# Multi-Robot coalition formation in real-time scenarios

José Guerrero [1] and Gabriel Oliver

*Universitat de les Illes Balears,*
*Mathematics and Computer Science Department, Palma de Mallorca, Spain,*
*e-mail:{jose.guerrero, goliver}@uib.es*

**Abstract**

Task allocation is one of the main issues to be addressed in multi-robot systems, especially when the robots form coalitions and the tasks have to be fulfilled before a deadline. In general, it is difficult to foresee the time required by a coalition to finish a task because it depends, among other factors, on the physical interference. The interference is a phenomenon produced when two or more robots want to access the same point simultaneously. This paper presents a new model to predict the time to execute a task. Thanks to this model, the robots needed to carry out a task before a deadline can be determined. Within this framework, the robots learn the interference and therefore, the coalition's utility, from their past experience using an on-line Support Vector Regression method (SVR). Furthermore, the SVR model is used together with a new auction method called 'Double Round auction' (DR). It will be demonstrated that by combining the interference model and the DR process, the total utility of the system significantly increases compared to classical auction approaches. This is the first auction method that includes the physical interference effects and that can determine the coalition size during the execution time to address tasks with deadlines. Transport like tasks run on a simulator and on real robots have been used to validate the proposed solutions.

[1]**Corresponding Author**: José Guerrero
**Address:** Universitat de les Illes Baleares, Departament de Matemàtiques i Informàtica, Cra. de Valldemossa, Km. 7.5, 07122, Palma (Balears), Spain
e-mail: jose.guerrero@uib.es
Tel:(+34)971171391 Fax:(+34)971173003

*Keywords:*
Multi-robot, Task allocation, auction, real time, learning

## 1. Introduction

Multi-robot systems can provide several advantages compared to single-robot systems, for example: robustness, flexibility and efficiency. To make the most of these potential benefits, several problems have to be solved, specially when two or more robots can constitute coalitions to execute tasks. Of all the issues reported in the specialized literature, this paper focuses on the methods to select the best robot or set of robots to execute a task, which is commonly referred to as the 'Multi-Robot Task Allocation' (MRTA) problem.

In [1] Gerkey and Mataric, broke down the MRTA problem in three orthogonal axes: Multi-task robots (MT) vs. Single-task robots (ST) depending on whether multiple tasks can be assigned to the same robot or not; Single-robot tasks (SR) vs. Multi-robot tasks (MR), where SR means that only one robot can be assigned to a task, while MR means that several robots (a coalition) can concurrently execute a task; and finally, Instantaneous assignment (IA) vs. Time-extended assignment (TE) where in IA the allocation is made by not taking into account the future incoming tasks. In terms of this taxonomy, the present work is concentrated on the ST-MR-IA problem. Moreover, special attention is paid to the tasks that have to be fulfilled before a deadline, in which case the knowledge of how long the coalition will take to finish a task is essential in order to determine the proper alliances. The time needed to finish a task is frequently unknown. Thus, in general, anticipating if certain coalitions can meet the task's deadline or estimate their utility can be quite difficult. In this context, it is very arduous to accurately model the execution time of a task because it depends on many complex and dynamic factors, among which this paper will center its attention on the physical interference. The interference appears when some individuals of a community compete for a shared resource. In Multi-robot systems, a frequent situation occurs when two or more robots need to access the same point simultaneously. It has been demonstrated in different studies [2, 3], that the physical interference has an important impact on the system performance. This effect can be dramatic when the system has to address tasks with deadlines because the interference increases the time needed by the coalition to finish the task

and, therefore, diminishes the coalition utility. This paper proposes a new MRTA method for creating coalitions of mobile robots in real time scenarios, that is, in environments where the tasks must be executed before a deadline. When a coalition finishes a task before its deadline, the system obtains the maximum utility associated with that task. Otherwise, the utility decreases following some time function. The goal of the proposed method is to maximize the total utility obtained by the system. Auction methods have been thoroughly studied and they are frequently used as a task allocation solution. Furthermore, the current MR auction methods cannot predict the expected execution time of a task nor the coalition size, thus, they are unsuitable for real time situations. To solve the stated problem, this work presents a new auction method, called Double Round auction (DR). The DR auction is used together with a model of the physical interference, based on Support Vector Regression (SVR) [4], to predict how long the task execution will last. This paper also analyzes the effect of the task progress' monitoring on the system performance. The experimental results here presented show that, using a correct interference model, the task execution time can be predicted and, therefore, the monitoring process and their sensorial and communication resources, are not necessary. To test the proposed system a foraging like task is used, where multiple robots can cooperate to transport the same object. The experiments have been obtained using both a simulator and a set of four real robots (Pioneer 3DX). The results prove that the new proposed method clearly outperforms those obtained with classical auction mechanisms. The main goals of this paper are:

- To present a new MRTA method that takes into account the physical interference and uses this information to estimate the tasks' execution time.
- To describe a coalition formation procedure that can determine on-line the size of the workgroup required to meet the tasks' deadlines.
- To develop a new auction method, called Double Round auction, that improves the classical auction approaches in terms of total utility achieved in environments with deadlines.

This paper extends and analyzes in greater detail our previous work on MRTA algorithms explained in [5]. Although the double round concept was preliminary introduced in [6], here we include a detailed description and

analysis of the algorithm, an error recovery strategy and new experimental results, including tests on real robots.

The paper is organized as follows: section 2 reviews the relevant work in MRTA with special attention paid to auction methods; section 3 presents a formal definition of the problem to solve and details the real time foraging task; section 4 explains the techniques used to predict the execution time; section 5 introduces the Double Round auction task allocation algorithm; the experimental results are shown and analyzed in section 6 and, finally, the conclusions and future work are presented in section 7.

## 2. MRTA related works

Nowadays, swarm intelligence and auction based methods are the MRTA methodologies mostly used. Swarm methods are inspired by insect colonies' behavior, such as bees or ants, where a global action emerges from the interaction between very simple entities. In general, the swarm systems do not need communication protocols to coordinate the robots, but the complexity of the tasks they can carry out is strongly limited. Most swarm systems can only deal with SR problems with homogeneous robots. This restriction is superseded in the swarm system proposed by Ducatelle et al. [7] dealing with tasks without deadlines. The K. Hsiang's et al. swarm method [8] also allows the coalition formation to execute a task, but with very complex communication protocols. Moreover, very few swarm methods address time restricted tasks; an exception being the work by Oliveira et al. work [9], where only a robot can be assigned to each task. Some swarm methods have tried to reduce the physical interference effect dividing the environment into several areas which only some of the robots can access simultaneously [10, 11], but none of these approaches gives a quantitative value of the interference effect for calculating the expected tasks' execution time.

Due to the swarm method limitations, this work is focused on auction-like mechanisms, based on an explicit communication protocol to coordinate the robots' actions. In these kind of systems, the robots act as selfish agents bidding for the tasks. The bids are adjusted to the robots' interest (capacity) to carry out the goal. Thus, the robot with the highest bid, that is the best robot, wins the auction process and gets the task. A lot of work have been done to address SR problems with auction based methods [12, 13, 14, 15, 16, 17, 18, 19], some of them partially adapting ideas originally conceived in the computer engineering context to solve the task allocation problem. The E.

Jones' et al. study [18] takes into account time restrictions within an auction process where only a robot can be assigned to each task. The Jones' method uses a learning SVR model to obtain the bid value, but does not estimate the execution time. T. Lemaire's auction method [17] allows soft deadlines with a single robot per task. Other auction like approaches [20, 19] can deal with deadlines but with many restrictions regarding the execution time or the kind of tasks to execute.

