A Token Passing Approach for Circle Formation by Multiple Mobile Robots

Avinash Gautam
Department of Computer Science and Information Systems
Birla Institute of Technology and Science
Pilani, India
avinash@pilani.bits-pilani.ac.in

Sudeept Mohan
Department of Computer Science and Information Systems
Birla Institute of Technology and Science
Pilani, India
sudeeptm@pilani.bits-pilani.ac.in

Virendra Singh Shekhawat
Department of Computer Science and Information Systems
Birla Institute of Technology and Science
Pilani, India
vsshekhawat@pilani.bits-pilani.ac.in

Abstract—**This paper proposes a weakly centralized distributed approach for positioning multiple mobile robots in a circular formation based on token passing. The problem of the circle formation with multiple robots which are arbitrarily placed on a 2D plane requires all robots to be uniformly positioned (i.e., at an equal angular distance of $\frac{2\pi}{N}$, where N = number of robots) on the circle circumference. The suggested approach is a leader-follower approach wherein it is the leader robot which computes the uniform positions on the circle circumference for all the follower robots. The problem of circle formation is divided into two sub-problems (a) leader selection and (b) finding enviable positions for the follower robots from the set of uniform positions computed by the leader robot. Both these problems are solved by token passing so as to reduce communication load on both the leader and the follower robots. The introduction of token passing makes it a weakly centralized framework thereby reducing the burden on the leader robot.**

Keywords-multi-robot coordination, pattern formation, software framework, leader-follower, token passing

I. Introduction

The interest in distributed autonomous robot systems has increased considerably in recent years. The problem of controlling a set of autonomous, mobile robots in a distributed fashion has been studied extensively, but almost exclusively from engineering and from an artificial intelligence point of view. Problems such as flocking, consensus, coverage and pattern formation are some of the important problems that have been studied over the past few years. The goal of these problems is to develop distributed protocols and control laws that allow the reaching of complicated global goals.

Pattern formation of multiple robots is highly desirable in almost all the problem domains where multiple robots are utilized and are required to coordinate. The formation is defined as a configuration in a bounded workspace, where each robot is at a desired distance from its neighbors. The desired formation is specified in terms of relative distances, so the formation can be achieved in any part of the workspace, and it can have any orientation.

The rest of this paper is divided into seven sections. Section-II reviews several approaches on the pattern formation problem, that is, how a group of robots can be controlled to get into and maintain a formation. Section-III describes the model of the system and several entities. A novel token passing based approach for positioning multiple mobile robots into a circular formation is described in Section-IV. In [5] the leader robot computes the uniform positions on the circle circumference for its followers and then directs the follower robots to their respective positions which are closest to them. This centralization has obvious disadvantages i.e., (a) single point of failure, (b) leader robot is burdened with additional responsibilities of distributing follower robots on the circle circumference and, (c) the amount of interaction, i.e., the number of messages exchanged between the leader and the follower robots are significant. On the other hand in the present approach we have eliminated the requirement of a leader to assist the followers in deciding their positions on circle circumference. The customization of token passing method reduces the number of messages exchanged between the leader robot and its followers and is discussed in Section-V. In Section-VI simulation results along with a comparison of the approach presented in [5] with the token passing approach is discussed. Conclusion and scope for the future work is discussed in Section-VII.

II. Related Work

This section examines some important research works in multi-robot pattern formation. The pattern formation problem is defined as the coordination of a group of robots to get into and maintain a formation with a certain shape, such as a wedge or a chain. Pattern formation is one of the key characteristics of many living beings which include birds flocking, fish schooling, and ant forming chains [6]. These species make patterns when they migrate from one feeding and breeding ground to another, prevent attacks of predators, etc. From an engineering perspective the best paradigm which captures the characteristics of cooperation in biological swarms (e.g., ants, termites,

This work is being funded by CAIR-DRDO, Ministry of Defense (Government of India) via its Grant# CAIR/ROB/CARS-2010/2. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding agency.

