Uniform Circle Formation by Asynchronous Robots: A Fully-Distributed Approach

Shan Jiang*, Jiannong Cao*, Jia Wang*, Milos Stojmenovic†, Julien Bourgeois‡
*The Hong Kong Polytechnic University, Hong Kong, China
†Singidunum University, 11000 Belgrade, Serbia
‡The FEMTO-ST Institute, 25200 Montbeliard, France
{cssjiang, csjcao, csjiawang}@comp.polyu.edu.hk , mstojmenovic@singidunum.ac.rs, bourgeois@cmu.edu

***Abstract*—Recent advances in robotics technology have made it practical to deploy a large number of inexpensive robots in a wide range of application domains. In many of those applications, a group of autonomous robots is required to form a predefined geometric shape such as a line or a circle. This problem, namely pattern formation, is one of the most important coordination problems in multi-robot systems. A particular pattern extensively studied in literature is the uniform circle, and the corresponding problem is called *uniform circle formation*. In uniform circle formation, a set of simple mobile robots (asynchronous, autonomous), starting from arbitrary positions on the plane, have to arrange themselves on the vertices of a regular polygon eventually. Towards addressing the problem, existing works usually make conveniently strong assumptions, i.e., the robots are regarded as mass points and have unlimited sensing and communication range. The question of whether the robots with actual size and limited sensing and communication range could form a uniform circle, to our knowledge, has remained open. In this paper, we propose a new approach towards addressing this issue. Three phases, consensus on the circle, circle formation, and uniform transformation, constitute our approach. We propose novel distributed algorithms for convex hull construction and cardinality estimation. Simulation results, theoretical analysis, and successful deployment have shown the effectiveness and practicability of our approach.**

***Index Terms*—multi-robot system, uniform circle formation, distributed algorithm**

I. Introduction

In recent years, advances in robotics, microelectronics and other related fields have made it feasible for engineers to fabricate inexpensive robots. It has been a trend in the robotics community to use a set of robots to accomplish the tasks instead of a single robot. The group of robots working in collaboration with each other is commonly referred to as a multi-robot system (MRS) [1][2]. The use of MRS provides better scalability, reliability, flexibility, versatility and helps in performing the tasks in a faster and cheaper way compared to single robot systems [3]. MRS can be very useful in search and surveillance applications, in particular in areas which are difficult or impossible for humans to access. Another benefit of MRS is that they have better spatial distribution [4]. Many applications such as underwater and space exploration, disaster relief, rescue missions in hazardous environments, military operations, medical surgeries, agriculture and smart homes can make use of distributed group of robots. It would not only be difficult but also may result in wastage of resources if such applications are developed using single robot systems.

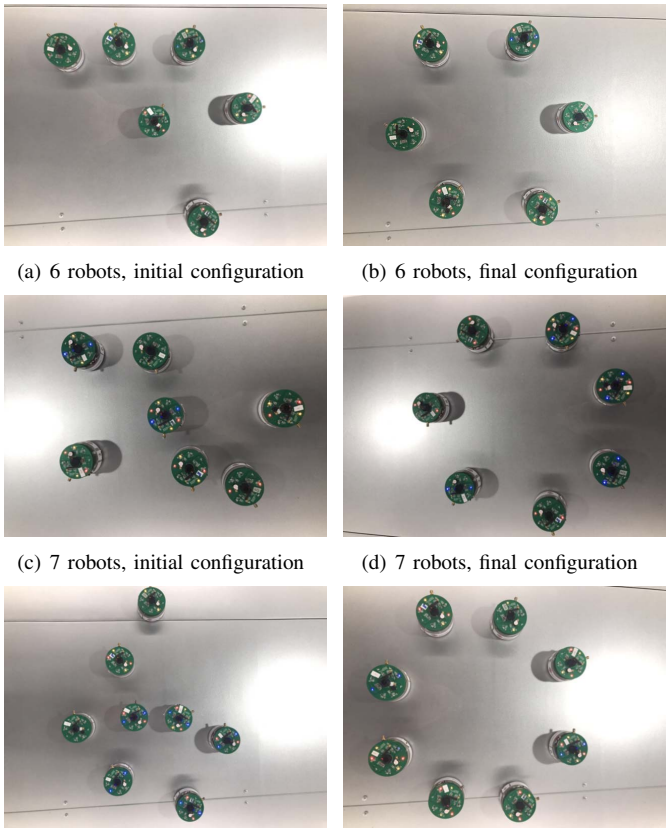
In many multi-robot applications, a group of autonomous robots is required to eventually form a predefined geometric pattern such as a circle [5][6] or a line. This problem, namely *pattern formation problem* [7], is one of the most important coordination problems for MRS. There are various advantages to forming a pattern such as enhancing coordination efficiency of the system and reducing outer impacts on the system. The pattern formation problem is also closely related to the agreement problem, which is a fundamental problem in distributed computing. For example, forming a single point corresponds to the *gathering* problem requiring all robots to gather at the same location, not determined in advance.

Pattern formation problem has been a hot research topic in the field of MRS for a long time [8]. A particular pattern extensively studied in literature is the *uniform circle* [9][10][11][12], in which the points form a regular polygon. The corresponding problem is called *uniform circle formation*. The problem of uniform circle formation plays a major role in the coordination problems in MRS due to the critical observation of formability by Suzuki and Yamashita [13]. Their observations indicate that uniform circles and points are the only patterns formable from arbitrary initial configuration in FSYNC (and thus also in SSYNC and ASYNC) [12].

The problem of uniform circle formation can be easily solved in a centralized MRS, in which there is a central computing station. The station knows all the information of the system and is responsible for computing all the actions of the robots. In this case, the problem of uniform circle formation can be mapped as a *minimum weight maximum matching problem (MWMMP)*, in which a set of robots are assigned with a set of target formation positions, and the weight is the sum of distances from each robot to its goal position. MWMMP is a typical combinatorial optimization problem, which can be modeled as an integer linear programming problem and optimally solved in polynomial time [14].

