Instituto de Sistemas e Robótica

Pólo de Lisboa

# Blocking Controllability of a Mobile Robot Population[1]

Francisco A. Melo
M. Isabel Ribeiro
Pedro Lima

May 2004
RT-601-04

ISR Torre Norte
Av. Rovisco Pais
1049 - 001 Lisboa
Portugal

[1] Work partially supported by Programa Operacional Sociedade de Informação (POSI) in the frame of QCA III and by the FCT Project *Rescue – Cooperative Navigation for Rescue Robots* (SRI/32546/99-00). The first author acknowledges the PhD grant SFRH/BD/3074/2000.

# Blocking Controllability of a Mobile Robot Population

Francisco A. Melo M. Isabel Ribeiro
Pedro Lima
Institute for Systems and Robotics
Instituto Superior Técnico
1049-001 Lisboa
e-mail: {fmelo,mir,pal}@isr.ist.utl.pt

May 5, 2004

**Abstract**

In this paper, the problem of determining if a population of mobile robots is able to travel from an initial configuration to a target configuration is addressed. This problem is related with the controllability of the automaton describing the system. To solve the problem, the concept of navigation automaton is introduced, allowing a simplification in the analysis of controllability. A set of illustrative examples is presented.

## 1 Introduction

Robotic navigation is a central topic of research in robotics, since the ability that a robot has to accomplish a given task may greatly depend on its capability to navigate in the environment. The related literature presents numerous works on the subject, and proposes different navigation strategies, in particular, Markov Models [1], dynamic behaviours [2] or Petri Nets [3].

In the last decades a great effort has been addressed to the subject of cooperative navigation. The existence of multiple robots sharing environment resources or pathways when simultaneously navigating in a common environment leads to new and challenging navigation problems. A common approach to solve the problem is the extension of known strategies for single robot navigation to the multi-robot case, for which there are several examples presented in the literature. In the work by Balch and Hybinette [4], potential fields are used to achieve multi-robot navigation. However, in the multi-robot navigation framework, new topics of investigation emerged, such as cooperation and formation control or flocking, as addressed, e.g., by Balch and Arkin [5], where a reactive behaviour-based approach to formation control is described.

In this paper, the problem of multi-robot navigation in an environment described by a topological map is addressed. It concerns the problem of driving the robots from some initial configuration to a final or target configuration. In particular, we develop analysis strategies in order to determine under which situations the target configuration becomes non-achievable, and, as such, prevent those situations. In a previous work [6], this analysis has been conducted for a set of homogeneous robots([1]). This paper extends those results to a set of generic robots, i.e., where robots with different capabilities may intervene.

The multi-robot system is modeled as a finite-state automaton (FSA) and the main contribution of this paper is the analysis of the blocking and controllability properties of the automaton. Since the automaton models the movement of the complete robot population in the environment, from a start configuration to given goal configuration, properties such as blocking and controllability have direct correspondence with the successful completion of this objective. A blocking state in the automaton corresponds to a distribution of the robots from which the desired goal configuration is not achievable (because one of the robots has reached a site from where it cannot leave, for example). Controllability of the automaton means that such blocking states are avoidable, and it is possible to disable some actions to prevent the robots from reaching such blocking configurations.

The results presented in this paper relate the blocking and controllability properties of the automaton modeling the multi-robot system (which can be a large-dimension automaton, for complex systems) with the blocking and controllability properties of smaller automata, named as *navigation automata*, that model the navigation of each individual robot in the population.

The paper is organized as follows. In Section 2, some basic concepts are introduced and the problem under study is described. Section 3 approaches the problem of determining the blocking properties of the automaton describing the system for a homogeneous and a non-homogeneous population. In Section 4, the results regarding controllability are presented both for homogeneous and non-homogeneous systems. Section 5 presents a set of illustrative examples. Finally, Section 6 concludes the paper and presents directions for future work.

## 2 Navigation Automata and the Multi-robot system

In this section, some basic concepts regarding automata are introduced and the notation used throughout this paper is described.

**Notation regarding automata [7]**:
A general automaton $Q$ is a six-tuple $Q = (X, E, f, \Gamma, x_0, X_m)$, where

- $X$ is the state space;
- $E$ is the set of possible events;
- $f : X \times E \longrightarrow X$ is the transition function;
- $\Gamma : X \longrightarrow 2^E$ is the active event function;
- $x_0$ is the initial state;
- $X_m$ is the set of marked states.

[1] We consider a set of robots to be homogeneous when all robots are alike, i.e., they have the same capabilities.

The events in $Q$ label each transition of the automaton, and a sequence of consecutive events in $Q$ is called a *string* generated by $Q$. The set of all strings generated by $Q$ is known as *language generated* by $Q$ and is denoted by $\mathcal{L}(Q)$. The set of all strings driving the system to a marked state is called *language marked* by $Q$, $\mathcal{L}_m(Q)$. An automaton is called *unmarked* if $X_m = \emptyset$.