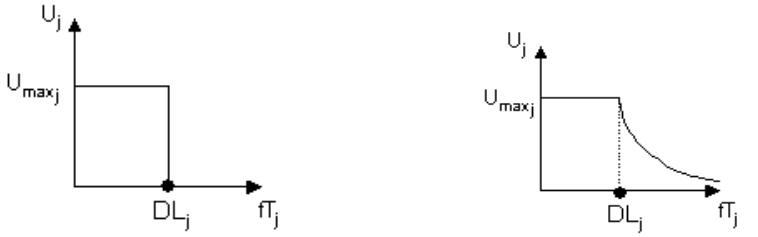
A few auction strategies, such as [21, 22, 23, 24, 25], allow us the addressing of MR problems, but none of them is able to deal with time restrictions associated with the tasks. E. Acebo and J. de-la-Rosa propose in [26] a swarm based method, with many characteristics from auction strategies, that solve MR problems with deadlines, but assume the execution time to be known in advance. Thus, not one of the referred methods takes into account the physical interference nor predicts the execution time of tasks. The MR auction method proposed in this paper overcomes these limitations and increases the number of finished tasks compared to the classical MRTA auction approaches.

## 3. Definition of the problem and complexity analysis

This section introduces the general MR problem to be solved and defines the transport task used to verify the proposed MRTA methods.

### *3.1. Problem statement*

The task allocation problem is defined as follows: Let $T = \{t_1, ...t_m\}$ be a set of m tasks and $R = \{r_1, ..., r_n\}$ the set of n robots to carry them out. Each task $t_j$ has the following characteristics: the amount of work needed to finish it ($taskWorkLoad_j$), a deadline ($DL_j > 0$), that is the time instant before which the task must be finished and, finally, a utility function $U_j(fT_j) > 0$, where $fT_j$ is the time passed since the task $t_j$ started. If the execution time of a task exceeds its deadline, $U_j$ decreases. The figure 1(a) shows an example of a hard deadline utility function, where $U_j$ is constant and equal $U_{maxj}$ while $fT_j < DL_j$ and abruptly drops to zero if the task cannot be executed by that time. Another kind of utility function is used for tasks with soft deadlines, an example of which can be seen in figure 1(b), in this case the value of $U_j$ progressively falls from $fT_j = DL_j$. Both, hard and soft deadline tasks have been used to test the performance of the proposed methods.




(a) Hard deadline utility function

(b) Soft deadline utility function

Figure 1: Utility function ($U_j$) examples

In general, each robot $r_i$ has a work capacity associated with each task $t_j$ ($workCapacity_{i,j}$) that represents the amount of work that the robot can process per time unit. Thus, the time to fulfill a task mainly depends on the number of robots of the coalition assigned to it, the $workCapacity_{i,j}$ of each of them and the physical interference between robots. Each coalition of robots $g$, in charge of a task $t_j$, has a work capacity ($groupCapacity_{g,j}$), that means the amount of work that the group of robots can perform per time unit. Let $C$ be the set of all valid assignments of coalitions of robots to the set $T$, an assignment is valid if every robot only belongs to a coalition and each coalition only executes a task. The goal of the MRTA method is to find an optimum assignment $C^* \in C$ that maximizes the sum of utilities, that is:

$$U = \sum_{t_j \in T} U_j(fT_j) \tag{1}$$

Certainly, the task execution time and, therefore $U_j$, depend on what coalition is assigned to each task. Thus, the stated question is similar to the Set Partition Problem (SPP), where a valid partition ($C^*$) of a set (set of all coalitions) that maximize a given utility function has to be found [27]. The SPP is a NP-Hard problem and, therefore, the one proposed in this paper is NP-hard too. Furthermore, as demonstrated in [28], the SPP cannot be approximated within a factor of $O(m^{1-\varepsilon})$ $\forall \varepsilon > 0$.

Few approaches have been proposed to solve the SPP in multi-robot scenarios and in general MR problems. The algorithm by L. Vig introduced in [27] has a complexity of $O(n^k \cdot m)$ where $k$ is the maximum number of robots in a coalition. T. Service and J. Adams in [28] improve the Vig's results with a new method whose complexity is $O(n^{\frac{3}{2}} \cdot m)$ regardless of the value

of $k$. In both cases, the algorithms' solution is, in general, suboptimal and does not take into account the tasks' deadlines. If the robots can be grouped in $j$ homogeneous sets, the Service's and Adams' algorithm can provide an optimal solution with a complexity $O(n^{2j} \cdot m)$. This algorithm requires a minimum number of robots to start executing a task and, when this number is reached, the utility of the task is a constant function. The problem proposed in this paper is indeed more general: the utility function can change over time, the number of robots needed to execute a task is not limited and unknown a priori and time restrictions for the tasks are allowed. Besides, in our task the number of homogeneous groups of robots can be equal to the number of robots, that is $j = n$. In this case, Service's and Adams' algorithm complexity is equal to $O(n^{2n} \cdot m)$.

### *3.2. Foraging with tasks with deadlines*

To validate the model explained in the next sections, an extended version of the classical foraging environment is used, adding tasks with deadlines in the following way. Some randomly placed robots must locate objects, randomly placed too, and carry them to a common delivery point. Each object to be gathered ($j$) has a weight ($taskWorkLoad_j$), a utility function ($U_j$), and a deadline ($D_j$). Each robot $i$ has a load capacity value ($loadCapacity_i$), defined as the maximum weight that it can carry. Thus, if a robot cannot carry the entire object at once, it takes a part of it, goes to the delivery point and comes back to the object for more bits. Of course, this is a particular case, simpler than the general case previously described but it allows us to efficiently isolate the interference effect from other factors that can appear in more complex scenarios. In any case, the methods that will be proposed can be extended to other kinds of tasks.

## 4. Prediction of the execution time

This section describes the process to predict the execution time of an specific task for solving MR problems. The prediction is based on a Support Vector Regression model and on the task definition given in the previous section. Thus, the expected time to finish a task performed by a group of robots $g$ is assumed to be:

$$DL_{g,j} = \frac{taskWorkLoad_j}{groupCapacity_{g,j}} \tag{2}$$

The $taskWorkLoad_j$ is easily known before executing the task, for example, the weight of an object to be transported or the area of a region to be covered by the robots' group. However, it is hard to determine precisely the $groupCapacity_{g,j}$ because it usually depends on dynamic and a priori unknown factors. Among all of these factors, it is frequently accepted that the one with the strongest effect is the physical interference. In this work the deviation from the ideal capacity of a robot's team is summarized in a single term called the interference factor $I$. Thus, the work capacity of a group is considered to be:

$$groupCapacity_{g,j} = idealCapacity_{g,j} - I \tag{3}$$

where $idealCapacity$ is the model of the group capacity. When the individual characteristics of the robots are known, the $idealCapacity$ can be easily estimated as follows:

$$idealCapacity_{g,j} = \sum_{1 \leq i \leq n_g} workCapacity_{i,j} \tag{4}$$

where $n_g$ is the number of robots of the group $g$. Although this is a simple model, it proves to be very effective, as will be shown later. The following subsections will describe how to get both the $I$ value and the ideal capacity of the group.

### *4.1. Individual Work Capacity Function*

The individual work capacity depends on the characteristics of the robot and on the kind of task to be executed. In the particular case of the transport task explained in section 3.2, it is assumed that the work capacity of a robot is the amount of an object's weight that this robot can transport to the delivery point, per time unit. Under ideal conditions, that is, in an open environment without any obstacle between the object and the delivery point, the robot's work capacity can be estimated in a simple way. Let $r_i$ be a robot, $v_i$ its maximum velocity, $d_j$ the distance between the object $j$ and the delivery point and $taskWorkLoad_j$ the weight of the object. The number of trips between the delivery point and the object that the robot must make to transport the whole object is $2 \cdot \frac{taskWorkLoad_j}{workCapacity_{i,j}}$. If the acceleration and deceleration time is neglected then for each of these trips the robot needs $\frac{d_j}{v_i}$ time units. Without loss of generalization, it can be considered that a robot needs one time unit to load and another one to unload each weight unit.