bees, wasps, and bacteria) is multi-agent systems [3]. In multi-agent systems the agents can be categorized as independent agents and cooperative agents. One of the chief characteristics of independent agents is defined as emergent cooperation [7] wherein the agents appear to be working together, but from the agent's viewpoint they are not. They are simply carrying out their own individual behavior. The emergent cooperation is surely visible in biological swarms. This intelligent group behavior is a result of actions performed by relatively simple individuals that are solely based on neighbor interactions and local information from the environment inhabited by the agent-collectively [8]. In particular, biologically inspired and reflex-based approaches involve deploying a large collection of minimalist agents (possessing a primitive sensing, locally-interacting, and weak decision-making capabilities) that cooperate to achieve quite remarkable task-performing capabilities [7]-[11]. Several empirical studies and simulations have been conducted in multi-robot pattern formation [12]-[15]. Also researchers have given strong theoretical frameworks from a mathematical perspective [16]-[18] which have laid down the foundations of such empirical studies. It is difficult to precisely establish a comparison between these approaches as the capabilities of the robots which are considered in all these research works are different. Also the assumptions and the model of the environment in which the robots are situated are different. Common assumptions about the sensing, navigation, communication and computational capabilities of the robots cannot be taken, yet what is observed is that many of the research works have considered low-end robots which do not share their coordinate system with other robots, have minimal or no understanding of other robots situated in the environment, and minimal or no communication between the robots. These robots communicate with each other through the environment, as they try to mimic the emergent behavior of the biological swarms [12], [15], [19]. The research works under this category lack guarantees that the actual formation will emerge. If the desired formation emerges out, it is a loose formation (not a strict geometric shape). On the other hand, there are few research works which consider the use of robots which are more involved with each other and share a lot of information. In [20], [21], [22], global knowledge is assumed: each robot knows the position of all others. Thus, when all robots are in place (positioned), the desired formation is established. The current approach discussed in this paper also makes use of global knowledge i.e., common global reference frame and explicit communication to bring multiple mobile robots in a strict circular shape/pattern.

III. THE SYSTEM MODEL

The system we have described in this paper is made up of several important entities which require separate treatment and demands independent descriptions:

(1) *Modeling Environment*: As described in [2] the environment should provide an exploitable design abstraction for building multi-agent system applications. Mainstream research work on cognitive agent systems [3, 4] treats the environment as a means of communication (i.e., message exchange), and a container for agents and resources. The environment is a software 2D plane in which the point robots are situated and provides a means of communication to the point robots. When the environment is first initialized, the robots are placed on arbitrary locations within the bounds of the environment. At any instance of time the state of the environment describes the number of robots lying on it along with their respective positions in terms of their x and y coordinates. Two important environmental primitives are: (a) Calculate smallest enclosing circle (SEC) [1] (b) Select the leader robot.

(2) *Modeling Robot*: The robots are, *homogeneous* (they can't be differentiated on their processing and sensing capabilities). All robots are independent of each other and there is no priori central authority. Each robot has a unique rank, which is used for resolving conflicts in various situations. At any instance of time the robot's state describes its distance from the center of the smallest enclosing circle (SEC) and it can also tell if it is positioned or not (i.e., if a robot is successfully positioned on the circumference of the SEC or not). The robots are *asynchronous* (there is no central clock), *mobile* (they are allowed to move on a 2D plane), *anonymous* (they are a priori indistinguishable), *oblivious* (they don't explicitly remember their past states and actions). The robots are identical network processes/ applications which are represented as a point object on a software 2D plane. Each robot is capable of generating and passing token to other robots.

(3) *Modeling RobotLeader*: A RobotLeader is a virtual entity [5] which is used to decorate a robot with leader capabilities once it is selected as a leader. The RobotLeader supports primitives to register, remove and notify the robot followers. Also it calculates the polar coordinates of the points which are at an equal angular distance θ, on the circumference of the SEC. We call these points as positions. Initially all positions are unoccupied. The leader robot notifies the follower robots one unoccupied position at a time, for them to compete among themselves and decide the winner of that position. If there are N robots the leader robot calculates N-1 positions for N-1 follower robots.

