ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

Μεταπτυχιακή Εργασία

The Collision-less Conjecture for the Go-To-The-Gabriel-Center (GTGC) algorithm

Λάμπρος Δημητρίου

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

13 Ιουνίου 2024

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

The Collision-less Conjecture for the Go-To-The-Gabriel-Center (GTGC) algorithm

Λάμπρος Δημητρίου

Επιβλέπων: Καθηγητής Μαυρονικόλας Μάριος

Εξεταστική Επιτροπή: Καθηγητής Μαυρονικόλας Μάριος
Επίκουρος Καθηγητής Πιερής Αντρέας
Καθηγήτρια Άννα Φιλίππου

Η Μεταπτυχιακή Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του μεταπτυχιακού Επιστήμη της Πληροφορικής του Τμήματος Πληροφορικής

13 Ιουνίου 2024

ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα διπλωματική εργασία είναι το αποτέλεσμα μιας σειράς αλληλεπιδράσεων με διαφορετικούς ανθρώπους, καθένας από τους οποίους έπαιξε σημαντικό ρόλο στην εξέλιξή της. Σε αυτό το σημείο λοιπόν νιώθω την ανάγκη να εκφράσω τις ειλικρινείς και θερμές μου ευχαριστίες σε όσους συνέβαλαν στην ολοκλήρωση αυτής της προσπάθειας.

Καταρχάς, θα ήθελα να εκφράσω τις ειλικρινείς μου ευχαριστίες στον επιβλέποντα καθηγητή μου, τον καθηγητή Μάριο Μαυρονικόλα, ο οποίος μου έδωσε την ευκαιρία και μου εμπιστεύτηκε την παρούσα διπλωματική εργασία. Ο Καθ. Μάριος Μαυρονικόλας με βοήθησε και με ενθάρρυνε σε όλα τα στάδια αυτής της εργασίας με συνεχή υποστήριξη και καθοδήγηση.

Τέλος, το μεγαλύτερο ευχαριστώ το οφείλω στην οικογένειά μου, τη φίλη μου και τα αγαπημένα μου πρόσωπα, που δέχτηκαν όλες τις επιλογές μου και με στήριξαν σε όλα τα χρόνια σπουδών μου και όσο έγραφα αυτή τη διατριβή. Με κράτησαν να συνεχίσω και αυτή η δουλειά δεν θα ήταν δυνατή χωρίς τη συμβολή τους.

Περίληψη

Στην παρούσα διπλωματική εργασία, μελετήθηκαν και παρουσιάστηκαν διάφοροι αλγόριθμοι συγκέντρωσης (gathering algorithms) που υπάρχουν στη βιβλιογραφία. Οι αλγόριθμοι συγκέντρωσης έχουν σαν στόχο τη συλλογή ενός αυθαίρετου n αριθμού από ρομπότ που υπάρχουν στο ευκλείδειο επίπεδο σε ένα μη προκαθορισμένο σημείο.

Συγκεκριμένα, δόθηκε περισσότερη έμφαση σε αλγορίθμους συνεχούς χρόνου που ακολουθούν το μοντέλο Look - Compute - Move. Δηλαδή, τον εντοπισμό των γειτόνων τους από το κοντινό τους περιβάλλον, τον υπολογισμό του σημείου προορισμού τους και τέλος, την κίνηση τους προς το σημείο προορισμού τους.

Από αυτούς τους αλγόριθμους, ξεχώρισε ο Go-To-The-Gabriel-Center (GTGC) αλγόριθμος, ο οποίος αποδείχθηκε ότι είναι collision-less στη μια διάσταση του ευκλείδειου επιπέδου και εικάζεται ότι είναι collision-less και στις δύο διαστάσεις. Κάτι το οποίο δεν αποδείχθηκε, παρά μόνο πάρθηκε το συμπέρασμα βάση προσομοιώσεων.

Σε αυτή την διπλωματική εργασία, ο κυριότερος στόχος ήταν να γίνει μια προσπάθεια να αποδειχθεί αυτή η εικασία ή να καταρριφθεί. Να αποδειχθεί δηλαδή αν ο αλγόριθμος Go-To-The-Gabriel-Center (GTGC) είναι collision-less ή όχι.

Αρχικά, αναλύθηκαν όλες οι πιθανές καταστάσεις που μπορεί να βρεθεί ένα ρομπότ και έγιναν κάποιες προσομοιώσεις βάση αυτών των καταστάσεων. Χρησιμοποιώντας τις προσομοιώσεις, τον τρόπο εκτέλεσης του αλγορίθμου και θεωρήματα από τη βιβλιογραφία, έγινε μια προσπάθεια να ληφθούν κάποια συμπεράσματα για το κατά πόσο μπορούν να υπάρξουν συγκρούσεις και υπό ποιες προϋποθέσεις.

Λαμβάνοντας υπόψη τα συμπεράσματα και τις προσομοιώσεις, προσπάθησε να αποδειχθεί η εικασία με μαθηματικό τρόπο. Αφού δημιουργήθηκε ένα σύστημα ανισοτήτων με εξισώσεις που πηγάζουν από τα συμπεράσματα, ελέγχθηκε αν υπάρχουν λύσεις στο σύστημα

ανισοτήτων, και αν υπάρχουν, τι συμπεράσματα μπορούν να ληφθούν.

Όπως παρουσιάζεται και στη συνέχεια, ο συγκεκριμένος αλγόριθμος αποδείχθηκε ότι μπορεί να θεωρηθεί collision-less εκτός από το σενάριο όπου μπορεί να βρεθεί σε μια συγκεκριμένη κατάσταση (cross-shaped graph). Λόγω του ότι η πιθανότητα να βρεθεί σε μια τέτοια κατάσταση (cross-shaped graph) τείνει στο 0, τότε λαμβάνεται σαν γενικό συμπέρασμα ότι ο αλγόριθμος Go-To-The-Gabriel-Center (GTGC) είναι collision-less (Lebesgue measure).

ABSTRACT

In this thesis, various gathering algorithms that exist in the literature were studied and presented. Gathering algorithms aim to gather an arbitrary number of n robots that exist in the Euclidean plane at a non predefined point.

In particular, more emphasis was placed on continuous-time algorithms that follow the Look - Compute - Move model. That is, the detection of their neighbors from their immediate environment, the calculation of their target point and finally, their movement towards their target point.

Among these algorithms, the Go-To-The-Gabriel-Center (GTGC) algorithm stood out, which was shown to be collision-less in one dimension of the Euclidean plane and conjectured to be collision-less in both dimensions. Something that was not proven, but was only concluded based on simulations.

In this thesis, the main aim was to make an attempt to prove this conjecture or to demolish it. That is, to prove whether the Go-To-The-Gabriel-Center (GTGC) algorithm is collision-less or not.

First, all the possible situations that a robot can find itself in, were analyzed and some simulations were made based on these situations. Using the simulations, how the algorithm is executed, and theorems from the literature, an attempt was made to draw some conclusions about whether collisions can occur and under what conditions.

Taking into account the conclusions and simulations, an attempt was made to prove the conjecture in a mathematical way. After creating a system of inequalities with equations derived from the conclusions, it is checked whether there are solutions to the system of inequalities, and if there are, what conclusions can be drawn.

As shown below, this particular algorithm is shown to be collision-less except for the scenario where it can be in a specific state (cross-shaped graph). Since the probability of being in such a state (cross-shaped graph) tends to 0, then it is taken as a general conclusion that the algorithm Go-To-The-Gabriel-Center (GTGC) is collision-less (Lebesgue measure).

Περιεχόμενα

Κατάλογος Σχημάτων

Κεφάλαιο 1

Εισαγωγή

1.1 Θεωρητικό Υπόβαθρο

Στον τομέα των παράλληλων, κατανεμημένων συστημάτων (Distributed Computing), εδώ και δεκαετίες, γίνεται μεγάλη έρευνα στον εντοπισμό αλγορίθμων οι οποίοι στοχεύουν στη συγκέντρωση των ρομπότ στο ευκλείδειο επίπεδο.

Πιο συγκεκριμένα, δεδομένης μιας ομάδας n αυτόνομων, αδιάστατων, ντετερμινιστικών και ανώνυμων ρομπότ, με περιορισμένο εύρος θέασης, ένας αλγόριθμος συλλογής (gathering algorithm) είναι ένας αλγόριθμος που συγκεντρώνει όλα τα n ρομπότ σε πεπερασμένο χρόνο, σε ένα συγκεκριμένο, μη προκαθορισμένο σημείο στο ευκλείδειο επίπεδο. Τέτοιοι αλγόριθμοι, πρέπει να είναι όσο το δυνατόν με καλύτερη χρονική πολυπλοκότητα (Time Complexity), όπως επίσης, και Collision-less, δηλαδή, κατά τη διάρκεια της εκτέλεσης τους να μην προκαλούνται συγκρούσεις μεταξύ των ρομπότ. Οι συγκεκριμένοι αλγόριθμοι έχουν αναπτυχθεί σε δύο χρονικά μοντέλα: διακριτό και συνεχές χρόνο. [1] [2]

Στους αλγόριθμους συγκέντρωσης που μελετήθηκαν, όλα τα ρομπότ λειτουργούν χωρίς κεντρικό έλεγχο (αυτόνομα), δεν έχουν επικοινωνία μεταξύ τους, δεν έχουν μοναδικά αναγνωριστικά (ανώνυμα) και δεν έχουν μακροπρόθεσμη μνήμη (δεν αποθηκεύουν πληροφορίες τοπικά) ή πλήρεις πληροφορίες για τις θέσεις όλων των άλλων ρομπότ. Επίσης, τα ρομπότ έχουν περιορισμένο εύρος θέασης το οποίο συνήθως δεν ξεπερνά το 1 (viewing range = 1). Η κίνηση των ρομπότ εξαρτάται αποκλειστικά από τις σχετικές θέσεις άλλων ρομπότ κοντά τους. [1] [2]

Όλα τα ρομπότ στο ευκλείδειο επίπεδο, εκτελούν ακριβώς τον ίδιο αλγόριθμο με τα ίδια βήματα. Συνήθως, ο αλγόριθμος υποδιαιρείται σε τρία βήματα: Look, Compute και Move. Αυτά τα τρία βήματα είναι επίσης γνωστά ως κύκλος LCM. Στο πρώτο βήμα, το ρομπότ συλλέγει τις πληροφορίες για το περιβάλλον, στη συνέχεια, χρησιμοποιώντας τις πληροφορίες από το προηγούμενο βήμα, το ρομπότ υπολογίζει το σημείο προορισμού του (target point) και τελικά κινείται προς αυτό το σημείο. [1] [2]

Ανάλογα με τον τρόπο ενεργοποίησης των ρομπότ, διακρίνουμε δύο χρονικά μοντέλα: διακριτό και συνεχές. Στο διακριτό μοντέλο η ενεργοποίηση του ρομπότ χωρίζεται σε γύρους. Κατά τη διάρκεια κάθε γύρου, το ρομπότ εκτελεί έναν κύκλο LCM. Τα μοντέλα διακριτού χρόνου χωρίζονται σε ασύγχρονα και ημισύγχρονα ανάλογα με το αν η ενεργοποίηση του ενός ρομπότ εξαρτάται από την ενεργοποίηση του άλλου. [1] [2]

Σε αντίθεση με τα συμβατικά μοντέλα διακριτού χρόνου, στα μοντέλα συνεχούς χρόνου, τα ρομπότ δεν ενεργούν σε κύκλους, αλλά προσαρμόζουν συνεχώς τις κατευθύνσεις κίνησης προς τα υπολογισμένα σημεία στόχου (target points), ενώ κινούνται με σταθερή ταχύτητα και σε ευθεία γραμμή. Γίνεται η υπόθεση ότι τα ρομπότ εκτελούν πλήρη κύκλο LCM σε κάθε χρονική στιγμή. Ως αποτέλεσμα, ο χρόνος εκτέλεσης δεν μπορούσε να οριστεί ως ο αριθμός των γύρων σε διακριτά μοντέλα. Αντίθετα, ορίζεται ως ο χρόνος που χρειάζεται να συγκεντρωθούν τα ρομπότ σε ένα σημείο. [1] [2] [3]

Παρόλο που έχουν αναπτυχθεί αρκετοί αλγόριθμοι συγκέντρωσης τόσο στο διακριτό όσο και στο συνεχές χρόνο, ακόμη δεν έχουν ανακαλυφθεί αλγόριθμοι οι οποίοι να έχουν αποδειχθεί ότι είναι Collision-less κυρίως στις δύο διαστάσεις που να μην χρησιμοποιούν πρόσθετες δυνατότητες.

Σε αυτή την διπλωματική εργασία, μελετήθηκαν κυρίως αλγόριθμοι συνεχούς χρόνου στις δύο διαστάσεις και διερευνήθηκε κατά πόσο υπάρχουν στην βιβλιογραφία αλγόριθμοι που ικανοποιούν αυτές τις συνθήκες αλλά παράλληλα **να έχουν αποδειχθεί ότι είναι Collision-less** χωρίς να χρησιμοποιούνται πρόσθετες δυνατότητες.

Ένας τέτοιος αλγόριθμος, είναι ο Go-To-The-Gabriel-Center algorithm (GTRC) [1]. Ο συγκεκριμένος αλγόριθμος συγκέντρωσης, παρουσιάστηκε με το μοντέλο συνεχούς χρόνου. Έχει αποδειχθεί ότι είναι Collision-less αλλά μόνο για τη μια διάσταση. Ο αρθρογράφος, ισχυρίζεται ότι αυτός ο αλγόριθμος είναι και στις δύο διαστάσεις Collision-less και εμβαθύνει τον ισχυρισμό του παρουσιάζοντας κάποιες προσομοιώσεις. Παρόλα αυτά, δεν υπήρξε κάποια

μαθηματική απόδειξη που επαληθεύει αυτό τον ισχυρισμό.

1.2 Στόχος Μεταπτυχιακής Εργασίας

Σε αυτή την διπλωματική εργασία, μελετήθηκαν αλγόριθμοι συγκέντρωσης που υπάρχουν στην βιβλιογραφία, οι οποίοι αφορούν κυρίως μοντέλα συνεχούς χρόνου και αναφέρονται στις δύο διαστάσεις στο ευκλείδειο επίπεδο.

Το κριτήριο που τέθηκε στη μελέτη των αλγορίθμων, είναι ότι αυτοί οι αλγόριθμοι πρέπει να είναι Collision-less και επιπλέον, να μην χρησιμοποιούν επιπλέον δυνατότητες (additional capabilities) για να θεωρούνται Collision-less.

Ο κυριότερος στόχος της διπλωματικής εργασίας, πέραν από την κατανόηση και μελέτη των αλγόριθμων συγκέντρωσης, ήταν να διερευνηθεί και να αποδειχθεί με τη χρήση μαθηματικών θεωρημάτων που υπάρχουν στην βιβλιογραφία, κατά πόσο ο αλγόριθμος συγκέντρωσης Go-To-The-Gabriel-Center algorithm (GTGC) είναι Collision-less ή όχι. Με λίγα λόγια, έγινε προσπάθεια απόδειξης ή κατάρριψης του ισχυρισμού του αρθρογράφου που παρουσίασε αυτόν τον αλγόριθμο ότι ο συγκεκριμένος αλγόριθμος ίσως να είναι Collision-less σε μοντέλα συνεχούς χρόνου στις δύο διαστάσεις στο ευκλείδειο επίπεδο, που περιέχει ένα αυθαίρετο αριθμό n από ρομπότ με $n \geq 4$.

Κεφάλαιο 2

Ανασκόπηση της Βιβλιογραφίας

2.1 Εισαγωγή

Σε αυτό το κεφάλαιο, θα παρουσιαστούν μερικοί από τους αλγόριθμους συγκέντρωσης (Gathering algorithms) που υπάρχουν στην βιβλιογραφία όπως είναι ο Go-To-The-Center algorithm (GTC) [4], ο Go-To-The-Relative-Center algorithm (GTRC) [2] και ο Go-To-The-Gabriel-Center algorithm (GTGC) [1].

Επίσης, θα επεξηγηθούν και αναλυθούν οντότητες όπως το σημείο προορισμού (target point), ο Minimal Enclosing Circle (MEC) και Gabriel graph (GG) οι οποίες χρησιμοποιούνται στον αλγόριθμο που μελετήθηκε ο οποίος είναι ο Go-To-The-Gabriel-Center algorithm.

Επιπλέον, θα γίνει αναφορά στο Planar graph και Delaunay triangulation και θα παρουσιαστούν μαθηματικά θεωρήματα που πηγάζουν μέσω αυτών και τα οποία χρησιμοποιήθηκαν για την απόδειξη που ακολουθεί στα επόμενα κεφάλαια.

2.2 Αλγόριθμοι Συγκέντρωσης

2.2.1 Go-To-The-Center algorithm (GTC)

Algorithm: Go-To-The-Center (GTC).

Require: Initial configuration

1: Robot r observes the positions of its neighbours in its UDG(r).

2: Robot r computes the minimum circle C(r) enclosing UDG(r). The centre T(r) of C(r) is the target point of r.

3: Robot r moves:

4: **if** r is already at T(r) **then**

5: Robot r remains at T(r) and moves in the same way as the target point does.

6: **else**

7: Robot r moves with maximum speed 1 towards T(r).

• **Επεξήγηση:**

Η επεξήγηση θα γίνει για το ένα ρομπότ (ρομπότ r) αλλά το ίδιο ισχύει για κάθε ρομπότ του ευκλείδειου επίπεδου που εκτελεί τον αλγόριθμο. Ο ίδιος αλγόριθμος επαναλαμβάνεται σε κάθε επανάληψη (κάθε κύκλο εκτέλεσης) από το σημείο 1. Αρχικά καθορίζονται τα initial configurations (αρχικές θέσεις των ρομπότ στο ευκλείδειο επίπεδο).

1. Στο πρώτο βήμα, το ρομπότ r εντοπίζει όλα τα ρομπότ που βρίσκονται στο ευκλείδειο επίπεδο τα οποία έχουν μέγιστη απόσταση 1 μεταξύ του (το viewing range καθορίστηκε στο 1), δηλαδή, τους γείτονες του στο UDG(r). Καθορίζοντας ως κάποιο ρομπότ έστω r2 ως γείτονα του r, τότε, τα 2 ρομπότ είναι connected στον UDG(r) (ομοίως και του UDG(r2)). Με το τέλος του πρώτου βήματος, θα υπάρχει το UDG(r) όπου θα υποδεικνύονται οι γείτονες του καθώς και οι θέσεις του.

2. Στο δεύτερο βήμα, ο σκοπός είναι να εντοπιστεί το target point του r (T(r)). Δηλαδή, σε πιο σημείο θα πρέπει να κινηθεί το ρομπότ r σε αυτή την επανάληψη. Για να εντοπιστεί το target point του r, υπολογίζεται ο ελάχιστος κύκλος εγκλεισμού του (MEC(r) -

Minimal Enclosing Circle), δηλαδή, ο ελάχιστος κύκλος που περικλείει το UDG(r). Το target point του r θα είναι το κέντρο του MEC(r).

3. Σε αυτό το σημείο, το ρομπότ r είναι έτοιμο να κινηθεί προς το target point το οποίο εντοπίστηκε στο προηγούμενο βήμα (σημείο 2):

   (α΄) **Αν** το ρομπότ r βρίσκεται ήδη στο target point T(r) το οποίο υπολογίστηκε από το βήμα 2, τότε το ρομπότ r θα παραμείνει σε αυτό το σημείο και θα κινείται με τον ίδιο τρόπο που κινείται και το T(r) (προϋποθέτει ότι θα αλλάξει το MEC(r)).

   (β΄) **Αλλιώς,** το ρομπότ r θα κινηθεί με μέγιστη ταχύτητα 1 προς το T(r) που υπολογίστηκε στο βήμα 2.

Στην επόμενη επανάληψη (στον επόμενο κύκλο) του αλγορίθμου για το ρομπότ r, ο αλγόριθμος θα ακολουθήσει τα ίδια βήματα ξεκινώντας από το σημείο 1.

2.2.2 Go-To-The-Relative-Center algorithm (GTRC)

Algorithm: Go-To-The-Relative-Center algorithm (GTRC).

Require: Initial configuration
1: Robot r observes the positions of its unit disk graph neighbors and calculates RNG(r).
2: Robot r computes the minimum circle C(r) enclosing RNG(r). The center T(r) of C(r) is the target point of r.
3: Robot r moves:
4: **if** r is already at T(r) **then**
5: Robot r remains at T(r) and moves in the same way as the target point does.
6: **else**
7: Robot r moves with maximum speed 1 towards T(r).

• **Επεξήγηση:**

Η επεξήγηση θα γίνει για το ένα ρομπότ (ρομπότ r) αλλά το ίδιο ισχύει για κάθε ρομπότ του ευκλείδειου επίπεδου που εκτελεί τον αλγόριθμο. Ο ίδιος αλγόριθμος επαναλαμβάνεται σε

κάθε επανάληψη (κάθε κύκλο εκτέλεσης) από το σημείο 1. Αρχικά καθορίζονται τα initial configurations (αρχικές θέσεις των ρομπότ στο ευκλείδειο επίπεδο).

1. Στο πρώτο βήμα, το ρομπότ r εντοπίζει όλα τα ρομπότ που βρίσκονται στο ευκλείδειο επίπεδο τα οποία έχουν μέγιστη απόσταση 1 μεταξύ του (το viewing range καθορίστηκε στο 1), δηλαδή, τους γείτονες του στο UDG(r). Για τον εντοπισμό και καθορισμό των γειτόνων, χρησιμοποιείται η μέθοδος RNG lenses η οποία θα επεξηγηθεί στη συνέχεια. Καθορίζοντας ως κάποιο ρομπότ έστω r2 ως γείτονα του r, τότε τα 2 ρομπότ είναι connected στον RNG του r (ομοίως και του r2). Με το τέλος του πρώτου βήματος, θα υπάρχει το Relative Neighborhood graph του r όπου θα υποδεικνύονται οι γείτονες του καθώς και οι θέσεις τους.

2. Στο δεύτερο βήμα, ο σκοπός είναι να εντοπιστεί το target point του r (T(r)). Δηλαδή, σε πιο σημείο θα πρέπει να κινηθεί το ρομπότ r σε αυτή την επανάληψη. Για να εντοπιστεί το target point του r, υπολογίζεται ο ελάχιστος κύκλος εγκλεισμού του (MEC(r) - Minimal Enclosing Circle), δηλαδή, ο ελάχιστος κύκλος που περικλείει το RNG(r). Το target point του r θα είναι το κέντρο του MEC(r).

3. Σε αυτό το σημείο, το ρομπότ r είναι έτοιμο να κινηθεί προς το target point το οποίο εντοπίστηκε στο προηγούμενο βήμα (σημείο 2):

   (α΄) **Αν** το ρομπότ r βρίσκεται ήδη στο target point T(r) το οποίο υπολογίστηκε από το βήμα 2, τότε το ρομπότ r θα παραμείνει σε αυτό το σημείο και θα κινείται με τον ίδιο τρόπο που κινείται και το T(r) (προϋποθέτει ότι θα αλλάξει το MEC(r)).

   (β΄) **Αλλιώς,** το ρομπότ r θα κινηθεί με μέγιστη ταχύτητα 1 προς το T(r) που υπολογίστηκε στο βήμα 2.

Στην επόμενη επανάληψη (στον επόμενο κύκλο) του αλγορίθμου για το ρομπότ r, ο αλγόριθμος θα ακολουθήσει τα ίδια βήματα ξεκινώντας από το σημείο 1.

2.2.3 Go-To-The-Gabriel-Center algorithm (GTGC)

Algorithm: Go-To-The-Gabriel-Center algorithm (GTGC).

Require: Initial configuration

1: Robot r observes the positions of its neighbours in its GG(r).

2: Robot r computes the minimum circle C(r) enclosing GG(r). The center T(r) of C(r) is the target point of r.

3: Robot r moves:

4: **if** r is already at T(r) **then**

5: Robot r remains at T(r) and moves in the same way as the target point does.

6: **else**

7: Robot r moves with maximum speed 1 towards T(r).

• **Επεξήγηση:**

Η επεξήγηση θα γίνει για το ένα ρομπότ (ρομπότ r) αλλά το ίδιο ισχύει για κάθε ρομπότ του ευκλείδειου επίπεδου που εκτελεί τον αλγόριθμο. Ο ίδιος αλγόριθμος επαναλαμβάνεται σε κάθε επανάληψη (κάθε κύκλο εκτέλεσης) από το σημείο 1. Αρχικά καθορίζονται τα initial configurations (αρχικές θέσεις των ρομπότ στο ευκλείδειο επίπεδο).

1. Στο πρώτο βήμα, το ρομπότ r εντοπίζει όλα τα ρομπότ που βρίσκονται στο ευκλείδειο επίπεδο τα οποία έχουν μέγιστη απόσταση 1 μεταξύ του (το viewing range καθορίστηκε στο 1), δηλαδή, τους γείτονες του στο GG(r). Για τον εντοπισμό και καθορισμό των γειτόνων χρησιμοποιείται το κριτήριο του Gabriel graph το οποίο θα επεξηγηθεί στη συνέχεια. Καθορίζοντας ως κάποιο ρομπότ έστω r2 ως γείτονα του r, τότε τα 2 ρομπότ είναι connected στον Gabriel graph του r (ομοίως και του r2). Με το τέλος του πρώτου βήματος, θα υπάρχει το Gabriel graph του r όπου θα υποδεικνύονται οι γείτονες του καθώς και οι θέσεις τους.

2. Στο δεύτερο βήμα, ο σκοπός είναι να εντοπιστεί το target point του r (T(r)). Δηλαδή, σε πιο σημείο θα πρέπει να κινηθεί το ρομπότ r σε αυτή την επανάληψη. Για να εντοπιστεί το target point του r, υπολογίζεται ο ελάχιστος κύκλος εγκλεισμού του (MEC(r) -

Minimal Enclosing Circle), δηλαδή, ο ελάχιστος κύκλος που περικλείει το GG(r). Το target point του r θα είναι το κέντρο του MEC(r). Το MEC(r) θα επεξηγηθεί καλύτερα στη συνέχεια, αλλά με λίγα λόγια υπολογίζεται ο μικρότερος κύκλος που μπορεί να υπάρξει ο οποίος θα περιέχει μέσα ή πάνω σε αυτόν όλους τους γείτονες του ρομπότ r (εννοείται και το ρομπότ r).

3. Σε αυτό το σημείο, το ρομπότ r είναι έτοιμο να κινηθεί προς το target point το οποίο εντοπίστηκε στο προηγούμενο βήμα (σημείο 2):

   (α΄) **Αν** το ρομπότ r βρίσκεται ήδη στο target point T(r) το οποίο υπολογίστηκε από το βήμα 2, τότε το ρομπότ r θα παραμείνει σε αυτό το σημείο και θα κινείται με τον ίδιο τρόπο που κινείται και το T(r) (προϋποθέτει ότι θα αλλάξει το MEC(r)).

   (β΄) **Αλλιώς,** το ρομπότ r θα κινηθεί με μέγιστη ταχύτητα 1 προς το T(r) που υπολογίστηκε στο βήμα 2.

Στην επόμενη επανάληψη (στον επόμενο κύκλο) του αλγορίθμου για το ρομπότ r, ο αλγόριθμος θα ακολουθήσει τα ίδια βήματα ξεκινώντας από το σημείο 1.

2.2.3.1 Go-To-The-Center algorithm (GTC) σαν αναφορά

Οι δύο πιο πάνω αλγόριθμοι (Go-To-The-Relative-Center algorithm (GTRC) και Go-To-The-Gabriel-Center algorithm (GTGC)), χρησιμοποιούν σαν αναφορά τον αλγόριθμο Go-To-The-Center (GTC). Στον Go-To-The-Center (GTC) αλγόριθμο, αφού υπολογιστούν όλα τα γειτονικά ρομπότ του ρομπότ r στο UDG(r), τότε χρησιμοποιείται η μέθοδος του ελάχιστου κύκλου εγκλεισμού (MEC) πάνω στο UDG(r) για να υπολογιστεί το σημείο προορισμού (target point) του r. Στους άλλους δύο αλγόριθμους, εκτελούνται ακριβώς τα ίδια βήματα εκτός από το σε πιο γράφο εκτελείται η μέθοδος του ελάχιστου κύκλου εγκλεισμού που υπολογίζει το σημείο προορισμού.

- Στον αλγόριθμο Go-To-The-Relative-Center (GTRC) χρησιμοποιείται η μέθοδος του ελάχιστου κύκλου εγκλεισμού (MEC) πάνω στο RNG(r) για να υπολογιστεί το σημείο προορισμού (προϋποθέτει ότι στο σημείο 1 του αλγορίθμου υπολογίστηκε ο RNG(r)).

- Στον αλγόριθμο Go-To-The-Gabriel-Center (GTGC) χρησιμοποιείται η μέθοδος του ελάχιστου κύκλου εγκλεισμού (MEC) πάνω στο GG(r) για να υπολογιστεί το σημείο προορισμού (προϋποθέτει ότι στο σημείο 1 του αλγορίθμου υπολογίστηκε ο GG(r)).

Με αυτό τον τρόπο, πετυχαίνεται διαφορετική ομαδοποίηση με πιο μικρό αριθμό από ρομπότ που γειτνιάζουν. Αυτό έχει σαν αποτέλεσμα την καλύτερη απόδοση των δύο τελευταίων αλγορίθμων, όπως επίσης, έχουν και πολύ μικρότερη πιθανότητα συγκρούσεων.

2.3 Σημείο Προορισμού

Σε κάθε εκτέλεση του αλγορίθμου, το σημαντικότερο κομμάτι του είναι ο υπολογισμός του σημείου προορισμού target point που θα πρέπει να κινηθεί το ρομπότ που εκτελεί τον αλγόριθμο μετά την εκτέλεση του. Στη βιβλιογραφία, παρουσιάστηκαν αρκετοί τρόποι με τους οποίους μπορεί να υπολογιστεί το σημείο προορισμού σε ένα αλγόριθμο. Πιο κάτω, παρουσιάζονται κάποιοι από αυτούς.

1. Ένας τρόπος είναι αυτός που χρησιμοποιείται στον αλγόριθμο Go-On-Bisector (GOB) που προτείνεται από τους Gordon et al. [5] και αργότερα μελετήθηκε από τους Kempkes et al. [3]. Στον αλγόριθμο Go-On-Bisector, τα ρομπότ που βρίσκονται στο όριο του τοπικού κυρτού κύτους (local convex hull), που ορίζεται γύρω από τη γειτονιά του UDG (Unit Disc Graph), κινούνται μέσα στο τοπικό κυρτό κύτος κατά μήκος της διχοτόμου της εσωτερικής γωνίας. Εάν τα ρομπότ δεν έχουν ακόμη συγκεντρωθεί σε ένα σημείο, τότε κάθε ρομπότ $r \in CH(R)$ κινείται μέσα στο κυρτό κύτος $CH(R)$ με ταχύτητα 1. Με άλλα λόγια, ο GOB είναι ένας αλγόριθμος συστολής (contracting algorithm).