Consequently, to load and unload the full object, the robot will spend $2 \cdot taskWorkLoad_j$ time units. Therefore, the total time required to completely transport the object is $T_{i,j} = 2 \cdot \frac{taskWorkLoad_j}{loadCapacity_i}(loadCapacity_i + \frac{d_j}{v_i})$ and, thus, the work capacity ($\frac{taskWorkLoad_j}{T_{i,j}}$) is:

$$workCapacity_{i,j} = \frac{loadCapacity_i \cdot v_i}{2 \cdot (loadCapacity_i v_i + d_j)} \tag{5}$$

From this value, and using the equation 4, the $idealCapacity_{g,j}$ of a coalition $g$ can be immediately calculated.

### *4.2. Interference prediction*

This paper proposes two approximations to estimate the $I$ value: the off-line interference prediction and an on-line interference learning procedure. The off-line method calculates the interference value before starting a task and the obtained value is not changed until the mission is finished. This estimation only takes into account one task and, therefore, it can only properly model the interference among robots of the coalition assigned to that task. The on-line method can modify the $I$ value during the mission execution, depending on the environment and on the robot characteristics. Thus, the on-line version also includes the interference between robots of different coalitions and with obstacles.

#### *4.2.1. Off-line interference prediction*

The creation of an off-line interference model depends on the task to be executed. To get this model for the objects transport task used in this study, the following experiment has been carried out: several robots must transport a unique object, formed by transportable bits, and the total weight transported by the robots after 40,000 time units is calculated. All the robots have the same load capacity (2 weight units in the experimental tests presented) and the same velocity (3 distance units/time unit), and therefore, they all have the same individual work capacity. Moreover, the environment does not have any obstacles only the robots, the object to be transported and the delivery point. Ten different distances between the object and the delivery point have been tested while the number of robots varied from 1 to 8. All these experiments have been executed using a multi-robot realistic simulator developed by the authors, called RoboCoT (Robot Colonies Tool) [29].

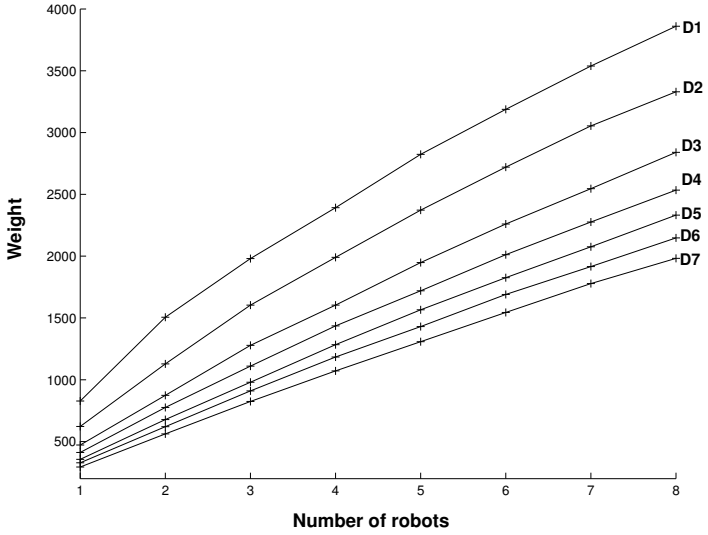



Figure 2: Total transported weight during the experiments to model interference effect.

Figure 2 shows the total transported weight as a function of the number of robots varying from 1 to 8 and for the following values of distance between the collecting and the delivery point: $D_1 = 140$ units, $D_2 = 180$ units, $D_3 = 250$ units, $D4 = 280$ units, $D5 = 330$ units, $D_6 = 360$ units and $D_7 = 400$ units. It can be observed that the transported weight is not proportional to the number of robots. As pointed out by different authors, the transported weight is lower than what would be expected if the robots could work without interferences. Thus, the interference effect reduces the group utility compared to the ideal capacity (see equation 4). The shorter the distance between the collection and delivery points and the higher the number of robots used, the stronger the interference effect becomes. For instance, when the object and the delivery point are separated by 140 distance units, ($D_1$) and with 8 robots cooperating in the task, the interference reduces the work capacity of the group down to 41.3% of the ideal situation. The difference between the expected $idealCapacity_{g,j}$ and the real capacity of the group from the experiments is the $I$ value.

Among all the methods tested to fit the experimental $I$ values to analytical expressions [30], the Support Vector Regression (SVR) model has provided the best performance. It has been proofed by other authors [31], that SRV outperforms the classical regression methods.

The goal of the SVR method is to find a function, $f(x)$, as flat as pos-

sible, that fits a given set of training data $(x_i, y_i)$, $i = 1..n_v$ where $x_i \in R^n$ represents the input data of $f(x)$ and $y_i \in R$ are the results. From this data, the function $f(x)$ is:

$$f(x) = \sum_{1 \leq i \leq n_v} (\alpha_i - \alpha_i^*) K(x, x_i) + b \tag{6}$$

where $\alpha_i$, $\alpha_i^*$ are the lagrange multipliers, $K(x, x_i)$ is a kernel function, $b$ is the bias and $n_v$ is the number of training data. During our experiments the radial basis function, shown in equation 7, is used as kernel:

$$K(x, x_i) = e^{-\gamma \|x - x_i\|^2} \tag{7}$$

where $\gamma$ is a parameter of this kernel. A detailed description of SVR method can be found in [32]. In the present case, each $x_i$ vector includes two features: the number of robots of the group and the distance between the object and the delivery point. The expected values of each coalition, $y_i$, is the $I$ value. Eighty vectors $(x_i, y_i)$ have been used to train the model, that is, $n_v = 80$, in such a way that, when a robot needs to know the group capacity and, therefore, the expected execution time of a task, the equation 6 is applied.

The implementation of the off-line SVR model is based on the *libsvm* library developed by C. Chang and C. Lin [33]. This library creates a model file from a given set of training data. In this study, the $\varepsilon - SVR$ method with a radial basis function as kernel has been used.

*4.2.2. On-line Interference prediction*

In the on-line SVR prediction process, the interference is not fully modeled before the task execution begins and, while it is in progress, the robots can adapt it. The on-line interference model is based on the J. Parrella's implementation of on-line SVR method [34] proposed by J. Ma et al [35], where samples can be removed from the model and new ones added. When the model is updated, the lagrange multipliers of the equation 6 must be modified to ensure that the $f(x)$ value is still optimum. The experiments performed show that this method is faster than the off-line version when the samples arrive one by one. The complexity of the algorithm is, in the worst case, $O(n_v^3) \cdot O(kernel)$, where $kernel$ is the time needed to calculate the kernel (equation 7).

With the on-line strategy, the robots start the execution with the off-line model, as described in 4.2.1. When a task finishes, a new sample $(x_i, y_i)$ is

created with the obtained interference and the data values, and it is added to the SVR model. Thus, the number of samples $n_v$ increases over the execution time and, therefore, the on-line SVR algorithm execution time raises too. To limit the execution time, the number of samples can be bound. If this number is above the established mentioned bound, the oldest vectors are removed from the model. Looking for a good tradeoff between process time and system performance in the experimental phase of this work, the maximum number of samples have been limited to 120.

## 5. Task Allocation: Double Round auction

The task allocation mechanisms, including the groups' formation, membership policy and task assignment is described in the following paragraphs. The DR algorithm splits the classical auction process into two steps: 'auction for a task' and 'auction for a robot'. In each task there is a robot, called the task leader, that will manage an auction process. Thus, several auction processes can be executed in parallel; one for each task.

### *5.1. Leader selection process*

Each task must have a single leader which is the robot that handles the coalition or work group, that is, it will be the auctioneer. In the initial stage, each robot is looking for a task. When a robot finds a new task, it tries to lead it. As there can only be one leader per task, the candidate will execute the leadership request process that can be seen in algorithm 1. In this algorithm the $TIME\_LEADERSHIP$ is the time that a robot $r_1$ waits to become the leader and $time_{init}$ is the instant when it starts the process. If $r_1$ receives a leadership request of a better robot before $TIME\_LEADERSHIP$, it will give up the process. The concept of 'better' depends on the tasks' characteristics, and in the present case, the better robot is the robot with the higher $workCapacity_{i,j}$, defined in equation 5.