However, the uniform circle formation problem becomes very challenging for distributed MRS, in which there is no centralized computing station. The robots have to make decisions by themselves independently, e.g., when and where to move, and how to avoid collisions. Moreover, each robot can only communicate with other robots inside its communication

978-1-5090-2991-4/17/$31.00 ©2017 IEEE



(a) 6 robots, initial configuration (b) 6 robots, final configuration

(c) 7 robots, initial configuration (d) 7 robots, final configuration

(e) 8 robots, initial configuration (f) 8 robots, final configuration

Fig. 1. several robots are forming uniform circles

range. In this case, the robots do not even know the total number of robots. Therefore, there are many difficulties on how to make a consensus on the circle to form and how to achieve the uniform circle through distributed coordination among the robots.

In this paper, we consider the problem of uniform circle formation for distributed MRS. To solve the problem, we first decompose it into three parts, namely consensus on circle, circle formation and uniform transformation. In the part of consensus on the circle, the robots estimate the total number of robots in the MRS and decide the center and the radius of the circle to form. In the part of circle formation and uniform transformation, the robots form a circle first and then move along the circle to make themselves evenly distributed on the circle. After designing our algorithm, we deployed it in our test-bed. Fig. 1 shows the running examples in which six, seven and eight robots are forming uniform circles respectively. We can see that our algorithm can successfully arrange the MRS into uniform circles with different numbers of robots and different initial configurations. The contributions of this work are:

- We formulate the uniform circle formation for distributed MRS and propose a three-phase solution, namely consensus on the circle, circle formation and uniform transformation, to this problem.
- We propose distributed algorithms for convex hull construction and cardinality estimation for MRS. We evaluate their performance and efficiency by simulation and theoretical analysis. The results show that our algorithms outperform existing ones. Furthermore, these algorithms can widely be applied in other distributed systems.
- We deploy in a realistic test-bed to test our solution. Successful experiments have implied the practicability and effectiveness of our solution.

We organize the rest of the paper as follows. Section II introduces the system model and problem definition. A three-phase fully-distributed approach, as well as the simulation and theoretical analysis, are discussed in Section III. In Section IV, successful deployment in our test-bed is demonstrated. Finally, related works are discussed in Section V and Section VI concludes the whole paper.

II. Preliminaries

In this section, we introduce the computational model of the system in Section II-A and formally formulate the uniform circle formation problem in Section II-B.

A. System Model

Consider a set of n computational entities $\mathcal{R} = \{r_1, \cdots, r_n\}$, namely *robots*, located on a Euclidean plane $\mathbb{R}^2$, on which they can move continuously. The robots are capable of localization, communication, and sensing. For robot r_i at time t, it is aware of its position and orientation $p_i^t = (x_i^t, y_i^t, \theta_i^t)$ under a common Cartesian coordinate system, where θ_i^t is the clockwise angle to the direction of the y-axis. Each robot can send and receive messages to and from its neighbors within a common communication range R_c. Besides communication, each robot can detect the relative positions of its neighbors within a common sensing range R_s. All robots are of identical size R, which is the radius of the smallest circle that wraps the robot. The only way to distinguish different robots is to use their unique identifiers $r_1, \cdots, r_n$. The identifiers can only be conveyed via wireless communication.

Definition 1. *(Configuration) $C^t = \{p_1^t, \cdots, p_n^t\}$ is the collection of positions of all robots at time t.*

Definition 2. *(Collision-free Configuration) A configuration C^t is said to be collision-free if the Euclidean distance between p_i^t and p_j^t is no less than $2R$ for all $1 < i < j < n$.*

Definition 3. *(Active Range) The active range R_a of the robots is defined as the minimal value between communication range R_c and sensing range R_s. Two robots are connected if they are within the connected range of each other.*

Definition 4. *(Connected Configuration) A configuration is said to be connected if for all p_i^t, there exists at least one p_j^t $(j \neq i)$ such that the Euclidean distance between p_i^t and p_j^t is no more than R_a.*

In the beginning, we assume that the whole system $\mathcal{R}$ is in a collision-free and connected configuration. This means

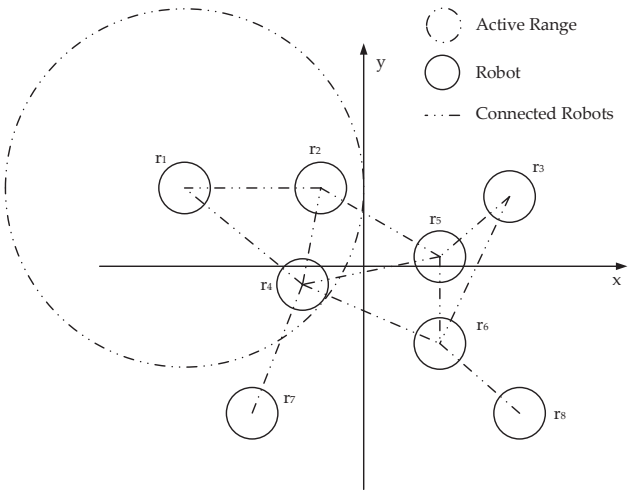


Fig. 2. An example of initial configuration. It is *collision-free* and *connected*.

the robots do not collide with each other, and there is no isolated robot in the system. Note that the assumption of full connectivity is necessary, otherwise the whole system will be divided into several small groups, and the problem is going to be solved in several small MRSs separately. Fig. 2 gives an example of an initial configuration of 8 robots. In the example, each robot can only sense and communicate locally, and all robots are identical in the system and form a distributed MRS.

Then, the robots repeatedly execute *Sense-Process-Act* cycles. Each cycle can be divided into three sequential phases as follows:

- *Sense*. The robot observes and collects data in the environment, itself and its neighbors. The collected data includes the position of itself, the relative positions of the robots within R_s and the messages from the robots within R_c;
- *Process*. The robot executes a given deterministic algorithm, which takes the collected data as input and outputs the next position to go to and the messages to deliver. Note that the algorithm is the same for all robots;
- *Act*: The robot moves directly toward the destination point (or stays still) along a line segment and delivers the messages. The destination point and the messages to deliver are computed in the previous *Process* phase.

