Protected Group Control System for Mobile Robots

Elena Basan [a], Maria Lapina [b], Massimo Mecella (*) [b,c]

[a] South Federal University, 2 Chekhov St., Taganrog, 347928, Russia
[b] North-Caucasus Federal University, 2, Kulakov prosp., Stavropol, 355029, Russia
[c] Sapienza Università di Roma, DIAG, via Ariosto 25, Roma, 00185, Italy

Abstract

The purpose of this work is to present a secure group control system for ground mobile robots. The paper includes the analysis of existing group control algorithms, a description of the implementation of some of the algorithms, their testing and analysis of characteristics. When we solve the problem, we adhere to the following aspects: The multivariance of simulated situations, the feasibility of algorithms within the framework of existing computer systems and their applicability within the framework of real group control systems, the use of cryptographic information security tools using distributed methods, the development of a model for group control of robots and testing of developed solutions should be carried out using a simulator, approbation of the developed solutions is carried out using a full-scale model, which includes a set of single-board computers, between which a wireless network is organized, robots perform the main task of analyzing the terrain and building a terrain map, while the terrain map should be distributed between robots and updated periodically, robots must be able to avoid obstacles and collisions with each other, perform the task of moving towards the goal and the task of following the group leader. Security solutions must ensure the property of integrity, proof of authorship, and trust in the data. Implementing the choice of the group leader must also provide the ability to select a trusted entity as the group leader.

Keywords [1]

Group control system, mobile robots, key generation procedure, signature, signature verification operations

1. Introduction

Robots are increasingly used to effectively solve practical problems, and some tasks are completely impossible to implement without automation. The first includes the tasks of production, protection, constant analysis of the environment. The second type of tasks include demining, exploration of dangerous areas, participation in hostilities. Robots often have to operate in a non-deterministic environment, i.e. environment that may have its own characteristics over time, for example, changing weather conditions, the structure of the territory itself (the appearance of new objects), etc.

The use of groups of robots is often most appropriate. There are several reasons for this:

- The production cost of one robot in a group is often lower than the production of a single prototype,
- The use of groups of robots can significantly speed up the execution of tasks assigned to such systems,
- The use of groups of robots allows expanding the scope of their application, due to greater efficiency, compared to single robots and a person.

YRID-2020: International Workshop on Data Mining and Knowledge Engineering, October 15-16, 2020, Stavropol, Russia
EMAIL: ele-barannik@yandex.ru (Elena Basan); norra7@yandex.ru (Maria Lapina); mecella@diag.uniroma1.it (Massimo Mecella)
(*) This work was performed while visiting researcher in North-Caucasus Federal University
ORCID: 0000-0001-6127-4484 (Elena Basan); 0000-0001-8117-9142 (Maria Lapina); 0000-0002-9730-8882 (Massimo Mecella)

- A group of robots is a set of similar or different types of robots, united by one or a group of common tasks. Identical robots are understood to have:
- Identical design features,
- Identical functional purposes,
- Same functionality.

Group Robot Control System (GRCS) controls group robots. GRCS can implement methods of centralized or decentralized management strategy. Centralized group management systems are distinguished by the presence of the so-called a leader which, based on the information coming from the robots of the group, is engaged in the overall coordination of the entire system. In decentralized control systems, such a leader is absent, therefore, each robot in this system is engaged in coordinating the entire system. The implementation of the GRCS involves many components:

- A set of robotic nodes,
- A set of goals,
- The presence of the environment.

Each robot is characterized by many parameters, for example, charge, carried weight, range of input devices (sensors), output (network devices) information. The goals given to a group of robots depend on the functions of the GRCS that it can perform. Among others, the following can be distinguished:

- Exploration of a given territory,
- Tasks for movement, patrolling, transportation of goods within a given deterministic territory,
- Execution of commands not related to the GRCS (for example, carrying out pollution measurements, notification of violations, etc.).

The environment is a space in which the subjects of the GRCS carry out their activities. Environments, as mentioned earlier, are divided into deterministic (predictable) and non-deterministic (unpredictable). The environment may contain obstacles in the form of stationary (walls) or moving objects (other robots).

