθούν περαιτέρω μετρήσεις.

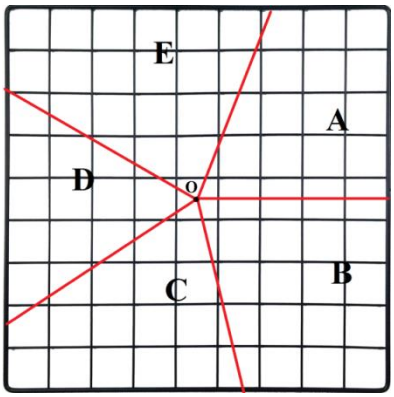


Σχήμα 35 Προσαρμογή του μέγιστου ύψους του κελιού ανάλογα με το σημείο τομής του από την γραμμή ορατότητας. Όταν συμβεί το γεγονός κέντρου, στο σημείο τομής των τριγώνων, το μέγιστο ύψος του κελιού επαναπροσδιορίζεται με βάση το ύψος του γεγονότος εξόδου.

Η μέγιστη πιθανή κλίση υπολογίζεται ανάλογα με την γωνία περιστροφής της γραμμής ορατότητας. Όπως απεικονίζεται και παραπάνω (Σχήμα 35), ο υπολογισμός για το μέγιστο ύψος διαφέρει πριν και μετά το γεγονός κέντρου του κάθε κελιού. Για γωνία μικρότερη του γεγονότος κέντρου κάθε κελιού, η μέγιστη κλίση υπολογίζεται από την σύγκριση της κλίσης για την θέση του γεγονότος εισόδου με την κλίση του για την θέση του γεγονότος κέντρου. Όταν συμβαίνει το γεγονός κέντρου κάθε κελιού, και αφού υπολογιστεί η ορατότητα του, γίνεται επαναπροσδιορισμός της μεγίστης τιμής του κελιού. Η μεγίστη τιμή της κλίσης για το κελί από το σημείο αυτό και μετά υπολογίζεται από τη σύγκριση της κλίσης του κελιού για την θέση του γεγονότος εξόδου και την κλίση του για την θέση του γεγονότος κέντρου. Η δεύτερη δικλείδα εφαρμόζεται όταν για οποιοδήποτε κελί, πριν το κελί που εξετάζεται, βρεθεί μεγαλύτερη τιμή ύψους. Σε αυτή την περίπτωση, ο υπολογισμός ορατότητας για το γεγονός διακόπτεται και δεν υπάρχει ορατότητα στο κελί. Με τον τρόπο αυτό αποφεύγονται περιττοί υπολογισμοί για τα υπόλοιπα κελιά της ευθείας.

3.5 Παραλληλοποίηση

Για την παράλληλη εκτέλεση του αλγορίθμου χρησιμοποιείται το MPI [28]. Το MPI κάνει χρήση διεργασιών που επικοινωνούν μεταξύ τους με στόχο την επιτάχυνση της εκτέλεσης του κώδικα. Οι διεργασίες εκτελούν τον ίδιο κώδικα με τη σειριακή εκτέλεση και βασίζονται στον προγραμματιστή να ορίσει διαφορετική είσοδο για την κάθε διεργασία, ώστε να παράγει διαφορετικά αποτελέσματα. Οι επιμέρους είσοδοι για κάθε διεργασία καθορίζονται με την επικοινωνία μεταξύ των διεργασιών λαμβάνοντας υπόψη των αριθμό των διεργασιών που χρησιμοποιεί ο κάθε χρήστης. Σημαντικό είναι η επικοινωνία μεταξύ των διεργασιών να αποφεύγεται, αν δεν είναι απαραίτητη, καθώς επιβραδύνει την εκτέλεση του κώδικα. Στον παρόντα αλγόριθμο απαιτείται η χρήση τουλάχιστον δυο διεργασιών. Η πρώτη διεργασία χρησιμοποιείται για να συγκεντρώνει τους υπολογισμούς των άλλων διεργασιών, ενώ οι υπόλοιπες διεργασίες αναλαμβάνουν τους υπολογισμούς της ορατότητας. Αρχικά, για την είσοδο των διεργασιών χωρίζεται το DEM σε κομμάτια με ίσο εμβαδό, τα οποία μοιράζονται σε όλες τις διεργασίες, εκτός από την πρώτη. Κάθε διεργασία εκτελεί τον αλγόριθμο για το κομμάτι που έχει αναλάβει και υπολογίζει την ορατότητα του.



Σχήμα 36 Παράδειγμα διαχωρισμού του DEM για την παράλληλη εκτέλεση του σε αλγορίθμου σε τομείς *A, B, C, D, E* με βάση τον παρατηρητή στο σημείο *O*.

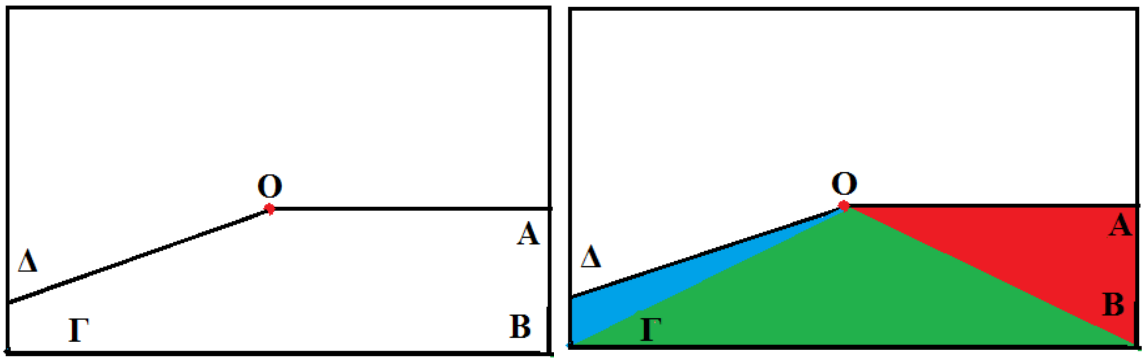
Ο διαχωρισμός των κομματιών για κάθε διεργασία γίνεται με βάση τη θέση του παρατηρητή, καθώς ο υπολογισμός εξαρτάται, τόσο στην απόσταση από τον παρατηρητή, όσο και στην γωνία περιστροφής της ημιευθείας που συμβαίνουν τα γεγονότα. Ο διαχωρισμός απεικονίζεται στο παραπάνω σχήμα (Σχήμα 36) για τα κομμάτια *A, B, C, D, E*. Κάθε διεργασία αναλαμβάνει ένα από τα παραπάνω κομμάτια και υπολογίζει η ορατότητα όλων των κελιών που περιλαμβάνει. Έπειτα, κάθε διεργασία επιστρέφει τις τιμές που υπολόγισε, για το κομμάτι που έχει αναλάβει, στην πρώτη διεργασία. Τέλος, η πρώτη διεργασία συγκεντρώνει τα αποτελέσματα και έχει ως έξοδο τα κελιά που είναι ορατά στον παρατηρητή για το σύνολο του DEM.

3.5.1 Διαχωρισμός ίσων κομματιών

Σκοπός του διαχωρισμού είναι να μοιραστούν τα κελιά του DEM, ώστε κάθε διεργασία να επεξεργάζεται ίσο αριθμό κελιών. Έτσι, οι διεργασίες πάντα διατηρούν παρόμοιο φόρτο, και δεν καθυστερεί μια διεργασία με δυσανάλογα μεγάλη είσοδο τον τερματισμό του αλγορίθμου. Επιπλέον, ο διαχωρισμός των επιμέρους κομματιών είναι σημαντικό να γίνει με τρόπο που τα καθιστά αυτόνομα για τον υπολογισμό της ορατότητας, ώστε να αποφευχθεί η επικοινωνία μεταξύ των διεργασιών. Επομένως, τα κομμάτια πρέπει να αρχίζουν από τον παρατηρητή και να καταλήγουν στα άκρα του DEM. Αυτό συμβαίνει, γιατί η μέθοδος με την οποία υπολογίζεται η ορατότητα, βασίζεται στη γωνία που η γραμμή ορατότητας περιστρέφεται γύρω από τον παρατηρητή και την απόσταση από τον παρατηρητή. Ως εκ τούτου, χωρίζεται το DEM σε αριθμό κομματιών ανάλογο με τον αριθμό των διεργασιών και με βάση τη θέση του παρατηρητή που ορίζει ο χρήστης. Αρχικά, η συνάρτηση δέχεται ως είσοδο το εμβαδό του DEM και τη θέση του παρατηρητή, και υπολογίζεται το εμβαδό που αναλογεί στην κάθε διεργασία από τον τύπο:

$$εμβαδό_{κάθε\ διεργασίας} = \frac{εμβαδό\ DEM}{(αριθμό\ διεργασιών - 1)}$$

Το σχήμα του κομματιού που αναλαμβάνει κάθε διεργασία εξαρτάται από την θέση του παρατηρητή και τον συνολικό αριθμό των διεργασιών. Συνεπώς, σε κάθε κομμάτι το σχήμα που προκύπτει, δεν έχει σταθερό αριθμό πλευρών. Για να υπολογιστεί το εμβαδό για οποιοδήποτε σχήμα του κομμάτι της κάθε διεργασίας, γίνεται ο υπολογισμός των επιμέρους τριγώνων από τα οποία αποτελείται. Τα τρίγωνα αυτά έχουν ως βάση κομμάτια από τις πλευρές του DEM και πλευρές τα ευθύγραμμα τμήματα που αρχίζουν από τη βάση για του τρίγωνο και καταλήγουν στη θέση του παρατηρητή. Στο σχήμα 37 γίνεται ο διαχωρισμός του πεντάπλευρου *ΑΒΟΓΔ* στα τρίγωνα *ΑΟΒ, ΒΟΓ, ΓΟΔ.*



Σχήμα 37 Υπολογισμός των κελιών που περιέχει ο τομέας ΑΒΟΓΔ. Αρχικά, διαχωρίζεται ο τομέας σε επιμέρους τρίγωνα και στη συνέχεια υπολογίζονται τα κελιά που περιέχουν τα τρίγωνα, υπολογίζοντας το εμβαδό τους.

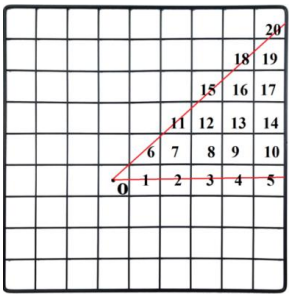
Για να υπολογιστούν τα εμβαδά των επιμέρους τριγώνων χρειάζεται να βρεθεί η βάση του κάθε τριγώνου. Για το κομμάτι *ΑΒΟΓΔ* που φαίνεται στο παραπάνω σχήμα (Σχήμα 37), αρχικά, υπολογίζεται το εμβαδό για το τρίγωνο που δημιουργείται με κορυφές τη θέση του παρατηρητή στο σημείο *Ο*, στο σημείο *Α* και στο σημείο *Β*, και απεικονίζεται με κόκκινο χρώμα. Στη συνέχεια, αφαιρείται από το μερίδιο που έχει αναλάβει η διεργασία, το εμβαδό του τριγώνου *ΑΟΒ*. Η διαδικασία επαναλαμβάνεται έως ότου το εμβαδό του τρίγωνο που δημιουργείται με βάση την επόμενη πλευρά του DEM να είναι μεγαλύτερο από το μερίδιο που αναλογεί στη διεργασία. Στην περίπτωση που παρουσιάζεται στο παραπάνω σχήμα (Σχήμα 37), η διαδικασία αρχίζει από το κόκκινο τρίγωνο και επαναλαμβάνεται για το πράσινο και το μπλε τρίγωνο. Κατά τον υπολογισμό του μπλε τριγώνου, το εμβαδό που αναλογεί σε ολόκληρη την πλευρά του είναι μεγαλύτερο από αυτό που απομένει στο μερίδιο της διεργασίας που υπολογίζεται. Στη περίπτωση αυτή χρησιμοποιείται ένα μέρος της πλευράς του DEM, το οποίο έχει εμβαδό ίσο με το υπόλοιπο μερίδιο της διεργασίας, όπως γίνεται στο σχήμα 37 για το τρίγωνο *ΓΟΔ*. Για τον υπολογισμό της βάσης του τελευταίου τριγώνου, χρησιμοποιείται ο παρακάτω τύπος :

$$Εμβαδό\ βάσης\ τριγώνου = \frac{2 * υπόλοιπο\ μερίδιο\ διεργασίας}{απόσταση\ πλευράς\ απο\ τον\ παρατηρητή} \quad (10)$$

Έπειτα, υπολογίζεται το σημείο στο οποίο τελειώνει το μερίδιο της διεργασίας, αφαιρώντας τη βάση του τριγώνου από την πλευρά του DEM, το οποίο συμβαίνει για το σχήμα 37 στο σημείο *Δ*. Χρησιμοποιώντας το σημείο αυτό, σε συνδυασμό με την εξίσωση της ευθείας, υπολογίζεται η γωνία στην οποία τελειώνει το μερίδιο της διεργασίας. Ο υπολογισμός τερματίζει στη προτελευταία διεργασία, και επιστρέφονται οι γωνίες εκκίνησης κάθε διεργασίας.