The robots are asynchronous with respect to their *Sense-Process-Act* cycles. That is to say, the execution of each *Sense-Process-Act* cycle of each robot is not only completely arbitrary but independent of the cycles of the other robots as well. In particular, it can be an arbitrarily long duration from the time a robot collects data to the time it moves based on the data. Also, one robot may be starting the *Sense* phase while another robot is performing the *Act* phase. This computational model is called asynchronous (ASYNC, also called CORDA) [15], which is the most general model in distributed systems.

There are two other computational models FSYNC and SSYNC, which have more restrictions than ASYNC. The robots are fully synchronous (FSYNC) if all robots start every *Sense-Process-Act* cycle simultaneously and synchronously execute each of its *Sense*, *Process*, and *Act*. In the semi-synchronous (SSYNC, also called SYM or ATOM) [16] setting, not all robots are active in every cycle, but all of those who start a certain cycle synchronously execute each of its *Sense*, *Process*, and *Act*. Fig. 3 compares the three models of execution of the *Sense-Process-Act* cycles.

Definition 5. *(Uniformly-circular Configuration) A configuration C^t is said to be uniformly-circular if there exists a regular n-gon P_n such that each p_i^t $(1 \leq i \leq n)$ equals the position of one of P_n's vertex.*

B. Problem Definition

With the definitions given in the Section II-A, we now formally define the problem of uniform circle formation to be solved in this paper as follows.

Definition 6. *Uniform Circle Formation: Given a set of n robots $\mathcal{R} = \{r_1, r_2, \cdots, r_n\}$ initially under a collision-free and connected configuration, arrange them under a collision-free and uniformly-circular configuration through a sequence of collision-free configurations.*

III. A FULLY-DISTRIBUTED APPROACH

In this section, we propose a fully-distributed approach to solving the uniform circle formation problem stated in Section II-B. In Section III-A, a general framework of the algorithm is introduced. Then detailed steps are discussed from Section III-B to Section III-G.

A. Algorithm Framework

When the system is just starting up, the robots are not aware of their neighbors. A robot network should be built up to enable the communication between the robots. This step is called *network construction*. In Section III-B, a network construction algorithm is proposed to construct neighbor lists for the robots. Note that this step is critical since the topology of the network remarkably affects the number and size of messages passing among the robots.

After the network construction, the robots can communicate with their neighbors. Since the circle to be formed is not given in advance, the robots should negotiate with each other to make a consensus on a common uniform circle to form. Two parameters, the radius, and the center are necessary to determine a uniform circle.

On the one hand, to make a consensus on the center, we propose a distributed convex hull construction algorithm in Section III-C. After execution of the algorithm, each robot will be aware of the convex hull of all robots. Then each robot makes the average of the positions of all robots as the center.

On the other hand, to reach consensus on the radius, a distributed cardinality estimation algorithm is proposed in Section III-D. In the MRS, no robot is aware of the total

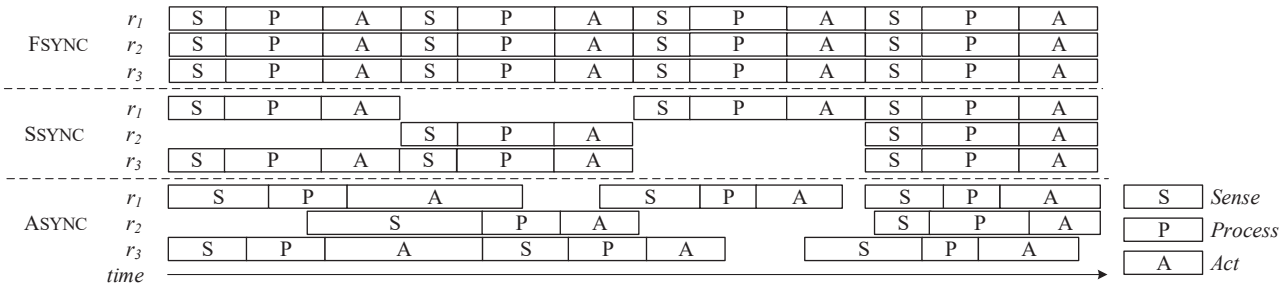


Fig. 3. An illustration of execution of the *Sense-Process-Act* cycles of three robots for the models of FSYNC, SSYNC, and ASYNC.

number of robots. Also, to count the exact number of entities inside a distributed system is computationally expensive. Therefore, our method is to estimate the approximate number of robots in the system. Note that the estimation is crucial since overestimation leads to unconnected configuration while underestimation results in insufficient space to place all robots.

Definition 7. *(Circular Configuration) A configuration C^t is said to be circular on center point O and radius r if the distances between each p_i^t ($1 \leq i \leq n$) and O are equal to r. Furthermore, a configuration C^t is said to be C-circular if C is a circle and C^t is circular on the center and radius of C.*

Definition 8. *Circle Formation: Given a circle C and a set of n robots $\mathcal{R} = \{r_1, r_2, \cdots, r_n\}$ initially under collision-free and connected configuration, arrange them under a collision-free, connected and C-circular configuration through a sequence of collision-free configurations.*

Definition 9. *Uniform Transformation: Given a set of n robots $\mathcal{R} = \{r_1, r_2, \cdots, r_n\}$ initially under a collision-free, connected and C-circular configuration, arrange them under a collision-free and uniformly-circular configuration through a sequence of collision-free configurations.*

After reaching a consensus on the common circle, we decompose the uniform circle formation problem into two subproblems, namely circle formation and uniform transformation seen above. Informally speaking, the two steps are to form a circle first and then to transform into a uniform circle. Here, we assume for simplification that all robots will not isolate themselves from the system during the movements of the whole system.

