
# Local Algorithms for the Continuous Gathering Problem [1]

**Dissertation**

zur Erlangung des akademischen Grades
**Doktor der Naturwissenschaften (Dr. rer. nat.)**
an der Fakultät für Elektrotechnik, Informatik und Mathematik
der Universität Paderborn

vorgelegt von
Pavel Podlipyan

Paderborn, November 15, 2017

[1]Die Arbeit wurde teilweise unterstützt vom DFG-Sonderforschungsbereich 901 (On-The-Fly Computing) und der International Graduate School Dynamic Intelligent Systems.

**Betreuer:**

Prof. Dr. Friedhelm Meyer auf der Heide, Universität Paderborn

**Gutachter:**

Prof. Dr. Friedhelm Meyer auf der Heide, Universität Paderborn

Prof. Dr. Christian Scheideler, Universität Paderborn

**Acknowledgements**

This work would not have been possible without the support of many people to whom I would like to express my acknowledgment. First of all, I would like to thank Prof. Dr. Friedhelm Meyer auf der Heide for the great support, guidance and freedom he gave me. We had many long discussions resulting in ideas that form the basis of this thesis. I feel very lucky to have been invited to the International Graduate School of Dynamic Intelligent Systems in the summer of 2012. This would not have been possible without the assistance of Astrid Canisius and Prof. Dr. Eckhard Steffen.

I am deeply thankful to all professional colleagues for supporting my work. Special thanks goes to the members of the Algorithms and Complexity Group and in particular my office partners Peter Pietrzyk, Christine Markarian and Shouwei Li. I would like to thank Alexander Mäcker for his time invested into proofread of this thesis. I am also grateful to all the students I had the pleasure to work with. I would like to thank my Bachelor students Johannes Schaefer and Arne Kemper. It was a great pleasure that Johannes joined the Algorithms and Complexity Group as a doctoral student.

Besides that, I would like to express my acknowledgement to my family. It would not have ever been possible to get so far without my parents Olga and Yevgeni. I am also very thankful to my sister Julia and her family Sasha, Sofia and Lena for so much care. Finally, I am deeply grateful to Josefine for making me happy every day. Last but surely not least, I would like to thank Anne and Johannes as well as all Schwaneyer, especially the Allenkreuers: Josef, Jutta, Elisabeth, Katharina and Maria. You are the best!

**Pavel Podlipyan**

Paderborn, September 21, 2017

# Abstract

We consider a group of mobile robots in the Euclidean plane. Robots have a limited vision range and do not have a central control. Each robot acts by depending solely on local information. Many algorithmic problems arise in such a setting. In this work, we explore the continuous gathering problem. The goal is to know how fast the robots can gather in one not-predefined point in the continuous time model. In this model, each robot continuously observes its local neighborhood and adapts its speed and direction following a local rule. We present a class of algorithms, which we call the *contracting algorithms*, that perform gathering in time $O(nd)$, where $n$ is the number of robots and $d$ is the diameter of the initial configuration. We also present several contracting algorithms and analyze their efficiency. Upper and lower bounds on the time needed to gather all robots in one not-predefined point are given. Besides that, we investigate how the use of proximity subgraphs of visibility graphs influences the gathering processes. Simulations exhibit a severe difference in the behavior of robots using a Gabriel or Relative Neighborhood graphs as the visibility graph. While a lot of collisions occur during the gathering process, typically only one collision (the final one) takes place if robots use proximity graphs. We present a contracting algorithm which ensures that no collision occurs during the gathering process. This algorithm requires the robots to have some additional capabilities, such as memory and chirality.

# Zusammenfassung

Im Zentrum unserer Betrachtung steht eine Gruppe von mobilen Robotern auf euklidischer Ebene. Roboter haben begrenzte Sicht und keine zentrale Steuerung. Sie agieren ausschließlich auf Grundlage lokaler Informationen. In diesem Rahmen ergeben sich viele algorithmische Probleme. In der vorliegenden Arbeit widmen wir uns dem Problem der kontinuierlichen Versammlung. Wir wollen herausfinden, wie schnell sich Roboter innerhalb eines kontinuierlichen Zeitmodells, an einem zuvor nicht definierten Punkt, sammeln können. In einem solchen Modell beobachtet ein Roboter seine unmittelbare Nachbarschaft kontinuierlich, während er gleichzeitig Geschwindigkeit und Richtung nach lokalen Vorgaben anpasst. Diesbezüglich legen wir eine Klasse von Algorithmen vor, die wir kontrahierende Algorithmen nennen. Algorithmen dieser Klasse veranlassen Roboter sich in der Zeit von $O(nd)$ zu sammeln, wobei $n$ die Anzahl der Roboter und $d$ den Durchmesser der Anfangskonfiguration beschreiben. Ferner zeigen wir einige konkrete kontrahierende Algorithmen, die wir auf ihre Effizienz hin untersuchen. Wir setzen Obere und Untere Schranken hinsichtlich der benötigten Zeit für die Versammlung von allen Robotern in einem zuvor nicht definierten Punkt. Daneben untersuchen wir, inwiefern sog. Proximity Untergraphen des Sichtbarkeitsgraph den Sammlungsprozess beeinflussen. Simulationen zeigen einen deutlichen Unterschied im Verhalten von solchen Robotern, die den Gabriel- oder Relativen Nachbarschaftsgraph als Sichtbarkeitsgraph nutzen. Während viele der Zusammenstöße während des Sammlungsprozesses auftreten, lässt sich bei der Verwendung von Proximity Graphen typischerweise nur ein Zusammenstoß (endgültig) ausmachen. In dieser Arbeit

präsentieren wir eine kontrahierende Strategie, welche einen kollisionsfreien Sammlungsprozess gewährleistet. Diese Strategie erfordert, dass Roboter einige zusätzliche Fähigkeiten besitzen.

# Contents

# Chapter 1

# Introduction

Autonomous aerial and ground vehicles are gradually playing a major role in search and rescue, monitoring, surveillance and inspection operations. In the near future, scenarios in which a group of small scale autonomous robots inspect narrow indoor or outdoor environments might be a reality. Mobile robots would actively explore unknown environments while avoiding collisions and creating maps. In such scenarios, ground or low-flying aerial robots cannot rely on the information from any global positioning system. A fully autonomous operation of robots acting solely on local information involves a number of challenges on different levels of robot design. For autonomous mobile robots, one of the main problems in such a settings is to build a given formation out of an arbitrary initial configuration. These problems are known as robot formation problems. The availability of only local information makes formation problems particularly interesting.

The features that allow robots to work with information are their *capabilities*. Examples include unique identification numbers, memory, and a compass. Typically, the aim is to use as few capabilities as possible in order to solve a given formation problem. In this work, we consider robots with a limited viewing range. Such robots

are capable of locating the positions of other robots only within a certain fixed range. Each robot knows only the position of some of the robots. However, robots are still required to build a given formation out of an arbitrary initial disposition.

In this work, we consider formation problems on a high level of abstraction. Simple models of the robot and the environment ensure that correctness and efficiency can be proved. The robots are represented by the points on the Euclidean plane. These robots neither block each other on their path while moving nor obstruct the vision of each other. The very natural aim in every formation problem is to use as few capabilities as possible so that the robots remain as inexpensive as possible. Therefore, in our model robots have *no common compass*: i.e., they do not agree on the orientation of their local coordinate system. Robots are *anonymous*, which means that they do not have any unique identifiers. In Chapter 3, this assumption will be relaxed. In order to achieve collisionless gathering, robots become *luminous*, i.e. they have one bit of visible external memory. This will allow them to see whether their neighbors belong to one of two groups. Robots are *silent*, meaning that they do not communicate directly. Robots can measure the exact relative positions of their neighbors within their viewing range: for example, the distances to the visible robots and the angles between the rays to these robots. Yet, this information cannot be stored. In Chapter 2, we consider *oblivious* robots, which do not use the memory to store the past. The robots therefore, have to base their decisions only on the current observed relative positions of their visible neighbors. In Chapter 3, robots will gain memory in order to achieve collisionless gathering. Besides that, in Chapter 3 the robots are *chiral*: i.e., they all agree either on left- or right-hand orientation.

The formation problem considered in this work is the gathering problem. The group of $n$ robots must gather in one point which is not predefined. During the gathering process the robots need to agree on the gathering point by depending only on local information. For every robot, the set of visible robots is not fixed. We call the graph that has an edge between any pair of robots that can see each other is a *visibility graph*. In order to gather all robots at one not-predefined point, we only demand that the visibility graph is connected at the beginning.

One part of the puzzle is the robot model. The other part of the puzzle is the time and activation model. The algorithms that we present in this work inherit the common *Look-Compute-Move* (LCM) model due to [CP04]. As the name of this model suggests, it consists of three steps. When a robot is activated, it first observes the environment within its limited viewing range and determines the relative positions of its neighbors in the visibility graph. In the second step, the robot calculates the *target point*: i.e., the point towards which the robot will move. Finally, the robot moves towards the target point computed in the previous step. Depending on how often robots execute the three LCM steps we split the time models into discrete and continuous. In this thesis we consider the continuous time model.

The goal of this thesis is to examine how efficiently the robots with a limited viewing range solve the gathering problem in a continuous time model. Besides that, we will also examine whether it is possible to use local information to gather the robots without collisions. The gathering problem in the model that we consider becomes trivial if an unlimited viewing range is granted to the robots. The challenge is to gather the robots efficiently with local information.

Let us consider a start configuration of $n$ robots such that the visibility graph is connected. For such a configuration and a given algorithm, we are interested in several things. The first of these is the *correctness* of the algorithm. The algorithm is *correct* if it preserves the connectivity between the robots with respect to the visibility graph. Besides that, we are interested in the *quality* of our algorithms. The definition of a quality measure depends on the time and activation model. In discrete time models, the quality is commonly measured in a number of rounds. For the continuous time model, a more suitable measure is the travel time of the robot.

Thus, our quality measure will be as follows. We use the maximum of the total time traveled by the robots until they are gathered at one not-predefined point. The maximum is taken over $n$ robots. We refer to this quality measure as the (maximum) *traveled time*. The last thing, though not the least point of interest, is the property of the algorithms to gather robots without collisions. We say that two robots collide if they are at the same point at the same time. We dedicate the second half of this thesis to the *collisionless* property of gathering algorithms.

## 1.1 Organization of the Thesis

We first start with an overview of the related work. In Chapter 2, we give a formal description of the gathering problem. Then, we define the quality measure for gathering algorithms in a continuous time model and the class of contracting algorithms. For this class, we present an upper and lower bound on the time needed to gather all robots using a contracting algorithm. The rest of Chapter 2 is dedicated to the analysis of the various local continuous gathering algorithms.

In Chapter 3, we study the collisionless property of gathering algorithms. In the simulation of the Go-To-The-Relative-Center and Go-To-The-Gabriel-Center algorithms on the Euclidean plane, we observe that robots gather without collisions from random configurations. We show that for one dimension, the gathering with both algorithms is collisionless. For the two-dimensional case, we show that collision takes place only at the linear number of specific points.

Using this structural information and additional algorithmic extensions in Chapter 4, we develop the Safe-Go-To-The-Relative-Center algorithm. This algorithm is local, collisionless, and contracting. We also apply the Safe-Go-To-The-Relative-Center algorithm to the near gathering problem. In Chapter 5, we conclude the thesis and raise some open questions.

## 1.2 Related Work

In the literature, robot formation problems are categorized depending on the formation that must be reached. For example, there are *chain problems* [DKLM06, DKMS07, KM09, KM11], where the goal is to achieve line formation; or there are *circle formation problems* [CMN04, DK02, FPSV14, Kat05], where the goal is to position all robots on the circle. The *gathering problem* is a formation problem with the simplest formation that must be reached. The goal of the robots is to reach a common single point, e.g. [ASY95, FJM17, GWB04].

The *convergence problem* [CDF$^+$11a, CP04, Kat11, SY99] is a relaxed version of the gathering problem. As the name suggests, the difference in this case, is that the robots do not need to reach some common point, but to converge to it. In the literature, a

convergence problem commonly refers to a problem in which the goal is a single point formation. For the various formation problems, such relaxation is also possible [KM11].

The particular formation problem is defined according to the model of the robot, the model of the environment, and the time and activation model. For any formation problem, there are two questions: (1) Is there an algorithm that solves the particular formation problem with a given robot model? (2) And if there is one, what is the quality of this algorithm? A major work in the field of formation problems has been done to answer the first question. The theoretical analysis or simulations have been used in order to show that robots with a certain set of capabilities are able to reach the necessary formation. Less attention was given to the second question. There are far fewer statements about the quality of the algorithm.

The model of the robot, namely the set of capabilities the robot requires to solve the particular formation problem as well as the quality of the solution heavily depend on the environment model and the time and activation model. In order to investigate this dependence, in the last 20 years researchers focused on the gathering and convergence problems. The simplicity of the desired formation allows us to get rid of many unnecessary complications in the analysis. In what follows, we shortly describe the most frequently used models in gathering and convergence problems and some other formation problems.

The algorithm of the robot is subdivided into three logical steps: 'Look', where the robot gets the information about the environment; 'Compute', where the robot computes a target destination based on the obtained information; 'Move', where the robot moves toward the target destination. When the robot does not perform

any operation, it is said to be idle. These three steps form a *cycle*. The activation model is a schedule of the robot and the robot's timings of the steps within a cycle. There are two main models: *asynchronous* and *semi-synchronous*.

In the general asynchronous model [CFPS03, CP04], the activation schedules of the robots are completely independent of each other. Robots may become active at any time and the idle state may split the cycle at any point. The only thing we know about the cycles is that its duration is finite and the robot is activated infinitely often. Due to this asynchronous model, only the correctness and the termination of the algorithm are usually considered, but no runtime bounds are given. In some cases, problems are considered experimentally [BCJKF14, CDF$^+$11b]. In this experiments authors were concentrating on the correctness and the quality of the algorithms.

In the semi-synchronous model [DP09, SDY09, SY99], activation schedules of the robots are synchronized and consist of rounds. One or more robots in each round are activated at the same time. In such a schedule, the information obtained by the robot is more consistent. No robot will ever observe the other robot during the motion. However, at each round, it is unknown which robots are activated. If all robots are activated in every round, then we get a *synchronous* activation model [ASY95, DKLM06, DKL$^+$11]. In this model, all robots perform their algorithm at the same time in sequential rounds. For the comprehensive survey of various discrete time models for the gathering and other related problems we refer the reader to the book [FPS12] by Flocchini, Prencipe and Santoro.

Apart from these three discrete time models mentioned above, there is at least one more. The *continuous* time model was proposed for the first time in [GWB04]. The authors of [KKM12] spec-

ify that the continuous time model may be viewed as the extreme instance of the discrete classical Look-Compute-Move model. Assuming a speed limit of one, the continuous time model arises from the discrete LCM model by fixing the maximum distance traveled per round by $\delta$ and letting $\delta \to 0$.

After the robot is activated, it performs a look operation and obtains an information about the surrounding environment. Depending on the vision range of the robot, there is an *unlimited* or a *limited visibility* model. With unlimited visibility, every robot obtains information about the whole environment. With limited visibility, just partial information about the surrounding environment is obtained by a robot. Usually, limited vision model is represented by the unit disk graph as in [ASY95]. Each robot can only see a certain distance away from its current position. Unlimited visibility as in [SY99] is the most commonly used model.

Cohen and Peleg show that if robots with unlimited visibility move towards the center of gravity, then they converge around the single point in a highly asynchronous model [CP04]. They also provide runtime bounds for the different activation models. Several of these bounds have been later improved in [CDF+11a]. For a more restricted asynchronous model, Katreniak present an algorithm that solves the convergence problem with limited visibility [Kat11]. The same algorithm solves the convergence problem with unlimited visibility in an asynchronous model, without the need to detect whether there is more than one robot at a given point (*multiplicity detection*). For robots that have a multiplicity detection, Cieliebak et al. in [CFPS03] propose an asynchronous algorithm that solves the gathering problem with unlimited visibility. The authors of [CDSN17] consider the variant of the gathering problem in an asynchronous time model in which the robots need to

gather at only some predetermined points in the plane. The authors of [ASY95] show that the robots with limited visibility acting in a synchronous discrete time model gather at one not-predefined point using a minimum enclosing circle algorithm. It has been shown in [DKL+11] that robots described in [ASY95] gather in $\Theta(n^2)$ rounds. A special semi-synchronous model is considered in [DKM10]. A round in this model consists of a movement of all robots in random order. For this model, the authors of [DKM10] present an algorithm that achieves gathering in $O(n^2)$ rounds in expectation.

In most of the works mentioned above the robots are placed on a Euclidean plane. The gathering and other formation problems on the grid are considered in [ACF+16, DSKN12, LM14]. In [CLFJM16], the gathering problem on the grid, in a synchronous time model, is solved by very simple robots in a linear number of rounds. The authors of [CLFJM16] use a pipeline technique similar to [KM09]. It is commonly assumed that robots are oblivious. They do not use the memory to store the past. However, in [CLFJM16], robots are able to remember a fixed number of steps. In [FJM17], a similar gathering problem on the grid is solved in a quadratic number of rounds with oblivious robots.

A common assumption in the models with unlimited visibility is an absence of unique identification – the robots are *anonymous*. Unique IDs or agreement on local coordinate system (i.e., a *compass*) together with unlimited visibility make the gathering problem trivial. It is shown in [DP09] that anonymous robots without a common compass with unlimited visibility gather in semi-synchronous models if and only if $n$ is odd. The various aspects of the models with a compass are studied in [IKIW07, KTI+07, SDY06, SDY09].

Independent of the visibility model, robots can either be trans-

parent or not. Often, robots are viewed as points and thus their visibility is *unobstructed*. The opposite assumption is made in [LFC+17]. It is shown that the *mutual visibility* problem can be solved without collisions by non-transparent, *luminous* robots with unlimited visibility. The aim in the mutual visibility problem is to reach a configuration in which each robot can see the rest (i.e., all robots are on the boundary of the convex polygon). Robots with lights or luminous robots are initially suggested by Peleg [Pel05]. Robots are called luminous if they have an external light with colors chosen from a fixed set. The light is visible to other robots depending on the visibility model.

Two point-shaped robots have a collision if they have the same position at the same time. The collisionless property becomes more important if the robots gain extent. For the first time, the Gathering problem for robots with an extent is considered in [CGP06], where the authors presented a solution for three and four robots by exhaustively considering all possible cases. Besides that, in [CGP06], the gathering itself is redefined, since robots with extent are not allowed to occupy the same position. Instead, gathering means forming a configuration for which the union of all discs representing the robots is connected. For the same model as in [CGP06], a general solution for the number of robots greater than four is presented by the authors of [AGM13] with an additional assumption of chirality. If robots agree on the left- and right-hand orientation, they are called *chiral*. Gathering of small groups of robots with extent in an asynchronous time model is also considered in [HPT14]. Commonly, there is no direct communication between the robots – they are *silent*. The collisionless motion in the group of mobile robots with an asynchronous communication is considered in [YDIW07].

The gathering of robots with an extent in an synchronous time model is defined differently in [CDF+11b]. The goal of the robots is to gather without touching, in such a way that disks representing the robots do not intersect. Robots gather around a predefined point that is already known by every robot. It is shown that robots gather in $O(nR)$ rounds, where $n$ is the number of robots and $R$ is the distance from the gathering point to the farthest robot. This bound has been improved in [SBMM17] by the tight linear bound.

Unlike in [CGP06] and [AGM13], the robots in [CDF+11b] have limited visibility. The robots with limited visibility but without extend are considered in [PPV15]. This work considers the near gathering problem, in which the aim of the robots is to get close enough to be in the vision range of each other without collisions and switch in this way to the setting in which the algorithms that utilize a global vision will work. Furthermore, the robots in [PPV15] are equipped with a compass so that they have a common coordinate system.

The continuous time model for the Gathering problem was first used in [GWB04]. Unlike in common discrete time models, robots do not act in rounds, but rather continuously adjust directions of movement towards calculated target points, while moving with constant velocity. As a result, runtime cannot be defined as the number of rounds. Instead, it is defined as the time that robots need to reach formation needed. The authors of [KKM12] analyze the runtime of the algorithm introduced in [GWB04], and show that the time needed for gathering is bounded by $O(\min\{n, OPT\, log(OPT)\})$, where $n$ is the number of robots and $OPT$ is the runtime of the optimal algorithms with unlimited visibility. There are no bounds for the minimum enclosing circle algorithm proposed in [ASY95] within the continuous time model.

## 1.3 Bibliography Note

Many of the results in this thesis have already been published as a preliminary version in conference proceedings. The analysis of the gathering problem and the collisionless property have been presented in [LMP16].

*Shouwei Li, Friedhelm Meyer auf der Heide, Pavel Podlipyan: The impact of the Gabriel subgraph of the visibility graph on the gathering of mobile autonomous robots. In: Algorithms for Sensor Systems, Proceedings of the 12th International Symposium on Algorithms and Experiments for Wireless Sensor Networks (ALGOSENSORS 2016), Springer-Verlag, 25 - 26 August 2016 (LNCS)*

An extended version of this paper has been invited for submission to a special issue of the *Theoretical Computer Science Journal*. The first local collisionless gathering algorithm was presented in [LMMP17].

*Shouwei Li, Christine Markarian, Friedhelm Meyer auf der Heide, Pavel Podlipyan: A Continuous Strategy for Collisionless Gathering. To appear in: Algorithms for Sensor Systems, Proceedings of the 13th International Symposium on Algorithms and Experiments for Wireless Sensor Networks (ALGOSENSORS 2017), Springer-Verlag, 7 - 8 September 2017 (LNCS).*

Some results from Chapter 2 and Chapter 4 have not been published yet.

# Chapter 2

# Gathering in the Continuous Time Model

In this chapter we will propose simple criterion that defines a class of algorithms that perform gathering on the Euclidean plane at time $O(nd)$, where $d$ is the diameter of the initial configuration. Next, for a given class of algorithms we consider upper and lower bound of maximum traveled time. Then we consider robots with limited visibility and for such robot model we present and analyze several examples of contracting algorithms.

## 2.1 Problem Description and Notation

We are given a set $R = \{r_1, \ldots, r_n\}$ of $n$ autonomous mobile robots. We denote by $r_i(t) \in \mathbb{R}^2$ the position of robot $r_i$ at time $t$. Robots agree on the unit distance. Each robot has its own local coordinate system. Robots are oblivious, meaning that they act depending only on the information about the current point of time. They are anonymous, meaning that they do not have IDs, and are also silent, meaning that they do not communicate. However, the robots can determine the positions of other robots within a viewing range 1 from its own position.

The given robot model is formalized by the unit disk graph. The Euclidean distance between two robots $r_i$ and $r_j$ at time $t$ is represented by $|r_i(t), r_j(t)|$. The two robots are open unit disk graph (open UDG) neighbors at time $t$ if $|r_i(t), r_j(t)| < 1$. Further, to save the space "open" will be dropped in the description of open unit disk graph, neighbors etc.