3.5.2 Σάρωση κελιών

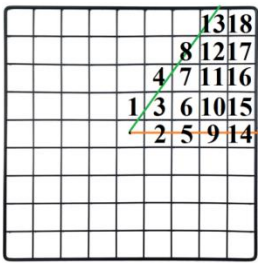
Ως συνέπεια του παραπάνω διαχωρισμού, μειώνεται το σύνολο των κελιών των οποίων υπολογίζονται οι μετρήσεις για κάθε διεργασία σε σχέση με την σειριακή σάρωση των κελιών. Επομένως, τροποποιείται και ο τρόπος που υπολογίζονται τα κελιά για την κάθε διεργασία.



Σχήμα 38 Παράδειγμα σάρωσης ενός τομέα για την παράλληλη εκτέλεση του αλγορίθμου. Μόνο τα αριθμημένα κελιά υπολογίζονται και η σάρωση πραγματοποιείται με την σειρά της αρίθμησης με βάση το τεταρτημόριο.

Όπως φαίνεται στο παραπάνω σχήμα (Σχήμα 38), για την παράλληλη επεξεργασία, η σάρωση διαφέρει από τη σειριακή, καθώς κάθε διεργασία υπολογίζει μόνο τα κελιά που περιέχει το κομμάτι που έχει αναλάβει. Συνεπώς, για να υπολογιστούν τα κελιά, απαιτείται ο υπολογισμός των ορίων του κομματιού κάθε διεργασίας. Ο υπολογισμός των ορίων πραγματοποιείται με την χρήση των γωνιών που υπολογίστηκαν στο κεφάλαιο 3.5.1 σε συνδυασμό με την εξίσωση της ευθείας και τη θέση του παρατηρητή. Για κάθε κομμάτι υπολογίζονται τα κελιά που βρίσκονται μεταξύ των δυο ευθειών. Τα κελιά στα όρια του κάθε κομματιού που δεν βρίσκονται μεταξύ των ευθειών, αλλά τέμνονται τουλάχιστον από μία από τις ευθείες, περιλαμβάνονται στα κελιά που υπολογίζει η διεργασία. Ο υπολογισμός των ορίων των κομματιών, με την χρήση της εξίσωσης της ευθείας έχει ως αποτέλεσμα τα όρια να υπολογίζονται σε πραγματικούς αριθμούς, ενώ οι θέσεις των κελιών να αντιστοιχούν σε ακέραιους αριθμούς. Επομένως, για όλα τα κομμάτια υπολογίζεται κομμάτι μεγαλύτερο από το απαιτούμενο κατά ένα κελί στον άξονα χ και ψ για τα άκρα του κομματιού. Η αύξηση αυτή εφαρμόζεται και για τους δύο άξονες, καθώς σφάλματα προκύπτουν όταν η γωνία περιστροφής προσεγγίζει τις γωνίες π και 3π/2 και για τον άξονα των ψ, και όταν η γωνία περιστροφής προσεγγίζει τις γωνίες μηδέν και π και για τον άξονα των χ. Η παραπάνω διαδικασία πραγματοποιείται ώστε να μην παραληφθεί κάποιο κελί, ενώ ταυτόχρονα γίνεται έλεγχος ώστε να μην υπάρχουν διπλότυπα για τα κελιά στα όρια των κομματιών. Για την αφαίρεση των διπλότυπων γεγονότων πραγματοποιούνται δύο ενέργειες. Τα γεγονότα που συμβαίνουν σε μικρότερη γωνία από αυτή της γωνίας εκκίνησης παραλείπονται, ενώ ταυτόχρονα η ανάγνωση της λίστας σάρωσης διακόπτεται αν ξεπεραστεί η γωνία τερματισμού της κάθε διεργασίας.

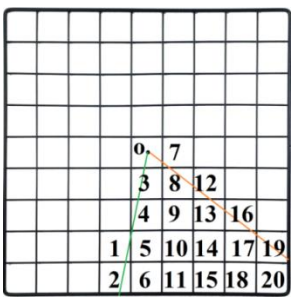
Για τον υπολογισμό των κομματιών ως είσοδος απαιτούνται τα δεδομένα που δέχεται η σειριακή σάρωση, ενώ είναι απαραίτητη και η γωνία στην οποία αρχίζει και στην οποία τελειώνει το κομμάτι. Με την χρήση των δύο αυτών γωνιών και την εξίσωση της ευθείας, υπολογίζονται οι συντεταγμένες των σημείων που οι δυο ευθείες τέμνουν τα άκρα του DEM. Στη συνέχεια, υπολογίζονται τα κελιά ανάλογα με το τεταρτημόριο στο οποίο βρίσκονται. Η πρώτη περίπτωση συμβαίνει όταν το κομμάτι βρίσκεται ολόκληρο σε ένα τεταρτημόριο.



Σχήμα 39 Παράδειγμα σάρωσης τομέα που βρίσκεται μόνο σε ένα τεταρτημόριο, ο υπολογισμός πραγματοποιείται με την αναγραφόμενη σειρά.

Όπως φαίνεται και στο παραπάνω σχήμα (Σχήμα 39) όταν το κομμάτι που υπολογίζεται δεν ξεπερνά τα όρια ενός μόνο τεταρτημόριου ο υπολογισμός γίνεται μεταξύ των δύο ευθειών, ενώ ταυτόχρονα γίνεται έλεγχος ώστε να μην ξεπεραστούν τα όρια του DEM. Ο υπολογισμός των κελιών αρχίζει από το κοντινότερο σημείο της πρώτης ευθείας στον παρατηρητή και τελειώνει όταν φτάσει το σημείο στα όριο της εικόνας. Όπως παρουσιάζεται και στο σχήμα 39, για κάθε στοιχείο της πορτοκαλί ευθείας υπολογίζονται τα κελιά στην ίδια στήλη, που βρίσκονται πάνω από τη γραμμή αυτή, και ταυτόχρονα κάτω από την πράσινη γραμμή. Στο κάθε κομμάτι ανήκουν και τα κελιά που τέμνονται από τις δύο ευθείες, ακόμα και αν το μεγαλύτερο μέρος τους δεν βρίσκεται ενδιάμεσα τους.

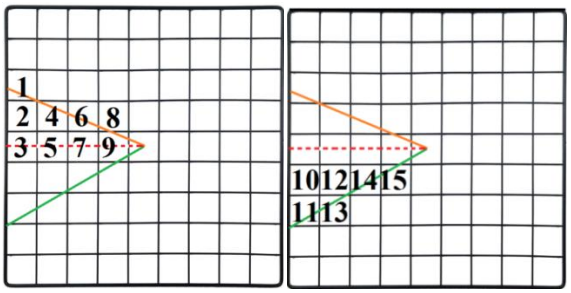
Η δεύτερη περίπτωση συμβαίνει όταν το κομμάτι βρίσκεται ταυτόχρονα στο πρώτο και δεύτερο τεταρτημόριο. Στην περίπτωση αυτή υπολογίζονται τα κελιά μεταξύ των δύο ευθειών και το όριο του DEM. Με ανάλογο τρόπο υπολογίζονται τα κομμάτια που βρίσκονται ταυτόχρονα στο τρίτο και τέταρτο τεταρτημόριο.



Σχήμα 40 Παράδειγμα σάρωσης τομέα που βρίσκεται ταυτόχρονα στο τρίτο και τέταρτο τεταρτημόριο, ο υπολογισμός πραγματοποιείται με την αναγραφόμενη σειρά.

Όπως φαίνεται και στο παραπάνω σχήμα (Σχήμα 40), όταν το κομμάτι που υπολογίζεται βρίσκεται ταυτόχρονα στο τρίτο και στο τέταρτο τεταρτημόριο, ο υπολογισμός δεν γίνεται μεταξύ των δύο ευθειών αλλά μεταξύ της κάθε ευθείας και τα όρια του DEM. Στην περίπτωση αυτή ο υπολογισμός των κελιών αρχίζει από το πιο απομακρυσμένο σημείο από τον παρατηρητή της πρώτης ευθείας και δεν τελειώνει στη θέση του παρατηρητή, αλλά συνεχίζεται ο υπολογισμός από το σημείο τομής των δυο ευθειών μέχρι το τέλος της δεύτερης ευθείας.

Η τελευταία περίπτωση συμβαίνει όταν το κομμάτι περιέχει της διεργασίας το τρίτο και το τέταρτο τεταρτημόριο. Σε αυτή τη περίπτωση γίνεται χρήση των παραπάνω μεθόδων, έτσι ώστε να υπολογίζονται τα κελιά στο πρώτο και δεύτερο τεταρτημόριο και στη συνέχεια στο τρίτο και τέταρτο τεταρτημόριο.

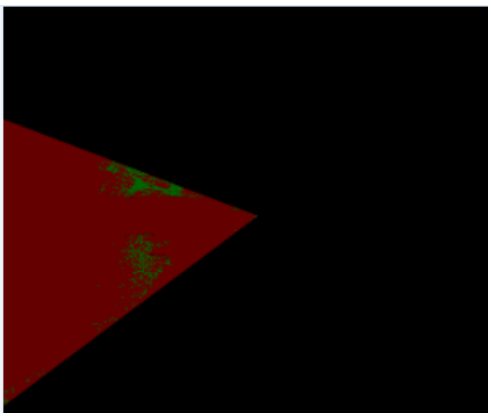


Σχήμα 41 Παράδειγμα σάρωσης τομέα που βρίσκεται μεταξύ δεύτερου και τρίτου τεταρτημόριου, ο υπολογισμός πραγματοποιείται με την αναγραφόμενη σειρά.

Όπως φαίνεται και στο παραπάνω σχήμα (Σχήμα 41), για τον υπολογισμό κομματιών που αρχίζουν από γωνία μικρότερη των π rad και καταλήγουν σε γωνία μεγαλύτερη των π rad πραγματοποιείται ξεχωριστά ο υπολογισμός για το τμήμα μέχρι π rad, και για αυτό που είναι μεγαλύτερο των π rad. Οποιαδήποτε από τις δύο μεθόδους που αναφέρθηκαν παραπάνω είναι κατάλληλη για τον υπολογισμό των κελιών του κάθε κομματιού. Οι υπολογισμοί του κάθε κομματιού περιλαμβάνουν εκτός από τα κελιά που περιέχονται σε κάθε κομμάτι και αυτά που απλά τέμνονται από τις ευθείες. Επομένως, για να αποφευχθεί η εισαγωγή διπλότυπων γεγονότων είναι απαραίτητο να μοιραστούν μόνο στο ένα κομμάτι τα κελιά που τέμνονται από την ευθεία με γωνία π rad, καθώς τέμνει κελιά που περιλαμβάνονται και στα δύο κομμάτια. Όπως απεικονίζεται και στα παραπάνω σχήματα (Σχήμα 41), τα κελιά που τέμνονται από την ευθεία, όταν βρίσκεται στην γωνία π rad υπολογίζονται με το κομμάτι που βρίσκεται στο δεύτερο τεταρτημόριο και παραλείπονται στις μετρήσεις του τρίτου τεταρτημόριου. Με ανάλογο τρόπο γίνεται και ο υπολογισμός των κελιών για την γωνία μηδέν rad. Ο υπολογισμός των κελιών στη περίπτωση αυτή πραγματοποιείται με το κομμάτι που βρίσκεται στο πρώτο τεταρτημόριο και παραλείπονται από την τελευταία διεργασία που περιέχει την γωνία 2π rad.