(4) *Modeling RobotFollower:* A RobotFollower is a virtual entity [5] which is used to decorate a robot with follower capabilities. All the robots except the one which is selected and decorated as a leader are decorated as RobotFollower. The RobotFollower support primitives to generate and pass tokens, using which they compete for their nearest position on the circumference of the SEC. All follower robots are notified by the leader robot, a position "p", on the circumference of the SEC. All the notified followers queries themselves to know if they want to compete for this position or not.

IV. THE APPROACH

To make multiple mobile robots fall in a circular formation we present a novel token passing based approach. The approach presented here is described in terms of coordination algorithms run by several entities which are described in Section-III.

A. The notations used by the Algorithms:

$rank_R \rightarrow$ rank of robot R

$R = \{R_1, R_2, R_3... R_N\} \rightarrow$ set of point robots

$pos(R_i)$ → i^{th} robot's current position on the plane
$SEC(R)$ → smallest enclosing circle of R
SEC_{center} → center of the smallest enclosing circle
SEC_{radius} → radius of the smallest enclosing circle
SEC_{circum} → circumference of the smallest enclosing circle
$dist(X,Y)$ → Euclidian distance between point X and point Y
R_L → Robot leader
LC → set of leader contender robots
$RF = \{RF_1, RF_2, RF_3 \ldots RF_N\}$ → set of follower robots
$P = \{P_1, P_2, P_3 \ldots P_N\}$ → set of uniform positions on the circumference of SEC (R)
$LP(P_i)$ → set of position contenders of position "P_i"
Token → T(X) = {id, rank, dist}, where id = rank of robot that has generated this token, rank = robot's rank, dist = distance of robot from point X.

Algorithm 1: Environment finds out the smallest enclosing circle, SEC(R).
Participating Entities: *Environment*

Input : *R*
Output : *SEC(R)*

The algorithm is proposed in [1] and is fairly efficient with a running time of O(nlogn). The algorithm is abstracted by the Environment which is exposed as an environmental primitives and support to the robots. The algorithm is locally run in the Environment (which is a software 2D plane) and the definition of the smallest enclosing circle, SEC(R) is communicated to all the robots. The decision for considering Algorithm-1 as an environmental primitive has eliminated the requirement of individual robots calculating SEC(R) for themselves.

Algorithm 2: Select the leader robot
Participating Entities: *Environment, Robot*

Leader selection is an abstraction of the Environment and is initiated by the Environment.

Input : *R, SEC(R)*
Output : *$rank_R$ of Robot nearest to SEC_{center}*

```
/* each robot calculates its distance from the center of the smallest enclosing circle and communicates it to the environment */
foreach Ri ε R {
    Ri.calculateDistance(SECcenter)
    Ri communicates its distance from the SECcenter to the Environment
}
/* for each robot the environment finds out if the distance of this robot is less than or equal to half of the radius of smallest enclosing circle. All such robots are added to the list of leader contenders */
foreach Ri ε R {
    if (dist(Ri, SECcenter) <= ½(SECradius))
    LC.add(Ri)
}
/* the environment randomly chooses one of the leader contender robot to generate a token of leadership. This robot is called as TokenGenerator */
Randomly choose Ri from LC
Let Ri generate T(SECcenter)
| T(SECcenter).id := rankRi                         |
| T(SECcenter).rank := rankRi                       | /*Token is initialized */
| T(SECcenter).dist := dist(pos(Ri), SECcenter)     |
/* the token is passed to the next robot on the virtual ring, for the robots to compete for leadership */
[Algorithm 2.1] rankleader := Compete for leadership(T(SECcenter), Ri+1)
/* decorate robot 'RL' whose rank is equal to the rank of leader robot as Robot-Leader */
Decorate RL as RobotLeader ‖ rankRi == rankleader
/* decorate all the robots except the 'RL' as RobotFollowers */
foreach Ri ε {R - RL} {
    Decorate Ri as RobotFollower
    RF.add(Ri) /* add them to the list of robot followers */

}
/* let the leader robot register all the follower robots */
foreach RFi ε RF { RL.register(RFi) }
```

Algorithm 2.1: Compete for leadership
Participating Entities: *Robot*

This algorithm enables the robots to compete for leadership by passing token of leadership to their successors on the virtual ring. The algorithm terminates when the token returns back to the TokenGenerator robot.