The set of robots that consists of the robot $r_i$ itself and all its UDG neighbors at time $t$ is called the UDG neighborhood of $r_i$ and denoted by $UDG_t(r_i)$. The UDG defined on all robots at some point in time $t$ is denoted by $UDG_t(R) = (R, E_t)$, where $(r_i, r_j) \in E_t$ iff $|r_i(t), r_j(t)| < 1$. We skip $t$ in the notation of the UDG neighborhood and UDG unless it needs to be mentioned explicitly.

The disposition of robots at some point in time $t$ on the plane is called a *configuration*. The disposition of robots at time $t = 0$ is called the *initial configuration*. Initial configurations are arbitrary except that all robots have distinct positions and if robots have limited visibility, then UDG over all robots at time $t = 0$ ($UDG_0(R)$) is connected. The goal is to gather all robots at one point that is not predefined.

We consider the continuous time model, first introduced by Gordon et al. in [GWB04] and later studied by Kempkes et al. in [KKM12]. The velocity of the robot depends solely on the relative positions of neighboring robots at the current point of time. It may change in a non-continuous manner since robots measure the relative positions of their neighbors without delay and instantly adjust their own movement with respect to the measurements. The maximum speed of the robot is assumed to be 1.

## 2.2 Contracting Algorithms

In this section, for the continuous time model we introduce a class of algorithms that perform gathering on the Euclidean plane at time $O(nd)$, where $d$ is the diameter of the initial configuration.

Let us first define the progress measure. Let $H_t(R) \subset \mathbb{R}^2$ be the closed convex hull around the positions of all robots at time $t$. We are particularly interested in robots that are corners of a convex hull. Namely, we consider the set of robots $CH_t(R) = \{c_i \in H_t(R) : \alpha_i(t) \in [0, \pi), i \in [1, k], k \leq n\}$, where $n$ is the total number of robots and $k$ is the number of robots that belong to the boundary of the convex hull and have the internal angle $\alpha_i(t) \in [0, \pi)$. We refer to $CH_t(R)$ as the *corner set* of the convex hull $H_t(R)$. Unless explicitly needed, we skip $t$ in the notation of the convex hull and corner set.

**Definition 1** (Contracting algorithm)**.** *In the continuous time model, a Gathering algorithm for $n$ robots on the Euclidean plane is contracting if for every time $t$ such that cardinality of $CH_t(R)$ is strictly greater than* 1*, every robot from $CH_t(R)$ moves with speed 1 in the direction that points into $H_t(R)$.*

If an algorithm is contracting, then we can bound the speed with which the length of the convex hull is decreasing. Let $l(t)$ be the length of the convex hull boundary at time $t$. Using the corner set we express the length as follows:

$$l(t) = \sum_{i=1}^{k} \left| c_i, c_{(i \bmod k)+1} \right|, \tag{2.1}$$

where $k$ is the number of robots in the corner set. Let $l'(t)$ be the speed with which the length $l(t)$ of the convex hull boundary changes. The length of the convex hull boundary will be the

progress measure for contracting algorithms. Before bound the $l'(t)$ from below we need to state one useful lemma.

**Lemma 1.** *For* $0 \leq \vartheta \leq 1$ *and* $0 \leq \alpha \leq \pi$ *it holds that*

$$\cos(\alpha\vartheta) + \cos(\alpha(1-\vartheta)) \geq \frac{2(\alpha-\pi)^2}{\pi^2}. \tag{2.2}$$

*Proof.* First we would like to note that the left part of inequality 2.2 is symmetric with respect to $\vartheta$. Thus, we only need to consider $0 \leq \vartheta \leq 1/2$. Let us get rid of $\vartheta$ by bounding the left part of inequality 2.2 from below as follows:

$$\cos(\alpha\vartheta) + \cos(\alpha(1-\vartheta)) \geq \cos(\alpha) + 1. \tag{2.3}$$

We use the interval method. The equation

$$\cos(\alpha) + 1 - (\cos(\alpha\vartheta) + \cos(\alpha(1-\vartheta))) = 0 \tag{2.4}$$

has the following roots: $\alpha_1 = 0, \alpha_2 = \pi$ if $0 < \vartheta \leq 1/2$ and $0 \leq \alpha \leq \pi$ if $\vartheta = 0$. It implies that the sign of the function

$$f(\alpha, \vartheta) = \cos(\alpha) + 1 - (\cos(\alpha\vartheta) + \cos(\alpha(1-\vartheta))) \tag{2.5}$$

does not change on the interval $0 \leq \alpha \leq \pi$ for any $0 < \vartheta \leq 1/2$. The sign is negative, since $f(\pi/2, 1/4) < -1/4$. If $\vartheta = 0$, then inequality 2.3 becomes an equality. It is left to show that

$$\frac{2(\alpha-\pi)^2}{\pi^2} \leq \cos(\alpha) + 1. \tag{2.6}$$

We use the same method. Roots on interval $0 \leq \alpha \leq \pi$ are $\alpha_1 = 0, \alpha_2 = \pi$ and the according sign is negative, e.g. for $\alpha = \pi/2$

$$\frac{2(\frac{\pi}{2}-\pi)^2}{\pi^2} - \cos\left(\frac{\pi}{2}\right) + 1 = -\frac{1}{2}. \tag{2.7}$$

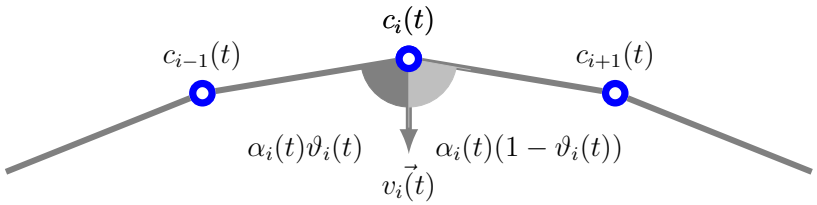



Figure 2.1: Velocity $\vec{v_i}(t)$ of robot $c_i$.

Due to this we can conclude that inequality 2.2 holds for $0 \leq \vartheta \leq 1$ and $0 \leq \alpha \leq \pi$. □

Now we are ready to bound from below the speed $l'(t)$ with which the length $l(t)$ of the convex hull boundary changes.

**Lemma 2.** *If a group of $n$ robots executes a contracting Gathering algorithm and the robots are not yet gathered (i.e., the cardinality of $CH_t(R)$ is strictly greater than 1), then the length $l(t)$ of the convex hull boundary at any point in time $t$ decreases with speed $l'(t) \geq \frac{8}{n}$.*

*Proof.* By definition of the contracting algorithm each robot $c_i \in CH(R)$ at time $t$ moves with speed 1 in the direction of $H(R)$. We do not know how velocity $\vec{v_i}(t)$ divides the internal angle $\alpha_i(t)$. This is illustrated in Figure 2.1. Due to this we introduce parameter $\vartheta_i(t) \in [0, 1]$ which defines two angles between the velocity vector $\vec{v_i}(t)$ and edges of the corner hull adjacent to robot $c_i$. Using parameter $\vartheta_i(t)$, the internal angle $\alpha_i(t)$ is given by

$$\alpha_i(t) = \alpha_i(t)\vartheta_i(t) + \alpha_i(t)(1 - \vartheta_i(t)). \tag{2.8}$$

Note that a change of $\vec{v_i}(t)$ causes just a change of parameter $\vartheta_i(t) \in [0, 1]$ within a given range.

Now we consider a line segment between $c_i$ and $c_{i+1}$. We call this line segment the edge of the corner set. The length of this edge is given by $d_{i,i+1}(t) = |c_i(t), c_{i+1}(t)|$. The velocity $\vec{d'}_{i,i+1}(t)$,

with which the length is changing is represented by $\vec{d'}_{i,i+1}(t) = \vec{P}v_i(t) + \vec{P}v_{i+1}(t)$, where $\vec{P}v_i(t)$, $\vec{P}v_{i+1}(t)$ are the orthogonal projection of velocity of robots $c_i$ and $c_{i+1}$ onto the line between these robots. Projections as well as $\vec{d'}_{i,i+1}(t)$ can be represented by scalar velocities $d'_{i,i+1}(t), Pv_i(t), Pv_{i+1}(t)$, where the sign of the scalar will represent whether the corresponding component of speed increases or decreases length $d_{i,i+1}(t)$. Further, in order to simplify the notation we skip time $t$. The value of $Pv_i$ depends on $\alpha_i(1-\vartheta_i)$ and can be represented as follows:

$$Pv_i(\alpha_i(1-\vartheta_i)) = \begin{cases} \cos(\alpha_i(1-\vartheta_i)) & \text{if } \alpha_i(1-\vartheta_i) \leq \pi/2; \\ -\cos(\alpha_i(1-\vartheta_i)) & \text{if } \alpha_i(1-\vartheta_i) > \pi/2. \end{cases} \tag{2.9}$$

In the same way we represent $Pv_{i+1}$, which depends on $\alpha_{i+1}\vartheta_{i+1}$. The scalar velocity with which the length of the edge between $c_i$ and $c_{i+1}$ is changing can now be expressed as follows:

$$d'_{i,i+1}(\alpha_i(1-\vartheta_i), \alpha_{i+1}\vartheta_{i+1}) = Pv_i(\alpha_i(1-\vartheta_i)) + Pv_{i+1}(\alpha_{i+1}\vartheta_{i+1}). \tag{2.10}$$

The overall speed with which the length of the corner hull is changing at time $t$ is represented by the sum of scalar velocities of each edge of the corner set

$$l' = \sum_{i=1}^{k} d'_{i,j}(\alpha_i(1-\vartheta_i), \alpha_j\vartheta_j) = \sum_{i=1}^{k} \left(Pv_i(\alpha_i(1-\vartheta_i)) + Pv_j(\alpha_j\vartheta_j)\right), \tag{2.11}$$

where $j = (i \bmod k)+1$. Although, instead of summing up over all edges of the corner hull we can sum up over all robots by rearranging components in $l'$ sum. For each robot $c_i$ we define the scalar

$l'_i(\alpha_i, \vartheta_i)$, which consist of two components, namely $l'_i(\alpha_i, \vartheta_i) = Pv_i(\alpha_i(1-\vartheta_i)) + Pv_i(\alpha_i\vartheta_i)$. Then the overall speed is represented by

$$l' = \sum_{i=1}^{k} l'_i(\alpha_i, \vartheta_i) = \sum_{i=1}^{k} \cos(\alpha_i\vartheta_i) + \cos(\alpha_i(1-\vartheta_i)). \tag{2.12}$$

From the useful Lemma 1 we know that the function $l'_i(\alpha_i, \vartheta_i)$ is lower bounded by ${}^{2(\alpha_i - \pi)^2}/_{\pi^2}, \forall\alpha_i \in [0, \pi), \forall\vartheta_i \in [0, 1]$. Using this fact, we can calculate the lower bound of speed with which the length of corner hull decreases, namely

$$l'(t) = \sum_{i=1}^{k} l'_i(\alpha_i, \vartheta_i) \geq \frac{2}{\pi^2} \sum_{i=1}^{k} (\alpha_i - \pi)^2; \tag{2.13}$$

To bound the obtained sum we use the sum of the square's inequality:

$$\sum_{i=1}^{i=k} {a_i}^2 \geq \frac{1}{k} \left( \sum_{i=1}^{k} a_i \right)^2. \tag{2.14}$$

It follows straight forward that

$$l'(t) \geq \frac{2}{k\pi^2} \left( \sum_{i=1}^{k} (\alpha_i - \pi) \right)^2. \tag{2.15}$$

It is well known that the sum of internal angles of a convex polygon is $\pi(k-2)$, where $k$ is the number of the polygon's vertices. Thus

$$l'(t) \geq \frac{2}{k\pi^2} \left( \left( \sum_{i=1}^{k} \alpha_i \right) - k\pi \right)^2, \tag{2.16}$$

$$l'(t) \geq \frac{2}{k\pi^2} \left( \pi(k-2) - k\pi \right)^2, \tag{2.17}$$

$$l'(t) \geq \frac{8}{k}. \tag{2.18}$$

As $k \leq n$, $l'(t) \geq \frac{8}{n}$. □

It is well known that the length of the convex hull boundary is $O(d)$, where $d$ is the diameter of the initial configuration. Hence, the theorem below follows directly.

**Theorem 1.** *Every contracting Gathering algorithm solves the Gathering problem in time $O(nd)$, where $d$ is the diameter of an initial configuration.*

For the robots with limited visibility by simple induction the length of the convex hull boundary is upper bounded as follows.

**Lemma 3.** *If we are given a connected unit disk graph with $n$ robots, then the length $l$ of the convex hull boundary is not greater than $2(n-1)$, where $n$ is a number of robots.*

*Proof.* We will show that each additional robot can increase the length of the convex hull boundary $l$ at most by 2. First, assume we have only one robot and $n = 1$, then the length of the convex hull boundary $l \leq 2(1-1) = 0$. If we want to increase the length by adding one more robot we need to place it at a maximal possible distance 1 away from the first one. In this case for $n = 2$ with two robots $l \leq 2(2-1) = 2$, since the convex hull boundary consists of two edges of length 1.

Assume that our inequality $l \leq 2(n-1)$ holds for an arbitrary number of robots $n$. If we want to increase the length of the convex hull boundary by adding one more robot $r$ to the existing $n$ robots, then to preserve connectivity we need to place it outside the existing convex hull $H(R)$ at the most a distance 1 away from one of the robots $c_i$ that belongs to the hull.

Let $(c_{p_1}, r)$ and $(r, c_{p_2})$, where $p_1, p_2 \in [1, n]$ are new edges of the corner hull around $n+1$ robots. Let $d = |c_{p_1}, r| + |r, c_{p_2}|$ be the length of these edges. When the number of robots was $k$, the length of corner hull between $c_{p_1}$, $c_i$ was $l(c_{p_1}, c_i) \geq |c_{p_1} c_i|$, and between $c_i$, $c_{p_2}$ the length was $l(c_i, c_{p_2}) \geq |c_i c_{p_2}|$. Now, using triangle inequality we can bound the length of new edges

$$|c_{p_1} r| \leq 1 + |c_{p_1} c_i| \leq 1 + l(c_{p_1} c_i), \tag{2.19}$$

$$|r c_{p_2}| \leq 1 + |c_i c_{p_2}| \leq 1 + l(c_i c_{p_2}). \tag{2.20}$$

By summing up inequalities we get the desired result

$$d = |c_{i-p_1} r| + |r c_{i+p_2}| \leq l(c_{i-1} c_i) + l(c_i c_{i+1}) + 2. \tag{2.21}$$

The difference in length between the convex hull boundaries of $n$ and $n+1$ robots is at most 2. □

For the robots with limited visibility, Theorem 1 and Lemma 3 yield the following.

**Corollary 1.** *If an initial configuration is a connected unit disk graph, then a contracting Gathering algorithm solves the Gathering problem in time $O(n^2)$.*

### 2.2.1 The Worst Contracting Algorithm

A continuous algorithm is contracting if at any point of time $t$ (unless robots are already gathered), every corner robot on the convex hull boundary moves with speed 1 inside the (closed) convex hull. Next we consider the worst initial configuration and according worst contracting algorithm that solves the gathering problem from

a given configuration in time $\Omega(nd)$, where $d$ is the diameter of an initial configuration.

**Worst algorithm:** Assume there is an initial configuration where all robots are corners of the convex hull boundary. Moreover, the boundary of the convex hull is a regular polygon. The target point of each robot is a left neighbor on the hull boundary with respect to the center of the regular polygon. Each robot moves with speed 1 towards its target point: i.e., left neighbor on the hull boundary. We call this algorithm the *Implicit-Go-To-The-Left* algorithm, since we do not specify how the robots in this algorithm know which neighbor is left one. Note that such Implicit-Go-To-The-Left algorithm is contracting. An instance with $n = 6$ robots is depicted in Figure 2.2.

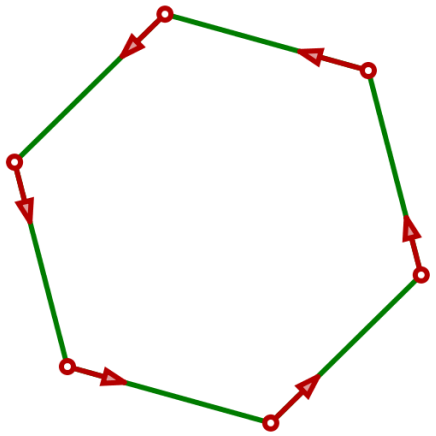


Figure 2.2: An instance of $n = 6$ robots that execute the Implicit-Go-To-The-Left algorithm.

The given algorithm together with a regular polygon as the initial configuration is better known as a symmetric $n$-bug[1] problem [Ber59, Nah07]. Using the technique from Lemma 2 we can bound

[1]Mice, dogs, etc.

from above the speed $l'(t)$ with which the length $l(t)$ of the convex hull boundary changes.

**Lemma 4.** *In a symmetric n-bug problem the length $l(t)$ of the convex hull boundary decreases with speed $l'(t) \leq \frac{2\pi^2}{n}$.*

*Proof.* Let us take a look at Equation 2.12 from Lemma 2:

$$l' = \sum_{i=1}^{k} l'_i(\alpha_i, \vartheta_i) = \sum_{i=1}^{k} \cos(\alpha_i \vartheta_i) + \cos(\alpha_i(1 - \vartheta_i)). \tag{2.22}$$

This equation holds for any contracting strategy. For the symmetric $n$-bug problem $k = n$, $\vartheta_i = 0$ and $\alpha_i = \frac{\pi(n-2)}{n}$ for all $i$. Thus,

$$l' = n\left(1 + \cos\left(\frac{\pi(n-2)}{n}\right)\right) = n\left(1 - \cos\left(\frac{2}{n}\pi\right)\right). \tag{2.23}$$

Next we use the fact that $cos(\alpha) \geq 1 - \frac{1}{2}\alpha^2$ for $\alpha \geq 0$:

$$l' \leq n\left(1 - \left(1 - \frac{1}{2}\left(\frac{2}{n}\pi\right)^2\right)\right) = \frac{2\pi^2}{n}. \tag{2.24}$$

□

It is well known that the length of the convex hull boundary is $\Omega(d)$, where $d$ is the diameter of the initial configuration. According to this fact and Lemma 4, in $n$-bug problems robots gather in time $\Omega(nd)$. If we use robots with limited visibility, then the next statement follows directly from Lemma 4 and Lemma 2.

**Corollary 2.** *If in a symmetric n-bug problem the initial configuration is a connected unit disk graph, then robots gather in time $t$, with $\frac{n^2}{2\pi^2} \leq t \leq \frac{n^2}{8}$.*

### 2.2.2 The Best Contracting Algorithm

Next we consider the best contracting algorithm, such that robots gather in time $O(d)$, where $d$ is the diameter of the initial configuration.

**Best algorithm:** Let us consider robots that are corners of the convex hull boundary, i.e. corner set $CH(R)$. Each robot moves with speed 1 on the inner angle bisector. The other robots stay idle. We call this algorithm the *Implicit-Go-On-Bisector* algorithm, since it does not specify how the robots get the information about their neighbors on the boundary of the convex hull. Such an Implicit-Go-On-Bisector algorithm is clearly contracting. An instance of $n = 11$ robots that execute the Implicit-Go-On-Bisector algorithm is depicted in Figure 2.3.

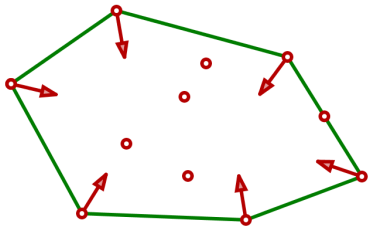


Figure 2.3: An instance of $n = 11$ robots that execute the Implicit-Go-On-Bisector algorithm.

Using the technique from Lemma 2 again, we can bound from below the speed $l'(t)$ with which the length $l(t)$ of the convex hull boundary changes if robots use the Implicit-Go-On-Bisector algorithm.

**Lemma 5.** *If robots use the Implicit-Go-On-Bisector algorithm, then the length $l(t)$ of the convex hull boundary decreases with speed $l'(t) \geq 4$.*

*Proof.* Let us take a look at the Equation 2.12 from Lemma 2 again:

$$l' = \sum_{i=1}^{k} l'_i(\alpha_i, \vartheta_i) = \sum_{i=1}^{k} \cos(\alpha_i \vartheta_i) + \cos(\alpha_i(1 - \vartheta_i)). \tag{2.25}$$

This equation holds for any contracting strategy. If robots use the Implicit-Go-On-Bisector algorithm, then $\vartheta_i = 1/2$ for all $i$. Thus,

$$l' = 2 \sum_{i=1}^{k} \cos\left(\frac{1}{2}\alpha_i\right). \tag{2.26}$$

Next we use the fact that $cos\left(\frac{\alpha}{2}\right) \geq 1 - \frac{\alpha}{2}$ for $\alpha \geq 0$:

$$l' \geq 2 \sum_{i=1}^{k} \left(1 - \frac{\alpha_i}{\pi}\right), \tag{2.27}$$

$$l' \geq 2k - \frac{2}{\pi} \sum_{i=1}^{k} \alpha_i. \tag{2.28}$$

It is well known that the sum of internal angles of a convex polygon is $\pi(k-2)$, where $k$ is the number of the polygon's vertices. Therefore it holds that $l' \geq 4$. □

It is well known that the length of the convex hull boundary is $O(d)$, where $d$ is the diameter of the initial configuration. According to this fact and Lemma 5, robots that use Implicit-Go-On-Bisector algorithms gather in time $O(d)$ from any initial configuration. If we use robots with limited visibility, then the next statement follows directly from Lemma 5 and Lemma 2.

**Corollary 3.** *If the initial configuration is a connected unit disk graph, then robots gather using the Implicit-Go-On-Bisector algorithm in time $O(n)$.*

## 2.3 Examples of Local Contracting Algorithms

In this section we consider robots with limited visibility and without any implicit information. The input information of each robot consists of the relative positions of other robots in its viewing range. Next we consider several gathering algorithms that use local information only and show that they are contracting.

### 2.3.1 Go-On-Bisector Algorithm

For robots with limited visibility the Go-On-Bisector algorithm proposed by Gordon et al. in [GWB04] and later studied by Kempkes et al. in [KKM12] is used.

**Go-On-Bisector algorithm:** Each robot $r \in R$ at every point in time $t$ observes the relative positions of all robots within the viewing range 1. Then the robot computes the local convex hull $LH_t(r)$ of these positions including itself. Depending on the disposition, the robot performs the following actions.

- If $r$ is strictly inside the convex hull, it does not move.
- If $r$ is on the edge of the local convex hull boundary, then it moves with this line, maintaining the ratio of distances between its two neighbors on the border.
- If $r$ is a corner of $LH_t(r)$, then it moves with speed 1 on the angle bisector towards the inner angle of $LH_t(r)$. In case $r$ has only 1 neighbor, the inner angle is zero and $r$ just moves towards its only neighbor.