3.5.3 Συγκέντρωση αποτελέσματος

Τελευταίο βήμα της παραλληλοποίησης του αλγορίθμου αποτελεί η συγκέντρωση των υπολογισμών των επιμέρους διεργασιών. Για να επιτευχθεί η συγκέντρωση του αποτελέσματος απαιτείται επικοινωνία μεταξύ των διεργασιών που εκτελούν τους υπολογισμούς με την πρώτη διεργασία. Η πρώτη διεργασία δεν εκτελεί υπολογισμό ορατότητας, αλλά συγκεντρώνει τα αποτελέσματα των υπολοίπων διεργασιών σε μια δισδιάστατη λίστα. Η επικοινωνία μεταξύ των υπόλοιπων διεργασιών με την πρώτη διεργασία πραγματοποιείται με τις εντολές comm.send και comm.recv. Οι διεργασίες, εκτός από την πρώτη, χρησιμοποιούν την εντολή comm.send. Η εντολή αυτή στέλνει την λίστα που περιέχει τα αποτελέσματα του υπολογισμού ορατότητας για κάθε διεργασίας στην πρώτη διεργασία. Η εντολή comm.send έχει παραμέτρους τη δισδιάστατη λίστα, που περιλαμβάνει τις συντεταγμένες *x* και *y* του κάθε κελιού που υπολογίστηκε, και μηδέν ή ένα που υποδεικνύει την ορατότητα του κελιού και προορισμό την διεργασία μηδέν. Η διεργασία μηδέν χρησιμοποιεί την εντολή comm.recv σε ένα βρόχο με παράμετρο την κάθε διεργασία από την οποία δέχεται επικοινωνία. Η εντολή αυτή αποδέχεται την επικοινωνία από τις υπόλοιπες διεργασίες και αποθηκεύει την λίστα που στέλνουν σε ένα δισδιάστατο πίνακα, στον οποίο συγκεντρώνονται τα αποτελέσματα όλων των διεργασιών.



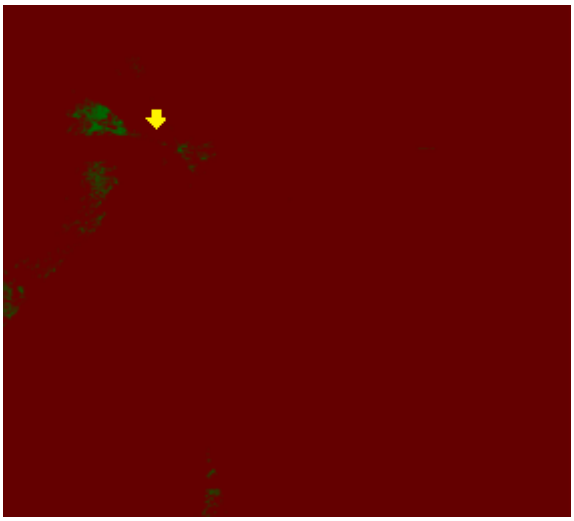
Σχήμα 42 Γραφική απεικόνιση του αποτελέσματος που υπολογίζει μια μόνο διεργασία. Με κόκκινο απεικονίζονται τα κελιά που δεν είναι ορατά από τον παρατηρητή, με πράσινο τα κελιά που είναι ορατά από τον παρατηρητή, ενώ με μαύρο παρουσιάζονται τα κελιά που υπολογίζονται από άλλες διεργασίες και δεν υπολογίζεται η ορατότητα τους.

Στο σχήμα 42, παρουσιάζεται ο υπολογισμός ορατότητας που πραγματοποιεί κάθε διεργασία για το κομμάτι που έχει αναλάβει. Με πράσινο απεικονίζονται τα κελιά που είναι ορατά από τον παρατηρητή, με κόκκινο αυτά που δεν είναι ορατά, ενώ με μαύρο είναι τα κελιά για τα οποία δεν υπολογίστηκε η ορατότητα από την παρούσα διεργασία και δεν υπάρχουν στοιχεία για την ορατότητα τους. Η διεργασία μηδέν πραγματοποιεί την συγχώνευση των επιμέρους κομματιών, δημιουργώντας έτσι το συνολικό αποτέλεσμα για ολόκληρο το DEM.

Ο φόρτος εργασίας της διεργασίας μηδέν αυξάνεται ανάλογα με τον αριθμό των διεργασιών που χρησιμοποιούνται κατά την εκτέλεση του αλγορίθμου. Για να αποφευχθεί η συμφόρηση στην διεργασία μηδέν είναι απαραίτητη η αφαίρεση των περιττών κελιών από την έξοδο των επιμέρους διεργασιών. Τα κελιά που δεν περιέχουν μετρήσεις είναι αυτά που απεικονίζονται στο σχήμα 42 με μαύρο χρώμα για την κάθε διεργασία. Για να παραληφθούν τα κελιά αυτά προσαρμόζεται η δομή της λίστας που στέλνεται στην διεργασία μηδέν. Η λίστα που δημιουργείται σε κάθε διεργασία, αποθηκεύει σε κάθε γραμμή της τόσο τις συντεταγμένες του κελιού, όσο και την ορατότητα του με μηδέν ή ένα, παραλείποντας έτσι τα περιττά κελιά κάθε διεργασίας και βελτιώνοντας σημαντικά την απόδοση του αλγορίθμου.

3.6 Έξοδος αλγορίθμου

Ο αλγόριθμος κατά την ολοκλήρωση του δημιουργεί μια δισδιάστατη λίστα που περιέχει όλα τα κελιά του DEM. Η λίστα περιέχει για κάθε κελί την τιμή μηδέν αν ο χρήστης δεν έχει ορατότητα στο κελί και ένα αν ο χρήστης έχει ορατότητα.



Σχήμα 43 Αποτέλεσμα της ανάλυσης οπτικού πεδίου, όπου με πράσινο απεικονίζονται τα κελιά που είναι ορατά από τον παρατηρητή, με κόκκινο απεικονίζονται τα κελιά που δεν είναι ορατά, και ο παρατηρητής βρίσκεται στο σημείο που δείχνει το κίτρινο βέλος.

Όπως φαίνεται στο σχήμα 43 τα κελιά που δεν είναι ορατά από τον παρατηρητή έχουν κόκκινο χρώμα ενώ τα κελιά που είναι ορατά παρουσιάζονται με πράσινο χρώμα. Σημαντικό είναι να αναφερθεί πως ο παρατηρητής δεν βρίσκεται πάντα στο κέντρο του σχήματος. Στην περίπτωση του παραπάνω σχήματος (Σχήμα 43), ο παρατηρητής βρίσκεται στο πάνω αριστερά μέρος του σχήματος, στο σημείο που δείχνει το κίτρινο βέλος.

Κεφάλαιο 4

Συμπεράσματα και μελλοντικές βελτιώσεις

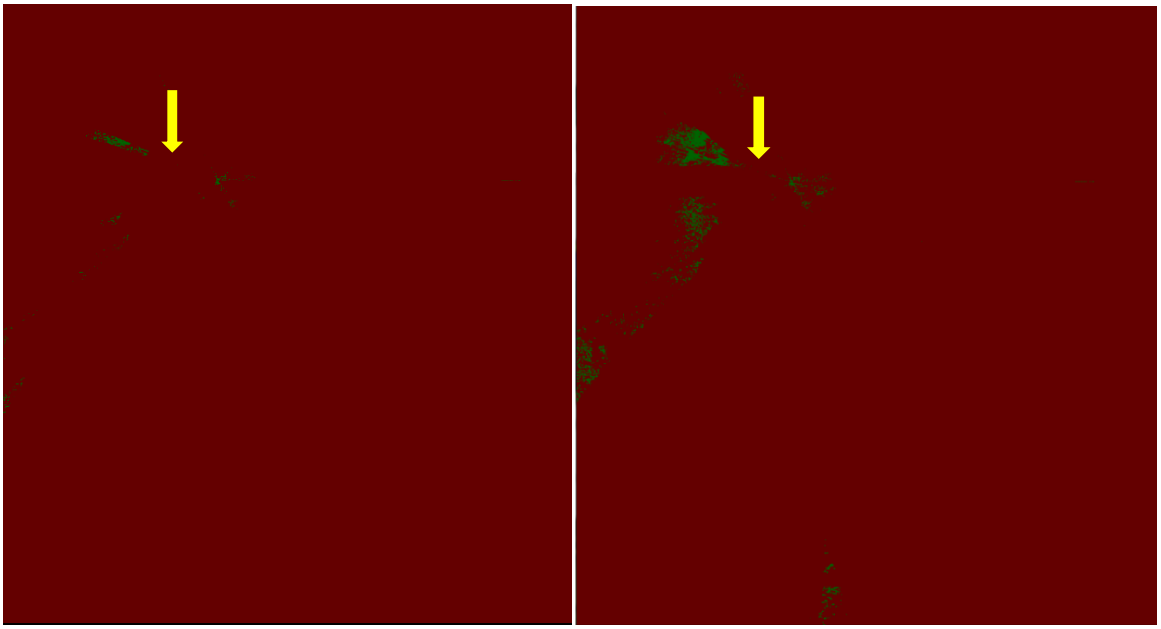
Στο παρακάτω κεφάλαιο καταγράφονται τα αποτελέσματα της παρούσας εργασίας. Στη συνέχεια, αναφέρονται τα προβλήματα που παρουσιάστηκαν κατά την συγγραφή του λογισμικού. Τέλος, αναφέρονται μελλοντικές επεκτάσεις που θα μπορούσαν να ενσωματωθούν στην εφαρμογή που υλοποιήθηκε.

4.1 Αποτελέσματα

Στην παρούσα εργασία χρησιμοποιήθηκε το ψηφιακό υψομετρικό μοντέλο, το οποίο προέρχεται από τον εθνικό δρυμό της Βαυαρίας. Η απόκτηση του πραγματοποιήθηκε μέσω της συνδρομής του κ. Marco Heurich[16], ως διαχειριστή του δρυμού, και του ΕΚΕΤΑ. Η περιοχή που εξετάζεται είναι το εθνικό πάρκο του δάσους της Βαυαρίας, που καλύπτει 243,69 km^2 και βρίσκεται στα σύνορα Γερμανίας και Τσεχίας. Το υψομετρικό μοντέλο, προήλθε από εναέρια λήψη που πραγματοποιήθηκε στο διάστημα τριών ημερών τον Ιούνιο του 2012. Η λήψη των δεδομένων έγινε από την εταιρία Milan Flug GmbH με την χρήση του λέιζερ σαρωτή Riegl 680i. Το παραπάνω υψομετρικό μοντέλο αποτελεί την είσοδο του αλγορίθμου για τα σχήματα που ακολουθούν, αλλά και για τις μετρήσεις του υποκεφαλαίου 4.1.1.

Ο αλγόριθμος που αναπτύχτηκε έχει ως στόχο να προσφέρει μια γρήγορη και ακριβή μέθοδο που πραγματοποιεί ανάλυση οπτικού πεδίου. Η ακρίβεια του αλγορίθμου που υλοποιήθηκε, αν συγκριθεί με αυτή του βασικού αλγορίθμου του Van Kreveld [1], είναι πολύ καλύτερη, αφού η μέτρηση του ύψους κάθε κελιού γίνεται με την τεχνική της παρεμβολής. Τα σχήματα 44, 45 και 46 απεικονίζουν κομμάτι της περιοχής του ψηφιακού υψομετρικού μοντέλου με έκταση 7,2 km^2. Η περιοχή επιλέχτηκε για λόγους ευκρίνειας, καθώς η απεικόνιση ολόκληρου του πάρκου θα έκανε δύσκολη την αναγνώριση των κελιών λόγω του μεγέθους της εικόνας.

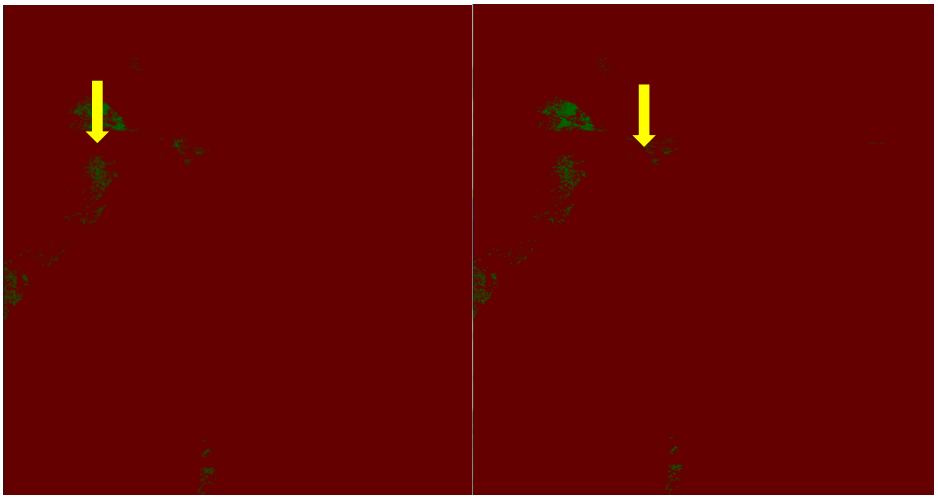
[16] Επικεφαλής του τμήματος διαχείρισης επισκεπτών και παρακολούθησης του εθνικού πάρκου της Βαυαρίας



Σχήμα 44 Αριστερά το αποτέλεσμα του αλγορίθμου του Van Kreveld, δεξιά ο αλγόριθμος που υλοποιήθηκε στην παρούσα διπλωματική. Ο παρατηρητής βρίσκεται στο σημείο που δείχνει το κίτρινο βέλος, τα πράσινα κελιά αντιπροσωπεύουν τα ορατά κελιά και τα κόκκινα κελιά αντιπροσωπεύουν τα μη ορατά κελιά.