Thus, the task of the group control system is reduced to determining at each moment of time such actions of each robot that will provide a given amount of change in social functionality over a time interval for given states of robots [1-4].

The goal is to develop and implement a model of a robotic group control system in a protected design. To achieve this goal, it is necessary to solve the following tasks:

- Analyze the architecture of group management,
- Conduct an analysis of group control algorithms,
- Demonstrate the implementation of the selected group control algorithms,
- Implement the described group control algorithms in an emulated environment or using single board computers connected in a wireless environment,
- Conduct experimental measurements to identify the characteristics of the algorithms under consideration.

2. Development of the architecture of a group control system for robots

Based on the hierarchical structure [5], it is necessary to single out several main components of the management system (MS). To do this, consider the levels of the control system and highlight the important tasks that are solved at each of them.

The strategic management level is the level that solves distribution problems for the entire group. The most important tasks here are:

- The task of target allocation,
- The task of forming groups of robots,
- The task of forming the formation of mobile robots,
- The task of exchanging trusted messages.

2.1. Development of the target allocation algorithm

The solution to this problem boils down to the fact that robots receive a list of goals and must decide how to choose goals in order to fulfill them most effectively. Efficiency is here understood as the smallest total time of movement of all robots to their chosen targets.

There are a large number of goal distribution algorithms, but most of them come down to compiling a matrix that includes robots, goals, and attractiveness coefficients [6]. In this paper, a general scheme for compiling such matrices will be given and I will give an example of calculating goals.

An MxN matrix is compiled, where M is the number of targets (horizontally), N is the number of robots ready to fulfill these goals (vertically). Each robot and each target is given its own serial number in this matrix. Further, at the intersection of each target and each robot, the coefficient "target attractiveness" is set. The larger this coefficient, the more convenient the target is for the robot. To calculate this coefficient, first of all, it is necessary to take into account the distance from the robot to the target, obtained taking into account the trajectory of this robot. Thus, the coefficient will be inversely proportional to the distance from the robot to the target. This value may already be enough to calculate the coefficient. But you can also use other values, for example, the remaining charge of the robot is directly proportional to the coefficient, the number of turns in the path from the robot to the target, inversely proportional to the coefficient. The more parameters are included in the calculation of this coefficient, the more accurately the optimal targets for robots will be calculated. Formula (1) shows an example of a formula for calculating the attractiveness coefficient from two values: the remaining charge in the battery (q) and the distance from the robot to the target (d):

$$k = \frac{q}{d} \quad (1)$$

Further, after the matrix is obtained, the algorithm for calculating the optimal goals is launched. It is executed until the number of assigned targets is equal to the number of robots, if there are fewer robots or the same number of targets [7]. Or until the number of targets assigned is equal to the number of targets, if there are more robots than targets.

The matrix usually includes the serial numbers of the robots along the vertical, the serial numbers of the targets along the horizontal and the coefficients of the attractiveness of a particular target for the robot at the intersection [8]. After such a matrix is ready, the goal calculation algorithm comes into operation [9]. There are several different variations of these algorithms. The following is the most general target allocation algorithm:

1. Cycle for I from 1 to N, where N is the number of robots;
    1.1. The maximum value of the row I of the matrix (maxTrgt) is selected;
    1.2. If maxTrgt > = of all values in column I then:
        1.2.1. Assign to robot I the target maxTrgt;
        1.2.2. All values in rows and columns numbered I = 0;
    1.3. else:
        1.3.1. Move to the next iteration of the cycle.

2.2. Development of an algorithm for organizing a group of robots in space

The task of organizing robots in space solves several issues, among which there were two main ones:

- Avoiding collisions of robots with obstacles and with each other,
- Implementation of the possibility of movement of a group of robots after the leader.

The method of potential fields [10] is based on the fact that the target should attract the robot, while obstacles, on the contrary, should repel it. The potential field method is usually used in a static environment to avoid static obstacles. It can also be used in dynamic environments. But in the case of static environments, the trajectory can be calculated in advance. In the case of dynamic environments, either constant recalculation of the vectors of interaction of the dynamic environment and constant correction of the trajectory according to these vectors by both robots, avoiding collisions with each

other, or modeling by each robot of its own behavior and the behavior of the robot with which a collision should be avoided, will be required.