2. Ένας άλλος τρόπος είναι μέσω του μέσου όρου των θέσεων όλων των γειτονικών ρομπότ. Αυτός ο αλγόριθμος είναι γνωστός ως αλγόριθμος Go-To-The-Gravity-Center (GTGrC), λόγω των Cohen et al. [6]. Ο GTGrC είναι επίσης ένας αλγόριθμος συστολής (contracting algorithm).

3. Ένας τρίτος τρόπος υπολογισμού του σημείου προορισμού, ο οποίος είναι και αυτός που μελετήθηκε σε αυτή την διπλωματική εργασία, είναι μέσω του υπολογισμού του ελάχιστου κύκλου εγκλεισμού (MEC: Minimal Enclosing Circle) που περικλείει όλα τα γειτονικά ρομπότ του ρομπότ r που εκτελεί τον αλγόριθμο (τη γειτονιά UDG του ρομπότ) και το σημείο προορισμού καθορίζεται ως το κέντρο του ελάχιστου κύκλου εγκλεισμού. Αυτός ο αλγόριθμος είναι γνωστός ως αλγόριθμος Go-To-The-Center (GTC), λόγω των Ando et al. [4].

   Αυτός ο τρόπος υπολογισμού του σημείου προορισμού χρησιμοποιήθηκε επίσης και

στον αλγόριθμο Go-To-The-Gabriel-Center (GTGC) [1] όπου στον συγκεκριμένο αλγόριθμο δεν υπολογίζεται ο ελάχιστος κύκλος εγκλεισμού σε όλα τα ρομπότ του ευκλείδειου επίπεδου που έχουν viewing range 1, αλλά στα γειτονικά ρομπότ (Gabriel neighbors) του ρομπότ r που εκτελεί τον αλγόριθμο με βάση τα κριτήρια του μοναδιαίου γραφήματος Gabriel (Gabriel graph).

Επιπλέον, αυτός ο τρόπος χρησιμοποιήθηκε και στον αλγόριθμο Go-To-The-Relative-Center (GTRC) [2] όπου στον συγκεκριμένο αλγόριθμο υπολογίζεται ο ελάχιστος κύκλος εγκλεισμού στα γειτονικά ρομπότ (relative neighbors) του ρομπότ r που εκτελεί τον αλγόριθμο με βάση τα κριτήρια του μοναδιαίου γραφήματος Relative Neighborhood.

Οι τρεις πιο πάνω αλγόριθμοι (CTC, GTGC, GTRC είναι όλοι αλγόριθμοι συρρίκνωσης (contracting alrorithm).

2.4 Ελάχιστος Κύκλος Εγκλεισμού (MEC)

Ο Ελάχιστος κύκλος Εγκλεισμού (Minimal Enclosing Circle) είναι η διαδικασία όπου γίνεται εύρεση του κύκλου με τη μικρότερη ακτίνα που περιέχει ένα δεδομένο σύνολο σημείων στο εσωτερικό του ή στα όριά του. Αυτός ο μικρότερος κύκλος είναι γνωστός ως ο ελάχιστος περικλειόμενος κύκλος (minimal enclosing circle). Είναι γνωστό και ως minimum covering circle problem, bounding circle problem, least bounding circle problem, smallest enclosing circle problem. Γενικότερα, είναι ένα πρόβλημα υπολογιστικής γεωμετρίας υπολογισμού του μικρότερου κύκλου που περιέχει όλα τα σημεία που βρίσκονται σε ένα σύνολο δεδομένων στο Ευκλείδειο επίπεδο (Euclidean plane).

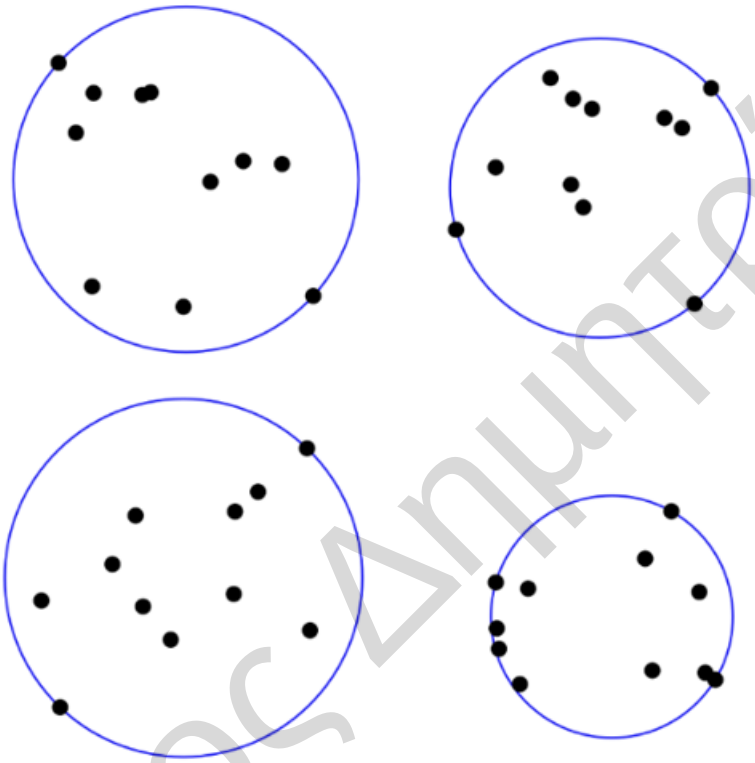
• **Χαρακτηριστικά και ιδιότητες:**

Οι περισσότερες από τις γεωμετρικές προσεγγίσεις για το πρόβλημα, αναζητούν σημεία που βρίσκονται στο όριο του ελάχιστου κύκλου και βασίζονται στα ακόλουθα γεγονότα:

- Ο ελάχιστος κύκλος κάλυψης είναι μοναδικός (minimal enclosing circle).
- Ο ελάχιστος κύκλος κάλυψης ενός συνόλου S μπορεί να προσδιοριστεί το πολύ από τρία σημεία του S που βρίσκονται στο όριο του κύκλου (περίμετρος του κύκλου).
  - Εάν προσδιορίζεται από δύο μόνο σημεία, τότε το ευθύγραμμο τμήμα που ενώνει

αυτά τα δύο σημεία πρέπει να είναι η διάμετρος του ελάχιστου κύκλου.

– Αν προσδιορίζεται από τρία σημεία, τότε το τρίγωνο που αποτελείται από αυτά τα τρία σημεία δεν είναι αμβλύ (obtuse), δηλαδή, δεν έχει γωνία μεγαλύτερη των 90°. Άρα το τρίγωνο είναι οξύ.



Σχήμα 2.1: Με μπλε χρώμα παρατηρούνται οι ελάχιστοι κύκλοι εγκλεισμού των σημείων του συνόλου. Όπως φαίνεται, στην περιφέρεια του κύκλου μπορούν να υπάρξουν το πολύ 3 σημεία του συνόλου. [7]

.

2.5 Γράφος Gabriel (GG)

Στα μαθηματικά και την υπολογιστική γεωμετρία, με τη χρήση του Gabriel graph στα σημεία ενός συνόλου S, δίνεται η δυνατότητα έκφρασης μιας έννοιας εγγύτητας (γειτονεύουν μεταξύ τους) για αυτά τα σημεία. Μέσω του Gabriel graph, μπορεί να καθοριστεί κατά πόσο δύο διακριτά σημεία στον ευκλείδειο χώρο είναι γειτονικά μεταξύ τους. Τυπικά, είναι το γράφημα G με σύνολο κορυφών S στο οποίο οποιαδήποτε σημεία $p \in S$ και $q \in S$ είναι γειτονικά ακριβώς εάν είναι διακριτά, δηλ. $p \neq q$, και ο κλειστός δίσκος του οποίου το ευθύγραμμο τμήμα pq είναι η διάμετρος του, δεν περιέχει στο εσωτερικό του άλλα στοιχεία του S.

•Υπάρχουν δύο κριτήρια για να καθοριστεί αν τα δύο διακριτά σημεία είναι γειτονικά μεταξύ τους (Gabriel neighbours):

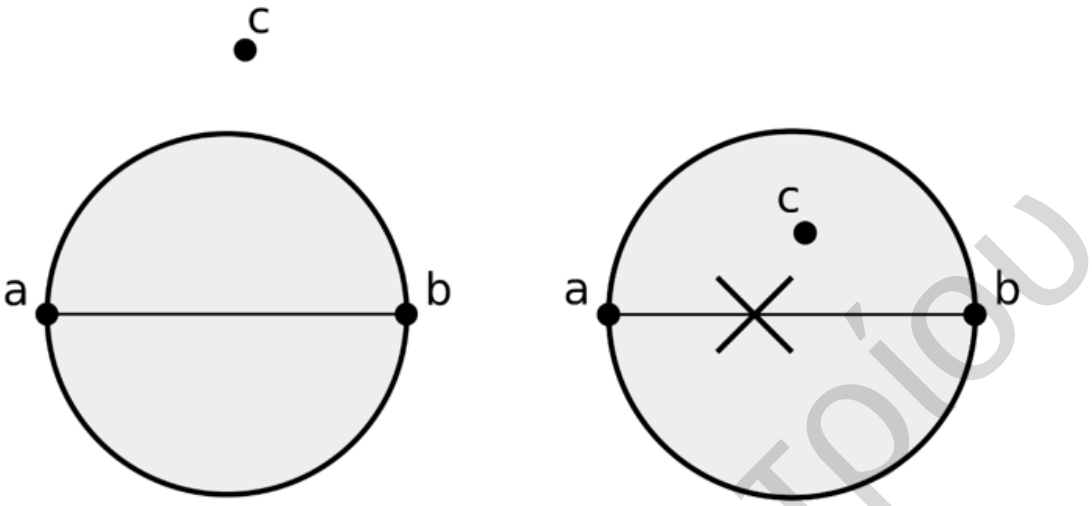
1. Έστω, τα 2 σημεία p και q, αν ο κλειστός δίσκος με διάμετρο pq δεν περιέχει άλλο σημείο στο εσωτερικό του, τότε τα 2 σημεία είναι γειτονικά μεταξύ τους. Με άλλα λόγια, τα σημεία p και q είναι γειτονικά, εκτός αν υπάρχει κάποιο άλλο σημείο z τέτοιο ώστε στο τρίγωνο pqz η γωνία που υποβάλλεται στο z να είναι $\geq 90^\circ$ (δηλ. το τρίγωνο πρέπει να είναι οξύ και όχι αμβλύ).Παράδειγμα αυτού του κριτηρίου απεικονίζεται στην εικόνα 2.2.

2. Ένας άλλος τρόπος για να χαρακτηριστεί το κριτήριο συνδεσιμότητας στον Gabriel graph, είναι μέσω της ευκλείδειας απόστασης. Ονομάζεται το κριτήριο γειτνίασης των ελαχίστων τετραγώνων, όπου το σημείο p είναι γείτονας με το σημείο q αν το τετράγωνο της απόστασης μεταξύ p και q είναι μικρότερο από το άθροισμα των τετραγώνων των αποστάσεων σε οποιοδήποτε άλλο σημείο z του συνόλου S στο οποίο ανήκουν και τα σημεία p, q. Παράδειγμα αυτού του κριτηρίου απεικονίζεται στην εικόνα 2.3.

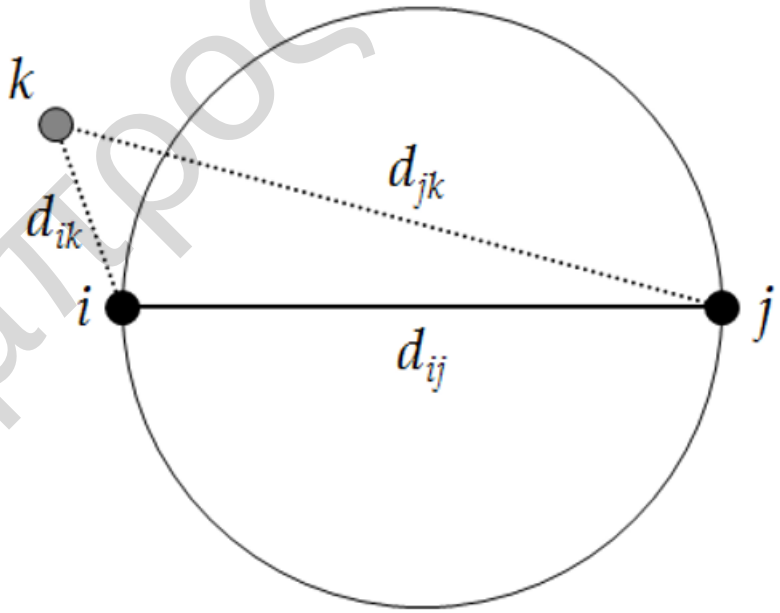
   - Τα σημεία p, q είναι γειτονικά μεταξύ τους αν και μόνο αν:

$$\mathbf{d}_{\mathbf{pq}}^2 < \mathbf{d}_{\mathbf{pz}}^2 + \mathbf{d}_{\mathbf{qz}}^2$$

   για όλα τα z τα οποία ανήκουν στο σύνολο S.



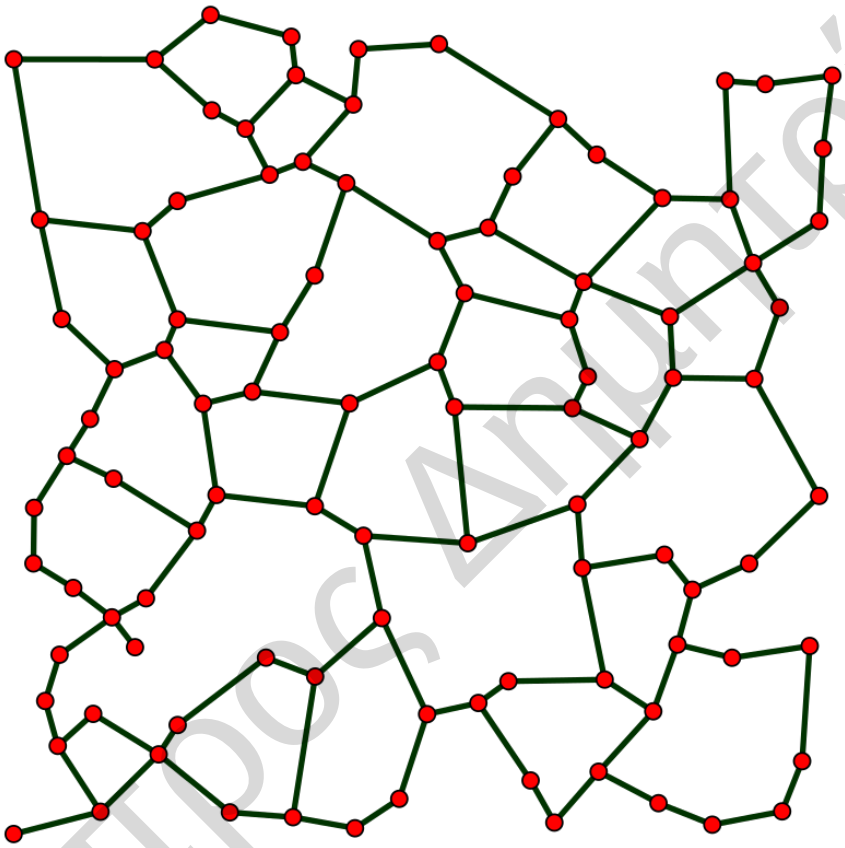
Σχήμα 2.2: Στην αριστερή εικόνα, παρατηρείται ότι δεν υπάρχει άλλο σημείο στο εσωτερικό του κύκλου με διάμετρο ab, έτσι ορίζεται ότι τα σημεία a, b είναι γειτονικά μεταξύ τους (στον Gabriel graph, είναι ενωμένα μεταξύ τους). Στην δεξιά εικόνα παρατηρείται ότι υπάρχει το σημείο c στο εσωτερικό του όμοιου κύκλου, οπόταν τα σημεία a, b δεν είναι γειτονικά μεταξύ τους. [7]



Σχήμα 2.3: Παρατηρείται ότι τα σημεία i, j είναι γειτονικά μεταξύ τους, αν το τετράγωνο της απόστασης μεταξύ τους, d_{ij}^2, είναι μικρότερο από το άθροισμα των τετραγώνων της απόστασης μεταξύ αυτών των σημείων και οποιουδήποτε άλλου σημείου k, $d_{ik}^2 + d_{jk}^2$. Δηλαδή, πρέπει να ισχύει: $d_{ij}^2 < d_{ik}^2 + d_{jk}^2$. [8]

2.6 Γράφος Relative Neighborhood (RNG)

Στην υπολογιστική γεωμετρία, το γράφημα σχετικής γειτονιάς (RNG) είναι ένα μη κατευθυνόμενο γράφημα που ορίζεται σε ένα σύνολο σημείων στο ευκλείδειο επίπεδο συνδέοντας δύο σημεία p και q με μια ακμή όταν δεν υπάρχει ένα άλλο τρίτο σημείο r που είναι πιο κοντά και στα δύο p και q από ό,τι είναι μεταξύ τους. [9] [10] [11]

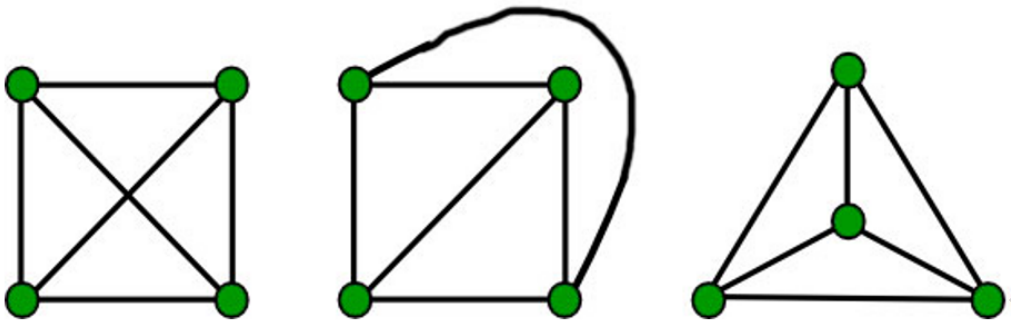


Σχήμα 2.4: Το σχετικό γράφημα γειτονιάς (Relative Neighborhood Graph) 100 τυχαίων σημείων σε μοναδιαίο τετράγωνο. [7]

.

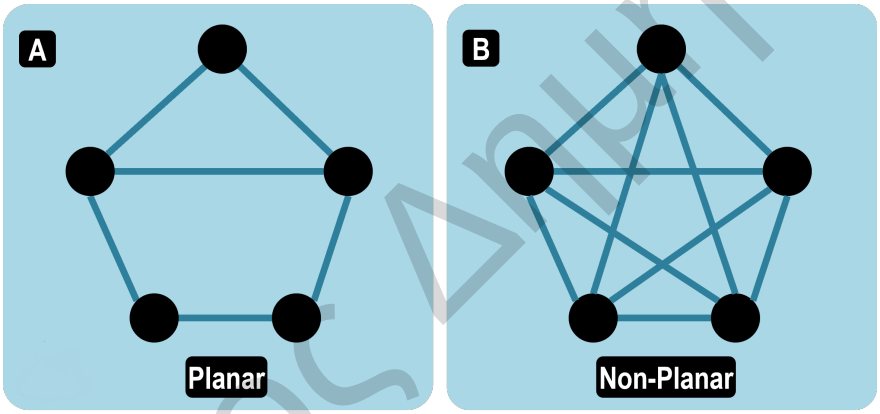
2.7 Επίπεδος γράφος

Στη θεωρία γράφων, ένας επίπεδος γράφος είναι ένας γράφος που μπορεί να ενσωματωθεί στο επίπεδο, δηλ. μπορεί να σχεδιαστεί στο επίπεδο με τέτοιο τρόπο ώστε οι άκρες του να τέμνονται μόνο στα τελικά τους σημεία. Με άλλα λόγια, μπορεί να σχεδιαστεί με τέτοιο τρόπο ώστε να μην διασταυρώνονται άκρες μεταξύ τους.



Σχήμα 2.5: Στην πρώτη εικόνα παρατηρείται ότι υπάρχουν δύο άκρες οι οποίες διασταυρώνονται μεταξύ τους. Άρα, συμπεραίνεται ότι αυτό το γράφημα δεν είναι planar graph. Στις άλλες δύο εικόνες, παρατηρείται ότι δεν υπάρχουν άκρες που να διασταυρώνονται μεταξύ τους. Οπόταν, συμπεραίνεται ότι τα γραφήματα είναι planar graphs. [12]

.



Σχήμα 2.6: Στην εικόνα A, παρουσιάζεται ένας planar graph. Στην εικόνα B, παρουσιάζεται ένας non planar graph. Στην εικόνα A παρατηρείται ότι δεν υπάρχουν άκρες που διασταυρώνονται μεταξύ τους ενώ στην εικόνα B υπάρχουν. [13]

.

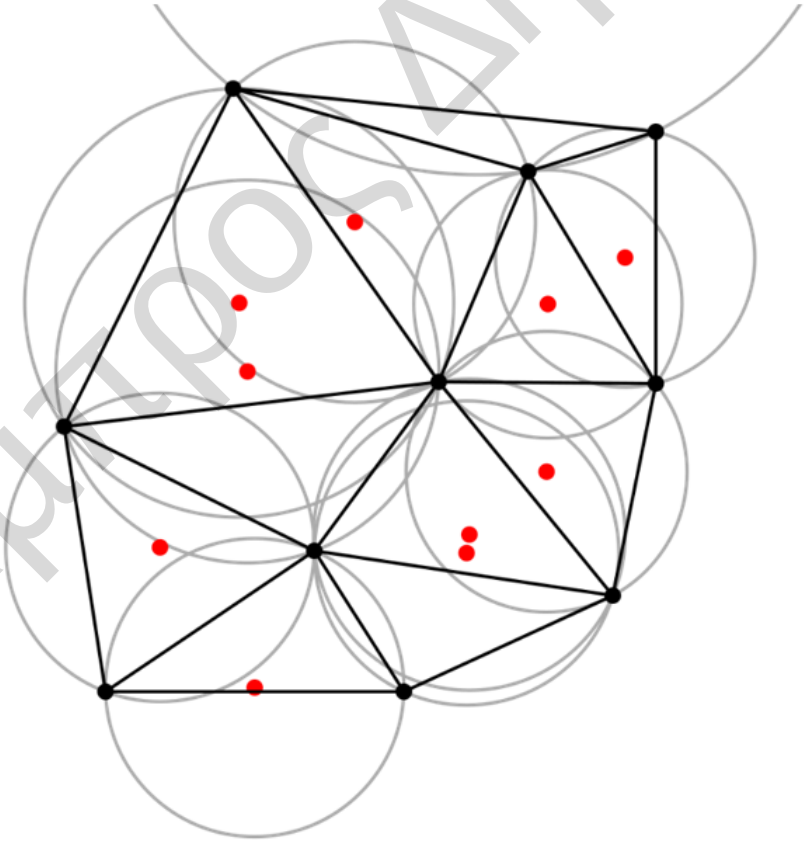
2.8 Τριγωνισμός Delaunay (DT)

Στα μαθηματικά και την υπολογιστική γεωμετρία, ένας τριγωνισμός Delaunay (DT), επίσης γνωστός ως τριγωνισμός Delone, για ένα δεδομένο σύνολο S διακριτών σημείων p_i, γενικά είναι ένας τριγωνισμός τέτοιος ώστε κανένα σημείο p_i του συνόλου S να μην βρίσκεται μέσα στον κύκλο οποιουδήποτε τριγώνου που δημιουργείται από τα σημεία του συνόλου S.

Πιο συγκεκριμένα, έστω τα σημεία a, b, c του συνόλου S. Έστω το τρίγωνο T_{abc} με κορυφές τα σημεία a, b, c. Σχεδιάζοντας τον μοναδικό κύκλο C_{abc} που προκύπτει, ο οποίος περιέχει στην περιφέρεια του τα σημεία a, b, c τότε δεν μπορεί να υπάρξει ένα άλλο σημείο d του συνόλου S, το οποίο να βρίσκεται στο εσωτερικό του κύκλου C_{abc}.

Για ένα σύνολο σημείων στην ίδια ευθεία δεν υπάρχει τριγωνισμός Delaunay (η έννοια του τριγωνισμού είναι εκφυλισμένη για αυτήν την περίπτωση). Για τέσσερα ή περισσότερα σημεία του ίδιου κύκλου (π.χ. οι κορυφές ενός ορθογωνίου) ο τριγωνισμός Delaunay δεν είναι μοναδικός: καθένας από τους δύο πιθανούς τριγωνισμούς που χωρίζουν το τετράγωνο σε δύο τρίγωνα ικανοποιεί την "συνθήκη Delaunay", δηλ. την απαίτηση ότι οι κυκλικοί κύκλοι όλων των τριγώνων έχουν άδειο εσωτερικό.

Με λίγα λόγια, σε ένα ευκλείδειο επίπεδο που περιέχει ένα αριθμό n από σημεία, για να ικανοποιείται ο τριγωνισμός Delaunay, τότε θα πρέπει να δημιουργηθούν τρίγωνα με κορυφές σημεία του ευκλείδειου επιπέδου και ακολούθως να δημιουργηθούν κύκλοι γύρω από το κάθε τρίγωνο. Δηλαδή, για κάθε τρίγωνο θα δημιουργηθεί ένας κύκλος ο οποίος θα περιέχει στην περιφέρεια του τις κορυφές του τριγώνου (σημεία του ευκλείδειου επιπέδου). Αν στο εσωτερικό αυτών των κύκλων δεν περιέχεται άλλο σημείο του ευκλείδειου επιπέδου, τότε ικανοποιείται η συνθήκη του τριγωνισμού Delaunay.



Σχήμα 2.7: Ο τριγωνισμός Delaunay με όλους τους κυκλικούς κύκλους και τα κέντρα τους (με κόκκινο). [7]

2.9 Χρήσιμα Θεωρήματα Βιβλιογραφίας

2.9.1 Ο γράφος Gabriel είναι υπογράφος του τριγωνισμού Delaunay

Όπως αποδείχθηκε στη βιβλιογραφία, το γράφημα Gabriel (Gabriel graph), είναι ένα υπογράφημα του τριγωνισμού Delaunay (Delaunay Triangulation). Αυτό σημαίνει ότι οι ιδιότητες και τα χαρακτηριστικά του τριγωνισμού Delaunay μπορούν να εφαρμοστούν και στο γράφημα Gabriel. Αυτές οι ιδιότητες έχουν χρησιμοποιηθεί για την απόδειξη που ακολουθεί σε επόμενα κεφάλαια. [14]

Το γράφημα του Gabriel σε ένα σύνολο κορυφών V είναι υπογράφημα του τριγωνισμού Delaunay για το V . Επιπλέον, η ακμή AB του τριγωνισμού Delaunay είναι μια ακμή του γραφήματος Gabriel εάν η ευθεία γραμμή που ενώνει το A με το B τέμνει το τμήμα οριακής γραμμής που είναι κοινό στα πολύγωνα Thiessen για το A και το B σε σημείο διαφορετικό από τα τελικά σημεία αυτού του τμήματος οριακής γραμμής. [14]

• **Απόδειξη:**

Για τα A, B ∈ V, ας υποθέσουμε ότι το μέσο της ευθείας μεταξύ A και B δεν έχει άλλες κορυφές του V τόσο κοντά όσο το A και το B. Τότε, ένα τμήμα της κάθετης διχοτόμου του AB, που εκτείνεται και στις δύο πλευρές του μέσου του AB, πρέπει να αποτελεί μέρος του ορίου των πολυγώνων Thiessen για το A και το B, και αντίστροφα. [14]

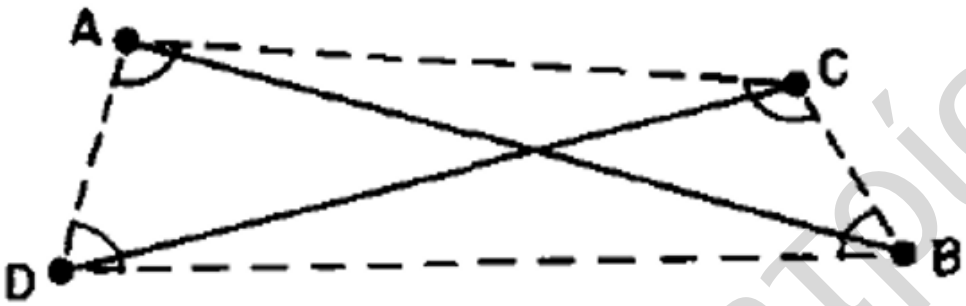
2.9.2 Ο γράφος Gabriel είναι και επίπεδος γράφος

Ακόμη ένα θεώρημα που αποδείχθηκε στη βιβιογραφία, είναι το ότι το γράφημα Gabriel (Gabriel graph), είναι και γράφημα Planar (Planar graph). Αυτό σημαίνει ότι οι ιδιότητες και τα χαρακτηριστικά του γραφήματος Panar μπορούν να εφαρμοστούν και στο γράφημα Gabriel. Αυτές οι ιδιότητες έχουν χρησιμοποιηθεί για την απόθειξη που ακολουθεί σε επόμενα κεφάλαια. [14]

• **Απόδειξη:**

Ας υποθέσουμε ότι οι γραμμές AB και CD ενός διαγράμματος γραφήματος Gabriel τέμνονται σε ένα εσωτερικό σημείο (βλ. εικόνα 2.8). Το άθροισμα των γωνιών ACB, CBD, BDA και DAC

πρέπει να ισούται με 360°, επομένως τουλάχιστον μία από αυτές, ας πούμε ACB, πρέπει να είναι τουλάχιστον 90°. Αλλά τότε το AB δεν είναι μια ακμή του γραφήματος Gabriel βάση των κριτηρίων του που αναφέρθηκαν πιο πάνω. Αυτή η αντίφαση αποδεικνύει το γεγονός ότι οποιοδήποτε γράφημα Gabriel είναι ένα επίπεδο γράφημα (Planar graph). [14]



Σχήμα 2.8: Απόδειξη ότι το γράφημα Gabriel είναι και γράφημα Planar. [14]

.

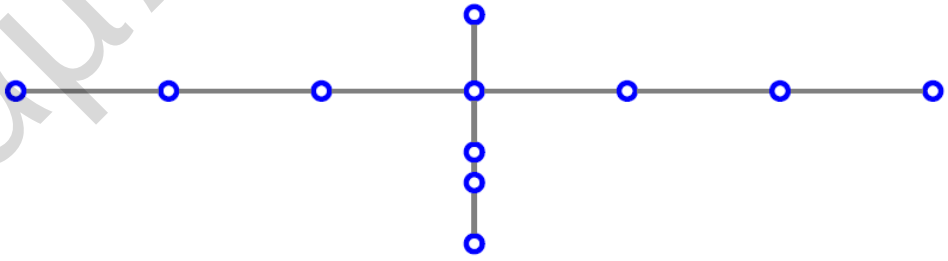
2.10 Σχετική Δουλειά

Ο σκοπός αυτής της διπλωματικής εργασίας ήταν να μελετηθούν αλγόριθμοι συγκέντρωσης (Gathering algorithms) συνεχούς χρόνου στις δύο διαστάσεις που υπάρχουν στη βιβλιογραφία, και να εξεταστούν κατά πόσο είναι collision-less. Επιπλέον, έγινε προσπάθεια να αποδειχθεί κατά πόσο είναι collision-less ένας από αυτούς τους αλγόριθμους (GTGC).