An automaton $Q$ is said to be *non-blocking* if $\overline{\mathcal{L}_m(Q)} = \mathcal{L}(Q)$ and *blocking* otherwise, i.e., if $\overline{\mathcal{L}_m(Q)} \subsetneq \mathcal{L}(Q)$. The notion of blocking does not apply to unmarked automata.

When it is clear from the context which initial state it refers to, $x = f(x_0, s)$ will be denoted $x(s)$. A set $X_C \subset X$ of states is said to be closed if $f(x, s) \in X_C$, for any $s \in \mathcal{L}(Q)$ and $x \in X_C$. A blocking automaton verifies $X_C \cap X_m = \emptyset$.

**The Problem** Consider a system of $N$ robots, navigating in a discrete environment (represented by a topological map) consisting of $M$ distinct sites. This is referred as a $N$-R-$M$-S situation ($N$ robots and $M$ sites) or a $N$-R-$M$-S system. The set of sites in the map is denoted by $\mathcal{S} = \{1, \ldots, M\}$.

When a robot is in site $i$, it will not generally be able to reach all other sites in a single movement. The function $\Omega_k : \mathcal{S} \to 2^{\mathcal{S}}$ establishes a correspondence between a site $i$ and a set $\mathcal{S}_i \subset \mathcal{S}$ of sites reachable from $i$ in one movement of robot $k$. If site $j$ is reachable from site $i$, i.e., if $j \in \Omega_k(i)$, then, for robot $k$, site $j$ is *adjacent* to site $i$. Function $\Omega_k$ is called the *adjacency function* for robot $k$.

This paper addresses the problem of driving the robots from an initial configuration $C_I$ to a final or target configuration $C_F$. The set of sites containing at least one robot in the final configuration is denoted by $\mathcal{S}_T$. The sites in $\mathcal{S}_T$ are called *target sites*. From the point of view of final configuration, no distinction is made among the robots, i.e., it is not important which robot is in each target site.

**Navigation Automata** Robot $k$ of the population moves in the environment defined by the topological map according to its own adjacency function and is described by an unmarked automaton $G_k = (Y_k, E_k, f_k, \Gamma_k, y_{0k})$.

The state space $Y_k$ is the set of all possible positions of robot $k$, verifying $Y_k = \mathcal{S}$. The event set $E_k$ is the set of all possible actions for robot $k$. Generally, an action consists of the command leading to the next site for the robot to move to. Since all robots have the same event space, to avoid ambiguity, action $i$ issued to robot $k$ is denoted by $Go_k(i)$, where $i$ is the next site for the robot $k$. Therefore, all events in the system correspond to movements of the robots. For simplicity, it is assumed that only one robot moves at a time.

The active event function $\Gamma_k$ when robot is in state $i$ corresponds to the sites reachable from $i$ in one movement. This means that $\Gamma_k = \Omega_k$.

**Definition 1 (Navigation automaton)** *Given a robot $k$ moving in a discrete environment consisting of $M$ distinct sites, the* navigation automata *for this robot are the marked automata $G_k(Y_m) = (Y_k, E_k, f_k, \Gamma_k, y_{0k}, Y_m)$, where:*

- *$Y_k$, $E_k$, $f_k$ and $\Gamma_k$ are defined as above;*
- *$Y_m$ is a set of target states, $Y_m \subset \mathcal{S}_T$.*

In the case of a homogeneous set of robots, i.e., in which $\Omega_1 = \ldots = \Omega_N$, all $G_i$ are alike, except for the initial condition $y_{0k}$. In this situation, when the initial condition is clear from the context, or not important, a navigation automaton will simply be denoted by $G(Y_m) = (Y, E_m, f_m, \Gamma_m, y_0, Y_m)$.

## 2.1 The Multi-Robot System

Consider the situation where $N$ robots (not necessarily similar) move along $M$ sites, with no constrains on the number of robots present at each site. In such situation, each robot can be in $M$ different positions and there are $M^N$ different possible configurations.

The system of all robots can be described by a FSA, $G = (X, E, f, \Gamma, x_0, X_m)$, where the state-space $X$ is the set of all possible robot configurations, yielding $|X| = M^N$. Each state is a $N$-tuple $(x_1, x_2, ..., x_N)$, where $x_i$ is the site where robot $i$ is.

Since any robot can aim at any target site, there is a set of states corresponding to the target configurations. Such states are called *compliant configurations*, and the set of all compliant configurations is denoted by $X_F \subset X$. Notice that, in automaton $G$, $X_m = X_F$.

Now consider the case of a homogeneous set of robots. In this situation, some of the states described above are equivalent, since the robots are indistinguishable. This leads to a simplification in the automaton, given that the equivalent configurations are merged into one single state. This simplified model has a state space $X$ consisting of $|X| = \begin{pmatrix} M+N-1 \\ M-1 \end{pmatrix}$ states, where each state consists of a $M$-tuple $(n_1, n_2, ..., n_M)$, $n_i$ being the number of robots in site $i$. Note that in this case of a homogeneous set of robots there is only one compliant configuration $X_F = \{x_m\} = X_m$.