B. Network Construction

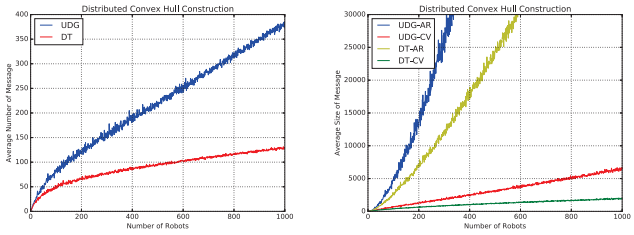
In this section, each robot constructs a local communication network with identifiers by running Algorithm 1. Initially, each robot r_i broadcasts a message containing its identifier and its position p_i to the neighboring robots within R_c. Upon receiving a piece of message $Msg_j = \{r_j, p_j\}$, robot r_i will testify whether there is any other robot within the circle whose diameter is the segment with ending points p_i and p_j. If there is no other robot in between, r_i will add r_j into its neighboring list N_i.

Algorithm 1 Network construction for each robot r_i

Input: r_i: robot identifier; R_a: active range; $S_i^{relative}$: the relative positions of the robots within R_s
Output: N_i: neighbor list of robot r_i
Begin:
  $N_i \leftarrow \emptyset$ ▷ Initialize the neighbor list as an empty set
  $S_i^{absolute} \leftarrow S_i^{relative} + (x_i, y_i)$
  Broadcast the message containing the identifier and the position $Msg_i = \{r_i, p_i\}$ to its neighbors
  while Receive a message $Msg_j = \{r_j, p_j\}$ from a neighboring robot **do**
    $d_{ij} \leftarrow distance(p_i, p_j)$▷ Measure the distance between r_i and r_j
    if $d_{ij} < R_a$ **then**
      $circle_{ij} \leftarrow$ a circle with midpoint of r_i and r_j as the center, and d_{ij} as the diameter
      if No points in $S_i^{absolute}$ are inside $circle_{ij}$ **then**
        $N_i \leftarrow N_i \cup \{r_j\}$
      end if
    end if
  end while
  return N_i
End

By running Algorithm 1, each robot r_i can construct a new local communication network N_i, which is also called the 1-hop neighbors of robot r_i. Compared to the original network which is a circle with radius R_c, the number of messages can be remarkably reduced by running the algorithm of convex hull construction and cardinality estimation. It is obvious that all the communication routes in the whole network are bi-directional.

To demonstrate the advantages of utilizing our algorithm of network construction, we compare it with the original network by running the convex hull construction algorithm. We run the distributed convex hull construction algorithm in our network and the original network in an MRS with 3 to 1000 robots and 10 different initial configurations for each number of robots. Then, we calculate the average number of messages needed in our network and the original network. Fig. III-C



(a) Average number of messages (b) Average size of messages

Fig. 4. Analysis on number of message and size of messages

shows the results, in which UDG (unit disk graph) means the original network while DT (Delaunay triangulation) means our network. It is oblivious that our network can significantly reduce the number of messages.

C. Convex Hull Construction

In this section, we propose a distributed convex hull construction algorithm (seen in Algorithm 2) to make all of the robots aware of the convex hull of all robots.

In the algorithm, each robot r_i maintains a local convex hull $conv_i$. When the algorithm is starting up, the primary convex hull for each robot r_i only contains the position p_i itself. Then, each robot r_i exchanges its local convex hull with all of its 1-hop neighbors. Upon receiving a convex hull from another robot, the robot locally runs Graham's scan [17] to merge its local convex hull with the received one. The resulting convex hull is saved as the new local convex hull. Each robot continues this procedure until the local convex hulls in two successive rounds are identical.

Another straightforward algorithm to make all robots aware of the common convex hull is that each robot sends all of its neighbors to its 1-hop neighbors repeatedly. In this algorithm, every robot can know all the robots in the system finally and can compute the convex hull. To evaluate our distributed convex hull construction algorithm, we compare it with the straightforward algorithm. We run the distributed convex hull construction algorithm and straightforward algorithm in DT network and UDG network in an MRS with 3 to 1000 robots and 10 different initial configurations for each number of robots. Then, we calculate the average number of messages as well as the average size of messages needed. Fig. 4 shows the results, in which AR means the straightforward algorithm while CV means our algorithm. According to the figure, we can see that the number and size of messages can be significantly reduced by using our convex hull construction algorithm.

D. Distributed Cardinality Estimation

In this section, we propose Algorithm 3 to estimate the number of robots in the whole system. Algorithm 3 is adapted from Algorithm 2 by modifying the content of messages and adjusting the updating rule upon receiving messages. After the execution of Algorithm 3, all robots will have a common sense on x_a^i for all $a \in [1, k]$. These parameters will be used for estimating the total number of robots in Section III-E.

Algorithm 2 Distributed convex hull Construction for each robot r_i

Input: N_i: One hop neighbor list; p_i: position; r_i: robot identifier
Output: $conv_i$: the convex hull of the MRS
Begin:

```
if not initialized then
    conv_i ← {(r_i, p_i)}                          ▷ Each robot maintain a local convex hull as conv_i. Initially, the convex hull of each robot is only the position of itself.
    conv'_i ← conv_i
    Fin ← ∅    ▷ Fin records the robots whose messages have been handled in current round.
    for each r_k ∈ N_i do
        msglist(r_k) ← an empty message queue
    end for
    Broadcast Msg_i = {r_i, conv_i} to its neighbors
else if receive message Msg_j = {r_j, conv_j} then
    if r_j ∈ Fin then
        msglist(r_j).push(conv_j)
    else
        conv'_i ← merge(conv'_i, conv_j)           ▷ Merge two convex hulls by running Graham's scan algorithm locally
        Fin ← Fin ∪ {r_j}
        if |Fin| = |N_i| then
            if conv'_i = conv_i then
                return conv_i
            end if
            conv_i ← conv'_i
            broadcast conv_i to r_i's neighbors
            Fin ← ∅
            for each r_k ∈ N_i do
                conv_k ← msglist(r_k).pop()
                conv'_i ← merge(conv_i, conv_k)
                Fin ← Fin ∪ {r_k}
            end for
        end if
    end if
end if
```

End

The general idea of Algorithm 3 is that each robot selects one of l slots to be placed and finally makes a consensus on all the occupied slots. This procedure is repeated k times. In the algorithm, l and k are parameters which can be used to achieve different accuracy requirements. The details will be discussed in Section III-E.