Η απόκλιση αυτή δημιουργείται εξαιτίας της δομής του αλγορίθμου, καθώς ένα σφάλμα κατά τον υπολογισμό του ύψος ενός κελιού επηρεάζει όλα τα κελιά που το περιλαμβάνουν στην γραμμή ορατότητας τους. Για αυτό το λόγο, η υψηλή ακρίβεια αποτελεί προτεραιότητα του αλγορίθμου, καθώς μικρές αλλαγές στον υπολογισμό δημιουργούν εκτεταμένη αστοχία στις μετρήσεις.

Ο αλγόριθμος που υλοποιήθηκε παρουσιάζει ακρίβεια παρόμοια με αυτή του αλγορίθμου που χρησιμοποιείται στο GRASS (Geographic Resources Analysis Support System) [27], το οποίο είναι ένα διεθνώς αναγνωρισμένο λογισμικό που χρησιμοποιείται για την επεξεργασία γεωχωρικών δεδομένων.



Σχήμα 45 Αριστερά το αποτέλεσμα από τον αλγόριθμο ανάλυσης οπτικού πεδίου που χρησιμοποιείται από το GRASS και δεξιά το αποτέλεσμα του αλγορίθμου που υλοποιήθηκε στην παρούσα διπλωματική. Ο

παρατηρητής βρίσκεται στο σημείο που δείχνει το κίτρινο βέλος, τα πράσινα κελιά αντιπροσωπεύουν τα ορατά κελιά και τα κόκκινα κελιά αντιπροσωπεύουν τα μη ορατά κελιά.

Όπως παρουσιάζεται παραπάνω, τα αποτελέσματα του αλγόριθμο που υλοποιήθηκε στην παρούσα εργασία και του αλγορίθμου του GRASS σχεδόν ταυτίζονται. Στο υποκεφάλαιο που ακολουθεί περιλαμβάνονται μετρήσεις για την πιο ακριβής σύγκριση των παραπάνω αλγορίθμων.

4.1.1 Μετρήσεις

Καθώς ο αλγόριθμος που υλοποιήθηκε έχει αρκετή απόκλιση από τον πρωτότυπο αλγόριθμο του Van Krevel, είναι απαραίτητα να εξεταστεί, αν η απόκλιση που προκύπτει είναι αποτέλεσμα σφαλμάτων στο αλγόριθμο που υλοποιήθηκε ή οφείλεται σε έλλειψη ακρίβειας του βασικού αλγορίθμου του Van Kreveld. Επομένως, παρακάτω εκτελούνται δυο συγκρίσεις. Η πρώτη σύγκριση έχει στόχο να δείξει την έλλειψη ακρίβειας του πρωτότυπου αλγορίθμου του Van Keveld σε σχέση με τον πιο ακριβή αλγόριθμο του GRASS. Η δεύτερη σύγκριση έχει στόχο να δείξει ότι τα αποτελέσματα του αλγορίθμου που υλοποιήθηκε είναι ορθά, αφού ταιριάζουν με αυτά του αλγόριθμο του GRASS. Στις μετρήσεις που ακολουθούν χρησιμοποιήθηκε έκταση 63,76 km^2 του υψομετρικού μοντέλου. Παρακάτω, πραγματοποιείται η πρώτη σύγκριση μεταξύ του πρωτότυπου αλγόριθμου του Van Kreveld με τον αλγόριθμο του GRASS.

	Ορατά κελιά GRASS	Μη ορατά κελιά GRASS	Ευστοχία χρήστη
Ορατά κελιά Van Kreveld	87332	3712	0,95922
Μη ορατά κελιά Van Kreveld	85222	6888733	0,98777
Ευστοχία παραγωγού	0,5061	0,9995	
Συντελεστή k	0,666	Συνολική ακρίβεια	0,9874

Πίνακας 1 Σύγκριση βασικού αλγορίθμου με τον αλγόριθμο του GRASS

Το συμπέρασμα που προκύπτει από τον παραπάνω πίνακα (πίνακας 1) είναι πως ο πρωτότυπος αλγόριθμος παρουσιάζει σημαντική αστοχία ως προς την ακρίβεια του. Ενώ τα κελιά που δεν είναι ορατά εντοπίζονται με επιτυχία, τα κελιά στα οποία ο παρατηρητής έχει ορατότητα παρουσιάζουν σημαντική απόκλιση. Η απόκλιση αυτή αντικατοπτρίζεται και στην τιμή του συντελεστή k. Ο συντελεστής k αποτελεί μια μεταβλητή, που εκφράζει την ομοιότητα των αποτελεσμάτων, και σε αντίθεση με την συνολική ακρίβεια, περιλαμβάνει στις μετρήσεις του την πιθανότητα τα κοινά σημεία που προκύπτουν να βρέθηκαν τυχαία. Η τιμή του κυμαίνεται μεταξύ του ένα και του μείον ένα. Όταν παίρνει την τιμή ένα υπάρχει πλήρης ομοιότητα, ενώ

για την τιμή μείον ένα υπάρχει πλήρης αντίθεση μεταξύ των αποτελεσμάτων. Στην περίπτωση που ο συντελεστής *k* έχει τιμή μηδέν, τότε οποιαδήποτε ομοιότητα των αποτελεσμάτων είναι τυχαία. Για κάθε τιμή μεγαλύτερη του μηδενός, τα αποτελέσματα του συντελεστή *k* οδηγούν σε διαφορετικό βαθμό συμφωνίας. Για τιμές από 0 έως 0,2 υπάρχει μικρή συμφωνία μεταξύ των αποτελεσμάτων, για τιμές από 0.21 έως 0,4 υπάρχει αρκετή συμφωνία, για τιμές από 0,41 έως 0,6 υπάρχει μέτρια συμφωνία, για τιμές από 0.61 έως 0,8 υπάρχει ουσιώδης σημασία και για τιμές από 0,81 έως 0,99 υπάρχει σχεδόν τέλεια συμφωνία [39].Ο πίνακας που ακολουθεί (πίνακας 2), περιλαμβάνει τις μετρήσεις για διαφορετικές θέσεις του παρατηρητή, ώστε να εξεταστεί η ακρίβεια του αλγορίθμου κατά μέσο όρο. Στον πίνακα 2 γίνονται οι ίδιες συγκρίσεις για τυχαίες θέσεις του παρατηρητή, ώστε να εξακριβωθεί ότι η αστοχία που παρατηρήθηκε δεν είναι τυχαία. Οι παρακάτω θέσεις του παρατηρητή επιλέχτηκαν από ένα σύνολο τυχαίων τιμών, ώστε να παρουσιάζουν μεγάλο πλήθος τόσο ορατών όσο και μη ορατών κελιών και να διατηρηθεί η ακεραιότητα των αποτελεσμάτων. Το σύνολο των τυχαίων τιμών υπολογίστηκε με την χρήση της βιβλιοθήκης random της Python.

Θέση παρατηρητή	Ευστοχία χρήστη για ορατά κελιά	Ευστοχία χρήστη για μη ορατά κελιά	Ευστοχία παραγωγού για ορατά κελιά	Ευστοχία παραγωγού για μη ορατά κελιά	Συντελεστής *k*
(1900,2125)	*0,943*	*0,9997*	*0,5537*	*0,9997*	*0,712*
(1450,606)	*0,974*	*0,9958*	*0,58*	*0,9998*	*0,7324*
(1577,2169)	*0,9115*	*0,9998*	*0,4228*	*0,99998*	*0,594*
(675,341)	*0,92659*	*0,999*	*0,5939*	*0,9998*	*0,7448*
(45,951)	*0,9818*	*0,9982*	*0,5820*	*0,9999*	*0,7349*
(1454,1297)	*0,9876*	*0,9910*	*0,5590*	*0,9998*	*0,7130*
(1640,1641)	*0,9624*	*0,9973*	*0,4192*	*0,9999*	*0,5896*
(1593,1898)	*0,9613*	*0,9999*	*0,5710*	*0,9999*	*0,7269*
(506,1667)	*0,9863*	*0,9985*	*0,5281*	*0,9999*	*0,6905*
(1057,1648)	*0,9840*	*0,9870*	*0,5354*	*0,9999*	*0,6915*

Πίνακας 2 Στοιχεία για διαφορετικές θέσης παρατηρητή για τον βασικό Van Kreveld

Όπως φαίνεται και στον παραπάνω πίνακα (πίνακας 2), ο αλγόριθμος του Van Kreveld συχνά δεν εντοπίζει κελιά, στα οποία έχει ορατότητα ο παρατηρητής, παρόλο που πετυχαίνει καλή ακρίβεια στα κελιά που δεν είναι ορατά για οποιαδήποτε από τις θέσεις του παρατηρητή. Αντίθετα, ο αλγόριθμος, που υλοποιήθηκε στην παρούσα διπλωματική, πετυχαίνει σημαντική

βελτίωση στην ακρίβεια, και εξασφαλίζει αποτελέσματα παρόμοια με τον αλγόριθμο του GRASS. Στο πίνακα 3, συγκρίνεται ο αλγόριθμος του GRASS με τον αλγόριθμο που υλοποιήθηκε στη παρούσα διπλωματική εργασία, και εξετάζεται η ακρίβεια των αποτελεσμάτων του.

	Ορατά κελιά GRASS	Μη ορατά κελιά GRASS	Ευστοχία χρήστη
Ορατά κελιά **αλγορίθμου διπλωματικής**	171167	172	0,9989
Μη ορατά κελιά **αλγορίθμου διπλωματικής**	1387	6892273	0,99979
Ευστοχία παραγωγού	0,992	0,999975	
Συντελεστής *k*	0,9919	Συνολική ακρίβεια	0,99978

Πίνακας 3 Σύγκριση του αλγορίθμου που υλοποιήθηκε με τον αλγόριθμο του GRASS

Είναι απαραίτητο όμως, να εξεταστούν αρκετά διαφορετικά σημεία, για να εξασφαλιστεί η ακεραιότητα των αποτελεσμάτων που παρουσιάζονται παραπάνω. Ο πίνακας, που ακολουθεί, (πίνακας 4) συγκρίνει τις μετρήσεις από μερικά τυχαία σημεία για του δύο αλγορίθμους. Οι θέσεις του παρατηρητή επιλέχτηκαν, όπως και για τον προηγούμενο πίνακα, από ένα σύνολο τυχαίων τιμών, ώστε να περιέχουν μεγάλο πλήθος, τόσο από ορατά, όσο και από μη ορατά κελιά.

Θέση παρατηρητή	Ευστοχία χρήστη για ορατά κελιά	Ευστοχία χρήστη για μη ορατά κελιά	Ευστοχία παραγωγού για ορατά κελιά	Ευστοχία παραγωγού για μη ορατά κελιά	Συντελεστής *k*
(1900,2125)	*0,984*	*0,9999*	*0,997*	*0,9999*	*0,9986*
(1450,606)	*0,9962*	*0,99996*	*0,9968*	*0,99996*	*0,9984*
(1577,2169)	*0,9953*	*0,9999*	*0,979*	*0,9999*	*0,9889*
(675,341)	*0,9951*	*0,9999*	*0,9961*	*0,9999*	*0,9980*
(45,951)	*0,9947*	*0,9999*	*0,9981*	*0,9999*	*0,9973*
(1454,1297)	*0,9954*	*0,9999*	*0,9997*	*0,9999*	*0,9976*
(1640,1641)	*0,9939*	*0,9999*	*0,9803*	*0,9999*	*0,9969*
(1593,1898)	*0,9799*	*0,9999*	*0,9970*	*0,9999*	*0,9898*
(506,1667)	*0,9792*	*0,9999*	*0,9998*	*0,9999*	*0,9894*
(1057,1648)	*0,9912*	*0,9999*	*0,9998*	*0,9997*	*0,9954*

Πίνακας 4 Στοιχεία για διαφορετικές θέσης παρατηρητή για τον αλγόριθμο που υλοποιήθηκε

Όπως παρατηρείται, οι μετρήσεις παραμένουν ακριβείς για όλες τις θέσεις του παρατηρητή, με τον συντελεστή *k* να παίρνει τιμές πολύ κοντά στο ένα.