**The event set** As seen before, robot $k$ has an available set of actions $E_k$, denoted by $Go_k(i)$. Therefore, the multi-robot system has a set of actions $E = \bigcup_k E_k$ available, all consisting of $Go_k(i)$ actions.

In the particular case of a homogeneous set of robots, it is not important which robot moves, but only from which site it moves. In such case, the possible actions for the robots will be of the type $Go(i, j)$, corresponding to a movement of a robot from site $i$ to site $j$. Notice that, if $R_i \subset \{1, \ldots, N\}$ is the set of robots in site $i$, the event $Go(i, j)$ in the homogeneous situation can be any of the events $Go_k(j)$, $k \in R_i$.

# 3 Blocking

Let $G$ be the automaton modeling a $N$-R-$M$-S system.

If $G$ is blocking, there is a closed set of states $C$, called *blocking set*, such that $C \cap X_m = \emptyset$. This, in turn, means that whenever the robots reach a configuration corresponding to a state $x \in C$ it is not possible to drive them to the desired configuration anymore.

Usually, blocking is checked by verifying $\overline{\mathcal{L}_m(G)} \subsetneq \mathcal{L}(G)$ exhaustively. In the present case, as the system can lead to relatively large automata for not so large $M$ and $N$, a more effective way to check the blocking properties of $G$ is desirable. This section addresses this problem.

## 3.1 Homogeneous System

In this subsection, the particular case of a set of homogeneous robots is analyzed, along the lines presented in [6]. The simple fact that the robot population is homogeneous has an immediate implication in terms of state-space and event set, since there are generally less events available and the state space is smaller than in the non-homogeneous situation. Moreover, in this case, $\Omega_1 = \Omega_2 = \ldots = \Omega_N = \Omega$.

From Section 2.1, the automaton $G$ describing the $N$-R-$M$-S system has a state space $X$ consisting of the $M$-tuples $(n_1, \ldots, n_M)$, where $n_i \in \{1, \ldots, N\}$ is the number of robots in site $i$. On the other hand, the set of possible events $E$ consists of all events of the type $Go(i, j)$, where $i, j \in \mathcal{S}$.

Considering the navigation automata $G(y_m)$ describing the navigation of any of the $N$ robots, Proposition 1 holds. An example of its application is presented in Section 5.

**Proposition 1** *The automaton $G$ describing a $N$-R-$M$-S homogeneous system is non-blocking iff* all *the navigation automata $G(y_m)$ are non-blocking, with any initial condition $y_0$ corresponding to a site from the initial configuration, and $y_m$ is a target state.*

**Proof:** See [6].

## 3.2 Non-homogeneous System

Let $G$ be the automaton modeling a generic $N$-R-$M$-S system. The state space $X$ of this automaton is the set of the $N$-tuples $(x_1, \ldots, x_N)$, where $x_i \in \{1, \ldots, M\}$ is the site where robot $i$ is. The set of possible events $E$ consists of all events of the type $Go_k(i)$, where $i \in \mathcal{S}$.

From automata theory, given any two automata $Q_i = (X_i, E_i, f_i, \Gamma_i, x_{0i}, X_{mi}), i = 1, 2$, the *parallel composition* of $Q_1$ and $Q_2$ is the automaton $Q = Q_1||Q_2 = (X, E, f, \Gamma, x_0, X_m)$ defined by [7]:

- $X = X_1 \times X_2$;
- $E = E_1 \cup E_2$;
- $f((x_1, x_2), e) = \begin{cases} (f_1(x_1, e), f_2(x_2, e)) & e \in \Gamma_1(x_1) \cap \Gamma_2(x_2); \\ (f_1(x_1, e), x_2) & e \in \Gamma_1(x_1) \backslash E_2; \\ (x_1, f_2(x_2, e)) & e \in \Gamma_2(x_2) \backslash E_1; \\ (x_1, x_2) & \text{otherwise}; \end{cases}$
- $\Gamma(x_1, x_2) = [\Gamma_1(x_1) \cap \Gamma_2(x_2)] \cup [\Gamma_1(x_1) \backslash E_2] \cup [\Gamma_2(x_2) \backslash E_1]$;
- $x_0 = (x_{01}, x_{02})$;
- $X_m = X_{m1} \times X_{m2}$.

Note that the automata describing each of the robots have disjoint event sets and a common state space. If the marked states of the automaton $G$ describing the overall system are disregarded, then $G = ||_i G_i$, where $||_i G_i$ stands for the parallel composition for all the automata $G_i, i \in \{1, \ldots, N\}$.

Recall from automata theory that

$$\mathcal{L}(Q_1||Q_2) = P_1^{-1}(\mathcal{L}(Q_1)) \cap P_2^{-1}(\mathcal{L}(Q_2)), \tag{1}$$

where $P_i^{-1}$ stands for the inverse projection operator w.r.t. $E_1 \cup E_2$ (see [7]). From (1), it becomes clear that, in the present system, $\mathcal{L}(G) = \bigcap_i P_i^{-1}(\mathcal{L}(G_i))$.