Input : *$T(SEC_{center}), R_{i+1}$*
Output : *$T(SEC_{center}).rank$*

```
do { /* run till the token doesn't come back to the TokenGenerator */
/* if the distance of the robot is greater than or equal to the distance to the SECcenter as specified in the token by its predecessor, then pass the token to the next leader contender robot without modification */
if((dist(Ri+1, SECcenter) >=  T(SECcenter).dist))
    pass T(SECcenter) to the next leader contender without modifying the token
/* if the distance of the robot is less than the distance specified by its predecessor in the token, then update the token by writing its own rank and distance to the SECcenter in the token fields, and pass on the token to the next robot */
else if((dist(Ri+1, SECcenter) <  T(SECcenter).dist))
    T(SECcenter).rank := rankRi+1
    T(SECcenter).dist := dist(Ri+1, SECcenter)
    pass T(SECcenter) to the next leader contender robot
}while(T(SECcenter).id != TokenGenerator.rank)
return T(SECcenter).rank /* return the rank of the robot specified in the token */
```

Algorithm 3: Calculate uniform positions on SEC_{circum}
Participating Entities: *RobotLeader*

The *RobotLeader* calculates N uniform positions on the SEC_{circum}, separated by an angular distance θ. The algorithm is described in [5]. Here it is described formally with pseudo code.

Input : *RF (set of robot followers)*
Output : *P (set of uniform positions on SEC_{circum})*

```
θ = (2*PI)/N, where N = size of RF
Randomly choose RFi, call it FirstFollower
Calculate
    x = FirstFollower.x – RobotLeader.x
    y = FirstFollower.y – RobotLeader.y
for i=0 to N {
    Pi.x = RobotLeader.x + x*cos(i*θ) – y*sin(i*θ)
    Pi.y = RobotLeader.y + y*cos(i*θ) – x*sin(i*θ)
    P.add(Pi)
}
```

Algorithm 4: Decide positions of *RobotFollowers* on SEC_{circum}
Participating Entities: *RobotLeader, RobotFollower*

Input : *P (set of uniform positions on SEC_{circum})*
RF (set of robot followers)
Output : *All the robot followers in RF, excluding those who were*

leader contenders are successfully positioned on the SEC_{circum}

---

foreach $P_i \varepsilon P$ { /* for each uniform position */
  foreach RF_i *in RF* { /* for each robot follower */
    /* the robot leaders checks if it is not positioned and is not a leader contender */
    if(!RF_i*.isPositioned() && !*RF_i*.isLeaderContender())*
      /* robot leader notifies the follower this position */
      R_L*.notifyOneUniformPositionAtTime(*P_i*)*
    /* if the distance of this robot follower from this position is less than the distance from the center of smallest enclosing circle */
    if (dist(RF_i*,* P_i*) < dist(*RF_i*,* SEC_{center}*))*
      /* add this follower to the list of point contenders */
      $LP(P_i).add(RF_i)$
  }
  /* the leader robot randomly chooses one of the follower robot from the list of position contenders to generate a token for this position */
  Randomly choose one RF_i *from* $LP(P_i)$
  Let RF_i *generate* $T(P_i)$
  $T(P_i).id := rank_{Pi}$
  $T(P_i).rank := rank_{Pi}$ */*Token is initialized */*
  $T(P_i).dist := dist(pos(RF_i), P_i)$
  /* the token is passed to the next follower robot on the virtual ring, for the follower robots to compete for this position*/
  [Algorithm 4.1] $rank_{positionWinner} :=$ *Compete for position(*$T(P_i)$*,* RF_{i+1}*)*
  /* robot follower whose rank is equal to the rank of position winner is asked to move to this position */
  $move(RF_x , P_i) \parallel rank_{RFx} == rank_{positionWinner}$
  /* set this robot follower as positioned and the position as occupied */
  RF_x*.setPositioned(true)* Λ P_i*.setOccupied(true)*
}
/* robot leader determines all such positions which are left unoccupied */
foreach $P_i \varepsilon P \parallel P_i$*.isUnoccupied() == true {*
  /* robot leader determines all the followers which are yet not positioned */
  foreach RF_i *in RF* $\parallel$ RF_i*.isPositioned() == false {*
    /* notify one unoccupied position at a time to all the follower robots which are not positioned */
    $R_L.notify(P_i, RF_i)$
    Robot follower which is nearest to P_i *moves to this position*
  }
}