**Proposition 1.** *The Go-On-Bisector algorithm is contracting.*

First of all, if robot $r$ is a corner robot of the convex hull $H_t(R)$, then it is also a corner robot of its own local convex hull $LH_t(r)$. Moreover, such a corner robot $r$ moves with speed 1 according to the Go-On-Bisector algorithm inside $H_t(R)$, since

$$\bigcup_{r \in R} LH_t(r) \subset H_t(R). \tag{2.29}$$

Therefore we can conclude that the Go-On-Bisector algorithm is contracting. However, our approach from Lemma 5 does not work with the Go-On-Bisector algorithm, since for any corner robot $r$ of $H(R)$ the bisector of the inner angle in $LH(r)$ does not necessarily coincide with the bisector of the inner angle in $H(R)$.

Kempkes, Kling and Meyer auf der Heide in [KKM12] show that if robots with limited visibility use the Go-On-Bisector algorithm, then they gather in time $O(\min\{n, OPT\ log(OPT)\})$, where $n$ is the number of robots and $OPT$ is the runtime of the optimal algorithm with unlimited visibility.

Besides the Go-On-Bisector there are several other algorithms that are well analyzed in the discrete time models, but there is no quality or correctness analysis in the continuous time model: e.g., the Go-To-The-Gravity-Center algorithm, due to Cohen et al. in [CP04], or the Go-To-The-Center algorithm proposed in [ASY95] within the continuous time model.

### 2.3.2 Go-To-The-Gravity-Center Algorithm

Let us consider robots with limited visibility in the continuous time model that execute the following algorithm:

**Go-To-The-Gravity-Center (GTGrC) algorithm:** Each robot $r \in R$, at every point in time $t$, observes relative positions of all robots

within viewing range 1. Then each robot computes its target point $T(r)$, which is the center of the gravity (average position) of all these positions including itself. Depending on disposition, the robot performs the following actions.

- If $r$ is not at the target point $T(r)$ yet, then the robot moves towards it with speed 1.
- If $r$ is already at the target point $T(r)$, then it follows the motion of the target point.

Note that the position of the target point might change in a discontinuous manner: for example, when a new robot appears in the viewing range of $r$.

**Proposition 2.** *The Go-To-The-Gravity-Center algorithm is contracting.*

*Proof.* Let us consider corner set $CH(R)$ of the convex hull $H(R)$. It is well known that the center of the gravity – i.e., the centroid or arithmetic mean of the convex polygon always lies inside the polygon. Due to this, for any robot $r \in CH(R)$ the corresponding target point $T(r)$ lies inside, the local convex hull $LH(r)$. Since

$$\bigcup_{r \in R} LH(r) \subset H(R) \tag{2.30}$$

we can state that any robot $r \in CH(R)$ also moves inside $H(R)$. It is left to show that if robots are not gathered yet, then the velocity of the corner robots is always 1.

Let us consider the corner robot $r \in CH(R)$ and corresponding internal angle $\alpha < \pi$. Let $b$ be a bisector of this angle and $S$ be the local coordinate system such that the origin is at $r$ and the $X$ axis lies on $b$. Let us consider the average coordinate $\bar{x}$ of all robots from

$UDG(r) \setminus r$ on the $X$ axis. Assume that $\bar{x}$ is outside the convex hull. Since the average cannot be greater than the maximum there must be a robot in $UDG(r) \setminus r$ outside of the convex hull.

Assume that $\bar{x}$ is at the origin and positive direction of the $X$ axis points outside the convex hull. All neighbors of $r$ are inside of the convex hull, thus the $X$ coordinate of every robot in $UDG(r) \setminus r$ is either negative or 0. If $\bar{x}$ is at the origin, then all summands are 0. In other words, all neighbors are on the $Y$ axis and $r$ is not a corner robot of the local convex hull $LH(r)$ and therefore $r \notin CH(R)$.

The arguments above imply that $\bar{x}$ is strictly negative. Thus the target point of $r$ does not coincide with the position of $r$. According to the GTGrC algorithm, robot $r$ moves with speed 1. □

Proposition 2 and Lemma 3 imply that robots with limited visibility gather in time $O(n^2)$. They do, but they gather at more than one non-predefined point because the unit disk graph might fall apart into connected components during the gathering process with GTGrC algorithm.

**Definition 2** (Connectivity)**.** *If robots execute some gathering algorithm and $UDG(r)$ is connected for all $t \geq 0$, then we say that the given gathering algorithm preserves connectivity.*

**Lemma 6.** *The Go-To-The-Gravity-Center algorithm does not preserve connectivity.*

*Proof.* Let us consider the following set of initial positions

$$\{x_n^-, ..., x_1^-, x_1^+, ..., x_n^+\} \tag{2.31}$$

of a set $R = \{r_{-1}, ..., r_{-n}, r_n, ..., r_1\}$ of $2n$ robots in one dimensions, where

$$x_i^+ = \left(\frac{1}{2} + \frac{i^2}{\frac{1}{10} + i^2}\right) \tag{2.32}$$

and $x_i^- = -x_i^+$, for $i = \{0, \ldots, n\}$. This set is illustrated in Figure 2.4.

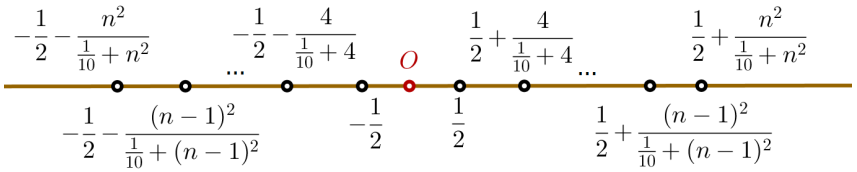



Figure 2.4: An example of the initial configuration of $2n$ robots that breaks into two connected components at any $t > 0$, if robots use the GTGrC algorithm and $n \geq 4$.

All robots execute the GTGrC algorithm. Let us consider robot $x_1^+$ situated at the position $1/2$. The unit disk graph neighborhood of $x_1^+$ except itself consists of $x_1^-$ and $\{x_2^+, ..., x_n^+\}$. Let us consider the target point $T(x_1^+)$. It is given by

$$T(x_1^+) = \frac{-\frac{1}{2} + \frac{1}{2} + \sum_{i=2}^{n} \left(\frac{1}{2} + \frac{i^2}{\frac{1}{10}+i^2}\right)}{n+1}. \tag{2.33}$$

The elements of the sequence 2.32 grow monotonically as $i$ grows. For $i \geq 2$ it holds that $x_i^+ \geq 9/10$. Using this fact we can bound from below $T(x_1^+)$ as follows:

$$T(x_1^+) > \frac{9(n-1)}{10(n+1)}. \tag{2.34}$$

For $n \geq 4$ it holds that $T(x_1^+) > 1/2$. In other words if $n \geq 4$, then target point of $x_1^+$ is strictly on the right of its position. Due to symmetry we can conclude that the target point of $x_1^-$ is strictly on the left of its position. The distance between $x_1^+$ $x_1^-$ is 1 – as soon as they start moving the whole configuration will be disconnected. □

### 2.3.3 Go-To-The-Center Algorithm

Ando, Suzuki, and Yamashita in [ASY95] present the Go-To-The-Center algorithm. They show that robots with limited visibility acting in a synchronous time model gather at one non-predefined point. In this section we will show that in the continuous time model the Go-To-The-Center algorithm gathers $n$ robots with limited visibility in one non-predefined point in time $O(n^2)$.

The smallest or minimum enclosing circle (MEC) plays a central role in this algorithm. The *minimum enclosing circle* is the smallest circle that contains all points of a given set on the Euclidean plane. Let us first consider the two-dimensional case. For the MEC in 2D, Chrystal has shown the following properties.

**Proposition 3.** (Chrystal [Chr85]) *Let $C$ be minimum enclosing circle of a point set $S$. Then either:*

1. *there are two points $m_1, m_2 \in S$ on the circumference of $C$ such that the line segment $(m_1 m_2)$ is a diameter of $C$, or*

2. *there are three points $m_1, m_2, m_3 \in S$ such that $C$ circumscribes $\triangle m_1 m_2 m_3$ and the center $c$ of $C$ is inside $\triangle m_1 m_2 m_3$, which means that $\triangle m_1 m_2 m_3$ is an acute or at most a right triangle.*

Let us consider a robot $r \in R$ at some point in time $t$ together with its unit disk graph neighborhood $UDG_t(r)$. The minimum enclosing circle $C_t(r)$ encircles all unit disk neighbors in $UDG_t(r)$.

With respect to the cases considered in Proposition 3 we will call the *minimum enclosing set* of the robot $r$ (at time $t$) the following sets: $MEC_t(r) = \{m_1, m_2\}$ or $MEC_t(r) = \{m_1, m_2, m_3\}$. We say that robots of $MEC_t(r)$ *form* the minimum enclosing circle $C_t(r)$ of robot $r$ at time $t$. Note that robot $r$ might belong to its own

minimum enclosing set, e.g. $MEC_t(r) = \{m_1, r\}$. In the one-dimensional case the smallest enclosing circle that encircles more than one robot is always formed by two robots.

The minimum enclosing circle of a point set is unique [Chr85]. The minimum enclosing circle (sphere) can be found in linear time in the Euclidean space of any constant dimension [Meg83]. In case there is more than one minimum enclosing set $MEC_t(r)$ that may form $C_t(r)$, then we assume that the robot selects one of them arbitrarily. Further, we skip $t$ in the notation of the minimum enclosing set and circle unless the time must be mentioned explicitly.

**Go-To-The-Center (GTC) algorithm:** Each robot $r \in R$ at every point in time $t$ observes the relative positions of all robots within viewing range 1, i.e. $UDG(r)$. Then the robot computes target point $T(r)$, which is the center of the minimum enclosing circle around all visible robots including itself. Depending on disposition, the robot performs the following actions:

- If $r$ is not at the target point $T(r)$ yet, then the robot moves towards it with speed 1.
- If $r$ is already at the target point $T(r)$, then it follows the motion of the target point.

Note that the position of the target point might change in a discontinuous manner: for example, when a new robot appears in the viewing range of $r$.

Let us first show that the GTC algorithm preserves connectivity.

**Lemma 7.** *Let us consider a group of robots $R$ on the Euclidean plane that follows the Go-To-The-Center algorithm. If $\{u, w\}$ is an edge in $UDG(R)$ at time 0, then $\{u, w\}$ is an edge in $UDG(R)$ at time $t$ for all $t \geq 0$.*

*Proof.* Let us consider robots $u$ and one of the unit disk edges of this robot, namely $\{u, w\}, w \in UDG(u)$. The area $Q_u$ is an intersection of unit discs of all unit disk neighbors $UDG(u)$ at time 0. The center $T(u)$ of the minimum enclosing circle $C(u)$ of $UDG(u)$ at time 0 is situated inside area $Q_u$ as well, since the radius of $C(u)$ is at most the radius of the unit disk and $C(u)$ encircles all robots in $UDG(u)$.

Robot $u$, which executes the GTC algorithm, goes towards its target point, namely the center of minimum enclosing circle $T(u)$. The line segment between $T(u)$ and $u$ is entirely contained in $Q_u$. Assume that at the end of the time interval $[0, t]$, the distance between robots $u$ and $w$ is greater than 1. Since the motion of the robots that follow the GTC algorithm is continuous, there exists a point in time $t' \in [0, t]$ such that at that time $t'$ the distance between $u$ and $w$ is exactly one.

Let $L$ be an intersection of the unit discs of robots $u$ and $w$ at time $t'$. Convex areas $Q_u$ and $Q_w$ are situated inside $L$, thus at time $t'$ robot $u$ (as well as $w$) can only move inside $L$. This contradicts our assumption. Thus, if $\{u, w\}$ is an edge in $UDG(R)$ at time 0, then $\{u, w\}$ is an edge in $UDG(R)$ for all $t \geq 0$. □

Next, in Lemma 8 and 9 we will show that the GTC gathering algorithm is contracting.

**Lemma 8.** *If robots follow the Go-To-The-Center algorithm, then the target point $T(r)$ of any robot $r \in R$ is in $H(R)$.*

*Proof.* Let us consider the minimum enclosing set $MEC(r)$ of robot $r$. All robots of $MEC(r)$ including $r$ itself are inside $H(R)$. In case $MEC(r)$ consists of two robots $m_1$ and $m_2$, the target point is the midpoint of the line segment $m_1m_2$. Since $H(R)$ is convex and $m_1$

and $m_2$ are in it, the line segment between $m_1$ and $m_2$ is inside $H(R)$ too. Therefore, the target point is in $H(R)$.

For the case where $MEC(r)$ consists of three robots $m_1$, $m_2$ and $m_3$, the target point is the center of the circumscribed circle around $\triangle m_1 m_2 m_3$. Let us consider without loss of generality one of the robots in $MEC(r)$, e.g. $m_1$. Let us now draw the line $l$ from $m_1$ through the target point $T(r)$. It is clear that line $l$ intersects the opposite side of $\triangle m_1 m_2 m_3$, namely $m_2 m_3$ at some point $a$. Since $H(R)$ is convex and $m_2$ and $m_3$ are in it, the line segment between $m_2$ and $m_3$ is inside $H(R)$ too, as well as point $a$. The target point $T(r)$ must be also be inside $H(R)$ because the same argument applies for the line segment $m_1 a$. □

**Lemma 9.** *If robot $r \in CH(R)$, then the target point $T(r)$ does not coincide with the position of robot $r(t)$.*

The proof of Lemma 9 is easily derived from the following proposition.

**Proposition 4.** (Chrystal [Chr85]) *Let $C$ be the minimum enclosing circle of a set of $n \geq 2$ points. Then there is no point-free arc with a length greater than $\pi$.*

Here, with respect to our notation, the point-free arc is the circular arc of the minimum enclosing circle $C(r)$ without any other robots on it. If for some robot $r \in CH(R)$ the center of the minimum enclosing circle – i.e., target point $T(r)$ coincides with the position of robot $r$ – then there is a robot-free arc (the part of the circle outside $H(R)$) with an arc length greater than $\pi$. This contradicts Proposition 4. It implies that all $r \in CH(R)$ move with speed 1 and according to Lemma 8 all $r \in CH(R)$ move inside the (closed) convex hull $H(R)$. In other words, the GTC algorithm is

contracting. Besides that, according to Lemma 7 the GTC algorithm preserves connectivity. Summing up all the statements above we get the following theorem.

**Theorem 2.** *The set of robots with limited visibility gathers at one non-predefined point using the Go-To-The-Center algorithm in the time* $O(n^2)$.

### 2.3.4 Go-To-The-Relative-Center Algorithm

Instead of calculating the minimum enclosing circle $C(r)$ with respect to $UDG(r)$, we will use the Relative neighborhood graph proposed in [Tou80]. The latter is defined in the two-dimensional Euclidean space as follows.

**Definition 3** (Relative neighborhood graph criterion)**.** *Any two robots* $u$, $v$ *are connected iff there does not exist any robot* $w \in R$ *satisfying* $|u, w| < |u, v|$ *and* $|v, w| < |u, v|$.

We denote by $RNG_t(r)$ (at time $t$) the subgraph obtained from the UDG neighborhood $UDG_t(r)$ by applying the Relative neighborhood graph criterion. We call $RNG_t(r)$ the unit *Relative graph neighborhood* of robot $r$. The Relative neighborhood graph (RNG) defined on all robots at time $t$ is denoted by $RNG_t(R)$.

The set of points on the Euclidean plane that corresponds to the intersection of open unit disks of all robots in $RNG(r)$ is denoted by $Q(r)$. The set of points $Q(r)$ is open and convex since it is the intersection of open unit disks that are convex [Sin97]. The circle $C_Q(r)$ with center at $r$ inscribed into $Q(r)$ is the *connectivity circle*. The radius of the connectivity circle is denoted by $\rho_Q$. Unless needed explicitly, we skip $t$ in the notation of the Relative neighborhood graph, neighborhoods, etc.

**Go-To-The-Relative-Center (GTRC) algorithm:** Every robot $r \in R$ at every point in time $t$ observes the relative positions of all robots within the viewing range 1, i.e. $UDG(r)$. From $UDG(r)$, the robot calculates $RNG(r)$. It then computes the target point $T(r)$ which is the center of the minimum enclosing circle around $RNG(r)$. Depending on disposition, the robot performs the following actions.

- If $r$ is not at the target point $T(r)$ yet, then the robot moves towards it with speed 1.
- If $r$ is already at the target point $T(r)$, then it follows the motion of the target point.

Note that the position of the target point might change in a discontinuous manner: for example, when new robot appears in the viewing range of $r$ or some robot violates the relative neighborhood criterion and changes $RNG(r)$.

**Proposition 5.** *For every robot $r \in R$ at any fixed point in time, $T(r) \in Q(r)$.*

Proposition 5 holds since the radius of $C(r)$ is less than 1 and $C(r)$ encircles all the robots in $RNG(r)$. The argumentation of the connectivity property is similar to the one used in Lemma 7 for the GTC algorithm.

**Lemma 10.** *Given a group of robots $R$ on the Euclidean plane executing the Go-To-The-Relative-Center algorithm, if $\{u, w\}$ is an edge in the open Relative neighborhood graph $RNG(R)$ at time 0, then there is a path from $u$ to $w$ in $RNG(R), \forall t \geq 0$.*

*Proof.* There are two cases to be considered. Both correspond to the situations in which during the time interval $[0, t]$ the edge $\{u, w\}$ does or does not violate the RNG criterion.

Let us consider the first case, where edge $\{u, w\}$ does not violate RNG criterion during $[0, t]$. Assume that at time $t_1$ it holds that $|u, w| = 1$. Let $L$ be an intersection of open unit disks of robot $u$ and $w$. Without loss of generality we consider robot $u$. According to our assumption there shall exist non-empty time interval $[t_a, t_b] \subset (0, t_1)$ such that during the time interval $[t_a, t_b]$ target point $T(u)$ of robot $u$ was outside $L$.

If there is no such interval, robot $u$ would not be able to leave $L$ since it is a convex set and line segment $uT(u)$ and therefore the whole trajectory is entirely inside $L$ at any point of the interval $[0, t_1]$.

However, existence of non-empty time interval $[t_a, t_b] \subset (0, t_1)$ such that during the time interval $[t_a, t_b]$ target point $T(u)$ of robot $u$ was outside $L$ contradicts Proposition 5, since $Q(u) \subseteq L$. We can therefore conclude that if edge $\{u, w\}$ does not violate the RNG criterion, then it remains the edge of $RNG(R)$ at $\forall t \geq 0$.

If an edge $\{u, w\}$ is not an edge in $RNG(R)$ at time $t_1$, due to the RNG criterion, then there is at least one robot $v$ that causes connection in $RNG(R)$ between $u$ and $w$ to be removed at some point in time $t' \in [t_0, t_1]$. Therefore, there shall be an alternative path via a robot that causes the connection to be removed [Tou80]. This is why there is a path from $u$ to $w$ in $GG(R), \forall t \geq 0$. □

The properties of the minimum enclosing circle do not depend on the subgraph of the unit disk graph that we use to calculate it. Thus, Lemma 8 and 9 also hold for the GTRC algorithm. In other words, the GTRC algorithm is contracting and, taking Lemma 10 into account, we can conclude the following.

**Theorem 3.** *The set of n robots with limited visibility gathers at one non-predefined point using the Go-To-The-Relative-Center al-*

*gorithm in time $O(n^2)$.*

The relative neighborhood graph does not have any impact on runtime since, Lemma 8 and 9 depend only on the properties on minimum enclosing circle. Because of this we can use any subgraph of the unit disk graph as long as connectivity is preserved during runtime. In [LMP16] we have studied the impact of the Gabriel graph on the gathering process of robots with limited visibility in the continuous time model.

There we presented the *Go-To-The-Gabriel-Center* (GTGC) algorithm, where the target point is the center of the minimum enclosing circle around the Gabriel neighborhood $GG(r)$. The Gabriel graph neighborhood is the subgraph of the unit disk graph neighborhood $UDG(r)$; it is calculated using the criterion proposed by Gabriel and Sokal in [GS69].

**Definition 4** (Gabriel graph criterion)**.** *Any two robots $u$, $v$ are connected if no other robot $w$ is present within the circle whose diameter is line segment $uw$. We call this circle a* Gabriel circle.

On first sight, the Gabriel graph and relative neighborhood graph do not have any severe impact on the gathering process. The runtime of the GTC does not improve.

**Theorem 4.** (Li et al. [LMP16]) *The set of $n$ robots with limited visibility gathers at one non-predefined point using the Go-To-The-Gabriel-Center algorithm in time $O(n^2)$.*

However, simulations exhibit a severe difference in the behavior of the GTC and GTRC algorithms: Whereas lots of collisions occur during a run of the GTC algorithm, typically only one, the final collision namely occurs during a run of the GTRC algorithm. We will investigate this observation in the next chapter.

# Chapter 3

# Collisionless Gathering

In this chapter we examine the difference between the behavior of the Go-To-The-Center and Go-To-The-Relative-Center algorithms. We are particularly interested in the *collisionless* gathering property. We say that the robots *collide* if they have the same position at the same time.

Note that if two robots with limited visibility collide, they move as one and share the same position for the rest of the execution of the algorithm. This happens since robots see the same neighborhood and therefore behave in the same way after the collision. At the end of the gathering process, all robots collide at one non-predefined point. We call this collision the *final* one. All other collisions are called *early* collisions. If we have no early collisions, the gathering is *collisionless*.

## 3.1 Go-To-The-Center Algorithm is not Collisionless

Next we present an example of a robust set of initial configurations on the Euclidean plane. For each of these, the GTC algorithm produces many early collisions. A set of initial configurations is

*robust* if it is defined by a set of $n$ disjoint circles so that the $n$ positions are drawn arbitrarily, one from each circle.

Let us consider the set of initial positions $\{x_1^-, ..., x_n^-, x_n^+, ..., x_1^+\}$ of a set $R = \{r_{-1}, ..., r_{-n}, r_n, ..., r_1\}$ of $2n$ robots on the Euclidean line, where $x_i^- = (-\frac{1}{2} + \frac{i}{3n})$ and $x_i^+ = (\frac{1}{2} - \frac{i}{3n})$, for $i = \{1, \ldots, n\}$. Compare Figure 3.1.

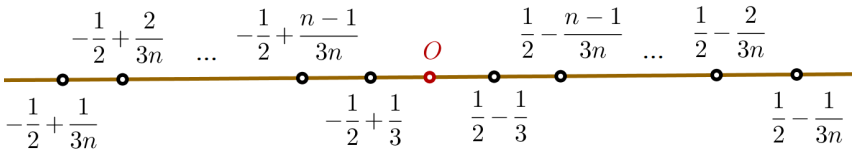



Figure 3.1: An example of the initial configuration that has an early collision, if robots use the GTC algorithm.

As all robots in this configuration see each other, they have the same unit disk graph neighborhood and therefore the same target point at the origin. Thus, according to the GTC algorithm, they move with speed 1 towards the origin. Every pair of robots $r_{-i}$ , $r_i$ produces a collision at time $(\frac{1}{2} - \frac{i}{3n})$. The first $n-1$ collisions are early collisions.