The method of slowing down movement [8] by one of the robots, including its complete stop, is also capable of solving problems of avoiding collisions with dynamic objects. The main stage in the implementation of the algorithm is to determine the point of collision of two objects, and to decide which of the two robots will wait. In addition, one should either take into account the errors (turning time, uneven surfaces), calculate the waiting time for one robot to pass another, or organize a network communication between the robots, then the waiting robot will continue to fulfill its goals only after it receives a message from the second that the collision point has been overcome.

The problem with the previous two methods is that they do not solve the problem of the oncoming movement of two robots in a narrow corridor and some other special cases. There are 2 ways out of these situations: either use other collision avoidance methods for these cases, or take all measures to prevent these situations from arising. For example, divide the map between robots into zones, and only one robot will have the right to explore each of them. The problem lies in the fact that, getting on the map, robots have no idea not only about its size, but also about its shape. In such a situation, there is no way to divide the map so that everyone gets about the same area for research, and that part of the space allocated for the robot for research is not fenced off from it.

Tracking the leader's position is carried out using network interaction and sensors for tracking one's own position and thus comes down to the equipment that will be used at the stand, which includes robots that implement group control. At the same time, tracking a leader in an emulated environment comes down to the selected mechanisms for transferring data from one node to another.

The movement towards the position of the leader is carried out using pathfinding algorithms. These include Dijkstra's algorithm [5] and the A * family of algorithms [6]. Dijkstra's algorithm is effective when exploring a territory whose structure is unknown to the robot. If there is a fully or partially explored map and it is necessary to construct a trajectory of movement along given points lying on the investigated areas, the A * family is used. This family includes a large number of algorithms, the effectiveness of which differs depending on the structure of the map, its size and other factors. Therefore, the use of the root A * algorithm will be the optimal solution at the initial stages.

The problem with the described methods is that they do not solve the problem of the oncoming movement of two robots in a narrow corridor and some other special cases. There are 2 ways out of these situations: either use other collision avoidance methods for these cases, or take all measures to prevent these situations from arising.

To avoid the possibility of robots colliding with each other, you can divide the map between robots into zones, in which only one robot can be at a time. The disadvantage of this approach is the impossibility of using it in non-deterministic spaces, since once on the map, robots have no idea not only about its size, but also about its shape. In such a situation, there is no way to divide the map so that everyone gets about the same area for research, and that part of the space allocated for the robot for research is not fenced off from it, as in Figure 1. In this figure, the robot in the upper left sector is responsible for exploring the upper left square, but it cannot reach all of its zones without crossing other robot's zones and will consider these unreachable zones impassable. In this case, it remains to rely only on the fact that the lower left robot detects a passage to such an area.

Figure 1 shows the map explored by the robot, where the robot is marked with a red circle, an impassable obstacle with a black square, and investigated zones with white and colored squares.

An effective solution for most situations will be the implementation of the potential fields method, which will be included in the work when the robots approach each other in a certain way. At the same time, such a method should calculate the trajectories of the robots bypassing each other in advance, taking into account obstacles on the map and surface irregularities.

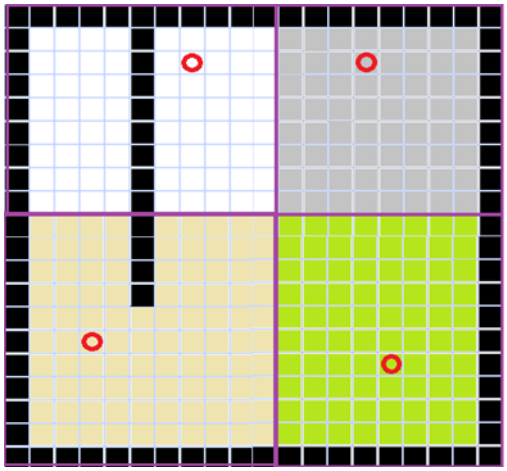

Figure 1: An example of incorrectly allocated study areas

The simplest solution to the problem of oncoming traffic in the corridor is to install charges of different sizes on the robots (for example, randomly in a given range, or according to their id). In this case, complete building-up can be avoided, but it will become more difficult to predict the behavior of such a system.