Όσο αφορά τον αλγόριθμο Go-To-The-Center (GTC), έχει ήδη αποδειχθεί ότι δεν είναι collision-less τόσο στη μια διάσταση όσο και στις δύο διαστάσεις του ευκλείδειου επιπέδου. [1]

Σχετικά με τον αλγόριθμο Go-To-The-Relative-Center (GTRC), και αυτός έχει αποδειχθεί ότι δεν είναι collision-less. Συγκεκριμένα, έχει αποδειχθεί ότι οι συγκρούσεις γίνονται μόνο σε ορισμένα σημεία, τα οποία ονομάστηκαν σημεία πρόσκρουσης (crash points). [2]

Για τον αλγόριθμο Go-To-The-Gabriel-Center (GTGC), έχει αποδειχθεί ότι είναι collision-less σε συνεχή χρόνο στη μία διάσταση του ευκλείδειου επιπέδου. Όσο αφορά τις δύο διαστάσεις, μέσω προσομοιώσεων, έχει δειχθεί ότι ίσως να είναι collision-less. Μόνο μια αρχική διάταξη (cross-shaped graph) (βρ. εικόνα 2.9) έχει δειχθεί μέσω προσομοιώσεων ότι προκαλούνται early collisions. Για αυτό, εικάζεται ότι ο συγκεκριμένος αλγόριθμος μπορεί να είναι collision-less αν δεν βρίσκεται σε αυτή την αρχική διάταξη. Σε περίπτωση που βρίσκεται σε αυτή τη διάταξη (cross-shaped), αναφέρεται ότι αν ¨ταρακουνηθεί¨ (¨shake¨) ελάχιστα η διάταξη έτσω ώστε τα ρομπότ να μετακινηθούν ελάχιστα σε τυχαία σημεία, τότε δεν θα υπάρξουν συγκρούσεις. [1]



Σχήμα 2.9: Ένα παράδειγμα του γραφήματος σε σχήμα σταυρού που οδηγεί στις πρώιμες συγκρούσεις με τον αλγόριθμο GTGC. [1]

.

Ένας αλγόριθμος που έχει παρουσιαστεί και αποδειχθεί ότι είναι collision-less στις δύο διαστάσεις του ευκλείδειου επιπέδου σε συνεχές χρόνο, είναι ο αλγόριθμος Safe-Go-To-

The-Relative-Center (S-GTRC). Αυτός ο αλγόριθμος είναι μια νέα έκδοση του αλγορίθμου Go-To-The-Relative-Center (GTRC), παρόλα αυτά γίνεται χρήση κάποιων επιπρόσθετων δυνατοτήτων additional capabilities πάνω στα ρομπότ, που οι προηγούμενοι αλγόριθμοι δεν τις είχαν. Για παράδειγμα, σε αυτό τον αλγόριθμο έχει αυξηθεί το viewing range από 1 σε 2. Αυτό χρειάστηκε για να αποφεθχθούν οι συγκρούσεις και να κάνει τον αλγόριθμο collision-less. [1]

Κεφάλαιο 3

Ανάλυση Περιπτώσεων

3.1 Εισαγωγή

Σε αυτό το κεφάλαιο παρουσιάζεται ο κυριότερος στόχος αυτής της διπλωματικής εργασίας, που ήταν να γίνει μια προσπάθεια να αποδειχθεί κατά πόσο ο αλγόριθμος Go-To-The-Gabriel-Center (GTGC) που είναι ένας αλγόριθμος συγκέντρωσης (gathering algorithm), είναι collision-less στις δύο διαστάσεις του ευκλείδειου επιπέδου και σε συνεχή χρόνο.

• **Για την απόδειξη θα ακολουθηθούν τα πιο κάτω βήματα:**

Θα χρησιμοποιηθούν χαρακτηριστικά και ιδιότητες από τη βιβλιογραφία και κυρίως:

- του ελάχιστου κύκλου ενγκλεισμού (MEC - Minimal Enclosing Circle)
- του γράφου Gabriel (Gabriel graph)
- του Planar graph, αφού υπάρχει θεώρημα που αποδεικνύει ότι ο Gabriel graph είναι και Planar graph
- του τριγωνισμού Delaunay (Delaunay triangulation), αφού υπάρχει θεώρημα που αποδεικνύει ότι το γράφημα Gabriel είναι υπογράφος (subgraph) του τριγωνισμού Delaunay

Ο αλγόριθμος που εκτελείται είναι κοινός για όλα τα ρομπότ. Θα γίνει η ανάλυση σε χρονικά διαστήματα (time intervals) της εκτέλεσης του αλγορίθμου σε ένα ρομπότ. Δηλαδή, θα εξεταστούν όλες οι καταστάσεις στις οποίες μπορεί να βρεθεί ένα ρομπότ (απεικονίζοντας το στιγμιότυπο της κατάστασης) και θα αναλυθεί η τροχιά (trajectory) που θα ακολουθήσει έτσι ώστε να φτάσει στο σημείο προορισμού του (target point). Με αυτό τον τρόπο, θα γίνει προσπάθεια να αποδειχθεί ότι δεν μπορεί να υπάρξουν τροχιές των ρομπότ που θα διασταυρωθούν (crossing trajectories, να υπάρχει σημείο τομής) ή έστω αν υπάρξουν, τότε δεν θα βρίσκονται ποτέ την ίδια χρονική στιγμή 2 ρομπότ στο σημείο τομής.

• **Για την απόδειξη, έχουμε τις πιο κάτω υποθέσεις οι οποίες αναγράφονται και στο άρθρο που παρουσιάζεται ο αλγόριθμος:**

- όλα τα ρομπότ κινούνται με μέγιστη ταχύτητα 1
- όλα τα ρομπότ έχουν viewing range 1
- όλα τα ρομπότ όταν κινούνται προς το σημείο προορισμού τους (target point), κινούνται σε ευθεία γραμμή. Δηλαδή, η τροχιά κίνησης τους είναι μια ευθεία γραμμή

• **Συγκρούσεις που γίνονται στα σημεία προορισμού (target points) ονομάζονται σημεία πρόσκρουσης (crash points) και είναι αποδεχτές λόγο της συγκέντρωσης (gathering).** [2]

Η απόδειξη έχει χωριστεί σε δύο ανεξάρτητα μέρη - σενάρια ανάλογα με την θέση του ρομπότ που εκτελεί τον αλγόριθμο σε σχέση με τον ελάχιστο κύκλο κάλυψης (MEC). Τα δύο σενάρια που πρέπει να εξεταστούν (καταστάσεις στις οποίες μπορεί να βρεθεί ένα ρομπότ) είναι τα ακόλουθα:

1. Το ρομπότ r1 που εκτελεί τον αλγόριθμο, βρίσκεται στην περιφέρεια του ελάχιστου κύκλου κάλυψης (MEC - Minimal Enclosing Circle)
2. Το ρομπότ r1 που εκτελεί τον αλγόριθμο, βρίσκεται στο εσωτερικό του ελάχιστου κύκλου κάλυψης (MEC - Minimal Enclosing Circle)

3.2 Σενάριο 1 - Το r1 βρίσκεται στην περιφέρεια του MEC

3.2.1 Εισαγωγή

Σε αυτό το σενάριο, το ρομπότ r1 (εκτελεί τον αλγόριθμο) είναι δεδομένο ότι θα βρίσκεται στην περιφέρεια του MEC ο οποίος υπολογίστηκε από το σημείο 2 του αλγορίθμου Go-To-The-Gabriel-Center (GTGC, βλ. αλγόριθμο εδώ: 2.2.3). Σε αυτό το σενάριο, το MEC μπορεί να έχει το πολύ 3 ρομπότ στην περιφέρεια του, βάση των ιδιοτήτων του ελάχιστου κύκλου κάλυψης (MEC, βλ. χαρακτηριστικά και ιδιότητες εδώ: 2.4). Οι καταστάσεις που μπορεί να προκύψουν από αυτό το σενάριο αναλύονται στα ακόλουθα υποκεφάλαια.

3.2.2 Περίπτωση 1 - Στην περιφέρεια του MEC βρίσκονται 2 ρομπότ

Σε αυτή την περίπτωση, το ρομπότ r1 που εκτελεί τον αλγόριθμο, βρίσκεται στην περιφέρεια του MEC και επίσης, στην περιφέρεια βρίσκεται ακόμη ένα άλλο ρομπότ r2. Συνολικά, στην περιφέρια του MEC βρίσκονται 2 ρομπότ, ένα εκ των οποίων εκτελεί τον αλγόριθμο.

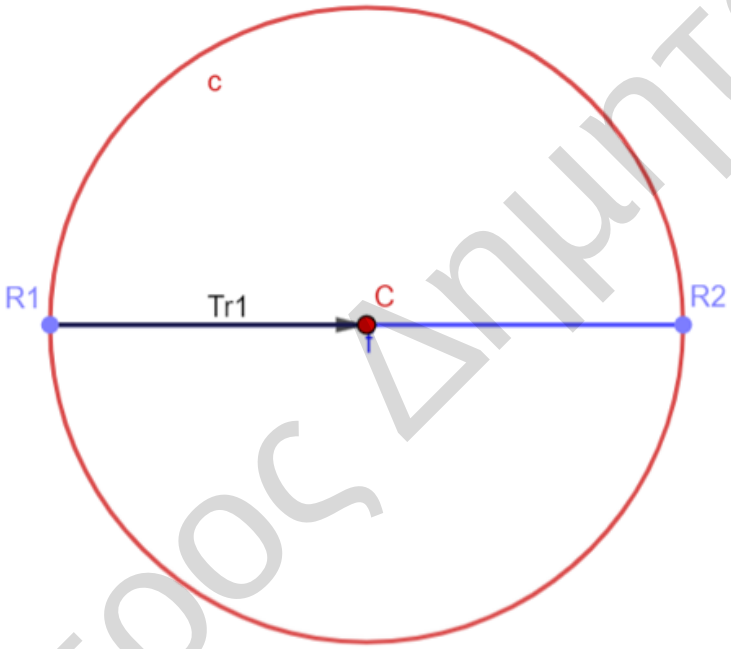
Σε αυτή την περίπτωση, είναι εύκολο να δειχθεί ότι η διάμετρος του MEC είναι η ευθεία γραμμή που προκύπτει από την ένωση των σημείων που βρίσκονται τα ρομπότ r1 και r2. Επίσης, μπορεί εύκολα να αποδειχθεί ότι οι γείτονες του ρομπότ r1 είναι **ΜΟΝΟ** το ρομπότ r2. Αν είχε και άλλους γείτονες, τότε είναι δεδομένο ότι το ρομπότ r3 δεν θα μπορούσε να βρίσκεται στο εσωτερικό του MEC επειδή αν βρισκόταν, δεν θα ίσχυε το κριτήριο 1 του Gabriel graph (βλέπε εδώ: 2.5) και έτσι, το r2 δεν θα θεωρείται γείτονας του r1.

Επίσης, είναι δεδομένο ότι το r1 δεν μπορεί να έχει γείτονα άλλο ρομπότ το οποίο βρίσκεται στο εξωτερικό του MEC, επειδή με βάση τον αλγόριθμο (βλ. εδώ: 2.2.3) θα προέκυπτε διαφορετικό MEC και όχι αυτός, αφού όλοι οι γείτονες του r1 πρέπει να βρίσκονται στο εσωτερικό ή στην περιφέρεια του MEC. Το μόνο σενάριο που δεν αναλύθηκε είναι η περίπτωση όπου ένα ρομπότ r3 βρίσκεται στην περιφέρεια του MEC όπου μπορεί να προκύψει αυτό το σενάριο και είναι αυτό που θα αναλυθεί στην κατάσταση 2.

Από τα πιο πάνω προκύπτει ότι το σημείο προορισμού (TP - target point του r1 στο οποίο θα πρέπει να κινηθεί είναι το κέντρο του MEC όπου είναι και το κέντρο της ευθείας γραμμής που ενώνει τα 2 ρομπότ.

Είναι ξεκάθαρο ότι από την τροχιά που θα ακολουθήσει το r1 για να φτάσει στο T(r1) δεν

μπορεί να προκύψει κάποια σύγκρουση αφού δεν θα υπάρξει κάποιο τρίτο ρομπότ που θα περάσει από την τροχιά που θα ακολουθήσει το r1. Η μόνη δυνατή σύγκρουση που μπορεί να προκύψει σε αυτή την κατάσταση και αυτό το σενάριο είναι η σύγκρουση μεταξύ του r1 και r2 στο σημείο προορισμού. Αυτό προϋποθέτει ότι το r2 δεν έχει άλλους γείτονες παρά το r1 (για να έχει τον ίδιο MEC και συνάμα το ίδιο T(r2)) και επίσης ότι τα 2 ρομπότ θα κινηθούν ακριβώς την ίδια χρονική στιγμή προς το σημείο προορισμού (αφού κινούνται με σταθερή ταχύτητα). Αυτή η σύγκρουση είναι αποδεχτή αφού είναι στο σημείο όπου γίνεται η συγκέντρωση - gathering (crash points).



Σχήμα 3.1: Απεικόνιση Σεναρίου 1 - Κατάστασης 1

.

Όπως φαίνεται και από την πιο πάνω εικόνα (βλ. εικόνα 3.1), στο εσωτερικό του κύκλου (MEC) δεν υπάρχει άλλο ρομπότ (δεν μπορεί να υπάρξει επειδή αν υπήρχε θα είχαμε διαφορετικό MEC) και στην περιφέρεια του υπάρχουν τα ρομπότ r1 και r2. Σε αυτή την κατάσταση, ο μοναδικός γείτονας του r1 είναι το r2 (αλλιώς θα είχαμε διαφορετικό MEC). Άρα είναι προφανές ότι δεν μπορεί να υπάρξει κάποια σύγκρουση του ρομπότ r1 με κάποιο άλλο ρομπότ αφού δεν πρόκειται να περάσει κάποιο άλλο ρομπότ από την τροχιά του r1 (μαύρη γραμμή). Στην περίπτωση που και το r2 έχει ως μοναδικό γείτονα το r1, τότε από τον αλγόριθμο θα προκύψει ότι το r1 και r2 θα έχουν τον ίδιο MEC και κατ' επέκταση το ίδιο σημείο προορισμού. Στην περίπτωση όπου ξεκινήσουν να κινούνται ακριβώς την ίδια χρονική στιγμή (σταθερή

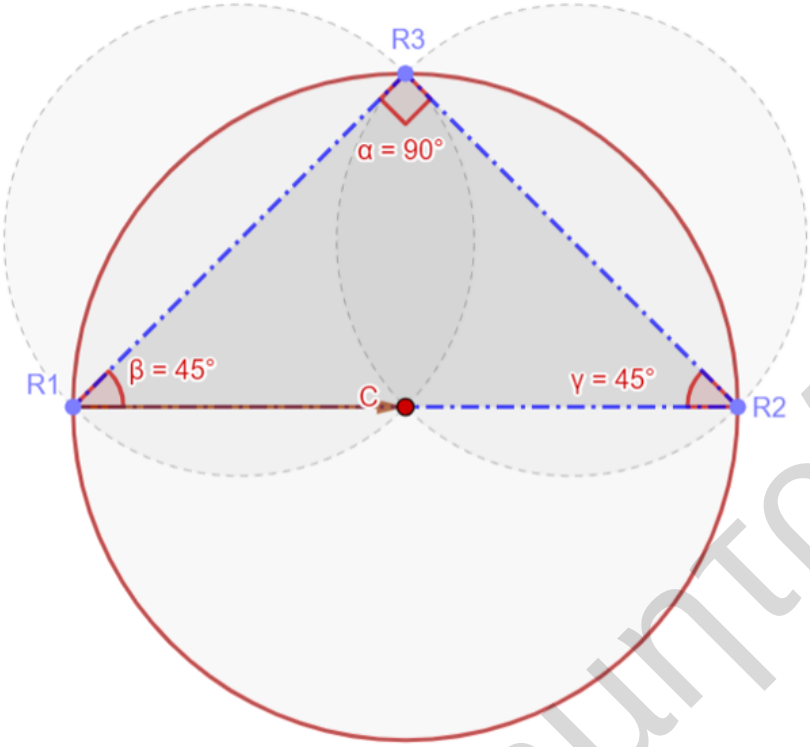
ταχύτητα) τότε θα υπάρχει σύγκρουση στο σημείο προορισμού το οποίο είναι το κέντρο του MEC. Αυτή η σύγκρουση είναι αποδεχτή αφού θα προκύψει στο σημείο προορισμού (crash point).

3.2.3 Περίπτωση 2 - Στην περιφέρεια του MEC βρίσκονται 3 ρομπότ

Σε αυτή την περίπτωση, το ρομπότ r1 που εκτελεί τον αλγόριθμο, βρίσκεται στην περιφέρεια του MEC και επίσης, στην περιφέρεια βρίσκεται ακόμη δύο άλλα ρομπότ r2 και r3. Συνολικά, στην περιφέρια του MEC βρίσκονται 3 ρομπότ, ένα εκ των οποίων εκτελεί τον αλγόριθμο.

Όπως αναφέρθηκε και πιο πάνω (βλ. χαρακτηριστικά και ιδιότητες MEC: 2.4), στην περιφέρεια του MEC, ο μέγιστος αριθμός ρομπότ που μπορεί να προκύψουν είναι 3. Αυτό ισχύει λόγο της ιδιότητας του MEC που λέει ότι το τρίγωνο που προκύπτει από την ένωση των 3 σημείων στην περιφέρεια του, πρέπει να είναι οξύ, δηλαδή όλες οι γωνιές να είναι $< 90°$. Όπως παρατηρήθηκε από τις προσομοιώσεις των καταστάσεων, δεν μπορεί να προκύψει ένα άλλο 4ο ρομπότ r4 το οποίο να βρίσκεται στο εσωτερικό του MEC όταν υπάρχουν είδη άλλα 3 ρομπότ στην περιφέρεια. Επίσης, όπως είδη αναφέρθηκε ούτε στην περιφέρεια του MEC μπορεί να υπάρξει ένα άλλο 4ο ρομπότ r4 επειδή, δεν θα ικανοποιείται η συνθήκη του MEC που πρέπει να ισχύει ότι το τρίγωνο που δημιουργείται μεταξύ των σημείων πρέπει να είναι οξύ.

Πιο κάτω, θα δειχθούν σε μορφή γραφημάτων οι διάφορες καταστάσεις που μπορεί να προκύψουν σε ένα τέτοιο σενάριο και θα αποδειχθεί γιατί δεν μπορεί να υπάρχει collision μεταξύ του ρομπότ r1 και κάποιου άλλου ρομπότ.



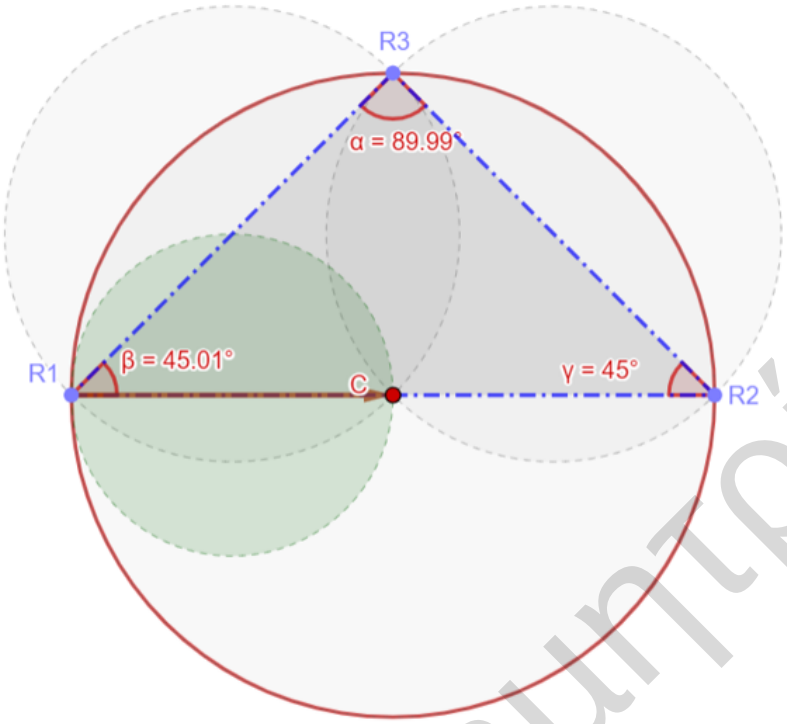
Σχήμα 3.2: Απεικόνιση Σεναρίου 1 - Κατάστασης 2.1

.

Στην πιο πάνω εικόνα (βλ. εικόνα Α΄.7), παρατηρείται ότι το τρίγωνο που προκύπτει μεταξύ των σημείων R1, R2, R3 είναι ορθογώνιο τρίγωνο. Αυτό το σενάριο δεν μπορεί να προκύψει επειδή με αυτό τον τρόπο καταρρίπτεται η συνθήκη του MEC που λέει ότι το τρίγωνο που προκύπτει είναι οξύ (βλ. χαρακτηριστικά και ιδιότητες MEC: 2.4). Σε αυτή την περίπτωση δεν είναι οξύ αλλά ορθογώνιο. Επίσης, ακόμη ένα κριτήριο που αποδεικνύει ότι δεν μπορεί να προκύψει αυτό το σενάριο είναι το κριτήριο 1 από τα κριτήρια του γραφήματος Gabriel (βλ. 2.5) όπου καθορίζονται οι γείτονες του r1, όπου αναφέρεται ότι ο κλειστός δίσκος που περικλείει τα σημεία r1 και r2 δεν πρέπει να περιέχει άλλο ρομπότ. Σε αυτή την περίπτωση, περιέχει το ρομπότ r3 στην περιφέρεια του.

Αποδείχθηκε με βάση τα χαρακτηριστηκά και ιδιότητες του MEC και βάση των κριτηρίων για τον καθορισμό των γειτόνων του γραφήματος Gabriel, ότι το πιο πάνω σενάριο δεν μπορεί να προκύψει από τον αλγόριθμο. Έπεται ότι δεν θα υπάρχουν collisions.

Στην κατάσταση που παρουσιάζεται στην πιο πάνω εικόνα (βλ. εικόνα 3.3), παρατηρείται ένα πιθανό σενάριο αφού πληρούνται όλα τα κριτήρια του MEC (βλ. χαρακτηριστικά και ιδιότητες MEC: 2.4) και του γραφήματος Gabriel (βλ. 2.5). Δηλαδή, το τρίγωνο που προκύπτει



Σχήμα 3.3: Απεικόνιση Σεναρίου 1 - Κατάστασης 2.2

μεταξύ των 3ων σημείων είναι οξύ (< 90°) και επίσης, οι δίσκοι που περικλείουν τα σημεία R1R3 και R1R2 αντίστοιχα, δεν περιέχουν άλλα ρομπότ στο εσωτερικό τους. Από τη στιγμή που αποδείχθηκε ότι μπορεί να προκύψει μια τέτοια κατάσταση, θα διερευνηθεί κατά πόσο μπορούν να υπάρξουν collisions.

Βάση του πιο πάνω γραφήματος (βλ. εικόνα 3.3), και λαμβάνοντας υπόψιν τον αλγόριθμο (βλ. αλγόριθμο 2.2.3), το R1 θα κινηθεί προς το σημείο προορισμού του target point που είναι το κέντρο του MEC (σημείο C). Η κίνηση του R1 θα είναι σε ευθεία γραμμή και με μέγιστη ταχύτητα 1 (βάση των πιο πάνω υποθέσεων: 3.1). Άρα, η τροχιά που θα ακολουθήσει το R1, θα είναι η διάμετρος του πράσινου κύκλου που φαίνεται πιο πάνω. Για να ελεγχθεί κατά πόσο μπορεί να προκύψουν collisions, θα πρέπει να διερευνηθεί αν θα υπάρξει κάποιο ρομπότ που θα περάσει από την τροχιά του R1.

Στο εσωτερικό του MEC δεν μπορεί να υπάρξει άλλο ρομπότ, επειδή, αν υπήρχε τότε δεν θα ίσχυε το κριτήριο του γραφήματος Gabriel (βλ. 2.5). Όπως φαίνεται και στην εικόνα (βλ. εικόνα 3.3), όλη η περιοχή του εσωτερικού του MEC είναι γκρίζα, που σημαίνει ότι δεν μπορεί να υπάρξει άλλο ρομπότ στο εσωτερικό του. Αν υπήρχε, τότε θα είχαμε διαφορετικό

MEC και άλλους γείτονες του R1 και κατ' επέκταση διαφορετικό γράφημα Gabriel.

Δεν μας ενδιαφέρουν τα ρομπότ που βρίσκονται στο εξωτερικό του MEC, επειδή είναι δεδομένο ότι δεν είναι γείτονες με το R1 (αν ήταν θα έπρεπε να βρίσκονται στο εσωτερικό του MEC και έτσι θα είχαμε διαφορετικό MEC), άρα δεν θα έχουν τροχιά προς το εσωτερικό του MEC.

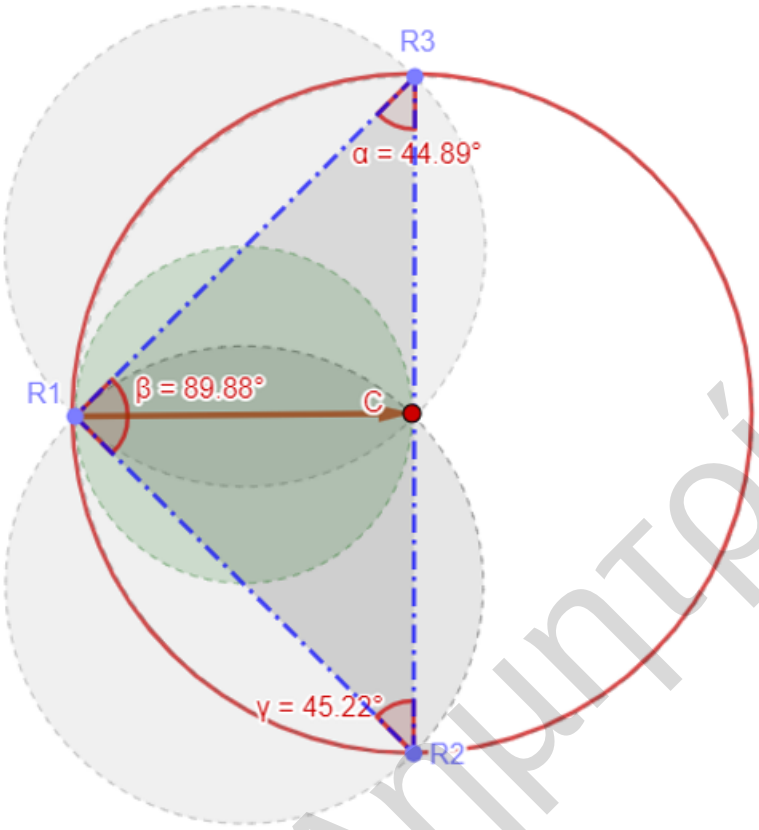
Επίσης, ο πράσινος κύκλος πιο πάνω, μας υποδηλώνει το safe zone του R1. Για να υπήρχε collision, θα έπρεπε να υπάρχει κάποιο ρομπότ στο εσωτερικό ή στην περίμετρο του πράσινου κύκλου το οποίο θα διασταύρωνε την τροχιά του R1. Όπως ήδη έχουμε αποδείξει, δεν μπορεί να υπάρξει τέτοιο σενάριο, άρα έπεται ότι δεν θα υπάρξουν collision ούτε σε μια τέτοια κατάσταση.

Πιο κάτω, στα παραρτήματα (βλ. Παράρτημα Α'), απεικονίζονται διάφορα γραφήματα μιας τέτοιας κατάστασης με διαφορετικές θέσεις των ρομπότ πάνω στην περιφέρεια του MEC, οι οποίες αποδεικνύουν το ότι δεν μπορεί να υπάρξει κάποιο collision σε μια τέτοια κατάσταση. Η απόδειξή έγινε πιο πάνω.

3.2.4 Περίπτωση 3 - Στην περιφέρεια του MEC βρίσκονται 3 ρομπότ και στο εσωτερικό του υπάρχουν ρομπότ

Σε αυτή την περίπτωση, υπάρχουν 3 ρομπότ στην περιφέρεια του MEC όπως και στην κατάσταση 2, αλλά σε αυτή την κατάσταση μπορεί να υπάρξει και ρομπότ στο εσωτερικό του MEC.

Αυτή η περίπτωση μπορεί να τύχει όταν η γωνία του τριγώνου R2R1R3 (η γωνία του ρομπότ R1) είναι κοντά στις 90° (βλ. εικόνα 3.4). Όσο πιο κοντά στις 90° βρίσκεται η γωνία του ρομπότ R1 και οι γωνίες των ρομπότ R2 και R3 είναι κοντά στις 45°, τόσο πιο μεγάλο εμβαδό θα υπάρχει στο εσωτερικό του MEC (άσπρο χρώμα) όπου μπορεί να υπάρξει ένα άλλο 4ο ρομπότ. Αυτό που επεξηγήθηκε, αναπαριστάται στην ακόλουθη εικόνα (βλ. εικόνα 3.4) και έτσι μπορεί να κατανοηθεί καλύτερα.



Σχήμα 3.4: Απεικόνιση Σεναρίου 1 - Κατάστασης 3.1

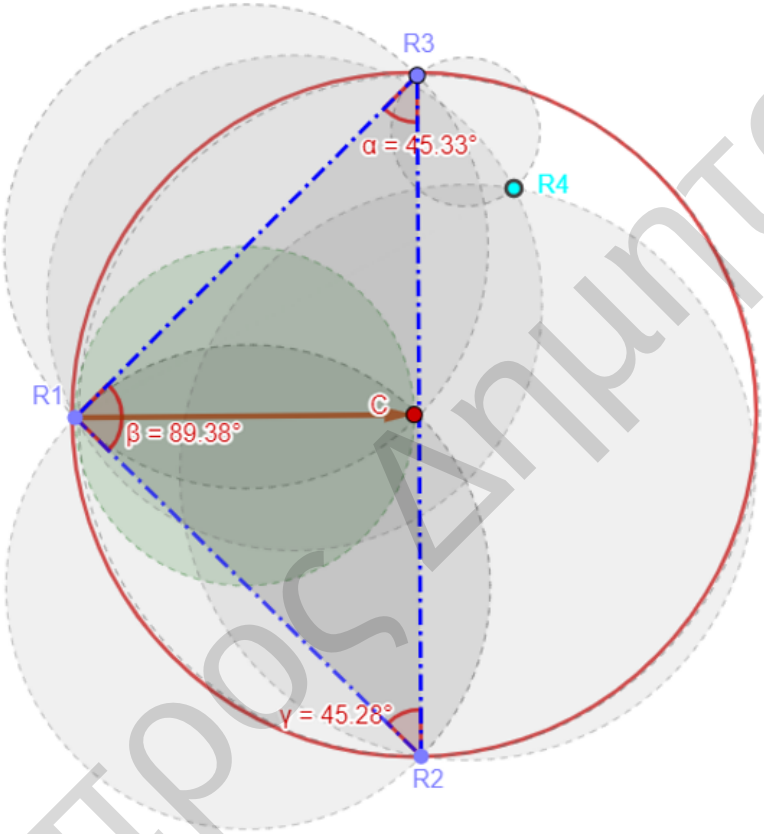
Στην πιο πάνω εικόνα (βλ. εικόνα 3.4), στο δεξί ημισφαίριο του MEC (άσπρο χρώμα), παρατηρείται η περιοχή όπου μπορεί να υπάρξει κάποιο άλλο ρομπότ.