### *5.2. Auction for a task*

If a robot is promoted to the leadership of a task, it creates, if necessary, a coalition of robots; that is, a set of robots that will cooperate to execute a specific task. In that case, the leader must decide which is the optimum group size and which robots are part of it. To take this decision, the leader uses the first step of the DR, called 'auction for a task'. In this process the leader is the auctioneer and the other robots bid using their work capacities. After starting

**Algorithm 1** Leadership request algorithm. Robot $r_1$ requests the leadership of task $t$

**Require:** $t$: task

1: Send a request for $t$
2: $time_{init} \leftarrow time$
3: **while** $time - time_{init} \leq TIME\_LEADERSHIP$ **do**
4:  **if** new message from $r_2$ **then**
5:  **if** $r_2$ is already the $t$ leader **then**
6:  Give up the request process
7:  **return**
8:  **end if**
9:  **if** $r_2$ requests $t$ **and** $r_2$ is better than $r_1$ **then**
10:  Give up the request process
11:  **return**
12:  **end if**
13:  **if** $r_2$ requests $t$ **and** $r_1$ is better than $r_2$ **then**
14:  Continue
15:  **end if**
16:  **end if**
17: **end while**
18: $r_1$ is the new task $t$ leader

the auction round, the leader waits a fixed time ($TIME\_AUCTION$) for the robots' bids. Then, it selects the robots with the highest work capacity using a greedy algorithm, until it detects that the group is able to reach the task deadline, that is, until this condition is verified:

$$DL_{g,j} \leq DL_j \tag{8}$$

where $DL_j$ is the deadline of the task $j$ and $DL_{g,j}$ is the expected time required by the group $g$ to finish the task, calculated using equation 2. Thus, the expected utility of the task is equal to $U_j(DL_{g,j})$. Furthermore, the leader does not include any selected robot in its group until it receives the confirmation from these robots. This confirmation is generated during the auction for a robot round, that will be explained in the next section. Of course, the leader collaborates in carrying out the task just as any other robot of the group.

*5.3. Auction for a robot*

Whether the deadline has been fulfilled or not, after the auction for a task round, the leader starts the 'auction for a robot' process. To get the selected robots, the leader bids for them sending the expected utility of the task. Therefore, now the leader becomes the bidder and the selected robots are the auctioneers. This can be seen in lines 18-20 of the algorithm 2. When a robot receives a bid from a leader, it waits a certain time ($TIME\_BID\_ACCEPTED$) for more bids from other leaders. After this time, the robot selects the coalitions with the greatest bid, becomes a member and sends a confirmation message ($ROBOT\_ALIVE$) to the corresponding leader. If the leader doesn't receive any answer from a robot or receives a reject message ($REFUSE$), it removes that robot from its list of selected robots. On the other hand, a robot is definitely added to the group, when the leader receives a confirmation message from it. The algorithm 3 specifies the steps executed by a non-leader robot to implement the auction for a robot stage.

When the robots implement the on-line SVR method, every time a task is finished, the leader receives the difference between the expected execution time using only the $idealCapacity_{g,j}$ (equation 4) and the real task duration. From this value, the leader calculates the $I$ value which, together with the average number of robots in the group and the distance to the delivery point, makes up a new SVR sample that is added to the model. Finally, the leader

**Algorithm 2** Leader's auction algorithm.

**Require:** $t_j$: task to execute
**Ensure:** List $R_p$ of selected robots and the AWARD message.

Send a bid request for the task $t_j$
$B_r \leftarrow \emptyset$
$time_{init} \leftarrow time$
**while** $time - time_{init} \leq TIME\_AUCTION$ **do**
  **if** a new bit $b_i$ received from robot $i$ **then**
    $B_r \leftarrow B_r \bigcup b_i$
  **end if**
**end while**
Sort $B_r$ from higher to lower work capacity
$DL_{g,j} \leftarrow \infty$
$k \leftarrow 0$
$R_p \leftarrow \emptyset$
**while** $i \leq |B_r|$ **and** $DL_{g,j} \geq DL_j$ **do**
  $R_p \leftarrow R_p \bigcup$ (robot's bid $B_r[k]$)
  Update $DL_{g,j}$ using the equation 6
  $i \leftarrow i + 1$
**end while**
**for all** $r \in R_p$ **do**
  Bid for the robot $r$ with the value $U_j(DL_{g,j})$
**end for**

sends the new SVR sample to the other robots that use this vector to update their own SVR models.

**Algorithm 3** No leader robot's auction for a robot algorithm

1: **if** $AWARD$ message, $b_0$, from a leader **then**
2: $B_l \leftarrow b_0$
3: $time_{init} \leftarrow time$
4: **while** $time - time_{init} \leq TIME\_BID\_ACCEPTED$ **do**
5: **if** $AWARD$ message, $b_i$, from a leader **then**
6: $B_l \leftarrow B_l \bigcup b_i$
7: **end if**
8: **end while**
9: $l_{best} \leftarrow$ best leader in $B_l$
10: Send a $ROBOT\_ALIVE$ message to $l_{best}$
11: **for all** $l_i$ in $B_l$ and $l_i \neq l_{best}$ **do**
12: Send $REFUSE$ message to $l_i$
13: **end for**
14: **end if**

*5.4. Complexity analysis*

The complexity of the proposed algorithms is relatively low and comparable to the current auction methods. The complexity of the auction process executed by the leader (algorithm 2) can be calculated as follows: Let $n$ be the number of bids received by the leader, then the complexity to sort the bids (line 9) using the marge sort or the binary tree sort method is $O(n \cdot log(n))$. The interference value estimation (line 15) has a complexity equal to $O(n_v^3 \cdot kernel) = O(n_v^3)$. Therefore, the complexity of the loop from line 13 to line 17 is $O(n \cdot n_v^3)$. Finally, the last loop (lines 18 to 20) has a complexity $O(n)$, and the whole algorithm's complexity is equal to $O(n \cdot log(n) + n \cdot n_v + n) \subset O(max(n \cdot log(n), n \cdot n_v))$. Moreover, the leader has to send a broadcast message asking for bids in the auction for a task process and, after that, it sends $n$ bids during the auction for a robot stage. Thus the leader sends a total of $O(2 \cdot n) = O(n)$ messages.

The complexity of the bidder algorithm during the auction for a task is $O(1)$ and it also has to send $O(1)$ messages. During the auction for a robot step the robots execute the algorithm 2 with a computational complexity of $O(n)$ to select the best leader and $O(n)$ to send the confirmation messages,

| Method | Computational | Communication |
|---|---|---|
| Double Round Auction | A: $O(max(nlog(n), n_v^3 n))$<br>B: $O(n)$ | A: $O(m)$<br>B: $O(m)$ |
| [28]<br>(T. Service et al. '10) | $O(n^{2j} \cdot m)$ | - |
| [23]<br>(L. Vig et al. '06) | $O(n^k m)$ | - |
| Dynamic Role [25]<br>(L. Chaimowicz '02) | A: $O(n)$<br>B: $O(1)$ | A: $O(n)$<br>B: $O(1)$ |

Table 1: Computational and communication complexity of several MRTA algorithms.

where $n$ is the number of received messages from the leaders. Thus, the complexity is equal to $O(2 \cdot n) = O(n)$. finally, it is easy to prove that the number of messages sent by each robot is $O(n)$ too.

The table 1 summarizes the computational and communication complexity of some MRTA algorithms that address MR problems. In this table $n$ is the number of robots, $m$ the number of tasks, $A$ the auctioneer complexity, $B$ the bidder complexity, $n_v$ the number of support vectors, $k$ the maximum number of robots in a coalition, and $j$ the number of homogeneous groups in the T. Service's algorithm.