We now define the following robust set of initial configurations in two dimensions. Let $\epsilon = \frac{1}{8n}$ and consider all initial configurations $\{y_1^-, ..., y_n^-, y_n^+, ..., y_1^+\}$ with $|y_i^-, (x_i^-, 0)| \leq \epsilon$ and $|y_i^+, (x_i^+, 0)| \leq \epsilon$.

All robots in this new configuration have distinct positions and can see each other. Thus, they have the same unit disk graph neighborhood and therefore the same target point $z$, the midpoint between $y_1^-$ and $y_1^+$. Note that $z$ does not change during the execution of the algorithm. The target point $z$ is at most $\epsilon$ away from the origin, thus it is between $y_n^-$ and $y_n^+$. The robots move with speed 1 towards $z$. As (at most) two robots, namely those with initial positions $y_1^-$ and $y_1^+$, arrive at the target point at the same time, we get for $n \geq 2$ at least $(n-1)$ and at most $(2n-3)$

early collisions at the target point. (For $n = 2$ there are no early collisions.)

## 3.2 Go-To-The-Relative-Center Algorithm in One Dimension

We are given a group of robots $R = \{r_1, ..., r_n\}$ in one dimension, ordered with respect to their initial positions, i.e. $r_1(0) < r_2(0) < \ldots < r_n(0)$. The initial positions of all robots are distinct, and the distance between any two consecutive robots is less or equal to one.

Let us consider time 0. It is easy to see that the relative neighborhood graph obtained from the unit disk graph is nothing but a path graph with edges $\{r_j(0), r_{j+1}(0)\}$, where $j = 1, 2, \ldots, n-1$. Thus, each $r_i$, with $2 \leq i \leq n-1$ has the two relative neighbors $r_{i-1}$ and $r_{i+1}$. The end robots $r_1$ and $r_n$ have only one relative neighbor, namely $r_2$ and $r_{n-1}$, respectively.

Now let $t_*$ be the time of a first collision of the GTRC algorithm started in this initial configuration. For every $t < t_*$, the order of the robots on the line remains unchanged, i.e. $r_1(t) < r_2(t) < \ldots < r_n(t)$. Thus the target points $T_i(t)$ at time $t$ are defined as follows:

$$T_i(t) = \begin{cases} \frac{r_1(t)+r_2(t)}{2} & \text{if } i = 1; \\ \frac{r_{i-1}(t)+r_{i+1}(t)}{2} & \text{if } i = 2, \ldots, n-1; \\ \frac{r_{n-1}(t)+r_n(t)}{2} & \text{if } i = n. \end{cases} \tag{3.1}$$

Let us first consider the runtime of this algorithm in one dimension.

**Theorem 5.** *In one-dimensional Euclidean space, the GTRC algorithm gathers $n$ robots in time $\Theta(n)$.*

*Proof.* The corner hull $CH(R)$ around the configuration where all

robots are placed on the same line consists of only two robots. Let us call them $c_1$ and $c_2$. Due to Lemma 9, the center of the minimum enclosing circle cannot coincide with the position of robot $c_1(t) \in CH(R)$, thus robot $c_1$ moves. By Lemma 8, the velocity vector $\vec{v}_1(t)$ of robot $c_1$ points inside the corner hull. The same holds for $c_2$. Robots $c_1$ and $c_2$ move with speed 1. Thus speed $l'(t)$ with which the length of the global corner hull $l(t)$ decreases is 4. The corner hull consists of two line segments $c_1c_2$ and $c_2c_1$, both decreasing with speed 2.

The length of the corner hull $l(t)$ in one dimension is at most $2(n-1)$. Thus, after time

$$t_* = \frac{l(t)}{4} \leq \frac{(n-1)}{2} \tag{3.2}$$

all robots are gathered at one point in the one-dimensional case. In the case where all robots are placed within maximum distance (viewing range) apart from each other, our inequality 3.2 becomes an equality. □

Next, we check the collisionless property of GTRC in one dimension. Let $M$ be the set of robots that collide at time $t_*$. We claim that $M = R$. Therefore, the first collision is the final collision; there are no early collisions.

**Theorem 6.** *In one dimension, gathering with the Go-To-The-Relative-Center algorithm is collisionless.*

*Proof.* Assume that $r_i$ is in $M$, but $r_{i-1}$ is not. Then $r_i$ and $r_{i+1}$ collided at time $t_*$. The distance $|r_i(t), r_{i+1}(t)|$ approaches 0 as we let $t \to t_*$. Due to the continuity, there exists a point $t' \in [0, t_*]$ such that $|r_{i-1}(t), r_i(t)| < |r_i(t), r_{i+1}(t)|$ for any $t \in [t', t_*]$.

This means that the target point of $r_i$ is on the left of $r_i$ at the time of a first collision. Therefore it was left of $r_i$ from time 0

and on. But this means that $r_i$ was running with speed 1 to the left from time 0 until time $t_*$. As $r_{i+1}$ was to the right of $r_i$ at time 0, it cannot reduce the distance to $r_i$ until time $t_*$. Thus, $r_i$ and $r_{i+1}$ can not collide at time $t_*$. Thus, the above assumption is wrong. Therefore, if $r_i \in M$, then $r_1, \ldots, r_i \in M$. A symmetric argument yields the following: if $r_i \in M$, then $r_i, ..., r_n \in M$. Thus $M = R$. □

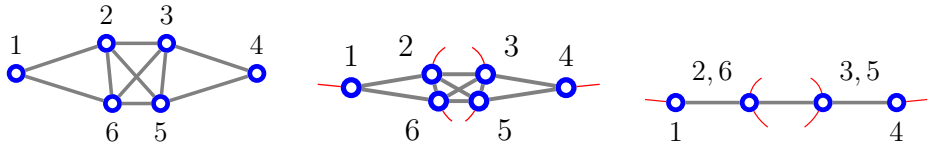
## 3.3 Go-To-The-Relative-Center Algorithm in Two Dimensions

Is gathering with the GTRC algorithm also collisionless in a two-dimensional Euclidean space? To answer this question we perform simulation – the continuous motion of the robots is replaced by the discrete one with a small discretization step. Two robots have a collision if the distance between them is less than a collision threshold, which is a few orders of magnitude less than the discretization step. The simulation process is terminated when the maximum distance between any two robots in the group is smaller than the final threshold, which is slightly greater than the number of robots times the discretization step.

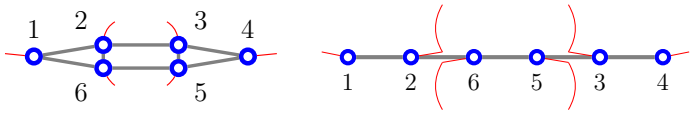
In our experiments, we observe that all randomly generated configurations had no early collisions with the GTRC algorithm. An example that highlights the different behavior of GTC and GTRC is shown in Figure 3.2.

The initial configuration in Figure 3.2 (a) is connected. Figures 3.2 (b) and 3.2 (c) represent the evolution of the group of robots that use the GTC algorithm. Figures 3.2 (d) and 3.2 (e) represent robots that use the GTRC algorithm at the same points in time $t_1 < t_2$. Robots $2, 6$ and $3, 5$ that use the unit disk graph neighborhood have

an early collision at time $t_2$ and behave as one since that point in time. On the other hand, robots that used the GTRC algorithm have no collisions.




(a) Initial configuration at time 0. (b) Unit disk graph at $t_1$. (c) Unit disk graph at $t_2$.




(d) Relative neighborhood graph at $t_1$. (e) Relative neighborhood graph at $t_2$.

Figure 3.2: Visibility graphs during the gathering of the robots that use original Go-To-The-Center and Go-To-The-Relative-Center. Blue circles represent robots; red lines represent trajectories of robots.

We perform three experiments with different inputs and measure the number of collisions between the robots during the run.

**Experiment 1:** The only initial configuration known to us that produces early collisions with the GTRC algorithm is a cross-shaped graph (see Figure 3.3), constructed as follows. The robots are split into two groups $H$ and $Q$. The robots of one group (e.g. $H$) are placed along the line $h$ at an equal distance apart from each other. The other group ($Q$) is placed on the line $q$ perpendicular to $h$. The line $q$ crosses $h$ at the midpoint between the end robots of $H$ on $h$. The robots of $Q$ are placed on the distinct positions on $q$ such that the whole graph is connected, and the distance between the end robots of $Q$ on $q$ is less than the distance between the end robots of $H$ on $h$.

In this configuration, all robots in $Q$ will have an early collision at the point where lines $q$ and $h$ cross. If we are going to "shake"

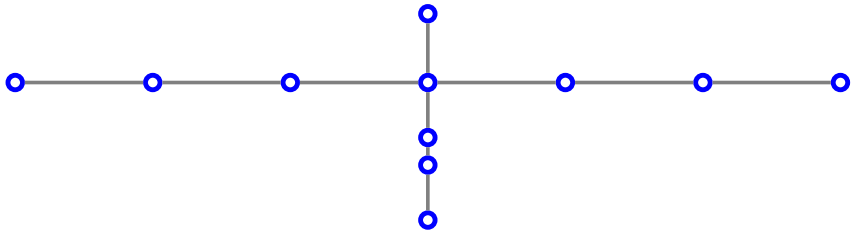


Figure 3.3: An instance of the cross-shaped graph that leads to early collisions with GTRC.

such a cross-shaped initial configuration – i.e., every robot moves independently at random in some small $\epsilon$-ball around its initial position such that connectivity is preserved – then the gathering in the simulation with GTRC will also be collisionless.

This feature in the behavior of the GTRC algorithm in two dimensions is reflected in the Collisionless Conjecture, which we formally describe as follows.

**Conjecture 1** (Collisionless conjecture)**.** *Let us consider the arbitrary initial configuration of robots that use the Go-To-The-Relative-Center algorithm in the two-dimensional Euclidean space. There is an* $\epsilon > 0$ *such that if the position of each robot is perturbed uniformly at random inside the* $\epsilon$*-ball around its initial position, then the probability of an early collision is* 0*.*

**Experiment 2:** In this experiment, we let the robots execute GTC and GTRC on random graphs (e.g. see Figure 3.4). The initial configuration – i.e., the random unit disk graph is created by the Monte Carlo method. Every robot is placed independently at random on the plane and is removed if it is not connected to the rest of the robots.

The statistical information about the runs of then the GTC algorithm with the random graphs of different size as an input is presented in Table 3.1. The number of early collisions grows with the size of the input. The deviation grows as well. In the same ex-

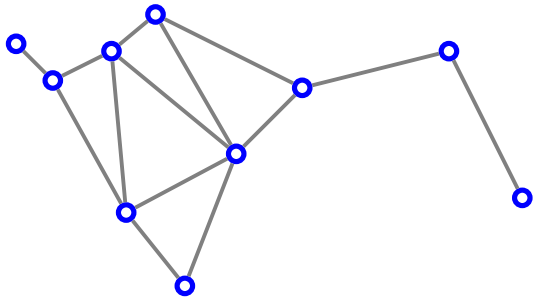


Figure 3.4: An instance of the random unit relative neighborhood graph

periment, with the random graph and the GTRC algorithm, robots had no collisions at all.

Table 3.1: Go-To-The-Center algorithm with the random unit disk graph as an input.

| Number of robots | | 10 | 20 | 30 | 40 | 50 | 60 |
|---|---|---|---|---|---|---|---|
| Number of collisions | Mean value | 0.18 | 1.87 | 3.52 | 8.45 | 15.41 | 19.7 |
| | Standard deviation | 0.67 | 3.19 | 4.55 | 8.38 | 11.57 | 14.29 |
| Sample size | | 100 | 100 | 100 | 100 | 100 | 100 |

**Experiment 3:** In the last experiment, we use an initial configuration for the graph with a particular structure. We aim to show that GTRC is what makes that difference, rather than the input. Here, the input is obtained from a path graph by placing additional $m$ neighbors very close to every $k$-th robot in the path, i.e. cluster. We call such a graph *clustered path graph* (see Figure 3.5). In this way, we make sure that every such cluster around the $k$-th robot will produce a collision.

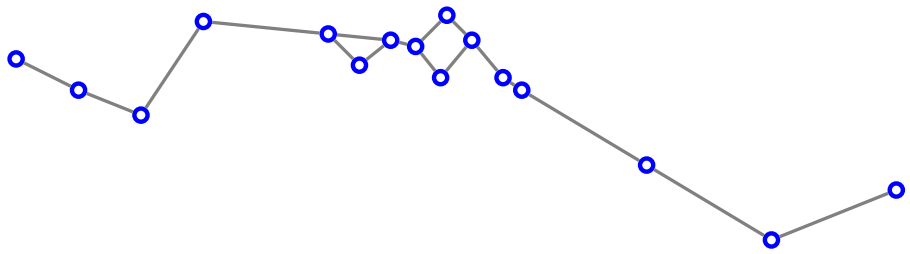


Figure 3.5: An instance of the random clustered path graph with relative neighborhood edges and a single cluster in the middle.

The statistical information for the experiments with the clustered path graph is presented in Table 3.2. The small deviation at every input size suggests that GTC with a clustered path as an input definitely produces early collisions. In the same experiment with a clustered path graph as an input, the GTRC algorithm had no collisions at all.

Table 3.2: Go-To-The-Center algorithm with a clustered path graph as an input.

| Number of robots | | 16 | 26 | 36 | 46 | 56 | 66 |
|---|---|---|---|---|---|---|---|
| Number of collisions | Mean value | 9.05 | 17.7 | 24.1 | 33.85 | 43.6 | 54.75 |
| | Standard deviation | 1.1 | 1.69 | 1.45 | 1.81 | 1.24 | 1.29 |
| Sample size | | 100 | 100 | 100 | 100 | 100 | 100 |

In the same three types of experiments, robots that used the Go-To-The-Gabriel-Center algorithm had no collisions at all as well. Our experiments strongly support the Collisionless Conjecture. The major open problem in this context is to prove it. However, this turns out to be difficult to do with common methods. In Section 3.5, we will discuss on one of the ways to prove the conjecture for $n = 4$ robots.

## 3.4 Collisions in Go-To-The-Relative-Center Algorithm

In this section we investigate the structural properties of collisions during a gathering with the GTRC algorithm. Recall that if two or more robots have the same position at the same point in time, then there is a collision. We refer to the set of robots in the minimum enclosing set $MEC(r)$ of a robot $r \in R$ as a *crash* point. We are able to show that collisions in GTRC take place only at crash points.

**Definition 5** (Crash point)**.** *For any robot $r$ with the minimum enclosing set $MEC(r)$, the midpoint between any two robots $m_1m_2 \in MEC(r)$ is a crash point $p(r)$. We say that robots $m_1$ and $m_2$ define the crash point $p(r)$.*

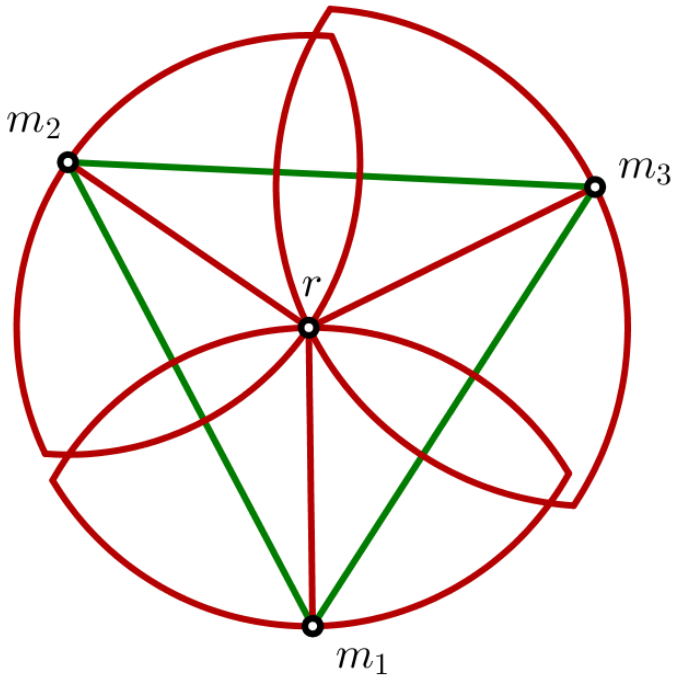
Note that if a minimum enclosing set consists of three robots $MEC(r) = \{m_1, m_2, m_3\}$, then the midpoint of every edge in $\triangle m_1m_2m_3$ is a crash point.

The set of Relative neighborhood edges adjacent to robot $r$ is denoted as $E_{RNG}(r)$. The set of all edges of the Relative neighborhood graph (RNG) is denoted by $E_{RNG}(R)$. In the same way, we define the set $E_{UDG}(r)$ of the unit disk graph edges adjacent to robot $r$ and the set $E_{MEC}(r)$ of the Relative neighborhood edges between robot $r$ and the members of the minimum enclosing set $MEC(r)$.

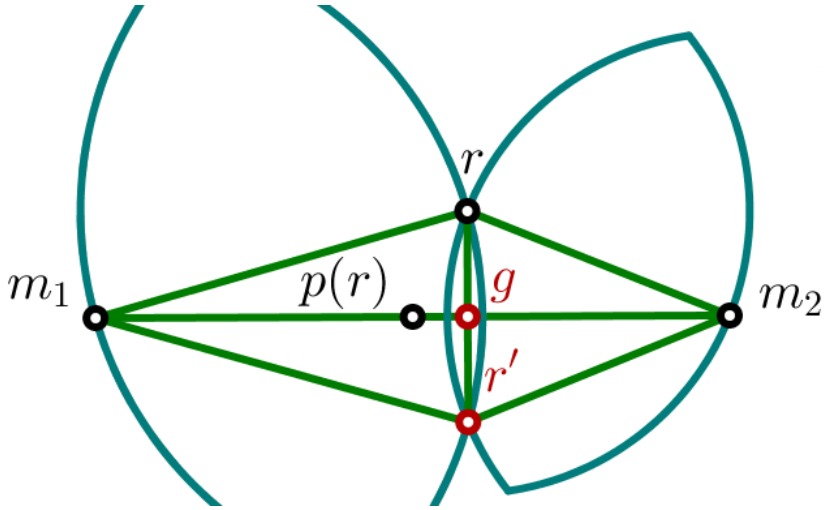
For the Relative neighborhood graph $RNG(R) = (R, E)$, we define the set-valued map $\mathfrak{R} : X \rightsquigarrow Y$, where $X \subset E$ and $Y \subset \mathbb{R}^2$ relate the RNG edges to the area occupied by the corresponding RNG lenses. For example, $\mathfrak{R}(\{r, u\}, \{r, w\})$ is the union of the RNG lenses that correspond to the RNG edges $\{r, u\}, \{r, w\}$.

**Definition 6** (Minimum enclosing set cover)**.** *The convex hull around robot $r$ and the members of $MEC(r)$ form the minimum enclosing set cover $K(r)$ of robot $r \in R$.*

Let us now consider a robot $r \in R$ and its minimum enclosing sets. We show that, in any case, the minimum enclosing set cover is a subset of the union of the RNG lenses that correspond to the RNG edges between robot $w$ and the robots of $MEC(w)$, namely $K(r) \subset \mathfrak{R}(E_{MEC}(r))$. Using basic geometric arguments, we conclude Lemma 11.




(a) Robot $r$ together with its minimum enclosing set $MEC(r)$ and RNG lenses, which correspond to the RNG edges between $r$ and members of $MEC(r)$.




(b) Robot $r$ together with a crash point $p(r)$ and the robots $m_1$, $m_2$ that define this crash point. Besides that, here we depict RNG lenses that correspond to the RNG edges $\{m_1, r\}$ and $\{m_2, r\}$.

Figure 3.6: Lenses of the Relative neighborhood graph.

**Lemma 11.** *If $MEC(r) = \{m_1, m_2, m_3\}$ is the minimum enclosing set over $RNG(r)$, then $\triangle m_1 m_2 m_3$ is completely covered by the RNG lenses that correspond to the RNG edges between $r$ and the robots of $MEC(r)$.*

*Proof.* We know that the center of the minimum enclosing circle always lies inside of $\triangle m_1 m_2 m_3$ since the given triangle is at most a right triangle. Now let us consider robot $r$ with the minimum enclosing set $MEC(r) = \{m_1, m_2, m_3\}$ and RNG lenses $C_1, C_2, C_3$ that $r$ creates between its neighbors $m_1, m_2, m_3$ as shown in Figure 3.6a. We would like to show that $\triangle m_1 m_2 m_3$ is entirely covered by the RNG lenses that form a non-convex figure $I = C_1 \cup C_2 \cup C_3$.

Let us consider one of the edges of $\triangle m_1 m_2 m_3$ and the two RNG lenses $C_1, C_2$ that pass through $m_1$ and $m_2$ as well as through robot $r$. The robots and lenses are depicted in Figure 3.6b. Let us consider $\triangle m_1 r r'$, where $r'$ is an intersection of lenses on the other side of $m_1, m_2$. This triangle is isosceles, i.e. $|m_1, r| = |m_2, r'|$, since $r'$ belongs to the circle with the center at $m_1$ and the radius $m_1 r$. Therefore $\triangle m_1 r g$ is a right triangle. Cathetus $m_1 g$ is shorter than hypotenuse $m_1 r$, thus it is inside the lens $C_1$. Since the lens is convex, the other cathetus $rg$ is also inside lens $C_1$. We show the same for $C_2$ and $\triangle m_2 r g$ and all remaining triangles that correspond to the edges of $\triangle m_1 m_2 m_3$. As a result, we prove that $\triangle m_1 m_2 m_3$ is entirely contained inside $I = C_1 \cup C_2 \cup C_3$. □

We can show similar results as in Lemma 11 for the minimum enclosing sets with different structures.

**Lemma 12.** *If $MEC(r) = \{m_1, m_2\}$ is the minimum enclosing set over $RNG(r)$, then $\triangle r m_1 m_2$ (Figure 3.6b) is completely covered by the RNG lenses between $r$ and the robots of $MEC(r)$.*

**Lemma 13.** *If $MEC(r) = \{m_1, m_2, r\}$ is the minimum enclosing set over $RNG(r)$, then $\triangle r m_1 m_2$ is completely covered by the RNG lenses between $r$ and the robots of $MEC(r)$.*

It remains to consider the case where $MEC(r) = \{m_1, r\}$ is the minimum enclosing circle over $RNG(r)$. We observe that the minimum enclosing circle $C(r)$ is a proper subset of the RNG lens between $m_1$ and $r$. This observation together with Lemmata 11, 12, and 13 yield the following corollary.

**Corollary 4.** *The union of the RNG lenses between robot $r$ and the robots of $MEC(r)$ is a superset of the minimum enclosing set cover $K(r)$, i.e. $K(r) \subset \mathfrak{R}(E_{MEC}(r))$.*

Note that the RNG lenses between the unit disk graph neighbors that satisfy the RNG criterion do not contain any other robots. If there is a robot inside one of the RNG lenses, then the corresponding part of the RNG will change.