Παρόλα αυτά, βάση του θεωρήματος που αναφέρει ότι το γράφημα Gabriel είναι ένα υπογράφημα subgraph του τριγωνισμού Delaunay (βλ. θεώρημα 2.9.1), δεν μπορεί να προκύψει το πιο πάνω σενάριο, επειδή ο τριγωνισμός Delaunay μας λέει ότι ο κύκλος που επικαλύπτει τις 3 κορυφές του τριγώνου που δημιουργούνται από τα 3 σημεία, δεν μπορεί να περιέχει άλλο σημείο στο εσωτερικό του. Στην προκειμένη περίπτωση, ο εν λόγο κύκλος είναι και ο MEC.

Οπόταν, με αυτό τον τρόπο αποδεικνύετε ότι σε ένα σενάριο όπου υπάρχουν 3 ρομπότ στην περιφέρεια του MEC, δεν μπορεί να υπάρξει 4ο ρομπότ στο εσωτερικό του. Έπεται ότι δεν θα υπάρξει σύγκρουση μεταξύ του R1 και κάποιου 4ου ρομπότ.

3.2.4.1 Διερεύνηση συγκρούσεων ΑΝ ΔΕΝ ΙΣΧΥΕ το θεώρημα: ¨Ο γράφος Gabriel είναι υπογράφος του τριγωνισμού Delaunay¨

Σε αυτό το υποκεφάλαιο, θα αναλυθεί κατά πόσο θα μπορούσε να υπάρχει κάποια σύγκρουση μεταξύ R1 και R4 **ΑΝ ΔΕΝ ΙΣΧΥΕ** το θεώρημα που αναφέρει ότι το γράφημα Gabriel είναι ένα υπογράφημα του τριγωνισμού Delaunay (βλ. θεώρημα 2.9.1) και έτσι θα μπορούσε να υπάρξει ένα 4ο ρομπότ στο εσωτερικό του MEC.



Σχήμα 3.5: Απεικόνιση Σεναρίου 1 - Κατάστασης 3.2

.

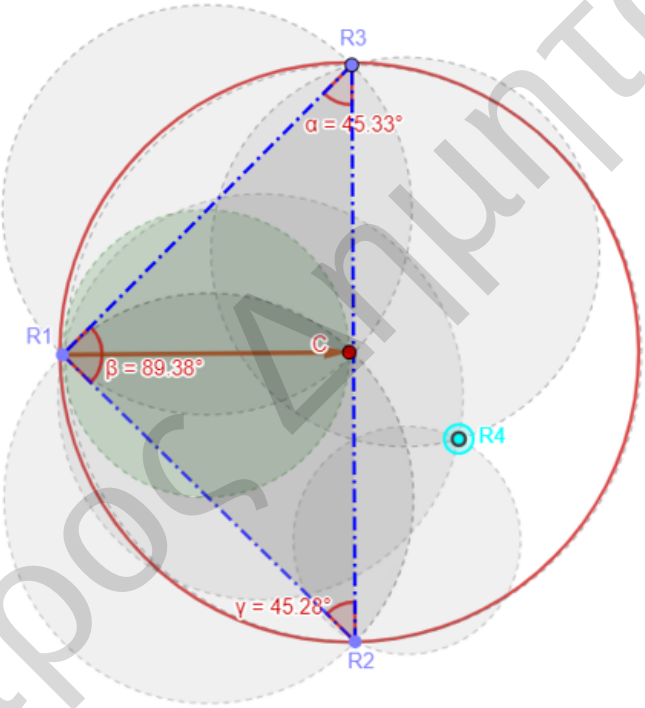
Στην πιο πάνω κατάσταση (βλ. εικόνα 3.5), όπου το MEC του R1 εμπεριέχει και ένα άλλο 4ο ρομπότ στο εσωτερικό του, το οποίο είναι και γείτονας με το R1, θα εξεταστεί αν μπορεί να υπάρξει collision μεταξύ R1 και R4. Το R4, εκτός από το R1, έχει και σαν γείτονες το R2 και R3 βάση του αλγορίθμου (βλ. αλγόριθμο 2.2.3).

Η μόνη περίπτωση που το R4 θα έχει διαφορετικό MEC από το R1, είναι να έχει και άλλους γείτονες. Μόνο αν έχει γείτονες στο εξωτερικό του πιο πάνω MEC, τότε θα μπορεί να έχει διαφορετικό MEC το R4. Αν ισχύει αυτή η περίπτωση, τότε είναι δεδομένο ότι το MEC του R4 θα είναι μετατοπισμένο προς τα δεξιότερα (μόνο εξωτερικά του δεξιού ημισφαιρίου μπορεί να

έχει άλλους γείτονες) και κατ' επέκταση και το σημείο προορισμού (target point) του R4 θα είναι δεξιότερα.

Αυτό σημαίνει ότι δεν μπορεί να υπάρξει σύγκρουση μεταξύ R1 και R4, επειδή, με βάση τα σημεία προορισμού τους, δεν θα διασταυρωθούν οι τροχιές τους.

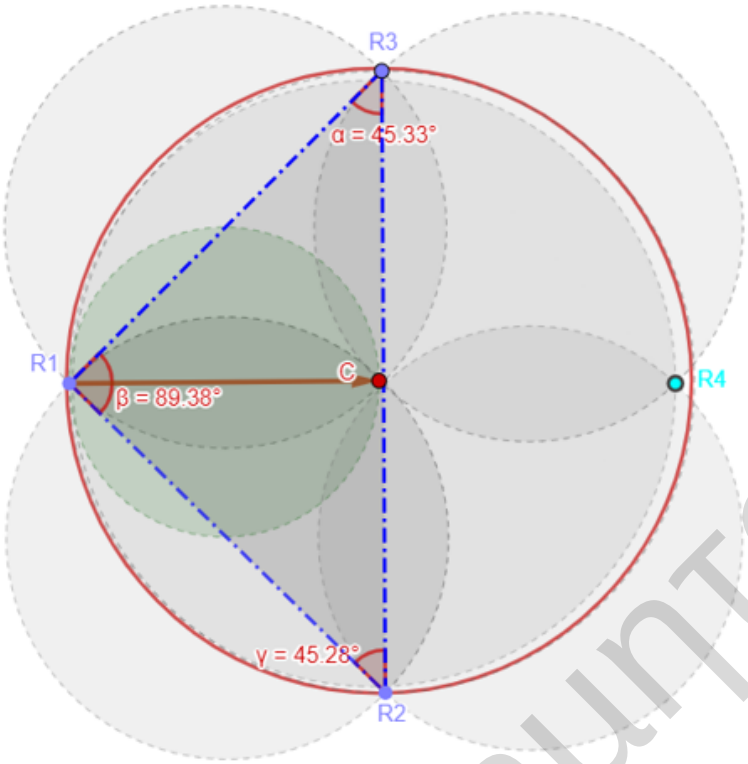
Στην περίπτωση όπου το R1 και R4 έχουν τον ίδιο MEC (και κατ' επέκταση το ίδιο σημείο προορισμού), τότε, το μόνο σημείο που μπορεί να συγκρουστούν είναι το κέντρο του MEC (σημείο προορισμού), που αυτή η περίπτωση είναι αποδεχτή όπως επεξηγήθηκε πιο πάνω (crash point, βλ. 3.1).



Σχήμα 3.6: Απεικόνιση Σεναρίου 1 - Κατάστασης 3.3

.

Στην εικόνα 3.6 πιο πάνω, παρατηρείται μια ανάλογη περίπτωση με το προηγούμενο σενάριο. Σε περίπτωση που στο εσωτερικό του MEC υπάρξει και ένα άλλο 5ο ρομπότ, τότε είναι δεδομένο με βάση τα κριτήρια του γραφήματος Gabriel (βλ. 2.5) ότι θα είναι γείτονας μόνο με το R4. Αυτό σημαίνει ότι το MEC δεν θα επηρεαστεί καθόλου και το R4 θα έχει ακριβώς το ίδιο σημείο προορισμού.



Σχήμα 3.7: Απεικόνιση Σεναρίου 1 - Κατάστασης 3.4

.

Το πιο πάνω σενάριο στην εικόνα 3.7, έχει μεγάλη πιθανότητα σύγκρουσης στο κέντρο του κύκλου σε περίπτωση που το R4 δεν έχει άλλους γείτονες πέραν του R1, R2 και R3. Παρόλα αυτά, όπως ήδη έχουμε αναφέρει, η σύγκρουση θα είναι αποδεχτή επειδή θα είναι στο σημείο προορισμού (crash point, βλ. 3.1).

Όπως ήδη αναφέρθηκε, σε αυτό το υποκεφάλαιο διερευνήθηκε κατά πόσο θα μπορούσαν να υπάρξουν συγκρούσεις **ΑΝ ΔΕΝ ΙΣΧΥΕ** το θεώρημα που αναφέρει ότι το γράφημα Gabriel είναι ένα υπογράφημα του τριγωνισμού Delaunay.

Παρόλα αυτά, αφού έχει αποδηχθεί στη βιβλιογραφία ότι ισχύει αυτό το θεώρημα (βλ. θεώρημα 2.9.1), έτσι είναι προφανές ότι τα σενάρια που παρουσιάστηκαν σε αυτό το υποκεφάλαιο δεν μπορούν να υπάρξουν, και έτσι, έπεται ότι δεν θα υπάρξουν συγκρούσεις.

3.3 Σενάριο 2 - Το r1 βρίσκεται στο εσωτερικό του MEC

3.3.1 Εισαγωγή

Σε αυτό το σενάριο, το ρομπότ r1 (εκτελεί τον αλγόριθμο) είναι δεδομένο ότι θα βρίσκεται στο εσωτερικό του MEC ο οποίος υπολογίστηκε από το σημείο 2 του αλγορίθμου Go-To-The- Gabriel-Center (GTGC, βλ. αλγόριθμο εδώ: 2.2.3). Σε αυτό το σενάριο, το MEC μπορεί να έχει το πολύ 3 ρομπότ στην περιφέρεια του, βάση των ιδιοτήτων του ελάχιστου κύκλου κάλυψης (MEC, βλ. χαρακτηριστικά και ιδιότητες εδώ: 2.4) αλλά όχι το ρομπότ r1. Οι καταστάσεις που μπορεί να προκύψουν σε αυτό το σενάριο αναλύονται στα ακόλουθα υποκεφάλαια.

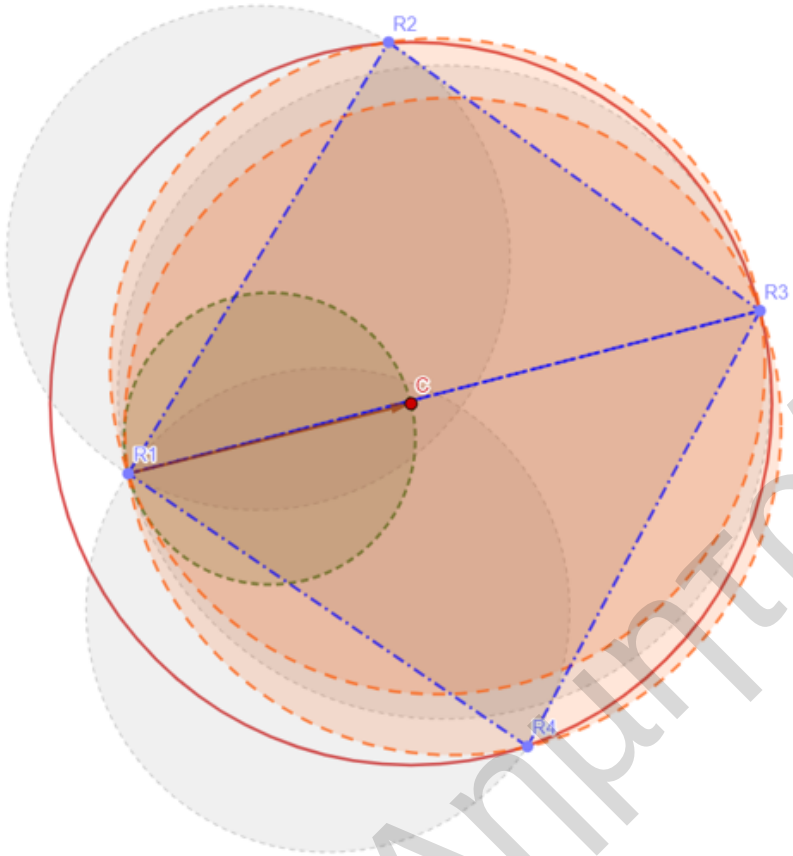
3.3.2 Περίπτωση 1 - Στην περιφέρεια του MEC υπάρχουν 3 ρομπότ που γειτνιάζουμε με το r1

Σε αυτή την περίπτωση, το ρομπότ r1 βρίσκεται στο εσωτερικό του MEC, και στην περιφέρεια του MEC υπάρχουν άλλα 3 ρομπότ (r2, r3, r4) τα οποία είναι γείτονες με το r1.

Σε αυτή την κατάσταση, είναι δεδομένο ότι τα 3 ρομπότ που είναι γείτονες του r1, δεν μπορούν να είναι και γείτονες μεταξύ τους. Δηλ.:

Το r1 βρίσκεται στο εσωτερικό του MEC και τα ρομπότ r2, r3, r4 βρίσκονται στην περιφέρεια του MEC. Είναι δεδομένο ότι το r1 έχει γείτονες το r2, r3, r4. Δεν μπορεί να ισχύουν οι ακόλουθες 3 συνθήκες ταυτόχρονα, παρά μόνο η μια από τις τρεις:

1. Το r2 έχει σαν γείτονες το r1, r3, r4
2. Το r3 έχει σαν γείτονες το r1, r2, r4
3. Το r4 έχει σαν γείτονες το r1, r2, r3



Σχήμα 3.8: Απεικόνιση Σεναρίου 2 - Κατάστασης 1.1

.

Όπως φαίνεται στην πιο πάνω εικόνα 3.8, το r1 έχει σαν γείτονες το r2, r3 και r4. Το r2 έχει σαν γείτονες το r1 και r3. Το r3 έχει σαν γείτονες το r1, r2 και r4. Το r4 έχει σαν γείτονες το r1 και r3. Άρα, από τις τρεις πιο πάνω συνθήκες, ισχύει μόνο η 2η.

Το r2 και r4 δεν μπορούν να είναι γείτονες μεταξύ τους επειδή αν ήταν τότε δεν θα ίσχυαν τα 2 κριτήρια του γραφήματος Gabriel (βλ. εδώ: 2.5).

Επίσης, βάση του θεωρήματος που μας λέει ότι ένα γράφημα Γαβριελ είναι και γράφημα πλαναρ (βλ. εδώ: 2.9.2), τότε δεν πρέπει να υπάρχουν ακμές που διασταυρώνονται μεταξύ τους σε ένα γράφημα planar. Όπως φαίνεται στην εικόνα 3.8, αν τα ρομπότ r2 και r4 ήταν γείτονες μεταξύ τους, τότε οι ακμές r1r3 και r2r4 θα διασταυρώνονταν μεταξύ τους, κάτι το οποίο δεν μπορεί να ισχύει.

Έπεται η απόδειξη ότι από τις τρεις πιο πάνω συνθήκες, μόνο η μια μπορεί να ισχύει. Συνοπτικά, σε ένα MEC το οποίο περιέχει 3 ρομπότ στην περιφέρεια του, και τα 3 αυτά ρομπότ είναι γειτονικά με ένα ρομπότ που βρίσκεται στο εσωτερικό του MEC και ο MEC

είναι αυτός του εσωτερικού ρομπότ, τότε τα 3 αυτά ρομπότ δεν μπορούν να γειτονεύουν όλα μεταξύ τους. Τουλάχιστον τα 2 από τα 3 ρομπότ, δεν θα γειτονεύουν μεταξύ τους (βλ. εικόνα 3.8, ρομπότ r2, r4).

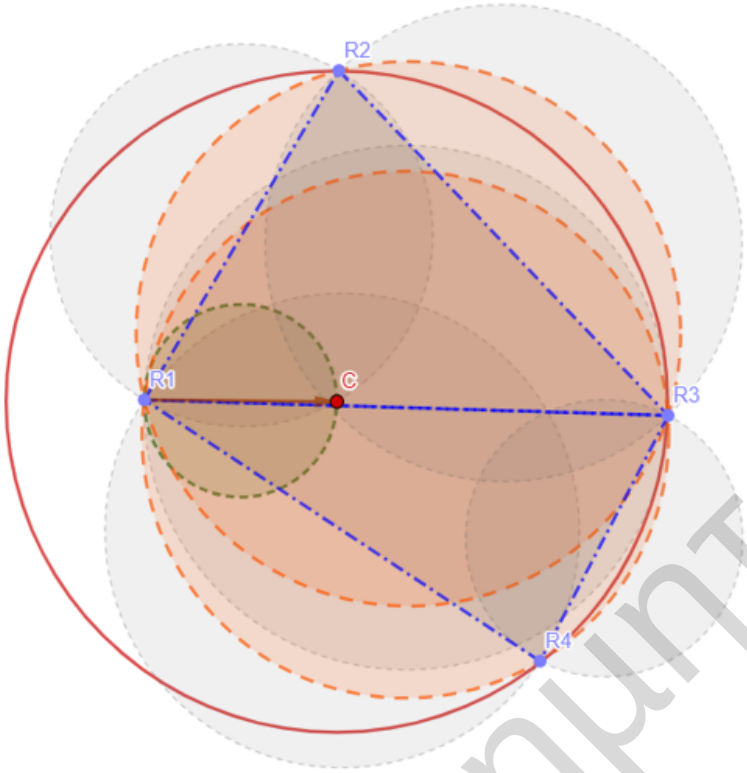
Πιο κάτω, στα παραρτήματα (βλ. Παράρτημα Β'), παραθέτονται διάφορες προσομοιώσεις αυτής της κατάστασης με διαφορετικές θέσεις των ρομπότ που ενδυναμώνουν αυτή την πρόταση.

Σε όλα τα πιο πάνω γραφήματα, παρατηρείται ότι ικανοποιούνται τα ακόλουθα θεωρήματα:

- Θεώρημα γραφήματος Gabriel (βλ. 2.5)
  - Σε κάθε κλειστό δίσκο που περικλείει 2 γειτονικά ρομπότ, δεν υπάρχει άλλο ρομπότ στο εσωτερικό του.
- Το γράφημα Gabriel είναι υπο-γράφος του τριγωνισμού Delaunay (βλ. 2.9.1)
  - Το θεώρημα του τριγωνισμού Delaunay μας λέει ότι ο δίσκος (εικόνα 3.8, πορτοκαλί χρώμα με διακεκομμένες γραμμές) που επικαλύπτει τις τρεις άκριες του τριγώνου που προκύπτει από τρία γειτονικά σημεία, δεν πρέπει να περιέχει άλλο ρομπότ στο εσωτερικό του. Παρατηρείται ότι ικανοποιείται το πιο πάνω θεώρημα.
- Το γράφημα Gabriel είναι και γράφημα Planar (βλ. 2.9.2)
  - Παρατηρείται ότι δεν υπάρχουν ακμές που να διασταυρώνονται μεταξύ τους (εικόνα 3.8, μπλε διακεκομμένες γραμμές).

Σε αυτό το σημείο, θα εξεταστεί κατά πόσο μπορούν να προκύψουν συγκρούσεις μεταξύ του ρομπότ r1 και των ρομπότ r2, r3, r4 (ρομπότ που βρίσκονται στην περιφέρεια του MEC του r1). Το κυριότερο που μας ενδιαφέρει είναι να εξεταστεί κατά πόσο οποιοδήποτε άλλο ρομπότ πέραν του r1, θα έχει τροχιά η οποία θα διαπεράσει από τον πράσινο κύκλο ο οποίος είναι και το safe zone του r1. Μόνο σε μια τέτοια περίπτωση μπορεί να υπάρξει σύγκρουση αφού οι τροχιές των δύο ρομπότ θα διασταυρωθούν.

Θα χρησιμοποιήσουμε σαν αναφορά το ακόλουθο γράφημα με την ακόλουθη κατάσταση (βλ. εικόνα 3.9).



Σχήμα 3.9: Απεικόνιση Σεναρίου 2 - Κατάστασης 1.8

.

Το ρομπότ r2, έχει σαν γείτονες το r1 και r3. Σε περίπτωση που δεν έχει άλλο γειτονικό ρομπότ, τότε το σημείο προορισμού του r2 (T(r2)) είναι δεδομένο ότι θα βρίσκεται στο πρώτο τεταρτημόριο του πιο πάνω MEC (MEC του r1, εικόνα 3.9). Σε περίπτωση που το r2 έχει και άλλους γείτονες, είναι δεδομένο ότι αυτοί θα βρίσκονται έξω από το MEC και συγκεκριμένα πιο πάνω από το r2 ως προς τον y άξονα. Αυτό οδηγεί στο συμπέρασμα ότι και το σημείο προορισμού του r2 θα βρίσκεται πιο πάνω. Και στις 2 περιπτώσεις, είναι δεδομένο ότι η τροχιά που θα ακολουθήσει το ρ2 για να φθάσει στο σημείο προορισμού του, δεν πρόκειται να διασταυρωθεί με την τροχιά που θα ακολουθήσει το r1.

Το ρομπότ r3, έχει σαν γείτονες το r1, r2 και r4. Σε περίπτωση που δεν έχει άλλο γειτονικό ρομπότ, τότε το σημείο προορισμού του (T(r3)) είναι προφανές ότι θα βρίσκεται δεξιότερα από το T(r1). Σε περίπτωση που το r3 έχει και άλλους γείτονες, είναι δεδομένο ότι αυτοί θα βρίσκονται έξω από το MEC του r1 και συγκεκριμένα δεξιότερα από το r3. Αυτό οδηγεί στο συμπέρασμα ότι και το σημείο προορισμού του r3 θα βρίσκεται δεξιότερα. Και στις 2 περιπτώσεις, είναι δεδομένο ότι η τροχιά που θα ακολουθήσει το r3 για να φθάσει στο σημείο προορισμού του, δεν πρόκειται να διασταυρωθεί με την τροχιά που θα ακολουθήσει

το r1.

Το ρομπότ r4, έχει σαν γείτονες το r1 και r3. Σε περίπτωση που δεν έχει άλλο γειτονικό ρομπότ, τότε το σημείο προορισμού του (T(r4)) είναι προφανές ότι θα βρίσκεται στο δεύτερο τεταρτημόριο του πιο πάνω MEC (MEC του r1). Σε περίπτωση που το r4 έχει και άλλους γείτονες, είναι δεδομένο ότι αυτοί θα βρίσκονται έξω από το MEC και συγκεκριμένα πιο κάτω από το r4 (το πιο πιθανό, μπορεί να βρίσκεται και δεξιότερα). Αυτό οδηγεί στο συμπέρασμα ότι και το σημείο προορισμού του r4 θα βρίσκεται πιο κάτω (ή δεξιότερα). Και στις 2 περιπτώσεις, είναι δεδομένο ότι η τροχιά που θα ακολουθήσει το r4 για να φθάσει στο σημείο προορισμού του, δεν πρόκειται να διασταυρωθεί με την τροχιά που θα ακολουθήσει το r1.

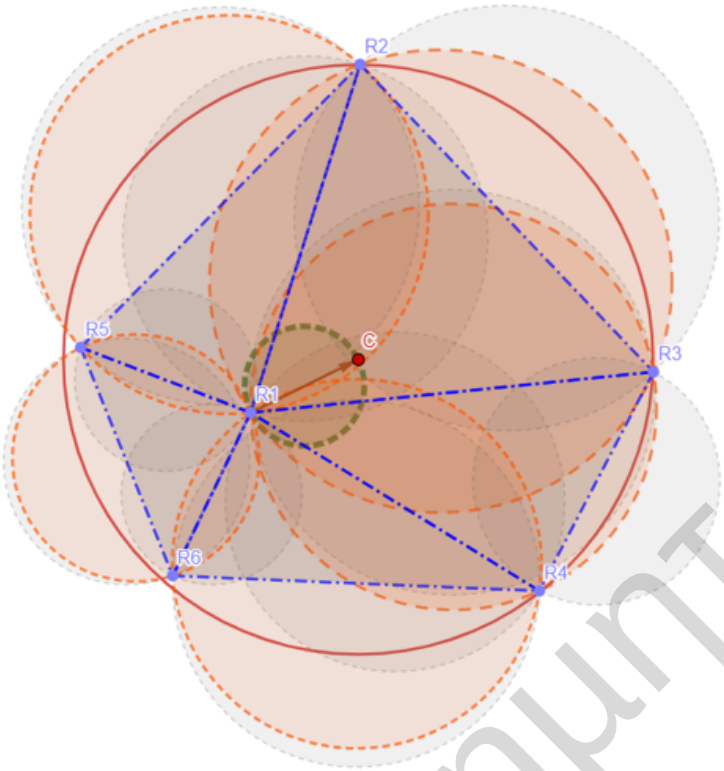
Η πιο πάνω ανάλυση καλύπτει πλήρως όλα τα σημεία που μπορεί να βρεθούν τα ρομπότ r2, r3 και r4 στην περιφέρεια του MEC καθώς και τα σημεία που μπορεί να βρεθεί το r1 μέσα στο MEC.

3.3.3 Περίπτωση 2 - Στην περιφέρεια του MEC υπάρχουν 3 ρομπότ που γειτνιάζουν με το r1 και στο εσωτερικό υπάρχουν και άλλα ρομπότ πέραν του r1

Σε αυτή την περίπτωση, το ρομπότ r1 βρίσκεται στο εσωτερικό του MEC, και στην περιφέρεια του υπάρχουν άλλα 3 ρομπότ (r2, r3, r4) τα οποία είναι γείτονες με το r1. Επιπλέον, στο εσωτερικό του MEC υπάρχει/υπάρχουν κι άλλο/άλλα ρομπότ πέραν του r1.

Οι τοποθεσίες που μπορεί να βρεθεί ένα ρομπότ στο εσωτερικό του MEC, είναι όπου δεν υπάρχει χρωματιστός κύκλος και το χρώμα της περιοχής είναι άσπρο (βλ. εικόνα 3.10). Αυτό ισχύει έτσι ώστε να ικανοποιούνται οι συνθήκες και τα θεωρήματα του γραφήματος Γαβριελ (2.5), του τριγωνισμού Delaunay (2.9.1) και του γραφήματος Planar 2.9.2.

Πιο κάτω στην εικόνα 3.10 παρουσιάζεται μια κατάσταση που μπορεί να προκύψει και είναι αποδεκτή, με 3 ρομπότ στην περιφέρεια του MEC του r1 και 3 στο εσωτερικό του.



Σχήμα 3.10: Απεικόνιση Σεναρίου 2 - Κατάστασης 2.1

.

- **Λίστα γειτνίασης:**
  - r1: r2 -> r3 -> r4 -> r5 -> r6
  - r2: r1 -> r3 -> r5
  - r3: r1 -> r2 -> r4
  - r4: r1 -> r3 -> r6
  - r5: r1 -> r2 -> r6
  - r6: r1 -> r4 -> r5

Το ρομπότ r2, έχει σαν γείτονες το r1, r3 και r5. Είτε έχει, είτε δεν έχει άλλο γειτονικό ρομπότ στο εξωτερικό του MEC, είναι δεδομένο ότι το σημείο προορισμού του r2 (T(r2)) θα βρίσκεται πιο πάνω από το σημείο προορισμού του r1 (T(r1)). Αυτό οδηγεί στο συμπέρασμα ότι η τροχιά που θα ακολουθήσει το r2 για να φθάσει στο σημείο προορισμού του, δεν πρόκειται να διασταυρωθεί με την τροχιά που θα ακολουθήσει το r1. Έπεται ότι δεν μπορεί να υπάρξει σύγκρουση μεταξύ r1 και r2.

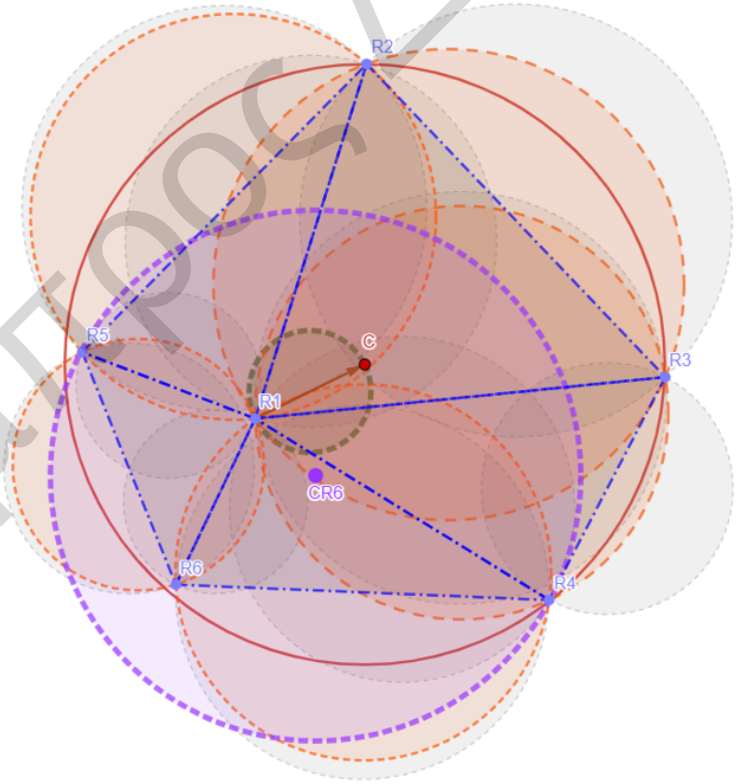
Το ρομπότ r3, έχει σαν γείτονες το r1, r2 και r4. Σε περίπτωση που δεν έχει άλλο γειτονικό ρομπότ, τότε το σημείο προορισμού του (T(r3)) είναι δεδομένο ότι θα βρίσκεται δεξιότερα από το σημείο προορισμού του r1 (T(r1)). Σε περίπτωση που το r3 έχει και άλλους γείτονες, είναι εύκολο να δειχθεί ότι αυτοί θα βρίσκονται έξω από το MEC και συγκεκριμένα δεξιότερα από το r3. Αυτό οδηγεί στο συμπέρασμα ότι και το σημείο προορισμού του r3 θα βρίσκεται δεξιότερα. Και στις 2 περιπτώσεις, συμπεραίνεται ότι η τροχιά που θα ακολουθήσει το r3 για να φθάσει στο σημείο προορισμού του, δεν πρόκειται να διασταυρωθεί με την τροχιά που θα ακολουθήσει το r1. Έπεται ότι δεν μπορεί να υπάρξει σύγκρουση μεταξύ r1 και r3.