Εξίσου σημαντικό είναι να εξεταστεί και η βελτίωση της απόδοσης που πετυχαίνει η παράλληλη εκτέλεση του αλγορίθμου. Η τεχνική αυτή εξαρτάται από το μέγεθος της εικόνας, την επεξεργαστική ισχύ του υπολογιστή που χρησιμοποιείται, αλλά και το πόσο ευδιάκριτα είναι τα ορατά κελιά. Πιο συγκεκριμένα, ο τελευταίος παράγοντας αφορά πόσο μεγάλες είναι οι διαφορές ύψους ανάμεσα στα κελιά. Αν τα κελιά περιμετρικά του παρατηρητή είναι πολύ ψηλότερα από όλα τα υπόλοιπα κελιά, ο υπολογισμός είναι εξαιρετικά γρήγορος. Αντίθετα, αν τα κελιά έχουν όλα το ίδιο ύψος, ο υπολογισμός είναι χρονοβόρος, αφού απαιτούνται αναλυτικοί υπολογισμοί για το σύνολο των κελιών.

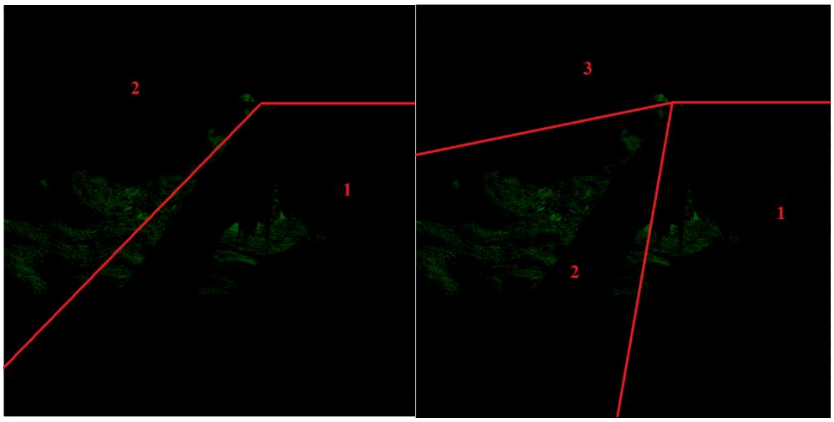
Παρακάτω, παρουσιάζεται ο χρόνος εκτέλεσης σε σχέση με τις διεργασίες που χρησιμοποιούνται, και συγκρίνεται ο χρόνος εκτέλεσης με αυτόν του αλγορίθμου του GRASS.

Διεργασίες	*1*	*15*	*30*	*45*	*60*	*75*	*90*
Χρόνος **αλγορίθμου διπλωματικής**	540sec	85sec	48sec	35sec	28sec	22sec	19sec
Χρόνος GRASS	55sec						

Πίνακας 5 Μετρήσεις χρόνου της παράλληλης εκτέλεσης του αλγορίθμου της διπλωματικής

Η απόκλιση στον χρόνο που παρουσιάζεται, οφείλεται στις τεχνικές διαχείρισης μνήμης σε συνδυασμό με τη χρήση δεικτών, που εφαρμόζονται στον αλγόριθμο του GRASS. Ο αλγόριθμος του GRASS είναι γραμμένος σε C++, η οποία παρέχει τη δυνατότητα στον προγραμματιστή να διαχειριστεί την διαθέσιμη μνήμη του υπολογιστή, ενώ αντίθετα, η Python περιορίζεται από το GIL (Global Interpreter Lock). Κατά συνέπεια, στην Python η παρέμβαση του προγραμματιστή στη μνήμη δημιουργεί προβλήματα, αφού παρεμβαίνει στη διαχείριση μνήμης που εκτελείται από τον διερμηνέα (interpreter). Ταυτόχρονα η C++ υποστηρίζει την χρήση δεικτών, μια δομή δεδομένων που καταλαμβάνει πολύ μικρότερο χώρο στη μνήμη από τους πίνακες που είναι διαθέσιμοι στην Python.

Από τον παραπάνω πίνακα (Πίνακας 5), εντοπίζονται και οι περιορισμοί της παράλληλης επεξεργασίας. Όπως φαίνεται και στον πίνακα 5, ο χρόνος εκτέλεσης δεν είναι ακριβώς ανάλογος των διεργασιών που χρησιμοποιούνται. Ένας παράγοντας που επηρεάζει τις επιδόσεις ιδιαίτερα σε μεγάλο αριθμό διεργασιών είναι οι περιορισμοί του υλικού του υπολογιστή που τρέχει τον κώδικα. Ο πιο βασικός παράγοντας, όμως, έχει να κάνει με την φύση του αλγορίθμου. Το πρόβλημα χωρίζεται σε ακριβώς ίσους τομείς με ίδιο αριθμό κελιών, αλλά κάθε τομέας δεν απαιτεί ίσο χρόνο για την ανάλυσή του. Παρόλα αυτά, δεν είναι δυνατό να χωριστούν τα κομμάτια ανάλογα με τον χρόνο υπολογισμού τους, αφού για να αναγνωριστεί ο απαιτούμενος χρόνος του κάθε τομέα είναι απαραίτητος πρώτα ο υπολογισμός του τομέα.



Σχήμα 46 Γραφική απεικόνιση του φόρτου για δυο και τρεις τομείς στην παράλληλη εκτέλεση του αλγορίθμου που υλοποιήθηκε.

Όπως απεικονίζεται και στο σχήμα 46 είναι δυνατό οι τρεις τομείς να έχουν χειρότερη απόδοση από τους δύο. Η περίπτωση αυτή προκύπτει, όταν το κομμάτι που αναλογεί στον τομέα περιέχει κελιά με παρόμοιο ύψος, καθιστώντας τον υπολογισμό του τομέα περίπλοκο. Όπως απεικονίζεται και στο σχήμα 46 για τρεις τομείς, ο τομέας δύο περιλαμβάνει περισσότερα ορατά κελιά, και κατά συνέπεια, απαιτεί πιο πολύ χρόνο για τον υπολογισμό του από τους υπόλοιπους τομείς. Αντίθετα, στο σχήμα 46 για δύο τομείς, τα ορατά σημεία διαχωρίζονται σε ίσα κομμάτια, μοιράζοντας έτσι καλύτερα το φόρτο του κάθε τομέα. Κάθε νέα διεργασία βελτιώνει τον χρόνο υπολογισμού της πιο αργής διεργασίας, μέχρι το σημείο που η βελτίωση αυτή θα είναι αμελητέα.

4.1.2 Συμπεράσματα

Ο κώδικας που αναπτύχτηκε παρουσιάζει πλεονεκτήματα σε σύγκριση με παρόμοιες υλοποιήσεις για εφαρμογές που επεξεργάζονται μεγάλες εκτάσεις εδάφους. Ο αλγόριθμος είναι ιδανικός για τον υπολογισμό μεγάλων περιοχών, αφού διασπάει το πρόβλημα σε κομμάτια μοιράζοντας έτσι και τον φόρτο του υπολογισμού. Η τεχνική της παραλληλοποίησης είναι ιδιαίτερα εξυπηρετική, όταν ο αλγόριθμος εκτελείται από ένα σύνολο υπολογιστών μικρότερης επεξεργαστικής ισχύς, καθιστώντας τους πιο αποδοτικούς από έναν πολύ πιο ισχυρό υπολογιστή. Ένα επιπλέον πλεονέκτημα είναι η υψηλή ακρίβεια του αλγορίθμου, αφού υπολογίζει με διαφορετικό τρόπο, τόσο το ύψος για τα γεγονότα εισόδου και εξόδου, όσο και για τις ενδιάμεσες θέσεις των κελιών. Η ακρίβεια είναι από τους πιο βασικούς παράγοντες για τις περισσότερες εφαρμογές του αλγορίθμου και είναι απαραίτητη ιδιαίτερα σε εφαρμογές αισθητικής αξίας. Η βελτίωση του αποτελέσματος σε σχέση με τον βασικό αλγόριθμο του Van Kreveld είναι εμφανής, όπως μπορεί να παρατηρηθεί και στο σχήμα 44.

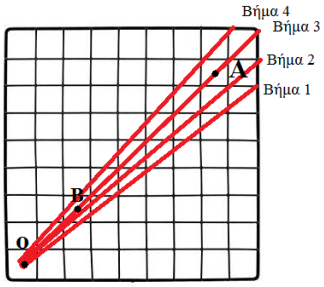
Οι μετατροπές που έγιναν στον αλγόριθμο δημιουργούν και προβλήματα σε ορισμένες περιπτώσεις. Αρχικά, ο διαχωρισμός που απαιτείται για την παραλληλοποίηση οδηγεί σε μικρές καθυστερήσεις για τον υπολογισμό κελιών μεταξύ δύο κομματιών και για την διαδικασία του διαχωρισμού. Επομένως, ο αλγόριθμος δεν είναι ο βέλτιστος δυνατός όσον αφορά στον υπολογισμό μικρών περιοχών. Επιπρόσθετα, οι υπολογισμοί που γίνονται για τα κελιά έχουν δραστική επίδραση στη δομή του αλγορίθμου. Ο αλγόριθμος θυσιάζει αρκετή από την απόδοση του για να προσφέρει βελτίωση στην ακρίβεια,. Ταυτόχρονα, λόγω τον αλλαγών που απαιτούνται, δεν είναι δυνατή η εύκολη μετατροπή του για οποιαδήποτε άλλη προσέγγιση που αφορά τη μέτρηση του ύψους, καθώς δεν γίνεται πλέον η χρήση του δέντρου, που είναι βασικό κομμάτι του αρχικού αλγορίθμου.

Ο αλγόριθμος, που υλοποιήθηκε, δίνει έμφαση στην ακρίβεια των μετρήσεων, αφού κρίνεται απαραίτητη στις εφαρμογές που χρησιμοποιείται η ανάλυση οπτικού πεδίου [40]. Η ακρίβεια του αλγορίθμου αποτελεί βασικό στόχο του αλγορίθμου, διότι μικρές αλλαγές είναι πιθανό να αλλοιώσουν σημαντικά την ορατότητα της θέσης που εξετάζεται. Επομένως, στον αλγόριθμο έγινε χρήση της τεχνικής της παρεμβολής, η οποία εξασφαλίζει την ακρίβεια των αποτελεσμάτων του αλγορίθμου, χωρίς να απαιτεί τη χρήση υψομετρικού μοντέλου με μικρότερα κελιά. Ταυτόχρονα, καθώς η ανάλυση οπτικού πεδίου χρησιμοποιείται σε ένα ευρύ φάσμα εφαρμογών, πολλές από τις οποίες αναφέρονται και στο κεφάλαιο 1, συχνά συνδυάζεται με άλλες τεχνικές ανάλογα με τις απαιτήσεις του έργου στο οποίο εφαρμόζεται. Κατά συνέπεια, η γλώσσα προγραμματισμού στην οποία υλοποιήθηκε ο αλγόριθμος οφείλει να ανταποκρίνεται στις ανάγκες για την πλειοψηφία των έργων αυτών. Έτσι, η χρήση της Python επιλέχτηκε λόγω της ευελιξίας που παρέχει [41]. Οι βιβλιοθήκες που είναι διαθέσιμες στην Python την καθιστούν ιδανική για χρήση σε μεγάλη ποικιλία εφαρμογών, καθώς μπορούν να εκφράσουν σύνθετους αλγόριθμους πολύ γρήγορα. Πιο συγκεκριμένα, η Python υποστηρίζει βιβλιοθήκες για την επεξεργασία γεωχωρικών δεδομένων, όπως η shapely, η rasterio και η gdal οι οποίες διευκολύνουν την ανάπτυξη σημαντικών εφαρμογών, όπως της Open Data Cube. Η αύξηση όμως της απόδοσης που παρέχει στον προγραμματιστή η Python, έχει ως αποτέλεσμα τη μείωση της απόδοσης του αλγορίθμου. Το συγκεκριμένο πρόβλημα μπορεί να μετριασθεί μέσω της παράλληλης εκτέλεσης του αλγορίθμου, όταν γίνει χρήση αρκετών διεργασιών. Συνολικά, ο αλγόριθμος διατηρεί μια ισορροπία μεταξύ απόδοσης και χρηστικότητας, και μπορεί να προσαρμοστεί περαιτέρω, ώστε να καλύψει οποιοδήποτε πρόβλημα προκύψει μελλοντικά, αφού ο πηγαίος κώδικας διατίθεται ελεύθερα (open source).

4.2 Προβλήματα που αντιμετωπίστηκαν κατά την υλοποίηση

Όπως είναι φυσικό σε ένα έργο τέτοιου μεγέθους κατά την υλοποίηση του αλγορίθμου υπήρξαν προβλήματα που δυσκόλεψαν την πρόοδο της εργασίας. Καθώς η εργασία αποτελείται μόνο από λογισμικό τα προβλήματα που προέκυψαν δεν αποτελούσαν εμπόδια στην λειτουργία του κώδικα, αλλά προκαλούσαν σημαντικές καθυστερήσεις στην εκτέλεση. Παρακάτω αναλύονται τα προβλήματα και ο τρόπος που αντιμετωπίστηκαν.