Assuming that the automaton $G$ describing the overall system is non-blocking, any string $s \in \mathcal{L}(G)$ is a prefix to a string in $\mathcal{L}_m(G)$. Let $y_i(s) = f_i(y_{0i}, P_i(s))$. Taking any string $s \in \mathcal{L}(G_i)$, and for any string $s' \in \mathcal{L}(G) \cap P_i^{-1}(s)$, the $i$th component of $x(s')$ is equal to $y_i(s)$. If $G$ is non-blocking, there is a string $s''$ such that $s'$ is a prefix of $s''$ and $x(s'') \in X_m$. This implies that $y_i(s'')$ is in a state corresponding to a target site, and, hence, if $Y_{mi} = \{x_i(s'')\}$, $G_i(Y_{mi})$ is non-blocking.

The previous paragraph proves that if $G$ is non-blocking, the same property holds for $G_i$ with all target states as marked states. The converse, however, is not true. As a particular case, in a homogeneous system, and according to Proposition 1, this condition is not enough to ensure $G$ to be non-blocking.

From what was stated above, Theorem 2 follows.

**Theorem 2** *In a generic N-R-M-S system, for its automaton $G$ to be non-blocking, all the navigation automata $G_i(Y_m)$ must non-blocking, with $Y_m = \mathcal{S}_T$. Similarly, if $G$ is blocking, there is at least one $i$ and one target state $y_m \in Y_m$ such that $G_i(y_m)$ is blocking.*

**Proof:** The first statement was proved above. The second statement is immediate, since, for $G$ to be blocking, there must be a configuration from which some robot $i$ cannot reach some target site $m$ corresponding to a target state $y_m$. Then, $G_i(y_m)$ is blocking. ■

Note that in Proposition 1 all navigation automata are used with individual states $y_m$ as marked states. Theorem 2 provides only necessary conditions for $G$ to be blocking or non-blocking. A necessary condition for $G$ to be blocking provides a sufficient condition to be non-blocking. Corollary 3 complements Theorem 2 in that it provides sufficient conditions for $G$ to be blocking or non-blocking.

**Corollary 3** *In a generic N-R-M-S system described by an automaton $G$, if all navigation automata $G_i(y_m)$ are non-blocking, then, so is $G$, with $y_m$ a target site. Conversely, if any navigation automata $G_i(Y_m)$ is blocking, with $Y_m = \mathcal{S}_T$, then so is $G$.*

**Proof:** If all $G_i(y_m)$ are non-blocking, the second condition of Theorem 2 fails, and, hence, $G$ is non-blocking. Conversely, if any $G_i(Y_m)$ is blocking, the first condition of Theorem 2 fails, and hence $G$ is blocking. ■

Theorem 2 together with its corollary provide necessary and sufficient conditions for the automata $G$ to be blocking or non-blocking. However, generally, it is not possible to determine the blocking properties of $G$ simply by taking into account the blocking properties of each navigation automata individually, since these properties depend on the relation between them.

### 3.2.1 Determination of the blocking properties of $G$

It is, however, possible to determine if the overall automaton $G$ is blocking or not, by comparing the blocking information regarding each of the automata $G_i(y_m)$.

Given an automaton $G = ||_i G_i$, define $G_{-k} = ||_{i \neq k} G_i$ and let $K(m)$ be the number of robots in target site $m$ in the final configuration $C_F$. More generally, if $U \subset \mathcal{S}_T$ is a set of target sites, define $K(U) = \sum_{m \in U} K(m)$. If for some robot $k$, all $G_k(y_m)$ are non-blocking, with $y_m$ being a target state, then $G$ is non-blocking if and only if $G_{-k}$ is non-blocking (with respect to all the possible compliant configurations). This result is proved, for generic automata, in Lemma 4.

From automata theory, given two automata $Q_1$ and $Q_2$,

$$\mathcal{L}_m(Q_1||Q_2) = P_1^{-1}(\mathcal{L}_m(Q_1)) \cap P_2^{-1}(\mathcal{L}_m(Q_2)). \tag{2}$$

**Lemma 4** *If $Q_1 = (Y_1, E_1, f_1, \Gamma_1, y_{01}, Y_{m1})$ is a non-blocking automaton, and $Q_2 = (Y_2, E_2, f_2, \Gamma_2, y_{02}, Y_{m2})$ is a generic automaton with $E_1 \cap E_2 = \emptyset$, then $Q = Q_1||Q_2$ is blocking if and only if $Q_2$ is blocking.*