Algorithm 4.1: Compete for position P_i
Participating Entities: *RobotFollower*

Input	:	$T(P_i), RF_{i+1}$
Output	:	$T(P_i).rank$

---

do { /* run till the token doesn't come back to TokenGenerator */
/* if the distance of the robot from position P_i is greater than or equal to the distance specified as specified in the token by its predecessor, then pass the token to the next position contender without modification */
$if((dist(RF_{i+1}, P_i) >= T(P_i).dist))$
  pass $T(P_i)$ *to the next position contender without modifying the token*
/* if the distance of the robot from position P_i is less than the distance specified by its predecessor in the token, then update the token with its own rank and distance from position P_i in the token fields, and pass on the token to the next position contender */
else $if((dist(RF_{i+1}, P_i) < T(P_i).dist))$
  $T(P_i).rank := rank_{RFi+1}$
  $T(P_i).dist := dist(RF_{i+1}, P_i)$
  pass $T(P_i)$ *to the next leader contender robot*
}while($T(P_i).id != RF_{i+1}.rank$*)*
return $T(P_i).rank$ /* return the rank of the robot specified in the token */

V. THE TOKEN PASSING & CUSTOMIZATION

Token Ring uses a ring topology whereby the data is sent from one machine to the next and so on around the ring until it ends up back where it started. A logical ring is constructed by *RobotFollowers*. Each *RobotFollower* knows who is next in line after itself using the ranks assigned to them, such that, $\{RF_0 \rightarrow RF_1 \rightarrow RF_2 \ldots RF_N \rightarrow RF_0\}$, with an exception that *RobotLeader* is not the part of the topology. Therefore the *RobotFollower* with a rank one less than the rank of the *RobotLeader* will have its next in line *RobotFollower* with a rank equals to the *RobotLeader's* rank plus 1.

To distribute the burden on any particular entity and the number of messages exchanged between various entities in the system, two rules are suggested:

Rule 1: when selecting the leader robot [*Algorithm 2*] only those robots which are in the inner circle see Fig. 1, are leader contenders and are eligible to compete for leadership. This leads to less number of messages being exchanged between the robots (i.e., the token is circulated among the leader contenders only). This creates a deadlock situation: *If all the robots are located in the inner circle see Fig. 1, then no robots will compete for leadership* – this deadlock is resolved by using the leader selection algorithm suggested in [5].

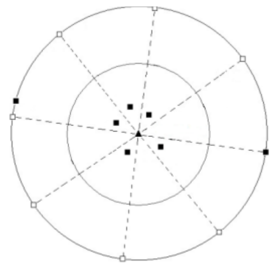

Fig. 1: All RobotFollowers are clustered around the SEC_{center}

Rule 2: Only those robot followers will compete for a given position "P_i" on the SEC_{circum} which are closer to the position "P_i" and away from the SEC_{center}. Less number of messages are exchanged between the *RobotFollowers* (i.e., the token is circulated between few position contenders only). This will result in two deadlock situations: (a) If all robot followers are clustered inside a circle with a radius equal to ½ (SEC_{radius}), such that, inside the inner circle, see Fig. 1, then no robots will compete for any of the uniform positions calculated by the follower. In this case the *RobotLeader* assigns the right positions to the *RobotFollowers* on the SEC_{circum} [5]. (b) If all or some of the *RobotFollowers* are clustered around a specific position(s), see Fig. 2, then all or some of the *RobotFollowers* will end up competing for only one position as they will always be closer to the SEC_{center} compared to other positions (i.e., some of the unoccupied positions will have no contenders), again such cases are resolved by *RobotLeader* by distributing *RobotFollowers* to their respective unoccupied positions (positions without any contenders) which are closer to them.