Let us now consider, in detail, the collisions that occur between the robots executing GTRC. The set of robots $M \subset R$ that have a collision at time $t_*$ is represented by a single robot $u$ for any $t \geq t_*$. We call $u$ the *representative* of $M$.

**Observation 1** (Early collision)**.** *The set of robots $M \subset R$ had a collision at time $t_*$ if the minimum enclosing circle around the RNG neighborhood of the representative $u$ has a diameter greater than zero.*

Note that the opposite of an early collision is a *final collision.* In other words, assume that the set of robots $M \subseteq R$ had a collision at time $t_*$ and the minimum enclosing circle around the RNG neighborhood of the representative $u$ has the diameter zero. This implies that, after the final collision, there are no other robots in

the unit disk graph neighborhood of the representative $u$. There are also no robots other than those in the unit disk graph neighborhood of robot $u$, due to the connectivity property of the GTRC algorithm shown in Lemma 10. Thus, if final collision takes place, then $M = R$. We say that the gathering is *collisionless* if there are no early collisions.

In the next Lemma 14, we utilize Corollary 4 to show that a robot $w$ can have early collision only at the crash point $p(w)$. The proof is a result of carefully checking all possible dispositions of $w$ and $K(w)$.

**Lemma 14.** *Let us consider robot $w$ during an arbitrary time interval $[a, b]$, such that during this interval, diameter $d_w$ of the minimum enclosing circle of robot $w$ is strictly greater than zero. If robot $w$ collides with some other robots $M \subset R$ during $[a, b]$, then:*

1. *collision takes place at the crash point $p(w)$ of robot $w$, and*
2. *robot $w$ does not belong to its own minimum enclosing set $MEC(w)$.*

*Proof.* We inspect the movement of robot $w$ before the collision during the time interval $[a, b]$. The diameter of the minimum enclosing circle $d_w$ of robot $w$ is strictly greater than zero during this time interval. Robot $w$ according to GTRC moves towards its target point $T(w)$ or, if it is already at $T(w)$, then robot $w$ follows the motion of the target point.

Let us consider the minimum enclosing set cover $K(w)$. We shall keep in mind that $K(w) \subset \mathfrak{R}(E_{MEC}(w))$ by the Corollary 4 and therefore there are no other robots (except $w$ and $MEC(r)$) that belong to $K(w)$. We represent the cover as the union $K(w) = K_{fr}(w) \cup K_{in}(w)$, where $K_{fr}(w)$ is the frontier or the boundary of $K(w)$ and $K_{in}(w)$ is an interior of $K(w)$.

By definition, $w \in K(w)$. The target point $T(w)$ is inside $K(w)$ too since it belongs to the triangle or the line segment that forms the minimum enclosing circle and, by the definition of $K(w)$, the according triangle or line segment is inside $K(w)$. Thus, it holds that $w, T(w) \in K(w) \subset \mathfrak{R}(E_{MEC}(w))$.

Assume that robot $w$ did not reach its target point $T(w)$ at the end of time interval $[a, b]$. Namely, $w$ moves towards the target point $T(w)$ with speed 1. The velocity of the robot always points towards $T(w)$. For robot $w$ during time interval $[a, b]$, we consider the set $I$ that consists of infinitely many line segments that connect robot $w$ and $T(w)$ at every point of the time interval $[a, b]$.

Assume that $w \in K_{in}(w)$ and $T(w) \in K_{in}(w)$ during $[a, b]$ depicted in Figure 3.7a. This implies that $I \subset K_{in}(w)$ since $K_{in}(w)$ is a convex set. Every point of $I$ together with a small enough ball around the point are in $K_{in}(w)$. Therefore, every point of the trajectory of robot $w$ during $[a, b]$ is in $K_{in}(w)$ together with small enough ball around this point. No other robot may approach arbitrarily close to and eventually collide with $w$ without changing the RNG during $[a, b]$.

Assume that $w \in K_{in}(w)$ and $T(w) \in K_{fr}(w)$ during $[a, b]$ as depicted in Figure 3.7b. If we assume that $w$ did not reach its target point $T(w)$ at the end of the time interval $[a, b]$, then $(I \setminus \bigcup_{[a,b]} T(w)) \subset K_{in}(w)$, where $\bigcup_{[a,b]} T(w)$ are the positions of $T(w)$ during time interval $[a, b]$. Every point of the trajectory of robot $w$ during $[a, b]$ is entirely inside of $K_{in}(w)$ together with a small enough ball around this point. Therefore, in this case as well, no other robot may approach arbitrarily close to and eventually collide with $w$ without changing the RNG during $[a, b]$.

Now we assume that $w \in K_{fr}(W)$ and $T(w) \in K_{in}(w)$ during $[a, b]$ as depicted in Figure 3.7c. The small enough ball $B(w)$ around

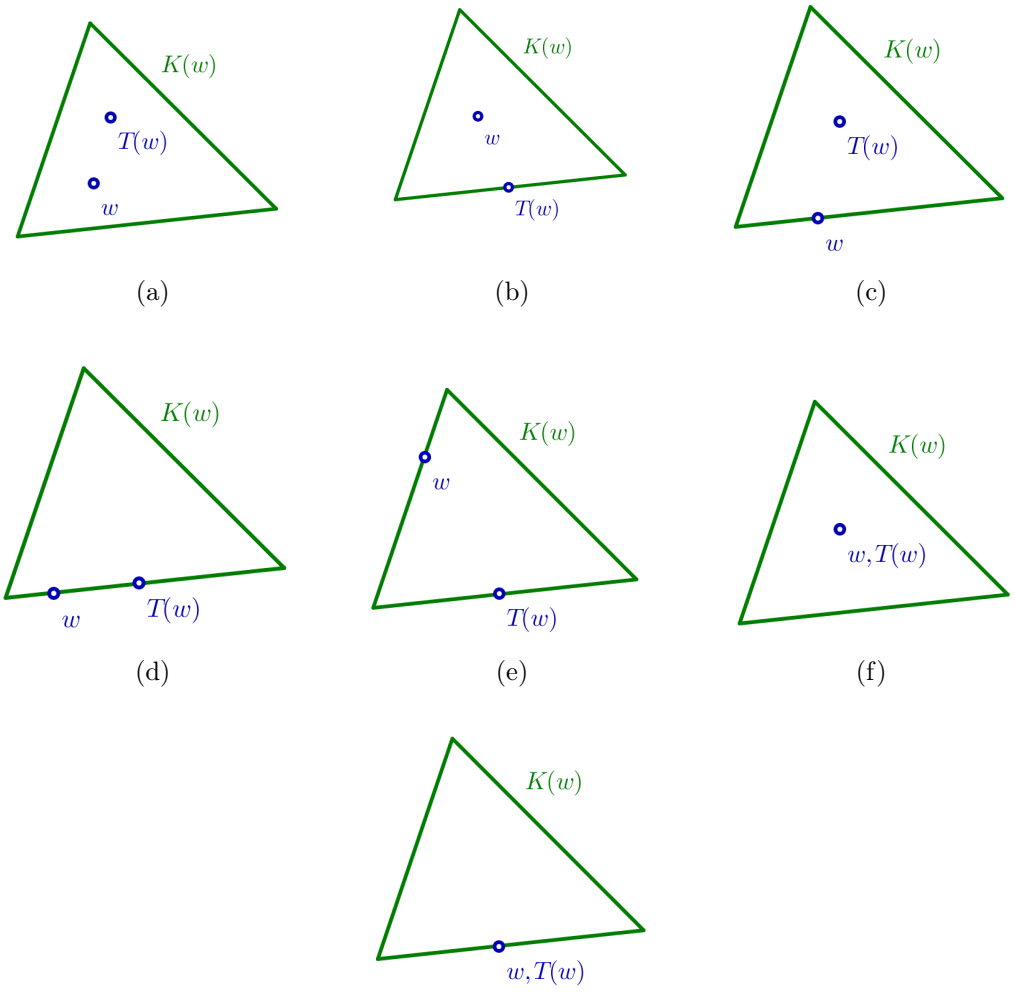



Figure 3.7: All possible dispositions of $w$ and $K(w)$.

the position of robot $w$ can be separated into two parts $B_{in}(w) = B(w) \cap K_{in}(w)$ and $B_{out} = B(w) \setminus (B_{in}(w) \cap K_{in}(w))$. According to the GTRC, robot $w$ moves towards $T(w) \in K_{in}(w)$. There are no robots at the points of $B_{in}(w)$ with a small enough ball around it. Therefore, no other robot may approach $w$ from $B_{in}(w)$ arbitrarily close to and eventually collide with $w$ without changing the RNG during $[a, b]$. There might be some robots inside $B_{out}$, but robot $w$ moves away from $B_{out}$ at maximum constant speed 1 and therefore according to [Sch95] even if some robots from $B_{out}$ pursue $w$ as a target point, they will only shorten the distance to $w$ but never reach it since all other robots move with the same or lower speed.

Let us now assume that both robots at position $w$ and the target point $T(w)$ are on the frontier during $[a, b]$, i.e. $w \in K_{fr}(W)$ and $T(w) \in K_{fr}(w)$. There are two possible dispositions of $w, T(w)$ and $K(w)$, depicted in Figure 3.7d and Figure 3.7e, respectively.

This is similar to the previous case. However, instead of $K(w)$, we take a look at $\mathfrak{R}(E_{MEC}(w)) = \mathfrak{R}_{fr}(E_{MEC}(w)) \cup \mathfrak{R}_{in}(E_{MEC}(w))$, where $\mathfrak{R}_{fr}(E_{MEC}(w))$ is the frontier or the boundary of $\mathfrak{R}(E_{MEC}(w))$ and $\mathfrak{R}_{in}(E_{MEC}(w))$ is an interior of $\mathfrak{R}(E_{MEC}(w))$. We split the small enough ball $B(w)$ into two parts such that $B_{in}(w) = B(w) \cap \mathfrak{R}_{in}(E_{MEC}(w))$ and $B_{out} = B(w) \setminus (B_{in}(w) \cap \mathfrak{R}_{in}(E_{MEC}(w))$.

There are no robots at the points of $B_{in}(w)$ and the small enough ball around it. In order to show that this statement is true, let us consider the frontier of $K(w)$. The latter consists of a line segment between robot $w$ and the robots of $MEC(w)$. If $w$ and $T(w)$ belong to the different line segments of $K_{fr}(w)$ (see Figure 3.7e), then we fall into the case where $w \in K_{fr}(W)$ and $T(w) \in K_{in}(w)$ during $[a, b]$, since robot $w$ did not reach its target point $T(w)$ at the end of the time interval $[a, b]$. This case is already considered.

Let us take a look at the situation where $w$ and $T(w)$ belong

to the same line segment of $K_{fr}(w)$ as depicted in Figure 3.7d. There are two ways the considered line segment might be formed. First, the line segment is formed by $wm_1$, where $m_1 \in MEC(w)$. Line segment $wm_1$ without endpoints is inside $\mathfrak{R}_{in}(E_{MEC}(w))$ since there must be an RNG lens between $w$ and $m_1$. The similar argument holds if the considered line segments of $K_{fr}(w)$ is formed by $m_1m_2$, where $m_1, m_2 \in MEC(w)$. Line segments $w_m1$ and $wm_2$ without endpoints are inside of $\mathfrak{R}_{in}(E_{MEC}(w))$ since there must be an RNG lens between $w$ and $m_1$ as well as between $w$ and $m_2$. According to our assumption, $T(w) \in K_{fr}(w)$ and therefore $T(w) \in \mathfrak{R}_{in}(E_{MEC}(w))$. There are no other robots inside $\mathfrak{R}_{in}(E_{MEC}(w))$ and therefore $B_{in}(w)$ does not contain any other robot either.

According to GTRC, robot $w$ moves towards $T(w) \in K_{fr}(w)$. There are no robots at the points of $B_{in}(w)$ and in a small enough ball around it. Therefore, no other robot may approach $w$ from $B_{in}(w)$ arbitrarily close and eventually collide with $w$ without changing the Relative neighborhood graph during $[a, b]$. There might be some robots inside $B_{out}$, but robot $w$ moves with maximum constant speed 1 away from $B_{out}$ and therefore, according to [Sch95], even if some robots from $B_{out}$ pursue $w$ as a target point, they will only shorten the distance to $w$ but never reach it, since all other robots move with the same or lower speed.

Summing up all the considered cases, we can state that if $w \neq T(w)$ during the time interval $[a, b]$, then robot $w$ cannot collide with some other robot from $u \in R$.

Now let us assume that $w = T(w)$ during the time interval $[a, b]$. According to GTRC, the robot follows the movement of $T(w)$ and might move slower than 1. If $w = T(w) \in K_{in}(w)$ (see Figure 3.7f) during time interval $[a, b]$, then every point of the trajectory of

robot $w$ during $[a, b]$ is entirely inside $K_{in}(w)$ together with a small enough ball around this point. Therefore, in this case also no other robot may approach arbitrarily close to and eventually collide with $w$ without changing the RNG during $[a, b]$.

The last to consider is the case where $w = T(w) \in K_{fr}(w)$ as depicted in Figure 3.7g. The small enough ball $B(w)$ around the position of robot $w$ can be separated into two parts $B_{in}(w) = B(w) \cap K_{in}(w)$ and $B_{out} = B(w) \setminus (B_{in}(w) \cap K_{in}(w))$. According to the GTRC algorithm, the robot follows the movement of $T(w)$ and might move slower than 1. Thus, robots from $B_{out}(w)$ can approach arbitrarily close to and eventually collide with $w$ without changing the RNG during $[a, b]$.

The target point $T(w) \in K_{fr}(w)$ only if it is a midpoint between two robots in $MEC(w)$. According to [Chr85], this is the case only if the line segment is the diameter of the minimum enclosing circle $C(w)$. The considered midpoint is nothing but the crash point $p(w)$ we have defined earlier.

We have shown that collision occurs at the crash point $p(w)$ of robot $w$. It is left to show that $w \notin MEC(w)$. Assume that $w \in MEC(w)$. In this case, if $w$ reached its own target point $T(w)$, then the diameter of the minimum enclosing circle $C(w)$ will be zero. This contradicts our statement that the minimum enclosing circle of robot $w$ has a diameter greater than zero. Therefore $w \notin MEC(w)$. □

## 3.5 The Collisionless Conjecture for Four Robots

Lemma 14 implies that an early collision may take place with the GTRC algorithm only at the crash points. There is at most a linear number of crash points at every point of time. If we are going to

count the robots involved in the early collision, then there must be at least 4 of them.

**Observation 2.** *For the set $R$ of $n < 4$ robots, the gathering is always collisionless with the GTRC or GTC algorithm.*

Four is the smallest number of robots that allows early collision, since for $n < 4$ there is simply not enough robots for it. If we had an early collision at time $t_*$ then according to Lemma 14, there exists robot $u$ that reached its own crash point $p$ and there are at least two robots $m_1, m_2 \in MEC(u)$ that do not coincide with robot $u$, since $u$ does not belong to its own minimum enclosing set $MEC(u)$. For an early collision, we need one more robot $w$ that at time $t_*$ reached a crash point of $u$ as well. Therefore we need at least 4 robots for an early collision with the GTRC algorithm. It is easy to see that the same observation also holds for the GTC algorithm. However, the RNG lenses around the crash point make the difference between the behavior of the GTC and GTRC algorithms. We can already illustrate this difference by using only $n = 4$ robots.

The crash point $p$ of robot $u$ is the midpoint between the two robots $m_1$ and $m_2$. When robot $w$ is close enough to $p$, then the RNG lenses corresponding to the RNG edges $\{m_1, w\}$ and $\{m_2, w\}$ form a narrow passage to the crash point $p$ as depicted in Figure 3.8. If some other robot $u$ has an early collision with $w$ at $p$, then its trajectory shall be entirely inside this narrow passage. Otherwise, if $u$ enters one of the RNG lenses, $p$ will not be the crash point of $w$ anymore. Nevertheless, proving the Conjecture 1 turns out to be a hard task even for $n = 4$ robots.

Let us consider one of the possible approaches. The main idea consists of two steps. The first step is to show that, shortly before an early collision, the set of a possible positions of one of the robots

is negligible, i.e. a set of measure zero on the Euclidean plane (i.e., line circle, etc.). The second step is to show that the flow related to the time-dependent vector field $V_r(t)$ (produced via the GTRC algorithm) of any robot $r \in R$ is the zero measure preserving in forwards and backward time.

As part of the first step, let us now consider the crash $p$ together with robots $m_1, m_2$. Assume that at time $t_*$ there was an early collision in $p$ between robots $u$ and $w$. Our aim is to show that during a short time interval $\delta$ before the collision, the positions of one of two robots, $u$ or $w$ belong to the set without an area.

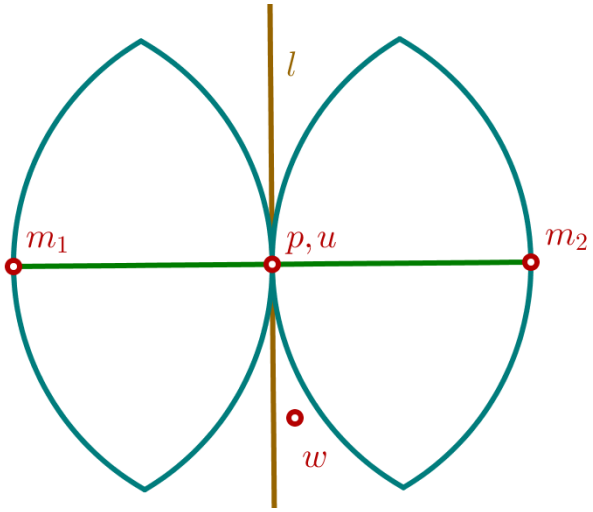



Figure 3.8: In this figure we have: robot $w$, the crash point $p$ of robot $u$ that coincides with the position of robot $u$, Relative neighbors $m_1$ and $m_2$ of robot $r$ with the according RNG lenses and line $l$ perpendicular to $m_1m_2$.

There are two cases to be considered. In the first case, both robots ($u$ and $w$) arrive at the crash point $p$ at the same time or one of the robots, w.l.o.g. $u$, is already at $p$. It is easy to see that in the second case, the only trajectory that does not cross the RNG lenses is the straight line $l$ perpendicular to the line segment $m_1m_2$ (see Figure 3.8). Line $l$ is a negligible set on the Euclidean plane.

Unlike the second case, the first one turns to be hard. Here it must also be show that the positions of both robots ($u$ and $w$)

that arrive at the crash point $p$ at the same time belong to the set without an area (i.e. a Jordan curve). One of the ways is to consider the travel time $\tau(x)$ as the function of the robot's initial position $x$. However, the function $\tau(x) : X \to p, X \subset \mathbb{R}^2$ that represents the travel time from point $x$ to the crash point $p$ is not well defined. In other words, we need some regularity properties of $\tau(x)$ in order to argue about the structure of level set $I_{t_c} = \{x \in \mathbb{R}^2 : \tau(x) = t_c\}$ of $\tau(x)$, using Implicit Function Theorem, for example. This is one of the open problems. Another open problem is to show that the flow related to the time-dependent vector field $V_r(t)$ (produced via the GTRC algorithm) of any robot $r \in R$ is zero measure preserving, i.e. on Euclidean plane such flow maps any set without an area to the set that has no area too.

# Chapter 4

# Collisionless Gathering with Some Algorithmic Extensions

So far the collisionless gathering was considered for robots with an extent, such as in [CGP06]. However, gathering itself in [CGP06] is redefined, since robots with extent are not allowed to occupy the same position. Instead, gathering means forming a configuration for which the union of all discs representing the robots is connected.

The problem that is closer to one that we consider in this work is studied in [CDF+11b]. The goal of the robots is to gather without touching, in such a way that disks representing the robots do not intersect. Robots gather around a *predefined* point that is already known by every robot.

The closest problem is considered in [PPV15]. The goal of the robots is to get close enough to each other without collisions so that every robot is in vision range of the others. Robots in [PPV15] are equipped with a compass so that they have a *common coordinate system*.

In this chapter, our aim is to show that robots with limited visibility are able to perform collisionless gathering without a predefined gathering point or common coordinate system. However, we would also need some additional capabilities in comparison to the robot

model used in Chapter 2.

## 4.1 Safe-Go-To-The-Relative-Center Algorithm

We extend the robot model and design the *Safe-Go-To-The-Relative-Center* algorithm (S-GTRC) using the contracting conditions from Lemma 2 and the structural properties from Lemma 14. The goal of S-GTRC is to gather without *early collisions* all the robots at one non-predefined point for any initial configuration. The initial configuration is arbitrary except that $UDG_0(R)$ is connected and all the robots have distinct positions.

The extended robot model is described as follows. The viewing range of a robot is 2. This will be needed to avoid collisions. Thus, a robot can see the UDG neighbors of its UDG neighbors. Note that S-GTRC preserves connectivity with respect to UDG. The open two-unit disk graph is defined analogously to the open unit disk graph. For all robots in $R$, we define 2-$UDG(R) = (R, 2\text{-}E_t)$, where $(r_i, r_j) \in 2\text{-}E_t$ iff for $r_i$ and $r_j$ it holds that $|r_i(t), r_j(t)| < 2$.

As in the common model, robots are anonymous and each robot has a local coordinate system that is not aligned with the coordinate systems of other robots. Unlike in the common model, robots here are chiral: i.e., they all agree either on left- or right-hand orientation. Robots are equipped with synchronized clocks. Robots are luminous: i.e., they have one bit of visible external memory like in [DFP$^+$12] by Das et al. The maximum speed of the robot is assumed to be $s \geq 1$.

Next, we describe S-GTRC and show that it performs gathering in the continuous time model for the extended robot model without early collisions in time $O(n^2)$.

The main idea of S-GTRC is described as follows. Robots are

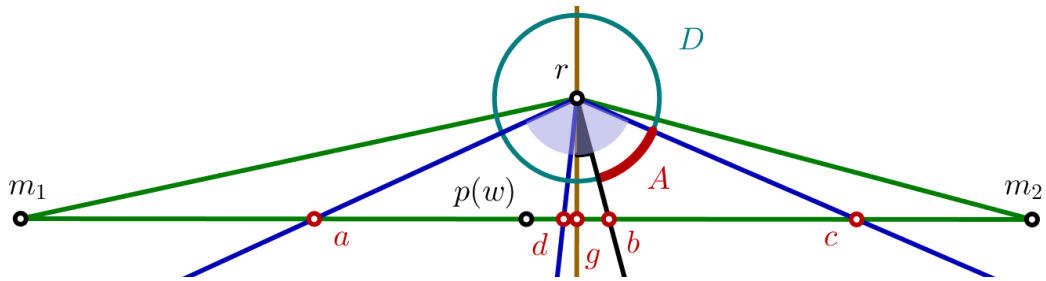



Figure 4.1: The construction of *target arc* $A \subset D$.

separated into two groups/states: *regular* and *safe*. In the regular group, robots execute GTRC and do not take into account the robots in the safe group. If some robots in the regular group are about to collide, at least one of them switches to the safe state and, independently from other robots, moves towards some specific, closely situated fixed point. This point is selected in such a way that collision is not possible. The state of the robot is automatically visible to its neighbors via the visible external memory.