Το ρομπότ r4, έχει σαν γείτονες το r1, r3 και r6. Είτε έχει, είτε δεν έχει άλλο γειτονικό ρομπότ στο εξωτερικό του MEC, είναι δεδομένο ότι το σημείο προορισμού του r4 (T(r4)) θα βρίσκεται δεξιότερα από το σημείο προορισμού του r1 (T(r1)). Αυτό οδηγεί στο συμπέρασμα ότι η τροχιά που θα ακολουθήσει το r4 για να φθάσει στο σημείο προορισμού του, δεν πρόκειται να διασταυρωθεί με την τροχιά που θα ακολουθήσει το r1. Έπεται ότι δεν μπορεί να υπάρξει σύγκρουση μεταξύ r1 και r4.

Το ρομπότ r5, έχει σαν γείτονες το r1, r2 και r6. Είτε έχει, είτε δεν έχει άλλο γειτονικό ρομπότ στο εξωτερικό του MEC, είναι δεδομένο ότι το σημείο προορισμού του r5 (T(r5)) θα βρίσκεται αριστερότερα από το r1. Αφού το r1 θα κινηθεί προς τα δεξιά και πάνω για να φθάσει στο σημείο προορισμού του, και το σημείο προορισμού του r5 (θα κινηθεί προς τα κει) είναι αριστερότερα από το r1, συμπεραίνεται ότι οι τροχιές που θα ακολουθήσουν τα 2 ρομπότ (r1, r5) δεν πρόκειται να διασταυρωθούν. Έπεται ότι δεν μπορεί να υπάρξει σύγκρουση μεταξύ r1 και r5.

Το ρομπότ r6, έχει σαν γείτονες το r1, r4 και r5. Πιο κάτω, στην εικόνα 3.11, φαίνεται με λιλά χρώμα το MEC του r6 που θα υπολογιστεί από τον αλγόριθμο σε περίπτωση που δεν έχει άλλους γείτονες έξω από το MEC του r1. Σε περίπτωση που έχει άλλους γείτονες, είναι ξεκάθαρο ότι το MEC του r6 θα μετακινηθεί προς τα αριστερότερα, όπως και το σημείο προορισμού του r6 (T(r6)) το οποίο συμβολίζεται με CR6 στην εικόνα 3.11. Σε αυτή την περίπτωση, είναι δεδομένο ότι η τροχιά που θα ακολουθήσει το r6, δεν θα διασταυρωθεί με αυτή του r1. Στην περίπτωση που το r6 δεν θα έχει άλλους γείτονες πέραν του r1, r4 και r5, το σημείο προορισμού του r6 που θα υπολογιστεί από τον αλγόριθμο, είναι αυτό που απεικονίζεται (CR6). Παρατηρείτε ότι είναι εχτός του safe zone του r1 (πράσινος κύκλος), και έτσι συμπεραίνεται ότι οι τροχιές των δύο ρομπότ δεν θα διασταυρωθούν.

Μια απορία που πιθανόν να δημιουργηθεί από τον αναγνώστη είναι η εξής: σε περίπτωση που μετακινηθούν οι γείτονες του r6 έτσι ώστε να μετακινήσουν το MEC του προς τα πάνω, τότε δεν θα μετακινηθεί και το T(R6) και να εμπίπτει μέσα στο safe zone του r1· Η απάντηση είναι η εξής: για να μετακινηθεί το MEC του r6 προς τα πάνω, θα πρέπει να μετακινηθεί ένα εκ των δύο ρομπότ που βρίσκονται στην περιφέρεια του (ή και τα δύο) τα οποία είναι το r4 και r5. Μια τέτοια μετακίνηση όμως, θα άλλαζε ριζικά το γράφημα 3.11, αφού θα άλλαξε και τους γείτονες του κάθε ρομπότ. Ως εκ τούτου θα άλλαζε και το MEC του κάθε ρομπότ και κατ' επέκταση το σημείο προορισμού του. Για παράδειγμα, έστω ότι το ρομπότ r5 μετακινηθεί ελάχιστα προς τα πάνω, τότε, μέσα στο δίσκο που περικλείει τα ρομπότ r5 και r6 (και τα καθιστά γείτονες – συνθήκη γραφήματος Gabriel), θα βρεθεί το r1. Αυτό οδηγεί στο γεγονός ότι πλέον το r5 και r6 δεν μπορούν να είναι γείτονες μεταξύ τους. Έτσι, το MEC του r6, θα αλλάξει ριζικά, αφού το r5 ήταν ένα ρομπότ στην περιφέρεια του MEC του r6 και έτσι το επηρεάζει άμεσα.



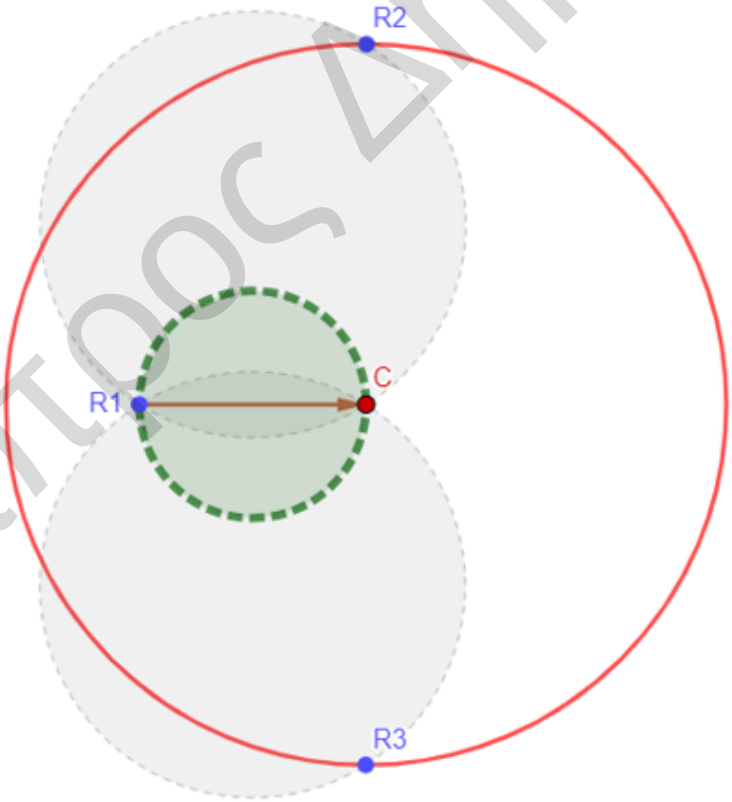
Σχήμα 3.11: Απεικόνιση Σεναρίου 2 - Κατάστασης 2.2

.

3.3.4 Περίπτωση 3 - Στην περιφέρεια του MEC βρίσκονται 2 ρομπότ και στο εσωτερικό του υπάρχουν ρομπότ

Σε αυτή την περίπτωση, το ρομπότ r1 βρίσκεται στο εσωτερικό του MEC, και στην περιφέρεια του MEC υπάρχουν άλλα 2 ρομπότ (r2, r3) τα οποία είναι γείτονες με το r1. Επιπλέον, στο εσωτερικό του MEC υπάρχει/υπάρχουν και άλλο/άλλα ρομπότ.

Σε αυτή την κατάσταση, η οποία περιέχει 2 ρομπότ (έστω r2, r3) στην περιφέρεια του MEC του r1, είναι προφανές ότι τα 2 αυτά ρομπότ θα είναι απέναντι το ένα από το άλλο, δηλαδή, η ευκλείδεια απόσταση μεταξύ των 2 ρομπότ θα ισούται με τη διάμετρο του MEC όπως φαίνεται και στην ακόλουθη εικόνα 3.12. Αυτό προκύπτει από τον αλγόριθμο, έτσι ώστε να υπολογιστεί το minimal enclosing circle (MEC) του r1. Επίσης, στην εικόνα 3.12, στο εσωτερικό του MEC, με άσπρο χρώμα παρουσιάζονται όλα τα πιθανά σημεία που μπορεί αν υπάρχει γειτονικό ρομπότ με το r1.

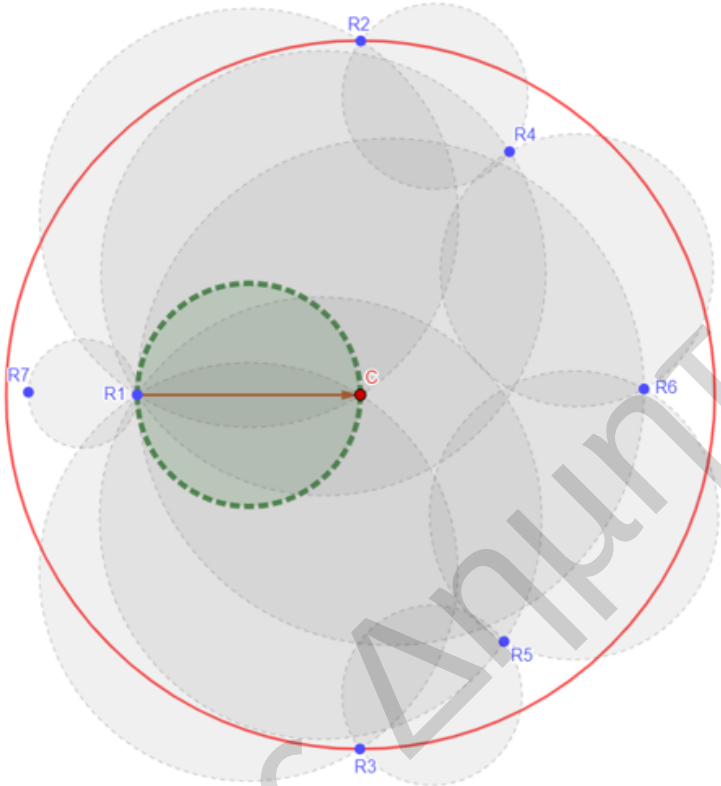


Σχήμα 3.12: Απεικόνιση Σεναρίου 2 - Κατάστασης 3.1

.

Σε αυτό το σημείο, θα εξεταστεί αν μπορεί να υπάρξουν συγκρούσεις σε μια διάταξη που

το MEC του r1 περιέχει 2 ρομπότ στην περιφέρεια του και 5 ρομπότ στο εσωτερικό του (συμπεριλαμβανομένου και του ρ1). Αποικονίζεται η συγκεκριμένη διάταξη στην εικόνα 3.13.



Σχήμα 3.13: Απεικόνιση Σεναρίου 2 - Κατάστασης 3.2

.

- **Λίστα γειτνίασης:**
  - r1: r2 -> r3 -> r4 -> r5 -> r6 -> r7
  - r2: r1 -> r4
  - r3: r1 -> r5
  - r4: r1 -> r2 -> r6
  - r5: r1 -> r3 -> r6
  - r6: r1 -> r4 -> r5
  - r7: r1

Το ρομπότ r2, το οποίο έχει γείτονες το r1 και r4, αν δεν έχει άλλους γείτονες στο εξωτερικό

του MEC, τότε το σημείο προορισμού του θα βρίσκεται μεταξύ του 1ου και 4ου τεταρτημόριου του MEC του r1 και πιο ψηλά από το σημείο προορισμού του r1. Αν το ρομπότ r2 έχει και άλλους γείτονες στο εξωτερικό του MEC, τότε το σημείο προορισμού του θα βρίσκεται σε ακόμη πιο ψηλό σημείο ως προς τον y άξονα. Βάση αυτών, είναι προφανές ότι η τροχιά του r2 δεν θα διασταυρωθεί με την τροχιά του r1. Έπεται ότι δεν θα υπάρξει σύγκρουση μεταξύ του r1 και r2.

Αντίστοιχα με το r2, και το ρομπότ r3, το οποίο έχει γείτονες το r1 και r5, αν δεν έχει άλλους γείτονες στο εξωτερικό του MEC, τότε το σημείο προορισμού του θα βρίσκεται στο 3ο τεταρτημόριο του MEC του r1 και πιο χαμηλά από το σημείο προορισμού του ρ1. Αν έχει και άλλους γείτονες στο εξωτερικό του MEC, τότε το σημείο προορισμού του r3 θα βρίσκεται σε ακόμη πιο χαμηλό σημείο ως προς τον y άξονα. Βάση αυτών, είναι προφανές ότι η τροχιά του r3 δεν θα διασταυρωθεί με την τροχιά του r1. Έπεται ότι δεν θα υπάρξει σύγκρουση μεταξύ του r1 και r3.

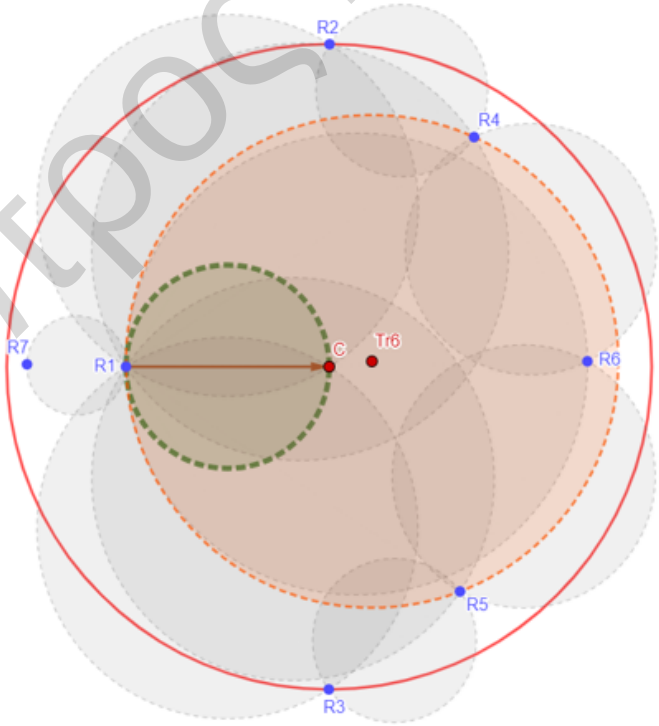
Τα ρομπότ r4, το οποίο έχει γείτονες το r1, r2 και r6, αν δεν έχει άλλους γείτονες στο εξωτερικό του MEC, τότε το σημείο προορισμού του θα βρίσκεται στο 1ο τεταρτημόριο του MEC του r1 και πιο ψηλά από το σημείο προορισμού του r1. Αν έχει και άλλους γείτονες στο εξωτερικό του MEC, τότε το σημείο προορισμού του r4 θα βρίσκεται σε ακόμη πιο ψηλό σημείο ως προς τον y άξονα ή ακόμη δεξιότερα από το σημείο προορισμού του r1 ως προς τον x άξονα. Βάση αυτών, είναι προφανές ότι η τροχιά του r4 δεν θα διασταυρωθεί με την τροχιά του r1. Έπεται ότι δεν θα υπάρξει σύγκρουση μεταξύ του r1 και r4.

Το ρομπότ r5, το οποίο έχει γείτονες το r1, r3 και r6, αν δεν έχει άλλους γείτονες στο εξωτερικό του MEC, τότε το σημείο προορισμού του θα βρίσκεται μεταξύ του 2ου και 3ου τεταρτημόριου του MEC του r1 και πιο χαμηλά από το σημείο προορισμού του r1. Αν έχει και άλλους γείτονες στο εξωτερικό του MEC, τότε το σημείο προορισμού του r5 θα βρίσκεται σε ακόμη πιο χαμηλό σημείο ως προς τον y άξονα. Βάση αυτών, είναι προφανές ότι η τροχιά του r5 δεν θα διασταυρωθεί με την τροχιά του r1. Έπεται ότι δεν θα υπάρξει σύγκρουση μεταξύ του r1 και r5.

Το ρομπότ r7, το οποίο έχει γείτονα το r1, αν δεν έχει άλλους γείτονες στο εξωτερικό του MEC, τότε το MEC του r7 θα έχει διάμετρο την ακμή μεταξύ των σημείων r1 και r7. Ως εκ τούτου, το σημείο προορισμού του r7 θα βρίσκεται μεταξύ του r1 και r7 (στη μισή απόσταση). Αν το r7 έχει και άλλους γείτονες στο εξωτερικό του MEC, τότε το σημείο προορισμού του

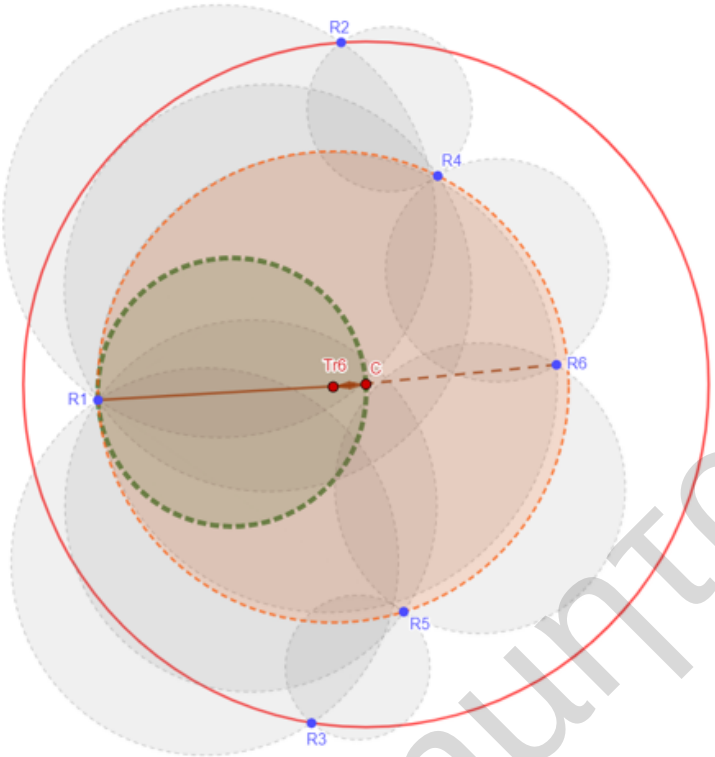
r7 θα βρίσκεται σε ακόμη πιο μακρινή απόσταση από το r1. Βάση αυτών, είναι προφανές ότι η τροχιά του r7 δεν θα διασταυρωθεί με την τροχιά του r1. Έπεται ότι δεν θα υπάρξει σύγκρουση μεταξύ του r1 και r7.

Το ρομπότ r6, το οποίο έχει γείτονες το r1, r4 και r5, αν δεν έχει άλλους γείτονες στο εξωτερικό του MEC, τότε το σημείο προορισμού του απεικονίζεται στο πιο κάτω γράφημα στην εικόνα 3.14 (Tr6) καθώς και το MEC του με πορτοκαλί χρώμα. Αν έχει και άλλους γείτονες στο εξωτερικό του MEC, τότε το σημείο προορισμού του r6 θα βρίσκεται σε ακόμη πιο μακρινή απόσταση από το safe zone (πράσινος κύκλος) του r1. Βάση αυτών, είναι προφανές ότι η τροχιά του r6 δεν θα διασταυρωθεί με την τροχιά του r1. Έπεται ότι δεν θα υπάρξει σύγκρουση μεταξύ του r1 και r6. Παρόλα αυτά, από όλα τα ρομπότ στην προκειμένη κατάσταση, το ρομπότ r6 είναι το πιο πιθανόν να βρεθεί σε μια κατάσταση όπου μπορεί να προκαλέσει σύγκρουση με το r1 για αυτό θα γίνει περισσότερη διερεύνηση στο σε ποιες καταστάσεις μπορεί να βρεθεί το συγκεκριμένο ρομπότ και με τι σημείο προορισμού. Το ρομπότ r7 δεν επηρεάζει αυτή την διερεύνηση, για αυτό και θα αφαιρεθεί για σκοπούς απλότητας.



Σχήμα 3.14: Απεικόνιση Σεναρίου 2 - Κατάστασης 3.3

.

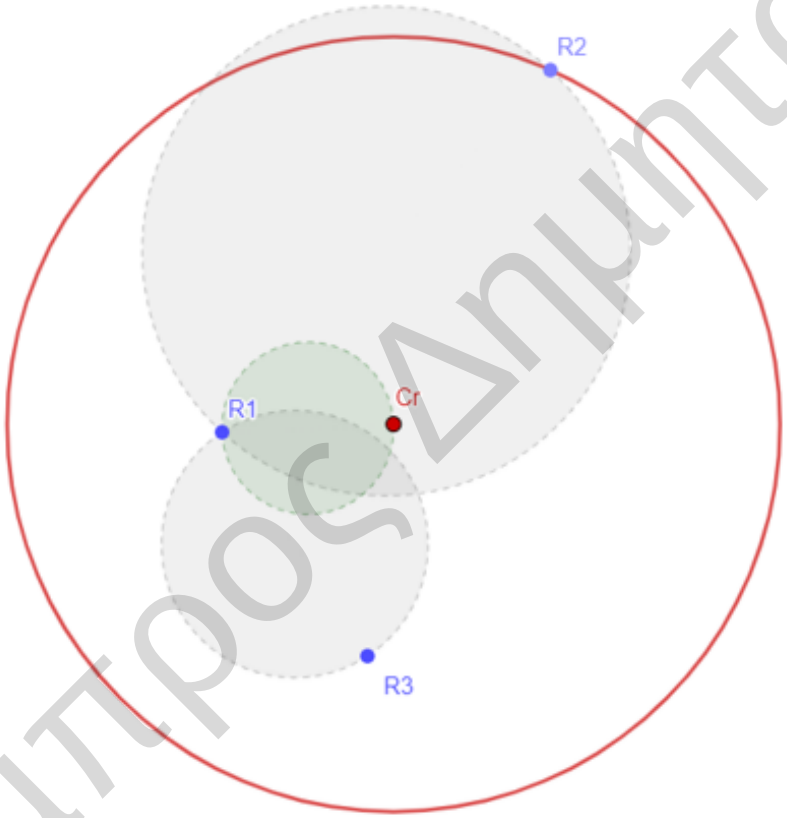


Σχήμα 3.15: Απεικόνιση Σεναρίου 2 - Κατάστασης 3.4

.

Στο πιο πάνω γράφημα, στην εικόνα 3.15, παρατηρείται ότι το σημείο προορισμού του r6 βρίσκεται μέσα στο save zone του r1 (πράσινος κύκλος) και πιο συγκεκριμένα βρίσκεται πάνω στην τροχιά που θα ακολουθήσει το r1. Αυτό υποδεικνύει ότι οι τροχιές των ρομπότ r1 και r6 έχουν κοινό σημείο (διασταυρώνονται) και αυτό προκαλεί μεγάλη πιθανότητα σύγκρουσης.

• **Σημ.:** σε περίπτωση που το r1 βρίσκεται στο εσωτερικό του MEC, τότε είναι δεδομένο ότι στην περιφέρεια του MEC θα υπάρχουν τουλάχιστον άλλα 2 ρομπότ (μέχρι 3) έτσι ώστε να διαμορφωθεί ο MEC. Αλλιώς το r1 θα βρισκόταν στην περιφέρεια του MEC. Πιο κάτω στην εικόνα 3.16 παρουσιάζεται ένα τέτοιο σενάριο το οποίο δεν είναι αποδεκτό. Το r2 βρίσκεται στην περιφέρεια του MEC, με το r1 να έχει σαν γείτονες το r2 και r3. Σε μια τέτοια περίπτωση, συμπεραίνεται εύκολα ότι ο κόκκινος κύκλος (MEC του r1), δεν είναι στην πραγματικότητα ο minimal enclosing circle, αφού μπορεί να υπολογιστεί και κύκλος με μικρότερη ακτίνα, δηλαδή, έχοντας και το r3 στην περιφέρεια του.



Σχήμα 3.16: Απεικόνιση Σεναρίου 2 - Κατάστασης 3.5

.

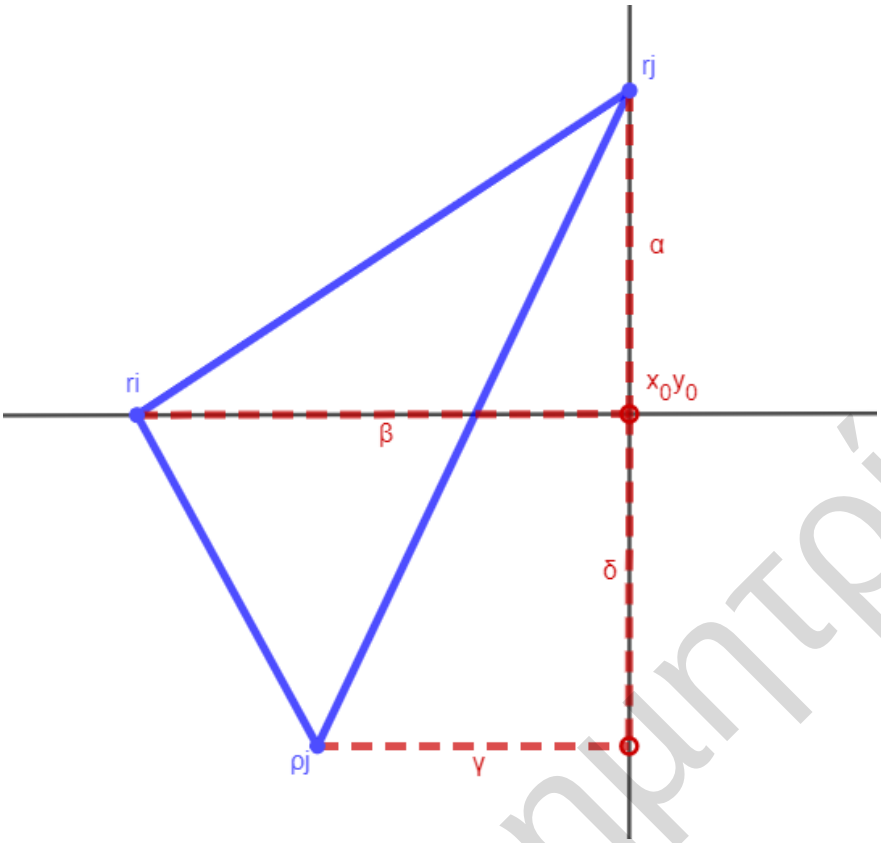
Κεφάλαιο 4

Μαθηματική Απόδειξη

4.1 Εισαγωγή

Σε αυτό το κεφάλαιο, βάση των συμπερασμάτων από τις προσομοιώσεις που παρουσιάστηκαν στο προηγούμενο κεφάλαιο, θα γίνει μια προσπάθεια να αποδειχθεί με μαθηματικό τρόπο, ότι, ο αλγόριθμος συγκέντρωσης (gathering) Go-To-The-Gabriel-Center (GTGC) είναι collision-less.

Χρησιμοποιήθηκε σαν αναφορά το γράφημα στην πιο κάτω εικόνα 4.1. Αρχικά, έχουμε σαν δεδομένο ότι το trajectory του ri θα έχει κατεύθυνση προς το σημείο x_0y_0. Για να υπάρχει σύγκρουση μεταξύ του ri και του rj, τότε το trajectory του rj θα έπρεπε να διασταυρώνει αυτό του ri. Αυτό σημαίνει ότι το rj, θα έπρεπε να είχε ένα gabriel neighbor για να το αναγκάσει να κινηθεί προς αυτή την κατεύθυνση (το ίδιο ισχύει και μεταξύ του ri και ρj). Έστω ότι, αυτός ο gabriel neighbor του rj είναι το ρj.



Σχήμα 4.1: Εικόνα αναφοράς για μαθηματική απόδειξη.

Αρχικά, έγιναν οι ακόλουθες υποθέσεις:

1. Αν το ri και rj είναι gabriel neighbors, τότε το ρj δεν μπορεί να βρίσκεται μέσα στον δίσκο με διάμετρο rirj (από gabriel graph κριτήριο 2.5).

2. Αν το ri και ρj είναι gabriel neighbors, τότε το rj δεν μπορεί να βρίσκεται μέσα στον δίσκο με διάμετρο riρj (από gabriel graph κριτήριο 2.5).

3. Αν το rj και ρj είναι gabriel neighbors, τότε το ri δεν μπορεί να βρίσκεται μέσα στον δίσκο με διάμετρο rjρj (από gabriel graph κριτήριο 2.5).

Χρησιμοποιώντας τις πιο πάνω υποθέσεις, και υπολογίζοντας τις εξισώσεις των τριών κύκλων με διαμέτρους rirj, riρj και rjρj ως προς τις μεταβλητές α, β, γ, δ, έγινε μια προσπάθεια για υπολογισμό κάποιων ανισοτήτων μεταξύ των μεταβλητών α, β, γ, δ. Βάση των πιο πάνω υποθέσεων, υπολογίστηκαν οι πιο κάτω ανισότητες:

$$d[\rho j, k(rirj)] > Rrirj$$

$$d[rj, k(ri\rho j)] > Rri\rho j$$

$$d[ri, k(rj\rho j)] > Rrj\rho j$$

Χρησιμοποιώντας το σύστημα ανισοτήτων, έγινε μια προσπάθεια να αποδειχθεί ότι αυτό το σύστημα δεν έχει λύση, που αυτό θα οδηγούσε σε αντίφαση των πιο πάνω υποθέσεων.

4.2 Υπολογιζόμενες εξισώσεις

4.2.1 Εξισώσεις κύκλων

Για να υπολογιστούν οι εξισώσεις των τριών κύκλων, πρώτα έπρεπε να υπολογιστούν τα κέντρα και οι ακτίνες αυτών των κύκλων και ακολούθως να χρησιμοποιηθεί η εξίσωση του κύκλου:

$$(x-h)^2 + (y-k)^2 = r^2$$

όπου (h,k) το κέντρο του κύκλου και r η ακτίνα του κύκλου.