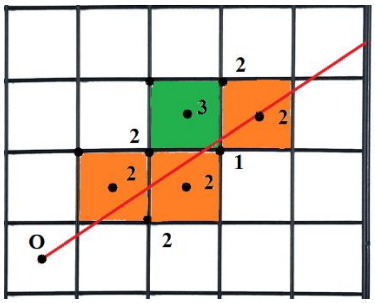
Ένα από τα πρώτα προβλήματα που προέκυψε αφορούσε την περιστροφή της ημιευθείας που περιγράφεται από τον αλγόριθμο του Van Kreveld. Η ημιευθεία περιστρέφεται γύρω από τον παρατηρητή και όταν το κελί τέμνεται στα κατάλληλα σημεία συμβαίνουν τα γεγονότα του κελιού. Στην αρχική μορφή του κώδικα η περιστροφή πραγματοποιούνταν από έναν βρόχο που σε κάθε επανάληψη αύξανε τις μοίρες περιστροφής της ημιευθείας, και υπολόγιζε για κάθε γωνία σε ποιο σημείο τέμνει τα κελιά χρησιμοποιώντας την εξίσωση της ευθείας. Καθώς είναι απίθανο η γωνία που δίνεται από το βρόχο να συναντά ακριβώς τα σημεία που συμβαίνουν τα γεγονότα κάθε κελιού η μέθοδος αυτή απαιτούσε για κάθε κελί να ελέγχεται τόσο για παραλήψεις, όσο και για διπλότυπα γεγονότα. Αυτή η υλοποίηση δημιουργούσε σημαντική καθυστέρηση στην εκτέλεση του αλγορίθμου, ειδικά για μεγάλα DEM καθώς, όσο μεγαλώνει η απόσταση από τον παρατηρητή, τόσο πρέπει να μικρύνει το βήμα του βρόχου για να καλύψει τα πιο απομακρυσμένα κελιά.



Σχήμα 47 Απεικόνιση των κελιών που τέμνονται από κάθε γωνία περιστροφής, τα κελιά που βρίσκονται κοντά στον παρατηρητή τέμνονται πολλαπλές φόρες ενώ τα πιο απομακρυσμένα μόνο μια φορά.

Όπως φαίνεται και στο σχήμα 47, το κελί *B*, επειδή είναι πιο κοντά στον παρατηρητή, τέμνεται από την ημιευθεία και στα τέσσερα βήματα του βρόχου ενώ το κελί A τέμνεται μόνο από τα βήματα 2-4. Κατά συνέπεια, αυξάνονται οι επαναλήψεις του βρόχου, ενώ ταυτόχρονα για τα κελιά κοντά στο παρατηρητή επαναλαμβάνονται περιττές μετρήσεις. Η λύση στο πρόβλημα δόθηκε όταν άλλαξε ο τρόπος προσέγγισης του προβλήματος. Όπως αναλύεται και στο κεφάλαιο 3.3. Στον παρών αλγόριθμο υπολογίζονται η γωνία, στην οποία συμβαίνουν τα γεγονότα για το κάθε κελί και στη συνέχεια τα γεγονότα ταξινομούνται κατά γωνία, ώστε να επιτευχθεί το ζητούμενο αποτέλεσμα χωρίς περιττούς υπολογισμούς.

Ένα ακόμα πρόβλημα που δημιουργήθηκε αφορά το δυαδικό δέντρο που περιγράφει ο αλγόριθμος του Van Kreveld. Η Python δεν έχει κάποια επίσημη βιβλιοθήκη που να χρησιμοποιείται ευρέως για την δημιουργία δέντρων. Επομένως, δημιουργήθηκε για τον αλγόριθμο ένα δέντρο χρησιμοποιώντας κλάσεις για το δέντρο και τους κόμβους του, όπως έχει υλοποιηθεί και σε άλλες εργασίες [42] και κατάλληλες συναρτήσεις για τις περιστροφές που απαιτούνται μετά τις εισαγωγές και τις εξαγωγές κόμβων. Αποτέλεσμα ήταν, όχι μόνο να είναι πιο δύσχρηστη η χρήση της δομής δεδομένων, αφού οποιαδήποτε επεξεργασία της απαιτούσε να γραφτεί από την αρχή κατάλληλη συνάρτηση, αλλά ταυτόχρονα να επιβραδύνει και τον αλγόριθμο λόγω των περιστρόφων που απαιτούνται για να διατηρηθεί η ισορροπία του δέντρου. Ο βασικότερος λόγος, όμως, που καθιστά το δέντρο περιττό, είναι οι παραλλαγές που εφαρμόστηκαν στον βασικό αλγόριθμο του Van Kreveld. Ο ρόλος του δέντρου ήταν να οργανωθούν τα ύψη των κελιών, ώστε με μία αναζήτηση του δέντρου για την εύρεση του κελιού, να παρέχει και το μέγιστο ύψος έως το κελί που αναζητήθηκε. Με τον τρόπο αυτό επιταχύνονταν σημαντικά ο υπολογισμός των γεγονότων κέντρου. Στα γεγονότα κέντρου η σύγκριση του ύψους του κελιού με το μέγιστο ύψος επέστρεφε το αποτέλεσμα για την ορατότητα με μια μόνο σύγκριση. Το πρόβλημα που δημιουργείται στην περίπτωση του αλγορίθμου που υλοποιήθηκε είναι πως τα κελιά δεν έχουν ένα σταθερό ύψος για όλη την περιοχή που καλύπτουν, αφού το ύψος τους υπολογίζεται με τη χρήση της τεχνικής της παρεμβολής. Συνεπώς, ακόμα και όταν πραγματοποιηθεί η ανάλογη σύγκριση, το ύψος που είναι αποθηκευμένο αποτελεί το μέγιστο πιθανό ύψος για το κελί. Επομένως, το κελί με το μέγιστο ύψος της λίστας αυτής, δεν θα είναι απαραίτητα το ίδιο με το κελί που έχει το μέγιστο αφού πραγματοποιηθεί λεπτομερής μέτρηση.



Σχήμα 48 Παράδειγμα τοπικού μέγιστου που οδηγεί σε λάθος συμπέρασμα. Το πράσινο κελί έχει τοπικό μέγιστο το 3 αλλά η τιμή του κελιού όταν υπολογιστεί με ακρίβεια τείνει στο 1.

Όπως φαίνεται παραπάνω, τα κελιά με πορτοκαλί χρώμα έχουν μέγιστο ύψος δυο, ενώ το πράσινο κελί έχει μέγιστο ύψος τρία. Αν δεν πραγματοποιηθούν ακριβείς μετρήσεις, όπως συμβαίνει στην δομή δέντρου του αλγορίθμου του Van Kreveld, το πράσινο κελί θα θεωρηθεί ορατό. Με μια πιο καλή προσέγγιση του ύψους του είναι όμως ξεκάθαρο ότι το κελί τείνει στο ένα και όχι στο τρία, και επομένως δεν είναι ορατό. Άρα, για κάθε γραμμή ορατότητας είναι

απαραίτητοι οι υπολογισμοί για όλα τα κελιά της γραμμής ορατότητας με μέγιστο ύψος μεγαλύτερο αυτού του κελιού που εξετάζεται. Για τους παραπάνω λόγους, το δυαδικό δέντρο αντικαταστάθηκε από μια λίστα για την οποία εφαρμόζεται δυαδική αναζήτηση μόνο για την εύρεση του κελιού.

Προβλήματα προέκυψαν όμως και κατά την μετατροπή του σειριακού αλγορίθμου σε παράλληλο, καθώς τα γεγονότα του αλγόριθμου του Van Kreveld εξαρτώνται από τη σειρά εκτέλεσης τους. Επομένως, ο διαχωρισμός των κομματιών πρέπει να γίνει με βάση την γωνία που συμβαίνουν για να διατηρηθεί η συνεχεία των γεγονότων. Επειδή όμως η ταξινόμηση κατά γωνία πραγματοποιείται αφού ολοκληρωθεί η σάρωση αρχικά, ο διαχωρισμός του αλγορίθμου στις διεργασίες γινόταν σε δύο σημεία. Πριν την σάρωση χωριζόταν το DEM σε ίσα κομμάτια τα οποία μετά τον υπολογισμό της σάρωσης συγχωνεύονταν και ξαναμοιραζόταν σε κομμάτια για τον υπολογισμό της ορατότητας. Σε αυτή την προσέγγιση δημιουργείται καθυστέρηση ανάλογη με το πλήθος των διεργασιών που χρησιμοποιούνται, επειδή απαιτείται επικοινωνία μεταξύ των διεργασιών σε δύο σημεία. Για να αντιμετωπιστεί αυτό το ζήτημα, δημιουργήθηκε η συνάρτηση που περιγράφεται στο κεφάλαιο 3.5.1. Η συνάρτηση αυτή διαχωρίζει τα κομμάτια της σάρωσης με βάση τις γωνίες τους, ώστε να μην χρειάζεται επικοινωνία των διεργασιών μέχρι το τέλος του αλγορίθμου, αφού η σάρωση και ο έλεγχος ορατότητας εξετάζει τα ίδια κομμάτια.

Ένα ακόμα πρόβλημα που δυσκόλεψε την διόρθωση του αλγορίθμου αφορά τον διαχωρισμό των κομματιών για την παράλληλη επεξεργασία και σχετίζεται με τον αριθμό των γεγονότων του κάθε κελιού. Αν ο αλγόριθμος κατά τη σάρωση ή την αρχικοποίηση του αλγορίθμου δεν εκτελέσει κάθε γεγονός των κελιών ακριβώς μια φορά, δημιουργείται αστοχία στο αποτέλεσμα ή διακόπτεται ολόκληρος ο αλγόριθμος. Η διακοπή της λειτουργίας συμβαίνει αν το γεγονός εισόδου ενός κελιού δεν πραγματοποιηθεί, ή αν συμβεί το γεγονός εξόδου για το ίδιο κελί περισσότερες από μία φόρες. Στην περίπτωση αυτή, κατά την εκτέλεση του γεγονότος εξόδου δεν θα βρεθεί το κελί στη λίστα ορατότητας και ο αλγόριθμος θα διακόψει τη λειτουργία του. Αστοχία στο αποτέλεσμα δημιουργείται στην περίπτωση που πραγματοποιηθούν δύο γεγονότα εισόδου ή δεν εκτελεστεί το γεγονός εξόδου για ένα κελί. Τότε, το κελί θα παραμείνει για ολόκληρη την περιστροφή της ημιευθείας στην λίστα ορατότητας, επηρεάζοντας το αποτέλεσμα. Για το πρόβλημα αυτό, ακόμα και όταν αναγνωρίστηκε η αιτία του σφάλματος, ήταν χρονοβόρα η επίλυση του, αφού η αστοχία στο αποτέλεσμα γίνεται αντιληπτή στα επόμενα κελιά, μόνο όταν πραγματοποιείται η απεικόνιση του αποτελέσματος, καθώς δεν εμπόδιζε την εκτέλεση του αλγορίθμου. Επομένως, δε διακρίνεται το πρόβλημα στο σημείο που δημιουργείται και ο μοναδικός τρόπος να βρεθεί το σφάλμα είναι με δοκιμές. Σε πολλές

περιπτώσεις το σφάλμα μπορεί να μην γίνει ορατό αν αφορά κελί με πολύ χαμηλό ύψος, που δεν επηρεάζει σημαντικά την ορατότητα.

Τέλος, προσπάθεια έγινε στον αλγόριθμο να συμπεριληφθεί και χρήση νημάτων για την επίτευξη παραλληλισμού σε συνδυασμό με το MPI [28] . Το πρόβλημα με την χρήση νημάτων στην Python σχετίζεται με τον μεταφραστή της. Ο μεταφραστής της Python για να πετύχει την ασφαλή χρήση των νημάτων, τα περιορίζει, ώστε να εκτελούνται ένα τη φορά. Αποτέλεσμα είναι η χρήση τους, είτε να μην αλλάζει τον χρόνο εκτέλεσης, είτε να τον επιβραδύνει [25]. Αυτό συμβαίνει καθώς τα νήματα, όχι μόνο εκτελούνται σειριακά, αλλά πραγματοποιούνται και εναλλαγές μεταξύ τους, καθυστερώντας έτσι τον αλγόριθμο.