2.3. Development of an algorithm for exchanging trusted messages

When a group of robots investigates a dynamically changing map, there is a possibility that different participants will store different maps of the area at the same time. In addition, the study of the map by a group of robots suggests an increase in the effectiveness of this study relative to a single one. To do this, you must use the card synchronization algorithm. The general card synchronization algorithm works as follows:

1. One of the nodes sends its map and position on the map by broadcasting,
2. Other clients validate (described below),
3. If the card is valid, then:
    i. Refresh the map in your local storage.
4. Otherwise:
    i. Fix the map,
    ii. Broadcast the corrected map and your position.

The algorithm presented above assumes the use of a decentralized system. In a centralized system, the initiator node sends a change request to the leader, who authenticates. Thus, the task of synchronizing the card is reduced to the implementation of network interaction in the group of robots and the implementation of the authentication algorithm.

The exchange of trusted messages can be implemented in two main ways: using cryptographic methods [11] and using algorithmic methods for determining trusted messages.

3. Development and implementation of an experimental study of the GRCS, in the Group2DEmul environment

As part of the experimental study, the following algorithms for group control of robots were implemented:

- Algorithm of target allocation,
- Algorithm of follow up the leader,
- Algorithm for the exchange of trusted messages using cryptographic functions,
- Algorithms for trajectory planning, including wave algorithm, algorithm A*.

In this section, the power of these algorithms will be assessed depending on the various structures of maps, as well as the temporal characteristics of the algorithms being implemented.

3.1. Experimental studies of the target allocation algorithm

The effectiveness of the target allocation algorithm depends on how the target calculation matrix is composed. To determine the speed of the algorithm, let's take two extreme situations:

- The matrix is compiled as effectively as possible, in such a way that in each row of numbers there is a maximum number greater than the maximum number from a column,
- The matrix is compiled as ineffectively as possible, so that the maximum number greater than the maximum number from the column is always in the last row [12].

With a matrix compiled most efficiently with a matrix dimension of N*N, the speed of the algorithm tends to O(N). With the matrix compiled as ineffectively as possible with the matrix dimension N*N, the speed of the algorithm tends to

$$O(\frac{(N+1)N}{2}) \quad (2)$$

We give the time characteristics for the matrix, compiled in the most ineffective way in table 1.

Table 1

Time characteristics of the target allocation algorithm for the N*N matrix

N, for a matrix of size N*N	Elapsed time, sec
50	0.01
100	0.032
500	0.76

Based on the above time characteristics, we can conclude that similar target allocation algorithms are well suited for GRCS, the number of robots in which is about 100 units. It should be understood that the measurements are demonstrated for a matrix composed in the most ineffective way, and the probability of obtaining such a matrix in systems with N robots is

$$\frac{1}{N!} \quad (3)$$

3.2. Experimental studies of the leader selection algorithm

In this section, the following leader selection algorithms will be analyzed:

- Bully Election Algorithm,
- Token Ring Election Algorithm,
- RAFT.

These algorithms solve the problem of choosing a leader by communicating nodes with each other. The order and structure of such messages are different in the algorithms under study and cannot be compared. However, we can compare the sheer number of messages transmitted over the network to establish a leader.

The Bully Election Algorithm works in such a way that each node sends messages to nodes whose priority is higher than its own and waits for an acknowledgment message. Thus, the number of messages in the network depends on which node starts the procedure for re-election of the leader. In the worst case, the node with the lowest priority starts this procedure. Then a group of N robots for re-election of a leader in a network with all available nodes needs

$$\frac{(N-1)(N-2)}{2} = \frac{N^2-3N+2}{2} \quad (4)$$

request messages,

$$\frac{(N-1)(N-2)}{2} = \frac{N^2-3N+2}{2} \quad (5)$$

confirmation messages and N-1 \ notification messages about the election of a new leader. Thus, all

$$N^2 - 2N + 1 = (N-1)^2 \quad (6)$$

messages to establish a new leader in the network, where the node that started the procedure is selected as ineffectively as possible.