### *5.5. Exceptions management*

An exceptional situation happens when a robot fails or when the communication errors cause a failure in the auction process. This section analyzes these situations and explains the implemented mechanisms to recover the system. These mechanisms are partially inspired in Sariel's work [36].

#### *5.5.1. Leaders duplicity*

If a leadership request message is lost or if it is received after the *TIME_LEADERSHIP* time, a task can have more than one leader. To solve this problem, a unique identifier is assigned to each robot before starting the mission. Each leader periodically broadcasts a message, called *LEADER_ALIVE*, with its identifier and the task position, to announce to the other robots that it is already the leader of that task. If another robot with a lower identifier and trying to be leader of the same task receives this message, it will leave the task leadership and will communicate it (message *EXIT_LEADER*) to

the robots of its group. When a robot receives an *EXIT_LEADER* message from its leader, it leaves the task and becomes free to be part of a new group.

*5.5.2. Leader failure*

Another exceptional situation happens when the leader of a task fails and cannot keep executing the task. To detect these situations, the *LEADER_ALIVE* message is used again, in such a way that when a robot in a group does not receive any *LEADER_ALIVE* message from its leader after a period of time, it will leave the group.

*5.5.3. Robot failure*

The non-leader robots that belong to a group periodically send a message to the group leader to inform it that they are still executing the task. If a leader does not receive any message from one of its robots after a certain time, it will remove it from the list of group members. After this dismissing step, the $DL_{g,j}$ is recalculated and a new auction round is started by the leader, if necessary.

*5.6. The Monitoring Process*

Some of the current auction methods need a monitoring process to evaluate the task progress. In the case presented here, the leader can periodically receive, during the execution of a task, information regarding the remaining weight of the object ($taskWorkLoad_j$) to be transported. The leader of the task uses this information to make a continuous estimation of the actual task workload remaining ($WL(t)$) using a linear equation:

$$WL(t) = WL(t_m) - groupCapacity_{g,j} * (t - t_m) \tag{9}$$

where $t_m$ is the time when the leader received real information about the task progress.

In a continual monitoring task progress execution, the leader knows at each moment the exact value of the $taskWorkLoad_j$, and therefore, it can start another auction process if the inequality 8 is not verified. In a non-monitoring task process execution, the leader only knows the $taskWorkLoad_j$ at the beginning of the task, and then it uses the equation 9 to predict the $taskWorkLoad_j$, with a unique constant $WL(t_m)$ during the whole process.

## 6. Experimental results

In this section we will show the results of several experiments carried out to study the effect of the double round auction process, the physical interference model, the monitoring process and the on-line SVR learning.

### *6.1. Experimental environment*

The simulator RoboCoT has been used to obtain the results presented here. The robots must execute the transport task explained in section 3.2. Three different kind of auction methods have been studied:

- Double Round auction (DR): the double auction mechanism explained in section 5.
- Single Round auction (SR): in this strategy the leader only uses the auction for a task method. Then, the robots select the first leader agreement message received.
- Greedy selection: all the leaders try to create a working group as numerous as possible, without taking into account the deadline value or the task characteristics. This strategy is like a SR in which the leader sends an agreement message to all the bidders.

The above indicated task allocation methods have been executed combined with the following strategies:

- M_I: the continual monitoring strategy is used together with the interference model.
- NM_I: the interference model is used, but without monitoring the progress of the task.
- M_NI: the continual monitoring strategy is used but without the interference method. That is, the expected execution time is calculated using only the ideal capacity.
- NM_NI: neither continual monitoring nor interference model are used.

Table 2 summarizes the experiments carried out, as a combination of the indicated auction method and monitoring/interference strategies. The greedy method does not implement any mechanism for limiting the coalition's size, so it only has sense to combine it with the NM_NI strategy.

| - | M_I | NM_I | M_NI | NM_NI |
|---|---|---|---|---|
| DR | X | X | X | X |
| SR | X | X | X | X |
| Greedy | | | | X |

Table 2: Summarize of the experiments carried out.

### *6.2. Experiments with the SVR off-line model*

To validate the SVR off-line model, two kinds of scenarios have been analyzed: homogeneous, where all the tasks have the same utility, and heterogeneous, where each task can have a different utility function.

#### *6.2.1. Homogeneous utility functions*

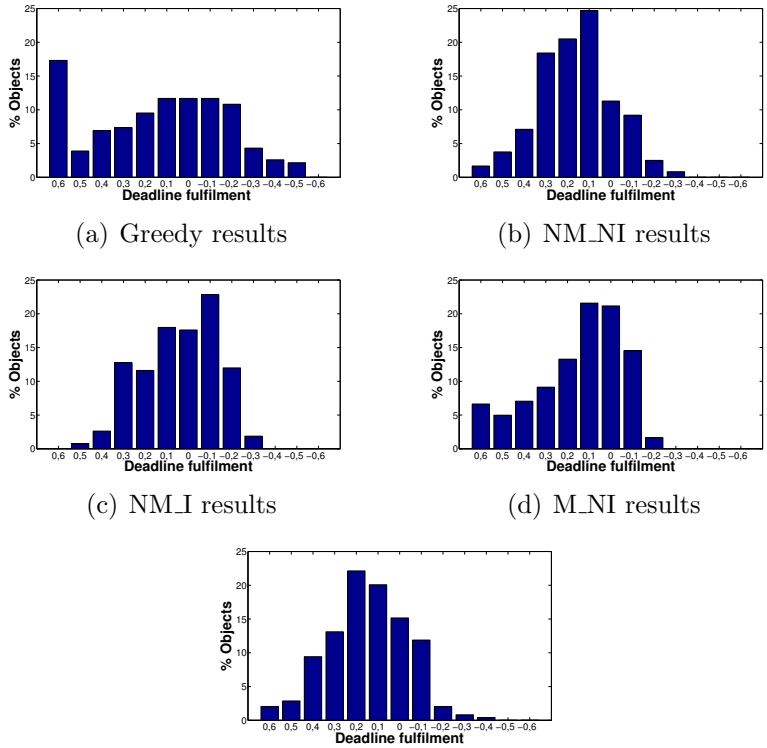
In the homogeneous utility functions experiments all the tasks have a hard deadline function with the same maximum utility $U_max$. In this way, the goal of the system is to finish as many tasks as possible before the deadline. Additionally, the objects must be transported as soon as possible even if they are over their deadline and their utility is already zero. In the auction for a task process, the leader tries to create a group to meet the tasks's deadline, without considering the task's utility. This utility is only taken into account in the auction for a robot stage, therefore, if all the tasks have the same utility, both the double and the single auction methods will give the same result. Thus, in this section only the single auction process is used, and the correctness of the SVR model can be studied regardless the kind of the auction model used.

All the experiments has been conducted using 10 robots and 3 objects to gather, with a weight equal to 40 weight units and the same deadline. This deadline time starts when the object appears in the environment. Despite having only 3 objects in the environment, when an object is fully transported to the delivery point, another one immediately appears at a random point of the working area. Thus, the initial conditions are maintained throughout all the mission execution. The robots have a load capacity of 2 weight units and a maximum velocity of 3 distance units per time unit. To simplify the analysis, the robots know the situation of each object in the environment, this information is provided by some central agent or user. The robots carry out the described mission during 30.000 time units. After this period, the time required to transport each object and the number of objects gathered is computed.