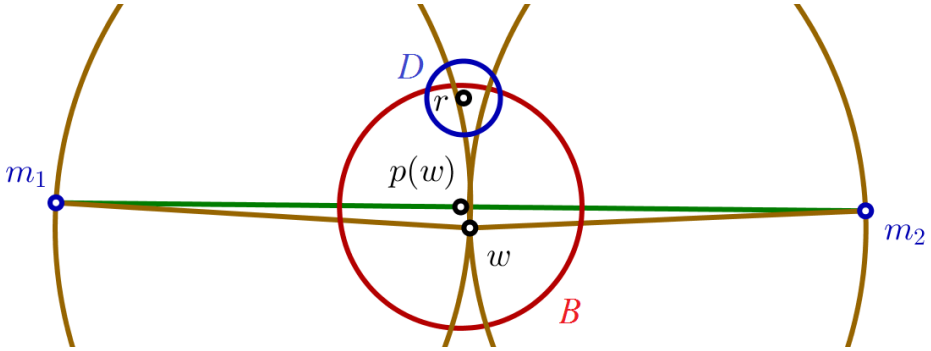



Figure 4.2: Robots $w$ with RNG lenses that correspond to RNG edges $\{w, m_1\}$ and $\{w, m_2\}$. Both robots are inside circle $B$ with the center at the crash point $p(w)$ and radius $1/2m|m_1, m_2|$. If robot $r$ performs a safe move, the target point during the safe move belongs to circle $D$.

From Lemma 14 we know that robot $r$ collides only at the crash point of some other robot $w$. Therefore, shortly before collision, both robots $r$ and $w$ together with their target points $T(w)$ and $T(r)$ are inside the relatively small ball $B$ with the center at the crash point $p(w)$ and radius $1/2m|m_1, m_2|$, where $m$ is a positive

parameter. Ball $B$ is depicted in Figure 4.2. In order to avoid collision, we let at least one of the robots move independently from the other robots. For a short period of time, the target point of the robot will be selected from the circle $D$ with the radius $1/k|w,r|$ centered at the position of $r$, where $k > 1$ is a positive parameter that will help us preserve connectivity. Circle $D$ together with $B$ is depicted in Figure 4.2. Circle $D$ is depicted in Figure 4.1. We refer to the arc $A$ of $D$ as the *target arc*. In the safe state, robots move towards the midpoint of the target arc. We construct the target arc in such a way that robots in the safe state satisfy the contracting conditions of Lemma 2. This is shown in Lemma 16.

The target arc is constructed as follows. Let us first consider arc $B \subset D$ such that the central angle $\angle arc = 3\pi/4$ and the bisector of $\angle arc$ coincide with the bisector of $\angle m_1 r m_2$. Then, we draw the line $rg$ perpendicular to $m_1 m_2$. With respect to this line, we either take left or right, depending on the chirality part of $B$, i.e. $B_L \subset B$. Finally we subtract from $B_L$ the arc that corresponds to the central angle $\angle grb = \pi/20$. What is left is *target arc* $A$, which we depict in Figure 4.1.

In order to select the suitable moment for the independent motion of robot $r$, we check whether $r$ is close to the crash point $p(w)$ of some robot $w$. Namely, robot $r$ checks if $\exists w : w, T(w), T(r) \in B_{1/2m|m_1,m_2|}(p(w))$, where robots $m_1, m_2$ define $p(w)$ and $|w,r| = \min_{u \in UDG(r) \setminus r}\{|r,u|\}$, where $UDG(r)$ consists of robots in both states (regular and safe) and $|w,r| \leq \rho_Q$ and $r$ is the leftmost (rightmost, depending on chirality) robot with respect to the direction towards $p(w)$.

We show in Lemma 17 that there exists a point in time at which this condition is satisfied, shortly before collision takes place. We refer to the logical expression above as the *safety condition*, denoted

by the function $\mathfrak{S}_r : X \to \mathbb{Z}_2$, where $X \subset \mathbb{R}^2$: i.e., $\mathfrak{S}_r(w) = true$ means that robot $w$ at its current position violates the safety condition with respect to the crash point $p(w)$ of robot $w$.

At every point in time $t$, each robot can read the positions and states of its neighboring robots in 2-$UDG(R)$. Every robot has access to the synchronized clock. Besides that, each robot can read and write into the two variables. These will be used to store either the position on the Euclidean plane, denoted by $M(r)$, or the point in time $\Delta(r)$ together with an additional state $L(r)$. The variable of robot $r$ that represents its state is called $S(r)$. This is set by default to *regular*. Our algorithm has three positive parameters, used by all of the robots: $s$, $m$ and $k$. Parameter $m$ defines how close to the crash point we check the safety condition. Parameter $s$ tells us the ratio between the speed of the robots in the safe and regular state. Robots in the safe mode are assumed to be faster. Parameter $k$ tells us what portion of the minimum distance to other robots does the robot cover during the motion in the safe state. The initial state at time 0 of every robot is regular. The initial value of the memory slots that correspond to $M(r)$, $\Delta(r)$ and $L(r)$ are *undefined*. The Safe-Go-To-The-Relative-Center algorithm is presented in pseudo-code as Algorithm 1.

The main difference between S-GTRC and GTRC lies in the states: regular and safe. In regular state, a robot moves according to GTRC. Early collisions may take place only in the regular branch of the algorithm. The safe branch is designed to avoid early collisions. The collision that takes place in S-GTRC without a safe branch is called a *potential* early collision. Next, we show that the safe branch avoids potential early collisions.

**Algorithm 1** Safe-Go-To-The-Relative-Center

**Require:** Initial configuration, parameters $m$ and $k$, velocity $s$.

Robot $r$ observes the positions of all its neighbors (*regular* and *safe*) in 2-$UDG(r)$ and checks:
**if** $\max_{a,b \in 2\text{-}UDG(r)} |a,b| \geq 1$ **then**
  Robot $r$ observes the positions of its *regular* neighbors in $UDG(r)$ and calculates $RNG(r)$.
  Robot $r$ computes the minimum circle $C(r)$ enclosing $RNG(r)$. The center $T(r)$ of $C(r)$ is the *target point* of $r$.
  Robot $r$ observes the positions of all *regular* robots in 2-$UDG(r)$. Using this information $r$ calculates crash points for every robot in $UDG(r)$. Then robot checks the following *safety* condition:
  Robot $r$ checks:
  **if** Robot $r$ is at its own crash point, i.e. $r = p(r)$ AND $\exists w : \mathfrak{S}_w(r) = true$ AND $S(w) = safe$ **then**
    $L(r) \coloneqq locked$
    $S(r) \coloneqq regular$
    $\Delta(r) \coloneqq \frac{|r,M(r)|}{s} + \#clock$
  **else**
    **if** $\exists w : \mathfrak{S}_r(w) = true$ **then**
      **if** $M(r) = undefined$ **then**
        $S(r) \coloneqq safe$
        $M(r) \coloneqq$ midpoint of *target arc* $A \subset D$, where $D$ is the circle with center at $r$ and radius radius $\frac{1}{k}|w,r|$.
    **else**
      $S(r) \coloneqq regular$
  **if** $L(r) = locked$ AND $\#clock = \Delta(r)$ **then**
    $S(r) \coloneqq undefined$
    $\Delta(r) \coloneqq undefined$
  Robot $r$ moves:
  **if** $S(r) = regular$ **then**
    **if** $r$ is already at $T(r)$ **then**
      Robot $r$ remains at $T(r)$ and moves in the same way as the target point does.
    **else**
      Robot $r$ moves with maximum speed 1 towards $T(r)$.
  **else**
    **if** $r$ is already at $M(r)$ **then**
      $S(r) \coloneqq regular$
      $M(r) \coloneqq undefined$
    **else**
      Robot $r$ moves with maximum speed $s$ towards $M(r)$.

### 4.1.1 Correctness and Runtime Analysis of the Safe-Go-To-The-Relative-Center Algorithm

First, we show that in the *safe* state, the robots preserve connectivity. Then, we consider the runtime and show that the robots in the safe state move only inside the convex hull, with speed $s$. In Lemma 15, we show that if parameter $k$ is big enough (i.e., the radius of circle $D$ is small enough), then at the end of the independent motion in the safe state, the robot will still have the same unit disk graph neighbors.

**Lemma 15.** *If robot $w$ is in $UDG(r)$ at time 0, when robot $r$ switches to safe state, then $w$ is still in $UDG(r)$ at time $t > 0$, when robot $r$ switches back to regular state.*

*Proof.* Let us consider the motion of the robot $r$ in the *safe* state. Assume that the safe state of some robot $r$ is triggered at time 0. This state is preserved until $t' = 1/k\,|w(0), r(0)|$ only if $|w(t), r(t)| \leq \rho_Q(t), t \in [0, t']$. The robot $r$ moves with speed $s$ towards midpoint $M(r)$ of the target arc $A$ on the circle with radius $1/k\,|w(0), r(0)|$ centered at $r(0)$.

Robot $r$ requires at least a time interval of length $1/k\,|w(0), r(0)| \leq 1/k\,\rho_Q(0)$ in order to reach $M(r)$. The motion of the other robots does not depend on the motion of $r$.

The connectivity circle $C_Q(r)$ cannot change much during the time interval $[0, t']$. All other robots move at most with speed $s$ and are able to cover at most the same distance as $r$ during $[0, t']$. Therefore it holds that $\rho_Q(t') \geq \rho_Q(0) - 2/k\,\rho_Q(0)$. Note that it does not matter if the robot $r$ reaches $M(r)$ or the criterion that triggers safe state does not hold anymore. If the criterion that triggers safe state is violated, then robot $r$ becomes regular earlier than $t'$.

For $k > 2$ it holds that $\rho_Q(t') > 0$ and thus any unit disk graph edge that robot $r$ had at time 0 will remain at the end of the safe motion. □

From Lemma 10 and Lemma 15 we can conclude that S-GTRC preserves connectivity.

**Corollary 5.** *Let us consider a group of robots $R$ on the Euclidean plane executing S-GTRC. If $\{u, w\}$ is an edge in the open Relative neighborhood graph $RNG(R)$ at time 0, then $\{u, w\}$ is an edge in $RNG(R)$ at $\forall t \geq 0$ or there is a path from $u$ to $v$ in $RNG(R), \forall t \geq 0$.*

Next, we analyze the runtime. We know that the robots in the regular state (i.e., those executing GTRC) satisfy the contracting conditions of Lemma 2. It remains to show that in the safe state the robots also satisfy the contracting conditions. In Lemma 16, we show that the target arc is constructed in such way that its middle point is always inside the local and consequently also a global convex hull.

**Lemma 16.** *If robot $r$ is in the safe state, then the target point $M(r)$ is inside the convex hull $H(R)$ over all robots, and it does not coincide with the position of the robot $r(t)$ at least until it switches into the regular state again.*

*Proof.* Let us consider robot $r$ that switches to the safe state at time 0. Robot $r$ comes back into the regular state again at time $t_1$. It is clear that the position of robot $r$ does not coincide with the target point $M(r)$ during the time interval $[0, t_1)$. It is left to show that the target point $M(r)$ of robot $r$ during the safe run lies inside $\triangle m_1 r m_2$, where robots $m_1, m_2$ form $p(w)$.

During the safe run, the motion of robot $r$ does not depend on the motion of other robots and vice versa. Thus we will first consider

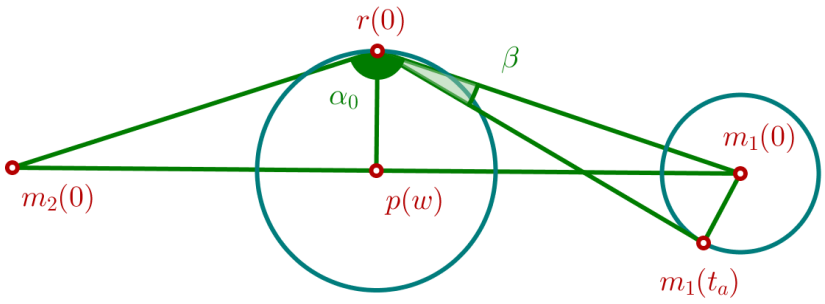



Figure 4.3: Robot $r$ at the position inside $B_{1/2m|m_1,m_2|}p(w)$ that minimizes angle $\alpha$ at time 0. On other hand, position $m_1(t_1)$ gives us a maximum decrease of the angle $\alpha$ at time $t_1$.

the motion of robots $m_1$ and $m_2$ independently from the motion of robot $r$ during the time interval $[0, t_1)$. Next, we show that $\angle m_1 r m_2$ cannot become too small during the safe run.

Let us denote $\angle m_1 r m_2$ at time 0 by $\alpha(0)$. Angle $\alpha(0)$ is at its smallest if position $r(0)$ is at the furthest position from line segment $m_1 m_2$ as depicted in Figure 4.3. In order to save space let $l = |m_1(0), m_2(0)|$. Since $r$ is in the safe state it holds that $|r(0), p(w)| \leq l/2m$. We consider $\triangle r(0) m_1(0) p(w)$, where it holds that $\cot(\alpha(0)/2) \leq 1/m$, thus $\alpha(0) \geq 2\,\mathrm{arccot}(1/m)$. If $\alpha(t_1)$ is $\angle m_1 r m_2$ at time $t_1$, then $\alpha(t_1) \geq \alpha(0) - 2\beta$, where $\beta$ is the maximum decrease of $\alpha(0)$ caused by the motion of $m_1$ and $m_2$. To bound $\beta$ we use first triangle inequality in $\triangle r(0) m_1(0) p(w)$, i.e $|r(0), m_1(0)| \leq l/2 + l/2m$ and then some trigonometry $\tan(\beta) \leq l/2mk\,(l/2 + l/2m)^{-1}$, thus $\beta \leq \arctan\left((mk\,(1 - 1/m))^{-1}\right)$. Combining our bounds on $\alpha(0)$ and $\beta$ we can show that $\alpha(t_1)$ is greater than some function dependent on $m$ and $k$, namely

$$\begin{aligned} \alpha(t_1) \geq \alpha(0) - 2\beta \geq \mathrm{arccot}\,(1/m) + \\ + \arctan\left((mk\,(1 - 1/m))^{-1}\right). \end{aligned} \tag{4.1}$$

For $k \geq 3$ and $m \geq 4$ the angle $\alpha(t_1) \geq \frac{3\pi}{4}$. According to our algorithm, target point $M(r)$ in the safe state is the midpoint of

the target arc $A$ and $A$ is a subset of arc $B \subset D$ such that central angle $\angle arc = {}^{3\pi}/_{4}$ and bisector of $\angle arc$ coincides with bisector of $\angle m_1 r m_2$. Therefore, if we select $m \geq 4$ and $k \geq 3$, then $M(r)$ is inside $\triangle m_1 r m_w \subset H(R)$. □

Lemma 16 implies that in the safe state any robot $r$ moves with speed $s \geq 1$ inside the convex hull. This means that S-GTRC is a contracting algorithm. Besides that, the length of the convex hull boundary around the initial configuration is not greater than $2(n-1)$, where $n$ is a number of robots, as shown in [LMP16].

**Theorem 7.** *The group of n robots executing S-GTRC gathers in time $O(n^2)$.*

### 4.1.2 Collisionless Property of the Safe-Go-To-The-Relative-Center Algorithm

Next, we investigate the collisions of robots executing S-GTRC. Robots that execute S-GTRC can be in one of the three states: regular, locked (regular) or safe, where a locked robot is a robot $r$ that is positioned at its own crash point $p(r)$ and $\exists w : \mathfrak{S}_w(r) = true$. It actually is in the regular state but the presence of $w$ in the proximity of its $p(r)$ prevents $r$ from switching to the safe state.

**Proposition 6.** *If robot r is in the safe state, then it does not collide with any other robot.*

This proposition holds since according to S-GTRC, during the motion in the safe state, a robot covers at most a distance ${}^{1}/_{k}|w, r| = {}^{1}/_{k} \min_{u \in UDG(r) \setminus r}\{|r, u|\}$. Due to the speed limit $s$, all other robots during the safe motion of $r$ can cover at most the same distance. For $k \geq 3$, the position of $r$ will never coincide with the position of some other robot. Similar argumentation works in Proposition 7 for the locked regular state.

**Proposition 7.** *If robot $r$ is in the locked state, then it does not collide with any other robot.*

*Proof.* If robot $r$ is locked, then its position coincides with the position of its crash point and there is a robot $w$ close to $r$ such that $\mathfrak{S}_w(r) = true$ and $w$ is in the safe state.

Assume that there is some other robot $u$ close to $r$. If $u$ is close enough it is also in the $UDG(w)$ and since $\mathfrak{S}_w(r) = true$ it holds that $|u, r| \geq |w, r| = \min_{u \in UDG(r) \setminus r}\{|r, u|\}$. Assume that robot $w$ switches to the safe mode at time $t_1$, at most at time $t_2$ it becomes regular again and changes the RNG graph such that crash point $p(r)$ does not exists after $t_2$ (we will show this in Lemma 18). If our assumption holds, then $r$ after time $t_2$ is not locked.

The safe mode of $w$ lasts for $|t_1, t_2| \leq 1/ks|w, r|$. Robot $r$ stays locked until $t_2$. During this time interval, robot $u$ can cover at most distance $1/k|w, r|$. During the same time robot $r$ covers at most $1/ks|w, r|$. Combining this together we can see that $|w(t_2), r(t_2)| \geq |w(t_1), r(t_1)|(1 - 1/ks - 1/k)$, And for $k \geq 3$, $s \geq 1$ it holds that $(1 - 1/ks - 1/k) > 0$. Thus $r$ cannot collide with some other robot $u$ while being locked, since both robots always move just a part of the minimum distance needed to be covered before the collision.

If $w$ is also locked and it cannot turn to the safe state, then there is $w_1$ such that $\mathfrak{S}_{w_1}(w) = true$. Note that the locked condition overrides the safe state and brings the robot back to a regular one. Since $w \in B_{1/m|m_1,m_2|}(p(r))$, where $m_1, m_2$ define the crash point $p(r)$ it holds that $|w, w_1| \leq 2/m|w, r|$. Using the similar arguments as before we can state that $|w(t_2), r(t_2)| \geq |w(t_1), r(t_1)|(1 - 2/ksm - 2/km)$. Furthermore, for $k \geq 3$, $s \geq 1$, $m \geq 2$ it holds that $(1 - 1/ks - 1/k) > 0$. We can continue our consideration. Such a chain of locks can be at most length $n$. However, $r$ cannot collide with some other robot $u$ while being locked since they will always move

just a part of the minimum distance needed to be covered before the collision.

Note that if some robot $w_0$ is locked by $w_1$, $w_1$ by $w_2$ etc., then $w_i$ is also locked $w_j$ for any $j > i$. But $w_i$ cannot be locked by $w_j$ for any $j < i$ since:

$$B_{1/m|m_{1,i+1},m_{2,i+1}|}(p(w_{i+1})) \subset B_{1/m|m_{1,i},m_{2,i}|}(p(w_i)), \qquad (4.2)$$

where $m_{1,i}, m_{2,i}$ and $m_{1,i+1}, m_{2,i+1}$ define according crash points $p(w_i)$ and $p(w_{i+1})$. □

In order to have a collision, according to Corollary 4, robot $r$ needs to reach the crash point $p(w)$ of some other robot $w$. An example of the disposition of robots $r$ and $w$ shortly before a collision at $p(w)$ is depicted in Figure 4.2.

Robot $r$ needs to pass through a shrinking gap between two lenses corresponding to the RNG edges between $w$ and $m_1, m_2$. Next, in Lemma 17, we show that the safe move always triggers before the potential collision by carefully analyzing the safety condition. In Lemma 18, we show that at the end of the motion in the safe state, a robot avoids potential collision by showing that during the run, a robot cannot reach the crash point of any other robot.

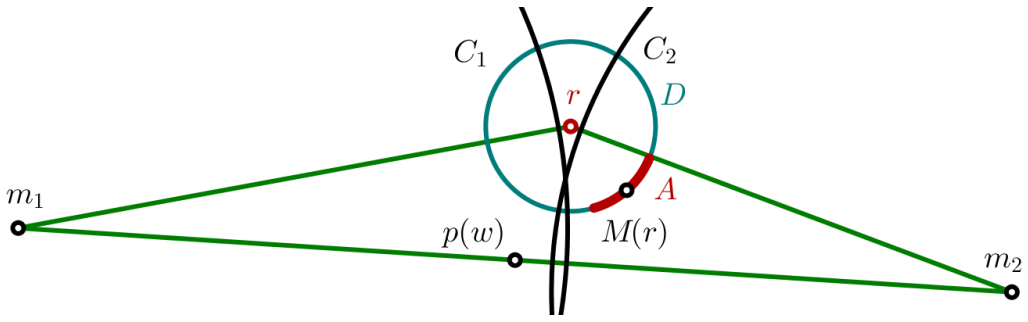



Figure 4.4: Robot $r$ in the safe state moves towards target point $M(r)$ the midpoint of the target arc $A \subset D$. Black arcs $C_1$, $C_2$ are the parts of according RNG lenses.

**Lemma 17.** *If a set of robots $M \subset R$ has a potential collision with robot $w$ at time $t_*$ in $p(w)$, then there exists robot $r \in M$ and a point in time $t < t_*$ such that $\mathfrak{S}_r(w) = true$.*

*Proof.* Assume that a set of robots $M$ arrives to the crash point $p(w)$ together with robot $w$ at time $t_*$. If our assumption holds, then there exists $r \in M$ and $t < t_*$ such that the safety condition is true, namely $\exists w : w, T(w), T(r) \in B_{1/2m|m_1,m_2|}p(w)$, where robots $m_1, m_2$ define $p(w)$ and $|w, r| = \min_{u \in UDG(r)\setminus r}\{|r, u|\}$, where $UDG(r)$ consists of robots in both states (regular and safe) and $|w, r| \leq \rho_Q$ and $r$ is leftmost (rightmost, depending on chirality) robot with respect to the direction towards $p(w)$. Let us now explain why.

Robots follow their target points, therefore in order to reach $p(w)$ the target point $T(r)$ of any robot $r \in M$ and target point $T(w)$ need to be at least inside the ball $B_{1/2m|m_1,m_2|}(p(w))$. Otherwise, robots will move outside this ball.

If robots in $M$ and $w$ collide, then for any robot $r \in M$ the limit of the distance $|w, r|$ is zero as time approaches $t_*$. Besides that, for any $r \in M$ it holds that $|w, r| = 0$ at time $t_*$. Therefore, due to continuity there shall be an interval $[t_1, t_*]$ where $|w, r|$ monotonically decreases up to 0.