**Proof:** Let $Q = (Y, E, f, \Gamma, y_0, Y_m)$.

Suppose that $Q_2$ is non-blocking. Then, $\overline{\mathcal{L}_m(Q_2)} = \mathcal{L}(Q_2)$. Take, then, $s \in \mathcal{L}(Q)$. Let $s_1 = P_1(s)$, $s_2 = P_2(s)$ and $y = (y_1, y_2) = f(y_0, s)$. Clearly, $y_1 = f_1(y_{01}, s_1)$ and $y_2 = f_2(y_{02}, s_2)$. Since both $Q_1$ and $Q_2$ are non-blocking, there are $s_1'$ and $s_2'$ such that $s_1 s_1' \in \mathcal{L}_m(Q_1)$ and $s_2 s_2' \in \mathcal{L}_m(Q_2)$. Since $E_1 \cap E_2 = \emptyset$, by making $s' = s_1' s_2'$, it becomes evident that $ss' \in \mathcal{L}(Q)$. Notice, however, that $y_1' = f_1(y_1, s_1') \in Y_{m1}$ and $y_2' = f_2(y_2, s_2') \in Y_{m2}$. But then, $y' = f(y, s') = (y_1', y_2') \in Y_{m1} \times Y_{m2} = Y_m$ and $ss' \in \mathcal{L}_m(Q)$.

On the other hand, if $Q_2$ is blocking, there is a blocking set $Y_C \subset Y_2$ such that $Y_C \cap Y_{m2} = \emptyset$ and there are no transitions out of $Y_C$. Take any $y_2 \in Y_C$ and any $y_1 \in Y_1$, and let $y = (y_1, y_2)$. Obviously, $y \notin Y_m$. If $Q$ is non-blocking, there is a string $s$ driving the automaton $Q$ from $y$ to $Y_m$. Let $s_2 = P_2(s)$. $s_2$ will drive $Q_2$ from $Y_C$ to $Y_{m2}$, which is impossible. Then, $Q$ must be blocking. ■

By Corollary 3, if $G_k(Y_m)$ is blocking for some $k$, $G$ is blocking. Conversely, if all $G_k(y_m)$ are blocking for some $y_m$, $G$ is also blocking, since it is possible to prevent all robots from reaching the corresponding target site.

Given any target site $m \in \mathcal{S}_T$, define as $B(m)$ the number of robots that block with respect to site $m$. Similarly, if $U \subset \mathcal{S}_T$ is a set of target sites, define $B(U)$ as the number of robots *simultaneously* blocking the sites in $U$. In general, $B(U) \neq \sum_{m \in U} B(m)$.

From what was said, if the number of robots blocking simultaneously the sites in some set $U \subset \mathcal{S}_T$ are such that $N - B(U) < K(U)$, then $G$ blocks, and therefore, $N - B(U) < K(U)$. This means that there are not enough "free" robots to go to the sites in $M$. This condition may be easily verified using the *blocking information matrix*.

**Definition 2 (Blocking Information Matrix)** *Given a generic $N$-$R$-$M$-$S$ system, the* blocking information matrix *(BIM) $\mathbf{B}_N$ is a $N \times N$ matrix such that element $(k, m)$ is 0 if $G_k(y_m)$ is blocking and 1 otherwise.*

Each of the $N$ lines of matrix $\mathbf{B}_N$ corresponds to a different robot. On the other hand, if a target site $m$ has $K(m)$ robots in the target configuration, matrix $\mathbf{B}_N$ will have $K(m)$ columns corresponding to this site.

Matrix $\mathbf{B}_N$ is easily computed from the analysis of the navigation automata $G_k(y_m)$, and the following result can now be proved.

**Theorem 5** *Given the Blocking Information Matrix $\mathbf{B}_N$ for a $N$-$R$-$M$-$S$ system, the automaton $G$ describing the overall system is blocking if and only if there is a permutation matrix $\mathbf{P}$ such that $\mathbf{PB}_N$ has only ones in the main diagonal.*

**Proof:** The permutation matrix simply conveys the fact that there is no ordering of the robots. Suppose, without loss of generality, that there are $N$ different target sites([2]) and suppose that $G$ is non-blocking. This means that, from any configuration, each robot $k$ must be able to move to a target site $m$. In terms of the matrix $\mathbf{B}_N$, this means that, in line $k$ of the matrix there must be at least a 1 in column $m$. But this is not enough, since all the robots must be able to move to different target sites. In other words, there must be a distribution of the robots among the target sites, such that each robot is in a different target site. In terms of the matrix $\mathbf{B}_N$ this means that it must be possible to rearrange the lines in the matrix so that there are only ones in the main diagonal.

The converse statement is immediate. ■

# 4 Supervisory Control

In this section, the problem of controllability of $G$ is addressed. The controllability problem is related to the design of a supervisor $S$, such that, when applied to the original system, the resulting system marks some desired language $\mathcal{K}$.

Although the automaton $G$ describing the system already marks the desired language, in a situation where the automaton is blocking, it is not desirable that the system reaches a blocking state, since this will prevent the final configuration to be reached. The presence of a supervisor $S$ in the system under study will necessarily relate to this situation where blocking must be prevented.