Figure 3 shows the time required to finish the tasks when the deadline is 900 time units. The bar with a label 0.6 represents the percentage of tasks whose execution time exceeds the deadline by a 60% or more. The bar labeled 0.5 represents the percentage of tasks executed within 1.5 and 1.6 times the deadline time, and so on. Negative labels represent the tasks that have fulfilled the deadline. For example, the bar with -0.2 are the tasks that required more than 0.7 but less than 0.8 of the deadline time. As can be seen, the greedy strategy correlates poorly the execution time of the task and its deadline. However, the number of tasks executed close to the deadline increases when the SR method is used. For example, using the greedy strategy 17.31% of tasks exceed the deadline time by 60%, but in the worst case of the experiments with auction methods (strategy M_NI) this percentage is only 6.64%. Table 3 shows the percentage of tasks executed before the deadline (column $t \leq DL$) and the tasks that needed less than 1.1 of the deadline to be finished (column $t < (DL + 10\%)$). Note that column $t \leq DL$ is included in the percentage of column $t < (DL + 10\%)$. From these data, it can be observed that the interference SVR model clearly improves the results, specially when it is used without monitoring (NM_I). It is also noticeable that the monitoring strategy can in some cases decrease the number of tasks that fulfill the deadline. This happens because the interference frequently stops or slows down the task progress. In this situation, the time needed to finish the task is estimated erroneously and condition 8 not verified anymore. Thus, the task leader tries to recruit new robots for its coalition increasing the interference and finally behaving like the greedy strategy. Finally, it is worth mentioning that more experiments have been carried out with different deadline times and very similar results have been obtained in all cases.

| - | $t \leq DL$ | $t < (DL + 10\%)$ |
|---|---|---|
| Greedy | 30.87 | 42.57 |
| NM_NI | 12,56 | 23.86 |
| NM_I | 36.67 | 54.27 |
| M_NI | 16.16 | 37.36 |
| M_I | 15.18 | 30.38 |

Table 3: Percentage of tasks executed before the deadline (column $t \leq DL$) and tasks that need less than 10% of its deadline to finished. The deadline is equal to 900 units and the utility function is homogeneous.




(a) Greedy results

(b) NM_NI results

(c) NM_I results

(d) M_NI results

(e) M_I results

Figure 3: Time required to finish the tasks with homogeneous utility functions and a deadline equals to 900 time units.

*6.2.2. Heterogeneous utility functions*

The results of the experiments carried out with heterogeneous utility functions, are explained here. In this situation the utility can be soft or hard and their maximum values can vary from one task to another. The goal of the system is to maximize the total utility given by equation 1. Moreover, it will be proved that with this type of tasks, the DR auction process increases the total utility compared to the SR auction and the greedy strategies.

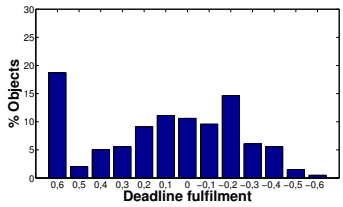
In these experiments the robots have to transport 400 objects and the total time needed to transport all of them is calculated. Fixing the number of objects at 400, the maximum utility that the system can generate is always the same. As in the single round experiments, there are only 3 objects in the environment simultaneously, when an object is completely transported, another one appears randomly placed in the environment. All the tasks have a deadline of 900 time units, but their maximum utility $U_{max_j}$ value can be different. Two utility functions have been tested, a hard deadline one and the following soft deadline function:

$$U_j = \begin{cases} U_{max_j} & \text{if } fT_j < DL_j \\ U_{max_j} \frac{0.07DL_j}{(fT_j - DL_i) + 0.07DL_j} & \text{sino} \end{cases} \quad (10)$$

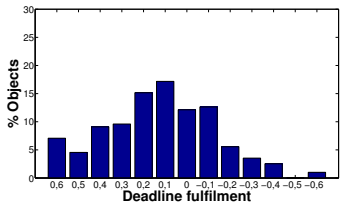
where $U_{max_j}$ is the maximum utility of the task $j$, $DL_j$ is the deadline of this task and $fT_j$ is the time required to finish it.

Figure 4 shows the time required to finish the tasks with the soft deadline function 10, using the DR auction process and heterogeneous utilities. The meaning of the bars is the same as was explained for figure 3. Similarly to the SR method and homogeneous utilities, the graphic shapes show a best correlation between the deadline and the number of tasks finished when the DR auction method is applied compared to the greedy strategy. Table 4 shows the percentage of tasks fulfilling their deadline (column $t \leq DL$) and the tasks that needed less than 1.1 times their deadline to finish (column $t < (DL + 10\%)$). Once again, the greedy and the NM_I strategies obtain the greatest percentage of tasks that strictly meet the deadline. The same experiments have been repeated with hard deadline functions obtaining very similar results.

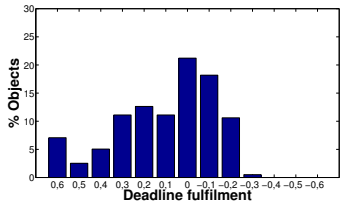
It has to be remembered that, apart from the results shown for the deadline fulfillment, the main stated goal of the methodology is to maximize the total utility of the system. Accordingly, it will be shown next that the DR auction with the interference method, clearly increases the total utility of the



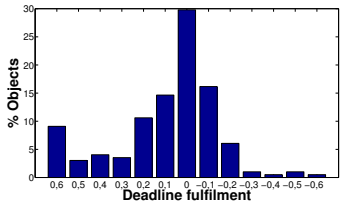

(a) Results with the greedy strategy



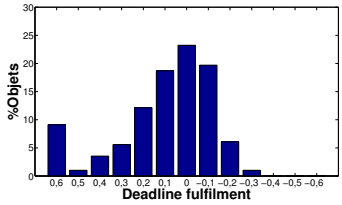

(b) NM_NI results




(c) NM_I results




(d) M_NI results




(e) M_I results

Figure 4: Time required to finish the tasks with soft deadline heterogenous utility functions (see equation 10) and a deadline equal to 900 time units. The double round auction process has been used.

system, outperforming the results of the other strategies.

Figure 5 shows the total utility (see equation 1) when the soft deadline function is used. As can be seen, the M_I and NM_I strategies present very similar results, therefore, it can be stated that the proposed interference model correctly predicts the evolution of a task without any monitoring process. Moreover, in both cases the total utility has been increased about 40% compared to the greedy method. In addition, this figure also highlights the impact of the interference on the system. Thus, the results of the methods that do not use the interference model (M_NI and NM_NI) and the results of the greedy system are very similar. Finally, the utility produced by the robots that use the single round auction strategy (SR) is also very similar to the greedy experiments results. Therefore, the auction for a robot round

| - | $t \leq DL$ | $t < (DL + 10\%)$ |
|---|---|---|
| Greedy | 37.88 | 48.48 |
| NM_NI | 25.30 | 37.40 |
| NM_I | 29.30 | 50.50 |
| M_NI | 25.25 | 55.05 |
| M_I | 26.76 | 50.00 |

Table 4: Percentage of tasks executed before the deadline (column $t \leq DL$) and tasks that need less than 10% of its deadline to finished (column $t \leq (DL + 10\%)$). The deadline is equal to 900 units and the utility function is heterogeneous following the equation 10. The double round auction process has been used in all cases.

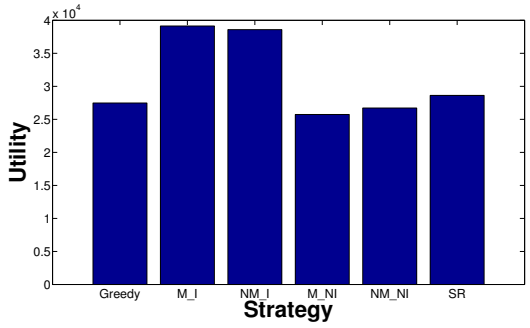



Figure 5: Total utility with the soft deadline function.

included in the DR methodology clearly increases the system performance.

The results of the experiments with the hard deadline function are shown in figure 6. The NM_I presents the best results, increasing by about 43% of the total utility of the system compared to the greedy strategy. Moreover, the non-monitoring strategy outperforms the monitoring methods. The utility of the SR strategy is slightly lower than the greedy method one, but, as it was shown in figure 3, using the SR strategy clearly results in the execution time of the tasks being more concentrated around the deadline.