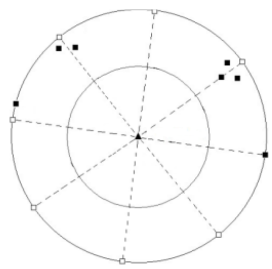

Fig. 2: RobotFollowers clustered around few positions, leaving some of the positions without any contenders

A sample run of the simulation for fifteen robots is shown below, wherein Fig. 3(a) is showing the spatial distribution of fifteen robots when the system is initialized on the 2D plane and Fig. 3(b) is showing the spatial distribution of the robots after one run of the algorithm. Also in Fig. 3(b) it can be noticed that the two kinds of clustering situations leaves the multi-robot formation in a non-uniform state (some of the positions remained unoccupied and some of the robots are not positioned).

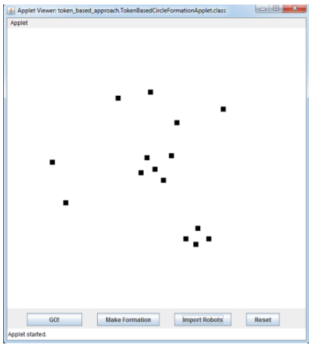

Fig. 3(a) Spatial distribution of Robots (N = 15)

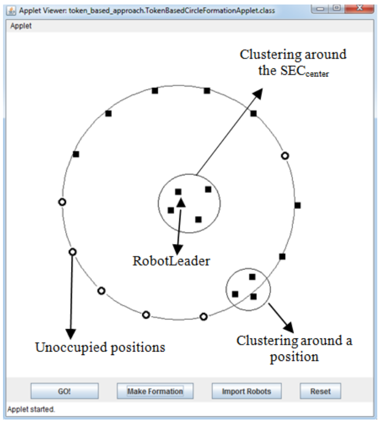

Fig. 3(b) Spatial distribution of Robots (N = 15) with unresolved clustering situations, after one run of the simulation

Fig. 3(c) is the best case of the simulation run wherein both the situations which lead to the issue of clustering are resolved. As discussed earlier, in the present approach, if the clustering situation arises, the leader robot handles them by brute-force methods both leader selection and position allocation.

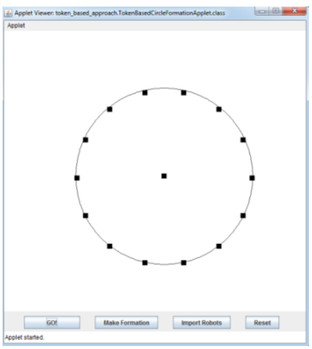

Fig. 3(c) Uniform circle formation of multiple robots (N = 15), after the second run of the simulation, with proper resolution of the two clustering situations

VI. SIMULATION RESULTS

This section compares the strongly centralized approach used in [5] for the circular formation of multiple mobile robots arbitrarily placed on a 2D plane with the novel token passing based approach developed in this paper. Simulations have been carried out for 5 to 50 robots. The number of messages that are exchanged between the robots (leader-follower and follower-follower) is greatly dependent on the spatial distribution of the robots on the 2D plane. Therefore we have maintained same spatial distribution for both the approaches in all the simulations.

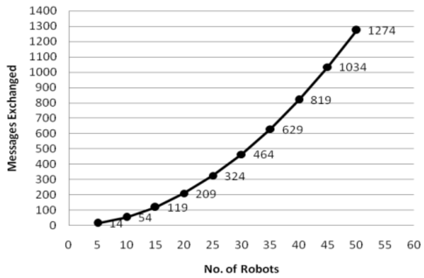

Fig. 4: Messages exchanged between the leader and the follower robots in strongly centralized approach [5]

The graph shown in Fig. 4 depicts the number of messages sent from the leader robot to the follower robots. The strongly centralized approach used in [5] requires the leader robot to compute and decide the uniform positions on the SEC_{circum} for all of its robot followers, therefore the communication is always initiated by the leader robot and the followers respond (i.e., strict leader-follower). On the other hand the token passing based approach presented in this paper has distributed decision making which requires both leader-follower, and follower-follower kind of interactions on various occasions as explained in the algorithms. The graph in Fig. 5(a) demonstrates the number of messages exchanged between the leader robot and all of its followers in token passing based approach. It can easily be observed from Fig. 4 and Fig. 5(a) that the number of messages exchanged between the leader and the follower robots have significantly gone down.