For a general automaton $Q$, if $\mathcal{K}$ is the desired marked language, there is a non-blocking supervisor $S$ such that $\mathcal{L}_m(S/Q) = \mathcal{K}$ if and only if $\overline{\mathcal{K}}E_{uc} \cap \mathcal{L}(Q) \subset \overline{\mathcal{K}}$ and $\mathcal{K} = \overline{\mathcal{K}} \cap \mathcal{L}_m(S/Q)$, [7].

As shown in Section 3, blocking states prevent the system from accomplishing the objective. For this reason, it is important to determine the existence of a supervisor which can prevent blocking, i.e., it is important to determine if the system is controllable. In the following analysis, the existence of unobservable events will be disregarded even if they make sense from a modelling point of view, as described in [6].

## 4.1 Homogeneous System

The problem of controllability will be first addressed on a homogeneous system, followed by its extension to a generic system. Complete details for homogeneous systems are presented in [6].

Consider that there is a non-empty set of uncontrollable events $E_{uc} \subset E$. These events may correspond to accidental movements of the robots which can't be avoided. As stated, supervisory control only makes sense when blocking is involved, which means that, from Proposition 1, at least one of the navigation automata $G(y_m)$ is blocking. Let then $\{G(y_i), i \in I\}$ be the set of blocking navigation automata for the system (at least one blocking $G(y_i)$ exists, since $G$ is assumed to be blocking). $Y_{Ci}$ will denote the blocking set of automaton $G(y_i)$. Define $Y_B = \bigcup_{i \in I} Y_{Ci}$ and $Y_{NB} = \mathcal{S}_T \backslash Y_B$.

**Proposition 6** *The blocking automaton $G$ describing a homogeneous $N$-$R$-$M$-$S$ is controllable iff the automaton $G_{uc}(Y_{NB})$ defined below is controllable, with*

[2]If this is not the case, consider a site $m$ with $K(m) > 1$ as being $K(m)$ different sites.

*respect to the language* $\mathcal{K} = \mathcal{L}_m(G_{uc}(Y_{NB}))$. *The automaton* $G_{uc}(Y_{NB})$ *is defined by the six-tuple* $(Y, E_m, f_m, \Gamma_m, y_0, Y_m)$, *where* $E_{uc} \subset E_m$ *is the set of uncontrollable events:*

- $Y$, $E_m$, $f_m$ *and* $\Gamma_m$ *are defined as in Definition 1;*
- $y_0$ *is the initial condition;*
- $Y_m = Y_{NB}$.

**Proof:** See [6].

### 4.2 Non-homogeneous System

Consider the automaton $G$ describing a generic $N$-R-$M$-S system. As stated before, blocking in $G$ is related to the number of robots "available" to fill each target site, when considering blocking sets. On the other hand, controllability relates with the ability of a supervisor to disable strings of events driving a robot to a blocking set.

Let $E_{uk} \subset E_k$ be the set of uncontrollable events for robot $k$. It is possible to include controllability information in matrix $\mathbf{B}_N$ in order to conclude about the controllability of $G$. If $G_k(y_m)$ is blocking but controllable with respect to the language $\mathcal{K} = \mathcal{L}_m(G_k(y_m))$, then the element $(k, m)$ of matrix $\mathbf{B}_N$ is set to $-1$. This motivates the following and most general form of Theorem 5.

**Theorem 7** *Given the Blocking Information Matrix* $\mathbf{B}_N$ *for a generic* $N$*-R-*$M$*-S system, the automaton* $G$ *describing the overall system is blocking if and only if there is a permutation matrix* $\mathbf{P}$ *such that* $\mathbf{PB}_N$ *has only ones in the main diagonal.*

*If* $G$ *is blocking, but there is a permutation matrix* $\mathbf{P}_1$ *such that* $\mathbf{P}_1\mathbf{B}_N$ *has only non-zero elements in the main diagonal, then* $G$ *is controllable with respect to the language* $\mathcal{K} = \mathcal{L}_m(G)$.

$E_{uc}$ *is the set of uncontrollable events of* $G$.

**Proof:** The first part is nothing but Theorem 5. To prove the second part, note that, if $G_k(y_m)$ is controllable, blocking can be prevented simply by disabling controllable events, and, hence, the controlled automaton is non-blocking. Then, from Theorem 5, blocking in $G$ is prevented by disabling only controllable events and, therefore, $G$ is controllable. On the other hand, if such is not possible for some combination of the $G_k(y_m)$, it is not possible to prevent at least one blocking configuration: $G$ is non-controllable. ■

Observe the relation between Theorem 7 and Proposition 6. In fact, Proposition 6 can be deduced from Theorem 7 when the robot set is homogeneous.