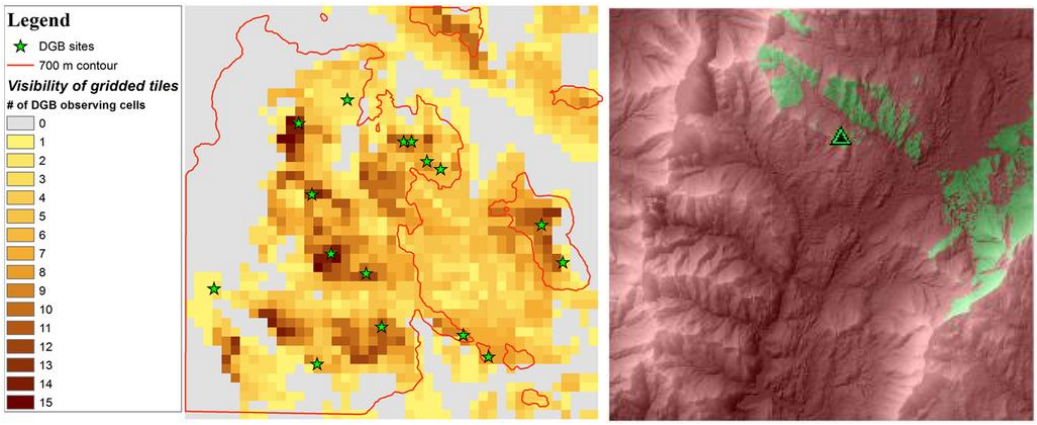
4.3 Μελλοντικές επεκτάσεις

Όπως αναφέρεται και παραπάνω η ανάλυση οπτικού πεδίου χρησιμοποιείται για ένα μεγάλο φάσμα εφαρμογών. Κατά συνέπεια είναι αναμενόμενο να χρειαστούν επεκτάσεις στον παρόντα αλγόριθμο, που θα εξυπηρετούν τις απαιτήσεις κάθε εφαρμογής. Επομένως, είναι σημαντικό ο αλγόριθμος να προσαρμόζεται εύκολα στις ανάγκες κάθε έργου που θα κληθεί να αναλάβει. Για αυτό το λόγο η παρούσα υλοποίηση διαχωρίζει τους υπολογισμούς σε ανεξάρτητες μεταξύ τους συναρτήσεις. Κάθε συνάρτηση μπορεί να αλλάξει αρκεί να παρέχει την ίδια έξοδο, δίνοντας έτσι την δυνατότητα για εύκολη αλλαγή η επέκταση οποιουδήποτε μέρους του αλγορίθμου.

Μία τεχνική που μπορεί να πετύχει την βελτίωση της απόδοσης του αλγορίθμου αφορά την χρήση τεχνικών παραλληλοποίησης με την χρήση της κάρτας γραφικών. Το κομμάτι που μπορεί να υλοποιηθεί με αυτή την τεχνική είναι αυτό της σάρωσης, καθώς αποτελείται από υπολογισμούς που είναι ανεξάρτητοι ο ένας από τον άλλο. Η υλοποίηση αυτή θα αποτελούσε μια υβριδική παραλληλοποίηση του αλγορίθμου, καθώς ο υπολογισμός της ορατότητας χρειάζεται να γίνει με συγκεκριμένη σειρά και δεν είναι ιδανικός για τη μέθοδο αυτή.

Υπάρχουν όμως και επεκτάσεις γενικού τύπου που θα εξυπηρετούσαν τους χρήστες. Ο παρών αλγόριθμος δίνει τιμή μηδέν ή ένα, ανάλογα με το αν υπάρχει ορατότητα ή όχι σε ένα σημείο. Η επέκταση αφορά τον υπολογισμό του απαραίτητου ύψους, ώστε να έχει ο παρατηρητής ορατότητα σε ένα σημείο. Η έξοδος του αλγορίθμου χρειάζεται να αλλάξει από μηδέν και σε έναν αριθμό που αντιπροσωπεύει το απαραίτητο ύψος, ώστε το σημείο να είναι ορατό από τον παρατηρητή. Αυτή η επέκταση θα εξυπηρετούσε χρήστες που εξετάζουν την ορατότητα μεταξύ δύο σημείων να βρουν ευκολότερα το απαιτούμενο ύψος, ώστε να έχει ο παρατηρητής ορατότητα στο σημείο.

Μία ακόμα επέκταση του αλγορίθμου αποτελεί και ο υπολογισμός της συσσωρευτικής ανάλυσης οπτικού πεδίου. Η συσσωρευτική ανάλυση οπτικού πεδίου αποτελεί το σύνολο των υπολογισμών της ανάλυσης οπτικού πεδίου για όλα τα στοιχεία του DEM που μελετάται. Πιο συγκεκριμένα, εκτελείται ανάλυση οπτικού πεδίου με παρατηρητή το κάθε στοιχείο της εικόνας. Κάθε εκτέλεση αποθηκεύει στο κελί με τις συντεταγμένες του παρατηρητή, τον αριθμό των κελιών στα οποία έχει ορατότητα. Αποτέλεσμα της συσσωρευτικής ανάλυση οπτικού πεδίου είναι κάθε κελί να έχει τιμή από μηδέν έως το πλήθος των κελιών του DEM. Η μέθοδος αυτή δείχνει τα σημεία με την βέλτιστη ορατότητα στην εικόνα και χρησιμοποιείται για εφαρμογές, όπως η τοποθέτηση πύργων τηλεπικοινωνιών.



Σχήμα 49 Αριστερά παράδειγμα συσσωρευτικής ανάλυσης οπτικού πεδίου και δεξιά παράδειγμα απλής ανάλυσης οπτικού πεδίου[17].

Η συσσωρευτική ανάλυση οπτικού πεδίου απεικονίζεται στην αριστερά στο σχήμα 49. Κάθε κελί έχει χρώμα ανάλογο με το πλήθος των σημείων στα οποία το κελί έχει ορατότητα. Η απλή ανάλυση οπτικού πεδίου που απεικονίζεται δεξιά έχει ως παρατηρητή το τρίγωνο και αποτελείται από δύο χρώματα, καθένα από τα οποία δείχνει αν υπάρχει ορατότητα από τον παρατηρητή στο κελί.

[17] Πηγή του σχήματος είναι η ιστοσελίδα https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0112191. Η επίσκεψη έγινε στις 11/5/2020

Παράρτημα

Οδηγίες εγκατάστασης και χρήσης του αλγορίθμου

Για την εκτέλεση του αλγορίθμου απαιτείται η εγκατάσταση της Python 3 από την ιστοσελίδα https://www.python.org/download/releases/3.0/ αλλά και η χρήση ενός αρχείου που περιέχει το ψηφιακό υψομετρικό μοντέλο μιας περιοχής. Για να εκτελεστεί ο κώδικα είναι απαραίτητες οι παρακάτω βιβλιοθήκες:

1. Math
2. PIL
3. Rasterio
4. Time
5. Mpi4py

Η εγκατάσταση των βιβλιοθηκών πραγματοποιείται χειροκίνητα με τη χρήση της εντολής pip install και το όνομα της βιβλιοθήκης.

Επιπλέον, είναι απαραίτητο το αρχείο με το υψομετρικό μοντέλο της περιοχής. Το αρχείο αυτό είναι της μορφής TIFF (Tagged Image File Format) και περιέχει το υψόμετρο του εδάφους για το κέντρο του κάθε σημείου της εικόνας. Το αρχείο πρέπει να βρίσκεται στο φάκελο που περιέχει τον κώδικα και να έχει το όνομα «DSM_Test.tif». Εναλλακτικά, μπορεί να γίνει αλλαγή στο όνομα του αρχείου με τις κατάλληλες αλλαγές στη γραμμή 52 του κώδικα, όπως φαίνεται παρακάτω.

```
51    #tif file with heights of every pixel
52    with rasterio.open('DSM_Test.tif', 'r+') as r:
```

Σχήμα 50 Απεικόνιση της γραμμής και του ονόματος του αρχείου που περιέχει το ψηφιακό υψομετρικό μοντέλο και χρησιμοποιείται ως είσοδος του αλγορίθμου.

Για να διευκρινιστεί η θέση και το ύψος του παρατηρητή μπορούν να τροποποιηθούν οι συντεταγμένες για τη λίστα vp, η οποία βρίσκεται στη γραμμή 30 του αλγορίθμου, όπως φαίνεται στο παρακάτω σχήμα (Σχήμα 51).

```
29    #viewpoint location
30    vp = [2047 ,785,0]#x,y,height
```

Σχήμα 51 Απεικόνιση της γραμμής και του ονόματος της λίστας που περιέχει τις συντεταγμένες του παρατηρητή που χρησιμοποιείται ως είσοδος του αλγορίθμου.

Όταν ολοκληρωθούν οι παραπάνω οδηγίες ο αλγόριθμος εκτελείται με την εντολή «mpirun –n *x* python example.py», όπου *x* είναι ο αριθμός των επεξεργαστών που θα χρησιμοποιηθούν και example.py είναι το όνομα που αρχείου με τον αλγόριθμο.

Περιγραφή λειτουργίας του κώδικα

Ο κώδικας που υλοποιήθηκε χωρίζεται σε τρία αρχεία, καθένα από τα οποία περιέχει συναρτήσεις που καλύπτουν διαφορετικά τμήματα του αλγορίθμου.

Το πρώτο αρχείο ονομάζεται par1.py, και αποτελεί τη βάση του αλγορίθμου. Οι τιμές που απαιτούνται για την εκτέλεση του αρχείου είναι το κελί που βρίσκεται ο παρατηρητής με τις συντεταγμένες του (χ,ψ), και το ύψος του παρατηρητή. Το αρχείο par1.py συντονίζει τις διεργασίες, και καλεί τις συναρτήσεις από τα άλλα αρχεία, ώστε να γίνει η σάρωση με της συναρτήσεις που βρίσκονται στο αρχείο par2.py, και ο υπολογισμός της ορατότητας με της συναρτήσεις που βρίσκονται στο αρχείο par3.py. Επομένως, περιέχει εκτός από τη main μόνο τη συνάρτηση split. Η έξοδος του αρχείου γίνεται κατά τον τερματισμό του αλγορίθμου, και είναι ένας πίνακας ο οποίος δίνει τιμή ένα στα κελιά που είναι ορατά και μηδέν σε αυτά που δεν είναι ορατά.

Η συνάρτηση split() είναι η πρώτη συνάρτηση που εκτελείται. Είσοδος της είναι ο αριθμός των κελιών που περιέχει το αρχείο TIFF. Η συνάρτηση υπολογίζει το μερίδιο κάθε διεργασίας ανάλογα με τον αριθμό των διεργασιών, και έχει ως έξοδο μια λίστα με τις γωνίες στις οποίες αρχίζει το κομμάτι που θα υπολογίσει η κάθε διεργασία. Η διαδικασία που ακολουθεί καταγράφεται στο κεφάλαιο 3.5.1.

Η συνάρτηση που εκτελείται στη συνέχεια, είναι η rotation(), η οποία ανήκει στο δεύτερο αρχείο με όνομα par2.py. Η συνάρτηση αυτή δέχεται ως είσοδο το αρχείο TIFF(Tagged Image File Format), που περιέχει το υψόμετρο του εδάφους για το κέντρο του κάθε εικονοστοιχείου και τις τιμές από τη λίστα που προέκυψε από τη split(). Η συνάρτηση υπολογίζει τη σάρωση της περιοχής για τον υπολογισμό των δεδομένων των κελιών, όπως περιγράφεται στο κεφάλαιο 3.3. Έξοδος της είναι η λίστα σάρωσης και η λίστα ορατότητας μετά τη αρχικοποίηση της.

Μέσα από τη συνάρτηση rotation καλείται αρχικά η συνάρτηση gridx() με ορίσματα τη γωνία εκκίνησης και τη γωνία τερματισμού του κομματιού που έχει αναλάβει η διεργασία. Ακολουθεί η gridy(), η οποία σε συνδυασμό με τη gridx() καταγράφει τις συντεταγμένες για τα καλιά που θα εξεταστούν. Η τεχνικές που χρησιμοποιούνται περιγράφονται στο κεφάλαιο 3.4.1. Τα ορίσματα της gridy() είναι τα ίδια με της gridx(), ενώ περιλαμβάνει ακόμα τη τιμή χ των κελιών που εξετάζονται, τη μεταβλητή currentpace που είναι το βήμα για τον βρόχο που χρησιμοποιείται, και τις fs και flag. Οι δύο τελευταίες αποτελούν μεταβλητές σημαίες που διευκρινίζουν ποια περίπτωση έχει προκύψει από αυτές που αναφέρονται στο κεφάλαιο 3.5.2.

Έξοδος της συνάρτησης είναι μια λίστα με τις συντεταγμένες (χ,ψ) των κελιών που περιέχει το κομμάτι που σαρώνεται.