E. Consensus on Circle

In this section, each robot determines the radius and the center of the circle to form. To determine the center, we use the resulting convex hull from Section III-C. To determine the radius, we consider two methods. In the first method, we calculate the area of the convex hull and divide the area of the convex hull by the area of one robot to get the maximum

Algorithm 3 Distributed cardinality estimation for each robot r_i

Input: N_i: One hop neighbor list; l: an integer parameter; k: another integer parameter
Output: $\hat{n_i}$: the estimation of total number of robots
Begin:

for $a \leftarrow 1$ to k **do**
$x_a^i \leftarrow$ a variable of l bits with all bits to be 0
$rand \leftarrow$ a random $0/1$ string of $l-1$ bits
$y \leftarrow$ the number of leading continuous '0's in $rand$ from left hand
Set x_a^i's $y+1$ bit to be 1
$x_a'^i \leftarrow x_a^i$
end for
In Algorithm 2, incorporate $\{x_1^i, \cdots, x_k^i\}$ into message Msg_i. The updating rule of x_a^i is $x_a^i \leftarrow (x_a^i \mid x_a^j)$ when receiving Msg_j. Finally, when returning $conv_i$, also returns x_a^i for all $a \in [1, k]$

End

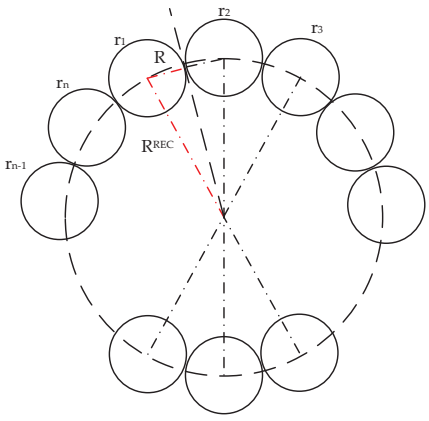


Fig. 5. Caclulating the radius of the common circle using the estimation of the number of robots.

number of robots in the area. In the second method, we estimate the number of robots using the results from Section III-D. Then, we take the minimum value of the results from the two methods as the estimated number of robots in the system. In this way, each root can have very high possibility to find a place on the circle boundary. The performance of the algorithm will be evaluated afterward.

After estimating the number of robots, each robot can calculate the radius of the common circle as shown in Fig. 5. The exact algorithm is shown in Algorithm 4.

In [18], the authors give a local cardinality estimator using the theorem proved in [19] as follows:

$$y = \Sigma_{a=1}^{k} y_a^i$$

$$\hat{p_i} = 1.2897 \times 2^{\frac{y}{k}}$$

The performance of the estimator can be guranteed by satisfying *accuracy requirement*. The accuracy requirement is said

Algorithm 4 Determination of the center and radius of the circle to be formed for each robot r_i

Input: x_a^i: parameters for cardinality estimation generated in Algorithm 3; k: parameter used in Algorithm 3; $conv_i$: convex hull generated in Algorithm 2
Output: O_i: the center of the circle; R_i^{CIR}: the radius of the circle
Begin:

for $a \leftarrow 1$ to k **do**
$y_a^i \leftarrow$ the number of leading continuous '1's in x_a^i
end for
$y \leftarrow \Sigma_{a=1}^{k} y_a^i$
$\hat{p_i} \leftarrow 1.2897 \times 2^{\frac{y}{k}}$ ▷ The result of distributed cardinality estimation
$O_i \leftarrow \frac{1}{|conv_i|} \Sigma_{j=1}^{|conv_i|} conv_i.p_j$ ▷ The center of the target circle
$area_i \leftarrow$ area of $conv_i$ ▷ Area of the convex hull
$\hat{q_i} \leftarrow \gamma \frac{area_i}{\pi R^2}$ ▷ The convex hull contains at most $\frac{area_i}{\pi R^2}$ robots. We multiply it by a parameter $\gamma \in [1, 2]$ without loss of generality.
$\hat{n_i} \leftarrow min(\hat{p_i}, \hat{q_i})$
$R_i^{CIR} \leftarrow R / \sin \frac{\pi}{\hat{n_i}}$
return $\{O_i, R_i^{CIR}\}$

End

to be satisfied if $Pr[|\hat{n} - n| \leq \beta n] \geq 1 - \alpha$, in which α is the *error probability* and β is the *confidence interval* (also called error bound). Adapting the theorem in [19] to our context, we get the constraints for k to guarantee the performance:

Theorem 1. *Given α and β, the accuracy requirement is satisfied if $k \geq \max\{[\frac{-\sigma_\infty c}{\log_2(1-\beta)}]^2, [\frac{\sigma_\infty c}{\log_2(1+\beta)}]^2\}$, where c is obtained by solving $1 - \alpha = erf(\frac{c}{\sqrt{2}})$, erf is the Gaussian error function.*

F. Circle Formation

After the previous steps, all the robots have made a consensus on a common circle to form. Let O be the center of the circle, R^{CIR} be the radius of the circle and CIR be the common circle. In this section, Algorithm 5 is proposed to solve the circle formation problem.

At the beginning, each robot r_i calculates the point D_i as the intersection point of the circle CIR and the ray Ray_i from O to p_i. And D_i serves as the target position of robot r_i. While robot r_i moves to D_i, it detects whether there is any other robot ahead along Ray_i. If so, it changes its target to the next position D_{i+1} that is clockwise available for a robot on the circle. Fig. 6 shows an example of Algorithm 5, in which solid arrows are the planned routes of the robots. Since there are enough spaces to place all the robots on the circle, there must be an available position for each robot.