## 5 Examples

We present three examples of the application of Theorem 7 in a simple indoor rescue situation. Consider a non-homogeneous set of three robots in which:

**The Crawler** ($Cr$) has tracker wheels and is capable of climbing and descending stairs. It is able to open doors only by pushing;

**The Puller** ($Pl$) is a wheeled mobile manipulator, able to open doors either by pushing or pulling. However, it is not able to climb stairs;

**The Pusher** ($Ps$) is a wheeled robot, able to open doors only by pushing. It cannot climb stairs.

The rescue operation takes place in the indoor environment depicted in Figure 1 (e.g., a fire scenario). On the left is the physical map of the place, and on the right is the corresponding topological map.

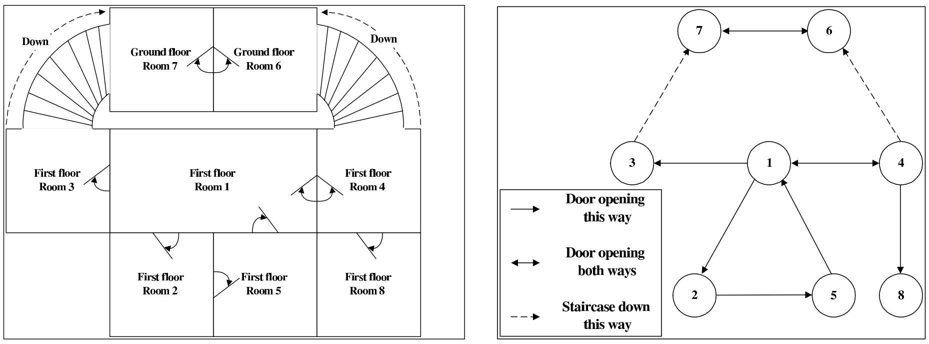


Figure 1: Map of the environment.

Each of the robots is described by a different automaton, as represented in Figure 2.

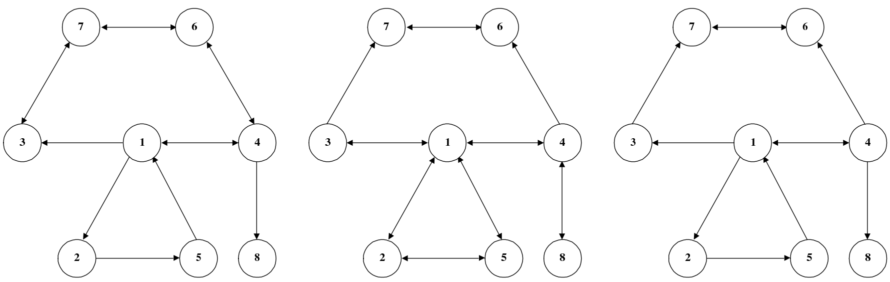

Figure 2: Automata for the robots.

The robots will leave Room 1 to assist three different victims, somewhere in the building. The doors open as shown in Figure 1 which limits the robots access to the different rooms. Moreover, when in Rooms 6 or 7, only the Crawler can go upstairs. Finally, when in Rooms 3 and 4, all the robots may fall downstairs, i.e., events $Go_k(6)$ and $Go_k(7)$ are uncontrollable for all $k$. The following examples illustrate the practical use of Theorem 7. We determine if there are configurations that prevent the success of a given rescue operation which, in terms of the framework proposed in this paper, correspond to blocking configurations. The situation where there are victims in sites $a, b$ and $c$ is referred to as the $a - b - c$ Rescue.

## 5.1 $6-7-8$ Rescue

In this situation, the BIM for the system is:

$$\mathbf{B}_3 = \begin{bmatrix} -1 & -1 & 0 \\ 1 & 1 & 0 \\ -1 & -1 & 1 \end{bmatrix}, \tag{3}$$

where the lines correspond to Pusher, Puller and Crawler and the columns correspond to target sites 6, 7 and 8, respectively.

Note that both $Ps$ and $Cr$, once inside Room 8, are not able to leave. This means that $G_{Ps}(6)$, $G_{Ps}(7)$, $G_{Cr}(6)$ and $G_{Cr}(7)$ are blocking. However, by disabling the events $Go_{Ps}(8)$ and $Go_{Cr}(8)$, this blocking can be prevented. Then, $\mathbf{B}_3(1,1) = \mathbf{B}_3(1,2) = \mathbf{B}_3(3,1) = \mathbf{B}_3(3,2) = -1$. On the other hand, if $Ps$ or $Pl$ get downstairs, they cannot go back upstairs. However, they cannot get to Room 8 without going through Room 4 and eventually falling to Room 6. But this cannot be avoided, since $Go_k(6)$ is uncontrollable. Then, $G_{Ps}(8)$ and $G_{Ps}(8)$ are blocking and uncontrollable, and $\mathbf{B}_3(1,3) = \mathbf{B}_3(2,3) = 0$. Finally, there is no room from which $Pl$ cannot reach Rooms 6 and 7, and from which $Cr$ cannot reach Room 8. Then, $G_{Pl}(6)$, $G_{Pl}(7)$ and $G_{Cr}(8)$ are non-blocking, and $\mathbf{B}_3(2,1) = \mathbf{B}_3(2,2) = \mathbf{B}_3(3,3) = 1$.