Στη συνέχεια, εκτελείται η συνάρτηση pixelcalc(), η οποία έχει ως είσοδο τις συντεταγμένες (χ,ψ) του κομματιού που υπολογίστηκε στη gridy(), τα δεδομένα του αρχείου TIFF και τις γωνίες εκκίνησης και τερματισμού του κομματιού. Η pixelcalc() υπολογίζει τα σημεία, στα οποία συμβαίνουν τα γεγονότα του κελιού με βάση το τεταρτημόριο, στο οποίο βρίσκονται, όπως περιγράφεται στο κεφάλαιο 3.3.1. Έξοδος της συνάρτησης είναι η λίστα με τα γεγονότα του κάθε κελιού.

Η τελευταία συνάρτηση που εκτελείται μέσα στη rotation() είναι η initlist(), η οποία δέχεται ως είσοδο τη λίστα σάρωσης και τη γωνία εκκίνησης. Η συνάρτηση αυτή πραγματοποιεί την αρχικοποίηση της λίστας ορατότητας. Σε αυτό το σημείο τελειώνει η κλήση της visibility() και η ροή επιστρέφει στη main().

Ακολουθεί, η συνάρτηση visibility(), η οποία βρίσκεται στο αρχείο par3.py, που πραγματοποιεί τον έλεγχο ορατότητας. Η visibility() έχει ως είσοδο τη λίστα σάρωσης, την αρχικοποιημένη λίστα ορατότητας, τη γωνία εκκίνησης και τερματισμού του κομματιού, τη θέση και το ύψος του παρατηρητή. Στόχος της συνάρτησης αυτής είναι εκτέλεση των γεγονότων, με αποτέλεσμα τον υπολογισμό της ορατότητας των κελιών, όπως περιγράφεται στο κεφάλαιο 3.4.3. Έξοδος της επομένως, είναι μια λίστα, στην οποία για κάθε κελί παίρνει τιμή ένα αν υπάρχει ορατότητα στο συγκεκριμένο κελί.

Μέσω της συνάρτησης visibilty() καλείται για τα γεγονότα εισόδου η συνάρτηση maxbisearch() με ορίσματα έναν πίνακα και μια μεταβλητή, και επιστρέφει τη θέση στην οποία η επόμενη μεταβλητή έχει μεγαλύτερη τιμή, ενώ ταυτόχρονα η προηγούμενη μεταβλητή έχει μικρότερη τιμή. Αντίστοιχα, τα γεγονότα κέντρου χρησιμοποιούν τη bisearch(), η οποία πραγματοποιεί με τα ίδια ορίσματα δυαδική αναζήτηση, και επιστρέφει τη θέση της μεταβλητής που δέχεται ως είσοδο. Με τον τερματισμό της συνάρτησης visibility(), η ροή του αλγορίθμου επιστρέφει στην main(), οπού συγκεντρώνεται το αποτέλεσμα του αλγορίθμου και στη συνέχεια ο αλγόριθμος τερματίζεται.

Βιβλιογραφία

1. **Kreveld, Marc Van.** *Variations on sweep algorithms: Efficient computation of extended viewsheds and classifications.* s.l. : In Proc. Symposium on Spatial Data Handling, 1996. σσ. 15-27.

2. **M. Sanfourche, B. Le Saux, A. Plyer and G. Le Besnerais.** *Environment mapping & interpretation by drone.* Lausanne : 2015 Joint Urban Remote Sensing Event (JURSE), 2015. σσ. 1-4.

3. **Heike Christian.** *Crowdsourcing geospatial data.* Institutfür Photogrammetrie und Geo Infomation, Leibniz Universität Hannover,Germany : ISPRS Journal of Photogrammetry and Remote Sensing,Volume 65, Issue 6, 2010. σσ. 550-557.

4. **Marcus V. A. Andrade, Salles V. G. Magalhaes, Mirella A. Magalhaes, Randolph W.Franklin, and Barbara M. Cutler.** *Efficient viewshed computation on terrain in external memory.* s.l. : GeoInformatica 15, 2011. σσ. 381-397.

5. **Wheatley David.** *Cumulative viewshed analysis: a GISbased method for investigating intervisibility, and its archaeological application.* s.l. : G Lock and Z Stancic eds GIS and Archaeology: a European Perspective, 1995. σσ. 171 -186.

6. **Anna Petrasova, Brendan Harmon, Vaclav Petras, Payam Tabrizian, Helena Mitasova.** *Viewshed Analysis. Tangible Modeling with Open Source GIS.* s.l. : Springer International Publishing, 2015. σσ. 77–82.

7. **Santos-Berbel, Maria Castro & César De.** *Spatial analysis of geometric design consistency and road sight distance.* s.l. : International Journal of Geographical Information Science 29, 2015. σσ. 1-14.

8. **Joel M. Caplan, Leslie W. Kennedy & Gohar Petrossian.** *Police-monitored CCTV cameras in Newark, NJ: A quasi-experimental test of crime deterrence.* s.l. : Journal of Experimental Criminology volume 7, 2011. σσ. 255–274.

9. **S. Neisha, Panel Jeffrey, Kodysha Olufemi,Omitaomuab Budhendra, Bhaduria Bradley.** *Methodology for estimating solar potential on multiple building rooftops for photovoltaic systems.* Oak Ridge, TN, USA : Sustainable Cities and Society 8, 2013.

10. **J. Edan, O. Idowut & I. Zango.** *Viewshed analisys of mtn telecommunication network in jimeta metropolis.* Yola : TJPRC Publication, 2013.

11. **Shitai Baoa, Ningchuan Xiaob, Zehui Laia, Heyuan Zhangc, Changjoo Kim.** *Optimizing watchtower locations for forest fire monitoring using location models.* s.l. : Fire Safety Journal,Volume 71, 2015. σσ. 100-109.

12. **S. Topouzi, S. Soetens, A. Gkiourou & A. Sarris.** *The Application of Viewshed Analysis in Greek Archaeological Landscape.* Crete, Greece : 6th Annual meeting of the European Association of Archaelogists, 2000. σσ. 1-15.

13. **David Izraelevitz.** *A Fast Algorithm for Approximate Viewshed Computation.* s.l. : Photogrammetric Engineering and Remote Sensing 69, 2003.

14. **Jeffrey Wilson, Greg Lindsey & Gilbert Liu.** *Viewshed characteristics of urban pedestrian trails.* s.l. : Journal of Maps 4, 2008. σσ. 108-118.

15. **Nobuhiko Yoshimuraa, Tsutom Hiura.** *Demand and supply of cultural ecosystem services: Use of geotagged photos to map the aesthetic value of landscapes in Hokkaido.* Hokkaido, Japan : Ecosystem Services 24, 2017. σσ. 68-78.

16. **Steve Carvera, Alexis Comberb, Rob McMorranc, Steve Nuttera.** *A GIS model for mapping spatial patterns and distribution of wild land in Scotland.* s.l. : Landscape and Urban Planning 104, 2012. σσ. 395-409.

17. **Runnström, Rannveig Ólafsdóttir & Micael C.** *How Wild is Iceland? Wilderness Quality with Respect to Nature-based Tourism.* Reykjavık, Iceland : Tourism Geographies 13), 2011. σσ. 280-298.

18. **Eric J. Maichak, Krysten L. Schuler.** *Applicability of Viewshed Analysis to Wildlife Population Estimation.* s.l. : American Midland Naturalist 152, 2004. σσ. 277-285.

19. **Peter L. Guth, P.** *Incorporating vegetation in viewshed and Line-of Sight algorithms.* s.l. : ASPRS/MAPPS 2009 Specialty Conference, 2009.

20. **Denis J. Dean, D.** *Improving the accuracy of forest viewsheds using triangulated networks and the visualpermeability method.* s.l. : Canadian Journal of Forest Research, 2011. σσ. 969-977.

21. **Gokhan, Yilmaz.** *Acceleration of Line of Sight Analysis Algorithms with Parallel Programming.* s.l. : Conference: USMOS 2017, 2017.

22. **W. Franklin, C. Ray and S. Mehta.** *Geometric Algorithms for Siting of Air Defense Missile Batteries.* s.l. : A Research Project for Battle, no. 2756, 1994.

23. **Aran J. Cauchi-Saunders, Ian J. Lewis.** *GPU enabled XDraw viewshed analysis.* s.l. : Journal of Parallel and Distributed Computing 84, 2015.

24. **Chaulio R. Ferreira, Marcus V. A. Andrade,Salles V. G. Magalhães, W. R. Franklin, Guilherme C. Pena.** *A Parallel Algorithm for Viewshed Computation on Grid Terrains.* s.l. : Vol. 5 No. 2 (2014): JOURNAL OF INFORMATION AND DATA MANAGEMENT,No. 2,Vol. 5 , 2014.

25. **Ami Marowka.** *On parallel software engineering education using Python.* s.l. : Education and Information Technologies 23, 2017.

26. **R. Chaulio, Ferreira, V. Marcus, A. Andrade, V. Salles, G. Magalhes, W. R. Franklin, Guilherme C. Pena.** *A Parallel Sweep Line Algorithm for Visibility Computation.* s.l. : Proceedings of the Brazilian Symposium on GeoInformatics, 2013. σσ. 85-96.

27. **H.J. Haverkort, Toma, L. Zhuang, Yi.** *Computing visibility on terrains in external memory*. s.l. : Journal of Experimental Algorithmics 13, 2008.

28. **William Gropp, Wing Luska, Nathan Doss, Anthony Skjellum.** A high-performance, portable implementation of the MPI message passing interface standard. *Parallel Computing.* 1996, σσ. 789-828.

29. **Philippe Flajolet, Andrew Odlyzko.** *Exploring binary trees and other simple trees.* s.l. : Proceedings of the 21st Annual Symposium on Foundations of Computer Science, 1980. σσ. 207–216.

30. **Vimal P. Parmar, Kumbharana CK.** *Comparing Linear Search and Binary Search Algorithms to Search an Element from a Linear List Implemented through Static Array, Dynamic Array and Linked List.* s.l. : International Journal of Computer Applications 121, 2015. σσ. 13-17. Volume 121 – No.3.

31. **Κ. Νικολακόπουλος, Κ. Κατσάνου, Ν. Λαμπράκης,.** Ψηφιακά μοντέλα αναγλύφου. *Υδρολογία με χρήση γεωγραφικών συστημάτων πληροφοριών και δεδομένων τηλεπισκόπησης,2015.* s.l. : [ηλεκτρ. βιβλ.] Αθήνα:Σύνδεσμος Ελληνικών Ακαδημαϊκών Βιβλιοθηκών, σσ. 147-148.

32. **Emil Johansson, Jacob Lunberg.** *Distributed Viewshed Analysis An Evaluation of Distribution Frameworks for Geospatial Information Systems.* Gothenburg, Sweden : s.n., 2016.

33. **James A. Stewart.** Fast Horizon Computation at All Points of a Terrain With Visibility and Shading Applications. *IEEE Transactions on Visualization and Computer Graphics.* 1998, σσ. 82 - 93.

34. **Peter Fisher.** First Experiments in Viewshed Uncertainty:. s.l. : Photogrammetric Engineering and Remote Sensing 57, 1992.

35. **Friedman, Milton.** *The Interpolation of Time Series by Related Series.* s.l. : Journal of the American Statistical Association,Vol. 57, No. 300, 1962. σσ. 729-757 .

36. **Young Hoo, Kima Sanjay Ranab, Steve Wisea.** *Exploring multiple viewshed analysis using terrain features and optimisation techniques.* s.l. : Computers & Geosciences 30, 2004. σσ. 1019-1032.

37. **Rossum, G. van.** *Python tutorial.* s.l. : Technical Report CS-R9526,Centrum voor Wiskunde en Informatica (CWI), 1995.

38. **Travis Oliphant.** *Python for Scientific Computing.* s.l. : Computing in Science and Engineering 9, 2007. σσ. 10-20.

39. **Rigby, Alan S.** *Statistical methods in epidemiology. v. Towardsan understanding of the kappa co effcient.* s.l. : disability and rehabilitation, 2000. σσ. 339-344.

40. **Heather A.Sandera, Steven M.Manson.** *Heights and locations of artificial structures in viewshed calculation: How close is close enough?* s.l. : Landscape and Urban Planning,Volume 82, Issue 4, 2007. σσ. 257-270.

41. **Mathieu Fourment, Michael R Gillings.** *A comparison of common programming languages used in bioinformatics.* s.l. : BMC Bioinformatics volume 9, 2008.

42. **Mark T Holder, Jeet Sukumaran.** *DendroPy: a Python library for phylogenetic computing.* s.l. : Bioinformatics 26, 2010. σσ. 69-71.

43. **Branko Kaucic, Borut Zalik.** *Comparison of Viewshed Algorithms on Regular Spaced Points.* s.l. : Proceedings of the 18th Spring Conference on Computer Graphics, 2002. σσ. 177–183.