Οι συντεταγμένες των σημείων ri, rj και ρj, βάση του γραφήματος 4.1 είναι οι ακόλουθες:

$$\begin{aligned} r_i &= (-\beta, 0) \\ r_j &= (0, a) \\ \rho_j &= (-\gamma, -\delta) \end{aligned} \tag{4.1}$$

όπου, λόγω του viewing range = 1, ισχύει ότι:

$$0 < a \leq 1, 0 < \beta \leq 1, 0 < \gamma \leq 1, 0 < \delta \leq 1$$

Το κέντρο του κάθε κύκλου έχει τις ακόλουθες συντεταγμένες:

$$C = (\frac{x_1 + x_2}{2}, \frac{y_1 + y_2}{2})$$

Βάση της πιο πάνω εξίσωσης, προκύπτουν τα κέντρα των 3ων κύκλων:

$$
\begin{aligned}
C_1 &= C_{ririj} = (-\frac{\beta}{2}, \frac{a}{2}) \\
C_2 &= C_{riρj} = (-\frac{(\beta+\gamma)}{2}, -\frac{\delta}{2}) \\
C_3 &= C_{rjjρj} = (-\frac{\gamma}{2}, \frac{a-\delta}{2})
\end{aligned}
\tag{4.2}
$$

Για τον υπολογισμό των ακτινών των 3ων κύκλων, θα χρησιμοποιηθεί η εξίσωση που υπολογίζει την ευκλείδεια απόσταση μεταξύ 2 σημείων:

$$
d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{4.3}
$$

Υπολογισμός ακτίνας r1 με διάμετρο τα σημεία ririj. Υπολογίζεται η ευκλείδεια απόσταση μεταξύ ri και C_1:

$$
\begin{aligned}
r_1 = R_{ririj} = d(r_i, C_1) &= \sqrt{[-\beta - (-\frac{\beta}{2})]^2 + (0 - \frac{a}{2})^2} \\
&= \sqrt{(-\beta + \frac{\beta}{2})^2 + (-\frac{a}{2})^2} \\
&= \sqrt{(-\frac{\beta}{2})^2 + (-\frac{a}{2})^2} \\
&= \sqrt{\frac{\beta^2}{4} + \frac{a^2}{4}} \\
&= \frac{\sqrt{a^2 + \beta^2}}{2}
\end{aligned}
\tag{4.4}
$$

Υπολογισμός ακτίνας r2 με διάμετρο τα σημεία riρj. Υπολογίζεται η ευκλείδεια απόσταση μεταξύ ri και C_2:

$$
\begin{aligned}
r_2 = R_{ri\rho j} = d(r_i, C_2) &= \sqrt{[-(\frac{\beta+\gamma}{2}) + \beta]^2 + (-\frac{\delta}{2} - 0)^2} \\
&= \sqrt{\frac{(\beta+\gamma)^2}{4} - (\beta+\gamma)\beta + \beta^2 + \frac{\delta^2}{4}} \\
&= \sqrt{\frac{\beta^2 + 2\beta\gamma + \gamma^2}{4} - \beta^2 - \beta\gamma + \beta^2 + \frac{\delta^2}{4}} \\
&= \sqrt{\frac{\beta^2 + 2\beta\gamma + \gamma^2 - 4\beta^2 - 4\beta\gamma + 4\beta^2 + \delta^2}{4}} \\
&= \sqrt{\frac{\beta^2 + \gamma^2 - 2\beta\gamma + \delta^2}{4}} \\
&= \frac{\sqrt{(\beta-\gamma)^2 + \delta^2}}{2}
\end{aligned}
\tag{4.5}
$$

Υπολογισμός ακτίνας r3 με διάμετρο τα σημεία rjρj. Υπολογίζεται η ευκλείδεια απόσταση μεταξύ rj και C_3:

$$
\begin{aligned}
r_3 = R_{rj\rho j} = d(r_j, C_3) &= \sqrt{(-\frac{\gamma}{2} - 0)^2 + (\frac{a-\delta}{2} - a)^2} \\
&= \sqrt{\frac{\gamma^2}{4} + \frac{(a-\delta)^2}{4} - (a-\delta)a + a^2} \\
&= \sqrt{\frac{\gamma^2}{4} + \frac{a^2 - 2a\delta + \delta^2}{4} - a^2 + a\delta + a^2} \\
&= \sqrt{\frac{\gamma^2 + a^2 - 2a\delta + \delta^2 + 4a\delta}{4}} \\
&= \frac{\sqrt{a^2 + 2a\delta + \delta^2 + \gamma^2}}{2} \\
&= \frac{\sqrt{(a+\delta)^2 + \gamma^2}}{2}
\end{aligned}
\tag{4.6}
$$

Με τη χρήση των σημείων του κέντρου και της ακτίνας του κάθε κύκλου, μπορούν να υπολογιστούν οι εξισώσεις των 3ων κύκλων ως ακολούθως:

Εξίσωση κύκλου 1 = Κύκλος rirj =

$$
(x + \frac{\beta}{2})^2 + (y - \frac{a}{2})^2 = \frac{a^2 + \beta^2}{4} \tag{4.7}
$$

Εξίσωση κύκλου 2 = Κύκλος riρj =

$$(x+\frac{\beta+\gamma}{2})^2+(y+\frac{\delta}{2})^2=\frac{(\beta-\gamma)^2+\delta^2}{4} \tag{4.8}$$

Εξίσωση κύκλου 3 = Κύκλος rjρj =

$$(x+\frac{\gamma}{2})^2+(y-\frac{a-\delta}{2})^2=\frac{(a+\delta)^2+\gamma^2}{4} \tag{4.9}$$

4.2.2 Εξισώσεις Ανισοτήτων βάση υποθέσεων

1. Από την 1η υπόθεση, θα υπολογιστούν οι εξισώσεις ανισοτήτων μεταξύ των μεταβλητών α, β, γ, δ. Η υπόθεση που έγινε είναι ότι το ρj, βρίσκεται έξω από τον κύκλο rirj.

$$\begin{aligned}
d[\rho_j, k(rirj)] &> R_{rirj}\\
d[(-\gamma,-\delta),(-\frac{\beta}{2},\frac{a}{2})] &> \frac{\sqrt{a^2+\beta^2}}{2}\\
\sqrt{(-\frac{\beta}{2}+\gamma)^2+(\frac{a}{2}+\delta)^2} &> \frac{\sqrt{a^2+\beta^2}}{2}\\
(\sqrt{(-\frac{\beta}{2}+\gamma)^2+(\frac{a}{2}+\delta)^2})^2 &> (\frac{\sqrt{a^2+\beta^2}}{2})^2\\
(-\frac{\beta}{2}+\gamma)^2+(\frac{a}{2}+\delta)^2 &> \frac{a^2+\beta^2}{4}\\
\frac{\beta^2}{4}-\beta\gamma+\gamma^2+\frac{a^2}{4}+a\delta+\delta^2 &> \frac{a^2+\beta^2}{4}\\
\frac{\beta^2}{4}-\frac{4\beta\gamma}{4}+\frac{4\gamma^2}{4}+\frac{a^2}{4}+\frac{4a\delta}{4}+\frac{4\delta^2}{4} &> \frac{a^2+\beta^2}{4}\\
\beta^2-4\beta\gamma+4\gamma^2+a^2+4a\delta+4\delta^2 &> a^2+\beta^2\\
4\gamma^2+4\delta^2-4\beta\gamma+4a\delta &> 0\\
\gamma^2+\delta^2-\beta\gamma+a\delta &> 0
\end{aligned} \tag{4.10}$$

2. Από την 2η υπόθεση, θα υπολογιστούν οι εξισώσεις ανισοτήτων μεταξύ των μεταβλητών α, β, γ, δ. Η υπόθεση που έγινε είναι ότι το rj, βρίσκεται έξω από τον κύκλο riρj.

$$
\begin{aligned}
d[rj, k(ri\rho j)] &> R_{ri\rho j} \\
d[(0,a),(-\frac{(\beta+\gamma)}{2},-\frac{\delta}{2})] &> \frac{\sqrt{(\beta-\gamma)^2+\delta^2}}{2} \\
\sqrt{[-\frac{(\beta+\gamma)}{2}-0]^2+(-\frac{\delta}{2}-a)^2} &> \frac{\sqrt{(\beta-\gamma)^2+\delta^2}}{2} \\
(\sqrt{[-\frac{(\beta+\gamma)}{2}-0]^2+(-\frac{\delta}{2}-a)^2})^2 &> (\frac{\sqrt{(\beta-\gamma)^2+\delta^2}}{2})^2 \\
[-\frac{(\beta+\gamma)}{2}-0]^2+(-\frac{\delta}{2}-a)^2 &> \frac{\sqrt{(\beta-\gamma)^2+\delta^2}}{4} \\
(-\frac{\beta+\gamma}{2})^2+\frac{\delta^2}{4}+a\delta+a^2 &> \frac{\beta^2-2\beta\gamma+\gamma^2+\delta^2}{4} \\
\frac{\beta^2+2\beta\gamma+\gamma^2}{4}+\frac{\delta^2}{4}+\frac{4a\delta}{4}+\frac{4a^2}{4} &> \frac{\beta^2-2\beta\gamma+\gamma^2+\delta^2}{4} \\
4\beta\gamma+4a\delta+4a^2 &> 0 \\
a^2+a\delta+\beta\gamma &> 0
\end{aligned}
\tag{4.11}
$$

3. Από την 3η υπόθεση, θα υπολογιστούν οι εξισώσεις ανισοτήτων μεταξύ των μεταβλητών α, β, γ, δ. Η υπόθεση που έγινε είναι ότι το ri, βρίσκεται έξω από τον κύκλο rjρj.

$$
\begin{aligned}
d[ri, k(rj\rho j)] &> R_{rj\rho j} \\
d[(-\beta, 0), (-\frac{\gamma}{2}, \frac{a-\delta}{2})] &> \frac{\sqrt{(a+\delta)^2+\gamma^2}}{2} \\
\sqrt{(-\frac{\gamma}{2}+\beta)^2+(\frac{a-\delta}{2})^2} &> \frac{\sqrt{(a+\delta)^2+\gamma^2}}{2} \\
(\sqrt{(-\frac{\gamma}{2}+\beta)^2+(\frac{a-\delta}{2})^2})^2 &> (\frac{\sqrt{(a+\delta)^2+\gamma^2}}{2})^2 \\
(-\frac{\gamma}{2}+\beta)^2+(\frac{a-\delta}{2})^2 &> \frac{(a+\delta)^2+\gamma^2}{4} \\
4(-\frac{\gamma}{2}+\beta)^2+4(\frac{a-\delta}{2})^2 &> (a+\delta)^2+\gamma^2 \\
4(\frac{\gamma^2}{4}-\beta\gamma+\beta^2)+\frac{4(a-\delta)^2}{4} &> (a+\delta)^2+\gamma^2 \\
4(\frac{\gamma^2}{4}-\beta\gamma+\beta^2)+a^2-2a\delta+\delta^2 &> a^2+2a\delta+\delta^2+\gamma^2 \\
\gamma^2-4\beta\gamma+4\beta^2+a^2-2a\delta+\delta^2 &> a^2+2a\delta+\delta^2+\gamma^2 \\
4\beta^2-4\beta\gamma-2a\delta &> 2a\delta \\
4\beta^2-4\beta\gamma-4a\delta &> 0 \\
\beta^2-\beta\gamma-a\delta &> 0
\end{aligned}
\tag{4.12}
$$

Επίσης, λόγω του viewing range που είναι 1, συμπεραίνεται ότι η διάμετρος του κάθε κύκλου είναι ≤ 1. Αυτό σημαίνει ότι η ακτίνα του κάθε κύκλου είναι $\leq \frac{1}{2}$.

Βάση αυτού, προκύπτουν οι ακόλουθες ανισότητες:

$$
\begin{aligned}
r_1 = R_{ritj} &\leq \frac{1}{2} \\
\frac{\sqrt{a^2+\beta^2}}{2} &\leq \frac{1}{2} \\
a^2+\beta^2 &\leq 1
\end{aligned}
\tag{4.13}
$$

$$
\begin{aligned}
r_2 = R_{ri\rho j} &\leq \frac{1}{2} \\
\frac{\sqrt{(\beta-\gamma)^2+\delta^2}}{2} &\leq \frac{1}{2} \\
(\beta-\gamma)^2+\delta^2 &\leq 1 \\
\beta^2-2\beta\gamma+\gamma^2+\delta^2 &\leq 1
\end{aligned}
\tag{4.14}
$$

$$
\begin{aligned}
r_3 = R_{rj\rho j} &\leq \frac{1}{2} \\
\frac{\sqrt{(a+\delta)^2+\gamma^2}}{2} &\leq \frac{1}{2} \\
(a+\delta)^2+\gamma^2 &\leq 1 \\
a^2+2a\delta+\delta^2+\gamma^2 &\leq 1
\end{aligned}
\tag{4.15}
$$

- **Σύνοψη ανισοτήτων:**
  - $\gamma^2+\delta^2-\beta\gamma+a\delta>0$
  - $a^2+a\delta+\beta\gamma>0$
  - $\beta^2-\beta\gamma-a\delta>0$
  - $a^2+\beta^2\leq 1$
  - $\beta^2-2\beta\gamma+\gamma^2+\delta^2\leq 1$
  - $a^2+2a\delta+\delta^2+\gamma^2\leq 1$
  - $0<a\leq 1$
  - $0<\beta\leq 1$
  - $0<\gamma\leq 1$
  - $0<\delta\leq 1$

4.3 Λύση συστήματος ανισοτήτων

Έχοντας το σύστημα που αποτελείται από τις ανισότητες που υπολογίστηκαν στο προηγούμενο section, γράφτηκε ένα πρόγραμμα στη γλώσσα προγραμματισμού python έτσι ώστε να διαπιστωθεί αν το σύστημα έχει λύση. Αν το σύστημα δεν είχε λύση, τότε οι αρχικές υποθέσεις δεν θα ίσχυαν και θα οδηγούμασταν σε αντίφαση.

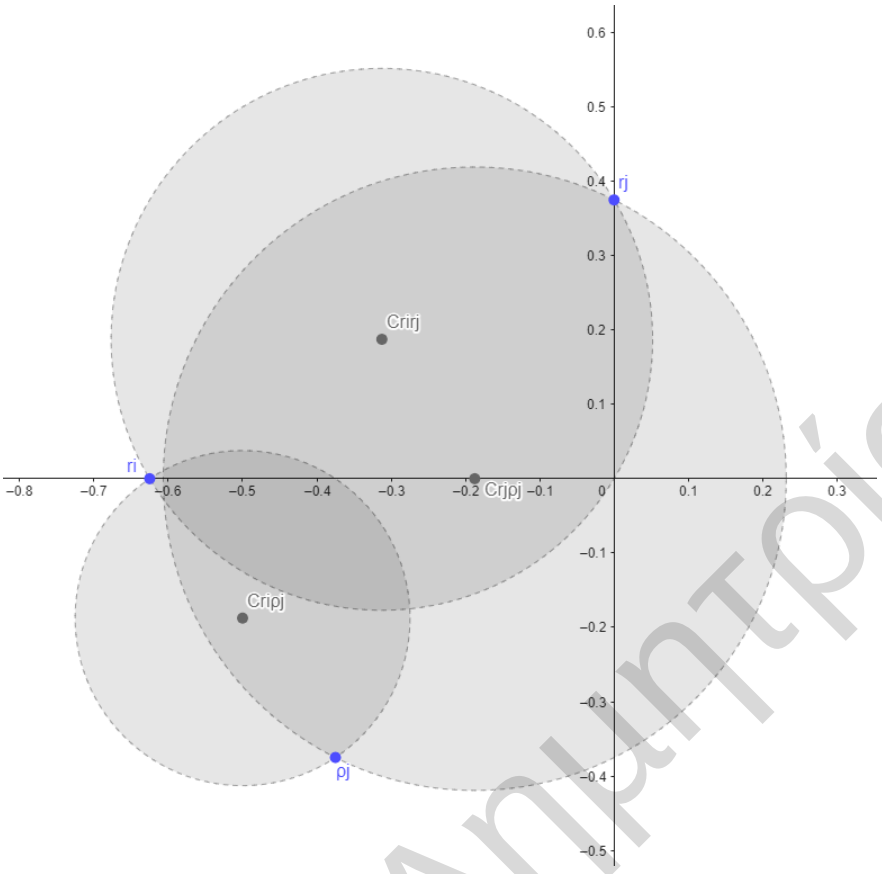
Ο κώδικας που υπολογίζει αν το σύστημα με τις ανισότητες έχει λύση, παρουσιάζεται στο Παράρτημα Ε΄.1 όπως και η εκτέλεση στο Παράρτημα Ε΄.2.

Η εκτέλεση του προγράμματος υπολόγισε λύση στο σύστημα με τις ακόλουθες τιμές στις μεταβλητές:

- α = 0.37500000186264515
- β = 0.62500000186264451
- γ = 0.37500000186264515
- δ = 0.37500000186264515

Αναθέτωντας τις υπολογιζόμενες τιμές των μεταβλητών στις εξισώσεις των κέντρων και ακτινών των κύκλων που υπολογίστηκαν στο προηγούμενο section, προκύπτει το ακόλουθο γράφημα (βλ. εικόνα 4.2). Όπως φαίνεται και στο γράφημα, οι εξισώσεις και οι υπολογιζόμενες τιμές των μεταβλητών είναι λογικές. Επίσης, παρατηρείται ότι οι αρχικές υποθέσεις ισχύουν αφού το σύστημα έχει λύση. Έτσι, δεν υπάρχει αντίφαση.

Στο γράφημα (βλ. εικόνα 4.2) οι κύκλοι με γκρίζο χρώμα είναι και το gabriel graph κριτήριο που καθορίζει αν τα ρομπότ είναι gabriel neighbors μεταξύ τους. Παρατηρείται ότι όλα τα ρομπότ γειτνιάζουν μεταξύ τους.



Σχήμα 4.2: Ανάθεση τιμών μεταβλητών α, β, γ, δ που υπολογίστηκαν από το πρόγραμμα python και σχεδιασμός κύκλων βάση των υπολογιζόμενων εξισώσεων των κύκλων.

4.4 Υπολογισμός MEC σημείων ri, rj, ρj

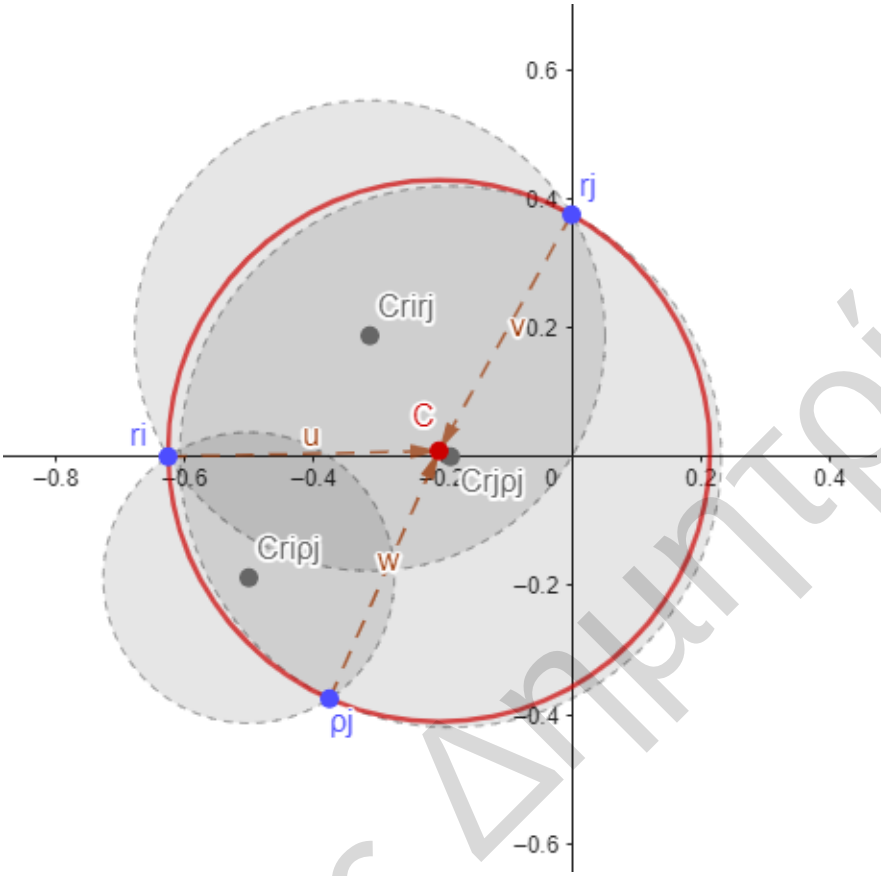
Για τον υπολογισμό του ελάχιστου κύκλου κάλυψης MEC των σημείων ri, rj, ρj, γράφτηκε ένα πρόγραμμα στη γλώσσα προγραμματισμού python, το οποίο υπολογίζει το κέντρο και την ακτίνα του ελάχιστου κύκλου κάλυψης αυτών των σημείων.

Ο κώδικας που υπολογίζει το κέντρο και την ακτίνα του ελάχιστου κύκλου κάλυψης (MEC), παρουσιάζεται στο Παράρτημα Δ΄.1 όπως και η εκτέλεση στο Παράρτημα Δ΄.2.

Η εκτέλεση του προγράμματος υπολόγισε τις ακόλουθες τιμές για το κέντρο και την ακτίνα του ελάχιστου κύκλου κάλυψης που περικλύει τα σημεία ri, rj και ρj:

- κέντρο = (-0.20535714 ,0.00892857)
- ακτίνα = 0.4197378332194894

Το επόμενο βήμα είναι να σχεδιαστεί ο υπολογιζόμενος κύκλος στο γράφημα 4.2. Το αποτέλεσμα παρουσιάζεται στο γράφημα 4.3.



Σχήμα 4.3: Σχεδιασμός ελάχιστου κύκλου κάλυψης (MEC) με βάση το κέντρο και την ακτίνα που υπολογίστηκαν από το πρόγραμμα python.

Όπως φαίνεται και στο πιο πάνω γράφημα (βλ. εικόνα 4.3), τα 3 σημεία ri, rj και pj βρίσκονται στην περιφέρεια του ελάχιστου κύκλου κάλυψης (κύκλος με κόκκινο χρώμα).

Αυτό, υποδεικνύει ότι και τα 3 σημεία έχουν το ίδιο σημείο προορισμού το οποίο είναι το κέντρο του ελάχιστου κύκλου κάλυψης (σημείο C). Έτσι, συμπεραίνεται ότι όποια σύγκρουση μπορεί να υπάρξει, θα είναι στο σημείο προορισμού η οποία είναι αποδεκτή.

Αν κάποιο από τα 3 ρομπότ ri, rj και pj, είχε επιπλέον gabriel neighbors, τότε η πιο πάνω διάταξη θα άλλαζε εντελώς, έχοντας διαφορετικό σημείο προορισμού. Αυτό σημαίνει ότι και το trajectory του κάθε ρομπότ θα μπορούσε να αλλάξει έχοντας σαν αποτέλεσμα το να μην διασταυρόνονται τα trajectories μεταξύ τους, δηλαδή να μην υπάρχουν collisions.

Το ρομπότ ri μπορεί να έχει επιπλέον gabriel neighbors σε αριστερότερο σημείο από αυτό ως προς τον x άξονα. Δεν μπορεί να έχει gabriel neighbor στο εσωτερικό των γκρίζων κύκλων έτσι ώστε να ικανοποιείται το κριτήριο του gabriel graph. Αυτό σημαίνει ότι και το σημείο

προορισμού του θα οδηγήτουν σε αριστερότερο σημείο αφού ο ελάχιστος κύκλος κάλυψης (MEC) του ri θα οδηγήτουν αριστερότερα έτσι ώστε να περικλύει και τα άλλα gabriel neighbors του.

Το ρομπότ rj μπορεί να έχει επιπλέον gabriel neighbors σε ψηλότερο σημείο από αυτό ως προς τον y άξονα ή σε δεξιότερο σημείο ως προς τον x άξονα. Δεν μπορεί να έχει gabriel neighbor στο εσωτερικό των γκρίζων κύκλων έτσι ώστε να ικανοποιείται το κριτήριο του gabriel graph. Αυτό σημαίνει ότι και το σημείο προορισμού του θα οδηγήτουν σε ψηλότερο ή δεξιότερο σημείο αφού ο ελάχιστος κύκλος κάλυψης (MEC) του rj θα οδηγήτουν ψηλότερα ή δεξιότερα έτσι ώστε να περικλύει και τα άλλα gabriel neighbors του.

Το ρομπότ pj μπορεί να έχει επιπλέον gabriel neighbors σε χαμηλότερο σημείο από αυτό ως προς τον y άξονα. Δεν μπορεί να έχει gabriel neighbor στο εσωτερικό των γκρίζων κύκλων έτσι ώστε να ικανοποιείται το κριτήριο του gabriel graph. Αυτό σημαίνει ότι και το σημείο προορισμού του θα οδηγήτουν σε χαμηλότερο σημείο αφού ο ελάχιστος κύκλος κάλυψης (MEC) του pj θα οδηγήτουν χαμηλότερα έτσι ώστε να περικλύει και τα άλλα gabriel neighbors του.

4.5 Έλεγχος αν υπάρχει πέραν της μιας λύσης στο σύστημα ανισοτήτων

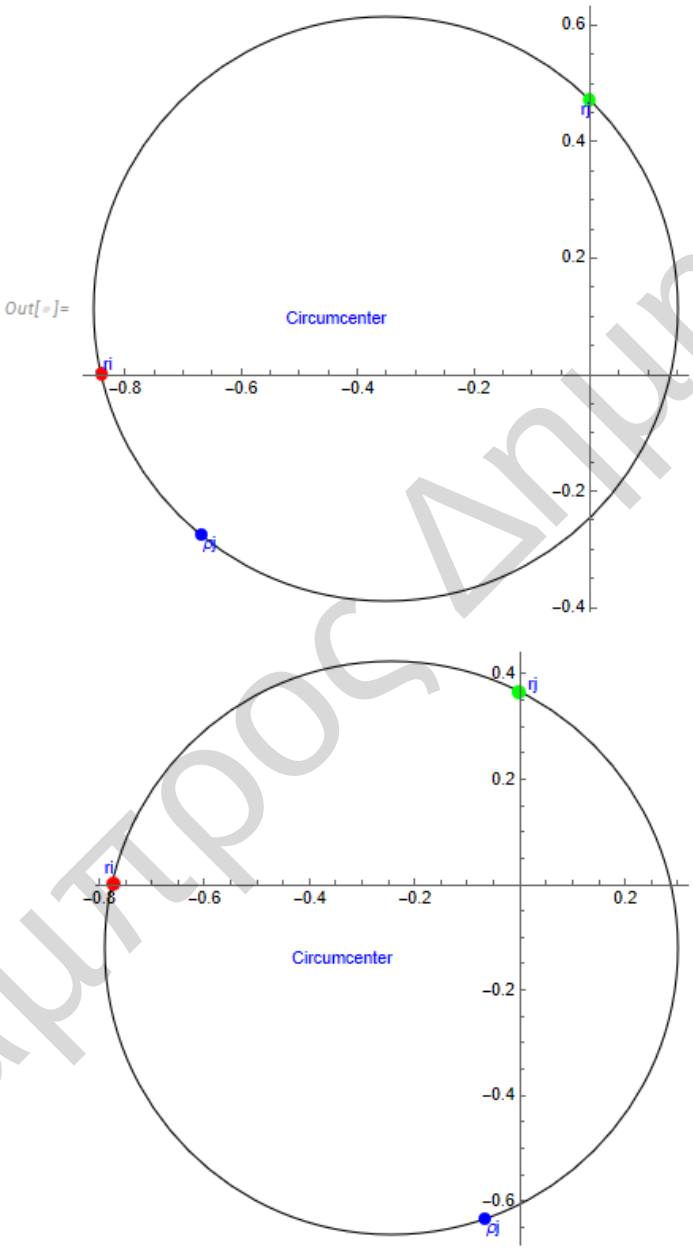
Για να ελεχθεί κατά πόσο το σύστημα ανισοτήτων που υπολογίστηκαν στο προηγούμενμο section έχει πέραν της μιας λύσης, έγινε χρήση του μαθηματικού εργαλείου mathematica.

Το mathematica υπολόγισε πέραν της μιας λύσης στο σύστημα ανισοτήτων. Συγκεκριμένα, ζητήθηκε να εντοπιστούν μέχρι 50 λύσεις στο σύστημα όπου και εντοπίστηκαν.

Ο κώδικας που υπολογίζει τις λύσεις στο σύστημα ανισοτήτων με τη χρήση του μαθηματικού εργαλείου mathematica, παρουσιάζεται στο Παράρτημα ΄.1 όπως και τα αποτελέσματα (λύσεις συστήματος) στο Παράρτημα ΄.2.

Αφού εντοπίστηκαν πέραν της μιας λύσης στο σύστημα, το επόμενο βήμα ήταν να υπολογιστεί ο ελάχιστος κύκλος κάλυψης MEC χρησιμοποιώντας λύσεις που υπολογίστηκαν για το σύστημα και να δημιουργηθεί το γράφημα με τον ελάχιστο κύκλο κάλυψης MEC και τα

3 σημεία. Ο κώδικας παρουσιάζεται στο Παράρτημα Ζ΄.1 και το γράφημα στην εικόνα 4.4. Σε περίπτωση που για όλες τις υπολογιζόμενες λύσεις, τα 3 σημεία βρίσκονταν πάνω στην περιφέρεια του ελάχιστου κύκλου κάλυψης, τότε ο αλγόριθμος θα θεωρείται collision-less, λόγω του ότι και τα 3 σημεία θα έχουν κοινό σημείο προορισμού, και οι συγκρούσεις στο κοινό σημείο προορισμού είναι αποδεκτές όπως επεξηγήθηκε προηγουμένως.



Σχήμα 4.4: Όπως παρατηρείται και στο γράφημα, τα 3 σημεία βρίσκονται στην περιφέρεια του MEC και στις δύο λύσεις. Αυτό υποδεικνύει ότι πιθανές συγκρούσεις θα είναι στο κοινό σημείο προορισμού τους που είναι αποδεκτό.

• Κέντρο ελάχιστου κύκλου κάλυψης (MEC):

$$k_{mec} = (-\frac{-a\beta^2 - a\gamma^2 - a^2\delta + \beta^2\delta - a\delta^2}{2(a\beta - a\gamma + \beta\delta)}, -\frac{-a^2\beta + a^2\gamma - \beta^2\gamma + \beta\gamma^2 + \beta\delta^2}{2(a\beta - a\gamma + \beta\delta)}) \quad (4.16)$$

• Ακτίνα ελάχιστου κύκλου κάλυψης (MEC):

$$R_{mec} = \sqrt{\frac{(-a^2\beta + a^2\gamma - \beta^2\gamma + \beta\gamma^2 + \beta\delta^2)^2}{4(a\beta - a\gamma + \beta\delta)^2} + (\beta - \frac{a\beta^2 - a\gamma^2 - a^2\delta + \beta^2\delta - a\delta^2}{2(a\beta - a\gamma + \beta\delta)})^2} \quad (4.17)$$

• Εξίσωση ελάχιστου κύκλου κάλυψης (MEC):

$$\begin{aligned}(x + \frac{-a\beta^2 - a\gamma^2 - a^2\delta + \beta^2\delta - a\delta^2}{2(a\beta - a\gamma + \beta\delta)})^2 + (y + \frac{-a^2\beta + a^2\gamma - \beta^2\gamma + \beta\gamma^2 + \beta\delta^2}{2(a\beta - a\gamma + \beta\delta)})^2 = \\ \frac{(-a^2\beta + a^2\gamma - \beta^2\gamma + \beta\gamma^2 + \beta\delta^2)^2}{4(a\beta - a\gamma + \beta\delta)^2} + (\beta - \frac{a\beta^2 - a\gamma^2 - a^2\delta + \beta^2\delta - a\delta^2}{2(a\beta - a\gamma + \beta\delta)})^2\end{aligned} \quad (4.18)$$

Χρησιμοποιώντας την εξίσωση του ελάχιστου κύκλου κάλυψης και 2 από τις λύσεις του συστήματος ανισοτήτων (μεταβλητές α, β, γ, δ), σχεδιάστηκε ο ελάχιστος κύκλος κάλυψης μαζί με τα 3 ρομπότ. Όπως φαίνεται και στο γράφημα 4.4, και στις 2 λύσεις του συστήματος τα 3 σημεία - ρομπότ εμπίπτουν πάνω στην περιφέρεια του ελάχιστου κύκλου κάλυψης. Αυτό σημαίνει ότι αν τα 3 ρομπότ δεν έχουν άλλα γειτονικά ρομπότ, όταν εκτελεστεί ο αλγόριθμος για το κάθε ρομπότ από τα 3, θα υπολογίσει ακριβώς τον ίδιο ελάχιστο κύκλο κάλυψης και κατ' επέκταση το ίδιο σημείο προορισμού. Έτσι, τα 3 ρομπότ θα ¨συναντηθούν¨ στο κοινό σημείο προορισμού τους. Αυτή η σύγκρουση θα είναι αποδεκτή λόγω του ότι είναι στο κοινό σημείο προορισμού.