Lets us now show that there exists $r \in M$ and $t < t_*$ such that $|w, r| = \min_{u \in UDG(r)\setminus r}\{|r, u|\}$ during $[t, t_*]$. We split the set as follows: $UDG(r) \setminus r = A \cap B$, where $A \subset M$ and $B \not\subset M$. The robots in $B$ do not take part in the collision. Let $\epsilon$ be the shortest distance from $r$ to any of the robots in $B$ during the time interval $[0, t_]$. Due to monotonicity there exists $r \in M : |w, r| = \min_{u \in B} |w, u|$ and some point of time $t \in [t_1, t_*)$ such that $|w, r| \leq \epsilon$.

There also exists a point in time $t \in [0, t_*)$ such that expression $|w, r| \leq \rho_Q$ is true. On one hand for any $r \in M$ the distance $|w, r|$

monotonically decreases on $[t_1, t_*]$. On the other hand, according to Lemma 10 $\rho_Q > 0$ during $[t_1, t_*]$, otherwise the connectivity property does not hold. This implies that there exists $r \in M$ and $t \in [t_1, t_*)$ such that $|w, r| \leq \rho_Q$.

Eventually, we can locally check the last expression of safety condition at any point in time. Namely, robot $r \in M$ checks whether it is a leftmost (rightmost, depending on chirality) robot with respect to the direction towards $p(w)$. There can be more than one robot that satisfy all previous condition, but the leftmost (rightmost) will be the unique one. □

**Lemma 18.** *If a set of robots $M \subset R$ has a potential collision with robot $w$ at time $t_*$ in $p(w)$, then there exists $r \in M$ such that the safety condition is triggered by $p(w)$ at time $t_1 < t_*$ and for $k = 4$, $m \geq 500$, $s \geq 10$ at the end of the safe motion at time $t_2 \in (t_1, t_*)$ the crash point $p(w)$ does not exist.*

*Proof.* Assume that there is a potential collision between robot $w$ in $p(w)$ and a set of robots $M \subset R$ at time $t_*$. Lemma 14 tells us that robots in the regular state collide only at the crash points of some other robots. Let us take a look at Figure 4.4 where we illustrate the position of some robot $r \in M$, $m_1$, $m_2$ and crash point $p(w)$ shortly before the potential collision at this crash point. Let $[0, t_*)$ be a short interval before the collision.

With respect to the position of robot $w$, there are two cases to be considered. Either $r$ is at the crash point $p(w)$ during the time interval $[0, t_*)$, or it is not. Let us consider the first case.

If $m_1$ or $m_2$ are in the safe state during $[0, t_*]$, then they are not taken into account by any $r \in M$ and $w$ for the calculation of target point in the regular state. However, if our assumption holds, then according to Lemma 14 there shall be some other robots $m_1'$

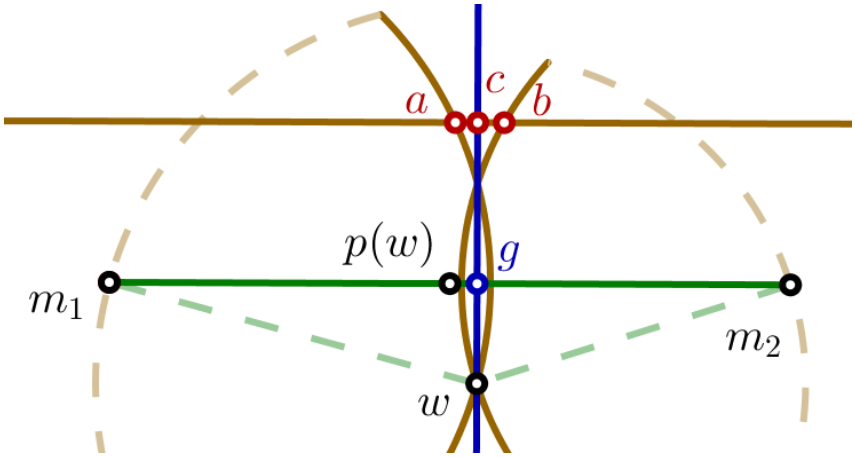
and $m_2'$ such that $p(w)$ is the midpoint between them. Thus during some positive interval $[0, t_*]$ robots $m_1, m_2$ and $w$ are in a regular state.

From Lemma 17, we know that there exist $r \in M$ and $t_1 \in [0, t_*)$ where robot $\mathfrak{S}_r(w) = true$. However, if $w = p(w)$ and $\exists r : \mathfrak{S}_r(w)$, then robot $w$ is locked. Locked robot $w$ stays in the regular state until robot $r$ finishes his independent motion in the safe state. We take a look once again at Figure 4.4. Let robot $r \in M$ be the one with $\mathfrak{S}_r(w) = true$. Assume that in Figure 4.4 the position of robot $r$ is depicted at time $t_1$. The motion of robots $m_1, m_2$ and $w$ does not depend on the motion of $r$. Assume that the positions of $m_1, m_2$ and $w$ are depicted in Figure 4.4 at time $t_2$ where robot $r$ becomes regular again.

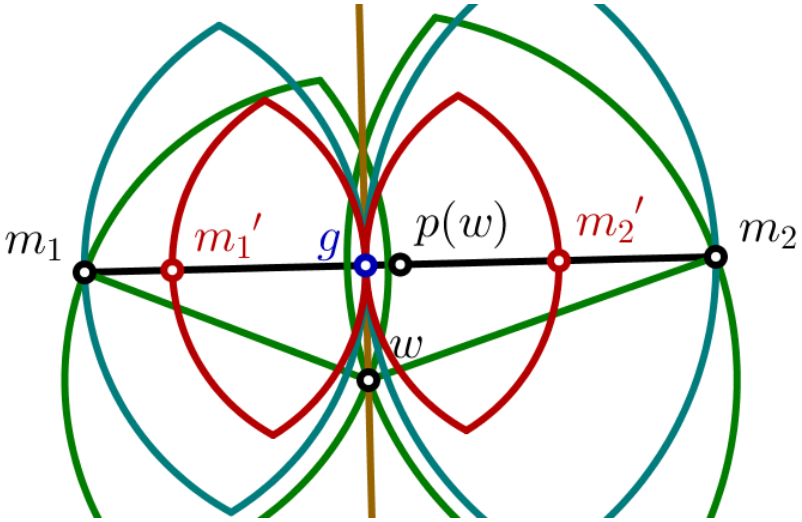
Robot $r$ starts an independent motion in the safe state at time $t_1$. It goes with speed $s$ towards the target point $M(r)$, which is the midpoint of target arc $A$. Next we show that $A$ is inside the left (or right) RNG lens, depending on chirality. Because of that, as robot $r$ turns back to regular state again at time $t_2$, $p(w)$ is not the crash point of $w$ since $w$ and $m_1$ are not RNG neighbors at time $t_2$ due to $r$.

Let us take a look at Figure 4.5a. Point $c$ is on the line that passes through $g$ perpendicular to $m_1m_2$. The distance to the line segment $m_1m_2$, i.e. $|c, g|$, we call the *height* over the crash point $p(w)$. The length of the line interval $ab$ we call the distance between the lenses at the height $|c, g|$.

Now let us take a look at Figure 4.5b. This figure illustrates how we can replace all given dispositions of $m_1, m_2$ and $w$ at the end of the safe motion of $r$ by the new bad one. By bad we mean that in the new disposition the distance between the lenses at any heights is greater than in the original one.




(a) Here we illustrate robot $w$ with its RNG lenses near the crash point $p(w)$. We call the length of the line segment $cg$ the height over the crash point and we call the length of the line segment $ab$ the distance between the RNG lenses at height $|c, g|$.




(b) Here we illustrate three steps needed to obtain the worst disposition. From the original (green lenses) disposition we move $w$ to $g$ (blue lenses) and then make RNG edges as short as possible (red lenses).

Figure 4.5: Disposition of the RNG lenses.

First we move robot $w$ to $g$. Since $|g, m_1| \leq |w, m_1|$ and $|g, m_2| \leq |w, m_2|$, the distance between the lenses at any height is greater. Then we move $m_1$ and $m_2$ to the new positions $m_1', m_2'$ as close as possible to $g$. Now we consider the bad disposition.

Let $l(t) = |m_1(t), m_2(t)|$, then the shortest length of RNG edge $\{m_1, w\}$ at time $t_1$ is $l(t_1)/2 - l(t_1)/2m$. Thus for the distance between $m_1$ and $m_2$ in the bad disposition it holds that $l'(t_1) = 2\,(l(t_1)/2 - l(t_1)/2m)$. During the safe motion of $r$, robots $m_1$ and $m_2$ may move towards each other, so that $l'(t_2) \geq 2\,(l(t_1)/2 - l(t_1)/2m) - 2/k|w(t_1), r(t_1)|$. If robots $m_1$ and $m_2$ do move towards each other, then we obtain *the worst* disposition, because then the distance between the RNG lenses in it for any height is greater than in any other disposition at time $t_2$.

The left part of the worst disposition is depicted in Figure 4.6. The right part is symmetric. It is easy to see that in the worst disposition the distance between $m_1'$ and $m_2'$ is bounded by $l'(t_2) \geq l(t_1)\,(1 - 1/m - 4/km)$. The distance between $r$ and line segment $m_1'm_2'$ at time $t$ we call $x(t)$. Since $\mathfrak{S}_r(w) = true$ it holds that $x(t) \leq l(t)/m$ where $t \in [t_1, t_2]$.

Since robot $r$ moves in the safe state in the direction of $m_1m_2$, it holds that $x(t_2) \leq x(t_1) + 1/kx(t_1)$. If we take into account that $x(t_1) \leq l(t_1)/m$ we get $x(t_2) \leq l(t_1)1/m\,(1/k + 1)$.

Let us now calculate the upper bound on the distance between two lenses at height $x(t_1)$ in the worst configuration. Namely, we need to upper bound $|a, c|$ depicted in Figure 4.6. In this picture $|c, g| = x(t_1)$ and $|m_1, g| = |m_2, g| = l'(t_2)/2$. We express the needed value as follows: $|a, c| = |m_1', g| - |m_1', b|$.

Let us begin with finding the angle $\theta$:

$$\sin(\theta) = \frac{|a, b|}{|m_1', b|} \leq \frac{2x(t_2)}{l'(t_2)} \leq \frac{2(1 + \frac{1}{k})}{m(1 - \frac{1}{m} - \frac{4}{mk})}. \qquad (4.3)$$

Now we can bound $|a, c|$:

$$|a, c| = \frac{l'(t_2)}{2} - \frac{l'(t_2)}{2}\cos(\theta), \tag{4.4}$$

$$\begin{aligned} |a, c| \leq \frac{l(t_1)}{2}\left(1 - \frac{1}{m} - \frac{4}{mk}\right) \cdot \\ \cdot \left(1 - \cos\left(\arcsin\left(\frac{2(1+\frac{1}{k})}{m(1-\frac{1}{m}-\frac{4}{mk})}\right)\right)\right). \end{aligned} \tag{4.5}$$

Let us denote the expression on the right by $\phi(m, k)$, then $|a, c| \leq {}^{l(t_1)}/_2\ \phi(m, k)$. The distance between the lenses $h(t_2)$ at height $x(t_1)$ for any disposition of $m_1, m_2$ and $w$ at time $t_2$ is bounded by $h(t_2) \leq 2|a, c| \leq l(t_1)\phi(m, k)$.

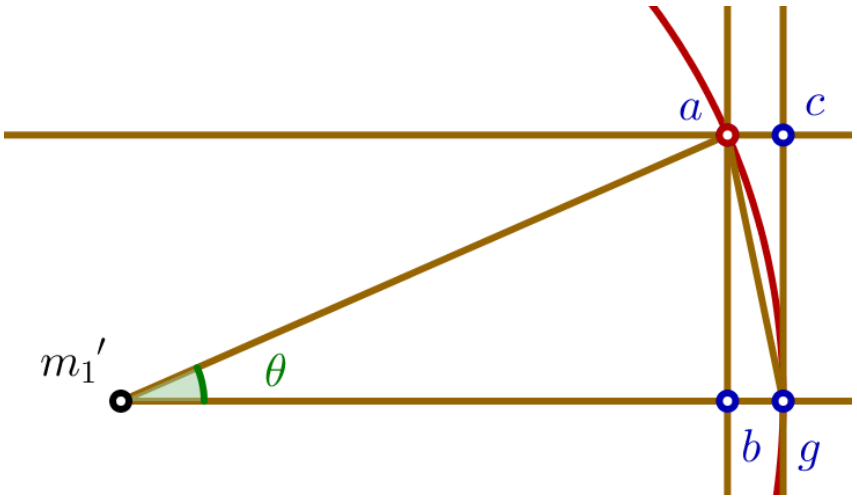



Figure 4.6: The worst disposition.

Next we consider target arc $A$ in details. But how do we know that circle $D$ depicted in Figure 4.4 intersects the lenses? So far in Lemma 16 and in Proposition 6 we have required $k \geq 3$. Now we fix $k = 4$. We consider the worst disposition again in Figure 4.6. If circle $D$ intersects the lenses in the worst disposition, then it intersects the lenses in any disposition that we consider. Let us

assume that $g$ is an origin, $gc$ is an $X$ axis and $bg$ is a $Y$ axis. If we let $|m_1', g| = 1$ then the boundary of the RNG lens is expressed by the function $\mu(x) = 1 - \sqrt{1 - x^2}$. The radius of the circle $B$ is less ore equal to the function $\rho(x) = {}^{x}/_{k} = {}^{x}/_{4}$. If $\mu(x) \leq 2\rho(x)$ then the circle $D$ intersects the RNG lenses. This holds for $x = {}^{16}/_{65}$. Thus, if ${}^{16}/_{65} \leq {}^{1}/_{m}$, i.e. $m \geq 5$, circle $D$ intersects the RNG lenses.

Let us now take a look at Figure 4.7. The locked robot $w$ moves $s$ times slower than robot $r$ in the safe state. Due to this we can show that there are arcs of $D$ – e.g., $yg$, $g'y'$ that stay inside the RND lenses $C_1$ and $C_2$. Namely, in Figure 4.7 the line through $yy'$ is parallel to the line segment $m_1m_2$ where $p(w)$ is the midpoint. Point $c$ depicts the position of robot $r$ at time $t_1$. Lines $l_1, l_3$ show us where $D$ intersects with the RNG lenses: $C_1, C_2$ in this case. Line segments $ab$, $b'a'$ have a length equal to the distance between the lenses at time $t_2$. Line segments $bc$, $cb'$ have a length equal to the distance that robot $w$ can cover during the safe motion of $r$.

At time $t_1$, robot $r$ was at some point in the gap between lenses $C_1$ and $C_2$. As robots $w$, $m_1$ and $m_2$ move the relative position of lenses with respect to circle $D$ with center $c$ might change. However, the gap between $C_1$ and $C_2$ lower than line segment $yy'$ is always between lines $l_1$ and $l_3$. In other words, arcs $yg$ and $g'y'$ shall be inside the RNG lenses.

Let us now calculate how the size of these arcs depends on parameters $m$, $k$ and $s$. Let us consider $\triangle cgx$ in Figure 4.7. Let us bound from above angle $\gamma$. Let us consider $\triangle cgx$ where $\sin(\gamma) = {}^{|g,x|}/_{|c,g|}$. Here $|c, g|$ is the radius of the circle $D$ and $|g, x| = |a, c|$ is the length of the line segment between $y$ and $c$ that can be still in the gap, i.e., not inside the RNG lenses.

Robot $r$ reaches with speed $s$ any point on $D$ in time $\frac{1}{ks}|w(t_1), r(t_1)|$, therefore robot $w$ can cover at most distance $\frac{1}{ks}|w(t_1), r(t_1)|$ with

speed 1. The length cathetus $gx$ is bounded from above by the distance between the lenses and the distance that robot $w$ can cover during $[t_1, t_2]$ as follows:

$$|g, x| \leq h(t_2) + \frac{1}{ks}|w(t_1), r(t_1)|. \tag{4.6}$$

Since $w$ and $r$ are relatively close, i.e., both robots are inside the $B_{1/2m|m_1,m_2|}p(w)$, we can write that:

$$|g, x| \leq h(t_2) + \frac{1}{ksm}l(t_1), \tag{4.7}$$

where $l(t_1)$ is the distance between robots $m_1$ and $m_2$.

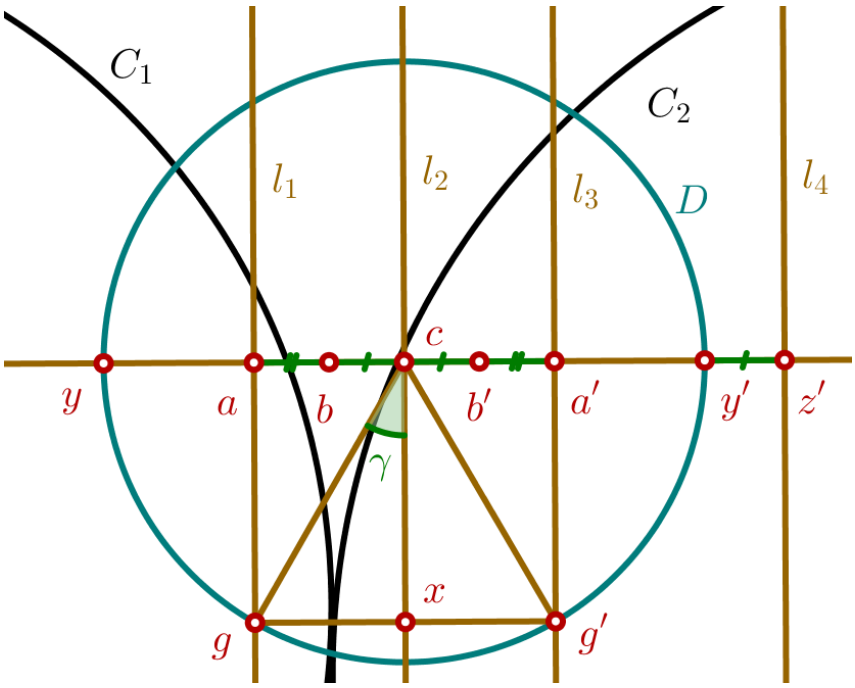



Figure 4.7: Here we illustrate the relative position of robot $r$ and the RNG lenses during the motion of robot $r$ in the safe state. Point $c$ here depicts the position of robot $r$ at the begin of the safe motion. The end position of the safe motion is on circle $D$. Lines $l_1, l_2, l_3, l_4$ show us where $D$ intersects with RNG lenses $C_1, C_2$.

Radius of the $D$ is $1/k|w(t_1), r(t_1)|$. Robots according to the algorithm perform calculation at every point in time. Due to this;

as soon as the safety condition is fulfilled the robot turns to the safe state. Due to the absence of delay we can say that $l(t_1)/2mk \leq 1/k|w(t_1), r(t_1)| \leq l(t_1)/mk$. Using this bound together with previous results we can bound $\sin(\gamma)$ as follows:

$$\sin(\gamma) \leq \frac{2mk\left(h(t_2) + \frac{1}{ksm}l(t_1)\right)}{l(t_1)}. \tag{4.8}$$

As we simplify this we get $\gamma \leq \arcsin\left(2mk\phi(m,k) + 1/s\right)$. This bound decreases as $m$ and $s$ grow and increases with the growth of $k$. For $s \geq 10$, $m \geq 500$ and $k = 4$ it holds that $\gamma < \pi/20$. Note that selected parameters also satisfy restrictions in other statements: $m \geq 4$, $k \geq 3$ in Lemma 16 and $m \geq 2$ $k \geq 3$, $s \geq 1$ in Proposition 7.

We use this bound on $\gamma$ together with the results of Lemma 16 in the construction of the target arc depicted in Figure 4.1. Due to this construction, the target point $M(r) \subset A$ of $r$ during the safe motion of $r$ is strictly inside the corresponding RNG lens. At the end of the safe motion crash point $p(w)$ does not exist since $r$ violates Relative neighborhood criterion between $w$ and of two robots $m_1$, $m_2$ that form the crash point.

Recall that, with respect to the position of robot $w$, there are two cases to be considered. Either $r$ is at the crash point $p(w)$ during the time interval $[0, t_*)$, or it is not. Let us consider the second case.

We assume again that there is a collision in the crash point $p(w)$ between robot $w$ and a set of robots $M \subset R$ at time $t_*$. According to Lemma 17 we know that there exists $r \in M$ and $t_1 \in [0, t_*)$ where robot $\mathfrak{S}_r(w) = true$. However, robot $w$ is not locked as in the previous case, since it is not at its own crash point $p(w)$ during $[0, t_*)$. The crash points of $r$ and $w$ might coincide and the safety

condition for robot $w$ might be also satisfied, i.e. $\mathfrak{S}_w(r) = true$. We lack information about the structure of the $MEC(r)$, thus we cannot guarantee that the crash point $p(r)$ does not exist after the safe motion of robot $w$. However, Proposition 6 guarantees that $w$ will not collide with some other robot during the motion in the safe state.

On the other hand, we can show that at the end of the safety motion of robot $r$ the crash point $p(w)$ does not exist. Robot $r$ might be locked during some proper subinterval of $[0, t_*)$, but as it cannot collide with some other robot while being locked by Proposition 7.

If our assumption of potential collision holds even if robot was locked during some proper subinterval of $[0, t_*)$,then according to Lemma 17 there is still a point in time $t_1 \in [0, t_*)$ where robot $\mathfrak{S}_r(w) = true$. Next we show that, independent of the motion of $w$ if robot performs the motion in the safety state, then the crash point $p(r)$ does not exist at the end of this motion.

Let us look back at Figure 4.7 once again. Robot $w$ moves either with speed 1 or $s$. Due to chirality with speed $s$, robot $r$ moves to the right in Figure 4.7. The RNG lenses intersect the line that passes through $yy'$ only inside the line segment $az'$, since robot $r$ to the left can only move with speed 1.

Robot $r$ that performs the safe motion starts at $c$ and moves towards target $M(r)$ on arc $yg$. From the previous case we know that $yg$ is inside RNG lens $C_1$ for $s \geq 10$, $m \geq 500$ and $k = 4$. Due to this, the target point $M(r)$ of $r$ (midpoint of $A$) during the safe motion of $r$ is strictly inside of corresponding RNG lens. At the end of the safe motion, crash point $p(w)$ does not exist, since $r$ violates Relative neighborhood criterion between $w$ and one of two robots $m_1$, $m_2$ that form the crash point. □

Using Proposition 6 and Proposition 7 together with Lemma 14, Lemma 17, Lemma 18 and Theorem 7, we prove one of our main results, namely Theorem 8.

**Theorem 8.** *For $s \geq 10$, $m \geq 500$, and $k = 4$, the Safe-Go-To-The-Relative-Center algorithm performs gathering in $O(n^2)$ without collisions.*

*Proof.* In this theorem, we would like to show that with S-GTRC the gathering is collisionless. In other words if $t_*$ is the time of the final collision, then for all $\in (0, t_*)$, for all $r, w \in R$ it holds that $|r, w| > 0$. Let us consider the trajectory of robot $r$. It consists of segments that correspond to the different states of the robot: regular, locked (regular) and safe. According to Proposition 6 and Proposition 7, in a safe and locked state robot $r$ cannot collide with some other robot $w$: i.e., in safe and locked state for all $t \in (0, t_*)$, for all $r, w \in R$ if $r$ is in the safe or locked state it holds that $|r, w| > 0$.