G. Uniform Transformation

After circle formation, the robots are going to do uniform transformation as stated in Section II-B. First, the robots

Algorithm 5 Circle Formation for each robot r_i

Input: O: the center of the circle; R^{CIR}: the radius of the circle; r_i: robot identifier; α: speed control parameter
Output: All robots are located on the boundrary of a circle
Begin:
$Cir \leftarrow$ the boundary of a circle with O as the center and R^{REC} as radius
$Ray_i \leftarrow$ the radial from O to p_i
$D_i \leftarrow$ the intersection point of O_i and Ray_i
while r_i does not reach Cir **do**
  if There is no other robot along Ray_i **then**
    Move along Ray_i to D_i
  else
    $D_{i+1} \leftarrow$ the next robot position clockwise adjacent to D_i on the Cir
    $Ray_{i+1} \leftarrow$ the ray from O to D_{i+1}
    if There is no other robots along the shortest way from p_i to Ray_{i+1} **then**
      Move along the shortest way from p_i to D_{i+1}
    else
      Wait in this round
    end if
  end if
end while
End

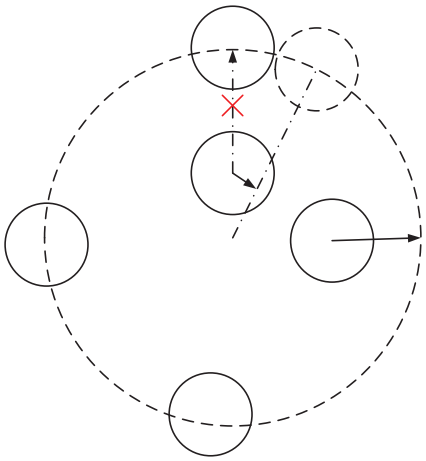


Fig. 6. Move to the boundary of the circle

can be aware of the total number of robots by clockwise passing information in a straightforward way. Then, each robot calculates the desired final robot inter-distance, namely d, on the uniform circle. For robot r, r^+ is notated as the clockwise neighboring robot on the circle. We adapt the algorithm in [11] to Algorithm 6 to solve the uniform transformation problem.

IV. Experimental Results

To demonstrate the usefulness and evaluate the performance of our solution proposed in Section III, we deploy a realistic

Algorithm 6 Uniform transformation for each robot r_i

Input:d: the desired final robot inter-distance; $p+$: the position of r_i's successor; CIR: the circle to form
Output: All robots are uniformly located along a circle
Begin:
$d^+ \leftarrow distance(p_i, p^+)$
if $d^+ > d$ **then**
  Move toward the point p on the circle CIR at distance d from p^+, remaining on the circle CIR during the movement.
end if
End

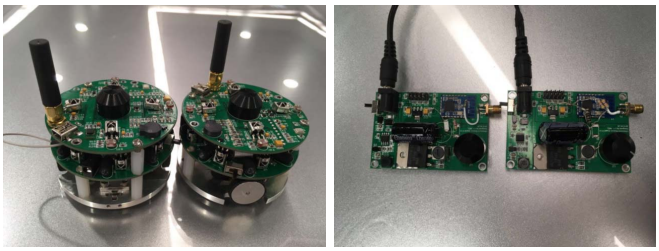


Fig. 7. The robots and the beacons

test-bed. Our realistic test-bed, as shown in Fig. 7, is composed of three components, which are a localization system, multiple (currently 8) intelligent robots and a programming environment.

The localization system consists of two anchor beacons, which are used for robotic localization. Each anchor beacon is composed of a 2.4G wireless communication module, an STM32 Microcontroller(MCU), an ultrasonic transmitter, a temperature sensor, and a battery. In the procedure of localization, the 2.4G wireless communication module and the ultrasonic transmitter will send signals simultaneously. Due to the different propagation velocities of the 2.4G wireless signal and the ultrasonic signal, each robot receives the two signals at different moments. As a result, each robot can calculate the distance between itself and the beacon using the principle of TDOA (time difference of arrival). After calculating the distance between the robot itself and the two anchor beacons, the robot can calculate its location on the planar platform. Since the velocity of the ultrasonic signal varies under different temperatures, we add a temperature sensor to achieve a more precise localization.

The robots, namely PiBots (The Hong Kong Polytechnic University Intelligent Robot), are the 2nd version of our design (version 1 is presented in [20]) in our laboratory. All the robots are in the same shape and with the same size, which is approximately a cylinder with 7cm radius and 18cm height. The struct diagram of the robot is shown in Fig. 8. Each robot is composed of two boards, the upper board and bottom board, which are powered by the power management unit. In both of the boards, there is an MCU which is responsible for data storage and processing. Inside the MCUs, data can be stored

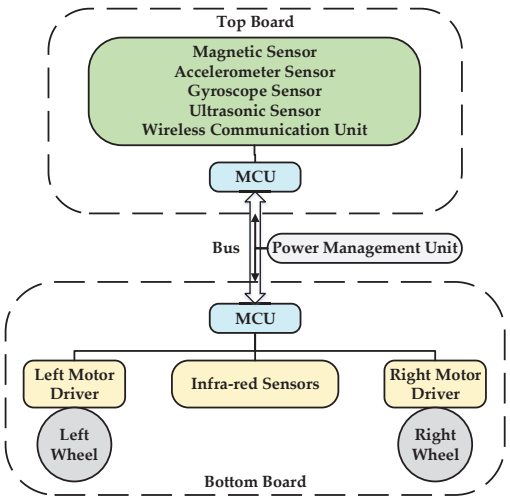


Fig. 8. Structure Diagram of a Robot

in either 8Mbit static random access memory (SRAM) or 512Kbit flash memory. The two boards are connected by a SPI (Serial Peripheral Interface) bus to achieve data transmission.