Στη συνέχεια, έπρεπε να ελεχθεί αν το σύστημα ανισοτήτων θα είχε λύση η οποία να εμφάνιζε κάποιο από τα 3 σημεία στο εσωτερικό του MEC. Αν το σύστημα είχε λύση, τότε το ρομπότ που θα βρίσκεται στο εσωτερικό θα είχε διαφορετικό σημείο προορισμού αφού θα είχε και διαφορετικό ελάχιστο κύκλο κάλυψης, και έτσι θα υπήρχε περίπτωση να προέκυπταν πρόωρες συγκρούσεις. Για να επιτευχθεί αυτό, στο σύστημα ανισοτήτων προστέθηκαν οι ακόλουθες ανισότητες (η μια ανεξάρτητα από την άλλη):

$$d[r_i, k(MEC)] < R_{mec}$$

$$\sqrt{(-\frac{-a\beta^2 - a\gamma^2 - a^2\delta + \beta^2\delta - a\delta^2}{2(a\beta - a\gamma + \beta\delta)} + \beta)^2 + (-\frac{-a^2\beta + a^2\gamma - \beta^2\gamma + \beta\gamma^2 + \beta\delta^2}{2(a\beta - a\gamma + \beta\delta)} - 0)^2} < \sqrt{\frac{(-a^2\beta + a^2\gamma - \beta^2\gamma + \beta\gamma^2 + \beta\delta^2)^2}{4(a\beta - a\gamma + \beta\delta)^2} + (\beta - \frac{a\beta^2 - a\gamma^2 - a^2\delta + \beta^2\delta - a\delta^2}{2(a\beta - a\gamma + \beta\delta)})^2} \tag{4.19}$$

$$d[r_j, k(MEC)] < R_{mec}$$

$$\sqrt{(-\frac{-a\beta^2 - a\gamma^2 - a^2\delta + \beta^2\delta - a\delta^2}{2(a\beta - a\gamma + \beta\delta)} - 0)^2 + (-\frac{-a^2\beta + a^2\gamma - \beta^2\gamma + \beta\gamma^2 + \beta\delta^2}{2(a\beta - a\gamma + \beta\delta)} - a)^2} < \sqrt{\frac{(-a^2\beta + a^2\gamma - \beta^2\gamma + \beta\gamma^2 + \beta\delta^2)^2}{4(a\beta - a\gamma + \beta\delta)^2} + (\beta - \frac{a\beta^2 - a\gamma^2 - a^2\delta + \beta^2\delta - a\delta^2}{2(a\beta - a\gamma + \beta\delta)})^2} \tag{4.20}$$

$$d[\rho_j, k(MEC)] < R_{mec}$$

$$\sqrt{(-\frac{-a\beta^2 - a\gamma^2 - a^2\delta + \beta^2\delta - a\delta^2}{2(a\beta - a\gamma + \beta\delta)} + \gamma)^2 + (-\frac{-a^2\beta + a^2\gamma - \beta^2\gamma + \beta\gamma^2 + \beta\delta^2}{2(a\beta - a\gamma + \beta\delta)} + \delta)^2} < \sqrt{\frac{(-a^2\beta + a^2\gamma - \beta^2\gamma + \beta\gamma^2 + \beta\delta^2)^2}{4(a\beta - a\gamma + \beta\delta)^2} + (\beta - \frac{a\beta^2 - a\gamma^2 - a^2\delta + \beta^2\delta - a\delta^2}{2(a\beta - a\gamma + \beta\delta)})^2} \tag{4.21}$$

Χρησιμοποιώντας τις ανισότητες που υπολογίστηκαν πιο πάνω (βλ. εδώ 4.2.2), αρχικά προστέθηκαν στο σύστημα ανισοτήτων και οι 3 ανισότητες που αφορούν τον ελάχιστο κύκλο κάλυψης και ελέχθηκε με τη χρήση του mathematica αν το σύστημα έχει λύση. Όπως ήταν αναμενόμενο, δεν προέκυψε κάποια λύση στο σύστημα.

Το επόμενο βήμα ήταν να προστεθεί στο σύστημα ανισοτήτων (βλ. εδώ 4.2.2) η μια εκ των τριών ανισοτήτων (σε κάθε σύστημα μια εκ των τριών ανισοτήτων) που αφορούν τον ελάχιστο

κύκλο κάλυψης και να δημιουργηθούν 3 ανεξάρτητα συστήματα ανισοτήτων. Δηλαδή, στο σύστημα ανισοτήτων, οι 3 πιο πάνω ανισότητες προστέθηκαν η μια ανεξάρτητα από την άλλη. Δημιουργήθηκαν 3 συστήματα ανισοτήτων όπου το καθένα περιείχε τις ανισότητες του προηγούμενο section και μια εκ των τριών πιο πάνω ανισοτήτων.

Με τη χρήση του mathematica, ελέχθηκε αν τα 3 ανεξάρτητα συστήματα ανισοτήτων είχαν λύση. Η εκτέλεση έδειξε ότι δεν υπάρχει λύση σε κανένα από τα 3 συστήματα ανισότήτων. Αυτό οδηγεί στο συμπέρασμα, ότι για αυτό το σενάριο δεν μπορεί να προκύψει περίπτωση όπου τουλάχιστον ένα εκ των τριών σημείων - ρομπότ να βρίσκονται στο εσωτερικό του ελάχιστου κύκλου κάλυψης. Έτσι, αφού τα 3 ρομπότ θα βρίσκονται πάντα στην περιφέρεια, θα έχουν και κοινό σημείο προορισμού. Έτσι, αν προκύψει κάποια σύγκρουση, θα είναι στο κοινό σημείο προορισμού και των τριών ρομπότ η οποια είναι αποδεκτή.

Ο κώδικας που γράφτηκε και εκτελέστηκε στο μαθηματικό εργαλείο mathematica παρουσιάζεται στο Παράρτημα Η΄.1. Το αποτέλεσμα της εκτέλεσης ήταν το κενό σύνολο. Οπόταν, μπορεί να διαπιστωθεί ότι όλες οι λύσεις του συστήματος, τοποθετούν τα 3 σημεία στην περιφέρεια του MEC και καμία λύση δεν υπάρχει που να τοποθετεί κάποιο από τα 3 ρομπότ στο εσωτερικό του MEC.

Κεφάλαιο 5

Ανασκόπηση και Μελλοντική Εργασία

5.1 Ανασκόπηση και Συμπεράσματα

Αφού μελετήθηκαν αρκετοί αλγόριθμοι συγκέντρωσης που υπάρχουν στη βιβλιογραφία και κατανοήθηκε ο τρόπος εκτέλεσης τους, το επόμενο βήμα ήταν να γίνει προσπάθεια να αποδειχθεί κατά πόσο ο αλγόριθμος συγκέντρωσης Go-To-The-Gabriel-Center (GTGC) είναι collision-less ή όχι.

Αρχικά, μέσω προσομοιώσεων και case analysis, έγινε προσπάθεια να καλυφθούν όλα τα σενάρια και καταστάσεις που μπορεί να βρεθεί ένα ρομπότ στο ευκλείδειο επίπεδο και με την χρήση θεωρημάτων που υπάρχουν στη βιβλιογραφία και τον τρόπο εκτέλεσης του αλγορίθμου, λήφθηκαν κάποια συμπεράσματα που αφορούν το αν ο αλγόριθμος είναι collision-less ή όχι.

Μέσω των προσομοιώσεων, πάρθηκε το συμπέρασμα ότι ο αλγόριθμος μπορεί να θεωρηθεί collision-less εκτός από μια συγκεκριμένη κατάσταση. Δηλαδή, το να βρεθεί σε ένα cross-shaped graph. Για τη συγκεκριμένη κατάσταση, δεν βρέθηκε κάτι το οποίο να πείθει ότι μπορούν να αποφευχθούν οι συγκρούσεις.

Στη συνέχεια, χρησιμοποιώντας τα συμπεράσματα που λήφθηκαν από τις προσομοιώσεις, έγινε μια προσπάθεια να αποδειχθεί κατά πόσο είναι collision-less ο συγκεκριμένος αλγόριθμος με μαθηματικό τρόπο. Για να επιτευχθεί αυτό, λήφθηκε σαν αναφορά μια κατάσταση η οποία αφορά ένα γενικό σενάριο, η οποία περιέχει τρία ρομπότ-σημεία, και στόχος ήταν να διερευνηθεί αν μπορούν τα trajectories που θα ακολουθήσουν τα ρομπότ, να διασταυρωθούν

μεταξύ τους. Δηλαδή, αν θα υπήρχαν συγκρούσεις.

Χρησιμοποιώντας το gabriel graph criterion, δημιουργήθηκε ένα σύστημα ανισοτήτων και ελέχθηκε κατά πόσο υπάρχει λύση στο σύστημα και αν υπάρχει, τότε, τι συμπερένουμε από τη λύση ή τις λύσεις του συστήματος. Από τις λύσεις του συστήματος, λήφθηκε το συμπέρασμα ότι οι μοναδικές συγκρούσεις που μπορούν να προκύψουν είναι στο κοινό σημείο προορισμού, όπου αυτές οι συγκρούσεις είναι αποδεκτές.

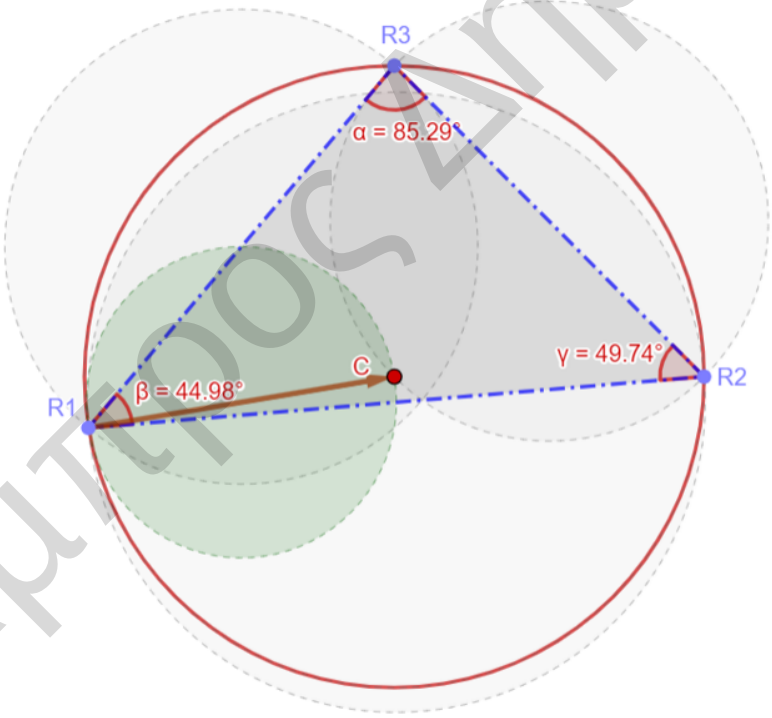
Από τα βήματα που περιγράφηκαν πιο πάνω, συμπεραίνεται ότι ο συγκεκριμένος αλγόριθμος μπορεί να θεωρηθεί collision-less εκτός από την κατάσταση που μπορεί να βρεθεί σε cross-shaped graph. Παρόλα αυτά, λόγω του Lebesgue measure, το οποίο αναφέρει ότι αν μια κατάσταση είναι πολύ σπάνια, δηλαδή η πιθανότητα εμφάνισης της τείνει προς το 0, τότε μπορεί να θεωρηθεί ότι έχει πιθανότητα εμφάνισης 0. Έτσι, σαν γενική εικόνα, λαμβάνεται το συμπέρασμα ότι ο αλγόριθμος συγκέντρωσης Go-To-The-Gabriel-Center (GTGC) είναι collision-less.

5.2 Μελλοντική Εργασία

Σαν μελλοντική εργασία θα θέλαμε να μελετήσουμε περαιτέρω αλγόριθμους συγκέντωσης σε συνεχή χρόνο και στις 2 διαστάσεις του ευκλείδειου επιπέδου και να γίνει μια προσπάθεια για δημιουργία ενός καινούργιου αλγόριθμου συγκέντρωσης ο οποίος να είναι collision-less για οποιαδήποτε κατάσταση βρεθεί. Για να επιτευχθεί αυτό, θα πρέπει να βρεθεί ένας διαφορετικός τρόπος με τον οποίο το κάθε ρομπότ θα καθορίζει τους γείτονες του από το ευκλείδειο επίπεδο και επίσης, να εντοπίζει το σημείο προορισμού του με διαφορετικό τρόπο ο οποίος να είναι πιο ασφαλές όσο αφορά τις συγκρούσεις. Εκτός από τη δημιουργία καινούργιου αλγόριθμου, θα μπορούσαν να εκμεταλεφθούν οι υφιστάμενοι αλγόριθμοι και προσθέτοντας κάποια additional capabilities που μπορεί να έχει το κάθε ρομπότ, όπως το να έχει στατική μνήμη ή να αυξηθεί το viewing range, να επιτευχθεί ο εντοπισμός αλγορίθμου ο οποίος θα είναι collision-less για όλες τις καταστάσεις.

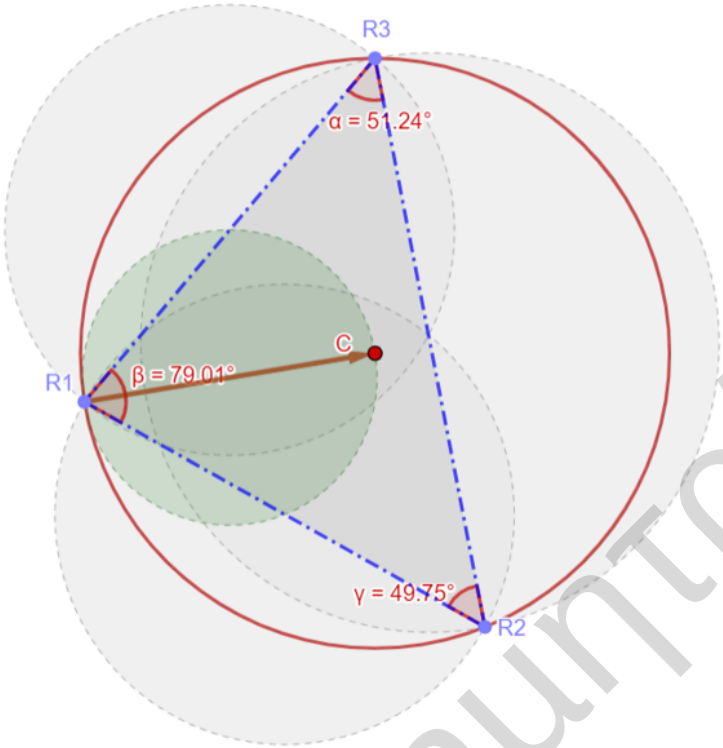
Παράρτημα Α΄

Γραφήματα Σεναρίου 1 - Περίπτωσης 2



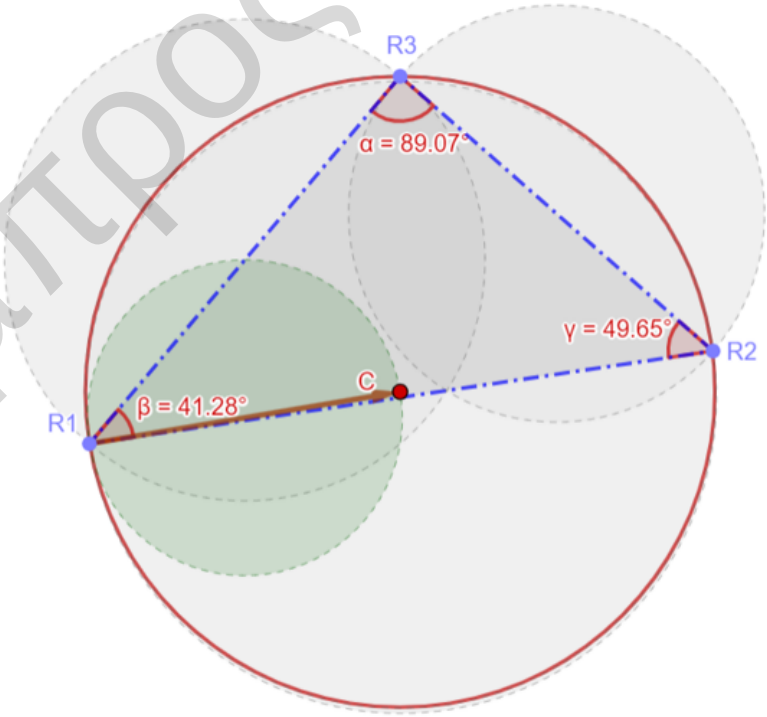
Σχήμα Α΄.1: Απεικόνιση Σεναρίου 1 - Κατάστασης 2.3

.



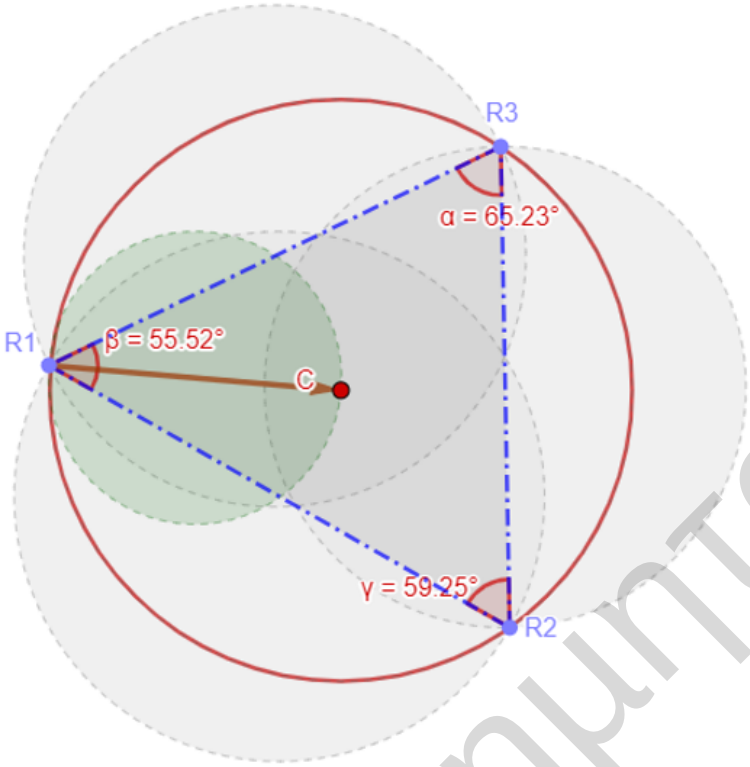
Σχήμα Α΄.2: Απεικόνιση Σεναρίου 1 - Κατάστασης 2.4

.



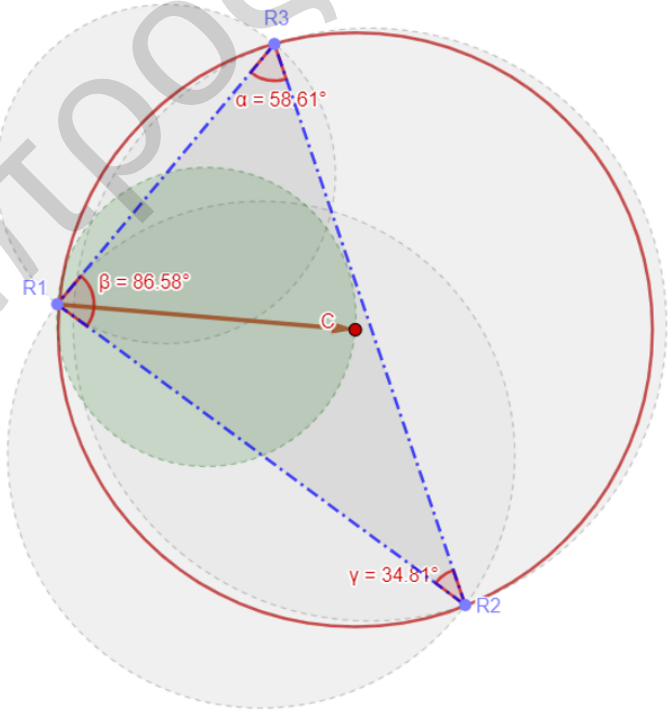
Σχήμα Α΄.3: Απεικόνιση Σεναρίου 1 - Κατάστασης 2.5

.



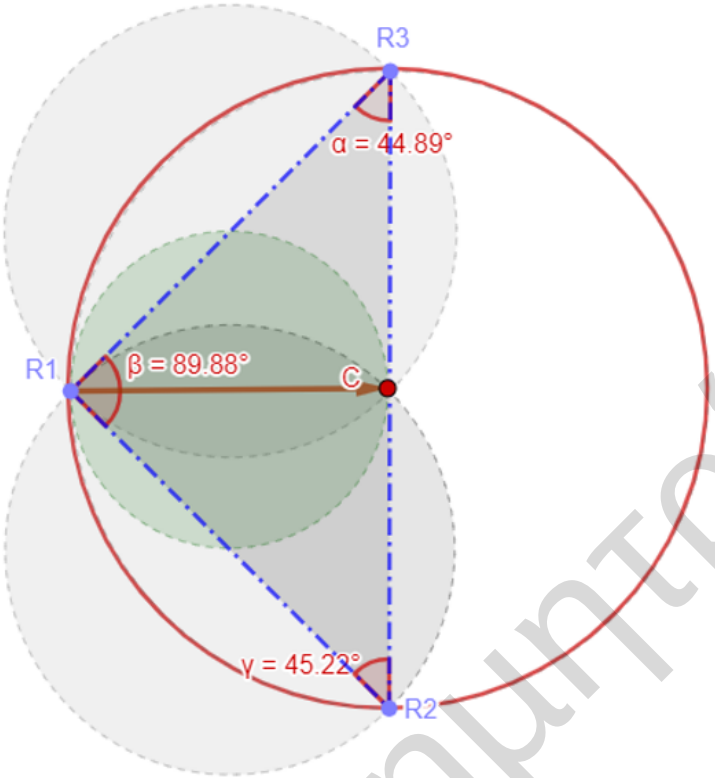
Σχήμα Α΄.4: Απεικόνιση Σεναρίου 1 - Κατάστασης 2.6

.



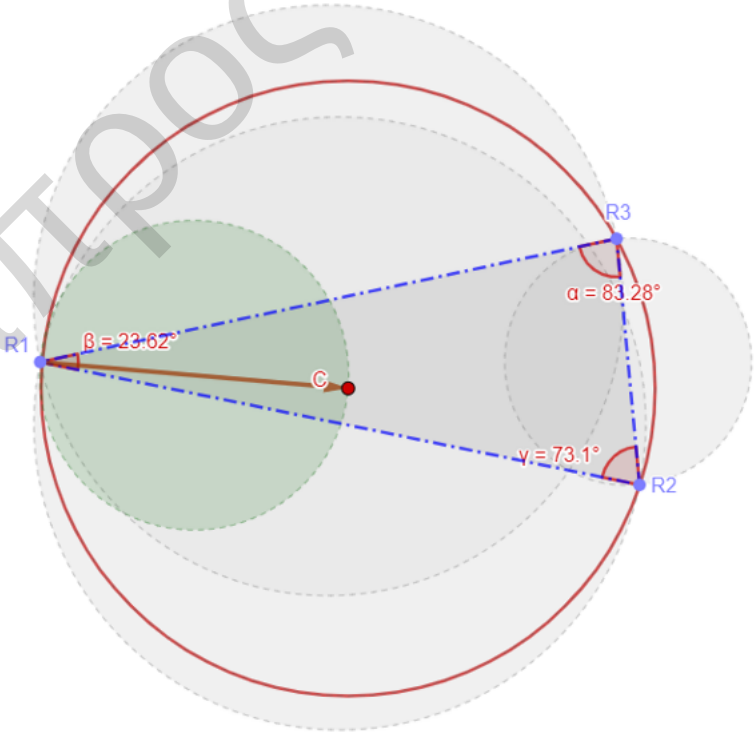
Σχήμα Α΄.5: Απεικόνιση Σεναρίου 1 - Κατάστασης 2.7

.



Σχήμα Α΄.6: Απεικόνιση Σεναρίου 1 - Κατάστασης 2.8

.

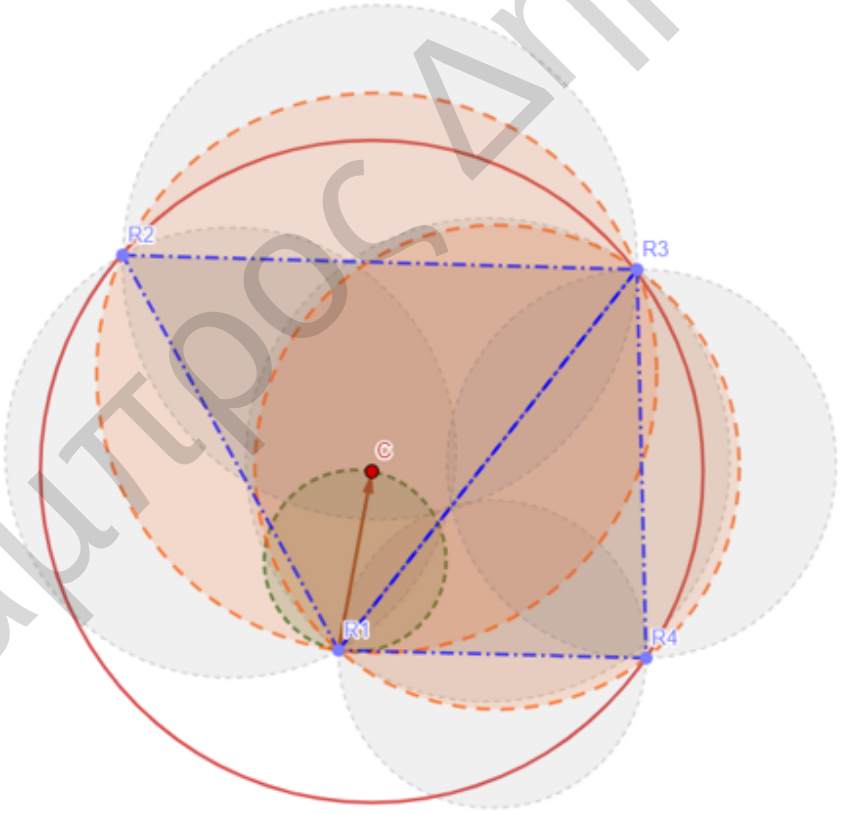


Σχήμα Α΄.7: Απεικόνιση Σεναρίου 1 - Κατάστασης 2.9

.

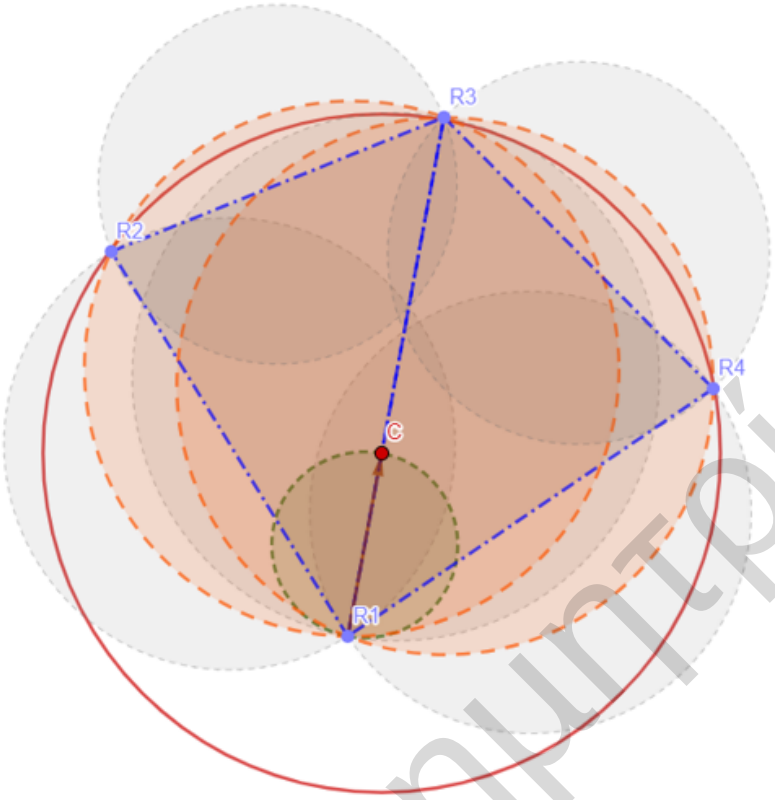
Παράρτημα Β΄

Γραφήματα Σεναρίου 2 - Περίπτωσης 1



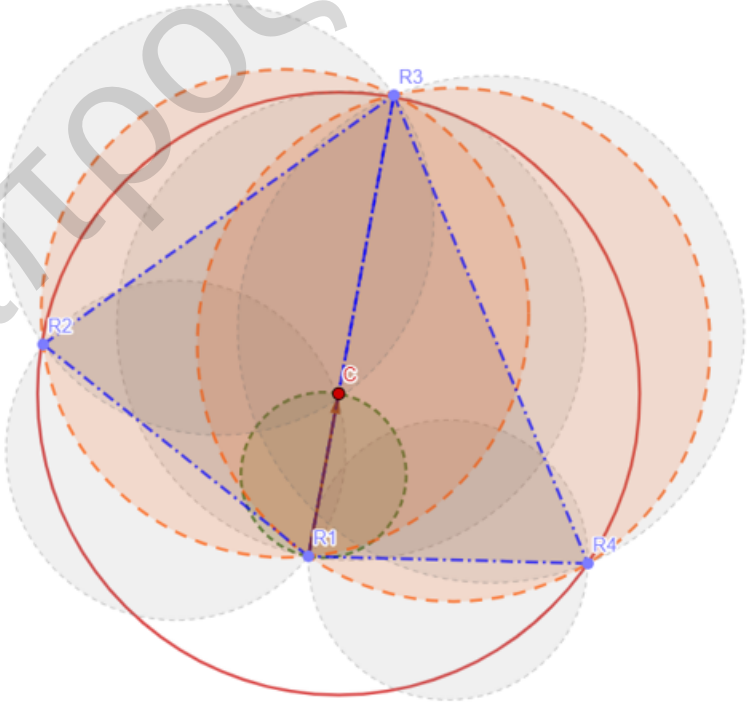
Σχήμα Β΄.1: Απεικόνιση Σεναρίου 2 - Κατάστασης 1.2

.