In the regular state according to Lemma 14, the set of robots $M \subset R$ is such that $r \in M$ can collide only at the crash point $p(w)$ of some robot $w \in M$. In order to reach crash point $p(w)$, robots from set $M \setminus w$ need to get in a regular state through the narrow gap between the RNG lenses as illustrated in Figure 4.4. However, this is impossible in the S-GTRC algorithm.

Let us consider the segment of the trajectory where the state of $r$ is regular starting from time $t_1 > 0$ and there is a potential collision of set $M \subset R$ and $w$ in $p(w)$ at time $t_4$. In other words, for any $u, v \in \{M, w\}$ it holds that $|u, v| = 0$ at time $t_4 > t_1$.

According to Lemma 17 and Lemma 18 there exists $t_2 < t_4$ and robot $u \in M$ starts a safe move at $t_2$. At the end of the safe move (at time $t_3 \in (t_2, t_4)$), robot $r$ changes configuration in such way

that point $p(w)$ is not the crash point of $w$.

At time $t_3$ at least robot $w$ has a different crash point $p'(w)$ and no other robot except $w$ can be at this point, according to Corollary 4. Therefore, for any $r \in R$ and $w \in R, r \neq w$ at time $t_3$ it holds that $|p'(w), r| > 0$. As we already mentioned in Lemma 14, in order to have a collision, robot $r$ shall reach the crash point of some other robot $w$, therefore also $|w, r| > 0$ for all $r \in R$ at time $t_3$. Starting from $t_3$ we repeat our consideration for every segment of the trajectory, where $r$ is in the regular state until all robots have gathered in a final collision. On every such segment for all $r, w \in R$ it holds that $|r, w| > 0$, therefore for all $t \in (0, t_*)$, for all $r, w \in R$ it holds that $|r, w| > 0$. □

## 4.2 The Near Gathering Problem

The near gathering problem for the robots with limited visibility was first considered in [PPV15]. The near gathering problem is a variation of the gathering problem, where the aim of the robots is to get close enough to be in vision range of each other without collisions. In [PPV15] this problem was solved with robots that are equipped with a compass so that they have a common coordinate system. We would like to show that with the extended robot model we can solve the gathering problem without agreement on a common coordinate system.

Let us consider group $R$ of $n$ robots within the extended robot mode. At time 0 we are given initial configuration such that $UDG(R)$ is connected. The goal is to gather all robots in one non-predefined circle $B_c$, where radius $c$ is some positive constant. Besides that, all robots shall have distinct positions, i.e. the gathering process shall be collisionless.

Before we describe the N-GTRC algorithm we need one definition. At every point in time in the N-GTRC algorithm, the robot checks whether there exists $r \in R$ such that

$$\max_{a,b \in 2\text{-}UDG(r)} |a,b| < \frac{1}{2}. \tag{4.9}$$

We refer to the logical expression above as the *terminal condition*, denoted by the function $\mathfrak{T}_r : X \to \mathbb{Z}_2$, where $X \subset \mathbb{R}^2$.

**Near gathering Go-To-The-Relative-Center algorithm (N-GTRC):** Every robot $r \in R$ at every point in time $t$ checks the terminal condition. If $\mathfrak{T}_r = true$, then the robot is idle. Otherwise, if $\mathfrak{T}_r = false$, then robot $r$ executes the S-GTRC algorithm.

First of all, we would like to make sure that all robots at the same time recognize that they have gathered. In Lemma 19 we consider the termination condition and make sure that if one robot is idle, then the rest also idle. Namely, robots have gathered inside of non-predefined circle $B_{\frac{1}{4}}$.

**Lemma 19.** *If there exists $r \in R$ such that $\mathfrak{T}_r = true$, then for all $r \in R$ it holds that $\mathfrak{T}_r = true$.*

*Proof.* Assume that there exists $r \in R$ such that $\mathfrak{T}_r = true$. It implies that all robots of 2-$UDG(r)$ are inside circle $B_{\frac{1}{4}}$.

Let us consider circle $B_{1\frac{1}{4}}$, which has the same center as $B_{\frac{1}{4}}$. There are no other robots in $B_{1\frac{1}{4}} \setminus B_{\frac{1}{4}}$, because if there exists robot $u$ in $B_{1\frac{1}{4}} \setminus B_{\frac{1}{4}}$, then either $|a,u| > |a,b|$ or $|b,u| > |a,b|$, which contradicts our assumption.

For any position $x$ in $B_{\frac{1}{4}}$ it holds that $B_{1\frac{1}{4}} \subset B_2(x)$, where $B_2(x)$ represents two unit disk centered in $x$. In other words any robot in $B_{\frac{1}{4}}$ can see whole $B_{1\frac{1}{4}}$. On the other hand, the S-GTRC algorithm preserves connectivity with respect to distance 1 according

to Lemma 10. But since there are no other robots in $B_{1\frac{1}{4}} \setminus B_{\frac{1}{4}}$ there cannot also be robots outside $B_{1\frac{1}{4}}$. All $r \in R$ are in $B_{\frac{1}{4}}$ and they all are idle: i.e., for all $r \in R, \mathfrak{T}_r = true$. □

The time needed to solve the near gathering problem with N-GTRC is clearly not greater than time needed to solve the gathering problem with S-GTRC for the same initial configuration. Therefore, for the extended robot model we can state the following.

**Theorem 9.** *The group of n robots solves the near gathering problem, i.e., it gathers in one non-predefined circle $B_{\frac{1}{4}}$ in time $O(n^2)$.*

# Chapter 5

# Conclusion and Outlook

The goal of this thesis was to examine how efficiently can robots with a limited viewing range solve the gathering problem in the continuous time model. Besides that, the core interest was to gather robots without collisions by using local information only.

Local information and the concurrency between the robots in the continuous time model were two major challenges in this thesis. From the related work, we know that the gathering problem in the model that we consider becomes trivial if the robots are granted unlimited viewing range.

On the other hand, robots models that are too weak (e.g., robots know only distance but not the direction) do not allow to solve the gathering problem even for two robots. The model with limited visibility lies between these two extreme assumptions.

We have studied the gathering problem in the continuous time model with limited visibility. For the given problem we have proposed the class of contracting algorithms which solve the gathering problem in time $O(nd)$, where $d$ is the diameter of the initial configuration. The definition of this class is so simple that we were able to use it as a criterion for the design and runtime analysis of several gathering algorithms.

Concerning the gathering problem, we have shown that there are implicit algorithms that match the lower and upper bounds for the class of contracting algorithms. Besides that, we have analyzed the quality and correctness of several known and new gathering algorithms.

In addition to theoretical research, we have also performed simulations with different gathering algorithms in order to obtain a better understanding of the dynamics during runtime. One of our observations was that Go-To-The-Relative-Center and Go-To-The-Gabriel-Center algorithms gather robots almost without collisions. On the basis of this observation for the Go-To-The-Relative-Center algorithm, we have shown that collisions take place only at specific points. This information, together with the contracting criterion, allowed us to develop the Safe-Go-To-The-Relative-Center algorithm. This algorithm is a contracting, local algorithm that solves the gatecrashing problem without collisions. We were also able to apply our new collisionless algorithm to the near gathering problem. An overview of the results can be found in Table 5.1.

Nevertheless, there are still plenty of open questions. Can robots gather faster than in quadratic time? The answer to this is yes since there is already a known algorithm due to [KKM12] that solves the gathering problem in time $O(\min\{n, OPT\ log(OPT)\})$, where $n$ is the number of robots and $OPT$ is the runtime of the optimal algorithm with unlimited visibility. Nevertheless, what is not known is whether there is a simple criterion for the corresponding class of linear time gathering algorithms.

The other obvious open question is the Collisionless Conjecture. It already seems to be difficult even for $n = 4$ robots. New methods from theory of dynamic systems are needed. With certain algorithmic extensions, i.e. new capabilities that we granted to the robots,

we have been able to design the collisionless contracting gathering algorithm. For many formation problems the open question is which capabilities of robots are necessary for a given global task, which are sufficient, and which are technically feasible.

Table 5.1: Overview of results.

| **Algorithm** | **Runtime** | **Connectivity property** | **Collisionless property** |
|---|---|---|---|
| Implicit-Go-To-The-Left (I-GTL) | $\frac{n^2}{2\pi^2} \leq t \leq \frac{n^2}{8}$ Cor. 2 | - | - |
| Implicit-Go-On-Bisector (I-GOB) | $O(n)$ Cor. 3 | - | - |
| Go-To-The-Gravity-Center (GTGrC) | $O(n^2)$ Pro. 2 | ✗ Lem. 6 | - |
| Go-To-The-Center (GTC) | $O(n^2)$ Thm. 2 | ✓ Lem. 7 | ✗ Sec. 3.1 |
| Go-To-The-Relative-Center on the line (GTRC on the line) | $\Theta(n)$ Thm. 5 | ✓ Lem. 10 | ✓ Thm. 6 |
| Go-To-The-Relative-Center (GTRC) | $O(n^2)$ Thm. 3 | ✓ Lem. 10 | ✗ Sec. 3.3 |
| Go-To-The-Gabriel-Center (GTGC) | $O(n^2)$ Thm. 4 | ✓ [LMP16] | ✗ [LMP16] |
| Safe-Go-To-The-Relative-Center (S-GTRC) | $O(n^2)$ Thm. 7 | ✓ Cor. 5 | ✓ Thm. 8 |
| Neat Gathering Go-To-The-Relative-Center (N-GTRC) | $O(n^2)$ Thm. 9 | ✓ Cor. 5 | ✓ Thm. 8 |

# Bibliography

[ACF+16] Abshoff, Sebastian, Andreas Cord-Landwehr, Matthias Fischer, Daniel Jung, and Friedhelm Meyer auf der Heide: *Gathering a closed chain of robots on a grid.* In *Proceedings of the 30th International Parallel and Distributed Processing Symposium (IPDPS)*, pages 689–699. IEEE, May 2016.

[AGM13] Agathangelou, Chrysovalandis, Chryssis Georgiou, and Marios Mavronicolas: *A distributed algorithm for gathering many fat mobile robots in the plane.* In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, pages 250–259. ACM, 2013.

[ASY95] Ando, Hideki, Ichiro Suzuki, and Masafumi Yamashita: *Formation and agreement problems for synchronous mobile robots with limited visibility.* In *Proceedings of 10th International Symposium on Intelligent Control*, pages 453–460. IEEE, 1995.

[BCJKF14] Bolla, Kálmán, Zsolt Csaba Johanyák, Tamás Kovács, and Gábor Fazekas: *Local center of gravity based gathering algorithm for fat robots.* In *Issues and Challenges of Intelligent Systems and Computational In-*

*telligence*, pages 175–183. Springer International Publishing, 2014.

[Ber59] Bernhart, Arthur: *Polygons of pursuit.* Scripta Math, 24:23–50, 1959.

[CDF+11a] Cord-Landwehr, Andreas, Bastian Degener, Matthias Fischer, Martina Hüllmann, Barbara Kempkes, Alexander Klaas, Peter Kling, Sven Kurras, Marcus Märtens, Friedhelm Meyer auf der Heide, Christoph Raupach, Kamil Swierkot, Daniel Warner, Christoph Weddemann, and Daniel Wonisch: *A new approach for analyzing convergence algorithms for mobile robots.* In *Automata, Languages and Programming - 38th International Colloquium (ICALP)*, pages 650–661. Springer, 2011.

[CDF+11b] Cord-Landwehr, Andreas, Bastian Degener, Matthias Fischer, Martina Hüllmann, Barbara Kempkes, Alexander Klaas, Peter Kling, Sven Kurras, Marcus Märtens, Friedhelm Meyer auf der Heide, Christoph Raupach, Kamil Swierkot, Daniel Warner, Christoph Weddemann, and Daniel Wonisch: *Collisionless gathering of robots with an extent.* In *Theory and Practice of Computer Science - 37th Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, pages 178–189. Springer, 2011.

[CDSN17] Cicerone, Serafino, Gabriele Di Stefano, and Alfredo Navarra: *Gathering of robots on meeting-points: feasibility and optimal resolution algorithms.* Distributed Computing, 2017.

[CFPS03] Cieliebak, Mark, Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro: *Solving the robots gathering problem.* In *Automata, Languages and Programming - 30th International Colloquium (ICALP)*, pages 1181–1196. Springer, 2003.

[CGP06] Czyzowicz, Jurek, Leszek Gasieniec, and Andrzej Pelc: *Gathering few fat mobile robots in the plane.* In *Principles of Distributed Systems - 10th International Conference (OPODIS)*, pages 350–364. Springer, 2006.

[Chr85] Chrystal, George: *On the problem to construct the minimum circle enclosing n given points in a plane.* Proceedings of the Edinburgh Mathematical Society, Third Meeting, pages 30–35, 1885.

[CLFJM16] Cord-Landwehr, Andreas, Matthias Fischer, Daniel Jung, and Friedhelm Meyer auf der Heide: *Asymptotically optimal gathering on a grid.* In *Proceedings of the 28th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 301–312. ACM, 2016.

[CMN04] Chatzigiannakis, Ioannis, Michael Markou, and Sotiris E. Nikoletseas: *Distributed circle formation for anonymous oblivious robots.* In *Experimental and Efficient Algorithms (WEA)*, pages 159–174, 2004.

[CP04] Cohen, Reuven and David Peleg: *Convergence properties of the gravitational algorithm in asynchronous robot systems.* In *12th Annual European Symposium on Algorithms (ESA)*, pages 228–239. Springer, 2004.

[DFP+12] Das, Shantanu, Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Masafumi Yamashita: *The power of lights: Synchronizing asynchronous robots using visible bits.* In *32nd International Conference on Distributed Computing Systems (ICDCS)*, pages 506–515. IEEE, 2012.

[DK02] Défago, Xavier and Akihiko Konagaya: *Circle formation for oblivious anonymous mobile robots with no common sense of orientation.* In *Proceedings of the 2002 Workshop on Principles of Mobile Computing (POMC)*, pages 97–104. ACM, 2002.

[DKL+11] Degener, Bastian, Barbara Kempkes, Tobias Langner, Friedhelm Meyer auf der Heide, Peter Pietrzyk, and Roger Wattenhofer: *A tight runtime bound for synchronous gathering of autonomous robots with limited visibility.* In *Proceedings of the 23rd Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 139–148. ACM, 2011.

[DKLM06] Dynia, Miroslaw, Jaroslaw Kutylowski, Pawel Lorek, and Friedhelm Meyer auf der Heide: *Maintaining communication between an explorer and a base station.* In *Biologically Inspired Cooperative Computing, IFIP 19th World Computer Congress, TC 10: 1st IFIP International Conference on Biologically Inspired Computing*, pages 137–146. Springer, 2006.

[DKM10] Degener, Bastian, Barbara Kempkes, and Friedhelm Meyer auf der Heide: *A local $O(n^2)$ gathering algorithm.* In *Proceedings of the 22nd Annual ACM Sym-*

*posium on Parallelism in Algorithms and Architectures (SPAA)*, pages 217–223. ACM, 2010.

[DKMS07] Dynia, Miroslaw, Jaroslaw Kutylowski, Friedhelm Meyer auf der Heide, and Jonas Schrieb: *Local strategies for maintaining a chain of relay stations between an explorer and a base station.* In *Proceedings of the 19th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 260–269. ACM, 2007.

[DP09] Dieudonné, Yoann and Franck Petit: *Self-stabilizing deterministic gathering.* In *Algorithmic Aspects of Wireless Sensor Networks, 5th International Workshop (ALGOSENSORS)*, pages 230–241. Springer, 2009.

[DSKN12] D'Angelo, Gianlorenzo, Gabriele Di Stefano, Ralf Klasing, and Alfredo Navarra: *Gathering of robots on anonymous grids without multiplicity detection.* In *Structural Information and Communication Complexity - 19th International Colloquium (SIROCCO)*, pages 327–338. Springer, 2012.

[FJM17] Fischer, Matthias, Daniel Jung, and Friedhelm Meyer auf der Heide: *Gathering anonymous, oblivious robots on a grid.* CoRR, abs/1702.03400, 2017.

[FPS12] Flocchini, Paola, Giuseppe Prencipe, and Nicola Santoro: *Distributed Computing by Oblivious Mobile Robots.* Synthesis Lectures on Distributed Computing Theory. Morgan & Claypool Publishers, 2012.

[FPSV14] Flocchini, Paola, Giuseppe Prencipe, Nicola Santoro, and Giovanni Viglietta: *Distributed computing by mo-*

*bile robots: Solving the uniform circle formation problem.* In *Principles of Distributed Systems - 18th International Conference (OPODIS)*, pages 217–232. Springer, 2014.

[GS69] Gabriel, K. Ruben and Robert R. Sokal: *A new statistical approach to geographic variation analysis.* Systematic Biology, 18(3):259–278, 1969.

[GWB04] Gordon, Noam, Israel A. Wagner, and Alfred M. Bruckstein: *Gathering multiple robotic a(ge)nts with limited sensing capabilities.* In *Ant Colony Optimization and Swarm Intelligence, 4th International Workshop (ANTS)*, pages 142–153. Springer, 2004.

[HPT14] Honorat, Anthony, Maria Potop-Butucaru, and Sébastien Tixeuil: *Gathering fat mobile robots with slim omnidirectional cameras.* Theoretical Computer Science, 557:1–27, 2014.

[IKIW07] Izumi, Taisuke, Yoshiaki Katayama, Nobuhiro Inuzuka, and Koichi Wada: *Gathering autonomous mobile robots with dynamic compasses: An optimal result.* In *Distributed Computing, 21st International Symposium (DISC)*, pages 298–312. Springer, 2007.

[Kat05] Katreniak, Branislav: *Biangular circle formation by asynchronous mobile robots.* In *Structural Information and Communication Complexity - 12th International Colloquium (SIROCCO)*, pages 185–199. Springer, 2005.

[Kat11] Katreniak, Branislav: *Convergence with limited visibility by asynchronous mobile robots.* In *Structural In-*

*formation and Communication Complexity - 18th International Colloquium (SIROCCO)*, pages 125–137. Springer, 2011.

[KKM12] Kempkes, Barbara, Peter Kling, and Friedhelm Meyer auf der Heide: *Optimal and competitive runtime bounds for continuous, local gathering of mobile robots.* In *Proceedings of the 24th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 18–26. ACM, 2012.

[KM09] Kutylowski, Jaroslaw and Friedhelm Meyer auf der Heide: *Optimal strategies for maintaining a chain of relays between an explorer and a base camp.* Theoretical Computer Science, 410(36):3391–3405, 2009.

[KM11] Kling, Peter and Friedhelm Meyer auf der Heide: *Convergence of local communication chain strategies via linear transformations: or how to trade locality for speed.* In *Proceedings of the 23rd Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 159–166. ACM, 2011.

[KTI+07] Katayama, Yoshiaki, Yuichi Tomida, Hiroyuki Imazu, Nobuhiro Inuzuka, and Koichi Wada: *Dynamic compass models and gathering algorithms for autonomous mobile robots.* In *Structural Information and Communication Complexity - 14th International Colloquium, (SIROCCO)*, pages 274–288. Springer, 2007.

[LFC+17] Luna, Giuseppe Antonio Di, Paola Flocchini, Sruti Gan Chaudhuri, Federico Poloni, Nicola Santoro, and Giovanni Viglietta: *Mutual visibility by luminous robots*

*without collisions.* Information and Computation, 254:392–418, 2017.

[LM14] Lukovszki, Tamás and Friedhelm Meyer auf der Heide: *Fast collisionless pattern formation by anonymous, position-aware robots.* In *Principles of Distributed Systems - 18th International Conference (OPODIS)*, pages 248–262. Springer, 2014.

[LMMP17] Li, Shouwei, Christine Markarian, Friedhelm Meyer auf der Heide, and Pavel Podlipyan: *A continuous strategy for collisionless gathering.* In *Algorithms for Sensor Systems, Proceedings of the 13th International Symposium on Algorithms and Experiments for Wireless Sensor Networks (ALGOSENSORS).* Springer, 2017. Full version: https://www.hni.uni-paderborn.de/pub/9531.

[LMP16] Li, Shouwei, Friedhelm Meyer auf der Heide, and Pavel Podlipyan: *The impact of the gabriel subgraph of the visibility graph on the gathering of mobile autonomous robots.* In *Algorithms for Sensor Systems, Proceedings of the 12th International Symposium on Algorithms and Experiments for Wireless Sensor Networks (ALGOSENSORS)*, pages 62–79. Springer, 2016.

[Meg83] Megiddo, Nimrod: *Linear-time algorithms for linear programming in $\mathbb{R}^3$ and related problems.* SIAM Journal on Computing, 12(4):759–776, 1983.

[Nah07] Nahin, Paul J.: *Chases and Escapes: The Mathematics of Pursuit and Evasion.* Princeton University Press, 2007.

[Pel05] Peleg, David: *Distributed coordination algorithms for mobile robot swarms: New directions and challenges.* In *Proceedings of 7th International Workshop on Distributed Computing (IWDC)*, pages 1–12. Springer, 2005.

[PPV15] Pagli, Linda, Giuseppe Prencipe, and Giovanni Viglietta: *Getting close without touching: near-gathering for autonomous mobile robots.* Distributed Computing, 28(5):333–349, Oct 2015.

[SBMM17] Sharma, Gokarna, Costas Busch, Supratik Mukhopadhyay, and Charles Malveaux: *Tight analysis of a collisionless robot gathering algorithm.* ACM Transactions on Autonomous and Adaptive Systems, 12(1):3:1–3:20, April 2017.

[Sch95] Schierscher, Georg: *Verfolgungsprobleme.* Berichte über Mathematik und Unterricht, (95-06), 1995.

[SDY06] Souissi, Samia, Xavier Défago, and Masafumi Yamashita: *Gathering asynchronous mobile robots with inaccurate compasses.* In *Principles of Distributed Systems - 10th International Conference (OPODIS)*, pages 333–349. Springer, 2006.

[SDY09] Souissi, Samia, Xavier Défago, and Masafumi Yamashita: *Using eventually consistent compasses to gather memory-less mobile robots with limited visibility.* ACM Transactions on Autonomous and Adaptive Systems, 4(1):9:1–9:27, 2009.

[Sin97] Singer, Ivan: *Abstract Convex Analysis.* Wiley, 1997.

[SY99] Suzuki, Ichiro and Masafumi Yamashita: *Distributed anonymous mobile robots: Formation of geometric patterns.* SIAM Journal on Computing, 28(4):1347–1363, 1999.

[Tou80] Toussaint, Godfried T.: *The relative neighbourhood graph of a finite planar set.* Pattern Recognition, 12(4):261–268, 1980.

[YDIW07] Yared, Rami, Xavier Défago, Julien Iguchi-Cartigny, and Matthias Wiesmann: *Collision prevention platform for a dynamic group of asynchronous cooperative mobile robots.* Journal of Networks, 2(4):28–39, 2007.