### *6.3. On-line SVR experiments*

In this subsection the impact of the on-line interference learning on the total utility generated by the system will be analyzed. In these experiments 7 homogeneous robots must transport 200 objects using the soft deadline utility function, the NM_I together with the double round auction. Two kind of experiments have been conducted:

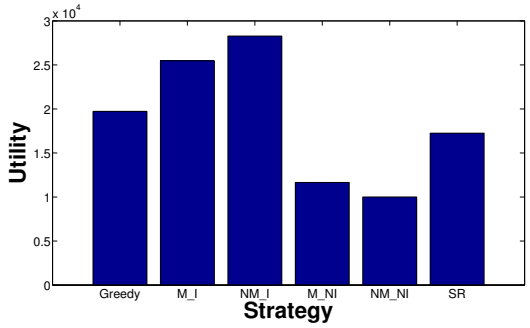



Figure 6: Total utility with the hard deadline function.

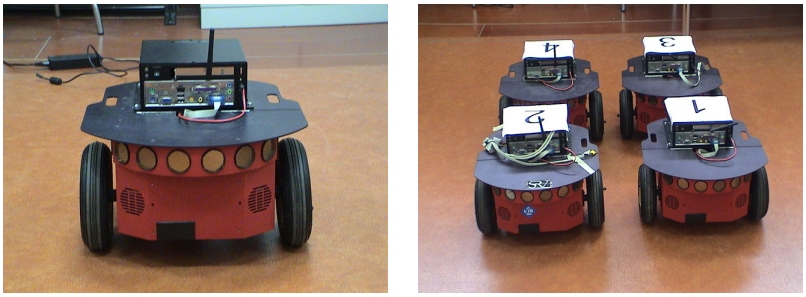
| | off-line SVR | on-line SVR |
|---|---|---|
| Without errors | 10802 | 11851 |
| With errors | 6290 | 10226 |

Table 5: Total Utility of the on-line SVR experiments

- Experiments without errors: in this case the robots know without any error all the experiment parameters, that is the robot's velocity and distance to the delivery point.

- Experiments with errors: in these experiments the velocity known by the robots is wrong. While the real velocity is 3.0, they suppose it is equal to 0.8 distance units per time units. This wrong velocity is used to get the ideal individual work capacity in equation 5. Thus, it can be observed what happens when there are errors in the kinematic model of the robot.

Table 5 shows the total utility of the system with both strategies, on-line and off-line SVR. It can be observed that when there are no errors in the system, the on-line SVR algorithm increases by 9.7% the utility compared to the off-line strategy. When the system has to deal with errors in the kinematic model this improvement is more remarkable and equal to 62.5%. Thus, it can be stated that the methods proposed here can correctly model the main phenomena that distort the ideal behavior of a MRTA system, that is: the physical interference produced by obstacles and robots from the same and other coalitions and the kinematic errors.



(a) Pioneer-3DX.

(b) The four robots used during the experiments.

Figure 7: Images of the robots Pioneer-3DX.

### *6.4. Real scenario*

To finish the experimental phase, some of the previously described experiments have been executed using real robots. In particular, to test the DR auction methods and the interference model, four Pioneer 3DX, shown in figure 7, have been used. Each vehicle, whose localization is calculated by odometry, is equipped with a ring of 16 regularly spaced sonars and has a maximum speed of 0.25m/s. The robots are endowed with a Via Epia mother board with an Eden 600MHz processor, 512MB of RAM and a wireless card for the communications. The well-known Player server [37] has been used for interacting with the robots' hardware. To accomplish the mission the robots have to collect two objects randomly placed in a 10m long by 5m wide workspace. Figure 8 shows the initial location of the four robots for one of these executions. The marks *O2* and *O1* on the floor, are the objects to gather and the *D* label, in the center of the image, is the delivery point.

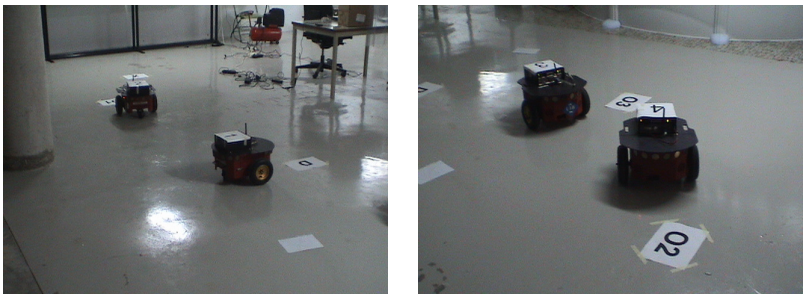
Figure 9 shows some snapshots taken during the task execution. In figure 9(a) robot 1 is getting closer to the delivery point *D*, meanwhile robot 2 is moving towards object 1. Figure 9(b) shows robots 3 and 4 collaborating to transport *O2*. Next, figure 10 shows the behavior of the system when a leader fails and how the robots recover the initial situation using the exception management mechanisms explained in section 5.5. Figure 10(a) shows the initial situation, where three robots are assigned to the object *O2* and only one to *O1*. In figure 10(b) the *O1*'s leader fails (it is manually withdrawn from the environment). Finally, figure 10(c) shows that a new leader is assigned to task *O1*, only 15s after the old leader failed. The system behavior during the experiment with real robots has been, in all cases,


Figure 8: Initial location of the robots during the experiments.

just as expected and in accordance with the described methods. Thus, it is proved that the described methods can be implemented on real robots without highly demanding requirements.

## 7. Conclusions and future work

This paper has presented a new auction method to fulfil the deadlines of a set of tasks using multi-robot coalitions. Because the stated problem is NP-hard, it has to be solved with a heuristic approach. One of the main problems to solve in real time environments is to predict the execution time of a task. This prediction is specially complex when the task has to been carried out by a coalition of robots because of the physical interference, among other factors. Several authors have pointed out the impact of the interference on the system performance, but none of them has used this information in an auction process. Thus, the method presented in this paper is the first auction based one that takes into account the interference effect and the first MRTA method that can determine the coalition size to satisfy real time restrictions. To determine the coalition size, we have proposed a SVR based model to predict the execution time. Our method has been tested with transport tasks using both off-line and on-line SVR methods, on a simulator and on real robots. The results of the experimental tests show that the results of the classical auction mechanisms can be improved combining the interference




(a) (b)

Figure 9: Some snapshots of the execution on real robots. (a) Robot 1 is getting to the delivery point and robot 2 is going toward to object 1. (b) Robots 3 and 4 are moving towards object 2.

(a) Initial situation. (b) The $O1$'s leader is withdrawn from the environment.

(c) System recovery.

Figure 10: System recovery example when a leader fails.

model and the double round auction method. Moreover, the SVR model renders unnecessary the monitoring of the task progress.

The work presented has some challenging aspects to add and to improve. We are working to use a preemption auction method, that is, a method that allows the exchange of robots between working groups. We will extend these experiments to other kind of tasks, such as exploration, cleaning, etc. Moreover, we will study other on-line SVR sample substitution strategies. For example, our first results show that the clustering techniques proposed by W. Wang and Z. Xu in [38] are not useful for on-line learning because they require a lot of time.

## Acknowledgments

This work has been partially supported by project DPI2008-06548-C03-02 and FEDER funding.

## References

[1] Gerkey, B.P., Mataric, M.. A formal analysis and taxonomy of task allocation in multi-robot systems. International Journal of Robotics Research 2004;23 (9):939–954.

[2] Lerman, K., Galstyan, A.. Mathematical model of foraging in a group of robots: Effect of interference. Autonomous Robots 2002;13(2):127–141.

[3] Rosenfeld, A., Kaminka, G.A., Kraus, S.. Coordination of Large Scale Multiagent Systems; chap. A Study of Scalability Properties in Robotic Teams. Springer-Verlag; 2006, p. 27–51.

[4] Theodoridis, S., Koutroumbas, K.. Pattern Recognition. Elsevier Science; 2006.

[5] Guerrero, J., Oliver, G.. Interference modelization in multi-robot auction methods. In: $6^{\text{th}}$ IFAC Symposium on Intelligent Autonomous Vehicles. Toulouse, France; 2007,.