From Theorem 7 the system is blocking but controllable. In fact, for example in the configuration where Crawler and Pusher are in room 8, it is impossible to reach the target configuration. However, this can be prevented, by disabling, for example, $Go_{\mathbf{Ps}}(8)$, which is a controllable event.

## 5.2 $2-8-8$ Rescue

In this situation, the BIM for the system is:

$$\mathbf{B}_3 = \begin{bmatrix} -1 & 0 & 0 \\ -1 & 0 & 0 \\ -1 & 1 & 1 \end{bmatrix}, \tag{4}$$

with the columns corresponding to target sites 2, 8 and 8, respectively. It becomes evident that the system is blocking but, unlike the previous example, it is uncontrollable. In fact, since two robots are required for site 8 and the only way to reach Room 8 is through Room 4, they will eventually move to Room 6 instead of moving to Room 8 (since $Go_k(6)$ is uncontrollable), once they get to Room 4. In this situation, it may be impossible to assist both victims in site 8. In fact, as long as there is more than one victim in site 8 ($K(8) > 1$), this problem will always exist. This happens because there are two robots which “helplessly” fall downstairs, blocking site 8 ($B(8) = 2$). Then, $N - B(8) = 3 - 2 = 1 < K(8)$, and the system is blocking. Since the only way to Room 8 is through Room 4, this situation cannot be prevented.

## 5.3 Homogeneous set

This last example is intended to relate Theorem 7 and Propositions 1 and 6. Suppose that, in Situation 5.2, the set of robots is homogeneous (i.e., either they are all Crawlers, or Pullers or Pushers). Then, all lines in matrix $\mathbf{B}_3$ are alike, and all elements of $\mathbf{B}_3$ must be one for the system to be non-blocking. But, by definition of BIM, this happens if all $G_k(y_m)$ are non-blocking, which is exactly what Proposition 1 states. By observing matrix (4), one can conclude that this never happens.

For the system to be controllable, all elements of matrix $\mathbf{B}_3$ must be non-zero. This means that either $G_k(y_m)$ is non-blocking and $B(k,m) = 1$, or $G_k(y_m)$ is controllable, and $B(k,m) = -1$. Since all navigation automata $G_k$ are alike, this happens only if the automaton $G_{uc}(Y_{NB})$ defined in Proposition 6 is controllable. Then, the only situation where the system is controllable is if all robots are Crawlers: $Y_{NB} = \{2\}$, and $G_{uc}(Y_{NB})$ is controllable.

## 6 Conclusions and future work

The problem of controlling the navigation of a set of mobile robots operating in a discrete environment was approached. Relevant results have been derived, that allow the use of small dimension automata (navigation automata) to infer about the blocking properties of the general automaton that describes the complete system. In a situation where a specific configuration is aimed for a set of robots, the presented results allow to determine, using global information, if the global objective is achievable, and if blocking configurations are avoidable.

An important extension of the present work is the determination of the relation between the blocking properties of the navigation automata and the ergodicity of the Markov Chain which can be used to model the complete system, when a probabilistic uncertainty is associated to the events representing the movements of the robots. Other interesting issue is the use of this local information in an optimal decision process, when a decentralized system is considered.

## Acknowledgements

Work partially supported by Programa Operacional Sociedade de Informação (POSI) in the frame of QCA III and by the FCT Project *Rescue—Cooperative Navigation for Rescue Robots* (SRI/32546/99-00). The first author acknowledges the PhD grant SFRH/BD/3074/2000.

## References

[1] R. Simmons and S. Koenig, "Probabilistic Robot Navigation in Partially Observable Environments," *Proceedings of the International Joint Conference on Artificial Intelligence, Montreal, Canada*, pp. 1080–1087, 1995.

[2] A. Steinhage, "Dynamical Systems for the Generation of Navigation Behaviour," Ph.D. dissertation, Institut für Neuroinformatik, Ruhr-Universität, Bochum Germany, November 1997.

[3] R. D. Hale, M. Rokonuzzaman, and R. G. Gosine, "Control of Mobile Robots in Unstructured Environments Using Discrete Event Modeling," *SPIE International Symposium on Intelligent Systems and Advanced Manufacturing, Boston, USA*, September 1999.

[4] T. Balch and M. Hybinette, "Social Potentials for Scalable Multirobot Formations," *IEEE International Conference on Robotics and Automation (ICRA-2000), San Francisco, USA*, 2000.

[5] T. Balch and R. C. Arkin, "Behavior-based formation control for multi-robot teams," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 6, pp. 926–939, December 1998.

[6] F. A. Melo, P. Lima, and M. I. Ribeiro, "Event-driven modelling and control of a mobile robot population," *Proceedings of the 8th Conference on Intelligent Autonomous Systems, Amsterdam Netherlands*, 2004.

[7] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, ser. The Kluwer International Series On Discrete Event Dynamic Systems. Kluwer Academic Publishers, 1999.