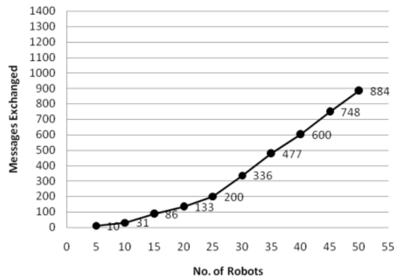

Fig. 5(a): Messages exchanged between the leader and the follower robots in token passing based approach

The graph shown in Fig. 5(b) corresponds to the number of messages exchanged among the follower robots to compete for a uniform position on the SEC_{circum}.

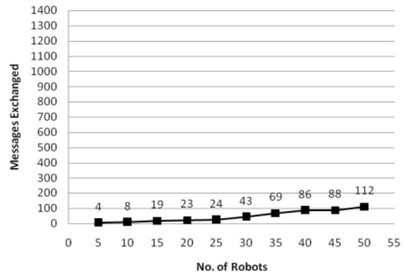

Fig. 5(b): Messages exchanged between follower robots in token passing based approach

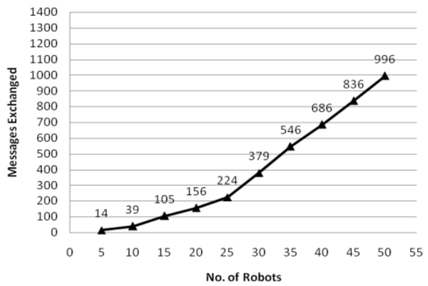

Fig. 5(c): Total number of messages exchanged in token passing based approach

In the end the graph of Fig. 5(c) stands for the total number of messages exchanged between the leader-follower and the follower-follower robots. Easy conclusions can be derived by looking at the graph of Fig. 4 and Fig. 5(c) that the token passing based approach not only reduces the communication between the leader and the follower robots but it is overall an effective technique which significantly cuts down the communication making it a weakly coordinated approach.

VII. CONCLUSION AND SCOPE FOR FUTURE WORK

The token passing based approach suggested in this paper is a superior approach compared with the centralized approach [5] for circular formation of multiple mobile robots. There is a significant fall in the number of messages sent and received by the leader robot to its followers. Within the followers also the number messages sent are very less. It can also be observed that the total number of messages exchanged in token passing based approach is much less than the total messages exchanged in the centralized approach. Clustering of robot followers around the SEC_{center} and around the positions have been dealt with the centralized algorithms used in [5]. For the future work dispersion algorithm(s) to avoid clustering can be implemented which would make the system more distributed and further reduce the centralization.

REFERENCES

[1] Skyum, S.; "A simple algorithm for computing the smallest enclosing circle", Information Processing Letters, Vol. 37, no. 3, Feb-18 1991, Pages 121-125, ISSN 0020-0190, 10.1016/0020-0190(91)90030-L.

[2] Weyns, D.; Omicini, A.; Odell, J.; "Environment as a first class abstraction in multiagent systems", Autonomous Agents and Multi-Agent Systems, Vol. 14, Issue 1, Feb. 2007, Pages 5-30.

[3] Huhns, MN.; Stephen, ML. Multiagent Systems and Societies of Agents. In Weiss, G. (Ed.), Multiagent systems, a modern approach to distributed artificial intelligence. Cambridge, MA, USA: MIT Press, 1998.

[4] Foundation for intelligent physical agents, FIPA abstract architecture specification. http://www.fipa.org/repository/bysubject.html [July 20, 2012].

[5] Gautam, A.; Mohan, S.; Misra, J.P., "A practical framework for uniform circle formation by multiple mobile robots," Industrial and Information Systems (ICIIS), 2012 7th IEEE International Conference on , vol., no., pp.1,5, 6-9 Aug. 2012.

[6] Camazine, S.; Deneubourg, J. L.; Franks, N. R.; J. Sneyd, G. Theraulaz, and E. Bonabeau. Self-Organisation in Biological Systems. Princeton University Press, NJ, 2001.