The Token Ring Election Algorithm operates in a ring structure, which minimizes the number of messages on the network. The chain of messages traverses such a ring 2 times, which means that in a group of N robots, the number of messages in the network for the implementation of the leader re-election algorithm will be equal to 2N.

The total number of messages of the RAFT algorithm cannot be estimated due to the fact that this algorithm works during the entire life cycle of the RAFT. But it is possible to evaluate only that part of the algorithm that is responsible for establishing a new group leader, from the same point of view as the previous two. To solve the problem of re-election of a leader, a node whose internal timer expires faster sends messages to the rest of the nodes in the network that it is a leader and is waiting for acknowledgments [13]. Thus, in a group of N robots, the number of messages in the network for implementing the leader re-election algorithm will be equal to

$$2(N-1). \tag{7}$$

Here is Table 1 illustrating the number of messages in the network for the described algorithms for a different number of nodes in the network:

Table 2

The number of messages required to solve the problem of leader re-election

	Number of nodes				
Algorithms	10	20	30	40	50
Bully Election	18	361	841	1521	2401
Token Ring Election	20	40	60	80	100
RAFT	18	38	58	78	98

From the results obtained, it can be seen that the number of messages required to solve the leader re-election problem for the Bully Election algorithm grows much faster than for other analyzed algorithms. At the same time, the number of messages of the Token Ring Election and RAFT algorithms is comparable and almost equal. However, due to the ring structure, the Token Ring Election algorithm is quite vulnerable, since if one node is lost in such a structure, the entire structure is destroyed and it must be reorganized. The RAFT algorithm avoids both of these problems by having a one-to-one structure and a fairly small number of messages required to solve the leader re-election problem.

3.3. Experimental studies algorithm follow up the leader

The algorithm for following the leader is reduced to calculating the positions of the nodes to place them around the leader. The algorithm calculates the position for each point using trigonometric functions. Thus, the speed of the algorithm for n nodes, the speed of the algorithm is O(n). The calculation time depends on the speed of the trigonometric operations in a specific programming language on a specific device. Here are the time characteristics for calculating the positions of nodes in table 3.

Table 3

Time characteristics of the leader movement algorithm for N nodes

N, number of nodes	Elapsed time, μs
1000	3982
5000	18395
10000	27970

Based on the results obtained, it can be concluded that this algorithm for placing robots around the leader is applicable even with a very large number of robots (about 10,000), which indicates the possibility of its use in GRCS.

3.4. Experimental studies of the algorithm for exchanging trusted messages using cryptographic functions

When implementing the algorithm for exchanging trusted messages using cryptographic functions, the time spent depends on the used DSA algorithm, the key length and the power of the computing facilities used. In chapter 4.3.1, the RSA algorithm for creating an DSA was considered, therefore, the time characteristics will be considered for it. As stated earlier, there are 3 main steps to signing a message:

- Generation of public and private keys,
- Formation of a message signed with a private key,
- Verification of the signature of the received message from the network.

We present the time characteristics for keys of different lengths for the indicated stages in tables 4-6. Signature and signature verification is carried out over a 1MB message. As you can see, the most time-consuming operation is the generation of keys, which is why it must be performed in advance, before the exchange of trusted messages begins. At the same time, signature and signature verification are rather fast operations and come down to raising numbers to a power. The duration of the signature and signature verification operations is the same in time. They differ only in the numbers raised to a power.

Table 4

Timing characteristics of generating public private keys of the RSA algorithm for keys of different lengths for message length

Key length, bit	Elapsed time, sec
128	0.028275012969970703
256	0.10507702827453613
512	0.24528908729553223
1024	3.6004180908203125

Table 5

Time characteristics of the signature of a 1 MB message using the RSA algorithm for keys of various lengths

Key length, bit	Elapsed time, sec
128	0.0003
256	0.0015
512	0.0024
1024	0.0081

Table 6

Timing characteristics of signature verification of a 1MB message using the RSA algorithm for keys of various lengths