[6] Guerrero, J., Oliver, G.. A multi-robot auction method to allocate tasks with deadlines. In: $7^{\text{th}}$ IFAC Symposium on Intelligent Autonomous Vehicles. Lecce, Italy; 2010,.

[7] Ducatelle, F., Förster, A., Caro, G.D., Gambardella, L.. Task allocation in robotic swarms: new methods and comparisons. Tech. Rep. IDSIA-01-09; IDSIA - Dalle Molle Institute for Artificial Intelligence; Lugano, Switzerland; 2009.

[8] Low, K.H., Leow, W.K., Ang, M.H.. Task allocation via self-organizing swarm coalitions in distributed mobile sensor network. In: 19th National Conference on Artificial Intelligence. San Jose, USA; 2004, p. 28–33.

[9] de Oliveira, D., Ferreira, P.R., Bazzan, A.L.. A swarm based approach for task allocation in dynamic agents organizations. In: 3th International Joint Conference on Autonomous Agents and Multiagent Systems; vol. 3. New York City, USA; 2004, p. 1252–1253.

[10] Lein, A., Vaughan, R.T.. Adaptive multi-robot bucket brigade foraging. In: 11st International Conference on the Simulation and Synthesis of Living Systems. Winchester, UK; 2008, p. 337–342.

[11] Goldberg, D., Mataric, M.J.. Robot Teams: From Diversity to Polymorphism; chap. Design and Evaluation of Robust Behavior-Based Controllers for Distributed Multi-Robot Collection Tasks. Ak Peters; 2002, p. 315–344.

[12] Dias, M.B., Stentz, A.. Traderbots: A market-based approach for resource, role, and task allocation in multirobot coordination. Tech. Rep. CMU-RI-TR-03-19; Carnegie Mellon University; 2003.

[13] Gerkey, B.P., Mataric, M.. Sold!: Auction methods for multi-robot coordination. IEEE Transactions on robotics and Automation, Special Issue on Multi-robot Systems 2002;18(5):758–768.

[14] Botelho, S., Alami, R.. Cooperative plan enhancement in multi-robot context. In: 6th International Conference on Intelligent Autonomous Systems. Venise, Italy; 2000, p. 131–138.

[15] Wei, S., Lihua, D., Hao, F., Haiqiang, Z.. Task allocation for multi-robot cooperative hunting behavior based on improved auction algorithm. In: 27th Chinese Control Conference. Beijing, China; 2008, p. 435–440.

[16] Thomas, G., Williams, A.B.. Sequential auctions for heterogeneous task allocation in multiagent routing domains. In: International Conference on Systems, Man, and Cybernetics. San Antonio, USA; 2009, p. 1995–2000.

[17] Lemaire, T., Alami, R., Lacroix, S.. A distributed tasks allocation scheme in multi-uav context. In: International Conference on Robotics and Automation (ICRA); vol. 4. New Orleans, USA; 2004, p. 3622–3627.

[18] Jones, E.G., Dias, M., Stentz, A.. Learning-enhanced market-based task allocation for disaster response. Tech. Rep. CMU-RI-TR-06-48; Carnegie Mellon University; 2006.

[19] Melvin, J., Keskinocak, P., Koenig, S., Tovey, C., Ozkaya, B.Y.. Multi-robot routing with rewards and disjoint time windows. In: International Conference on Intelligent Robots and Systems (IROS). San Diego, USA; 2007, p. 2332–2337.

[20] Campbell, A., Wu, A., Shumaker, R.. Multi-agent task allocation: Learning when to say no. In: $10^{\text{th}}$ annual conference on Genetic and evolutionary computation. Atlanta, USA; 2008, p. 201–208.

[21] Jones, E.G.. Multi-robot coordination in domains with intra-path constraints. Ph.D. thesis; The Robotics Institute Carnegie Mellon University; 2009.

[22] Viguria, A., Maza, I., Ollero, A.. S+t: An algorithm for distributed multirobot task allocation based on services for improving robot cooperation. In: International Conference on Robotics and Automation (ICRA). Pasadena, USA; 2008, p. 3163–3168.

[23] Vig, L., Adams, J.A.. Market-based multi-robot coalition formation. In: $8^{\text{th}}$ International Symposium on Distributed Autonomous Robotic Systems. Minneapolis, USA; 2006, p. 227–236.

[24] Zlot, R.M., Stentz, A.. Market-based multirobot coordination for complex tasks. International Journal of Robotics Research, Special Issue on the 4th International Conference on Field and Service Robotics 2006;25(1):73–101.

[25] Chaimowicz, L., Campos, M.F.M., Kumar, V.. Dynamic role assignment for cooperative robots. In: International Conference on Robotics and Automation (ICRA). Washington, USA; 2002, p. 292–298.

[26] del Acebo, E., de-la Rosa, J.L.. Introducing bar systems: A class of swarm intelligence optimization algorithms. In: AISB Convention Communication, Interaction and Social Intelligence. Aberdeen, Scotland; 2008, p. 18–23.

[27] Vig, L.. Multi-robot coalition formation. Phd thesis; Gratuate School of Vanderbilt University; Nashville, USA; 2006.

[28] Service, T., Adams, J.. Coalition formation for task allocation: theory and algorithms. Autonomous Agents and Multi-Agent Systems (En Prensa) 2010;.

[29] Guerrero, J., Ortiz, A., Oliver, G.. Experiments design using a mobile robots simulator. In: IEEE-TTTC International Conference on Automation, Quality and Testing Robotics. Cluj-Napoca, Rumania; 2002, p. 29–34.

[30] Guerrero, J., Oliver, G.. Physical interference impact in multi-robot task allocation auction methods. In: IEEE Workshop on Distributed Intelligent Systems. Praga, Czech Republic; 2006, p. 19–24.

[31] Juutilainen, I., Röning, J., Laurinen, P.. A sutdy on differences in interpolation capabilities of models. In: IEEE Mid-Summer Workshop on Soft Computing in Industrial Applications. Espoo, Finland; 2005, p. 202–207.

[32] Smola, A.J., Schölkopf, B.. A tutorial on support vector regression. Statistics and Computing 2004;14:199–222.

[33] Chang, C., Lin, C.. LIBSVM: a library for support vector machines (http://www.csie.ntu.edu.tw/ cjlin/libsvm); 2001.

[34] Parrella, F.. Online support vector regression. Master's thesis; Department of Information Science; University of Genoa, Italy; 2007.

[35] Ma, J., Theiler, J., Parkins, S.. Accurate on-line support vector regression. Journal Computation 2003;15:2683–2703.

[36] Sariel, S., Balch, T., Erdogan, N.. Incremental multi-robot task selection for resource constrained and interrelated tasks. In: International Conference on Intelligent Robots and Systems (IROS). San Diego, USA; 2007, p. 2314–2319.

[37] Collett, T.J., MacDonald, B.A., Gerkey, B.P.. "player 2.0: Toward a practical robot programming framework". In: Australasian Conference on Robotics and Automation. Sydney, Australia; 2005,.

[38] Wang, W., Xu, Z.. A heuristic training for support vector regression. Neurocomputing 2004;61:259–275.

José Guerrero received his degree in computer since from the University of Balearic Islands (UIB) where he is currently PhD candidate. His research interest includes multi-robot task allocation mechanisms and auction and swarm coordination mechanisms.

Gabriel Oliver received a degree in Physics from the Universitat Autònoma de Barcelona. He earned his Ph.D. in Computer Science from the Universitat Politècnica de Catalunya. His major research interests are real-time vision and control architectures for autonomous mobile robots. He has been the leading researcher of projects granted by the local administration, the Spanish Scientific Council (CICYT) and the European Commission (FP7). He is currently Professor at the Department of Mathematics and Computer Science at the Universitat de les Illes Balears where he is the Director of the Systems, Robotics and Computer Vision Group (SRV).