[7] E. Bonabeau.; M. Dorigo.; G. Theraulaz.; Swarm Intelligence: From Natural to Artificial Systems, Oxford University Press, 1999, pp. 183–203.

[8] G. Beni.; J. Wang.; "Distributed robotic systems and swarm intelligence", IEEE International Conference on Robotics and Automation, 1991, pp. 1914–1919.

[9] G. Dudek.; M. Jrenkin.; E. Milios.; D. Wilkes.; "A taxonomy for swarm robots", IEEE/RSJ International Conference on Intelligent Robots and Systems, Japan, 1993, pp. 441–447.

[10] Mataric M. J. Interaction and intelligent behavior, PhD Thesis, Department of Electrical Engineering and Computer Science, MIT, May 1994.

[11] K. Sugawara.; T. Watanbe.; "Swarming robots—foraging behavior of simple multi-robot system", IEEE/RSJ International Conference on Intelligent Robots and Systems, EPFL, Lausanne, Switzerland, 2002, pp. 2702–2707.

[12] Yamaguchi, H.; Arai, T.; Beni, G.; "A distributed control scheme for multiple robotic vehicles to make group formations", Robotics and Autonomous Systems, Vol. 36, Issue 4, 30 September 2001, Pages 125-147, ISSN 0921-8890, 10.1016/S0921-8890(01)00133-6.

[13] Barfoot, T.D.; Clark, C.M.; "Motion planning for formations of mobile robots", Robotics and Autonomous Systems, Vol. 46, Issue 2, 29 February 2004, Pages 65-78, ISSN 0921-8890, 10.1016/j.robot.2003.11.004.

[14] Krishnanand, K.N.; Ghose, D.; "Formations of minimalist mobile robots using local-templates and spatially distributed interactions", Robotics and Autonomous Systems, Vol. 53, Issues 3–4, 31 December 2005, Pages 194-213, ISSN 0921-8890, 10.1016/j.robot.2005.09.006.

[15] Ikemoto, Y.; Hasegawa, Y.; Fukuda, T.; Matsuda, K.; "Gradual spatial pattern formation of homogeneous robot group", Information Sciences, Vol. 171, Issue 4, 13 May 2005, Pages 431-445, ISSN 0020-0255, 10.1016/j.ins.2004.09.013.

[16] Chen, F.; Chen, Z.; Liu, Z.; Xiang, L.; Yuan, Z.; "Decentralized formation control of mobile agents: A unified framework", Physica A: Statistical Mechanics and its Applications, Vol. 387, Issues 19–20, August 2008, Pages 4917-4926, ISSN 0378-4371, 10.1016/j.physa.2008.04.018.

[17] Défago, X.; Souissi, S.; "Non-uniform circle formation algorithm for oblivious mobile robots with convergence toward uniformity", Theoretical Computer Science, Vol. 396, Issues 1–3, 10 May 2008, Pages 97-112, ISSN 0304-3975, 10.1016/j.tcs.2008.01.050.

[18] Elor, Y.; Bruckstein, A. M.; "Uniform multi-agent deployment on a ring", Theoretical Computer Science, Vol. 412, Issues 8–10, 4 March 2011, Pages 783-795, ISSN 0304-3975, 10.1016/j.tcs.2010.11.023.

[19] Fredslund, J.; Mataric, M.J.; "A general algorithm for robot formations using local sensing and minimal communication", IEEE Transactions on Robotics and Automation, Vol. 18 , Issue: 5, Oct 2002, Pages 837-846.

[20] Balch, T.; Arkin, R. C.; "Behavior-based Formation Control for Multi-robot Teams", IEEE Transactions on Robotics and Automation, Vol. 14, No. 6, December 1998, pp. 926–939.

[21] Chen, Q.; Luh, J. Y. S.; "Coordination and Control of a Group of Small Mobile Robots", IEEE International Conference on Robotics and Automation (ICRA-1994), pp. 2315–2320, San Diego, 1994.

[22] M. Anthony Lewis, Kar-Han Tan, High Precision Formation Control of Mobile Robots Using Virtual Structures Autonomous Robots 4, pp. 387–403 (1997).