Key length, bit	Elapsed time, sec
128	0.0003
256	0.0015
512	0.0024
1024	0.0081

Based on the results obtained, we can conclude that the key generation procedure is indeed the longest in relation to the signature and signature verification operations. The key generation time depends on its length; therefore, the task is to choose the most optimal option for the key length in terms of its length, and, consequently, information security and generation time. Based on the data given in Table 4, we can conclude that for the RSA algorithm the most effective key from this point of view is a 512-bit key. However, it must be remembered that this length may vary for other signature algorithms. In addition, the preparation of signing keys usually occurs before the very

procedure of interaction between robots and, therefore, does not affect the efficiency of such systems. The data described in Tables 5 and 6 indicate that although the signing and signature verification time also grows with the growth of the key length, it is still small enough to be taken into account.

4. Conclusion

The systems of group control were analyzed, the algorithms of the GRCS were described, their comparative analysis was carried out and their temporal performance characteristics were measured. The described algorithms were implemented in the emulated Group2DEmul environment and their temporal characteristics are given. The efficiency of the algorithms was evaluated in relation to the time of their execution for various input data. Based on the experimental measurements, it was concluded that the algorithms under study are effective.

The developed emulated environment has the functionality for describing most of the algorithms of the GRCS. The data obtained from this system can be used to make adjustments to the proposed algorithms before introducing them into real GRCS. In the future, it is planned to add tools for customizing the user interface through the user interface, add the ability to generate automatic reports on the effectiveness of the algorithm, which could be attached as a justification for the use of the algorithm when implemented in real GRCS.

5. References

[1] Pshikhopov V.Kh., Soloviev V.V., Titov A.E. and others. Group control of mobile objects in uncertain environments – 2015 – 9 p.

[2] Basan, A.S., Basan, E.S., Lapina, M.A., Kormakova, V.N., Lapin, V.G. Security methods for a group of mobile robots according to the requirements of Russian and foreign legislation: IOP Conference Series: Materials Science and Engineering, 2020, 873(1), 012031 DOI: 10.1088/1757-899X/873/1/012031

[3] Basan E., Lapina M. and Orel D. Host-based Method and System for Detecting Anomalies in Network Traffic for a Robotic System: CEUR Workshop Proceedings YSIP3 – Proceedings of the Young Scientist's Third International Workshop on Trends in Information Processing, 2019.

[4] Proshkin, N.A., Basan, E.S., Lapina, M.A., Klepikova, A.G., Lapin, V.G. Developing models of IoT infrastructures to identify vulnerabilities and analyse threats: IOP Conference Series: Materials Science and Engineering, 2020, 873(1), 012018 DOI: 10.1088/1757-899X/873/1/012018

[5] Description of the hierarchical structure of group control systems [Electronic resource]. – URL: http://roboticslib.ru/books/item/f00/s00/z0000003/st014.shtml/ (date of access: 22.09.20).

[6] D.A. Beloglazov, V.V. Soloviev et al. Target distribution method in groups of intelligent mobile robots – 2016 – 122 p.

[7] Description of the Bully Election Algorithm [Electronic resource]. – URL: https://www.cs.colostate.edu/~cs551/CourseNotes/Synchronization/BullyExample.html (date of access 22.09.20).

[8] Description of the Ring Election Algorithm [Electronic resource]. – URL: https://www.cs.colostate.edu/~cs551/CourseNotes/Synchronization/RingElectExample.html (date accessed 22.09.20).

[9] Description of the RAFT algorithm [Electronic resource]. – URL: http://thesecretlivesofdata.com/raft (date of access 22.09.20).

[10] Description of the method of potential fields [Electronic resource]. – URL: https://keldysh.ru/papers/2001/prep40/prep2001_40.html (date of access 22.09.20).

[11] Chichikin G.Ya., Semyonov D.A. Cryptosystem RSA, 2019 – 15 p.

[12] Implementation of the wave algorithm in Python. [Electronic resource]. – URL: https://pythondigest.ru/view/2681/ (date of treatment 22.09.20).

[13] Implementation of the A * algorithm in Python. [Electronic resource]. – URL: https://www.pvsm.ru/python/77932/ (date of access 22.09.20).