The bottom board includes two motor drivers, eight infra-red sensors, a motor feedback signal processor and an STM32 MCU. The motor drivers can control the wheels with given speeds separately and the motor feedback signal processor can get the current speeds of the wheels. The eight infra-red sensors can detect whether there are obstacles or not in eight directions. The upper board includes an STM32 MCU, three controlled buttons, a 2.4G wireless communication unit, a 2K EEPROM, an ultrasonic receiver, a 3-axis accelerator sensor, a 3-axis gyroscope, a 3-axis geomagnetic sensor, etc. The wireless communication unit, which uses IEEE 802.15.4 as its protocol, enables communication between robots. Other functionalities achieved by the bottom board includes localization, direction awareness, etc.

The programming environment is based on FreeRTOS, a popular real-time operating system kernel for embedded devices. In FreeRTOS, programmers can define a set of tasks (similar to threads in operating systems) with priorities to be executed concurrently. For example, in our implementation of uniform circle formation, a task to control motors, a task for communication, and a task for the purpose of user-defined application are used. Then, in every time unit, the tasks will be executed one by one according to the pre-defined priorities. Also, to simply the programming task, we utilize the programming model proposed in [21]. Finally, we deploy our algorithm for uniform circle formation in our test-bed with parameters $l = 10$, $k = 4$ and $\gamma = 1.2$. The running examples are shown in Fig. 1, in which scenarios with different numbers of robots and different initial configurations are considered. The results demonstrate the effectiveness and practicability of our algorithm.

V. Related Works

The circle formation problem was first discussed and further improved by Sugihara and Suzuki [22]. Their approach is based on heuristics and works in SSYNC for the formation of an approximate circle instead of a perfect one. Later on, researchers cast light on a particular case of circle formation called uniform circle formation in which the robots must be arranged at regular intervals on the boundary of a circle. A remarkable progress is attained by Defago and Konagaya [6]. They proposed a protocol in the SSYNC model and formally prove their approach to converge toward a uniform circle. With respect to ASYNC model, Flocchini et al. [12] addressed the uniform circle formation problem by moving to smallest enclosing circle (SEC) and avoiding *pre-regular* circumstance.

However, all the above algorithms are based on the assumptions of unlimited visibility and the punctiform hypothesis. With unlimited visibility, each robot is aware of the positions of all robots. However, robots can only sense their surroundings within a certain range in reality. With limited visibility, a robot might not even know the total number of robots. Also, assuming unlimited visibility makes this procedure unscalable and computationally expensive, since each robot has to process the information of all robots. On the other hand, robots are represented as points under the punctiform hypothesis. These assumptions greatly simplify the problem of uniform circle formation by avoiding the details of the formation of a consensus of the circle size, collision avoidance, etc.

The circumstance of limited visibility is understandably challenging. To our knowledge, only a small number of algorithmic results are known under the limited visibility scenario, even in problems other than uniform circle formation. Multi-robot gathering with limited visibility is well discussed and investigated [23][24]. On uniform circle formation, known results are conducted by Dutta et al. [25] and Datta et al. [26]. However, both of them assume for convenience that the circle to form is given in advance. In reality, consensus on the circle is not trivial at all.

The boundary is a "tightly" wrapped contour around the configuration of robots. In this paper, we incorporate the convex hull for boundary detection. In computational geometry, the algorithm of Grahams scan [17] is well-known to compute the convex hull of a set of points in the two-dimensional space. The algorithm of Grahams scan requires the computation to be performed in a central computer, which does not meet our requirement that the robots form a distributed system and are with limited visibility. To our knowledge, only few algorithms [27][28][29] are known for distributed boundary detection. However, these algorithms only make each robot aware of whether or not it is on the boundary. Each robot is not aware of all the robots on the boundary.

Cardinality estimation, i.e., counting the approximate number of tags in a given region is widely discussed in RFID systems [30][31]. However, in existing works, only the central computation station, i.e., the RFID readers, are aware of the estimation result. To our knowledge, the problem of cardinality

estimation has not been discussed yet in fully-distributed systems.

VI. Conclusion

The problem of uniform circle formation is one of the important coordination problems in multi-robot systems. The question of whether robots with actual size and limited sensing and communication range could form a uniform circle, to our knowledge, has remained open. In this paper, we formulate the uniform circle formation problem in a distributed multi-robot system and propose a three-phase approach, namely consensus on circle, circle formation and uniform transformation towards solving it. Inside our approach, we propose several new algorithms, i.e., distributed convex hull construction and distributed cardinality estimation, which can be used in general distributed systems. After designing our distributed algorithm, we deploy it in our realistic test-bed and have done solid experiments. The results imply the effectiveness and practicability of our algorithms.

VII. Acknowledgement

The research is partially supported by the ANR/RGC Joint Research Scheme with number A-PolyU505/12, NSFC/RGC Joint Research Scheme with number N_PolyU519/12 and NSFC with number 61332004. This work was partially supported by grant TR32054: "Digital signal processing, and the synthesis of an information security system", Serbian Ministry of Science and Education.

References

[1] L. Luo, N. Chakraborty, and K. Sycara, "Provably-good distributed algorithm for constrained multi-robot task assignment for grouped tasks," *Robotics, IEEE Transactions on*, vol. 31, no. 1, pp. 19–30, 2015.

[2] D. Panagou, D. M. Stipanović, and P. G. Voulgaris, "Distributed coordination control for multi-robot networks using lyapunov-like barrier functions," *IEEE Transactions on Automatic Control*, vol. 61, no. 3, pp. 617–632, 2016.

[3] T. Arai, E. Pagello, and L. E. Parker, "Editorial: Advances in multi-robot systems," *IEEE Transactions on robotics and automation*, vol. 18, no. 5, pp. 655–661, 2002.

[4] Z. Yan, N. Jouandeau, and A. A. Cherif, "A survey and analysis of multi-robot coordination," *International Journal of Advanced Robotic Systems*, vol. 10, 2013.