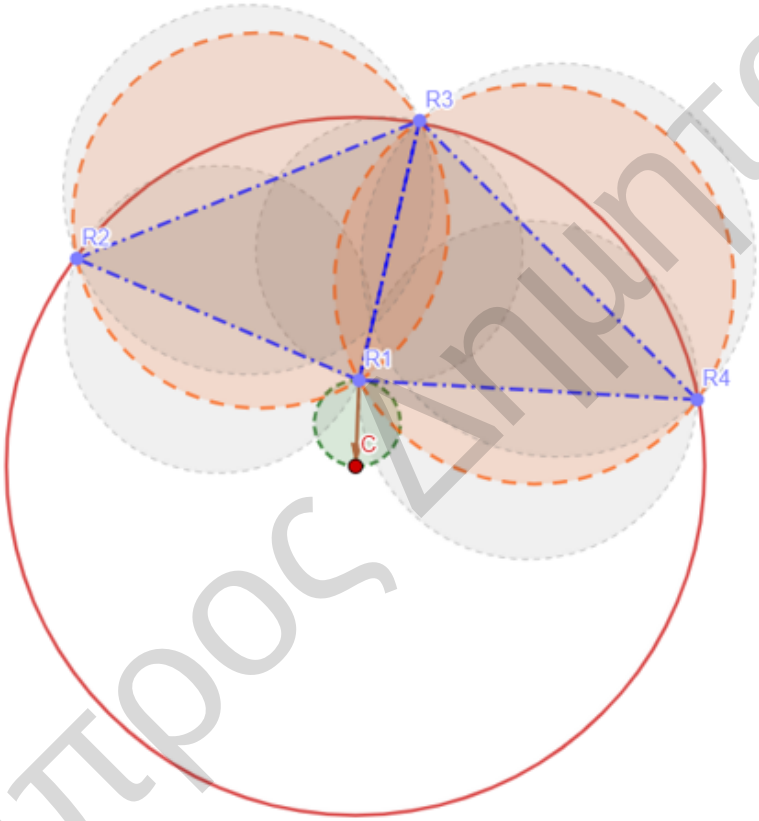
Σχήμα Β΄.2: Απεικόνιση Σεναρίου 2 - Κατάστασης 1.3

.



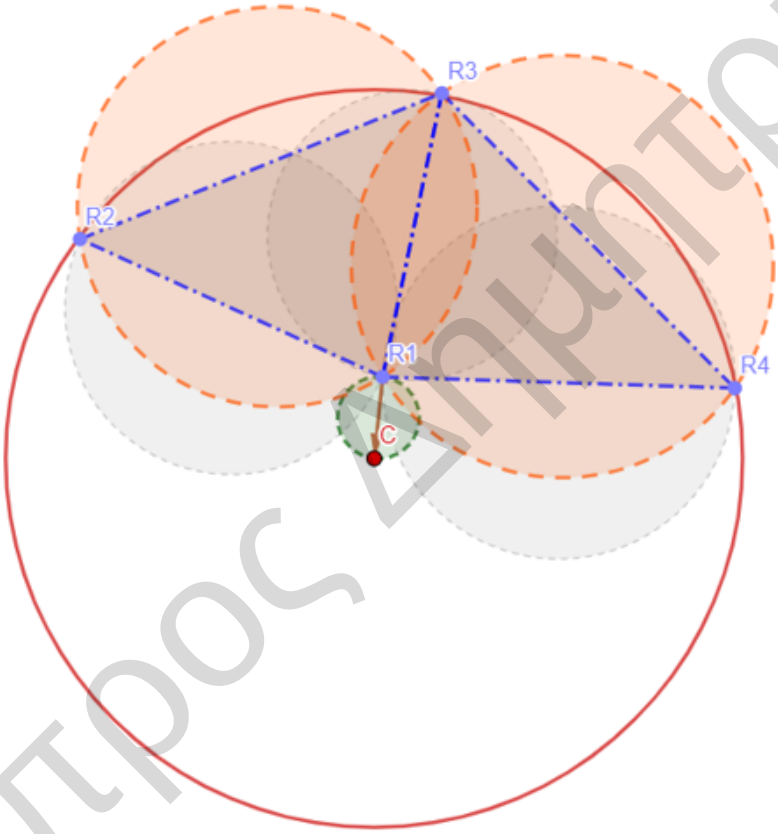
Σχήμα Β΄.3: Απεικόνιση Σεναρίου 2 - Κατάστασης 1.4

.



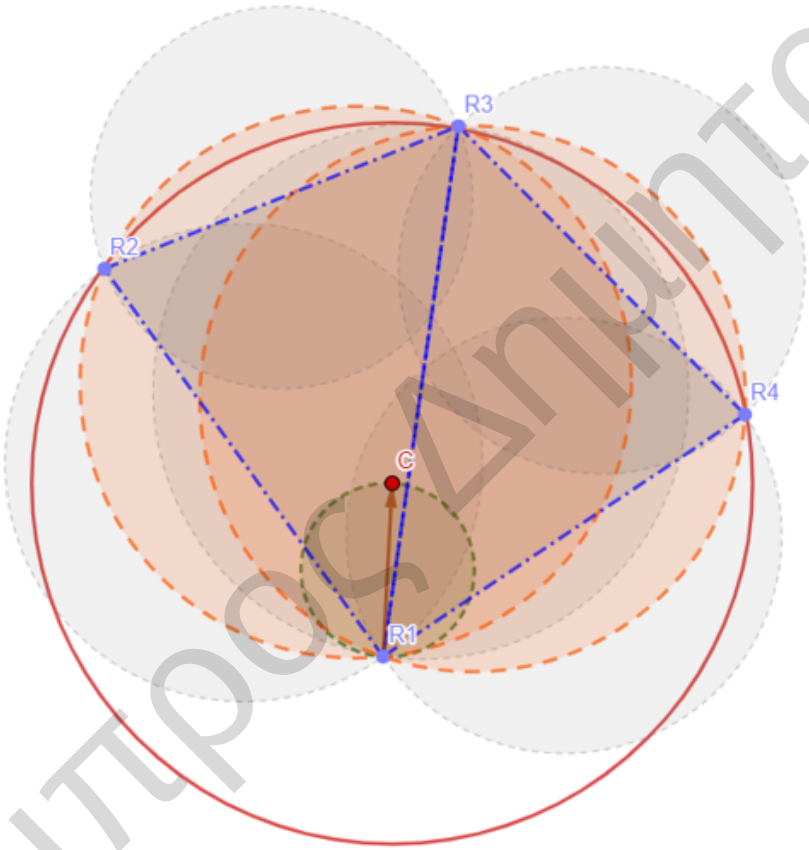
Σχήμα Β΄.4: Απεικόνιση Σεναρίου 2 - Κατάστασης 1.5

.



Σχήμα Β΄.5: Απεικόνιση Σεναρίου 2 - Κατάστασης 1.6

.



Σχήμα Β΄.6: Απεικόνιση Σεναρίου 2 - Κατάστασης 1.7

.

Παράρτημα Γ΄

Κώδικας Python για υπολογισμό συστήματος ανισοτήτων

Γ΄.1 Κώδικας

```
import numpy as np
from scipy.optimize import minimize

# Define the inequalities as constraint functions
def constraint1(x):
    c, d, a, b = x
    return c**2 + d**2 - b*c + a*d  # satisfy c^2 + d^2-bc+ad>0

def constraint2(x):
    c, d, a, b = x
    return a**2 + a*d + b*c    # satisfy a^2+ad+bc>0

def constraint3(x):
    c, d, a, b = x
    return b**2 - b*c - a*d    # satisfy b^2-bc-ad>0
```

```
def constraint4(x):
    c, d, a, b = x
    return 1 - (a**2 + b**2)  # satisfy a^2+b^2<=1

def constraint5(x):
    c, d, a, b = x
    return 1 - (b**2 -2*b*c + c**2 + d**2)  # satisfy b^2-2bc+c^2+d^2<=1

def constraint6(x):
    c, d, a, b = x
    return 1 - (a**2 + 2*a*d + d**2 + c**2)  # satisfy a^2+2ad+d^2+c^2<=1

def constraint7(x):
    c, d, a, b = x
    return a - 1e-6  # Slightly positive to satisfy a > 0

def constraint8(x):
    c, d, a, b = x
    return b - 1e-6  # Slightly positive to satisfy b > 0

def constraint9(x):
    c, d, a, b = x
    return c - 1e-6  # Slightly positive to satisfy c > 0

def constraint10(x):
    c, d, a, b = x
    return d - 1e-6  # Slightly positive to satisfy d > 0


def constraint11(x):
    c, d, a, b = x
    return 1 - a     # satisfy a <= 1
```

```
def constraint12(x):
    c, d, a, b = x
    return 1 - b     # satisfy b <= 1

def constraint13(x):
    c, d, a, b = x
    return 1 - c     # satisfy c <= 1

def constraint14(x):
    c, d, a, b = x
    return 1 - d     # satisfy d <= 1


# Combine the constraints into a list
constraints = [
    {'type': 'ineq', 'fun': constraint1},
    {'type': 'ineq', 'fun': constraint2},
    {'type': 'ineq', 'fun': constraint3},
    {'type': 'ineq', 'fun': constraint4},
    {'type': 'ineq', 'fun': constraint5},
    {'type': 'ineq', 'fun': constraint6},
    {'type': 'ineq', 'fun': constraint7},
    {'type': 'ineq', 'fun': constraint8},
    {'type': 'ineq', 'fun': constraint9},
    {'type': 'ineq', 'fun': constraint10},
    {'type': 'ineq', 'fun': constraint11},
    {'type': 'ineq', 'fun': constraint12},
    {'type': 'ineq', 'fun': constraint13},
    {'type': 'ineq', 'fun': constraint14}
]
```

```
# Define a dummy objective function (we only care about the constraints)
def objective(x):
    return 0

# Initial guess (should be within the feasible region if known)
x0 = [0.5, 0.5, 0.5, 0.5]

# Run the optimization to check for feasibility
result = minimize(objective, x0, constraints=constraints, method='SLSQP', options={

# Check if a feasible solution was found
if result.success:
    print('A feasible solution exists:')
    print(f'Solution: c = {result.x[0]}, d = {result.x[1]}, a = {result.x[2]}, b = {
else:
    print('No feasible solution found.')
```

Γ΄.2 Εκτέλεση προγράμματος

```
C:\Users\lambrosd\Desktop\python>python .\calculate_inequalities_system.py
Optimization terminated successfully    (Exit mode 0)
            Current function value: 0
            Iterations: 1
            Function evaluations: 6
            Gradient evaluations: 1
A feasible solution exists:
Solution: c = 0.37500000186264515, d = 0.37500000186264515, a = 0.37500000186264515, b = 0.6250000018626451
```

Σχήμα Γ΄.1: Εκτέλεση προγράμματος για υπολογισμό συστήματος ανισοτήτων.

Παράρτημα Δ΄

Κώδικας Python για υπολογισμό MEC

Δ΄.1 Κώδικας

```
import numpy as np
from scipy.spatial import ConvexHull, Delaunay

def min_circle(points):
    def make_circle(points):
        if len(points) == 0:
            return (None, None), -np.inf
        elif len(points) == 1:
            return points[0], 0
        elif len(points) == 2:
            c = np.mean(points, axis=0)
            r = np.linalg.norm(points[0] - c)
            return c, r

        # Use convex hull to find a starting point
        hull = ConvexHull(points)
        start_triangle = points[hull.vertices]

        if len(start_triangle) == 3:
```

```
            c, r = circumcenter(start_triangle[0], start_triangle[1], start_triangle
        else:
            c, r = make_circle(start_triangle[:-1])

            if np.linalg.norm(start_triangle[-1] - c) > r:
                c, r = make_circle(np.append(start_triangle[:-1], [start_triangle[-

        return c, r

    def circumcenter(a, b, c):
        d = 2 * (a[0] * (b[1] - c[1]) + b[0] * (c[1] - a[1]) + c[0] * (a[1] - b[1]))
        ux = ((np.linalg.norm(a) ** 2) * (b[1] - c[1]) + (np.linalg.norm(b) ** 2) *
        uy = ((np.linalg.norm(a) ** 2) * (c[0] - b[0]) + (np.linalg.norm(b) ** 2) *
        return np.array([ux, uy]), np.linalg.norm(np.array([ux, uy]) - a)

    center, radius = make_circle(points)
    return center, radius

# Given points
a = 0.37500000186264515
b = 0.6250000018626451
c = 0.37500000186264515
d = 0.37500000186264515

points = np.array([[-b, 0], [0, a], [-c, -d]])

# Calculate minimum enclosing circle
center, radius = min_circle(points)

# Print the results
print("Center of the minimum enclosing circle:", center)
print("Radius of the minimum enclosing circle:", radius)
```

Δ΄.2 Εκτέλεση προγράμματος

```
C:\Users\lambrosd\Desktop\python>python .\calculate_mec.py
Center of the minimum enclosing circle: [-0.20535714  0.00892857]
Radius of the minimum enclosing circle: 0.4197378332194894
```

Σχήμα Δ΄.1: Εκτέλεση προγράμματος για υπολογισμό κέντρου και ακτίνας MEC.

Παράρτημα Ε΄

Κώδικας Python για υπολογισμό συστήματος ανισοτήτων

Ε΄.1 Κώδικας

```
import numpy as np
from scipy.optimize import minimize

# Define the inequalities as constraint functions
def constraint1(x):
    c, d, a, b = x
    return c**2 + d**2 - b*c + a*d  # satisfy c^2 + d^2-bc+ad>0

def constraint2(x):
    c, d, a, b = x
    return a**2 + a*d + b*c    # satisfy a^2+ad+bc>0

def constraint3(x):
    c, d, a, b = x
    return b**2 - b*c - a*d    # satisfy b^2-bc-ad>0
```

```
def constraint4(x):
    c, d, a, b = x
    return 1 - (a**2 + b**2)  # satisfy a^2+b^2<=1

def constraint5(x):
    c, d, a, b = x
    return 1 - (b**2 -2*b*c + c**2 + d**2)  # satisfy b^2-2bc+c^2+d^2<=1

def constraint6(x):
    c, d, a, b = x
    return 1 - (a**2 + 2*a*d + d**2 + c**2)  # satisfy a^2+2ad+d^2+c^2<=1

def constraint7(x):
    c, d, a, b = x
    return a - 1e-6  # Slightly positive to satisfy a > 0

def constraint8(x):
    c, d, a, b = x
    return b - 1e-6  # Slightly positive to satisfy b > 0

def constraint9(x):
    c, d, a, b = x
    return c - 1e-6  # Slightly positive to satisfy c > 0

def constraint10(x):
    c, d, a, b = x
    return d - 1e-6  # Slightly positive to satisfy d > 0


def constraint11(x):
    c, d, a, b = x
    return 1 - a     # satisfy a <= 1
```

```
def constraint12(x):
    c, d, a, b = x
    return 1 - b     # satisfy b <= 1

def constraint13(x):
    c, d, a, b = x
    return 1 - c     # satisfy c <= 1

def constraint14(x):
    c, d, a, b = x
    return 1 - d     # satisfy d <= 1


# Combine the constraints into a list
constraints = [
    {'type': 'ineq', 'fun': constraint1},
    {'type': 'ineq', 'fun': constraint2},
    {'type': 'ineq', 'fun': constraint3},
    {'type': 'ineq', 'fun': constraint4},
    {'type': 'ineq', 'fun': constraint5},
    {'type': 'ineq', 'fun': constraint6},
    {'type': 'ineq', 'fun': constraint7},
    {'type': 'ineq', 'fun': constraint8},
    {'type': 'ineq', 'fun': constraint9},
    {'type': 'ineq', 'fun': constraint10},
    {'type': 'ineq', 'fun': constraint11},
    {'type': 'ineq', 'fun': constraint12},
    {'type': 'ineq', 'fun': constraint13},
    {'type': 'ineq', 'fun': constraint14}
]
```

```
# Define a dummy objective function (we only care about the constraints)
def objective(x):
    return 0

# Initial guess (should be within the feasible region if known)
x0 = [0.5, 0.5, 0.5, 0.5]

# Run the optimization to check for feasibility
result = minimize(objective, x0, constraints=constraints, method='SLSQP', options

# Check if a feasible solution was found
if result.success:
    print('A feasible solution exists:')
    print(f'Solution: c = {result.x[0]}, d = {result.x[1]}, a = {result.x[2]}, b =
else:
    print('No feasible solution found.')
```

Ε΄.2 Εκτέλεση προγράμματος

```
C:\Users\lambrosd\Desktop\python>python .\calculate_inequalities_system.py
Optimization terminated successfully    (Exit mode 0)
            Current function value: 0
            Iterations: 1
            Function evaluations: 6
            Gradient evaluations: 1
A feasible solution exists:
Solution: c = 0.37500000186264515, d = 0.37500000186264515, a = 0.37500000186264515, b = 0.6250000018626451
```

Σχήμα Ε΄.1: Εκτέλεση προγράμματος για υπολογισμό συστήματος ανισοτήτων.

Παράρτημα ΄

Κώδικας Mathematica για υπολογισμό λύσεων στο σύστημα ανισοτήτων

΄.1 Κώδικας

```
In[•]:= (*Define the inequalities*)
inequalities = {c^2 + d^2 - b*c + a*d > 0, a^2 + a*d + b*c > 0, b^2 - b*c - a*d > 0, a^2 + b^2 ≤ 1, b^2 - 2*b*c + c^2 + d^2 ≤ 1,
   a^2 + 2*a*d + d^2 + c^2 ≤ 1, a > 0, b > 0, c > 0, d > 0, a ≤ 1, b ≤ 1, c ≤ 1, d ≤ 1};

(*Use FindInstance to find a solution*)
solution = FindInstance[inequalities, {a, b, c, d}, Reals, 50]
```

Σχήμα ΄.1: Κώδικας mathematica για υπολογισμό 50 λύσεων στο σύστημα ανισοτήτων.

΄.2 Εκτέλεση προγράμματος

```
In[•]:= N[solution]
Out[•]= {{a → 0.363091, b → 0.771065, c → 0.0648649, d → 0.634803}, {a → 0.469759, b → 0.837931, c → 0.666667, d → 0.275597},
  {a → 0.429471, b → 0.767742, c → 0.328042, d → 0.515192}, {a → 0.590423, b → 0.781931, c → 0.0457516, d → 0.165493}, {a → 0.415575, b → 0.909559, c → 0.0497653, d → 0.510642},
  {a → 0.063381, b → 0.35035, c → 0.183997, d → 0.687023}, {a → 0.462861, b → 0.696246, c → 0.392509, d → 0.423181}, {a → 0.791962, b → 0.405904, c → 0.26506, d → 0.0611187},
  {a → 0.427472, b → 0.102439, c → 0.0334686, d → 0.00975285}, {a → 0.475357, b → 0.851323, c → 0.743091, d → 0.165551}, {a → 0.585524, b → 0.126214, c → 0.101124, d → 0.004909},
  {a → 0.585624, b → 0.68239, c → 0.597183, d → 0.0978865}, {a → 0.43297, b → 0.901408, c → 0.075741, d → 0.453333}, {a → 0.125162, b → 0.713568, c → 0.555118, d → 0.706609},
  {a → 0.379786, b → 0.925074, c → 0.133633, d → 0.521008}, {a → 0.030091, b → 0.977612, c → 0.438356, d → 0.842142}, {a → 0.257523, b → 0.452874, c → 0.0303216, d → 0.742017},
  {a → 0.844547, b → 0.535482, c → 0.0529101, d → 0.154053}, {a → 0.372588, b → 0.778688, c → 0.560807, d → 0.347087}, {a → 0.875737, b → 0.482788, c → 0.457143, d → 0.0134456},
  {a → 0.969609, b → 0.199422, c → 0.0305939, d → 0.0173086}, {a → 0.116765, b → 0.468931, c → 0.257392, d → 0.473684}, {a → 0.0930721, b → 0.995659, c → 0.662857, d → 0.425968},
  {a → 0.456463, b → 0.889742, c → 0.0264966, d → 0.504784}, {a → 0.453764, b → 0.85495, c → 0.12766, d → 0.538054}, {a → 0.43297, b → 0.901408, c → 0.0202399, d → 0.472803},
  {a → 0.538438, b → 0.49852, c → 0.133005, d → 0.318766}, {a → 0.460962, b → 0.717687, c → 0.323276, d → 0.485343}, {a → 0.139858, b → 0.990172, c → 0.373965, d → 0.787585},
  {a → 0.335199, b → 0.814672, c → 0.55618, d → 0.495863}, {a → 0.204239, b → 0.616236, c → 0.0105526, d → 0.287879}, {a → 0.89893, b → 0.381081, c → 0.182116, d → 0.0750791},
  {a → 0.013396, b → 0.99991, c → 0.641414, d → 0.753799}, {a → 0.562431, b → 0.575163, c → 0.0949367, d → 0.228571}, {a → 0.13296, b → 0.956098, c → 0.357414, d → 0.652174},
  {a → 0.0927722, b → 0.818182, c → 0.402655, d → 0.470588}, {a → 0.223533, b → 0.744898, c → 0.107647, d → 0.390533}, {a → 0.403879, b → 0.908084, c → 0.120301, d → 0.588859},
  {a → 0.323103, b → 0.767152, c → 0.0136488, d → 0.0310559}, {a → 0.450265, b → 0.892895, c → 0.322314, d → 0.284038}, {a → 0.504249, b → 0.863558, c → 0.632479, d → 0.270329},
  {a → 0.286314, b → 0.821429, c → 0.114342, d → 0.707127}, {a → 0.122263, b → 0.388306, c → 0.113991, d → 0.132353}, {a → 0.271818, b → 0.342105, c → 0.0506757, d → 0.151899},
  {a → 0.306108, b → 0.460875, c → 0.459091, d → 0.0026795}, {a → 0.5, b → 0.5, c → 0.206897, d → 0.153992}, {a → 0.692592, b → 0.721329, c → 0.0496454, d → 0.293814},
  {a → 0.154654, b → 0.987969, c → 0.36005, d → 0.778279}, {a → 0.948216, b → 0.0350877, c → 0.0128517, d → 0.000743579}, {a → 0.204239, b → 0.616236, c → 0.00323861, d → 0.669231}}
```

Σχήμα ΄.2: Εκτέλεση προγράμματος για υπολογισμό 50 λύσεων στο σύστημα ανισοτήτων.

Παράρτημα Ζ΄

Κώδικας Mathematica για υπολογισμό ΜΕC με βάση τις υπολογιζόμενες τιμές του συστήματος ανισοτήτων.

Ζ΄.1 Κώδικας

```
In[*]:=
ri = {-0.8379310344827586, 0};
rj = {0, 0.4697590722783165};
ρj = {-0.6666666666666666, -0.2755969202216134};

(*Calculate midpoints of the sides of the triangle formed by the points*)
midpointAB = (ri + rj) / 2;
midpointBC = (rj + ρj) / 2;

(*Calculate the slopes of the sides AB and BC*)
slopeAB = (rj[[2]] - ri[[2]]) / (rj[[1]] - ri[[1]]);
slopeBC = (ρj[[2]] - rj[[2]]) / (ρj[[1]] - rj[[1]]);

(*Calculate the slopes of the perpendicular bisectors*)
perpSlopeAB = -1 / slopeAB;
perpSlopeBC = -1 / slopeBC;

(*Equations of the perpendicular bisectors*)
eqnPerpAB = y - midpointAB[[2]] == perpSlopeAB (x - midpointAB[[1]]);
eqnPerpBC = y - midpointBC[[2]] == perpSlopeBC (x - midpointBC[[1]]);

(*Solve for the intersection of the perpendicular bisectors,which gives the circumcenter*)
circumcenter = Solve[{eqnPerpAB, eqnPerpBC}, {x, y}];
circumcenter = {x, y} /. circumcenter[[1]];

(*Calculate the radius of the circumcircle*)
radius = Sqrt[(circumcenter[[1]] - ri[[1]])^2 + (circumcenter[[2]] - ri[[2]])^2];

(*Plot the points and the circumcircle*)
Graphics[{(*Draw the circumcircle*)Circle[circumcenter, radius], (*Draw the points*)Red, PointSize[Large], Point[ri], Green, PointSize[Large], Point[rj],
  Blue, PointSize[Large], Point[ρj], (*Label the points*)Text["ri", ri, {-1, -1}], Text["rj", rj, {1, 1}], Text["ρj", ρj, {-1, 1}], (*Label the circumcenter*)
  Text["Circumcenter", circumcenter, {1, 1}]}, Axes → True, AspectRatio → 1]
```

Σχήμα Ζ΄.1: Κώδικας mathematica για υπολογισμό ΜΕC με βάση τις υπολογιζόμενες τιμές του συστήματος ανισοτήτων.

Παράρτημα Η΄

Κώδικας Mathematica για έλεγχο αν υπάρχει λύση λύση στο σύστημα όπου ένα σημείο να βρίσκεται στο εσωτερικό του MEC

Η΄.1 Κώδικας

```
(*Define the points*)
ri = {-b, 0};
rj = {0, a};
ρj = {-c, -d};

(*Calculate midpoints of the sides of the triangle formed by the points*)
midpointAB = (ri + rj) / 2;
midpointBC = (rj + ρj) / 2;

(*Calculate the slopes of the sides AB and BC*)
slopeAB = (rj[[2]] - ri[[2]]) / (rj[[1]] - ri[[1]]);
slopeBC = (ρj[[2]] - rj[[2]]) / (ρj[[1]] - rj[[1]]);

(*Calculate the slopes of the perpendicular bisectors*)
perpSlopeAB = -1 / slopeAB;
perpSlopeBC = -1 / slopeBC;

(*Equations of the perpendicular bisectors*)
eqnPerpAB = y - midpointAB[[2]] == perpSlopeAB (x - midpointAB[[1]]);
eqnPerpBC = y - midpointBC[[2]] == perpSlopeBC (x - midpointBC[[1]]);

(*Solve for the intersection of the perpendicular bisectors,which gives the circumcenter*)
circumcenter = Solve[{eqnPerpAB, eqnPerpBC}, {x, y}];
circumcenter = {x, y} /. circumcenter[[1]];

(*Calculate the radius of the circumcircle*)
radius = Sqrt[(circumcenter[[1]] - ri[[1]]) ^2 + (circumcenter[[2]] - ri[[2]]) ^2];

(*Calculate the Euclidean distance*)
disRiMEC = EuclideanDistance[ri, circumcenter];
disRjMEC = EuclideanDistance[rj, circumcenter];
disPjMEC = EuclideanDistance[ρj, circumcenter];

(*Define the inequalities    , disRiMEC < radius, disRjMEC < radius, disPjMEC < radius*)
inequalities1 = {c^2+d^2-b*c+a*d>0, a^2+a*d+b*c>0, b^2-b*c-a*d>0, a^2+b^2≤1, b^2-2*b*c+c^2+d^2≤1, a^2+2*a*d+d^2+c^2≤1, a>0, b>0, c>0, d>0, a≤1, b≤1, c≤1, d≤1, disRiMEC < radius};
inequalities2 = {c^2+d^2-b*c+a*d>0, a^2+a*d+b*c>0, b^2-b*c-a*d>0, a^2+b^2≤1, b^2-2*b*c+c^2+d^2≤1, a^2+2*a*d+d^2+c^2≤1, a>0, b>0, c>0, d>0, a≤1, b≤1, c≤1, d≤1, disRjMEC < radius};
inequalities3 = {c^2+d^2-b*c+a*d>0, a^2+a*d+b*c>0, b^2-b*c-a*d>0, a^2+b^2≤1, b^2-2*b*c+c^2+d^2≤1, a^2+2*a*d+d^2+c^2≤1, a>0, b>0, c>0, d>0, a≤1, b≤1, c≤1, d≤1, disPjMEC < radius};

(*Use FindInstance to find a solution*)
solution1 = FindInstance[inequalities1, {a, b, c, d}, Reals, 1];
solution2 = FindInstance[inequalities2, {a, b, c, d}, Reals, 1];
solution3 = FindInstance[inequalities3, {a, b, c, d}, Reals, 1];
```

Σχήμα Η΄.1: Κώδικας mathematica για έλεγχο αν υπάρχει λύση λύση στο σύστημα όπου ένα σημείο να βρίσκεται στο εσωτερικό του MEC.

Bibliography

[1] S. Li, F. Meyer auf der Heide, and P. Podlipyan, "The impact of the gabriel subgraph of the visibility graph on the gathering of mobile autonomous robots," in *Algorithms for Sensor Systems: 12th International Symposium on Algorithms and Experiments for Wireless Sensor Networks, ALGOSENSORS 2016, Aarhus, Denmark, August 25-26, 2016, Revised Selected Papers 12*. Springer, 2017, pp. 62–79.

[2] S. Li, C. Markarian, F. Meyer auf der Heide, and P. Podlipyan, "A continuous strategy for collisionless gathering," in *Algorithms for Sensor Systems: 13th International Symposium on Algorithms and Experiments for Wireless Sensor Networks, ALGOSENSORS 2017, Vienna, Austria, September 7-8, 2017, Revised Selected Papers 13*. Springer, 2017, pp. 182–197.

[3] B. Kempkes, P. Kling, and F. Meyer auf der Heide, "Optimal and competitive runtime bounds for continuous, local gathering of mobile robots," in *Proceedings of the twenty-fourth annual ACM symposium on Parallelism in algorithms and architectures*, 2012, pp. 18–26.

[4] H. Ando, I. Suzuki, and M. Yamashita, "Formation and agreement problems for synchronous mobile robots with limited visibility," in *Proceedings of Tenth International Symposium on Intelligent Control*. IEEE, 1995, pp. 453–460.

[5] N. Gordon, I. A. Wagner, and A. M. Bruckstein, "Gathering multiple robotic a (ge) nts with limited sensing capabilities," in *Ant Colony Optimization and Swarm Intelligence: 4th International Workshop, ANTS 2004, Brussels, Belgium, September 5-8, 2004. Proceedings 4*. Springer, 2004, pp. 142–153.

[6] R. Cohen and D. Peleg, "Convergence properties of the gravitational algorithm in asynchronous robot systems," *SIAM Journal on Computing*, vol. 34, no. 6, pp. 1516–

1528, 2005.

[7] “Wikipedia.” [Online]. Available: https://www.wikipedia.org/

[8] “Gabriel graph.” [Online]. Available: https://www.passagesoftware.net/webhelp/Gabriel_Graph.htm

[9] J. Katajainen and O. Nevalainen, “Computing relative neighbourhood graphs in the plane,” *Pattern Recognition*, vol. 19, no. 3, pp. 221–228, 1986.

[10] G. T. Toussaint, “The relative neighbourhood graph of a finite planar set,” *Pattern recognition*, vol. 12, no. 4, pp. 261–268, 1980.

[11] J. W. Jaromczyk and G. T. Toussaint, “Relative neighborhood graphs and their relatives,” *Proceedings of the IEEE*, vol. 80, no. 9, pp. 1502–1517, 1992.

[12] “Planar graph.” [Online]. Available: https://www.geeksforgeeks.org/mathematics-planar-graphs-graph-coloring/

[13] “Planar graph.” [Online]. Available: https://transportgeography.org/contents/methods/graph-theory-definition-properties/planar-non-planar-graph/

[14] D. W. Matula and R. R. Sokal, “Properties of gabriel graphs relevant to geographic variation research and the clustering of points in the plane,” *Geographical analysis*, vol. 12, no. 3, pp. 205–222, 1980.