[5] I. Chatzigiannakis, M. Markou, and S. Nikoletseas, "Distributed circle formation for anonymous oblivious robots," in *International Workshop on Experimental and Efficient Algorithms*. Springer, 2004, pp. 159–174.

[6] X. Défago and A. Konagaya, "Circle formation for oblivious anonymous mobile robots with no common sense of orientation," in *Proceedings of the second ACM international workshop on Principles of mobile computing*. ACM, 2002, pp. 97–104.

[7] N. Fujinaga, Y. Yamauchi, H. Ono, S. Kijima, and M. Yamashita, "Pattern formation by oblivious asynchronous mobile robots," *SIAM Journal on Computing*, vol. 44, no. 3, pp. 740–785, 2015.

[8] K.-K. Oh, M.-C. Park, and H.-S. Ahn, "A survey of multi-agent formation control," *Automatica*, vol. 53, pp. 424–440, 2015.

[9] Y. Dieudonné, O. Labbani-Igbida, and F. Petit, "Circle formation of weak mobile robots," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 3, no. 4, p. 16, 2008.

[10] Y. Dieudonné and F. Petit, "Squaring the circle with weak mobile robots," in *International Symposium on Algorithms and Computation*. Springer, 2008, pp. 354–365.

[11] P. Flocchini, G. Prencipe, and N. Santoro, "Self-deployment algorithms for mobile sensors on a ring," in *International Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics*. Springer, 2006, pp. 59–70.

[12] P. Flocchini, G. Prencipe, N. Santoro, and G. Viglietta, "Distributed computing by mobile robots: uniform circle formation," *Distributed Computing*, pp. 1–45, 2014.

[13] I. Suzuki and M. Yamashita, "Distributed anonymous mobile robots: Formation of geometric patterns," *SIAM Journal on Computing*, vol. 28, no. 4, pp. 1347–1363, 1999.

[14] W. Cook and A. Rohe, "Computing minimum-weight perfect matchings," *INFORMS Journal on Computing*, vol. 11, no. 2, pp. 138–148, 1999.

[15] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer, "Hard tasks for weak robots: The role of common knowledge in pattern formation by autonomous mobile robots," in *Algorithms and Computation, 10th International Symposium, ISAAC'99, Chennai, India, December 16-18, 1999, Proceedings*, 1999, pp. 93–102.

[16] I. Suzuki and M. Yamashita, "Distributed anonymous mobile robots," in *SIROCCO'96, The 3rd International Colloquium on Structural Information & Communication Complexity, Siena, Italy, June 6-8, 1996*, 1996, pp. 313–330.

[17] R. L. Graham, "An efficient algorith for determining the convex hull of a finite planar set," *Information processing letters*, vol. 1, no. 4, pp. 132–133, 1972.

[18] C. Qian, H. Ngan, Y. Liu, and L. M. Ni, "Cardinality estimation for large-scale rfid systems," *IEEE transactions on parallel and distributed systems*, vol. 22, no. 9, pp. 1441–1454, 2011.

[19] P. Flajolet and G. N. Martin, "Probabilistic counting algorithms for data base applications," *Journal of computer and system sciences*, vol. 31, no. 2, pp. 182–209, 1985.

[20] S. Jiang, J. Liang, J. Cao, and R. Liu, "An ensemble-level programming model with real-time support for multi-robot systems," in *Pervasive Computing and Communication Workshops (PerCom Workshops), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1–3.

[21] S. Jiang, J. Cao, Y. Liu, J. Chen, and X. Liu, "Programming large-scale multi-robot system with timing constraints," in *Computer Communication and Networks (ICCCN), 2016 25th International Conference on*. IEEE, 2016, pp. 1–9.

[22] K. Sugihara and I. Suzuki, "Distributed algorithms for formation of geometric patterns with many mobile robots," *Journal of robotic systems*, vol. 13, no. 3, pp. 127–139, 1996.

[23] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer, "Gathering of asynchronous robots with limited visibility," *Theoretical Computer Science*, vol. 337, no. 1-3, pp. 147–168, 2005.

[24] J. Lin, A. S. Morse, and B. D. Anderson, "The multi-agent rendezvous problem. part 2: The asynchronous case," *SIAM Journal on Control and Optimization*, vol. 46, no. 6, pp. 2120–2147, 2007.

[25] A. Dutta, S. G. Chaudhuri, S. Datta, and K. Mukhopadhyaya, "Circle formation by asynchronous fat robots with limited visibility," in *International Conference on Distributed Computing and Internet Technology*. Springer, 2012, pp. 83–93.

[26] S. Datta, A. Dutta, S. G. Chaudhuri, and K. Mukhopadhyaya, "Circle formation by asynchronous transparent fat robots," in *International Conference on Distributed Computing and Internet Technology*. Springer, 2013, pp. 195–207.

[27] Y. Wang, J. Gao, and J. S. Mitchell, "Boundary recognition in sensor networks by topological methods," in *Proceedings of the 12th annual international conference on Mobile computing and networking*. ACM, 2006, pp. 122–133.

[28] J. McLurkin and E. D. Demaine, "A distributed boundary detection algorithm for multi-robot systems," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*. IEEE, 2009, pp. 4791–4798.

[29] P. Guo, J. Cao, and K. Zhang, "Distributed topological convex hull estimation of event region in wireless sensor networks without location information," *IEEE transactions on parallel and distributed systems*, vol. 26, no. 1, pp. 85–94, 2015.

[30] M. Kodialam and T. Nandagopal, "Fast and reliable estimation schemes in rfid systems," in *Proceedings of the 12th annual international conference on Mobile computing and networking*. ACM, 2006, pp. 322–333.

[31] X. Liu, K. Li, J. Wu, A. X. Liu, X. Xie, C. Zhu, and W. Xue, "Top-k queries for multi-category rfid systems," in *Computer Communications, IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on*. IEEE, 2016, pp. 1–9.